# Tools related to Meso-NH model

N. Asencio, J. Duron, J. Escobar, D. Gazen, P. Jabouille, I. Mallet

March 21, 2005

## Contents

# 1 Introduction

After initialisation, run of the model or computation of diagnostics, output Meso-NH files can be convert into other formats of files. The present documentation aims at describ the differents tools which can be applied to the binary part of FM files (their suffix is **.lfi**). Most of these tools can be run on the user local computer (Linux PC or HP workstation).

First, the compression tool `lfiz` and the conversion tool `conv2dia` dealing with FM files (synchronous and diachronic) as input and output, are described. The next sections concern tools dealing with other formats than FM: conversions with `lfi2cdf`, `lfi2grb` and `lfi2v5d`. A set of tools for reading diachronic FM files and dealing with diachronic informations is presented: `extractdia`, `mesonh2obs` and `obs2mesonh` (the 2 latest aim at help users to compare MesoNH outputs to observations).

The figure 1 shows when a FM file is either synchronous (contains the values of all the fields corresponding to the same instant of the simulation) or diachronic (contains time series of some fields obtained during the run of the model). Then the figure 2 resumes the tools which can be applied to a FM file according its type, one of the two previous ones.

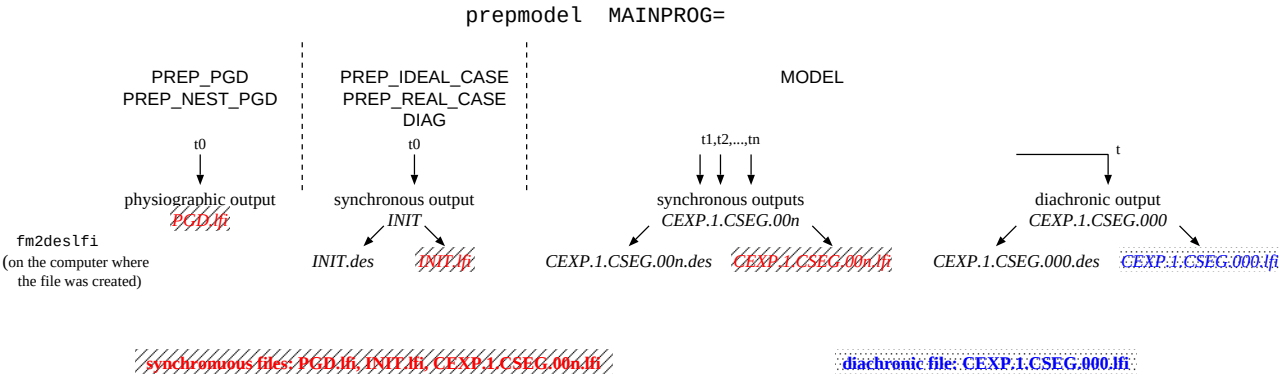


Figure 1: Type of FM files after a MesoNH program

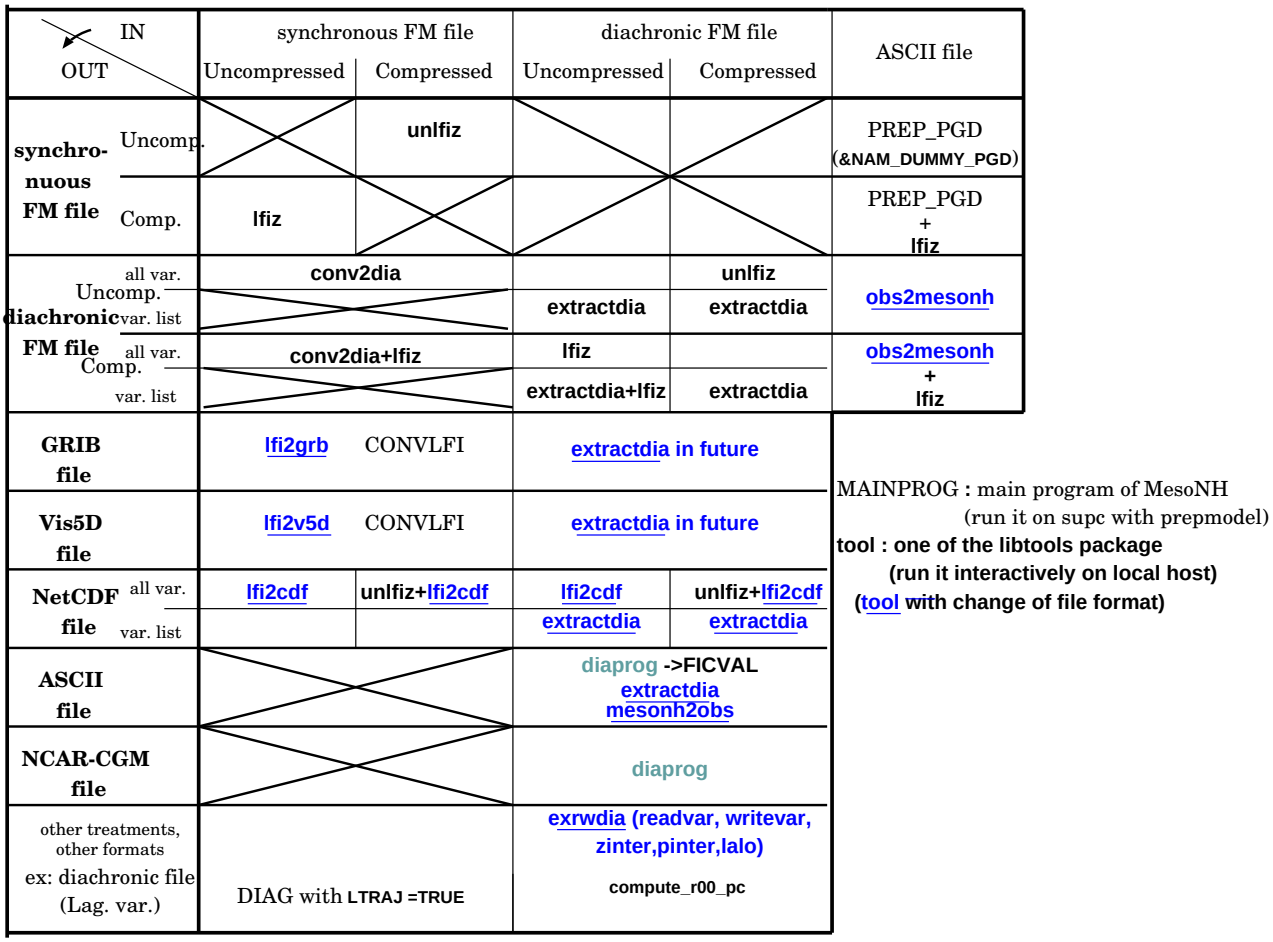


Figure 2: Which tools on FM files?

# 2 Compression of FM files

A specific compression tool has been developed for FM files. This tool, called `lfiz`, was first devoted for files that will be explored by the graphic utility `diaprog`. In fact, it is also used for files used during a simulation (initial and coupling files) to reduce the data storage. Some information of how the compression works is given here, its execution is particularly easy.

