

数值天气预报课程作业

吴浩杰 20221170212

1 实习三

麦卡托投影的地图放大系数和科里奥利参数

1.1 实习目的

通过编程计算，使学生掌握麦卡托投影的地图放大系数和科里奥利参数的计算方法。

1.2 实习要求

编写并提交计算麦卡托投影的地图放大系数和科里奥利参数的相关程序。要求学生在机房现场操作，参考沈桐立等 (2015) 关于麦卡托投影相关内容，撰写实习报告，教师进行随堂讲解和指导。

1.3 实习内容

根据实际计算麦卡托投影地图放大系数的公式，输入 a 、 d 、 Ω 和格点坐标，计算地图放大系数和科里奥利参数。

Listing 1: 麦卡托投影地图放大系数

```
1  import numpy as np
2  import pandas as pd
3  import matplotlib.pyplot as plt
4
5  # 设置matplotlib中文字体支持
6  plt.rcParams['font.sans-serif'] = ['SimHei'] # 设置默认字体
   为黑体
7  plt.rcParams['axes.unicode_minus'] = False # 解决负号显示问
   题
8
9  # 初始化参数
10 d = 100
```

```

11     phy0 = 5
12     a = 6371
13     omega = 7.292e-5
14     l_ref = a * np.cos(np.deg2rad(22.5)) * np.tan(np.deg2rad(
15         phy0))
16
17     # 定义起始经纬度
18     lon_start = 100 # 起始经度 (°E)
19     lat_start = 22.5 # 起始纬度 (°N)
20
21     # 初始化矩阵
22     m = np.zeros((6, 7))
23     f = np.zeros((6, 7))
24
25     # 计算映射系数m和科氏力参数f
26     for In in range(7):
27         for Jn in range(6):
28             # 计算当前点的纬度 (考虑到矩阵翻转)
29             current_lat = lat_start - (5-Jn) * (d / (a * np.pi /
30                 180)) # 5-Jn是因为矩阵会被翻转
31             # 地图放大系数只与纬度有关
32             m[Jn, In] = 1 / np.cos(np.deg2rad(current_lat))
33             # 科氏力参数只与纬度有关
34             f[Jn, In] = 2 * omega * np.sin(np.deg2rad(
35                 current_lat))
36
37     # 翻转矩阵
38     m = np.flipud(m)
39     f = np.flipud(f)
40
41     # 打印结果
42     print("映射系数m:")
43     print(m)
44     print("\n科氏力参数f:")
45     print(f)
46
47     # 将结果保存到Excel文件
48     df_m = pd.DataFrame(m)
49     df_f = pd.DataFrame(f)
50
51     df_m.to_excel('map_scale.xlsx', index=False, header=False)
52     df_f.to_excel('coriolis_force.xlsx', index=False, header=
53         False)

```

```

50
51 # 计算经纬度网格
52 lon_start = 100 # 起始经度 (°E)
53 lat_start = 22.5 # 起始纬度 (°N)
54
55 # 计算每个网格点对应的经纬度
56 lon_grid = np.zeros((6, 7))
57 lat_grid = np.zeros((6, 7))
58
59 for i in range(7):
60     for j in range(6):
61         # 计算经度变化 (根据距离和纬度计算)
62         delta_lon = (i * d) / (a * np.cos(np.deg2rad(
63             lat_start)) * np.pi / 180)
64         # 计算纬度变化 (根据距离计算)
65         delta_lat = (j * d) / (a * np.pi / 180)
66
67         lon_grid[j, i] = lon_start + delta_lon
68         lat_grid[j, i] = lat_start - delta_lat # 纬度向南递
69         减
70
71 # 翻转经纬度网格以匹配m和f矩阵
72 lon_grid = np.flipud(lon_grid)
73 lat_grid = np.flipud(lat_grid)
74
75 # 创建图形和子图
76 plt.figure(figsize=(15, 6))
77
78 # 绘制地图放大系数的热力图
79 plt.subplot(121)
80 im1 = plt.imshow(m, cmap='viridis')
81 plt.colorbar(im1, label='地图放大系数')
82 plt.title('地图放大系数分布')
83
84 # 设置经纬度刻度
85 xticks = np.arange(7)
86 yticks = np.arange(6)
87 plt.xticks(xticks, [f'{lon_grid[0,i]:.1f}°E' for i in range
88     (7)])
89 plt.yticks(yticks, [f'{lat_grid[i,0]:.1f}°N' for i in range
90     (6)])
91 plt.xlabel('经度')
92 plt.ylabel('纬度')

