PASS-SWIO Online Training – Exercise

Exercise: Cross Validating altimeter & tide gauge sea level data

Objective:

Access the new altimeter time series data set and to validate against the Toamasina tide gauge data.

Getting the altimeter and tide gauge data

We first need to download the altimeter and tide gauge data:

- The PASS-SWIO altimeter time series data set:
  - Open web browser
  - Go to PASS-SWIO project website: www.satoc.eu/projects/pass-swio/training.html
  - Click on "Download along track satellite altimeter data"
  - Open downloaded zip archive: "altimetry_along_track.zip" (14.2 MB)
  - Save downloaded files on your laptops under \Shared\Data\C-RISe\AltimetryByPass

- Updated Toamasina Tide Gauge data
  - Go to PASS-SWIO project website: www.satoc.eu/projects/pass-swio/training.html
  - Click on "Download Toamasina Tide Gauge data"
  - Open downloaded zip archive: "HA_112_000001_058741_GS_02_TS_Toamasina.t2k.zip" (1.3 MB)
  - Copy to \Shared\Data\TideGauge\C-RISe_TG_validation\C-RISE_IOCSLMF

Python Set Up

- Laptop set up instructions
  - www.satoc.eu/projects/pass-swio/training.html
  - Click on "Guidelines for preparation of laptops"

- Python Code – new routines needed for new data.
  - www.satoc.eu/projects/pass-swio/training.html
  - Click on "Download python code archive"
  - Open downloaded zip archive: "python_code.zip"
  - Copy to \Shared\Software\Windows\SeaLevel\SeaLevelValidation\python_code
  - You may have to overwrite previous routines

Python Set Up (3.6)

- The new code is written for python 3.6, with the following libraries installed:
  - scipy
  - netCDF4
  - matplotlib
  - NB may need to reinstall older version of matplotlib – in python window conda install matplotlib=3.2

- Code should be installed in the location: *C:/Shared/Software/Windows/SeaLevel/SeaLevelValidation/python_code*
  - If you put it somewhere else, you will need to edit the python code to look in that location
  - All figures generated by the code are saved to ./figures
  - Statistics and processed data are saved to ./data

- Instructions for setting up the python environment are in the "Guidelines for preparation of laptops" document. The following instructions assume you have set up a "pass-swio" environment with the correct libraries installed.

Exercise 1: Validating Altimeter against Toamasina Tide Gauge

- Open Anaconda-Navigator application
- Select "Environments"
- Choose pass-swio and "Open Terminal"
- Change directory to the location of the code

```
> cd C:/ Shared/Software/Windows/SeaLevel/SeaLevelValidation/python_code
```

- Run the genStats code, it needs 2 arguments, the tide gauge location (e.g. *Toamasina*) and the satellite code (*j1j2j3*, *s3a*, or *s3b*)

```
>python genStats.py Toamasina j1j2j3
```

If successful the programme will output a series of figures, in the folder ./figures

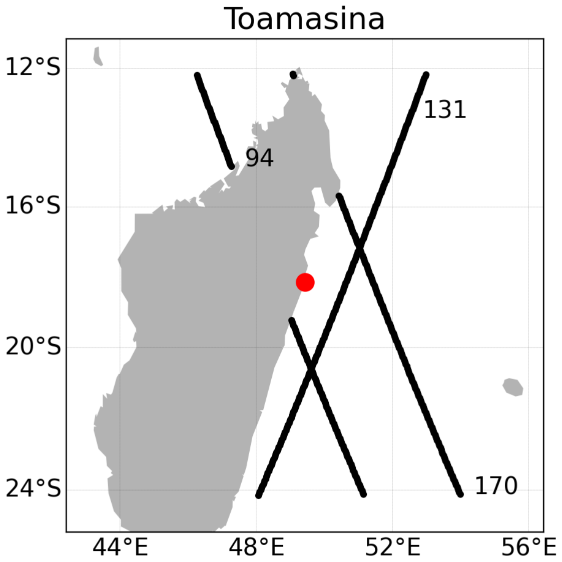
- Toamasina_passes_*satcode*.png
- corr_dist2Coast_*track*_Toamasina_*satcode*.png
- rms_dist2Coast_*track*_Toamasina_*satcode*.png
- Series_maxCorr__*track*_Toamasina_*satcode*.png