## 2.1 `lfiz` tool

The `lfiz` tool works on the binary part (LFI file) of a FM file, synchronous or diachronic. It is a lossy compression tool. The compressed articles are exclusively the 2-dimensional or 3-dimensional `REAL` fields. When dealing with 3D fields the tool works with each 2D plane on every vertical level. The initial values stored with 64-bit `REAL` precision are first converted into 32-bit `REAL` precision and then compressed by mapping the 32-bit real values upon 16-bit integer values (with a possible isolation of extrema values). The better compression is achieved for fields with small value range. For fields with missing value (e.g. 2-dimensional fields with land-sea mask), the extremum value is excluded and the compression is done on significant values of the field. The minimum compression ratio is 4 for each 2D or 3D `REAL` compressed field.

## 2.2 `unlfiz` tool

The `unlfiz` tool will restore the 64-bit `REAL` value size to all the compressed LFI articles. However, each previously compressed article will gain no more than a 32-bit `REAL` precision because of the lossy technique involved above.

## 2.3 Usage

The binary part of the FM file is required in the current directory. To compress the file `myfile.lfi`, you can type:

```
lfiz myfile.lfi
```

This will produce the compressed file `myfile.Z.lfi`

In the same way, to uncompress the file `myfile.Z.lfi`, you can type:

```
unlfiz myfile.Z.lfi
```

The output file `myfile.lfi` is a valid LFI file but the LFI articles previously compressed are 64-bit `REAL` with no more than 32-bit `REAL` precision.

# 3 Conversion of FM synchronous file to diachronic format

Short description is given here, readers must refer to the original documentation on the Meso-NH web site: "TRAITEMENT GRAPHIQUE DES FICHIERS SYNCHRONES PRODUITS PAR LE MODÈLE MESONH, J. Duron".

## 3.1 Synchronous and diachronic formats

The Meso-NH graphic utility (`diaprog`) works on FM files which are on diachronic format. A diachronic FM file is either

- a file produced during the simulation which contain time series of self-documented informations (e.g. file with name CEXP.1.CSEG.000). An information is one of the following:

  - a 3-dimensional, 2-dimensional, 1-dimensional or 0-dimensional field (eventually time-averaged, or compressed in one direction): type CART,

  - a set of vertical profiles at points checking some criteria: type MASK,

  - spectral coefficients obtained by FFT along the X or Y direction: type SPXY,

  - pseudo-observations (ground station: type SSOL; dropsonde: type DRST; radiosonde: type RSPL; airborne radar: type RAPL).

