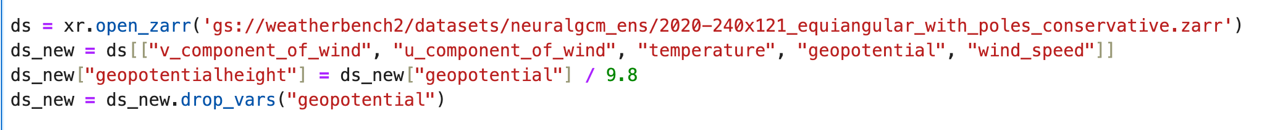
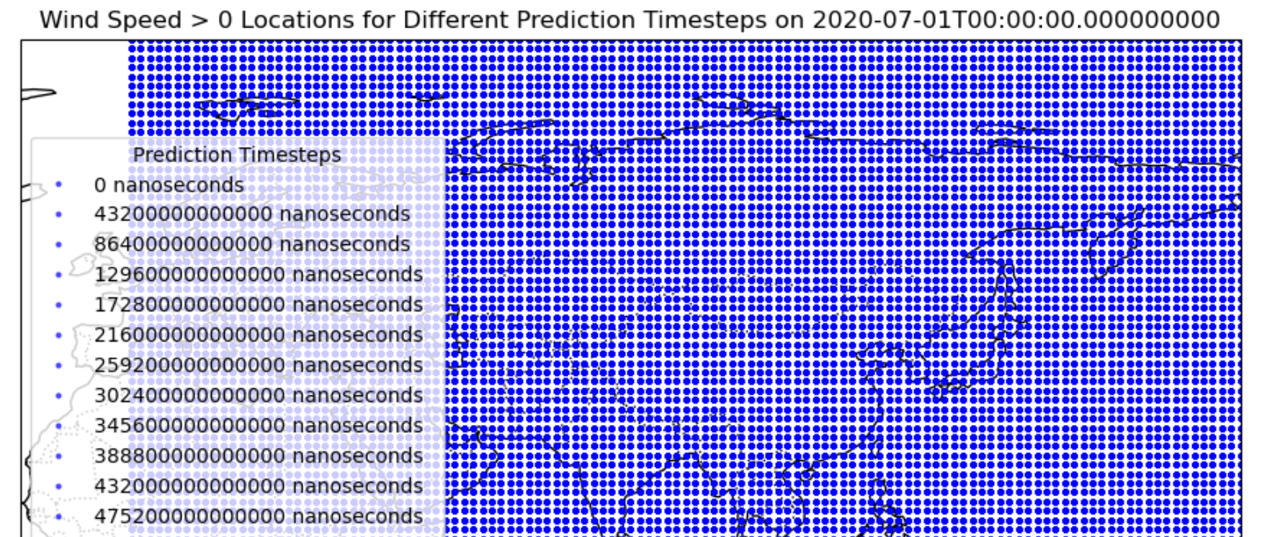
Extract NeuralGCM data and plot the graph:



We extract the v\_component\_of\_wind, u\_component\_of\_wind, temperature, geopotential, wind\_speed. And change geopotential to geopotential height.

Then we want to plot the graph of predicited track, with initial time = 7.1, and prediction time = 15 days.

