姓名： 栗浩宇 学号： 20222490 d

《GNSS定位新技术及数据处理方法》课后大作业（二）

**GNSS 空间天气监测分析实验**

**任务部分**：每位同学从<http://www.igs.gnsswhu.cn>或者<https://rinex.geodetic.gov.hk/rinex3>或者下载原始观测数据，然后利用<http://192.144.235.90/NEU2DTEC>处理生成VTEC数据，在此基础上，至少完成如下两项任务中的一项：

**任务一：**1）对比分析至少6个不同纬度测站的电离层差异，并探讨原因；2）利用多项式拟合生成香港区域电离层模型。

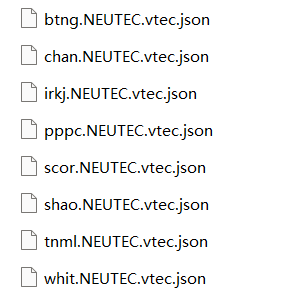
1. **算法**

1 不同纬度测站的电离层差异分析

从http://www.igs.gnsswhu.cn下载了2025年4月22日（年积日112）不同纬度测站的观测值文件,一共下载了8个测站，其纬度信息如下表所示，然后下载了2025年4月22日的精密星历文件，导入NEU2DTEC生成VTEC数据。

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 测站 | BTNG | PPPC | TNML | SHAO | CHAN | IRKJ | WHIT | SCOR |
| 纬度/° | 1.439 | 9.773 | 24.798 | 31.100 | 43.791 | 52.219 | 60.751 | 70.485 |

转换后的VTEC数据如下，共有8个json文件：



接下来可以用网站中提供的绘图代码绘制这8个测站的VTEC随时间（历元）的变化情况， 其中发现BTNG测站数据极少，TNML测站数据不符合要求，所以采用剩余的6个测站进行分析，用Python对这**6个**测站进行统计分析，统计内容包括：均值、最大值、最小值、变化幅度，以及相关系数的计算和相关性是否显著的判断。

**相关系数**的计算基于如下公式(Pearson公式)：

其中：

和 是两个变量的观测值（此处为纬度和平均VTEC值）

和 是它们各自的平均值

是样本数量（此处为测站数量）

相关系数范围是 1表示完全正相关，-1表示完全负相关，0表示无线性相关

**相关性检验**使用T分布，检验统计量t计算公式如下：

其中是相关系数，是样本数量。

然后根据检验统计量来确定值，如果值小于0.05，则“相关性显著”。

以上算法结果的展示以及电离层差异原因的分析均在下文中进行。

2 多项式拟合香港地区电离层模型

同样地，下载了2025年4月22日香港地区的18个测站的观测值文件，然后进行了VTEC的计算，计算后发现，其中HKFN、HKCL、HKMW、HKQT测站的数据有问题，直接排除掉这4个测站，使用剩下的**14个**测站进行后续的模型生成。要注意的是：这里面还要用到测站的观测值文件(.rnx)，来读取测站的经纬度信息。

下面是本次多项式拟合香港地区电离层模型的算法介绍：

首先要进行经纬度的归一化，所用公式如下：

将经纬度数据归一化到区间，这是为了防止高阶多项式计算中出现的数值不稳定问题。

接下来进行多项式特征的构建，将二维输入(经度,纬度)转换为多项式特征矩阵。例如，当多项式阶数为5时，对于输入特征（归一化后的经纬度），会生成21个特征：

接着就是多项式系数的求解问题，这里引入ElasticNet回归方法，主要是为了处理过拟合问题和提高模型泛化能力。

如果是传统的多项式拟合，使用普通的最小二乘法，其目标函数为：

其中，是多项式特征矩阵，是待求系数向量，是VTEC观测值。

ElasticNet回归的目标函数为：

其中：，，。

综合上述过程，任意一点的VTEC值可通过下面这个公式预测：

其中，是归一化到[-1,1]区间的经度和纬度坐标 ，是多项式的最高阶数(代码中的DEGREE参数，是通过ElasticNet回归求解的系数)。

后续还用到的评估指标有：

决定系数()：

均方根误差()：

平均绝对误差()：

对于残差插值，还用到三次样条插值公式：

以上就是多项式拟合香港地区电离层模型的所有算法，下面就用14个测站的数据，选择手动选择某一时刻进行电离层模型的多项式拟合。

1. **代码**

1 不同纬度测站数据的统计分析(vtec\_statistics.py)

import json

import numpy as np

import pandas as pd

import os

import matplotlib.pyplot as plt

*# 设置中文字体*

plt.rcParams['font.sans-serif'] = ['SimHei', 'DejaVu Sans', 'Arial Unicode MS']  *# 优先使用的中文字体*

plt.rcParams['axes.unicode\_minus'] = False  *# 解决负号显示问题*

*# 测站信息*

stations = {

    "PPPC": 9.773,

    "SHAO": 31.100,

    "CHAN": 43.791,

    "IRKJ": 52.219,

    "WHIT": 60.751,

    "SCOR": 70.485

}

*# 按纬度从低到高排序测站*

sorted\_stations = dict(sorted(stations.items(), key=lambda item: item[1]))

*# 读取VTEC数据*

def load\_vtec\_data(station, data\_dir="./"):

    """

    读取VTEC JSON数据

    参数:

        station: 测站名称

        data\_dir: 数据文件目录

    返回:

        vtec\_data: 包含各卫星VTEC数据的字典

    """

*# 查找匹配的文件*

    station\_files = []

    for file in os.listdir(data\_dir):

*# 检查文件名是否包含测站名称且为JSON文件*

        if station.lower() in file.lower() and file.endswith(".vtec.json"):

            station\_files.append(os.path.join(data\_dir, file))

    if not station\_files:

        print(f"警告: 未找到测站 {station} 的VTEC数据文件")

*# 如果找不到文件，返回空字典或使用模拟数据*

        return {}

*# 使用找到的第一个文件*

    file\_path = station\_files[0]

    print(f"读取文件: {file\_path}")

    try:

        with open(file\_path, "r") as file:

            vtec\_data = json.load(file)

        return vtec\_data

    except Exception as e:

        print(f"读取文件 {file\_path} 时出错: {e}")

        return {}

*# 分析VTEC数据统计特征*

def analyze\_vtec\_statistics(vtec\_data):

    """分析VTEC数据的统计特征"""

    all\_values = []

    for prn, values in vtec\_data.items():

*# 过滤掉0值*

        valid\_values = [v for v in values if abs(v) > 1.0e-6]

        if valid\_values:

            all\_values.extend(valid\_values)

    if not all\_values:

        return {

            "mean": 0,

            "max": 0,

            "min": 0,

            "std": 0,

            "range": 0

        }

    return {

        "mean": np.mean(all\_values),

        "max": np.max(all\_values),

        "min": np.min(all\_values),

        "std": np.std(all\_values),

        "range": np.max(all\_values) - np.min(all\_values)

    }

*# 计算每个时间点的平均VTEC*

def calculate\_hourly\_vtec(vtec\_data):

*# 获取第一颗卫星的数据长度*

    prns = list(vtec\_data.keys())

    if not prns:

        return []

    epoch\_count = len(vtec\_data[prns[0]])

    hourly\_vtec = []

    for epoch in range(epoch\_count):

        values = []

        for prn, data in vtec\_data.items():

*# 检查数据有效性并过滤零值和异常值*

            if epoch < len(data) and abs(data[epoch]) > 1.0e-6 and data[epoch] < 200:

                values.append(data[epoch])

*# 只有当有足够有效值时才计算平均值*

        if len(values) >= 3:  *# 至少需要3颗卫星的有效数据*

            hourly\_vtec.append(np.mean(values))

        else:

*# 使用插值而非0值*

            if hourly\_vtec:

                hourly\_vtec.append(hourly\_vtec[-1])  *# 使用前一个有效值*

            else:

                hourly\_vtec.append(np.nan)  *# 标记为缺失值*

*# 对NaN值进行插值处理*

    hourly\_vtec = np.array(hourly\_vtec)

    mask = np.isnan(hourly\_vtec)

    hourly\_vtec[mask] = np.interp(

        np.flatnonzero(mask),

        np.flatnonzero(~mask),

        hourly\_vtec[~mask]

    )

    return hourly\_vtec.tolist()

*# 分析所有测站*

def analyze\_all\_stations():

    """分析所有测站的VTEC数据"""

    station\_stats = {}

    hourly\_data = {}

    for station in sorted\_stations:

        vtec\_data = load\_vtec\_data(station)

        stats = analyze\_vtec\_statistics(vtec\_data)

        hourly\_vtec = calculate\_hourly\_vtec(vtec\_data)

        station\_stats[station] = stats

        hourly\_data[station] = hourly\_vtec

    return station\_stats, hourly\_data

*# 绘制结果*

def plot\_results(station\_stats, hourly\_data):

