| **SEDOO-AERIS** | |  |
| --- | --- | --- |
| **MIMOSA - Modélisation Isentrope du transport Méso-échelle de l'Ozone Stratosphérique par Advection** | | |
|  | **Technical Document**  **& User Manual** | |

|  | **Name** | **Organisation** | **Date** | **Visa** |
| --- | --- | --- | --- | --- |
| **Written by:** | Daria Malik | Magellium | 24/10/2023 |  |
|
| **Checked by:** | Vanessa Pedinotti | Magellium |  |  |
| **Approved by:** | Béatrice Berthelot | Magellium |  |  |
|  |  |  |  |  |

| **Document reference:** |  |
| --- | --- |
| **Edition.Revision:** | 2.0 |
| **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 | 24/10/2023 | Creation of document |  |
| 2 | 0 | 06/05/2024 | Document update |  |

**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. MIMOSA Overview 3**](#_ziu01fwmppg)

[1.1 MIMOSA architecture 3](#_t951qf3gsjgg)

[1.2 MIMOSA input files 4](#_w8z3nm9xkiag)

[1.2.1 input.namelist 4](#_44epz9fptgdt)

[1.3 MIMOSA Output Files 6](#_8evsalk92iec)

[1.3.1 Output estimated variables 6](#_y4lcir3kpq0e)

[1.3.2 Recovery files 8](#_646gsaocjpwa)

[1.3.3 Stations files output 8](#_lrnx0hflxfhf)

[**2. MIMOSA simulation for User 8**](#_f9lvwopfha6)

[2.1 MIMOSA Singularity container 8](#_mne4m4kc3ydy)

[2.2 MIMOSA User interface 9](#_rzmondgn8u2e)

[2.2.1 Bash script for the simulation launch 9](#_mvk443rx8ber)

[2.2.2 User configuration file 10](#_qle7pxd9ov0w)

[2.2.3 Python post-processing script 11](#_8a4me6qpta55)

[2.3 MIMOSA simulation launch 12](#_x8psglesk7qi)

[2.3.1 Interactive launch 12](#_k2wslxerc165)

[2.3.2 Non-interactive launch 12](#_o1k3szsthp8a)

[2.3.3 Bind option 13](#_4h0w1hmcgd7j)

[2.4 MIMOSA working directory structure 13](#_6r1v2cx7zhbf)

[**3. ECMWF data extraction for MIMOSA 15**](#_nbbah2x7fy28)

[3.1 GRIB data fields and format 15](#_u4ubp4m4mmn6)

[3.2 ECMR data fields and format 16](#_vlqe2go11i4w)

[3.3 Data extraction input parameters 17](#_66vfpur28xme)

[3.4 Data extraction launch 18](#_dyfyn9fgh5ui)

[**Bibliography 19**](#_i5emtyb1ladi)

[**Annexes 20**](#_7z6lwgcps987)

[Annexe 1 : input.namelist syntax 20](#_ubg7rgbdnx00)

[Annexe 2 : get GRIB grid size 23](#_24pixhlzhzec)

# 1. MIMOSA Overview

MIMOSA is a high-resolution potential vorticity advection model developed in Fortran by A. Hauchecorne (Hauchecorne et al., 2002). It is initialized at a time t from ECMWF data (horizontal wind fields U and V, temperature and pressure) on an orthogonal grid centered on the north pole. MIMOSA calculates then advects the potential vorticity on isentropic surfaces with a resolution of 1/3 or 1/6 degree in latitude and longitude.

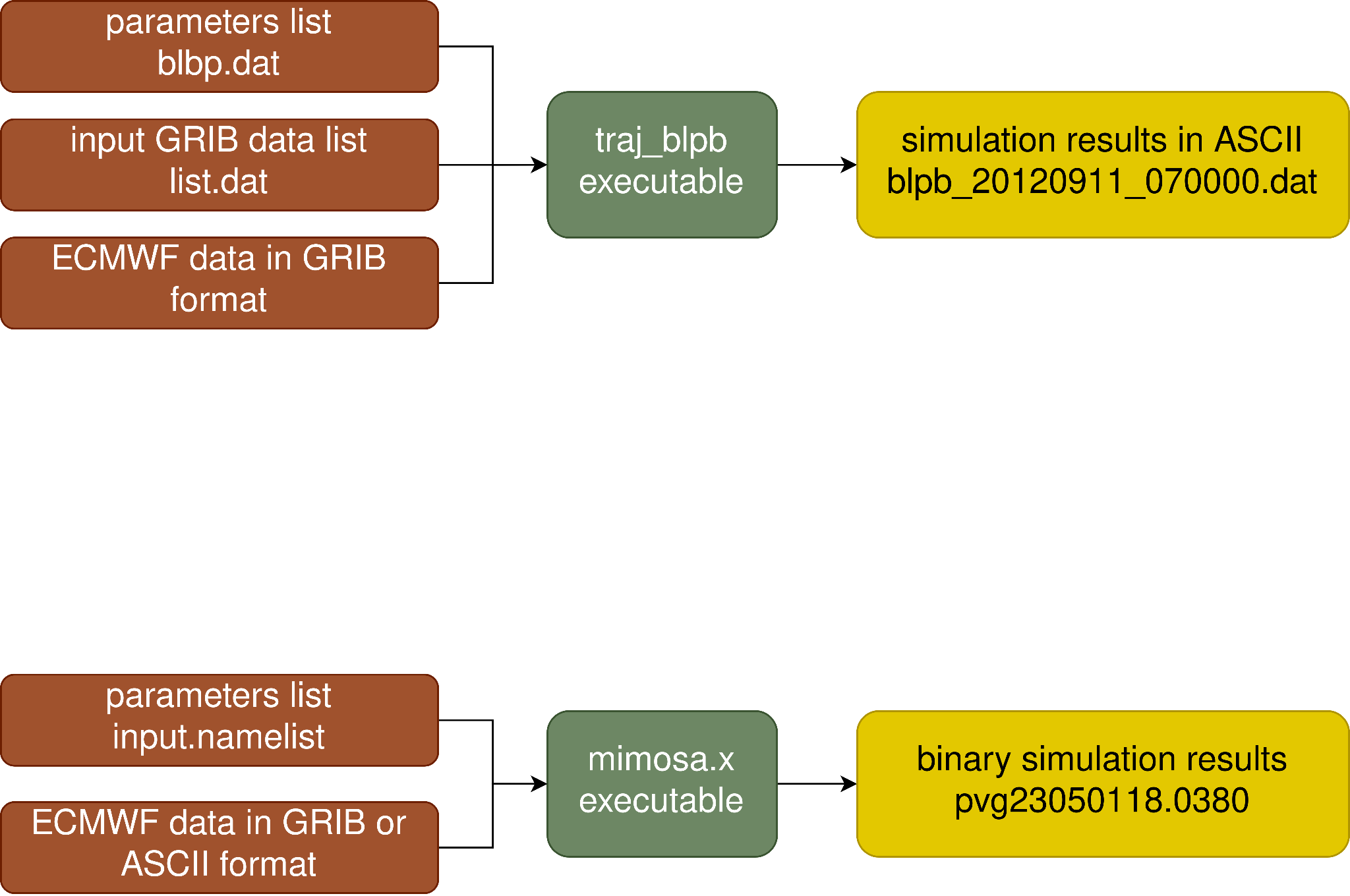
MIMOSA has been updated to allow the use of ECMWF input data on model levels. This document describes the prerequisites necessary for the operational version of MIMOSA, the extraction of the ECMWF data necessary for the simulation, the simulation progress and the output of the simulation.

