Metropolitan State University Denver – Meteorology Department

MTR 3450 – Atmospheric Dynamics

Justin Richling



IDL into Python: Atmospheric Dynamics Lab

# Abstract (should be short and sweet, correct??)

I had the pleasure and challenge of converting Dr. Keah Scheunemann’s atmospheric dynamics lab from her old Interactive Data Language (IDL) version into a semi-recent environment of Python, specifically the working notebook Jupyter.

The primary motivation was to get the students a crash course in Python, as the school doesn’t have a computer science Python course that fits the meteorology department’s computer science requirement just yet. The other main motivation was to help Dr. Scheunemann update her lab from the now expensive and limited IDL language into up and coming open source language. The Python library from Unidata called MetPy are intended for quick data analysis and raw coding.

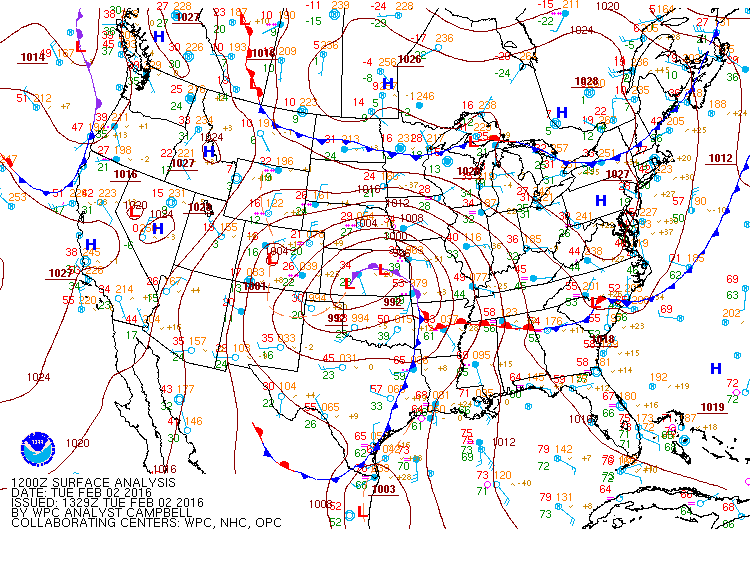
Although the syntax of IDL and Python did have multiple similarities, there were still significant translational issues. The largest obstacle was getting the raw data from the netCDF file into Python and then plotting. There were no similarities between Python and IDL at that point. This forced me to dive deeper into understanding Python and all the different available libraries.

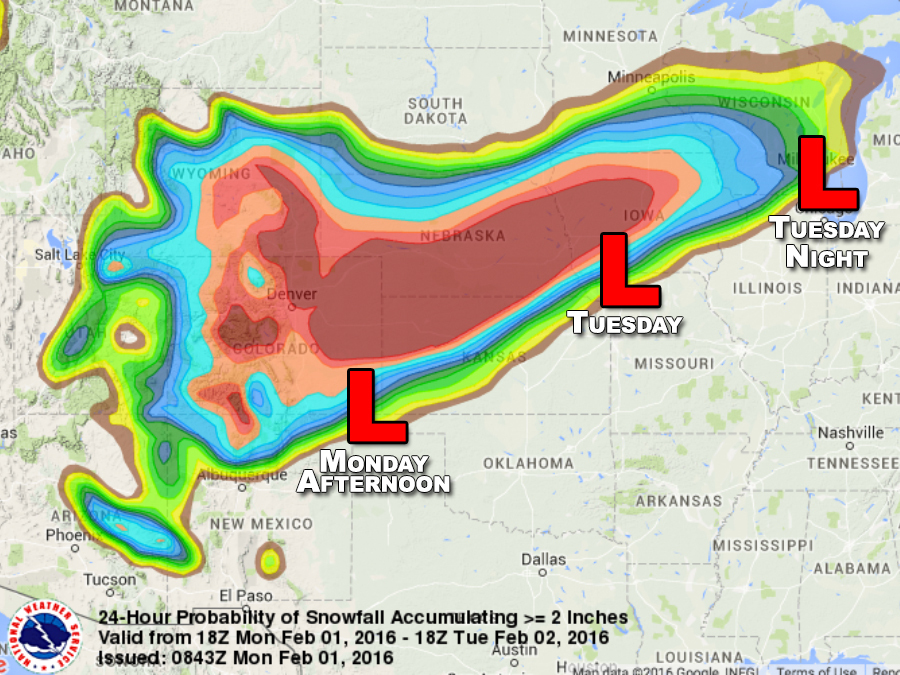
Obstacles:

