Danny Siu

Discovery Research Program as a Data Science sophomore,

Working with post-doc Chi Li on the BEHR project,

Trying to replace the AMF calculations with ML models that replicate the same output more efficiently after training on satellite data and WRF-Chem inputs (relevant to $NO_2$ profile)

# BEHR Project Log (2/1/19 - 8/9/19)

Feb 1, 2019 – Talked to Ron Cohen, got introduced to Chi after expressing interest in the BEHR project.

Feb 5, 2019 – Group meeting with Discovery Research Program undergrads and grad student researchers. Learned a bit about BEACON and talked with Chi more about the BEHR project.

Created my account to have access to the servers with satellite (OMI) and WRF-Chem model data

Plus, got Berkeley's VPN to connect with servers while working from home

Feb 8, 2019 –

Talked a lot with Chi about the BEHR project more in detail.

Understood more about AMF and WRF-Chem,

Going more into details about what data we are using to simulate the AMF calculation.

[will take pictures and paste down better records of what I learned]

Plan for the next week:

Joining the WRF-Chem pixels' data with OMI data by geolocation coordinates (long & lat) and time(hr)

1. Align OMI with WRF (same geolocation and same time)
2. Anything that's 3D, interpolate to OMI Pressure Coordinate
   (Convert WRF-Chem Pressure measurements to the corr. OMI's vertical coordinates (bc WRF-Chem has 29 layers while OMI has 33))

Future Steps:

1. Find nearest pixel.
2. Find a 5x5 window in WRF-Chem.
3. Collect from last hour to now.
4. Calculate wind velocity for each.

Weekly Plan: Going to meet with Chi on Mon and Fri, 2-6pm

My next task for Monday in more detail:

Use OMI information to find WRF-Chem pixels:

Take inspiration from this MATLAB code which already does this. (Code is in “BEHR-core-utils/AMF-tools/rProfile_WRF.m”)

(Options: can either convert this to Python (make my own code) or repurpose this code to run on our files)

Test this process for year 2013 data – test this for 1 week in a loop first

- We can use the servers to calculate for entire years (so I don’t have to run huge computations and store lots of data locally).

Another note: we can convert time to more useful data. Instead of seconds, can use a DateTime object to store day, month, yr, hour,

Feb 15, 2019 –

**Coding work today:**

I began by reading the background behind the code for BEHR, listed in “BEHR_Readme.pdf.” It gave helpful information on the formatting of the BEHR code, reading their data, and some additional notes on variables and function explanations that I didn’t fully get to.
(Code is in “/Documents/Spring 2019/BEHR_Proj/Danny_Siu_Behr_Proj_Model.ipynb”)
Afterwards, I set up presudocode to get information from OMI and WRF_Chem pixels (to be coded up).
Then, I wrote code to assign all corresponding WRF_Chem pixels to each OMI pixel using corresponding location.
I tested that this works, assuming that I can successfully grab arrays from OMI and WRF_Chem pixel files.
Then, I planned next steps with Chi, working out details on how to store this data from OMI and WRF-Chem pixels into dataframes. Eventually, we might make a visual map that can easily show WRF-Chem data superimposed onto a dynamic OMI gridding system (bc OMI pixels change location constantly).

Note for stored variables involved:

WRF_Chem pixels have a CENTER variable, which stores long and latitude of the center

OMI pixels have corner variables, which have longs and lats of each corner

Most of my code today was on my function "interpolate_WRF_to_OMI(wrf_pixel_arr, omi_pixel_arr)".

It tests that WRF_Chem pixels' center is in the box formed by OMI pixel corners; this is how the code in "..\BEHR_Proj\BEHR-core-utils\AMF_tools\rProfile_WRF.m" calculates it in the function "avg_apriori()".)

Goal: Essentially, we want to create a map of all the WRF-Chem pixels' measurements onto a grid of OMI tiles. This map can be created visually later, after I create the dataframes.

Feb 18, 2019 – Holiday. I have a break!

Feb 22, 2019 –

**Coding work today:** **I spent a lot of today, just trying to figure out how to read in hdf5 files (for OMI) and netcdf files (for WRF-Chem) and access their variables.**

I might have some storage issues for just storing the data from WRF-Chem files (it had a Memory Error when I used the xarray's open_dataset(ds).to_dataframe() method). So, Pandas might not be a good option for dealing with all this data, just because of the quantity of data that I'm dealing with.

Using pathlib, Path objects for easy filepaths. *(from pathlib import Path)*

**Several possible ways to open netCDF files (WRF-Chem): (* = seems to work well)**

***Xarray library -** *xr.open_dataset()*

***netCDF4 library, importing Dataset,** which opens the netCDF file as a sort of Python dictionary object

*file = Dataset(filename, 'r')*

**netCDF library**

```
# from netcdf import netcdf as nc

# root, is_new = nc.open('file_*.nc')

# print(root.files)
```

**Trying to open HDF5 files (OMI)**

**Tried to use Pandas to read HDF files into a dataframe (pd.read_hdf(filename, header/variable??))**

**For WRF-Chem files:**

Variables:

CEN_LAT and CEN_LON = centers of whole domain

X_LAT and X_LON = centers of the pixels – use these to align with OMI pixels! These will come out as a list of all the different WRF-Chem pixels.

The netCDF files has the entire WRF-Chem pixel domain, so it has measurements for all pixels in one hour, one day. Contrary to what I thought, it's for ALL the pixels, not just one.

**Grabbing data and placing it into dataframes:**

For OMI, we are going to pull everything – all the information will be relevant for calculation.

For WRF-Chem, pull those specific variables that Chi specified (listed in “Calculate T and RH from WRF.doc”). I’m going to run calculations for these variables and insert them into a dataframe.

**Immediate next steps:**

Continue figuring out how to access variables and all the information in the HDF5 and netCDF files. This is critical so that I can put this information into dataframes.

This includes the task of parsing the naming scheme of the WRF-Chem files and the ‘hour’-folder that it belongs to in order to identify the timestamps of these files. Then I can interpolate these files with the OMI data, along with location (which I finished last time).

**Next possible steps:**

Convert WRF-Chem pixels to OMI matrix → (That’s ~5 WRF-Chem pixels to 1 OMI gridbox)

Trying to make it easy for data training and correlation -> allow for easy 1 to 1 data correlation (instead of having to correlate 5 datapoints with 1 OMI pixel when training the data model).

In order to perform this conversion of data,

1. Determine which WRF-Chem pixels belong to the specific OMI pixel (at that hour and day)
2. Average these 5 pixels’ measurement values and put this data into a corresponding OMI matrix pixel.
3. Now, we’ll have 2 OMI-gridlike matrices, which we can use to train the data on when using both datasets to correlate to the AMF output.

**TBD:**

Need to align all the WRF_Chem_pixels with matching DATE_TIME as well, with the corresponding OMI tiles.
(Another factor to account for: WRF-Chem pixels have fixed location, while OMI locations are constantly changing.)

Notes about variables:

> WRF-Chem pixels have fixed location, while OMI locations are constantly changing.
>
> WRF-Chem pixel date can be found in filename, calc hour using "TIME" variable

Next steps:

1. Create a dataframe with OMI pixel as the primary index.
   First columns will hold OMI_pixel_index, OMI_pixel_time (might need to use indices for time to refer to another table with diff times, avoids many duplicates)
   Secondary columns will hold list of WRF_Chem_pixel_indices.

2. Create a second dataframe with WRF_Chem_pixel_indices as primary index.
   Columns will hold the different measurements we want to gather for each WRF_Chem_pixel (air pressure, temp, and RH)

3. In the future, we can create a map of all the WRF-Chem pixels' measurements essentially superimposed onto a grid of OMI tiles.

**(TLDR) Needs to be done next time:**

Align dates, (in addition to location), for WRF_Chem pixels and OMI pixels.

Create dataframes for easy access to information relating WRF_Chem pixels and OMI tiles.

3/4/19 –

Idea from Vincent to convert my text files to pandas dataframes:

pandas.to_csv() - use to convert basically any text file to csv!

pandas.read_csv() - read in data!

3/8/19 –

Tested the pandas “to_csv” method. Doesn’t work very well because input files must be in a very specific format. This method works well for converting dataframes to csv.

Using the H5py python library to open hdf5 files from BEHR (satellite data).

> Helpful code to open hdf5 files in this StackOverflow post: https://stackoverflow.com/questions/28170623/how-to-read-hdf5-files-in-python

I am able to extract DateTime object data (YYYY-MM-DD) from filenames, so that I can align the data by date AND location in Pandas later. Was able to do this for both BEHR and WRF-Chem files.

Discovered that the BEHR files that we were using did not have the time and geodata that we needed. We are now switching directories and using the BEHR records in the directory: "Z:\share-sat\SAT\BEHR\BEHR_Files_v3-0B\us\daily\" on server 19, (rather than the published files in the WEBSITE folder w/ limited data).

> Note: These BEHR records are stored as MATLAB files!! I have to use something else to open these files in python. Able to use scipy.io library module to open matlab files as numpy.ndarrays.

> Note: h5py is no longer necessary, since we are no longer using hdf5 files for the BEHR records 😐.

Completed:

- Already able to align with geolocation using my function
- Able to create a DateTime object from WRF-Chem filenames, with accuracy up to the hour.
- Able to now open both WRF-Chem and BEHR files

Plans:

- Still need to extract the hour from BEHR records, so we can also create a DateTime object from BEHR records
    - Already can grab date from BEHR filename, just need to grab the nearest hour (rounded up) from the MATLAB variable "TIME"
    - Will need to calculate time if this is stored as seconds since a certain reference date
- Planning to first enact temporal alignment first (according to BEHR records), and then spatial alignment for WRF-pixels and BEHR records.
- Need to familiarize with working w/ MATLAB files to extract out the BEHR data records
    - Make a list of variables that exist within these MATLAB files for future reference

Code reference:

```
import os.path
os.path.filename()
self.actualhr = ref.sec - self.sec
import datetime, calendar
```

3/15/19 – Ongoing plan:

Still need to collocate the pixels by date and location

Should be able to grab hour from BEHR MatLab files (in numpy arrays)

Then try to insert data from BEHR and WRF-Chem files into separate dataframes.

The "join"-ed dataframes will be formatted as the following:

- ✓ Multi-Index order: DateTime, Location, Behr-pixels

Scikit-learn can be used to perform regressions and maybe even neural networking.