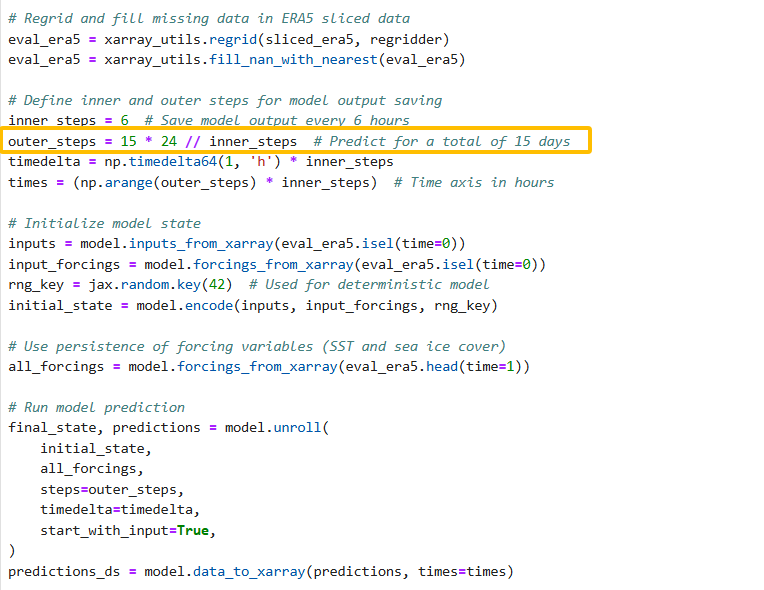
However, there is a issue when we plotting the graph. The dimension of the variable is 6, we fixed time=7.1, and we use a for loop in taking the prediction\_timedelta in order to fix the prediction\_timedelta in each iteration, then the dimension of variable will reduce to 4. We use wind\_speed to plot the track of wind, and filter the data with wind\_speed >0. And we want to use the lat and lon to show the track, but we don’t know the meaning of realization, besides in the previous data, we choose level = 300,400,500. In the graph plotting, we don’t know which level should we plot.



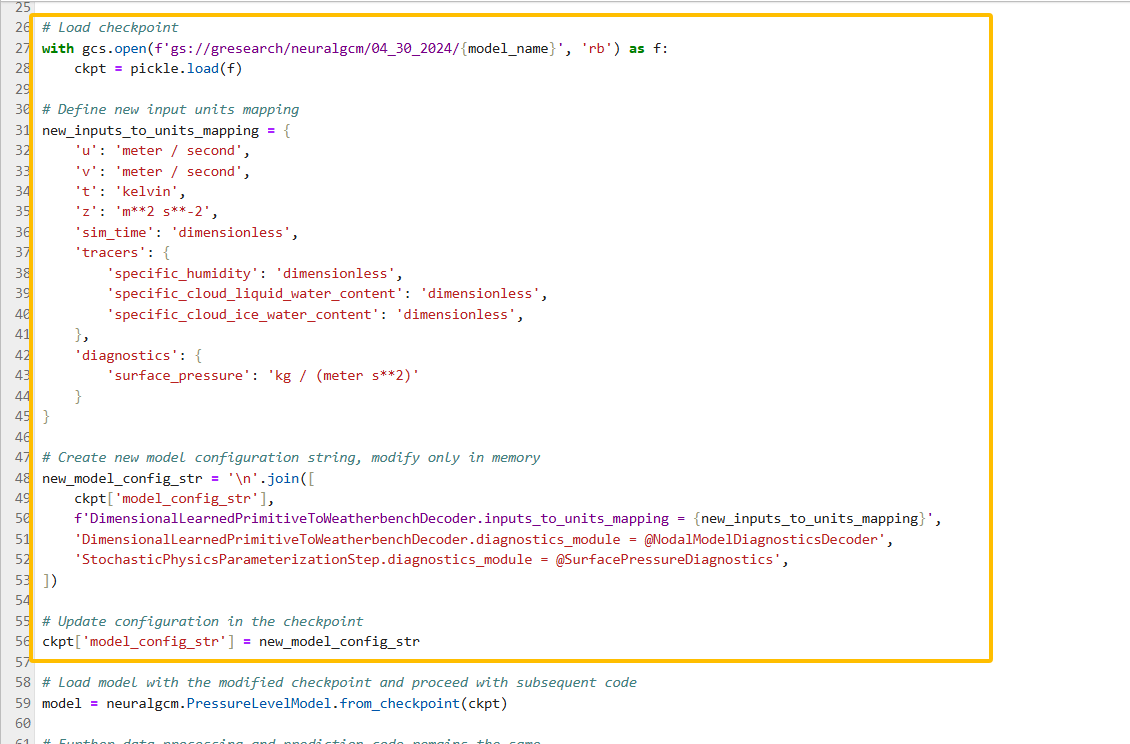
Extract the code for NeuralGCM

Please see **getonemonth.py**

Minor problem:



“15” should be “30”.