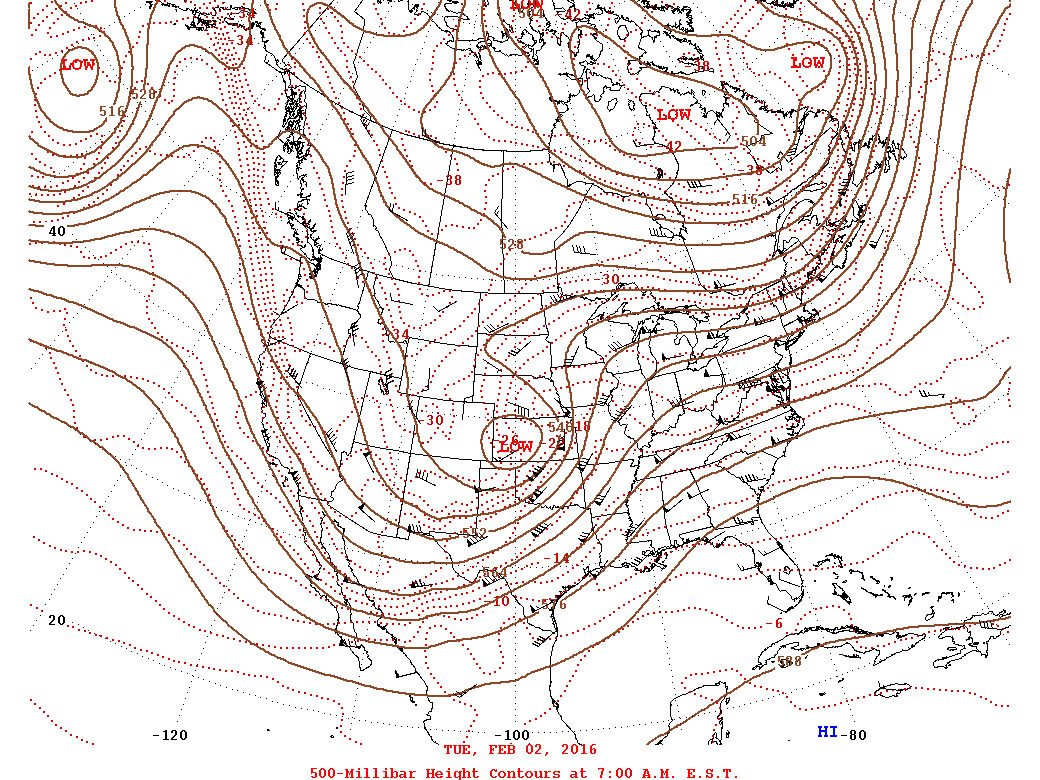
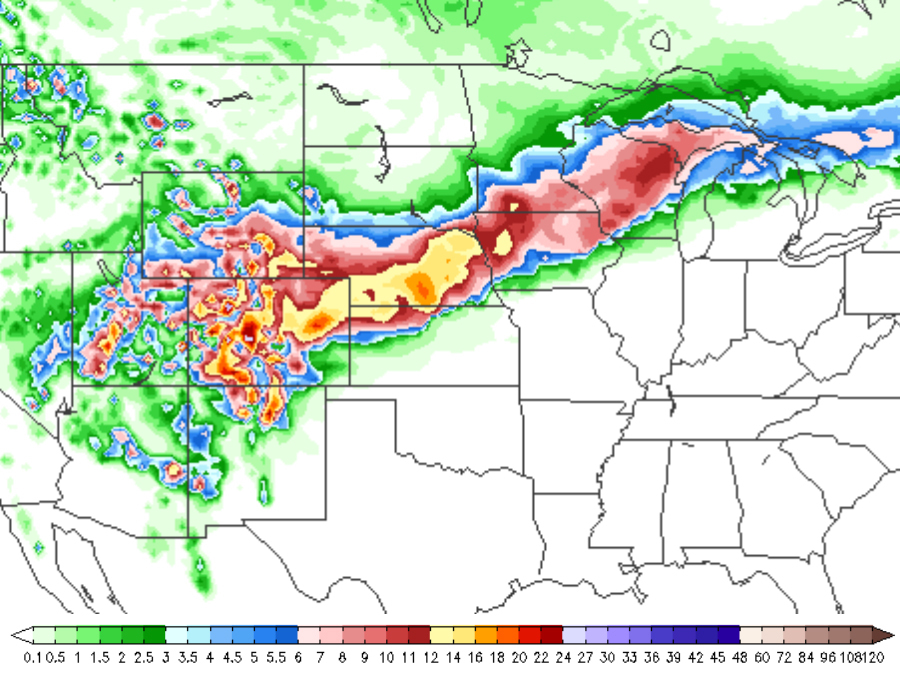
* Python software behaving with the IDV and AWIPS software; this is the most important as the other classes use the lab.
* Trying to time everything as best as possible for both skills taught and buried in student’s memory and small amount of time; we only have a couple of weeks for the whole scheme. This put significant uncertainty
* Varying computer and programming skill levels

# Case Study Intro

The lab was built around the February 2nd, 2016 Groundhog’s Day snowstorm that dropped XX” of snow around the Denver metro area. The synoptic setup included a strong low-pressure system somewhere over Kansas/Oklahoma, giving rise to some upslope flow over north-central Colorado. The occlusion around the low indicates that there is some warm, moist air aloft from the Gulf of Mexico also adding to vertical velocities??