Time: seconds since Jan 1, 1993

[took a long break from research due to midterms and Spring Break]

4/12/19

Goal for the next week:

Figure out how to:

- Grab hour from the OMI files (figure out grabbing the Time array from Matlab file)
- Grab latitude and longitude from OMI and WRF-Chem files (to compare geolocation)

Then, after we can successfully grab these things…

1. Scan through each OMI file, through each swath (out of 4), which has 258 grids that it scans for each record before moving to the next full scan, and look at each scan (there are 60 scans – one per grid).
2. Then, match the time of the grid (DateTime object) to the filename of a (or several) WRF-Chem files (which tells us the DateTime as well). Basically, we're aligning the times of the OMI file scan to the time of nearest WRF-Chem files, to the nearest hour.
3. After this, we can check which WRF-Chem file has the correct corresponding geolocation to the OMI file. Our interpolation method comes in handy here: we are trying to see which WRF-Chem record lies within the 4 corners of the OMI scan.
4. Then in the dataframe: we can place the id (maybe index) of the WRF-Chem record for reference. This allows us to match all the corr. WRF-Chem records to the OMI records, without creating huge dataframes, allowing faster retrieval of WRF-Chem record information, and allowing for easier modification of this dataframe (won't need to copy all of the WRF-Chem record information over and over).

Today, I completed grabbing the "Time", "Latitude", and "Longitude" arrays from Matlab files. I understood some more about what we are going to need to do to align WRF-Chem and OMI files (resulting in the above explanation of our next steps).

Next steps for aligning the hours:

- Calculate hour from the OMI "Time" field in seconds since Jan 1, 1993; times in UTC
- Round to nearest hour when aligning OMI and WRF-Chem files

Next Time:

So, I ended on starting to calculate hour using the OMI time field of total seconds since Jan 1, 1993. Check email for Chi's function, which exactly tackles this problem (uses calendar module to avoid the issue of accounting for leap years). Reference this to code for next time!

Also: Need to make sure to NOT use "Longitude" and "Latitude" fields. We need to use the Corner longitude and latitudes for OMI pixels, to allow for checking if WRF-Chem pixels belong to these OMI Pixels. **MAKE SURE TO GRAB ARRAYS OF SIZE 4, AKA CORNER LONGITUDES AND LATITUDES FOR OMI!!!

4/26/19 -

Successfully calculated the hour for OMI records using the OMI time field and calendar module.

This means that for now, I've completed the first 2 parts of my pseudocode.

***Pseudocode:***

1. Extract the DateTime from WRF-Chem and BEHR files
2. Extract the Location from WRF-Chem and BEHR files

However, I ran into some issues with how the data is stored in BEHR files. The BEHR file is a Matlab file, which stores its records in the Data field, which contains 4 swaths, aka 4 scans of the US, per day.

This means that the data structure we're working with is a nested array, where 1 level down is the swaths, and the next level down is the records themselves. We want to avoid the hassle and messiness of having to index into swaths and then individual OMI scan records.

For next time:

Make a function to "flatten" the OMI files [Data] 4x1 array into a 1D array with thousands of records in an OMI file.

First try for loops, then use map or apply functions to save time.

5/3/19 -

Flatten NumPy function for MatLab file matrix?

Or copy swaths 3 times into 1 array

Make a function that prints if we can print the alignment for 1 day – geolocation coordinates and time for BEHR and WRF-Chem files

First match 1 BEHR record with its corresponding WRF-Chem file records

OMI – satellite sensor instruments

NASA has a standard product (sp in the code) from OMI – aka the data they grab from their satellites

BEHR – modification of the NASA product by incorporating NO2 profiles; it's somewhat based

VCD and SCD of NO2 columns

AMF is used to convert SCD to VCD

NASA has their own AMF, and we use our own AMF model. We use their SCD to convert to VCD, and implement some corrections too. We use WRF-Chem to produce NO2 profiles from their SCD and easily compute to VCD.

So, we're trying to use our WRF-Chem model inputs (BEHR-product) and correlate to its outputs (BEHR file outputs "BEHRAMFTrop" and "BEHRAMFTropVisOnly").

AMF is what we want to produce.

My current job:

Aligning WRF-Chem input data to the BEHR data format bc SCD and SW are stored in BEHR. All three of these are used to calculate AMF (SCD, SW, and NO2 profiles produced from the WRF-Chem model)

Note: WRF-Chem model inputs are stored in the dataset that I have for WRF-Chem. We are grabbing the SCD and SW from the BEHR files.

We are trying to align all of these to predict the AMF calculations.

Eventually, we want our ML data model to replace the WRF-Chem model and produce the VCD calculation by correlating the SCD and SW from BEHR and the WRF-Chem inputs with the corresponding AMF outputs. We want to produce AMF data from our ML model bc VCD is an easy calculation from AMF.

NO2 is a pollutant itself; VCD (vertical column density) gives an indication of the surface pollution.

NO2 is also a precursor that leads to the production of other air pollutants, like particulate matter and Ozone.

Ozone itself is a greenhouse gas. It protects us from too much UV radiation from the sun, but too much ozone near the surface is bad. It is a greenhouse gas (absorbs and reemits radiation near the ground and increases temperature), and it harms our health bc it's a poisonous gas.

NO and NO2 chemically combine to form NOx, a family of gases.

NOx + VOC + Heat and Sunlight => Ozone

LOOK at the EPA website to find out more about Ozone pollution, NOx, etc….

(There's a research paper on this!) VCD can be used to calculate surface NO2 pollution; could perhaps utilize to map NO2 pollution over time in geographical areas.

Chi drew a diagram to explain our project task again before we continue for the summer.

[insert diagram here]

---

Might use BOTH or EITHER TiledCornerLongitude or FoV75CornerLong/Lat variables in the BEHR files.

Need to decide this with Chi, see which is better/ what's the difference!

---

6/3/19 – Today I start working for the summer – Day 1 of Paid research 😊

Goal for the week: Check that we collocate the data correctly

Read the paper from Chi: "*Application of random forest regression to the calculation of gas-phase chemistry within the GEOS-Chem chemistry model v10*" by Christoph A. Keller and Mat J. Evans

- All of Section 2: 2.1 – understand chem model, 2.4, 2.5, 2.7
- Section 3
- Section 4

Read the paper from Ron: *"A neural network radiative transfer model approach applied to TROPOMI's aerosol height algorithm"* by Swadhin Nanda, Martin de Graaf, J. Pepijn Veefkind, Mark ter Linden, Maarten Sneep, Johan de Haan, and Pieternel F. Levelt

- https://www.atmos-meas-tech-discuss.net/amt-2019-143/

Work today:

Was able to successfully concatenate 2d arrays from the 4 swaths into 1 large 2d array. This was usable for 2d variables. I finally was able to "flatten" 2d variables, as I was working on before, and almost finish my intended function "extract_var_1d".

Worked w Chi to figure out how to deal with 3d variables (such as the data for corner longitudes and corner latitudes. These are stored as 4 groups of longitudes and latitudes. So dimensions were 4x75x60 for a large "flattened" 3D array) We are choosing to "stack" or concatenate on the column axis (aka axis=-

1), so that we can do this consistently for higher dimension variables. Otherwise, when stacking on the row axis, it might change as it did for 3D (from axis 0 to axis 1) and we would have to find the row dimension for higher dimension data. ** We are assuming that the column axis is always going to be the last axis.

Reference my notebook to see what "stacking by column" (aka concatenating on the last axis) looks like. I don't need to like the data structure as much as the machine just needs to understand it. Be careful when accessing this data in the future because I need to remember what it looks like, if trying to grab certain coordinates out.

For next time:

Re-create my "extract_var_anyd" function using Chi's code, basically just being inspired by the axis=-1 part.

Make sure it works for both 2d and 3d array data. Note to self: I can keep my 2d array part to make it quicker. Check the dimension of the data, then choose which code part to run: if 2d, use my concatenate function that I finished; if 3d, use code inspired by Chi's stacking and appending.

6/4/19 – Decided to concatenate using the **row-axis**. This allows the data structure for the Matlab file variables to look the same, allows using the np.concatenate function, and is easy to use for 2d and 3d variables, which are the most useful variables for our data model. I was able to generalize this into a method that works for most variables called "extract_var_anyd."

I figured out how to identify the row axis and utilize np.concatenate and list comprehensions to allow efficient extraction of variables of any dimension* from the nested swaths data structure. (*We are excluding 1xn variables, text variables, and variables that generally are not 2d and 3d data. May need to re-evaluate building the dataframes if these variables prove to be necessary).

Currently going to work on collocating the Behr and WRF-Chem geolocations, since now I am able to extract corner longitudes and latitudes for the BEHR pixels.

This is important for forming the multi-index of our dataframes: organizing by DateTime (to the nearest hour), then organizing by Geolocation (grouping WRF-Chem pixels to their corr. BEHR pixels).

Created the "WRF_extract_location" method which grabs coordinate for the WRF-Chem pixels' centers. Had to review opening netCDF files from WRF-Chem to grab the center longitude and latitude coordinates.

**New information:**

Learned some intrinsic details about the coordinates for our BEHR pixels in particular. Normal longitude and latitude variables come as a 795 x 60 array. Meanwhile, the FoV75CornerLongitude and

FoV75CornerLatitude variables come as 4x795x60 size arrays. The “4” comes from the Corner Longitude/Latitude variables containing 4 corners for every pixel. There are 795x60 OMI pixels that we need to collocate for the scan of the entire US. It takes 4 separate swaths for the BEHR satellite to scan the whole US, since it scans a certain area in a polar orbit over the US. The swath is the rectangular area that the BEHR satellite scans over the US. The resolution at which it scans each swath is 60 pixels across. The 258 and 21 dimensions in the swath arrays are the length of the scans (i.e. in each swath, the FoV75CornerLongitude array is either 4x258x60 or 4x21x60). This results in 795x60 total pixels to scan the entire US, aka 795x60 total pixels to collocate for each OMI file to WRF-Chem files.

An important thing to note is that the BEHR satellite actually only scans the US about once a day. This would imply that it scans certain locations around the same time every day (useful for double-checking data), and most locations only get scanned once a day. This is another reason for why we are collocating WRF-Chem pixel data to OMI pixels – since OMI pixels would not have enough data to match to WRF-Chem pixel data (which has information for every location at every hour), but vice versa would be guaranteed to have enough information.

When visualizing the scan of BEHR on a 2D map, it will not look quite rectangular but will look convoluted somewhat. This is because a 2D map distorts the view of an ellipsoid earth. Since the OMI satellite is a polar orbit satellite, sitting about 700 km above the surface of the earth in a Sun-synchronous orbit, it is orbiting and scanning its swaths almost longitudinally at an angle. So, the route of the scan looks somewhat strange on a 2D map, because a map shows the same degrees of longitude at the equator much closer together at the poles. This result in the BEHR satellite route having an hourglass shape – covering a seemingly wider area at the poles than at the equator, when in fact, the swath is always the same width (divided into 60 pixels, scanning approximately 2600 km in width). One interesting fact when looking at the OMI specifications in the paper is that the spectral measurement of scans is 780, so it measures 780 wavelengths.