    """绘制分析结果"""

*# 1. 不同测站VTEC统计值*

    stations = list(station\_stats.keys())

    latitudes = [sorted\_stations[s] for s in stations]

    means = [station\_stats[s]["mean"] for s in stations]

    maxes = [station\_stats[s]["max"] for s in stations]

    mins = [station\_stats[s]["min"] for s in stations]

    ranges = [station\_stats[s]["range"] for s in stations]

*# 创建图形*

    plt.figure(figsize=(10, 8))

*# VTEC随纬度变化*

    plt.plot(latitudes, means, 'o-', label='平均VTEC')

    plt.plot(latitudes, maxes, '^-', label='最大VTEC')

    plt.plot(latitudes, mins, 'v-', label='最小VTEC')

    plt.plot(latitudes, ranges, 's-', label='VTEC变化范围')

    plt.xlabel('纬度 (°N)')

    plt.ylabel('VTEC (TECU)')

    plt.title('不同纬度测站的VTEC统计特征')

    plt.grid(True)

    plt.legend()

    plt.tight\_layout()

    plt.savefig('./result/vtec\_latitude\_analysis.png', dpi=300)

    plt.close()

*# 3. 生成统计表格*

    stats\_df = pd.DataFrame({

        '测站': stations,

        '纬度(°N)': latitudes,

        '平均VTEC(TECU)': means,

        '最大VTEC(TECU)': maxes,

        '最小VTEC(TECU)': mins,

        'VTEC变化幅度(TECU)': ranges

    })

print(stats\_df)

    try:

        os.makedirs('./result', exist\_ok=True)

        stats\_df.to\_csv('./result/vtec\_statistics.csv', index=False, encoding='utf-8-sig')

    except Exception as e:

        print(f"保存UTF-8-sig格式失败: {e}")

    return stats\_df

*# 主分析函数*

def main\_analysis(data\_dir="./"):

    """

    主分析函数

    参数:

        data\_dir: 数据文件目录

    """

*# 检查数据文件*

    print(f"检查目录 {data\_dir} 中的数据文件...")

    all\_files = os.listdir(data\_dir)

    vtec\_files = [f for f in all\_files if f.endswith(".vtec.json")]

    print(f"找到 {len(vtec\_files)} 个VTEC数据文件: {vtec\_files}")

*# 分析所有测站*

    station\_stats = {}

    hourly\_data = {}

    for station in sorted\_stations:

        print(f"\n处理测站: {station}，纬度: {sorted\_stations[station]}°N")

        vtec\_data = load\_vtec\_data(station, data\_dir)

        if not vtec\_data:

            print(f"未找到测站 {station} 的有效数据，跳过此测站")

            continue

        print(f"成功读取 {station} 测站数据，包含 {len(vtec\_data)} 颗卫星的VTEC记录")

        stats = analyze\_vtec\_statistics(vtec\_data)

        hourly\_vtec = calculate\_hourly\_vtec(vtec\_data)

        station\_stats[station] = stats

        hourly\_data[station] = hourly\_vtec

        print(f"测站 {station} 统计数据: 平均VTEC={stats['mean']:.2f} TECU, 最大值={stats['max']:.2f} TECU")

    if not station\_stats:

        print("未找到任何测站的有效数据，无法进行分析")

        return

    stats\_df = plot\_results(station\_stats, hourly\_data)

*# 分析结果*

    print("\n不同纬度测站电离层VTEC差异分析结果:")

    print("-" \* 60)

*# 找出最大VTEC的测站和纬度*

    max\_vtec\_station = stats\_df.loc[stats\_df['最大VTEC(TECU)'].idxmax()]

    print(f"最大VTEC值出现在 {max\_vtec\_station['测站']} 测站 (纬度: {max\_vtec\_station['纬度(°N)']}°N)")

*# 低纬度和高纬度的差异*

    low\_lat\_df = stats\_df[stats\_df['纬度(°N)'] < 30]

    high\_lat\_df = stats\_df[stats\_df['纬度(°N)'] > 55]

    if not low\_lat\_df.empty and not high\_lat\_df.empty:

        low\_lat\_mean = low\_lat\_df['平均VTEC(TECU)'].mean()

        high\_lat\_mean = high\_lat\_df['平均VTEC(TECU)'].mean()

        diff\_percent = (low\_lat\_mean - high\_lat\_mean) / high\_lat\_mean \* 100 if high\_lat\_mean > 0 else 0

        print(f"低纬度地区平均VTEC比高纬度地区高约 {diff\_percent:.1f}%")

*# 电离层赤道异常特征*

    eq\_anomaly\_stations = stats\_df[(stats\_df['纬度(°N)'] >= 10) & (stats\_df['纬度(°N)'] <= 20)]

    if not eq\_anomaly\_stations.empty:

        print(f"观察到电离层赤道异常现象，赤道附近±15°纬度区域VTEC值较高")

*# 尝试计算纬度相关性*

    try:

        from scipy.stats import pearsonr

        if len(stats\_df) >= 3:  *# 至少需要3个点才能计算有意义的相关性*

            corr, p\_value = pearsonr(stats\_df['纬度(°N)'], stats\_df['平均VTEC(TECU)'])

            print(f"VTEC与纬度的相关系数为：{corr:.6f}")

            if p\_value < 0.05:

                print(f"   相关性显著：(p-value={p\_value:.6f})")

            else:

                print(f"   相关性不显著：(p-value={p\_value:.6f})")

        else:

            print("测站数量不足，无法计算VTEC与纬度的相关性")

    except Exception as e:

        print(f"计算相关性时出错: {e}")

    return stats\_df

*# 执行分析*

if \_\_name\_\_ == "\_\_main\_\_":

    data\_dir = "vtecs"

main\_analysis(data\_dir)

2 多项式拟合电离层模型(hk\_ionosphere\_polyfit.py)

import os

import json

import math

from pathlib import Path

import numpy as np

import matplotlib.pyplot as plt

from matplotlib.path import Path as MplPath

from scipy.spatial import ConvexHull, Voronoi, voronoi\_plot\_2d

from sklearn.preprocessing import PolynomialFeatures

from sklearn.linear\_model import ElasticNet

from scipy.interpolate import griddata

import matplotlib.pyplot as plt

*# 设置中文字体*

plt.rcParams['font.sans-serif'] = ['SimHei', 'DejaVu Sans', 'Arial Unicode MS']

plt.rcParams['axes.unicode\_minus'] = False

*# 基本参数*

RNX\_DIR  = Path(r'hkdata')   *# RINEX数据目录*

VTEC\_DIR = Path(r'HKVTEC')  *# VTEC数据目录*

DEGREE     = 5        *# 多项式阶次*

ALPHA      = 0.0009  *# ElasticNet正则化强度*

L1\_RATIO   = 0.5    *# ElasticNet的L1比例*

SAMPLE\_SEC = 8\*3600  *# 选取 8:00*

INTVL\_SEC  = 30     *# 观测采样间隔（秒）*

def ecef2geodetic(x: float, y: float, z: float):

    """WGS-84 椭球下 ECEF ➜ (lat, lon)  (度)"""

    a  = 6378137.0            *# 长半轴*

    e2 = 6.69437999014e-3     *# 第一偏心率平方*

    b  = a \* math.sqrt(1 - e2)

    lon = math.atan2(y, x)

    p   = math.hypot(x, y)

    theta = math.atan2(z \* a, p \* b)

    lat = math.atan2(z + e2 \* b \* math.sin(theta)\*\*3,

                     p - e2 \* a \* math.cos(theta)\*\*3)

    return math.degrees(lat), math.degrees(lon)

def read\_rinex\_positions(folder: Path):

    """返回 {站名: (lon, lat)}"""

    pos = {}

    for file in folder.glob('\*.rnx'):

        sta = file.stem[:4].lower()

        with file.open('r', errors='ignore') as fh:

            for line in fh:

                if 'APPROX POSITION XYZ' in line:

                    x, y, z = map(float, line.split()[:3])

                    lat, lon = ecef2geodetic(x, y, z)

                    pos[sta] = (lon, lat)

                    break

    return pos

def read\_vtec\_at\_epoch(folder: Path, epoch\_idx: int):

    """返回 {站名: vtec\_value}"""

    vtec = {}

    for file in folder.glob('\*.json'):

        sta = file.stem[:4].lower()

        with file.open('r', encoding='utf-8') as fh:

            data = json.load(fh)

        vals = []

        for prn, series in data.items():

            if epoch\_idx < len(series):

                val = series[epoch\_idx]

                if val is not None and not (isinstance(val, float) and math.isnan(val)):

                    vals.append(val)

        if vals:

            vtec[sta] = float(np.mean(vals))

    return vtec

def normalize(arr):

