AOS C111

Jack Skari

Dr. Alex Lozinski

06 December 2023

Using Machine Learning to Connect Tropical Cyclone Radii to Intensity, CAPE and Subsaturation

Introduction

Climate modeling of tropical cyclones is a difficult challenge, due to their highly dependent nature on environmental parameters. However, we can better understand the tropical cyclone's relationship with its environment by using reanalysis data. Oftentimes reanalysis data is lacking crucial information, which decreases the value of the dataset. One such example of this is TempestExtremes (TE), a subset of ERA5 reanalysis data used to track tropical cyclones using a specific algorithm. One of the details TE misses is the radii of the systems it deems to be tropical cyclones. Due to this, it becomes significantly harder to determine trends in the environment and the tropical cyclone as there is uncertainty to where a tropical cyclone begins and ends. Understanding how we can determine the radii of a tropical cyclone from given data is crucial to be able to draw conclusions from data in order to optimize climate models.

In order to combat the problem with this dataset, I collected TE data for storms from 2002 to 2013 in the Atlantic basin for each individual storm's minimal pressure, convective

available potential energy in the lower free troposphere ($CAPE_L$), and subsaturation in the lower free troposphere ($SUBSAT_L$) at every 6 hour timestep. The objective of doing this is to average the $CAPE_L$ and $SUBSAT_L$ from both inside and outside of the storm (discussed further in the data section). Doing this will allow me to plot these two variables in relation to minimal pressure at each point as a 3D plot. I will be able to employ a support vector machine (SVM) algorithm in order to classify whether a given point is inside or outside of tropical storm force winds.

From this, I found success using a 3D SVM, and managed to create a decision boundary and a large percentage of the data points functioned as support vectors. With it, I was able to reach an accuracy of 81.0% according to the Jaccard score, and create a 3D shape that divided the inside and outside values. I found the SVM method to work very well when creating the decision boundary. The same cannot be said when the support vectors are plotted alone. Due to the heavy overlap in the data, the support vectors proved difficult in determining approximately where the decision boundary would fall. In conclusion, I found that it is possible to use $CAPE_L$, $SUBSAT_L$, and pressure values both inside and outside the storm in order to determine the radii of a TC via the support vector classifier method.

Data

In order to begin constructing this model, I first took preprocessed tropical cyclone data from the TE algorithm mentioned above. TE is a subset of ERA5 data following a tropical cyclone detection algorithm. This algorithm works by searching for two criteria: a location with a large pressure drop, and an upper-level warm core. (Ulrich et al. 2021) The algorithm then stitches these storms in time steps of 6 hours apart as long as both criteria are present. TE

inherently also includes environmental conditions, such as equivalent potential temperature (θ_e) at multiple locations in the atmosphere. With this preprocessed data, I was able to extract θ_e in the atmospheric boundary layer (θ_{eB}), saturated θ_e in the lower free troposphere (θ^*_{eL}), and unsaturated θ_e in the lower free troposphere (θ_{eL}) for each storm at each time step. Thus I had access to every tropical cyclone TE had identified from 2002 to 2013, and the global environment around them.

Using the equation derived from "A Process-Oriented Diagnostic to Assess Precipitation-Thermodynamic Relations and Application to CMIP6 Models" by Fiaz Ahmed and David Neelin, I was able to convert θ_e, θ^*_{eL}, and θ_{eL} into $CAPE_L$ and $SUBSAT_L$. I now had each tropical cyclone and its intensity in its environment with $CAPE_L$ and $SUBSAT_L$. However, I lacked the ability to discern the exact radii of each storm, as this was the problem I was having my algorithm solve. To work around this, I took the dataset of tropical cyclones by wind radii from "Dataset of outer tropical cyclone size from a radial wind profile," by Albenis Pérez-Alcarón et al. Due to time constraints, I only was able to take tropical cyclones from the North Atlantic basin. I classified the "inside" of the tropical cyclone as the radius (in km) of tropical storm-force winds from the center, and the "outside" to be all values 2.5 degrees out from the wind radii, as shown in figure 1. With this, I manually fed the radii to code to average inside and outside of the tropical cyclones. In addition, I ignored storm time steps that had passed 34.75 degrees latitude or whose radii plus outer 2.5 degrees passed this value as TE is missing data from beyond this latitude.