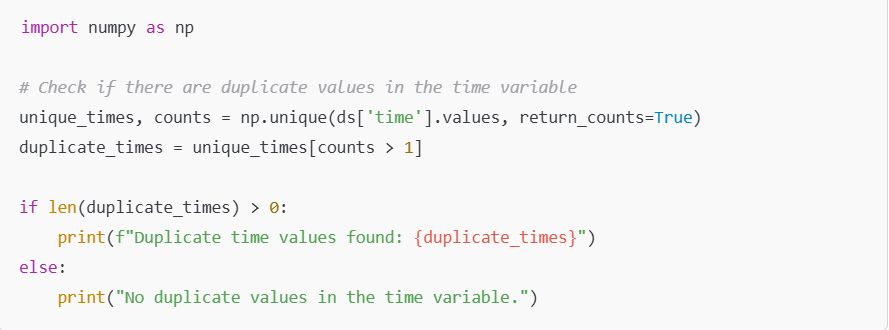
New parts compared to the original code in “Forecasting quick start”.

Then, please see **addingunits\_onemonth.py.** There are 12 steps in total. We can get **predictions\_output\_onemonth\_updated.nc** file.

However, we are still encountering issues related to time and floating-point data types that prevent TE from running successfully. Specific issues include cases where the "time" variable, whether in int or double format, does not allow the data to execute. It's unclear if "unit" is the sole critical factor for making TE run properly, and there are overlapping segments in the time parameters. As a result, we ran **TEready\_onemonth.py** to further optimize our original **updated\_onemonth.nc** dataset. Our output is **predictions\_output\_onemonth\_TE\_ready.nc** file.

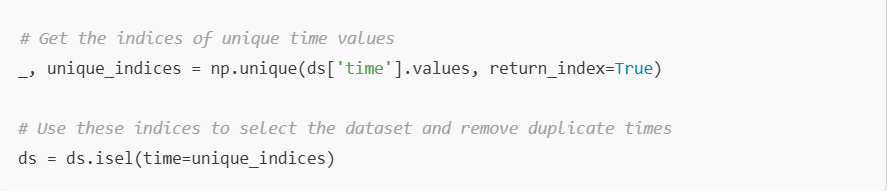
Specifically, when processing a NetCDF dataset, the time variable may contain duplicate time values, which can cause errors when running tools like TempestExtremes (TE). For example, logs may show that all times are recognized as 1970-01-01 00:00:00, typically due to improper handling of the time variable.

First, we need to confirm whether the time variable contains duplicate time points. This can be achieved using the following method:



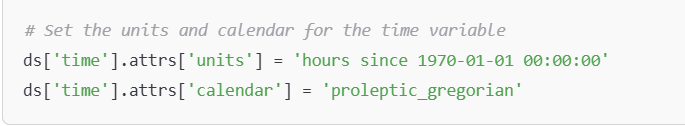
The np.unique function returns the unique values in the array and the number of times each unique value appears. By selecting unique\_times[counts > 1], to identify any time values that occur more than once. Using the isel method with unique\_indices, we can select these unique time steps to remove duplicates.

Then, we remove the duplicate time steps using the following method:



The np.unique function returns unique time values and their first occurrence indices.

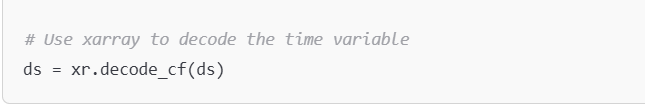
The isel method selects the dataset using these unique indices, effectively removing any duplicate time steps. To ensure that TE can correctly interpret the time variable, we set the units and calendar attributes:



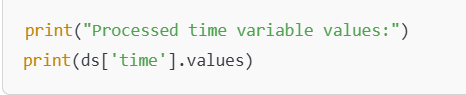
Ensure that the time variable's data type is float64 to avoid precision loss:



Utilize xarray's decode\_cf function to decode the time variable according to the Climate and Forecast metadata conventions:

