Online Trajectory Module in COSMO - A short user guide

Document version: 1.0 (as of June 2014)

Annette K. Miltenberger, Stephan Pfahl, Anne Roches, Heini Wernli
IAC and C2SM, ETH Zurich
Contact: stephan.pfahl@env.ethz.ch

1 Introduction

An online trajectory module is implemented in COSMO 5.1 and higher versions. The module is a diagnostic tool that enables to compute trajectories (i.e. to follow air parcels) in a very accurate manner, since the position of a trajectory can be updated at each model time step based on the model wind fields. Furthermore, the module enables to trace any prognostic variable as well as diagnostic variables along the trajectories. A detailed description of the algorithm used, of the workflow, of the functionality provided by the module and an example of scientific application can be found in Miltenberger et al. (2013). The module has been tested on several platforms and applied in multiple research projects, demonstrating its versatility and scientific soundness.

This document is based on the user guide published as supplement material of Miltenberger et al. (2013). Discrepancies that can be observed between both documents reflect the changes made to the code between the first version and the version that is currently implemented in the official COSMO version 5.1 and higher. This user guide describes shortly the implementation in the official COSMO code and presents the technical aspects that are relevant for the user of the module.

2 Implementation overview

The online trajectory module consists of two different source files, namely `data_traj.f90` and `src_traj.f90`. While the first file contains the definition of datatypes and the declaration of some variables used across the whole module (in particular some maximal values for the dimensioning of module arrays), the allocation of the module variables, the forward integration of the trajectories, the communication between the processors and the I/O subroutines are available in the second file. The

interfaces to the rest of the COSMO code are slim: four calls to the organizing routine of the module and a global switch to activate/deactivate the module.

The module is controlled by a namelist and requires a text file containing the start positions of the trajectories (startfile).

In the following sections the technical details related to the use of the module and its implementation are described. Section 3 provides an overview of all relevant namelist swiches and section 4 explains how the start information of the trajectories should be specified. Reading these two sections should be enough to setup a trajectory computation. In section 5, more information about the output files produced by the trajectory module is provided while section 6 details the current implementation and should ease code comprehension and further extension.

3 Namelist settings

The online trajectory module can be switched on and off by an additional namelist parameter in the namelist group `RUNCTL` named `ltraj`, which is set to `FALSE` by default. The user-defined parameters for the trajectory calculation can be specified in a separate namelist named `INPUT_TRAJ` in a namelist group called `TRAJCTL`. These parameters are described below.

- `istart_mode_traj` : Specification mode of the trajectory start times:
  - `istart_mode_traj= 1` : A comma separated list of start times has to be given through the namelist parameter `hstart_traj` or `nstart_traj`. The list contains the start times in hours after model start (`hstart_traj`) or in model time steps (`nstart_traj`). Trajectories are always started at the same spatial locations for each specified start time. `nstart_traj` has priority over `hstart_traj` if both are specified.
  - `istart_mode_traj = 2` : Trajectories are started repeatedly at fixed time intervals from the same spatial locations. The first start time (in hours after the model start), the last start time (in hours after the model start) and the time interval (in hours) are specified through the namelist parameter `hcomb_start_traj` (a comma separated list of the three elements). The same triplet can be expressed in model time steps through the namelist parameter `ncomb_start_traj`. `ncomb_start_traj` has priority over `hcomb_start_traj` if both are specified.
- `startfile_traj` : Path of the start file. The path can be a relative path and is limited to 100 characters. A description of the start file is provided in section 4.
- `hstop_traj` or `nstop_traj`: Time at which the trajectory calculation stops in hours after model start or in model time steps, respectively. It only has to be specified if the trajectory calculation should stop before the model integration. It has to be a multiple of the model time step. `nstop_traj` has priority over `hstop_traj` if both are specified.
- `ydir_traj` : Directory to which the trajectory output will be written. The directory specification is limited to 100 characters.
- `hinc_out_traj` or `ninc_out_traj`: Output frequency of the trajectory positions (and traced variables) in hours or in number of model time steps respectively. `hinc_out_traj` has to be a multiple of the model time step and `ninc_out_traj` has to be a whole number. If both parameters are specified, `ninc_out_traj` has priority over `hinc_out_traj`.

