Mapping The Elevation of Pine Island Glacier between 2009 and 2015

Introduction

This report analyses the elevation of the Pine Island Glacier between 2009 and 2015, by analysing flight lines from NASA LVIS Sensor data. Five tasks have been completed, with the methodology, outputs and limitations of the python code examined. While all tasks can successfully run, there are definite improvements to be made which will be considered in this report.

Site Context

Figure One: Location of Pine Island Glacier (European Space Agency, 2023)

Pine Island Glacier (shown in figure one) is located on the West Antarctic Ice Sheet and drains an area worth two thirds of the United Kingdom (iStar, n.d). The Pine Island Glacier is among the fastest changing glaciers, rapidly retreating since the 1940's (British Antarctic Survey, 2016) and losing over a trillion tons of ice over the last decade (Rydt *et al,* 2021).

Task One

Methodology

This task utilised object orientated programming, where software is organised around objects (Gillis, 2024) by using the class PlotLVIS, inheriting from LvisData. This class reprojects the LVIS to 3031 from 4326 so it is designed for mapping Antarctica. This Lvis is then plotted using the matplotlib package, retrieving the waves information of the user specified index and finally saving as a png image.

Output

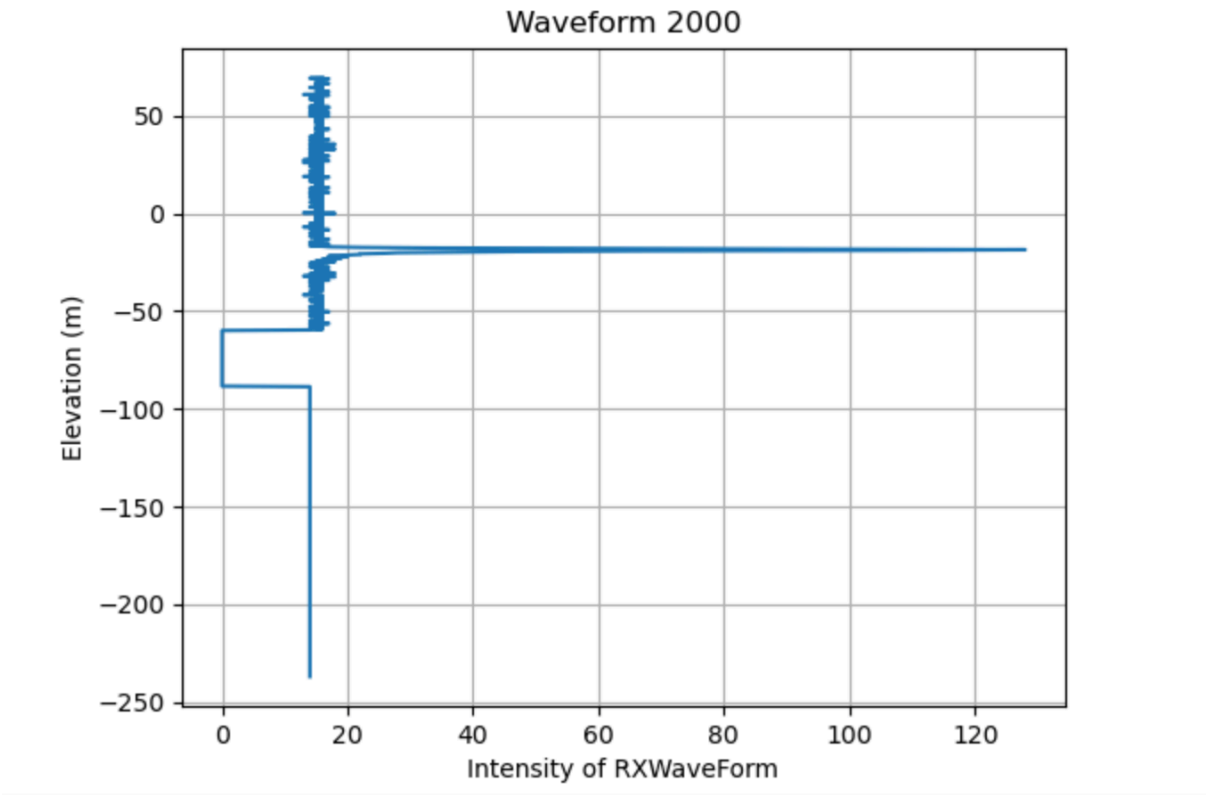

Figure Two: Graph of LVIS Waveform

Limitations and Improvements

This task utilises object orientated programming effectively to produce a matplotlib graph, fulfilling the task requirements. However, the line thickness could have been reduced to improve readability. An interesting improvement would be to add another line showing the average waveform of the whole LVIS file, enabling comparison with this specified index.

