# CMSC 6950 Project

Yashar Tavakoli

June 2021

## 1 Introduction

Argo is the name of an international data-collection program in oceanography. The program consists of a fleet of instruments around the globe, which operate autonomously to measure and report different ocean variables. In this report, the name argo refer to both the program, and the instruments. Argo data is massive and can be nonhomogeneous. Thanks to argopy (which is a light-weight python library) however, accessing argo data can be quite hassle-free. The mentioned library, also provides data manipulation and visualization facilities.

This project consists of two computational tasks, which in a nutshell, aim at fetching data from `argopy`, manipulating the data a bit, and presenting the results through proper visualization. In my experience, out-of-the-box visualization provisions of argopy, is quite primitive. So I have employed more powerful python libraries for visualization.

Before diving in, here is the workflow of this project (each python script is responsible for a single computational task):

$$\left.\begin{array}{ll} \text{2d\_density\_argos\_locations.py} & \left\{ \text{hexbin\_argos\_locations.png} \right. \\ \text{correlation\_between\_variables.py} & \left\{ \begin{array}{l} \text{map\_of\_locations.png} \\ \text{correlation1.png} \\ \text{correlation2.png} \\ \text{correlation3.png} \\ \text{correlation4.png} \end{array} \right. \end{array} \right\} \text{report.pdf}$$

Now without further further adieu, I will explain the two computational tasks I took on.

## 2 Density of argos' locations

Every argo during its lifetime, visits a sequence of locations (at different depths). Each location during a time-period therefore, can be visited by any number of argos. Provided that the location does not point to land of course. Now an interesting question one can ask, would be: how frequent a location is visited by argos during a certain time-period? Or more interestingly, one can look at a certain geographical region instead. A scatter-plot over a longitude-latitude box, would be a quick approach to answer the mentioned question. That said, scatter-plots suffer from certain shortcomings:

- When the data is dense, a scatter plot would be messy and less interpretable to the eye.
- There is no visual component accompanying scatter-plots, whereby one can learn the number of points in a given area.

A workaround in this situation, would be some sort of 2-dimensional *destiny plot*, whereby the distribution of data is more readily observable.

One of the more sophisticated density plots perhaps, is `seaborn.kdeplot`. But seaborn is not integrated with `mpl_toolkits.basemap`, which is the library I will use for the underlay geographical map (a map-less plot would also suffer from poor interpret-ability). Instead of `seaborn.kdeplot`, I will use `matplotlib.pyplot.hexbin` which is well integrated with `basemap`, and provides comparable results.

To fetch geographically bounded argo data in a given time-period:

- One can use `argo_loader.region().to_xarray()` which takes longitude, latitude, and time period as arguments, and returns the argo data in multidimensional `xarray`.

- Since working with Pandas (2-dimensional) data-frames is more straightforward, I have flattened the `xarrays` (using `argo.point2profile()`), and converted them to Panda's data-frames (using `to_dataframe()`).

- Flattening a multidimensional `xarray` to 2-dimensional produces a multi-index data-frame. In computational task no.1, we do not need any of the indices so I have discarded the indices (using `reset_index()`). Furthermore I have dismissed all the columns but longitude and latitude. Lastly, before drawing the `matplotlib.pyplot.hexbin` on the map, I have prepared the data for `mpl_toolkits.basemap`.

As depicted in Fig.1, the region I have chosen for this task, stretches from 140 to 150 in longitude, and from 35 to 50 in latitude (around Japan). The time-frame is chosen to be a six months period between 2015-06-01 and 2015-12-30. The image depicts the result. One last step I performed as manifest in the image, is highlight some locations on the map to be able to better explain the plot: It seems to be fair to say that argos cover offshore more frequently than ports such as Sendai. One exception perhaps, would be off the coast of Sapporo (in Hokkaido) with more than 1000 reports. One other curious fact perhaps, is argos' low frequency around north of Kunashiri.

# 3 A correlation analysis of Level, Pressure, Salinity, and Temperature.

A question one might ask regarding argo data, would be the correlation between each pair in {Level, Pressure, Salinity, Temperature}. To keep the computations light, I have singled out four profiles from four different argos located in Indian Ocean, South Atlantic, North Atlantic, and Pacific (using the service provided by fleetmonitoring.euro-argo). To fetch the data, I have used `argo_loader.profile().to_xarray()`, which takes in the argo

