数值天气预报课程作业

吴浩杰 20221170212

1 实习九

时间平滑

1.1 实习目的

通过编程实现时间平滑，使学生掌握边界和区域内网格点空时间平滑的基本原理及计算方法。

1.2 实习内容

(1) 编写时间平滑子程序，输入前后三个时间层的 u、v、z 和平滑系数，输出平滑结果。

(2) 采用欧洲中心 ERA5 再分析资料的 500 hPa 重力位势高度场，采用兰勃特投影方式，以 1979 年 1 月 10 日 00 时为例进行时间平滑。

Listing 1: 时间平滑

```
import os
import numpy as np
import xarray as xr
import matplotlib.pyplot as plt
import cartopy.crs as ccrs
from cartopy.util import add_cyclic_point
from tqdm import tqdm

def time_smooth_fields(u, v, z, smooth_factor=0.5):
    """
    时间平滑子程序
    参数:
        u, v, z: 三个时间层的风场和位势高度场数据，每个数组形状为(3, lat, lon)
        smooth_factor: 平滑系数，默认为0.5
    返回:
```

```
        平滑后的u、v、z场
    """
    def smooth_single_field(field):
        # 获取中间时刻的场
        field_smooth = field[1].copy()
        # 对非边界点进行平滑
        field_smooth[1:-1, 1:-1] = field[1][1:-1, 1:-1] +
            smooth_factor * (
             field[0][1:-1, 1:-1] + field[2][1:-1, 1:-1] - 2.0 *
                 field[1][1:-1, 1:-1]
        ) / 2.0
        return field_smooth

    # 分别对u、v、z场进行平滑
    u_smooth = smooth_single_field(u)
    v_smooth = smooth_single_field(v)
    z_smooth = smooth_single_field(z)

    return u_smooth, v_smooth, z_smooth

def load_500hpa_data(nc_file):
    """
    读取500hPa位势高度场数据
    """
    print("正在读取数据...")
    with xr.open_dataset(nc_file, engine='netcdf4', decode_times
        =False) as ds:
        # 选择500hPa层的连续三个时次数据
        z = ds['z'].sel(pressure_level=500).isel(valid_time=
            slice(0, 3))
        height = (z / 9.80665).values  # 转换为米

        # 获取原始经纬度数据
        lon = ds.longitude.values
        lat = ds.latitude.values

        # 确保经度范围在[-180, 180)之间
        lon = ((lon + 180) % 360) - 180
        idx = np.argsort(lon)
        lon = lon[idx]

        # 调整数据的经度顺序
        height = height[:, :, idx]
```

```
55
56      return height, lon, lat
57
58  def plot_height_fields(height_original, height_smooth, lon, lat)
        :
59      """
60      绘制原始场和平滑后的场对比图
61      """
62      # 添加循环点
63      height_original_cyc, lon_cyc = add_cyclic_point(
            height_original, coord=lon)
64      height_smooth_cyc, _ = add_cyclic_point(height_smooth, coord
            =lon)
65
66      # 创建兰勃特投影
67      proj = ccrs.LambertConformal(
68          central_longitude=120,
69          central_latitude=50,
70          standard_parallels=(30, 60)
71      )
72
73      # 创建画布和子图
74      fig = plt.figure(figsize=(15, 7))
75
76      # 绘制原始场
77      ax1 = fig.add_subplot(121, projection=proj)
78      plot_single_field(ax1, height_original_cyc, lon_cyc, lat, "
            原始500hPa位势高度场")
79
80      # 绘制平滑场
81      ax2 = fig.add_subplot(122, projection=proj)
82      plot_single_field(ax2, height_smooth_cyc, lon_cyc, lat, "时
            间平滑后500hPa位势高度场")
83
84      # 添加总标题
85      title = '1979年1月10日 500hPa位势高度场分析\n'
86      subtitle = '单位: m'
87      plt.suptitle(title + subtitle, fontsize=16, y=0.98)
88
89      # 调整子图之间的间距
90      plt.subplots_adjust(wspace=0.2, top=0.85)
91
92  def plot_single_field(ax, field, lon, lat, title):
```

```
    """
    绘制单个高度场子图
    """
    # 设置地图范围
    ax.set_extent([60, 180, 20, 80], crs=ccrs.PlateCarree())

    # 添加海岸线
    ax.coastlines('50m', linewidth=0.8)

    # 添加网格线
    gl = ax.gridlines(crs=ccrs.PlateCarree(), draw_labels=True,
                      linestyle='--', linewidth=0.5, color='gray')
    gl.top_labels = False
    gl.right_labels = False
    gl.xlocator = plt.FixedLocator(np.arange(60, 181, 30))
    gl.ylocator = plt.FixedLocator(np.arange(20, 81, 10))
    gl.xlabel_style = gl.ylabel_style = {'size': 10}

    # 绘制等值线
    levels = np.arange(5000, 5751, 150)
    cs = ax.contour(lon, lat, field, levels=levels,
                    colors='black', linewidths=1.2,
                    transform=ccrs.PlateCarree())

    # 添加等值线标签
    plt.clabel(cs, fmt='%d', inline=True, fontsize=9)

    # 添加子图标题
    ax.set_title(title, fontsize=14, pad=10, y=1.05)

def main():
    # 设置中文字体
    plt.rcParams['font.sans-serif'] = ['SimHei']
    plt.rcParams['axes.unicode_minus'] = False

    # 数据文件路径
    nc_file = 'geo_197901.nc'
    if not os.path.exists(nc_file):
        raise FileNotFoundError(f"找不到文件: {nc_file}")

    # 读取数据
    height, lon, lat = load_500hpa_data(nc_file)
```

```
135
136     # 进行时间平滑
137     print("正在进行时间平滑...")
138     # 由于示例中只需要对z场进行平滑，我们传入相同的数据作为u和v
139     _, _, height_smooth = time_smooth_fields(height, height,
            height)
140
141     # 绘制对比图
142     print("正在绘制高度场对比图...")
143     plot_height_fields(height[1], height_smooth, lon, lat)
144
145     # 显示图像
146     plt.show()
147
148 if __name__ == '__main__':
149     main()
```

