# CMSC 6950 Project

Yashar Tavakoli

June 2021

## 1 Introduction

The ocean is a key component of the Earth climate system. It thus needs a continuous real-time monitoring to help scientists better understand its dynamic and predict its evolution. All around the world, oceanographers have managed to join their efforts and set up a Global Ocean Observing System among which Argo is a key component.

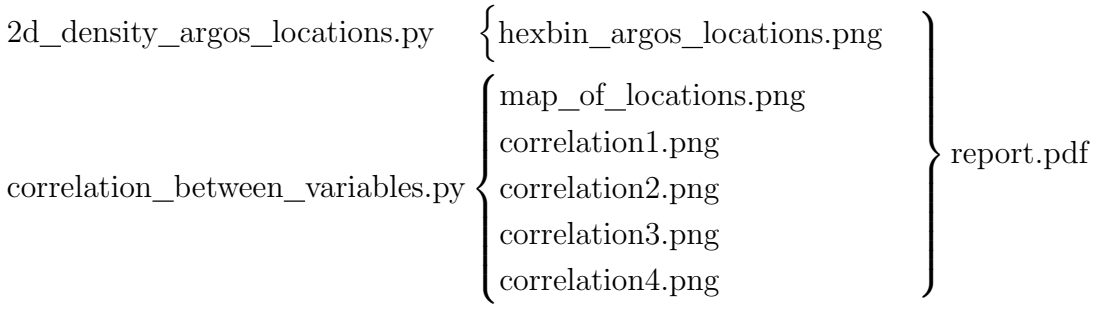
Argo is a global network of nearly 4000 autonomous probes measuring pressure, temperature and salinity from the surface to 2000m depth every 10 days. The localisation of these probes is nearly random between the 60th parallels (see live coverage here). All probes data are collected by satellite in real-time, processed by several data centers and finally merged in a single dataset (collecting more than 2 millions of vertical profiles data) made freely available to anyone through a ftp server or monthly zip snapshots.

This project in a nutshell, aims at fetching data from `argopy`, manipulating the data a bit, and presenting the results through proper visualization. `argopy` landing page states that it aims to

1. ease argo data access,
2. manipulation, and,
3. visualization.

In my experience, visualization facilities of the argopy is quite limited. So I have employed more powerful python libraries for visualization.

Here is the workflow of this project (each python script is responsible for a single computational task):




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 sequence of locations (at different depth). Having said that, the frequency of locations (in form longitude and latitude pairs) within every geographical boundary over a certain period of time, would be non-uniform. So an interesting question would be: in given a geographical boundary in the form of a box defined by latitude and longitude coordinates, which points are visited by argos and how much?

Scatter-plot would be a quick approach to this problem. Scatter-plots however suffer from certain shortcomings:

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

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`.

Now regarding the geographically bounded argo data, during 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 computation 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 Fig.1 in 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 is highlighting some locations on the map to be able to better explain the plot: It seems to be more-or-less 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 profile reports. One curious fact is argos' low frequency 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, and 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

