SEDOO-AERIS

REPROBUS - Reactive Processes Ruling the Ozone Budget in the Stratosphere

Technical Document & User Manual

	Name	Organisation	Date	Visa
Written by:	Daria Malik	Magellium		
Checked by:	Vanessa Pedinotti	Magellium		
Approved by:	Béatrice Berthelot	Magellium		

Document reference:	
Edition.Revision:	1.0
Date Issued:	06/05/2024
Customer:	
Ref. Market, consultation:	

Distribution List

	Name	Organisation	No. copies
Sent to :	D.Boulanger P. Henry	OMP OMP	1 1
Internal copy :	V. Pedinotti B.Berthelot	Magellium Magellium	1 1

Document evolution sheet

Ed.	Rev.	Date	Purpose evolution	Comments
1	0	06/05/2024	Creation of document	

Dissemination level

PU	Public	x
PP	Restricted to other programme participants	
RE	Restricted to a group specified by the consortium	
CO	Confidential, only for members of the consortium	

Contents

1. REPROBUS Overview

REPROBUS (Reactive Processes Ruling the Ozone Budget in the Stratosphere) is a 3D chemistry-transport model calculating the temporal evolution of 55 chemical species via 147 reactions chemicals (Lefèvre et al, 1998) (Fig. 1). Heterogeneous reactions are also taken into account. Liquid PSCs (Polar Stratospheric Clouds) are described using the model of Carslaw et al (1995). Wind and temperature fields are imposed every 3 hours and come from ECMWF analyses. The time step used for the chemical calculations is 15 minutes.

REPROBUS is a Fortran based numeric model. For the easier installation, maintenance and portability, REPROBUS executables and dependencies have been wrapped up into a Singularity container.

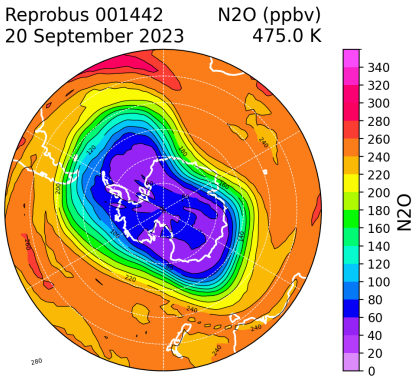

Figure 1 : N2O concentration in ppbv above the South Pole estimated by the REPROBUS model for September 20th, 2023

1.1. REPROBUS architecture

The processing chain is based on a main program called `reprobus_l137_YYYYMMDD.f`. Every year in July, a new version of the program is updated in order to integrate the forcings of the current year of the species NOy, Cly, Bry below the lowest level of chemistry (nivbas). The date of integration of these values allows the update of the program version: `YYYYMMDD` in the program name. Currently the version of the program is that of `20230723`. It is the project manager who ensures the integration of the forcing data for the current year.

The program takes as input ECMWF meteorological variables and the parameters files provided with the source code of the REPROBUS (Fig. 2); these parameters are not to be changed by the user.

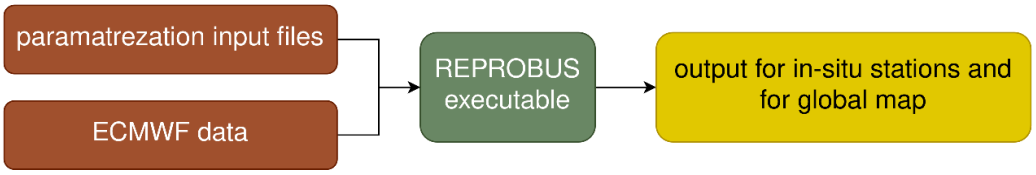

Figure 2 : scheme of the REPROBUS simulation input/output

1.2 REPROBUS input files

As the input script and txt files are immutable, the only user-dependent input is the meteorological data. The section below describes in detail immutable input files and ECMWF data that must be used for the REPROBUS simulation.

