# Mathematics and big data technology development to visualize, deliver, and analyze IMS and ARGO data

Tyler Tucker

San Diego State University

April 23rd 2018

# Table of Contents

# Table of Contents

# Thesis overview



- Both climate and oceanographic science disciplines are feeling a data explosion and are struggling to scale.
- Unprecedented amounts of data are made available by governmental agencies like NOAA and NASA and used by the scientific community to draw inferences about the Earth.
- The traditional visualization and delivery tools can be improved to handle the surge of big data.
- This presentation describes techniques used to create visualization and data retrieval products.
- Global data sets need modern web app technologies for professional and amateur to visualize, retrieve, and analyze data as quickly, accurately, and painlessly as possible.

# Focus on two data sets



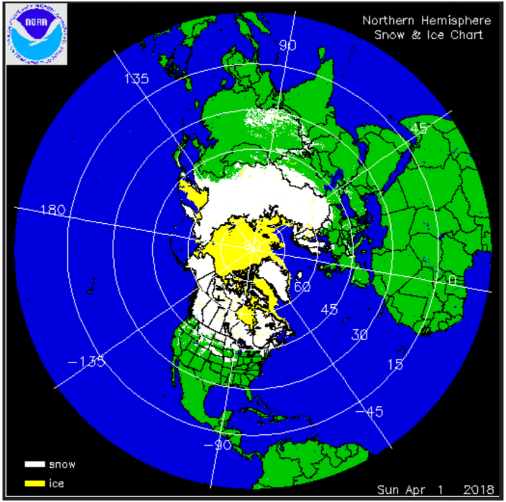


Figure: IMS snow cover, `http://www.natice.noaa.gov/ims/`

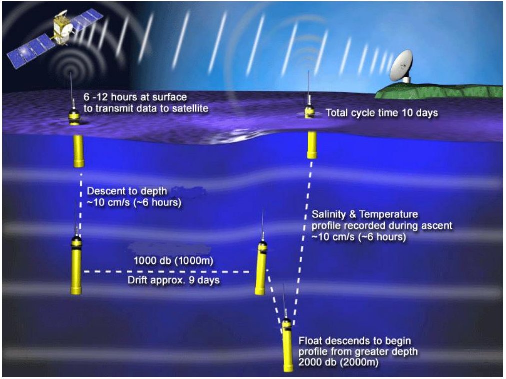


Figure: Detail of one profile cycle
`http://www.argo.ucsd.edu/How_Argo_floats.html`

# Table of Contents

# Interactive Multisensor Snow and Ice Mapping System



- Daily snow/ice coverage.
- Matrix style output on FTP site
- Current tools for visualization

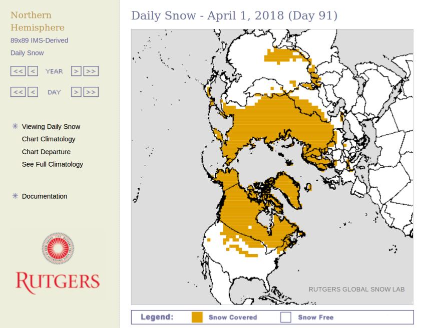


Figure: IMS snow cover, https://climate.rutgers.edu/snowcover/

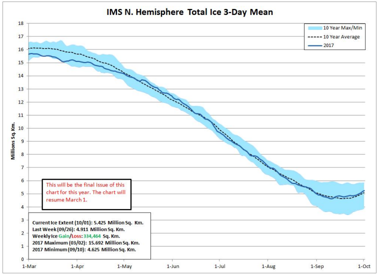


Figure: IMS snow area, http://www.natice.noaa.gov/ims/

## Research topics

- Calculate grid area?
- Estimate snow/ice coverage in $km^2$
- Estimated error vs other products?

# Introducing IMS - Tibet Snow Man



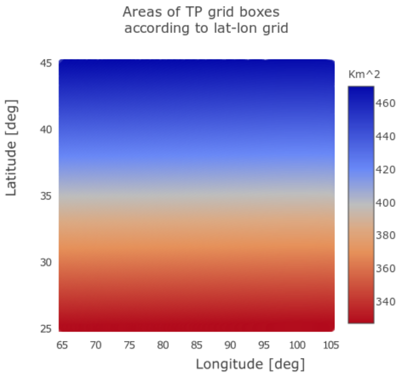


Figure: Time series comparison between 24 x 24km 2 assumption and area calcu- lation via shoe-lace formula.

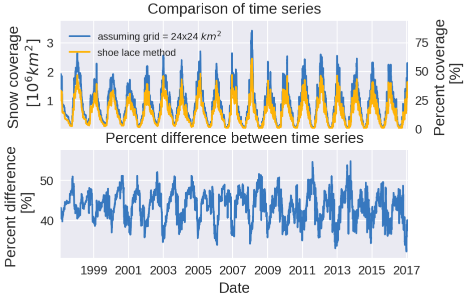


Figure: Time series comparison between $24x24km^2$ assumption and area calculation via shoe-lace formula.

