pymmaster Documentation

Luc Girod, Robert McNabb, Chris Nuth

Oct 15, 2020

Contents

This is the documentation for MMASTER-workflows and pymmaster, a set of tools for the processing of improved DEMs from ASTER.

CHAPTER 1

Installation and Setup

The following is a (non-exhaustive) set of instructions for getting setup to run MMASTER on your own machine. Note that this can be an **extremely** computationally intensive process, so we don't really recommend trying to run this on your personal laptop.

As this is a (non-exhaustive) set of instructions, it may not work 100% with your particular setup. We are happy to try to provide guidance/support, **but we make no promises**.

1.1 Installing MicMac

Detailed installation instructions for MicMac on multiple platforms can be found here, but we've added a short summary to help guide through the process.

First, clone the MicMac repository to a folder on your computer (you can also do this online via github):

```
/home/bob/software:~$ git clone https://github.com/micmacIGN/micmac.git
...
/home/bob/software:~$ cd micmac
/home/bob/software/micmac:~$ git fetch
/home/bob/software/micmac:~$ git checkout IncludeALGLIB
```

This will clone the MicMac git repository to your machine, fetch the remote, and switch to the *IncludeALGLIB* branch. Check the **README.md** (or **LISEZMOI.md**) file to install any dependencies, then:

```
/home/bob/software/micmac:~$ mkdir build && cd build/
/home/bob/software/micmac/build:~$ cmake .. -DWITH_QT5=1 -DWERROR=0 -DWITH_CCACHE=OFF
...
/home/bob/software/micmac/build:~$ make install -j$n
```

where $n is the number of cores to compile MicMac with. The compiler flag **-DWERROR=0** is needed, as some of the dependencies will throw warnings that will force the compiler to quit with errors if we don't turn it off.

Finally, make sure to add the MicMac bin directory (/home/bob/software/micmac/bin in the above example) to your $PATH environment variable, in order to be able to run MicMac. You can check that all dependencies are installed by

running the following:

```
/home/bob:~$ mm3d CheckDependencies
git revision : v1.0.beta13-844-g21d990533

byte order   : little-endian
address size : 64 bits

micmac directory : [/home/bob/software/micmac/]
auxilary tools directory : [/home/bob/software/micmac/binaire-aux/linux/]

--- Qt enabled : 5.9.5
    library path:  [/home/bob/miniconda3/envs/bobtools/plugins]

make:  found (/usr/bin/make)
exiftool:  found (/usr/bin/exiftool)
exiv2:  found (/usr/bin/exiv2)
convert:  found (/usr/bin/convert)
proj:  found (/usr/bin/proj)
cs2cs:  found (/usr/bin/cs2cs
```

You should also see the following output from the **mm3d SateLib ApplyParallaxCor** command:

```
/home/bob:~$ mm3d SateLib ApplyParallaxCor
*****************************
*  Help for Elise Arg main  *
*****************************
Mandatory unnamed args :
  * string :: {Image to be corrected}
  * string :: {Paralax correction file}
Named args :
  * [Name=Out] string :: {Name of output image (Def=ImName_corrected.tif)}
  * [Name=FitASTER] INT :: {Fit functions appropriate for ASTER L1A processing (input
↪'1' or '2' : version number)}
  * [Name=ExportFitASTER] bool :: {Export grid from FitASTER (Def=false)}
  * [Name=ASTERSceneName] string :: {ASTER L1A Scene name (Only for and MANDATORY for␣
↪FitASTERv2)}
```

In a nutshell, the basic idea is: clone the MicMac git repository, then build the source code. Simple!

1.2 Optional: Preparing a python environment

If you like, you can set up a dedicated python environment for your MMASTER needs. This can be handy, in case any packages required by pymmaster clash with packages in your default environment. Our personal preference is conda, but your preferences may differ.

The git repository has a file, mmaster_environment.yml, which provides a working environment for pymmaster and conda. Once you have conda installed, simply run:

```
conda env create -f mmaster_environment.yml
```

This will create a new conda environment, called mmaster_environment, which will have all of the various python packages necessary to run pymmaster. To activate the new environment, type:

```
conda activate mmaster_environment
```

And you should be ready to go. Note that you will have to activate this environment any time you wish to run MMASTER and pymmaster scripts and tools, if it is not already activated in your terminal.

1.3 Installing MMASTER and pymmaster

Once you have MicMac installed and an (optional) python environment set up, you can install pymmaster.