1.2.1 Parametrization files

The parametrization files are used by the main `reprobus_l137_YYYYMMDD.f` script and are mandatory for the compilation process. These files are immutable and the user does not need to modify them. The parametrization files are :

- Fortran routines `jno.f90` and `altitude.f90`
  - `jno.f90` allows the parameterization of the photodissociation of NO according to Minschwaner and Siskind, a new calculation of nitric oxide photolysis in the stratosphere, mesosphere, and low thermosphere, j. geophys. res. 98, 20401-20412, 1993
  - `altitude.f90` allows the calculation of geometric altitude in km according to http://mtp.jpl.nasa.gov/notes/altitude/altitude.html
- txt immutable files `h2s04.txt`, `mopitt_corrected.txt`, `jstrato.txt`, `relief.txt` and `ecmwf_137_levels.txt`
  - `h2so4.txt` is for H2SO4 forcing
  - `mopitt_corrected.txt` if for CO forcing below the lowest level of chemistry based on the mopitt measurements (according to J.L. Attié, LA),
  - `jstrato.txt` contains parameters for the calculation of photodissociation coefficients
  - `relief.txt` is for reading of the relief in km
  - `ecmwf_137_levels.txt` contains coefficients of transition from ECMWF model levels to pressure levels

Another input file that is sometimes needed is the `qinit2d` file. It is the initialization file written by the program author; it contains values needed by the program when it needs to be initialized on day 1 of the simulation. Along with this file the ozone meteorological values also must be extracted and provided to the program. The initialization commonly takes place on December 1st for the arctic winter, and on May 1st for the Antarctic winter. The `qinit2d` file must be maintained and updated when necessary by the program providers.

1.2.2 Meteorological data

The data used for the REPROBUS simulations are extracted from the ECMWF server. The mandatory meteorological variables needed for the modelization are:

- U component of wind (U)[1]
- V component of wind (V)
- temperature (t)
- specific humidity (q)
- vertical velocity (w)

The data is extracted on the 137 model levels of the ECMWF. The analysis data is extracted for the hours 0,6,12, and 18 of the day; the 3-hour intervals in-between the analysis data are filled with the forecast data.

The data is extracted in a GRIB format, but must undergo post-processing in order to be read correctly by the REPROBUS tool. The data is then organized by date with the `ecmwf_yyyymmdd` naming convention.

After the post-processing, the data can be divided by records as follows :

Hour	00h										03h										...	21h									
Model level	1	1	1	1	1	1	2	2	...	137	1	1	1	1	1	1	2	2	...	137	...	1	1	1	1	1	1	2	2	...	137
Variable	lnsp	t	w	q	u	v	t	w		v	lnsp	t	w	q	u	v	t	w		v		lnsp	t	w	q	u	v	t	w		v

Figure 3 : record organization in a ecmwf_yyyymmdd file after the GRIB post-processing

In this scheme, the `lnsp` variable for the hour 00h is the first record, the temperature (`t`) variable on model level 1 for 00h is the second record, vertical velocity (`w`) on model level 1 for 00h is the third record and so on, until the last record which is the V component of wind (`v`) on the level 137 for the 21h time. Each of these records follows a bytes order with the big-endian convention schematized on Fig. 4.

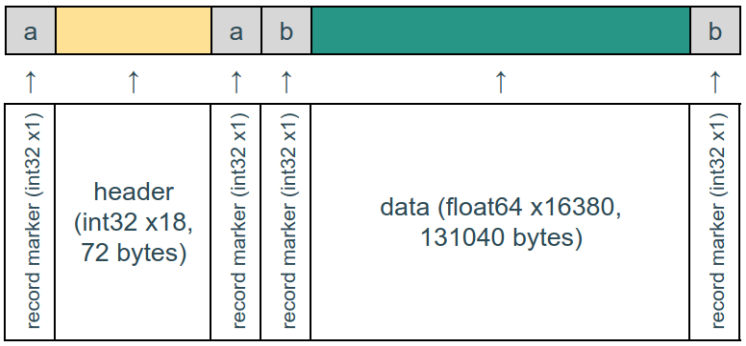

Figure 4 : bytes order of a record in the ecmwf_yyyymmdd file after the post-processing

The header contains such information as variable ID number from the ECMWF parameter database, date and hour of the data or model level number. The data is

[1] the short name of the field in the parentheses corresponds to the short name of this field in the ECMWF MARS database

then written sequentially with the longitude axis index changing fastest. The size of the data must be 180x91 pixels as these values are hard-coded in the source Fortran scripts of the program; hence there are 16380 data values on the latitude-longitude grid in the meteorological data file.