```

```

89
90 # 绘制科氏力参数的热力图
91 plt.subplot(122)
92 im2 = plt.imshow(f, cmap='viridis')
93 plt.colorbar(im2, label='科氏力参数')
94 plt.title('科氏力参数分布')
95
96 # 设置经纬度刻度
97 plt.xticks(xticks, [f'{lon_grid[0,i]:.1f}°E' for i in range
98                    (7)])
99 plt.yticks(yticks, [f'{lat_grid[i,0]:.1f}°N' for i in range
100                   (6)])
101 plt.xlabel('经度')
102 plt.ylabel('纬度')
103
104 # 调整子图之间的间距
105 plt.tight_layout()
106
107 # 显示图形
108 plt.show()
109
110 # 打印经纬度网格
111 print("\n经度网格 (°E):")
112 print(lon_grid)
113 print("\n纬度网格 (°N):")
114 print(lat_grid)
115
116 print("\n结果已保存到Excel文件中")

```

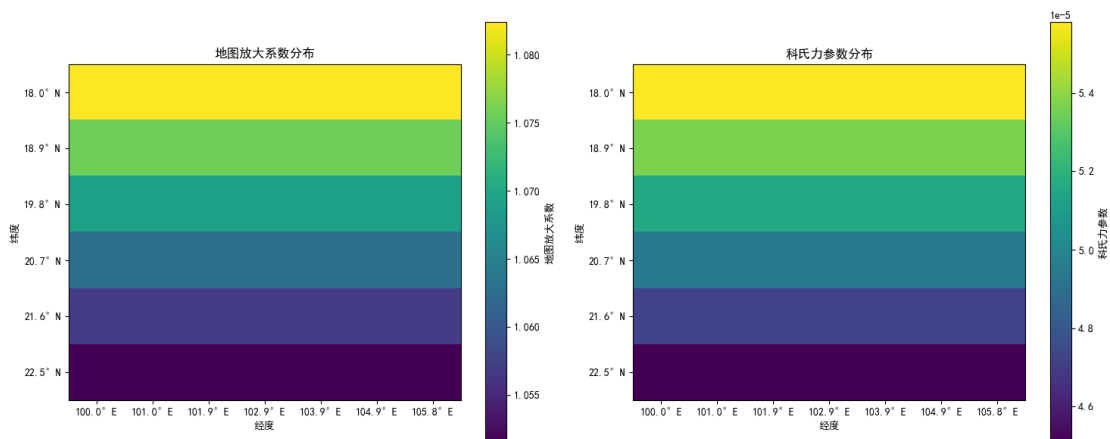


Figure 1: 参考点 O 为 (100°E, 5°N)

1.4 习题一

把麦卡托投影标准纬度变为 30°N ，选取一矩形网格，左下角原点 O 为 $(100^{\circ}\text{E}, 50^{\circ}\text{N})$ ，网格距 100 km，网格点向北 6 个，向东 7 个。计算网格上各点的地图放大系数和科里奥利参数