Chi told me that I can find more information about how OMI satellites work and give us the BEHR product are in the following papers.

- *The Berkeley High Resolution Tropospheric NO2 product* by Joshua L. Laughner, Qindan Zhu, and Ronald C. Cohen https://www.earth-syst-sci-data.net/10/2069/2018/essd-10-2069-2018.pdf
- Ozone Monitoring Instrument (OMI): Data User’s Guide https://acdisc.gesdisc.eosdis.nasa.gov/data/s4pa/Aura_OMI_Level2G/OMTO3G.003/doc/README.OMI_DUG.pdf
- OMI Instrument Characteristics: https://projects.knmi.nl/omi/research/instrument/characteristics.php?tag=full

**Next steps:**

Plan to collocate at least 1 variable of OMI and WRF-Chem pixels, aka map WRF-Chem pixels’ data into a file formatted and indexed according to the OMI layout (and matching DateTime and Geolocation).

It would be great if we could provide some visualization of this collocation. For example, either a map of some WRF-Chem data matching the OMI layout or a graph showing some regression. We shall see next week how it goes!

For future use: Consider using the Numba library for optimizing usage with NumPy arrays and data structures. Keep this in mind for running large calculations for machine learning with tons of climate data; may need to use this library among other computing tricks to optimize computational runtime.

6/5/19 –

I should utilize my new *extract_var_anyd* method to correct the method for *BEHR_extract_full_date*. It allows easy extraction of my "Time" variable, so that I can easily return a numpy array. I already performed the extraction of arrays; use it here!

**Work today:**

Replaced the "BEHR_extract_date" and "BEHR_extract_hour" methods. Found out that I had not fully finished the "BEHR_extract_full_date" method, bc previously, I had run into issues w/ indexing into swaths for the "Time" variable in the BEHR files. Was able to utilize my "extract_var_anyd" method to grab the flattened "Time" numpy array and now, my "BEHR_extract_full_date" method returns an array of datetime.datetime(yr, month, day, hr) instances.

First lunch w the Cohen research team! Met Leah, the other undergrad working on BEHR, Catherine, and others. I found out that the Cohen group has 4 projects of different interests: BEACO2N, BEHR, LIF – work on trees and absorption of NOx by planting them, and some lab work that I can't remember.

Examined how to extract location from BEHR files. The structure of the corner longitudes and latitudes' arrays are 3d and I must figure out how to team up my previous geolocation interpolation method to take in or accommodate these corner array structures. More details about this can be found in my Jupyter notebook "Danny_Siu_BEHR_Model_v2".

Options:

1. Make the "BEHR_extract_location method" to massage the corner lon/lat arrays from BEHR files into 2 4-element arrays

    Feed this into the "interpolate_WRF_to_OMI" in order to collocate BEHR pixels w WRF-Chem pixels

2. Change the "interpolate_WRF_to_OMI" method to loop thru the BEHR "CornerLon"/"CornerLat" variable arrays.

    Must be able to index into the Lon/Lat arrays correctly.

    Will require changing the "interpolate_WRF_to_OMI" method and not just assuming that it works.

Consider using some numpy array indexing function that can take the corresponding index element from multiple arrays, or will have to make this function myself.
https://docs.scipy.org/doc/numpy/reference/routines.array-manipulation.html - considering the split functions in particular.

**For next time:**

Start reading the research papers to understand more about OMI and the BEHR research.

Choose the option for extracting location from BEHR files. Currently, must try and figure out how to easily collocate the BEHR Pixels w/ WRF-Chem pixels by geolocation, since the corner longitudes and latitudes for OMI are stored in 4x795x60 size arrays.

Something to consider: Might want to find out the difference between TiledCornerLongitude or FoV75CornerLong/Lat variables in the BEHR files:

Which one is better suited for our purposes??

6/6/19 –

Trying to think through how to loop the CornerLon and CornerLat arrays in the OMI files.

Decided to read the "OMNO2 README Document: Data Product Version 3.0" paper to get a better idea of how the OMI data is gathered and understand the variables that I'm dealing w much better.

There are a lot of good clarifications that I discovered:

- OMI is the Ozone Monitoring Instrument, which is used on NASA's Aura satellite to detect NOx column levels. The visible wavelength (VIS) detector on OMI is used to gather NO2 column measurements. The Level 2 (L-2) NO2 product (OMNO2) includes stratospheric, tropospheric, and total columns.

As communicated to Howie, this is my current understanding for what I'm doing and perhaps a good summary for my current work for the summer, continuing from the past semester:

My research group focuses on air pollutants, generally dealing with atmospheric science data.

The paper I sent you is about the OMI instrument on NASA's Aura satellite, which has produced the data that I have to work with. So, I'm trying to understand some of the variables that it's measuring and how it's scanning by reading this paper: https://aura.gesdisc.eosdis.nasa.gov/data/Aura_OMI_Level2/OMNO2.003/doc/README.OMNO2.pdf?fbclid=IwAR1p1Jg5xGvSkact79GIGq02A1RfQ2KCS0LXKYiCmVGdzBWFAVoo8khve3c.

They use this computational climate model, coupled with chemical reaction computations, to output NO2 measurements (among others), but it takes lots of expensive computational power to run this model.

I'm trying to create a data model using ML, which uses the same inputs and can hopefully predict the same outputs as this computational climate model by correlating the inputs to the outputs and making unseen relations using ML. This could replace(!) their model if it's good enough, but for now, it's just for making good predictions and proof of concept.

**Things to consider:**

Still need to understand the difference between “TiledCornerLon/Lat” and “FoV75CornerLon/Lat” variables in the BEHR files.

Maintaining Data Quality: (page 12-13 of the “OMNO2 README Document: Data Product Version 3.0” paper)

There are fill-values for certain data due to the Row Anomaly and the Zoom modes. Some data is missing due to the orbits with measurements done in Zoom mode. ****We can see which is good data by selecting only data for which the least-significant bit of VcdQualityFlags is zero, indicating good data.**

**Next time:**

Still need to experiment w/ how to loop the CornerLon and CornerLat arrays in the OMI files.

Continue reading the “OMNO2 README Document: Data Product Version 3.0” paper to get a better idea of how the OMI data is gathered and understand the variables that I’m dealing w much better.

6/7/19 –

Read BEHR documentation and worked on figuring out work details.

6/10/19 –

Figured out how to make my life a lot easier for using the longitude and latitudes!

I can actually just take the max and min longitudes and latitudes of the BEHR pixels and put that into an array, instead of putting all 4 corners into an array and calculating this same measurement for the “interpolate_WRF_to_OMI” method.

Maybe I’ll implement this later

Success!

Finally, I was able to create a method, “BEHR_align_location”, to align the geolocations of WRF-Chem and OMI pixels. Given a WRF-Chem file and OMI file, I return a Boolean numpy array, which shows whether the single WRF-Chem pixel given matches in geolocation with the corresponding OMI pixel.

- I received help from Chi on how to index so that I get the same corresponding element in a 3d array, which allowed easy retrieval of the lon/lat coordinates for 4 corners of the OMI pixel.

I can use this method on all the WRF-Chem and OMI pixels that match in DateTime first.

**Next steps:**

Start looping through a subset of WRF-Chem and OMI files and try to match them in DateTime. Then match them in Geolocation. Afterwards, create a dataframe w the main index as the OMI pixel, and other indices being the WRF-Chem pixel filenames that match w the OMI pixel in DateTime and geolocation.

Continue to read the OMI paper next time.

Figure out logging work hours by Friday (before biweekly pay cycle ends) – should use CalTime to log in my hours

Might need to send Ron my excel sheet w hours logged

Data Science things to consider (from Kevin):

Ask for resources to prep for DS position interviews – from career advisors maybe

PyCharm – a work environment to consider! It's an IDE that looks like Sublime and is color-coded.

Look at shortcuts , such as “””<enter> to make method comments on variables

Write python unit tests

Language information:

Stata – for social sciences

R – specifically for statisticians

C++ and Python are useful in general

Insight – Data Science boot camp where you pay them after you find a job (sim. to coding boot camps)

Learned about the ***sci-kit image library*** from Chi. Consider using in the future

6/11/19 –

I found out from reading the BEHR_NO2 file that the 'VcdQualityFlags' attribute of the OMI data had values 0 for good quality data. I checked and found that the OMI MatLab data had all good data, aka the 'VcdQualityFlags' attribute was set to 0 for the data. So I don't need to worry about parsing out bad data bc I am already using a “good quality” dataset.

**Fully describing BEHR by reading the BEHR_Readme document:**

“BEHR was initiated by Ashley Russell, who completed her Ph. D. in 2012. She showed that using high-resolution albedo and terrain data improved the OMI NASA Standard Product retrieval…”

Reviewed that BEHR requires 4 satellite products:
1. OMNO2 level 2 (the NASA OMI $NO_2$ product)
2. OMPIXCOR (a NASA pixel corner product)

3. Aqua-MODIS cloud product (MYD06)
4. Combined MODIS BRDF product (MCD43C1).

BEHR runs by:
Reading NASA files in MatLab:

1. It reads all relevant variables from OMNO2 _les into Matlab.
2. It reads pixel corner data from the OMPIXCOR product and matches it up to the standard product data.
3. It averages the MYD06 L2 cloud product data to each OMI pixel.
4. It calculates the BRDF albedo from the MCD43C1 BRDF parameters for each pixel.
5. It averages the GLOBE terrain data to each OMI pixel, converting from altitude to terrain pressure.

Recalculate AMF and Tropospheric Column – Key component of BEHR

1. It uses the TOMRAD look up table from NASA's OMNO2 product to generate scattering weights (a.k.a. box AMFs) for each pixel, but using MODIS albedo and GLOBE terrain pressure. This is done for the clear and cloudy cases.
2. Reads in $NO_2$ profiles generated from WRF-Chem and bins them to each OMI pixel.
3. Calculates new AMFs by combining the WRF profiles and scattering weights.
4. The new AMF is then applied to the tropospheric slant column, found by multiplying the NASA AMF with their vertical column.
5. Finally selected data fields are gridded onto a 0:05 ° x 0:05 ° .

There is the option of creating maps of BEHR NO2 data.

MATLAB is used to run BEHR itself. Python is another language that the PSM gridding code and some utilities for BEHR are written in currently.