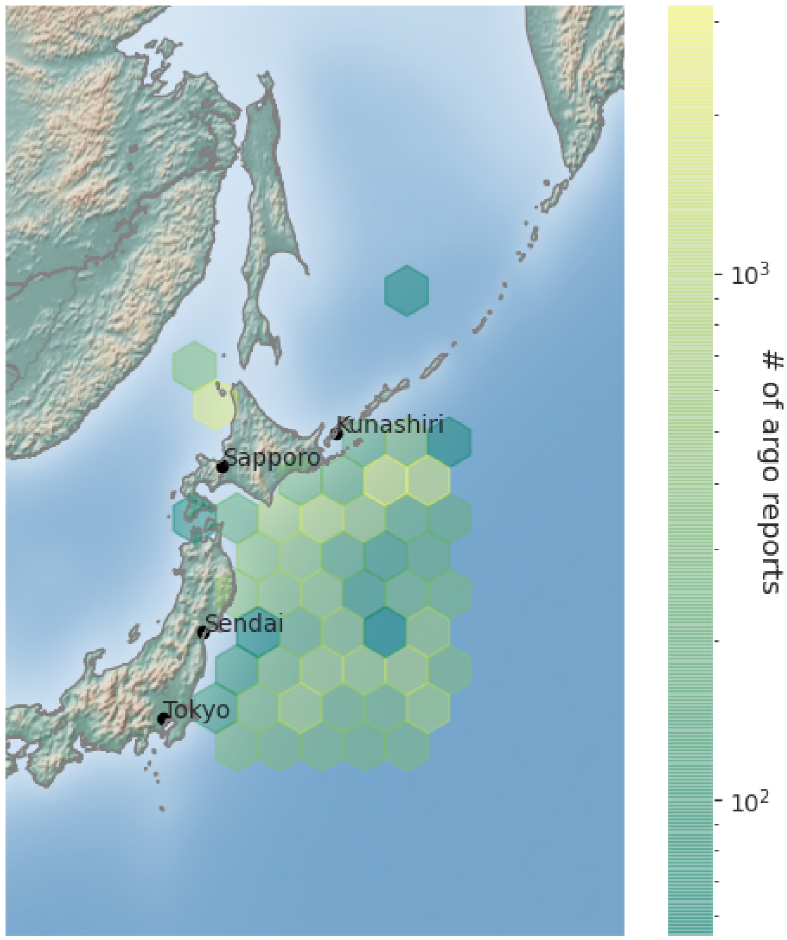



Figure 1: Hexbin plot of argo data

WMO and profile numbers, and returns the data in the form of four different `xarrays` (I have chosen profile numbers manually, so that the produced plots show decent diversity).

Next, as explained in task no.1, I have flattened the `xarrays` and converted them to Panda's data-frames. Here, unlike task no.1, the produced indices in the data-frames are useful: the Level variable in the data-frames, is defined as index. As the first step, I have reintroduced Level as a column and dismissed all the column except Level, Pressure, Salinity, and Temperature.

Before moving forward, let's take a look at a map produced by `basemap` hinting at the spots where the chosen argos resided or have resided (Fig.2). Since each profile consists of a sequence of longitudes and latitudes, I have taken a mean to get an idea about the whereabouts of the argos pertaining to the selected profiles.

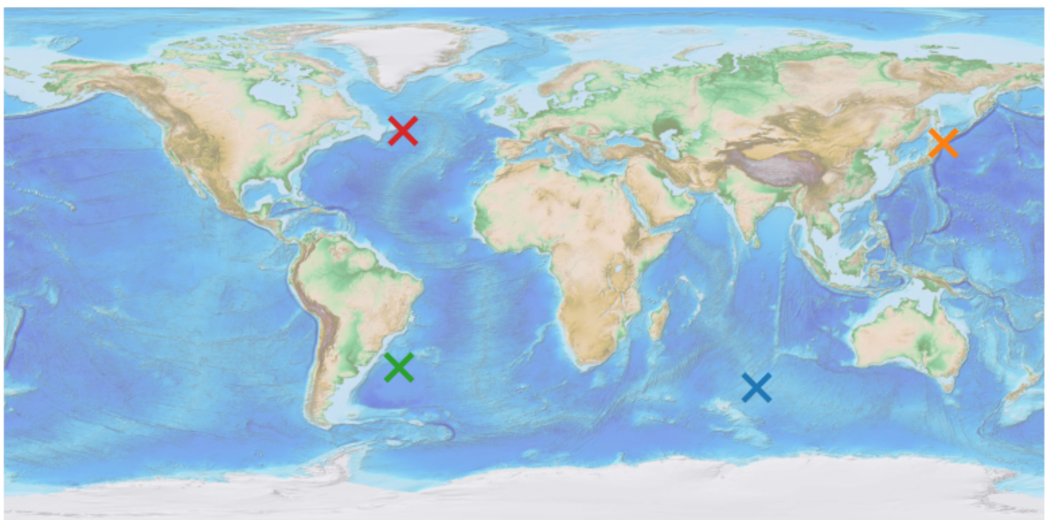


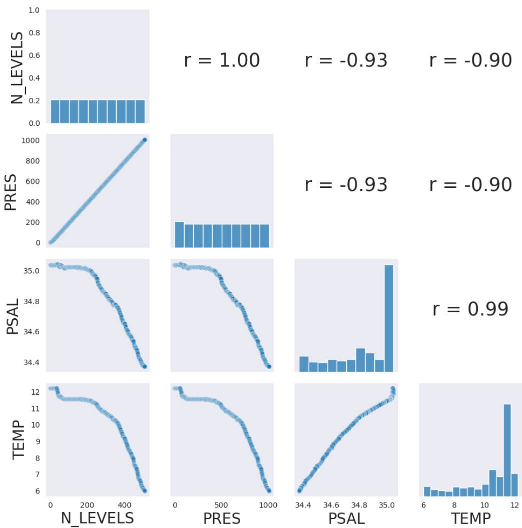
Figure 2: Locations of the argos

Now back to the main task at hand, we need a proper visualization which help us get an idea of the correlation between the variables in question. To do this, I have devised a composite plot from three components:

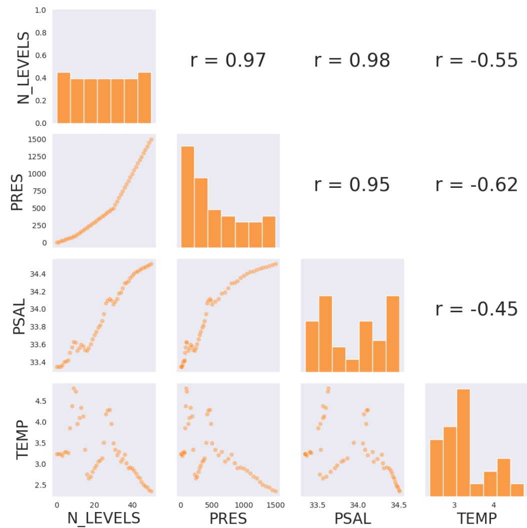
1. The lower-triangle consists of pairwise scatter-plots.

2. The diagonal consists of histogram bars illustrating the distribution of variables.

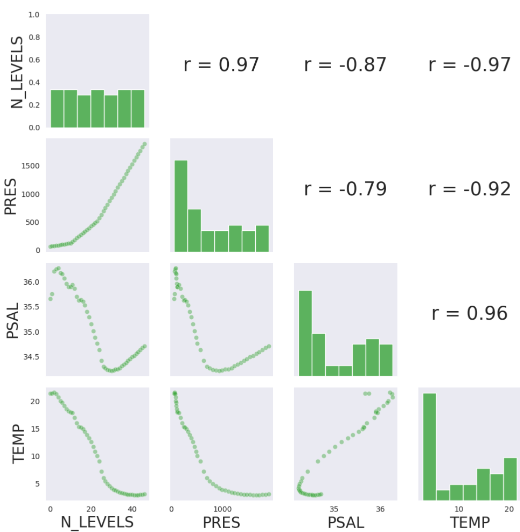
3. The upper-triangle indicates the Pearson's r coefficient for each pair of variables (implemented though `reg_coef()` in the code). The Pearson's r ranges from 1 to -1. A value close to 1 (-1) indicates a (reverse) near perfect linear relationship between the variables. A value close to 0, implies that there is almost no linear correlation between the variables.



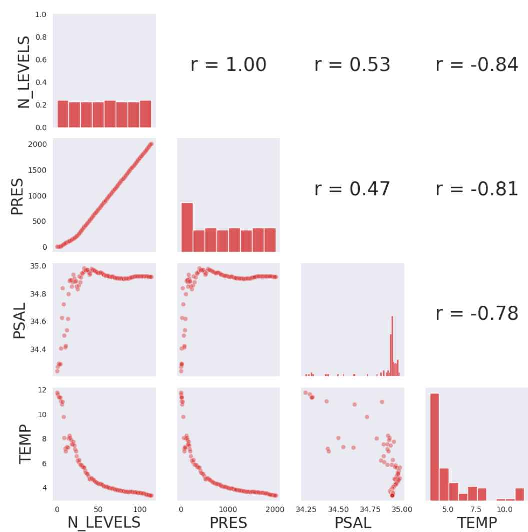

(a) Location: Indian Ocean




(b) Location: North Pacific




(c) Location: South Atlantic




(d) Location: North Atlantic

Figure 3: Correlation between the variables

Now lets take a couple of notes regarding the correlation between the variables as illustrated in Fig.3:

- The scatter plots and coefficients for all the cases indicate a near perfect linear relationship between Pressure and Level.
- All the plots also demonstrate that there exist a significant reverse linear relationship between Temperature and Level, as well as Temperature and Pressure.
- The rest of the pair of variables show non-conclusive behavior. The situation here perhaps, would entail a more educated investigation. That said, curiously there is no near-zero coefficient for any of the pairs.
- The variables are similarly correlated pertaining to profiles from Indian Ocean, and South Atlantic. Same with North Atlantic and North Pacific. Hence a crude non-educated guess: the divide is between northern and southern hemisphere (?).

# 4 Conclusion

In this project, I implemented two computational tasks involving argopy, where data is fetched from external sources and visualized properly through python plotting libraries.

# References

[1] Maze et al., *argopy: A Python library for Argo ocean data analysis.*. Journal of Open Source Software, 5(53), 2425, 2020.