- `tracevar_traj` : Names of the variables which should be traced along the trajectories. The naming should be consistent with the table in the COSMO source file `src_setup_vartab.f90`, which is also used for the COSMO model output.

An example of the namelist `TRAJCTL` is given in Listing 1.1.

Listing 1.1. Namelist example

```
cat > INPUT_TRAJ << end_input_traj
 &TRAJCTL
  nstop_traj= 4320,
  hstop_traj = 24.0,
  istart_mode_traj = 1,
    hstart_traj = 3.0,6.0,12.0,
    nstart_traj = 540,1080,2160,
    !hcomb_start_traj=
    !ncomb_start_traj=
  startfile_traj = './startfile_case_n.txt',
  ydir_traj = './output_traj',
  !hinc_out_traj = 0.0333,
  ninc_out_traj = 6,
  tracevar_traj = 'T','P','QV','QC','QR','RELHUM',
/
end_input_traj
```

4 Start file

A start file containing the start positions (and times) of the trajectories has to be provided. It is specified by the user via the namelist parameter `startfile_traj` (see section 3). It contains the rotated longitudes, the rotated latitudes and the height and optionally the times at which the trajectories are started.

It is a textfile and it must comply with strict formatting rules.

The first 3 lines are header lines:

- First line: It contains the reference date, i.e. the date at which the COSMO simulation starts. This line is not considered by the program.
- Second line: Description of the file data columns. It should either be `"time lon lat z"` or `"lon lat z"` (without quotes). Only the first four characters of this line will be analyzed by the program and in case the first four characters match the keyword `time`, the program will read the first data column as being the start times and expect the second column to be the rotated longitudes. If the first four characters do not match the keyword `time`, the first data column is assumed to contain the rotated longitudes.
- Third line: It is not relevant for the program. It traditionally contains only `"----"` symbols.

The following lines in the file are content lines. They are formatted in one of these two ways:

- Formatting in case `time` is present
  - first column (1 position): empty
  - second column (4 positions): time
  - third column (1 position): empty
  - fourth column (8 positions): longitudes
  - fifth column (1 position): empty
  - sixth column (7 positions): latitudes
  - seventh column (1 position): empty
  - eighth column (8 positions): height
- Formatting in case `time` is not present
  - first column (1 position): empty
  - second column (8 positions): longitudes
  - third column (1 position): empty
  - fourth column (7 positions): latitudes
  - fifth column (1 position): empty
  - sixth column (8 positions): height

The longitudes have to range between `-180.0` and `180.0` degrees and the latitudes can be comprised between `-90.0` and `90.0` degrees. They are expressed in the same referential than COSMO (i.e. same rotated pole coordinates). Both can have up to 3 decimal positions. The height has to be given in meters AMSL and cannot be bigger than 99999.999 meters. It has up to 3 decimal positions. If time data is provided, it must be in hours after model start and must be smaller or equal to 99.99 hours. It has up to 2 decimal positions. Please note that even if provided the time data is not further used in the program.

The start file is read in an endless loop and all lines (except the 3 header lines) are considered as data.

Make sure that your start file fulfill the format requirements explained above!

A startfile should look similar to the example provided in Listing 1.2.

Listing 1.2. Start file example

```
Reference Date 19870725_0000
time lon lat z
----------------
 10.00    -8.000   -2.500 9000.000
 10.00    -8.000   -2.500 9100.000
 10.00    -8.000   -2.500 9200.000
```

5 Output

For all trajectories, their position as well as the value of the traced variables along them are written to file at the output frequency specified by the user via the namelist parameter `hinc_out_traj` or `ninc_out_traj` (see section 3) in the output directory specified by the user (`ydir_traj`, see section 3). Only NetCDF format is supported.
The output is split in the following way: the trajectories that start together at a given start time are considered as a group. If the group is too large to be contained in one single file, the trajectories are further split over several files, according to their position in the start file. A limit of 2 GB is set as maximal file size in order to ease postprocessing. At all output time steps comprised between the start of the trajectory group and the end of the trajectory simulation (see namelist parameter `hstop_traj` or `nstop_traj` in section 3), the positions and traced variable values for the group of trajectories are written to one (or several) files. The files can thus have different lengths (for different start times).

