PROFFAST User Manual

This manual is intended to provide concise instructions for the use of PROFFAST. PROFFAST is a software package for retrieving trace gas concentrations from interferograms measured with Bruker EM27/SUN solar absorption FTIR spectrometer by using Bruker OPUS software. The PROFFAST software package is developed at the Karlsruhe Institute of Technology (KIT) and funded by the European Space Agency (ESA). Recently, PROFFASTpylot was created to run PROFFAST under Python.

The combination of a common instrumental standard for EM27/SUN spectrometers and a common data analysis procedure with PROFFAST provides a standard for greenhouse gas measurements within the COllaborative Carbon Column Observing Network (COCCON) [1] [2].

Contents

1. Naming, required software and legend

Naming

The software described in this manual is called PROFFAST. It includes the trace gas analysis packages PROFFASTpreprocess, PROFFASTpcxs, PROFFASTinvers, and the Python interface PROFFASTpylot. The first three parts together are often referred to as PROFFAST in following. This structure becomes clearer when looking at the Schematic representation of the PROFFAST algorithm on page 15.

Required external software	*Software used in this manual*
• git	git [3] *for Windows*
• A program to run git	*git bash command line interface (CLI)*
• A program to run ftp (file transfer protocol)	*filezilla* [4]
• Python [5]	*Python 3.11*
• A CLI to run python	*Windows PowerShell*

git bash is installed within the git package.

These software packages are all open-source and available for all common operating systems.

*[Note: Git is used to get the PROFFASTpylot package onto your system. A method without git will also be introduced, but git is **highly recommended** as it makes updating much easier.]*

The examples in this manual were carried out under Windows 10 Enterprise. Shells and corresponding commands in other operating systems may differ. Information is provided at the relevant point.

Legend

To be done **once** to set up PROFFAST(pylot) on your system.

To be done for **each new** PROFFAST(pylot) retrieval.

Retrieval

Measured input data.

Output data.

Data

`Command Line Interface` The command line interface (e.g., git bash) used for a specific operation.

`Shell commands, file names and paths are written in this font.`

2. Literature overview

There are already documentations for both, PROFFAST and PROFFASTpylot, that build the fundament for this manual. These can be found here:

- PROFFAST: You will find some documentation in `prf/docs` when you have installed PROFFAST (see in Installation of PROFFAST on page 3).
- PROFFASTpylot Documentation [6] (website), GitLab PROFFASTpylot [7] (GitLab repository)
- Material from the COCCON telephone conferences, including Q&A and presentations of measurement campaigns, can be found here: COCCON telcos [8].

3. Installation of PROFFAST

In this section, the download and installation of the software packages are demonstrated. The presented information can be found in more detail in PROFFASTpylot documentation [6] → Installation. *[As mentioned above, one can get the PROFFASTpylot package in two ways: With git (recommended) or by downloading a zip-archive. Both are explained in the following.]*

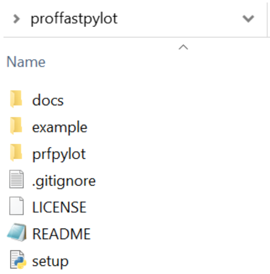
Download PROFFASTpylot with git

What to do	What should happen
Navigate: `cd *Path where you want to store PROFFAST*`	`user@EM27/SUN/~> cd proffast_path/` `user@EM27/SUN/proffast_path>` — Git Bash *(In this manual, git bash is used to run git from a command line under Windows.)*

Clone git repository:
 	
`git clone https://gitlab.eudat.eu/coccon-kit/proffastpylot.git`
	`user@EM27/SUN/proffast_path> git clone https://gitlab.eudat.eu/cocc`
	`fastpylot.git`
	`Cloning into 'proffastpylot'...`
	`remote: Enumerating objects: 2566, done.`
	`remote: Counting objects: 100% (68/68), done.`
	`remote: Compressing objects: 100% (64/64), done.`
	`remote: Total 2566 (delta 28), reused 0 (delta 0), pack-reused 2498`
	`Receiving objects: 100% (2566/2566), 1.48 MiB | 9.97 MiB/s, done.`
	`Resolving deltas: 100% (1642/1642), done.`
	`user@EM27/SUN/proffast_path>`

Alternative: ***Download zip-file*** *from https://gitlab.eudat.eu/coccon-kit/proffastpylot/-/archive/master/proffastpylot-master.zip and* ***unpack it****. You will find a directory* `proffastpylot-master` *with all necessary files. Change the name to* `proffastpylot`.

New directory `proffastpylot` created!
(The `.gitignore` file only shows up with git.)



Update PROFFASTpylot with git

What to do	What should happen
Navigate to the `proffastpylot` folder: `cd proffastpylot`	`user@EM27/SUN/proffast_path> cd proffastpylot/` `user@EM27/SUN/proffastpylot>` — Git Bash

Pull the newest version from the remote repository:
 	
`git pull`
	`user@EM27/SUN/proffastpylot> git pull`
	`Already up to date.`
	
	(This feedback is different, if a new version is available.)

Alternative: *Repeat the download of the zip-file as shown above.*

Add PROFFAST

What to do

Download the newest version of PROFFAST from the KIT website:

https://www.imk-asf.kit.edu/english/3225.php

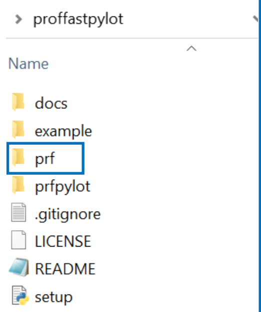
Unzip the zip-file and **move** the `prf` folder to your `proffastpylot` folder.