Task Two

Methodology

This task uses the class LvisGround, drawing on LVISdata, to estimate ground elevation from waveform data. This class consists of five functions to set a noise threshold, compute the centre of gravity of the denoised waveform, calculate waveform noise mean and standard deviation, denoise the waveform and convert the processed elevation data into a geotiff file.

A function is then utilised to plot the processed geotiff using the matplotlib package, opening the file using rasterio.

Output

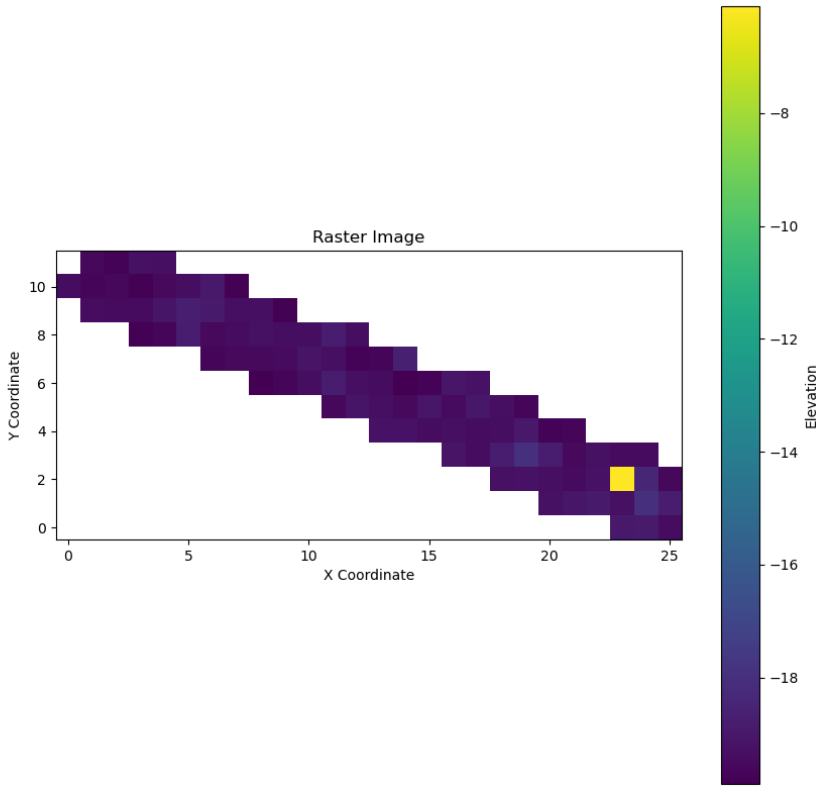

Figure Three: Graph of LVIS Geotiff

Limitations and Improvements

This task mostly fulfils the brief, showing a DEM which fluctuating elevation, however there are definite improvements that could be implemented. One of the most significant drawbacks is the resolution is still hardcoded at 0.01, despite the brief stating that a 30m resolution should be used. When a 30m resolution was implemented, the graph did not output the LVIS elevation, instead showing a blank square. A more successful output would improve upon this to allow the resolution to be set by the user. This could perhaps be solved by re-examining the coordinate system and to see if reprojection to 3031 could be applied. However, when this was previously tried it did not alter the graph output.

The function PlotTiff is also not within the class LvisGround. Despite still being functional, this omission decreases the overall readability and quality of this code, and if repeated this function would be altered to be included within this class.

