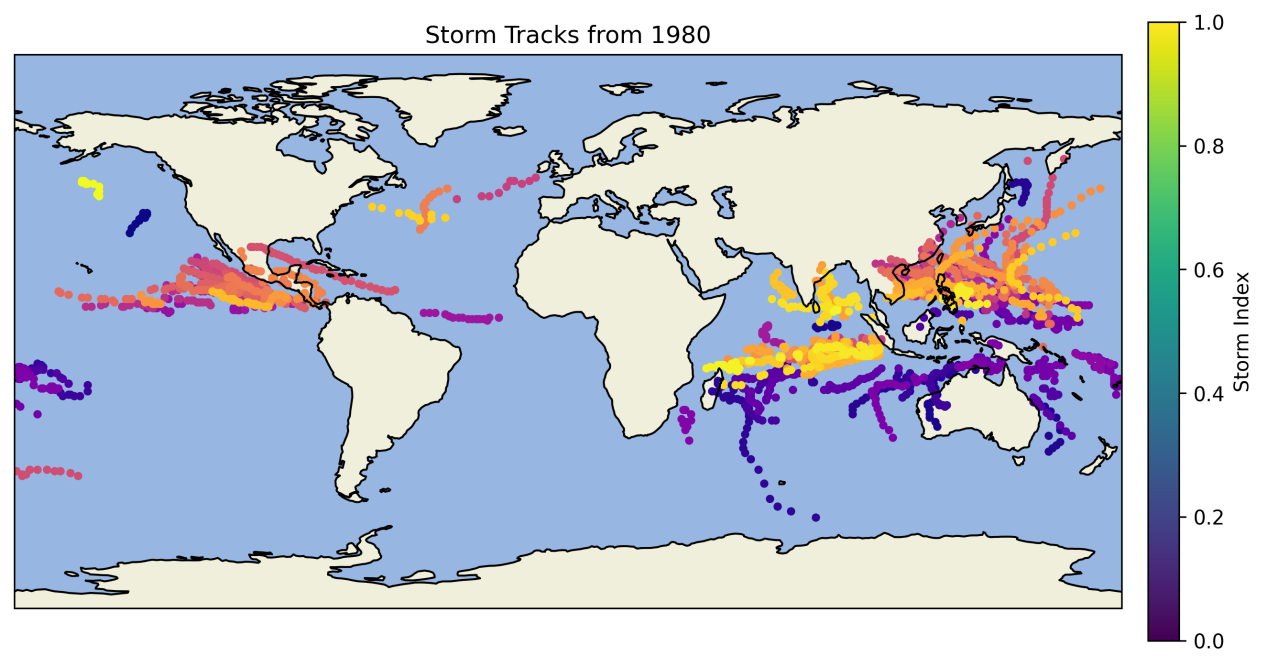
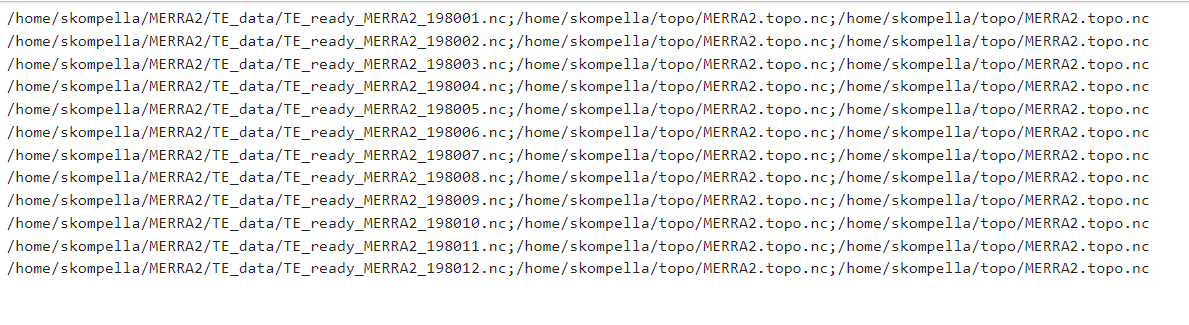
1. MERRA2 Data