    """线性映射到 [-1, 1]"""

    mn, mx = float(np.min(arr)), float(np.max(arr))

    return 2\*(arr - mn)/(mx - mn) - 1, mn, mx

*# 读取数据*

stations\_ll = read\_rinex\_positions(RNX\_DIR)

epoch\_idx   = SAMPLE\_SEC // INTVL\_SEC

stations\_vt = read\_vtec\_at\_epoch(VTEC\_DIR, epoch\_idx)

common\_keys = sorted(set(stations\_ll) & set(stations\_vt))

if len(common\_keys) < 4:

    raise RuntimeError('有效测站不足，无法拟合多项式')

lon\_arr = np.array([stations\_ll[k][0] for k in common\_keys])

lat\_arr = np.array([stations\_ll[k][1] for k in common\_keys])

vtec\_arr = np.array([stations\_vt[k]    for k in common\_keys])

*# 多项式拟合*

lon\_n, lon\_min, lon\_max = normalize(lon\_arr)

lat\_n, lat\_min, lat\_max = normalize(lat\_arr)

X\_norm = np.column\_stack((lon\_n, lat\_n))

poly = PolynomialFeatures(degree=DEGREE, include\_bias=True)

X\_poly = poly.fit\_transform(X\_norm)

model = ElasticNet(alpha=ALPHA, l1\_ratio=L1\_RATIO, fit\_intercept=False, tol=1e-8,max\_iter=10000)

model.fit(X\_poly, vtec\_arr)

r2 = model.score(X\_poly, vtec\_arr)

print(f'>>> 已拟合 {DEGREE} 阶多项式，R² = {r2:.3f}，样本数 = {len(common\_keys)}')

*# 生成网格*

lon\_lin = np.linspace(lon\_min, lon\_max, 200)

lat\_lin = np.linspace(lat\_min, lat\_max, 200)

grid\_lon, grid\_lat = np.meshgrid(lon\_lin, lat\_lin)

grid\_pts = np.column\_stack((grid\_lon.ravel(), grid\_lat.ravel()))

*# 利用测站凸包裁剪，避免外插*

hull = ConvexHull(np.column\_stack((lon\_arr, lat\_arr)))

hull\_path = MplPath(np.column\_stack((lon\_arr, lat\_arr))[hull.vertices])

inside = hull\_path.contains\_points(grid\_pts)

*# 归一化 + 预测*

lon\_n\_g = 2\*(grid\_pts[:, 0] - lon\_min)/(lon\_max - lon\_min) - 1

lat\_n\_g = 2\*(grid\_pts[:, 1] - lat\_min)/(lat\_max - lat\_min) - 1

Z\_pred  = model.predict(poly.transform(np.column\_stack((lon\_n\_g, lat\_n\_g))))

Z\_grid = np.full(grid\_lon.shape, np.nan)

Z\_grid.ravel()[inside] = Z\_pred[inside]

*# 计算残差*

station\_pred = model.predict(X\_poly)

residuals = vtec\_arr - station\_pred

abs\_residuals = np.abs(residuals)

*# 图1: 残差热力图*

plt.figure(figsize=(8, 6))

plt.title('残差空间分布热力图', fontsize=15)

grid\_x, grid\_y = np.mgrid[lon\_min:lon\_max:100j, lat\_min:lat\_max:100j]

grid\_residuals = griddata((lon\_arr, lat\_arr), residuals, (grid\_x, grid\_y), method='cubic')

hull\_mask = np.zeros\_like(grid\_residuals, dtype=bool)

for i in range(grid\_x.shape[0]):

    for j in range(grid\_x.shape[1]):

        if not hull\_path.contains\_point((grid\_x[i, j], grid\_y[i, j])):

            hull\_mask[i, j] = True

grid\_residuals = np.ma.array(grid\_residuals, mask=hull\_mask)

cmap = plt.cm.coolwarm

im = plt.pcolormesh(grid\_x, grid\_y, grid\_residuals, cmap=cmap, shading='auto',

                    vmin=-np.max(abs\_residuals), vmax=np.max(abs\_residuals))

plt.colorbar(im, label='残差 (TECU)')

plt.scatter(lon\_arr, lat\_arr, c='k', s=30, marker='^', label='测站位置')

for i, txt in enumerate(common\_keys):

    plt.annotate(f'{txt}\n({residuals[i]:.2f})', (lon\_arr[i], lat\_arr[i]),

                 fontsize=9, ha='center', va='bottom', color='k')

plt.xlabel('经度 (°E)')

plt.ylabel('纬度 (°N)')

plt.grid(True, linestyle='--', alpha=0.3)

plt.legend(loc='best')

rmse = np.sqrt(np.mean(residuals\*\*2))

plt.figtext(0.5, 0.01,

            f"均方根误差(RMSE): {rmse:.3f} TECU | "

            f"最大残差: {np.max(abs\_residuals):.3f} TECU | "

            f"平均绝对残差: {np.mean(abs\_residuals):.3f} TECU",

            ha='center', fontsize=11, bbox=dict(boxstyle="round,pad=0.5",

                                              fc="white", ec="gray", alpha=0.8))

plt.tight\_layout(rect=[0, 0.05, 1, 0.95])

plt.savefig('./result/残差热力图.png', dpi=300, bbox\_inches='tight')

plt.show()

*# 图2: 3D表面*

fig = plt.figure(figsize=(8, 6))

ax = fig.add\_subplot(111, projection='3d')

vtec\_cmap = plt.cm.plasma

surf = ax.plot\_surface(grid\_lon, grid\_lat, Z\_grid,

                       rstride=4, cstride=4,

                       cmap=vtec\_cmap, edgecolor='none', alpha=0.90)

ax.scatter(lon\_arr, lat\_arr, vtec\_arr, c='white', s=35, edgecolor='black', label='测站')

ax.set\_xlabel('经度 (°E)')

ax.set\_ylabel('纬度 (°N)')

ax.set\_zlabel('VTEC (TECU)')

ax.set\_title(f'香港 VTEC 3D表面，使用 {DEGREE} 阶多项式拟合\n'

             f't = {SAMPLE\_SEC//3600:02d}:{SAMPLE\_SEC%3600//60:02d}')

fig.colorbar(surf, shrink=0.55, aspect=12, label='VTEC (TECU)')

plt.legend()

plt.tight\_layout()

plt.savefig('./result/VTEC\_3D表面.png', dpi=300, bbox\_inches='tight')

plt.show()

*# 图3: 2D空间分布图*

plt.figure(figsize=(8, 6))

plt.title('香港地区 VTEC 2D空间分布图', fontsize=15)

contour\_filled = plt.contourf(grid\_lon, grid\_lat, Z\_grid, levels=15, cmap='jet', alpha=0.8)

contour\_lines = plt.contour(grid\_lon, grid\_lat, Z\_grid, levels=8, colors='k', linewidths=0.5, alpha=0.7)

plt.clabel(contour\_lines, inline=True, fontsize=8, fmt='%.1f')

plt.scatter(lon\_arr, lat\_arr, c='white', s=40, edgecolor='black', marker='^', label='测站位置')

for i, txt in enumerate(common\_keys):

    plt.annotate(txt, (lon\_arr[i], lat\_arr[i]), fontsize=9, ha='center', va='bottom')

cbar = plt.colorbar(contour\_filled, label='VTEC (TECU)')

cbar.set\_label('VTEC (TECU)', fontsize=12)

plt.xlabel('经度 (°E)', fontsize=12)

plt.ylabel('纬度 (°N)', fontsize=12)

plt.grid(True, linestyle='--', alpha=0.3)

plt.legend(loc='best')

plt.axis('equal')

plt.tight\_layout()

plt.savefig('./result/VTEC\_2D空间分布图.png', dpi=300, bbox\_inches='tight')

plt.show()

print("\n图表生成完成！共生成3张图：")

print("1. 残差热力图")

print("2. VTEC\_3D表面图")

print("3. VTEC\_2D空间分布图")