Task Three

Methodology

This task initially follows the same methodology as task two to calculate and write the geotiff. However, this becomes more complex due to the amount of 2015 files to process. The function process_files processes .h5 files, calculates the bounds based on the pine glacier coordinates, making sure to convert the longitude from negative to positive to ensure an accurate output as the glacier is in Antarctica. To reduce memory processing, only every third .h5 is processed and the data is subdivided into tiles. Two subsequent functions then merge (merge_tif) and reproject (reproject_to_3031) the files through rasterio, both then being saved as png's after being plotted using matplotlib.

Output

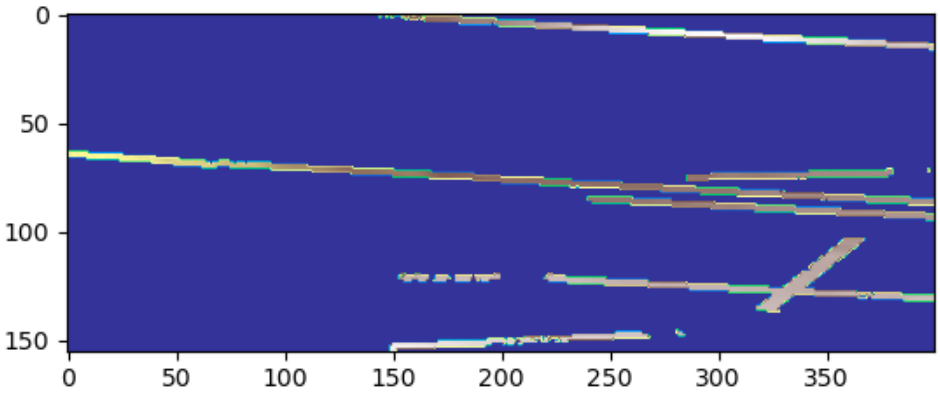

Figure Four: Merged 2015 Tiff

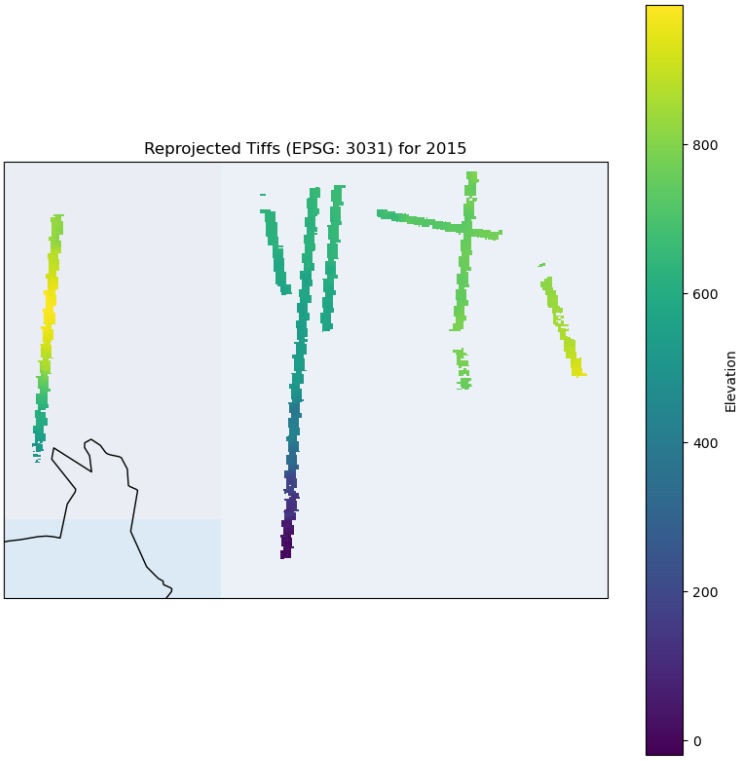

Figure Five: Reprojected Merged 2015 Tiff

Limitations and Improvements

While this code does successfully output a merged tiff, there are significant limitations to this task. The most pertinent is the failure to reduce the processing below 2GB, despite subdividing into tiles and only processing a small section of the 2015 files. Attempt was also made in LvisClass.py to slice the arrays to save memory using 'lazy loading', where data is only loaded/processed when required (CloudFlare, n.d). The memory limitation was tested in the Forth command line (--mem 2G) and through using tracemalloc (Python, n.d). A way to solve this would be to further increase the tile size and process even fewer tiles.

