**Get Data**

1. U10, V10, SLP in single levels

import cdsapi

dataset = "reanalysis-era5-single-levels"

request = {

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

"variable": [

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

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

"mean\_sea\_level\_pressure"

],

"year": ["2022"],

"month": ["03"],

"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()

1. T, H in pressure levels

import cdsapi

dataset = "reanalysis-era5-pressure-levels"

request = {

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

"variable": [

"geopotential",

"temperature"

],

"year": ["2022"],

"month": ["03"],

"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"],

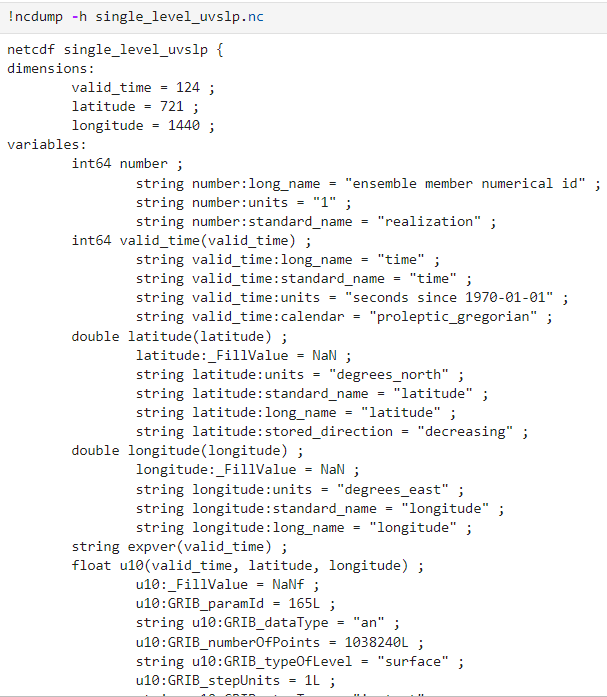
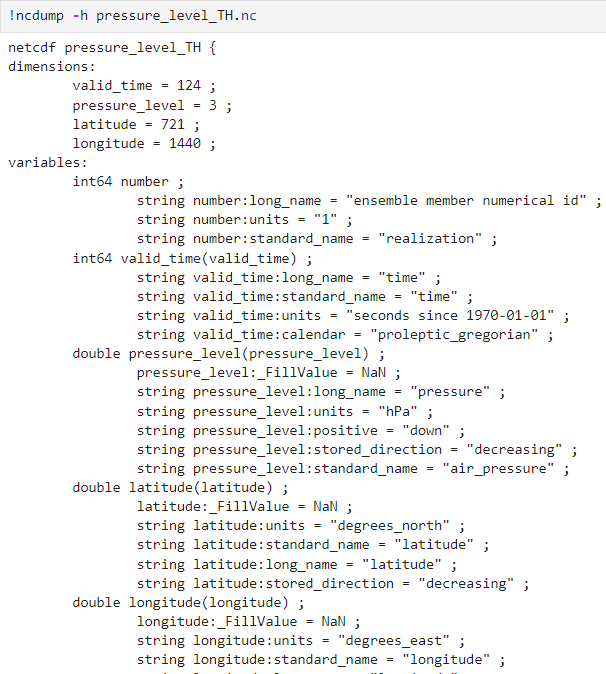
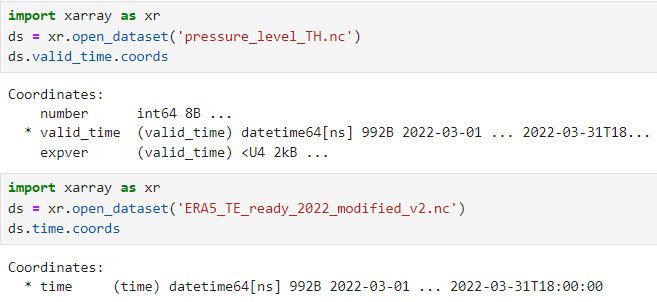
"data\_format": "netcdf",

"download\_format": "unarchived"

}

client = cdsapi.Client()

client.retrieve(dataset, request).download()



**Merge Data**

We try to adjust the attributes of the obtained data to match the attributes of TE\_ready\_198001.nc.