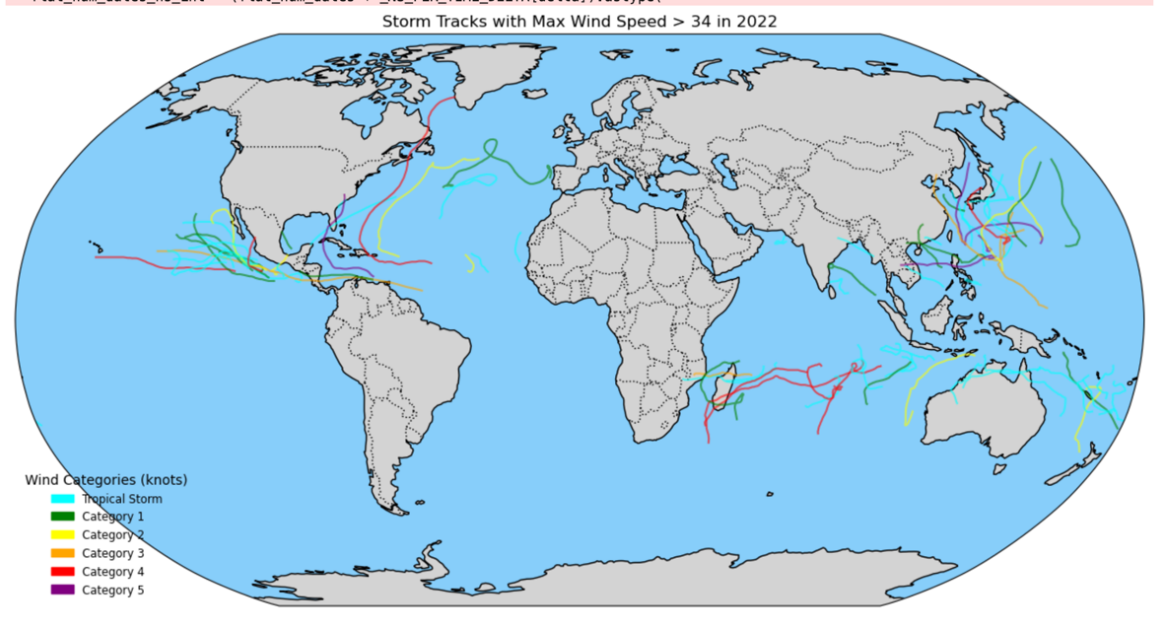




We conducted a global storm track reanalysis for the 12 months of 1980 using MERRA2 data through TempestExtremes (TE). The resulting data (.dat files) can be viewed on GitHub, along with the corresponding data labeling and visualizations.

1. IBTrACS

We plot the track of storm in 2022 with max wind speed larger or equal to 34k.



ds = xr.open\_dataset('/home/zy2608/zy2608/9.22/IBTrACS.last3years.v04r01.nc')

ds\_2022 = ds.where(ds['season'] == 2022, drop=True)

max\_wind\_per\_storm = ds\_2022['usa\_wind'].max(dim='date\_time')

storms\_with\_high\_wind = max\_wind\_per\_storm.where(max\_wind\_per\_storm > 34, drop=True)

high\_wind\_storms = ds\_2022.sel(storm=storms\_with\_high\_wind['storm'])

latitudes = high\_wind\_storms['usa\_lat']

longitudes = high\_wind\_storms['usa\_lon']

winds = high\_wind\_storms['usa\_wind']