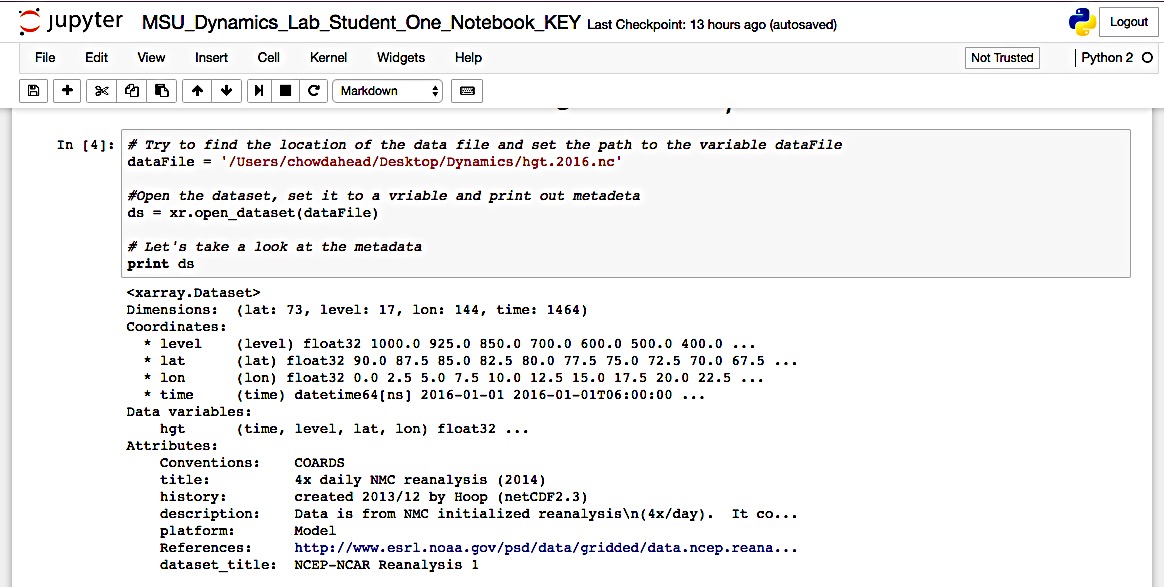




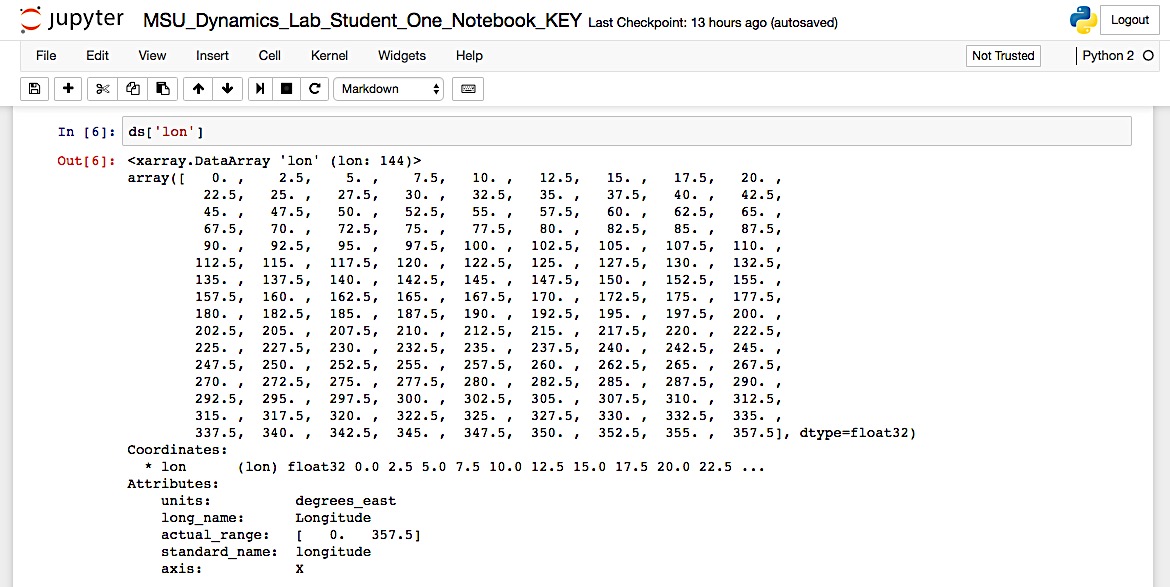
# Jupyter Notebooks Intro

There exists an interactive version of Python in which the user has control of how much and where in the code script is run. Called IPython, it has several advantages relating to user interface and transparency. This type of *working* *notebook* is a called ***Jupyter notebook***.





With this setup, the user can run specific cells – code blocks separate from each other – to test specific pieces of code or just keep everything nice and separated for personal organization. The individual cells also allow for easy diagnosis without having to run the whole script, say when one wants to know the value of some data or what the current variable is equal to.



# Lab Intro (what we did with this lab…)

In MSU Denver’s atmospheric dynamics course, there are numerous topics covered, but the lab only touched on a couple major ones to highlight Python’s calculation and plotting abilities. The students begin the lab with a crash course into programming data types and Python syntax.

One lab session gives the students and opportunity to understand how Jupyter notebooks are different than regular Python and other programming scripts.

