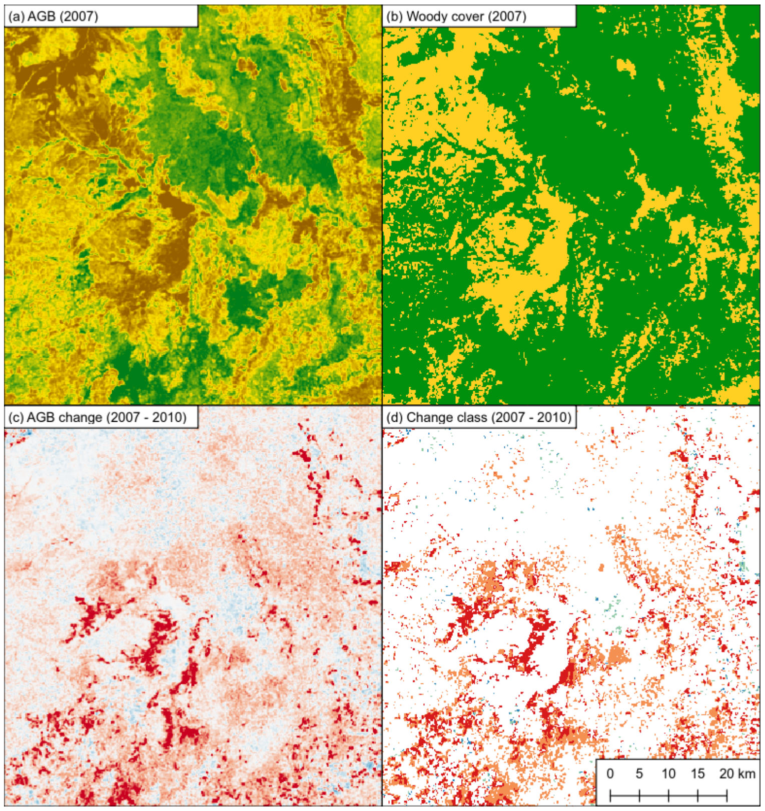
biota Documentation

Release 1

Samuel Bowers

Aug 02, 2022

Contents



The BIOmass Tool for Alos (`biota`) is a Python library that uses data from JAXA's ALOS mosaic product to generate maps of:

- Aboveground biomass
- Forest cover
- Forest cover change

CHAPTER 1

Who do I talk to?

Written and maintained by Samuel Bowers (sam.bowers@ed.ac.uk).

CHAPTER 2

Contents:

2.1 Setup instructions

2.1.1 Preamble

biota is written in Python and was developped on a Linux platform. Provided you can use Python and can install modules on your computer, it should be possible to run on most OS or a Linux server.

On this page, we explain how to set up your Linux or Windows machine to use *biota*.

2.1.2 Installing Anaconda

We recommend running `biota` in Anaconda.

If you are using a Linux machine, open a terminal window, change directory to the location you downloaded Anaconda Python, and run the following commands:

```
wget https://repo.anaconda.com/archive/Anaconda3-2020.11-Linux-x86_64.sh
bash Anaconda3-2020.11-Linux-x86_64.sh
```

Once complete, you'll need to add this version of Python to your .bashrc file as follows (replacing `~/` with your installation directory):

```
# Substitute root for the path to your system's installation and .bashrc file.
echo 'export PATH="~/anaconda3/bin:$PATH"' >> ~/.bashrc
exec -l $SHELL
```

If this has functioned, on executing `python` in a terminal window, you should see the following:

```
Python 2.7.12 |Anaconda custom (64-bit)| (default, Jul  2 2016, 17:42:40)
[GCC 4.4.7 20120313 (Red Hat 4.4.7-1)] on linux2
Type "help", "copyright", "credits" or "license" for more information.
Anaconda is brought to you by Continuum Analytics.
```

(continues on next page)

(continued from previous page)

```
Please check out: http://continuum.io/thanks and https://anaconda.org
>>>
```

For Windows users, go to the Anaconda website) and download the installer for your version of Windows (64-bit or 32-bit). Once the download is finished, run the installer. This may take a while, but when it is done you will be able to open the *Anaconda Prompt*.

2.1.3 Setting up your Anaconda environment

To ensure you are working with the appropriate version of Python as well as the correct modules, we recommend that you create an Anaconda virtual environment set up for running `biota`. This is done by running the following commands in your terminal or the *Anaconda prompt* (recommended procedure):

```
conda create -n biota -c conda-forge python=3.7 tqdm scikit-image pillow pyshp gdal
```

Note: the GDAL package is notoriously temperamental. If this step fails, try again and add ‘ openssl=1.0‘ at the end of the line

Activate the `biota` environment whenever opening a new terminal window by running this command:

```
conda activate biota
```

Note: Remember to activate the `biota` environment whenever you want to use `biota`.

If you are SURE you won’t use anything else than *biota*, you can do without virtual environments. In this case, just type:

```
conda install -c conda-forge python=3.7 tqdm scikit-image pillow pyshp gdal
```

If you want to use the *biota* graphical interface, you need an extra package called *PyQt5*. To install it, type:

```
pip install pyqt5
```

2.1.4 Installing biota

Navigate to the folder where you want to install *biota*. To do this in both Linuw and Windows, type the following:

```
cd /full/path/to/your/favorite/folder/
```

Note: If you don’t want to type the full path (and really, you souldn’t), copy it from your file explorer into the terminal after the ‘cd ‘ command.

To install `biota`, you will need to use the version control software `git` (if you don’t have `git`, follow the instructions here). You can collect the `biota` source code with the command:

```
git clone https://github.com/smfm-project/biota.git
```

To install `biota`, run the following command:

```
python setup.py install
```

Congratulations, you are now ready to use *biota*.

2.2 The various forms of biota

biota comes in 3 forms: a Python module, a command line function, and a Graphical User Interface (GUI). We recommend the use of the Python module for users with good knowledge of Python who wish to produce and adapt their own results. The command line form of *biota* is particularly adapted for batch jobs, and has been used to produce country-scale maps of biomass loss and probably deforestation with little scripting. Finally, the *biota* GUI provides an accessible tool for users who are less comfortable with scripting and terminals.

2.2.1 Using biota in Python

If your installation was successful, you should be able to import `biota` in Python:

```
import biota
```

You may now use biota in your Python scripts! See the worked examples to explore the different methods and attributes of *biota* in Python. Using *biota* in python will give you the most flexibility.

2.2.2 Using biota from the command line

For most applications, the command line interface will be the most straightforward way of using `biota`. It's also the best way to run batch jobs.

To avoid having to reference the full path of the Python scripts in biota when using command line tools, add the following line to your .bashrc file:

```
alias biota='_biota() { python ~/full/path/to/biota/cli/"$1".py $(shift; echo "$@") ;}
↪; _biota'
```

This creates a function that enables you to call `biota` just by typing `biota` in your terminal window. To run this function, restart your terminal or run `bash` (you will only need to do this once). You will then need to activate the `biota` environment once again.

Take a look at the worked examples to explore the possibilities when you using *biota* from the command line.

2.2.3 Using the biota Graphical User Interface (GUI)

The `biota` Graphical User Interface (GUI) allows you to use *biota* with minimal usage of the command line. In your terminal or the *Anaconda prompt* (if you're using Windows), navigate to the folder where you installed *biota* by typing:

```
cd /full/path/to/biota/
```

This folder should have */biota/* at the end of its path. In this folder you will find a folder called *Biota_app/*. Navigate to it by typing:

```
cd Biota_app/
```

To open the GUI, type

```
python Biota.py
```

The GUI window should then open. Now you can use *biota* several times with minimal typing.

2.3 Command line tools

`biota` was initially developed as a tool for use in Python. However, command line functionalities are available to download data and produce outputs of:

- Calibrated gamma0 backscatter
- Forest cover
- Aboveground biomass
- Aboveground biomass change
- Classified forest change types (i.e. deforestation, degradation etc.)

These are available through the command line tool `biota`.

2.3.1 Getting ALOS mosaic data

Data from the ALOS mosaic product can be accessed from JAXA through a graphical interface online after signing up. The data is delivered in either 1x1 degree tiles or 5x5 degree collections of tiles (recommended).

To obtain national-scale data using the web interface is possible, but for large-scale applications `biota` contains a command line interface to automate the download and decompression process. `download.py` takes latitude, longitude and years as inputs, and returns a downloaded an decompressed and ready-to-use product in a format the `biota` will understand.

The `biota` tool includes a script to download ALOS mosaic tiles directly from the JAXA FTP server. The user specifies either a 1x1 or 5x5 degree tile by its upper-left corner latitude and longitude and a year or set of years. The downloader handles access of both the older ALOS-1 filename structure and the ALOS-2 filename structure. The script also optionally decompress images after download.

Help for `download.py` can be viewed by typing `biota download --help`:

```
usage: download.py [-h] [-lat DEG] [-lon DEG] [-l] [-y Y [Y ...]] [-o DIR]
                [-r]

Download ALOS-1/2 data from JAXA, specifying a particular year and
latitude/longitude.

Required arguments:
-lat DEG, --latitude DEG
                        Latitude of tile upper-left corner.
-lon DEG, --longitude DEG
                        Longitude of tile upper-left corner.

Optional arguments:
-l, --large           Download large tiles. ALOS mosaic tiles are available
                        in 1x1 or 5x5 degree tiles. If downloading large
                        volumes of data, it's usually better to use the
                        latter. If this option is chosen, you must select a
```

(continues on next page)

(continued from previous page)

```
                        lat and lon that's a multiple of 5 degrees.
-y Y [Y ...], --years Y [Y ...]
                        Year of data to download. Defaults to downloading all
                        data.
-o DIR, --output_dir DIR
                        Optionally specify an output directory. Defaults to
                        the present working directory.
-r, --remove          Optionally remove downloaded .tar.gz files after
                        decompression.
```

For example, to download data for the 1x1 tile at 38 degrees longitude and -8 degrees latitude for the years 2007 and 2010, input:

```
biota download -lon 38 -lat -8 -y 2007 2010
```

To download all tiles for the 5x5 degree area (recommended) covering 35 - 40 degrees longitude and -5 - -10 degrees latitude for the years 2007 and 2010, input:

```
biota download -lon 35 -lat -5 --large -y 2007 2010
```

To specify an output directory, we can use the `-o` option:

```
biota download -lon 38 -lat -8 -y 2007 2010 -o /path/to/output_dir
```

ALOS mosaic data are delivered as compressed files, which `biota` will decompress. To remove the original files after decompression, use the `-r` flag:

```
biota download -lon 38 -lat -8 -y 2007 2010 -r
```

2.3.2 Mapping vegetation properties

The `biota` tool features a command line option to produce a raster of vegetation properties for a given year. The user specifies the directory where the data are stored, then specifies the designated location and year like for the `download`. The user may choose to produce rasters for any or all of the following properties: Gamma0, Aboveground Biomass, Woody Cover. Filtering, downsampling and options specific to each property are available.

