Fiducial Reference Measurements for Ground-Based DOAS Air-Quality Observations

ESA Contract No. 4000135355/21/I-DT-lr

Deliverable D8.4

TIP BrO stratospheric profiling

Date: Friday 15th May, 2026
Version: 1.0

Stratospheric BrO profiling TIP

Martina M. Friedrich, Caroline Fayt, Alexis Merlaud, Michel Van Roozendael

Friday 15th May, 2026

Acronyms

BASCOE Belgian Assimilation System for Chemical ObsErvations.

BIRA-IASB Belgian Institute for Space Aeronomy.

CCN contract change notice.

CTM Chemical transport Model.

DISORT DIscrete SOrdinate Radiative Transfer.

DSCD Differential Slant Column Density.

FRM4DOAS Fiducial Reference Measurements for Ground-Based DOAS Air-Quality Observations.

GDFB Gridded DataFile by BIRA.

GRIB GRIdded Binary or General Regularly-distributed Information in Binary form.

L1 level 1 (i.e. corrected for dark current, offset, as well as possibly straylight) spectra.

L2 level 2 FRM4DOAS module specific netCDF4 output file containing both retrieved quantities as well as used dSCDs for traceability..

LUT Look-up table.

netCDF Network Common Data Form.

NRT near-real-time.

OEM Optimal Estimation Method.

SCD slant column density.

SZA Solar zenith angle.

TIP Technical implementation plan.

WP Work package.

Contents

1 Introduction

This document is the Technical implementation plan (TIP) for the stratospheric BrO profiling algorithm, Work package (WP)-8 of Fiducial Reference Measurements for Ground-Based DOAS Air-Quality Observations (FRM4DOAS)-2.0, contract change notice (CCN)-02. It describes the necessary steps to integrate the prototype algorithm described in Deliverable:D8.2v.2 in the FRM4DOAS processing chain. While the stratospheric BrO profiling is conceptually similar to the stratospheric NO_2 profiling already implemented in the FRM4DOAS processing, it is different in that it does not use pre-compiled Look-up table (LUT). As such, it means performing the Chemical transport Model (CTM) simulations with Belgian Assimilation System for Chemical ObsErvations (BASCOE) on a daily basis. This does not only effect the computational costs, but also the work-flow.

In Sect. 2 we outline the conceptual steps of the workflow to be carried out before the profile inversion. In Sect. 3 we describe the implementation steps needed for the algorithm described in Deliverable:D8.2v.2. In Sect. 4 we desribe the requirements regarding libraries for BASCOE, DIscrete SOrdinate Radiative Transfer (DISORT) as well as the python-based inversion algorithm. We also provide rough time estimates for the processing. Finally, in Sect. 5 we estimate the required work to implement the different steps. We separate tasks into configuration `C`, implementation `I`, wrapper `W` and single setup and testing `S` tasks.

2 Tasks to perform prior to inversion

In order to perform the retrievals, the following steps have to be performed prior to the profile inversion.

1. On a daily basis: Analyse spectra with `QDOAS`, implementing the settings developed in FRM4DOAS-2.0:D8.1.
2. On a daily basis: Perform global BASCOE simulations with a time step of $\Delta t = 10$ min and online photolysis rate calculations and extract profiles at the station locations.
3. Once prior to the retrieval: Construct for each station a regulatization matrices, for Optimal Estimation Method (OEM) this is S_a, according to the receipe used in Deliverable:D8.2v.2

2.1 SCD fitting

The slant column density (SCD) fitting will be performed with `qdoas`. `qdoas` is naturally already a part of the FRM4DOAS processing chain. The input to `qdoas` are level 1 (i.e. corrected for dark current, offset, as well as possibly straylight) spectra (L1) spectrum files that are already submitted to the processing system. Therefore, no new wrappers and no new configuration system need to be set up. What is needed are instrument specific configuration-files. The recommended fitting window and fitting parameters have been presented in FRM4DOAS-2.0:D8.1. The task to create `QDOAS` station specific configuration files is called `C1`.