First, clone the git repository:

```
git clone https://github.com/luc-girod/MMASTER-workflows.git
```

Next, use **pip** to install the scripts and python modules:

```
pip install -e MMASTER-workflows
```

Note: the *-e* allows you to make changes to the code (for example, from git updates or through your own tinkering), that will then be updated within your python install. If you run *pip install* without this option, it will install a static version of the package, and any changes/updates will have to be explictly re-installed.

Assuming that you haven't run into any errors, you should be set up. You can verify this by running:

```
mmaster_bias_correction.py -h
```

From the command line (in a non-Windows environment; *Windows instructions coming soon-ish*).

You should see the following output (or something very similar):

```
usage: mmaster_bias_correction.py [-h] [-s SLAVEDEM] [-a EXC_MASK]
                                  [-b INC_MASK] [-n NPROC] [-o OUTDIR] [-p]
                                  [-l]
                                  masterdem indir [indir ...]

Run MMASTER post-processing bias corrections, given external elevation data.

positional arguments:
    masterdem             path to master DEM/elevations to be used for bias
                          correction
    indir                 directory/directories where final, georeferenced
                          images are located.

optional arguments:
    -h, --help            show this help message and exit
    -s SLAVEDEM, --slavedem SLAVEDEM
                          (optional) name of DEM to correct. By default,
                          mmaster_bias_correction.py looks for MMASTER DEMs of
                          the form AST_L1A_..._Z.tif
    -a EXC_MASK, --exc_mask EXC_MASK
                          exclusion mask. Areas inside of this shapefile (i.e.,
                          glaciers) will not be used for coregistration [None]
    -b INC_MASK, --inc_mask INC_MASK
                          inclusion mask. Areas outside of this mask (i.e.,
                          water) will not be used for coregistration. [None]
    -n NPROC, --nproc NPROC
                          number of processors to use [1].
    -o OUTDIR, --outdir OUTDIR
                          directory to output files to (creates if not already
```

(continues on next page)

(continued from previous page)

```
                      present). [.]
  -p, --points          process assuming that master DEM is point elevations
                      [False].
  -l, --log             write output to a log file rather than printing to the
                      screen [False].
```

If you don't see this, feel free to ask for help by sending an e-mail, though it can also be helpful to google around for some solutions first. If you do send us an e-mail, be sure to include the specific error messages that you receive. Screenshots are also helpful.

Good luck!

CHAPTER 2

Running MMASTER with bias correction

This tutorial is designed to provide the basic steps for creating DEMs from raw ASTER imagery, using MMASTER.

If you are planning to run this process on many DEMs, it might be worth having a look at scripts such as *process_mmaster.sh*, or *RunMicMacAster_batch.sh*, which are designed to automatically process many, many scenes at a time.

And of course, you are always free/encouraged to write your own scripts as needed.

2.1 Downloading data

ASTER data can most easily obtained from NASA Earthdata. Be sure to search the ASTER L1A Reconstructed Unprocessed Instrument Data, not the terrain-corrected data.

Once you have found a suitable number of scenes, be sure to order the **Geotiff**-formatted data.

After some time, you'll get an e-mail containing links to download your data. **Be sure to download both the zipped image and the metadata (.met) file**. At this point, you're ready to run MMASTER.

2.2 Running MMASTER to process ASTER DEMs

2.2.1 Pre-processing/sorting

If you have downloaded one or more **strips** (i.e., more than one ASTER scene that were acquired continuously), you can use *sort_aster_strips.py* to sort these scenes into strips of typically no more than three individual scenes.

2.2.2 Processing

From there, you can use *WorkFlowASTER.sh* to run MMASTER and extract a DEM from the ASTER imagery:

```
WorkFlowASTER.sh -s SCENENAME -z "utm zone +north/south" -a -i 2
```

This will run the MMASTER workflows on a folder called SCENENAME (usually of the form AST_L1A_003MMDDYYYHHMMSS), with outputs in UTM ZONE N/S. It will use version 2 of the fitting routine (more details here), and it will create track angle maps to be used in the bias removal stages. Support is also included for Polar Stereographic coordinate systems; check *WorkFlowASTER.sh* for more details, and for other command-line options.

Depending on the number of scenes, this may take some time (on our 80-core processing server at UiO it's about 1.5 hours per 3-scene strip; around 20-30 minutes for an individual scene). As stated in *Installation and Setup*, it will likely take significantly longer on most personal laptops, and thus we don't really recommend it.