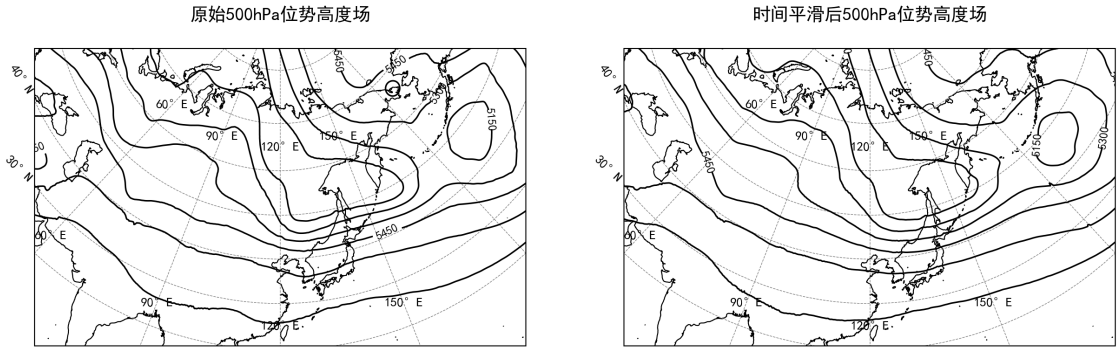

Figure 1: 平滑前后对比

可见时间平滑后的的等值线更为光滑。

1.3 实验九习题

1.3.1 习题一

采用兰勃特投影方式，对欧洲中心 ERA5 再分析资料 1979 年 1 月 10 日 00 时的 500hPa 重力位势高度场进行三点时间平滑。

Listing 2: 实验九习题一

```
1     import os
2     import numpy as np
3     import xarray as xr
4     import matplotlib.pyplot as plt
5     import cartopy.crs as ccrs
6     from cartopy.util import add_cyclic_point
7     from tqdm import tqdm
```

```
8
9       def load_200hpa_fields(nc_file):
10          """
11          读取200hPa的u、v、z要素场数据的连续三个时次
12          返回：三个时次的要素场数据、经度、纬度
13          """
14          print("正在读取数据...")
15          with xr.open_dataset(nc_file, engine='netcdf4',
                decode_times=False) as ds:
16              # 选择200hPa层的连续三个时次数据
17              fields = {}
18              for var in ['u', 'v', 'z']:
19                  data = ds[var].sel(pressure_level=200).isel(
                        valid_time=slice(0, 3))
20                  if var == 'z':
21                      data = data / 9.80665  # 转换为位势米
22                  fields[var] = data.values
23
24              # 获取原始经纬度数据
25              lon = ds.longitude.values
26              lat = ds.latitude.values
27
28              # 确保经度范围在[-180, 180)之间
29              lon = ((lon + 180) % 360) - 180
30              idx = np.argsort(lon)
31              lon = lon[idx]
32
33              # 调整所有时次数据的经度顺序
34              for var in fields:
35                  fields[var] = fields[var][:, :, idx]
36
37          return fields, lon, lat
38
39      def time_smooth(data_n_minus_1, data_n, data_n_plus_1, s
            =0.5):
40          """
41          对中间时次进行三点时间平滑
42          参数:
43          data_n_minus_1, data_n, data_n_plus_1: 连续三个时次的数
                据
44          s: 平滑系数，默认0.5
45          """
46          data_smooth = data_n.copy()
```

```
47          data_smooth[1:-1, 1:-1] = data_n[1:-1, 1:-1] + s * (
48              data_n_minus_1[1:-1, 1:-1] + data_n_plus_1[1:-1,
                    1:-1] - 2.0 * data_n[1:-1, 1:-1]
49          ) / 2.0
50          return data_smooth
51
52      def plot_fields(field, field_smooth, lon, lat, var_name,
            units):
53          """
54          在两个子图中分别绘制原始场和平滑后的场
55          """
56          # 添加循环点
57          field_cyc, lon_cyc = add_cyclic_point(field, coord=lon)
58          field_smooth_cyc, _ = add_cyclic_point(field_smooth,
                coord=lon)
59
60          # 创建兰勃特投影
61          proj = ccrs.LambertConformal(
62              central_longitude=120,
63              central_latitude=50,
64              standard_parallels=(30, 60)
65          )
66
67          # 创建画布和子图
68          fig = plt.figure(figsize=(15, 7))
69
70          # 绘制原始场
71          ax1 = fig.add_subplot(121, projection=proj)
