FYS-3023 Assignment 2: Altimetry

Truls Karlsen

October 15, 2024

Introduction

As we've learnt in the lectures, satellite radar altimeters can be used to measure changes in the surface elevation of the Earth's ice sheets, including the Greenland Ice Sheet (GrIS). This provides crucial data for monitoring the surface mass balance or 'health' of the ice sheet: is it gaining mass from snow accumulation or losing mass from excess melt?

The ESA Climate Office describe the need for Ice Sheet data from satellites `https://climate.esa.int/en/projects/ice-sheets-greenland/about/`

In this exercise we will process data from the ESA CryoSat-2 radar altimeter from Level 1B (radar waveforms) up to a full 3D surface elevation model of the Greenland Ice Sheet (Level 3). We will then compare it to a 30-year time series of surface elevation to investigate where and by how much the ice sheet has changed. Some code snippets are provided to give you an introduction to coding with geospatial data, while elsewhere we expect you to design a few sections of code yourself.

You can access the data required for the assignment and the latest CryoSat-2 Product Handbook here `https://uitno.box.com/s/bbthiwao5m1txaxtto103ubsqupin4l9`

1 CryoSat-2 Echogram

The 'ESA L1B' directory contains a track from CryoSat-2 while the sensor is in so-called 'Low resolution mode' (LRM) and records the radar echoes with a conventional pulse-limited method.
Original data source: CryoSat-2 Data server `https://earth.esa.int/eogateway/missions/cryosat/data`

1.1

Display the LRM track from the NetCDF file
CS_OFFL_SIR_LRM_1B_20220712T151419_20220712T151831_E001.nc

Load the radar waveform echogram from the file **pwr_waveform_20_ku** and display the echogram as an image. It should look like the following

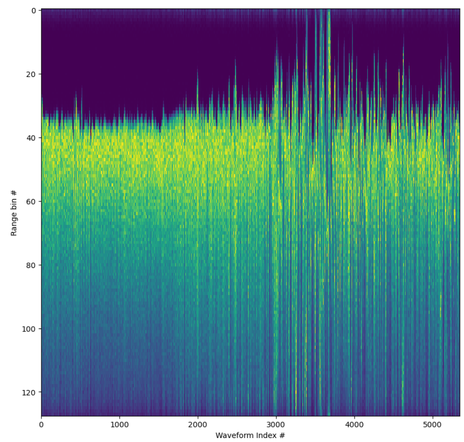

Figure 1: Echogram from task 1.1

Below is example code for opening a netCDF file in Python, and accessing its content

```
# library for opening netCDF files
import netCDF4 as nc

# path to netCDF file
netCDF_path = 'path_to_netCDF_file.nc'

# opening the netCDF file
netCDF_file = nc.Dataset(netCDF_path)

# printing out the contents of the file
print(netCDF_file)

# printing out the list of variables contained in the file
for variable in netCDF_file.variables.values():
    print(variable)

# accessing the variable 'name of variable'
netCDF_data = netCDF_file.variables['name_of_variable']
```

1.2

Each column represents a single power waveform or echo, and each row represents one range bin within the sensor's range window. There are 128 bins in the range window in LRM mode, from bin index 0:127, and the total range window is 60 m wide.

Plot one random waveform with the power normalized from zero to one. It should look like the following