2.2.3 Post-processing and cleaning

MicMac will produce a number of intermediate files, which most users will not find useful. Once you have finished processing a directory, you can run *PostProcessMicMac.sh*, which will search each image directory in the current folder. It creates a new directory called **PROCESSED_FINAL**, with sub-directories for each image within the current folder. In that folder, it will create the following files:

- **AST_L1A_003MMDDYYYYHHMMSS_Z.tif** - the DEM, masked to the visible image (and to successful correlation values).
- **AST_L1A_003MMDDYYYYHHMMSS_V123.tif** - the orthorectified false-color composite of ASTER bands 1-3, typically at a resolution of 15 m.
- **AST_L1A_003MMDDYYYYHHMMSS_HS.tif** - the hillshade of the DEM.
- **AST_L1A_003MMDDYYYYHHMMSS_CORR.tif** - the correlation score (from 0-100) for each pixel in the DEM.
- **TrackAngleMap_3N.tif, 3B.tif** - the track pointing angle for the 3N and 3B bands for each pixel in the DEM.
- **AST_L1A_003MMDDYYYHHMMSS..zip.met** - the original metadata files downloaded for each individual scenes.

You can also remove the intermediate files from the processing directories using *CleanMicMac.sh*, which will create a new directory called **PROCESSED_INITIAL**, with sub-directories for each image within the current folder. In those folders, it will save only the final steps of the MicMac automask, correlation, orthoimage, and DEM, in addition to the zipped raw data. This is a useful way to save space, especially when dealing with hundreds or even thousands of images.

As detailed in Girod et al (2017), this process will remove a significant portion of the cross-track bias in the DEM, as well as improve the matching, but some cross-track bias will remain, as well as all of the along-track bias. In order to remove the remaining biases, we have to use the **pymmaster** python tools.

2.3 Using pymmaster to remove remaining biases

2.3.1 External sources of elevation data

In order to correct the remaining biases in the ASTER DEMs, we first need an external elevation dataset. There are number of potential sources, in addition to any DEMs that you may have, and we have listed some options here (in no particular order):

- **SRTM** - The Shuttle Radar Topography Mission (SRTM) covers the entire globe (between 60N and 56S) at a spatial resolution of approximately 30m. It was acquired using a C-band radar instrument aboard the Space Shuttle Endeavour in February 2000, and is one of the better globally-consistent products available. It can be

obtained from many different sources, including Earth Explorer. A less-complete X-band product is available from DLR.

- **ASTER GDEM** - The ASTER Global DEM is a 30m DEM produced from a mosaic of ASTER imagery acquired before 2011. Tiles can be downloaded from NASA Earthdata.
- **TanDEM-X 90m DEM** - The TanDEM-X 90m DEM is a global DEM product from TanDEM-X imagery. It is freely available from DLR, and more information can be found from DLR.
- **ArcticDEM** - The ArcticDEM covers the Arctic (most land above 60N plus all of Alaska). It is produced from high-resolution optical imagery, and is available at a resolution of 2m. More information, and downloads, can be found at the Polar Geopspatial Center.
- **REMA** (Antarctica only) - The **Reference Elevation Map of Antarctica** covers Antarctica. Like the Arctic DEM, it is produced from high-resolution optical imagery, and is provided at a resolution of 8m. More information, and downloads, can be found at the Polar Geospatial Center.
- **ICESat** - At present, ICESat is supported through pybob, though the data must be stored in a particular format. Files containing ICESat data for each of the RGI regions can be obtained from one of the authors listed below. **planning to put them on Google Drive, or somewhere similar**

When selecting a reference dataset, several considerations should be made. The highest resolution datasets are not necessarily the best options - storage and memory can be a concern, and the ASTER scenes have a resolution of at best 15m (the default MMASTER processing is 30m). Radar-based DEMs such as the TanDEM-X 90m DEM and SRTM can have penetration biases over snow, ice, and vegetation, which can have an affect on the bias removal process. ICESat is globally consistent, but sparse, with relatively poor spatial coverage at lower latitudes. The important thing is to consider your application, and be sure to familiarize yourself with the products you are using.

2.3.2 Masking non-stable terrain

In order to properly correct motion-related biases in the ASTER DEMs, it is important to mask non-stable terrain (i.e., water bodies, glaciers, large landslides). The main routine in *mmaster_tools*, **mmaster_bias_removal**, is set up to accept two types of masks:

- exclusion masks (i.e., glaciers, water bodies, landslides, areas of deforestation) - areas where large changes are known to have occured, and will therefore mask the true bias signal. These are most easily provided as a path to a shapefile outlining the areas to exclude.
- inclusion masks (i.e., land) - areas where the ground is expected to be stable, such as mask outlining land areas. This is most easily provided as a path to a shapefile outlining the areas to include (in other words, the opposite of an exclusion mask).