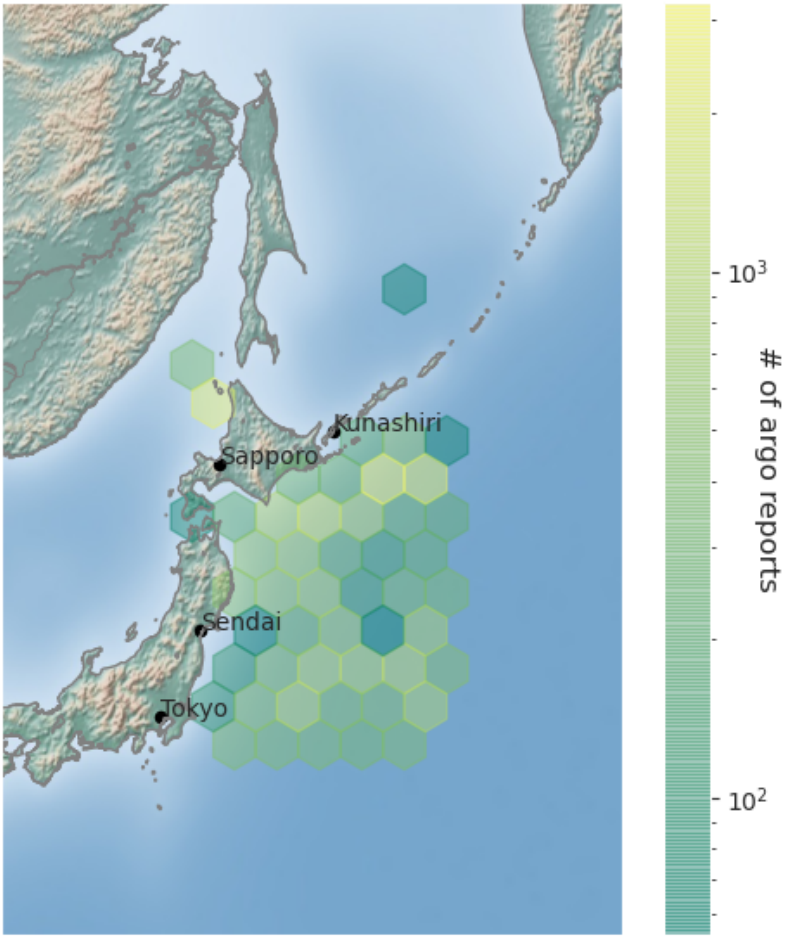



Figure 1: Hexbin plot of argo data

the service provided by fleetmonitoring.euro-argo). Now to fetch the data I have used `argo_loader.profile().to_xarray()`, which takes the argo 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 not useless: the Level variable the data-frames is defined

as index. As a 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.

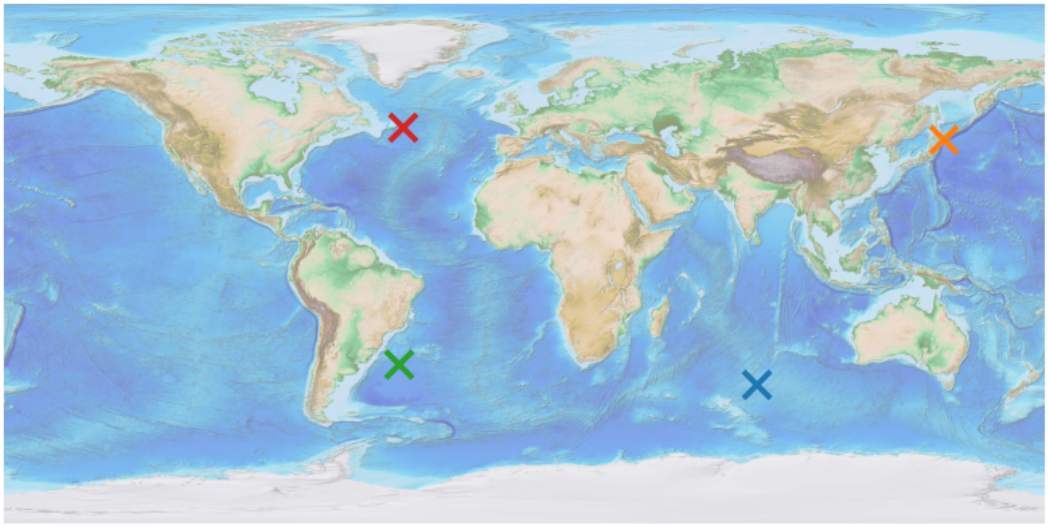
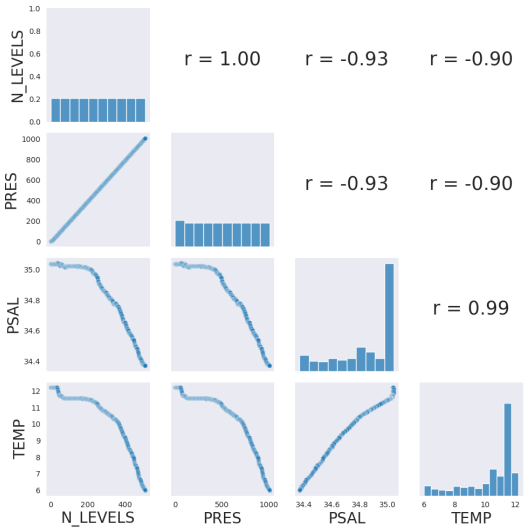


