**NASA DEVELOP National Program**



Langley Research Center

*Spring 2015*

CALIPSO: Global Health & Air Quality

*Creating a Tool with the ability to Classify Aerosols within CALIPSO Data to Help Improve Future Research and Decision Making*

**Technical Report** 

Final Draft - March 27, 2015

Jordan Vaa (Project Lead)

Courtney Duquette

Ashna Aggarwal

Dr. Kenton Ross, National (Science Advisor)

Nathan Owen, Langley DEVELOP (Center Lead)

Chad Smith, Langley DEVELOP (Assistant Center Lead)

# I. Abstract

The Cloud-Aerosol Lidar and Infrared Pathfinder Satellite Observations (CALIPSO) mission has been providing researchers with information about the global distribution of aerosols and clouds since 2006. Aboard the CALIPSO satellite is the Cloud-Aerosol Lidar with Orthogonal Polarization (CALIOP), which sends laser pulses of 532 nm and 1064 nm into the Earth’s atmosphere. By measuring backscatter, researchers are able to map the distribution of aerosols (such as pollutants, dust, and smoke) and clouds. However, it remains difficult to track specific objects as they progress through the environment, especially as some types of aerosols are more difficult to identify than others. To solve this issue, the Langley DEVELOP team created a tool that will allow researchers to identify, select, and categorize aerosol objects. The objects are then exported to an easily accessible database. This method will allow researchers to follow key objects as they move through time and space. The CALIPSO science team will use this tool to identify smoke plumes and explore their compositions. The composition of smoke plumes varies significantly depending on the fuel type. Monitoring how these compositions change with time will help researchers understand the impact of smoke on air quality downstream of a source fire.

**Keywords**

CALIPSO, CALIOP, Lidar, Atmospheric Aerosols, Air Quality, Python

# II. Introduction

**Background Information**

On April 20, 2006 the Cloud-Aerosol Lidar Infrared Pathfinder Satellite Observation (CALIPSO), was launched from Vandenberg Air Force base in California (Winker et al, 2007). CALIPSO was the first satellite-based lidar systems in orbit, following the Lidar In-space Technology Experiments (LITE). Before LITE, lidar observations had only been made with ground based installations and a small number of airplane based system and lidar in space was only theoretical. Whereas LITE was a temporary test of the viability of using lidar in space to observe the atmosphere, CALIPSO was a semi-permanent system (Winker et al, 1996). Its purpose was to observe clouds and aerosols in the atmosphere, letting researchers track and understand their effects. Due to the relatively new quality of lidar data, formatting data and accessibility can be enhanced in many ways (Winker et al., 2013). CALIPSO’s instrument package was comprised principally of the Cloud-Aerosol Lidar with Orthogonal Polarization (CALIOP), along with a Wide Field Camera (WFC) and an Imaging Infrared Radiometer (IIR) (Winker et al, 2007).

CALIOP sends pulses of 532nm and 1064nm into the atmosphere. By utilizing two receiver channels for the 532 nm wavelength and a total attenuated backscatter at the 1064 nm wavelength CALIPSO had the ability to identify several different types of aerosols, both anthropogenic (biomass burning and urban pollution) and biogenic ( desert dust, natural, sea salt, and volcanic) (Omar et al, 2009). With CALIPSO data, scientists were able to further chemically classify the aerosols (black carbon, organic carbons, sulfates, etc.) (Omar et al, 2009). CALIPSO determined the type based on an algorithm consisting of both backscatter measurements and volume depolarization ratio measurements (Omar et al, 2009). The algorithm used at least five input parameters to make this determination: altitude, integrated attenuated backscatter, location, surface type, and volume depolarization ratio (Omar et al, 2009). Although this algorithm generated the aerosol subtype, it could not discriminate between the sources of the aerosol (Omar et al, 2009).

One aerosol that required such attention is smoke. Because of its various compositions and change in structure over time, it was difficult to track smoke plumes as they moved away from source fires. Generally, smoke particles began as chains of nanometer sized particles. However, “with aging these chains tend to collapse into irregular aggregates of small particles,” (Sun et al, 2013). These chains were more difficult to detect with the algorithms listed above. Our proposed tool will help researchers document smoke plume objects with a human eye, and track them as they move across the environment.

