Bathymetry Tutorial: 2

An IS2 ATL03 file have been provided for you:

* processed\_ATL03\_20190614091833\_11780301\_005\_01.h5

To run the bathymetry extraction, all scripts have been provided for you. There are 4 main scripts. Two that hold the core functions and workflow but which you will not interact with directly. You can look into these if you want to know how the code works in detail:

* bathy\_utils.py = This python script stores all of the functions that the bathymetry extraction uses. When we run the code the commands and functions are called from here.
* run\_bathy.py = A script that takes the code arguments and sticks the functions in Bathy\_utils.py together to create a single coherent workflow.

And two scripts that you will interact with:

* 2\_1\_create\_run\_shell.py = A script that calls the python files above, parsing in the inputs and arguments and creating a new file that you will run
* 2\_2\_run\_bathy.sh = The shell file that runs the bathy code and does the extraction. You will create this file in this tutorial

Firstly, open 2\_1\_create\_run\_shell.py in a code editor. Here you will see comments that explain the steps in the code.

Now you should navigate to the /code/ folder in this tutorial using your terminal/prompt, for example:

cd /Users/Bob/Documents/BathymetryTutorial/code

Now you should run:

python 2\_1\_create\_run\_shell.py

This will create the shell file 2\_2\_run\_bathy.sh. If you open this file you will see a list of the python commands that will be run. This will be a series of commands that will call run\_bathy.py with all the arguments listed. This file therefore contains all of the commands you will need to run and saves you from typing them in each time. If you wanted to run one particular command you can use the one listed in this file as an example.

Example call:

python run\_bathy.py -i ../data/processed\_ATL03\_20190614091833\_11780301\_005\_01.h5 -l 1 -th 30

To run the whole shell script we will use a program called gnu parallel. This splits up commands passed to the terminal/prompt and assigns them to the number of cores available.

parallel < 2\_2\_run\_bathy.sh

This will run the commands on all available cores so your machine will slow down dramatically if it’s being used for other tasks. To limit the number of cores used, you can use the -j flag and specify the number of cores to use.

parallel -j2 < 2\_2\_run\_bathy.sh

This will create a number of outputs in the data folder, split into png and gpkg files. The first is a png file which shows you the transect with the selected photons overlain in red. You can use these to judge whether the algorithm with the selected threshold value has worked well or not. Choose the image that looks the best and then you will see an associated gpkg file that accompanies it. This gpkg (GeoPackage) file has the selected (red) photon depths and lat/lon – you can use it as you would a shapefile.

The benefit of running the code in this way is that you don’t always know what the right threshold value will be, so it allows you to run them all and then just select out the ones that look the best. Typically, I run all lasers with just **one** threshold value and check the outputs before running them all. Sometimes the h5 files are not suitable, particularly if you download 10s of them at a time. Just running one threshold value allows you to discard ones that have no potential. You can then run the full range of thresholds on the promising looking ones.

Generally, the aim is to pick the runs with the highest signal-to-noise ratio – selecting the maximum bathy photons for the minimum number of noise photons. This can often mean sacrificing some bathy photons to make sure no noise is captured, but as the code allows you to run through lots of files with little effort, it is better to focus on building up a training bank from lots of files than focus on one file specifically. If this is needed however, you can start using very specific thresholds as needed.

In the code we have run, I would likely choose the threshold value of 20. While this does not capture all of the bathy values it selects a good sample with minimum noise. In the folder ‘other\_examples’ you can see a range of files processed with different bathymetry surfaces and with different threshold values as a demonstration of the algorithm.