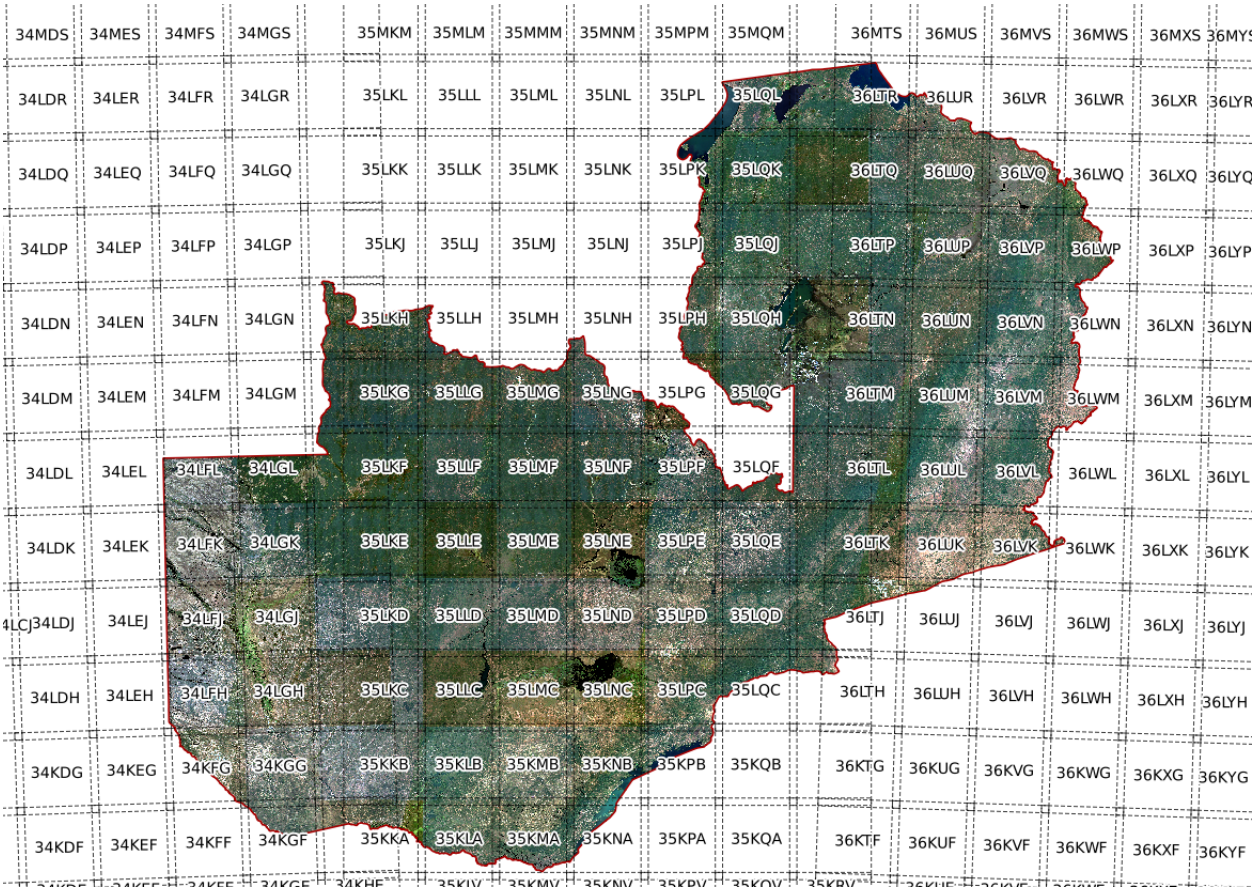
sen2mosaic Documentation

Release 1

Samuel Bowers

Dec 02, 2020

Contents



This is a set of tools to aid in the production of large-scale cloud-free seasonal mosaic products from Sentinel-2 data.

Building cloud-free mosaics of Sentinel-2 data for land cover mapping is difficult, with existing tools still under-development and hard to use. The goal of these tools to streamline this processing chain with a set of straightforward command line tools.

This repository contains four command-line based scripts to perform the following tasks:

- Downloading Sentinel-2 data from the Copernicus Open Access Hub for a particular tile, specifying date ranges and degrees of cloud cover. This is based on the Sentinelsat utility.
- Executing the sen2cor tool to perform atmospheric correction, and performing simple improvements to its cloud mask.
- Mosaicking cloud-free Sentinel-2 pixels from .SAFE files into larger GeoTIFF files that are suitable for image classification.

CHAPTER 1

How do I get set up?