1. **结果**
2. 不同纬度测站的电离层差异分析
   1. 电离层数据的时序可视化

这部分直接使用网站提供的代码，得到结果如下：

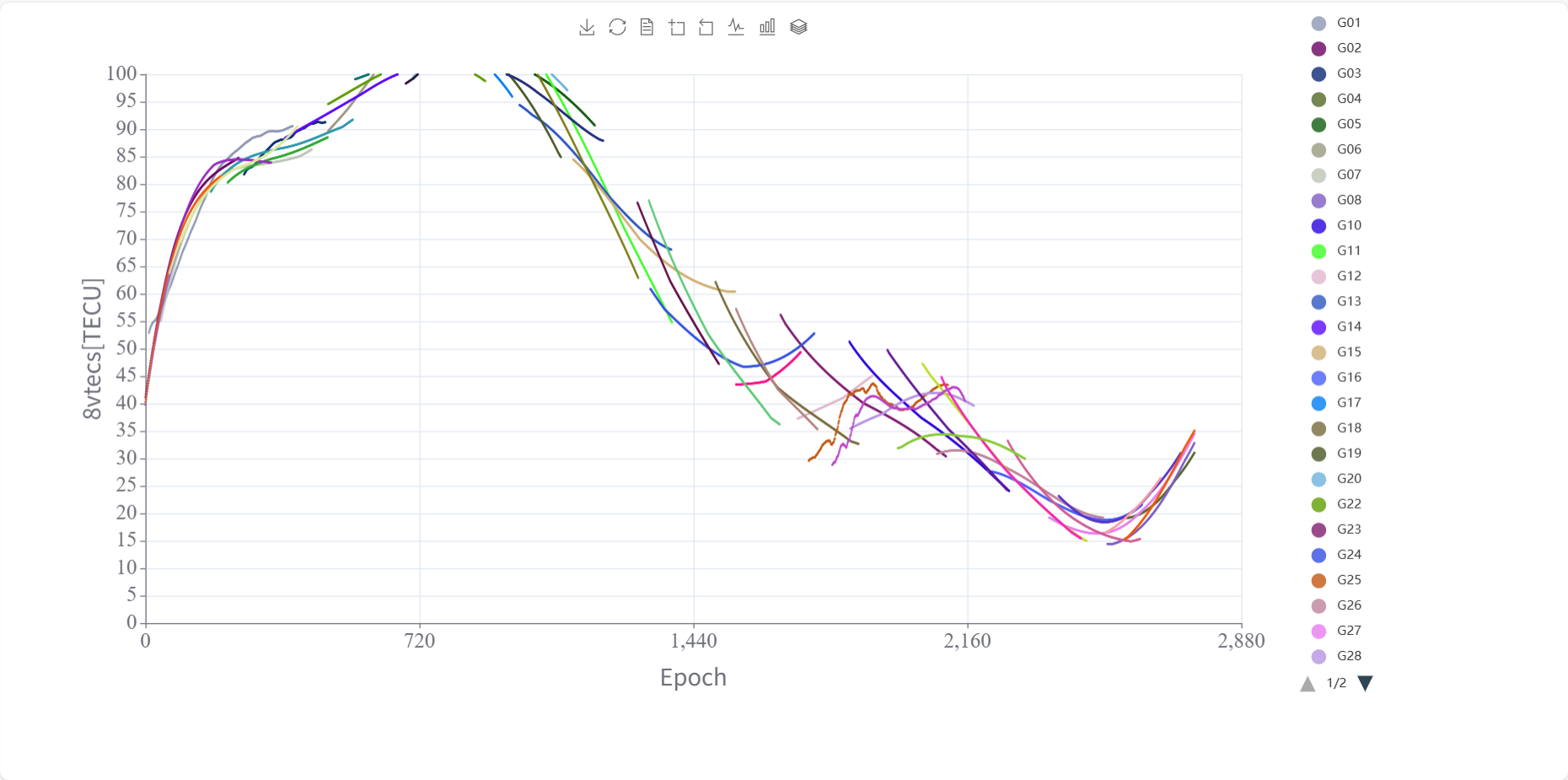


图1 PPPC测站

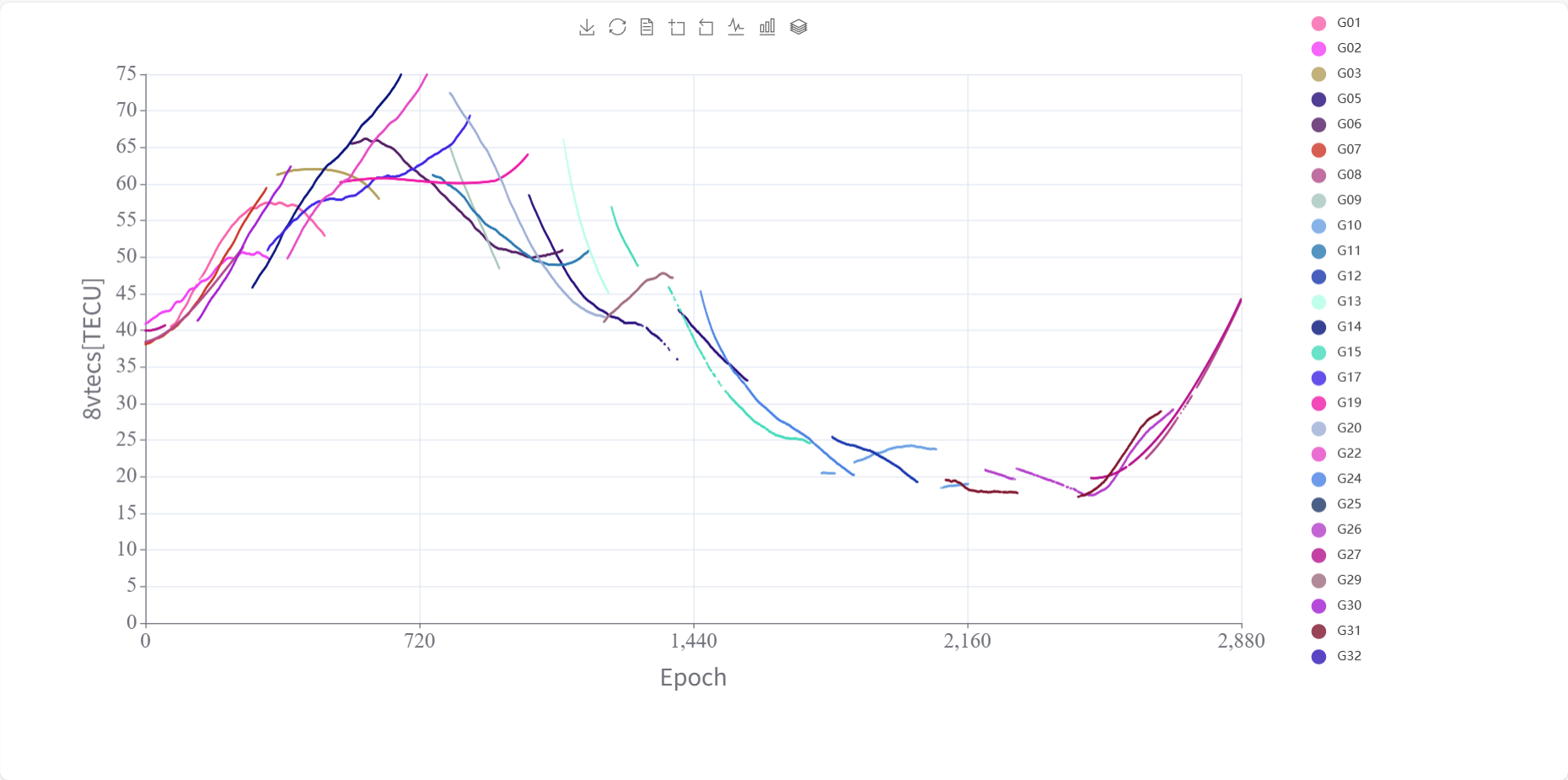


图2 SHAO测站

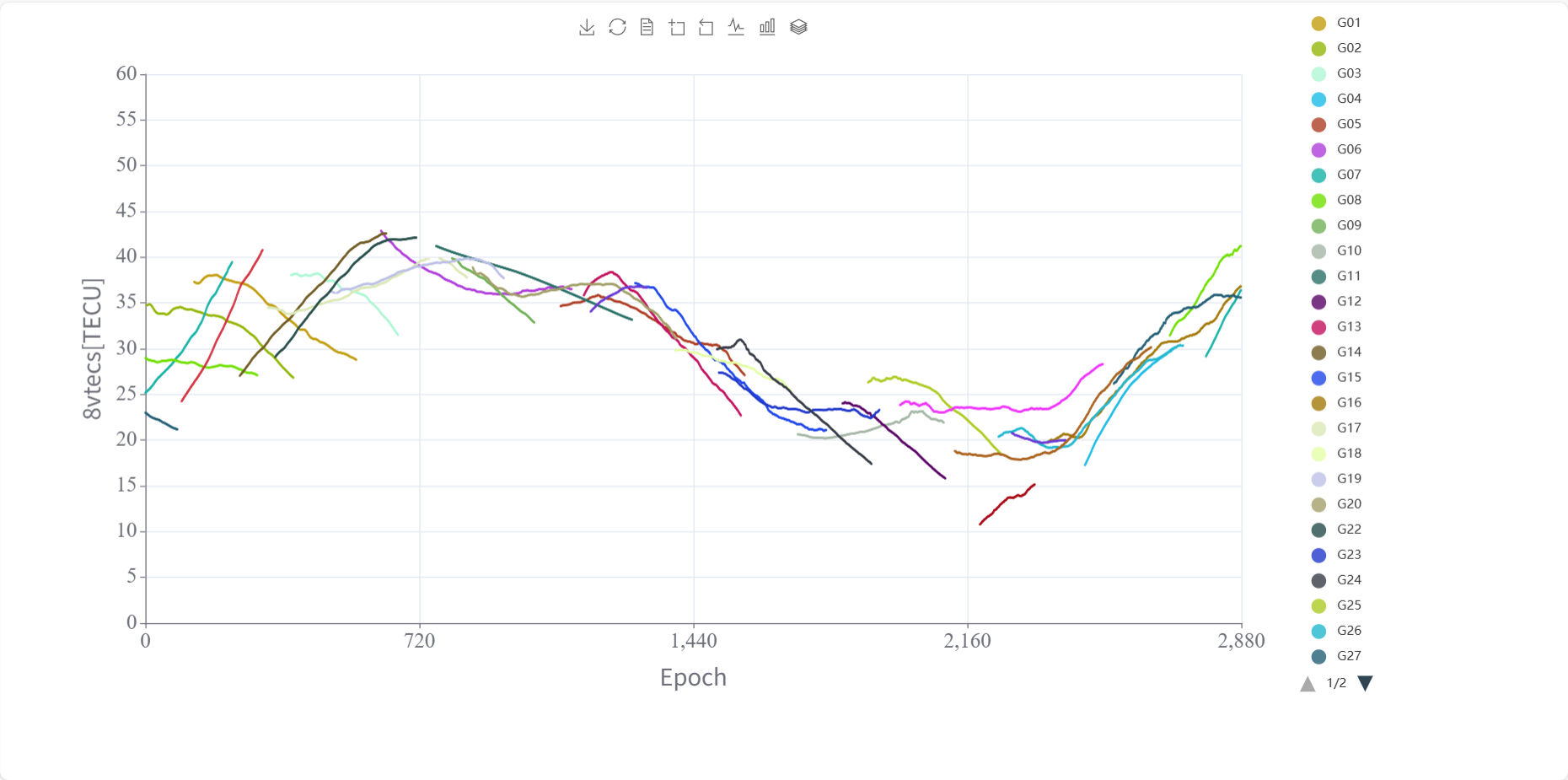


图3 CHAN测站

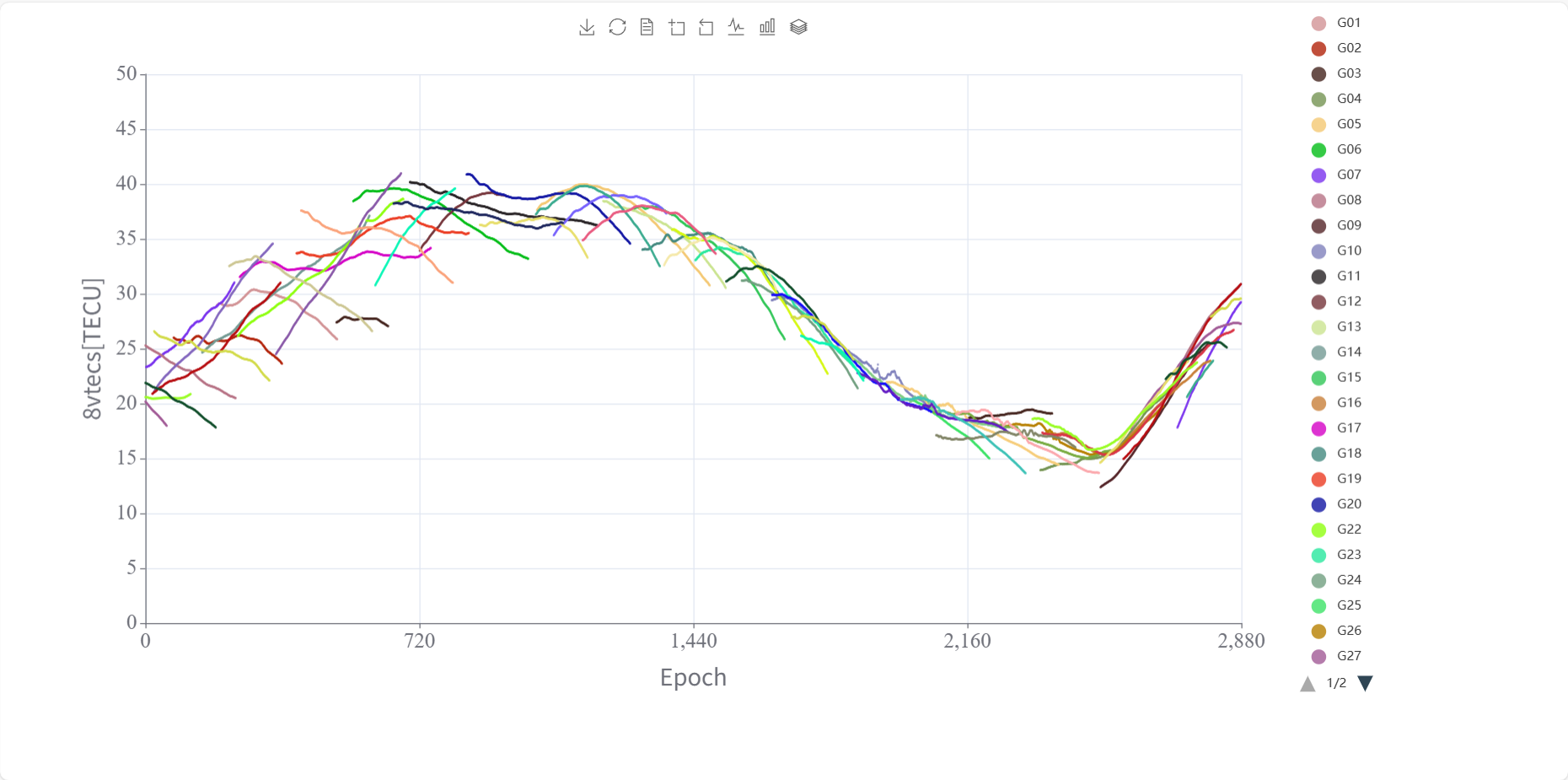


图4 IRKJ测站

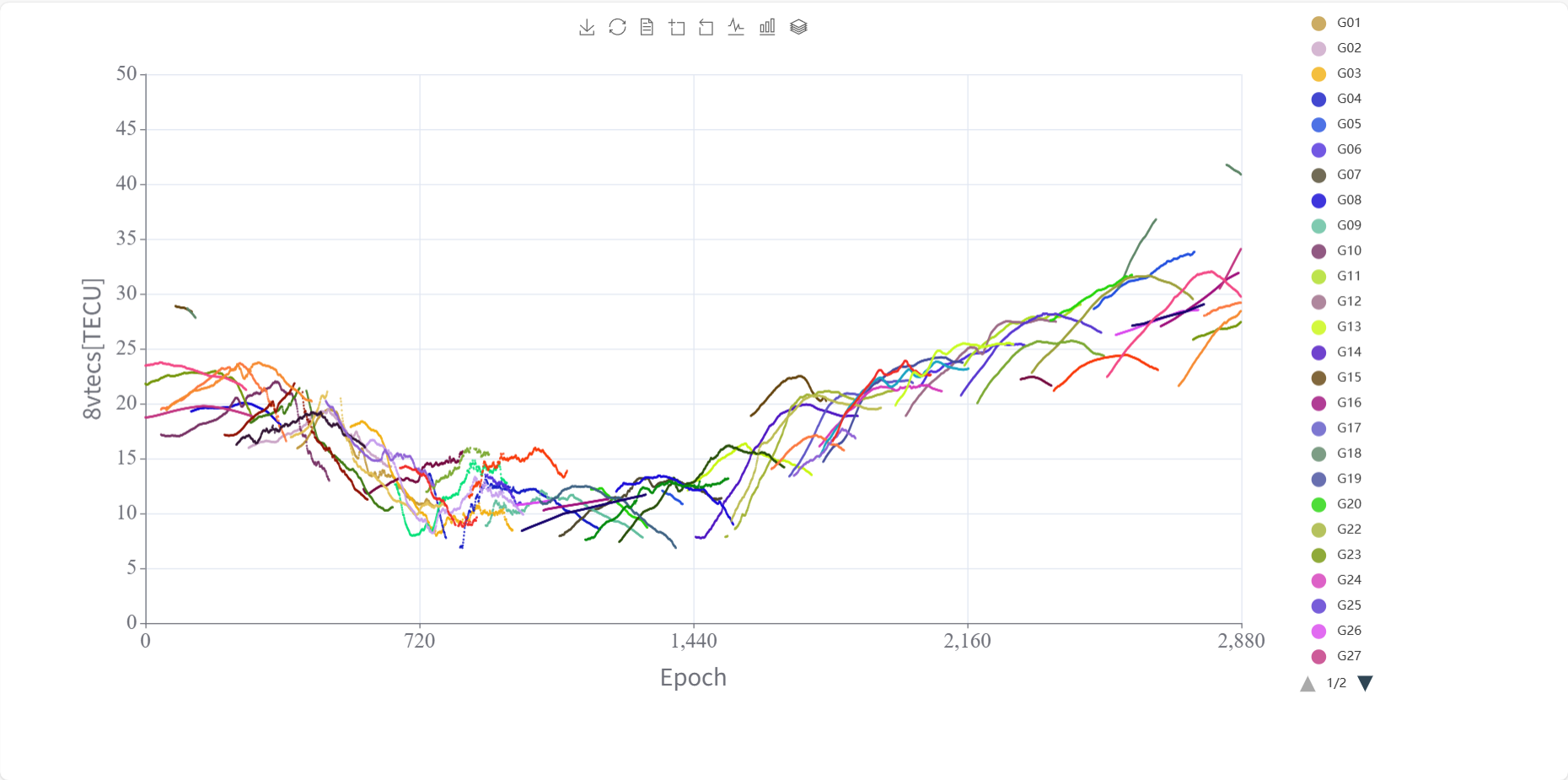