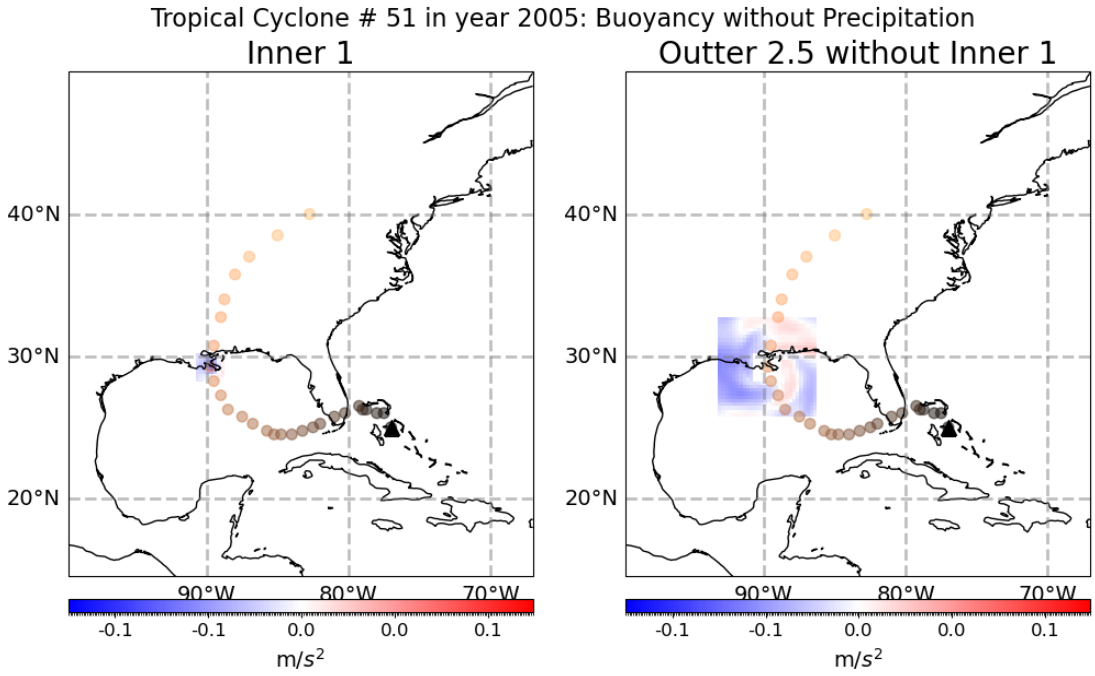

Figure 1: An environmental factor (buoyancy) with only values inside (left) and only outside values (right) for hurricane Katrina right before landfall

Due to the radii from Pérez-Alcarón et al. being in kilometers, and not in degrees latitude and longitude, I both converted the radii into degree latitude and rounded to the nearest .25 due to the grid resolution being .25 degrees latitude and longitude. To make the data size more manageable, I averaged the $CAPE_L$ and $SUBSAT_L$ for both the inside and outside of the tropical cyclone, as well as taking the minimal pressure for each time step. For a single time step, both inside values and outside values share a minimal pressure, due to the desire to relate $CAPE_L$ and $SUBSAT_L$ to pressure. As shown in figure 2, I then plotted both the internal and external values as a 3D plot. The data was now ready to be fed into a machine learning algorithm to classify.