A further limitation is the argument parsing. The wealth of arguments needed to run this task drastically reduces readability and user accessibility. Attempt was made to simplify this by combining the merge and reproject command lines, however when altered the task did not run.

Task Four

Methodology

Task Four is included in Task_Three.py as the function fill_gaps. This code uses rasterio fill no data (Rasterio, n.d) to fill between the flight lines by interpolating the values from neighbouring valid data to fill the gaps, saving as a png using rasterio and matplotlib. An alternative method was used initially, involving reducing the resolution to 500. However, the output of the rasterio method was deemed of a higher quality and so was chosen for the final product.

As part of this task both the 2009 and 2015 data was processed, merged, reprojected and gap filled.

Output

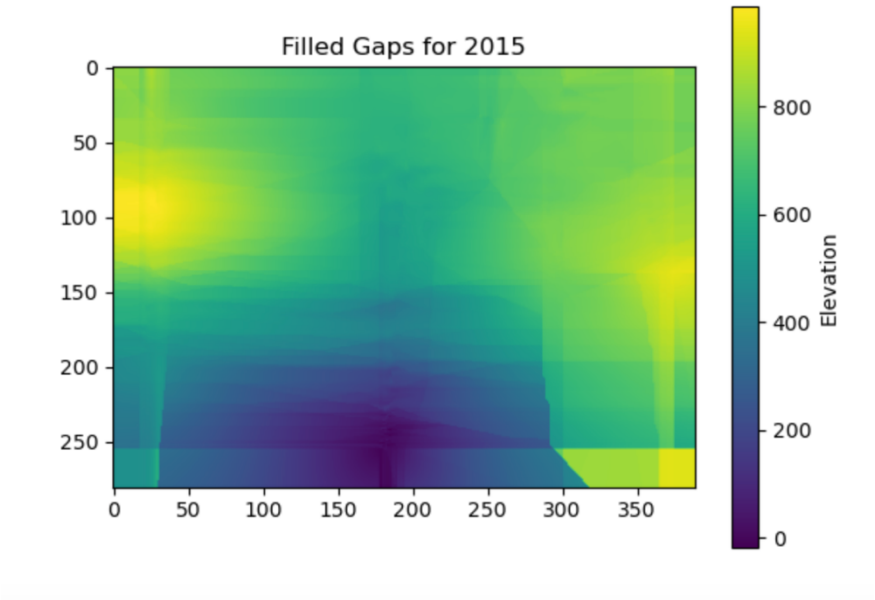

Figure Six: Gap Filled 2015 Data

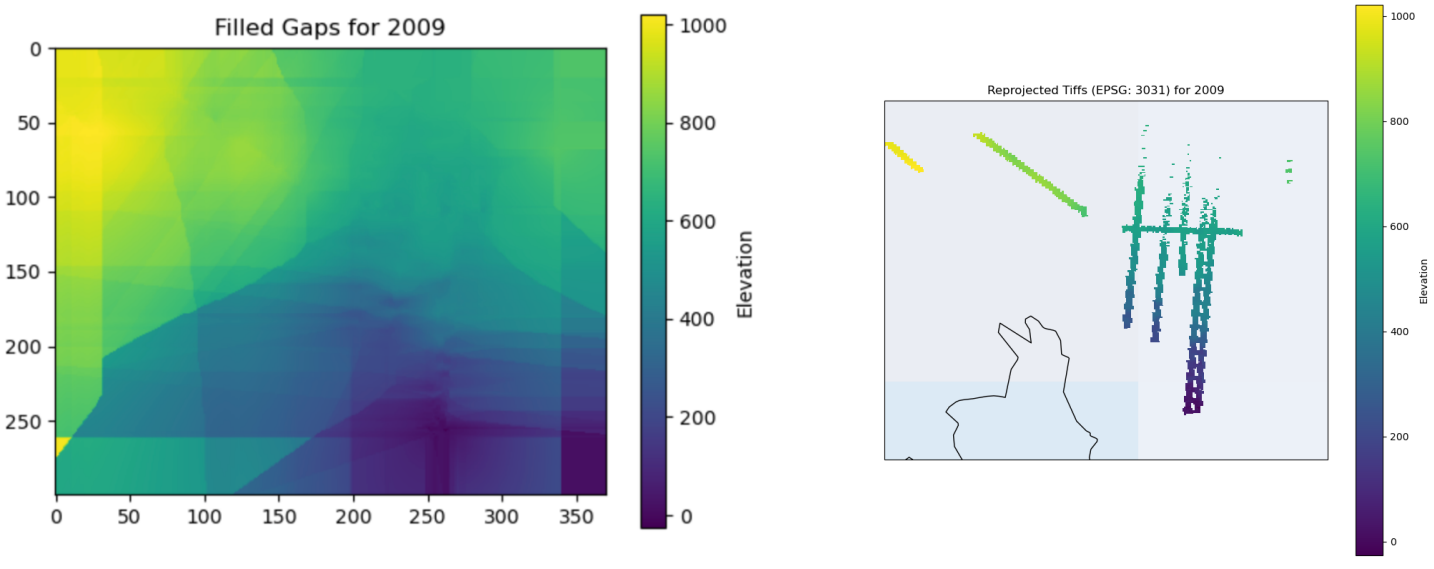

Figure Seven: Gap Filled 2009 Data (Left) and Reprojected 2009 Data (Right)

Limitations and Improvements

This was one of the most successful parts of this project. Nevertheless, the filled gaps algorithm, and by association the entire Task_Three.py file, could be improved by being altered to fit within the LvisGround class. Figure Six and Figure Seven show a successful gap fill, but this is not smooth. Increasing the smoothing alterations could have improved the final output. However, increasing this caused the entire plot to show as a white square. Further analysis of the code and rasterio documentation could potentially solve this issue.

Task Five

Methodology

This task is based around functional programming, rather than using an object orientated programming approach. The read_tiff function opens a geotiff file (the gap filled output geotiff from either 2009 or 2015) and reads the pixel values into a NumPy array, also retrieving georeferencing information. Once both geotiffs have been processed, the resample_to_match function resamples the second geotiff so it matches the reprojection and resolution of the first geotiff. The calculate_overlap function then calculates where the 2009 and 2015 data intersects and extracts this. Finally, the write_tiff and save_as_png functions write the elevation change geotiff and save this as a png.