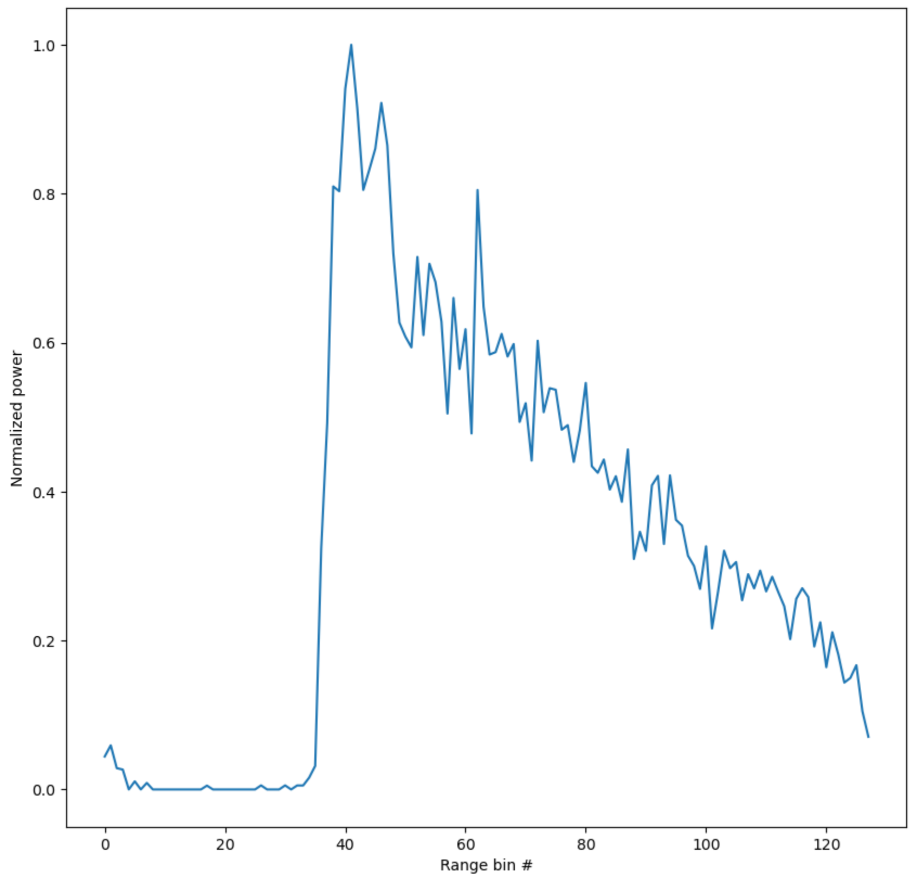

Figure 2: Normalized waveform from index 99

1.3

You can see the noise floor in the first 20-30 range bins, which corresponds to the part of the range window before the pulse hits the surface. Then you can see how the power rapidly rises on the waveform leading edge as the radar pulse intersects the surface before it falls in the trailing edge as the pulse annulus propagates over the surface. See Figure 20 in the CryoSat Product Handbook for an overview of the waveform parameters.

If you look back at the echogram you can see the range bin containing the leading edge oscillates a little. Since the surface of the GrIS varies in elevation the satellite has to keep updating the position of the range window compared to the spacecraft in order to keep the surface in the window. To illustrate this, plot the waveform at index 3680.

You can see the sensor 'loses lock' at this location and the surface is not contained in the range window. Load and plot the latitude and longitude of the observations on a basemap of Greenland to see where they've been collected. Adding a map to a plot in Python can be done through the use of the *cartopy* package

```
# loading cartopy and ticklabel formatters
import cartopy.crs as ccrs
from cartopy.mpl.ticker import LatitudeFormatter,
                               LongitudeFormatter

# initializing a figure (choose width and height.
# dpi can also be adjusted)
fig = plt.figure(figsize=[width, height], dpi=300)

# setting the geographic projection
ax = plt.axes(projection=ccrs.PlateCarree())

# plotting longitude and latitude
ax.plot(longitude, latitude)

# adding a basemap (here you can also choose other maps,
# but this is an easy way to do it)
ax.stock_img()

# adding coastlines to make a better outline
ax.coastlines()

# setting geographical extent
ax.set_extent([xmin, xmax, ymin, ymax])

# setting x and y-ticks (should be a list/array of ticks)
ax.set_xticks(xticks)
ax.set_yticks(yticks)

# formatting the ticks
lon_formatter = LongitudeFormatter(zero_direction_label=True)
lat_formatter = LatitudeFormatter()
ax.xaxis.set_major_formatter(lon_formatter)
ax.yaxis.set_major_formatter(lat_formatter)
```

The map should look something like the following

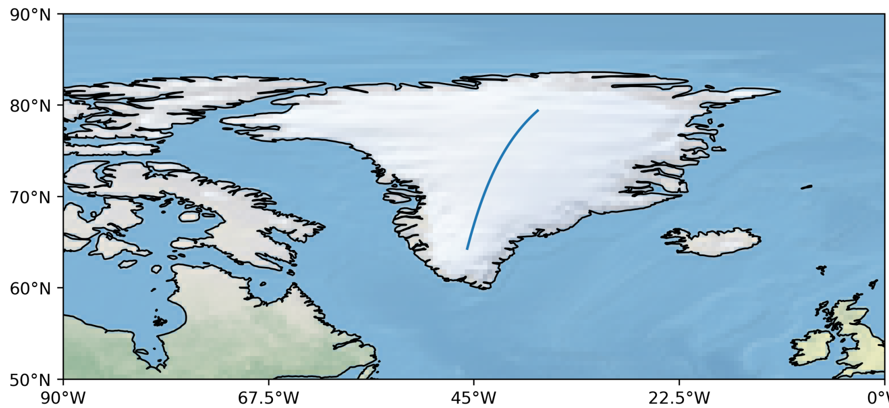

Figure 3: Satellite track over a basemap of Greenland

It is up to you how you format the map and which geographic projection you use. You can see some of the options here `https://scitools.org.uk/cartopy/docs/v0.15/crs/projections.html` and if you want to know more about geographic and projected coordinate systems, this is a great reference `https://mgimond.github.io/Spatial/chp09_0.html`.