Help for this functionality can be viewed by typing `biota property --help` or `biota property -h`:

```
usage: property.py [-h] [-dir DIR] [-lat DEG] [-lon DEG] [-y Y [Y ...]]
                                   [-o {Gamma0,AGB,WoodyCover,all}] [-nf] [-ds N] [-
↪od DIR]
                                   [-v] [-p POL] [-u units] [-ft tC/ha] [-at ha]

Process ALOS-1/2 mosaic data to prpoduce estimates of forest cover and
biomass.

Required arguments:
  -dir DIR, --data_directory DIR
                                                Path to directory containing ALOS␣
↪mosaic data.
  -lat DEG, --latitude DEG
                                                Latitude of tile to process (upper-
↪left corner).
  -lon DEG, --longitude DEG
```

(continues on next page)

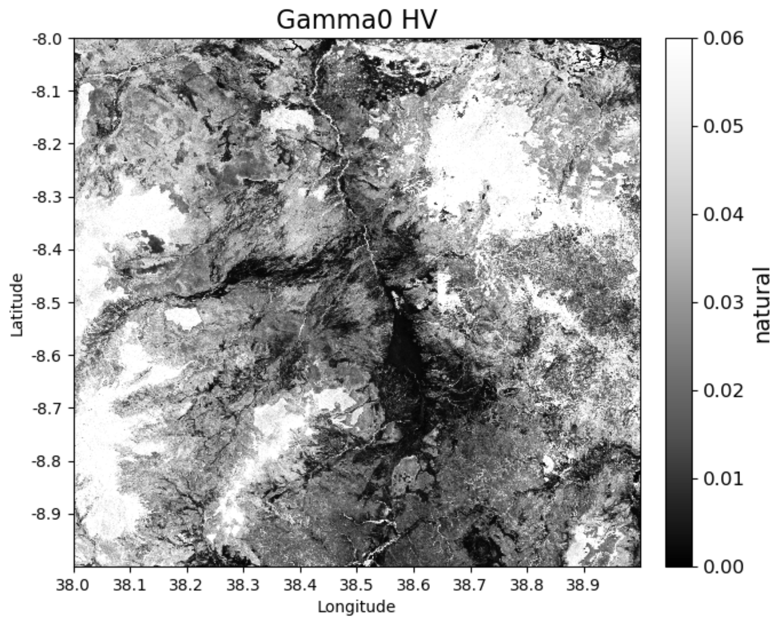
(continued from previous page)

```
                                               Longitude of tile to process (upper-
↪left corner).
  -y Y [Y ...], --years Y [Y ...]
                                               Years of data to process.

Optional arguments:
  -o {Gamma0,AGB,WoodyCover,all}, --output {Gamma0,AGB,WoodyCover,all}
                                               Choose which kind of output you want.␣
↪Defaults to all
                                               possible outputs.
  -nf, --nofilter       Use this flag if you don't want to apply a speckle
                                               filter.
  -ds N, --downsample_factor N
                                               Apply downsampling to inputs by␣
↪specifying an integer
                                               factor to downsample by. Defaults to␣
↪no downsampling.
  -od DIR, --output_dir DIR
                                               Optionally specify an output␣
↪directory. Defaults to
                                               the present working directory.
  -v, --verbose         Print progress to terminal. Defaults to False.

Output-specific arguments:
  -p POL, --polarisation POL
                                               If you have selected Gamma0 as an␣
↪output, choose the
                                               polarisation. Defaults to HV.
  -u units, --units units
                                               If you have selected Gamma0 as an␣
↪output, choose the
                                               outputs units. Defaults to 'natural'␣
↪units.
  -ft tC/ha, --forest_threshold tC/ha
                                               If you have selected WoodyCover as an␣
↪output, choose
                                               the miminum forest biomass threshold.␣
↪Defaults to 10
                                               tC/ha.
  -at ha, --area_threshold ha
                                               If you have selected WoodyCover as an␣
↪output, choose
                                               the minimum forest area threshold.␣
↪Defaults to 0 ha.

        usage: property.py [-h] [-dir DIR] [-lat DEG] [-lon DEG] [-y Y [Y ...]]
                                                  [-o {Gamma0,AGB,WoodyCover,all}] [-
↪lf] [-ds FACTOR]
                                                  [-od DIR] [-pz {HV,HH,VH,VV}] [-ft␣
↪THRESHOLD]
                                                  [-at THRESHOLD]
```

Gamma0 backscatter

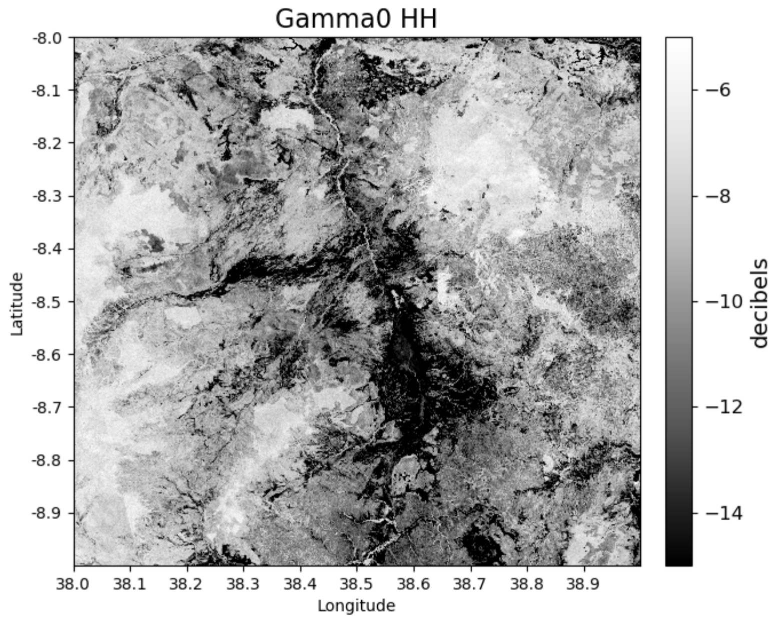
For example, to produce a calibrated image of Gamma0 backscatter for the HV polarisation for the downloaded 1x1 tile at 38 degrees longitude and -8 degrees latitude for the year 2007, run:

```
biota property -dir /path/to/data/ -lon 38 -lat -8 -y 2007 -o Gamma0
```



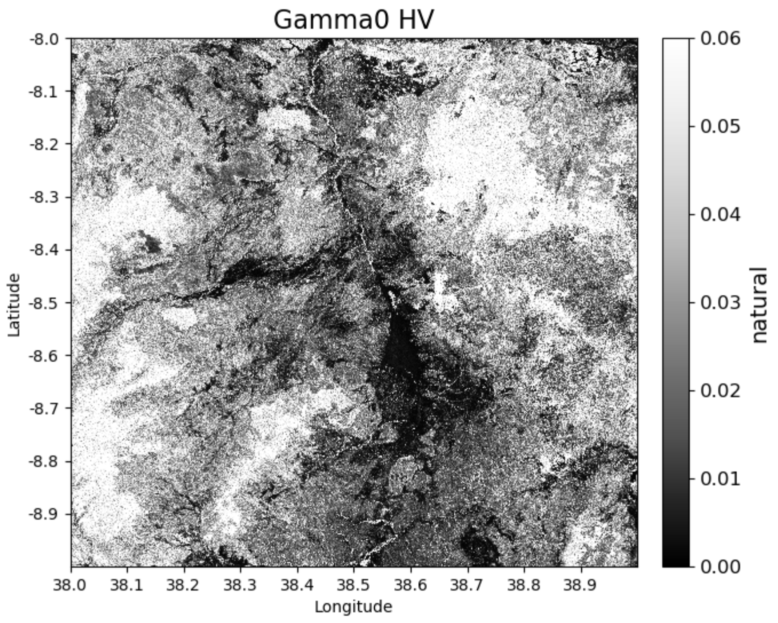
To produce the same image, but using the HH polarisation and with units of decibels:

```
biota property -dir /path/to/data/ -lon 38 -lat -8 -y 2007 -o Gamma0 -u decibels -p HH
```



If you prefer not to apply a speckle filter:

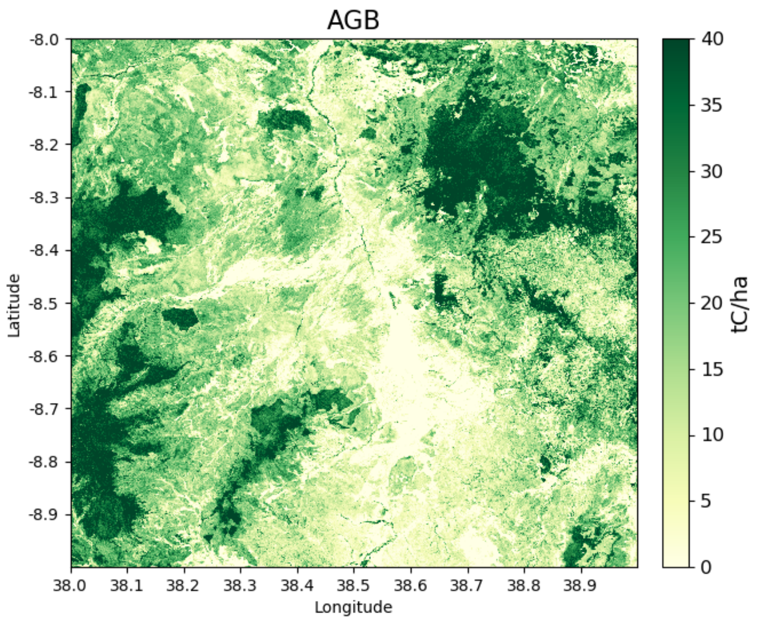
```
biota property -dir /path/to/data/ -lon 38 -lat -8 -y 2007 -o Gamma0 --nofilter
```



Aboveground biomass

To produce a map of aboveground biomass (based on a generic southern African biomass-backscatter model, in units of tC/ha) for the same tile run:

```
biota property -dir /path/to/data/ -lon 38 -lat -8 -y 2007 -o AGB
```

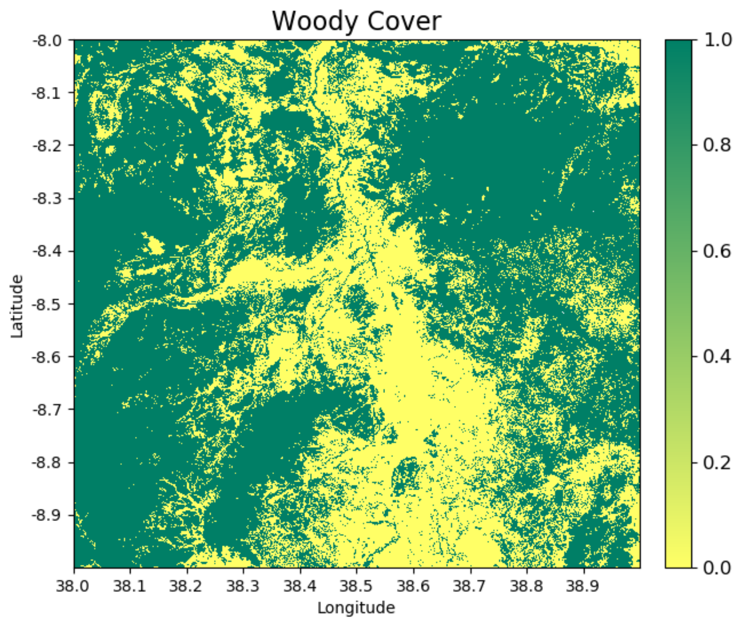


Woody cover

`biota` can also be used to produce a map of forest cover (or woody cover), based on a threshold of biomass. By default the 10 tC/ha threshold separates forest from nonforest.