longitudes = ((longitudes + 180) % 360) - 180

categories = {

'Tropical Storm': {'min': 34, 'max': 63, 'color': 'cyan'},

'Category 1': {'min': 64, 'max': 82, 'color': 'green'},

'Category 2': {'min': 83, 'max': 95, 'color': 'yellow'},

'Category 3': {'min': 96, 'max': 112, 'color': 'orange'},

'Category 4': {'min': 113, 'max': 136, 'color': 'red'},

'Category 5': {'min': 137, 'color': 'purple'},

}

plt.figure(figsize=(15, 10))

ax = plt.axes(projection=ccrs.Robinson())

ax.set\_global()

ax.coastlines()

ax.add\_feature(cfeature.BORDERS, linestyle=':')

ax.add\_feature(cfeature.LAND, facecolor='lightgray')

ax.add\_feature(cfeature.OCEAN, facecolor='lightskyblue')

plt.title("Storm Tracks with Max Wind Speed > 34 in 2022")

for storm\_idx in range(latitudes.sizes['storm']):

storm\_lat = latitudes.isel(storm=storm\_idx).dropna('date\_time')

storm\_lon = longitudes.isel(storm=storm\_idx).dropna('date\_time')

max\_wind = max\_wind\_per\_storm.isel(storm=storm\_idx)

for category, props in categories.items():

if ('min' in props and 'max' in props and props['min'] <= max\_wind <= props['max']) or \

('min' in props and max\_wind >= props['min'] and 'max' not in props):

storm\_category = category

color = props['color']

break

if len(storm\_lon) > 1:

split\_indices = np.where(np.abs(np.diff(storm\_lon)) > 180)[0] + 1

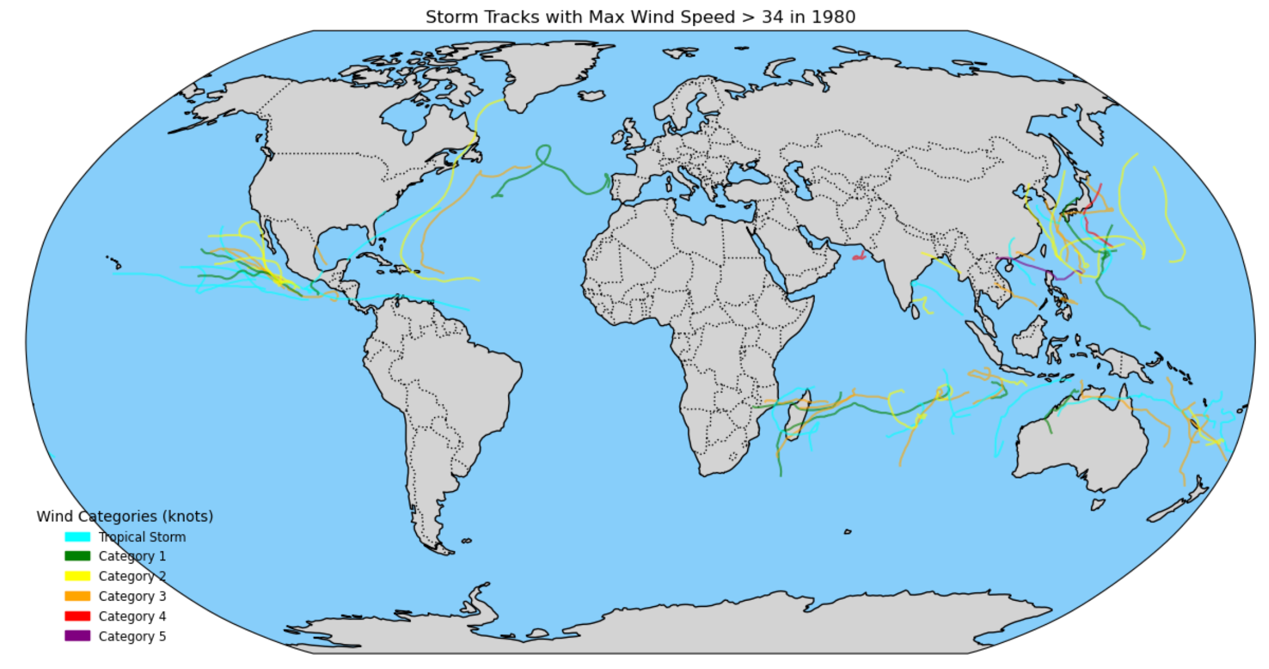
split\_lon = np.split(storm\_lon, split\_indices)

split\_lat = np.split(storm\_lat, split\_indices)