Listing 2: 习题一

```
1  import numpy as np
2  import pandas as pd
3  import matplotlib.pyplot as plt
4
5  # 设置matplotlib中文字体支持
6  plt.rcParams['font.sans-serif'] = ['SimHei'] # 设置默认字体
   为黑体
7  plt.rcParams['axes.unicode_minus'] = False # 解决负号显示问
   题
8
9  # 初始化参数
10 d = 100 # 网格距离，单位km
11 phy0 = 30 # 标准纬度 $30^{\circ}\text{N}$ 
12 a = 6371 # 地球半径，单位km
13 omega = 7.292e-5 # 地球角速度
14
15 # 计算参考长度，使用原点位置( $100^{\circ}\text{E}$ ， $50^{\circ}\text{N}$ )
16 l_ref = a * np.cos(np.deg2rad(50)) * np.tan(np.deg2rad(phy0)
   )
17
18 # 定义起始经纬度
19 lon_start = 100 # 起始经度 ( $^{\circ}\text{E}$ )
20 lat_start = 50 # 起始纬度 ( $^{\circ}\text{N}$ )
21
22 # 初始化矩阵
23 m = np.zeros((6, 7)) # 地图放大系数矩阵
24 f = np.zeros((6, 7)) # 科氏力参数矩阵
25
26 # 计算映射系数m和科氏力参数f
27 for In in range(7):
28     for Jn in range(6):
29         # 计算当前点的纬度（考虑到矩阵翻转）
30         current_lat = lat_start - (5-Jn) * (d / (a * np.pi /
   180)) # 5-Jn是因为矩阵会被翻转
31         # 地图放大系数只与纬度有关
32         m[Jn, In] = 1 / np.cos(np.deg2rad(current_lat))
33         # 科氏力参数只与纬度有关
```

```

34         f[Jn, In] = 2 * omega * np.sin(np.deg2rad(
35             current_lat))
36
37     # 翻转矩阵使得左下角为原点
38     m = np.flipud(m)
39     f = np.flipud(f)
40
41     # 打印结果
42     print("地图放大系数m:")
43     print(m)
44     print("\n科氏力参数f:")
45     print(f)
46
47     # 将结果保存到Excel文件
48     df_m = pd.DataFrame(m)
49     df_f = pd.DataFrame(f)
50
51     df_m.to_excel('map_scale_exp1.xlsx', index=False, header=
52         False)
53     df_f.to_excel('coriolis_force_exp1.xlsx', index=False,
54         header=False)
55
56     # 计算经纬度网格
57
58     # 计算每个网格点对应的经纬度
59     lon_grid = np.zeros((6, 7))
60     lat_grid = np.zeros((6, 7))
61
62     for i in range(7):
63         for j in range(6):
64             # 计算经度变化（根据距离和纬度计算）
65             delta_lon = (i * d) / (a * np.cos(np.deg2rad(
66                 lat_start)) * np.pi / 180)
67             # 计算纬度变化（根据距离计算）
68             delta_lat = (j * d) / (a * np.pi / 180)
69
70             lon_grid[j, i] = lon_start + delta_lon
71             lat_grid[j, i] = lat_start - delta_lat # 纬度向南递
              减

```

```

72
73 # 创建图形和子图
74 plt.figure(figsize=(15, 6))
75
76 # 绘制地图放大系数的热力图
77 plt.subplot(121)
78 im1 = plt.imshow(m, cmap='viridis')
79 plt.colorbar(im1, label='地图放大系数')
80 plt.title('地图放大系数分布')
81
82 # 设置经纬度刻度
83 xticks = np.arange(7)
84 yticks = np.arange(6)
85 plt.xticks(xticks, [f'{lon_grid[0,i]:.1f}°E' for i in range
86                    (7)])
87 plt.yticks(yticks, [f'{lat_grid[i,0]:.1f}°N' for i in range
88                    (6)])
89 plt.xlabel('经度')
90 plt.ylabel('纬度')
91
92 # 绘制科氏力参数的热力图
93 plt.subplot(122)
94 im2 = plt.imshow(f, cmap='viridis')
95 plt.colorbar(im2, label='科氏力参数')
96 plt.title('科氏力参数分布')
97
98 # 设置经纬度刻度
99 plt.xticks(xticks, [f'{lon_grid[0,i]:.1f}°E' for i in range
100                   (7)])
101 plt.yticks(yticks, [f'{lat_grid[i,0]:.1f}°N' for i in range
102                   (6)])
103 plt.xlabel('经度')
104 plt.ylabel('纬度')
105
106 # 调整子图之间的间距
107 plt.tight_layout()
108
109 # 显示图形
110 plt.show()
111
112 # 打印经纬度网格
113 print("\n经度网格 (°E):")
114 print(lon_grid)

