Uptrop

Release 2.0.0-alpha

Eloise Marais

Mar 08, 2023

OVERVIEW OF CAPABILITIES

Uptrop is python software to convert satellite observations of total atmospheric column densities of air pollutants to vertical profiles in the global troposphere using the cloud-slicing technique.

The software was developed using total column densities of nitrogen dioxide (NO_2) from the ESA Sentinel-5P TROPOMI instrument to derive NO_2 mixing ratios in the global upper troposphere (8-12 km) at ~100 km spatial resolution. The software is now being updated to also retrieve NO_2 and ozone (O_3) mixing ratios in five vertical layers throughout the troposphere.

This brief documentation provides details of relevant references, data products, sample output, guidance on installing and using the software, and how to ask for help or report bugs, issues and suggested code updates.

CHAPTER

ONE

CONTENTS

1.1 Overview of Capabilities

Uptrop-Py is python coded software used to apply the so-called cloud-slicing technique to satellite observations of column densities of nitrogen dioxide (NO_2) to retrieve vertical profiles of mixing ratios of NO_2 throughout the troposphere on a global scale.

Code output includes data in NetCDF format, plots in postscript format, and metrics at the end of the log file. Sample plots and logfile metrics are in the Gallery.

So far the code has been applied to Sentinel-5P TROPOMI instrument obesrvations of total columns of NO_2 using cloud information from two distinct TROPOMI cloud products to obtain NO_2 concentrations in a single layer in the upper troposphere from 450 hPa (~8 km) to 180 hPa (~12 km).

The steps in the retrieval algorithm are detailed in the open access peer-reviewed Marais et al. (2021) paper published in Atmospheric Measurement Techniques.

The data generated for this paper are available for public download from the UCL Data Repository.

A static version of the software that was used in the above paper to generate the above data is hosted on Zenodo and available for public download and use as version 1.1.0.

1.2 Code Updates in Progress

A new and improved version of the software is under development. It includes many updates that are categorised below.

1.2.1 Data / Output

- Change regression fit from reduced major axis to Theil, so that cloud-slicing regression fit is less impacted by non-uniformally distributed data leading to an overestimate in mixing ratios over remote locations.
- No longer bias correct tropospheric columns of NO_2, as this bias correction was due to application of the reduced major axis regression fit to non-uniformally distributed data.

1.2.2 Increased Flexibility

- Adjust the code to be able to apply it to multiple pollutants. NOt just NO2. Currenlty under development is application to TROPOMI ozone to retrieve veritcal profiles of tropospheric ozone, a potent greenhouse gas.
- Ability to obtain mixing ratios in all layers in the troposphere, with the flexibility for the user to specify the pressure range, limited to a range that is at least 100 hPa for the cloud-slicing routine to succeed.
- Enable flexibility to select any start and end date, not just limited to processing data for a single season.

1.2.3 Structural Changes

- Replace Basemap with Cartopy for generating sample plots.
- Restructure code directories
- Improve variable names in the Python code and in the output NetCDF data file

If you're interested in using a version of the software currently under development, please reach out to the developers via the GitHub Issues page.

1.3 Code Structure

1.3.1 Core Code

Used for cloud-slicing satellite observations of total column densities of NO2:

./erc-uptrop/uptrop/cloud_slice_tropomi_no2.py

Module that reads in TROPOMI data, calls the cloud slicing routine, grids data, calculates seasonal means, outputs data and sample plots.

./erc-uptrop/uptrop/cloud_slice_no2.py

Routine that cloud-slices the TROPOMI NO2 data prepared in `cloud_slice_tropomi_no2.py`. Calculates NO2 concentration and associated error in pptv.

./erc-uptrop/uptrop/height_pressure_converter.py

Routine to convert cloud top height to cloud top pressure. Used for the TROPOMI CLOUD_OFFL product only.

./erc-uptrop/uptrop/boostrap.py

Routine that calculates the reduced major axis regression slope and intercept values and errors estimated with bootstrapping. Adapted into Python from the IDL GAMAP package.