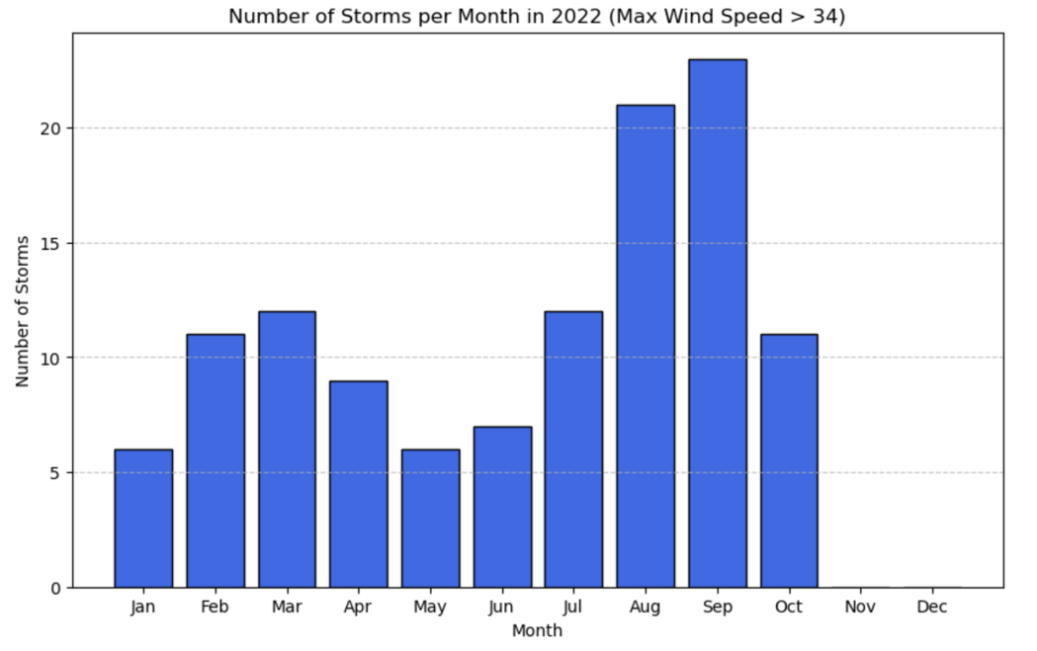
for lon\_segment, lat\_segment in zip(split\_lon, split\_lat):

plt.plot(lon\_segment, lat\_segment, color=color, transform=ccrs.PlateCarree(), alpha=0.6)

We also plot the track of storm in 1980 with max wind speed larger or equal to 34k.



We made the bar chart of number of storms per month in 2022 (Max Wind Speed > 34)



ds = xr.open\_dataset('/home/zy2608/zy2608/9.22/IBTrACS.last3years.v04r01.nc')

iso\_time = pd.to\_datetime(ds.iso\_time.astype(str).values.flatten()).to\_numpy().reshape(ds.iso\_time.shape)

ds['iso\_time'] = xr.DataArray(iso\_time, dims=ds.iso\_time.dims, coords=ds.iso\_time.coords)

ds\_2022 = ds.where(ds['iso\_time'].dt.year == 2022, drop=True)

max\_wind\_per\_storm = ds\_2022['usa\_wind'].max(dim='date\_time')

storms\_with\_high\_wind = max\_wind\_per\_storm.where(max\_wind\_per\_storm > 34, drop=True)

high\_wind\_storms = ds\_2022.sel(storm=storms\_with\_high\_wind['storm'])