Same structure and process is applied to the extraction of the ozone data needed for the program initialization.

The process of the data extraction and corresponding scripts will be detailed later in this document.

1.3 REPROBUS output files

REPROBUS output files are divided into 3 categories : station files, profile files and history files.

1.3.1 Station files

Station files contain integrated column data for O3 and passive O3 in Dobson units, and integrated column data for NO2 and OClO in mol.cm-2 for 40 Arctic, Antarctic or mid-latitude stations. These files are named `stations_YYYYMMDDHH_V`, where :

- `YYYYMMDD` is the integration date
- `HH`=12pm
- `V` is a model version

These files are written in the ASCII format and the data is structured into columns (Fig. 5).

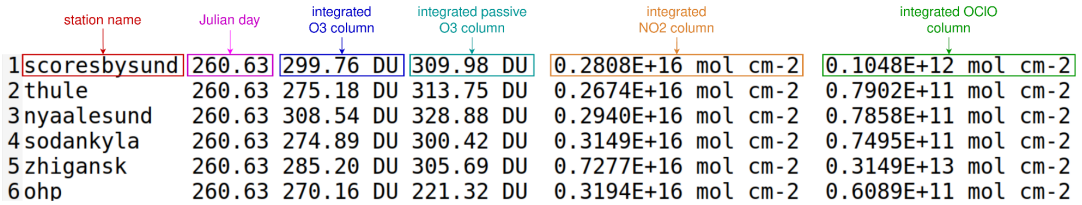

Figure 5 : example content of the file stations_2023091912_001442

1.3.2 Profile files

Profile files contain profiles of all the molecules calculated for 12 stations with 3-hour timestep on 137 ECMWF model levels. These files are named `reprobus_S_YYYYMMDDHH_V`, where :

- `YYYYMMDD` is the integration date
- `HH`=12pm
- `S` is the name of the station
- `V` is the model version

These files are written in the ASCII format and the data is structured into columns with a header in the beginning of the file (Fig. 6). Each variable is calculated on the 137 ECMWF model levels.

```
 1 RUN 001442
 2
 3 REPROBUS JULIAN DAY = 260.6250
 418   SEPTEMBER  2023  15 0UT
 5
 6 YAKUTSK
 7
 8 LONGITUDE =  129.620 LATITUDE =   62.030
 9 ILON      =       66 ILAT     =       15
10
11 SOLAR ZENITH ANGLE =  116.00
12
13  ALTITUDE    PRESSURE    TEMPERATURE THETA       DENSITY     N2O
14
15  0.7834E+02  0.1000E-01  0.1916E+03  0.5139E+04  0.3783E+15  0.6505E-10
16  0.7287E+02  0.2551E-01  0.1988E+03  0.4081E+04  0.9301E+15  0.1009E-09
17  0.7033E+02  0.3884E-01  0.2045E+03  0.3722E+04  0.1377E+16  0.1667E-09
18  0.6790E+02  0.5747E-01  0.2114E+03  0.3442E+04  0.1970E+16  0.2660E-09
                                . . .
151  0.2052E+00  0.9875E+03  0.2739E+03  0.2749E+03  0.2613E+20  0.3357E-06
152
153  PRESSURE    CH4         H2O         NOy         HNO3        N2O5
154
155  0.1000E-01  0.1660E-06  0.5267E-05  0.1802E-09  0.1136E-17  0.2282E-23
156  0.2551E-01  0.1698E-06  0.6299E-05  0.3558E-09  0.1440E-16  0.4819E-14
157  0.3884E-01  0.1780E-06  0.7096E-05  0.6011E-09  0.3891E-16  0.1998E-13
158  0.5747E-01  0.1907E-06  0.7103E-05  0.7474E-09  0.7302E-16  0.4701E-13
159  0.8287E-01  0.2087E-06  0.7205E-05  0.1132E-08  0.1864E-15  0.1258E-12
                                . . .
291  0.9875E+03  0.1919E-05  0.5754E-02  0.3000E-09  0.1939E-09  0.9347E-12
292
293  PRESSURE    Cly         Ox          CO          OClO        Passive Ox
294
295  0.1000E-01  0.3226E-08  0.7751E-05  0.6245E-06  0.2349E-12  0.3861E-05
296  0.2551E-01  0.3225E-08  0.3038E-05  0.3222E-06  0.4319E-12  0.3953E-05
297  0.3884E-01  0.3224E-08  0.2338E-05  0.9692E-07  0.5759E-12  0.4094E-05
298  0.5747E-01  0.3223E-08  0.1978E-05  0.9227E-07  0.7410E-12  0.4149E-05
```