```

```

111     print("\n纬度网格 (°N):")
112     print(lat_grid)
113
114     print("\n结果已保存到Excel文件中")

```

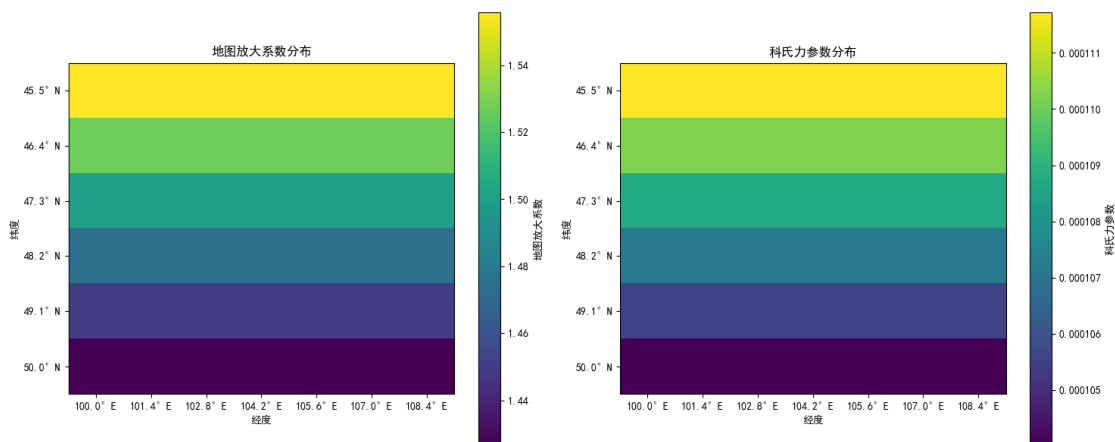


Figure 2: 中央经度为 45° 的麦卡托投影放大系数分布

1.5 习题二

中尺度天气预报模式 WRF 的麦卡托投影如书上所示，为减小模式网格上的投影形变，标准纬度可随投影范围改变。设模拟区域为中南半岛地区，尝试各种语言编写程序，计算麦卡托投影在东亚地区的地图放大系数和科里奥利参数，并与 WPS 相同设置的结果进行比较

Listing 3: 习题二

```

1     import numpy as np
2     import pandas as pd
3     import matplotlib.pyplot as plt
4
5     # 设置matplotlib中文字体支持
6     plt.rcParams['font.sans-serif'] = ['SimHei'] # 设置默认字体
7     # 为黑体
8     plt.rcParams['axes.unicode_minus'] = False # 解决负号显示问
9     # 题
10
11     # 初始化参数 - 中南半岛地区
12     d = 100 # 网格距离，单位km
13     phy0 = 15 # 标准纬度15°N（适合中南半岛地区）
14     a = 6371 # 地球半径，单位km
15     omega = 7.292e-5 # 地球角速度
16
17     # 计算参考长度，使用原点位置(100°E，20°N) - 中南半岛区域