For the easier installation, maintenance and portability, MIMOSA executables and dependencies have been wrapped up into a Singularity container.

## 1.1 MIMOSA architecture

The main Fortran program that launches simulations is the mimosa.f90 script. The makefile added with the source files in the src/ directory of the git repository allows the MIMOSA compilation with the gfortran compiler installed in the Singularity container. The executable will be located in the container in the following path : /usr/local/MIMOSA/src/mimosa.x

The executable needs to be launched from the working directory where the input file input.namelist and the ECMWF meteorological data are present (Fig. 1). This input file contains multiple simulation parameters such as start and end date of the simulation, spatial and temporal resolutions of the input data and output results, and others. These parameters will be detailed later in the document. The output files [pvg/tg/ug/vg]\_[date][time].[theta] are in binary format and correspond respectively to the potential vorticity, temperature, U wind field, V wind field. The ”g” suffix means that the result is computed for the global map. Other possible suffixes are ”n” and ”s” for the North and South hemispheres respectively.



*Figure 1 : scheme of the MIMOSA simulation input/output*

## 1.2 MIMOSA input files

### 1.2.1 input.namelist

The input.namelist file contains multiple parameters that allow to define the MIMOSA simulation. The syntaxe of the file is detailed in the Annexe 1. The Table 1 describes in detail the parameters that must be set for any simulation.

*Table 1 : MIMOSA input parameters description*

| **Parameter name** | **Description** | **Values** |
| --- | --- | --- |
| **zone** | Defines the geographical area of the simulation | 1 = north (-10N, 90N)  2 = south (-90N, 10N)  3 = global (-90N, 90N) |
| **intype** | Defines the type of input files | 1 = ASCII (.ECMR)  2 = GRIB (.grib) |
| **iand** | Defines the starting date and time of the simulation | year (YY) |
| **moisd** | month (MM) |
| **jourd** | day (DD) |
| **iheured** | hour (HH) |
| **ianf** | Defines the ending date and time of the simulation | year (YY) |
| **moisf** | month (MM) |
| **jourf** | day (DD) |
| **iheuref** | hour (HH) |
| **initpv** | Defines if the simulation is new or a restart. | 0 = if a ph\* file from a previous run should be read  1 = initialization is needed |
| **teta** | Defines the isentropic surface (K), shouldn't be greater than 950K for isobaric input files (intype = 1) |  |
| **nx** | Number of grid points along longitudes in input ECMWF files |  |
| **ny** | Number of grid points along latitude in input ECMWF files |  |
| **np** | Number of pressure levels (intype = 1) or number or model levels (inType = 2) |  |
| **pres** | Allows to define the pressure levels of isobaric file (intype = 1). Not needed for GRIB files (intype = 2). If there is more than 50 levels, declaration of pres variable in mimosa.f95 | value1, value2, value3… (separated by the comma) |
| **paslat** | Input ECMWF data resolution along latitude |  |
| **paslong** | Input ECMWF data resolution along longitude |  |
| **latminecmr** | Define the minimum and maximum latitude of ECMWF grid |  |
| **latmaxecmr** | Define the maximum and maximum latitude of ECMWF grid |  |
| **ndeg** | Defines the number of MIMOSA grid points per degree of latitude and longitude | either 3 or 6 |
| **nlis2d** | Defines the number of points use for the smooth | 15 for default |
| **nhgrid** | Defines the number of hours between two call to regrid | 6 for default |
| **nwrite** | Defines the number of hours between two outputs of PV files | 6 for default |
| **nhmod** | Defines the number of hours between two ECMWF files |  |
| **nhrelax** | Defines the number of hours of relaxation time | 240 for default |
| **nprhmod** | Defines the hour of the first ECMWF file |  |
| **nprwrite** | Defines the first hour of PV file |  |
| **indifexpl** | Defines if explicit diffusion is activated | 0 = no explicit diffusion  1 = explicit diffusion |
| **diff** | Defines the value of the explicit diffusion if it is activated | 4050 for default |
| **nrun** | Defines the name of the directory where the output will be stored | 1 → directory RUN01  2 → directory RUN02  15 → directory RUN15 |
| **nwtemp** | Defines the time between two output of temperature or wind |  |
| **wind\_out** | Defines if wind horizontal components files will be saved as output | 0 = no output of wind files  1 = output of wind files |
| **t\_out** | Defines if temperature files will be saved as output | 0 = no output of temperature  1 = output of temperature |
| **stations\_out** | Defines if PV, temperature and PV profiles of stations files will be saved | 0 = no output of station files  1 = output of station files |