We started by adjusting each variable of the data one by one.

Lat:

import xarray as xr

ds = xr.open\_dataset('/data0/zy2608/ERA5\_ready.nc')

ds = ds.rename({'latitude': 'lat'})

if 'standard\_name' in ds['lat'].attrs:

del ds['lat'].attrs['standard\_name']

if 'stored\_direction' in ds['lat'].attrs:

del ds['lat'].attrs['stored\_direction']

ds['lat'].attrs.update({

'long\_name': 'latitude',

'units': 'degrees\_north',

'vmax': 1.e+15,

'vmin': -1.e+15

})

ds.to\_netcdf('/data0/zy2608/ERA5\_lat\_ready\_updated.nc')

SLP:

import xarray as xr

ds = xr.open\_dataset('/data0/zy2608/ERA5\_lat\_ready\_updated.nc', mode='r+', engine='netcdf4')

ds = ds.rename({'slp': 'SLP'})

attrs\_to\_remove = [

'GRIB\_paramId', 'GRIB\_dataType', 'GRIB\_numberOfPoints', 'GRIB\_typeOfLevel',

'GRIB\_stepUnits', 'GRIB\_stepType', 'GRIB\_gridType', 'GRIB\_uvRelativeToGrid',

'GRIB\_NV', 'GRIB\_Nx', 'GRIB\_Ny', 'GRIB\_cfName', 'GRIB\_cfVarName',

'GRIB\_gridDefinitionDescription', 'GRIB\_iDirectionIncrementInDegrees',

'GRIB\_iScansNegatively', 'GRIB\_jDirectionIncrementInDegrees',

'GRIB\_jPointsAreConsecutive', 'GRIB\_jScansPositively',

'GRIB\_latitudeOfFirstGridPointInDegrees', 'GRIB\_latitudeOfLastGridPointInDegrees',

'GRIB\_longitudeOfFirstGridPointInDegrees', 'GRIB\_longitudeOfLastGridPointInDegrees',

'GRIB\_missingValue', 'GRIB\_name', 'GRIB\_shortName', 'GRIB\_totalNumber', 'GRIB\_units',

'GRIB\_surface', 'coordinates'

]

for attr in attrs\_to\_remove:

if attr in ds['SLP'].attrs:

del ds['SLP'].attrs[attr]

ds['SLP'].attrs.update({

'long\_name': 'sea\_level\_pressure',

'units': 'Pa',

'fmissing\_value': '1.e+15f',

'standard\_name': 'sea\_level\_pressure',

'vmax': '1.e+15f',

'vmin': '-1.e+15f',

'valid\_range': ['-1.e+15f', '1.e+15f'],

'add\_offset': '0.f',

'scale\_factor': '1.f',

'missing\_value': '1.e+15f'

})

encoding = {

'SLP': {

'\_FillValue': 1.e+15,

}

}

ds.to\_netcdf('/data0/zy2608/ERA5\_ready\_slp\_modified.nc', mode='w', engine='netcdf4', encoding=encoding)

Then we use ERA5\_ready\_slp\_modified.nc updated data to reduce and set attributes in slp:

import xarray as xr

ds = xr.open\_dataset('/data0/zy2608/ERA5\_ready\_slp\_modified.nc', mode='r+', engine='netcdf4')

ds = ds.rename({'U10': 'U10M'})

attrs\_to\_remove = [

'GRIB\_paramId', 'GRIB\_dataType', 'GRIB\_numberOfPoints', 'GRIB\_typeOfLevel',

'GRIB\_stepUnits', 'GRIB\_stepType', 'GRIB\_gridType', 'GRIB\_uvRelativeToGrid',

'GRIB\_NV', 'GRIB\_Nx', 'GRIB\_Ny', 'GRIB\_cfName', 'GRIB\_cfVarName',

'GRIB\_gridDefinitionDescription', 'GRIB\_iDirectionIncrementInDegrees',

'GRIB\_iScansNegatively', 'GRIB\_jDirectionIncrementInDegrees',

'GRIB\_jPointsAreConsecutive', 'GRIB\_jScansPositively',

'GRIB\_latitudeOfFirstGridPointInDegrees', 'GRIB\_latitudeOfLastGridPointInDegrees',