At present, only one mask of each type is supported. If both types are provided, the resulting mask created will be the (**symmetrical difference?**) of the two masks.

For glaciers, a good globally-complete source of data is the Randolph Glacier Inventory. In some areas where large changes have occurred in the past decade or so, it may be necessary to update the glacier mask. It's always a good idea to compare the glacier outlines provided with satellite images acquired around the same time as the ASTER scenes (including the **orthorectified** ASTER scenes produced by MMASTER), as well as images acquired around the same time as the reference dataset (where applicable).

For land and/or water masks, a good source of data is . . .

2.3.3 mmaster_bias_correction.py

Once you have the supplementary data needed, and have extracted the DEM(s) from raw ASTER imagery, you can run *mmaster_bias_correction.py*. This script is most useful for areas where the reference (master) DEM or elevation data is contained within a single, smaller file. In this case, you can run *mmaster_bias_correction.py* as follows:

```
mmaster_bias_correction.py path/to/reference_dem.tif AST* -a path/to/exc_mask -b path/
↪to/inc_mask
```

This will run on each subdirectory of the form AST*, masking unstable terrain using both an inclusion and an exclusion mask, as discussed above. It will run each directory in sequence, printing the log to stdout. Results will be stored in the default folder *biasrem* within each subdirectory.

2.3.4 bias_correct_tiles.py

For DEMs, or groups of DEMs, that cover a larger area, you can run the bias correction routines by using DEM tiles, rather than a single DEM covering the whole region. Before running *bias_correct_tiles.py*, you first have to produce a shapefile of the extents of the DEM tiles. You can do this by first navigating to the directory where you have stored the DEM tiles, and running image_footprint.py:

```
image_footprint.py *.tif -o Reference_Tiles.shp
```

This will produce a shapefile, Reference_Tiles.shp, which contains a footprint for each .tif file in the directory, as well as attributes with the path and filename of each DEM tile. *bias_correct_tiles.py* will use this path, and the shapefile, to create a Virtual Dataset (vrt) with each of the files - thus, it's important to note that the tiles have the same spatial reference system.

With a shapefile of reference DEM tiles, you can run *bias_correct_tiles.py* in the **PROCESSED_FINAL** directory created earlier. The following example will run on all folders in **PROCESSED_FINAL**, using 4 cores and writing results to a subdirectory of each DEM folder called biasrem:

```
bias_correct_tiles.py path/to/Reference_Tiles.shp AST* -a path/to/exclusion_mask -b␣
↪path/to/inclusion_mask -n 4 -o biasrem
```

When using more than one core, *bias_correct_tiles.py* will, like *bias_correct_tiles.py*, write a log for each directory to a log file in that directory, so as to not clutter up your screen with multiple outputs from multiple DEMs at the same time. The end results will be the same as when running *mmaster_bias_correction.py* - the only difference is the input reference DEM.

Good luck!

CHAPTER 3

pymmaster

pymmaster modules and scripts, sorted by language.

3.1 bash

pymmaster bash scripts

3.1.1 CleanMicMac.sh

3.1.2 Link_MMASTER_Files.sh

3.1.3 PostProcessMicMac.sh

3.1.4 RunMicMacAster_batch.sh

3.1.5 WorkFlowASTER.sh

3.1.6 WorkFlowASTER_onescene.sh

3.1.7 WorkFlowASTER_onestrip.sh

3.1.8 WorkFlow_WaterMask.sh

3.1.9 process_mmaster.sh

3.2 python

pymmaster python modules and shell scripts

3.2.1 scripts

apply_mmaster_corrections.py

Apply MMASTER post-processing bias corrections to a MMASTER DEM.

```
usage: apply_mmaster_corrections.py [-h] [-i INPUT_DIR] [-m CORR_MASK]
                                    [-t THRESHOLD] [-c CROSS_TRACK_PARAMS]
                                    [-l] [-f ALONG_TRACK_FULL]
                                    [-w ALONG_TRACK_LOW] [-o OUTFILENAME]
                                    dem
```

Positional Arguments

dem Filename of DEM to load.

Named Arguments

-i, --input_dir Input directory to use. Default is current working directory.

Default: "."

-m, --corr_mask (optional) filename of correlation mask to apply.

-t, --threshold (optional) correlation threshold to use. Default is 60.

