

End of Internship Report

Margaret Trimpin

Towson University

Code 610

10 December, 2021

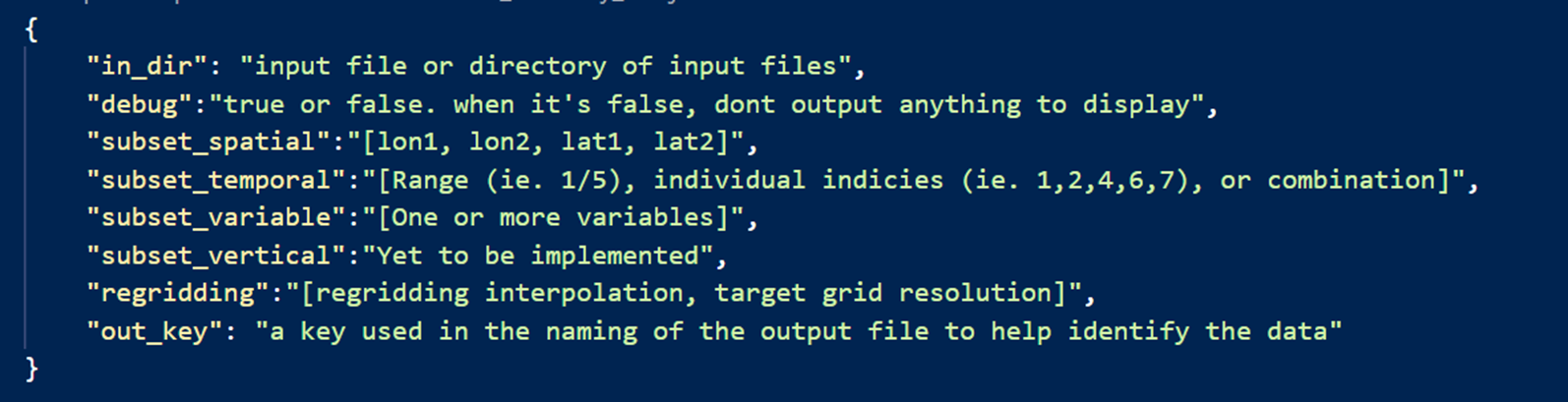
# **1.** **Accomplishments**

Over the course of my internship experience, I was given the task of developing a subsetter and regridder for the Goddard Earth Sciences Data Information Services Center (GES DISC). This tool was to be able to carry out the same processes as the current tool in use, the Level 3/ 4 Regridder and Subsetter (LEARS). These processes include spatial, temporal, and variable subsetting, as well as regridding based on a variety of interpolations and resolutions. Once the new tool was able to do these processes, I had to verify that it is superior to the old tool, whether it be in terms of functionality, customizability, or efficiency.