**Continued reading the OMI Paper**
(https://aura.gesdisc.eosdis.nasa.gov/data/Aura_OMI_Level2/OMNO2.003/doc/README.OMNO2.pdf)

Learned about:

The OMI NO2 Processing Algorithm: The OMI Level 1B data product contains calibrated earthshine radiance spectra I for each FOV. Earthshine radiances are divided by the solar irradiance spectrum
F to give a normalized spectrum R = I/F. The trace gas (NO2, H2O, CHOCHO) slant column amounts are retrieved from the normalized spectrum. The slant column amounts are combined with stratospheric and tropospheric air mass factors toobtain vertical columns.

**Notes on UCB employment to investigate more at home:**

- Getting out of the UCB Defined Contribution plan

- If not able to, make sure that I can recover retirement savings funds https://ucnet.universityofcalifornia.edu/news/2013/05/former-employees-may-claim-retirement-savings-funds.html
- Removing any extra fees from my paycheck
- Just check that I don't have extra deductions and that my information is correct
- Submit any needed forms to supply insurance info, exempt from automatic health plans, etc.

**Next steps:**

***Might want to take a look at parts of the BEHR code because they have also performed alignment between WRF-Chem and each OMI pixel! Check out what they do, and see if it helps me in aligning my pixels together. The BEHR_Readme doc specifies that they do this – it is highlighted in yellow above ^.

6/12/19 –

Making a list of the papers that I was given to read by Ron and Chi, as well as some I found to be useful:

(Unread) Ron - Manuscript under review about neural networking for aerosol retrieval from hyperspectral imagery: https://www.atmos-meas-tech-discuss.net/amt-2019-228/

Citation: Mauceri, S., Kindel, B., Massie, S., and Pilewskie, P.: Neural Network for Aerosol Retrieval from Hyperspectral Imagery, Atmos. Meas. Tech. Discuss., https://doi.org/10.5194/amt-2019-228, in review, 2019.

(Unread) Chi – Paper about the GEOS-Chem chemistry model v10. From this data set we train random forest regression models to predict the concentration of each transported species after the integrator, based on the physical and chemical conditions before the integrator: https://www.geosci-model-dev.net/12/1209/2019/gmd-12-1209-2019.html

Citation: Keller, C. A. and Evans, M. J.: Application of random forest regression to the calculation of gas-phase chemistry within the GEOS-Chem chemistry model v10, Geosci. Model Dev., 12, 1209-1225, https://doi.org/10.5194/gmd-12-1209-2019, 2019.

(Unread) Ron and Chi – Manuscript under review about neural networking applied to TROPOMI's aerosol height algorithm: https://www.atmos-meas-tech-discuss.net/amt-2019-143/

Citation: Nanda, S., de Graaf, M., Veefkind, J. P., ter Linden, M., Sneep, M., de Haan, J., and Levelt, P. F.: A neural network radiative transfer model approach applied to TROPOMI's aerosol height algorithm, Atmos. Meas. Tech. Discuss., https://doi.org/10.5194/amt-2019-143, in review, 2019.

(Started) Me - OMNO2 README Document: Data Product Version 3.0

-describes the version 3.0 release of the OMI NO2 Standard Product, omno2, the version 3.0 release of the OMI NO2 gridded Level-2 (omno2g), and the version 3.0 release of the gridded omno2d product produced from it: https://aura.gesdisc.eosdis.nasa.gov/data/Aura_OMI_Level2/OMNO2.003/doc/README.OMNO2.pdf

Citation: written by the The OMI Nitrogen Dioxide Algorithm Team in September 2016, Document Version 7.0

(Finished) BEHR Readme Document – about the BEHR Product. The Berkeley High Resolution Retrieval is a high resolution NO2 retrieval based on the NASA OMNO2 product from the Ozone Monitoring Instrument (OMI) aboard the Aura satellite. This retrieval improves the standard OMNO2 product in several ways. This doc describes the current state of BEHR, including its file structure and a changelog.

Citation: Written by Josh Laugher in March 12, 2018

**Work today:**

Finish reading the OMNO2 Readme Document,

Take a look at the BEHR code for aligning WRF-Chem and OMI Pixels

Start to organize the dataframe by which WRF-Chem pixels align w OMI ones.

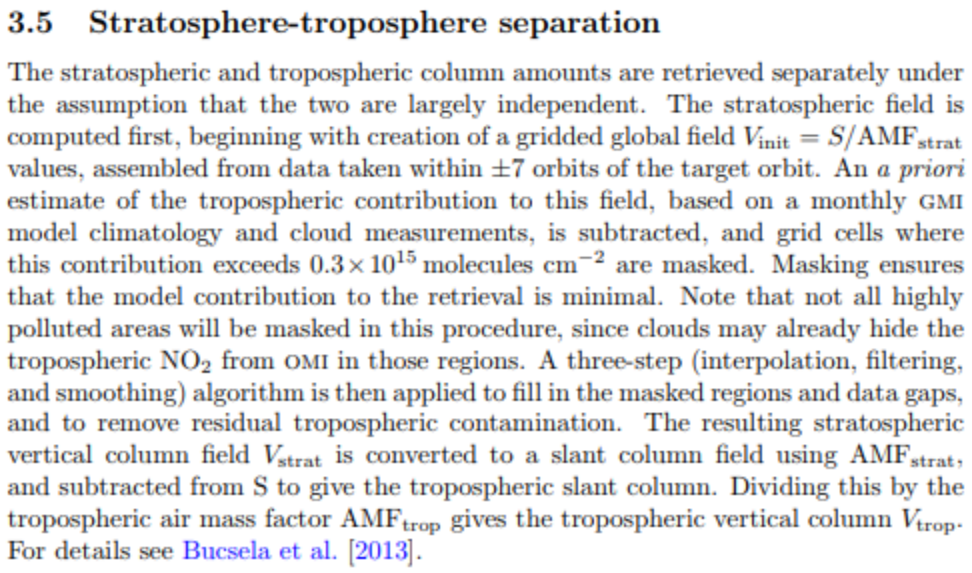
Find a sample of 2 files with the same date in the filename first!

Align by DateTime

Need to learn Bash!

Still reading the OMNO2 Readme Document, progress at page 17.

This passage below is what Chi previously explained to me as what I am trying to predict. I want to correlate the input of the BEHR model with the specific output of SCD (Slant Column Density) and then we can easily calculate the VCD (Vertical Column Density) by using our AMF calculations.



### 3.5 Stratosphere-troposphere separation

The stratospheric and tropospheric column amounts are retrieved separately under the assumption that the two are largely independent. The stratospheric field is computed first, beginning with creation of a gridded global field $V_{\mathrm{init}} = S/\mathrm{AMF}_{\mathrm{strat}}$ values, assembled from data taken within ±7 orbits of the target orbit. An *a priori* estimate of the tropospheric contribution to this field, based on a monthly GMI model climatology and cloud measurements, is subtracted, and grid cells where this contribution exceeds $0.3 \times 10^{15}$ molecules $\mathrm{cm}^{-2}$ are masked. Masking ensures that the model contribution to the retrieval is minimal. Note that not all highly polluted areas will be masked in this procedure, since clouds may already hide the tropospheric $NO_2$ from OMI in those regions. A three-step (interpolation, filtering, and smoothing) algorithm is then applied to fill in the masked regions and data gaps, and to remove residual tropospheric contamination. The resulting stratospheric vertical column field $V_{\mathrm{strat}}$ is converted to a slant column field using $\mathrm{AMF}_{\mathrm{strat}}$, and subtracted from S to give the tropospheric slant column. Dividing this by the tropospheric air mass factor $\mathrm{AMF}_{\mathrm{trop}}$ gives the tropospheric vertical column $V_{\mathrm{trop}}$. For details see Bucsela et al. [2013].

Hurray for better understanding of the OMI data! Next steps to take remain the same for tmrw essentially.

6/13/19 –

Finished reading the BEHR_Readme Document (at least most of the parts that seemed relevant for me to understand, and which I COULD understand).

**Plans:**

- Transfer my code to pycharm IDE.
- Get a subset of WRF-Chem and OMI files to collocate together.
- Use my functions to write code that starts organizing these files together and create dataframes w correctly organized indices.
    - -might take a look at how other data scientists arrange the data into dataframes, what tools they use, Bash scripting, etc.
- Start to organize the dataframe by which WRF-Chem pixels align w OMI ones. Use a test subset.

Chi mentioned to not average pixels' vertical data but to make sure that we average data that is in the same general area location-wise. So, collocation and averaging happens horizontally, NOT vertically. The vertical measurements, in their arrays or single measurements, will be manipulated in a different way or left the same.

Going thru the OMNO2 Readme Document is going to help me w BEHR bc the BEHR Product is based off of improvements made on the NASA OMNO2 product. So, the OMNO2 product will help me understand some of the variables used in the BEHR Product (because they are the same) and to see how variables are calculated and relate to one another.

**Tonight:**

Going to buy the Machine Learning and/or Deep Learning course from Udemy (while it's still $10)

Planning to use their videos to familiarize with Machine Learning basics, which were confusing in Data100, learn concepts and programming techniques and maybe even learn R.

Also, still figuring out how to separate my Python Jupyter Notebook into files to create my workspace in Pycharm. Need to set up my workspace, and then start collocating a subset of WRF-Chem and OMI files.

If it's not working or this process is taking too long, just code in Jupyter Notebook for now.

6/14/19 –

Setting up my workspace in Pycharm and transferring my code from Jupyter Notebook. This is the more traditional workspace in industry, so I should get used to using an IDE to run Bash, instead of Jupyter Notebooks, used primarily in education.

Buying and starting my online ML course from Udemy. Also, might begin watching 189 lectures from Spring 2019 taught by Jonathan Shewchuk.

Just enrolled in the Edx.org course: "Machine Learning with Python: from Linear Models to Deep Learning" - Provided by Massachusetts Institute of Technology (MITx)

10-14 hr/week for 13 weeks. Will see if I can cut down the amt of work needed, but it will definitely help in the future, even if I don't finish the course by the end of summer.

6/17/19 –

Remind Ron to submit my timesheet before the day ends. **Deadline for timesheet is today.**

Starting the online ML courses from Udemy and MITx (on Edx).

Continue creating my workspace in Pycharm.

Start collocating a subset of WRF-Chem and OMI files.

Started taking the Udemy ML course. It starts kinda slow with the introductions and the video content is not quite as edited as edx.org course lectures are. It's debatable how good the course and instructors are, but I might have to check out the sample code and go thru examples to see for myself.

Started taking the Edx ML course from MIT. **Need to finish HW0 before Wednesday.**

> Edx seems to have improved since I took the Python course 6 years ago.
>
> With HW0 and listed pre-reqs, I might have to get better at probability and stats in order to successfully understand and do ML. Also, Math53 might actually be necessary – I will either need to take it or learn it on my own now.

6/18/17 –

Continuing on the ML courses and setting up PyCharm environment.