```



```

16 lon_start = 100 # 起始经度 (°E)
17 lat_start = 20 # 起始纬度 (°N)
18 l_ref = a * np.cos(np.deg2rad(lat_start)) * np.tan(np.
    deg2rad(phy0))
19
20 # 初始化矩阵
21 m = np.zeros((6, 7)) # 地图放大系数矩阵
22 f = np.zeros((6, 7)) # 科氏力参数矩阵
23
24 # 计算映射系数m和科氏力参数f
25 for In in range(7):
26     for Jn in range(6):
27         # 计算当前点的纬度 (考虑到矩阵翻转)
28         current_lat = lat_start - (5-Jn) * (d / (a * np.pi /
            180)) # 5-Jn是因为矩阵会被翻转
29         # 地图放大系数只与纬度有关
30         m[Jn, In] = 1 / np.cos(np.deg2rad(current_lat))
31         # 科氏力参数只与纬度有关
32         f[Jn, In] = 2 * omega * np.sin(np.deg2rad(
            current_lat))
33
34 # 翻转矩阵使得左下角为原点
35 m = np.flipud(m)
36 f = np.flipud(f)
37
38 # 打印结果
39 print("地图放大系数m (麦卡托投影, 中南半岛区域):")
40 print(m)
41 print("\n科氏力参数f (麦卡托投影, 中南半岛区域):")
42 print(f)
43
44 # 将结果保存到Excel文件
45 df_m = pd.DataFrame(m)
46 df_f = pd.DataFrame(f)
47
48 df_m.to_excel('map_scale_exp2.xlsx', index=False, header=
    False)
49 df_f.to_excel('coriolis_force_exp2.xlsx', index=False,
    header=False)
50
51 # 计算每个网格点对应的经纬度
52 lon_grid = np.zeros((6, 7))
53 lat_grid = np.zeros((6, 7))

```

```

54
55     for i in range(7):
56         for j in range(6):
57             # 计算经度变化（根据距离和纬度计算）
58             delta_lon = (i * d) / (a * np.cos(np.deg2rad(
                    lat_start)) * np.pi / 180)
59             # 计算纬度变化（根据距离计算）
60             delta_lat = (j * d) / (a * np.pi / 180)
61
62             lon_grid[j, i] = lon_start + delta_lon
63             lat_grid[j, i] = lat_start - delta_lat # 纬度向南递
                    减
64
65     # 翻转经纬度网格以匹配m和f矩阵
66     lon_grid = np.flipud(lon_grid)
67     lat_grid = np.flipud(lat_grid)
68
69     # 创建图形和子图
70     plt.figure(figsize=(15, 12))
71
72     # 绘制地图放大系数的热力图
73     plt.subplot(221)
74     im1 = plt.imshow(m, cmap='viridis')
75     plt.colorbar(im1, label='地图放大系数')
76     plt.title('麦卡托投影地图放大系数分布（中南半岛区域）')
77
78     # 设置经纬度刻度
79     xticks = np.arange(7)
80     yticks = np.arange(6)
81     plt.xticks(xticks, [f'{lon_grid[0,i]:.1f}°E' for i in range
                    (7)])
82     plt.yticks(yticks, [f'{lat_grid[i,0]:.1f}°N' for i in range
                    (6)])
83     plt.xlabel('经度')
84     plt.ylabel('纬度')
85
86     # 绘制科氏力参数的热力图
87     plt.subplot(222)
88     im2 = plt.imshow(f, cmap='viridis')
89     plt.colorbar(im2, label='科氏力参数')
90     plt.title('麦卡托投影科氏力参数分布（中南半岛区域）')
91
92     # 设置经纬度刻度

```

```

93     plt.xticks(xticks, [f'{lon_grid[0,i]:.1f}°E' for i in range
94                       (7)])
95     plt.yticks(yticks, [f'{lat_grid[i,0]:.1f}°N' for i in range
96                       (6)])
97
98     # 计算WPS设置下的地图放大系数和科氏力参数
99     # 使用c.m文件中的公式进行计算
100    m_wps = np.zeros((6, 7))
101    f_wps = np.zeros((6, 7))
102
103    for In in range(7):
104        for Jn in range(6):
105            # 根据c.m文件中的公式计算
106            m_wps[Jn, In] = np.sqrt((a * np.cos(np.deg2rad(
107                lat_start)))**2 + (Jn * d)**2) / a
108            f_wps[Jn, In] = 2 * omega * np.sin(Jn * d / np.sqrt
109                ((a * np.cos(np.deg2rad(lat_start)))**2 + (Jn * d
110                )**2))
111
112    # 翻转矩阵
113    m_wps = np.flipud(m_wps)
114    f_wps = np.flipud(f_wps)
115
116    # 绘制WPS设置下的地图放大系数热力图
117    plt.subplot(223)
118    im3 = plt.imshow(m_wps, cmap='viridis')
119    plt.colorbar(im3, label='地图放大系数 (WPS)')
120    plt.title('WPS设置下的地图放大系数分布')
121
122    # 设置经纬度刻度
123    plt.xticks(xticks, [f'{lon_grid[0,i]:.1f}°E' for i in range
124                      (7)])
125    plt.yticks(yticks, [f'{lat_grid[i,0]:.1f}°N' for i in range
126                      (6)])
127    plt.xlabel('经度')
128    plt.ylabel('纬度')
129
130    # 绘制WPS设置下的科氏力参数热力图
131    plt.subplot(224)
132    im4 = plt.imshow(f_wps, cmap='viridis')
133    plt.colorbar(im4, label='科氏力参数 (WPS)')

