1.402

SHARPPY: FUELING THE PYTHON CULT

Kelton T. Halbert*, W. Greg Blumberg*, Patrick T. Marsh**
University of Oklahoma School of Meteorology, Norman, Oklahoma*
Storm Prediction Center, Norman Oklahoma**

1. INTRODUCTION

In 2011, work began to remedy a need for an open source, comprehensive, cross-platform, upper air sounding analysis program that could meet both the operational and research standards of the atmospheric science community. This work manifested itself in the form of SHARPpy, a Python-based rewrite of the Storm Prediction Center's (SPC) Skew-T and Hodograph Analysis Research Program (SHARP) (Hart et al. 1999) led by Patrick Marsh and John Hart. Due to wide adoption of the Python programming language in the atmospheric science community, in addition to its open-source and cross platform nature, this rewrite was done in Python. An early alpha version of this program was released at the American Meteorological Society's Annual Meeting in January 2012 (Marsh and Hart 2012).

Since then, two new lead developers have adopted the SHARPpy project and continued its development. As a result of our developments and the collaboration of several SPC employees, the package has now reached a status comparable to SHARP and contains several new analysis routines previously internal to the SPC. With respect to its original release, this updated package is released with a beta status at this meeting.

2. TIMELINE OF DEVELOPMENTS

The original alpha version of SHARPpy in 2011 contained various routines that could perform parcel lifts to calculate thermodynamic indices (i.e. CAPE, CIN, LCL) as well as routines to analyze kinematic indices (i.e. shear, storm-relative helicity, mean-wind) from soundings. This original release only utilized packages default to the Python interpreter. However, additional developments utilized third party, open-source packages such as Numpy (for faster array operations) and PySide (plotting for the Qt framework).

When new development of SHARPpy began in early 2014, the authors began working to 1.) Expand and stabilize the available set of analysis routines and 2.) Develop the plotting ability of the package through the development of a graphical user interface (GUI) similar to SHARP. An example of the SPC SHARP GUI for the Norman, Oklahoma sounding on April 27th, 1991 is shown in Figure 1.

Several new features added to SHARPpy include the Storm Slinky, the Possible Hazard Type, and the Sounding Analogue Retrieval System (SARS) which all were previously internal to the SPC. The Storm Slinky is a simplified 3D parcel trajectory in which a parcel that is lifted to its level of free convection (LFC) and then given a 5 m/s upward nudge. Afterwards, the parcel is accelerated upwards using its own buoyancy and the parcel is advected horizontally by the storm-relative winds until the parcel reaches its equilibrium level (EL). The Possible Hazard Type function is a fuzzy-logic decision tree that attempts to identify the likely hazard associated a given sounding and is loosely based off of the forecasting experience of SPC forecasters. The last new key function, the SARS routine, is a sounding matching program. SARS matches computed indices (i.e. CAPE, storm-relative helicity) from an input sounding to indices computed from a database of past tornado and hail proximity soundings. From the matched soundings, SARS computes a probability of tornadoes or severe hail and also provides a list of the closest matches (Jewell 2010).

3. COLLABORATION WITH NWS/SPC

Additional porting of SHARP routines into SHARPpy was difficult without outside assistance as many SHARP routines are internal to the SPC. As a result, the authors began contacting employees at the SPC in order to gain information about these additional routines. Employees were enthusiastic about assisting the project. For example, Ryan Jewell and Rich Thompson provided information critical to implementing the SARS routines. In addition, Bryan Smith and Rich Thompson provided original SHARP code that expanded the available set of insets that could be displayed in the GUI. Through email and visits to SPC, this collaborative relationship ensured the further development and testing of SHARPpy.

Benefits of this collaboration were not one-sided. Optimizations to the SHARPpy code base were shared with SPC employees, which were then implemented in SPC's version of SHARP. These implementations led to a 30% speed increase in the SHARP program. In addition, several SPC employees expressed interest in having a personal copy of SHARPpy, due to the fact that the routines behind it have decades of testing and verification. SPC employees have also given suggestions for additional GUI functionality that they would like, and are currently under development.

Additional routines were added to SHARPpy that were not included in the SHARP program. In order to provide climatological context for each sounding passed to the program, SHARPpy includes a precipitable water vapor climatology database provided by Matt Bunkers from the National Weather Service (NWS) Weather Forecast Office (WFO) in Rapid City, South Dakota

(UNR). This database provides climatological means and variance of precipitable water vapor (PW) at different sounding sites around the United States. The index displayed on the GUI will change colors based on where the sounding lies within the distribution, in addition to triggering the Possible Hazard Type window for flash flood risks.