./erc-uptrop/uptrop/gamap_colormap.py

The white-yellow-green-orange-red colorbar from the GCPy Python package.

./erc-uptrop/uptrop/constants.py

Constants and conversion factors used for cloud-slicing.

1.3.2 Additional Code

Used by Marais et al. (2021) to assess the TROPOMI NO2 columns and cloud products:

./erc-uptrop/uptrop/fresco_cld_error.py

Routine to compare two TROPOMI cloud products used to cloud-slice TROPOMI NO2.

./erc-uptrop/uptrop/read_pandora.py

Routine to read in ground-based Pandora total and tropospheric NO2 column data in the format provided by the Pandonia Global Network.

./erc-uptrop/uptrop/compare_tropomi_pandora.py

Routine to sample and compare coincident data from TROPOMI and Pandora total and tropospheric NO2 columns at high-altitude sites.

1.4 Installation

To download the latest stable version of the cloud-slicing uptrop-py code, type:

```
$ git clone https://github.com/eamarais/erc-uptrop.git
```

This command clones the source code to a folder called erc-uptrop

To download GEOS-Chem Classic source code into a folder named something other than `erc-uptrop`, supply the name of the folder at the end of the **`git clone`** command. For example:

```
$ git clone https://github.com/eamarais/erc-uptrop.git my-code-dir
```

This will download the source code into `my-code-dir` instead of `erc-uptrop`.

Once the **`git clone`** process starts, you should see output similar to this:

```
Cloning into 'erc-uptrop'...
remote: Enumerating objects: 1568, done.
remote: Counting objects: 100% (485/485), done.
remote: Compressing objects: 100% (277/277), done.
remote: Total 1568 (delta 322), reused 297 (delta 195), pack-reused 1083
Receiving objects: 100% (1568/1568), 1.25 MiB | 10.35 MiB/s, done.
Resolving deltas: 100% (1086/1086), done.
```

When the **`git clone`** process has finished, get a directory listing:

```
$ ls -CF erc-uptrop/*
```

and you will see output similar to this:

```
docs/  environment.yml  LICENSE  README.md  tests/  uptrop/
```

Navigate into the `erc-uptrop` folder and confirm that the code is in the main branch:

```
$ cd erc-uptrop/
$ git branch
```

You will see output similar to this:

```
* main
```

If you plan to modify the code, either to add new features or fix bugs, create a branch to store these changes.

To do so, type:

```
$ git branch feature/my-git-updates
$ git checkout feature/my-git-updates
```

Instead of `feature/my-git-updates`, you may choose a name that reflects the nature of your updates (e.g. `feature/fix_bug`, `feature/add_compound` etc.) If you now type:

```
$ git branch
```

You will see that we are checked out onto the branch that you just created.

```
* feature/my-git-updates
main
```

1.5 Usage

Instructions below assume python is being simulated from the command line using a conda virtual environment and python version 3.9.10.

Dependencies are listed in the environment.yml file that is downloaded with the source code. These include:

```
Numpy, NetCDF4, Dateutil, Matplotlib, Basemap
```

To cloud-slice TROPOMI NO2 to obtain seasonal mean NO2 mixing ratios for June-August 2019 at 180-320 hPa on a 1 degree x 1 degree global grid, enter the following at the command line:

```
$ python cloud_slice_tropomi_no2.py --trop_dir="path-to-tropomi-data/" --out_dir="path-
↪to-output-directory/" --cloud_product="fresco-wide" --no2_prod="OFFL" --cloud_
↪threshold="07" --grid_res="1x1" --year="2019" --pmax="180" --pmin="450" --season="jja"␣
↪> log_file
```

All arguments are input as strings.

Default input arguments are:

```
--grid_res="1x1"
--cloud-product="fresco-wide"
--cloud-threshold="07"
--pmin="180"
--pmax="450"
--no2-prod="PAL"
--version="v1"
```

Resolution options are limited to, in degrees latitude by degrees longitude:

```
1x1, 2x2.5, 4x5
```

The TROPOMI satellite data being read in assumed to be stored in the format ./NO2_$no2_prod/$year/$month/, where year and month are extracted from the season input information.