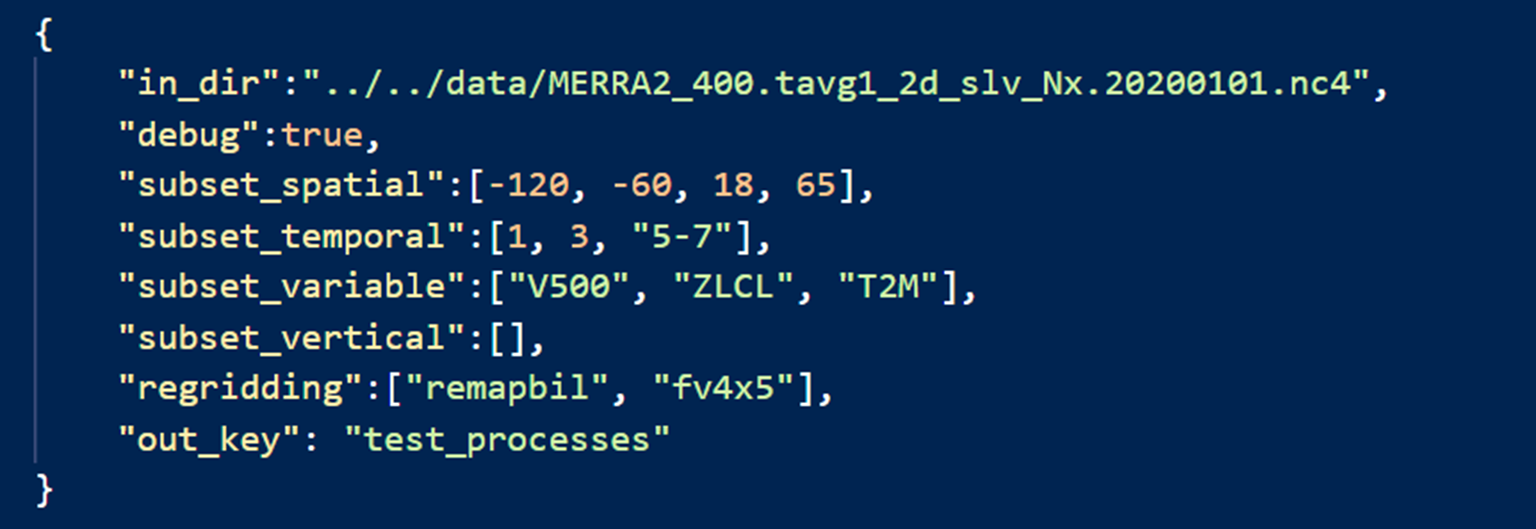
## GRaSS

The tool that I developed is called the GES DISC Regridding and Subsetting System, or GRaSS for short. GRaSS was developed using the Xarray and xESMF python packages and using data from the GES DISC website for testing purposes.

GRaSS takes input in the form of a json file passed as a sole parameter. Said json file contains the following information:



Below is a sample input json file. As described in the template above, each key whose field is not empty corresponds to the operations to perform on the dataset. For this dataset, we wish to perform spatial, variable, and temporal subsetting, as well as regridding. The spatial subset field tells us that the output file will show data only between the coordinates of -120 to -60 degrees longitude and 18 to 65 degrees latitude. The numbers indicated in the temporal subset field tell us that the output dataset will only contain data for the first, third and fifth through seventh hour after midnight on the daily dataset. The subset variable field shows the variables we wish to include in the output dataset, and finally, the regridding field shows us two keys- one corresponding to the interpolation method, and one corresponding to the resolution. Both of these keys are interpreted in GRaSS. For example, “remapbil” translates to bilinear interpolation, and “fv4x5” translates to a 4x5 degree global resolution.

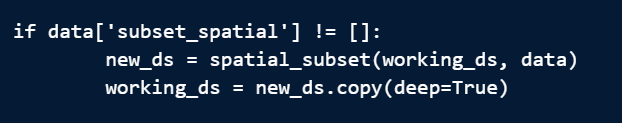


Once all of the operations are complete, a new netCDF file is outputted to an arbitrary temporary directory, with a name identifiable by the output key.

GRaSS reads the json file in the main function, simply called GRaSS(). This function opens the json file passed to the function, and uses xarray to open the dataset(s) to be processed. It then passes the dataset to a function called perform\_operation(), where the json file values are read, and the corresponding operations are performed on the dataset.

The first task performed by the function is to read the dataset units, identify the variables corresponding to "degrees\_east" and "degrees\_north", and change the value names to "lon" and "lat", accordingly. Many datasets have inconsistently named coordinate values (ie. X and Y, latitude and longitude, etc.), so the program must name them uniformly before performing any operations.

Then, the script sets "debug" to the value under data['debug'] from the json file, and reads each of the fields for subsetting and regridding. For each field, the program simply uses if statements to evaluate "if [field] != []", or if the key is not empty. If not, the program calls the function corresponding to the value. For example,



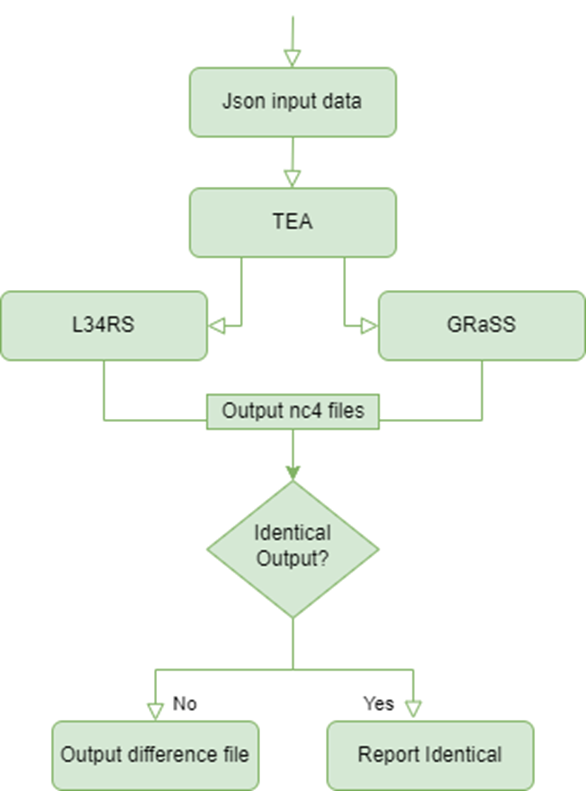
In plain English, this code means that if the field for spatial subsetting is not empty, then the program should call the spatial subset function, pass the working dataset and json information, and then update the working dataset with the new subsetted dataset.