4. VERIFYING THE ROUTINES

Verification of the routines contained within SHARPpy came from a combination of comparing the SHARPpy output with the SPC online soundings, as well through collaborations with SPC employees. SPC employees assisted by providing data from their historical sounding archive and screenshots from SHARP to compare the routines not available online. Figure 1 illustrates the GUI output from each program used in such a comparison. In this case, most of the program output is the same or similar, but there are several discrepancies between the coloring and values of the indices. In addition, the SHARPpy SARS Supercell analogues do match to many of the same analogs that the SHARP version does, but it misses a few, which could be a consequence of slight differences in the indices computed. Overall, much of the program is the same.

In order to evaluate the SHARPpy code against SPC's SHARP, text data archived from the SPC online soundings website was run through the SHARPpy routines. This text data spanned the Norman, Oklahoma NWS soundings between February 18th, 2014 12 UTC and October 18th, 2014 00 UTC (n=485). Each text file contains both the raw sounding data run through SPC's SHARP and a subset of the sounding's corresponding indices generated by SHARP.

A subset of the errors between SHARPpy and SHARP for different indices is shown in Figure 2. The indices chosen are indices essential to the core functionality of the program. The top panel of Figure 2 shows the validity of the SHARPpy most unstable (MU), surface-based (SB), and 100 mb mixed-layer (ML) thermodynamic indices. Most convective available potential energy (CAPE) and convective inhibition (CIN) absolute errors are within 1-3 J/kg and most significant levels (lifted condensation level; LCL, level of free convection; LFC, equilibrium level; EL) have little error. The MUEL and MULCL have larger errors, which may be due to slight precision differences causing different most-unstable parcels to be found by the SHARPpy algorithms.

The second panel of Figure 2 shows kinematic and composite indices. This comparison includes PW, the 700-500 mb lapse rate (LR75), surface to 1 km shear (SHR01), and the surface to 6 km shear (SHR06). The largest errors come from the routines involving the storm relative helicity (0-1 storm-relative helicity; SRH1, 0-3 storm-relative helicity; SRH3, and effective storm relative-helicity; EFFSRH) as these helicity calculations are dependent on how the storm motion vector is calculated. Composite indices such as the significant tornado parameter STP (fixed-layer; STPF, with CIN; STPC), the Severe Hail Parameter (SHIP), and supercell composite indices (SCP) are also shown to have minimal error.

5. FUTURE WORK

Additional evaluations of SHARPpy are underway, as some discrepancies do exist, as is seen in Figures 1 and 2. These differences are minimal and do not seem to affect the overall functionality of the program. However, as of this AMS meeting, SPC and NWS employees listed above have begun evaluating SHARPPy, and we expect additional improvements will be made to the package as their comments are made. This package is downloadable at: https://github.com/sharppy

6. ACKNOWLEDGEMENTS

The authors wish to acknowledge and thank the multitude of people who were critical to the further development of this program. Without their contributions, SHARPpy would not as high of a quality program as it is today. Such people include: Rich Thompson (SPC), Ryan Jewell (SPC), Bryan Smith (SPC), Matthew Bunkers (NWS Rapid City, SD), Tim Supinie (Oklahoma School of Meteorology), and David John Gagne (Oklahoma School of Meteorology). The authors also thank Dave Turner (National Severe Storms Laboratory) for archiving the SPC online soundings that were used in verifying the SHARPpy algorithms.

7. REFERENCES

Jewell, R., 2010: The Sounding Analog Retrieval System (SARS). *Preprints*, 25th Conf. Severe Local Storms, Denver CO.

Hart, J. A., J. Whistler, R. Lindsay, and M. Kay, 1999: NSHARP, version 3.10. Storm Prediction Center, National Centers for Environmental Prediction, Norman, OK, 33 pp.

Marsh, P. T. and J.A. Hart, 2012: SHARPPY: A Python Implementation of the Skew-T/Hodograph Analysis and Research Program, *2nd Symposium on Advances in Modeling and Analysis Using Python*. New Orleans, LA, Amer. Meteor. Soc.