The initpv parameter allows to define if the simulation has to be launched with the initialisation, or if the recovery files (described further in the document) should be read from a previous launch. In summary, the simulation for days D1 through D2 must be launched in the initialization mode if there was no simulation ending on the day D1-1. If there were a simulation for days DX through D1-1, recovery files will be present in the working directory of this simulation, and the simulation D1→D2 can be launched *in the same working directory* but with the initpv=0 in the configuration file. If the D1→D2 simulation is launched in a new directory, recovery files will not be found by the algorithm, and the code will exit on error. Launching in a new empty directory must be made with the initpv=1 parameter to initialize the simulation. If the new simulation covers an absolutely new date range and is not in a continuity of the existing simulations, the initpv parameter must be set to 1 too.

## 1.3 MIMOSA Output Files

The MIMOSA simulation produces two types of binary outputs: 1) potential vorticity/temperature/wind fields binary files which are estimated for the requested dates and hours, and 2) recovery files needed to simulate long time series, and station files (if requested).

### 

### 1.3.1 Output estimated variables

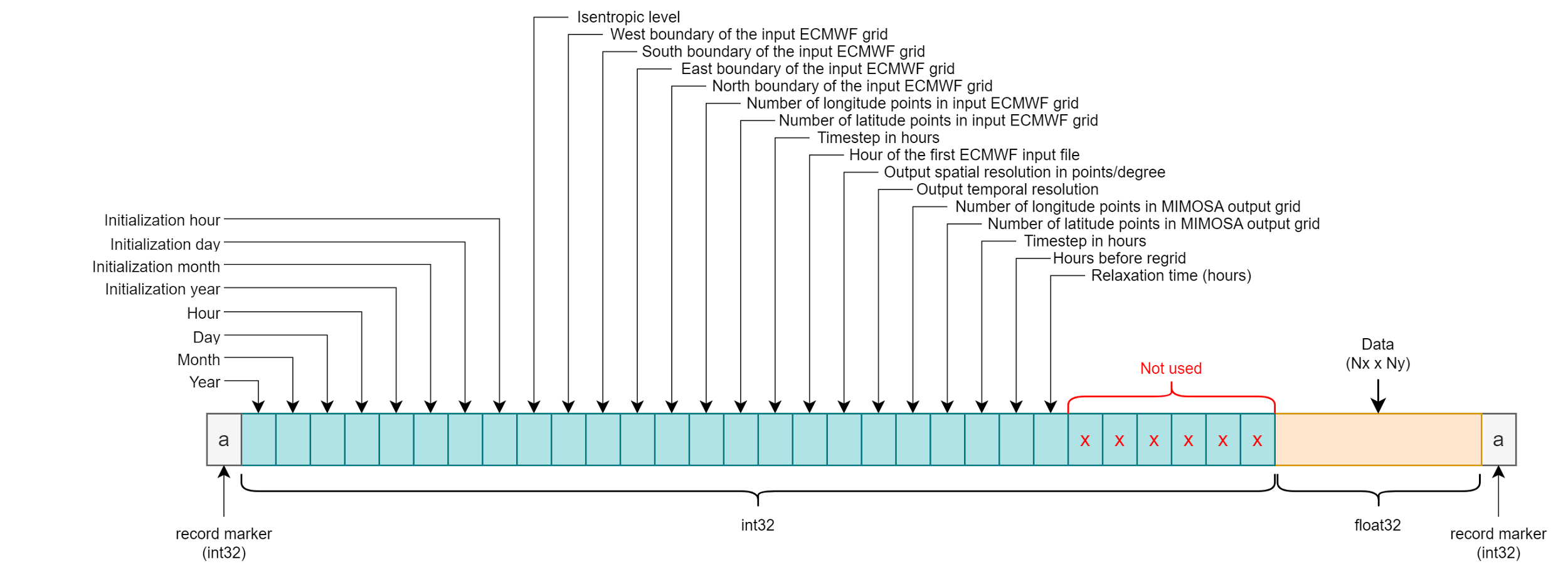
Output binary files of the estimated variables are named with the following syntax:

* pvXYYMMDDHH.THETA
* tXYYMMDDHH.THETA
* uXYYMMDDHH.THETA
* vXYYMMDDHH.THETA

where

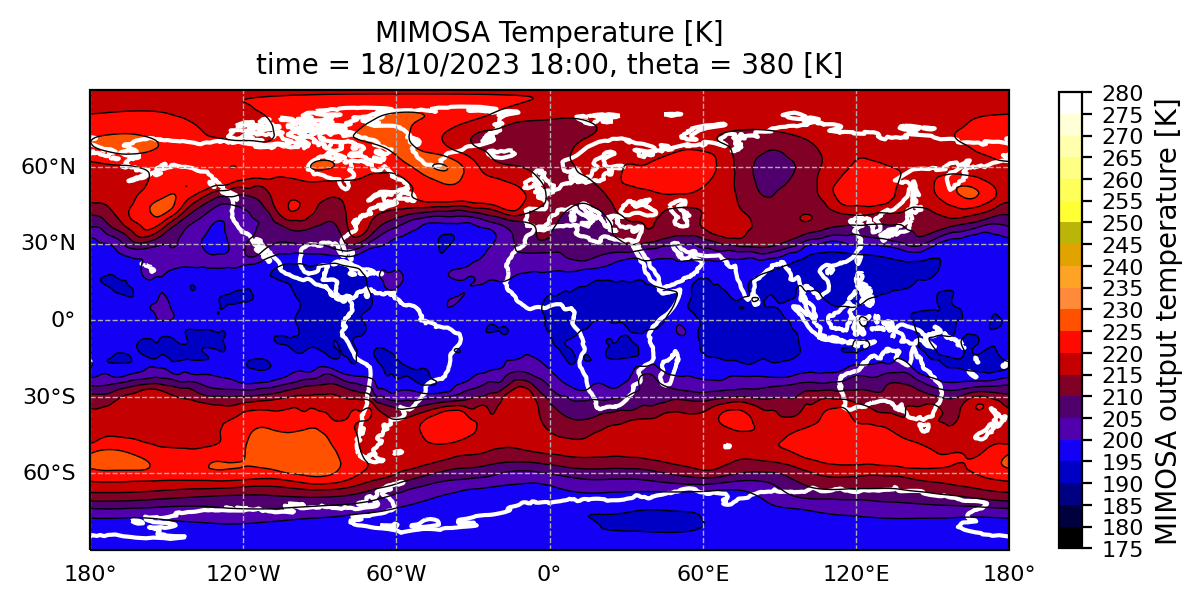
* pv/t/u/v correspond to the above mentioned variables
* X = g/n/s depending on the requested simulation geographical zone
* YYMMDDHH is the output date and time (2 digits year, month, day and hour)
* THETA is the value of the isentropic level of this output

The output files are in binary unformatted Fortran format and contain two parts: the header and the data itself. The data follows little endian convention. Fig. 2 shows the organization of the data in the binary output.

[](https://app.diagrams.net/?page-id=VGcWQ1uA2lOwWw8U6AAi&scale=auto#G1zTjAJhYo_0OS5C_VTIOaMg-ly95jQKHr)

*Figure 2 : header description of the MIMOSA binary output*

The order of the data is a row order with the column index changing the fastest. It means that the first N points is the first row of the data, the next N points is a second row etc. This data can be plotted as 2D variables to visualize simulation results (Fig. 3). The data size is defined respectively via header values “Number of longitude points in MIMOSA output grid” and “Number of latitude points in MIMOSA output grid”.



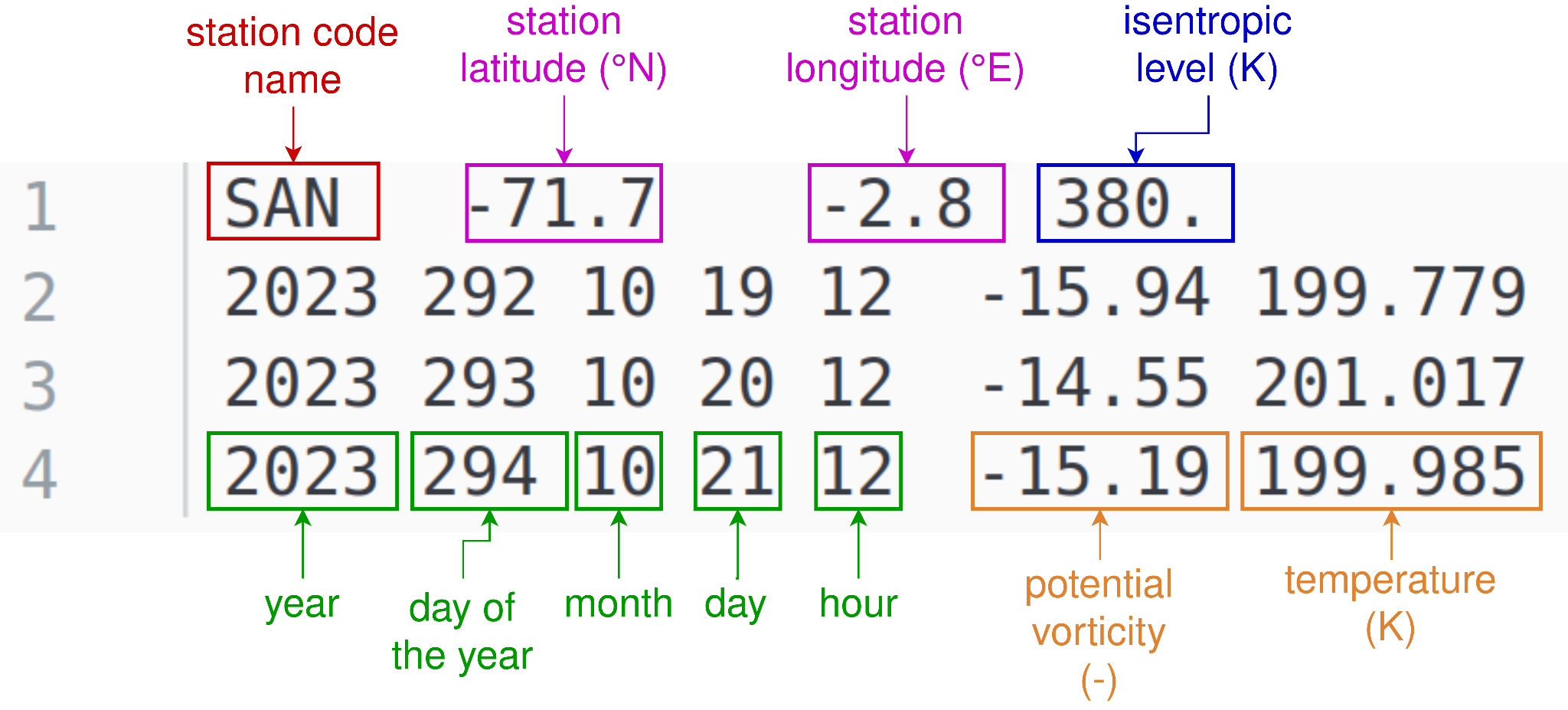
*Figure 3 : example of the MIMOSA output temperature visualization*