These tools are written in Python for use in Linux. You will need to have first successfully installed the following:

- Sentinelhub: A library for searching and downloading Sentinel-2 products.
- Sen2cor: Atmospheric correction and cloud masking for Sentinel-2.

The sen2cor tool is built around the Anaconda distribution of Python. The modules used in these scripts are all available in Anaconda Python.

CHAPTER 2

Who do I talk to?

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

CHAPTER 3

See also

We have also developed a very similar tool to produce mosaics of Sentinel-1 C-band radar backscatter data. See sen1mosaic.

CHAPTER 4

Contents:

4.1 Setup instructions

4.1.1 Requirements

This toolset is written for use in Linux.

You will need access to a PC or server with at least:

- Python 3
- sen2mosaic
- 8 GB of RAM to run sen2cor.

4.1.2 Installing Anaconda Python

These tools are written in Python. We recommend the Anaconda distribution of Python, which contains all the modules necessary to run these scripts.

To install Anaconda Python, open a terminal window, change directory to the location you'd like to install Anaconda Python, and run the following commands:

```
wget https://repo.anaconda.com/archive/Anaconda3-2019.03-Linux-x86_64.sh
chmod +x Anaconda3-2019.03-Linux-x86_64.sh
./Anaconda3-2019.03-Linux-x86_64.sh
```

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

```
Python 2.7.14 |Anaconda, Inc.| (default, Dec  7 2017, 17:05:42)
[GCC 7.2.0] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

4.1.3 Setting up your Anaconda environment

Note: The Anaconda environment required for sen1mosaic and sen2mosaic is identical. If you already have a sen1mosaic environment set up, it can be used in place of a new 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 `sen2mosaic`. This is done by running the following commands in your terminal or the Anaconda prompt (recommended procedure):

```
conda create -n sen2mosaic -c conda-forge python=3.7 scipy pandas psutil scikit-image␣
↪gdal opencv pyshp
```

Activate the `sen2mosaic` environment whenever opening a new terminal window by running this command:

```
conda activate sen2mosaic
```

4.1.4 Installing sen2cor

sen2cor is an ESA program to perform atmospheric correction and cloud masking of Sentinel-2 level 1C images. It generates a new file containing bottom of atmosphere reflectance values and a cloud mask.

For further details and up-to-date installation instructions, see the sen2cor website.

At the time of writing, sen2cor can be installed using the following commands. sen2cor must be installed after Anaconda Python. Open a terminal window, change directory to the location you'd like sen2cor to be installed, and run the following commands:

```
wget http://step.esa.int/thirdparties/sen2cor/2.8.0/Sen2Cor-02.08.00-Linux64.run
chmod +x Sen2Cor-02.08.00-Linux64.run
./Sen2Cor-02.08.00-Linux64.run
```

Once complete, you need to reference this software in your .bashrc file as follows:

```
echo "source ~/Sen2Cor-02.08.00-Linux64/L2A_Bashrc" >> ~/.bashrc
exec -l $SHELL
```

To test the installation, type `L2A_Process --help` in a terminal window to show running instructions. You should see something that looks like the following:

```
usage: L2A_Process.py [-h] [--mode MODE] [--resolution {10,20,60}]
                      [--datastrip DATASTRIP] [--tile TILE]
                      [--output_dir OUTPUT_DIR] [--work_dir WORK_DIR]
                      [--img_database_dir IMG_DATABASE_DIR]
                      [--res_database_dir RES_DATABASE_DIR]
                      [--processing_centre PROCESSING_CENTRE]
                      [--archiving_centre ARCHIVING_CENTRE]
                      [--processing_baseline PROCESSING_BASELINE] [--raw]
                      [--tif] [--sc_only] [--cr_only] [--debug]
                      [--GIP_L2A GIP_L2A] [--GIP_L2A_SC GIP_L2A_SC]
                      [--GIP_L2A_AC GIP_L2A_AC] [--GIP_L2A_PB GIP_L2A_PB]
                      input_dir

Sentinel-2 Level 2A Processor (Sen2Cor). Version: 2.8.0, created: 2019.02.20,
supporting Level-1C product version 14.2 - 14.5.
```

(continues on next page)

(continued from previous page)