2.2 Perform Bascoe simulations

In the current setup, BASCOE is driven by 6-hourly ERA-5 re-analysis files. For the tests performed for Deliverable:D8.2v.2, we use the ERA-5 data re-gridded on a regular lat-lon grid (2.5° x 2°). The re-gridding is performed via an in-house program (gdfb_mflux) further developing a program described in Segers et al. (2002) (mflux). The input to this are the GRIdded Binary or General Regularly-distributed Information in Binary form (GRIB) files of the quantities expressed as spherical harmonics.

The GRIB files are currently automatically downloaded daily to BIRA servers. However, there is a delay (they are re-analysis files) of 6 days. In practice this means that the operational product, without further changes, can run with a 6 day delay. In order to do so, as mentioned above, the gridding step has to be peformed before BASCOE can be run. While the downloading is in place, the remaining steps need to be implemented: a wrapper to start the re-gridding program (once daily) has to be created, we will refer to this as `W1`. Further, a wrapper to run 1 day of BASCOE initialized with the previous full output needs to be created, `W2`.

However, if the delay of 6 days is not acceptable, then a different approach need to be taken, possibly using meteorological near-real-time (NRT) products. This would however mean a significant amount of work since programs to convert the output to the specific Gridded DataFile by BIRA (GDFB) expected by BASCOE have to be written. We call this task `I1`.

2.3 Creation of S_a matrix for each station

For the examples presented in Deliverable:D8.2v.2, the variance at each altitude above the three stations at 80°, 85° and 90° Solar zenith angle (SZA) during both sunset and sunrise was derived from 2 (for Harestua) or 3 (for Kiruna and Lauder) years only. However, for an operational implementation, the variance as a function of height should rather be estimated from 10 years minimum, resulting in roughly 300 values for each monthly variance estimate at a specific SZA at sunrise and sunset, respectively. This task, `S1`, has to be performed only once per station. Currently, this is performed manually for each station.

3 Tasks for the implementation of the Inversion

In order to perform the inversion, the output from `qdoas` needs to be read in. Information of where to find which variable can be stored in task- and possibly station specific configuration files, depending on whether group names are likely to change from station to station. Additionally, locations of additional input, as the S_a, are needed. Depending on the instrument, different SZAs retrieval ranges might be specified. Hence, station specific configuration files need to be created, `C2` and a system to process these `I2`. The specific reading routine to ingest FRM4DOAS formatted Network Common Data Form (netCDF)4 `qdoas` files need to be written, task `I3`.

The inversion algorithm is implemented in python and can mostly be used as is, however severe cleaning needs to be performed. Since the idea to perform the retrieval at several (at least 2) SZA as an additional consistence check has to be implemented, a restructuring of the code is needed: It is currently written to perform the retrieval at a single SZA and the retrievals at different retrieval SZA are performed by repeated calls to `uvspec`. However, the only detail that changes are the provided NO_2 and O_3 profiles since, as opposed to the BrO profile, they are treated as 1D (depending on height) variables. However, since the interesting output for us is the wavelength specific SCD and weighting function K, this should not depend on the ozone and NO_2 profiles. The restructuring and cleaning of the algorithm is named `I4`.

Regarding the output file, some work has to be invested in order to arrive at the level of level 2 FRM4DOAS module specific netCDF4 output file containing both retrieved quantities as well as used dSCDs for traceability. (L2) files for other FRM4DOAS products. Currently, the original input (e.g. SZAs, Differential Slant Column Densities (DSCDs)) is stored in a sub-optimal way together with the retrieved profile and the retrieval settings and *a priori* information on the main level of the file. A module configuration file has to be created defining the output file structure, `C3`. A proper write routine has to be implemented to convert the output into the structure dictated by the configuration file, `I5`. Finally, in order to run the inversion task, a wrapper has to be created, `W3`.

Finally, tests need to be performed to ensure that the data is read and processed as intended, units are interpreted correctly, interpolations are performed correctly. This should not be confused with the verification tasks assigned to CCN-03. We name this task `S2`.