### 1.3.2 Recovery files

Recovery files are also binary outputs which allow running long simulations and are reused by MIMOSA for quicker computations. The name of these files follows the syntaxe [phn/phs]YYMMDDHH.THETA, where the YYMMDDDHH and THETA parts have the same meaning as in the variable output name syntaxe. These recovery files will be used by the simulation if initpv=0, and if the new simulation launched in this same directory starts on the next day from the end of the previous simulation.

### 1.3.3 Stations files output

The stations outputs correspond to the estimated PV and temperature values extracted above ground sites. These files are written in the ASCII format and are composed of two main parts: the header and the estimated variables (Fig. 4).



*Figure 4 : MIMOSA stations output example*

# 2. MIMOSA simulation for User

## 2.1 MIMOSA Singularity container

MIMOSA 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 mimosa-container.def is a Singularity Definition File, which 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 mimosa.sif mimosa-container.def |
| --- |

⚠️ 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 MIMOSA User interface

The user interface for the easier simulation launches is a bash script mimosa-user-script.sh which requires a configuration file from the user as input, mimosa.conf. The script then takes care of writing an input.namelist file for MIMOSA Fortran executable, launching the simulation itself and creating additional output files for users. This bash script must be executed inside of the MIMOSA 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-mimosa.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 global/North pole/South pole maps
* one netCDF file for each of the binary outputs
* netCDF files where multiple binary outputs are combined along the time dimension

### 2.2.1 Bash script for the simulation launch

Bash script constitutes a link between user configuration file and the MIMOSA executable in the Singularity container. The script reads the user configuration file, writes a special configuration for the Fortran executable (input.namelist), launches the simulation 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 mimosa.sif  Singularity>  Singularity> ./mimosa-user-script.sh --help  ### .-'';'-.  ### ,' <\_,-.`. MIMOSA tool for  ### /) ,--,\_>\\_\ high-resolution  ### |' ( \\_ | potential vorticity  ### |\_ `-. / | advection model  ### \`-. ; \_(`/  ### `.( \/ ,'  ### `-....-'  ###  ### This script handles the MIMOSA simulation input  ### parameters and launches the simulation, as well as  ### the post-processing for the output additional  ### reformating 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 mimosa.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.

The variables defined in this file correspond to the variables described for the input.namelist file.

| SIMUDIR="/root/mimosa\_simulation\_1"  NRUN=3  SYEAR=23  SMONTH=10  SDAY=18  SHOUR=12  EYEAR=23  EMONTH=10  EDAY=21  EHOUR=12  ZONE=3  INTYPE=1  INITPV=1  THETA=( 380 475 550 675 )  NX=180  NY=91  NP=17  PRES="300.,250.,200.,150.,100.,70.,50.,30.,20.,10.,7.,5.,3.,2.,1."  PASLAT=2  PASLONG=2  LATMIN=-90  LATMAX=90  NDEG=6  NLIS2D=15  NHGRID=6  NWRITE=6  NHMOD=12  NHRELAX=240  NPRHMOD=0  NPRWRITE=0  INDIFEXPL=0  DIFF=4050  NWTEMP=6  WINDOUT=0  TOUT=1  STATIONS=1 |
| --- |

### 2.2.3 Python post-processing script

The Python script post-process-mimosa.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 simulation outputs, and one netCDF file for each of the variable/isentropic level combinations. In these combined files, the data is merged along the time dimension, while in the individual file the data corresponds to one variable on one isentropic level at a time t.

For the simulation launched via the bash script, the post-processing Python script must be located in the same directory as the Bash script; it is launched automatically after the simulation has finished. If needed, the Python code can also be used individually for already existing simulations results (see the usage below).

| usage: post-process-mimosa.py [-h] [--start-date START\_DATE] [--end-date END\_DATE] [--out-dir OUT\_DIR] [--im-dir IM\_DIR]  Post-processing of the MIMOSA Fortran output files for the creation of netCDF copies and results visualization  optional arguments:  -h, --help Show this help message and exit  --start-date START\_DATE Start date of the files to process in YYMMDD format  --end-date END\_DATE End date of the files to process in YYMMDD format  --out-dir OUT\_DIR Path to the directory with the Fortran binary output files  --im-dir IM\_DIR Path to the directory where to save visualization |
| --- |

## 2.3 MIMOSA simulation launch

The Bash script has to be executed inside the Singularity container where MIMOSA was installed (mimosa.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 |
| --- |

1. launch Bash script with the simulation configuration

| Singularity>  Singularity> /my/path/to/mimosa-user-script.sh --config /my/path/to/mimosa.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/mimosa-user-script.sh --config /my/path/to/mimosa.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 MIMOSA working directory structure

MIMOSA bash script can be called from any directory, as long as the correct path to the working directory with input files is indicated in the configuration file. The rules and the structure of the working directory are detailed below.