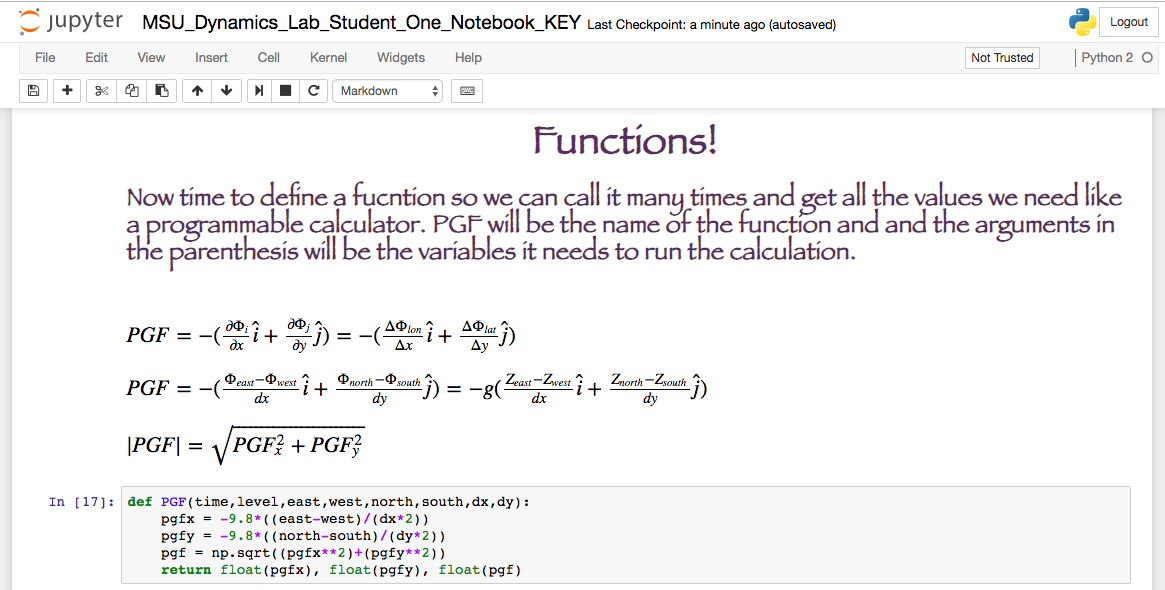
The next set of time is designed for the students to learn the *netCDF* data file type; one of many different file types used in atmospheric sciences. In doing so we will quickly cover locating and opening the *netCDF* file and reading the variables along with the metadata; specific info of the variables. In addition, the students will learn how to make a function so that they could calculate and fill ***arrays*** with the *PGF*, *Coriolis Factor*, *Geostrophic Winds*, and *Absolute Vorticity*, all from the *latitude/longitude* and *Geopotential Height* variables. Finally, the students will create a new *netCDF* file and fill with the calculated arrays as they progress. This will be helpful so when they come back to their notebooks, they won’t have to wait for the calculations to run every time. The new *netCDF* file holds all the new data arrays. The students will have a new Jupyter notebook for the data analysis and calculations.

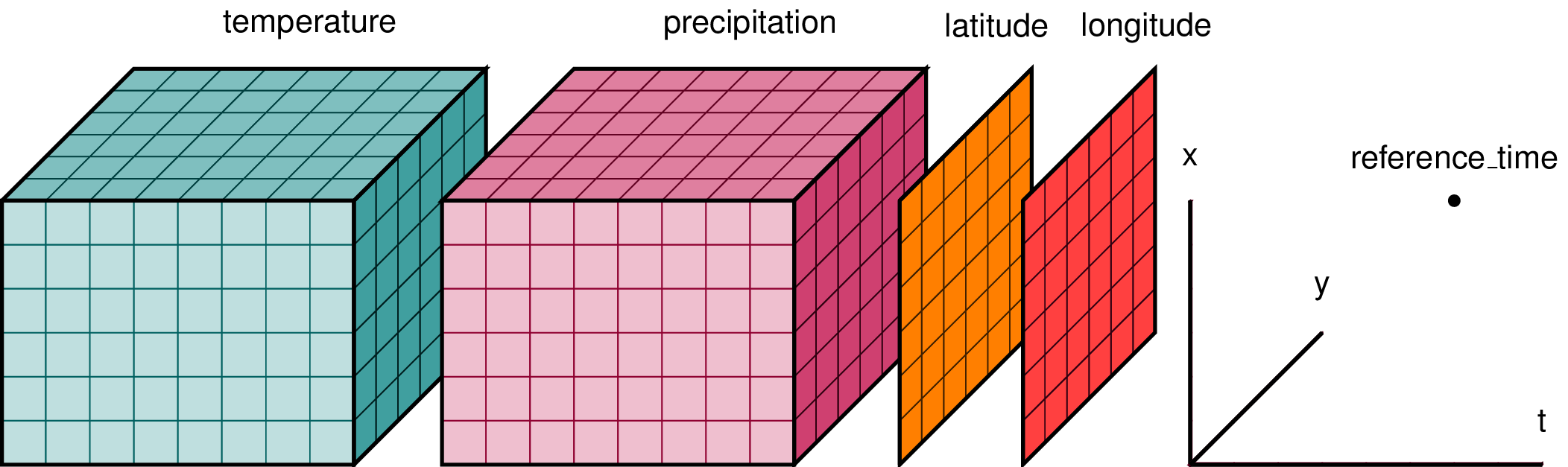
The subsequent time allotted is meant for plotting the array data they calculated. We will go over the CartoPy, NumPy, and Matplotlib libraries. CartoPy will set up the map background/plotting based off projection of data. NumPy will take care of any mathematical regimes we need for things like pi and trig functions as well as the actual arrays we will use to house the multidimensional data. Matplotlib is the plotting library to set up the figure and axes to plot. This library will also allow for customization of plotting functions like contour labels, axis titles, colorbars, etc. Finally, the students will read the new calculated data into a new Jupyter notebook designed for plotting only.

