The National Weather Service (NWS) launches weather balloons (also known as radiosondes or rawinsondes) twice daily at 92 sites. Balloon launches just before 00 and 12 UTC provide measurements of temperature, dew point, wind speed, and wind direction at fixed mandatory heights, as well as at heights of significant change. These data are used by forecasters and in numerical weather prediction models to quantify and understand the state of the atmosphere. From the sounding data, a number of parameters are calculated, some empirically derived, others based on thermodynamics. The amount of energy available to air parcels in the atmosphere to convect, known as Convective Available Potential Energy (CAPE) and the energy barrier preventing convection, known as Convective Inhibition (CIN) are of great interest in potential severe weather situations. Here we present the process of implementing the seemingly simple calculations into MetPy, a community meteorological calculation and plotting package supported by Unidata. CAPE and CIN may be calculated with a number of minor corrections, use of different approximations, and different atmospheric parcels. The exact implementation used in many meteorological tools such as SharpPy, GEIMPACT, and others is often poorly documented. We attempted to implement CAPE in the most versatile way possible to provide the maximum flexibility to users.

In an attempt to understand how a typical research meteorologist would use MetPy and the Unidata suite of tools, we chose to reproduce a famous paper that compares CAPE calculations with and without the virtual temperature correction by Doswell and Rasmussen in 1994.

Doswell and Rasmussen found that the magnitude of the virtual temperature correction on CAPE (always positive) increased with increasing CAPE values, but that the relative importance of the correction decreased with increasing CAPE values. The authors suggest a standardized CAPE calculation in which the simple pseudoadiabatic profile is used, the virtual temperature correction made, and the most unstable parcel in the lowest 300 hPa of the atmosphere used.

Our results are similar (right), but appear to contain many more soundings than the original paper. The original data used were not presented in tabular form, leaving the differences unexplained. In general, trends and magnitudes are preserved. Further comparison with other sounding analysis tools is in progress to further cross-compare results from different analysis tools.

### Design Challenges

**Assumptions Object**

Most of the complexity in the calculation lies in the air parcel path determination. We have prototyped an assumptions object that holds assumptions about the calculation to be made. Future plans include adding assumptions onto other calculations and building a solver that automatically determines the best and valid path from one set of parameters to another derived set of parameters.

This assumptions object avoids putting the data into an arbitrary “sounding object” or similar. This enables the user to easily work with their own data of any format, as long as it can be read into a Numpy array without needing to learn a custom MetPy data model.

```python
def __init__(self):
    self.use_virtual_temperature = True
    self.most_unstable = True
    # other attributes
```

**CAPE/CIN Wrappers**

The core CAPE/CIN calculation is a simple integral, but depends upon calculation of a number of thermodynamically significant levels and the starting parcel and associated parcel path. This core calculation is available to the user wanting to do a complex or custom calculation, but we have provided wrappers for the most common types of CAPE/CIN calculated by research and operational meteorologists.

- **Surface Based CAPE**
- **Mixed Layer CAPE**
- **Most Unstable CAPE**

Calculations that require parcel path information can either take a parcel path provided by the user (in a research situation with non-standard paths for example), calculate a default path, or calculate a path given an assumptions object. This allows easy standard use cases, but maximum flexibility for advanced users.

### Conclusions

API/ Library Design

During the course of developing this frequently requested feature, the MetPy team was able to crystallize some ideas about where complexity in calculations should lie in the package and how assumptions/different solutions to the same problem could be addressed. The team intends to continue adding a variety of meteorological calculations to the package as requested by the community and implement a solver to help automatically determine the calculations required to get from the input variables to the desired quantity.

- Complexity should be pushed to the lowest, most fundamental level.
- Assumptions should be integrated into a single method to calculate the desired quantity.
- Methods should accept arguments in any physically sensible unit and handle conversions internally with the appropriate assumption.

Reproducibility

In the course of reproducing the results of a classic paper, we encountered several challenges. The largest of which was finding the same data used as no table of data was included. The numerical results of the study were not included either, and poorly scanned PDFs of the paper prevented accurate reverse engineering of the data from the graphs. The vast majority of the same data were likely found, but this poses a challenge for true reproducibility.

- Finding data without data tables is difficult.
- No numerical results were given in the paper.
- No code was included with the paper.

Our hope is that with the thoroughly documented and citable MetPy and tools like GitHub LFS, that such a study done now could be made easily reproducible.