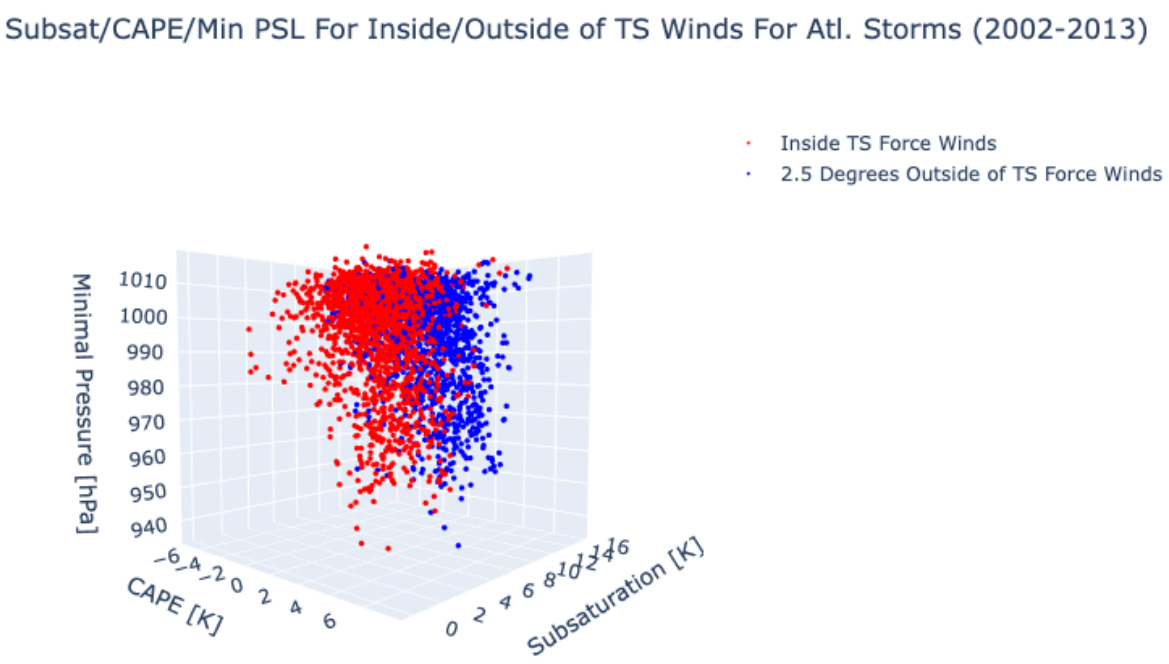

Figure 2: All averaged $CAPE_L$ and $SUBSAT_L$ values for both inside and outside of the tropical cyclones

Modeling

Due to the nature of the problem, a classifier model was needed. Neural networks would not serve well here, as I found that the decision boundary needed here was significantly simpler and didn't need the complex training entailed with it. This meant I needed a supervised classifier model rather than an unsupervised one. As I was looking for a specific equation to use a classifier for, decision trees were deemed to not be a good candidate. This left me with the logistic equation and SVM algorithms as potential choices. Since I sought a non uniformly shaped boundary (as there was a heavy overlap at weaker intensities), I found the SVM algorithm to be most ideal for this problem.

Interestingly, I had to do little to optimize the algorithm function. The two values I mostly had tinkered with were the regularization parameter and tolerance. The regularization parameter affects how hard a boundary is chosen for the classification at the cost of making the decision boundary trend very close to the support vectors and less able to be generalized. This results in a stronger classifier, but runs the risk of doing poorly on unseen data. Tolerance is how hard the algorithm works to converge on its classification, with smaller values meaning the algorithm needs to spend more time converging, while greater values means it can stop sooner. When I increased the regularization parameter from 1 to 10, the accuracy of the decision boundary fell slightly by .2% and the amount of support vectors decreased by about 8.25%, meaning the algorithm was able to increase the margin between the decision boundary and them. This had the opposite effect of what increasing the regularization parameter should do. When I decreased the tolerance, the amount of support vectors decreased by a negligible amount, while the accuracy fell by about .1%. In general, altering the parameters here doesn't optimize the algorithm much. As seen in figure 3, the kernel chosen for the SVM is the linear kernel, as more complex kernels have trouble in the 3D space when creating the decision boundary, and due to the desire for a straight forward boundary.

```
sv_fit = SVC(C=1,kernel='linear',tol=1e-3) #creating the support vector machine
mySVM=sv_fit.fit(trueallarray,decision) #fitting the SVM
svs = mySVM.support_vectors_ #for the support vectors
z = lambda x,y: (-mySVM.intercept_[0]-mySVM.coef_[0][0]*x-mySVM.coef_[0][1]*y) / mySVM.coef_[0][2]
#for the decision boundary in 3D
print("The shape of the support vectors is {}.".format(svs.shape)) #get the amount of support vectors
print("The equation relating pressure, CAPE, and substauration is (- {:.2f} - {:.2f}*CAPE - {:.2f}*SUBSAT)/{:.2f
```

```
The shape of the support vectors is (1515, 3).
The equation relating pressure, CAPE, and substauration is (- 29.99 - -0.07*CAPE - -0.88*SUBSAT)/-0.02 = Pressure
```

```
hit_rate = [] #I want to get the accuracy of the decision boundary
for_inner=[]
for_outer=[]
yes_check = np.concatenate((yes,yes),axis=None) #due to the jaccard not working well for detecting misses
for r in range(len(cape_inner)):
    if (z(cape_inner[r],subs_inner[r]) > psl[r]):
        hit_rate.append(1) #1 if it on the correct side of the decision boundary
        for_inner.append(1) #if I want the jaccard score of just inner or outter
    else:
        hit_rate.append(0)
        for_inner.append(0) #0 if it is on the incorrect side of the decision boundary
for r in range(len(cape_outer)):
    if (z(cape_outer[r],subs_outer[r]) < psl[r]):
        hit_rate.append(1) #1 if it is on the correct side of the decision boundary, should be 0 but the jaccard
        for_outer.append(1) #flip since it doesn't work well for 0 values
    else:
        hit_rate.append(0) #0 if it is on the incorrect side of the decision boundary, should be 1 but the jacca
        for_outer.append(0) #flip since it doesn't work well for 0 values
accuracy=jaccard_score(yes_check, hit_rate) #compares calculated values to actual values
print("The accuracy for predicting inner values: {}".format(jaccard_score(yes, for_inner)))
print("The accuracy for predicting outer values: {}".format(jaccard_score(yes, for_outer)))
print("The overall acurracy of the decision boundary: {}".format(accuracy))
```

```
The accuracy for predicting inner values: 0.823237885462555
The accuracy for predicting outer values: 0.7973568281938326
The overall acurracy of the decision boundary: 0.8102973568281938
```