Before the DEVELOP spring 2015 term, the CALIPSO science team viewed this data using an IDL program that was developed in early 2007 for the team. Although this program has undergone updates and improvements over the years, due to the proprietary nature of IDL it has remained difficult for researchers outside of NASA to use. A program, called the CloudSat and CALIPSO plotting tool (ccplot), was built in 2010 that mimics the IDL program used by the CALIPSO science team. This tool was developed for a simple method of visualizing CloudSat and CALIPSO data (Kuma, 2010). Ccplot allows a user to specify HDF files (specifically the CloudSAT and CALIPSO data files) and create an output based on those files. This output, generally an image, provides visualization of the data provided, such as depolarization or aerosol types detected by the lidar. However, it cannot be manipulated by the user. Additionally, it is a command line program, and thus requires several Python libraries and modules to install correctly (Kuma, 2010). It is the goal of this project to update ccplot so that it is easier to install and use, as well as add the ability to manipulate the images created by it, such as selecting elements.

**Previous Studies**

In the recent years, there has been a number of algorithms developed to automatically discern cloud and aerosol layers. For example, the selective, iterated boundary location (SIBYL) algorithm scans the backscatter signals through multiple iterations and averages the signal depending on the background noise (Vaughan et al, 2009). Then, additional algorithms such as the Scene Classification Algorithm (SCA) attempt to identify what type of aerosol or cloud the SIBYL detected. The Hybrid Extinction Retrieval Algorithm (HERA) uses known lidar ratios and scattering ratios to further create backscatter profiles for clouds and aerosols (Winker et al, 2006). There is an existing tool used to analyze spectrometer imaging by combining high spectral resolution and spatial data presentation. The Spectral Image Processing System (SIPS) is an integrated system and user interface developed in IDL that allowed analysis, utilities for formatting, and visualizing data sets from AVIRIS, GERIS, and Eos HIRIS (Kruse et. al., 1993). The user interface of this tool had the ability to zoom and edit the image created based off the images in addition to saving the image in a BSQ format (Kruse et. al., 1993). In addition to the built in User Interface, SIPS could export any data to a GIS software in order to allow the user to perform further analysis (Kruse et. al., 1993).

**Objective**

The goal of this project was to replace current IDL scripts with a tool in the open source Python programming environment for a wider distribution base. The project improved upon a pre-existing IDL tool. The previous tool allows a user to upload an HDF file containing L1 and L2 data products from CALIPSO, and visualize these objects as images on a grid with axes of time and altitude. The user was then be able to select cloud and aerosol objects of interests by freehand in order to classify the data. The tool created in this project was designed to maintain those functionalities while also introducing new functionalities. Selected objects can be exported into a web-based database, supporting a consolidation of all prerequisite code and tools. Previously, using the older code and tools it was challenging to collect and run the code due to the disparate nature of its component modules. Moreover, the project aimed to simplify the selection of objects within CALIOP data, specifically smoke plumes. These plumes have a significant impact on air quality in a region. However, smoke is difficult to track over time because of its various possible compositions and similarity to clouds. With an organized, accessible database, researchers will be able to better recognize the impact of smoke as it travels further from source fires.

**Area of Study**

This project had a global area of study, as the tool could be used with any CALIPSO data.

**Period of Study**

2006-Present

**National Application Addressed**

This project addressed the health and air quality application area. It would allow researchers to better document the life-cycles and effects of various atmospheric aerosols throughout the globe which affect the overall quality of the air.

**Project Partners**

Our partners included the NASA CALIPSO science team, with the point of contact being Dr. Charles Trepte and Dr. Amber Soja. Dr. Trepte guided effort towards the creation of a location for organized CALIOP data. Additionally, Dr. Amber Soja would like to see the database assist the effort of detecting smoke plumes as they travel through the environment.