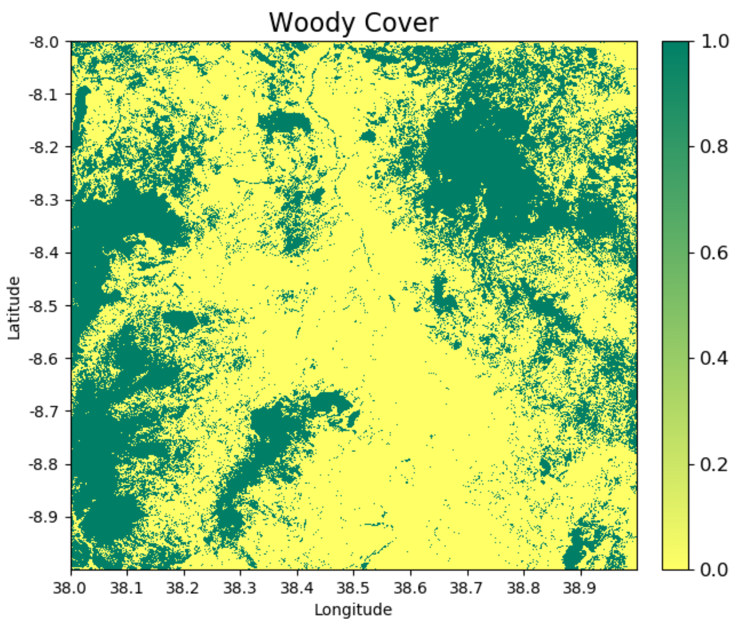
To produce a map of woody cover run:

```
biota property -dir /path/to/data/ -lon 38 -lat -8 -y 2007 -o WoodyCover
```



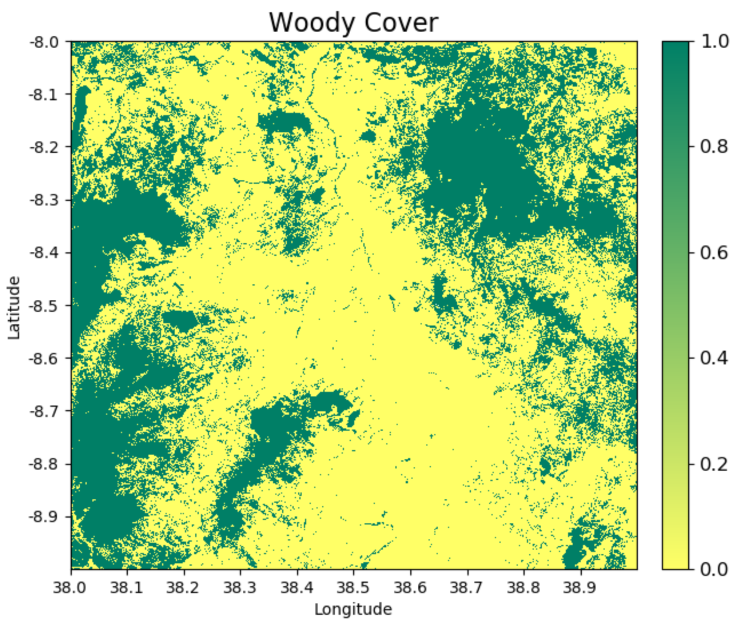
To use a custom forest/non-forest threshold (e.g. 20 tC/ha), use the `-ft` option:

```
biota property -dir /path/to/data/ -lon 38 -lat -8 -y 2007 -o WoodyCover -ft 20
```



Forest definitions often contain a minimum area threshold (e.g. a forest patch must be greater than 1 ha in size) to be counted as forest. To also add a minimum area threshold with `biota` use the `-at` option:

```
biota property -dir /path/to/data/ -lon 38 -lat -8 -y 2007 -o WoodyCover -ft 20 -at 1
```



2.3.3 Producing vegetation change rasters

The `biota` tool features a command line option to produce a raster of vegetation change between two given years. The user specifies the directory where the data are stored, then specifies the designated location and two years to compare.

Help for this functionality can be viewed by typing `biota change --help` or `biota change -h`:

```
usage: change.py [-h] [-dir DIR] [-lat DEG] [-lon DEG] [-y1 YR] [-y2 YR]
                                 [-o {AGBChange,ChangeType,all}] [-nf] [-ds N] [-od␣
↪DIR] [-v]
                                 [-ct ha] [-mt tC/ha] [-it PC] [-ft tC/ha] [-at ha]

Process ALOS-1/2 moasic data to output biomass and woody cover change between
2 years.

Required arguments:
  -dir DIR, --data_directory DIR
                                               Path to directory containing ALOS␣
↪mosaic data.
  -lat DEG, --latitude DEG
                                               Latitude of tile to process (upper-
↪left corner).
  -lon DEG, --longitude DEG
                                               Longitude of tile to process (upper-
↪left corner).
  -y1 YR, --year1 YR    First year of data to process.
  -y2 YR, --year2 YR    Second year of data to process.

Optional arguments:
  -o {AGBChange,ChangeType,all}, --output {AGBChange,ChangeType,all}
                                               Choose which kind of output you want.␣
↪Defaults to all
                                               possible outputs.
  -nf, --nofilter       Use this flag if you don't want to apply a speckle
                                               filter.
  -ds N, --downsample_factor N
                                               Apply downsampling to inputs by␣
↪specifying an integer
                                               factor to downsample by. Defaults to␣
↪no downsampling.
  -od DIR, --output_dir DIR
                                               Optionally specify an output␣
↪directory. Defaults to
                                               the present working directory.
  -v, --verbose         Print progress to terminal. Defaults to False.

Output-specific arguments:
  -ct ha, --change_area_threshold ha
                                               If you have selected ChangeType as an␣
↪output, choose a
                                               threshold for a minimum change in␣
↪forest area required
                                               to be flagged as a change. Defaults␣
↪to 0 ha.
  -mt tC/ha, --change_magnitude_threshold tC/ha
                                               If you have selected ChangeType as an␣
↪output, choose
                                               the minimum absolute change in␣
↪biomass to be flagged
                                               as a change. Defaults to 0 tC/ha.
  -it PC, --change_intensity_threshold PC
```

(continues on next page)

(continued from previous page)

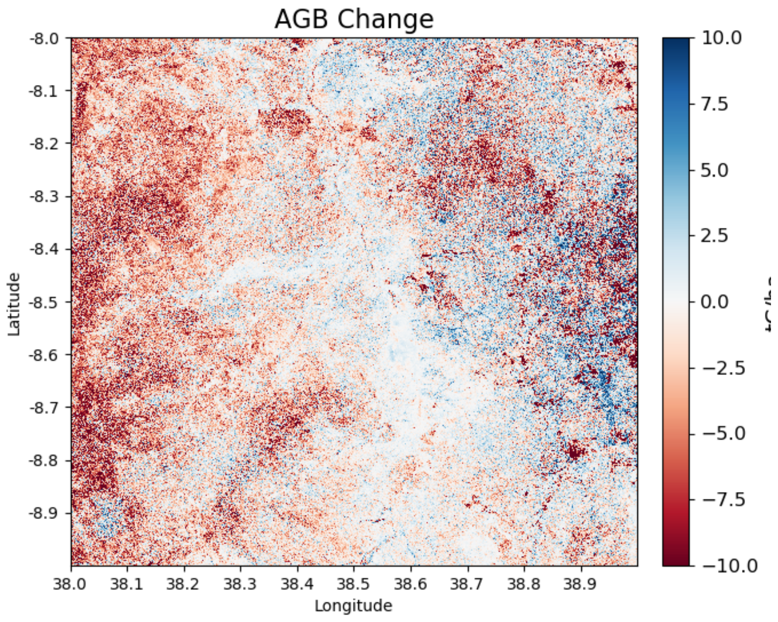
```
                                          If you have selected ChangeType as an␣
↪output, choose
                                          the minimum relative change in␣
↪biomass to be flagged
                                          as a change. Defaults to 0 percent.
  -ft tC/ha, --forest_threshold tC/ha
                                          If you have selected ChangeType as an␣
↪output, choose
                                          the miminum forest biomass threshold␣
↪in each input
                                          image. Defaults to 10 tC/ha.
  -at ha, --area_threshold ha
                                          If you have selected ChangeType as an␣
↪output, choose
                                          the minimum forest area threshold in␣
↪each input image.
                                          Defaults to 0 ha.
```

AGB Change

`biota` can produce estimates of AGB change

To produce a map of biomass change for the downloaded 1x1 tile at 38 degrees longitude and -8 degrees latitude between 2007 and 2010, run:

```
biota change -dir /path/to/data/ -lon 38 -lat -8 -y1 2007 -y2 2010 -o AGBChange
```



Change detection

Changes in AGB are classified based on a series of thresholds:

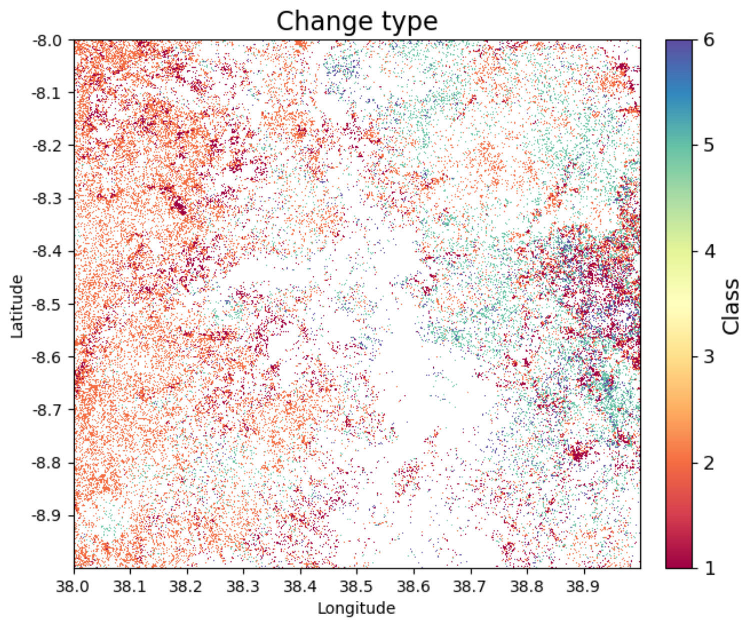
Threshold	Description
`forest_threshold`	The minimum AGB that defines a forest area (tC/ha).
`change_area_threshold`	The minimum area over which a change must occurr (ha).
`change_magnitude_threshold`	The minimum absolute change of AGB that defines a change event (tC/ha).
`change_intensity_threshold`	The minimum proportional change of AGB that defines a change event (0-1).

There are 7 change types described in `biota`, each of which is defined as a number 0 to 6 in the output GeoTiff. Change types are:

Change class	Pixel value	Description
Deforestation	1	A loss of AGB from that crosses the `forest_threshold`.
Degradation	2	A loss of AGB in a location above the `forest_threshold` in both images.
Minor Loss	3	A loss of AGB that does not cross the `change_area_threshold`, `change_magnitude_threshold`, or `change_intensity_threshold`.
Minor Gain	4	A gain of AGB that does not cross the `change_area_threshold`, `change_magnitude_threshold`, or `change_intensity_threshold`.
Growth	5	A gain of AGB in a location above the `forest_threshold` in both images.
Aforestation	6	A gain of AGB that crosses the `forest_threshold`.
Nonforest	0	Below `forest_threshold` in both images.