Default: 60.0

-c, --cross_track_params (Relative) path to cross-track correction parameter file. Default is [input_dir]/biasrem/params_CrossTrack_Polynomial.txt

Default: "biasrem/params_CrossTrack_Polynomial.txt"

-l, --low_freq Only apply low-frequency along-track corrections. Default applies full frequency.

Default: False

-f, --along_track_full (Relative) path to full-frequency along-track correction parameter file. Default is [input_dir]/biasrem/params_AlongTrack_SumofSines.txt

Default: "biasrem/params_AlongTrack_SumofSines.txt"

-w, --along_track_low (Relative) path to low-frequency along-track correction parameter file. Default is [input_dir]/biasrem/params_AlongTrack_SumofSines_lowfreq.txt

Default: "biasrem/params_AlongTrack_SumofSines_lowfreq.txt"

-o, --outfilename Output filename for corrected DEM. Default is inputdem_XAJ.tif

batch_coregister_tiles.py

Iteratively calculate co-registration parameters for two DEMs, as seen in Nuth and Kaeaeb (2011).

```
usage: batch_coregister_tiles.py [-h] [-a EXC_MASK] [-b INC_MASK] [-n NPROC]
                                 [-p] [-f]
                                 master_tiles infiles [infiles ...]
```

Positional Arguments

master_tiles path to master DEM tiles to be used for co-registration

infiles path to slave DEM(s) to be co-registered

Named Arguments

-a, --exc_mask Glacier mask. Areas inside of this shapefile will not be used for coregistration [None]

-b, --inc_mask Land mask. Areas outside of this mask (i.e., water) will not be used for coregistration. [None]

-n, --nproc number of processors to use [1].

Default: 1

-p, --points Process assuming that master DEM is ICESat data [False].

Default: False

-f, --full_ext Write full extent of master DEM and shifted slave DEM. [False].

Default: False

bias_correct_tiles.py

Run MMASTER post-processing bias corrections, given external elevation data.

```
usage: bias_correct_tiles.py [-h] [-s SLAVEDEM] [-a EXC_MASK] [-b INC_MASK]
                             [-n NPROC] [-t TMPDIR] [-o OUTDIR] [-p] [-l] [-z]
                             master_tiles indir [indir ...]
```

Positional Arguments

master_tiles Shapefile of master DEM footprints to be used for bias correction

indir directory/directories where final, georeferenced images are located.

Named Arguments

-s, --slavedem (optional) name of DEM to correct. By default, mmaster_bias_correction.py looks for MMASTER DEMs of the form AST_L1A_003…_Z.tif

-a, --exc_mask exclusion mask. Areas inside of this shapefile (i.e., glaciers) will not be used for coregistration [None]

-b, --inc_mask inclusion mask. Areas outside of this mask (i.e., water) will not be used for coregistration. [None]

-n, --nproc number of processors to use [1].

Default: 1

-t, --tmpdir tmp directory to process files to [None]

-o, --outdir	directory to output files to (creates if not already present). [.] Default: "biasrem"
-p, --points	process assuming that master DEM is point elevations [False]. Default: False
-l, --log	write output to a log file rather than printing to the screen [False]. Default: False
-z, --zip	extract from zip archive, keep a minimum of output files (logs and pdfs only) [False]. Default: False

create_micmac_xml.py

Given a GDAL dataset, create a MicMac xml worldfile.

```
usage: create_micmac_xml.py [-h] [-m MASK] [-g GEOM] filename
```

Positional Arguments

filename	List of DEM files to read and stack.

Named Arguments

-m, --mask	Path to mask file [./MEC-Malt/Masq_STD-MALT_DeZoom1.tif] Default: "./MEC-Malt/Masq_STD-MALT_DeZoom1.tif"
-g, --geom	MicMac Geometry name [eGeomMNTEuclid] Default: "eGeomMNTEuclid"

mmaster_bias_correction.py

Run MMASTER post-processing bias corrections, given external elevation data.

```
usage: mmaster_bias_correction.py [-h] [-s SLAVEDEM] [-a EXC_MASK]
                                  [-b INC_MASK] [-n NPROC] [-o OUTDIR] [-p]
                                  [-l]
                                  masterdem indir [indir ...]
```

Positional Arguments

masterdem	path to master DEM/elevations to be used for bias correction
indir	directory/directories where final, georeferenced images are located.