```
positional arguments:
input_dir             Directory of Level-1C input

optional arguments:
-h, --help            show this help message and exit
--mode MODE           Mode: generate_datastrip, process_tile
--resolution {10,20,60}
                        Target resolution, can be 10, 20 or 60m. If omitted,
                        only 20 and 10m resolutions will be processed
--datastrip DATASTRIP
                        Datastrip folder
--tile TILE           Tile folder
--output_dir OUTPUT_DIR
                        Output directory
--work_dir WORK_DIR   Work directory
--img_database_dir IMG_DATABASE_DIR
                        Database directory for L1C input images
--res_database_dir RES_DATABASE_DIR
                        Database directory for results and temporary products
--processing_centre PROCESSING_CENTRE
                        Processing centre as regex: ^[A-Z_]{4}$, e.g "SGS_"
--archiving_centre ARCHIVING_CENTRE
                        Archiving centre as regex: ^[A-Z_]{4}$, e.g. "SGS_"
--processing_baseline PROCESSING_BASELINE
                        Processing baseline in the format: "dd.dd", where
                        d=[0:9]
--raw                 Export raw images in rawl format with ENVI hdr
--tif                 Export raw images in TIFF format instead of JPEG-2000
--sc_only             Performs only the scene classification at 60 or 20m
                        resolution
--cr_only             Performs only the creation of the L2A product tree, no
                        processing
--debug               Performs in debug mode
--GIP_L2A GIP_L2A     Select the user GIPP
--GIP_L2A_SC GIP_L2A_SC
                        Select the scene classification GIPP
--GIP_L2A_AC GIP_L2A_AC
                        Select the atmospheric correction GIPP
--GIP_L2A_PB GIP_L2A_PB
                        Select the processing baseline GIPP
```

4.1.5 Installing sentinelsat

Sentinelsat is the toolset used to access data from the Sentinel-2 archive at the Copernicus Open Access Data Hub.

Up-to-date installation instructions can be found here.

At the time of writing, the installation process is as follows:

```
pip install sentinelsat
```

4.1.6 Installing sen2mosaic

sen2mosaic can be downloaded to a machine from its repository . To do this, open a terminal window and input:

```
git clone https://github.com/smfm-project/sen2mosaic.git
```

To install sen2mosaic, navigate to the sen2mosaic directory and run the following within your sen2mosaic environment.

```
python setup.py install
```

To avoid having to reference the full path of the Python scripts in sen2mosaic, it's a good idea add the following line to your `.bashrc` file:

```
echo "alias s2m='_s2m() { python ~/sen2mosaic/cli/\"\$1\".py \$(shift; echo \"\$@\") ;
↪}; _s2m'" >> ~/.bashrc
```

4.1.7 Is there a Dockerfile?

Coming soon!

4.1.8 Where do I get help?

For help installing sen2cor, it's best to refer to the ESA STEP forum. For assistance in setting up and using sen2mosaic, email sam.bowers@ed.ac.uk.

4.2 Command line tools

The most straightforward way of using sen2mosaic it to call its various stages from the Linux command line. Here the functionality of each of the four commands is explained. In the next section we show how it can be used by example.

Note: Remember, each command line script has a help flag, which can be examined when in doubt.

4.2.1 Getting L1C data

Data from Sentinel-2 are available from the Copernicus Open Access Data Hub, which has a graphical interface to download scenes from selected areas. Whilst useful for smaller areas, generating mosaics at national scales requires a volume of data which makes this extremely labour intensive.

Note: If you already have access to Sentinel-2 data, you can skip straight to `preprocess.py`. This may be the case if you're using a cloud platform where Sentinel-2 data archives are stored at the same location as servers.

The alternative is to download data using the API Hub. This system allows users to search for files using conditions on the command line, and automatically download files. To interface with the API hub, we use an excellent open source utility called Sentinelsat. This operates both as a command line tool, and as a Python API, which we use here. You will need to sign up for an account at Scihub.

`download.py` is a program to interface with Sentinelsat to download Sentinel-2 files, specifying a particular tile, date ranges and degrees of cloud cover. It will also decompress and tidy up .zip files, ready for use with `preprocess.py`.

Help for `download.py` can be viewed by typing `s2m download --help`:

```
usage: download.py [-h] -u USER -p PASS -t [TILES [TILES ...]] [-l LEVEL]
                   [-s START] [-e END] [-c %] [-m MB] [-o PATH] [-r]

Download Sentinel-2 data from the Copernicus Open Access Hub, specifying a
particular tile, date ranges and degrees of cloud cover.