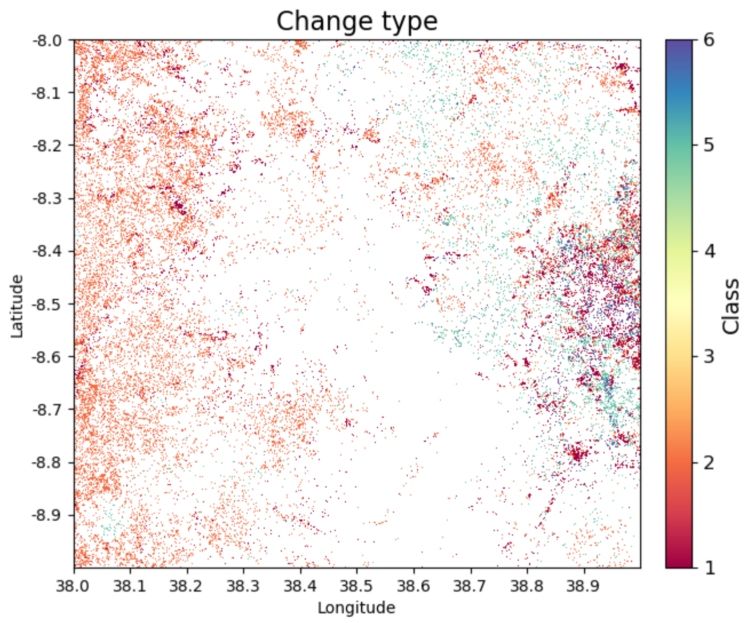
To identify change types with default parameters, use:

```
biota change -dir /path/to/data/ -lon 38 -lat -8 -y1 2007 -y2 2010 -o ChangeType
```



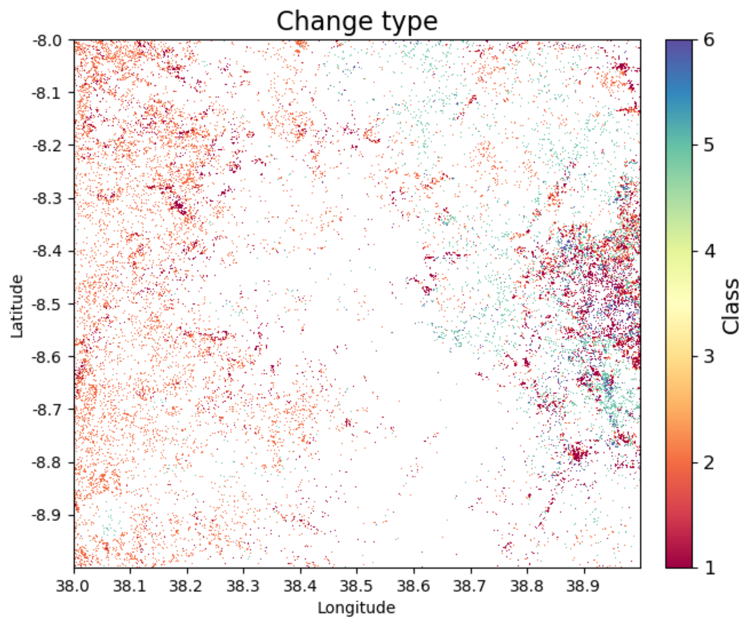
To apply a minimum area of 1 hectare for a given change:

```
biota change -dir /path/to/data/ -lon 38 -lat -8 -y1 2007 -y2 2010 -o ChangeType -ct 1
```



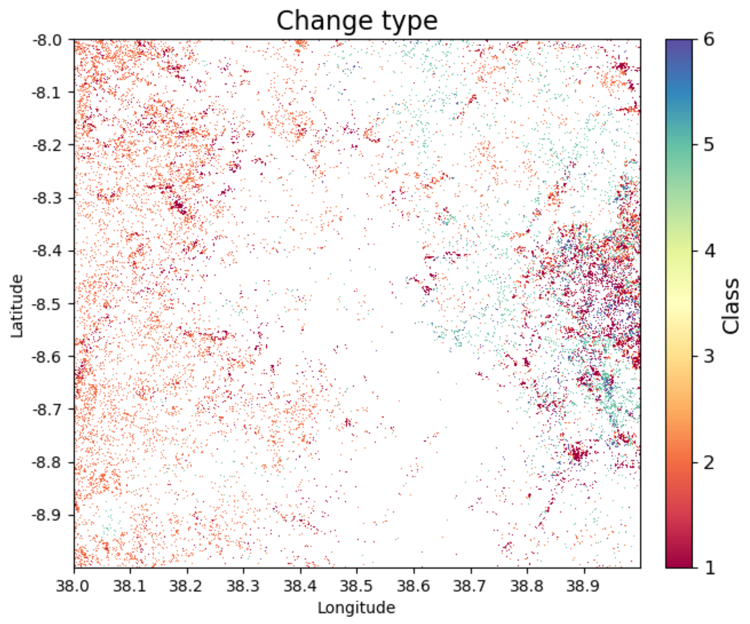
To also apply a minimum absolute change threshod of 5 tC/ha:

```
biota change -dir /path/to/data/ -lon 38 -lat -8 -y1 2007 -y2 2010 -o ChangeType -ct␣
↪1 -mt 5
```



To also apply a minumum relative change threshold of 25 % of biomass change:

```
biota change -dir /path/to/data/ -lon 38 -lat -8 -y1 2007 -y2 2010 -o ChangeType -ct␣
↪1 -mt 5 -it 25
```



Change detection recipes

To produce high-quality estimates of change, the user will need to carefully define the multiple parameters determing change parameters based on experimentation and ecological knowledge. Some recipes that have worked well for us include the following.

For production of deforestation estimates in Zimbabwe:

```
biota change -dir /path/to/data/ -lon 38 -lat -8 -y1 2007 -y2 2010 -o ChangeType -ct␣
↪1 -it 25
```

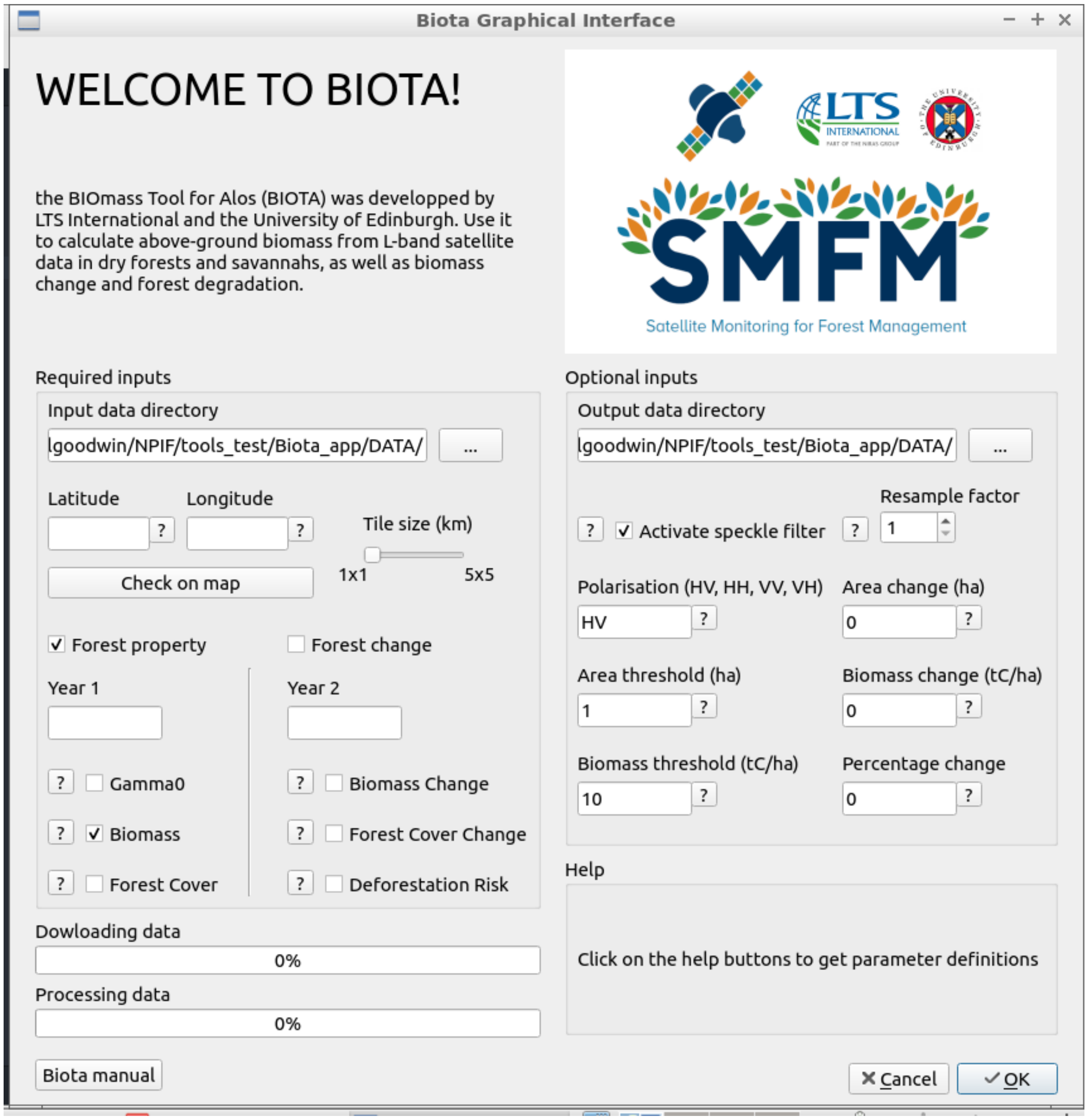
More recipes to follow after the Nairobi SMFM workshop (11th - 15th March 2019).

2.4 Graphical User Interface

In this section, we describe the features of the *biota* GUI. Use this form of *biota* if you are not comfortable with terminals or scripting, or if you simply want to process a single tile easily.

2.4.1 The biota window

The *biota* GUI has three panels: Primary inputs, Secondary inputs, and Help. By clicking on the interrogation buttons, you will display offline help for a given parameter.



Biota Graphical Interface

WELCOME TO BIOTA!

the BIOmass Tool for Alos (BIOTA) was developped by LTS International and the University of Edinburgh. Use it to calculate above-ground biomass from L-band satellite data in dry forests and savannahs, as well as biomass change and forest degradation.

Required inputs

Input data directory: lgoodwin/NPIF/tools_test/Biota_app/DATA/ ...

Latitude ? Longitude ?

Tile size (km) 1x1 5x5

Check on map

✓ Forest property

Forest change

Year 1

Year 2

? Gamma0

? ✓ Biomass

? Forest Cover

? Biomass Change

? Forest Cover Change

? Deforestation Risk

Dowloading data 0%

Processing data 0%

Biota manual

Optional inputs

Output data directory: lgoodwin/NPIF/tools_test/Biota_app/DATA/ ...

? ✓ Activate speckle filter

Resample factor ? 1

Polarisation (HV, HH, VV, VH): HV ?

Area change (ha): 0 ?

Area threshold (ha): 1 ?

Biomass change (tC/ha): 0 ?

Biomass threshold (tC/ha): 10 ?

Percentage change: 0 ?

Help

Click on the help buttons to get parameter definitions

✕ Cancel ✓ OK

2.4.2 Primary inputs

Primary inputs are those that need your attention. You NEED to select an input data directory (even if there is a default), and you NEED to specify the top-left latitude and longitude for the tile you will process. You also NEED to enter at least the Year 1 parameter. If something goes wrong, an error message will appear.

2.4.3 Secondary inputs

Secondary does not mean they are not important! Parameters in this panel are filled with default values, but if you delete them you will get an error message. See the individial help message to decide which value to attribute to each parameter.

2.5 Worked example 1: Downloading data in the command line

In this worked example we'll run through an example based on an area of Southeastern Tanzania called Kilwa District. This is a location where the University of Edinburgh coordinates a network of forest plots where there is good data on aboveground biomass. We'll use `biota` to download ALOS mosaic data for the Kilwa region.

2.5.1 Preparation

First ensure that you've followed the setup instructions successfully.

Open a terminal, and use `cd` to navigate to the location you'd like to store data.

```
cd /home/user
mkdir worked_example_biota
cd worked_example_biota
```

Use `mkdir` to make a directory to contain the ALOS mosaic data:

```
mkdir DATA
```

Here we'll demonstrate the process for downloading the large tile (5x5 degrees) covering Kilwa District.