# Introducing IMS - Tibet Snow Man



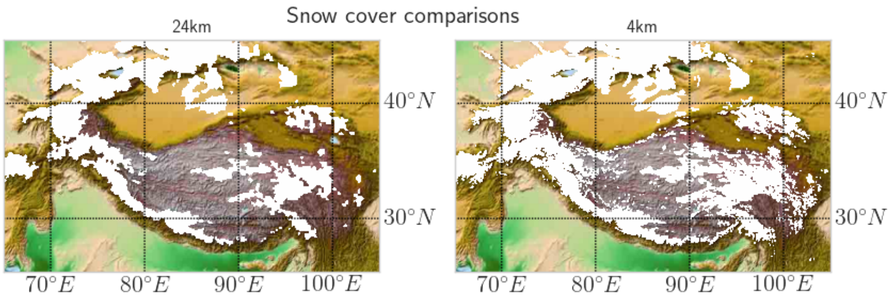


Figure: Time series comparison between 24 x 24km 2 assumption and area calculation via shoe-lace formula. Time lapse is available at `https://www.youtube.com/watch?v=Kz82dwMKnLY&t=0s`

# Describing the Data



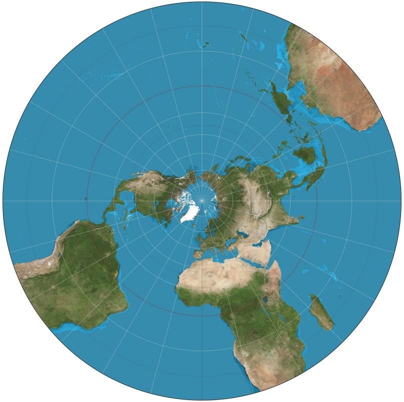

Figure: Stereographic projection. Taken from https://en.wikipedia.org/wiki/Stereographic_projection

## Stereographic Projection

Equations to map a sphere point $(\phi, \lambda)$ to a Cartesian point $(x, y)$ are below:

$$x = 2R\, k_0 \tan\left(\frac{\pi}{4} - \frac{\phi}{2}\right) \sin(\lambda - \lambda_0)$$

$$y = -2R\, k_0 \tan\left(\frac{\pi}{4} - \frac{\phi}{2}\right) \cos(\lambda - \lambda_0)$$

The length scale factor $k$ is

$$k = \frac{2k_0}{1 + \sin(\phi)},$$

and $(\phi_0, \lambda_0)$ is a designated center point.

# Describing the Data



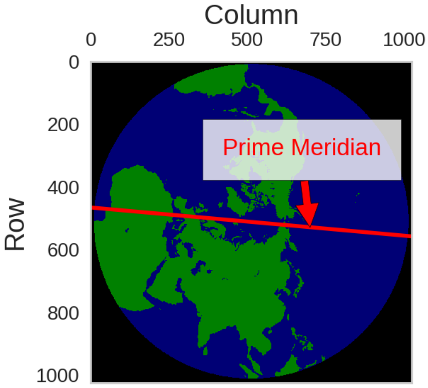


Figure: Grid coordinate ticks of NH without snow and ice cover. IMS provides daily files in an n x n for- mat. The data is given in .asc files but mirrored to the figure shown. The file represents a stereographic projec- tion, with the 80$^\circ$ meridian line pointing up. IMS rotates the prime meridian by about 10.22$^\circ$ clockwise to the hori- zontal. This figure shows the 24km resolution data, which has n = 1024.

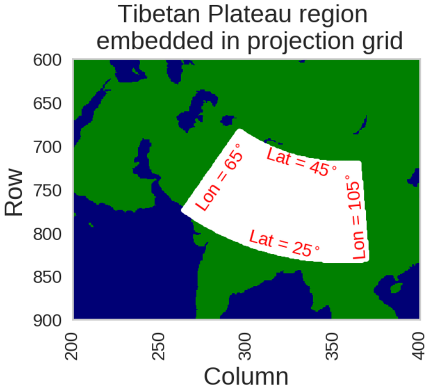


Figure: The TP region (25$^\circ$ - 45$^\circ$ N and 65$^\circ$ - 105$^\circ$ E) shown bounded in white is shown on a stereographic projection.

# Describing the Data



- IMS claims a 24x24, 4x4, and 1x1 km resolution products. In reality, this is true at about 60° latitudes.
- IMS is a non-uniform grid.
- Data is coarser at the pole!
- Areas along latitude vary by almost x2
- Why make equator data finer for a snow/ice product?
- What are the consequences calculating snow/ice coverage with a uniform grid assumption?

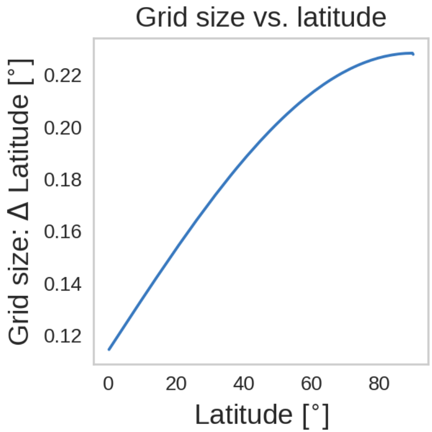


Figure: Latitude was taken from IMS latitude file: rows 0 to 512, column 512, points plotted lie approximately at $80^\circ E$. Grid points are distributed densely near the equator, with latitude grid spacing of $.11^\circ$ at the equator. The latitude grid spacing increases to $.23^\circ$ at the north pole.

# Describing the Data



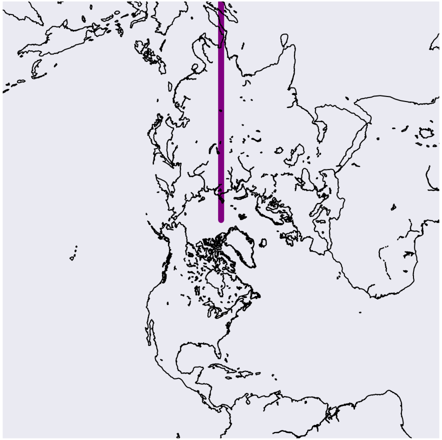

Figure: Latitude was taken from IMS latitude file: rows 0 to 512, column 512, points plotted lie approximately at $80^\circ E$.