months = high\_wind\_storms['iso\_time'].dt.month

unique\_storm\_months\_list = []

for storm in high\_wind\_storms['storm']:

storm\_months = months.sel(storm=storm).dropna('date\_time').values

unique\_months = np.unique(storm\_months)

unique\_storm\_months\_list.extend(unique\_months)

all\_unique\_months = np.array(unique\_storm\_months\_list)

monthly\_counts = pd.Series(all\_unique\_months).value\_counts().sort\_index()

all\_months\_index = pd.Index(np.arange(1, 13), name="month")

monthly\_counts\_full = monthly\_counts.reindex(all\_months\_index, fill\_value=0)

plt.figure(figsize=(10, 6))

plt.bar(monthly\_counts\_full.index, monthly\_counts\_full.values, color='royalblue', edgecolor='black')

plt.xlabel('Month')

plt.ylabel('Number of Storms')

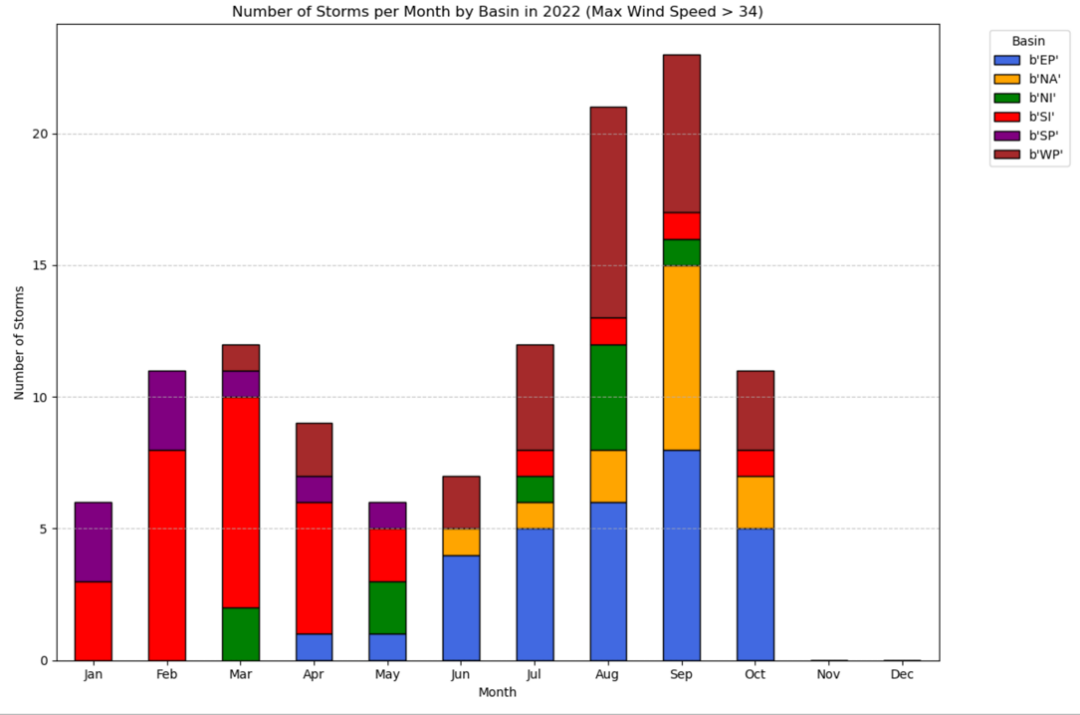
plt.title('Number of Storms per Month in 2022 (Max Wind Speed > 34)')

plt.xticks(ticks=np.arange(1, 13), labels=['Jan', 'Feb', 'Mar', 'Apr', 'May', 'Jun', 'Jul', 'Aug', 'Sep', 'Oct', 'Nov', 'Dec'])

plt.grid(axis='y', linestyle='--', alpha=0.7)

plt.show()

