Altimetry Analytical Tools User Guide

Author: Evan Burgess

evanburgess@gmail.com

April 1st, 2014

1 Background

This code was designed to enable easy access to the entire UAF Altimetry program now 20 year long dataset in a python interface that easily allows data to be analyzed and visualized. The toolset is based on a Postgres database spatially enabled with PostGIS that holds all of the altimetry data, a modified version of the RGI 4.0 called the ergi, tidewater calving fluxes (McNabb et al., 2015) and length changes (McNabb et al., 2014) from Bob-o (robertmcnabb@gmail.com) over specific intervals, and tidewater and lake calving fluxes from (Burgess et al., 2013). The toolset here, allows one the flexibility to easily select an compare and analyze any set of surveys the user wishes, and then apply those selections to an extrapolation using methods described in the supplemental section of Larsen et al., (2015).

2 Software

To work this software requires Postgres 9 or higher with the PostGIS 2.0 extension. This database can be on the local machine or on a server elsewhere. The client machine must have Python 2.7 and the following libraries:

|  |
| --- |
| * ConfigParser |
| * copy |
| * datetime |
| * errno |
| * functools |
| * glob |
| * itertools |
| * matplotlib |
| * matplotlib |
| * numpy |
| * os |
| * osgeo |
| * ppygis |
| * re |
| * scipy |
| * signal |
| * simplekml |
| * subprocess |
|  |
|  |
| * sys * time |
| * time |
| * types |
| * types |
| * unicodedata |
| * warnings |
| * xlrd |

The user will also need GDAL and OGR installed and I suggest homebrew or something similar. This all should work on a PC or on Linux/Unix as python is cross platform as is postgres but it has not been tested in this case. There are many python development environments available, PyCharm CE is a good one. The client machine also must have secure access to baird as it takes lamb output files as inputs. Anthony has his own code to derive estimates of dhdt and this method is separate from the LAMB method run on baird.

2 Postgres setup

Before going into the code, I will discuss the tables setup in postgres. First, when I started this I hadn’t worked with postgres in 8 years so I forgot about foreign keys and constraints etc. To make this robust those constraints and keys should be cleaned up both in postgres and in the code. Also need to point out there is what seems to be an odd able structure and I think it is important to discuss why it was setup this way because, now that things have changed the code can be simplified. When we started, lamb processing integrated the dhdt curve over a hypsometry derived from a polygon that was digitized by Austin or Lee. Usually they would take the RGI polygon and tweak it as they saw fit. But the extrapolation was undoubtably going to need to use the entire RGI, so I set up a bunch of tables that would allow us have multiple glacier outlines for the same glacier, which seems weird and as it turned out we didn’t need it, so a lot of the structure is extraneous. Through time I have begun to transition the code to the RGI but never had the time to fully make the transition and clean it up. So in that respect it is kinda a mess, and the more time I get to work on this the more I will get to clean this up.