  A diachronic file can contains informations of one or several previous types stored at different time frequency. For a whole description about the diachronic file type, reader must refer to the original documentation on the Meso-NH web site: "CRÉATION ET EXPLOITATION DE FICHIERS DIACHRONIQUES, J. Duron".

or

- a 'pseudo'-diachronic file resulting of the conversion of a synchronous file (e.g. with name CEXP.1.CSEG.00n where n>0). Recall that such a file contains all the pronostic fields of the model at one instant (initial or during the simulation). When converted it is a 'pseudo'-diachronic file, because it contains only one instant and one type of diachronic information (CART). The next subsection presents the conversion tool (named `conv2dia`) to apply to synchronous files, necessary step to use `diaprog` graphic tool.

## 3.2 `conv2dia` tool

The conversion tool works on files produced by the initialisation programs (PREP_PGD, PREP_IDEAL_CASE, PREP_REAL_CASE), the model simulation, or the post-processing program (DIAG). It allows to convert one synchronous file onto one diachronic file, as well as merge several synchronous files with chronological times (outputs of one run, or files initialised from large-scale model) onto one diachronic file.

With `conv2dia.elim` tool, you can choose not to convert all the fields of the input file(s). The pronostic fields at $t - dt$ instant, or at $t$ instant, or any other fields can be eliminated. With `conv2dia.select` tool, you have to indicate the fields to select for conversion. This is done to reduce the size of the output file.

The output file contains informations whose type is CART stored in arrays with size of `(IIU*IJU*IKU)`, `(IIU*IJU)`, `(IIU*IKU)`, or 1.

## 3.3 Example

Only the binary (LFI) part of the input FM files is required in the current directory (split the FM file with the `fm2deslfi` script if not).

All characters typed on keyboard are saved in `dirconv.elim` or `dirconv.select` file, it can be appended and used as input (after being renamed) for the next call of the tool (e.g. `conv2dia.elim < dirconv.elim.ex`).

Below is the example of questions when `conv2dia.elim` is invoked.

```
ENTER NUMBER OF INPUT FM FILES
2
ENTER FM FILE NAME
CEXP.1.CSEG.001
ENTER FM FILE NAME
CEXP.1.CSEG.002
ENTER DIACHRONIC FILE NAME
CEXP.1.CSEG.1-2.dia
DELETION OF PARAMETERS AT TIME t-dt ?  (enter 1)
DELETION OF PARAMETERS AT TIME t ?  (enter 2)
NO DELETION ? (enter 0)
2
Do you want to suppress others parameters ?  (y/n)
y
Enter their names in UPPERCASE (1/1 line)
End by END
DTHCONV
DRVCONV
END
```

# 4 Conversion to NetCDF files

## 4.1 `lfi2cdf` tool

The `lfi2cdf` tool converts the binary part (or LFI file) of a FM file (synchronous or diachronic) into a NetCDF file. All the fields (or more precisely all the LFI articles) contained in the input LFI file are copied to the NetCDF output file with their values unchanged. As a LFI article does not hold any information on the variable, the tool tries to describe the corresponding NetCDF variable by using :

- 3 LFI articles: `IMAX, JMAX,` and `KMAX` if they are available in the LFI input file. These articles may provide the NetCDF dimensions `DIMX, DIMY,` and `DIMZ` of an array variable. If these variables are not available in the input file, the tool treats each array variable as a 1D array.
- a small database implemented as a structure array in the `lfi2cdf` source file `fieldtype.f90`. This array holds the type (`REAL, INTEGER, LOGICAL...`) of every common LFI article. When an article is not present in this database, its name is displayed on `stdout` by the running tool, and the corresponding values are always considered as `REAL` values. A new LFI article type description can be easily added in the `fieldtype.f90` source file and the tool must be then recompiled.

### 4.1.1 Usage

The binary part of the FM file is required in the current directory. The following commands convert a file `myfile.lfi` from LFI to NetCDF:

```
lfi2cdf myfile.lfi
```

or

```
lfi2cdf myfile
```

The output NetCDF file is named: `myfile.cdf`. It can easily be manipulated by NetCDF tools[1] like `ncdump`, `ncview`, or `NCO` operators.

In the same way, you will convert a NetCDF file `myfile.cdf` back to LFI format by typing:

```
cdf2lfi myfile.cdf
```

or

```
cdf2lfi myfile
```

The output LFI file is then named: `myfile.lfi`

## 4.2 `extractdia` tool

The `extractdia` tool converts a diachronic FM file into a NetCDF file after an extraction of a list of fields and an optional extraction of a sub-domain. See the section 5.1.

[1] see freely available NetCDF software at http://www.unidata.ucar.edu/packages/netcdf/software.html

# 5 Dealing with diachronic files

The Meso-NH program of post-processing (DIAG) treats synchronous files from initialization or simulation. For a given need, one wants to work on fields stored in a diachronic file before exploration with `diaprog` or with another graphical tool to possibly compare with observations.

- The `extractdia` tool allows to extract fields from a diachronic file, on the whole domain or on a part of it, to interpole them (horizontal grid and/or vertical grid) and to write them in some other given formats (section 5.1). This program is based on a routine of reading and a routine of writing of diachronic variables: they are the essential source lines to deal with a diachronic file. These 2 routines can be used in the user own program to match his personal needs. An example of such a program `exrwdia.f90` and how to compile it is given in section 5.2.