Once all of the operations are complete, a new nc4 file is outputted to the folder of the original input file, with a name identifiable by the name of the tool, the input json, the nc4 file for processing, and the out\_key provided by the json key. If an out\_key is not provided, then the output file name = json file name + input file name. Example: GRaSS\_regridding\_MERRA2\_400.tavg1\_2d\_slv\_Nx.20200101\_key.nc4.

An additional parameter, "NaN\_to\_missing\_val", is used to determine whether or not to convert any NaN values within a dataset to the "missing\_value" specified in the attributes of each dataset. With a default value of "True," this parameter prompts the system to read from the dataset attributes, obtain the default "missing value," and convert all NaN values within the dataset to that value. This functionality comes in handy when we use our comparison tool, TEA, to evaluate the results of multiple different tools' output files of the same dataset, to ensure they have their missing values set to the same number.

## TEA

In order to evaluate the accuracy and performance of the new tool, an additional script was created to compare the output files of GRaSS against those of LEARS. This comparison script is called TEA, the Tool Equivalence Analyzer. TEA's purpose is to compare files subsetted and regridded by GRaSS to those subsetted and regridded by LEARS, using the same input values. The files will be compared using the NCO tool, nccmp.

TEA reads the input json file and searches for existing output files from each tool for that file. If the files do not exist, then TEA runs each tool on the input file, where an output netCDF file is generated. TEA then runs the nccmp operation on the generated output files in the command line. If the output files already exist, TEA simply compares the existing files.

If the files are identical, the program reports as such. If differences in the comparison files are detected, they are outputted to a text file in a temporary directory, with a name corresponding to the names of the two tools and json file under evaluation.

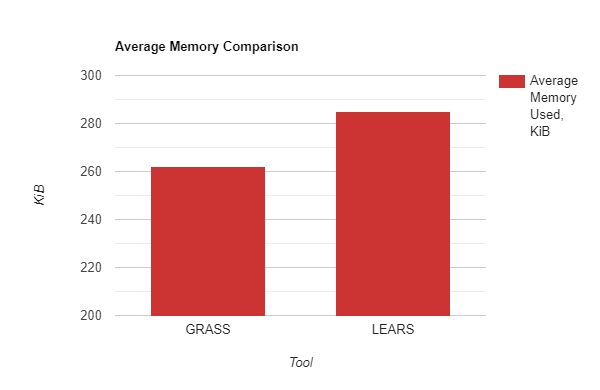
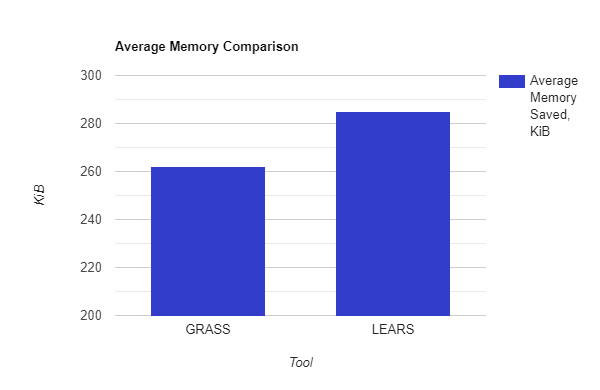
When compared to the LEARS tool, the completed GRaSS tool was observed to produce identical output files to those of LEARS. All functions were tested by TEA, with many different input varieties for each process. Additionally, a test was run executing both GRaSS and LEARS on a json file containing every possible operation, and both tools executed with no problem.