We also made the bar chart of number of storms per month by basin in 2022 (Max Wind Speed > 34)



ds = xr.open\_dataset('/home/zy2608/zy2608/9.22/IBTrACS.last3years.v04r01.nc')

iso\_time = pd.to\_datetime(ds.iso\_time.astype(str).values.flatten()).to\_numpy().reshape(ds.iso\_time.shape)

ds['iso\_time'] = xr.DataArray(iso\_time, dims=ds.iso\_time.dims, coords=ds.iso\_time.coords)

ds\_2022 = ds.where(ds['iso\_time'].dt.year == 2022, drop=True)

max\_wind\_per\_storm = ds\_2022['usa\_wind'].max(dim='date\_time')

storms\_with\_high\_wind = max\_wind\_per\_storm.where(max\_wind\_per\_storm > 34, drop=True)

high\_wind\_storms = ds\_2022.sel(storm=storms\_with\_high\_wind['storm'])

months = high\_wind\_storms['iso\_time'].dt.month

basins = high\_wind\_storms['basin']

unique\_storm\_months\_list = []

unique\_storm\_basins\_list = []

for storm in high\_wind\_storms['storm']:

storm\_months = months.sel(storm=storm).dropna('date\_time').values

storm\_basins = basins.sel(storm=storm).dropna('date\_time').values

unique\_months = np.unique(storm\_months)

unique\_basins = np.unique(storm\_basins)

for month in unique\_months:

unique\_storm\_months\_list.append(month)

unique\_storm\_basins\_list.append(unique\_basins[0])

storm\_data = pd.DataFrame({

'month': unique\_storm\_months\_list,

'basin': unique\_storm\_basins\_list

})

monthly\_basin\_counts = storm\_data.groupby(['month', 'basin']).size().unstack(fill\_value=0)

all\_months\_index = pd.Index(np.arange(1, 13), name="month")

monthly\_basin\_counts = monthly\_basin\_counts.reindex(all\_months\_index, fill\_value=0)

monthly\_basin\_counts.plot(

kind='bar',

stacked=True,

figsize=(12, 8),

color=['royalblue', 'orange', 'green', 'red', 'purple', 'brown', 'pink'],

edgecolor='black'

)

plt.xlabel('Month')

plt.ylabel('Number of Storms')

plt.title('Number of Storms per Month by Basin in 2022 (Max Wind Speed > 34)')

plt.xticks(ticks=np.arange(0, 12), labels=['Jan', 'Feb', 'Mar', 'Apr', 'May', 'Jun', 'Jul', 'Aug', 'Sep', 'Oct', 'Nov', 'Dec'], rotation=0)

plt.grid(axis='y', linestyle='--', alpha=0.7)

plt.legend(title='Basin', bbox\_to\_anchor=(1.05, 1), loc='upper left')

plt.tight\_layout()

plt.show()

1. ERA5 Data

We noticed that TempestExtremes requires both surface-level data and pressure-level data to accurately identify and track atmospheric features like cyclones. Pressure-level data provides information about the atmospheric state at various altitudes, which is crucial for detecting features that are not apparent at the surface.

**Extract Data from CDS**

①ERA5 hourly data on single levels from 1940 to present

i. To get ERA5 data in surface level:

import cdsapi

dataset = "reanalysis-era5-single-levels"

request = {

"product\_type": ["reanalysis"],

"variable": [

"10m\_u\_component\_of\_wind",

"10m\_v\_component\_of\_wind",

"2m\_temperature",

"mean\_sea\_level\_pressure",

"geopotential"

],

"year": ["2022"],

"month": [

"01", "02", "03",

"04", "05", "06",

"07", "08", "09",

"10", "11", "12"

],

"day": [

"01", "02", "03",

"04", "05", "06",

"07", "08", "09",

"10", "11", "12",

"13", "14", "15",

"16", "17", "18",

"19", "20", "21",

"22", "23", "24",

"25", "26", "27",

"28", "29", "30",

"31"