# III. Methodology

**Data Used**

CALIPSO provided 6 L1 and 10 L2 data products for roughly 6 hour spans. These data projects were provided in the Hierarchical Data Format (HDF) which is used for storing large scientific data samples. HDF requires specialized software to open, dependent on the source of the data and the intended use (HDF Group). This is compounded by the fact that CALIPSO data is meant to viewed in a graphical format, showing the “curtain view” of the lidar reflectance values. Graphical data representations are more complicated than a simple chart of numbers. The CALIPSO science team uses a command-line program written in interactive data language (IDL) known as PDFreader which allows them to view the data and select objects within it for further consideration as well as active manipulation. However, IDL is a proprietary language of Research Systems Inc. (RSI) and NASA, which makes it difficult for users to get access to it. To this end the current project is written in Python, which is an open source language, in contrast to proprietary languages such as IDL and MatLab. Ccplot was released under a two part Berkeley Software Distribution license, meaning it may be used and integrated into other products, so long as the original license is retained (Kuma, 2010). This means that any tool that includes ccplot must also include CCPLOT’s BSD. Over the five years, Python has been used more frequently in the atmospheric sciences due to the added features of array handling and other data structures (Lin, 2012). Python also has a clear and natural syntax that is easy to read by resembling pseudo-code. Finally, since Python is an open-source language, the users have to ability to use or create their own customized packages (Lin, 2012). Since Python can utilize several different packages with a single interpreted environment, the code is more concise and flexible (Lin, 2012).

**Project Process**

The project team has created a graphical user interface (GUI) to aid in the plotting of CALIPSO data products. This will increase ease of use, as a GUI is more user friendly than a command line program implemented for data analysis. The project was written in Python 2.7, using the Python modules TKinter, PILLOW, and ccplot. The TkInter module is particularly useful for building interfaces. PILLOW is an extension of the Python Image Library (PIL), which imports and processes images for display. Ccplot, is a module built in Python exclusively to plot CALIPSO and CloudSAT data products.

In the CALIPSO\_Visualization Tool package, users are provided with a comprehensive “READ\_ME” text file that explains the overall structure of the script. Information includes required imports, class structure, global variables, and functions. In addition to the technical information, there is also a list of things that the team feels still needs to be done. In general, the class “CALIPSO” uses the attributes root, file, and imagefilename to load the desired Level 1 CALIPSO data from .hdf file format and output it as a .png image. The major Tkinter widgets implemented in this program are Canvas (used to display the main graph), PanedWindow (to hold the top and bottom portions of the interface), Frame (to group the other widgets together), and the x and y scrollbars (attached to the Canvas) which are only active when the zoom functions are active.

Then the major portions of the GUI are separated into various class methods. These include methods allocated to three portions of the interface. Before any methods can be called, an instance of the class “CALIPSO” must be created by passing in a Tkinter screen. Then the first method that must be called is the “setUpWindow()”. This method adds the title to the class’s root Tkinter screen and ensures that the screen will appear in the center of the users screen. Once the root window is set up, the menu bar is added to the screen using the “setUpMenu()” method. This method instantiates a new instance of a Menu and then adds the two separate menus that are being used called “File” and “Help”. Within both of those submenus different commands are added; “Import File”, “Export File, “Save”, “Save As”, and “Exit”, to the file menu and “Tutorial” and “About” to the help menu. Some of these have yet to be implemented. The “Import File” uses tkFileDialog to communicate with the user’s directory. The method call “dlg=tkFileDialog.open()” opens the selected file, which is then assigned to the variable “fl” using “fl= dlg.show().” The “fl” variable is then assigned to the class variable “self.file” to be used freely throughout the code.