One or several files `traj_tXXXXXX_pYYY.nc` are thus produced, where `XXXXXX` is the start time of the trajectory group expressed in minutes after the model start and `YYY` is the number of the file in the file set containing the trajectories starting at `XXXXXX` after the model start.

The output files follow the CF conventions. Some global attributes (reference date, duration of the simulation for the group of trajectories, coordinates of the rotated Pole and output frequency) are written to each file. A variable `time` indicates the time elapsed since the model start in seconds. For each instance of `time` (first dimension) and for each trajectory identified by its ID (`id` as second dimension), the longitude, latitude and height (variables `longitude`, `latitude` and `z`) are written as well as the value of the traced variables. For each of these variables, the short name, the long name and the units are provided as attributes.

The header of an output file (here for 13507 trajectories starting after 120 minutes) thus looks like the example provided in Listing 1.3.

Listing 1.3. Output file example

```
netcdf traj_t000120_p001 {
dimensions:
        id = 13507 ;
        time = 541 ;
variables:
        float time(time) ;
                time:standard_name = "time" ;
                time:long_name = "time" ;
                time:units = "seconds" ;
        float longitude(time, id) ;
                longitude:standard_name = "grid_longitude" ;
                longitude:long_name = "rotated longitudes" ;
                longitude:units = "degrees" ;
        float latitude(time, id) ;
                latitude:standard_name = "grid_latitude" ;
```

```
                latitude:long_name = "rotated latitudes" ;
                latitude:units = "degrees" ;
        float z(time, id) ;
                z:standard_name = "height" ;
                z:long_name = "height above mean sea level" ;
                z:units = "m AMSL" ;
        float T(time, id) ;
                T:standard_name = "air_temperature" ;
                T:long_name = "temperature" ;
                T:units = "K" ;
        float P(time, id) ;
                P:standard_name = "air_pressure" ;
                P:long_name = "pressure" ;
                P:units = "Pa" ;
        float U(time, id) ;
                U:standard_name = "grid_eastward_wind" ;
                U:long_name = "U-component of wind" ;
                U:units = "m s-1" ;
        float QV(time, id) ;
                QV:standard_name = "specific_humidity" ;
                QV:long_name = "specific humidity" ;
                QV:units = "kg kg-1" ;
        float QI(time, id) ;
                QI:standard_name = "mass_fraction_of_cloud_ice_in_air" ;
                QI:long_name = "specific cloud ice content" ;
                QI:units = "kg kg-1" ;
        float RELHUM(time, id) ;
                RELHUM:standard_name = "relative_humidity" ;
                RELHUM:long_name = "relative humidity" ;
                RELHUM:units = "%" ;

// global attributes:
                :ref_year = 2011 ;
                :ref_month = 11 ;
                :ref_day = 20 ;
                :ref_hour = 12 ;
                :ref_min = 0 ;
                :ref_sec = 0 ;
                :duration_in_sec = 10800. ;
                :pollon = -171.5 ;
                :pollat = 47.5 ;
                :output_timestep_in_sec = 20. ;
}
```

6 Implementation details

6.1 Workflow overview

An organizing routine `organize_traj` is called four times from `lmorg.f90`:

1. Once at the beginning of the initialization phase with the action `'input'`. The namelist `INPUT_TRAJ` is read as well as the start file. Necessary checks are performed in order to ensure that the specifications made by the user are meaningful. The user can proof the settings by inspecting the `YUSPECIF` file.