Required arguments:
-u USER, --user USER  Scihub username
-p PASS, --password PASS
                      Scihub password
-t [TILES [TILES ...]], --tiles [TILES [TILES ...]]
                      Sentinel 2 tile name, in format ##XXX

Optional arguments:
-l LEVEL, --level LEVEL
                      Set to search and download level '1C' (default) or
                      '2A' data. Note that L2A data may not be available at
                      all locations.
-s START, --start START
                      Start date for search in format YYYYMMDD. Defaults to
                      20150523.
-e END, --end END     End date for search in format YYYYMMDD. Defaults to
                      today's date.
-c %, --cloud %       Maximum percentage of cloud cover to download.
                      Defaults to 100 % (download all images, regardless of
                      cloud cover).
-m MB, --minsize MB   Minimum file size to download in MB. Defaults to 25
                      MB.
-o PATH, --output_dir PATH
                      Specify an output directory. Defaults to the present
                      working directory.
-r, --remove          Remove level 1C .zip files after decompression.
```

For example, to download all data for tile 36KWA between for May and June 2017, with a maximum cloud cover percentage of 30 %, specifying an output location and removing decompressed .zip files, use the following command:

```
s2m download -u user.name -p supersecret -t 36KWA -s 20170501 -e 20170630 -c 30 -r -o␣
↪/path/to/DATA_dir/
```

Note: What if I want data before 6th December 2016?.

The format in which Sentinel-2 data is distributed was modified in December 2016, and the earlier format is not well supported. As there is a limited volume of data from before this date, we recommend downloading the data from Scihub.

A nice tool to help out with this is aria2. After adding products to your basket at Sentinelhub, you'll download a metadata file called `products.meta4`. Use aria2 to download the file's contents as follows:

```
aria2c --http-user=username --http-passwd=supersecret --check-certificate=false --max-
↪concurrent-downloads=2 -M products.meta4
```

4.2.2 Processing to L2A

Once you have Sentinel-2 (Level 1C) data, the next step is to perform atmospheric correction and identify clouds and cloud shadows. This step is based on sen2cor.

Note: If you already have access to Sentinel-2 L2A, skip straight to `mosaic.py`. This may be the case if you're using a cloud platform where Sentinel-2 data archives are stored at the same location as servers. You can also skip this step if you're happy to build a mosaic using L1C data, but be aware that the output will be of lower quality.

`preprocess.py` takes a list of level 1C .SAFE files as input, initiates sen2cor, and performs simple modifications to improve the quality of it's cloud and cloud shadow mask.

Help for `preprocess.py` can be viewed by typing `s2m preprocess --help`:

```
usage: preprocess.py [-h] [-t TILE] [-g GIPP] [-o DIR] [-res 10/20/60]
                     [-s PATH] [-s255 PATH] [-p N] [-v]
                     [L1C_FILES [L1C_FILES ...]]

Process level 1C Sentinel-2 data from the Copernicus Open Access Hub to level
2A. This script initiates sen2cor, which performs atmospheric correction and
generate a cloud mask. This script also performs simple improvements to the
cloud mask.

Positional arguments:
L1C_FILES             Sentinel 2 input files (level 1C) in .SAFE format.
                        Specify one or more valid Sentinel-2 .SAFE, a
                        directory containing .SAFE files, a Sentinel-2 tile or
                        multiple granules through wildcards (e.g.
                        *.SAFE/GRANULE/*), or a file containing a list of
                        input files. Leave blank to process files in current
                        working directoy. All granules that match input
                        conditions will be atmospherically corrected.

Optional arguments:
-t TILE, --tile TILE  Specify a specific Sentinel-2 tile to process. If
                        omitted, all tiles in L1C_FILES will be processed.
-g GIPP, --gipp GIPP  optionally specify a custom L2A_Process settings file.
-o DIR, --output_dir DIR
                        Specify a directory to output level 2A files. If not
                        specified, atmospherically corrected images will be
                        written to the same directory as input files.
-res 10/20/60, --resolution 10/20/60
                        Process only one of the Sentinel-2 resolutions, with
                        options of 10, 20, or 60 m. Defaults to processing all
                        three. N.B It is not currently possible to only the 10
                        m resolution, an input of 10 m will instead process
                        all resolutions.
-s PATH, --sen2cor PATH
                        Path to sen2cor (v2.8), if not callable with the
                        default 'L2A_Process'.
-s255 PATH, --sen2cor255 PATH
                        Path to sen2cor (v2.5.5), required if processing
                        Sentinel-2 data with the old file format.
-p N, --n_processes N
                        Specify a maximum number of tiles to process in
                        paralell. Bear in mind that more processes will
                        require more memory. Defaults to 1.
-v, --verbose         Make script verbose.
```

For example, to run `preprocess.py` on a set of level 1C Sentinel-2 files in a directory, processing only 20 m resolution data, use the following command:

```
s2m preprocess -res 20 /path/to/DATA_dir/
```

The pre-processing script supports parallel processing of L1C files. Be aware that this will entail greater processing and memory requirements than are available on most standard desktop PCs. To parallel process 3 tiles for the 20 m resolution, input:

