I would start by the self-introduction first. So good morning everyone, this is Chi Zhang, working as a data science intern in the project of lightning jump detection and the severe weather forecast model construction.

In this set of slides, I will briefly go through the Lightning Jump related Algorithm as well as the Severe Hail index collection algorithm with some real-life or artificial case studies as examples.

Before we go deep into the detailed content of this talk, the overall flowchart of the project is listed in here. Beginning with the raw data analysis, followed by valid data collection and preprocessing, then move to the next step of using the 2-sigma method to detect lightning jump and apply the lightning jump centroid to capture the severe hail storm related information from the radar station. All these eight steps are built based on specific case-study for reference at the start, while the final step listed here is for enlarging the case-study dataset by applying the parallel processing with the generic algorithm.

And now, let's move to the first step of this flowchart, which is the preliminary analysis of the data that points out some information about the lightning data.

All the raw lightning data are stored in a CSV file, which contains a per-second level record during the whole year. In this project, year 2014, 2021, 2022 data are applied for producing case-study for constructing the sample code and models. Since each of this CSV file is extremely large and text splitting with bytes (which will trigger the lost of valid instance) are not allowed, an alternative way of data file preprocessing is applied. We split the yearly CSV file into a daily format (for example, there are 365 sub-file in 2021), which will take less memory occupation when loading it to the algorithm. Once the pre-construction was finished, we now look into the data information, which is listed in this slide, including the time, location and type of lightning as well as its amplitude.

With this information, we now could develop the code with some case-study, which is a specific area at the specific date with severe weather that happened before. Here, I will select the Brisbane 2014-11-27 case as an example for the next part illustration.

So, after we've learnt something about the lightning data and made some pre-construction of the data file already, the next step is to develop a preprocessing algorithm to handle the data based on some case studies.

Since we want to clarify which lightning group in a specific area is the identifier of the severe weather, a clustering algorithm is applied first.
However, before we take that step, there are still some assumptions that need to be made. Based on the lightning data, we know that there are two types of flashes, including the 'Inter-Cloud (known as IC)' and the 'Cloud to Ground (known as CG)'. Since there is only a very small proportion of CG lightning in each investigated moment and some draft visualisation shows a sparse scattering of the CG instance, we chose to ignore the lightning type by treating them as 'lightning' broadly speaking.

Then based on this assumption, we now move to the clustering section. Since we don't have any pre-assigned group label for each ground-based detected lightning, an unsupervised method is preferred for preprocessing. The DBSCAN, a Density-based spatial clustering method is chosen in this case for grouping the lightning jump within a specific period of time-interval. Presented in this slide, we can see that the DBSCAN only requires two variables that need to be pre-set, which are the minimum lightning instance within a valid cluster and the maximum distance between the lightning within a valid cluster, and we could take a look of how this method works on the real data set.

On the left plot, is the lightning visualisation in Brisbane on 2014-11-27 at 00:03 recorded in UTC style. From this plot, we could kind of see some groups of lightning and some outliers scattered around them. And then, with these variables set, we apply the DBSCAN to do clustering and reach the result in the right plot. You could see that the lightning flashes are clustered into two groups with different colours and the outliers are excluded. Besides, based on the upper bound set for the distance between lightnings, a standard for splitting the cluster is also applied.

Afterward, we will develop the method of tracking each of the clusters during the investigated day, by capturing the closest cluster in the next moment considering the current moment, or adding a new cluster from the next moment as a new track if it is not close to any of the currently investigated cluster. Present in this slide, I take one of the cluster, No.7 recorded from

Brisbane 2014-11-27 case study as an illustration of how the cluster is tracked and how it moves. On this GIF plot, there is a bounding box of the Brisbane area and the curve arrow there indicates the moving track of the cluster.

Now, let's start the GIF and see how it goes along the trail. And so, this is how the lightning cluster will be tracked with our developed algorithm and each moment of the cluster is stored to a preprocessed CSV that is used for computing the lightning jump for the further usage.

And once the preprocessing step is done, we will move to the next step of detecting the lightning jump from each tracked cluster within the case study.

In this project, we apply the 2-sigma method as standard for identifying a lightning jump from a time-period of lightning detection. For computing the sigma level with this method, 7 time-stamp of records are required. In this slide, I create some artificial data for reference, with A to G in the second row as arbitrary data, while the third row was manually created and let's focus on it. We first compute the difference between each two time period lightning flashes amount, and then calculate the standard deviation of the first five differences to obtain the sd value. Afterwards, divide the last difference by this sd value to obtain the sigma level. Once the sigma value is greater than 2 (a default case), the first condition of lightning jump is satisfied. Then, we look at the last two moment's lightning amount, if both of them are greater than 20 (a default threshold), the second condition is also fulfilled and the lightning jump is identified as 'Happen' at the 7th moment from the table.

Once the lightning jump algorithm is finished, we would pay attention to the next case of lightning jump happening along the time in each day. Presented in this huge table, which is still an artificial example for illustration, we could see several situations of the lightning jump happening. For example, the first case on the left, which is a normal case of having a lightning jump at the beginning, and the next 6 minutes range belong to the same jump (which is designed by default). Or the third case on the right, where the lightning jump happens at the middle, which we treat with a similar strategy by turning the next 6 minutes to the same jump.

What is most special is the Case 2, which I double circle it out here. After the first jump takes place, another jump happens within the six minute range. In this case, we decide to treat it as a similar jump to the previous one, and extend another 6 minutes of the same jump range presented in here for further analysis.

So that is how we identify the lightning jump at this stage, and based on this information, we could then move to my last step of using the lightning jump information to capture the severe hail information which is used for validation testing purposes.

In this slide, I select the case 1 as presented above as an example of how to collect the SHI data. We take the first moment of a lightning jump as a collector by setting its coordinate as centroid, and then create a bounding box with a range of 0.5 degree to scan through all the radar stations nearby, and pick the one which is the closest. Then, from that radar station datafile, we will use the lightning jump time as starting point, and record the next 50 minutes range SHI information, including the maximum and 90th percentile of the SHI value, the coordinate with the max SHI data as well as how large does this lightning jump detection bounding box has occupied the radar range. And with this method, the SHI information will be recorded and stored in a CSV which is handled by my Colleague Jonathan as the next stage.

An additional step is added at the end as creating more data. Since the previous six steps are constructed based on some specific case study, the information they have revealed might not be generic. In this case, by applying the mpi4py package function and the NCI platform, we could process the parallel processing in two types of way, including creating a yearly data record of a specific Area, let say Brisbane from January the first to the end of year 2021. Or we could obtain X many case studies from different areas with the specific datetime. Or in the future, if even more data is preferred, these two functions can be combined and create a yearly recorded data for all the investigated areas.

Before I finish my part, in these final two slides, all the tunable variables are listed which you could refer back to the specific slide mentioned here, where all of them are marked in red. And we currently focus on the first three variables for creating the lightning cluster at the beginning since it will affect the number of clusters as well as the lightning jump detection from the bottom, which is the foundation of the project source code.

And that is broadly about the current progress of the lightning jump and severe hail detection algorithm. Thank you for listening and I will pass to Jonathan for the next session.