| **/root**  └──mimosa\_wdir/  ├──GRIB/  │ └──2023/  │ └──05/  │ ├──D23050100.grib  │ ├──D23050106.grib  │ └──...  ├──RUNxx/  │ ├──DATA/  │ │ ├──pvg23050118.0380  │ │ ├──pvg23050118.0380.nc  │ │ ├──pvg23050200.0380  │ │ ├──pvg23050200.0380.nc  │ │ └──pvg.0380.nc  │ ├──IMAGES/  │ │ ├──pvg23050118.0380.png  │ │ └──pvg23050200.0380.png  │ ├──phn230501012.0380  │ └──phn230502012.0380  ├──mimosa.conf  ├──input.namelist  └──mimosa.x | -> working directory  -> **mandatory** GRIB folder in working directory with GRIB data inside organized by YYYY and MM subdirectories  -> output folder created by the simulation (simulation results and recovery files)  -> user configuration file  -> created MIMOSA configuration file  -> MIMOSA executable |
| --- | --- |

If it is not possible to run simulations in the directory where the input ECMWF data is placed or to copy the data into the working directory, a potential solution would be to create a symbolic link [1] from the working directory to your data directory. For example, if your data is stored at /home/data/ECMWF/2023 and your working directory is /home/user/mimosa/simulation\_1, you can do the following :

| [home] $ cd /home/user/mimosa/simulation\_1  [simulation\_1] $ ln -s /home/data/ECMWF ./ECMWF |
| --- |

If the command performed successfully, you will see in the working directory a ECMWF folder with indication ->/home/data/ECMWF. Now the newly created ECMWF “file” will point to the true ECMWF data directory that will be seen by the simulation. The simulation can then be launched as usual.

# 3. ECMWF data extraction for MIMOSA

The input data for the simulations is meteorological data : wind, temperature and logarithm of surface pressure, coming from the ECMWF database. To extract and prepare the data in the correct format, the script mimosa-extract-grib.sh or mimosa-extract-ecmr.sh should be used. These scripts extract the data either in the GRIB or ASCII (ECMR) format, respectively. Both of the formats are supported by the MIMOSA tool but the input configuration of the simulation itself must take into account which of the formats should be used. For this, the corresponding parameter intype must be set either to 1 or 2 (see section *1.2.1* or *2.2.2*). The user can also configure the start and end date of the data to extract, as well as the spatial resolution and the data class (only in the GRIB version).

The configuration of two data extractions are as follows:

* GRIB data
  + extracted on ECMWF 137 model levels
  + the timestep is 3 hours if the requested date range is up to J+6; if the end date exceeds the J+6 limit, the timestep is 6 hours
* ECMR data
  + extracted on 17 pressure levels
  + the timestep is 12 hours

The script must be launched on the ECMWF MARS server (ecs, hpc or other). The data extraction was tested with a member-state user account. Other more public accounts might customize the script based on the MARS services or APIs available for their type of user. The data is extracted and stored in the directory requested in the input configuration; afterwards, the data can be used for the simulation.

## 3.1 GRIB data fields and format

In the case where the input data is in the GRIB format, the data has to be organized in a certain manner: all variables for the same date and time should be packed together in one file. The name of the file must follow the syntax DYYMMDDHH.grib, where YYMMDDHH correspond to the year (2 digits), month, day and hour of the data packed in the file.

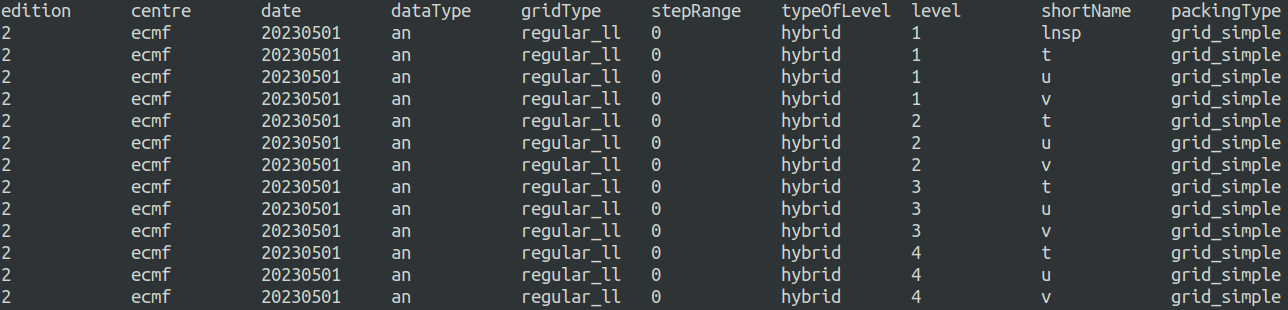
The mandatory meteorological fields are:

* U component of wind (U)[[1]](#footnote-0)
* V component of wind (V)
* temperature (t)
* logarithm of surface pressure (lnsp)

The data in the file must be stored in the following order :

1. lnsp
2. t,u,v on the level 1
3. t,u,v on the level 2
4. t,u,v on the level 3
5. etc

An example of the grib\_ls command on one of the files is presented on Fig. 5 below.



*Figure 5 : example of grib\_ls command output for the MIMOSA meteorological input grib file*