## Efficiency/ Functionality Evaluation against LEARS

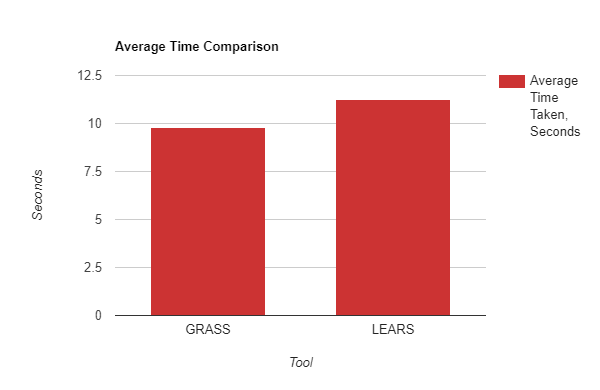
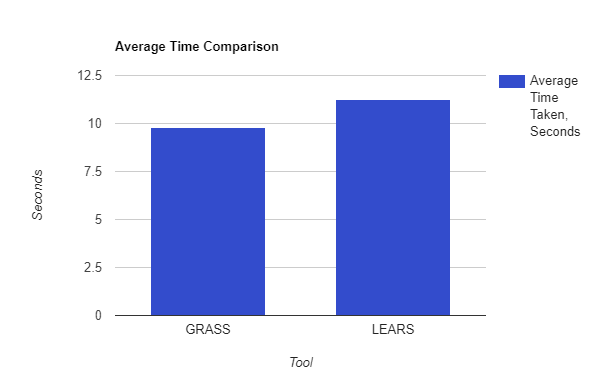
Upon functionality and efficiency analysis, we found that as a whole, GRASS provides more functionality than the existing LEARS tool. Thanks to its xarray structure, it provides a variety of advanced operations and more customizability than otherwise available with CDO (the library used by LEARS). Aspects that set GRASS apart from LEARS include remote file processing functionality, the ability to take json files as input, and the ability to process multiple json files as an input and process all of them accordingly.

In addition to increased functionality, the time and memory resource allocation for each tool was analyzed. A script was developed to run each tool 25 times and take the average amount of time and memory used for execution. These values were then compared, and the following conclusions were drawn:

* For all operations, GRASS requires less memory than LEARS, with an average of 25.33 KiB of memory saved per execution.



* Additionally, with regards to the conda environment dependencies required for the successful execution of each tool, the minimized LERARS is larger than that of GRASS, making GRASS more memory efficient in more ways than one.
* In the case of spatial subsetting, multi-variable subsetting, and certain types of temporal subsetting, GRASS is faster than LEARS, with an average of 5.628 seconds saved per execution. Though GRASS is slower than LEARS in single-variable subsetting, regridding and some temporal subsetting cases, the average amount of time saved by LEARS is only 3.266 seconds per execution. Thus, GRASS on average saves more time than LEARS.



# **2. Challenges Faced/ Future Works**

There were aspects that caused challenges in the process of designing this tool. For example, in xarray datasets, temporal information is of type datetime64. On the other hand, some of the datasets for analysis display time in integer form, to represent hours since a certain time. In the development of GRASS, an extra step had to be taken to make these two datatypes compatible.

Another issue is that xarray cannot translate slicing data spatially across the anti-meridian. Thus, GRASS has to take an extra step to divide a subset on both sides of the meridian and concatenate them, producing the expected result.

In terms of future works, GRASS still has limitations on global regridding- xESMF does not yet support regridding at the poles. Additionally, vertical subsetting has not yet been implemented, and once complete, GRASS will be put into a Docker container for agency use.

# **3.** **Contribution**

As mentioned, my primary focus for the fall semester was to develop a new tool for the GES DISC. The goal of the GES DISC is to enable researchers and educators to maximize knowledge of the Earth. This is done by making data accessible to all, continuing to make advancements and building to integrate new technologies. Though not completely ready for deployment at the moment, GRASS has proven to be a reliable subsetting and regridding tool. GRASS not only provides more extensive functionality and customizability due to its Xarray and xESMF backend- It also proves more efficient in memory and, in some cases, time resource allocation. It is in my hopes that GRaSS will one day be able to replace the current tool in use, and prove valuable to the betterment of the DISC. I believe that by making data more accessible to the world, GRaSS will contribute to our overall knowledge and understanding of Earth and Space data, and aid in accomplishing the DISC’s mission.

# **4.** **Retrospect**

I feel as though this internship experience has greatly aided me in my future career endeavors. Not only has it improved my existing coding and research skills; It also provided me with opportunities for networking and gave me experience working in the real world and applying my skills to real-world problems. Through this experience I also gained the desire to further expand my range of skills, by showing me all of the different options I have, and all of the areas of study I could one day explore. Before this internship, I never had the desire to pursue further education, but I believe that working at NASA changed my outlook. Regardless of where I end up, I believe that through this experience, I am much better equipped to graduate and enter the real workforce.