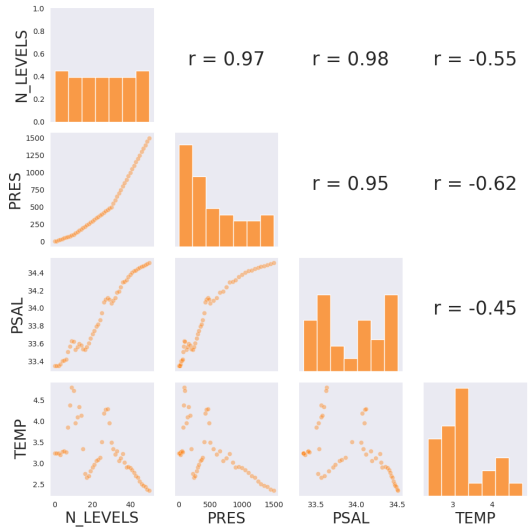
Figure 2: Locations of the argos

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 thee components:

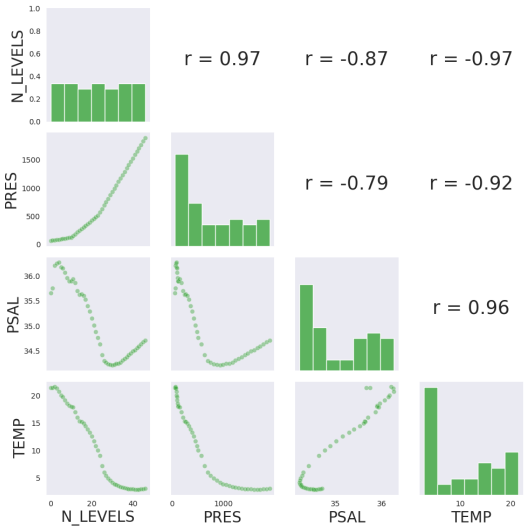
1. The lower-triangle consists of pairwise scatter-plots.
2. The diagonal consists of histogram bars illustrating the distribution of variables.
3. The upper-triangle indicates the Pearson's r coefficient for each pair of variables (implemented though `reg_coef()` in the code). The Pearson's r ranges from 1 to -1. A value close to 1 (-1) indicates a (reverse)



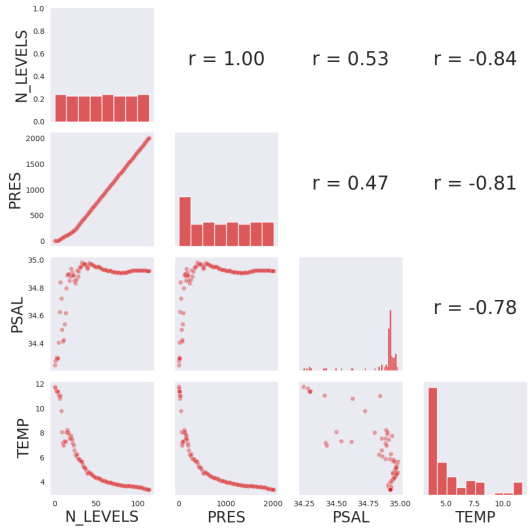

(a) Location: Indian Ocean




(b) Location: North Pacific




(c) Location: South Atlantic




(d) Location: North Atlantic

Figure 3: Correlation between the variables

near perfect linear relationship between the variables. A value close to 0, implies that there is almost no linear 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-perfectly 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 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, South Atlantic. Same with North Atlantic and North Pacific. So 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.