Set up safety training time with Erin

Just got my account approved for my UCB Institutional Linux cluster. I now have access to Savio!

6/19/17 –

**Need to do:**

Start to learn bash scripting again

> Learn by looking at Chi's example job scripts.

Work on collocating a subset of BEHR and WRF-CHEM files

Finish HW0 for edx.org

See what I need to understand in ML research papers, look for relevant lectures in my online course

**Done today:**

Setting up my cluster account, so that I'm able to log into BRC (Berkeley Research Computing) Clusters

Completed lab walkthrough and safety training with Erin,

Managed to access cluster and also look at Chi's job script (in bash). Can make a copy and move to my own directory in the cluster, will need to edit the first 3 lines to match my account and specify which cluster I'm running on.

> Check the BRC website to find out how to submit jobs on Savio and for more tips on writing the Bash scripts

Finished HW0 for the MIT Edx ML course. Very frustrating bc idk some of the math, but managed to finish it in the end. **Make sure to finish the other assignments and videos that are due June 25!!** (next Tues)

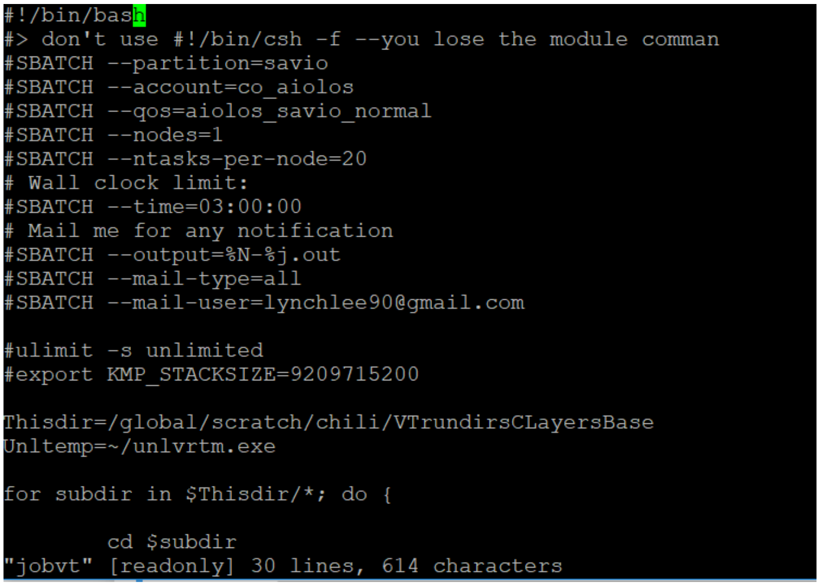
> Recommend to finish assignments at least 1 day early, bc the time zone for deadlines is midnight UTC.

**Work to do:**

Have to take "Chemical Safety Training" courses – 101, 105, 106

> Complete by the end of the week. Shoot Erin an email after finishing these courses!
>
> -can be found on the Cohen research group page>Safety tab. Also, check email for link
>
> Maybe check out the COC Ergonomics Matching Funds?...

Below is what one of Chi's job scripts looks like. I will need to tweak a few things for my scripts obviously, but this is the layout. Also, I need to look into downloading my own preferred text editor into my cluster directory; I could use nano or sublime! Or I could get good at using vim, like a true OG programmer lol. Look up tutorials on how to do this perhaps…



```
#!/bin/bash
#> don't use #!/bin/csh -f --you lose the module comman
#SBATCH --partition=savio
#SBATCH --account=co_aiolos
#SBATCH --qos=aiolos_savio_normal
#SBATCH --nodes=1
#SBATCH --ntasks-per-node=20
# Wall clock limit:
#SBATCH --time=03:00:00
# Mail me for any notification
#SBATCH --output=%N-%j.out
#SBATCH --mail-type=all
#SBATCH --mail-user=lynchlee90@gmail.com

#ulimit -s unlimited
#export KMP_STACKSIZE=9209715200

Thisdir=/global/scratch/chili/VTrundirsCLayersBase
Unltemp=~/unlvrtm.exe

for subdir in $Thisdir/*; do {

        cd $subdir
"jobvt" [readonly] 30 lines, 614 characters
```

6/20/19 –

Continued working on Edx ML course. VERY frustrating bug with installing packages for Proj 0. Ended up having to remove software and restart my computer.

Completed EHS 101 “Fundamentals of Lab Safety” course. Took a couple hours!

6/21/19 –

Plan:

In the morning:

Begin to re-learn bash scripting

Learn by looking at Chi’s example job scripts and BRC website tips

Work on collocating a subset of BEHR and WRF-CHEM files

In the afternoon:

Take “Chemical Safety Training” courses –105, 106

Watch ML lectures and do assignments for Edx ML Course; HW and Proj due next Tues!! (Finish by Mon)

**Work**:

Completed “Chemical Safety Training” courses –105, 106

Worked on Edx ML Course

6/24/19 –

Finish watching lectures and doing assignments (Proj 0 and 1) on Edx ML course due tmrw

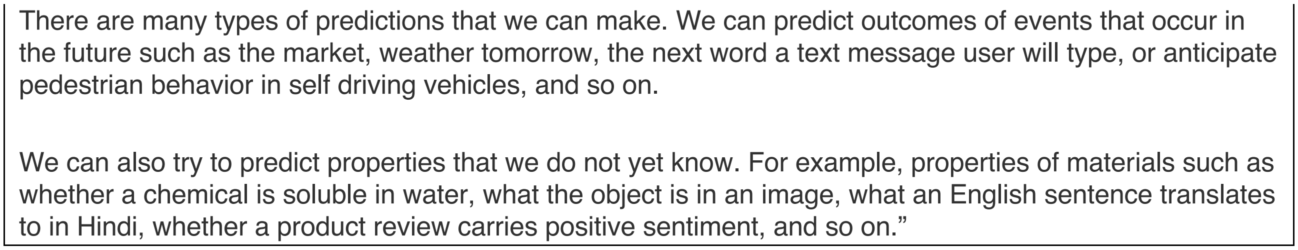
Purpose to take ML course:

Learning about Machine Learning is how I intend to grow intellectually. I want to see whether I want to pursue a career in Machine Learning or related research in the future, or to focus in other areas of Data Science. This is going to help me in my future required ML course for my major, and it is going to help me understand more about ML as I gain more repetition and familiarity with ML techniques and applications.

Begin to learn bash scripting and making my own job script

Good intro material from ML course:

“Machine learning as a discipline aims to design, understand, and apply computer programs that learn from experience (i.e. data) for the purpose of modelling, prediction, and control. We will start with **prediction** as a core machine learning task.



There are many types of predictions that we can make. We can predict outcomes of events that occur in the future such as the market, weather tomorrow, the next word a text message user will type, or anticipate pedestrian behavior in self driving vehicles, and so on.

We can also try to predict properties that we do not yet know. For example, properties of materials such as whether a chemical is soluble in water, what the object is in an image, what an English sentence translates to in Hindi, whether a product review carries positive sentiment, and so on.”

Hypothetical question: can I make an autocomplete better than Apple and Google?? Aka a better ML algorithm that applies their data better?

6/25/19 –

Finish watching lectures and doing assignments (Proj 0 and 1) on Edx ML course due tmrw

Begin to learn bash scripting and making my own job script

Start collocating the files for OMI and WRF-Chem; test on a subset of files

6/26/19 –

Continue w Edx ML course

6/27/19 –

May need to take another safety training course -  the LHAT “Risk and Safety Solutions” safety course…

Working on collocating the files for OMI and WRF-Chem; test on a subset of files

-regained access to the servers; through Kevin, made a user account w/ access to server 19. *Note: Must use AIRBEARS2 or LAN (Ethernet) connection to access the servers.

-write job script to collocate files locally

**To do:**

Write a script that outputs the file names of OMI files that contain OMI pixels aligning in geolocation and time with a certain WRF-Chem file.

Continue Edx ML course and maybe skip to learning sci-kit learn and how I should utilize random forest (if basing my ML algorithm off of the research papers sent by Ron. Look over those research papers and investigate whether random forest is useful for what we want to do.)

**Work Done:**

Edited the .bash_profile file in C:\Users\UserName to allow opening any text files in Sublime Text 3 using the terminal command “subl <text_file>”.

Using the Youtube tutorials by OMGenomics (Maria Nattestad) to learn Bash scripting. Link to her bash tutorial at https://www.youtube.com/watch?v=F-gskSl4pwQ.

Managed to create a test bash script successfully. Then I made the setup for my "collocation_script" in "C:\Users\chat2\Documents\Spring 2019\BEHR_Proj\BEHR_job_scripts". I will use this job script to test out my collocation of WRF-Chem and OMI files. It is already set up to run on the savio node utilizing parallelization.

Details on setting up job scripts for the Berkeley servers can be found here: http://research-it.berkeley.edu/services/high-performance-computing/running-your-jobs#Job-submission-with-specific-resource-requirements

6/28/19 –

Try my job script out! Need to at least test collocating a subset of WRF-Chem and OMI files.

If not done today, at least make good process on writing the Bash script for it.

**Process of writing my job script and figuring out how to run it on the servers:**

Talked w Chi and, through Qindan, I received the directory location of OMI and WRF-Chem files on the BRC servers (check my email). Should be able to run my job script on these files directly; however, Qindan recommended copying all of these files into my scratch directory, so that I can look and parse through these files more quickly. (Warning from Chi: might just copy over a year of data bc there may not be enough space to copy over that many files.)

*Distinguish between my Scratch dir, home directory, and ___ directory

**Chi's recommendations for job scripts:**

Set up a virtual environment, so that I can download my own packages needed for my code (i.e. NumPy, SciPy, … and also Sublime Text 3, which I'll use for editing my Python and Bash scripts).

**Kevin's recommendations on bash scripting:**

He sent a bit of code to my email, which activates the virtual environment in the server directories. I will need to change some of the directories which are "cd"-ed into, and modify according to my own directories and set-up on the server.

Use argparser to pass in files for running my job script,
Have intermittent printing of information (i.e. job start time, job end time, starting some process, looking at which files, etc.)
Construct my output files within my home directory or within my scratch user directory (or a mix of both)

**Workflow:**

Initialize a virtual environment in my home directory on my account for the server

Intro to virtual environments: https://docs.python.org/3/tutorial/venv.html

Installing packages using pip into my VE: https://packaging.python.org/guides/installing-using-pip-and-virtual-environments/

Install packages in a requirements.txt using pip install; this file will list all the packages needed to run my code and record the package versions as well.

Set up virtual environment directory in home directory, and place my whole project and code in the VE (virtual environment) directory; Basically, treat my VE as its own operating system.