*Additionally for **Linux users**: Run the installation script to create the executables:*

```
bash install_proffast_linux.sh
```

New folder prf added!

(The `.gitignore` file only shows up with git.)



Create a virtual python environment (venv)

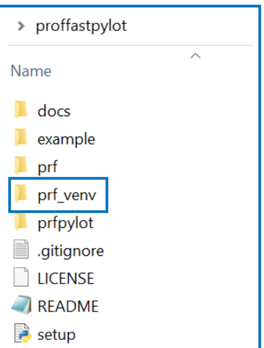
What to do	What should happen

Navigate to `proffastpylot` directory and **create the virtual python environment:**
 	
`python -m venv prf_venv`
	`user@EM27/SUN/proffast_path> cd .\proffastpylot\`
	`user@EM27/SUN/proffastpylot> python -m venv prf_venv`
	`user@EM27/SUN/proffastpylot>`
	Windows PowerShell

Using a [virtual environment](#) [17] avoids version conflicts within the base python installation on your system.

New folder `prf_venv` created!

(The .gitignore file only shows up with git.)



Activate the virtual environment

What to do	What should happen

In the `proffastpylot` directory, **run:**
`.\prf_venv\Scripts\Activate.ps1`
(Windows PowerShell)*
`.\prf_venv\Scripts\activate`
(Windows CMD)
`source prf_venv/bin/activate`
(Linux)
 	
[To deactivate: `deactivate`**]**
	`user@EM27/SUN/proffastpylot> .\prf_venv\Scripts\Activate.ps1`
	`(prf_venv) user@EM27/SUN/proffastpylot>`
	Windows PowerShell*
	Virtual environment activated!
	
	`(prf_venv) user@EM27/SUN/proffastpylot> deactivate`
	`user@EM27/SUN/proffastpylot>`

**To run scripts in the PowerShell, you may need to set the execution policy to at least "RemoteSigned". Run:* `Set-Execution Policy RemoteSigned -Scope CurrentUser`. *See* [15] *for more information.*

Install the PROFFASTpylot repository

What to do	What should happen

In the **activated virtual environment**, run:
Important!
`pip install --editable .`
	`(prf_venv) user@EM27/SUN/proffastpylot> pip install --editable .`
	... installing process will take around one minute ...
	`Successfully installed PROFFASTpylot-1.2 PyYAML-6.0.1 certifi-2023.7.22 cffi-1.16.0 charset-normalizer-3.3.0 colorama-0.4.6 fortranformat-2.0.0 h3-3.7.6 idna-3.4 numpy-1.26.0 pandas-2.1.1 pycparser-2.21 python-dateutil-2.8.2 pytz-2023.3.post1 requests-2.31.0 six-1.16.0 timezonefinder-6.2.0 tqdm-4.66.1 tzdata-2023.3 urllib3-2.0.6 wheel-0.41.2`
	Windows PowerShell

This step will also install all required python packages.

Final folder structure

```
proffastpylot
├── docs
├── example
│   ├── input_sodankyla_example.yml
│   ├── log_type_pressure.yml
│   ├── run.py
│   └── ...
├── prf
│   ├── docs
│   ├── inp_fast
│   ├── inp_fwd
│   ├── preprocess
│   ├── source
│   ├── out_fast
│   ├── wrk_fast
│   └── ...
├── prf_venv
├── prfpylot
├── ...
└── setup.py
```

4. Test PROFFAST using the example dataset

After successfully setting up the software environment, an `example` folder is included in the `proffastpylot` directory. In this section, it is shown how an example dataset can be downloaded and then processed to generate a first test result that is of interest for most users.

Download and run the example dataset

What to do	What should happen
Navigate to the `proffastpylot` directory. **Activate the virtual environment** as shown on page 5. **Navigate** to the `example` folder. Finally, **run** the `run.py` script **with python**. `python run.py` **Enter "yes"** to download the example dataset.	See the console output below.

```
user@EM27/SUN/proffast_path> cd .\proffastpylot\
user@EM27/SUN/proffastpylot> .\prf_venv\Scripts\Activate.ps1
(prf_venv) user@EM27/SUN/proffastpylot> cd .\example\
(prf_venv) user@EM27/SUN/example> python .\run.py
Example data where not found on disk.
Do you like to download them? This will download 104 MB of dat
a to your disk.
Enter 'yes' to download the data or 'no' to abort:
yes
 98%|
```

… The retrieval will take a few minutes to run …

```
2023-10-12 10:45:06,013, INFO: Removing temporary files ...
2023-10-12 10:45:06,085, INFO: Done.

(prf_venv) user@EM27/SUN/example>
```

Windows PowerShell

Depending on the operating system, the retrieved gas concentrations can differ slightly. Please follow the instructions given in [19] → ***"6. Test the installation by running an example dataset". The*** `Reference_Output_Example_Sodankyla.csv` ***is only available in joss branch.***

Example folder structure

```
proffastpylot
└── example
    ├── analysis
    │   └── Sodankyla_SN039
    │       ├── 170608
    │       │   ├── cal
    │       │   ├── pT
    │       │   └── VMR_dim
    │       └── 170609
    │           └── ...
    ├── input_data
    │   ├── interferograms_sodankyla
    │   ├── map_sodankyla
    │   ├── pressure_sodankyla
    │   └── coords.csv
    ├── results
    │   └── Sodankyla_SN039_170608-170609
    │       ├── input_files
    │       │   └── ...
    │       ├── logfiles
    │       │   └── ...
    │       ├── comb_invparms_Sodankyla_SN039_170608-170609.csv
    │       ├── input_sodankyla_example
    │       └── ...
    ├── input_sodankyla_example.yml
    ├── log_type_pressure.yml
    └── run.py
```