Named Arguments

-s, --slavedem (optional) name of DEM to correct. By default, mmaster_bias_correction.py looks for MMASTER DEMs of the form AST_L1A_003…_Z.tif

-a, --exc_mask exclusion mask. Areas inside of this shapefile (i.e., glaciers) will not be used for coregistration [None]

-b, --inc_mask inclusion mask. Areas outside of this mask (i.e., water) will not be used for coregistration. [None]

-n, --nproc number of processors to use [1].

Default: 1

-o, --outdir directory to output files to (creates if not already present). [.]

Default: "biasrem"

-p, --points process assuming that master DEM is point elevations [False].

Default: False

-l, --log write output to a log file rather than printing to the screen [False].

Default: False

mosaic_micmac_tiles.py

Re-stitch images tiled by MicMac.

```
usage: mosaic_micmac_tiles.py [-h] [-filename FILENAME] [-imgdir IMGDIR]
```

Named Arguments

-filename MicMac filename to tile [Z_Num9_DeZoom1_STD-MALT]

Default: "Z_Num9_DeZoom1_STD-MALT"

-imgdir Directory containing images [.]

Default: "."

sort_aster_strips.py

Sort ASTER scenes into folders based on whether or not they form continuous strips.

```
usage: sort_aster_strips.py [-h] [--folder FOLDER] [--max_length MAX_LENGTH]
```

Named Arguments

--folder Folder with raw, unsorted ASTER data.

--max_length Maximum number of scenes to put into a strip [3].

Default: 3

3.2.2 modules

mmaster_tools

pymmaster.mmaster_tools provides most of the routines used for removing jitter-related bias in ASTER DEMs produced using MicMac ASTER.

pymmaster.mmaster_tools.**calculate_dH**(*mst_dem*, *slv_dem*, *pts*)

Calculate difference between master and slave DEM. Master and slave DEMs must have the same shape. First, differences are calculated, then data are filtered using a 5x5 median filter, and differences greater than 100 m, or differences on high (>50deg) or low (< 0.5deg) slopes are removed.

Parameters

- **mst_dem** – master DEM (i.e., external DEM/ICESat data)
- **slv_dem** – slave DEM (i.e., ASTER scene)
- **pts** – True if master DEM is ICESat data, False if master DEM is a DEM.

Returns dH: filtered elevation differences between master and slave DEMs.

pymmaster.mmaster_tools.**fitfun_sumofsin**(*xx*, *p*)

The Base function for the sum of sines fitting to elevation differences in the along track (xx) direction of the ASTER flightpath. DEPRECATED - uses only one angle instead of two. see fitfun_sumofsin_2angle

pymmaster.mmaster_tools.**fitfun_sumofsin_2angle**(*xxn*, *xxb*, *p*)

Fit sum of sines function for two track angles.

Parameters

- **xxn** (*array-like*) – along-track distance in nadir direction
- **xxb** – along-track distance in back-looking direction
- **p** (*array-like*) – sum of sines parameters

Returns sum_of_sines sum of sines evaluated at the given along-track distances.

pymmaster.mmaster_tools.**get_aster_footprint**(*gran_name*, *proj4='+units=m +init=epsg:4326'*, *indir=None*, *polyout=True*)

Create shapefile of ASTER footprint from .met file

Parameters

- **gran_name** (*str*) – ASTER granule name to use; assumed to also be the folder in which .met file(s) are stored.
- **proj4** (*str, dict*) – proj4 representation for coordinate system to use [default: '+units=m +init=epsg:4326', WGS84 Lat/Lon].
- **indir** (*str*) – Directory to search in [default: current working directory].
- **polyout** (*bool*) – Create a shapefile of the footprint in the input directory [True].

Returns footprint shapely Polygon representing ASTER footprint, reprojected to given CRS.

pymmaster.mmaster_tools.**get_atrack_coord**(*mst_dem*, *myangN*, *myangB*)

Createx numpy arrays for along- and cross-track distances from DEM and angle maps.

Parameters

- **dem** (*pybob GeoImg*) – DEM to get x,y positions from
- **myangN** (*pybob GeoImg*) – MMASTER 3N track angle (i.e., track angle in nadir image)

- **myangB** (*pybob GeoImg*) – MMASTER 3B track angle (i.e., track angle in back-looking image)

Returns yyn, yyb along-track distances in 3N and 3B track directions.

pymmaster.mmaster_tools.**get_fit_variables**(*mst_dem*, *slv_dem*, *xxn*, *pts*, *xxb=None*)
Get input variables for bias fitting.

