Update 25.1.22

拆分 MonthYear.acc2.nc

批处理

提取 BC 负荷，合并成一个 nc

重构：合并飞行数据集

替换每一条记录

垂直剖面重构

boxplot 重构

拆分 MonthYear.acc2.nc

复制 scaleacc.sh 脚本，运行

```
chmod +x scaleacc #授权
./scaleacc JUN2011.acc2.nc tijl #aij aijl taij taijl
# scaleacc acc-file acc_array_name[,name2] [remap_file]
```

得到 JUN2011.taij2.nc

批处理

批量复制 copy_file.sh

```
target_dir="../2_results" #目标文件夹路径
mkdir -p "$target_dir" # 创建目标文件夹（如果不存在）

# 遍历所有符合格式的文件
for file in [A-Z][A-Z][A-Z]20[0-9][0-9].acc2.nc; do
    # 检查文件是否存在（避免无匹配文件时报错）
    if [[ -f "$file" ]]; then
        echo "Copying file: $file to $target_dir"
        # 复制文件到目标文件夹
        cp "$file" "$target_dir"
    else
        echo "No files found matching the pattern."
        break
    fi
done
```

批处理

批量执行 scaleacc：scaleacc_files.sh

```
months=("JAN" "FEB" "MAR" "APR" "MAY" "JUN" "JUL" "AUG" "SEP" "OCT" "NOV" "DEC")
start_year=2009
end_year=2018
file_list=() # 预生成所有可能的文件名
for month in "${months[@]}"; do
    for year in $(seq "$start_year" "$end_year"); do
        file_list+=("$month$year.acc2.nc")
    done
done
for file in "${file_list[@]}"; do # 遍历预生成的文件名列表
    if [[ -f "$file" ]]; then # 检查文件是否存在
        echo "Processing file: $file"
        ./scaleacc "$file" taijl # 运行 ./scaleacc 命令
        ./scaleacc "$file" taij
        ./scaleacc "$file" aij
        ./scaleacc "$file" aijl
    fi
done
echo "Processing completed!"
```

提取 BC 负荷

在 taij 中有 5 个不同的混合状态下的 BC 质量浓度

```
dataset_name = [
    "M_BC1_BC",
    "M_BC2_BC",
    "M_BCS_BC",
    "M_BOC_BC",
    "M_MXX_BC",
]
```

合并成一个 nc

```
for year in range(start_year, end_year + 1):
    for month_num in range(0, 12, 1):
        file_names.append(month[month_num] + str(year) + ".taijl2.nc")
        filename = file_names[-1]
        path = r"..\数据集\giss\\" + filename
        if not os.path.exists(path):
            continue
        else:
            print(path)
            dataset = nc.Dataset(path, "r")
            BC_conponent = np.zeros((5, 40, 90, 144))
            for i in range(5):
                BC_conponent[i, :, :, :] = dataset.variables[dataset_name[i]][:].data
            BC_sum = np.sum(BC_conponent, axis=0)
            final_result[year - start_year, month_num, :, :, :] = BC_sum
        path = r"..\数据集\giss\\" + month[month_num] + str(year) + ".aijl2.nc"
        if not os.path.exists(path):
            continue
        else:
            print(path)
            dataset = nc.Dataset(path, "r")
            Temprature = dataset.variables["TempL"][:].data
            Temprature_result[year - start_year, month_num, :, :, :] = Temprature
```

合并成一个 nc

```python
output_file_name = "giss_monthly_data.nc"
with nc.Dataset(output_file_name, "w", format="NETCDF4") as ds:
    ds.createDimension("year", end_year - start_year + 1)
    ds.createDimension("month", 12)
    ds.createDimension("level", 40)
    ds.createDimension("lat", 90)
    ds.createDimension("lon", 144)

    year = ds.createVariable("year", np.int32, ("year",))
    month = ds.createVariable("month", np.int32, ("month",))
    level = ds.createVariable("level", np.int32, ("level",))
    lat = ds.createVariable("lat", np.float32, ("lat",))
    lon = ds.createVariable("lon", np.float32, ("lon",))
    bc_mass = ds.createVariable(
        "bc_mass", np.float32, ("year", "month", "level", "lat", "lon")
    )
    pressure = ds.createVariable("pressure", np.float32, ("level",))
    temprature = ds.createVariable(
        "temprature", np.float32, ("year", "month", "level", "lat", "lon")
    )

    year[:] = np.linspace(start_year, end_year, end_year - start_year + 1)
    month[:] = np.linspace(1, 12, 12)
    level[:] = np.linspace(1, 40, 40)
    lat[:] = model_lat
    lon[:] = model_lon
    pressure[:] = model_pressure

    bc_mass[:, :, :, :, :] = final_result
    temprature[:, :, :, :, :] = Temprature_result

print(f"NetCDF file '{output_file_name}' created successfully.")
```

image-20250122150736365

重构：合并飞行数据集

```python
for campaign_name in campaign_names:
    path = "..\数据集\dataset\\" + campaign_name + "_data.csv"
    df = pd.read_csv(path)
    df.insert(0, "Campaign", campaign_name)
    df["location"] = df["location"].apply(lambda x: "sea" if x == 1 else "land")
    final_result = final_result.append(df, ignore_index=True)
```

image-20250121185758382

替换每一条记录

通过对于飞行观测记录的每一条记录，找到其在模拟数据集中对应的年月、高度层和经纬度进行替换

```python
for index, row in tqdm(flight_data.iterrows()):
    year_index = np.argmin(np.abs(years - row["date"].year))
    month_index = np.argmin(np.abs(months - row["date"].month))
    level_index = np.argmin(np.abs(pressures - get_pressure_mbar(row["alt"])))
    lat_index = np.argmin(np.abs(lat - row["lat"]))
    lon_index = np.argmin(np.abs(lon - row["lon"]))
    replace_bc_mass = get_mass_ng_per_m3(
        model_bc_mass[year_index, month_index, level_index, lat_index, lon_index],
        row["alt"],
        temprature[year_index, month_index, level_index, lat_index, lon_index],
    )
    flight_data.loc[index, "mass"] = replace_bc_mass
```