```
s2m preprocess -res 20 -n 3 /path/to/DATA_dir/
```

4.2.3 Processing to a mosaic

The final `sen2mosaic` processing step creates a composite image of multiple Sentinel-2 level 2A images in user-specified tiling grid. This script takes L2A data as input, identifies the tiles that fall within the specified spatial extent, and builds a composite image using available data to produce single-band GeoTiff files for easy use in classification systems.

Note: You can also build a mosaic using L1C data, but be aware that the output will be of lower quality.

`mosaic.py` takes a directory containing Sentinel-2 .SAFE files, an output image extent (xmin, ymin, xmax, ymax) and projection EPSG code as inputs, along with a series of options to modify the compositing approach.

Help for `mosaic.py` can be viewed by typing `s2m mosaic --help`:

```
usage: mosaic.py [-h] -te XMIN YMIN XMAX YMAX -e EPSG [-res m] [-l 1C/2A]
                 [-st START] [-en END] [-pc PC] [-m [N [N ...]]] [-b] [-i]
                 [-t DIR] [-o DIR] [-n NAME] [-p N] [-v]
                 [PATH [PATH ...]]

Process Sentinel-2 data to a composite mosaic product to a customisable grid
square, based on specified UTM coordinate bounds. Data are output as GeoTiff
files for each spectral band, with .vrt files for ease of visualisation.

positional arguments:
PATH                   Sentinel 2 input files (level 1C/2A) in .SAFE format.
                         Specify one or more valid Sentinel-2 .SAFE, a
                         directory containing .SAFE files, a Sentinel-2 tile or
                         multiple granules through wildcards (e.g.
                         *.SAFE/GRANULE/*), or a file containing a list of
                         input files. Leave blank to process files in current
                         working directoy. All granules that match input
                         conditions will be included.

required arguments:
-te XMIN YMIN XMAX YMAX, --target_extent XMIN YMIN XMAX YMAX
                         Extent of output image tile, in format <xmin, ymin,
                         xmax, ymax>.
-e EPSG, --epsg EPSG  EPSG code for output image tile CRS. This must be UTM.
                         Find the EPSG code of your output CRS as
                         https://www.epsg-registry.org/.
-res m, --resolution m
                         Specify a resolution in metres.

optional arguments:
-l 1C/2A, --level 1C/2A
                         Input image processing level, '1C' or '2A'. Defaults
```

(continues on next page)

(continued from previous page)

```
                        to '2A'.
-st START, --start START
                        Start date for tiles to include in format YYYYMMDD.
                        Defaults to processing all dates.
-en END, --end END    End date for tiles to include in format YYYYMMDD.
                        Defaults to processing all dates.
-pc PC, --percentile PC
                        Specify a percentile of reflectance to output.
                        Defaults to 25 percent, which tends to produce good
                        results.
-m [N [N ...]], --masked_vals [N [N ...]]
                        Specify SLC values to not include in the mosaic (e.g.
                        -m 7 8 9). See http://step.esa.int/main/third-party-
                        plugins-2/sen2cor/ for description of sen2cor mask
                        values. Defaults to 'auto', which masks values 0 and
                        9. Also accepts 'none', to include all values.
-b, --colour_balance  Perform colour balancing between tiles. Not generally
                        recommended, particularly where working over large
                        areas. Defaults to False.
-i, --improve_mask    Apply improvements to Sentinel-2 cloud mask. Not
                        generally recommended, except where a very
                        conservative mask is desired. Defaults to no
                        improvement.
-t DIR, --temp_dir DIR
                        Directory to write temporary files, only required for
                        L1C data. Defaults to '/tmp'.
-o DIR, --output_dir DIR
                        Specify an output directory. Defaults to the present
                        working directory.
-n NAME, --output_name NAME
                        Specify a string to precede output filename. Defaults
                        to 'mosaic'.
-p N, --n_processes N
                        Specify a maximum number of tiles to process in
                        paralell. Bear in mind that more processes will
                        require more memory. Defaults to 1.
-v, --verbose         Make script verbose.
```

For example, to run `mosaic.py` in the directory `/path/to/DATA_dir/` which contains level 2A files to create a 200 x 200 km output tile in the UTM36S projection at 20 m resoluton, input:

```
s2m mosaic -te 700000 7900000 900000 8100000 -e 32736 -res 20 /path/to/DATA_dir/
```

To do the same operation, but specifying an output directory, a name to prepend to outputs from this tile, and performing inter-scene colour balancing and corrections to the sen2cor mask, input:

```
s2m mosaic -te 700000 7900000 900000 8100000 -e 32736 -res 20 -o /path/to/output/ -n␣
↪my_output -b -c /path/to/DATA_dir/
```

4.3 Worked example on the command line

Here we'll show you by example how the sen2mosaic processing chain works in practice. We will focus on an example from southern Mozambique, with the aim of creating a cloud-free composite GeoTiff product for the area **500,000 - 600,000** m Eastings and **7,500,000 - 7,700,000** m Northings (**UTM 36S**). This area is covered by Sentinel-2 tiles

36KWA and **36KWB**. We'll limit this mosaic to the early dry season (**May and June**) of **2017**, in anticipation of multiple seasonally-specific mosaics improving classification accuracy.

4.3.1 Preparation

First ensure that you've followed setup successfully.

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

```
cd /home/user/DATA
mkdir worked_example
cd worked_example
```

Use mkdir to make a separate folder for each of the granules you intend to download.

