August 24th, 2020

1. Kick off meeting w/Mun and Aneesh to go over what we have done so far.
2. Will meet with Mun every other Friday, starting Sep 4th (1pm).
3. Long-term goals:
    a. Understand how high wind events have changed in Boulder
    b. Data streams:
        i. SONIC (new) anemometer
        ii. WXT (older) anemometer
        iii. Radiosonde data
        iv. Re-analysis data
4. FIrst steps:
    a. Download data from CHORDS. See data download page and follow instructions, if too problematic, we will contact EOL engineers.
    b. Use sample data to start plotting and scripts
        i. Timeseries files of SONIC and WXT, overlaid on each other, wind magnitude vs time.
        ii. Wind roses for both SONIC and WXT. Here is documentation on NCL windrose for general information. Wind roses tell us the prevailing wind direction, mean, and standard deviation for a specific location. This will help us determine how things have changed over the years, and based on instruments.
5. Next steps:
    a. Look at wind rose by month (season) for both SONIC and WXT
    b. Look at max wind: full timeseries, and by month (season) for both SONIC and WXT.

September 3rd, 2020

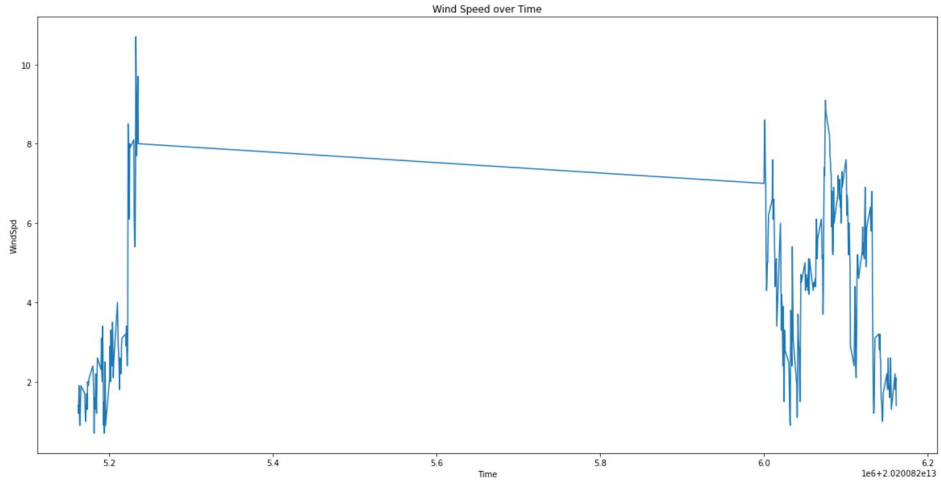
1. Timeseries plot from sample data is looking good with the exception of the transition from one day to the next due to timestamp. Moon will work on creating a time array that will produce consistent and smooth data across the x axis.
    a. Pick a time for the reference time… say, Jan 1, 2020 (or Aug 1 2020 if you want less missing data). Unit will be hours.
    b. Create time array using missing data (example, set missing metadata parameter to -999)... Aug 1 to Aug 24 will be missing… Aug 25 and aug 26, actual data.
    c. This is to get used to plotting timeseries with a reasonable x-axis and to understand how to handle missing data. Example:
        i. time (0) = 0, wind(0) = -999 (for missing data if starting at aug 1)
        ii. time(n) = <hour value for aug 25th 00z, i.e. 24 days * 24 hours per day>, wind (n) = first actual value
2. Once have timeseries, and missing data plots, will move to wind rose

3. Played with csv in excel software… showed Moon how to cut and paste columns if she wants to avoid reading in the entire csv file in python.