'GRIB\_longitudeOfFirstGridPointInDegrees', 'GRIB\_longitudeOfLastGridPointInDegrees',

'GRIB\_missingValue', 'GRIB\_name', 'GRIB\_shortName', 'GRIB\_totalNumber', 'GRIB\_units',

'GRIB\_surface', 'coordinates'

]

for attr in attrs\_to\_remove:

if attr in ds['U10M'].attrs:

del ds['U10M'].attrs[attr]

if '\_FillValue' in ds['U10M'].attrs:

del ds['U10M'].attrs['\_FillValue']

ds['U10M'].attrs.update({

'long\_name': '10-meter\_eastward\_wind',

'units': 'm s-1',

'fmissing\_value': float(1.e+15),

'standard\_name': '10-meter\_eastward\_wind',

'vmax': float(1.e+15),

'vmin': float(-1.e+15),

'valid\_range': [float(-1.e+15), float(1.e+15)],

'add\_offset': float(0.0),

'scale\_factor': float(1.0),

'missing\_value': float(1.e+15)

})

ds.to\_netcdf('/data0/zy2608/ERA5\_ready\_U10M\_modified.nc', mode='w', engine='netcdf4')

However, there is something wrong with the code:

We don’t know how to solve it.

Therefore, we change another method to update attributes of the data:

import xarray as xr

ds\_surface = xr.open\_dataset('single\_level\_uvslp.nc')

ds\_surface = ds\_surface.rename({

'latitude': 'lat',

'longitude': 'lon',

'valid\_time': 'time',

'msl': 'SLP',

'u10': 'U10M',

'v10': 'V10M'

})

ds\_pl = xr.open\_dataset('pressure\_level\_TH.nc')

ds\_pl = ds\_pl.rename({

'latitude': 'lat',

'longitude': 'lon',

'valid\_time': 'time',

'pressure\_level': 'lev',

't': 'T',

'z': 'H',

})

# Convert geopotential to geopotential height (H) in meters

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'

# Select variables to merge

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

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

# Merge datasets

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

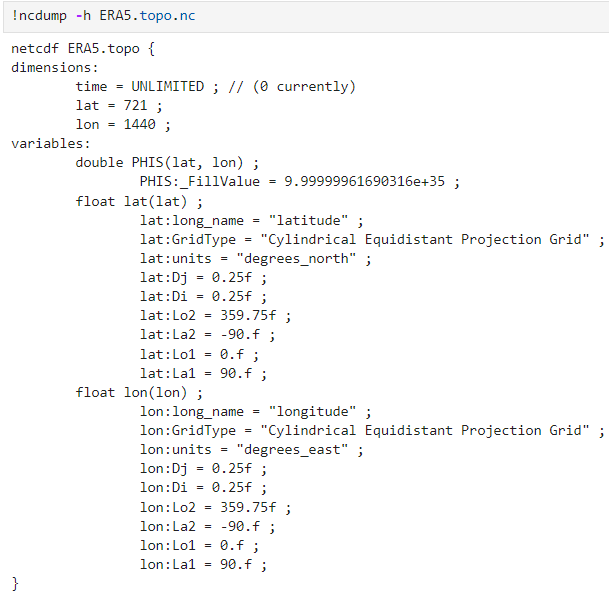
# Ensure time coordinates are consistent

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

# Save to NetCDF

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





**Unity of attributes and variables**

Please see the codes in Github