Some other tools based on the 2 routines of reading and writing are also available to allow easier comparisons with observation data (sections 5.3 and 5.4):

- `mesonh2obs` to get MesoNH field values at given observation points (the format of output file is ASCII),
- `obs2mesonh` to put observation values on a given MesoNH grid (the output file has diachronic FM format), observations can then be plotted with `diaprog` tool.
- `compute_r00_pc` to catenate evolution of Lagrangian tracers back to the model start (as done in DIAG program, see documentation "Lagrangian trajectory and air-mass tracking analyses with MesoNH by means of Eulerian passive tracers", Gheusi and Stein, 2005).

The figure 3 resumes the input and output of these tools.

Remark: for all the following tools, the input diachronic files can be located in another directory than the one in which the tool is invoked (as for `diaprog`). In this case, initialise the following shell variable

```
export DIRLFI=directory_files_diachro
```

Shell links will be automatically performed during the execution and will be removed by the mesonh-shell-tool `rmlink` at the execution end.

## 5.1 Extracte fields, domain, change format with `extractdia` tool

The input file is a FM diachronic file, either a 'true' diachronic one (its name is ended by **.000** and it contains time series of informations obtained during the run of the model), or a 'pseudo'-diachronic one (it is the result of the conversion of a synchronous file, see section 3.1), compressed (with `lfiz`) or not.

The format of the output file is chosen by the user among one of the following:

- a FM DIAChronic file,

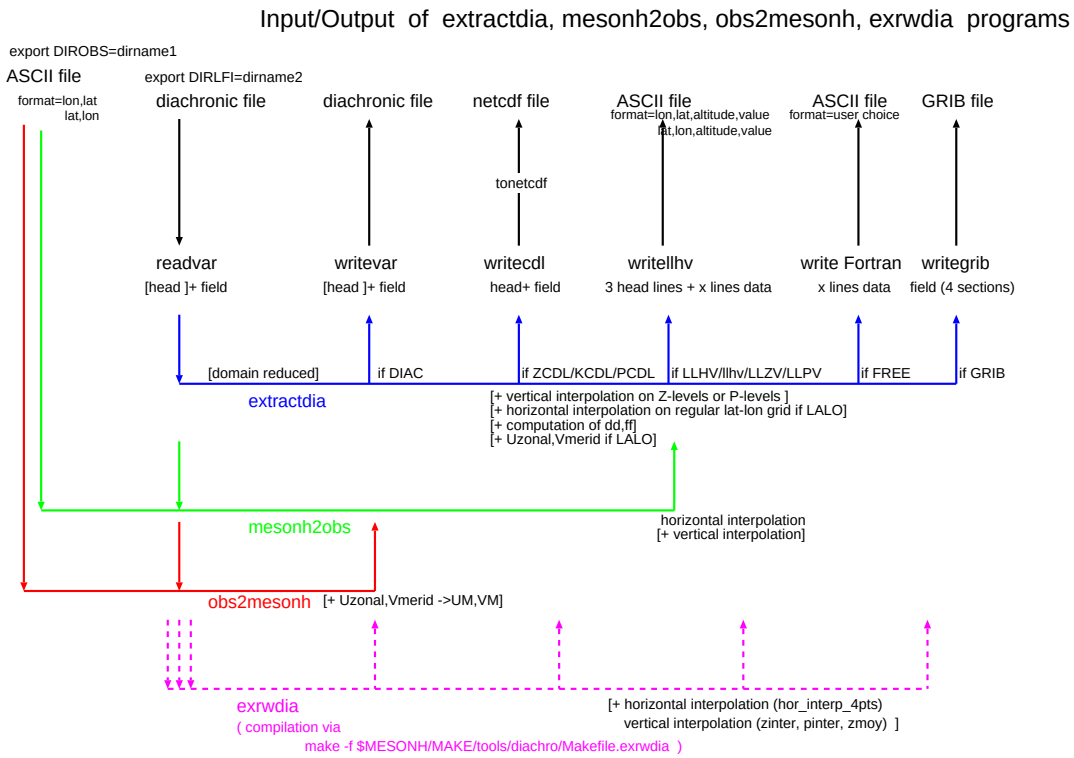


Figure 3:

- an ASCII file with Latitude-Longitude-Height-Value or latitude-longitude-height-value,
- ASCII files with FREE format defined by the user (one file per field),
- a CDL file (converted to NetCDF format at the end of the program, with `ncgen` utility of NetCDF package inside the mesonh-shell-tool `tonetcdf`),
- a GRIB file (in the future),
- a Vis5D file (in the future).

The main program is an interactive one: the name of input diachronic file, the output format, the coordinates of the part of the domain, the name of fields to be read and written are required. All that is typed on keyboard is saved in `direxter.`fmt file, it can be appended and used as input (after renaming it) for the next call of the tool (e.g. `mv direxter.DIAC dirDIAC1 ; extractdia < dirDIAC1`).

