# glacier_lengths

*Release 0.1.2*

**glacier_lengths contributors**

**May 23, 2022**

# CONTENTS:

Often when glacier lengths are calculated, only the glacier centerline is considered. This is arguably not a statistically representative measure for the entire front, as it just considers one point on the glacier outline. The *glacier_lengths* package aims to simplify length calculations along an arbitrary amount of lines buffered around the glacier centerline.

CHAPTER

# ONE

# SIMPLE USAGE

```
import geopandas as gpd

import glacier_lengths
from glacier_lengths import examples

# Read the example data
outlines = gpd.read_file(examples.get_example("rhone-outlines")).sort_values("year")
old_outline = outlines.iloc[0]
new_outline = outlines.iloc[1]
centerline = gpd.read_file(examples.get_example("rhone-centerline")).iloc[0]

# Generate ~40 buffered lines around the glacier centerline
old_buffered_lines = glacier_lengths.buffer_centerline(centerline.geometry, old_outline.
↪geometry)
# Cut the newly generated lines to the new_outline
new_buffered_lines = glacier_lengths.cut_centerlines(old_buffered_lines, new_outline.
↪geometry)

# Measure the lengths of the old and new glacier centerlines.
old_lengths = glacier_lengths.measure_lengths(old_buffered_lines)
new_lengths = glacier_lengths.measure_lengths(new_buffered_lines)

# Print the results.
print(f"""
{old_outline['year']}: {old_lengths.mean():.1f}±{old_lengths.std():.1f} m
{new_outline['year']}: {new_lengths.mean():.1f}±{new_lengths.std():.1f} m
""")
```

prints:

```
Downloading latest examples...

1928: 10782.8±39.5 m
2020: 9692.0±22.2 m
```

## 1.1 glacier_lengths package

### 1.1.1 Submodules

#### glacier_lengths.core module

Core functions in the glacier_lengths package.

glacier_lengths.core.**buffer_centerline**(*centerline: shapely.geometry.LineString*, *glacier_outline: shapely.geometry.MultiPolygon*, *min_radius: float = 1.0*, *max_radius: float = 50*, *buffer_count: int = 20*)

Return buffered glacier centerlines (lines parallel to the centerline).

Note that the centerline coordinates should be ordered from glacier start to glacier end.

**Parameters**

- **centerline** – The glacier centerline.
- **glacier_outline** – The glacier outline polygon.
- **min_radius** – The minimum buffer radius in georeferenced units.
- **max_radius** – The maximum buffer radius in georeferenced units.
- **buffer_count** – The amount of buffers to create. Will return approximately twice the count (one for each side).

**Returns** Multiple buffered glacier centerlines.

glacier_lengths.core.**cut_centerlines**(*centerlines: Union[shapely.geometry.LineString, shapely.geometry.MultiLineString]*, *cutting_geometry: Union[shapely.geometry.LineString, shapely.geometry.Polygon, shapely.geometry.MultiPolygon]*, *max_difference_fraction: float = 0.2*, *warn_if_not_cut: bool = True*) → Union[shapely.geometry.LineString, shapely.geometry.MultiLineString]

Cut glacier centerlines with another geometry.

The other geometry could be a glacier outline or a glacier front line.

**Parameters**

- **centerlines** – One or multiple glacier centerlines.
- **cutting_geometry** – A supported geometry to cut the centerlines with.
- **max_difference_fraction** – The maximum difference of a centerline compared to the longest centerline. This is a filtering step to not include extremely small cut centerlines. A larger value will allow more centerlines to be valid. Defaults to 0.2 (80% of the longest centerline length).
- **warn_if_not_cut** – Issue a warning if any of the centerlines were not cut by the cutting geometry.

**Returns** Cut glacier centerlines.

glacier_lengths.core.**geometry_to_line**(*geometry*) → Union[shapely.geometry.LineString, shapely.geometry.MultiLineString]

Try to convert a given geometry to a line.

**Parameters** **geometry** – A shapely geometry object.

**Raises** `ValueError` – If the geometry is in an unsupported format.

**Returns** A LineString or MultiLineString representing the given geometry.

`glacier_lengths.core.iter_geom`(*geometry*) → Iterable
Return an iterable of the geometry.

Use case: If 'geometry' is either a LineString or a MultiLineString. Only MultiLineString can be iterated over normally.

`glacier_lengths.core.measure_lengths`(*centerlines: Union[shapely.geometry.LineString, shapely.geometry.MultiLineString]*) → np.ndarray
Measure the lengths of the given glacier centerlines.

**Parameters** `centerlines` – One or multiple glacier centerlines.

**Returns** An array of lengths with shape (N,) where N is the amount of centerlines.

### glacier_lengths.examples module

Example data auxiliary functions.

`glacier_lengths.examples.download_examples`(*overwrite: bool = False*) → str
Download examples from the GitHub repo to a temporary directory.

**Parameters** `overwrite` – Overwrite the files even though they exist?

**Raises** `ValueError` – If the data could not be fetched from the GitHub repo.

**Returns** A filepath to the temporary directory.

`glacier_lengths.examples.get_example`(*name: str*)
Retrieve the path to an example file.

Files will be downloaded from GitHub if they cannot be found.

**Returns** An absolute filepath to the given example.

### glacier_lengths.plotting module

Auxiliary plotting functions.

`glacier_lengths.plotting.plot_centerlines`(*centerlines: Union[shapely.geometry.LineString, shapely.geometry.MultiLineString], glacier_outline: Optional[Union[shapely.geometry.Polygon, shapely.geometry.MultiPolygon]] = None, plt_ax: Optional[plt.Axes] = None, centerline_kwargs: dict[str, Any] = None, outline_kwargs: dict[str, Any] = None*) → None
Plot glacier centerlines.

*plt.show()* or similar has to be run to display the figure.

**Parameters**

- `centerlines` – One or multiple glacier centrelines.
- `glacier_outline` – Optional. Glacier outline to give the centerlines context.
- `plt_ax` – Optional. A matplotlib axis to draw on. Defaults to the current axis.
- `centerline_kwargs` – Optional. Keyword arguments to supply the centerline matplotlib plot() call.

- **outline_kwargs** – Optional. Keyword arguments to supply the outline matplotlib plot() call.

glacier_lengths.plotting.**plot_length_change**(*dates: list[Union[datetime, float]]*, *lengths: list[np.ndarray]*, *plt_ax: Optional[plt.Axes] = None*) → None

Plot length change as boxplots with associated errors.

len(dates) have to be equal to len(lengths)

**Parameters**

- **dates** – The dates of the length measurements.
- **lengths** – A list of length measurements (one array per date).
- **plt_ax** – Optional. A matplotlib axis to draw on. Defaults to the current axis.

### 1.1.2 Module contents

Tools to statistically measure glacier lengths.

CHAPTER

TWO

# INDICES AND TABLES

- genindex
- modindex
- search

# PYTHON MODULE INDEX

## g

# INDEX