Output

Total Ice Volume Change from 2009 to 2015: -2796046059849.919 cubic metres

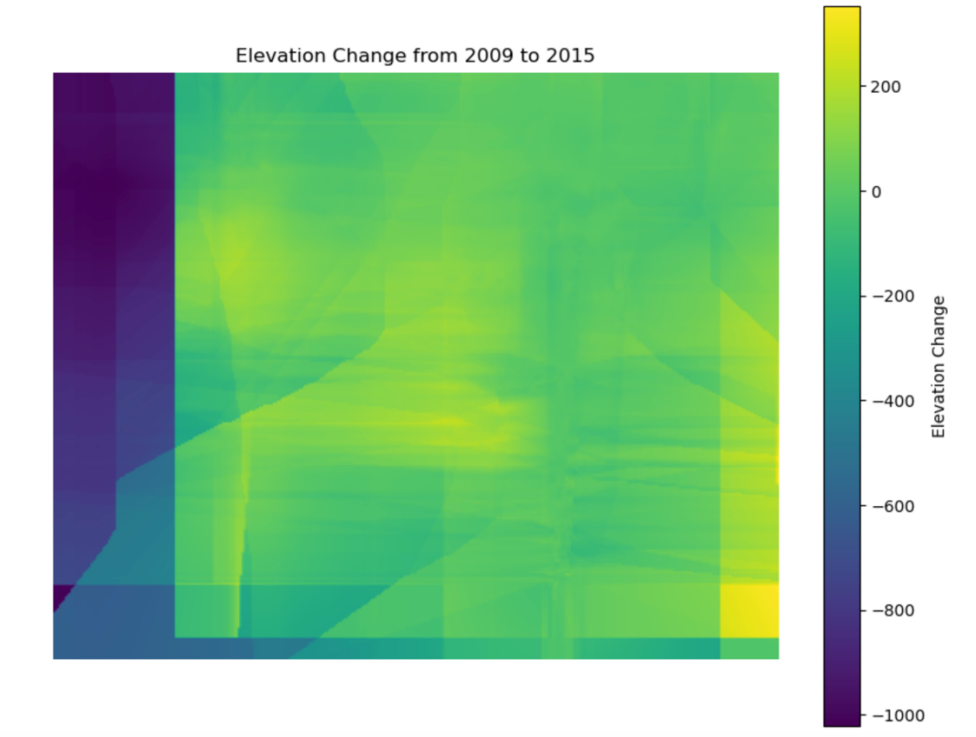

Figure Eight: Elevation Change Geotiff

Limitations and Improvements

The most obvious limitation is the lack of object orientated programming methodology. To improve this, this task could be restructured to be in a class. This task does successfully identify the negative elevation change, however Figure Eight could be further improved by increasing the resolution to

smooth the harsh lines. A base map could also be added using matplotlib to see the outline of the Pine glacier, similar to Figure Five.

Discussion

With significant improvements, these tools could be applied to a variety of scientific questions, especially glacial change. Vivero and Lambiel (2024) calculate the surface elevation decreases of Rock Glaciers in the Swiss Alps, creating DEM's from UAV Surveys. Figure Nine shows the surface elevation decreases, with elevation data drawn from DEM's. Chen *et al* (2024) also calculate elevation change on the Tibetan Plateau, using an algorithm to extract crossover points of satellite data. Applying Task Five to charting glacial retreat would be invaluable to understand climate change effects. However, improvements would need to be made to create a higher quality graph, perhaps through increasing the smoothing iterations and supplying a base map. This is due to the gap filling methodology in Task Four, and so alternative methodologies could be explored, such as 'gapfill' (Gerber, 2018) and using a random forest gap-filling model (Disarm-Platform, 2019).