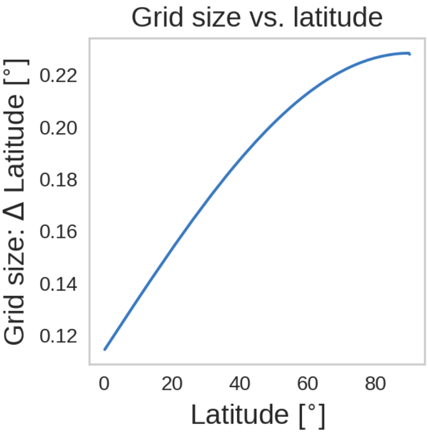


Figure: Grid points are distributed densely near the equator, with latitude grid spacing of $.11^\circ$ at the equator. The latitude grid spacing increases to $.23^\circ$ at the north pole.

# Calculating Areas



- Question: Given a latitude-longitude grid, how can we estimate areas?
- Answer: Assume each point is a center of a 4-point polygon on a 2d equal-area projection. We can estimate the corners and find the areas of each cell using Green's theorem.

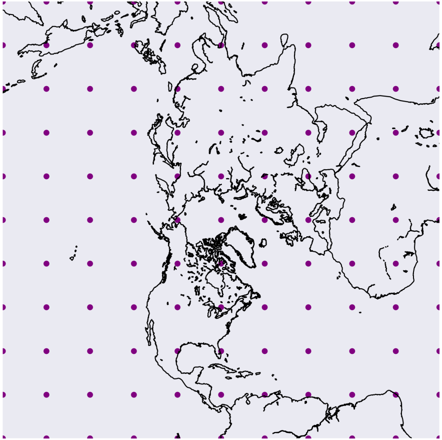

Figure: Sample grid

# Calculating Areas: Use an Equal Area Projection



Figure: Lambert azimuthal equal area projection. Taken from `https://en.wikipedia.org/wiki/Lambert_azimuthal_equal-area_projection`

## Lambert Azimuthal Equal Area Projection

Equations to map a sphere point $(\phi, \lambda)$ to a Cartesian point $(x, y)$ are below:

$$x = Rk' \cos\phi_1 \sin(\lambda - \lambda_0)$$

$$y = Rk'[\cos\phi_1 \sin\phi - \sin\phi_1 \cos\phi \cos(\lambda - \lambda_0)]$$

$$k' = \frac{\sqrt{2}}{\sqrt{1 + \sin\phi_1 \sin\phi + \cos\phi_1 \cos\phi \cos(\lambda\lambda_0)}}$$

Where $R = 6,371km$ is the Earth's radius. For the TP, $\phi_1 = 35^\circ$, $\lambda_0 = 85^\circ$

# Calculating Areas: Estimate Grid Cell Area



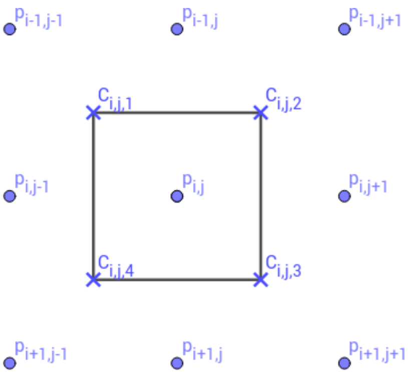


Figure: Grid box showing grid box corners $C_{i,j,k}$ for lat-lon point $p_{i,j}$ calculated using shoe-lace formula.

## Corner/Centroids

A grid point $p_{i,j}$ has four corners $C_{i,j,k}$. Corners are the centroids of the bounding square.

$$C_ij1 = \frac{1}{4}(p_{i-1,j-1} + p_{i,j-1} + p_{i,j} + p_{i-1,j})$$

$$C_ij2 = \frac{1}{4}(p_{i-1,j} + p_{i-1,j+1} + p_{i,j+1} + p_{i,j})$$

$$C_ij3 = \frac{1}{4}(p_{i,j} + p_{i,j+1} + p_{i+1,j+1} + p_{i+1,j})$$

$$C_ij4 = \frac{1}{4}(p_{i-1,j} + p_{i,j} + p_{i+1,j} + p_{i-1,j+1})$$

# Calculating Areas: Estimate Grid Cell Area



## Shoelace/Surveyor's Formula

Discreet Green's Theorm estimates area on a 2D surface. Given a set of vectors $x_k$, $y_k$ of the four corners of point $p_{i,j}$, area $A_{i,j}$ is computed to be

$$A_{ij} = \frac{1}{2}\left|\sum_{k=1}^{3} x_k y_{i+1} + x_n y_1 - \sum_{k=1}^{3} x_{k+1} y_k - x_1 y_n\right|$$

# LEAP Areas Over TP of '$24km^2$' Resolution



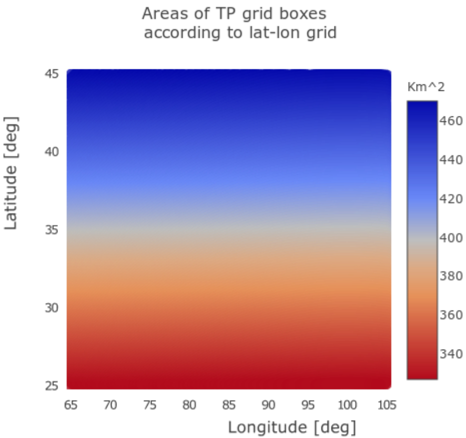


Figure: The area of a grid box varies according to latitude.