For most users, the file containing the "combined inverted parameters" will be of main interest. **The meaning of the contained quantities is described in Appendix C on page 16.**

5. Prepare a retrieval with your own dataset

The example dataset used in the section above provides a first impression of the folder structure of a PROFFAST retrieval. In this section, we learn step by step, how to adapt another dataset to the required input data structure. The recommended folder structure is as follows (PROFFASTpylot documentation [6] → Folder Structure).

```
E:
└── Your_EM27_RawData
    ├── interferograms
    ├── maps
    └── pressure
```

Measured data (spectra + pressure) + Map files. May be stored on an external storage.

```
D:
└── Your_PROFFAST_OutputData
    ├── analysis
    └── results
        └── Site_SNXXX_StartDate-EndDate
            ├── logfiles
            ├── ...
            └── comb_invparms_Site_SNXXX_StartDate-EndDate.csv
```

PROFFAST output data. comb_invparms = "Combined inverted parameters" including trace gas concentrations, total column amounts, pressure, etc.

```
D:
└── Your_Retrieval
    ├── input_file.yml
    ├── coords.csv
    ├── log_type_pressure.yml
    └── run.py
```

Retrieval inputs: run script, pressure data specification, coordinates, input file.

```
D:
└── proffastpylot
    ├── docs
    ├── prf
    ├── prf_venv
    └── prfpylot
```

Installation shown above. Don't need to be changed.

***But:** 4 monthly updates (p. 3) recommended.*

For a more complex retrieval with many measurement days and sites, see Appendix A on page 14.

Interferograms

These are recorded by an EM27/SUN in combination with the OPUS software. The term "SNXXX" showing up in the folder structure refers to the serial number of the used EM27/SUN instrument. "YYY" in the file name is a number counting the sequential measurements.

Generic folder structure	Exemplary file list

└── Your_EM27_RawData
    └── interferograms
        ├── SNXXX
        │   ├── Date1
        │   │   ├── Date1SN.YYY
        │   │   └── (more interferograms)
        │   └── (more days)
        └── (more instruments)
	170608SN.350
	170608SN.400
	170608SN.450
	170608SN.500
	170608SN.550
	170608SN.600

Map files

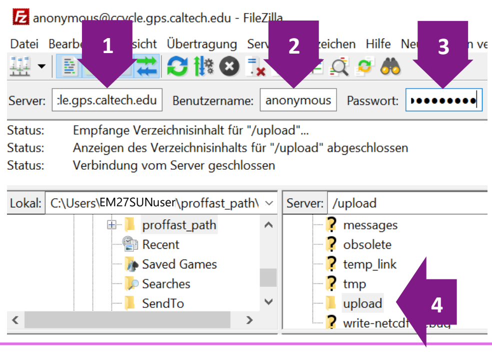
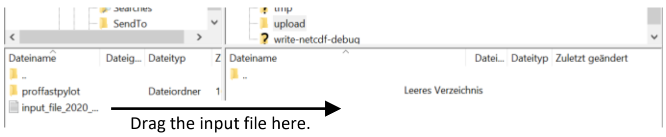
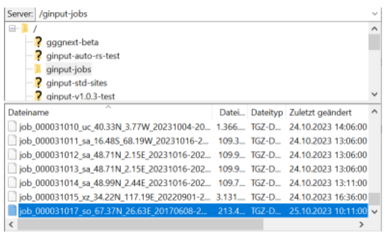
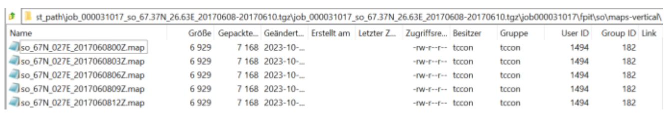
The map files contain modeled atmospheric information at the measurement site at the measurement time, e.g., *a-priori VMR (volume mixing ratio) profiles. They are automatically generated on a Caltech server.*

[The measured absorption spectra are fitted to those of the a-priori profiles. These profiles are generated model-based [9]*. More information can be found in literature, e.g., in* [1]*. The map files can be the same for different measurement sites, if the distance between them is less than ≈ 20 km.]*

Generic folder structure	Exemplary file list

└── Your_EM27_RawData
    └── maps
        ├── site-abbrev_latN_longE_DateTime.map
        └── (more map files)
	so_67N_027E_2017060800Z
	so_67N_027E_2017060803Z
	so_67N_027E_2017060806Z
	so_67N_027E_2017060809Z
	so_67N_027E_2017060812Z
	so_67N_027E_2017060815Z