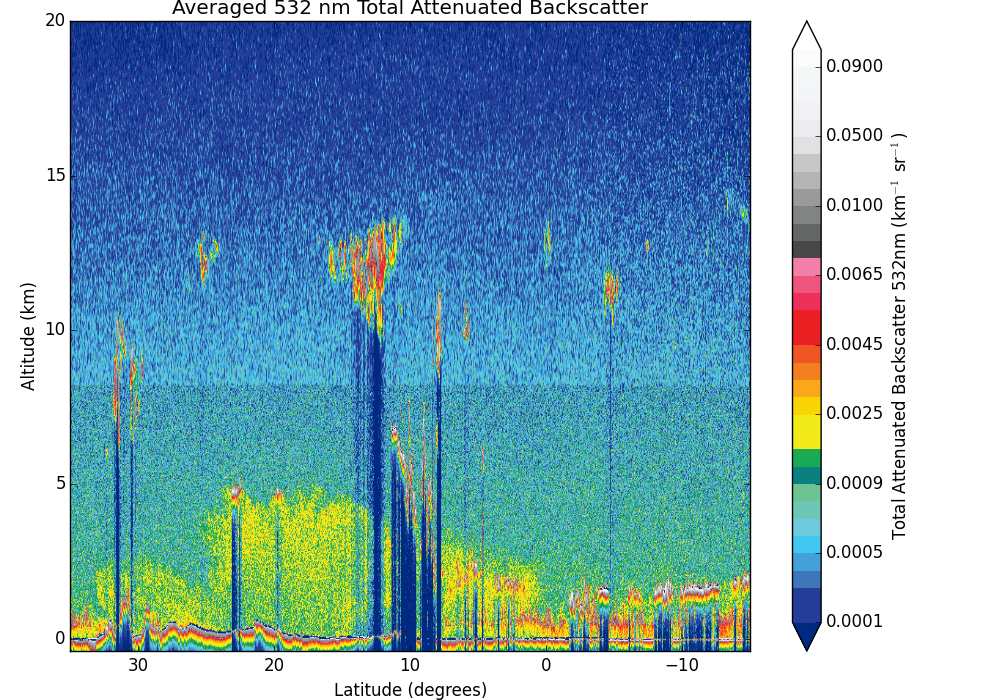
Finally, “setUpMainScreen()” is called. This method calls two other methods, “topPanedWindow()” and “selPlot(0)”. The “selPlot(0)” will set the default image to a stored image within the package. The “topPanedWindow()” is the method that sets up the different buttons displayed on the top on the screen such as “Browse”, “Zoom In”, “Zoom Out”, “Reset”, and “Plot Type”. The “Browse” is a different way to import a file so that button simply recalls the method used in the menu bar. The “Plot Type” uses a menu of several Tkinter radio-buttons to assign a unique integer value to “plotType” to indicate where the user has selected either the Backscatter, Depolarization, or VFM plot. When the user selects the plot type of interest, a lambda function is called which allows the button to activate a method, “selPlot(plotType)”, to pass in a variable. Within in “selPlot(plotType)”, an “if” statement will decide which image the user intends to display and execute the necessary script. Those scripts return a “.png” file which is then sized to the correct dimensions and converted to a “PhotoImage” which can then be displayed on the main canvas. The “Zoom In” button scales the size of the PhotoImage up based on the number of times the button is clicked. The rescaled image then replaces the standard-sized image by adding the image to the canvas. In addition to the image change, a scroll bar is activated on the screen to allow the user to observe the whole image at the new size. “Zoom Out” works similarly and checks to see when the image has returned to the original size.