# Calculating Areas: Estimate Grid Cell Area



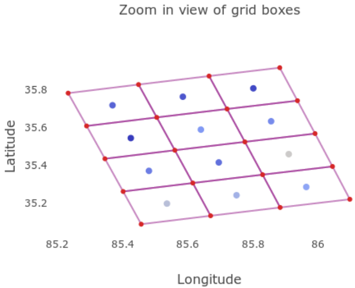


Figure: A zoom-in display of grid boxes and their centroid points provided a detailed view on how areas are estimated. Each central circle represents a lat-lon point provided by IMS. TSM calculates the red corner points followed by the bounding boxes' area.

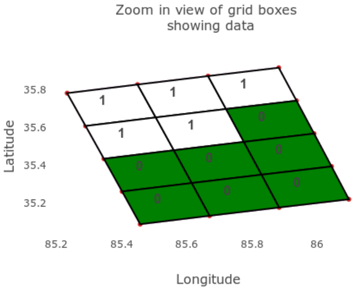


Figure: A zoom-in view of the IMS data with snow/no snow illustrations. Cells covered in snow are in white and cells with no snow are shown in green.

# Comparing LEAP Areas With Uniform Assumption



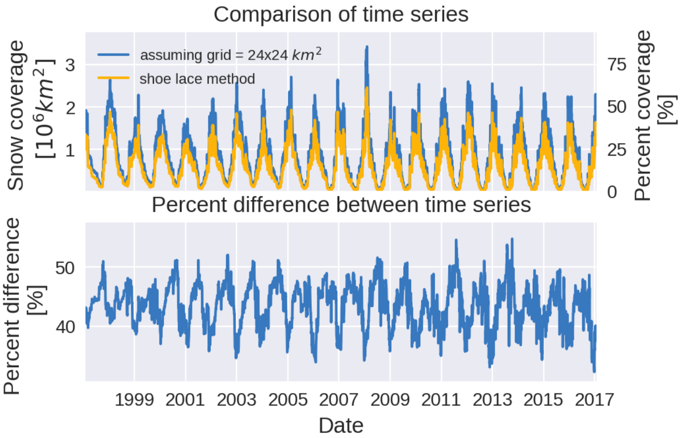


Figure: Time series comparison between $24 \times 24km^2$ assumption and area calculation via shoe-lace formula.

# Comparing '$24 km^2$' with '$4 km^2$' Resolutions



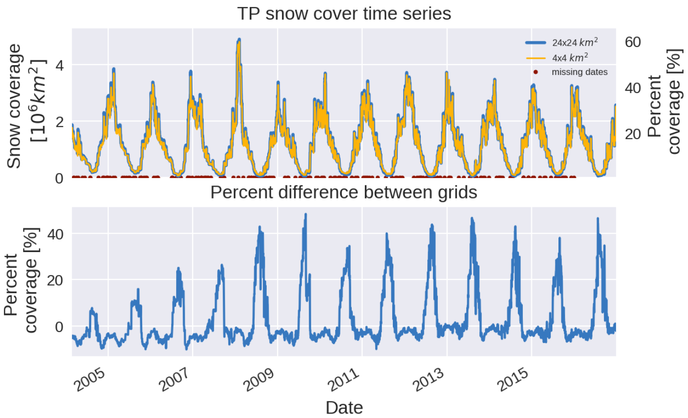


Figure: The snow cover area time series comparison between two resolutions: 4 km and 24 km. Percent difference calculated using percent difference $100 * \left(\frac{ts_{24\ km} - ts_{4\ km}}{ts_{4\ km}}\right)$.

# Climatology: 5 Day Binning



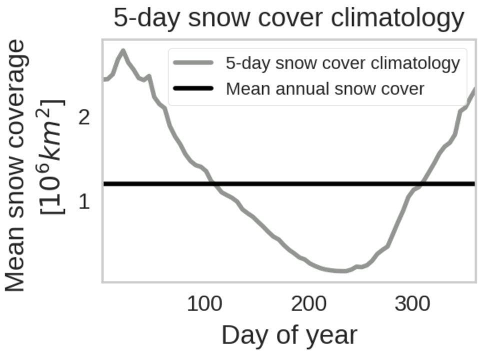


Figure: 5 day climate averages of the 24$x$24 km annual snow cycle.

# Climatology: STL via LOESS Regression



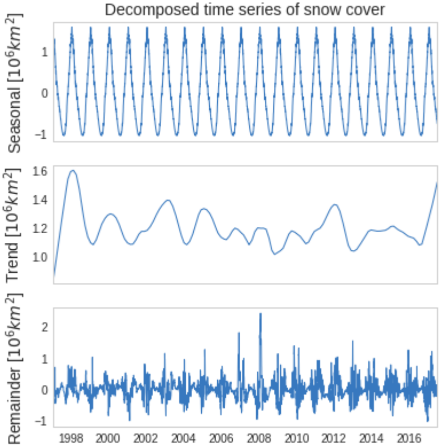


Figure: Deseasonalized using STL algorithm described in Appendex.

## STL

Seasonal Decomposition Based on LOESS by Cleveland et al 1990 allows a non stationary time series with a seasonal trend to be decomposed into three parts: The trend $T_t$, season $S_t$, and the residual $R_t$. Where the original signal $Z_t$ is combination of all three.

$$Z_t = T_t + S_t + R_t$$

Implementation is done using the stl() function in R.

# Anomalies



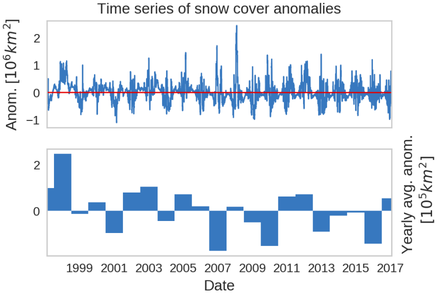


Figure: Time series of Snow coverage anomalies with included trend line, in addition to yearly averaged anomalies, shown by the bar chart.