Request and download the map files

What to do	What should happen
You need FTP[1] access to the Caltech server. Map files are generated there. **If you don't already have access, follow the instructions** on the TCCON website [16].	*You will get access with the **username "anonymous"** and your **email address as password**.* Not necessarily the same email address!
Create text file: `input_file_2020*.txt`, containing information on your measurements. (* = *empty* or more characters. The only variable part is *.)	*input_file_2020_example - Editor Datei Bearbeiten Format Ansicht `so` Personal site abbreviation `20170608` Start date YYYYMMDD `20170610` End date + 1 Day `67.366` Latitude `26.630` Longitude `user@em27SUN.edu` Email to receive map files[2] This exact form is required, no extra spaces or characters allowed!
Connect via FTP[1] to the Caltech server `ccycle.gps.caltech.edu` (1). **Enter** username "anonymous" (2) and your password (3). **Press Enter**. **Navigate** to the `upload` directory (4).	
Drag the input file from your local folder to the remote `upload` directory.	 *(The dragged file will immediately disappear, and a successful transfer message will appear.)*
The generation of the map files takes some hours. **Navigate** to the `ginput-jobs` folder and check for `job_XXXXX_site_lon_lat_date.tgz` corresponding to your input file. *[**or** follow the instruction in the email received by "tccon Account".]*[2]	
Download the generated archive. **Open the archive** (e.g., with 7-Zip) and follow the path as shown to the right.	 Path: `\job_000031017_so_67.37N_26.63E_20170608-20170610.tgz\job000031017\fpit\so\maps-vertical\`

[1]*FTP = File Transfer Protocol. Here, the open source FileZilla client* [4] *is used to connect via FTP. It is available for all common operating systems. There are other ways to connect to the FTP server, such as a command line interface.*

[2]*Apparently receiving the tccon email does not work for some email addresses or servers. Downloading map files via FileZilla still works without the email.*

More detailed information on the Caltech website [18].

Pressure data

The pressure dataset contains the surface pressure at the EM27/SUN location at the time of measurement. It is measured by an external sensor at the measurement site.

[Surface pressure is critical to the retrieval process as it is determined by the air mass in the column above the instrument. The accuracy of the surface pressure must be within 0.2 mbar. Therefore, the pressure sensor must be calibrated to an absolute reference pressure. Finally, any difference in height between the EM27/SUN and the pressure sensor must be taken into account, as 0.2 mbar corresponds to a height difference of approximately 2 m. Information about pressure measurement strategies can be found in published material describing various COCCON field campaigns, e.g. [10].*]*

<table>
<tr><th>Generic folder structure</th><th>Exemplary file list (extract from example)</th></tr>
<tr><td><pre>└── Your_EM27_RawData
    └── pressure
        └── (pressure files as specified in pressure type file)</pre></td><td><pre>UTCdate____   UTCtime___   BaroTHB40
"08.06.2017" "00:10:00"   +9.9892e+02
"08.06.2017" "00:20:00"   +9.9902e+02
"08.06.2017" "00:30:00"   +9.9902e+02
"08.06.2017" "00:40:00"   +9.9912e+02
"08.06.2017" "00:50:00"   +9.9912e+02</pre></td></tr>
</table>

Pressure type file

The pressure type file provides PROFFAST the information about the **file format, time format and other specifications** of the pressure dataset you use for your retrieval.

[The pressure data is often measured by individual sensors, depending on the availability or practicability at the measurement location. Consequently, the data usually has different file formats, time formats, units and so on. Therefore, the pressure data needs to be specified in more detail to be treated by PROFFAST. For this purpose, a file "log_type_pressure.yml" must be adapted to the dataset used.]

Set up the pressure type file

What to do

Copy the `log_type_pressure.yml` from the `example` directory into your own retrieval directory.

Open the `log_type_pressure.yml` file with a text editor.

Follow the instructions given in the file and **adjust the example file to your own pressure dataset**.

Most **important parameters** are:

- "dataframe_parameters": "key" ↔ name of column, "fmt" ↔ format of quantity in column, "csv_kwargs: sep" ↔ separator between columns in pressure data file (\t = tab)
- "time_key" can be anything, must not be UTC time.
- "UTC offset of data"

For more information, see the PROFFASTpylot documentation [6] → *Pressure Input.*

Coordinates

The coordinates of the measurement locations must be given either in the input file or in an external file. The latter is recommended, especially when measuring at more than one location. They can be measured e.g., by a GPS sensor and manually written to a `.csv` file.

[The recommended accuracy is 0.001°, which is equivalent to 100 m.]

Generic folder structure	File structure with exemplary data
└── Your_Retrieval └── coords.csv	1 Site, Latitude, Longitude, Altitude_kmasl, Starttime 2 Sodankyla, 67.366, 26.630, 0.181, 2014-01-01 3 Thessaloniki, 40.634, 22.956, 0.067, 2021-10-04 4 Karlsruhe, 49.103, 8.436, 0.130, 2014-01-01

Input File

The input file here refers to the PROFFASTpylot input file, not to be confused with the one used to generate the map files. This file specifies the paths for all input and output files. Other specifications can be made in this file as well. Most of the entries can be left as defaults for the standard measurement.

Set up the input file

What to do

Copy the example `input_sodankyla_example.yml` into your own retrieval directory and **rename** it. `D:/Your_Retrieval` represents an exemplary general retrieval directory.

Open the new input file with a text editor.

Adjust the entries of the file names and paths according to your own folder structure. In the example below, the left column shows an exemplary ***input folder structure*** whereas the right column the corresponding entries in the ***input file***.

```
E:
└── Your_EM27_RawData
    ├── interferograms
    ├── maps
    └── pressure
D:
└── Your_PROFFAST_OutputData
    ├── analysis
    └── results
        └── Site_SNXXX_StartDate-EndDate
            ├── logfiles
            ├── ...
            └── comb_invparms_Site_SNXXX_StartDate-EndDate.csv
D:
└── Your_Retrieval
    ├── input_file.yml
    ├── coords.csv
    ├── log_type_pressure.yml
    └── run.py

D:
└── proffastpylot
    ├── docs
    ├── prf
    ├── prf_venv
    └── prfpylot
```

```
coord_file: coords.csv¹

interferogram_path: E:/Your_EM27_RawData/interferograms²
map_path: E:/Your_EM27_RawData/maps
pressure_path: E:/Your_EM27_RawData/pressure

pressure_type_file: log_type_pressure.yml

analysis_path: D:/Your_PROFFAST_OutputData/analysis
result_path: D:/Your_PROFFAST_OutputData/results
```

[1] No path specified here, because coords.csv is in same folder as run.py, as for pressure type file.