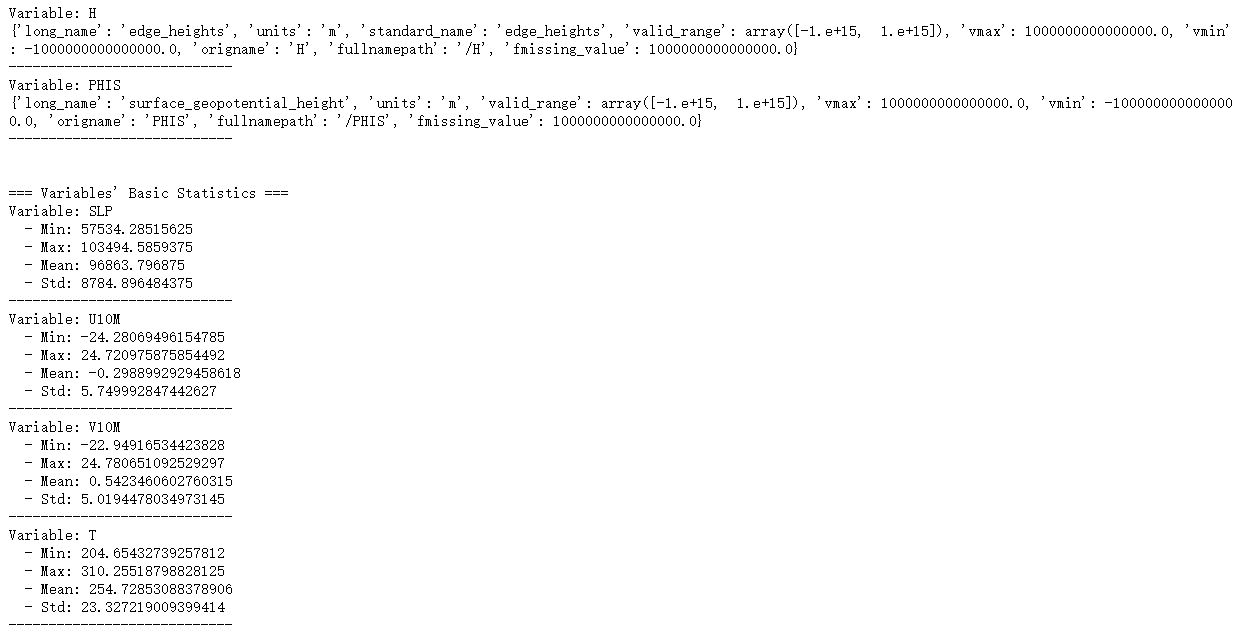
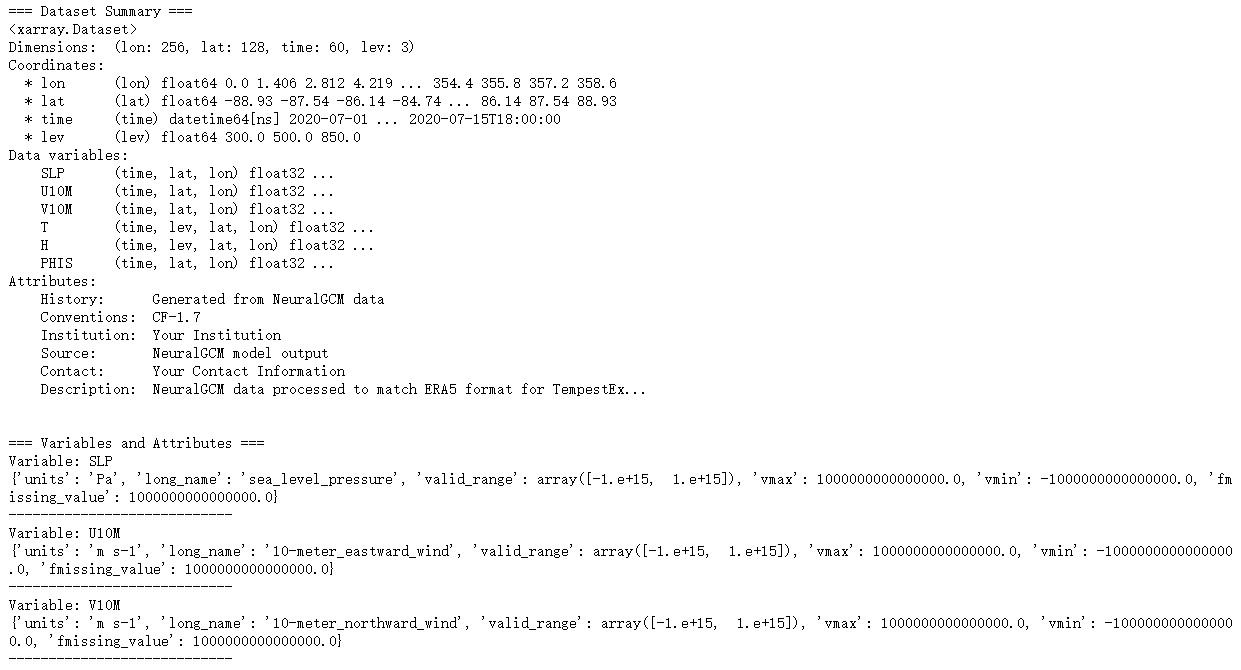


