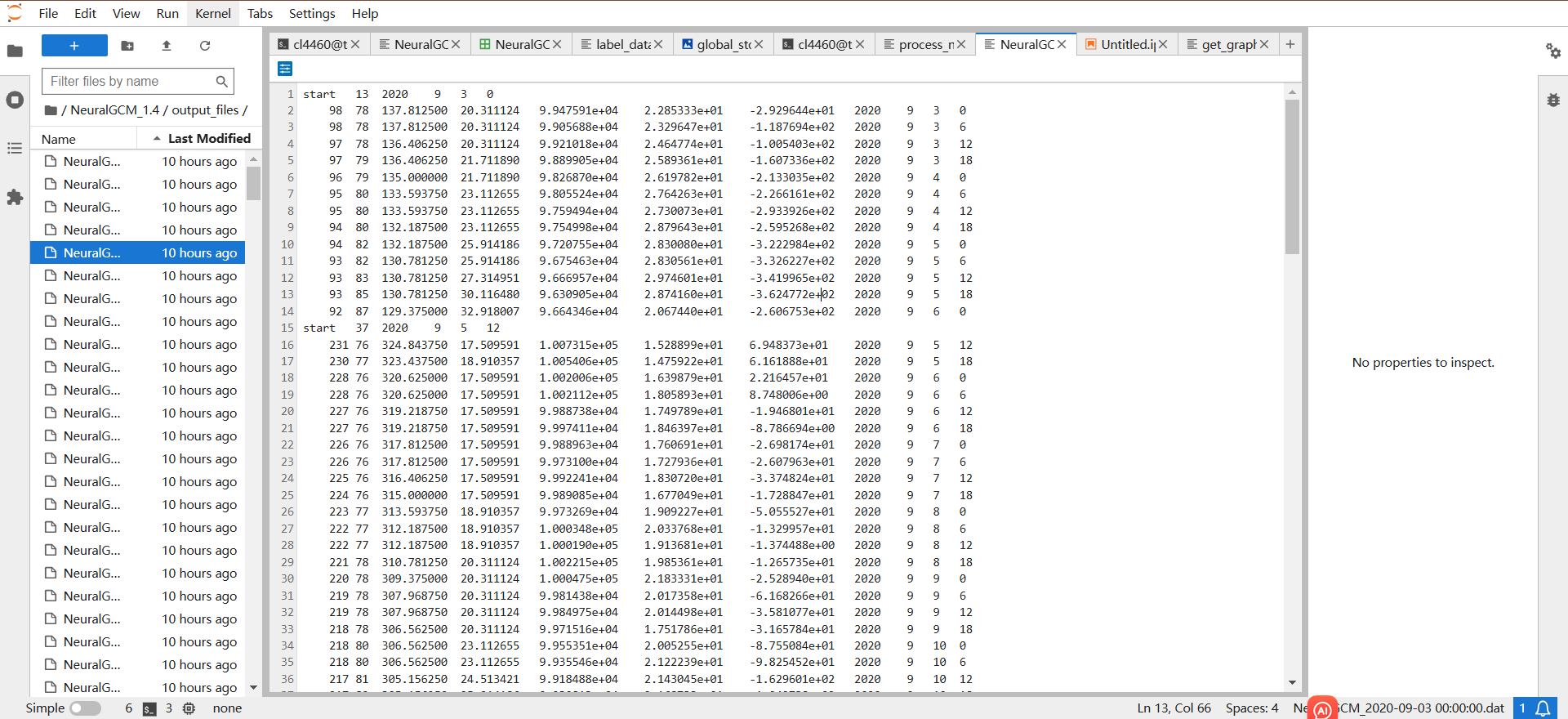
We first process the resulting data in parallel by modifying the shell script file, see **process\_nc\_files.sh**