[2] Be aware that forward slashes (/) are used for paths throughout this manual. For some APIs (Application Programming Interfaces), backslashes (\) could be necessary.

For more information, see the PROFFASTpylot documentation [6] → Folder Structure.

Run Script

The final step is to adjust the `run.py` script that carries out the PROFFAST retrieval.

Set up the run script

What to do	What should happen
Copy the `run.py` from the example into your own retrieval directory and **rename** it. **Open** your new run script with a text editor. **Remove** the code that is only needed for the example retrieval: The comments at the top of the script and the download of the example dataset. **Enter** the name of your **input file**. **Set** the desired **number of parallel processes**[1].	*You should now have created a folder structure as shown on page 7.* Edited `run.py` script:

```
"""Ready-to use example to demonstrate the usage of
PROFFASTpylot.


To execute this file from .../proffastpylot/example as your
working directory.

The Sodankyla example data set will be downloaded if not
present.
All steps of the retrieval with PROFFAST will be executed
by Pylot.run() automatically.
"""

from prfpylot.download_example import ExampleDownloadHandler
from prfpylot.pylot import Pylot


# This statement needs to be executed in all run scripts to
prevent problems
# with the multiprocessing on windows
if __name__ == "__main__":

    # Check if example input data is already available on
    disk,
    # if not download it.
    # This is not needed for your personal PROFASTpylot run

ExampleDownloadHandler().check_and_download_example_data()

    # The following part can be adapted to your own
    retrieval
    input_file = "input_sodankyla_example.yml"
    input_file = "input_file.yml"
    MyPylot = Pylot(input_file, logginglevel="info")
    MyPylot.run(n_processes=2)
```

[1]*Process the data of different measurement days parallel on different processor cores. The recommended maximum n_processes is the number of processor cores of your system.*

For information on running PROFFAST partially (e.g. only preprocessing), read in the [PROFFASTpylot documentation](#) [6] → *Usage.*

6. Running your own retrieval

Now that you have prepared the dataset and input files, three more steps must be carried out to complete the retrieval:

Update PROFFASTpylot (p. 3) and make sure you have the newest version of PROFFAST (p.4).

Activate the virtual environment (p. 5).

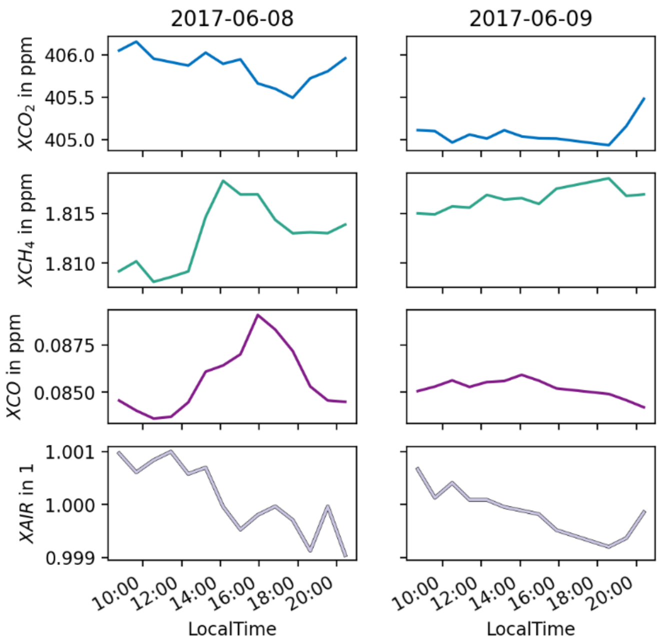
Run the run script (p. 11).

Wait for the results. Depending on the dataset and the processing power of your system, this will take at least several minutes. For datasets with many measurement days and minutely scans, it might take several hours. The results can be found in the `results` folder.

[One important scientific reason for updates are the ILS (instrumental line shape) parameters. The ILS parameters are individually determined for each EM27/SUN because every instrument differs slightly in interferometric alignment, optical aberrations etc. (read in [11]*). A list with the current ILS parameters is provided in the* `prfpylot` *directory.* ***These should be used for a COCCON compliant retrieval****. Information on the implementation of ILS parameters in PROFFAST can also be found in the PROFFASTpylot documentation* [6] *→ ILS Parameters.]*

7. Results

A graphical representation of the results is a convenient way for their interpretation.



For the exemplary dataset, the **column-averaged dry-air mole fractions *Xgas*** for the most important greenhouse gases are shown on the right side. The use of dry-air mole fractions has the advantage of eliminating concentration variations caused by variable ground pressure and the highly variable mixing ratio of water vapor (0 - 4%). Compared to total column amounts, using *Xgas* also eliminates systematic errors induced by surface pressure. **Careful** **calibration** [12] is required to obtain absolute trace gas concentrations (for more information, see, e.g., in [11]).
The quantity ***XAIR* can serve as a measure of system stability** with an ideal value of 1. If data derived from observations recorded at low to moderate airmass (Solar Zenith Angle (SZA) < 70°) and during stable weather conditions show deviations in *XAIR* of more than 1% from unity, this might indicate problems with the FTIR measurements or with the pressure recording and would require further investigation. More information can be found in [1] or other contributions within the COCCON framework.