```
mkdir 36KWA
mkdir 36KWB
```

Here we'll demonstrate the process for the tile 36KWA. We'll leave 36KWB for you to do without guidance.

To begin, navigate to the 36KWA folder.

```
cd 36KWA
```

4.3.2 Downloading data

The first step is to download Sentinel-2 level 1C data from the Copernicus Open Access Data Hub.

For this we use the `L1C.py` tool. We will need to specify a Scihub username and password (sign up for an account at Scihub), the tile to download, a start and end date in the format YYYYMMDD, and a maximum degree of cloud cover to download. For the purposes of this demonstration, we'll set maximum cloud cover to 30 %.

These options can be encoded as follows:

```
s2m download -u user.name -p supersecret -t 36KWA -s 20170501 -e 20170630 -c 30
```

As we didn't specify the option `-o` (`--output`), data will output to the current working directory. We also didn't include the `-r` (`--remove`) flag, meaning that intermediate .zip files downloaded from the internet won't be deleted. This can quickly result in large volumes of data building up, so if you're limited by disk space use the `-r` flag.

Wait for all files to finish downloading before proceeding to the next step. By the time the processing is complete, your `36KWA/` directory should contain the following files (show files in the currenty working directory with the command `ls`).

```
S2A_MSIL1C_20170506T074241_N0205_R049_T36KWA_20170506T075325.SAFE
S2A_MSIL1C_20170506T074241_N0205_R049_T36KWA_20170506T075325.zip
S2A_MSIL1C_20170516T072621_N0205_R049_T36KWA_20170516T075513.SAFE
S2A_MSIL1C_20170516T072621_N0205_R049_T36KWA_20170516T075513.zip
S2A_MSIL1C_20170519T075221_N0205_R092_T36KWA_20170519T080547.SAFE
S2A_MSIL1C_20170519T075221_N0205_R092_T36KWA_20170519T080547.zip
S2A_MSIL1C_20170526T074241_N0205_R049_T36KWA_20170526T074901.SAFE
S2A_MSIL1C_20170526T074241_N0205_R049_T36KWA_20170526T074901.zip
S2A_MSIL1C_20170529T073611_N0205_R092_T36KWA_20170529T075550.SAFE
S2A_MSIL1C_20170529T073611_N0205_R092_T36KWA_20170529T075550.zip
S2A_MSIL1C_20170605T072621_N0205_R049_T36KWA_20170605T075534.SAFE
S2A_MSIL1C_20170605T072621_N0205_R049_T36KWA_20170605T075534.zip
```

(continues on next page)

(continued from previous page)

```
S2A_MSIL1C_20170608T075211_N0205_R092_T36KWA_20170608T080546.SAFE
S2A_MSIL1C_20170608T075211_N0205_R092_T36KWA_20170608T080546.zip
S2A_MSIL1C_20170628T075211_N0205_R092_T36KWA_20170628T080542.SAFE
S2A_MSIL1C_20170628T075211_N0205_R092_T36KWA_20170628T080542.zip
```

4.3.3 Atmopsheric correction and cloud masking

The next step is to perform atmospheric correction (removes the effects of the atmosphere on refectance values of images) and cloud masking (identifies clouds in images.) to generate Sentinel-2 level 2A data. We do this with the ESA program `sen2cor`.

Note: As of Q4 2018, ESA is generating the L2A product systematically and making it available for download. If you download the L2A data directly, this preprocessing step can be skipped.

To perform atmospheric correction and cloud masking we call the tool `preprocess.py`. We need to specify Sentinel-2 level 1C input files, a directory containing level 1C files, or a single tile within a .SAFE file `*.SAFE/GRANULE/*`).

To process all .SAFE files for the tile 36KWA (in the current working directory) at 20 m resolution, we can submit the following line:

```
s2m preprocess -res 20 -t 36KWA -v
```

This command will loop through each Sentinel-2 level 1C file and process them one at a time. You might alternatively want to run several commands similtaneously using the `-p` flag, but bear in mind that this will require access to a large quanity of memory.

Here we didn't specify the options `-o` (`--output_dir`) and `--g` (`--gipp`), which can be used to output data to a location other than the directory containing input files, or the `-r` (`--remove`) option, which would delete Sentinel-2 level 1C data once data is finished processing.

Wait for all files to be processed to level 2A before proceeding. If you run `ls` again, your `36KWA/` directory should now contain a new set of files:

```
...
S2A_MSIL2A_20170506T074241_N0205_R049_T36KWA_20170506T075325.SAFE
S2A_MSIL2A_20170516T072621_N0205_R049_T36KWA_20170516T075513.SAFE
S2A_MSIL2A_20170519T075221_N0205_R092_T36KWA_20170519T080547.SAFE
S2A_MSIL2A_20170526T074241_N0205_R049_T36KWA_20170526T074901.SAFE
S2A_MSIL2A_20170529T073611_N0205_R092_T36KWA_20170529T075550.SAFE
S2A_MSIL2A_20170605T072621_N0205_R049_T36KWA_20170605T075534.SAFE
S2A_MSIL2A_20170608T075211_N0205_R092_T36KWA_20170608T080546.SAFE
S2A_MSIL2A_20170628T075211_N0205_R092_T36KWA_20170628T080542.SAFE
```

4.3.4 Repeat for other tiles

The download and atmospheric correction stages need to be repeated for each tile for your area of interest.

Now it's your turn! `cd` to the 36KWB folder, and generate a Sentinel-2 level-3 image using the methods we've just employed for tile 36KWA.

4.3.5 Generating a cloud-free mosaic image

Each of these Sentinel-2 level 2A images is now atmospherically corrected, but each still contains masked areas of cloud. The goal of this step is to combine the cloud-free pixels of each image to generate a single cloud-free composite image composed of multiple satellite overpasses. This step also converts data from the Sentinel-2 .SAFE format to the easy to work with GeoTiff format, and allows the specification of a customised tiling grid. We recommend a grid of tiles that's approximately equal to the area of four Sentinel-2 tiles (~200,000 x 200,000 m).

Here we only have two tiles (36KWA and 36KWB), so we'll just perform a small-scale demonstration, generating an output with the limits **500,000 - 600,000** m Eastings and **7,500,000 - 7,700,000** m Northings (**UTM 36S**) at **20** m resolution.

To perform this step we call the tool `mosaic.py`. We need to specify the location of all input files (with wildcards), the exent of the output image and the EPSG code describing the output coordinate reference system (UTM 36S = 32736). We'll also give output data a name to identfy this tile.

First cd to the directory containing all Sentinel-2 L2A data.