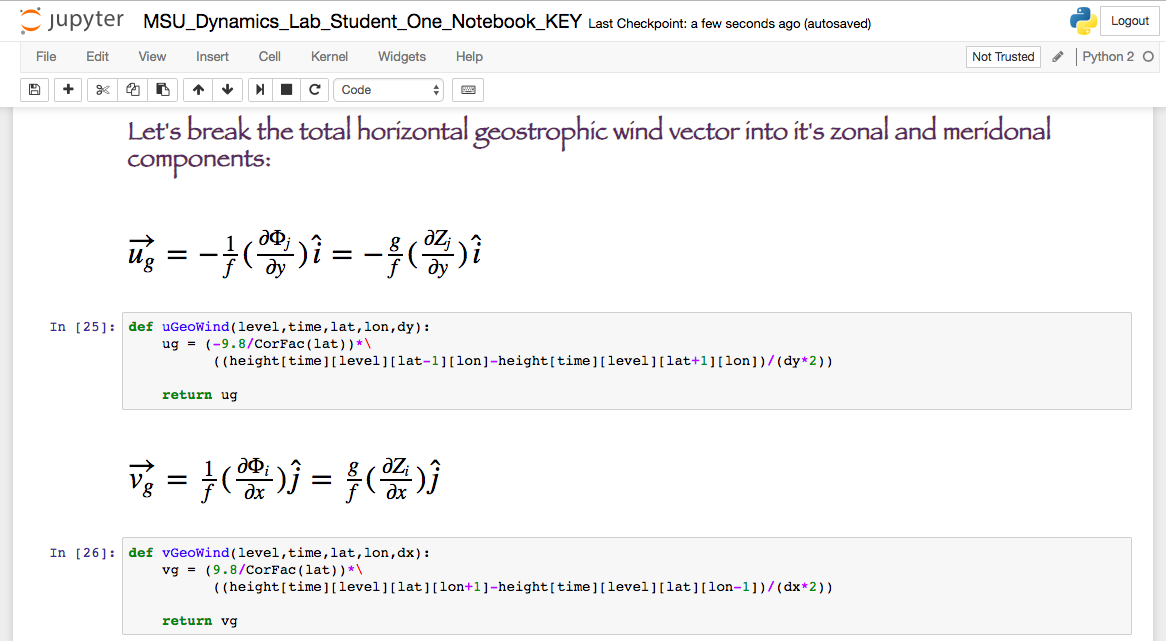
The original *netCDF* file included **times**, **latitudes**, **longitudes**, and **geopotential heights** for the whole globe four times daily for the calendar year. We only focused on the continental US and most of Canada for the three time stamps 06, 12, and 18 Z of February 2nd. The time in the original *netCDF* file was stored based off hours after a certain year and it was confusing, so the students manipulated the data based off the index number that corresponded to the timestep number, ie if we wanted Feb 2nd at 12Z, our index number would be 4 x daily for the number of days from Jan 1st to Feb. 2nd +time steps to get to 12Z.

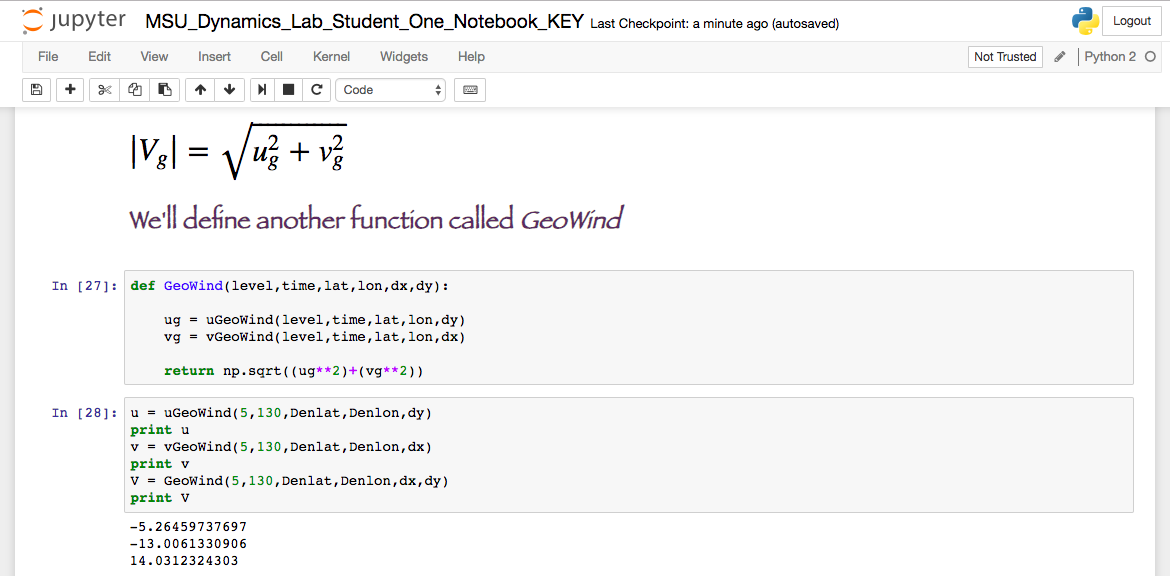
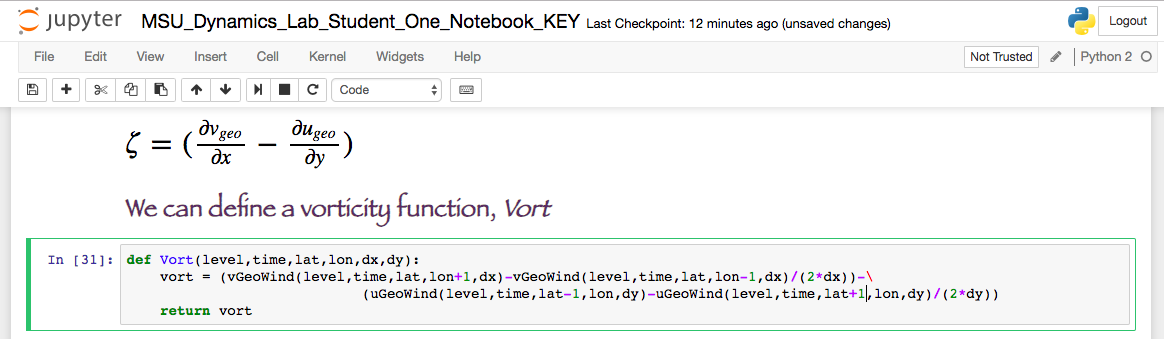
Thus Feb. 2nd 12Z was index number 130.