So, my set-up is Home_directory > Virtual_environment_dir > All_my_proj_code.

Also, set up so that most of my code is in my Python file. Do 1% of code in Bash, the rest in the language that I'm familiar w. So, I can do the scanning and looping of files within my Python file maybe (using global) and utilize my functions within my Python file (that I've used in Jupyter notebook).

Read up on virtual environments to get familiar with it, and also to set it up properly in my account on the servers.

Add Kevin's snippet of code to my bash script to activate the virtual environment for my code to run in and use all my needed packages.

Transfer my Jupyter notebook code into many Python files; consolidate needed functions into one location.

Learn more about accessing files and directories in my Python code.

Copy the OMI and WRF-Chem files into my scratch directory (once I make my own).

**From my talk w Chi on data output:**

I will need to output some form of file w the OMI layout.

For each OMI file, I will have an output file w the same layout which contains a grid of OMI pixels. For each pixel, I have a list of WRF-Chem filenames which match it, and for each WRF-Chem file, I must specify which pixels in the WRF-Chem file match the OMI pixel.

> Note: for each WRF-Chem file, they should all match identically in terms of geolocation layout, so it's easier to work w since it's standardized organization (unlike OMI files' variable geolocations).

Chi recommended outputting an HDF or NetCDF file for this huge matrix output. I agree, bc I don't enjoy dealing w Matlab files, so that is NOT the option for me; however, if people want to keep all the data in Matlab format, they can easily read from HDF or NetCDF files into Matlab files.

Turns out that I do have a user directory on the server already!

My "scratch" space directory is in "global/scratch/ds46429/".

When logging into the server, I am automatically placed in my home directory ds46429.

My home directory will be where most of my code and the job scripts that I create will reside.

My scratch directory will be where all my data is (for temporary storage). So, my OMI and WRF-Chem files that I am working with will be here, as well as my input and output files.

Supposedly, the scratch space will be cleared after a month or two. So, if I want to keep my presumably large output files, we can move them into the "aiolos" data/file servers for long-term storage; my home directory will probably not have enough space for storing those large files. (I can talk w Chi and Qindan when it gets to that point.)

Soccer now!

7/1/19 –

Need to:

Copy over year 2014 for WRF-Chem data

Ask Kevin for what priority to set my jobs to, what to run in parallel, and other details about running job scripts and what I should submit to the BRC cluster

Work on edx ML course

Wasn't as productive today tbh, because I'm dealing w some stuff rn.

7/2-7/3/19 –

Resume and finish my copying of the "2014" directory of WRF-Chem data

Learned how to use the argparse library – allows for parsing input variables in the command line

-default arg is a string, can make args optional or not, can modify its description and data type

Documentation: https://docs.python.org/2/howto/argparse.html

Tutorial on usage: https://towardsdatascience.com/learn-enough-python-to-be-useful-argparse-e482e1764e05

Finished Lecture 2 of Edx ML course, going to review Lecture 3, HW1, Lecture 4, HW 2, and Proj 1

Don't worry about my grade so much; just learn the material!!

Also going to be reading the lecture notes from CS 189. It looks pretty good; there is a textbook of the background material that I need to know. Use this to prep for 189; go over this in the summer!!

7/8/19 –

Go over 189 material

Collocate my data files; work on creating my dataframe

Work on my bash script

Read the papers sent by Ron on neural networking, Chi on random forest regression, and ML applied to models

Download programs onto new laptop, get ready to transport work environment to new computer

For my data collocation:

Copy WRF-Chem and OMI files into my scratch directory for the year 2014

"2014" for WRF-Chem, ______ for OMI ???

Try to apply my bash script to take WRF-Chem file and OMI file as inputs (utilize arg parse for now)

Figure out how to create my dataframe output file (HDF or NetCDF file probably for this huge matrix output)

7/9/19 –

Set up virtual environment, and download my imported packages (write in requirements.txt)

This is the location of the BEHR and WRF-Chem files on the cluster for reference: (from Qindan's email Jun28).

BEHR:

```
/clusterfs/aiolos/laughner/SAT/BEHR/BEHR_Files
```

WRF–Chem:
/clusterfs/aiolos/laughner/SAT/BEHR/WRF_NO2_Profiles

## How to access the data on the cluster:

Directories (filepaths) containing the data that I copied over to my scratch user directory:

BEHR – "/clusterfs/aiolos/laughner/SAT/BEHR/BEHR_Files/us/daily"    (Two cds back from "global")

> These BEHR files are the same as those on server 19, in the "OMI_BEHR-DAILY_US_v3-0B" folder
>
> **There are 3553 BEHR files on server 19, but 2915 BEHR files in the cluster directory we referenced.
>
> From the filenames, we know that we have data from Jan1, 2005 to Dec31, 2014.
> When calculated, we should have 3652 files if having daily files from Jan1, 2005 to Dec31, 2014.

WRF-Chem – "/clusterfs/aiolos/laughner/SAT/BEHR/WRF_NO2_Profiles/Daily/us"

> This folder has data from years 2005 to 2014.
>
> These are the same files in "Z:\share-wrf1\BEHR-WRF\Outputs\us" – years 2010-2014, "Z:\share-wrf2\BEHR-WRF\Outputs\us" – years 2005-2009, "Z:\share-wrf3\BEHR-WRF\Outputs\us" – year 2011

Destination for my scratch directory – "/global/scratch/ds46429"

**How to access the data that I accessed:**

Go to SFTP Net Drive 2017 program to log into servers 19 and 45.

Go to Putty to log into my cluster account, and cd into the database on the cluster by doing "cd <above file paths for BEHR/WRF-Chem>"

**After talking to Chi:**

Stick to using year 2013 and on – the most recent data

For now, just use the 2014 folder for both WRF-Chem and BEHR files to generate the dataset that will train the model.

> **Keep in mind: We're also using some subset of data for training and test data. Not gonna use all the data.**

**Need To Do:**

After separating my Jupyter notebook code into Python files,

and copying over my "2014" files for WRF-Chem and BEHR files,

Set up virtual environment in my Home (user) directory, and download my imported packages (write in requirements.txt)

To finalize my job script, need to:

Cd back into my scratch directory to find files (bc I'm running my job script in a virtual environment in my home user directory in the cluster)

Utilize arg parse (to take in files to look for; maybe I can instead write the input as filepaths or do some conjunction of cd-ing and writing part of the filename to look for input files)

Utilize my Python files' code to use functions and collocate my OMI and WRF-Chem files

Need to figure out how my output file is generated!!

Going to output an HDF or netCDF file, but need to figure out details of what information I'm outputting, as well as the indices (bc I have many OMI records in one pixel, and many WRF-Chem pixels/records? per one OMI record).

Linux commands:

```
ls | wc -l  #Print out number of files and directories in current directory
```

```
dir  #Lists all files and directories in the current directory
```

```
dir *.exe  #Lists all files and directories ending in ".exe" in the current directory
```

For BEHR files, I am looking for data from the year 2014, so I look for corresponding filenames w this: "OMI_BEHR-DAILY_US_v3-0B_2014*.mat"

**Linux commands run to copy over the 2014 data for WRF-Chem and OMI:**

```
cd /global/scratch/ds46429

cp -R /clusterfs/aiolos/laughner/SAT/BEHR/BEHR_Files/us/daily/OMI_BEHR-DAILY_US_v3-0B_2014*.mat BEHR_2014/

nohup rsync -auv /clusterfs/aiolos/laughner/SAT/BEHR/WRF_NO2_Profiles/Daily/us/2014/. WRFChem_2014/ >logsync 2>&1 &
```

```
ls -l    #list all the details of the files
```

```
tail logsync    #check the "tail end" (last several lines) of the logsync text file
```

**Work done:**

Chose to copy over the OMI and WRF-Chem data from the year 2014.

Set up and created my virtual environment

~~Install gcp (so that I can finish copying the 2014 WRF-Chem data)~~

~~gcp -a /clusterfs/aiolos/laughner/SAT/BEHR/WRF_NO2_Profiles/Daily/us/2014/. WRFChem_2014/~~

Used rsync to copy over my WRF-Chem "2014" directory. The key is to start the job on the data transfer node. I can also run the data transfer in the background by using "nohup rsync -auv source dest >logsync 2>&1 &". Lesson: **Have to copy over files using the data transfer node**

Another option for an SSH program (which Chi uses): MobaXterm for PC SSH

Info on the BRC data transfer node and which commands/file transfer software to use for data transfer: http://research-it.berkeley.edu/services/high-performance-computing/transferring-data

**Still Need To Do for Next Time:**

Check that my "2014" directories for WRF-Chem and BEHR files are all complete and have copied over. (look inside my scratch directory)

Activate my virtual environment, then begin downloading my imported packages (write in requirements.txt) – important to figure out how this works

Separate my Jupyter notebook code into Python files.

To finalize my job script, need to:

Cd back into my scratch directory to find files (bc I'm running my job script in a virtual environment in my home user directory in the cluster)

Utilize arg parse (to take in files to look for; maybe I can instead write the input as filepaths or do some conjunction of cd-ing and writing part of the filename to look for input files)

Utilize my Python files' code to use functions and collocate my OMI and WRF-Chem files

Need to figure out how my output file is generated!!

Going to output an HDF or netCDF file, but need to figure out details of what information I'm outputting, as well as the indices (bc I have many OMI records in one pixel, and many WRF-Chem pixels/records? per one OMI record).

***Anytime I log into my brc cluster account, type this command to activate my virtual environment “my_venv”. Do this before any pip install commands and Python commands and running Python job scripts! (this is a Linux command bc the cluster runs on a Unix system)

```
source my_venv/bin/activate
```

**Other things I need to do:**

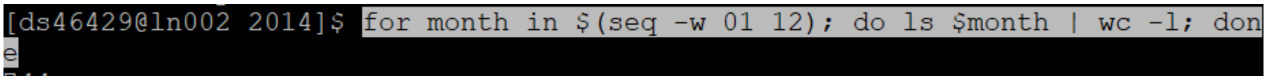
- Learn more about utilizing ML. Maybe instead of keeping up w the Edx ML course, I can opt to learn to use sci-kit learn and 189 material at my own pace.
- Figure out how to create my output file containing collocated OMI/WRF-Chem data.
- Read the research papers that Ron sent me

7/10/19 –

Double-checked that my “2014” directories for WRF-Chem and BEHR files are all complete and have copied over. (looked inside my scratch directory, and ensured that the file count was correct)

How Chi helped me to check the file count in folders:

for month in $(seq -w 01 12); do ls $month | wc -l; done



```
[ds46429@ln002 2014]$ for month in $(seq -w 01 12); do ls $month | wc -l; don
e
```

Note: currently my job scripts are in BEHR_Proj>BEHR_job_scripts

Trying to move all my Jupyter notebook files to a python file

I’ve moved all my methods into the Python file “wrfomi_collocation.py”. Double-check that everything is there and working properly later.