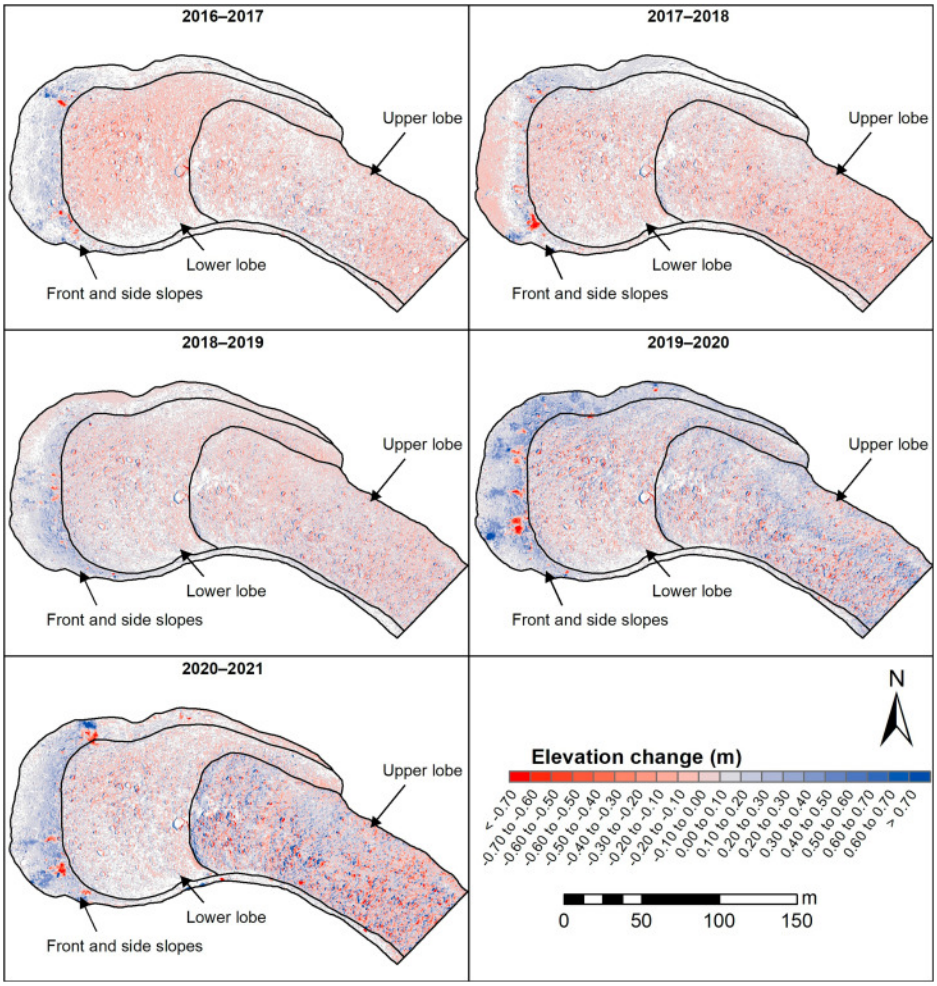

Figure Nine: Surface Elevation change maps of Les Cliosses from 2016 to 2021 (Vivero and Lambiel, 2024).

This code could be drastically improved. A main limitation was the failure in Task Three to reduce the memory processing. If this code was to be used to processed higher file quantities, this would need to be altered. Beyond this, all the code could be altered to fit within an object orientated structure to increase readability, utilising classes and objects to access the methods within a class, not creating methods outside of it. In conclusion, this code fulfils much of the brief, however, does need alterations to be deployed in a production environment or used for further research purposes.

Bibliography

British Antarctic Survey. (2016). *New study shows when Pine Island Glacier retreat began - British Antarctic Survey*. [online] Available at: https://www.bas.ac.uk/media-post/new-study-reveals-when-west-antarcticas-largest-glacier-started-retreating/ [Accessed 4 Apr. 2025].

Chen, T., Wang, J., Che, T., Hao, X. and Li, H. (2024). High spatial resolution elevation change dataset derived from ICESat-2 crossover points on the Tibetan Plateau. *Scientific Data*, [online] 11(1). doi:https://doi.org/10.1038/s41597-024-03214-2.

Cloudflare.com. (n.d). *What is lazy loading?* [online] Available at: https://www.cloudflare.com/en-gb/learning/performance/what-is-lazy-loading/.

Disarm-Platform. (2019). Gap-Filling Rasters [Online]. Available at: https://github.com/disarm-platform/gapfilling_rasters