Parameters

- **mst_dem** (*pybob GeoImg, ICESat*) – Master DEM to use for fitting
- **slv_dem** (*pybob GeoImg*) – Slave DEM to use for fitting
- **xxn** (*array-like*) – along- or cross-track distance in the nadir track direction
- **pts** (*bool*) – True if mst_dem is ICESat points, False if mst_dem is a GeoImg.
- **xxb** (*array-like*) – along- or cross-track direction in the back-looking track direction. If not provided, fit variables are provided only for one angle.

Returns xx, dH, grp_xx, grp_dH, xx2, grp_sts track distance, elevation differences, group distances, group median dH group values, group distances in the back-looking direction, group statistics.

pymmaster.mmaster_tools.**get_group_statistics**(*invar*, *indata*, *indist=500*)
Calculate statistics of groups of pixels grouped by along-track distance.

Parameters

- **invar** (*array-like*) – Input array that determines how to group data (i.e., along-track distance)
- **indata** (*array-like*) – Data to calculate group statistics for.
- **indist** (*float*) – Distance by which to group pixels.

Returns grp_stats group statistics for input data.

pymmaster.mmaster_tools.**get_track_angle**(*fprint*, *track_dist*)
Calculate the angle made by the ASTER flight track from scene footprint.

Parameters

- **fprint** (*shapely Polygon*) – Footprint of ASTER scene
- **track_dist** (*float*) – Distance along flight track within scene at which to calculate angle.

Returns track_angle angle, in degrees, of flight track

pymmaster.mmaster_tools.**get_xy_rot**(*dem*, *myang*)
Rotate x, y axes of image to get along- and cross-track distances.

Parameters

- **dem** (*pybob GeoImg*) – DEM to get x,y positions from.
- **myang** (*float*) – angle by which to rotate axes

Returns xxr, yyr arrays corresponding to along (x) and cross (y) track distances.

pymmaster.mmaster_tools.**make_group_id**(*inmat*, *grpid*)
Make a unique ID for statistical analysis.

Parameters

- **inmat** (*array-like*) – Input array to create ID for.

- **grpid** (*array-like*) – Array of input group IDs

Returns outmat Output of group IDs

pymmaster.mmaster_tools.**make_mask**(*inpoly*, *pts*, *raster_out=False*)

Create a True/False mask to determine whether input points are within a polygon.

Parameters

- **inpoly** (*shapely Polygon*) – Input polygon to use to create mask.
- **pts** (*array-like, pybob.GeoImg*) – Either a list of (x,y) coordinates, or a GeoImg. If a list of (x,y) coordinates, make_mask uses shapely.polygon.contains to determine if the point is within inpoly. If pts is a GeoImg, make_mask uses ogr and gdal to "rasterize" inpoly to the GeoImg raster.
- **raster_out** (*bool*) – Kind of output to return. If True, pts must be a GeoImg.

Returns mask An array which is True where pts lie within inpoly and False elsewhere.

pymmaster.mmaster_tools.**mask_raster_threshold**(*rasname*, *maskname*, *outfilename*, *threshold=50*, *datatype=<MagicMock id='140562390329664'>*)

Filter values in an input raster if the correlation mask raster falls below a threshold value. Removes noise from low-correlation areas (i.e., clouds) using a binary opening operation.

Parameters

- **rasname** (*str, pybob.GeoImg*) – path to raster (i.e., DEM) to mask, or GeoImg object.
- **maskname** (*str, pybob.GeoImg*) – path to correlation raster to determine mask values, or GeoImg object.
- **outfilename** (*str*) – filename to write masked raster to.
- **threshold** (*float*) – correlation value to determine masked values.
- **datatype** (*numpy datatype*) – datatype to save raster as.

Returns masked_ras GeoImg of masked input raster.

pymmaster.mmaster_tools.**mmaster_bias_removal**(*mst_dem*, *slv_dem*, *glacmask=None*, *landmask=None*, *pts=False*, *work_dir='.'*, *tmp_dir=None*, *out_dir=None*, *return_geoimg=True*, *write_log=False*, *zipped=False*, *robust=True*)

Removes cross track and along track biases from MMASTER DEMs.

Parameters

- **mst_dem** (*str, pybob.GeoImg, pybob.ICESat*) – Path to filename or GeoImg dataset representing "master" DEM or ICESat.
- **slv_dem** (*str, pybob.GeoImg*) – Path to filename or GeoImg dataset representing "slave" DEM (developed for ASTER).
- **glacmask** (*str*) – Path to shapefile representing points to exclude from co-registration consideration (i.e., glaciers).
- **landmask** (*str*) – Path to shapefile representing points to include in co-registration consideration (i.e., stable ground/land).