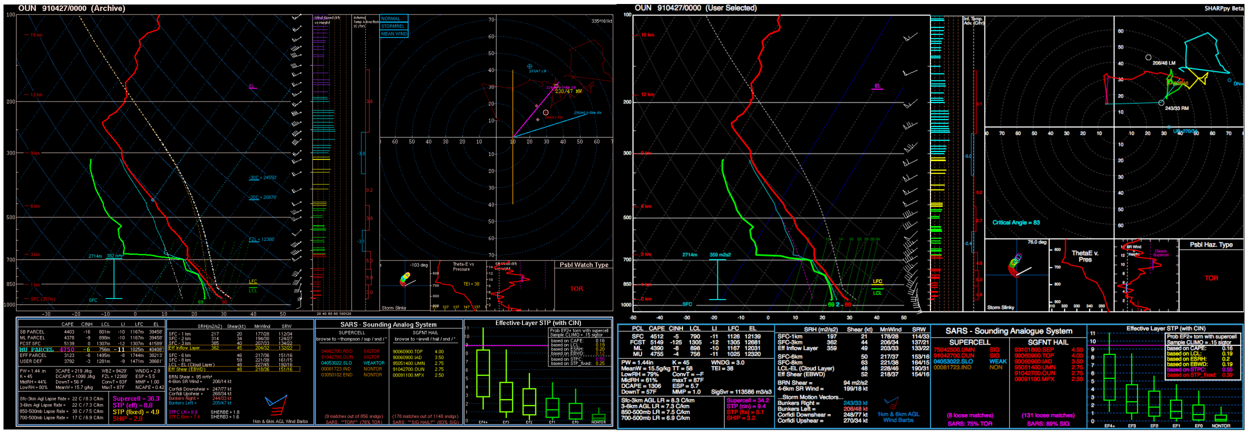

Figure 1. The SHARP GUI (left) and SHARPpy GUI (right) for the Norman, Oklahoma April, 27th 1991 00 UTC sounding with an example boundary motion plotted.

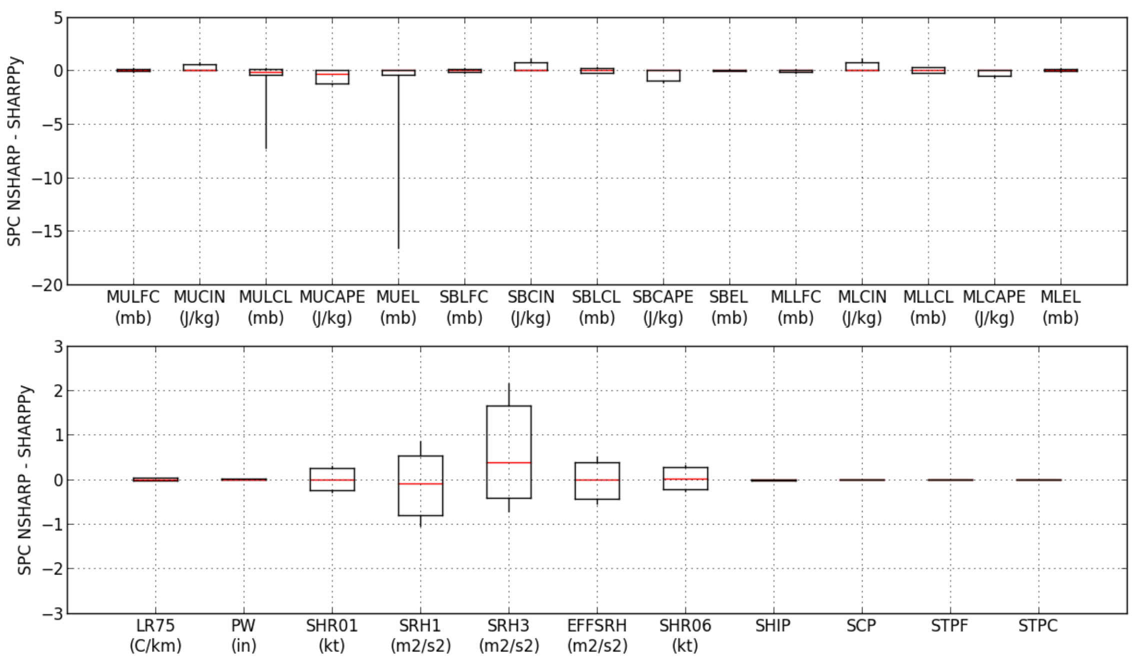

Figure 2. Box and whisker plots to show the distribution of errors between the SHARPPy-generated indices and those generated by SPC's SHARP program within the verification dataset. The whiskers of the box and whisker plots indicate the 20th to 80th percentiles.