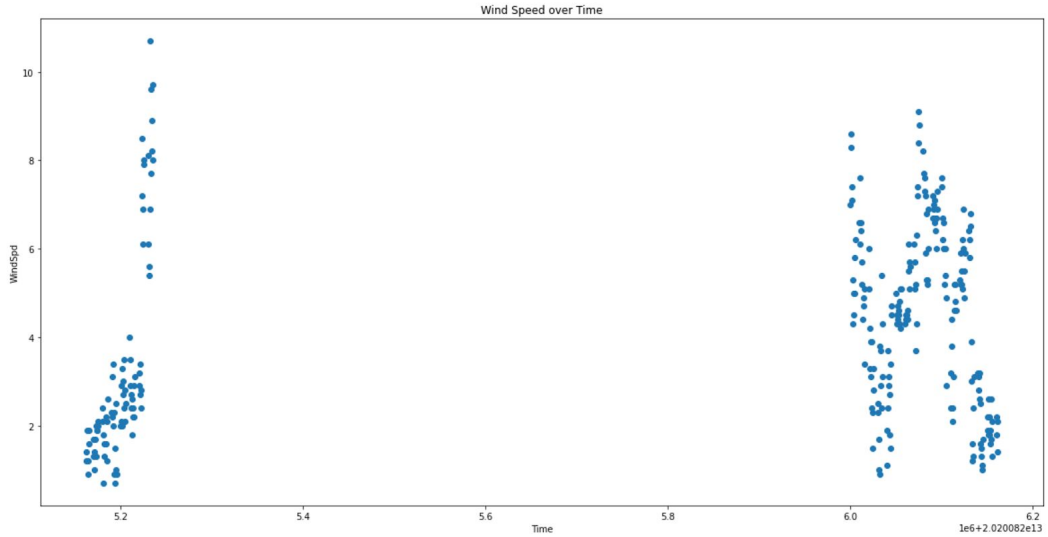
September 11th, 2020

1. In regards to my call with Aneesh, I decided to play around with different plotting techniques and see which one lended itself the most useful.
    a) He recommended I try and use a scatter plot. Even though we can't as clearly see the relationship between the wind speed and time, it at least gets rid of the weird line connecting the two days and throwing off the data.

Here are some Comparison Images.



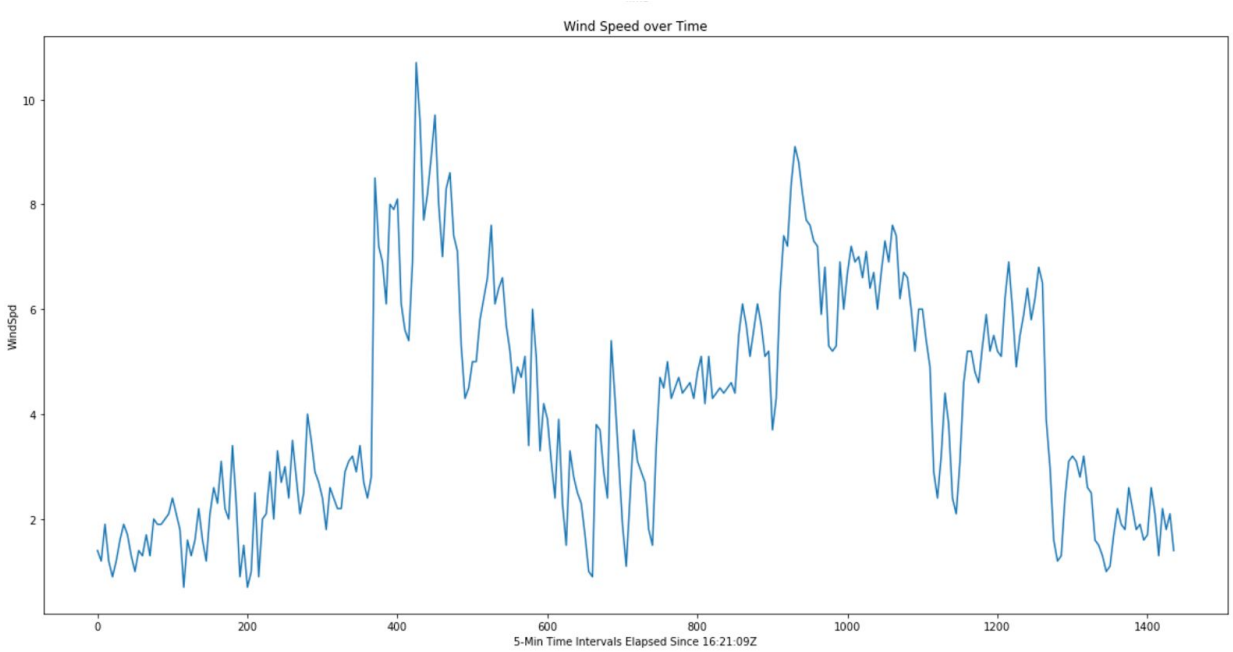
This is my Original Plot. As you can see, it makes a connection between Aug 25th and Aug 26th, even though none should exist.



