Extraction and Analysis of Degree Heating Weeks Data in NetCDF Files

Tonoya Ahmed

Background

- Raised temperatures are leading to the increase of frequency in warm water events, resulting in more frequent and more intense coral bleaching events
- To understand how to quantify and measure this process, Degree Heating Weeks (DHW) are used to calculate exactly how much thermal stress a specific coral reef is under at a given time.
- A Degree Heating Week is calculated by accumulating the temperatures that are over the highest summer mean for a particular location for a specific amount of time.
- Example: if the temperature of a location is 2°C above the maximum for 4 weeks, the DHW measure is 8.
- This value can quantify how much heat has built up in that area over a long period of time.
- Worldwide temporal data is stored by the National Oceanic and Atmospheric Administration (NOAA) in a file collection, measured using Degree Heating Weeks

Project Goal

The goal of this project is to utilize the NOAA file collection to conduct data analysis that will simplify complex temporal data files into a more digestible format, making the data easier to read, interpret, and wrangle.

Data

- The geographical and temporal data is all stored in a collection of NetCDF files created by the NOAA
- NetCDF files are commonly used for storing climate data in multidimensional arrays, organized in a hierarchical manner by parameters such as coordinate locations, time, and temperature
- Each NetCDF file contains data for a 24 hour period
- Each file contains the DHW values for different longitude and latitude pairs across the globe for the specified day
- Dimensions and variables:

```
    dimensions(sizes): time(1), lon(7200), lat(3600)
    variables(dimensions): int32 time(time), float32 lat(lat), float32 lon(lon), int16 degree_heating_week(time, la
t, lon), uint8 mask(time, lat, lon), int16 crs()
```

Data Analysis and Code

Part 1 - Unpacking NetCDF Files

- Unpackaging and viewing the data in NetCDF files required using the netCDF4 package, a python package designed to aid extracting information from NetCDF files and perform data processing tasks
- Used netCDF4 functions to analyze metadata of files and understand hierarchical organization of stored data.
- Example metadata:

```
    source: Coral Reef Watch Daily Global 5km Satellite Sea Surface Temperature v3.1 (CoralTemp v3.1)
    spatial_resolution: 0.05 degrees
    standard_name_vocabulary: CF Standard Name Table v27
    summary: This is a product of NOAA Coral Reef Watch Daily Global 5km Satellite Coral Bleaching Heat Stress Moni
toring Product Suite
    time_coverage_duration: P1D
    time_coverage_end: 20231108T000000Z
    time_coverage_resolution: P1D
    time_coverage_start: 20231107T000000Z
    title: NOAA Coral Reef Watch Daily Global 5km Satellite Coral Bleaching Degree Heating Week
    uuid: 24c7f597-298c-4be6-a0c2-ad97b993f60f
    dimensions(sizes): time(1), lon(7200), lat(3600)
    variables(dimensions): int32 time(time), float32 lat(lat), float32 lon(lon), int16 degree_heating_week(time, la
t, lon), uint8 mask(time, lat, lon), int16 crs()
    groups:
```

Part 2 - Accessing DHW Data

- Sampled a small selection of observations from the datasets to access the Degree Heating Weeks data
- Sampling methods:
  - Geographical - Isolated data regarding the Palmyra Islands using a coordinate based bounding box
  - Time - Selected several days in October and November, a time period in which DHW measurements are expected to be higher, due to buildup of summer heat
- Initial unprocessed data:

```
dhw = file.variables['degree_heating_week']
data = dhw[:]
data
```

```
masked_array(
  data=[[[--, --, --, ..., --, --, --],
         [--, --, --, ..., --, --, --],
         [--, --, --, ..., --, --, --],
         ...,
         [--, --, --, ..., --, --, --],
         [--, --, --, ..., --, --, --],
         [--, --, --, ..., --, --, --]]],
  mask=[[[ True,  True,  True, ...,  True,  True,  True],
         [ True,  True,  True, ...,  True,  True,  True],
         [ True,  True,  True, ...,  True,  True,  True],
         ...,
         [ True,  True,  True, ...,  True,  True,  True],
         [ True,  True,  True, ...,  True,  True,  True],
         [ True,  True,  True, ...,  True,  True,  True]]],
  fill_value=-32768)
```