Currently setting up my PyCharm workspace: (PyCharm is a Python IDE)

PyCharm Quick Start Guide: https://www.jetbrains.com/help/pycharm/quick-start-guide.html

Still need to resolve library import errors. Going to be installing libraries in my “venv” virtual environment (already set up in my project workspace directory “Danny_BEHR_Code”)

**Next steps:**

Finish setting up my PyCharm environment. Ensure no other errors in editing my Python files in this project code directory.

Continue working on my job script and add the needed code mentioned beforehand.

(Reference previous entries (7/9/19) for my to-do list on this job script).

7/11-7/12/19 –

Somehow resolved my PyCharm environment library import issues. **JK, it's not resolved.**

Watched edx ML course Lecture 3: Hinge loss, Margin boundaries, and Regularization

Tmrw: I can continue working on ML course (lectures due today), reading research papers, and working on my bash script and creating my dataframe. I want to try and make my dataframe hdf file by the end of next week.

7/15-7/16/19 –

Organize my Chrome tabs and windows

Import libraries for my Pycharm environment

Work on my bash script

Plan out my output hdf file for my dataframe

Research into how other people generate dataframes from their datasets

The most relevant information about making my job script and anticipated code starts from **page 26**.

**Make sure to submit my job script w intermittent outputting (--verbose or sthg).

7/17/19 –

Resolve the issue w importing libraries, unless it doesn't matter bc it'll be run in the virtual environment anyways

Set up my virtual environment and import necessary libraries (sublime, numpy, scipy, etc.)

Talk to Chi about how to create my hdf file

Research into how other people generate dataframes from their datasets

Done today:

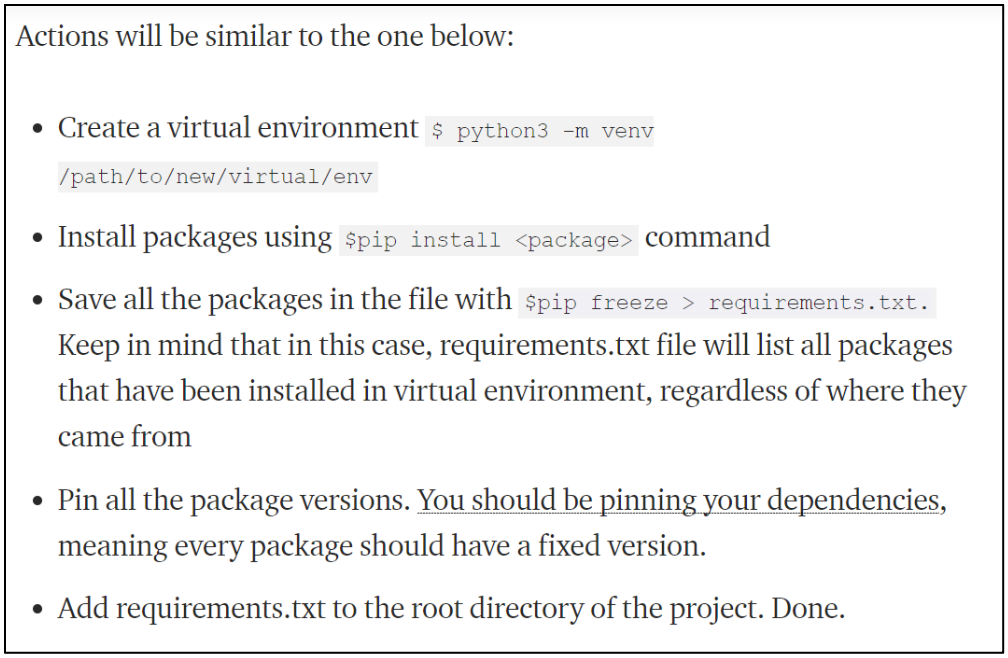
Managed to pip install necessary libraries into my "venv" virtual environment (activate venv using `. /c/Users/chat2/Documents/Spring 2019/BEHR_Proj`). Then saved all my packages into a file "requirements.txt".

Also, just copied over the "requirements.txt" file to my cluster home directory, ran the command to load all the packages listed there, and have now installed all my needed packages for my python file to load during my script runs. Set up my virtual environment for the cluster!

```
In my virtual environment on the BRC cluster server, activate the virtual
environment, make sure to just copy over this requirements.txt file, and run
pip install -r requirements.txt
```

This will ensure that all the correct packages are installed to be able to run my python file and Bash script on the server.

Anytime that I want to create a project and keep track of its dependencies and packages, just create a virtual environment for the project, activate it, pip install needed packages, and save everything to the requirements.txt file using "pip freeze". OR, activate a virtual environment, and download the needed packages using a requirements.txt file that I already have using the above command ("pip install -r requirements.txt").



Actions will be similar to the one below:

- Create a virtual environment `$ python3 -m venv /path/to/new/virtual/env`
- Install packages using `$pip install <package>` command
- Save all the packages in the file with `$pip freeze > requirements.txt.` Keep in mind that in this case, requirements.txt file will list all packages that have been installed in virtual environment, regardless of where they came from
- Pin all the package versions. You should be pinning your dependencies, meaning every package should have a fixed version.
- Add requirements.txt to the root directory of the project. Done.

Used scp to copy over the requirements.txt file from my laptop to the server

Also installing Sublime Text 3 into my cluster account home directory. Finish this tmrw.

Info to install on the cluster: https://research-it.berkeley.edu/services/high-performance-computing/accessing-and-installing-software

Sublime Text 3 website (need the tarball 64 version): https://www.sublimetext.com/3

Finish things more quickly by talking to Chi. Work quickly to make this dataframe; you're running out of time!

7/18/19 –

Installing Sublime Text 3 into my cluster account home directory. Finish this; **activate my virtual environment before doing anything!**

Info to install on the cluster: https://research-it.berkeley.edu/services/high-performance-computing/accessing-and-installing-software

Sublime Text 3 website (need the tarball 64 version): https://www.sublimetext.com/3

POSTPONED -- Sublime Text 3 download on cluster. Okay, might not be worth it to try and figure out how to download and run sublime in the cluster to open files. I'm just gonna use vim and nano to open files here and refresh on how to use it, OR I can just edit files locally anyways. If it comes to it later, I can ask Kevin why my installation didn't work – I was unable to run the "./configure" command, followed by "make", "make install". Here is a post that explains better: https://askubuntu.com/questions/25961/how-do-i-install-a-tar-gz-or-tar-bz2-file. Maybe I'll need this later when Kevin comes back. Another useful link from Sublime Text 3's forum: http://docs.sublimetext.info/en/latest/getting_started/install.html.

*Chi sent me an email today w subject title "argparse", which shows some of his code when using argparse for python files. Reference the code snippets here for later.

Been looking at argparse again and figuring out how to use it. Might need for my Bash script if I'm taking in arguments; we shall see.

**Next Steps:**

Figure out next steps for writing my Bash script.

Find out more about using argparse – sthg about using it in my python files, which will then be useful when I use Bash scripts to call python files.

To finalize my job script, need to:  (from 7/9/19 post)

Cd back into my scratch directory to find files (bc I'm running my job script in a virtual environment in my home user directory in the cluster)

Utilize arg parse (to take in files to look for; maybe I can instead write the input as filepaths or do some conjunction of cd-ing and writing part of the filename to look for input files)

Utilize my Python files' code to use functions and collocate my OMI and WRF-Chem files

Need to figure out how my output file is generated!!

Going to output an HDF or netCDF file, but need to figure out details of what information I'm outputting, as well as the indices (bc I have many OMI records in one pixel, and many WRF-Chem pixels/records? per one OMI record).

Basically, the biggest concern is figuring out how to generate my HDF output file and what I want it to look like. With a better idea of what output I want to make, and how to manipulate and collocate files using my Python file methods and the Bash script, I can hopefully have a more organized approach and idea for writing my job script.

7/19/19 –

Updated checklist - To finalize my job script, need to:

First activate virtual environment, so Python file/script has all needed packages imported

Cd back into my scratch directory to find files (bc I'm running my job script in a virtual environment in my home user directory in the cluster)

Utilize arg parse (to take in files to look for; maybe I can instead write the input as filepaths or do some conjunction of cd-ing and writing part of the filename to look for input files)

Utilize my Python files' code to use functions and collocate my OMI and WRF-Chem files

Need to figure out how my output file is generated!!

Going to output an HDF or netCDF file, but need to figure out details of what information I'm outputting, as well as the indices (bc I have many OMI records in one pixel, and many WRF-Chem pixels/records? per one OMI record).

7/22-7/25/19 –

Ask Kevin how to install new software on the cluster (in my case, Sublime Text 3); otherwise, just get used to vim or nano

Didn't end up figuring out how to install the x86_64 tarball executable of Sublime Text 3.

According to Chi and Kevin, I don't need to install a graphical text editor on the cluster; the cluster should only need to use vim or nano to view files and should be primarily used for running executables, not for editing files or scripts. If I actually ran Sublime on the cluster, I would need to also use a GUI to display the text editor.

So, we're scrapping the idea of installing Sublime on the cluster. Just activate the virtual environment and use the needed packages for my Python files.