图5 WHIT测站

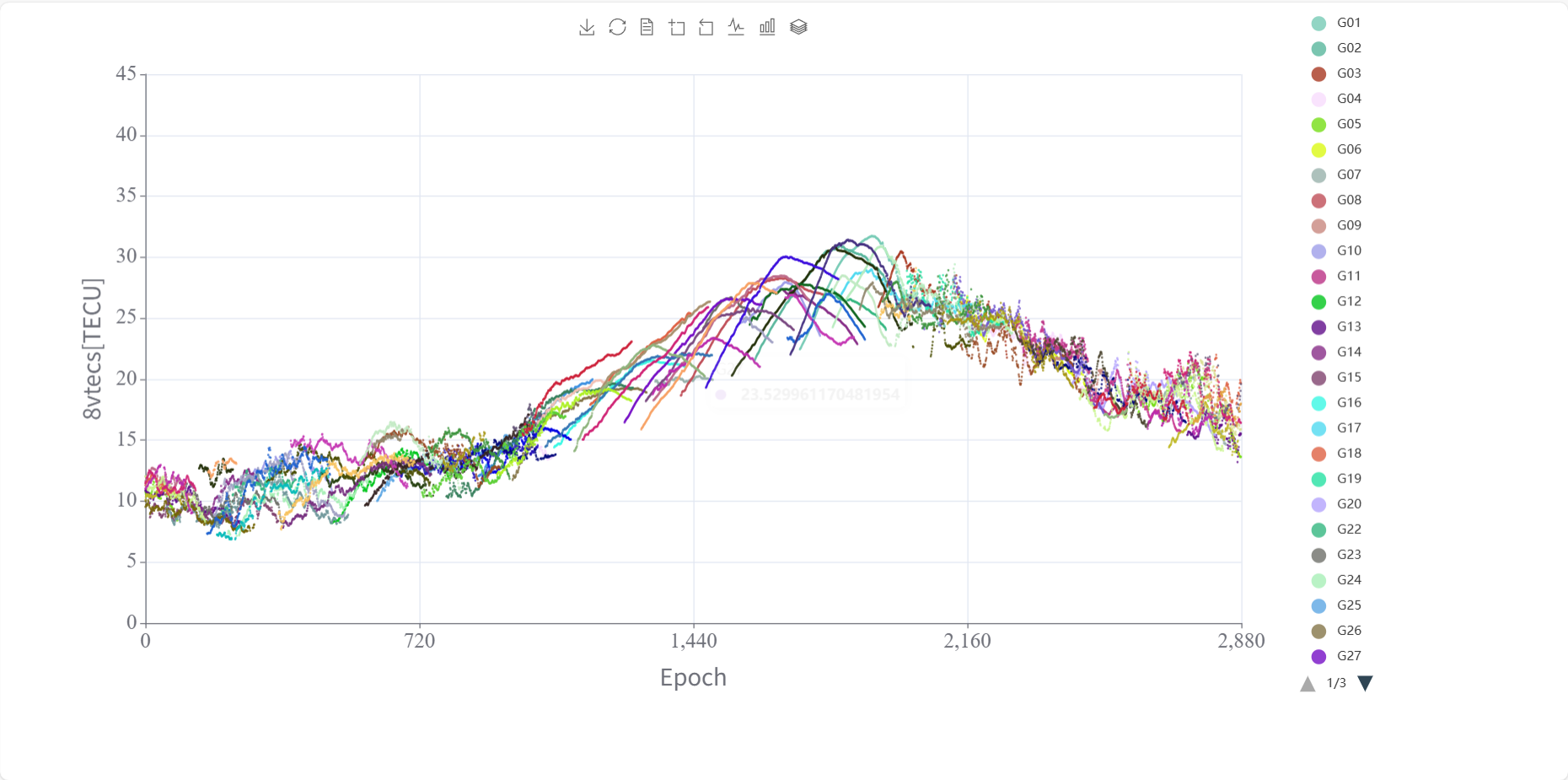
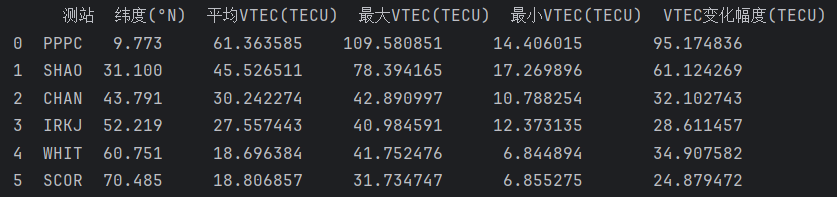


图6 SCOR测站

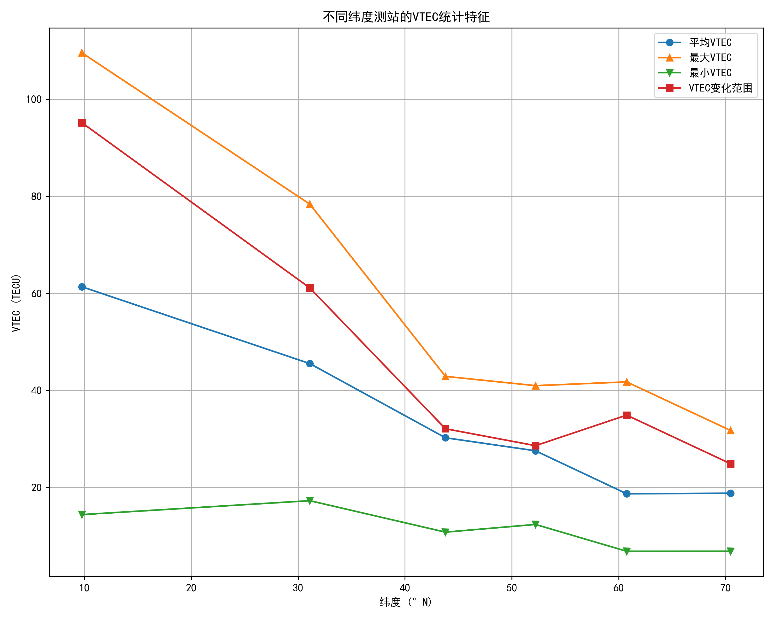
1.2 统计结果

运行代码后，会得到这6个测站的统计数据结果(打印并输出csv文件)，如下所示：



除此之外，还会得到差异分析结果和统计数据可视化结果：





1.3 对比分析6个测站电离层差异

我把纬度低于 30°划为低纬度地区，纬度在30°到55°之间划为中纬度地区，纬度高于 55°划为高纬度地区。

统计发现：

**低纬度地区**的VTEC值最高，显示出明显的高VTEC特征。低纬度地区的VTEC波动较大，尤其是最大值和最小值之间的差距较大。

**中纬度地区**的VTEC值低于低纬度地区，整体呈现逐渐降低的趋势。虽然中纬度地区的VTEC值相较于低纬度有所下降，但仍然具有一定的变化幅度，尤其在最大值和最小值之间存在显著差距。

**高纬度地区**的VTEC值最低，且变化幅度相对较小，WHIT和SCOR测站的VTEC值接近。高纬度地区的VTEC值明显低于低纬度和中纬度地区。

根据输出结果还可以看出：低纬度地区显示出比高纬度地区（纬度 > 55°N）更高的平均VTEC值。具体来说，低纬度地区的平均VTEC值比高纬度地区高约227.2%。