European Space Agency. (2023). *Thwaites and Pine Island Glaciers in West Antarctica*. [online] Available at: https://www.esa.int/ESA_Multimedia/Images/2023/01/Thwaites_and_Pine_Island_Glaciers_in_West_Antarctica [Accessed 4 Apr. 2025].

Gerber, F., de Jong, R., Schaepman, M.E., Schaepman-Strub, G. and Furrer, R. (2018). Predicting Missing Values in Spatio-Temporal Remote Sensing Data. *IEEE Transactions on Geoscience and Remote Sensing*, 56(5), pp.2841–2853. doi:https://doi.org/10.1109/tgrs.2017.2785240.

Gillis, A. (2024). *What Is Object-Oriented Programming (OOP)?* [online] TechTarget. Available at: https://www.techtarget.com/searchapparchitecture/definition/object-oriented-programming-OOP.

Istar. (n.d). *iSTAR – NERC Ice Sheet Stability Programme | Fact file – Pine Island Glacier*. [online] Available at: https://www.istar.ac.uk/press-media/fact-file-pine-island-glacier/.

Python (n.d.). *tracemalloc — Trace memory allocations — Python 3.11.0 documentation*. [online] Available at: https://docs.python.org/3/library/tracemalloc.html.

Rasterio. (n.d). *rasterio.fill module — rasterio 1.5.0.dev documentation*. [online] Available at: https://rasterio.readthedocs.io/en/latest/api/rasterio.fill.html [Accessed 4 Apr. 2025].

Rydt, J., Reese, R., Paolo, F.S. and Gudmundsson, G.H. (2021). Drivers of Pine Island Glacier speed-up between 1996 and 2016. *The Cryosphere*, 15(1), pp.113–132. doi:https://doi.org/10.5194/tc-15-113-2021.

Vivero, S. and Lambiel, C. (2024). Annual surface elevation changes of rock glaciers and their geomorphological significance: Examples from the Swiss Alps. *Geomorphology*, 467, p.109487. doi:https://doi.org/10.1016/j.geomorph.2024.109487.