```
cd /home/user/DATA/worked_example/
```

To run `mosaic.py`,

```
s2m mosaic -te 500000 7500000 600000 7700000 -e 32736 -res 20 -n worked_example -v ./
↪36KW*
```

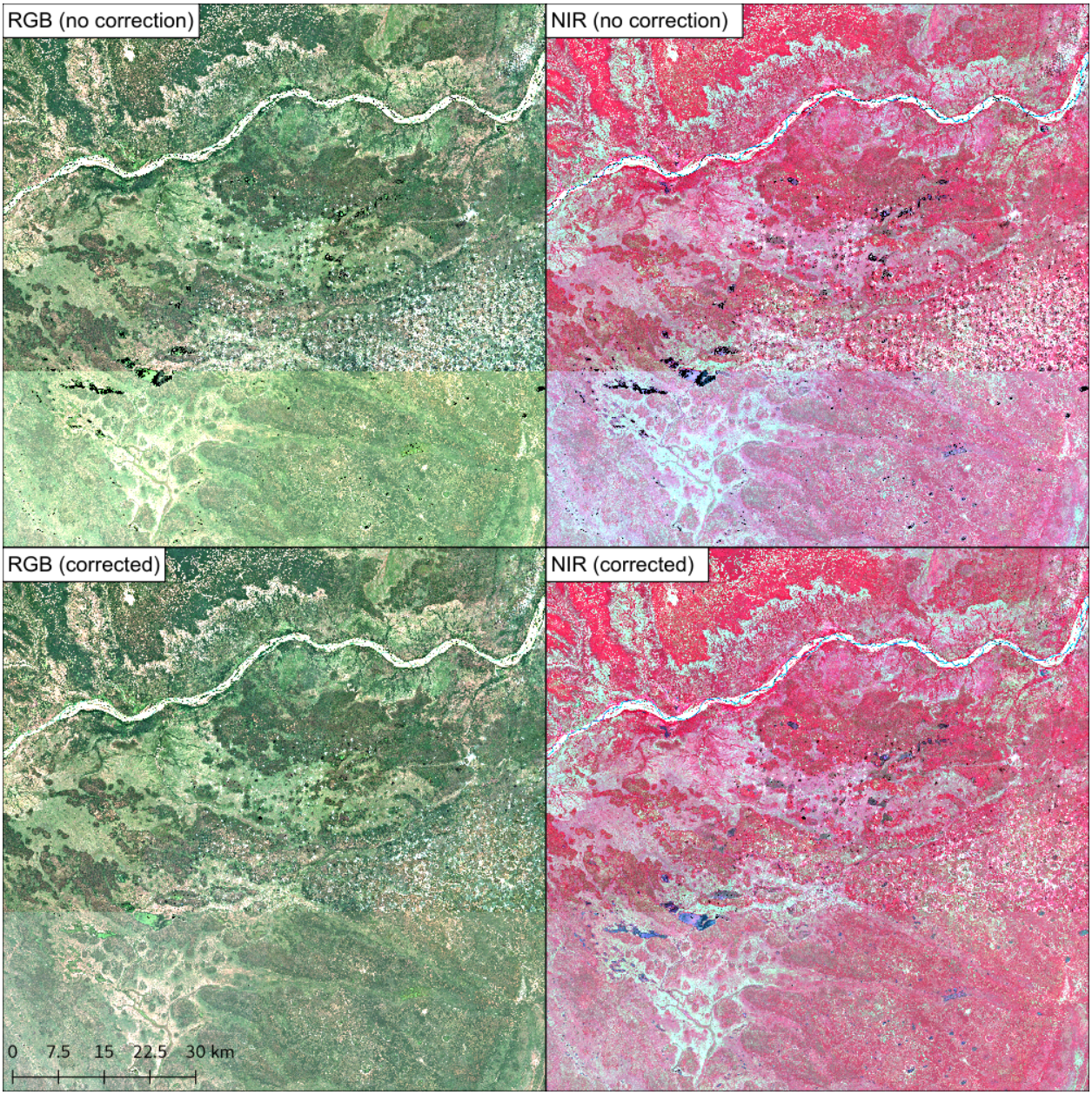
Here we didn't specify the `-o` (`--output_dir`) option, meaning that results will be output to the current working directory. Once processing is complete, you can use `ls` to view the newly created output files:

```
worked_example_R20m_B02.tif
worked_example_R20m_B03.tif
worked_example_R20m_B04.tif
worked_example_R20m_B05.tif
worked_example_R20m_B06.tif
worked_example_R20m_B07.tif
worked_example_R20m_B09.tif
worked_example_R20m_B11.tif
worked_example_R20m_B12.tif
worked_example_R20m_B8A.tif
worked_example_R20m_SLC.tif
```

The files `B01` to `B12` represent individual Sentinel-2 spectral bands, *, and ''SCL'* records the mask value for each pixel for each band (generally acceptable values are: 4 = vegetation, 5 = bare soils, 6 = water).

4.3.6 Viewing data

In addition to a GeoTiff file for each Sentinel-2 band, `mosaic.py` outputs two 3-band GDAL virtual dataset files (`.vrt`). These are labelled `_RGB.vrt` and `_NIR.vrt`, and can be opened in QGIS to show a true colour (Red, Green, Blue) and false colour composite (NIR, Red, Green) image.



4.4 Using sen2mosaic in Python

This is harder than the command line, but you may be interested in importing the sen2mosaic functions into Python in order to customise the processing chain.

To make sen2mosaic accesible in Python, edit your `.bashrc` file (located at `~/.bashrc`) to contain the line:

```
export PYTHONPATH=$PYTHONPATH:/path/to/sen2mosaic/
```

You should now be able to import each of the four modules in Python as follows:

```
import sen2mosaic.download
import sen2mosaic.preprocess
import sen2mosaic.mosaic
import sen2mosaic.utilities
```

Help for each function can be accessed interactively from Python. For example:

```
>>> help(sen2mosaic.download.connectToAPI)
        Help on function connectToAPI in module sen2mosaic.download:

        connectToAPI(username, password)
        Connect to the SciHub API with sentinelsat.

        Args:
            username: Scihub username. Sign up at https://scihub.copernicus.eu/.
            password: Scihub password.
```

On this page each of the functions from the `download`, `preprocess`, and `mosaic` modules are documented. A further module named `utilities` contains generic functions for processing Sentinel-2 data. Note that the `main()` function in each is what is driven by the command line tools, so in addition to it's component parts you can call the entire processing chain from Python.

4.4.1 Download module

4.4.2 Preprocess module

4.4.3 Mosaic module

4.4.4 Utilities module

CHAPTER 5

Indices and tables

- genindex
- modindex
- search