Figure 6 : example content of the file reprobus_yakutsk_2023091912_001442

1.3.3 History files

History files contain model states at the end of an integration for a given day and constitute the calculation resumption file for the following day calculations. These files are named `MODEL_history_YYYYMMDDHH_V`, where :

- `YYYYMMDD` is the integration date
- `HH`=12pm
- `V` is a model version

Below is the visual representation of the bytes organization inside the file (Fig. 7).

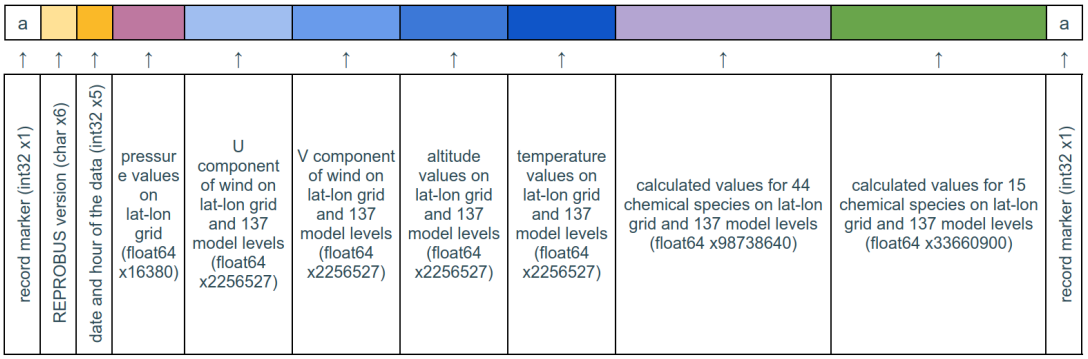

Figure 7 : bytes order of a MODEL_history_YYYYMMDDHH_V file

The lat-lon grid has the size of 180x91 pixels, hence there are 180 x 91=16380 values for the pressure variable which is on the simple lat-lon grid. When the variable has also a third dimension (137 model levels), there are accordingly 16380 x 137 = 2256527 values. For the chemical species variables, the 2256527 value is multiplied either by 44 species for the first variable (violet block on Fig. 7) and by 15 species for the second variable (green block on Fig. 7).

The chemical species in the two last variables are ordered as follows:

- 44 chemical species block : `N2O, CH4, H2O, NOy, HNO3, N2O5, Cly, Ox, CO, OClO, Passive Ox, H2SO4, HCl, ClONO2, HOCl, Cl2, H2O2, ClNO2, HBr, BrONO2, NOx, HNO4, ClOx, BrOx, Cl2O2, HOBr, BrCl, CH2O, CH3O2, CH3O2H, CFC-11, CFC-12, CFC-113, CCl4, CH3CCl3*, CH3Cl, HCFC-22*, CH3Br, H-1211*, H-1301, Bry, CH2Br2*, HNO3 GAS`
- 15 chemical species block : `O(1D), OH, Cl, O(3P), O3, HO2, NO2, NO, Br, N, ClO, BrO, NO3, H, CH3`

2. REPROBUS simulation for user

2.1 REPROBUS Singularity container

REPROBUS Fortran source code is containerized into a Singularity container, which allows any user to run it without previous libraries installation and Fortran configurations (except the installation of the Singularity application itself).