Appendix

A A more sophisticated retrieval setup

The folder structure presented in [Prepare a retrieval with your own dataset?] (p. 7) needs to be extended if you have many measurement days and different measurement sites. In this case, the following adjustments provide an extended folder structure.

Set up an extended folder structure

What to do

Match the paths specified in the ***input files*** against the ***input data structure and the desired output structure***. *Here, a folder structure for an exemplary dataset with 2 instruments measuring at 3 measurement sites is shown.*

```
E:
└── Your_EM27_RawData
    ├── interferograms
    │   ├── SN037
    │   └── SN039
    ├── maps*
    └── pressure
        ├── Site1
        ├── Site2
        └── Site3
D:
└── Your_PROFFAST_OutputData
    ├── analysis
    └── results
        ├── Site1_SN039_StartDate1_EndDate1
        │   ├── logfiles
        │   ├── ...
        │   └── comb_invparms_Site1_SN039_StartDate1_EndDate1.csv
        ├── Site2_SN037_StartDate2_EndDate2
        │   ├── logfiles
        │   ├── ...
        │   └── comb_invparms_Site2_SN037_StartDate2_EndDate2.csv
        └── Site3_SN037_StartDate3_EndDate3
            ├── logfiles
            ├── ...
            └── comb_invparms_Site3_SN037_StartDate3_EndDate3.csv
D:
└── Your_Retrieval
    ├── run.py
    ├── pressure_type_files
    │   ├── log_type_pressure_Site1.yml
    │   ├── log_type_pressure_Site2.yml
    │   └── log_type_pressure_Site3.yml
    ├── coords.csv
    └── input_files
        ├── input_file_Site1.yml
        ├── input_file_Site2.yml
        └── input_file_Site3.yml
D:
└── proffastpylot
    ├── docs
    ├── prf
    ├── prf_venv
    └── prfpylot
```

Exemplary parameters for `input_file_Site1.yml`:

```
coord_file: coords.csv

interferogram_path: E:/Your_EM27_RawData/interferograms/SN039
map_path: E:/Your_EM27_RawData/maps
pressure_path: E:/Your_EM27_RawData/pressure/Site1

pressure_type_file: pressure_type_files/log_type_pressure_Site1.yml

analysis_path: D:/Your_PROFFAST_OutputData/analysis
result_path: D:/Your_PROFFAST_OutputData/results
```

Adjust the `run.py` script to **loop over all input files** ***or*** use **different run scripts for the different input files**. An exemplary code could be:

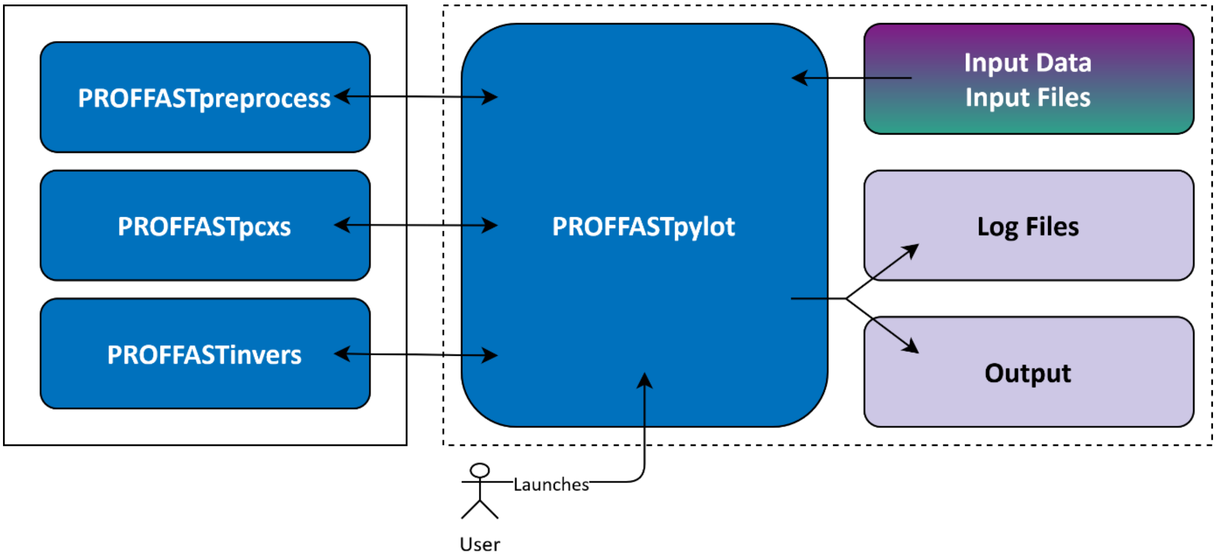
```
input_file_numbers = range(1,4)
for i in input_file_numbers:
    if __name__ == "__main__":

        input_file = (
            "input_files/input_file_Site" + str(i) + ".yml")
        MyPylot = Pylot(input_file, logginglevel="info")
        MyPylot.run(n_processes=2)
```

**The map files can be the same for different measurement sites, if the distance between them is less than approximately 20 km.*