To begin, navigate to the DATA folder.

```
cd DATA
```

2.5.2 Downloading data

The first step is to download Sentinel-2 level 1C data from JAXA.

For this we use the `download.py` tool. To do this we need to specify the latitude/longitude, and years to download. Kilwa District is contained within the large tile at longitude 35 to 40 degrees and latitude -5 - -10 degrees. For the purposes of this demonstration, we'll download ALOS-1 data from 2007 and 2010.

These options can be encoded as follows:

```
biota download -lon 35 -lat -5 --large -y 2007 2010 -r
```

Note: If you're unsure what the *download* command does, type, *biota download -h* or *biota download –help*.

As we didn't specify the option `-o` (`--output`), data will output to the current working directory. We also included `-r` (`--remove`) flag, meaning that intermediate .zip files downloaded from the internet will be deleted after download.

Wait for the two files (2007 and 2010) to finish downloading before proceeding to the next step. By the time the processing is complete, your `DATA/` directory should contain the following new directories (show files in the current working directory with the command `ls`).

```
S05E035_07_MOS
S05E035_10_MOS
```

2.6 Worked example 2: Your first output

In this section we'll use `biota` to generate maps of gamma0 backscatter, AGB, and forest cover. This example covers the 3 forms of *biota*.

2.6.1 Open Python and import biota

Open a terminal window (right-click the Desktop, and select 'Open Terminal'), and run the command `python` or `ipython`. We recommend use of `ipython` if available, which has a range of features that make it more user-friendly than standard Python. If successful, you should see something that looks like the following:

To load the biota module, type:

```
>> import biota
```

2.6.2 Loading an ALOS tile

Data from the ALOS mosaic is provided as a series of 1 x 1 degree tiles. To load a tile in memory, we need to tell `biota` what directory the ALOS mosaic data are stored in, and what latitude and longitude we want to load. To save us from writing them out repeatedly, we can store these as variables:

```
>>> data_dir = '~/DATA/'
>>> latitude = -9
>>> longitude = 38
```

The biota function to load an ALOS tile can be called with the function `biota.loadTile()`, which takes inputs of (i) the data directory, (ii) the latitude, (iii) the longitude, and (iv) the year (in this order). Here we'll load in data for 2007 using the three variables we previously defined:

```
>>> tile_2007 = biota.LoadTile(data_dir, latitude, longitude, 2007)
```

The new object called `tile_2007` has a range of attributes. These can be accessed as follows:

```
>>> tile_2007.year
2007
>>> tile_2007.lat
-9
```

(continues on next page)

(continued from previous page)

```
>>> tile_2007.lon
38
>>> tile_2007.directory
'~/DATA/S05E035_07_MOS/'
>>> tile_2007.satellite
'ALOS-1'
>>> tile_2007.xSize, tile_2007.ySize # Raster size, in pixels
(4500, 4500)
>>> tile_2007.xRes, tile_2007.yRes # Pixel resolution in meters
(24.401, 24.579)
```

Advanced: The tile also contains projection information for interaction with `GDAL`:

```
>>> tile_2007.extent # Extent in the format minlon, minlat, maxlon, maxlat
(38.0, -10.0, 39.0, -9.0)
>>> tile_2007.geo_t # A geo_transform object
(38.0, 0.00022222222222222223, 0.0, -9.0, 0.0, -0.00022222222222222223)
>>> tile_2007.proj # Projection wkt
'GEOGCS["WGS 84",DATUM["WGS_1984",SPHEROID["WGS 84",6378137,298.257223563,AUTHORITY[
↪"EPSG","7030"]],AUTHORITY["EPSG","6326"]],PRIMEM["Greenwich",0,AUTHORITY["EPSG",
↪"8901"]],UNIT["degree",0.0174532925199433,AUTHORITY["EPSG","9122"]],AUTHORITY["EPSG
↪","4326"]]'
```

There are a few other options that can be specified when loading an ALOS tile, but we'll return to these in the see the furtheroptions section.

2.6.3 Extracting backscatter information

The `biota` module is programmed to calibrate ALOS mosaic data to interpretable units of backscatter. This is performed with the `getGamma0()` function. The data are returned as a masked `numpy` array:

```
>>> gamma0_2007 = tile_2007.getGamma0()
>>> gamma0_2007
masked_array(data =
[[0.0669537278370757 0.04214984634805357 0.05141784577914017 ...,
0.029133617952838833 0.024789602664736045 0.040281545637899534]
[0.031600461516752214 0.04214984634805357 0.05141784577914017 ...,
0.03435099209051573 0.028222499657083098 0.03354230142969638]
[0.031600461516752214 0.04050920492690238 0.06216969020533775 ...,
0.037654602824076254 0.04403078198836734 0.025848435873858728]
...,
[0.0900164548052426 0.0662958895217059 0.07768386584418481 ...,
0.049509525268380976 0.0346139149132766 0.021227103665645366]
[0.08548700525257016 0.0888309264753313 0.11198792676214335 ...,
0.08441404357533155 0.06655132961906884 0.05196509713141002]
[0.07134665398730806 0.05708835833035639 0.07595717558689226 ...,
0.021496125937039534 0.027866832136739485 0.0629132766445086]],
            mask =
[[False False False ..., False False False]
[False False False ..., False False False]
[False False False ..., False False False]
...,
[False False False ..., False False False]
[False False False ..., False False False]
```

(continues on next page)

(continued from previous page)

```
[False False False ..., False False False]],
    fill_value = 1e+20)
```

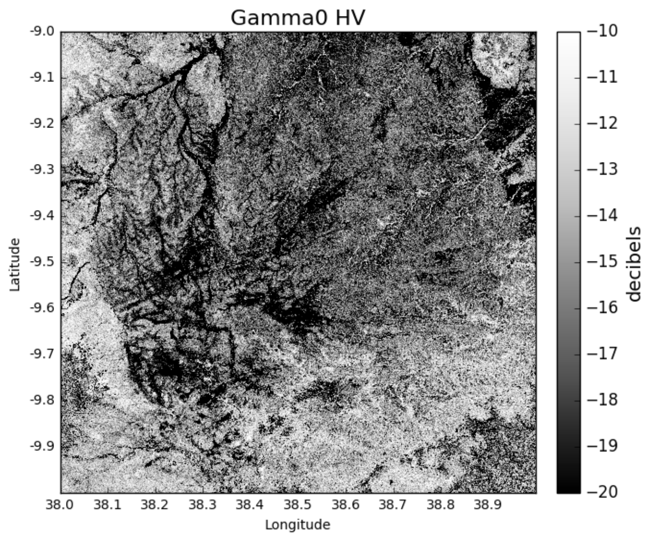
By default the image loaded is 'HV' polarised in 'natural' units. It's also possible to specify options for the polarisation ('HV' *[default]* or 'HH') and the units ('natural' *[default]* or 'decibels') as follows:

```
>>> gamma0_HH_2007 = tile_2007.getGamma0(polarisation = 'HH', units = 'decibels')
>>> gamma0_HV_2007 = tile_2007.getGamma0(polarisation = 'HV', units = 'decibels')
```

If we want to visualise this data, we can run:

```
>>> gamma0_2007 = tile_2007.getGamma0(polarisation = 'HV', units = 'decibels', show =␣
↪True)
```

Which produces the following image:



If we want to save this data to a geoTiff, we can run:

```
>>> gamma0_2007 = tile_2007.getGamma0(polarisation = 'HV', units = 'decibels', output␣
↪= True)
```

This will write a GeoTiff file to the current working directory. This file can be processed and visualised in standard GIS and remote sensing software (e.g. QGIS, GDAL).

To load these tiles and save a raster of backscatter through the command line, run:

```
biota snapshot -dir /path/to/data/ -lat -9 -lon 38, -y 2007 -o Gamma0 -lf
```

To change the default polarisation setting, add the flag `-pz` and enter the desired polarisation. For instance, to get 'HH' data:

```
biota snapshot -dir /path/to/data/ -lat -9 -lon 38, -y 2007 -o Gamma0 -lf -pz HH
```

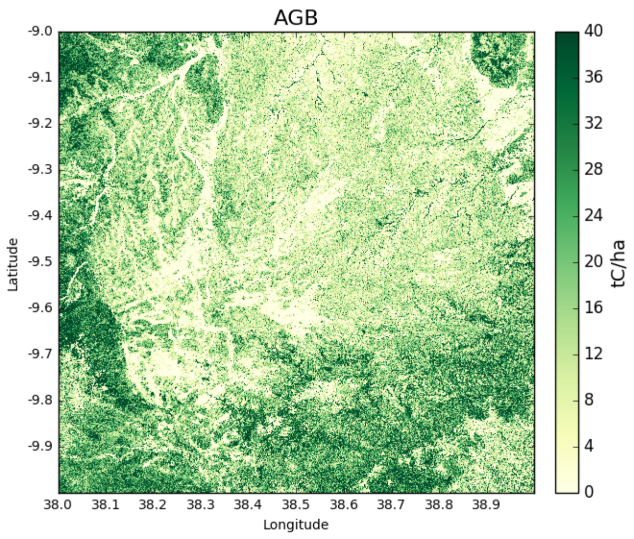
NB; biota does not support data visualisation in the command line, as many users will not have a graphic interface from their terminal. To visualise DATA, load the output raster in GIS software or plot it with Python.

2.6.4 Building a map of AGB

In a similar way to loading gamma0 backscatter, we can show maps of AGB.

```
>>> agb_2007 = tile_2007.getAGB(show = True)
```

Areas in darker green show denser forest:



Like the previous function (and most others in the `biota` module), we can output a GeoTiff as follows:

```
>>> agb_2007 = tile_2007.getAGB(output = True) # To output AGB map to a GeoTiff
```

Or, from the command line, run:

```
biota snapshot -dir /path/to/data/ -lat -9 -lon 38, -y 2007 -o AGB -lf
```

Note: By default `biota` uses an equation calibrated for ALOS-1 in miombo woodlands of Southern Africa. It's advisable to have a locally calibrated biomass-backscatter equation to improve predictions.

2.6.5 Building a forest cover map

A forest cover map (or 'woody cover') can be generated as follows:

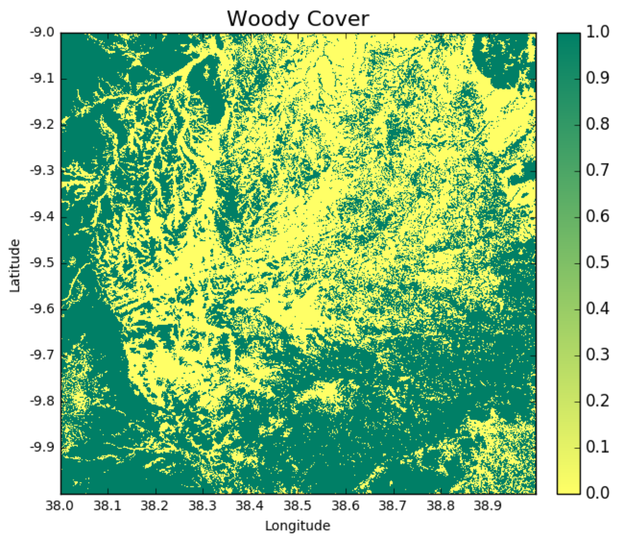
```
>>> woodycover_2007 = tile_2007.getWoodyCover(show = True)
```

and output:

```
>>> woodycover_2007 = tile_2007.getWoodyCover(output = True)
```

To execute this from the command line, run:

```
biota snapshot -dir /path/to/data/ -lat -9 -lon 38, -y 2007 -o WoodyCover -lf
```



By default `biota` will use a generic definition of forest of 10 tC/ha with no minimum area. In the next section we'll discuss how this and other forest definitons can be customised.