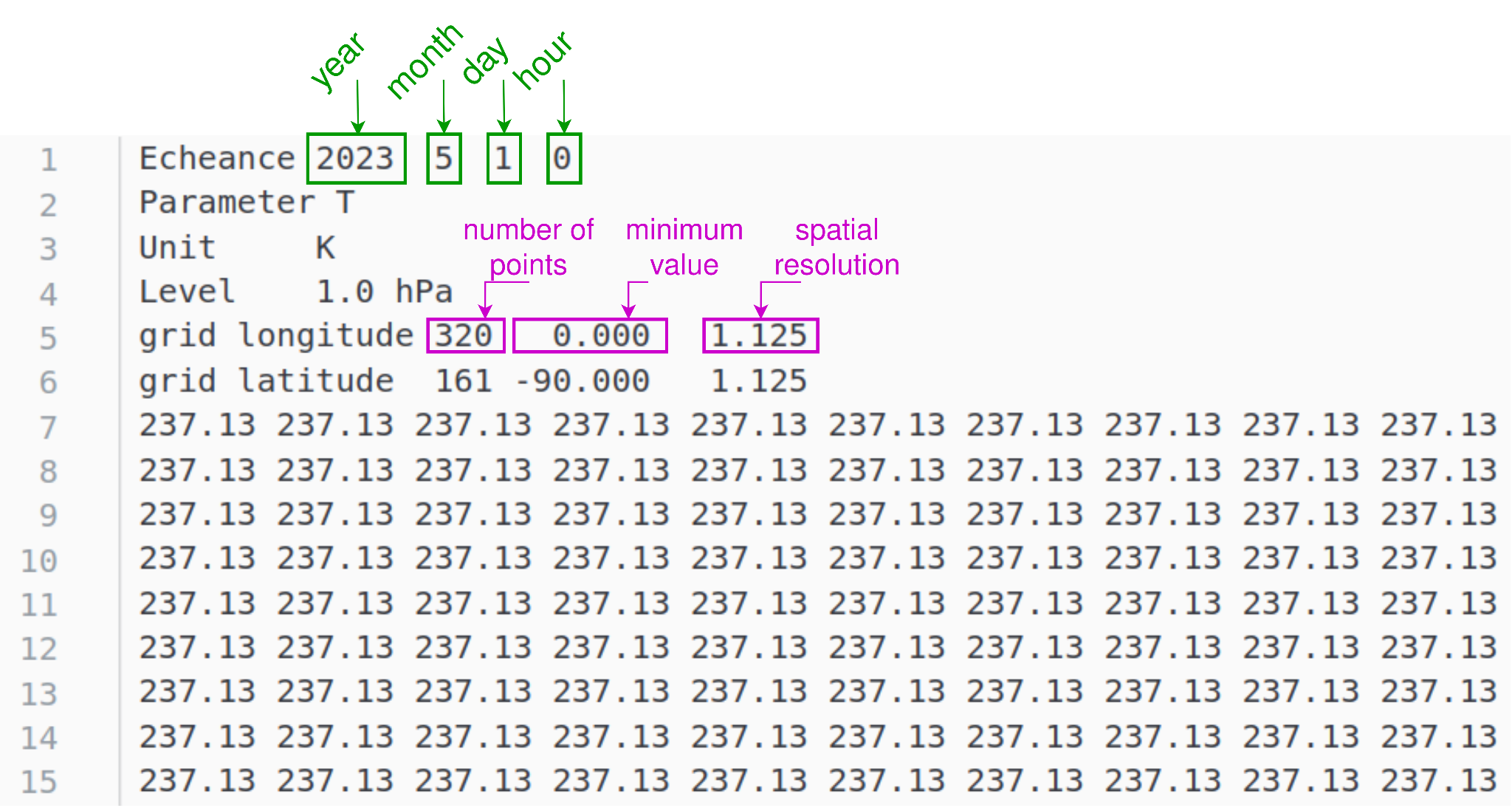
The script mimosa-extract-grib.sh allows to extract the data, sort it in the correct order and rename files with correct names for the Fortran executable. Based on the start and end date, the data will be extracted from the analysis and/or forecast database. The timestep is 3 hours if the requested date range is up to J+6; if the end date exceeds the J+6 limit, the timestep is 6 hours.

## 3.2 ECMR data fields and format

The script mimosa-extract-ecmr.sh allows to extract the data from ECMWF server and format it in the ASCII format with .ECMR extension. The variables in this case are only temperature and U/V wind fields, without the logarithm of surface pressure.

The data is extracted on the 17 pressure levels: 1,2,3,5,7,10,20,30,50,70,100,150,200,250,300,400,500 hPa. Based on the start and end date, the data will be extracted from the analysis and/or forecast database. The timestep is 12 hours.

The data in ASCII files is organized in matrixes of 10 columns, so that the (number of rows x 10) = total number of data points. An example of such a matrix is presented on Fig. 6; matrices for other variables and levels for this date are saved further in the file one after another.



*Figure 6 : example of the ASCII content of the MIMOSA input ECMWF data*

As in the case of the GRIB data, the Bash script allows the user to extract the data, store it in the correct order and format and rename files according to the rules.

## 3.3 Data extraction 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="20230101"  END\_DATE="20230105"  DATA\_CLASS="od"  SPATIAL\_RESOLUTION="1.125"  DATA\_DIR=$(pwd)  WORKING\_DIR=$(pwd) | START\_DATE="20231018"  END\_DATE="20231022"  SPATIAL\_RESOLUTION="1.125"  DATA\_DIR=$(pwd)  WORKING\_DIR=$(pwd) |
| --- | --- |
| *Input parameters for the GRIB extraction* | *Input parameters for the ECMR extraction* |

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.

## 3.4 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

| $ ./mimosa-extract-[grib/ecmr].sh --config path/to/the/conf\_file.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/mimosa-extract-[grib/ecmr].sh --config path/to/the/conf\_file.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.

# Bibliography

[1] Linux man page of the ln command

[*https://www.mankier.com/1/ln*](https://www.mankier.com/1/ln)

[2] SLURM sbatch command input options

[*https://slurm.schedmd.com/sbatch.html*](https://slurm.schedmd.com/sbatch.html)

[3] ECMWF MARS server filesystems

[*https://confluence.ecmwf.int/display/UDOC/HPC2020%3A+Filesystems*](https://confluence.ecmwf.int/display/UDOC/HPC2020%3A+Filesystems)

[4] MARS catalogue

[*https://apps.ecmwf.int/mars-catalogue/*](https://apps.ecmwf.int/mars-catalogue/)

# Annexes

## Annexe 1 : input.namelist syntax

!==========================================================================!