# IV. Results & Discussion

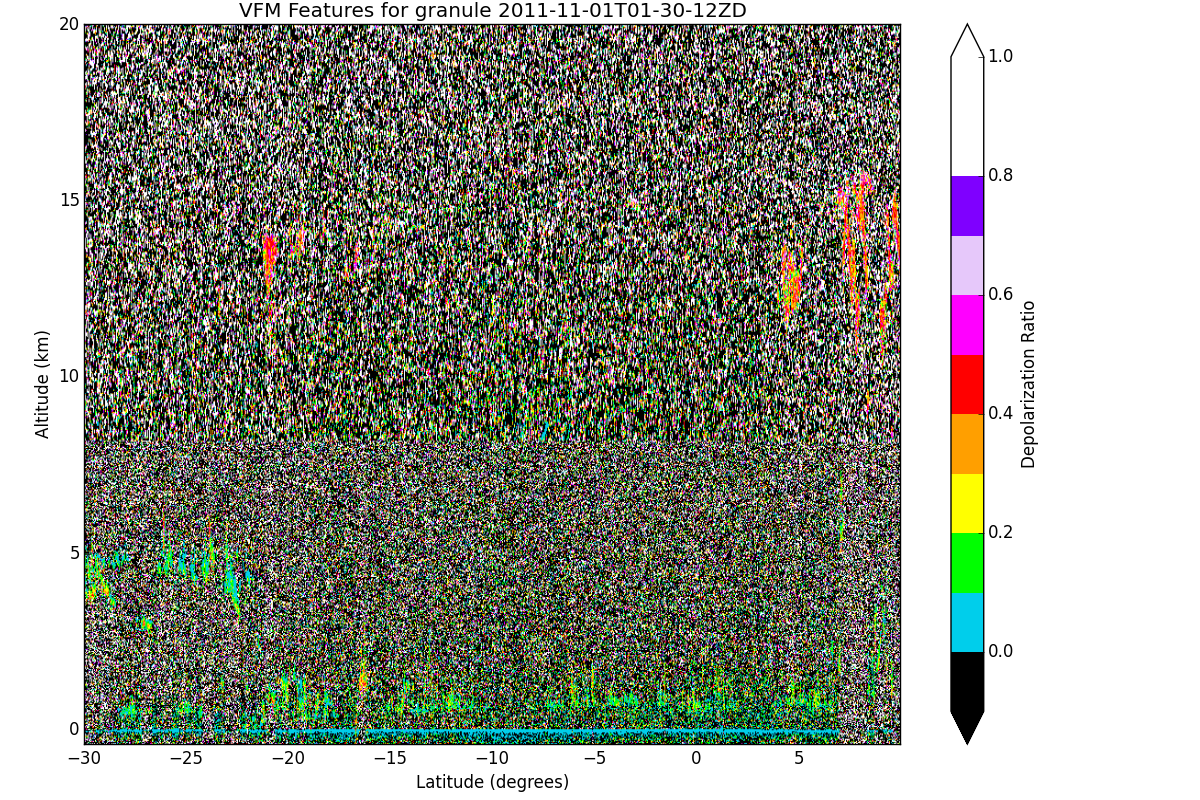
**Analysis of Results**

As a first term project, the CALIPSO team has progressed in two major ways. The first includes consolidating ccplot with Python. The team did this by providing the directory of Level 1 CALIPSO data directly into a set of provided scripts. These codes import matplotlib, ccplot, and numpy to scale and plot aerosol data from 0 to 20 km in altitude, and -15 to 35 km in latitude at a given data and time noted in the file names. With these programs, users have the ability to execute ccplot directly from a Python script. This gives programmers the option to generate graphs based on Level 1 CALIPSO data including the Total Attenuated Backscatter and Depolarization ratio plots without using the command line program. This has added a level of user friendliness to these scripts.

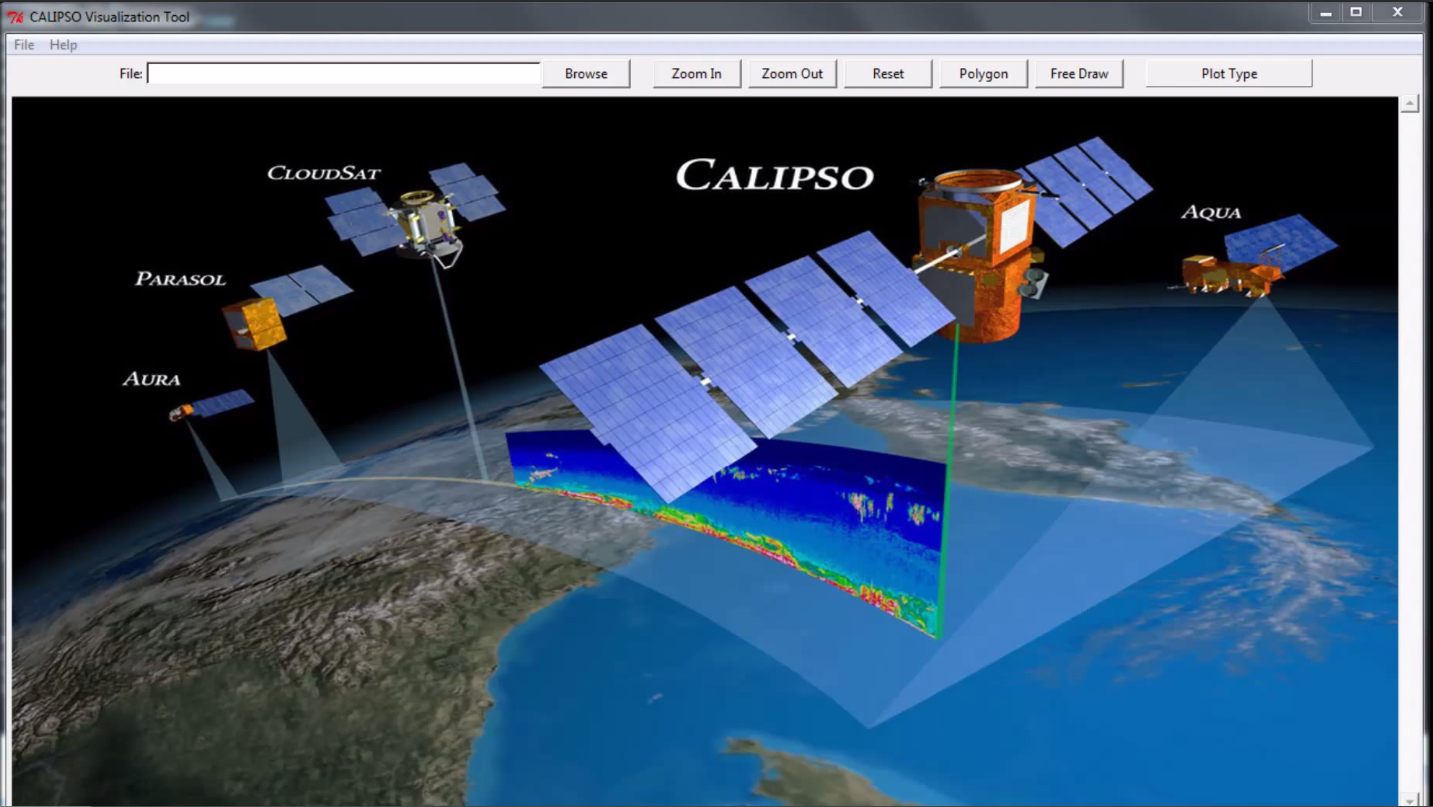
The Backscatter image (below) provides a color code of the Total Attenuated Backscatter coefficients of aerosols. This coefficient represents the intensity of the scattered signal received in relation to the intensity of an initial signal sent at 532 nm. This plot type is useful for identifying particle sizes and shapes.