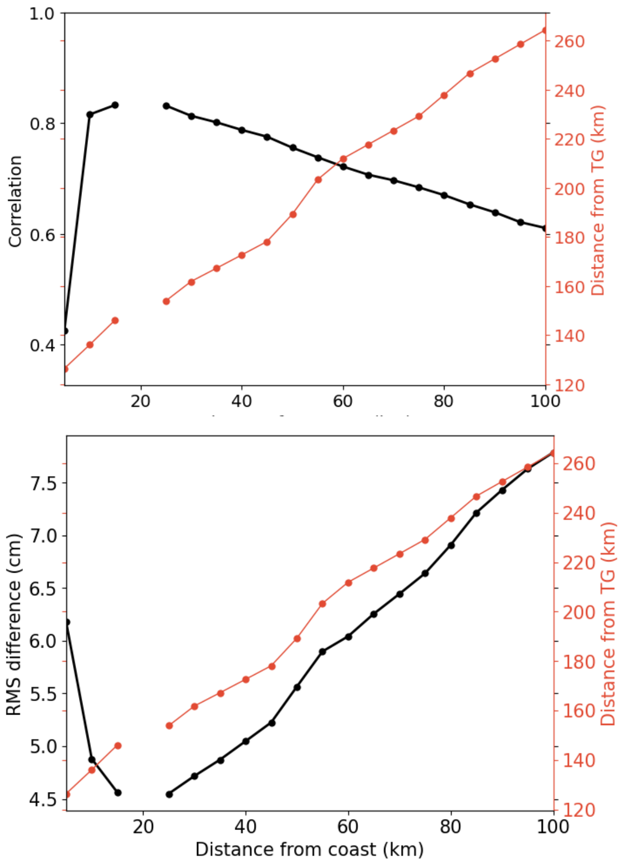
It should also have generated some data files in ./data. 3 for each satellite code (j1j2j3, s3a, or s3b)

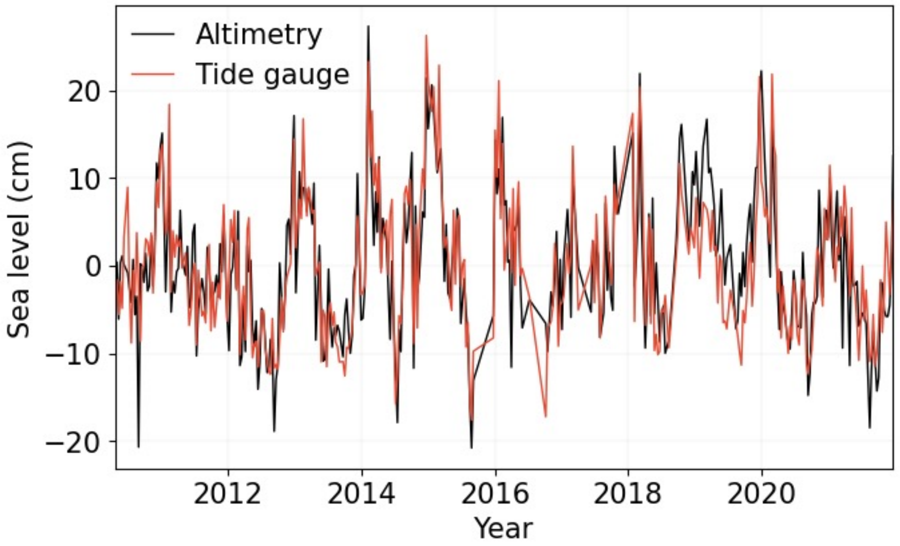
- altimetry_Toamasina_series_*satcode* (the altimeter sea level data)
- tideGauge_Toamasina_series _*satcode*, (the tide gauge sea level data at the time of the altimeter over-pass)
- altimetry_Toamasina_series_time _*satcode* (the time, in decimal years, of the altimeter / tide gauge data pairs)

These files will be in ./data.