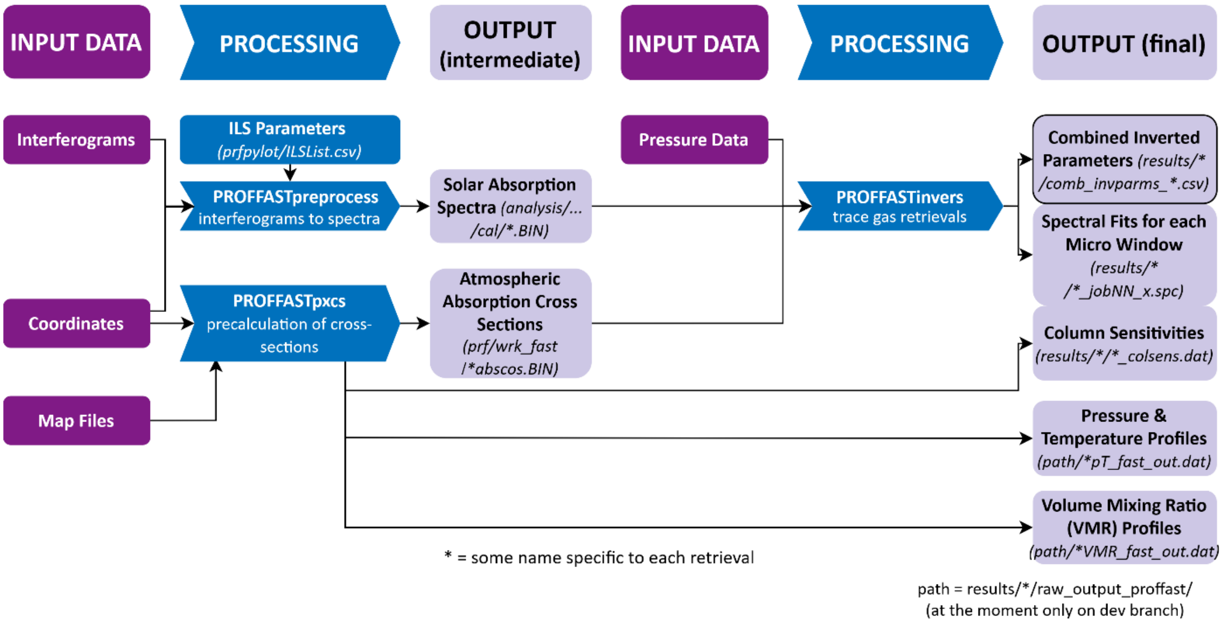
B Schematic representation of the PROFFAST algorithm

The schematic representations in the following shall mainly serve for the general understanding of the workflow of this software.

The first figure shows the schematic interaction between PROFFASTpylot, the other PROFFAST components and the retrieval data. For a basic trace gas retrieval with PROFFAST, the user only provides all needed input data and launches PROFFASTpylot. The latter reads the input data, passes it on to the corresponding processing scripts and finally writes the results to the desired folders.



The second figure shows a schematic of the trace gas analysis within PROFFAST. For simplicity, PROFFASTpylot is not included in this figure. Also, only the input parameters presented here in this manual are shown. However, the output data in this figure contains information more than the usual "combined inverted parameters" and may be of interest to advanced users.



More information on Column sensitivities [13].

C Interpretation of Results: Combined inverted parameters

In this section, the quantities contained in the `results/comb_invparms*.csv` file are specified. The main quantities related to trace gas concentrations are:

- Total column amount (TC) of molecules in $\frac{\mathrm{molec.}}{\mathrm{m}^2}$: $TC_{\mathrm{gas}} = \int_{z_0}^{\infty} \mathrm{d}z\, \rho_{\mathrm{N}}$ with the ground height z_0 and the number density of the trace gas ρ_{N} (in $\frac{\mathrm{molec.}}{\mathrm{m}^3}$);
- Column-averaged dry-air mole fraction (DMF) in ppm: $X_{\mathrm{gas}} = \frac{TC_{\mathrm{gas}}}{TC_{\mathrm{DryAir}}}$.

UTC	LocalTime	spectrum	JulianDate	UTtimeh	gndP	gndT	latdeg
Coordinated Universal Time	Time at measurement site	Spectrum measured in specific time interval	Count of days [14]	Decimal UTC hours	Surface pressure in millibars	Surface temperature in Kelvin	Latitude in degrees
londeg	**altim**	**appSZA**	**azimuth**	**XH2O**	**XAIR**	**XCO2**	**XCH4**
Longitude in degrees	Altitude in meters	Solar zenith angle in degrees	Solar azimuth angle (south=0°) in degrees	DMF of water vapor in ppm	DMF of dry air in 1 (ideally XAIR = 1)	DMF of CO_2 in ppm	DMF of CH_4 in ppm
XCO	**XCH4_S5P**	**H2O**	**O2**	**CO2**	**CH4**	**CO**	**CH4_S5P**
DMF of CO in ppm	DMF of CH_4 calculated like Sentinel-5P[1] in ppm	TC of H_2O in molecules/m²	TC of O_2 in molecules/m²	TC of CO_2 in molecules/m²	TC of CH_4 in molecules/m²	TC of CO in molecules/m²	TC of CH_4 calculated like Sentinel-5P[1] in molecules/m²

[1] CH_4 abundance that is calculated in the same spectral window (channel 2) as CO. This window is also used by the Sentinel-5P satellite.

Bibliography

[1] M. Frey, M. K. Sha, F. Hase, M. Kiel, T. Blumenstock, R. Harig, G. Surawicz, N. M. Deutscher, K. Shiomi, J. E. Franklin, H. Bösch, J. Chen, M. Grutter, H. Ohyama, Y. Sun, A. Butz, G. Mengistu Tsidu, D. Ene, D. Wunch, Z. Cao, O. Garcia, M. Ramonet, F. Vogel and J. Orphal, "Building the COllaborative Carbon Column Observing Network (COCCON): long-term stability and ensemble performance of the EM27/SUN Fourier transform spectrometer," *Atmospheric Measurement Techniques,* vol. 12, no. 3, pp. 1513-1530, 2019.