Following the scatter plot documentation, we manage to get rid of that line altogether and see the wind speed across separate days. I could consider putting in an axis break so that they're not as spaced apart. The Time here also needs to be fixed still with the Python Date/Time

documentation that Aneesh sent. I haven't gotten around to implementing it just yet but next week I'm hoping to do that.

Here is the final thing I did, which was to change time into an interval of 5 minute elapsed periods.



This is by far the best results I've gotten. The x-axis will need some more work in terms of readability as I'm sure it won't make sense, but basically it's minutes elapsed since ____. So the 1400 at the end of the graph is saying, 1400 minutes elapsed since ____ Zulu. Or whatever our beginning timestamp may be.

Goals:

Next Week, what I'm going to try to do is try and fidget around with the timestamp information Aneesh sent me so that maybe I can work with time in the way it's supposed to be instead of having to convert it to 5 minute intervals. I'm also going to begin plotting more sample data and see if I can produce a couple graphs like the one I produced above. Once that looks good, we can start on Wind Roses!

September 18th, 2020

1. Gary Granger has provided the csv files for 2018-2020! They have been uploaded to the drive here
2. The SONIC is the 1 sec data (larger files) and the old "WXT" is the 5 minute data (smaller files).
    a. SONIC: variables are: datetime,Dir,Spd,Status,U,V

    b. WXT: variables are: datetime,wdir,wspd,wspd_max,tdry,rh,pres,rain_accum,batt
3. Units (from Gary):
    a. Wind speeds are in m/s,
    b. temperatures in Celsius,
    c. pressures in hPa,directions
    d. compass degrees from true north,
    e. rain accumulation in mm.
    f. I think the wind direction is downwind and not upwind, but I did not verify that.
    g. WXT "old" sensor: RAL website for ML
4. To-do tasks
    a. Python: figure out how to ingest CSV file w/o automatically having all variables as a string.
        i. In NCL, routines to read in ascii files (and csv files). Is there something like this in python?
        ii. https://www.ncl.ucar.edu/Applications/read_ascii.shtml
        iii. We want to avoid ingesting data as a string.
    b. Plot SONIC (one file to start), add all files once you've got that complete
    c. Plot WXT ""
    d. Try plotting both SONIC and WXT together
        i. Take mean of SONIC for 5 minutes (300 sec, time steps), save that into a different array
        ii. Plot the new array w/WXT -- both will have the same time slices
        iii. Plot max_spd in WXT files with a max windspeed we will compute w/ SONIC : Similarly to d.i. -> take the "max" for 300 sec time steps, save that max value into a different and the compare with WXT.

October 2nd, 2020

1. Username is mpasha - You have an account on Cheyenne, Casper, (NCAR supercomputers) with this username! Use Duo to log on. Instructions are here.