],

"time": [

"00:00", "06:00", "12:00",

"18:00"

],

"data\_format": "netcdf",

"download\_format": "unarchived"

}

client = cdsapi.Client()

client.retrieve(dataset, request).download() ERA\_data.nc

ii. To get geopotential data(topo) in surface level:

import cdsapi

dataset = "reanalysis-era5-single-levels"

request = {

"product\_type": ["reanalysis"],

"variable": ["geopotential"],

"year": ["2022"],

"month": [

"01", "02", "03",

"04", "05", "06",

"07", "08", "09",

"10", "11", "12"

],

"day": [

"01", "02", "03",

"04", "05", "06",

"07", "08", "09",

"10", "11", "12",

"13", "14", "15",

"16", "17", "18",

"19", "20", "21",

"22", "23", "24",

"25", "26", "27",

"28", "29", "30",

"31"

],

"time": [

"00:00", "06:00", "12:00",

"18:00"

],

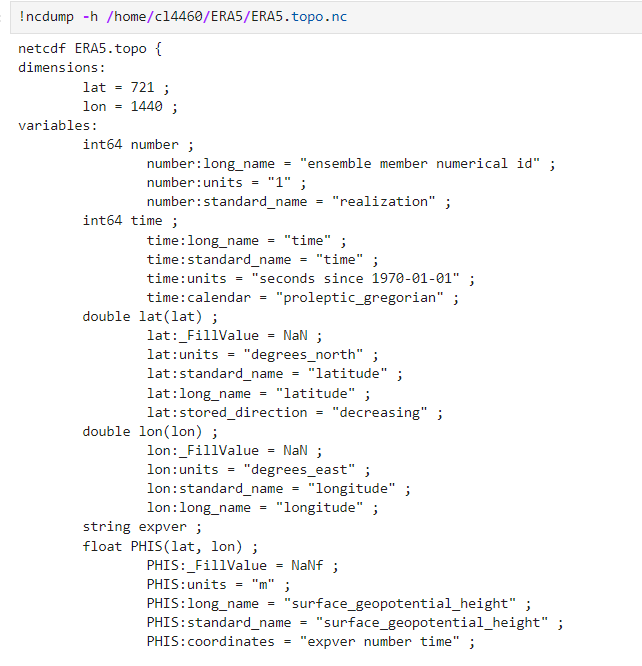
"data\_format": "netcdf",

"download\_format": "unarchived"

}

client = cdsapi.Client()

client.retrieve(dataset, request).download() ERA5.topo.nc (4M)



Potential problem:

In our ERA5\_TE\_ready\_2022.nc, the time variable has:

time:units = "seconds since 1970-01-01"

In contrast, the MERRA2 data that worked has:

time:units = "hours since 1980-01-01" ;

import xarray as xr

import numpy as np

ds\_topo = xr.open\_dataset('/home/cl4460/ERA5/ERA5.topo.nc')

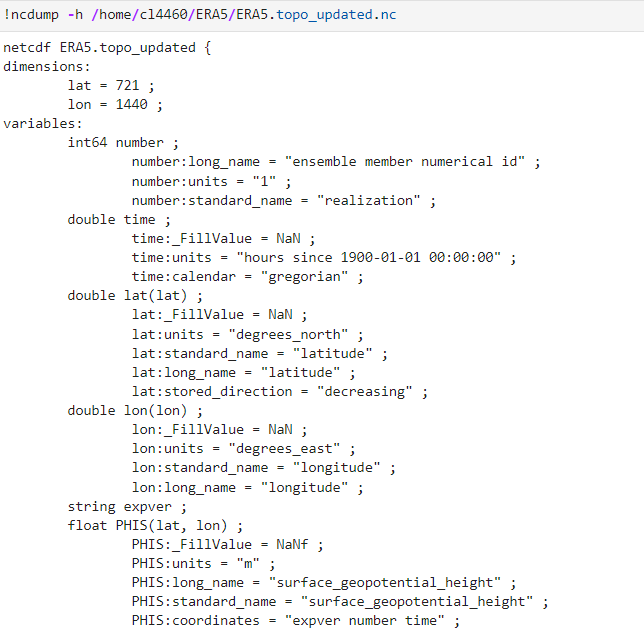
ds\_topo['time'] = ds\_topo['time'].astype('datetime64[ns]')