2 Waveform Retracking

We will use a TFMRA (Threshold First-Maximum Retracking Algorithm) to identify the point on the leading edge of the waveform that we assume to represent the mean ice sheet surface elevation. For pulse-limited echoes this is at 50% of the leading edge power (see Figure 20 in the Handbook again). Remind yourself how this algorithm works by checking back to slides from the Altimetry 1 and Altimetry 2 lectures.

2.1

Design your own TFMRA algorithm. It should include the following steps applied to each waveform in the echogram, in turn:

1. Estimate the noise floor power. This can be done by taking the 1st derivative of the normalized power (P) for a given index (dP), and averaging the normalized power from the 1st instance of $dP > 0$ to $dP > 0.15$. This essentially averages the normalized power of the first 20-30 range bins.

2. Find the first clear maximum after the leading edge of the waveform (this is often the overall maximum but occasionally won't be if there is a strong secondary peak on the waveform trailing edge). The first maximum can

be identified from the 1st and 2nd derivative of P, by locating where $dP > 0.1$ and $d^2P > 0.05$. To check if the leading edge is indeed the overall maximum, check if the normalized power is larger for any of the next 10 points.

3. Normalize the echo between the noise floor and first-maximum. Set any negative values to 0.

4. Oversample the echo by several orders of magnitude in range. For python, this can be done using the *interp1d* function from *scipy.interpolate* for instance.

5. Find the fractional range bin index corresponding to 50% of the leading edge power (this is known as the 'retracking point' or 'n' in the Handbook)

There are no perfect ways to complete these steps, so just do your best! For instance, for waveform #100 the 'retracking point' is at a range bin index of approximately 37.04, where the bin indices are labelled 0:127.

Apply the retracking algorithm to every waveform in the track. You will have to filter out invalid waveforms, for instance where the range window is out of lock or the waveform has a strange shape, that you cannot retrack. Be prepared to discard up to 1000 samples. The criteria for accepting a sample is that the average of the first 10 P values is below 0.1 and that the noise floor is below 0.03. Set any sample not accepted to *NaN*. Plot the retracker output on top of the echogram image. What is the total number of discarded waveforms?

3 Finding the Surface Elevation

The retracking point represents a height of the mean ice sheet surface with respect to the range window, so now we have to place the range window in true space.

3.1

On page 46 of the CryoSat Handbook you can see the equation for calculating the range from the satellite to the central bin of the range window. To implement this you need to load the two-way window time delay **window_del_20_ku** and use the number of samples for LRM mode.

The surface height is calculated from the satellite altitude minus the range, but first we need to account for various geophysical corrections as discussed in the lectures. Add the corrections listed in Table 5 of the Handbook under *LRM-mode Interior Land Ice* to the range to obtain a corrected range. The corrections are all available in the L1B product, but only at a 1 Hz sampling whereas the waveform data is at 20 Hz. You will therefore need to associate the 1 Hz corrections to the 20 Hz measurements via **ind_meas_1hz_20_ku** based on the instructions on page 52 of the Handbook. If there are multiple options for a certain correction, you can just choose one.

The final surface heights, which are referenced to the WGS84 earth ellipsoid, should range from around 2200 to 3000 meters. Plot the result on a basemap of Greenland.

4 Creating a Monthly Map of the Surface Elevation

We need a lot of profiles from consecutive orbits of the spacecraft to create a geospatial map of the GrIS surface elevation. Lucky for us the ESA "CryoTEMPO" project is processing new data in near real-time, so we can look at a whole month of their Level 2 observations from July 2022 `http://www.cpom.ucl.ac.uk/cryotempo/index.php?hillshade=0&baseline=B&version=001&theme=landice&parameter=elevation&area=greenland_all_grounded&month=7&year=2022&availability_area=all&availability_type=files&map_or_stats_type=map`

4.1

View one of the netcdf files by using the *print* function in Python. You can see how simple they are compared to the raw L1B product with far fewer variables.

Load all the CryoTEMPO netcdf files, including the variables: latitude, longitude, surface_type, elevation and uncertainty. Convert the latitude and longitude data to a projected coordinate system, for instance the Polar Stereographic North (PSN) projection `https://epsg.io/3996` which can be done in the following way

```
# using pyproj to transform the coordinates from lat, lon
to PSN from pyproj import Transformer

# initiating a transformer object, which transforms from
# EPSG:4326 (lat, lon) to EPSG:3996 (PSN)
transformer = Transformer.from_crs('EPSG:4326', 'EPSG:3996')

# lat, lon converted to PSN
x, y = transformer.transform(latitude, longitude)
```

Only retain samples with surface type = 1, "grounded ice". Remove samples where the elevation or uncertainty is *NaN*. Plot the elevation and uncertainty on a basemap of Greenland.