The advantages for each output format are the following:

- the wind direction (dd) and wind intensity (ff) could be asked.
- fields are eventually interpolated according output format, first vertically and then horizontally. For vertical interpolation, the user specifies the type of levels (Z or P), the number of levels and their values (in m or in hPa). No vertical interpolation if the type of levels is K (model levels).

For horizontal interpolation on regular grid in longitude and latitude, the program chooses the optimum values computed for the model grid.

If interpolations are required, the wind components are transformed in zonal and meridian components.

These interpolations do not allow interpolation in a required cross-section, the FICVAL file obtained during a `diaprog` session gives this interpolation.

- for the DIAChronic format, the output file will be reduced in size since it contains only some fields on a part of the domain without any interpolations . It can still be plotted with `diaprog`.

- for the LL*V/ll*v format, the fields can be interpolated onto a regular grid in longitude and latitude (LALO option) or can remained on the conformal model grid. (LLZV/llzv option for interpolation on constant altitude levels, LLPV/llpv option for interpolation on constant pression levels LLHV/lhzv option to stay on MesoNH vertical levels). Three header lines give zoom, unit, variable name and temporal informations and are followed by four values on each line.

- for the CDL format, the fields can be horizontally interpolated onto a regular grid in longitude and latitude (LALO option), and eventually vertically on some prescribed levels (ZCDL option for interpolation on constant altitude levels, PCDL option for interpolation on constant pression levels, KCDL option to stay on MesoNH vertical levels). The CDL format is transformed to binary Netdcf format at the end of the program run by the mesonh-shell-tool `tonetcdf`.

- the FREE format allows to get the interpolated values (vertical or horizontal interpolations) without any geographical locations: just values list are available after one header line.

## 5.2 Personal modifications: `exrwdia` program

The `extractdia` program uses 2 routines of reading (`readvar.f90`) and writing (`writevar.f90`) of MesoNH variables as they are stored in diachronic files (that is in 6-dimensional arrays). These 2 routines can be used in your own program: an example of such a program is `exrwdia.f90`. The source code contains extended comments, and there are some examples of computation with the extracted fields (module and direction of components of wind, interpolation on some Z levels, maximum of a 3D field along the vertical direction, vertical average between two Z levels).

The use of this method need to be familiar with the Mesonh specificities: seven grids (Gal-Chen) for the storage of the variables, the U,V wind components are referenced in the Mesonh grid and are different from the Uzonal and Vmeridian components.

### 5.2.1 Routines of reading and writing

A diachronic file contain time series of informations that are self-documented (section 3.1). The self-documentation is provided by the header of the file, which contains a list of pre-defined records, and each field (or information) is stored by several records, the

number of them varies from 8 to 11, according to the type of the information (CART, MASK, SPXY, SSOL, DRST, RSPL or RAPL).

The subroutine `readvar.f90` reads the required field. At the first call, the file is opened, its header is read (the dimensions of the total domain (IMAX, JMAX, KMAX), the orography...) and some characteristics are computed (the conformal coordinates, the map factor...). The required field is then read and available in a 6-dimensional array: XVAR(i,j,k,t,n,p)[2].

The subroutine `writevar.f90` writes the field if the wanted output format is DIAchronic one. If it is the first call the header is written, then the field is stored by the same number of records than when it was read.

The personal code can be inserted in the main program between the call of the two previous subroutines. For the FREE format, the writing code lines are to be written in the main program.

### 5.2.2 Compilation

You have to

- create a sub-directory `src` to put your own source files
- copy `$MESONH/MAKE/tools/diachro/src/EXTRACTDIA/exrwdia.f90` to `src/my_prog.f90` and modify it
- initialize the shell variable `ARCH` which refers to your system and the compiler used (see examples as the suffix of files in `$MESONH/MAKE/conf` directory).
- compile with
  `gmaketools PROG=my_prog OBJS="my_routine1.o my_routine2.o"`
  (the $MESONH/MAKE/tools/diachro/`Makefile.exrwdia` version will be used).

To update the routines dependances directly inside the Makefile:

- initialize the following shell variables:
  - `MNH_LIBTOOLS` which is the directory where the reference sources for the libraries and tools are,
  - `ARCH` which refers to your system and the compiler used (see examples as the suffix of files in `$MNH_LIBTOOLS/conf` directory).
- copy the `$MNH_LIBTOOLS/tools/diachro/Makefile.exrwdia` file in your working directory, rename it to `Makefile`,
- compile with `gmake`

[2]For a whole description of the diachronic file type, reader must refer to the original documentation on the Meso-NH web site: "CRÉATION ET EXPLOITATION DE FICHIERS DIACHRONIQUES, J. Duron".

## 5.3 Compare to observations with `mesonh2obs` tool

### 5.3.1 Input and output

The `mesonh2obs` tool allows to interpolate MesoNH fields at given points (such as points where observation data are available).

The input files are an ASCII file indicated the position of the points by their latitude and longitude coordinates as well as vertical dimension if a vertical profile is required, and one or several diachronic FM file(s) with fields to interpolate at previous points.

