The Journal of Open Source Software

PolarToolkit: Python Tools for Convenient, Reproducible, and Open Polar Science

Matthew D. Tankersley [1,2]

1 Victoria University of Wellington, New Zealand **2** GNS Science, New Zealand

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor: Hugo Ledoux

Reviewers:

- @PennyHow
- @JessicaS11

Submitted: 10 February 2024
Published: unpublished

Summary

PolarToolkit (formerly known as Antarctic-Plots) is a Python package with the goal of making Polar (i.e. Antarctic, Arctic, Greenland) research more efficient, reproducible, and accessible. The software does this by providing; 1) convenient functions for downloading and pre-processing a wide range of commonly used polar datasets, 2) tools for common geospatial tasks (i.e. changing data resolution, subsetting data by geographic regions), and 3) code to easily create publication-quality maps, data profiles, and cross-sections. Additionally, `PolarToolkit` provides an easy means for exploring datasets with pre-defined or interactively-chosen geographic regions.

Statement of need

A common workflow for a geospatial scientist might be: navigate to an online repository and download a dataset, place this downloaded file in a local folder on their computer, perform some preprocessing steps, such a re-projecting, or interpolating the data, possibly using tools like GMT (Wessel et al., 2019), perform their scientific analysis on the data, and then create a map with this data using a graphical user interface (GUI) application such as QGIS (QGIS Development Team, 2024). These workflows typically require many separate tools (i.e. internet browser, file browser, spatial analysis software, and mapping software), and are often manually repeated many times throughout a manuscript revision process, and throughout the career of the scientist.

`PolarToolkit` aims to consolidate this workflow to be entirely contained within Python, making it both easier and faster to perform all these steps. Scripting workflows like this has several advantages: 1) it decreases the chance of human errors, for example using an old-version of the downloaded data or accidentally altering a pre-processing steps, such as referencing a raster of elevation data to the geoid instead of the ellipsoid, and 2) it allows entire workflows to be shared easily between collaborators with a single python file or Jupyter Notebook. Although a popular and well-designed similar package exists (Antarctic Mapping Tools, Greene et al., 2017), PolarToolkit is unique in its open-access without the need for a paid MatLab license.

Written in easy-to-learn Python, and utilizing common geospatial data structures, `PolarToolkit` is designed to be familiar to use for experienced Python users, while also being approachable for beginner coders. It is built upon several open-source packages, such a Pooch for data downloading (Uieda et al., 2020), PyGMT for creating figures (Uieda et al., 2021), and xarray and verde for geospatial data processing (Hoyer & Hamman, 2017; Uieda, 2018).

Comprehensive documentation, API reference, tutorials and how-to guides are available at https://antarctic-plots.readthedocs.io/en/, and development occurs in the GitHub repository.

PolarToolkit Modules

The key functionality of `PolarToolkit` is organized into five modules:

Module	Description
regions	Pre-defined or interactively chosen geographic regions
fetch	Functions to download, pre-process, and retrieve cached data
maps	Create high-quality maps with functions specifically tailored to polar settings
profile	Define a line, sample layers and data along it, and plot the results
utils	Useful functions for common tasks (e.g. coordinate conversion, masking)

Example

The below example demonstrates the functionality of `PolarToolkit`. Running the code will perform the following steps:

1) Download (or retrieve previously downloaded) datasets from various online repositories:
   - Surface and bed elevation, and ice thickness data from Bedmap2 (Fretwell et al., 2013)
   - Antarctic coastline and groundingline shapefiles (Depoorter et al., 2013)
   - Antarctic ice shelf boundary shapefiles (Mouginot et al., 2017)
   - Imagery data from LIMA (Bindschadler et al., 2008)
2) Pre-process the data
   - convert the Bedmap2 `.tif` files into compressed `.zarr` files
   - resample the grid resolutions from 1 km to 2 km
   - extract the portion of the grids around the Saunders Coast region
   - calculate the water column thickness ([surface - ice thickness] - bed)
   - mask the grid outside of the floating ice shelves
3) Create a map
   - plot a basemap of imagery
   - plot water column thickness data
   - add a colorbar histogram to show data distribution
   - add features like a scale bar, inset map and a title

```python
# import modules
from polartoolkit import regions, fetch, utils, maps

# define a region
region = regions.saunders_coast

# download and pre-process bedmap2 data
water_thickness = fetch.bedmap2(
  layer="water_thickness",
  region=region,
  spacing=2000)
```

```python
# mask areas outside of ice shelves
water_thickness = utils.mask_from_shp(
    shapefile=fetch.measures_boundaries(version="IceShelf"),
    xr_grid=water_thickness,
    masked=True,
    invert=False)

# plot map and set options
fig = maps.plot_grd(
    grid=water_thickness,
    cmap="dense",
    title="Saunders Coast Ice Shelves",
    cbar_label="Water column thickness (m)",
    imagery_basemap=True,
    coast=True,
    inset=True,
    scalebar=True,
    hist=True)

# display figure
fig.show()
```