Output directory assumes to have ./Data/ subdirectory to store NetCDF files and ./Images/ subdirectory to store plots.

The –cloud_threshold input argument should be "07", "08", "09", "10" to limit use to optically thick clouds and associated contamination from the target compound below clouds.

Recommended cloud top pressure ranges to use include 180-320 hPa, 320-450 hPa, 450-600 hPa, 600-800 hPa, and 800-1000 hPa. The code also offers the flexibility for the user to decide on cloud top height ranges difference to those recommended, but the difference between –pmin and –pmax must exceed 100 hPa. If not, the code will stop with an error message.

1.6 Help and reporting issues

GitHub is used to ask for help and reporting issues or bugs.

To ask a question, report a bug, or make a suggestion, open a new issue. When you do, please include your name and institution and include all relevant information for replicating the bug or issue.

1.7 Gallery

1.7.1 Data

The NetCDF file generated at the end of a successful cloud-slicing simulation has the following contents:

```
netcdf file:/path/to/file/tropomi-ut-no2-fresco-07-1x1-jja-2019-v010101.nc {
  dimensions:
    lon = 361;
    lat = 181;
  variables:
    float lon(lon=361);
      :units = "degrees_east";
      :long_name = "longiitude";

    float nobs(lon=361, lat=181);
      :units = "unitless";
      :long_name = "Number of observations in each gridsquare used to obtain cloud-
↪sliced UT NO2 mixing ratios";

    float cld_top_p_ceil(lon=361, lat=181);
      :units = "hPa";
      :long_name = "Gridded mean ceiling of cloud top pressures used to cloud-slice␣
↪TROPOMI NO2";

    float cld_top_p_range(lon=361, lat=181);
      :units = "hPa";
      :long_name = "Gridded mean range in cloud top pressures used to cloud-slice␣
↪TROPOMI NO2";

    float utno2err(lon=361, lat=181);
      :units = "pptv";
      :long_name = "Standard error of the NO2 mixing ratios in the UT (180-450 hPa)␣
↪obtained using cloud-slicing";
```

(continues on next page)

(continued from previous page)

```
    float utno2(lon=361, lat=181);
      :units = "pptv";
      :long_name = "NO2 mixing ratios in the UT (180-450 hPa) obtained using cloud-
↪slicing";

    float lat(lat=181);
      :units = "degrees_north";
      :long_name = "latitude";

  // global attributes:
}
```

Variables output in the NetCDF file include:

lon
: Centre longitude values for the target grid (1 degree in the above example) [1D]

lat
: Centre latitude values for the target grid (1 degree in the above example) [1D]

nobs
: The number of cloud-sliced NO2 data points in each gridbox [2D]

utno2
: Gridded mean cloud-sliced NO2 mixing ratios in pptv (seasonal means for a single year in the above example) [2D]

utno2err
: Gridded mean standard error on the cloud-sliced NO2 mixing ratios in pptv [2D]

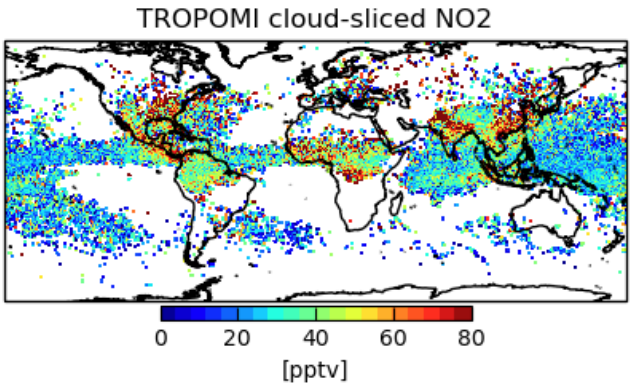
cld_top_p_range
: Gridded mean cloud top pressure range in hPa [2D] Useful if comparing cloud-sliced NO2 to in situ or model output,as ensures data cover consistent altitude ranges.