Each output file, one for each input FM file, is an ASCII one with six possible options for lines format (LLHV, llhv, LLZV, llzv, LLPV, llpv).

In the input ASCII file, each line indicates the location of one point, all lines have the same format, one of the following :

| | |
|---|---|
| lon lat | and altitudes will be asked by the `mesonh2obs` program |
| lat lon | and altitudes will be asked by the `mesonh2obs` program |
| lon lat altitude(m) | |
| lat lon altitude(m) | |

The output ASCII file contains lines with the same format, one of the following according to the option:

| | | |
|---|---|---|
| lon lat model_level_altitude(m) | option LLHV | |
| lat lon model_level_altitude(m) | option llhv | |
| lon lat altitude(m) | option LLZV | –interpolation routine `zinter.f90` for 3D fields |
| lat lon altitude(m) | option llzv | ” |
| lon lat pression(hPa) | option LLPV | –interpolation routine `pinter.f90` for 3D fields |
| lat lon pression(hPa) | option llpv | ” (pressure variable is read in input FM file) |

### 5.3.2 Usage

The tool is an interactive one: the option for the lines format of the output file, the name of the ASCII file with the location of the observation points are first asked. Then the name of the input diachronic files is asked in a loop, and the name of the fields to interpolate in a second loop:

```
    mesonh2obs << eof
format_output_file # line format of output file (LLHV/llhv/LLZV/llzv/LLPV/llpv)
format_input_file  # LL (lon,lat)ou ll (lat,lon)
altitude_in_input_file # 0 (altitude_in_m on the third colon)/N
if N, number_vertical_levels # number of vertical levels above
                             # each lat,lon points
      list_of_these_levels   # exemple: (in metres or hPa): 500 1500
obs_file                   # name of the Obs file
0                          # control prints (0/1/2/3)
diachronic_file1           # file with fields to be interpolated (without .lfi)
field1_of_diachronic_file1 # field to be interpolated
field2_of_diachronic_file1
END                        # end of extraction in diachronic_file1
diachronic_file2           # file with fields to be interpolated (without .lfi)
fieldi_of_diachronic_file2 # field to be interpolated
```

```
fieldj_of_diachronic_file2
END                          # end of extraction in diachronic_file2
END                          # end of diachronic files list
eof
```

If `field_of_diachronic_file` contains 'AC' string (for ACcumulated precipitation), you can substract values of the same field from a previous diachronic file. Then after line `field('AC')_of_diachronic_file`, answer the question:

```
"Pluie cumulee, voulez-vous faire la difference avec un instant anterieur
(o\/O\/y\/Y\/n\/N) ?"
```

if `Y/O`, indicate the name of `diachronic_file_previous` (without .lfi) in a second supplementary line.

### 5.3.3 Method

The main program retrieves first the $X$ and $Y$ conformal coordinates of each observation point, then for each read field interpolates it vertically if required (vertical profile field with option LLZV, llzv, LLPV or llpv, LLHV, llhv), and finally interpolates horizontally the field and the array of the vertical profile.

## 5.4 Compare to observations with `obs2mesonh` tool

### 5.4.1 Input and output

The `obs2mesonh` tool allows to replace observations on a MesoNH grid. The output file has diachronic FM format: it can be used as input for `diaprog` to plot observations in the same background as MesoNH fields.

The input files are one or several ASCII file(s), each of it contains the values of one type of observation (one per line, all lines have the same format: lon-lat-alt_in_meters-value or lat-lon-alt_in_meters-value), and a diachronic FM file whose grids (spatial and temporal) will be used to replace previous observation values.

The output file is a diachronic file with the orography and the grids of the input diachronic one, each field corresponds to each input observation file. One or two fields are added for each observation field treated: N_field_name for the number of observation averaged in each grid points and if 2D type, ALT_field_name for the altitudes of the observation.

### 5.4.2 Usage

The tool is an interactive one:

```
    obs2mesonh << eof
file_diachronic_with_zs    # initialize MesoNH spatial and temporal grids
0/1/2/3                 # verbosity level
LL                      # format of obs file (LL=lon lat alt value,
                        #                     ll=lat lon alt value)
file1_obs               # name of obs file (undefined value=999.0)
name_new_field1         # name of the obs field  in output file
```

```
unit_new_field1          # free characters string for unit
1D/2D/3D                 # profil of the  obs field
                         # for the 2D case, only K=1 will be initialised
LL                       # format of obs file (LL=lon lat alt value,
                         #                     ll=lat lon alt value)
file2_obs
name_new_field2
unit_new_field2
1D/2D/3D
END                      # closing of output diachronic file
eof
```

### 5.4.3 Method

For each observation read in an input file:
- the MesoNH grid point I,J containing this observation is searching,
- then for observation with 3D profil, the vertical level K is searched (the MesoNH vertical grid (Gal-Chen) at I,J is taken into account); for observation with 2D or 1D profil, the first level K=1 is attributed,
- the value of the observation is stored on grid point (I,J,K).