The students were instructed to calculate Coriolis Parameter, Geostrophic Winds, and Relative Vorticity. They learned what it was like to work with ***arrays*** of data and to create their own for storing new variables made from calculations they preformed with functions they created, working on raw data.

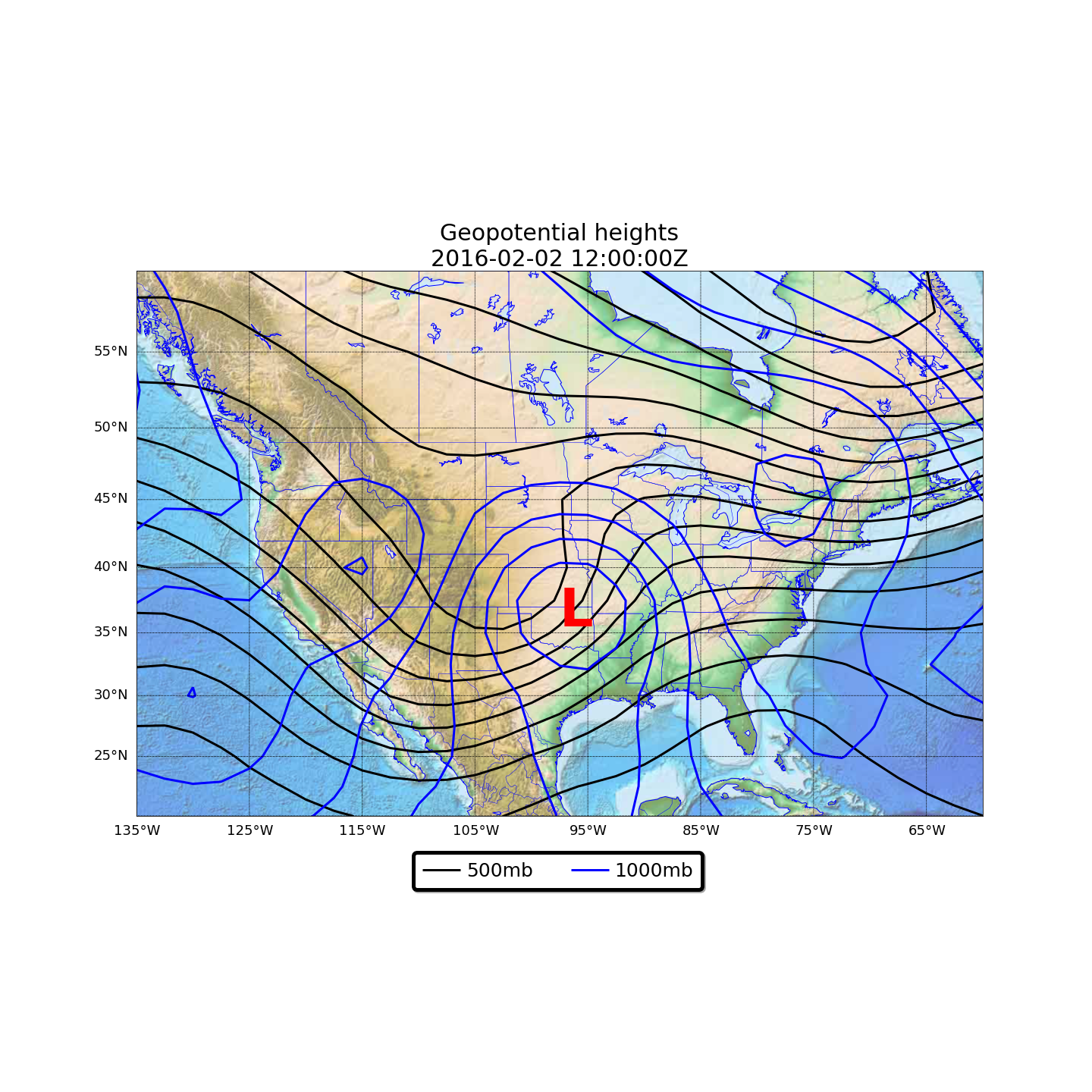








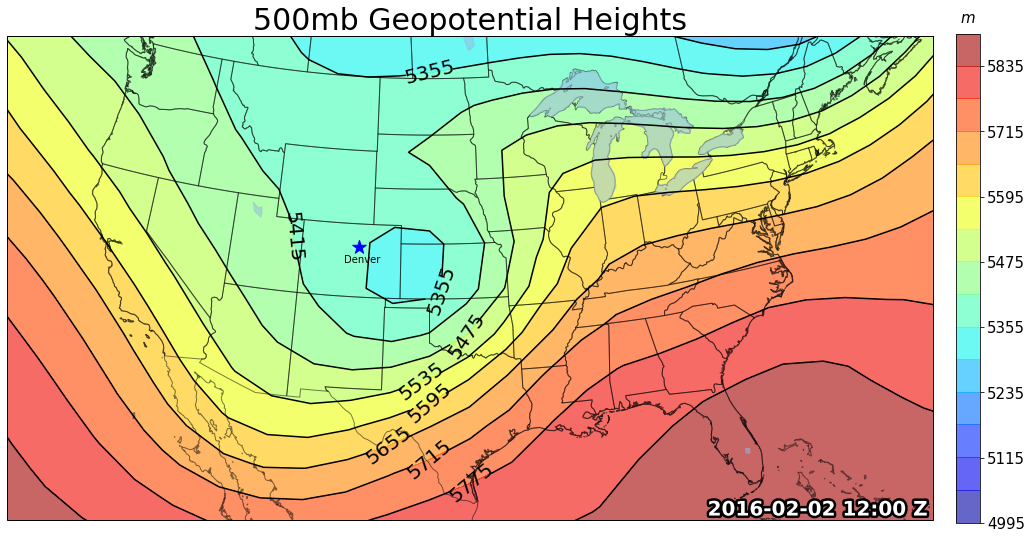
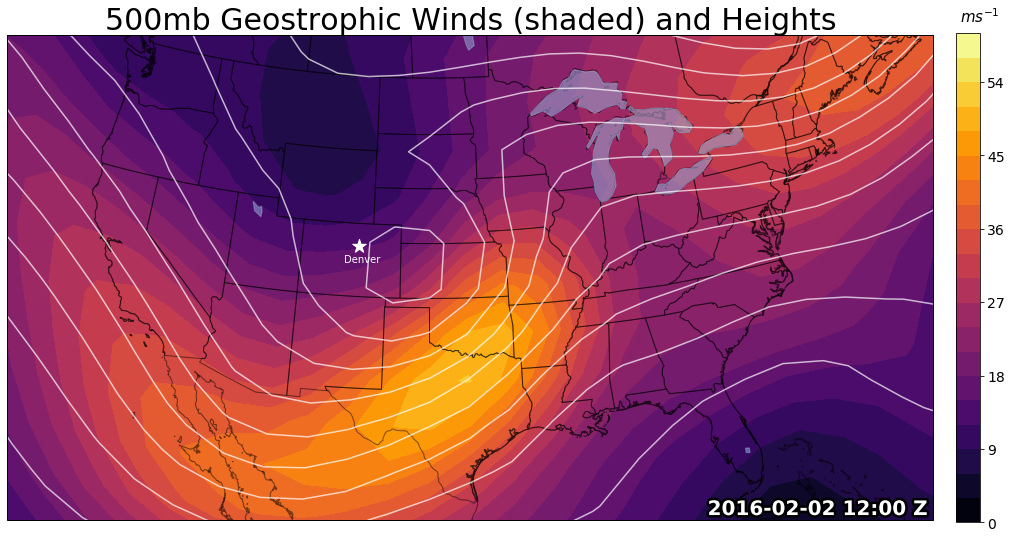
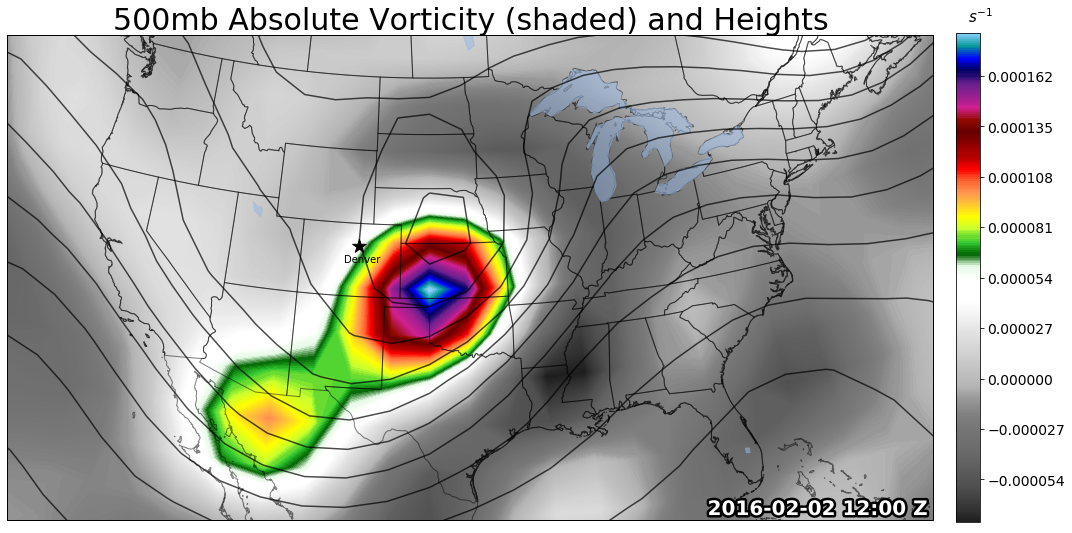
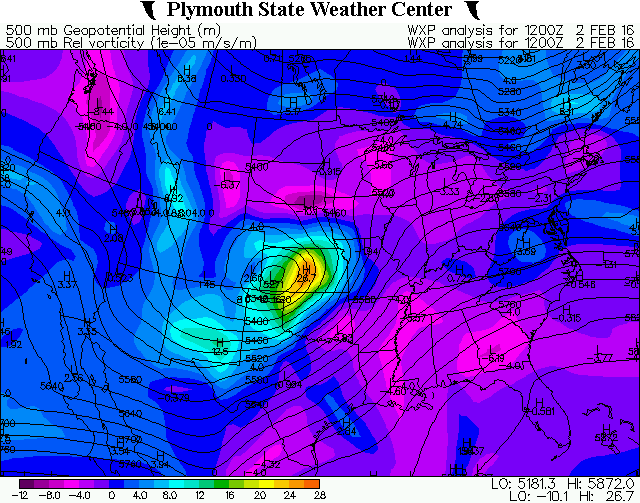
# Outputs