2. Reminder:
   The SONIC is the 1 sec data (larger files) and the old "WXT" is the 5 minute data (smaller files).
    a. SONIC: variables are: datetime,Dir,Spd,Status,U,V
    b. WXT: variables are: datetime,wdir,wspd,wspd_max,tdry,rh,pres,rain_accum,batt
3. See September 18th, To-do's: (#4)

October 14th, 2020

1. Pandas has proved GREAT results! No longer need lines and lines of code to convert the variables to proper types. It does it automatically as it's imported into a dataframe.

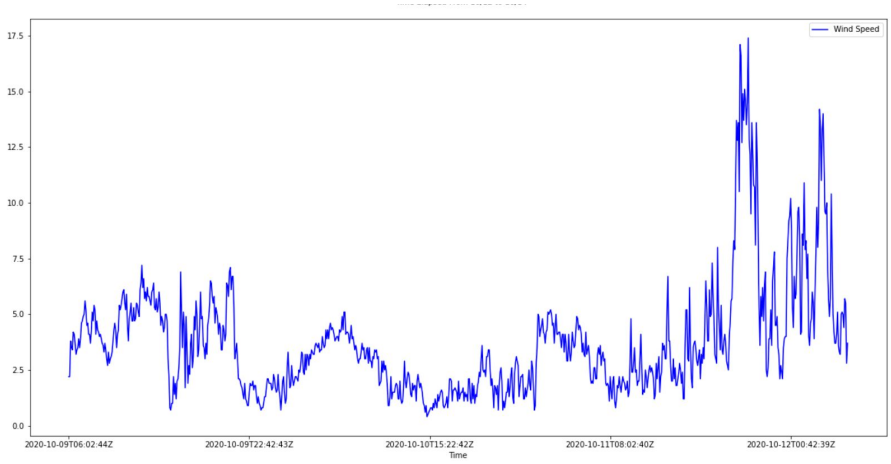
a.

```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
from matplotlib.pyplot import figure
import csv
df = pd.read_csv('RecentWindData1012-1014.csv', encoding='utf-8')
df2 = pd.read_csv('WindData1009-1011.csv', encoding = 'utf-8')

#df.head(18)
#df.info()

ax = df.plot(kind = 'line', x = 'Time', y = 'Wind Speed', color='red', label = "Wind Speeds from 10/12 to 10/15", figsize = (20, 10))
df2.plot(kind = 'line', x = 'Time', y = 'Wind Speed', color='blue', label = "Wind Speeds from 10/09 to 10/12", figsize = (20, 10), ax = ax)
ax.set_xlabel("Time Elapsed")
ax.set_ylabel("Wind Speed in m/s")
ax.legend()
```

b. Always nice to see a 80-90 line program be reduced to just a few lines.

2. JupyterHub on Cheyenne wasn't working for me for some reason. First it was their clean up, but then when I clicked on the links provided for it, it said I couldn't get into the page or that the page was unavailable. More maintenance (?) So instead, knowing that October 12th marked the start of "Wind Season", in Colorado I wanted to see what wind graphs looked like!
3. I plotted here a graph from the WXT Anemometer Data (http://portal.chordsrt.com/data) from October 9th to October 12th first, just to see what Wind Speed trends looked like prior to "Wind Season" beginning.

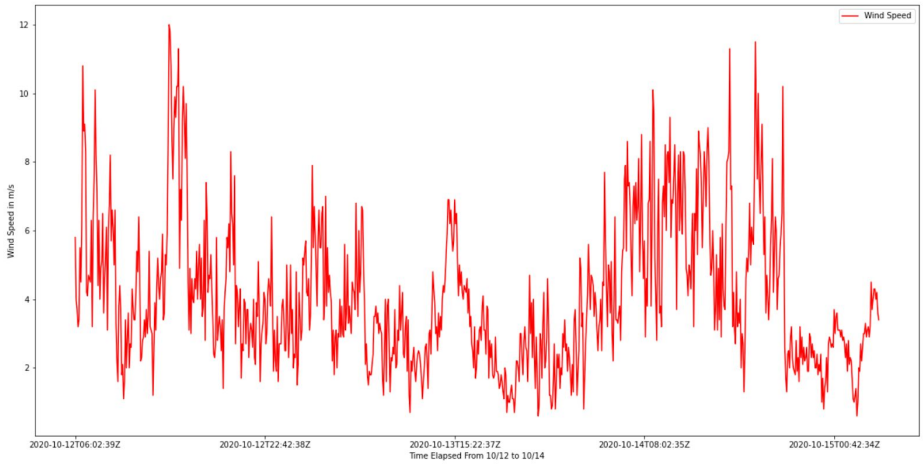
a.