If several values are stored at the same grid point, arithmetic average of values is done (when unit is $dBz$, the average is computed in $Ze$). If there is no values at a grid point, undefined value is put. The observations whose altitude is below the altitude of the first MesoNH level are stored at level K=1, a warning message is printed in this case.

The wind components are considered zonal and meridian in the observation and are transformed to wind components in the Mesonh grid.

## 5.5 Catenation of Lagrangian trajectory with `compute_r00_pc` tool

### 5.5.1 Input and output

The `compute_r00_pc` tool allows to compute advanced diagnostics. related to Lagrangian tracers activated during the model simulation (`LLG=.TRUE.` in namelist `NAM_CONF`): it is based on the subroutine `compute_r00` used in the DIAG program. See section 2.2 of documentation "Lagrangian trajectory and air-mass tracking analyses with MesoNH by means of Eulerian passive tracers" (Gheusi and Stein, 2005).

The input files are one or several diachronic FM file(s) containing Lagrangian tracers (`LGXM,LGYM,LGZM`) simply converted by `conv2dia` after simulation, or after DIAG (in the latter case, only Lagrangian basic diagnostics were asked: `LTRAJ=.TRUE.` in namelist `NAM_DIAG` with the namelist `NAM_STO_FILE` empty, and additional diagnostic fields can be asked: `CISO='EV'` and `LMOIST_E=.T.` for the example of 5.5.2), and an ASCII file named `compute_r00.nam` with namelist format.

The output file is a diachronic file containing advanced diagnostics: initial coordinates resulting from catenation process, initial values of basic diagnostic fields (present in the input diachronic files) that the Lagrangian parcels had at initial time(s).

### 5.5.2 Usage

The ASCII file `compute_r00.nam` looks as the following:

```
&NAM_STO_FILE CFILES(1)='AR40_mc2_19990921.00d.Z',
              CFILES(2)='AR40_mc2_19990920.12d.Z',
              CFILES(3)='AR40_mc2_19990920.00d.Z',
              CFILES(4)='AR40_mc2_19990919.12d.Z',
              CFILES(5)='AR40_mc2_19990919.00d.Z',
              NSTART_SUPP(1)=3                   /
&NAM_FIELD  CFIELD_LAG(1)='THETAE',
            CFIELD_LAG(2)='POVOM' /
```

The namelist `NAM_STO_FILE` is the same as in the file `DIAG1.nam`. The namelist `NAM_FIELD` indicates the other quantities for which initial values have to be computed.

Then to run the tool,

```
#  initialise the following shell variable (optional if input file
#  is in the current directory):
export DIRLFI=directory_files_diachro
#  initialise the variable ARCH (LXNAGf95 for PC, HPf90 for HP)
export ARCH=LXNAGf95
#  execute
$MESONH/MAKE/tools/diachro/$ARCH/compute_r00_pc
```

### 5.5.3 Method

The structure of the program and the interpolation subroutine (`interpxyz`) are the same as in the DIAG program, the subroutines of reading and writing are those for handling diachronic files (`readvar` and `writevar`).

# 6 Conversion to GRIB or Vis5D files

## 6.1 Presentation

FM synchronous file can be convert into GRIB or Vis5D format. This section aims at describ how the converter works and how use it.

The GRIB (GRId in Binary) format is a standard meteorological one, defined by the WMO. GRIB files can be plotted with METVIEW graphic interface (developped at ECMWF), or R2[3] software.

The Vis5D format is specified for using Vis5D[4] software (following the GNU General Public License): 3 spatial dimensions, time dimension, $5^{th}$ dimension for enumeration of variables. It is rather designed for animation of 3D plotting.

Choice was made to put together the two file formats in a same conversion program because in both cases specificities of Meso-NH grids have to be treated in the same way (horizontally: Arakawa C-grid, vertically: Gal-Chen coordinate $\hat{z}$ following terrain). However, the user has to choose one of the two formats available when running the tool (see section 6.2).

## 6.2 Usage

The interactive tool is called `lfi2grb` or `lfi2v5d` according the wanted output file format, but it runs the same program. Some questions are to be answered to indicate the number and type of vertical levels, the type of horizontal domain, and the name of the variables to write into the output file. All that is typed on keyboard is saved in `dirconv.grb` or `dirconv.v5d` file, it can be appended and used as input (after renaming it) for the next call of the tool (e.g. `mv dirconv.grb dirgrb ; lfi2grb < dirgrb`).

For historical reasons, a program with the same goal of conversion to GRIB or Vis5d has been first developped as a main program of MesoNH, as DIAG program is. This program called **CONVLFI** runs with the MesoNH procedure **prepmodel** and a namelist file `CONVLFI1.nam` (see 6.2.5).

To use the converter after a **DIAG prepmodel** job, the Meso-NH file must remain a synchronous file, not transformed onto a diachronic file: in **prepmodelrc** specify `OUTFILE_TOOLS='fm'` (default is 'conv2dia' to convert with `conv2dia`).