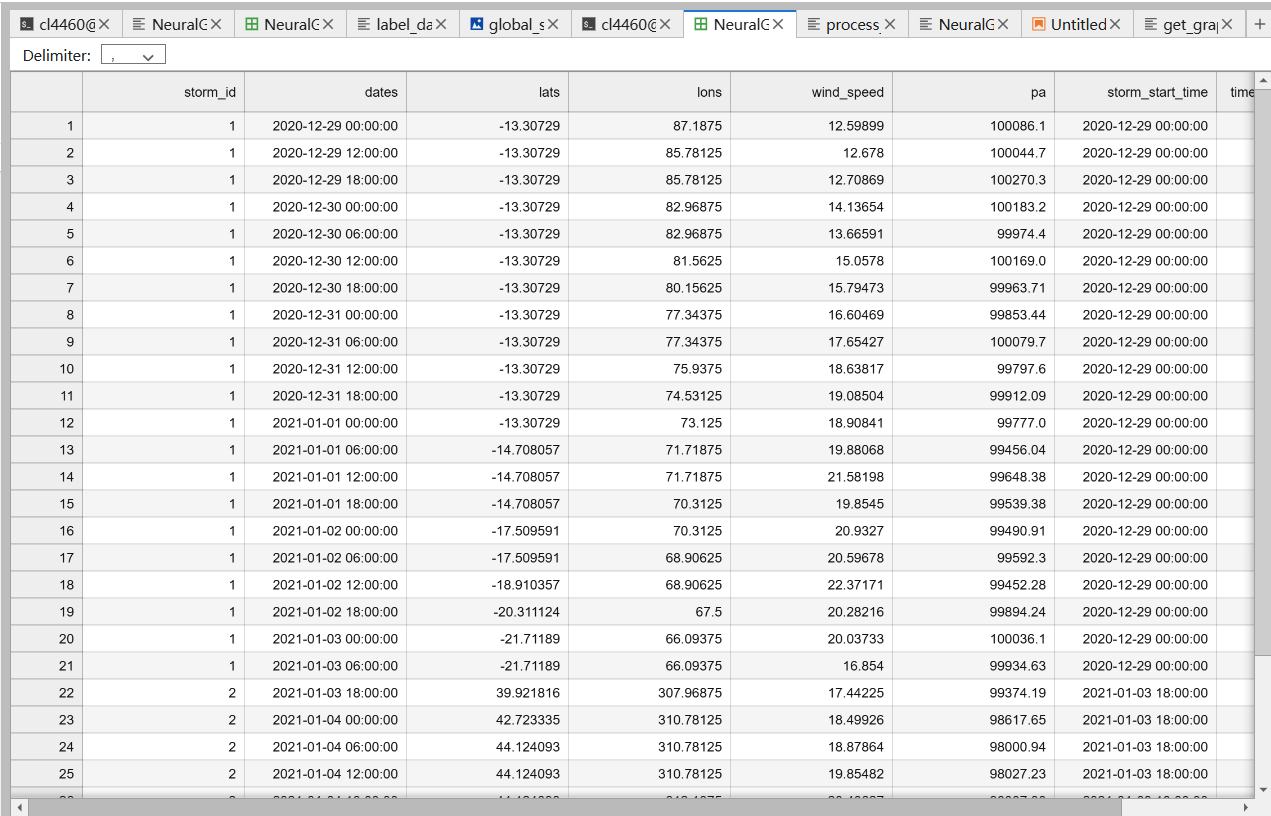


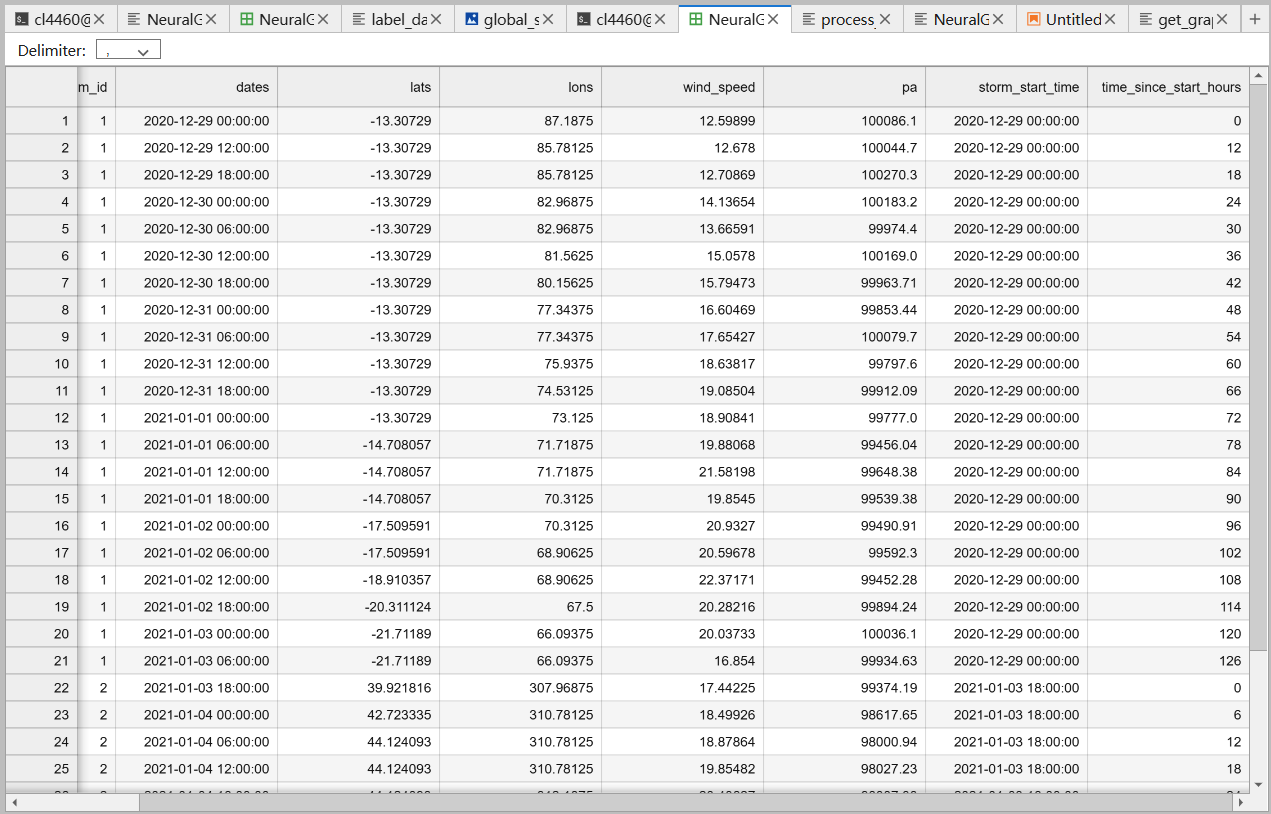
The output forms can be seen below:



For all the resulting dat files, we use **label.data.py** to labelize them.

Please see the code and one of excel table below.





Within each DataFrame, used the 'storm\_id' column to group data points belonging to the same storm, This allows for separate processing of each storm within a dataset.

**grouped = df.groupby('storm\_id')**

For each storm, sorted the data points chronologically to ensure the storm track is plotted in the correct temporal sequence.

**group = group.sort\_values('dates')**

Created an empty list to hold DataFrames from all datasets

**df\_list = []**

Iterated over all 365 datasets, processed as described above, and appended each to the list.

**for processed\_file in processed\_file\_paths:**

**df = pd.read\_csv(processed\_file)**

**...processing...**

**df\_list.append(df)**

We used a predefined list of distinct colors to represent different storms within a single dataset.

**storm\_colors = ['red', 'yellow', 'blue', 'green', 'orange', 'purple', 'cyan', 'magenta', 'brown', 'black']**

For each storm in the dataset, we assigned a color from the list based on the storm's index. The modulo operation (%) ensures that colors are reused cyclically if the number of storms exceeds the number of available colors. Colors were assigned cyclically to storms across all datasets. This means after every 10 storms, the colors repeat.

**color = storm\_colors[storm\_idx % len(storm\_colors)]**

This approach balances distinction between storms and practical limitations of color variety.