## Anomalies

5-day average climate model $clim(t)_5$ is subtracted from the time series, $Snow(t)$.

$$Anomalies(t) = Snow(t) - clim(t)_5$$

Note that anomalies and STL remainder are attempting to deseasonalize the signal. Bother are not variance stationary.

# Anomalies



Table: Anomaly Distribution Properties

| Statistic | Value |
|---|---|
| Mean | $-309.9 km^2$ |
| Median | $-30992 km^2$ |
| Min | $-1.112\ 10^6 km^2$ |
| Max | $2.446\ 10^6 km^2$ |
| Standard Deviation | $.34259971\ 10^6 km^2$ |
| Skew | 1.0006 |
| Kurtosis (Pearson's) | 4.351 |

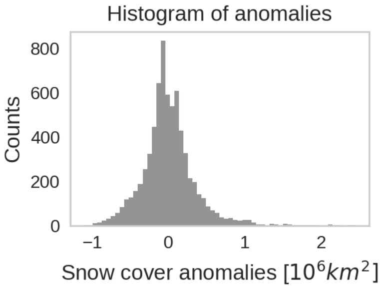


Figure: Histogram of the $24 \times 24$ anomalies.

# TSM as a Product



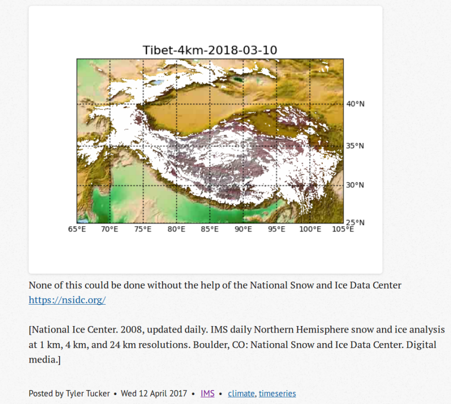


Figure: TSM is currently used to generate daily snow mapping at `http://www.itsonlyamodel.us/daily-snow.html`

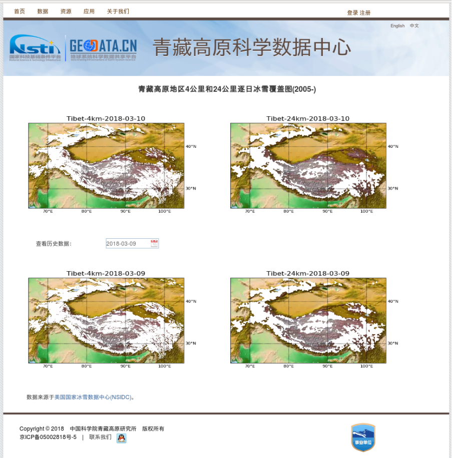


Figure: Screenshot of `http://www.tpedatabase.cn/tibetSnow.jsp` showing daily snow plot at archives at 24 and 4 km resolution.

# TSM as a Product



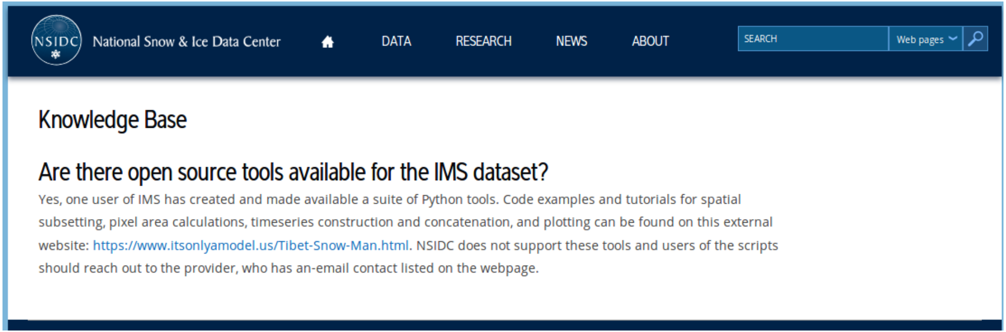


Figure: TSM is cited by NSIDC at at `http://nsidc.org/support/faq/are-there-open-source-tools-available-ims-dataset`

# Next Steps and Conclusion




- Build code into a web app instead of a python library. Interactive web apps attract more users and researchers.
- IMS error is difficult to characterize. How can we estimate error, reduce bias? Can we trust these products?
- Apply this when they release a Southern Hemisphere Product

# Table of Contents

# Argo



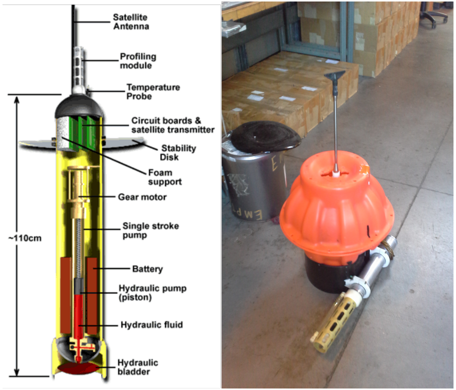


Figure: APEX float (left). Image taken from http://argofloats.wikispaces.com/home Deep Solo prototype (right) Taken by me at SIO.