The file `reprobus-ifort-container.def` and `reprobus-nvidia-container.def` are Singularity Definition Files. A Singularity Definition File is a set of blueprints explaining how to build a custom container. It includes specifics about the base OS to build or the base container to start from, software to install, environment variables to set at runtime, files to add from the host system, and container metadata. The command which allows the build of a `.sif` Singularity image is:

```
$ sudo singularity build reprobus-ifort.sif reprobus-ifort-container.def
```

or

```
$ sudo singularity build reprobus-nvidia.sif reprobus-nvidia-container.def
```

The difference between the two definition files is that ifort container will use Intel Fortran compiler which runs on CPUs, while the nvidia container is designed to work on the GPUs if the host machine has one. The choice between the two containers must be made by the user based on the available hardware.

⚠️ One must have sudo rights on the system where the container is built. If it is not possible, the option `--fakeroot` can be used in order to build a container without sudo rights. In case of a multi-user server or machine, if the `--fakeroot` option throws an error, the one should contact the administrator of the machine to add one's user to white list. Once the container is built, the image can be run anywhere and without sudo rights.

2.2 REPROBUS User interface

The user interface for the easier simulation launches is a bash script `reprobus-simulation.sh` which requires a configuration file from the user as input, `reprobus-user.conf`. The script then takes care of adapting compilation based on the configuration, launching the simulation itself and creating additional output files for users. This bash script must be executed *inside of the REPROBUS Singularity container*.

The input configuration file is a txt file where principal simulation parameters are defined by the user. After the simulation is done, the script calls a Python `post-process-reprobus.py` script which takes the binary output from Fortran and recreates it in different additional formats for future user's analysis. For each simulation, additional outputs from Python script are :

- PNG plots of the estimated variable on the North pole between December and March, and on the South pole between May and September
- netCDF version of the station and profile output files

2.2.1 Bash script for the simulation launch

Bash script constitutes a link between user configuration file and the REPROBUS executable in the Singularity container. The script reads the user configuration file, launches the simulation according to parameters set by the user and calls the post-processing Python script. The only input for this Bash interface is the path to the user configuration file described earlier.

```
$ singularity shell reprobus-ifort.sif
Singularity>
Singularity> ./reprobus-simulation.sh --help

###             .-'';'-.
###            ,'   <_,-.`.          REPROBUS tool for
###           /)   ,--,_>\_\           reactive processes
###          |'   (      \_ |            ruling the ozone budget
###          |_    `-.    / |              in the stratosphere
###           \`-.   ;  _(`/
###            `.(    \/ ,'
###              `-....-'
###
### This script handles the REPROBUS simulation input
### parameters and launches the simulation, as well as
### the post-processing for the output additional
### formatting and results visualization
###
### Usage:  [options] arguments
### Options:
###   -h, --help              Show this help message and exit
### Arguments:
###   --config conf_fielpath  This argument must correspond to the configuration
###                           file where the user defines input parameters needed
###                           for the simulation

Singularity>
```

2.2.2 User configuration file

User configuration file is a txt file `reprobus-user.conf` (see an example content of this file below) where it is possible to set up multiple different variables and

parameters of the simulation. The file can be renamed if needed, as its path is given as input to the bash script by the user.

```
EXP="001442"
NSTART=1
COMPILER="ifort"
WDIR="/sedoo/resos/reprobus/WORK"
DATA_DIR="/sedoo/resos/reprobus/ECMWF"
RESTART_DIR="/sedoo/resos/reprobus/RESTART"
RES_DIR="/sedoo/resos/reprobus/RES"
IMAGES_DIR="/sedoo/resos/reprobus/IMAGES"
START_DATE="20230922"
END_DATE="20230923"
```

The `EXP` variable is the version of the reprobus; the `NSTART` variable allows to define whether the simulation must be launched based on the history files from previous simulations or if it should be initialized with the ozone files if no simulation was performed beforehand. `RESTART_DIR` is the directory where the `MODEL_history_YYYYMMDDHH_V` files will be stored, and the `RES_DIR` is the directory where the profile and the station output files will be stored.