The Depolarization image (below) provides a color code of the Depolarization Ratio of aerosols. This ratio, on a range of 0 to 1, represents the direction of the scattered signal in relation to the initial signal. If the scattered signal is perpendicular to the initial signal (completely scattered) the depolarization ratio is 1. If the scattered signal is parallel to the initial signal (no scatter) the depolarization ratio is 0.



For the second major accomplishment, the team created a GUI using Tkinter, the standard GUI library for Python. From this interface users can use the browse button, which is located at the top of the screen, to navigate through their directories to find the desired Level 1 CALIPSO data file. Next, either a Total Attenuated Backscatter or Depolarization ratio plot based on the uploaded data can be generated and displayed in the GUI window. If no file has been uploaded, a screen will pop up to inform the users a file is needed.



The “zoom in” and “zoom out” options let a user zoom in on the image, allowing the user to focus on objects of interest. The Reset button allows users to return to the home screen and remove the file from the program. For more details, see the “Methodology” section.

**Errors and Uncertainty**

A problem confronted by the team was that one of the plot types, the Vertical Feature Mask or VFM, previously translated into a Python script, would not run on the computers used by the team. Running the code produced a memory error and a blank screen when the empty plot was forced to display the canvas. The team discussed the error with the scripts’ author, but no resolution to the issue could be found as of yet. Additionally, four extra gigabytes of RAM were added to one of the team’s computers in hopes that it would address the memory error. However, the issue was not resolved. The author was able to display the plot on a Macbook Pro, which implies that the error is a result of the Windows OS environment used by the team.

**Future Work**

In upcoming terms, the unimplemented buttons and features in the User Interface must be completed. These include the two drawing buttons located on the top of the main screen. The “Polygon” button should allow users to select a designated shape, either a circle or rectangle, and outline a feature they wish to investigate. The shape should be scalable as well as able to be placed anywhere on the plot. The “Free Draw” button will allow the user to draw free-hand shapes on the screen, outlining the feature that they are interested in. This will be an update from the original IDL tool that only features the “Polygon” function. Once the image is outlined, the user should be prompted to classify the image with various information. Additionally, several options located in the menu bars are currently unimplemented. Under the file menu, the “Export Image” function needs to be implemented. It should scan the image for an outlined object and isolate it from the main file. The object should then be added to a database created from many like objects. This database should also be designed and implemented in a future project. Also under the file menu is the “Save” and “Save As” functions. These functions will give users the ability to both save any changes to a data file that they are working on and change the current directory and name of the image, respectively. The final function on the menu bar that needs to be implemented is “Tutorial” under the help menu. This function should link users to any documentation or manuals that will assist with the functionality of the tool.

Currently, there are many types of plots for CALIPSO data. These plots have not yet properly coded into python, but as they are, they should be added to the tool. This process should be made in such a way that a user with little to no Python experience is able to add modules that integrate additional plot types. This will require setting the tool up in such a way as to add these new functions.