Figure 3: The setup of the SVM, for both the decision boundary and the support vectors, and the calculation of the accuracy using the Jaccard score.

Results

As seen in figure 4, I plugged the equation found in figure 3 into the graph with pressure functioning as z. This results in a decision boundary that splits the data in two with only significant overlap happening at weaker intensities. The decision boundary managed to reach an accuracy of 81.0% correct prediction on the data. The accuracy was higher for predicting $CAPE_L$ and $SUBSAT_L$ values within the TC than for outside of the TC, at 82.3% and 79.4% respectively. The decision boundary function results in a shape that is greatly elongated, but for the sake of seeing the individual points better, I limited the vertical height of the decision boundary by reducing the z-axis.

Figure 5 shows the support vectors instead of the decision boundary. Most support vectors are clustered towards the dividing point between the two values, as expected. There are a few support vectors very far away from where the decision boundary is. Many of these far away support vectors are towards the higher negative value $CAPE_L$ and high $SUBSAT_L$. As figure 3 suggests, there are a total of 1515 support vectors, which totals to about roughly 41% of the values serving as said vectors.

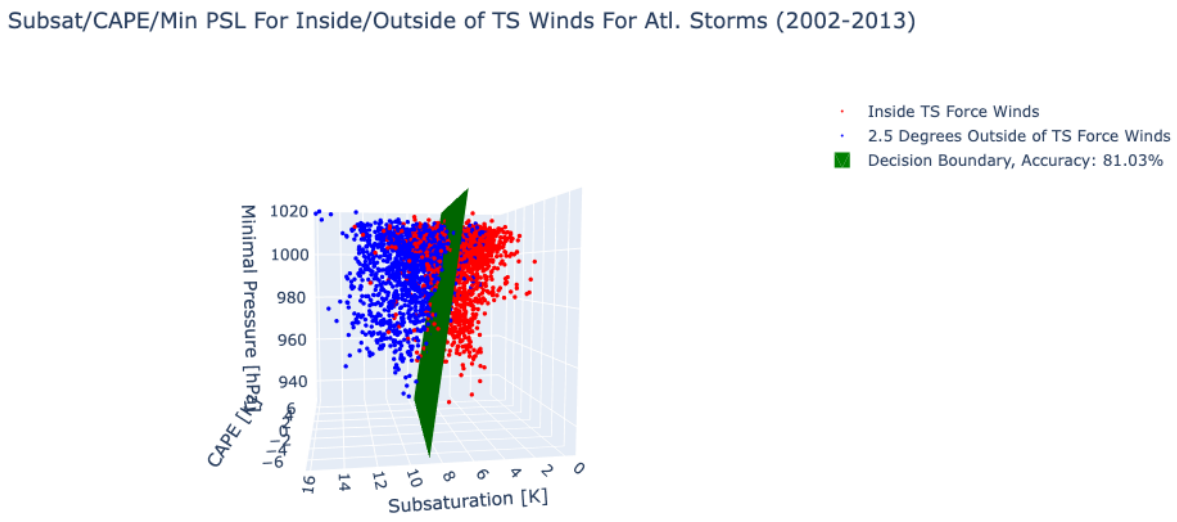

Figure 4: The data points with the decision boundary, there is some overlap at weaker intensities, but not much at higher intensities