[2] KIT IMK-ASF, "IMK-ASF - About IMK-ASF - Research Groups - Ground-Based Remote Sensing - COCCON," [Online]. Available: https://www.imk-asf.kit.edu/english/COCCON.php.

[3] Git, "Git - Downloads," [Online]. Available: https://git-scm.com/downloads. [Accessed 23 11 2023].

[4] FilleZilla, "FileZilla - The free FTP solution," [Online]. Available: https://filezilla-project.org/. [Accessed 23 11 2023].

[5] Python, "Download Python," Python.org, [Online]. Available: https://www.python.org/downloads/.

[6] L. Feld, B. Herkommer and D. Dubravica, "PROFFASTpylot documentation," 06 09 2023. [Online]. Available: https://www.imk-asf.kit.edu/english/4261.php. [Accessed 23 11 2023].

[7] L. Feld, B. Herkommer and D. Dubravica, "docs · master · coccon-kit / PROFFASTpylot · GitLab," GitLab, 12 05 2023. [Online]. Available: https://gitlab.eudat.eu/coccon-kit/proffastpylot/-/tree/master/docs.

[8] KIT IMK-ASF, "IMK-ASF - About IMK-ASF - Research Groups - Ground-Based Remote Sensing - COCCON - Telephone Conferences," [Online]. Available: https://www.imk-asf.kit.edu/english/3934.php. [Accessed 12 12 2023].

[9] J. Laughner, S. Roche, M. Kiel, G. C. Toon, D. Wunch, B. C. Baier, S. Biraud, H. Chen, R. Kivi, T. Laemmel, K. McKain, P.-Y. Quéhé, C. Rousogenous, B. B. Stephens, K. Walker and P. O. Wennberg, "A new algorithm to generate a priori trace gas profiles for the GGG2020 retrieval algorithm," *Atmospheric Measurement Techniques,* vol. 16, no. 5, pp. 1121-1146, 2023.

[10] P. L. Schmid, "Quantification of Greenhouse Gas Emissions in Thessaloniki, Greece," KIT, Karlsruhe, 2023.

[11] C. Alberti, F. Hase, M. Frey, D. Dubravica, T. Blumenstock, A. Dehn, P. Castracane, G. Surawicz, R. Harig, B. C. Baier, C. Bès, J. Bi, H. Boesch, A. Buth, Z. Cai, J. Chen, S. M. Crowell, N. M. Deutscher, D. Ene, J. E. Franklin, O. García, D. Griffith, B. Grouiez, M. Grutter, A. Hamdouni, S. Houweling, N. Humpage, N. Jacobs, S. Jeong, L. Joly, N. B. Jonas, D. Jouglet, R. Kivi, R. Kleinschek, M. Lopez, D. J. Medeiros, I. Morino, N. Mostafavipak, A. Müller, H. Ohyama, P. I. Palmer, M. Pathakoti, D. F. Pollard, U. Raffalski, M. Ramonet, R. Ramsay, M. K. Sha, K. Shiomi, W. Simpson, W. Stremme, Y. Sun, H. Tanimoto, Y. Té, G. M. Tsidu, V. A. Velazco, F. Vogel, M. Watanabe, C. Wei, D. Wunch, M. Yamasoe, L. Zhang and J. Orphal, "Improved calibration procedures for the EM27/SUN spectrometers of the COllaborative Carbon Column Observing Network (COCCON)," *Atmospheric Measurement Techniques,* vol. 15, no. 8, pp. 2433-2463, 2022.

[12] KIT IMK-ASF, "2021-04-30_Instrument-Calibration.pdf," [Online]. Available: https://www.imk-asf.kit.edu/downloads/Coccon/2021-04-30_Instrument-Calibration.pdf.

[13] KIT IMK-ASF, " IMK-ASF - About IMK-ASF - Research Groups - Ground-Based Remote Sensing - COCCON - Column Sensitivities," [Online]. Available: https://www.imk-asf.kit.edu/english/3330.php . [Accessed 12 12 2023].

[14] Wikipedia, "Julian day," [Online]. Available: https://en.wikipedia.org/w/index.php?title=Julian_day&oldid=1180339802. [Accessed 23 11 2023].

[15] S. Wheeler, "about Execution Policies - PowerShell," [Online]. Available: https://learn.microsoft.com/en-us/powershell/module/microsoft.powershell.core/about/about_execution_policies?view=powershell-7.4.

[16] J. Laughner, "EM27ProffastSupport < Main < TCCON Wiki," Caltech, 06 02 2023. [Online]. Available: https://tccon-wiki.caltech.edu/Main/EM27ProffastSupport.

[17] Python, "Python documentation: venv — Creation of virtual environments," Python, [Online]. Available: https://docs.python.org/3/library/venv.html. [Accessed 23 11 2023].

[18] J. Laughner, "ObtainingGinputData < Main < TCCON Wiki," Caltech, 06 02 2023. [Online]. Available: https://tccon-wiki.caltech.edu/Main/ObtainingGinputData#Custom_locations_and_EM27s. [Accessed 28 11 2023].

[19] L. Feld, B. Herkommer and D. Dubravica, "docs/1-1_installation · joss · coccon-kit / PROFFASTpylot · GitLab," GitLab, 04 09 2023. [Online]. Available: https://gitlab.eudat.eu/coccon-kit/proffastpylot/-/blob/joss/docs/1-1_installation.md?ref_type=heads.