通过计算VTEC与纬度的相关系数，结果为“-0.981738”，显示出强烈的负相关关系。这意味着**随着纬度的增加，VTEC值通常下降**。并且相关性检验的p值为0.000497，远小于0.05，表明此相关性是**显著的**。

1.4 探讨差异原因

**1. 太阳辐射的纬度分布效应**

低纬度地区太阳入射角接近垂直，单位面积接收的太阳辐射能量显著高于高纬度地区。电离层主要由太阳极紫外(EUV)和X射线辐射电离大气产生，垂直入射时的电离效率最高。根据Chapman电离理论，电离产生率与太阳天顶角的余弦值成正比，从PPPC(9.773°N)到SCOR(70.485°N)，太阳平均天顶角逐渐增大，导致电离效率急剧下降，这是造成VTEC随纬度增加而递减的根本原因。

**2. 地磁场配置的控制作用**

地磁场在不同纬度的倾角差异显著影响等离子体的分布和输运。赤道地区地磁场近乎水平，而高纬度地区近乎垂直。由于等离子体扩散主要沿磁力线进行，赤道地区的水平磁场限制了等离子体的垂直扩散，使其更容易在电离层F层积累，形成较高的电子密度。相反，高纬度地区的垂直磁场允许等离子体更容易向下扩散并与中性成分复合，导致稳态电子密度较低。

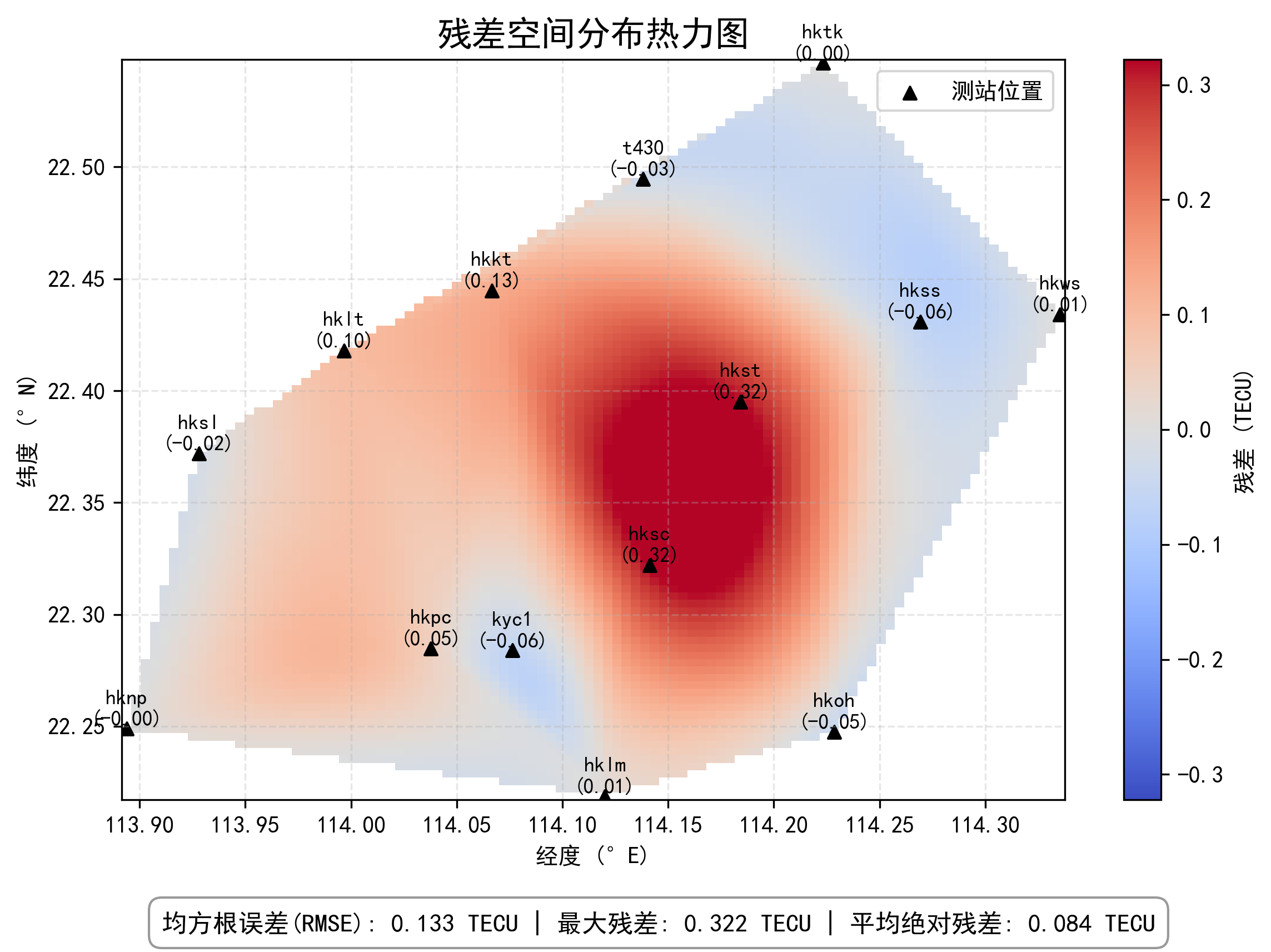
**3. 赤道电离层异常现象**

赤道电离层异常(EIA)是低纬度VTEC显著增强的重要机制。在地磁赤道地区，东向电场与水平磁场相互作用产生E×B垂直向上漂移，将等离子体抬升到高层。随后在重力和压力梯度作用下，等离子体沿磁力线向南北两侧扩散，在地磁纬度±15-20°附近形成两个电子密度峰值。结果数据完美验证了这一现象：PPPC测站(9.773°N)平均VTEC达61.36 TECU，正好位于赤道异常北峰附近，而SHAO测站(31.100°N)仍部分受到异常区影响，VTEC为45.53 TECU。