2.6.6 Further options when loading an ALOS tile

`biota` supports a range of options for data processing and forest definitions. These options should be specified when loading a tile. These various options can be specified in any combination, but be aware that when analysing change the pre-processing steps for each tile should be identical.

Speckle filtering

Radar data are often very noisy as the result of 'radar speckle', which can be supressed with a speckle filter. The `biota` module has an Enhanced Lee speckle filter, which can be applied to the ALOS tile. By default, no filtering is applied. The speckle filter should be specified on loading the tile:

```
>>> tile_2007 = biota.LoadTile(data_dir, latitude, longitude, 2007, lee_filter = True)
```

Filtering results in an AGB map is noticeably less noisy than those from unfiltered ALOS image.

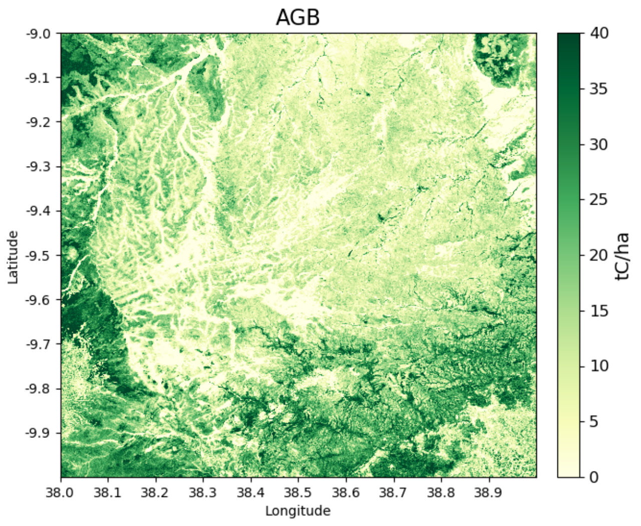
```
>>> tile_2007.getAGB(show = True)
```

In the command line, the flag `-lf` deactivates the speckle-filtering (ON by default). To keep the filter on, simply do not type the flag:

```
biota snapshot -dir /path/to/data/ -lat -9 -lon 38, -y 2007 -o AGB
```

Downsampling

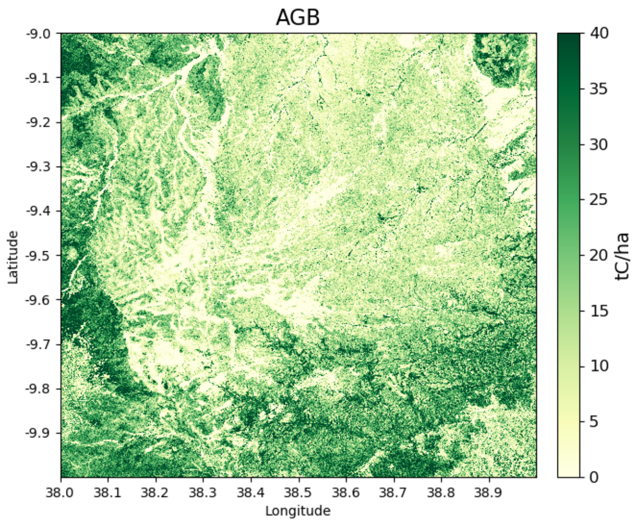
Data volumes can be reduced through downsampling. This comes at a cost to resolution, but does have the positive effect of reducing speckle noise. By default, no downsampling is appied. For example, to halve the resolution of output images, set the parameter `downsample_factor` to 2:



```
>>> tile_2007 = biota.LoadTile(data_dir, latitude, longitude, 2007, downsample_factor␣
↪= 2)
```

With a `downsample_factor` of 2, the resolution of the image is halved:

```
>>> tile_2007.getAGB(show = True)
```



In the command line, the flag `-ds` activates downsampling and is followed by the downsampling factor. To reproduce the result above, run:

```
biota snapshot -dir /path/to/data/ -lat -9 -lon 38, -y 2007 -o AGB -lf
```

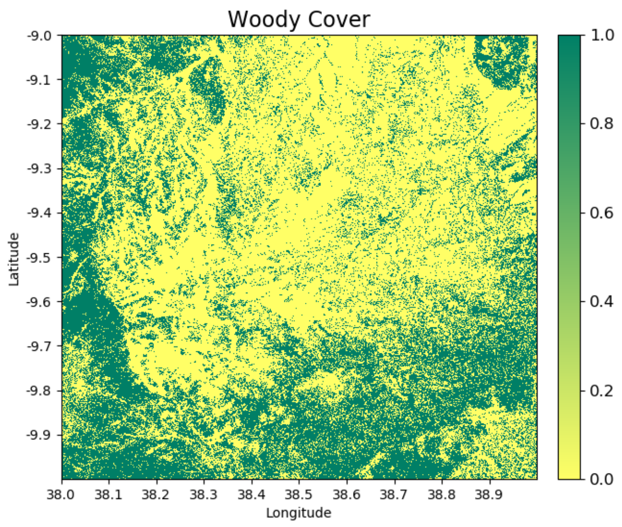
Changing forest definitions

For many purposes it's useful to classify regions into forest and nonforest areas. To achieve this with `biota` a threshold AGB (`forest_threshold`) and a minimum area (`area_threshold`) that separate forest from nonforest can be specified. By default the forest_threshold is 10 tC/ha and the area_threshold is 0 ha. For example, for a forest definition of 15 tC/ha with a minimum area of 1 hecatare:

```
>>> tile_2007 = biota.LoadTile(data_dir, latitude, longitude, 2007, forest_threshold␣
↪= 15, area_threshold = 1)
```

A higher `forest_threshold` or `minimum_area` results in a reduced forest area:

```
>>> tile_2007.getWoodyCover(show = True)
```



In the command line, the flag – activates downsampling and is followed by the downsampling factor. To reproduce the result above, run:

```
biota snapshot -dir /path/to/data/ -lat -9 -lon 38, -y 2007 -o WoodyCover -lf -ft 15 -
↪at 1
```

Changing output directory

By default, GeoTiffs are output to the current working directory. This may not be the best place to output GeoTiff files, a different output directory can be specified as follows:

```
>>> tile_2007 = biota.LoadTile(data_dir, latitude, longitude, 2007, output_dir = '~/
↪output_data/)
```

From the command line: .. code-block:: console

biota snapshot -dir /path/to/data/ -lat -9 -lon 38, -y 2007 -o WoodyCover -lf -od /path/to/output/

2.6.7 Masking data

The ALOS mosaic product is supplied with a basic mask indicating locations of radar show and large water bodies. For many applications this may not be sufficient. For example, radar backscatter from ALOS is strongly influenced by soil moisture changes, which will be particularly severe around rivers.

For some biomass mapping applications and for change detection, we might want to mask out rivers or other features. The `biota` library can generate an updated mask with either classified GeoTiffs or shapefiles.

NB: biota does not support masking from the command line, since it does not output direct visualisations.

Masking with a shapefile

For this example, we'll use a publically available shapefile of inland water in Tanzania from Diva GIS. Download the shapefile here, unzip it, and save it somewhere accessible.

This can be done on the command line as follows:

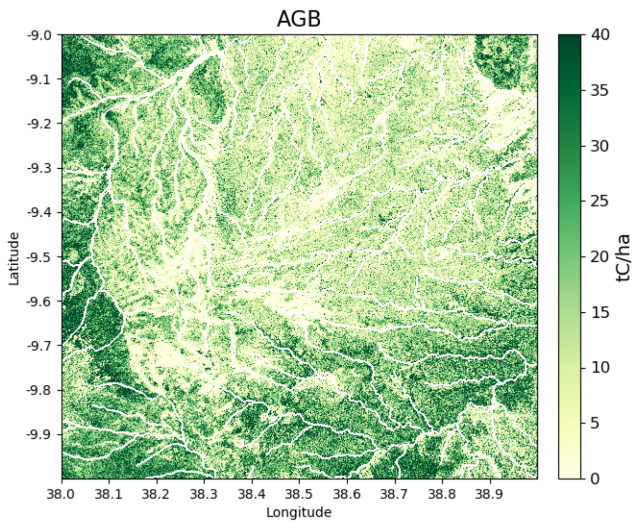
```
mkdir auxillary_data
cd auxillary_data
wget http://biogeo.ucdavis.edu/data/diva/wat/TZA_wat.zip
unzip TZA_wat.zip
```

We can use this shapefile to update the mask in `biota`, applying a 250 m mask around river lines, as follows:

```
>>> tile_2007.updateMask('auxillary_data/TZA_water_lines_dcw.shp', buffer_size = 250)
```

River lines and 250 m buffer now appear in white in the resultant image:

```
>>> tile_2007.getAGB(show = True)
```



Masking with a GeoTiff

Perhaps we aren't interested in mapping known agricultural land, we might want to mask out areas of agriculture from a land cover map.

Here we'll use the ESA CCI land cover map to locate areas of agriculture. The 2007 map is available to download from ESA.

With the command line:

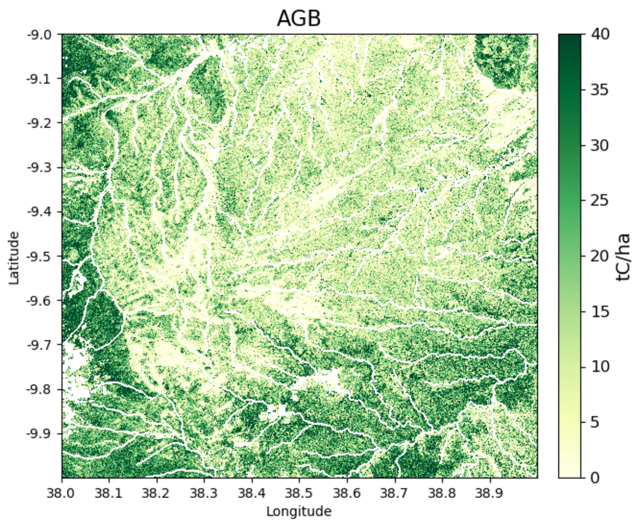
```
cd auxillary_data
wget ftp://geo10.elie.ucl.ac.be/v207/ESACCI-LC-L4-LCCS-Map-300m-P1Y-2007-v2.0.7.tif
```

In the ESA CCI data product the values `10`, `20`, `30`, and `40` correspond to locations with agriculture. We can mask out this class in `biota` as follows:

```
>>> tile_2007.updateMask('auxillary_data/ESACCI-LC-L4-LCCS-Map-300m-P1Y-2007-v2.0.7.
↪tif', classes = [10, 20, 30])
```

Areas to the south-west of the image now appear in the white mask.

```
>>> tile_2007.getAGB(show = True)
```



Note, that the `updateMask()` function added to the previous water mask rather than replacing it. `updateMask()` can be run multiple times to make use of multiple datasets.