Trying to learn the benefits of Python vs Bash scripts rn

Ask Chi about creating a Bash script or Python script

Reference and understand one of Chi's Bash scripts to create my own (make sure I know how to do the bulleted list below in my job script)

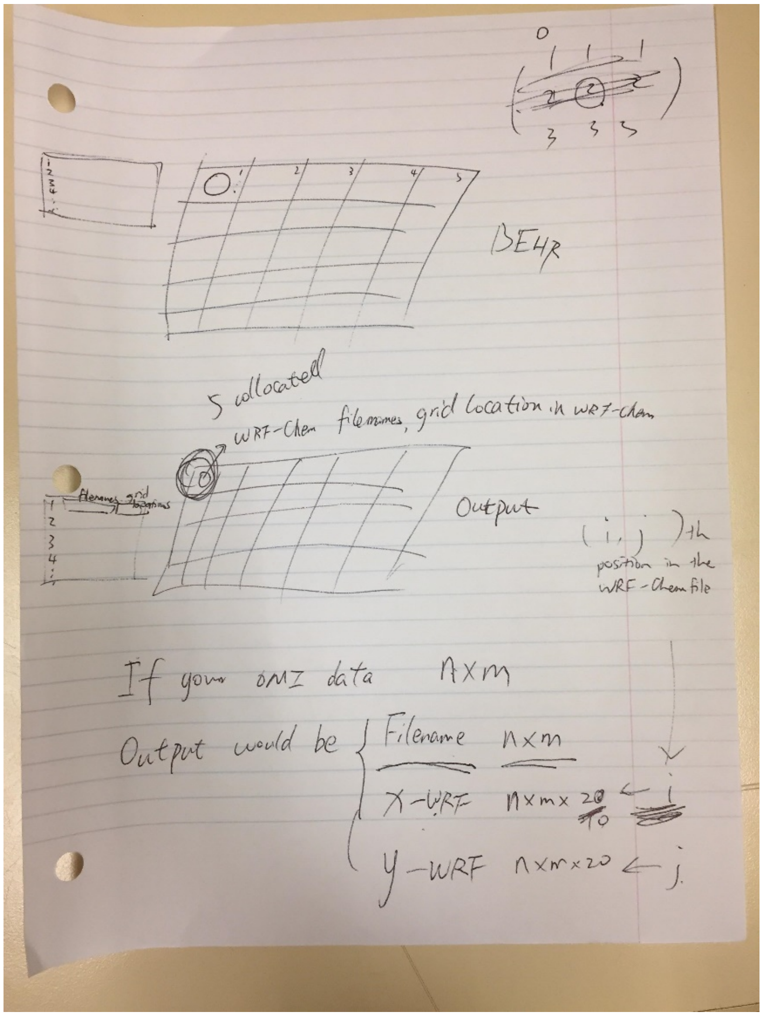
Talk about how to create my resulting HDF file

In my job script, need to be able to:

- Change directory into OMI and WRF-Chem directories in my scratch directory
- Loop over files in directories
- Use functions in my Python file
- Output an hdf file after collocating OMI and WRF-Chem files

Create a Python script for my Bash script to call upon and execute…

I think I will perform my looping within the Python script (when performing the file record collocation) using the OS module. The Bash script can just be used to specify the job details for the cluster and specify what Python executable to run.



If your OMI data n×m

Output would be { Filename n×m
X-WRF n×m×20 ← i
Y-WRF n×m×20 ← j

Chi discussed some details about my output hdf file in the above picture.

Should just have to first arrange by DateTime (to the nearest hour), then include grid locations in the WRF-Chem file (aka, the i,jth spot in the WRF-Chem gridded file). This will all be in the same data format as the OMI file hopefully.

I'm getting so confused w how this output will look like, especially bc these are multidimensional arrays that we are collocating and arranging.

I still need to figure out how to write this in code – first by creating the hdf to have the same dimensions as the OMI files, searching through directories and the correct filenames, and then parsing through the files to find the correct indices of where the record is in the WRF-Chem gridded file. Need to use my Python functions for this part and create some new functions.

Each HDF will correspond to an OMI file; I will basically generate many HDF files during this process of collocation.

I feel so lost and this output is just so hard to visualize. I really hope this works.

7/30/19 –

Aim for finishing my dataframe by this week!!

Also, gotta practice for coding challenges every other night at least!

### Work done:

Created most of my Project log summary for summer 2019 to gauge my progress, as well as assess what needs to get done in these last 2 weeks that I work for the summer.

Planning my work for the last 2 weeks of summer

Edited my Bash script to correctly activate the virtual environment and ensure all installed packages are accessible to my code

### For this week, Jul 30-Aug 2:

Backup and reset my Dell laptop (might take time to reinstall settings on my computer)

****Finish much of my job script in Bash in order to create some output HDF files****

When I run into issues, talk to Chi and figure out how to correctly generate HDF files

### For next week, Aug 5-Aug 9:

If all goes well, figure out the automation process of generating these HDF files

Divide my data into training and test datasets,

Review the ML process when planning for the next semester

Hopefully submit multiple jobs to the cluster to generate HDF files, which I can use for the fall semester research and utilize for my ML training set

To-do: Finish Proj log summary, Continue editing Bash script to call my Python functions

7/31/19 –

In order to attempt and solve some of my laptop's software issues, I backed up my laptop to an external hard drive, reset my laptop to the factory image, restored my files, and re-downloaded the programs that I use.

8/1/19 –

Updated my laptop to resolve some software and OS issues hopefully.

Reviewed my written notes to review what I've discussed w Chi and to try and better understand the output HDF file that I want to produce. I really need to write this out on paper and write some pseudocode to visualize the layout of these files.

Work hard on code till 4

8/5/19 – Still kind of stuck on what to do for my OMI and WRF-Chem pixel collocation.

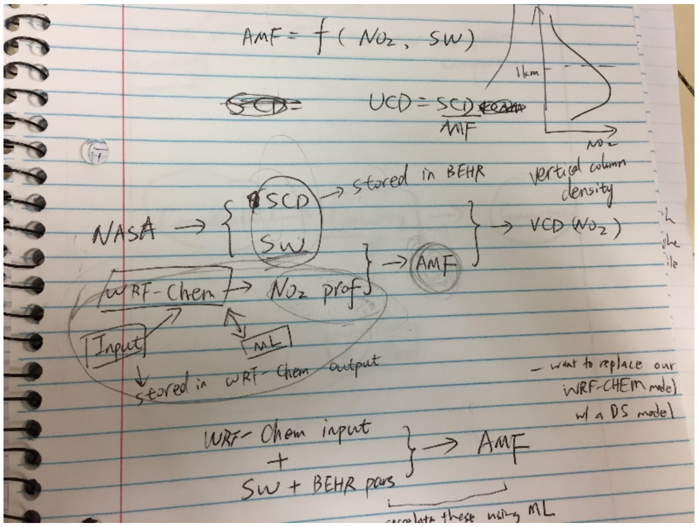
Basically just wrote some more pseudocode for this part, and organized my countless notes and tabs to clear up some of my laptop space and clutter and make myself more efficient.

8/6/19 –

Chi just told me some amazing news. **I don't have to collocate the data for BEHR and WRF-Chem pixels anymore!**

The focus of my ML work can instead be done simply by focusing on the data found in the WRF-Chem files! The WRF-Chem outputs have inputs stored in the files, as well as the NO2 profile outputs. If I can use ML and train on the inputs to predict the NO2 profile vectors, then that would be very instrumental to helping replace the need for the WRF-Chem model runs, which is what we wanted in the first place. Right now, I can basically try and find any statistical relations between the WRF-Chem input variables and the NO2 profiles, perhaps trying a multivariate regression among other things. I can also begin performing Principal Component Analysis (PCA) to determine the most influential variables on the predictions of the NO2 profiles.

The only concern is whether or not we can successfully predict the NO2 profiles, which are vectors, rather than the AMF, which is a number, which is what we would've predicted if the OMI-WRF-Chem pixel collocation was not so difficult to perform and grasp for me.



To the left, the circled portion of the diagram (which shows the transformation of data from NASA and WRF-CHEM into the VCD NO2 product) displays the process of what I want to focus on now. Instead of trying to predict the AMF using OMI data and the inputs from WRF-Chem, we're going to try and just predict the vector of NO2 profiles by training the model on the WRF-CHEM inputs and NO2 profile outputs. This can hopefully eliminate the need for computationally expensive WRF-CHEM model runs.

Yeah, anyways, this is great news! I can basically begin by just using the WRF-Chem files to analyze the pixels within them and train on the datasets already found within them, rather than having to create a database before I can begin training the model.

The idea with this new approach is that I can start using my knowledge of machine learning to get started on this part of the research, and not be so concerned about the details of aligning the two datasets, which may not really be needed.

**Other exciting news:** Professor Cohen might hire some other graduate student to help with the Machine Learning segment of predicting NO2 profiles using WRF-Chem inputs. Also, Josh Laughner (the guy in charge of answering questions about the BEHR dataset!) expressed some interest in learning machine learning and becoming involved with this work himself! So, I can maybe finally get some mentorship with people who are more familiar with machine learning, or can work with me directly and help me do some of this work more quickly. I can hopefully get something completed in the fall then and have something to show for all my efforts! It would definitely help me learn more quickly and feel like I accomplished something with actual results. Yessss.

Re-downloaded Panoply and Java 8 in order to open WRF-Chem files (netCDF I think). Just a reminder, Panoply can open HDF, NetCDF, and GRIB files.

**Work to be done now:**

Use sci-kit learn and matplotlib to perform regressions and plot data visualizations to determine what WRF-CHEM inputs correspond strongly to NO2 profile vectors.

Eventually start trying to use PCA or some method of feature selection to see which predictors are the best indicators for predicting NO2 profiles correctly.

Chi's email w subject "" has 3 variables not listed in WRF-CHEM that I should use as WRF-CHEM predictors for my model. Add this to my training dataset and see if they're related to the output.

This means… I should move back to Jupyter notebook! Lol. I can more easily generate plots there and test different variables. Also, according to Duncan, Jupyter notebooks are very common and suitable for data science research anyways; PyCharm is moreso for industry work.

It seems like random forest algorithm is a good place to look for my ML approach as well, besides feature selection, PCA, and random plotting of variables. It is used for predicting categorical data.

I should look more at Chi's papers on the use of the random forest alg. Also, I should learn more ML formally, review Data 100, look at what sci-kit learn has to offer, see what regressions I should use, and practice more on this classification prediction problem. Oh! And check out EdX or Udemy on how these things work!

8/7-8/8/19 –

Reverting back to my Jupyter notebook code for the BEHR ML Project.

Reviewing code that I made for my prior classifier model and regression models for Data100

Beginning to create visualizations and plots to try and understand relationships between variables and NO2 outputs

Going to use feature selection techniques as specified in the article, “Feature Selection Techniques in Machine Learning with Python”, found at https://towardsdatascience.com/feature-selection-techniques-in-machine-learning-with-python-f24e7da3f36e.

Work done:

Reading articles from medium.com and towardsdatascience.com, which discuss machine learning more in detail.

- Choosing which ML model to use: https://towardsdatascience.com/which-machine-learning-model-to-use-db5fdf37f3dd
- ML definitions: https://medium.com/technology-nineleaps/some-key-machine-learning-definitions-b524eb6cb48
- Techniques to learn ML: https://towardsdatascience.com/6-techniques-which-help-me-study-machine-learning-five-days-per-week-fb3e889fad80
- Developing a ML Model from Scratch: https://towardsdatascience.com/machine-learning-general-process-8f1b510bd8af

8/9/19 –

Finished my “BEHR Project Log Summary Summer19_Danny Siu” document. Sending this to Ron today! It details what I’ve been working on from week to week this summer – roadblocks I ran into, things that I needed to clarify, documentation I needed to read and understand, and substantial steps that I took in writing my Bash script and setting up my cluster account for script submission.

My iPhone froze last night. Trying to restore it; otherwise, going to have to go to the Apple store tmrw.