**4. 中性风系统的纬度差异**

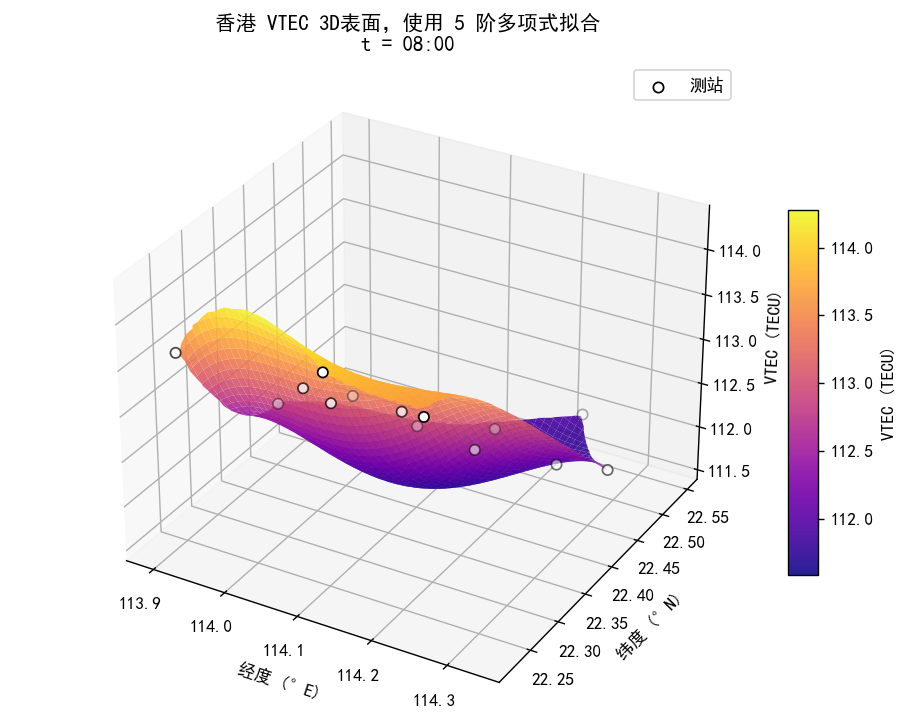
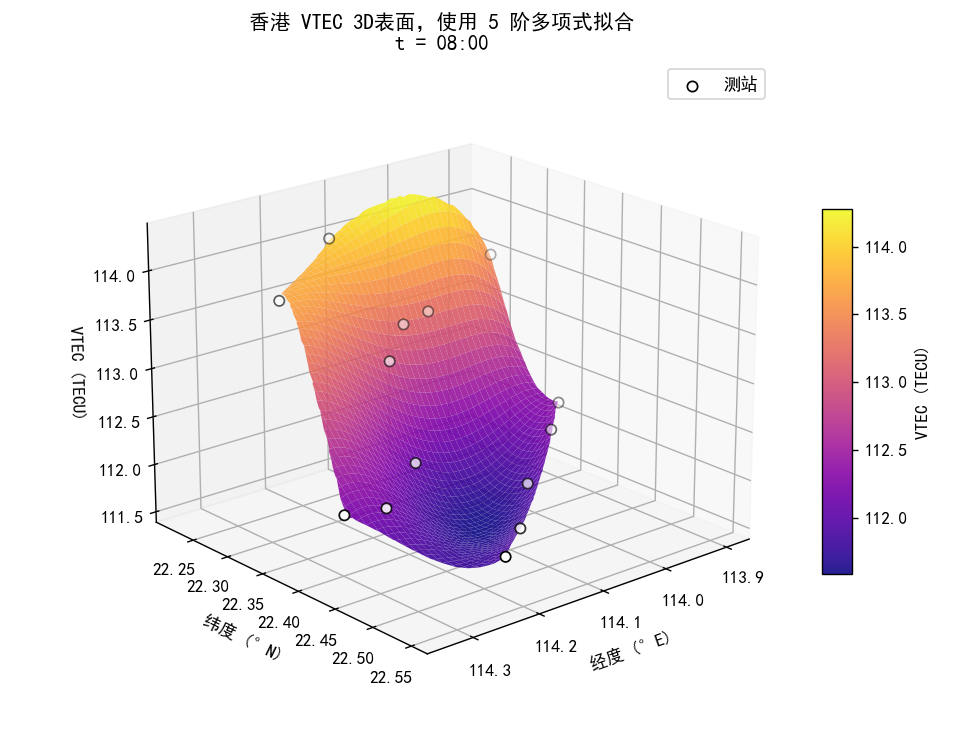
热层中性风对电离层的影响随纬度变化显著。白天从赤道向极地的子午风推动等离子体沿磁力线运动，在低纬度地区由于磁倾角小，上行中性风能有效提升电离层高度，延长等离子体寿命。而在高纬度地区，由于磁倾角大，中性风对电离层垂直分布的影响相对较小。此外，纬向风通过与磁场的相互作用在低纬度产生更强的垂直漂移，进一步增强了低纬度的VTEC值。

2 多项式拟合香港地区电离层模型

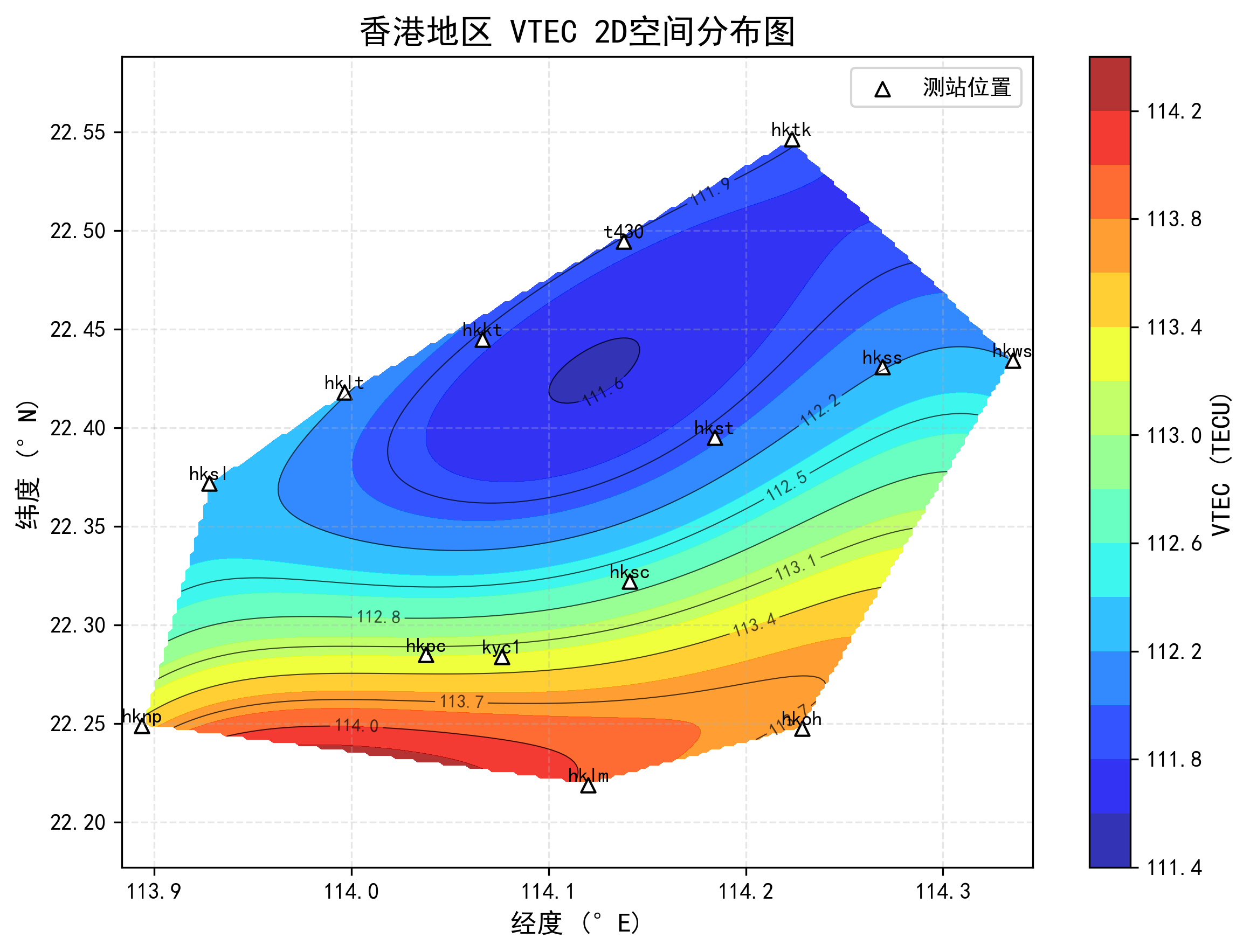


从残差空间分布热力图我们可以看出，残差呈现明显的系统性空间分布，中心区域（约114.15°E，22.35°N）显示较大正残差（0.2-0.3 TECU），周边区域为负残差（-0.1到-0.2 TECU）。统计特征：RMSE（0.133 TECU）相对较小，表明整体建模精度良好最大残差（0.322 TECU）出现在中心区域。平均绝对残差（0.084 TECU）较低，说明模型总体性能可接受。

电离层模型的3D可视化(使用**5阶**多项式拟合)：

电离层模型的2D可视化：



控制台打印的模型信息：



这里R²=0.962 意味着5阶多项式模型解释了96.2%的电离层VTEC变化。效果还是非常好的。

1. **总结**

本次任务围绕GNSS电离层监测展开，从指定网站下载2025年4月22日不同纬度测站的观测值文件及精密星历文件，利用NEU2DTEC生成VTEC数据，在筛选有效数据后，针对不同纬度测站电离层差异分析和香港区域电离层建模两项任务展开研究。

在不同纬度测站电离层差异分析中，选取6个有效测站（纬度范围9.733°N至70.485°N），通过统计均值、最大值、最小值、变化幅度等特征并计算相关系数，发现低纬度地区VTEC值最高且波动较大，中纬度地区VTEC值逐渐降低，高纬度地区VTEC值最低且变化幅度较小，低纬度地区平均VTEC值比高纬度地区高约227.2%，纬度与VTEC的相关系数为-0.9817，p值为0.000497，相关性显著，这主要是太阳辐射纬度分布、地磁场配置、赤道电离层异常及中性风系统差异等因素共同作用的结果。

在利用多项式拟合香港区域的电离层模型的任务中，使用14个有效测站数据，通过经纬度归一化、5阶多项式特征构建及ElasticNet回归拟合模型，结果显示模型决定系数R²为0.962，均方根误差为0.133 TECU，平均绝对误差为0.084 TECU，残差热力图显示中心区域有正残差、周边为负残差，3D和2D可视化清晰呈现了香港地区VTEC的空间分布，模型精度较高且能有效捕捉区域电离层特征。

本次实验通过数据处理、统计分析和模型构建，验证了电离层纬度差异的物理机制，构建了适用于香港地区的电离层模型。在这个过程中，通过一个个问题的解决，也锻炼了我发现问题、处理问题的能力。