b. Notice how the wind speed SPIKES on October 12th!

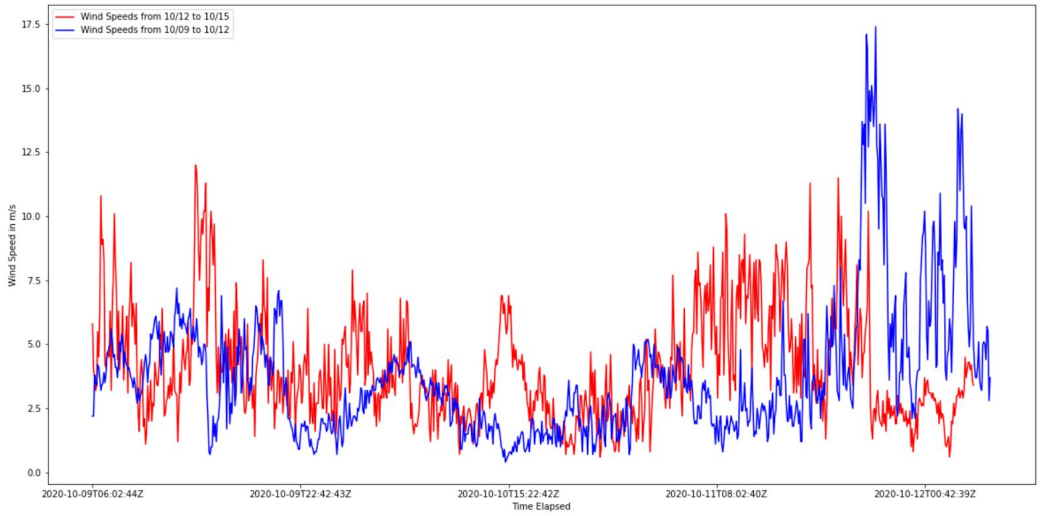
4. Next, I plotted data from October 12th through the 15th again for the WXT data.

a.



b. You can see in this graph how crazy the winds were over the last 3 days! Really shows the emergence of this “Wind Season”.

5. Finally, to test out some of my skills, I wanted to overlay both of these charts on top of eachother.

a. 

b. I’ve labeled the **Red Line Plot** as Wind Speed Data from October 12th, 2020 - October 15th, 2020.

c. I’ve labeled the **Blue Line Plot** as Wind Speed Data from October 09th, 2020 - October 12th, 2020.

6. I was actually able to get Sonic data from the entire year loaded into a Pandas Dataframe, however my machine doesn’t have the capability to graph that immense amount of data. So the only way to handle Sonic data is to do it on Cheyenne which I’m still having issues getting to.

a.
```
#Use Pandas and comma as delimiter.
import pandas as pd
df = pd.read_csv('s131Mhdfntjn', encoding='utf-8')
df.head(5)
```

	datetime	Dir	Spd	Status	U	V
0	2018-01-24T19:13:01.0522	4	1.76	60.0	-0.12277	-1.7557
1	2018-01-24T19:13:02.0527	359	1.90	60.0	0.03316	-1.8997
2	2018-01-24T19:13:03.0496	354	2.27	60.0	0.23728	-2.2576
3	2018-01-24T19:13:04.0508	346	1.77	60.0	0.42820	-1.7174
4	2018-01-24T19:13:05.0520	342	1.33	60.0	0.41099	-1.2649

```
df.info()
plt.show()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 29410107 entries, 0 to 29410106
Data columns (total 6 columns):
 #   Column    Dtype
---  ------    -----
 0   datetime  object
 1   Dir       int64
 2   Spd       float64
 3   Status    float64
 4   U         float64
 5   V         float64
dtypes: float64(4), int64(1), object(1)
memory usage: 1.3+ GB
```