- Try importing to Microsoft excel and reproduce the time series plot.
- You could also try to fit a long-term trend to the tide gauge and altimeter sea level data.


Toamasina
12°S
16°S
20°S
24°S
44°E
48°E
52°E
56°E
94
131
170


Correlation
Distance from TG (km)
RMS difference (cm)
Distance from coast (km)


Altimetry
Tide gauge
Sea level (cm)
Year
2012
2014
2016
2018
2020

Other python routines for plotting and extracting data

- All the following routines need the input file: "*|Shared|Software|Windows|SeaLevel|SeaLevelValidation|data|statsAltiTracks_mission*". Where "*mission*" is *j1j2j3, s3a* or *s3b*.
- This file is created by running "genStatsTracks.py" as follows:

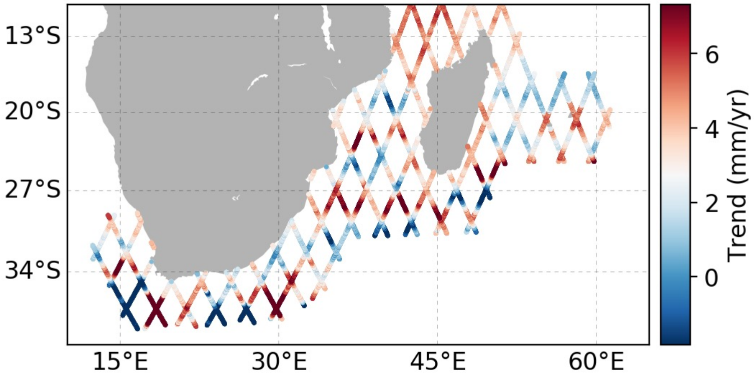
> python genStatsTracks.py *mission* (*mission* is *j1j2j3* or *s3a* or *s3b*)

(it will take some time to run)

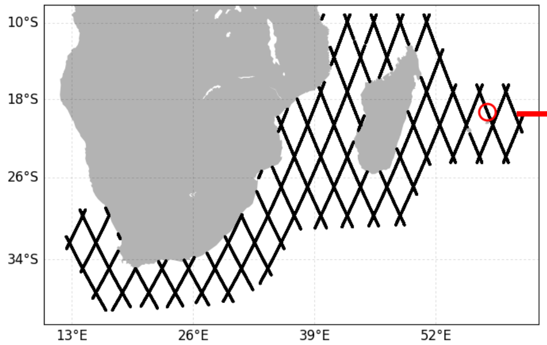
- *plotStatsMap.py* python routine to plot maps of a preselected domain of
    - amplitude and phase of the sea-level annual and semi-annual cycles,
    - the spatial distribution of sea-level trends,
    - standard deviation of detrended and de-seasoned sea-level anomalies
- *plotSeriesSelect.py* is an ipython routine that selects a point along a satellite track and plots the sea-level anomalies for that point.
- *Extract_stats_local.py* is python routine that extracts the altimeter derived sea-level statistics for a selected location