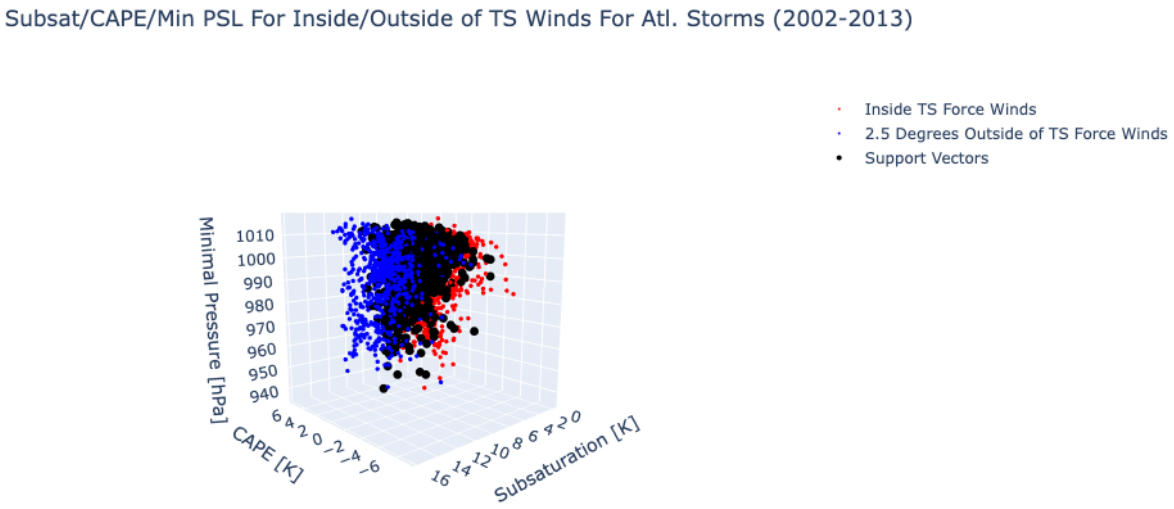

Figure 5: The data points with support vectors, most are close to where the decision boundary is, but a few are scattered far from the dividing point

Discussion

This calculated decision boundary by the SVM algorithm has worked well as seen by the relatively high accuracy. Storms with a lower pressure than what the decision boundary predicts are noted as being internal values, while storms with higher pressures than predicted are listed as external values. Due to the large number of weaker storms, I found that there is significant overlap of internal and external values, which means that no matter what, the decision boundary cannot have perfect accuracy. As the storms intensify, the decision boundary becomes more accurate. According to the Integrated Ocean Observing System, a category 1 hurricane has an average minimal pressure of 980 hPa. When I took only storms that had a lower pressure than this, accuracy goes up by about 10%. The accuracy in this case is 92.2% for all values, and 90.1% and 94.3% for inner values and outer values respectively. However, when looking at

tropical cyclones that have a minimal pressure of 980 hPa or above, the accuracy falls by about 2% for each respective category. With nearly 8 times more storms in this category, I found that the high accuracy for stronger storms does not offset the lesser accuracy for the weaker storms.

Interestingly, the decision boundary when given an unbounded z-axis has an odd polygonal shape with a maximal/minimal value far greater/lesser than the range of the data as seen in figure 6. When I took the maximal values of $CAPE_L$ and $SUBSAT_L$ and plugged them into the equation, I obtained a value of 633.4 hPa, over 300 hPa less than the minimal value of the dataset. Taking the minima the same way, I obtained a value of 1249.4 hPa, over 200 more than the maximal value of the dataset. The odd shape can be explained by the fact that not every value of $CAPE_L$ and $SUBSAT_L$ are explored upon and the boundary is only fed the values from the dataset.