- **pts** (*bool*) – If True, program assumes that masterDEM represents point data (i.e., ICESat), as opposed to raster data. Slope/aspect are then calculated from slaveDEM. masterDEM should be a string representing an HDF5 file continaing ICESat data.
- **work_dir** (*str*) – Location where output files and directory should be saved [.]
- **tmp_dir** – Location where files are processed
- **out_dir** (*str*) – Location to save bias removal outputs.
- **return_geoimg** (*bool*) – Return GeoImg objects of the corrected slave DEM and the co-registered master DEM [True]
- **write_log** (*bool*) – Re-direct stdout, stderr to a log file in the work directory [False]
- **zipped** (*bool*) – extract from zip archive, keep a minimum of output files (logs and pdfs only) [False]
- **robust** (*bool*) – solve low-frequency and jitter components of along-track bias separately. If spatial sampling is limited, this is a better approach than solving along-track as one single step.

Returns slv_corr, mst_coreg corrected MMASTER DEM, co-registered master DEM (if return_geoimg)

pymmaster.mmaster_tools.**nmad**(*data*)

Calculate the Normalized Median Absolute Deviation (NMAD) of the input dataset.

Parameters **data** (*array-like*) – input data on which to calculate NMAD

Returns nmad NMAD of input dataset.

pymmaster.mmaster_tools.**orient_footprint**(*fprint*)

Orient ASTER footprint coordinates to be clockwise, with upper left coordinate first.

Parameters **fprint** (*shapely polygon*) – footprint to orient

Returns o_fprint re-oriented copy of input footprint.

pymmaster.mmaster_tools.**plot_bias**(*xx*, *dH*, *grp_xx*, *grp_dH*, *title*, *pp*, *pmod=None*, *smod=None*, *plotmin=None*, *txt=None*)

data : original data as numpy array (:,2), x = 1col,y = 2col gdata : grouped data as numpy array (:,2) pmod,smod are two model options to plot, numpy array (:,1)

pymmaster.mmaster_tools.**polynomial_fit**(*x*, *y*)

[DEPRECATED] A polynomial search function for orders 1-6 given dependent (x) and independent (y) variables. Uses the numpy polynomial package.

pymmaster.mmaster_tools.**preprocess**(*mst_dem*, *slv_dem*, *glacmask=None*, *landmask=None*, *work_dir='.'*, *out_dir='biasrem'*, *pts=False*)

Pre-process ASTER scene to enable cross- and along-track corrections. Co-registers the ASTER (slave) and external (master) DEMs/ICESat, and shifts the orthoimage and correlation mask based on the offset calculated. Results are saved by default in a folder called 'coreg' which is moved to the 'bias_removal' folder at the end of this function

Parameters

- **mst_dem** (*str, pybob.Geoimg, pybob.ICESat*) – Path to filename or GeoImg dataset representing external "master" DEM or ICESat dataset.
- **slv_dem** (*str, pybob.GeoImg*) – Path to filename or GeoImg dataset representing ASTER DEM
- **glacmask** (*str*) – Path to shapefile representing areas to exclude from co-registration consideration (i.e., glaciers).

- **landmask** (*str*) – Path to shapefile representing areas to include in co-registration consideration (i.e., stable ground/land).
- **work_dir** – Working directory to use [Assumes '.'].
- **out_dir** (*str*) – Output directory for the new coregistration folder
- **pts** – True if mst_dem is point data, False if mst_dem is a DEM.

Returns mst_coreg, slv_coreg, shift_params co-registered master, slave datasets, as well as a tuple containing x,y,z shifts calculated during co-registration process.

pymmaster.mmaster_tools.**reproject_geometry**(*src_data*, *src_crs*, *dst_crs*)

Reproject a geometry object from one coordinate system to another.

Parameters

- **src_data** (*shapely geometry*) – geometry object to reproject
- **src_crs** (*str, dict*) – proj4 description of source CRS
- **dst_crs** (*str, dict*) – proj4 description of destination CRS

Returns dst_data reprojected data.

pymmaster.mmaster_tools.**robust_polynomial_fit**(*xx*, *yy*)

Given sample data xx, yy, compute a robust polynomial fit to the data. Order is chosen automatically by comparing residuals for multiple fit orders.

Parameters

- **xx** (*array-like*) – input x data (typically across-track distance)
- **yy** (*array-like*) – input y data (typically elevation differences)

Returns coefs, order polynomial coefficients and order for the best-fit polynomial

CHAPTER 4

Indices and tables

- genindex
- modindex
- search

Python Module Index

p

Index