4.2

Plot the elevation as a 3D model including your own processed track in another colour so you can compare the two processing chains. Does your track line up well with the CryoTEMPO data? Are there any strange outliers, and if so why?

OPTIONAL: create a quantitative comparison between your track and the nearest data from CryoTEMPO.

5 Long-term Surface Elevation Change

The final objective is to compare the altimetry surface elevation data from July 2022 to past observations, so we can see how much the GrIS might have thickened or thinned over recent years/decades.

The dataset **Surface_Elevation_Greenland_Monthly_5km_Grid.nc** comes from a 30-year time series of gridded ice sheet surface elevation from multiple present and past altimeter missions (including CryoSat-2) created by Zhang et al 2022: `https://essd.copernicus.org/articles/14/973/2022/`

5.1

In this final task we will resample the CryoTEMPO data to the same 5-km grid as used by Zhang et al. To do this, load the latitude and longitude grids from the Zhang netcdf file (Be careful to use new variable names!). The latitude and longitude grids will have to be converted to PSN coordinates.

Design your own algorithm, or use one you find online, to sample the CryoTEMPO observations onto this grid. The function *cKDTree* from *scipy.spatial* can be used for this. The radius for each grid point can be set to 2.5km, but you are free to use other search radii as well. To sample the elevation data onto the grid, all data points within a given radius of a grid point should be averaged using the normalized weights from the inverse of the uncertainty measurements

$$\bar{h} = \frac{\sum_i h(i) * w(i)}{\sum_i w(i)}$$

where $h(i)$ is the elevation measurement at index i and $w(i)$ is the weight from the inverse of the uncertainty at index i.

Plot the gridded surface on a basemap of Greenland. Don't worry if the gridded elevation data appears 'tracky' with a lot of missing grid cells - this is all the data we have for the month, so we just have to work with it! The plot should look something like Figure 4.

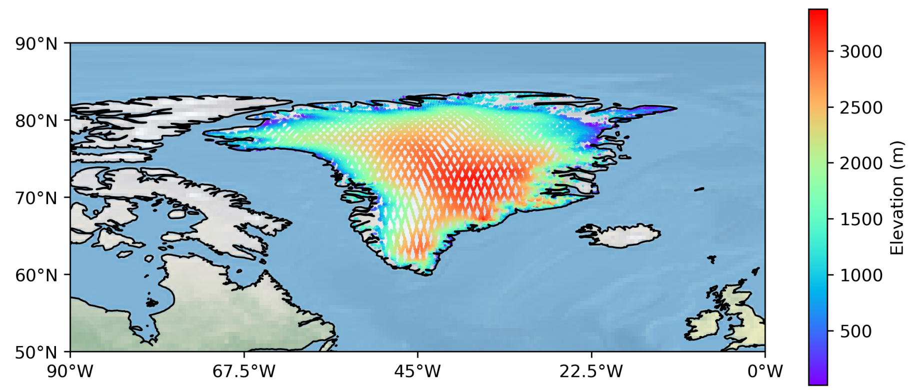

Figure 4: Gridded elevation data

5.2

Now load the time series of gridded surface elevation from the Zhang et al dataset, including the variables 'time', 'basin' and 'elev_abs'. The time is provided in fractional years from 1991 to 2020.

The basin variable shows different drainage basins for the GrIS, which you can use to make time series plots for different regions `https://earth.gsfc.nasa.gov/cryo/data/polar-altimetry/antarctic-and-greenland-drainage-systems`. Plot the basins on a map.

Add your July 2022 grid to the end of the Zhang et al gridded record. Plot a map of the elevation difference between the start and end of the CryoSat-2 record: July 2011 to July 2022.

Try plotting the elevation differences between other months of the record too.

5.3

You can also plot a time series of the mean elevation for all measurements within a certain region of the ice sheet e.g. Northwest (region 8). Due to the many missing grid cells in the July 2022 data, this data must be discarded from the time series to avoid biasing the results.

What patterns do you see? What does it suggest is changing about the Greenland Ice Sheet? Which regions/ice streams have been changing the most and when did they start changing?

What are the strengths of satellite altimetry for decadal monitoring of the Earth's ice sheets? Can you describe some limitations?

OPTIONAL: try making maps of the long-term trends for each grid cell. Where are the trends statistically significant..?