! \_\_ \_\_ \_\_\_\_\_ \_\_ \_\_ \_\_\_\_ \_\_\_\_\_ !

! | \/ |\_ \_| \/ |/ \_\_ \ / \_\_\_\_| /\ !

! | \ / | | | | \ / | | | | (\_\_\_ / \ !

! | |\/| | | | | |\/| | | | |\\_\_\_ \ / /\ \ !

! | | | |\_| |\_| | | | |\_\_| |\_\_\_\_) / \_\_\_\_ \ !

! |\_| |\_|\_\_\_\_\_|\_| |\_|\\_\_\_\_/|\_\_\_\_\_/\_/ \\_\ !

!==========================================================================!

!

! CARACTERISCS OF THE RUN

!

&run

!-----

! ZONE defines the geographical area :

! -> 1 for the Northern Hemisphere [-10N, 90N]

! -> 2 for the Southern hemisphere [-90N, 10N]

! -> 3 for the both Hemisphere [-90N, 90N]

zone = 3

!-----

! INTYPE defines the type of input files :

! -> 1 for ASCII isobaric files (\*.ECMR)

! -> 2 for GRIB encoded model levels files (\*.grib)

intype = 2

!-----

! IAND, MOISD, JOURD, IHEURED define the starting date of the simulation

! -> Year (YY)

iand = 23

! -> Month (MM)

moisd = 05

! -> Day (DD)

jourd = 01

! -> Hour (HH)

iheured = 12

!-----

! IANF, MOISF, JOURF, IHEUREF define the final date of the simulation

! -> Year (YY)

ianf = 23

! -> Month (MM)

moisf = 05

! -> Day (DD)

jourf = 02

! -> Hour (HH)

iheuref = 12

!-----

! INITPV defines if the simulation is new or a restart

! -> 1 if initialization is need

! -> 0 if a ph\* file from a previous run should be read

initpv = 1

!-----

! TETA defines the isentropic surface (K)

! -> shouldn't be greater than 950K for isobaric input files (intype = 1)

teta = 625

/

!

! CARACTERISCS OF ECMWF GRID

!

&grid

!-----

! NX, NY and NP define the number of points of the ECMWF grid

! -> Number of grid points along longitudes

nx = 320

! -> Number of grid points along latitudes

ny = 161

! -> Number of pressure levels (intype = 1) or number or model levels (inType = 2)

np = 137

!-----

! PRES allows to define the pressure levels of isobaric file (intype = 1)

! -> this variable is not needed for GRIB files (intype = 2)

! -> If there is more than 50 levels, declaration of pres variable in mimosa.f95

pres(1) = 1000.,975.,950.,925.,900.,875.,850.,825.,800.,775.,750.,700.,650.,600.,550.,500.,450.,400.,350.,300.,250.,225.,200.,175.,150.,125.,100.,70.,50.,30.,20.,10.,7.,5.,3.,2.,1.

!-----

! PASLAT and PASLONG define the horizontal resolution of the ECMWF grid

! -> resolution along latitude

paslat = 1.125

! -> resolution along longitude

paslong = 1.125

!-----

! LATMINECMR and LATMAXECMR define the minumum and maximum latitude of ECMWF grid

! -> minimum latitude

latminecmr = -90

! -> maximum latitude

latmaxecmr = 90

/

!

! CONFIGURATION OF THE SIMULATION

!

&config

!-----

! NDEG defines the number of MIMOSA grid points per degree of latitude and longitude

! -> value should be 3 or 6

ndeg = 6

!-----

! NLIS2D defines the number of points use for the smooth

nlis2d = 15

!-----

! NHGRID defines the number of hours between two call to regrid

nhgrid = 6

!-----

! NWRITE defines the number of hour between two outputs of PV files

nwrite = 6

!-----

! NHMOD defines the number of hours between two ECMWF files

nhmod = 6

!-----

! NHRELAX defines the number of hours of relaxation time

nhrelax = 240

!-----

! NPRHMOD defines the hour of the first ECMWF file

nprhmod = 0

!-----

! NPRWRITE defines the first hour of PVF file

nprwrite = 0

!-----

! Explicit diffusion

! INDIFEXPL defines if explicit diffusion is activated

! -> 0 no explicit diffusion

! -> 1 explicit diffusion

indifexpl = 0

! DIFF defines the value of the explicit diffusion if it is activated

diff = 4050

/

!

! OUTPUT OF THE SIMULATION

!

&output

!-----

! NRUN defines the name of the directory where MIMOSA outputs will be saved

! -> if nrun = 5, files will be saved in RUN05 directory

! -> if nrun = 16, files will be saved in RUN16 directory

nrun = 1

!-----

! NWTEMP defines the time between two output of temperature or wind

nwtemp = 6

!-----

! WIND\_OUT defines if wind horizontal components files will be saved

! -> 0 no output of wind files

! -> 1 output of wind files

wind\_out = 0

!-----

! T\_OUT defines if temperature files will be saved

! -> 0 no output of temperature files

! -> 1 output of temperature files

t\_out = 1

!-----

! STATIONS\_OUT defines if PV and temperature and PV profiles at stations files will be saved

! -> 0 no output of stations files

! -> 1 output of stations files

stations\_out = 0

/

## Annexe 2 : get GRIB grid size

Below is the command that can be used inside the MIMOSA singularity container in order to retrieve the x/y size of the meteorological fields. The first value is the number of points in longitude direction, the second is the number of points in latitude direction.

| $ singularity shell [--bind path/to/bind] mimosa.sif  Singularity>  Singularity> grib\_get -p Ni,Nj your\_grib\_file.grib | head -1  320 161 |
| --- |

1. the short name of the field in the parentheses corresponds to the short name of this field in the ECMWF MARS database [↑](#footnote-ref-0)