Finally, the tool must be packaged into an executable program and installer, which will allow users to easily transfer the program from one machine to the next. The installation process will package the supporting Python modules, including ccplot and Basemap, preventing the user from having to search for them independently. This will simplify the initial set-up process for the tool greatly.

# V. Acknowledgments

* Dr. Kenton Ross (NASA DEVELOP National Program): Science Advisor
* Dr. Charles Trepte (NASA CALIPSO Science Team): Partner POC
* Dr. Amber Soja (National Institute of Aerospace): Partner POC
* Brian Magill (NASA CALIPSO Science Team)
* Nathan Owen (Langley DEVELOP Center Lead)
* Chad Smith (Langley DEVELOP Assistant Center Lead)

This material is based upon work supported by NASA through contract NNL11AA00B and cooperative agreement NNX14AB60A.

# VI. References

The HDF Group - Information, Support, and Software. (n.d.). Retrieved March 23, 2015, from http://www.hdfgroup.org/

Kruse, F., Lefkoff, A., Boardman, J., Heidebrecht, K., Shapiro, A., Barloon, P., & Goetz, A. (1993). The spectral image processing system (sips): Interactive visualization and analysis of imaging spectrometer data. *Remote Sensing of Environment*, 44(2-3). 145-163. doi: [10.1016/0034-4257(93)90013-N](http://dx.doi.org/10.1016/0034-4257(93)90013-N)

Kuma, P. (2010). Visualising Data from CloudSat and CALIPSO Satellites. Bachelors Thesis. *Bratislava Faculty of Mathematics, Physics, and Informatics, Comenius University*.

Lin, W. (2012): Why Python Is the Next Wave in Earth Sciences Computing. *Bull. Amer. Meteor. Soc.*, 93, 1823–1824. doi: http://dx.doi.org/10.1175/BAMS-D-12-00148.1

Omar, A., Winker, D., Vaughan, M., Hu, Y., Trepte, C., Ferrare, R., Lee, K., Hostetler, C., Kittaka, C., Rogers, R., Kuehn, R., & Liu, Z, (2009) The CALIPSO Automated Aerosol Classification and Lidar Ratio Selection Algorithm. *J. Atmos. Oceanic Technol.*,*26*(10). doi: <http://dx.doi.org/10.1175/2009JTECHA1231.1>

Setvák, M., Bedka, K., Lindsey, D., Sokol, A., Charvát, Z., Šťástka, J., & Wang, P. (2013). A-Train observations of deep convective storm tops. *Atmospheric Research,* *123*, 229-248. doi: [10.1016/j.atmosres.2012.06.020](http://dx.doi.org/10.1016/j.atmosres.2012.06.020)

Sun, W., Liu, Z., Videen, G., Fu, Q., Muinonen, K., Winker, D., Lukashin, C., Jin, Z., Lin, B., Huang, J., (201) For the depolarization of linearly polarized light by smoke particles. *Journal of Quantitative Spectroscopy and Radiative Transfer.* doi: [10.1016/j.jqsrt.2012.03.031](http://dx.doi.org/10.1016/j.jqsrt.2012.03.031)

Vaughan, M., Powell, K., Winker, D., Hostetler, C., Kuehn, R., Hunt, W.... Mcgill, M. (2009). Fully Automated Detection of Cloud and Aerosol Layers in the CALIPSO Lidar Measurements. *Journal of Atmospheric and Oceanic Technology, 26*(10), 2034-2050. doi: <http://dx.doi.org/10.1175/2009JTECHA1228.1>

Winker, D., Couch, R., & Mccormick, M. (1996). An overview of LITE: NASA's Lidar In-space Technology Experiment.*Proceedings of the IEEE,* *84*(2), 164-180. doi: [10.1109/5.482227](http://dx.doi.org/10.1109/5.482227)

Winker, D., Hostetler, C., Vaughan, M., Omar, A., (2006) CALIOP Algorithm Theoretical Basis Document. *PC-SCI-202 Part 1 Release 2.0*

Winker, D., Hunt, W., & Mcgill, M. (2007). Initial performance assessment of CALIOP.*Geophysical Research Letters,* *34*(19). doi: 10.1029/2007GL030135

Winker, D, Pelon, J, Trepte, C, & Barroso, C. (2013) CALIPSO: Proposal to the 2013 Earth Science Division Senior Review Panel.