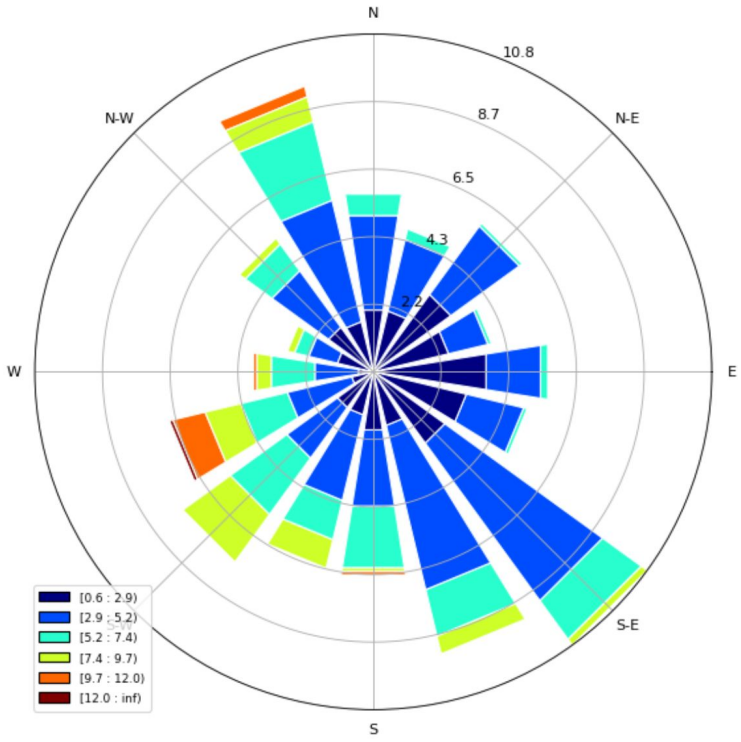
b. As you can see, I was able to load in sonic data, but again my local python distribution can’t handle the actual plotting.

7. Windrose Practice! I took data from WXT Anemometer from October 12-October 15, and decided to put it into a WindRose!

a.

```
[7]: import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
from matplotlib.pyplot import figure
from windrose import WindroseAxes
from matplotlib import pyplot as plt
import matplotlib.cm as cm
import csv

df = pd.read_csv('RecentWindData1012-1014.csv', encoding='utf-8')
df.head(18)
windSpeedList = df['Wind Speed'].tolist()
windDirList = df['Wind Direction'].tolist()
ax = WindroseAxes.from_ax()
ax.bar(windDirList, windSpeedList, normed=True, opening=0.8, edgecolor='white')
ax.set_legend()
```

b.



c. Here is my windrose for data across those 3 days!

8. TO-DO Tasks
   a. Try and Log onto Cheyenne still.
   b. Get sonic data loaded in, get it into pandas, and plot it against WXT data for the same year!
   c. Plot consecutive files for one long timeseries (conjoining the data together)
   d. Repeat plotting sonic and wxt together..... you may have to figure out to do timestamps given that they may not exactly line up.
   e. Windrose: put into mph and figure out if frequency is in % or # of counts. (2 for sonic, 2 for WXT, one will be all of the data, one will be the top 10%)

i. Here is an example of how to read a windrose https://www.mtavalanche.com/weather/windrose#:~:text=A%20wind%20rose%20is%20a,for%20a%20certain%20time%20period.&text=The%20most%20important%20data%20is,coming%20from%20a%20particular%20direction.Max function in python (Descending order, take top 10% of data).
ii. And then Plot this ^.