Current version of the Bash user interface does not support the launch with NSTART=0 due to some hard-coded issues. You might want to contact REPROBUS tool providers for more information if necessary.

2.2.3 Python post-processing script

The Python script `post-process-reprobus.py` allows to recreate the binary simulation results into a more readable format such as netCDF, and create visual representations of the estimated variables. The script creates one netCDF file for each of the outputs in `RES_DIR`, and also creates figures for calculated chemical species from `MODEL_history_YYYYMMDDHH_V` files either for Northern or Southern hemisphere based on the simulation date. The Python script is launched automatically once the simulation is done, so the script must be placed in the same path as the Bash script that launches the simulation. If needed, the Python code can also be used individually for already existing simulations results (see the usage below).

```
usage: post-process-reprobus.py [-h] [--date DATE] [--restart-dir RESTART_DIR]
[--res-dir RES_DIR] [--image-dir IMAGE_DIR]

Post-processing of the REPROBUS Fortran output files

optional arguments:
  -h, --help            show this help message and exit
  --date DATE           End date of the simulation in YYMMDD format
  --restart-dir RESTART_DIR
```

```
                        Path to the directory with the MODEL_history Fortran binary
files
  --res-dir RES_DIR     Path to the directory where the station result files are
stored
  --image-dir IMAGE_DIR
                        Path to the directory where to save the figures
```

2.3 REPROBUS simulation launch

The Bash script has to be executed inside the Singularity container where REPROBUS was installed (`reprobus-ifort.sif` or `reprobus-nvidia.sif`). There are multiple methods to do so.

2.3.1 Interactive launch

If you want to launch the simulation in the interactive mode and be able to continue to work in the Singularity container, you can follow these steps:

1. launch Singularity container in the shell mode

```
$ singularity shell path/to/the/container.sif
```

2. launch Bash script with the simulation configuration

```
Singularity>
Singularity> /my/path/to/reprobus-simulation.sh --config /my/path/to/reprobus.conf
```

2.3.2 Non-interactive launch

It is also possible to launch the script in one go with the singularity exec command :

```
$ singularity exec path/to/the/container.sif /my/path/to/reprobus-simulation.sh
--config /my/path/to/reprobus.conf
```

All of the methods described above can be wrapped into cron, bash scripts or slurm jobs depending on one's needs and number of simulations.

2.3.3 Bind option

If during the simulation the input ECMWF data, or other files and/or directories, were not found, while the configuration is correct and the data is present, this could mean that the Singularity container does not recognize the directory where the GRIB data is stored. To fix this problem, the `bind` option can be used.

The `--bind` option allows to map directories on the host system to directories within the container. When Singularity 'swaps' the host operating system for the one inside your container, the host file system becomes partially inaccessible. The system administrator has the ability to define what bind paths will be included automatically inside each container. Some bind paths are automatically derived (e.g. a user's home directory) and some are statically defined (e.g. bind paths in the Singularity configuration file). In the default configuration, the directories `$HOME`, `/tmp`, `/proc`, `/sys`, `/dev`, and `$PWD` are among the system-defined bind paths. Thus, in order to read and/or write files on the host system from within the container, one must bind the necessary directories if they are not automatically included. Here's an example of using the `--bind` option and binding `/data` on the host to `/mnt` in the container (`/mnt` does not need to already exist in the container):

```
$ ls /data
bar  foo
$ singularity exec --bind /data:/mnt my_container.sif ls /mnt
bar  foo
```

You can bind multiple directories in a single command with this syntax:

```
$ singularity shell --bind /opt,/data:/mnt my_container.sif
```

This will bind `/opt` on the host to `/opt` in the container and `/data` on the host to `/mnt` in the container.