72          plot_single_field(ax1, field_cyc, lon_cyc, lat, f"原始{
                var_name}场")
73
74          # 绘制平滑场
75          ax2 = fig.add_subplot(122, projection=proj)
76          plot_single_field(ax2, field_smooth_cyc, lon_cyc, lat, f
                "时间平滑后{var_name}场")
77
78          # 添加总标题
79          title = f'1979年1月10日 200hPa {var_name}场分析\n'
80          subtitle = f'单位: {units}'
81          plt.suptitle(title + subtitle, fontsize=16, y=0.98)
82
83          # 调整子图之间的间距
84          plt.subplots_adjust(wspace=0.2, top=0.85)
```

```
85
86      def plot_single_field(ax, field, lon, lat, title):
87          """
88          绘制单个要素场子图
89          """
90          # 设置地图范围
91          ax.set_extent([60, 180, 20, 80], crs=ccrs.PlateCarree())
92
93          # 添加海岸线
94          ax.coastlines('50m', linewidth=0.8)
95
96          # 添加网格线
97          gl = ax.gridlines(crs=ccrs.PlateCarree(), draw_labels=
                True,
98                            linestyle='--', linewidth=0.5, color='
                                gray')
99          gl.top_labels = False
100         gl.right_labels = False
101         gl.xlocator = plt.FixedLocator(np.arange(60, 181, 30))
102         gl.ylocator = plt.FixedLocator(np.arange(20, 81, 10))
103         gl.xlabel_style = gl.ylabel_style = {'size': 10}
104
105         # 绘制等值线
106         cs = ax.contour(lon, lat, field, colors='black',
                linewidths=1.2,
107                         transform=ccrs.PlateCarree())
108
109         # 添加等值线标签
110         plt.clabel(cs, fmt='%d', inline=True, fontsize=9)
111
112         # 添加子图标题
113         ax.set_title(title, fontsize=14, pad=10, y=1.05)
114
115     def main():
116         # 设置中文字体
117         plt.rcParams['font.sans-serif'] = ['SimHei']
118         plt.rcParams['axes.unicode_minus'] = False
119
120         # 数据文件路径
121         nc_file = 'geo_197901_200hpa.nc'
122         if not os.path.exists(nc_file):
123             raise FileNotFoundError(f"找不到文件：{nc_file}")
124
```

```
        # 读取数据
        fields, lon, lat = load_200hpa_fields(nc_file)

        # 变量名称和单位
        var_info = {
            'u': ('纬向风', 'm/s'),
            'v': ('经向风', 'm/s'),
            'z': ('位势高度', 'm')
        }

        # 对每个要素场进行时间平滑并绘图
        print("正在进行时间平滑和绘图...")
        for var in tqdm(fields):
            # 获取三个时次的数据
            data_n_minus_1 = fields[var][0]
            data_n = fields[var][1]
            data_n_plus_1 = fields[var][2]

            # 进行时间平滑
            data_smooth = time_smooth(data_n_minus_1, data_n,
                data_n_plus_1)

            # 绘制对比图
            plot_fields(data_n, data_smooth, lon, lat, var_info[
                var][0], var_info[var][1])
            plt.show()

    if __name__ == '__main__':
        main()
```