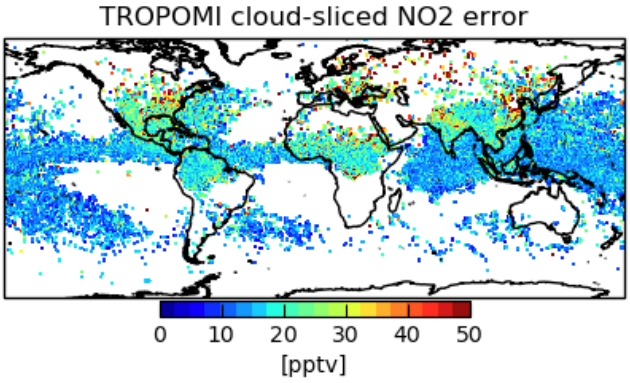
cld_top_p_ceil
: Gridded mean cloud top pressure ceiling in hPa [2D] Useful if comparing cloud-sliced NO2 to in situ or model output,as ensures data cover consistent altitude ranges.

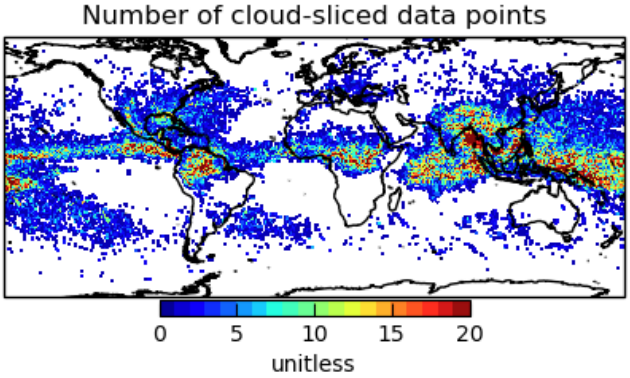
1.7.2 Sample plots

Plots output with uptrop using cartopy include three panels of global maps showing cloud-sliced nitrogen dioxide (NO2) mixing ratios at the pressure range of interest (top), the estimated error on the cloud-sliced NO2 mixing ratios, and the number of cloud-sliced values. These are not publication quality plots, but are merely for sanity checking and benchmarking.

The example below is from cloud-slicing TROPOMI NO2 in June-August 2019 at 450-180 hPa (~8-12 km):


TROPOMI cloud-sliced NO2
0
20
40
60
80
[pptv]


TROPOMI cloud-sliced NO2 error
0
10
20
30
40
50
[pptv]


Number of cloud-sliced data points
0
5
10
15
20
unitless

1.7.3 Log file

The log file tracks the progress of the code and at the end of the simulation outputs metrics. These include the maximum number of satellite pixels in the target grid, the number of satellite pixels removed in each data filtering step, the total number of successful cloud-slicing retrievals compared to the total number that could have been retrieved, and the percent of total TROPOMI pixels used for cloud slicing:

```
Max no. of data points in a gridsquare:  64.0
(1) Too few points:  280605
(2) Low cloud height range:  260019
(3) Low cloud height std dev:  2105
(4) Large error:  0
(5) Significantly less than zero:  15664
(6) Outlier (NO2 > 200 pptv):  0
(7) Non-uniform stratosphere:  133461
(8) Successful retrievals:  83516
(9) Total possible points:  775370
Mean % points retained:  2.141713715255334
```

The printout above indicates that the gid with the most cloud-sliced NO2 retrievals has 64 data points, that, of all valid clusters of satellite pixels within the pressure range of interest, 280,605 have too few coincident points, 260,019 have a cloud height range that is less than required, 2,105 have a cloud height stnadrd deviation that is less than required, 0 have too large a cloud-sliced NO2 error, 15,664 have cloud-sliced NO2 that is statistically significantly less than zero, that 0 are outliers, and that for 133,461 the overlying stratospheric NO2 is not uniform. Also included in the printout is that there were 83,516 successful retrievals out of a total of 775,370, and that of all satellite pixels 2.14% are good quality, fall within the cloud pressure range of interest over optically thick clouds.

INDEX

Symbols

C

L

N

U