Resetting a mask

To return the mask to it's original state, run:

```
tile_2007.resetMask()
```

2.6.8 Putting it all together

Using the commands above, we can create a script to automate the pre-processing of an ALOS tile to generate outputs of gamma0 (HV backscatter in units of decibels), AGB and forest cover for the year 2007. We'll filter the data and specify a forest threshold of 15 tC/ha with a minimum area of 1 hectare, Using a text editor:

```
# Import the biota module
import biota

# Define a variable with the location of ALOS tiles
data_dir = '~/DATA/'

# Define and output location
output_dir = '~/outputs/'

# Define latitude and longitude
latitude = -9
longitude = 38

# Load the ALOS tile with specified options
tile_2007 = biota.LoadTile(data_dir, latitude, longitude, 2007, lee_filter = True,␣
↪forest_threshold = 15., area_threshold = 1, output_dir = output_dir)

# Add river lines to the mask with a 250 m buffer
tile_2007.updateMask('auxillary_data/TZA_water_lines_dcw.shp', buffer_size = 250)

# Calculate gamma0 and output to GeoTiff
gamma0_2007 = tile_2007.getGamma0(output = True)

# Calculate AGB and output to GeoTiff
gamma0_2007 = tile_2007.getAGB(output = True)

# Calculate Woody cover and output to GeoTiff
gamma0_2007 = tile_2007.getWoodyCover(output = True)
```

Save this file (e.g. `process_2007.py`), and run on the command line:

Advanced: To process multiple tiles, we can use nested `for` loops. We can also add a `try`/`except` condition to prevent the program from crashing if an ALOS tile can't be loaded (e.g. over the ocean).

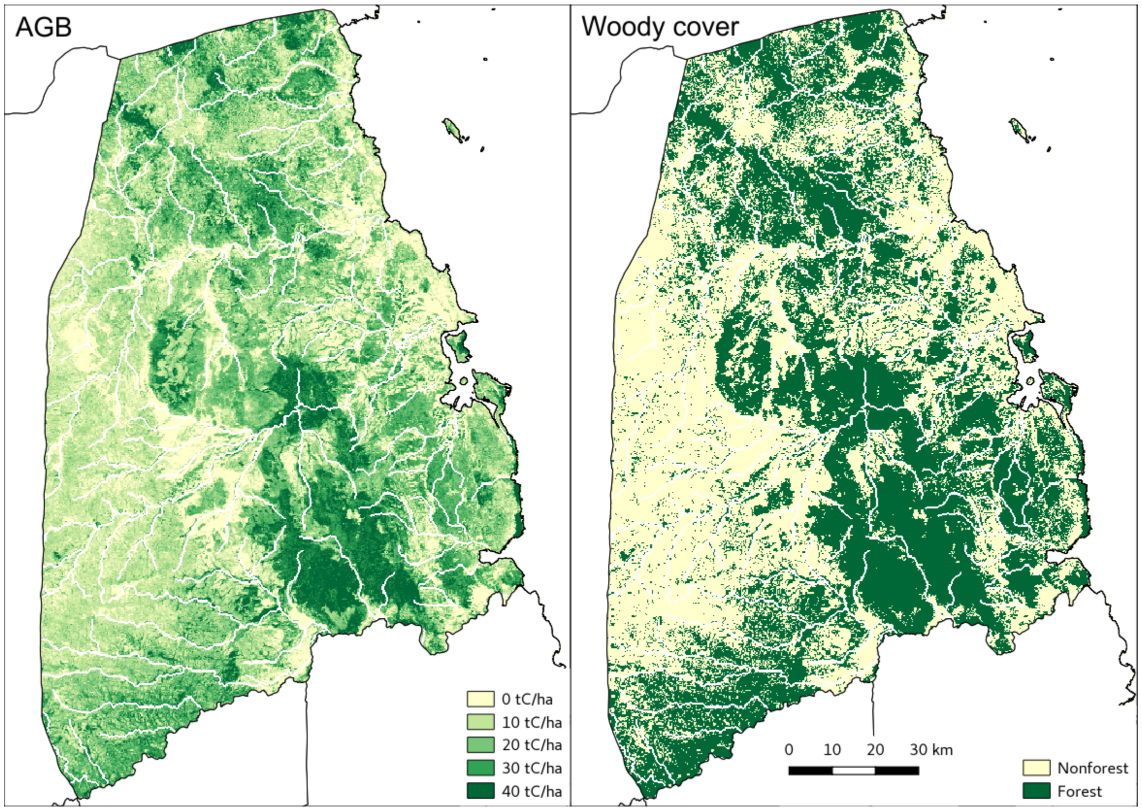
```
# Import the biota module
import biota

# Define a variable with the location of ALOS tiles
data_dir = '~/DATA/'

# Define and output location
output_dir = '~/outputs/'

for latitude in range(-9,-7):
    for longitude in range(38, 40):

        # Print progress
        print 'Doing latitude: %s, longitude: %s'%(str(latitude), str(longitude))

        # Load the ALOS tile with specified options
        try:
            tile_2007 = biota.LoadTile(data_dir, latitude, longitude, 2007, lee_
↪filter = True, forest_threshold = 15., area_threshold = 1, output_dir = output_dir)

        except:
            continue

        # Add river lines to the mask with a 250 m buffer
```

(continues on next page)

(continued from previous page)

```
        tile_2007.updateMask('auxillary_data/TZA_water_lines_dcw.shp', buffer_size =␣
↪250)

        # Calculate gamma0 and output to GeoTiff
        gamma0_2007 = tile_2007.getGamma0(output = True)

        # Calculate AGB and output to GeoTiff
        gamma0_2007 = tile_2007.getAGB(output = True)

        # Calculate Woody cover and output to GeoTiff
        gamma0_2007 = tile_2007.getWoodyCover(output = True)
```

Visualised in QGIS, the resulting biomass and woody cover maps for Kilwa District are:



2.6.9 Producing an output with the GUI

once the window is open, select a Latitude and Longitude, then 'Forest property'. Select the year for which you want to donwload and process the data (yes, the download is automatic in the GUI) and tick the boxes you want to output. If you want to refine your analysis, modify the Area threshold and the Biomass threshold. That's it!

2.7 Worked example 3: Generating change maps

In this section we'll use `biota` to generate change maps based on comparisons of AGB estimates.

2.7.1 Loading a pair of ALOS tiles

The first step to making a change map with `biota` is to load ALOS tiles from two time steps. Here we'll use the same options specified in the previous section. Note: The two tiles must be for the same location, and when specifying processing options (e.g. filtering, forest thresholds), these will need to be identical for the two tiles.

```
# Import the biota module
import biota

# Define a variable with the location of ALOS tiles
data_dir = '~/DATA/'

# Define and output location
output_dir = '~/outputs/'

# Define latitude and longitude
latitude = -9
longitude = 38

# Load the ALOS tile for 2007...
tile_2007 = biota.LoadTile(data_dir, latitude, longitude, 2007, lee_filter = True,␣
↪forest_threshold = 15., area_threshold = 1, output_dir = output_dir)

# ...and for 2010
tile_2010 = biota.LoadTile(data_dir, latitude, longitude, 2010, lee_filter = True,␣
↪forest_threshold = 15., area_threshold = 1, output_dir = output_dir)
```

The two ALOS tiles can be combined into a single object with the following function:

```
>>> tile_change = biota.LoadChange(tile_2007, tile_2010)
```

2.7.2 Calculating AGB Change

The change detection approach of `biota` uses estimates of AGB to identify where forests are changing. AGB change can be calculated though biota as:

```
>>> agb_change = tile_change.getAGBChange()
>>> agb_change
    masked_array(data =
[[-3.403055420785666 -3.2185805294699428 -3.842888490812964 ...,
-5.498752002499962 -6.6389547847411405 -7.520097496689651]
[-1.5990121702260573 -1.6973552573226378 -2.9678666519388592 ...,
-4.4864224739504195 -5.864341812148449 -6.796168587735821]
[-1.02131529023627 -1.5413389977882552 -2.414254618973583 ...,
-2.830552006729352 -3.8702527731167393 -4.528010392506495]
...,
[0.6013737223757403 2.802579609051506 0.34130080537234875 ...,
-6.128205502443066 -4.106561725890309 -1.5234573142258547]
[-1.9193522392770461 2.0279646123649755 0.16270392947076573 ...,
-8.413927340963976 -5.367616290175992 -2.8658223482473666]
[-3.794662904080411 -1.1240199233975332 -2.050815071218274 ...,
-7.369375820470982 -4.486499426633067 -2.4614887987073963]],
            mask =
[[False False False ..., False False False]
[False False False ..., False False False]
[False False False ..., False False False]
```

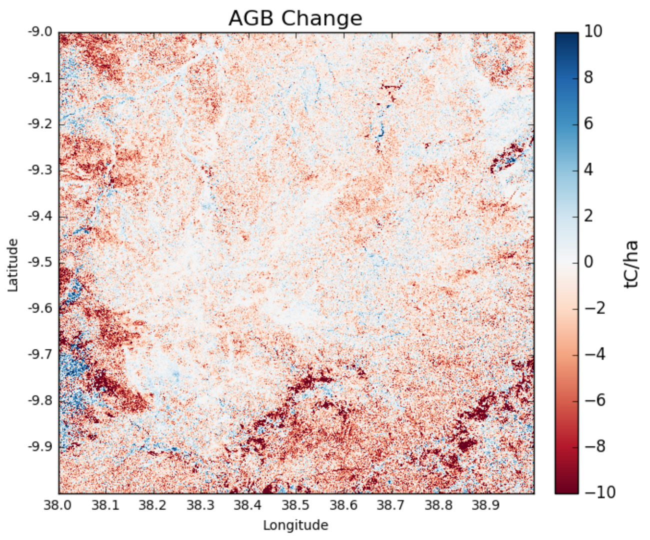
(continues on next page)

(continued from previous page)

```
...,
[False False False ..., False False False]
[False False False ..., False False False]
[False False False ..., False False False]],
    fill_value = 1e+20)
```

Like with a single tile, `biota` can display and output AGB change to GeoTiff:

```
>>> agb_change = tile_change.getAGBChange(show = True, output = True)
```



The predominance of red colours in this tile indicate a general reduction to AGB, with areas of dark red showing locations of deforestation.

To reproduce this result in the command line, run:

```
biota change -dir /path/to/data/ -lat -9 -lon 38, -y1 2007 -y2 2010 -o AGB -lf
```

2.7.3 Classifying change type

Changes in AGB are classified based on a series of thresholds:

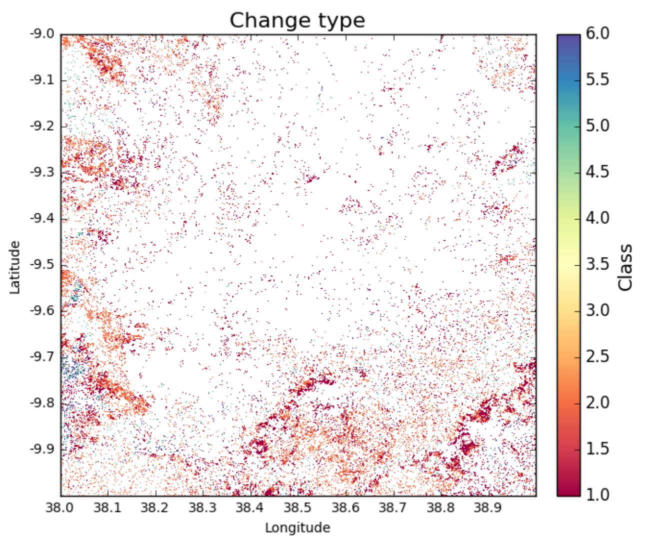
Threshold	Description
`forest_threshold`	The minimum AGB that defines a forest area (tC/ha).
`change_area_threshold`	The minimum area over which a change must occurr (ha).
`change_magnitude_threshold`	The minimum absolute change of AGB that defines a change event (tC/ha).
`change_intensity_threshold`	The minimum proportional change of AGB that defines a change event (0-1).