```

```

129     plt.title('WPS设置下的科氏力参数分布')
130
131     # 设置经纬度刻度
132     plt.xticks(xticks, [f'{lon_grid[0,i]:.1f}°E' for i in range
133                        (7)])
134     plt.yticks(yticks, [f'{lat_grid[i,0]:.1f}°N' for i in range
135                        (6)])
136
137     plt.xlabel('经度')
138     plt.ylabel('纬度')
139
140     # 调整子图之间的间距
141     plt.tight_layout()
142
143     # 显示图形
144     plt.show()
145
146     # 计算差异
147     m_diff = m - m_wps
148     f_diff = f - f_wps
149
150     # 打印差异结果
151     print("\n地图放大系数差异 (自定义 - WPS):")
152     print(m_diff)
153     print("\n科氏力参数差异 (自定义 - WPS):")
154     print(f_diff)
155
156     # 打印经纬度网格
157     print("\n经度网格 (°E):")
158     print(lon_grid)
159     print("\n纬度网格 (°N):")
160     print(lat_grid)
161
162     print("\n结果已保存到Excel文件中")

```

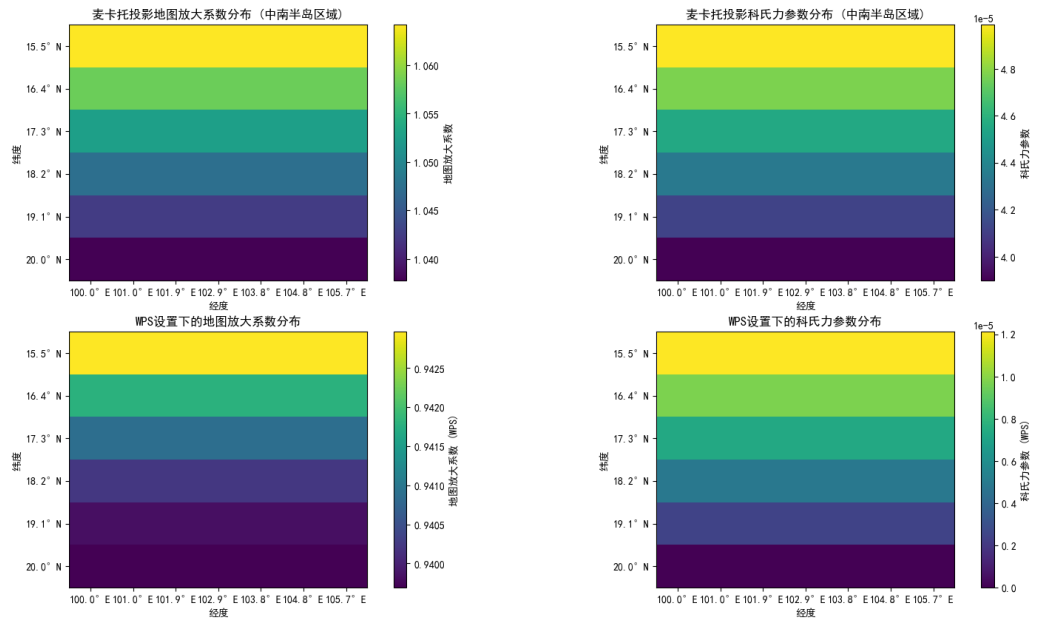


Figure 3: 对比分析