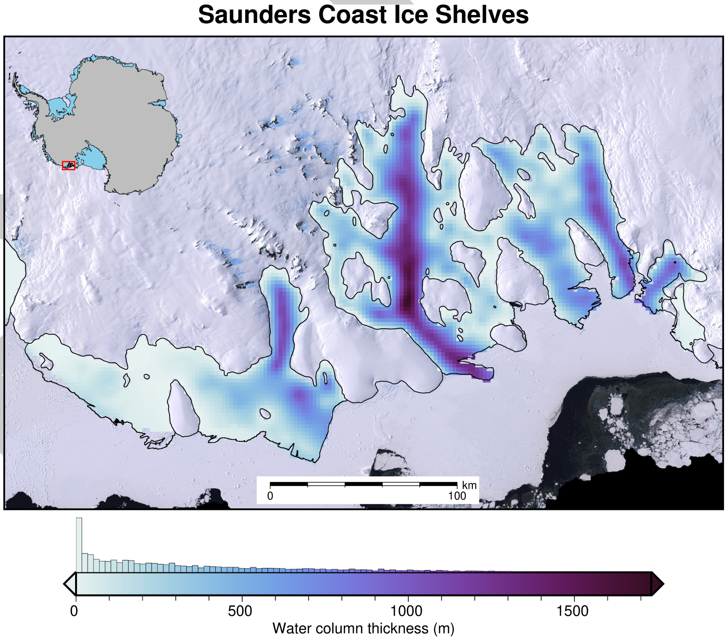

Figure 1: Example map output from above code implemented in `PolarToolkit`. Water column thickness (Fretwell et al., 2013) beneath the ice shelves of Antarctica's Saunders Coast. Inset map shows figure location. Grounding line and coastlines shown by black line (Depoorter et al., 2013). Background imagery from LIMA (Bindschadler et al., 2008). Colorbar histogram shows data distribution.

Acknowledgements

I would like to acknowledge the support in the early stages of this project from Wei Ji Leong, who taught me many of the skills necessary for developing this package, as well as the team at PyGMT, Fatiando a Terra, and the community of Software Underground for all the technical support and providing open-source software which inspired this project.

References

Bindschadler, R., Vornberger, P., Fleming, A., Fox, A., Mullins, J., Binnie, D., Paulsen, S. J., Granneman, B., & Gorodetzky, D. (2008). The Landsat Image Mosaic of Antarctica. *Remote Sensing of Environment*, *112*(12), 4214–4226. https://doi.org/10.1016/j.rse.2008.07.006

Depoorter, M. A., Bamber, J. L., Griggs, J., Lenaerts, J. T. M., Ligtenberg, S. R. M., van den Broeke, M. R., & Moholdt, G. (2013). Antarctic masks (ice-shelves, ice-sheet, and islands), link to shape file. In *In supplement to: Depoorter, MA et al. (2013): Calving fluxes and basal melt rates of Antarctic ice shelves. Nature, 502, 89-92, https://doi.org/10.1038/nature12567*. PANGAEA. https://doi.org/10.1594/PANGAEA.819147

Fretwell, P., Pritchard, H. D., Vaughan, D. G., Bamber, J. L., Barrand, N. E., Bell, R. E., Bianchi, C., Bingham, R. G., Blankenship, D. D., Casassa, G., Catania, G., Callens, D., Conway, H., Cook, A. J., Corr, H. F. J., Damaske, D., Damm, V., Ferraccioli, F., Forsberg, R., … Zirizzotti, A. (2013). Bedmap2: Improved ice bed, surface and thickness datasets for Antarctica. *The Cryosphere*, *7*(1), 375–393. https://doi.org/10.5194/tc-7-375-2013

Greene, C. A., Gwyther, D. E., & Blankenship, D. D. (2017). Antarctic Mapping Tools for Matlab. *Computers & Geosciences*, *104*, 151–157. https://doi.org/10.1016/j.cageo.2016.08.003

Hoyer, S., & Hamman, J. (2017). Xarray: N-D labeled Arrays and Datasets in Python. *Journal of Open Research Software*, *5*(1), 10. https://doi.org/10.5334/jors.148

Mouginot, J., Scheuchl, B., & Rignot, E. (2017). MEaSUREs Antarctic Boundaries for IPY 2007-2009 from Satellite Radar, Version 2. In *National Snow and Ice Data Center*. https://nsidc.org/data/nsidc-0709/versions/2.

QGIS Development Team. (2024). *QGIS geographic information system* [Manual]. QGIS Association.

Uieda, L. (2018). Verde: Processing and gridding spatial data using Green's functions. *Journal of Open Source Software*, *3*(29), 957. https://doi.org/10.21105/joss.00957

Uieda, L., Soler, S., Rampin, R., van Kemenade, H., Turk, M., Shapero, D., Banihirwe, A., & Leeman, J. (2020). Pooch: A friend to fetch your data files. *Journal of Open Source Software*, *5*(45), 1943. https://doi.org/10.21105/joss.01943

Uieda, L., Tian, D., Leong, W. J., Jones, M., Schlitzer, W., Toney, L., Grund, M., Yao, J., Magen, Y., Materna, K., Newton, T., Anant, A., Ziebarth, M., Wessel, P., & Quinn, J. (2021). *PyGMT: A Python interface for the Generic Mapping Tools*. Zenodo. https://doi.org/10.5281/zenodo.5607255

Wessel, P., Luis, J. F., Uieda, L., Scharroo, R., Wobbe, F., Smith, W. H. F., & Tian, D. (2019). The Generic Mapping Tools version 6. *Geochemistry, Geophysics, Geosystems*, *20*(11), 5556–5564. https://doi.org/10.1029/2019GC008515