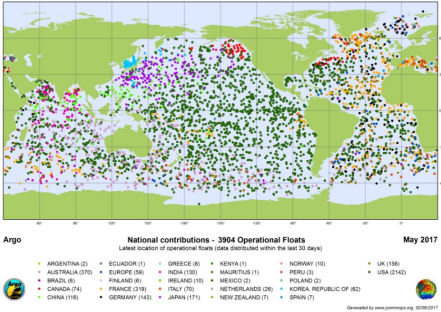


Figure: Image taken from http://argofloats.wikispaces.com/home

## Research topics

- How to access data?
- How to visualize data set?
- How to handle errors?

# Argo Terminology



- Profile: Prray of measurements taken by float/platform.
- Platform: Device that takes profiles. Platform $\equiv$ Float.
- Selection: Profiles grouped by area, date, depth, etc.

# Coriolis Visualization Application



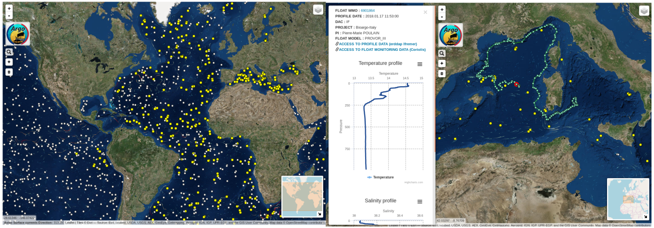


Figure: Argo Active network map (left) `http://www.argo-france.fr/en/argo-active-network-map/`Showing argo profiles owned by the Coriolis (French) DAC. Yellow dots represent floats operated by the Coriolis project. Selecting a dot reveals a side panel with additional information (right). Velocity field animations overlay the profile dots showing currents generated via ANDRO. Screenshot taken from `http://map.argo-france.fr/`

**Pros:**

- Fast
- Intuitive
- Interactive

**Cons:**

- Can't select regions
- Can't select platforms
- Limited to recent profiles
- Highcharts aren't FOSS

San Diego State University

# NASA Sea Level Change Data Analysis Tool



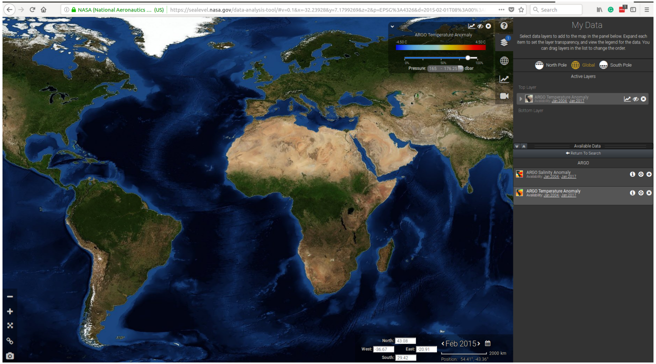


Figure: DAT web app format with Data tab displayed on the side bar. Argo derived gridded salinity and temperature products from January 2004 to January 2017 are available to plot. Screenshot taken from `https://sealevel.nasa.gov/data-analysis-tool/`

## Pros:

- Regional selection
- Offers several data sets
- Time series
- Contour images overlay

## Cons:

- Cuts off in middle of ocean
- Slow
- Non-intuitive
- Beta product (includes bugs)
- Limited to gridded products

# 4D Visual Delivery of Big Climate Data: MrSharky



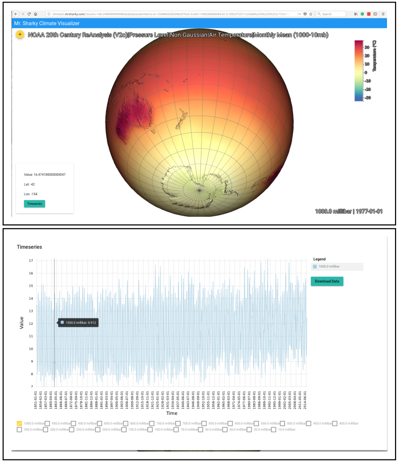


Figure: Clicking a point on the globe will generate a box element showing the point's value and latitude-longitude coordinates, shown in the lower left corner of the (top) subfigure. Clicking the teal button will generate a time series (bottom) that can be downloaded as a .csv file. Screenshots taken from `http://climate2.mrsharky.com/`

## Pros:

- Rich with custom viewing features
- View in seveal projections
- Provides time series
- Uses several products

## Cons:

- Resource heavy
- Little documentation on how to use
- Limited chart capability
- Limited to gridded products

# Visualization and Data Retrieval Tool: Argovis



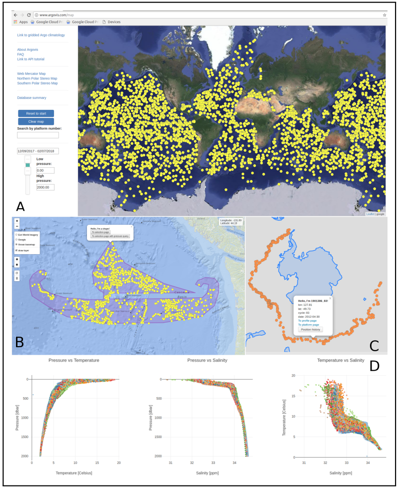


Figure: Argovis features: interactive map (top) Selection(mid right), stereo projection (left), data charts (bottom)

## Pros:

- Custom viewing features
- Includes stereographic projection
- Interactive map
- API included
- Video tutorials! `https://www.youtube.com/watch?v=IlNJOowuTHM`

## Cons:

- No time series
- Only Argo data
- No gridded products

# Argo Data



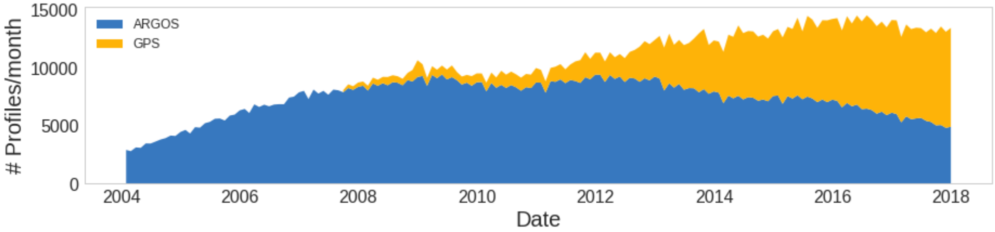


Figure: History of Argo profiles reported each month since the programs in- ception. Trends towards GPS measurements indicate a need for a data retrival service. Argo data generation is increasing in the number of profiles reported and by the amount of data reported by each profile.

