

```

In [1]: import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import folium

file_path = "EV Data-2012-Dec-03_08-40-11.csv"
df = pd.read_excel(file_path)

def plot_trajectory_with_folium(latitudes, longitudes, title="车辆轨迹图"):
    """
    使用Folium绘制交互式车辆轨迹

    参数:
    latitudes: 纬度列表
    longitudes: 经度列表
    title: 地图标题
    """

    # 计算中心点
    center_lat = sum(latitudes) / len(latitudes)
    center_lon = sum(longitudes) / len(longitudes)

    # 创建地图
    m = folium.Map(location=[center_lat, center_lon],
                    zoom_start=13,
                    tiles='OpenStreetMap')

    # 添加轨迹线
    points = list(zip(latitudes, longitudes))
    folium.PolyLine(points,
                    color='red',
                    weight=5,
                    opacity=0.8,
                    popup=title).add_to(m)

    # 添加起点和终点标记
    folium.Marker([latitudes[0], longitudes[0]],
                  popup='起点',
                  icon=folium.Icon(color='green', icon='play')).add_to(m)

    folium.Marker([latitudes[-1], longitudes[-1]],
                  popup='终点',
                  icon=folium.Icon(color='blue', icon='stop')).add_to(m)

    # 添加轨迹点标记
    for i, (lat, lon) in enumerate(points):
        if i % 5 == 0: # 每5个点标记一次
            folium.CircleMarker([lat, lon],
                                radius=3,
                                popup=f'点 {i}',
                                color='black',
                                fill=True).add_to(m)

    # 保存为HTML文件
    m.save('vehicle_trajectory.html')
    print("交互式地图已保存为 'vehicle_trajectory.html'")

    return m

```

ChatGPT

提取经纬度数据

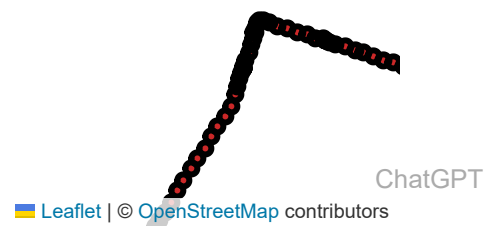
```
latitudes = df["Latitude"].tolist()
longitudes = df["Longitude"].tolist()

print(f"读取了 {len(latitudes)} 个数据点")

# 绘制交互式地图
plot_trajectory_with_folium(latitudes, longitudes, "从CSV文件读取的车辆轨迹")
```

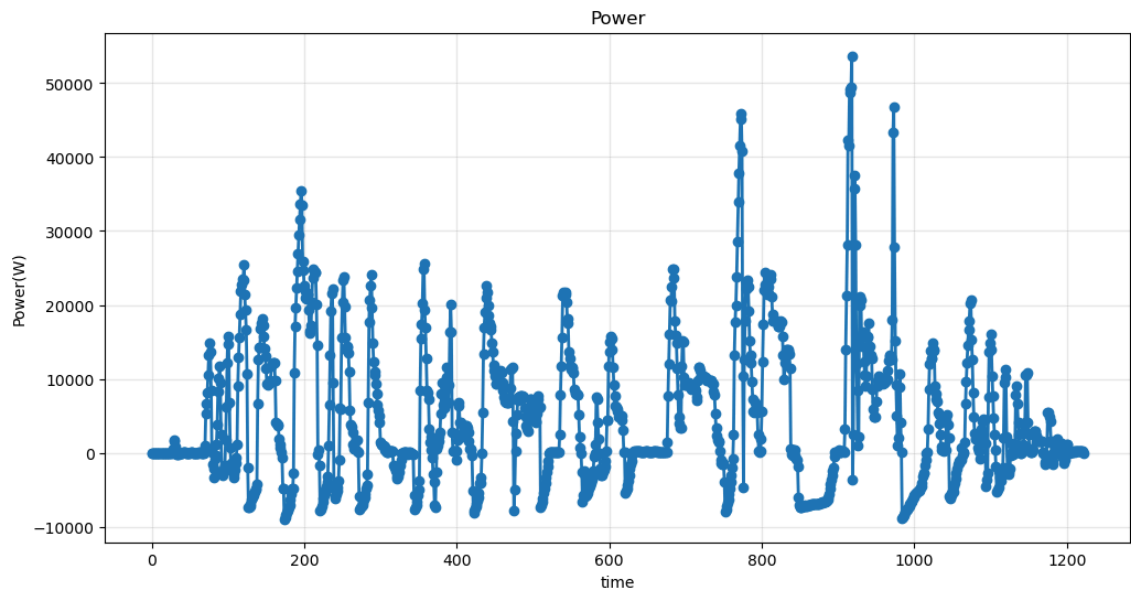
读取了 1223 个数据点
交互式地图已保存为 'vehicle_trajectory.html'