Generated outputs include: geopotential heights at 1000 and 500mb, Coriolis parameter, pressure gradient force, geostrophic winds at 500mb, and absolute vorticity at 500mb, all globally. Then the students plotted the 500mb Heights, 500mb geostrophic winds with heights, and 500mb vorticity with heights.

# Old IDL Outputs

Hv;jnl’km

# Python Outputs



# Student Responses

I asked the students to give some feedback about the lab after the completion. This was intended to help me and Keah figure out if the new lab was worth it and get the student’s perspective on the pace and difficulty.

# Conclusion (what did I learn from this…)

After testing the new lab out on the fall 2018 Dynamics students, there were obvious successes and setbacks.

Success:

The most evident success from this project was that it worked and got the students to engage in Python. Plotting of the variables worked out nicely and was able to customize the maps nicely.

The definitions and array manipulation appeared to be a great estimate of the quantities we were covering.

Setbacks:

There was a lot of hand holding by my part, mostly due to lack of programming knowledge and time constraints as the course only allowed for 5 or 6 lab days.

One of the major variables with this lab is pacing the lab days based on the how the students are adjusting and understanding the code.

Getting all the Python libraries to play nicely; one of Python’s biggest hurdles.

Major Changes

* Update for Python 3 and Cartopy - Python 2 and Basemap are being depreciated
* Create a lab manual
* Create a better final write up – ask a bit more than I did this year
* Possibly add to homework – for maybe the vorticity section??
* Get Keah up to speed on Python!!
* More explanation of code?
* Double check numbers with the original IDL lab…
* Allow for commenting and HTML customization
* Change the maps to one generic map function and call it with different maps

# Thanks

There are numerous thanks to be given for this project. First and foremost to Dr. Scheunemann for allowing me the opportunity to use her class as a guinea pig to implement my Python notebooks and for being able to learn how to teach. She lent me the trust which …

Thanks to Chris Kimmett for hunting bugs with the misbehaving Python libraries and not crashing AWIPS. Without you the project wouldn’t even been able to get off the ground.

Metropolitan State University of Denver

Erin Regan my fellow learning assistant who learned some Python and helped students in the lab where she could.

Unidata for supplying a wonderful resource for atmospheric sciences and Python.

Brooke Evans for allowing me to bill my work as a learning assistant to accomplish the interface of my project and the dynamics course.

Finally – The Dynamics students!

# Resources

Figures 1 and 5:

* <https://www.thedenverchannel.com/storm-shield/storm-shield-featured/blizzard-from-colorado-to-michigan>

Figure 2:

* <https://www.wpc.ncep.noaa.gov/archives/web_pages/sfc/sfc_archive_maps.php?arcdate=02/02/2016&selmap=2016020212&maptype=namussfc>

Figure 3:

* <https://www.wpc.ncep.noaa.gov/dailywxmap/dwm_500ht_20160202.html>

Figure 4:

* <https://vortex.plymouth.edu/myo/upa/ovrmap-a.html>

Figure 6:

* <https://www.bouldercast.com/groundhog-day-2016/>