2. Once later on during the initialization phase with the action `'init'`. Arrays required for the whole trajectory simulation are allocated. A data structure is filled for all traced variables (see section 6.4). The rank of the eight surrounding processing elements for each processing element is defined in a parallel case. The position of the trajectories are initialized using the information read from the start file. Also, a status flag indicating if the trajectory has to be computed (`alive`), has left the domain (`dead`) or is not yet started (`embryonic`) is initialized using the start times read from or derived from the namelist settings (see section 3). The number of required output files is computed and the split of the trajectories among these files is determined. All NetCDF files are created and the global attributes are written to the files. The variables are defined and their attributes written. The NetCDF files will be kept open for the whole trajectory simulation.

3. Once at the end of each time step, where the other diagnostics are computed, with the action `'compute'`. For each compute time step of the trajectories (i.e. for each time step comprised between the start of the first trajectory and the end of the trajectory computation, both specified by the user via namelist parameters), it is checked for each trajectory if it should be computed or not on each processor. In case it has to be computed, the forward integration is performed (see Miltenberger et al., 2013 for more details) and in case the trajectory leaves the processor domain after its integration, it is added to a buffer for communicating the new position to the relevant neighbor processor. After this has been done for all trajectories, the communication of all trajectories that have to be passed to a neighbor processor is performed. Finally output is performed if the time step corresponds to an output time step.

4. Once at the end of the simulation with the action `'finalize'`. All NetCDF files are closed and the memory used for the trajectory simulation is freed.

6.2 Maximal values for array allocation

Some restrictions have to be set regarding the maximal size of arrays used in the module. All the maximal values are defined as parameters in the module `data_traj.f90`.
In case these maximal values have to be increased, one can simply change the values of the parameters listed below in `data_traj.f90` and test if the model still works on the platform used.

The parameters are:

- nmax_traj_starttime=100: Maximal number of trajectory start times. It means that up to 100 values can be provided by the user through the namelist parameters hstarts_traj or nstarts_traj or that the namelist triplet hcomb_start_traj or ncomb_start_traj does not lead to more than 100 values. This parameter should be increased with caution!
- nmax_traj_startpt=50000: Maximal number of trajectories that can start at a given start time (from different locations). It thus means that a maximum of 100*50000=5 mio. trajectories is imposed. Please note that so far, the module has been tested only with up to about 2 mio. of trajectories. If increasing this parameter or the previous one, intensive testing should be done.
- nmax_traced_var=30: Maximal number of variables that can be traced along the trajectories. This parameter could be increased if more variables should be traced but testing is recommended.
- nmax_out_per_start_t=100: Maximal number of output files per start time. It corresponds to the part YYY in the output filename traj_tXXXXXX_pYYY.nc. Please note that this could not be increased to more than 999 without modifying the naming scheme of the output files. At the moment, we cannot produce more than 100*100=10'000 output files (nmax_traj_starttime times nmax_out_per_start_t).
- nmax_nc_size=$2*2^{30}$: Maximal size of an output file. The file size is currently limited to 2 GB in order to ease post-processing. If increasing this value, some post-processing facilities might fail.
- header_nc_size=$2*2^{20}$: Reserved space for the header in an output file (2 MB). If more information is added to the header in the future, it might be necessary to increase this number.

6.3 Restrictions of use

Currently, the trajectory module cannot be used in combination with a few other model settings:

- Restart (hstart > 0). The restart is not yet implemented for the trajectories.
- Periodic boundary conditions (lperi_x and lperi_y). Actually this setting works but the trajectories themselves are not cycled.

Other configurations are supported.