Out[1]:



```
In [44]: power = df["Pack_Power(W)"].tolist()
# 创建时间索引（假设是等间隔时间）
time = range(len(power))

plt.figure(figsize=(12, 6))
plt.plot(time, power, marker='o', linewidth=2, markersize=6)
plt.xlabel('time')
plt.ylabel('Power(W)')
plt.title('Power')
plt.grid(True, alpha=0.3)
plt.show()
```

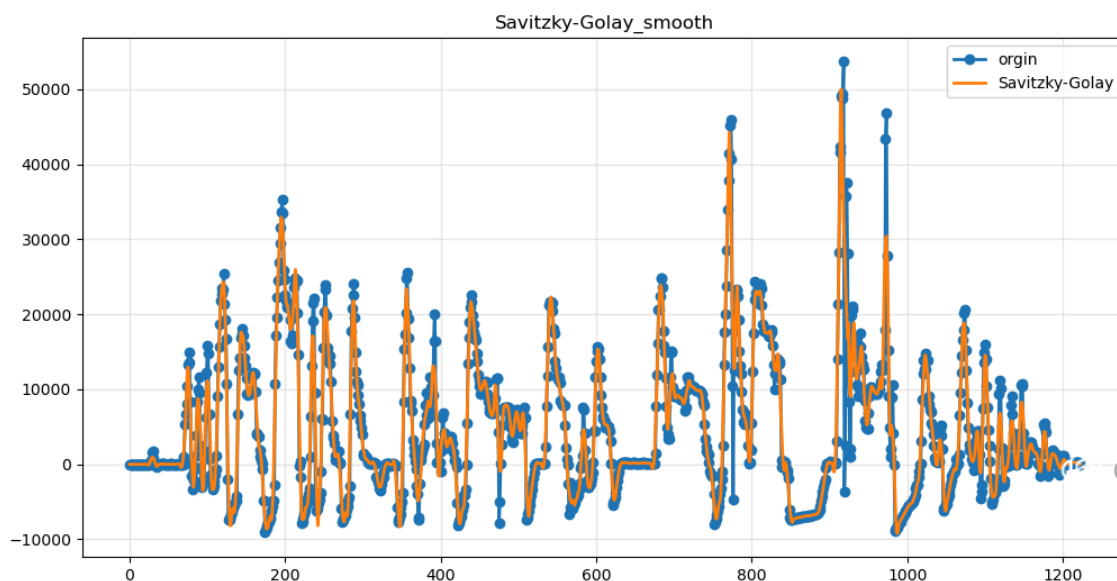


```
In [52]: from scipy.signal import savgol_filter

# Savitzky-Golay 滤波器（保留特征的同时平滑）
window_length = 11 # 窗口长度（必须为奇数）
polyorder = 3      # 多项式阶数

power_sg = savgol_filter(power, window_length, polyorder)

plt.figure(figsize=(12, 6))
plt.plot(time, power, marker='o', label='origin', linewidth=2, markersize=6)
plt.plot(time, power_sg, label='Savitzky-Golay', linewidth=2)
plt.legend()
plt.grid(True, alpha=0.3)
plt.title('Savitzky-Golay_smooth')
plt.show()
```



```
In [63]: from scipy import integrate
# 辛普森法数值积分
def simpson_integration(time, power_values):
    """使用辛普森法计算积分"""
    return integrate.simpson(power_values, time)

simpson_integral = simpson_integration(time, power_sg)
print(f"平滑后总积分（辛普森法）: {simpson_integral:.2f} 焦耳")
```

平滑后总积分（辛普森法）: 5963390.24 焦耳