SAN DIEGO STATE UNIVERSITY

# Argo FTP



## Index of ftp://usgodae.org/pub/outgoing/argo/latest_data/

Up to higher level directory

| Name | Size | Last Modified | |
|---|---|---|---|
| File: D20180112_prof.nc | 88595 KB | 1/13/18 | 12:01:00 AM PST |
| File: D20180113_prof.nc | 53831 KB | 1/13/18 | 3:00:00 AM PST |
| File: D20180115_prof.nc | 55136 KB | 1/15/18 | 11:00:00 PM PST |
| File: D20180116_prof.nc | 31304 KB | 1/16/18 | 10:02:00 PM PST |
| File: D20180117_prof.nc | 53875 KB | 1/18/18 | 12:00:00 AM PST |
| File: D20180118_prof.nc | 31498 KB | 1/18/18 | 9:00:00 PM PST |
| File: D20180118_prof_0.nc | 99863 KB | 1/18/18 | 4:03:00 AM PST |
| File: D20180119_prof.nc | 1928 KB | 1/20/18 | 12:00:00 AM PST |
| File: D20180120_prof.nc | 1367 KB | 1/20/18 | 3:00:00 AM PST |
| File: D20180122_prof.nc | 2035 KB | 1/22/18 | 8:00:00 PM PST |
| File: D20180123_prof.nc | 31686 KB | 1/23/18 | 5:00:00 PM PST |
| File: D20180124_prof.nc | 22981 KB | 1/24/18 | 6:00:00 PM PST |
| File: D20180125_prof.nc | 79559 KB | 1/25/18 | 3:00:00 PM PST |
| File: D20180125_prof_0.nc | 100842 KB | 1/25/18 | 2:00:00 PM PST |
| File: D20180127_prof.nc | 696 KB | 1/27/18 | 3:00:00 AM PST |
| File: D20180129_prof.nc | 81714 KB | 1/30/18 | 12:00:00 AM PST |
| File: D20180130_prof.nc | 25430 KB | 1/30/18 | 1:00:00 PM PST |
| File: D20180131_prof.nc | 76379 KB | 1/31/18 | 8:00:00 PM PST |
| File: D20180201_prof.nc | 96713 KB | 2/1/18 | 4:00:00 PM PST |
| File: D20180201_prof_0.nc | 99828 KB | 2/1/18 | 5:00:00 PM PST |

# Argovis JSON output



Listing 1: JSON schema of mongoDB profile.

```
{
  "_id": <String>,
  "max_pres": <Int>,
  "measurements": <List[{temp: <Number>,
                         psal : <Number>,
                         pres : <Number>,
                         ...}]>,
  "date": <ISODate>,
  "POSITIONING_SYSTEM": <String>,
  "PLATFORM_TYPE": <String>,
  "DATA_MODE": <String>,
  "PI_NAME": <String>,
  "cycle_number": <Int>,
  "lat", <Number>,
  "lon", <Number>,
  "geoLocation", {"type": "Point",
                "coordinates": [<Number[2]>],
  "dac": <String>,
  "platform_number": <String>,
  "station_parameters": <List[String]>,
  "nc_url": <String>
}
```

# API: Profile function



Listing 2: Python function to retrieve profile data. Uses 'requests' library

```
def get_profile(profile_number):
    resp = requests.get('http://www.argovis.com/catalog/profiles/'+profile_number)
    # Consider any status other than 2xx an error
    if not resp.status_code // 100 == 2:
        return "Error: Unexpected response {}".format(resp)
    profile = resp.json()
    return profile
```

# API: Platform function



Listing 3: Python function to retrieve platform data

```
def get_platform_profiles(platform_number):
    resp = requests.get('http://www.argovis.com/catalog/platforms/'+platform_number)
    # Consider any status other than 2xx an error
    if not resp.status_code // 100 == 2:
        return "Error: Unexpected response {}".format(resp)
    if resp.status_code == 500:
        return "Error: 500 status {}".format(resp)
    platformProfiles = resp.json()
    return platformProfiles
```

# API: Selection function



Listing 4: Python function to retrieve selection data

```
def get_selection_profiles(startDate, endDate, shape, presRange=None):
    baseURL = 'http://www.argovis.com/selection/profiles'
    startDateQuery = '?startDate=' + startDate
    endDateQuery = '&endDate=' + endDate
    shapeQuery = '&shape='+shape.replace(' ','')
    if not presRange == None:
        pressRangeQuery = '&presRange=' + presRange
        url = baseURL + startDateQuery + endDateQuery + pressRangeQuery + shapeQuery
    else:
        url = baseURL + startDateQuery + endDateQuery + shapeQuery
    resp = requests.get(url)
    # Consider any status other than 2xx an error
    if not resp.status_code // 100 == 2:
        return "Error: Unexpected response {}".format(resp)
    if resp.status_code == 500:
        pdb.set_trace()
        return "Error: 500 status {}".format(resp)
    selectionProfiles = resp.json()
    return selectionProfiles
```