ds\_topo['time'] = ((ds\_topo['time'] - np.datetime64('1900-01-01T00:00:00')) / np.timedelta64(1, 'h'))

ds\_topo['time'].attrs['units'] = 'hours since 1900-01-01 00:00:00'

ds\_topo['time'].attrs['calendar'] = 'gregorian'

ds\_topo.to\_netcdf('/home/cl4460/ERA5/ERA5.topo\_updated.nc', mode='w')



②ERA5 hourly data on pressure levels from 1940 to present

1. To get ERA5 data in pressure level:

import cdsapi

dataset = "reanalysis-era5-pressure-levels"

request = {

"product\_type": ["reanalysis"],

"variable": [

"geopotential",

"temperature",

"u\_component\_of\_wind",

"v\_component\_of\_wind"

],

"year": ["2022"],

"month": [

"01", "02", "03",

"04", "05", "06",

"07", "08", "09",

"10", "11", "12"

],

"day": [

"01", "02", "03",

"04", "05", "06",

"07", "08", "09",

"10", "11", "12",

"13", "14", "15",

"16", "17", "18",

"19", "20", "21",

"22", "23", "24",

"25", "26", "27",

"28", "29", "30",

"31"

],

"time": [

"00:00", "06:00", "12:00",

"18:00"

],

"pressure\_level": ["300", "400", "500"],

"data\_format": "netcdf",

"download\_format": "unarchived"

}

client = cdsapi.Client()

client.retrieve(dataset, request).download() ERA5\_pressure\_level\_data.nc (28.2G)

**Merge data:**

import xarray as xr

ds\_surface = xr.open\_dataset('ERA\_data.nc')

ds\_surface = ds\_surface.rename({

'latitude': 'lat',

'longitude': 'lon',

'valid\_time': 'time',

'msl': 'SLP',

'u10': 'U10M',

'v10': 'V10M',

'z': 'z\_surface'

})

z\_surface = ds\_surface['z\_surface'].isel(time=0)

PHIS = z\_surface / 9.80665

PHIS = PHIS.rename('PHIS')

PHIS.attrs['units'] = 'm'

PHIS.attrs['long\_name'] = 'surface\_geopotential\_height'

PHIS.attrs['standard\_name'] = 'surface\_geopotential\_height'

ds\_topo = PHIS.to\_dataset()

ds\_topo.to\_netcdf('ERA5.topo.nc')

ds\_pl = xr.open\_dataset('ERA5\_pressure\_level\_data.nc')

ds\_pl = ds\_pl.rename({

'latitude': 'lat',

'longitude': 'lon',

'valid\_time': 'time',

'pressure\_level': 'lev',

't': 'T',

'z': 'H', # Geopotential, will convert to height

'u': 'U',

'v': 'V'

})

ds\_pl['H'] = ds\_pl['H'] / 9.80665

ds\_pl['H'].attrs['units'] = 'm'

ds\_pl['H'].attrs['long\_name'] = 'geopotential\_height'

ds\_pl['H'].attrs['standard\_name'] = 'geopotential\_height'

ds\_surface\_vars = ds\_surface[['SLP', 'U10M', 'V10M']]

ds\_pl\_vars = ds\_pl[['T', 'H', 'U', 'V']]

ds\_combined = xr.merge([ds\_surface\_vars, ds\_pl\_vars])

ds\_combined['time'] = ds\_surface['time']

ds\_combined.to\_netcdf('ERA5\_TE\_ready\_2022.nc') ERA5\_TE\_ready\_2022.nc (47.7GB)