2.4 REPROBUS working directory structure

REPROBUS bash script can be called from any directory, as long as the correct paths to the working and output directories are indicated in the configuration file. The rules and the structure of the working directory are detailed below.

```
/root
 ├─path/to/wdir/                          -> working directory where the input files
 |   ├─ecmwf_137_levels.txt                needed for compilation are linked; this
```

```
|   ├─jno.f90                                          directory can be the same for multiple
|   ├─reprobus_1442.f                                  simulations, it is only needed for the
|   └─…                                                compilation and simulation running
|                                                      purposes; the clean-up is performed by the
|                                                      simulation launch script
|
├─path/to/restart/dir/                                 -> history files stored in the RESTART_DIR
|   └─MODEL_history_2023091912_001442
├─path/to/res/dir/                                     -> output folder with profile and station
|   ├─reprobus_airesadour_2023091912_001442.nc         files in the original binary format + netCDF
|   ├─reprobus_airesadour_2023091912_001442            format created with the post-processing
|   ├─reprobus_ddu_2023091912_001442.nc
|   ├─reprobus_ddu_2023091912_001442
|   ├─...
|   ├─stations_2023091912_001442.nc
|   └─stations_2023091912_001442
└─path/to/images/dir/                                  -> directory with the created images of
    ├─2023091912_BrOx_435.png                          calculated chemical species
    ├─2023091912_BrOx_475.png
    ├─2023091912_HCl_435.png
    ├─2023091912_HCl_475.png
    └─...
```

The simulation run needs input files and data being organized in a certain manner, but the simulation bash script takes care of providing symbolic links to the data and files if needed.

3. ECMWF data extraction for REPROBUS

The input data for the simulations is meteorological data : wind, temperature, relative humidity, vertical velocity and logarithm of surface pressure, coming from the ECMWF database. To extract and prepare the data in the correct format (detailed in the section 1.2.2) the script `reprobus-extract-ecmwf.sh` should be used. The user can configure the start and end date of the data, as well as the working and output directories for the data extraction. The extracted and processed data can then be directly used as input for a corresponding REPROBUS simulation.

3.1 Input parameters

The script takes as input a configuration txt file. This configuration file contains all the parameters that the user can change and configure:

```
START_DATE="YYYYMMDD"
END_DATE="YYYYMMDD"
DATA_DIR=""
WORKING_DIR=""
```

The `DATA_DIR` and `WORKING_DIR` variables correspond respectively to the directory where the extracted data will be stored and where the working files (like request files) will be stored. These two variables can point to the same or different directories. If these directories do not exist, they will be created by the script. The start and end dates are included in the time period of the data extraction. For example, in the case of a start date being 20231101 and an end date 20231102, the data will be extracted for 01/11/2023 AND for 02/11/2023.

3.2 Data extraction launch

To extract the data, you should follow next steps:

1. log in to the ECMWF MARS server
2. set up parameters in your configuration file [.conf]
3. do

```
$ ./reprobus-extract-ecmwf.sh --config path/to/the/reprobus_ecmwf.conf
```

The script will launch the extraction based on your set up parameters. It is advised to launch this script as a SLURM job on the MARS server, otherwise the shell will be inaccessible and the script might take longer to perform. To launch this script as a job, you can use the following syntax :

```
$ sbatch --wrap="path/to/the/reprobus-extract-ecmwf.sh --config
path/to/the/reprobus_ecmwf.conf"
```

This command will launch a SLURM job with your configuration file and create a `.out` log file in the directory from where the command was called. You can consult the SLURM documentation [2] to see how you can customize this call and add features like job name or mail user for notifications.

3.3 O3 extraction for the simulation initialization

In cases where the initialization is needed, the ozone data must be extracted for this special type of the simulation. The script reprobus-extract-o3.sh must be used. The usage of this script is the same as for the reprobus-extract-ecmwf.sh, except that there is no END_DATE parameter in the configuration file. The start date can also be one of the dates when the re-initialization of the model is needed.

```
START_DATE="YYYYMMDD"
DATA_DIR=""
WORKING_DIR=""
```