# API: Time Series Applications



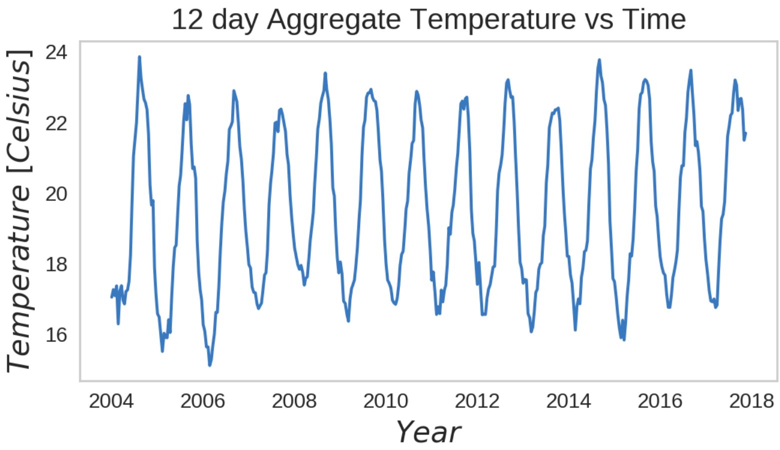


Figure: API includes a special time series generator for a selection.

# API: Time Series Applications



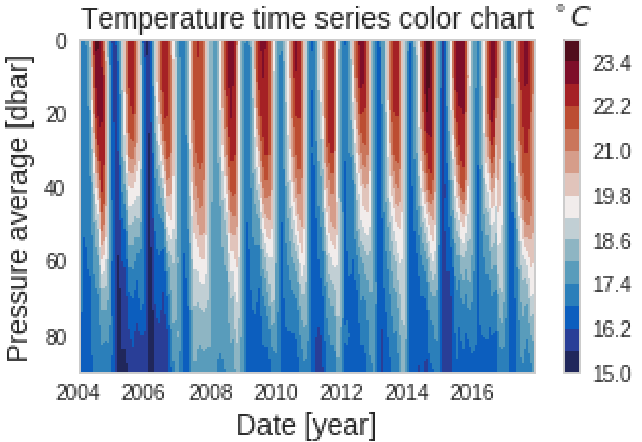


Figure: Time series at ten equally spaced partitions along the first 100 meters of the water column (10 meters/partition). Temperature is displayed as color and the y-axis is plotted as pressure (depth).

# API: Histogram on Basemap



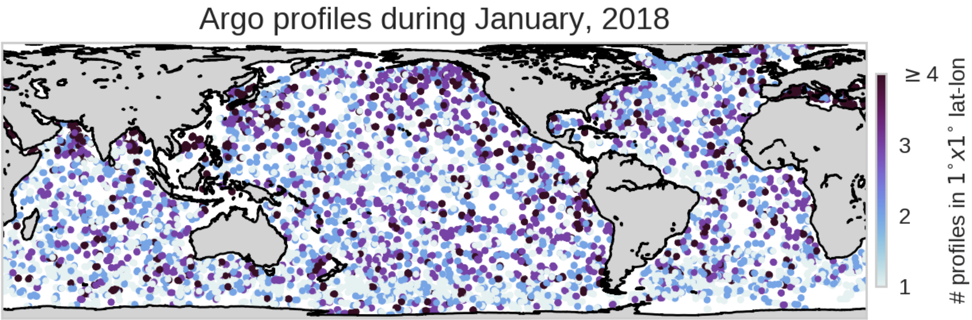


Figure: SST time series of selection. Color chart was made using a cubic interpolation of 10 different depths aggregated every 10 dbar.

# API: View Float Types



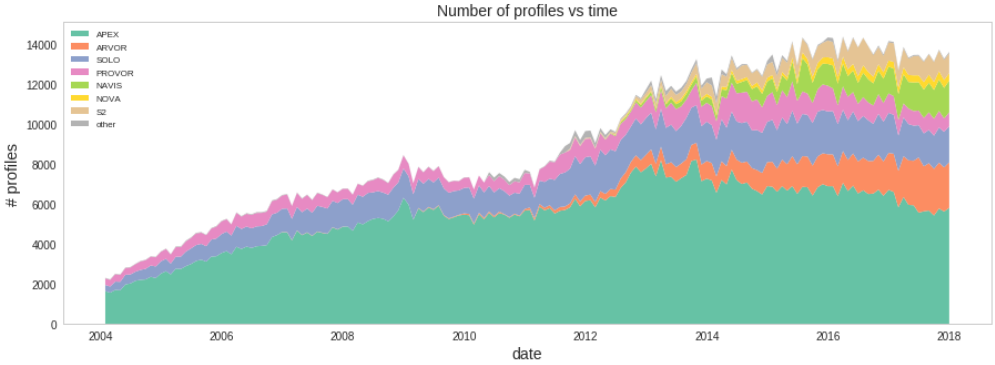


Figure: Number of profiles reported per month vs Time.

# Next Steps



- Create Kaggle.com competition using Argovis.com's database
- Include API in other languages, (R, MATLAB, Julia, etc.)
- Publish website release in a scientific journal
- Apply for a research grant for further development/testing.
- Included a gridded product section to the website.

# Table of Contents

# Conclusion



- It is possible to create websites, blogs, videos, apps with open source materials
- The methods and approaches in this thesis serves the public, specifically the next generation of scientist
- Web-based toolkits serve the user community.
- Agencies are actively pursuing similar websites (NASA, NOAA, Argo France). This thesis applies to them.