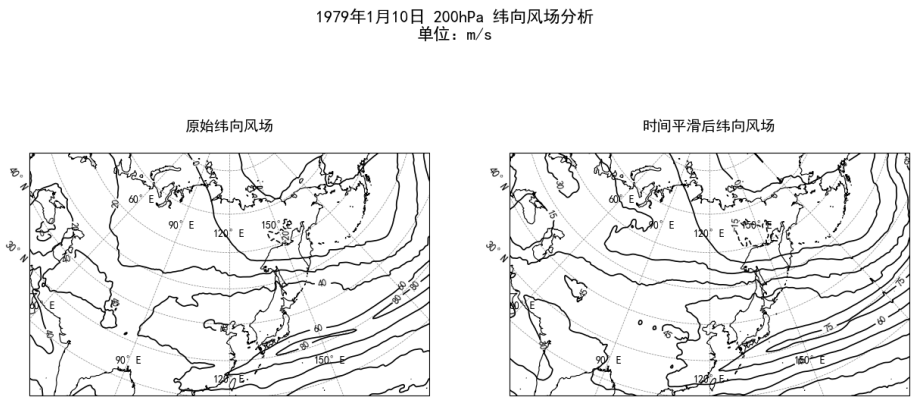

Figure 2: u 分量场

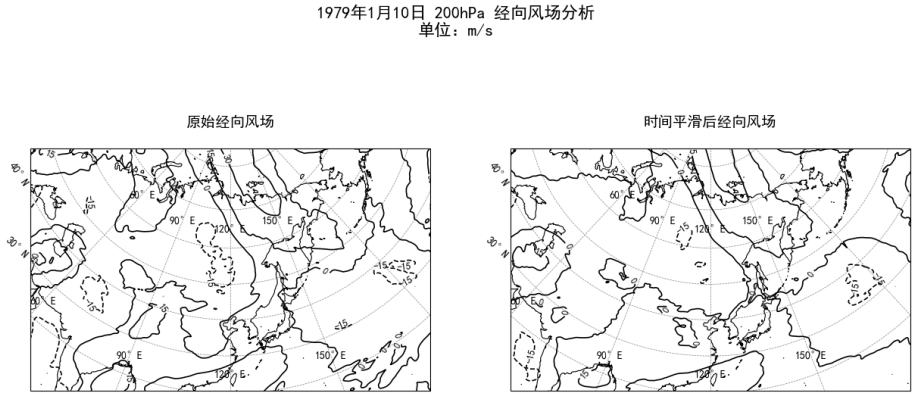

Figure 3: v 分量场

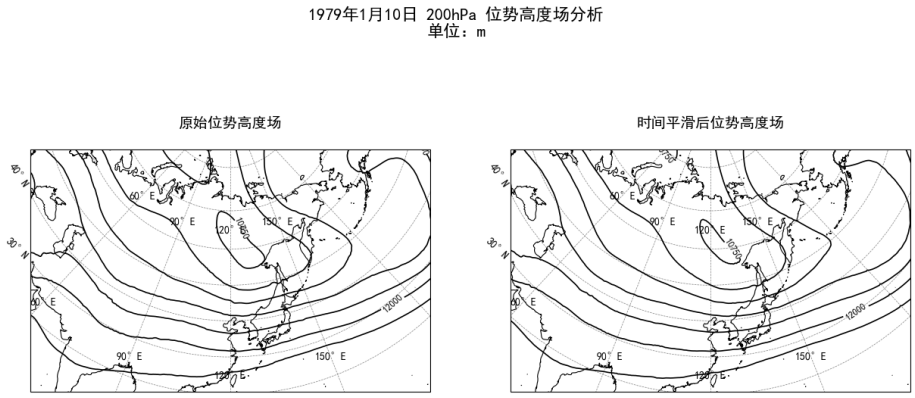

Figure 4: z 分量场

可以发现，三点时间平滑后极值都被平滑掉，但总体的空间分布还是稳定的。