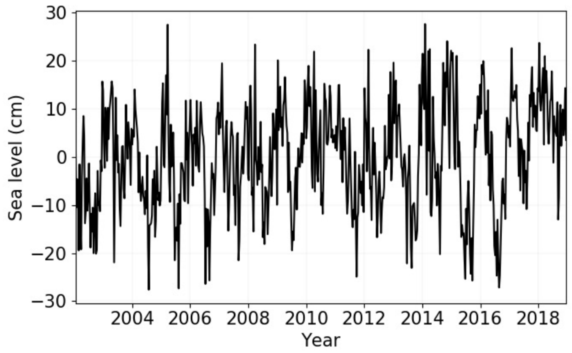
Mapping Altimeter Sea Level Data

Sea Level Trends

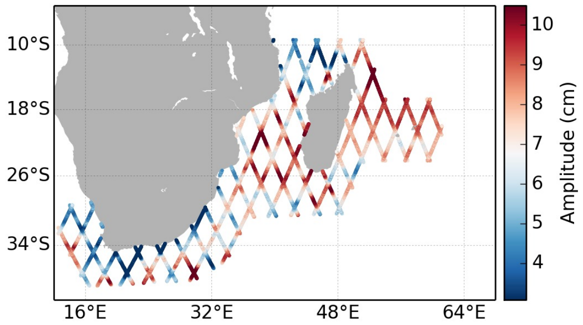


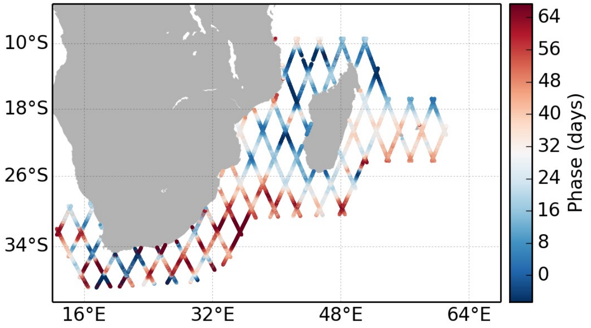
Time Series at Chosen Sites





Mapping Annual Cycles





Maps of sea level statistics - PlotStatsMap.py

- *plotStatsMap.py* python routine to plot maps of
  - amplitude and phase of the sea-level annual and semi-annual cycles,
  - the spatial distribution of sea-level trends,
  - standard deviation of detrended and de-seasoned sea-level anomalies

python plotStatsMap.py lon_min lon_max lat_min lat_max stat mission (mission is *j1j2j3*, *s3a* or *s3b*)

- *lon_min, lon_max, lat_min, lat_max* define the four corners of the area you want to plot. They represent coordinates and have units of degree.
- The fifth input "stat" denotes the statistic you want to plot:
  - *amp_an* : amplitude of the annual cycle; *phase_an* : phase of the annual cycle
  - *amp_semian* : amplitude of the semi-annual cycle; *phase_semian* : phase of the semi-annual cycle
  - *trend* : trend
  - *sdev* : standard deviation of de-trended and de-seasoned sea-level anomalies
- The input 'mission' identifies the altimetry dataset: '*j1j2j3*' (Jason missions) and '*s3a*' or '*s3b*' (Sentinel 3a / 3b).

The following will produce a map of regional trends in the domain (10°E-65°E,40°S-10°S):

python plotStatsMap.py 10 65 -40 -10 trend j1j2j3

The figure will be produced in the directory "\Shared\Software\Windows\SeaLevel\SeaLevelValidation\figures

Extracting and plotting time series for a selected point - *plotSeriesSelect.py*

- The function *plotSeriesSelect.py* selects a point along a satellite track, plots the sea-level anomalies for that point and saves the time series data to a file.

- To run it, start the "ipython" interpreter:

  > *ipython*

  > *from plotSeriesSelect import plotSeriesSelect*

  > *plotSeriesSelect('j1j2j3')*

- This function takes one parameter denoting the altimetry dataset: 'j1j2j3' (Jason missions), 's3a' or 's3b' (Sentinel 3a / 3b).
- The statement above will open a map showing all the altimetry tracks (maximize the window to see the tracks more clearly). Select a point on the tracks using the mouse. This will produce a figure of the sea level anomalies at the selected point (it might take several seconds while it extracts the data), and also a figure of the tracks with the selected location highlighted. Save the figures by clicking on the file icon.
- To exit the iPython interpreter

  > *quit()*

The sea level anomaly time series, and times (in decimal years), will be saved in separate files created under \data. They can be imported to excel and combined to generate a time series for further analysis.

”Interrogating” the altimeter statistics – extract_stats_local.py

- *extract_stats_local.py* prints the sea level trend and annual / semi-annual cycles statistics for a selected location.
- It requires the routine *tabulate.* You need to add this to the pass-swio python environment through the anaconda navigator application
- The *extract_stats_local.py* function takes three arguments:
  - loni : longitude of the site
  - lati : latitude of the site
  - mission : one of 'j1j2j3', 's3a' or 's3b’

```
> python extract_stats_local.py 51 -20 j1j2j3
```

- The function will print a table summarizing the stats.
- If the location you're requesting stats for is more than 5 degrees away from the closest altimetry measurement, the function will raise an error and print the following message: "This point is likely outside the altimetry data domain"

Questions ?