There are 7 change types described in `biota`, each of which is defined as a number 0 to 6. Change types are:

Change class	Pixel value	Description
Deforestation	1	A loss of AGB from that crosses the `forest_threshold`.
Degradation	2	A loss of AGB in a location above the `forest_threshold` in both images.
Minor Loss	3	A loss of AGB that does not cross the `change_area_threshold`, `change_magnitude_threshold`, or `change_intensity_threshold`.
Minor Gain	4	A gain of AGB that does not cross the `change_area_threshold`, `change_magnitude_threshold`, or `change_intensity_threshold`.
Growth	5	A gain of AGB in a location above the `forest_threshold` in both images.
Aforestation	6	A gain of AGB that crosses the `forest_threshold`.
Nonforest	0	Below `forest_threshold` in both images.

To classify each pixel by its change type, use the function getChangeType(): .. code-block:: python

```
>>> change_types = tile_change.getChangeType(show = True)
```



To reproduce this result in the command line, run:

```
biota change -dir /path/to/data/ -lat -9 -lon 38, -y1 2007 -y2 2010 -o ChangeType -lf
```

2.7.4 Further options for calculating change

Like for a single ALOS tile, `biota` offers a range of parameters for detection of change which should be specified when loading the change onbject. It's worth thinking carefully about these parameters, how they fit with a forest definition, and how they might affect the mapped outputs. The options are:

Change area threshold

Setting this to 1 requires a change to occur over at least 1 hectare for the change to be counted. The default of `change_area_threshold` is 0 ha.

```
>>> tile_change = biota.LoadChange(tile_2007, tile_2010, change_area_threshold = 1)
```

To reproduce this result in the command line, run:

```
biota change -dir /path/to/data/ -lat -9 -lon 38, -y1 2007 -y2 2010 -o ChangeType -lf␣
↪-cat 1
```

Change magnitude threshold

Setting this to 5 requires a change of magntiude at least 5 tC/ha to occur before being counted. The default of `change_magnitude_threshold` is 0 tC/ha.

```
>>> tile_change = biota.LoadChange(tile_2007, tile_2010, change_magnitude_threshold =␣
↪5)
```

To reproduce this result in the command line, run:

```
biota change -dir /path/to/data/ -lat -9 -lon 38, -y1 2007 -y2 2010 -o ChangeType -lf␣
↪-cmt 5
```

Change intensity threshold

Setting this to 0.2 requires a change of 20 % or greater of AGB to be classified as deforestation, degradation, growth, or afforestation. The default of `change_intensity_threshold` is 0.

```
>>> tile_change = biota.LoadChange(tile_2007, tile_2010, change_intensity_threshold =␣
↪0.2)
```

To reproduce this result in the command line, run:

```
biota change -dir /path/to/data/ -lat -9 -lon 38, -y1 2007 -y2 2010 -o ChangeType -lf␣
↪-cit 0.2
```

Change forest threshold

Changes will also be sensitive to the forest definitions used when loading individual tiles. The `forest_threshold` should be itentical in both of the loaded tiles, with stricter forest definitions typically reducing change areas:

Changing output directory

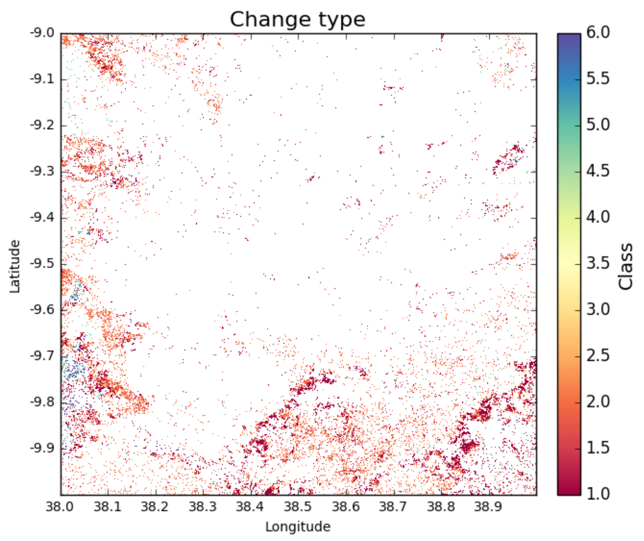
The output directory for GeoTiffs can be specified in a similar way to a single ALOS tile:

```
>>> tile_change = biota.LoadChange(tile_2007, tile_2010, output_dir = '~/outputs/)
```

2.7.5 Classifying change type with options

If we repeat the classification of change type, but this time with a `change_area_threshold` (1 hectare) and a `change_magnitude_threshold` (5 tC/ha), the mapped change shows significantly reduced noise:

```
>>> tile_change = biota.LoadChange(tile_2007, tile_2010, change_area_threshold = 1,␣
↪change_magnitude_threshold = 5)
>>> tile_change.getChangeType(show = True)
```



To reproduce this result in the command line, run:

```
biota change -dir /path/to/data/ -lat -9 -lon 38, -y1 2007 -y2 2010 -o ChangeType -lf␣
↪-cat 1 -cmt 5
```

2.7.6 Masking data

Masks to the change layer are drawn from the two input tiles. Where either one of the input tiles are masked, the change output will also be masked. This means that masks to water bodies etc. should be applied at the `LoadTile()` step.

```
tile_2007.updateMask('auxillary_data/TZA_water_lines_dcw.shp', buffer_size = 250)
tile_change = biota.LoadChange(tile_2007, tile_2010)
tile_change.getAGBChange(show = True)
```

2.7.7 Other functionality

[To follow]

2.7.8 Putting it all together

```
# Import the biota module
import biota

# Define a variable with the location of ALOS tiles
data_dir = '~/DATA/'

# Define and output location
output_dir = '~/outputs/'

# Define latitude and longitude
latitude = -9
longitude = 38

# Load the ALOS tiles with specified options
tile_2007 = biota.LoadTile(data_dir, latitude, longitude, 2007, lee_filter = True,␣
↪forest_threshold = 15., area_threshold = 1, output_dir = output_dir)

tile_2010 = biota.LoadTile(data_dir, latitude, longitude, 2010, lee_filter = True,␣
↪forest_threshold = 15., area_threshold = 1, output_dir = output_dir)

# Add river lines to the mask with a 250 m buffer
tile_2007.updateMask('auxillary_data/TZA_water_lines_dcw.shp', buffer_size = 250)

# Load change between tiles, with options
tile_change = biota.LoadChange(tile_2007, tile_2010, change_area_threshold = 1,␣
↪change_magnitude_threshold = 5)

# Calculate AGB change and output
agb_change = tile_change.getAGBChange(output = True)

# Calculate change type and output
change_type = tile_change.getChangeType(output = True)
```

Save this file (e.g. `process_change.py`), and run on the command line:

Advanced: To process multiple tiles, we can use nested `for` loops. We can also add a `try`/`except` condition to prevent the program from crashing if an ALOS tile can't be loaded (e.g. over the ocean).

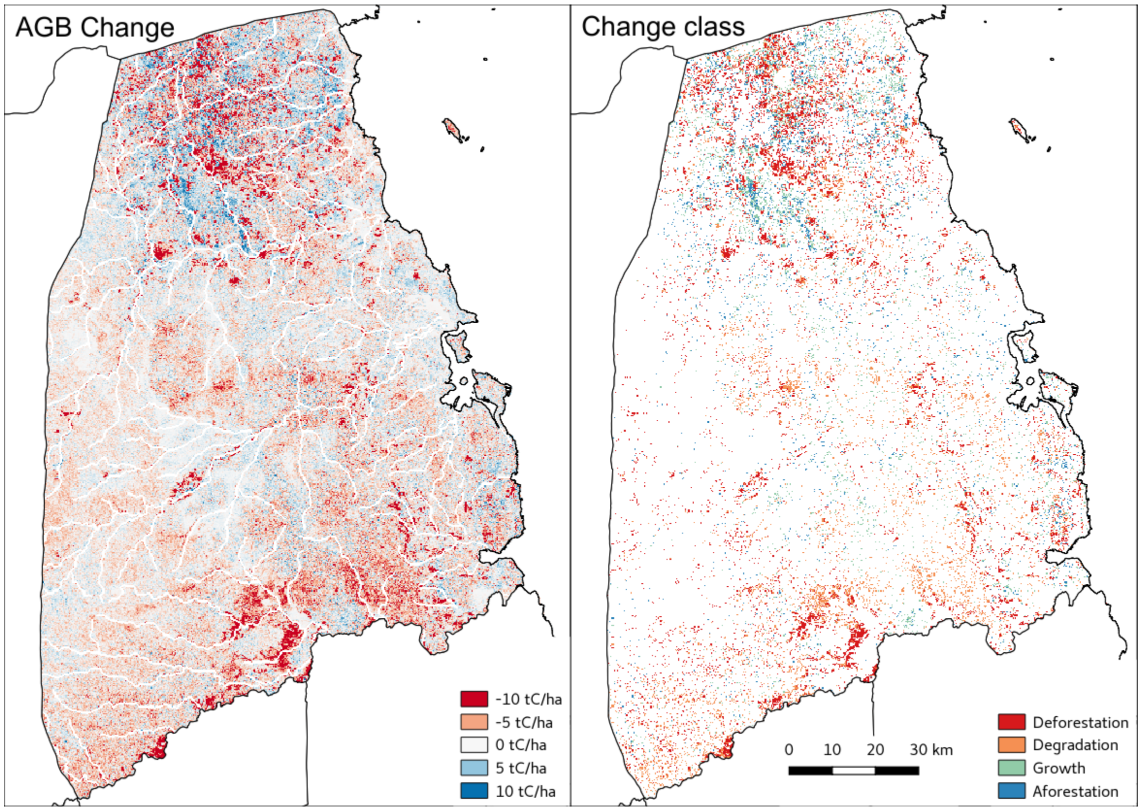
```
# Import the biota module
import biota

# Define a variable with the location of ALOS tiles
data_dir = '~/DATA/'

# Define and output location
output_dir = '~/outputs/'

for latitude in range(-9,-7):
    for longitude in range(38, 40):

        # Update progress
        print 'Doing latitude: %s, longitude: %s'%(str(latitude), str(longitude))

        # Load the ALOS tile with specified options
        try:
            tile_2007 = biota.LoadTile(data_dir, latitude, longitude, 2007, lee_
↪filter = True, forest_threshold = 15., area_threshold = 1)
            tile_2010 = biota.LoadTile(data_dir, latitude, longitude, 2010, lee_
↪filter = True, forest_threshold = 15., area_threshold = 1)
```

(continues on next page)

(continued from previous page)

```
            tile_change = biota.LoadChange(tile_2007, tile_2010, output_dir = output_
↪dir)

        except:
            continue

        # Add river lines to the mask with a 250 m buffer
        tile_2007.updateMask('auxillary_data/TZA_water_lines_dcw.shp', buffer_size =␣
↪250)

        # Load change between tiles, with options
        tile_change = biota.LoadChange(tile_2007, tile_2010, output_dir = output_dir)

        # Calculate AGB change and output
        agb_change = tile_change.getAGBChange(output = True)

        # Calculate change type and output
        change_type = tile_change.getChangeType(output = True)
```

Visualised in QGIS, the resulting AGB change and change type maps for Kilwa District are:



2.7.9 Producing an output with the GUI

If you have produced forest properties with worked example 2, this should be no problem: simply tick the Forest change box, select a second year to make comparisons with and select the desired output(s). If you want to refine your analysis, modify any of the secondary parameters. That's it!

2.8 biota package

2.8.1 Submodules

2.8.2 biota.download module

2.8.3 biota.core module

#Module contents #————— # #.. automodule:: biota # :members: # :undoc-members: # :show-inheritance:

CHAPTER 3

Indices and tables

- genindex
- modindex
- search