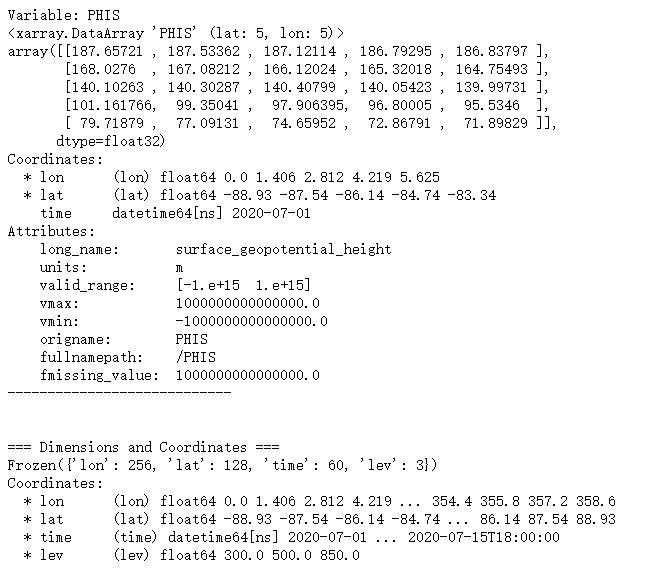
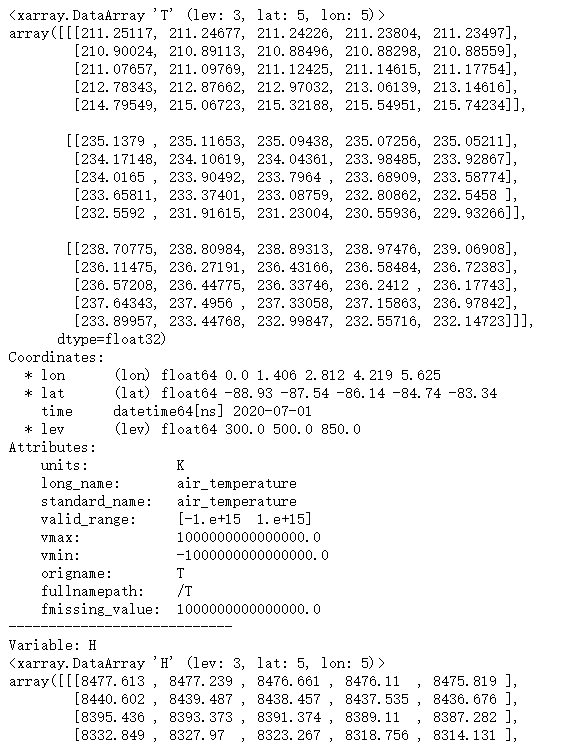
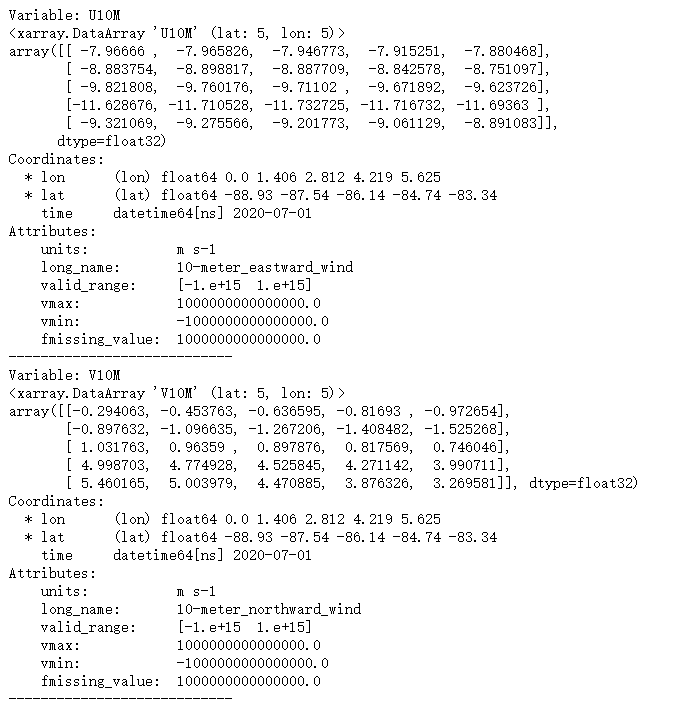
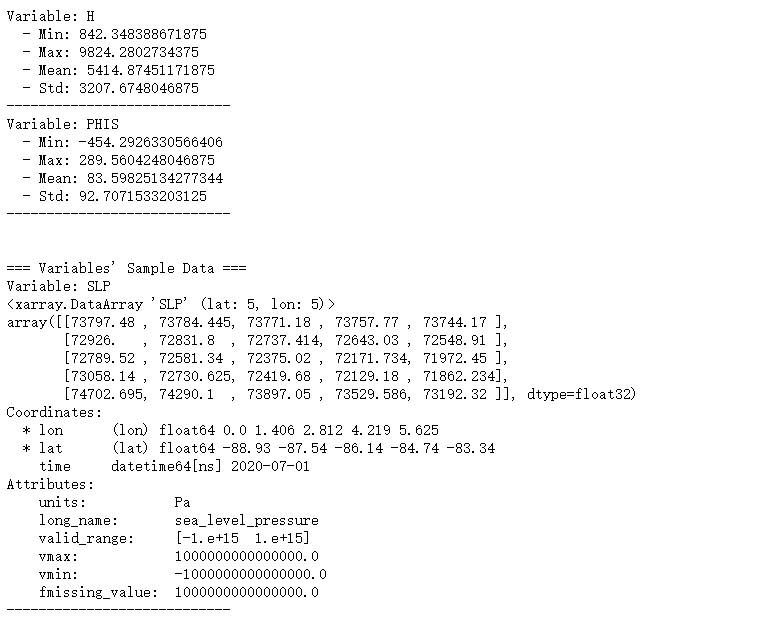
After completing the above steps, print and inspect the time variable to ensure its correctness:



Please see the code.

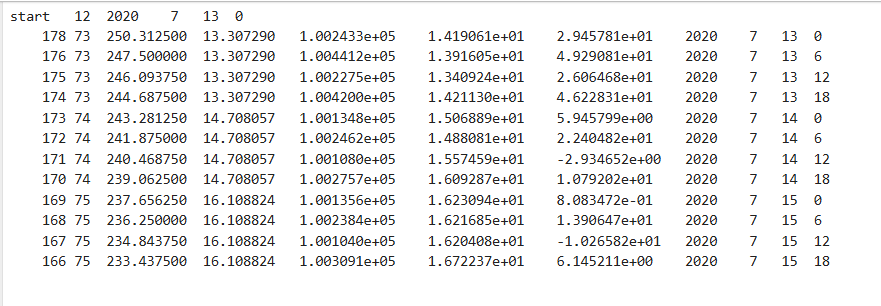
We can check data through

**checkdata.py**:



After we get this data, we put it into TE shell script to run it. Please see **track\_onemonth.sh**.

Reanalysis for 15 days data in NerualGCM using TE can be seen below:



What we are doing and future plan:

The plan is to extract a full year of data from the NeuralGCM dataset on the Forecasting Quick Start website, then perform reanalysis with TE to generate images. The data extraction process is very slow, although each extracted dataset is relatively small.