4 Requirements

4.1 uvspec and DISORT

Currently, we use the classic ifort (i.e. not ifx) compiler for compilation. Currently, `uvspec` is compiled with the following system libraries:

- linux-vdso.so.1
- libm.so.6
- libc.so.6
- libgcc_s.so.1

4.2 BASCOE

Bascoe is compiled with ifort on . The following system libraries are needed:

- linux-vdso.so.1
- libnetcdff.so.7
- libnetcdf.so.19
- libmkl_intel_lp64.so
- libmkl_intel_thread.so
- libmkl_core.so
- libiomp5.so
- libm.so.6
- libc.so.6
- libgcc_s.so.1
- libstdc++.so.6
- libtirpc.so.3

- libhdf5_hl.so.100
- libhdf5.so.103
- libsz.so.2
- libz.so.1
- libcurl.so.4
- libjpeg.so.62
- libdl.so.2
- libgfortran.so.5
- libquadmath.so.0
- libpthread.so.0
- libgssapi_krb5.so.2
- libnghttp2.so.14
- libidn2.so.0
- libssh.so.4
- libpsl.so.5
- libssl.so.3
- libcrypto.so.3
- libldap_r-2.4.so.2
- liblber-2.4.so.2
- libzstd.so.1
- libbrotlidec.so.1
- libkrb5.so.3
- libk5crypto.so.3
- libcom_err.so.2
- libkrb5support.so.0
- libunistring.so.2
- libjitterentropy.so.3
- libsasl2.so.3
- libbrotlicommon.so.1
- libkeyutils.so.1
- libresolv.so.2
- libselinux.so.1
- libpcre2-8.so.0

4.3 Python packages for the inversion

The inversion code is currently run using python version 3.12.3 and it uses the following packages:

package	version	version in FRM4DOAS
cftime	1.6.4	1.6.4
netCDF4	1.6.5	1.6.5
numpy	1.26.4	1.26.4
pandas	2.2.2	2.2.2
scipy	1.17.0	1.13.1
xarray	2025.1.2	2025.1.2

Only the version of scipy differs; however, especially the scipy.linalg was update significantly during the last 3 mayor releases, hence it is highly recommended to upgrade the scipy package for the implementation:
https://docs.scipy.org/doc/scipy-1.17.0/release/1.17.0-notes.html#scipy-linalg-improvements
https://docs.scipy.org/doc/scipy-1.16.0/release/1.16.0-notes.html#scipy-linalg-improvements
https://docs.scipy.org/doc/scipy-1.15.0/release/1.15.0-notes.html#scipy-linalg-improvements

4.4 Computation time estimates

- BASCOE: As previously stated in Deliverable:D8.2v.2, running 1 day of global BASCOE with a time step of $\Delta t = 10$ min takes approximately 20 min on Belgian Institute for Space Aeronomy (BIRA-IASB) compute servers.
- `QDOAS`: While this will not be a new task, it might extend the processing time if 1 (or more) additional fitting windows are included. The upper limit is of the order of 1 min/ day/ station.
- Profile inversion: The profile inversion takes about 3 min per day (i.e. sunrise+sunset) per station on a single processor.

5 Estimated implementation costs

Task class	Tasks	Work load / PM
C	configuration	1
	C1: QDOAS configuration	
	C2: input configuration	
	C3: output configuration	
W	wrapper	1
	W1: gridding wrapper	
	W2: BASCOE wrapper	
	W3: inversion wrapper	
I	implementation	2.5 (without I1)
	(I1: changing to NRT meteorology for BASCOE	2)
	I2: configuration processing	
	I3: preprocessing	
	I4: cleaning/ re-organizing inversion	
	I5: output routine	
S	single setup and test	1
	S1: Create S_a matrices for each station	
	S1: Testing the system (without changes due to I1)	

Acknowledgments

Simon Chabrillat and Quentin Errera are the main developers of BASCOE. We are thankful for the permission of using BASCOE in the framework of FRM4DOAS and for the time and effort they put into helping us getting acquainted with the use. Further, we thank François Hendrick, who developed many of the prototype retrieval algorithms that this current work builds upon, for making his extensive code base available to us and for giving us a rough road map of the tasks.

References

FRM4DOAS-2.0 (2026). *ATBD v2*. Deliverable D8.2v.2. FRM4DOAS-2.0.

FRM4DOAS-2.0:D8.1 (2025). *Technical note on BrO Slant Column Fitting Optimisation*. FRM4DOAS-2.0 D8.1.

Segers, Arjo, Peter van Velthoven, Bram Bregman, and Maarten Krol (2002). "On the computation of mass fluxes for Eulerian transport models from spectral meteorological fields". In: *International Conference on Computational Science*. Springer, pp. 767–776.