其中对于气压的计算使用公式

$$P = P_0 \cdot \left(1 - \frac{L \cdot h}{T_0}\right)^{\frac{g \cdot M}{R \cdot L}}$$

各个参数如下

```
def get_pressure_mbar(alt):
    P0 = 101325  # 海平面标准大气压 (Pa)
    T0 = 288.15  # 海平面标准温度 (K)
    L = 0.0065  # 温度递减率 (K/m)
    g = 9.80665  # 重力加速度 (m/s²)
    M = 0.028964  # 干空气的摩尔质量 (kg/mol)
    R = 8.314  # 通用气体常数 (J/(mol·K))
    return P0 * (1 - (L * alt) / T0) ** (g * M / (R * L)) / 100
```

对于密度单位换算公式

$$C_{\mathrm{BC}}\,(\mathrm{kg/m^3}\ ) = C_{\mathrm{BC}}\,(\mathrm{kg/kg\ air}) \times \frac{P}{R \cdot T}$$

各个参数如下，乘号后为空气密度，×100 为从 10^-10^kg/kg air 转化为 ng/m^3^

```python
def get_mass_ng_per_m3(mass_kg_per_kg_air, alt, temperature_K):
    R = 287  # 干空气的比气体常数
    P = get_pressure_mbar(alt) * 100  # mbar -> Pa
    result = mass_kg_per_kg_air * P / (R * temperature_K) * 100
    return result
```

由于没有 11 年以前的模拟数据，这些目前是 nan

重构：垂直剖面

重构了一下以前的狗屎代码，以下是模式数据的垂直剖面。

```python
def draw_profile(df, i):
    dataset_name = df["Campaign"].unique()[0]
    df["alt_group"] = df["alt"] // 500 * 500
    grouped_stats = (
        df.groupby("alt_group")["mass"].agg(["mean", "median", "std"]).reset_index()
    )
    plt.title(dataset_name, fontsize=14)
    plt.xlabel("BC_concentration  ($ng/m^3$)", fontsize=14)
    plt.ylabel("Altitude(m)", fontsize=14)
    plt.plot(
        grouped_stats["mean"],
        grouped_stats["alt_group"],
        linestyle="-",
        linewidth=3,
        color=colors[i],
    )
    plt.plot(
        grouped_stats["median"],
        grouped_stats["alt_group"],
        linestyle="--",
        color=colors[i],
    )
    plt.ylim(0, 13000)
    plt.errorbar(
        grouped_stats["mean"],
        grouped_stats["alt_group"],
        xerr=grouped_stats["std"],
        ecolor=colors[i],
        linestyle="None",
        capsize=0,
        linewidth=0.5,
    )
    plt.xlim(0, None)
    return 0


if __name__ == "__main__":
    path = "../数据集/model_replaced_flight_data.csv"
    df = pd.read_csv(path)
    dataset_names = pd.unique(df["Campaign"]).tolist()
    dataset_names.remove("GOAMAZON")
    plt.figure(figsize=(15, 15))
    for i in range(len(dataset_names)):
        plt.subplot(3, 4, i + 1)
        selected_df = df[df["Campaign"] == dataset_names[i]]
        draw_profile(selected_df, i)
    plt.subplots_adjust(hspace=0.3, wspace=0.5)
    plt.show()
```

重构：boxplot

重构代码，以下是模拟数据的

boxplot

```python
if __name__ == "__main__":
    path = "../数据集/model_replaced_flight_data.csv"
    df = pd.read_csv(path)
    dataset_names = pd.unique(df["Campaign"]).tolist()
    grouped_df = df.groupby(["Campaign"])
    box_data = []
    for campaign, group in grouped_df:
        box_data.append(group["mass"].tolist())
    fig, ax = plt.subplots(figsize=(16, 9), dpi=200)
    ax.set_title("Model BC concentration Box Plot", fontsize=20)
    ax.set_xlabel("BC concentration ($ng/m^3$)", fontsize=14)
    bplot = ax.boxplot(
        box_data,
        vert=False,  # 将vert参数设置为False以实现横纵颠倒
        patch_artist=True,
        labels=dataset_names,
        whiskerprops=dict(linewidth=0.5),
        boxprops=dict(linewidth=0.5),
        showmeans=True,
        meanprops=dict(marker="o", markeredgecolor="black", markerfacecolor="black"),
        showcaps=True,
        showfliers=False,
        medianprops=dict(linewidth=0.5, color="black"),
        whis=[5, 95],
    )
    for patch, color in zip(bplot["boxes"], colors):
        patch.set_facecolor(color)
    ax.set_yticklabels(dataset_names, fontsize=14)
    ax.tick_params(axis="both", which="major", labelsize=14)
    ax.grid(axis="x", alpha=0.5)
    plt.show()
```

待办

- 2011 年以前的模拟数据
- 操作观测数据的代码重构