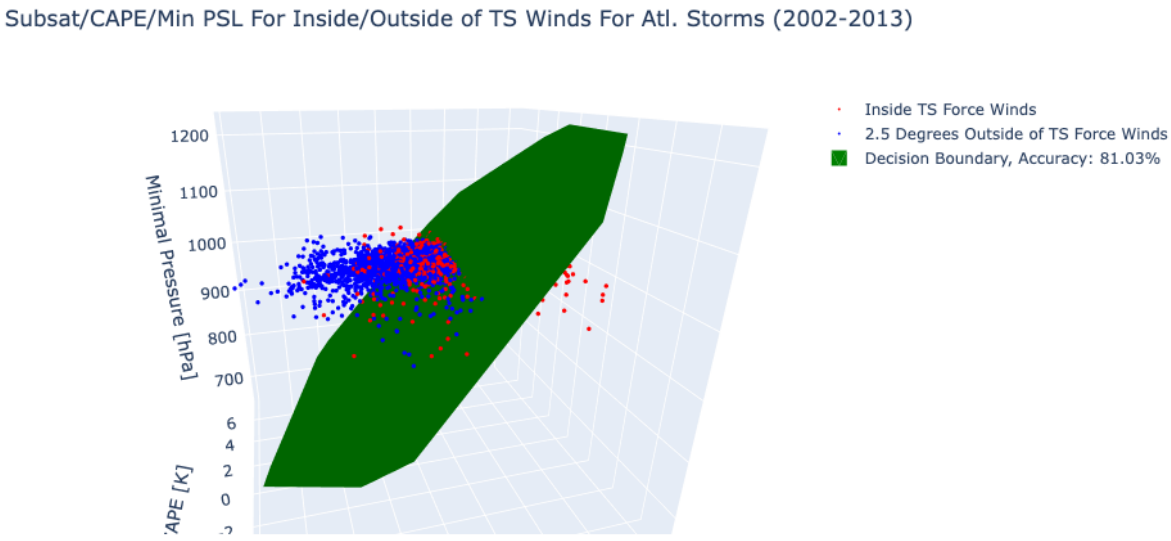

Figure 6: The full decision boundary

In addition, this also results in a large number of support vectors. With a high percentage of support vectors, this means that there might be a slight overfitting of the data. I found that with the heavy overlapping of the data, there have to be many support vectors to fit the intricacies of the overlap. In addition, the margin between the decision boundary and the support vectors is relatively small, meaning that the data is likely overfitted. Unfortunately, I found that trying to decrease the amount of support vectors brought them even closer to the decision boundary. Thus a compromise is made for the accuracy and the number of support vectors. The support vectors still clearly indicate where the decision boundary is going to be so I see it working well.

Conclusion

Oftentimes, ERA5 data is rather lacking when it comes to certain information, which is expected as reanalysis is heavily dependent on the data it pulls from. However, as I have shown here, there are ways to work around this challenge. By using $CAPE_L$, $SUBSAT_L$, and minimal pressure, I was able to successfully implement a machine learning algorithm to decide the radii of a tropical cyclone.

I collected this data by taking already obtained tropical cyclone radii by applying it to a data collecting code. I was able to do this by taking the average $CAPE_L$, $SUBSAT_L$, and pressure of inside tropical storm force winds and the outside 2.5 degrees latitude and longitude without the inner values. I only was able to take values for storms in the North Atlantic basin and below 34.75 degrees latitude due to time constraints and missing data respectively. Due to the nature of needing a classifier that could produce a potentially nonlinear equation, I chose the SVM

algorithm. With this, I received a decision boundary with an accuracy of about 81.0% where 42% of the data points served as support vectors. The accuracy of the decision boundary increased with intensity, likely due to less data points. The support vectors had very small margins, meaning it was very fit to the data.

With this, a question is raised on how well this decision boundary could fit other data points. A large margin is the goal of the SVM algorithm, as ideally this results in a versatile classifier. However, I found that this can come at the cost of accuracy in the decision boundary. Overall, I found that this dataset traded the margin size for relatively high accuracy. As a result, this decision boundary may not be the most ideal for other datasets. However, this can be tested by applying it to other datasets to see if it can work just as well. With this relationship established, it is now possible to backwards engineer this equation. It is now feasible to obtain a rough idea of what the relation between the average $CAPE_L$ and $SUBSAT_L$ are for a tropical cyclone based on the radii of the tropical cyclone. This in turn opens the door for more accurate environmental analysis for current and past tropical cyclones.

References

Ahmed, Fiaz, and J. David Neelin. "A process‐oriented diagnostic to assess precipitation‐thermodynamic relations and application to CMIP6 models." *Geophysical Research Letters*, vol. 48, no. 14, 26 June 2021, https://doi.org/10.1029/2021gl094108.

"Hurricane Glossary." *SECOORA*, IOOS, secoora.org/hurricane_glossary/. Accessed 6 Dec. 2023.

Pérez-Alarcón, Albenis, et al. “Dataset of outer tropical cyclone size from a radial wind profile.” *Data in Brief*, vol. 40, Feb. 2022, p. 107825, https://doi.org/10.1016/j.dib.2022.107825.

Ullrich, Paul A., et al. “Tempestextremes v2.1: A community framework for feature detection, tracking, and analysis in large datasets.” *Geoscientific Model Development*, vol. 14, no. 8, 13 Aug. 2021, pp. 5023–5048, https://doi.org/10.5194/gmd-14-5023-2021.