- Final processed data:

```
import numpy as np

lat = file.variables['lat'][:]
lon = file.variables['lon'][:]

lat_min = 5.5
lat_max = 7.0
lon_min = -163.0
lon_max = -161.5

lat_inds = np.where((lat >= lat_min) & (lat <= lat_max))[0]
lon_inds = np.where((lon >= lon_min) & (lon <= lon_max))[0]

degree_heating_week = f.variables['degree_heating_week'][:, lat_inds, lon_inds]

print(degree_heating_week)
```

```
[[[1.57 1.58 1.57 1.58 1.73 1.87 1.88 1.73 1.88 2.18 2.18 2.33 2.79 2.8
   2.81 2.94 2.82 2.82 2.83 2.82 2.66 2.64 2.36 2.35 2.19 2.2  2.04 1.89
   1.9  2.05]
  [1.59 1.59 1.73 2.03 2.04 1.89 2.05 2.19 2.2  2.05 2.19 2.64 2.79 2.8
   2.97 2.97 2.99 3.02 3.01 2.98 2.68 2.66 2.52 2.22 2.21 2.2  1.9  1.9
   1.9  2.05]
```

Part 3 - Converting data into tabular format

- The last part of processing the data involved taking the complex NetCDF files and converting them into a simple tabular format that can easily be used for data query and analysis
- The code involved:
  - Importing necessary packages
  - Retrieving parsed data about DHW, longitude, latitude, time
  - Creating data grids to store parsed data
  - Unpackaging the multidimensional parsed data into one-dimensional arrays
  - Creating tabular dataframe
  - Writing the streamlined data into the data frame
  - Storing the data frame in a CSV file
  - Displaying example observations

```
import cdsapi
import netCDF4
from netCDF4 import num2date
import numpy as np
import pandas as pd

dhw = f.variables['degree_heating_week']
time_dim, lat_dim, lon_dim = dhw.get_dims()
time_var = f.variables[time_dim.name]
times = num2date(time_var[:], time_var.units)
latitudes = f.variables[lat_dim.name][:]
longitudes = f.variables[lon_dim.name][:]

output_dir = './'
filename = os.path.join(output_dir, 'output_table.csv')

times_grid, latitudes_grid, longitudes_grid = [
    x.flatten() for x in np.meshgrid(
    times, latitudes, longitudes, indexing='ij')]
df = pd.DataFrame({
    'time': [t.isoformat() for t in times_grid],
    'latitude': latitudes_grid,
    'longitude': longitudes_grid,
    'dhw': dhw[:].flatten()})

df.to_csv(filename, index=False)
example_output = df[df["dhw"]>0]
example_output = example_output.reset_index()
                               .drop(columns = ["index"])
example_output
```

	time	latitude	longitude	dhw
0	2023-11-07T12:00:00	83.625000	-32.674999	0.15
1	2023-11-07T12:00:00	83.625000	-32.625000	0.15
2	2023-11-07T12:00:00	83.625000	-32.575001	0.15
3	2023-11-07T12:00:00	83.625000	-32.525002	0.16
4	2023-11-07T12:00:00	83.625000	-32.474998	0.16
...	...	...	...	...
4653477	2023-11-07T12:00:00	-59.924999	170.824997	0.15
4653478	2023-11-07T12:00:00	-59.924999	170.875000	0.15
4653479	2023-11-07T12:00:00	-59.924999	170.925003	0.15
4653480	2023-11-07T12:00:00	-59.924999	170.975006	0.15
4653481	2023-11-07T12:00:00	-59.924999	171.024994	0.15

4653482 rows × 4 columns

Challenges

- Had difficulty unpacking the organizational structure of NetCDF files, due to the complexity and unfamiliarity of the file structure
- Had trouble accessing DHW data, since so many of the values inside the array were masked
- Initially started sampling at January 1st, which mostly contained DHW values of 0, due to the fact that it is in the middle of winter, resulting in the misconception that all the values were missing

Future Work

- Generalize the data analysis onto all of the NetCDF files outside of the sample selection
- Use the outputted CSV files to create a querying system, involving a simple interface that allows the user to query temporal data using a date and coordinate pair as query input