6.4 Traced variables

A new data type trajtrace_type is defined in data_traj.f90 to hold the needed information to handle properly the traced variables. Most of the information is inherited from the var structure available in src_setup_vartab.f90. It contains the following elements:

- `name`: name of the traced variable
- `levtyp`: level type (GRIB related)
- `ntri`: time range indicator (GRIB related)
- `rank`: rank
- `p4`: pointer to a 4-dimensional field
- `p3`: pointer to a 3-dimensional field
- `idef_stat`: status indicating if the variable has been computed already or if it should be derived from other model variables
- `istag`: staggering flag (`0`: no staggering; `1`: zonal staggering; `2`: meridional staggering; `3`: vertical staggering)
- `units`: units (NetCDF related)
- `sdname`: standard name (NetCDF related)
- `lgname`: long name (NetCDF related)

Only COSMO model variables can be traced. The name of the traced variables must match exactly the name of the corresponding variable in the `var` structure (in `src_setup_vartab.f90`). Only 4-dimensional and 3-dimensional variables in space can be traced. The traced variables must be instantaneous values (i.e. have a time range indicator of 0), since it is not clear what should be traced in case of accumulated values or statistical quantities (mean, min, max). The variables can be on the half-levels or on the full-levels. They can be staggered in zonal direction (`U`) or in meridional direction (`V`). All variables that are valid at each time step in the model (`idef_stat=1` in `src_setup_vartab.f90`), i.e. all prognostic variables and several diagnostics, can be traced. The other model variables (`idef_stat=0` in `src_setup_vartab.f90`) must be derived before being written to the COSMO output. Only a subset of these variables has been implemented for tracing so far:

- `P`: total pressure
- `RELHUM`: relative humidity

Other variables could easily be implemented by making use of the subroutines used in `src_output.f90` (e.g. `potential_vorticity_rho`).

At each output step, the traced variables are interpolated to the trajectory position using a linear interpolation. The computed values for the traced variables as well as the trajectory ID and position (longitude, latitude and height) are gathered on processor 0, which writes the data to file.

6.5 Communication strategy

For each trajectory that leaves a processor, its ID as well as the three components of its position (longitude, latitude and height) are packed in a buffer dimensioned with the 8 neighbors and with the number of trajectories times the four needed pieces of information (ID, longitude, latitude and time). After having looped over all trajectories, actual communication is performed. For this, a "pairwise" non-blocking (`MPI_ISEND` and `MPI_IRECV`) communication strategy has been chosen. Two slices of the buffer for opposite neighbor processors (e.g. Northern and Southern neighbor) are communicated at a time. This is faster than communicating in each direction one after the other and requires less memory than a communication in all directions at a time. This solution is thus a compromise between performance in terms of communication speed and memory consumption. This could easily be changed if desired. The `MPI_IRECV` are pre-posted in order to improve performance.

6.6 Trajectory positions

The trajectory positions in the index space must be determined several times during a trajectory computation time step: to determine the winds at the trajectory position in order to update its position, to interpolate the orography at the trajectory position (in order to check if the trajectory has reached the ground), and finally to interpolate the traced variables to the trajectory location in case output must be performed. This computation is sensitive to the reference system and rounding errors thus occur when changing the domain decomposition. They are very small (typically differences of $O(10^{-9})$ for the longitudes and latitudes, $O(10^{-6})$ for height, $O(10^{-7})$ for traced temperature, $O(10^{-8})$ for traced wind or pressure, $O(10^{-10})$ for the traced humidity variables are observed) and do not affect the physical interpretation of the trajectories.

The required horizontal interpolation combined with the formulation of the lower boundary condition in the model (wind extrapolation) leads to trajectories that intersect the terrain, which is physically impossible. Any trajectory that reaches the ground is then lost for the rest of the simulation. In order to avoid unreasonable loss of trajectories, a trajectory that reaches the ground is reset to 10 meters above ground during the Euler forward integration (see Miltenberger et al., 2013 for more details). A hardcoded switch `ljump` controls this feature and could easily be set to `FALSE`, in order to let the trajectories leave the domain at the ground during the forward integration. Similarly, the value of the hardcoded parameter `hjump` that defines the height at which the trajectories are reset (currently 10 meters above ground) can easily be modified.

7 References

A. K. Miltenberger, S. Pfahl and H. Wernli (2013). An online trajectory module (version 1.0) for the nonhydrostatic numerical weather prediction model COSMO. Geosci. Model Dev., 6, 1989-2004, doi:10.5194/gmd-6-1989-2013.