### 6.2.1 `lfi2grb` tool

When `lfi2grb` tool is invoked, you must indicate, after the name of the input file, first the horizontal grid (type, eventually type of interpolation and domain), the vertical grid (type and levels), then the list of the 3-dimensional fields to convert, and the list of the 2-dimensional ones.

For the horizontal grid, you can either keep the one of MesoNH file (cartesien or conformal projection) or interpolate onto a lat-lon regular grid. In the first case, you can replace all the fields on mass points (A-grid) or keep the native grid (C-grid). In the second case, you have to indicate the bounds of the domain with north and south latitudes and west and east longitudes, as well as the type of horizontal interpolation:

[3] used in the GMME/MICADO team at CNRM

[4] home page `http://www.ssec.wisc.edu/~billh/vis5d.html`

nearest-neighbour value or bilinear interpolation with the 4 surrounding values. The resolution of the lat.-lon. grid is automatically initialized with the equivalent value of the grid-mesh where the map scale is minimum. The program also indicates the number of grid points of the Meso-NH domain inside the prescribed lat-lon domain. If there are points of lat-lon domain outside Meso-NH one, the value of the interpolated fields at these points will be a missing one.

The vertical grid can be either the native K levels or pressure levels. In the first case (`K`), all levels are kept and no interpolation is done: the height specified in the GRIB header is the one of the grid without orography. In the second case (`P`), the list of pressure levels is either specified manually or computed using a linear function from user-specified minimum, maximum and increment values. If a prescribed level is below the lower Meso-NH level or above the upper MesoNH level, the value of the field at this level will be a missing one. Otherwise, the value is computed from a linear interpolation in log(P).

The 3-dimensional fields to convert are specified as follows: one field per line with first the name of the record in the input file following by its grib code (tabular character is allowed). Note that no test is done on the value of grib code (GRIB header ISEC1(6)): you choose it to easily identify the field with the software used after the conversion. The end of the list is indicated by the keyword `END`.

The 2-dimensional fields to convert are specified as follows: one field per line with first the name of the record in the input file (it can be a K-level of a 3-dimensional field too), following by its grib code and possibly level indicator and level value (tabular character is allowed). Note that the value of the level indicator (ISEC1(7)) is optional (the default value is 105: 'specified height above ground'). So is the level value (ISEC1(8)), the default value is the altitude of the first mass point of the K-levels. The end of the list is indicated by the keyword `END`.

### 6.2.2 Example of `lfi2grb` use

- to convert onto a GRIB file with horizontal and vertical interpolations in P levels: (all that is typed on keyboard (in *italic* in the example below) is saved in `dirconv.grb`)

```
- ENTER FM synchronous FILE NAME (without .lfi) ?
CEXP.1.CSEG.001d                    <- the input file must be splitted in .des and .lfi
- Horizontal interpolation to lat-lon regular grid?  (Y/y/O/o/N/n)
y
- Type of interpolation?  NEARest-neighbour (default) or BILInear
NEAR
- NSWE target domain bounds (in degrees)?
55. 35. -20. 10.
- Vertical grid:  type K or P ?
P
- Type of vertical grid:  given by linear FUNCTN (default) or MANUALly ?
FUNCTN
- Enter number of P levels ?
5
- Values of the 5 P levels (hPa, from bottom to top):
1000. 850. 700. 500. 300.
- Enter 3D variables to CONVERT (1/1 line, end by END):
MesoNH field name, grib parameter indicator
```

*UM 33*
- next 3D field or END ?
*VM 34*
- next 3D field or END ?
*END*
- Enter 2D variables to CONVERT (1/1 line, end by END):
MesoNH field name, grib parameter indicator, eventually level indicator and level value
*T2M 13 105 2*
- next 2D field or END ?
*THM_K_2 13*
- next 2D field or END ?
*END*
2 fields (3D), and 2 fields (2D) written in CEXP.1.CSEG.001d.GRB

### 6.2.3 `lfi2v5d` tool

When `lfi2v5d` tool is invoked, you must indicate, after the name of the input file, first the vertical grid (type and levels), then the list of the 3-dimensional fields to convert, and the list of the 2-dimensional ones.

No horizontal interpolation is available for the Vis5D format output: all the converted fields are replaced on mass points (A-grid) of the MesoNH grid (cartesien or conformal projection).

The vertical grid can be either the native K levels, altitude levels or pressure levels. In the first case (`K`), all levels are kept and the fields are interpolated on the levels of the lowest point of the domain. In the second and third cases (`Z` and `P`), the list of levels is either specified manually or computed using a linear function from user-specified minimum, maximum and increment values. The value of the field is computed from a linear interpolation in Z or in log(P).

The 3-dimensional fields to convert are specified with one record name per line. The end of the list is indicated by the keyword `END`.

Then the 2-dimensional fields, or a K-level of 3-dimensional fields, to convert are specified in the same way.

### 6.2.4 Example of `lfi2v5d` use

- to convert onto a Vis5D file with vertical interpolation in Z levels:
  (all that is typed on keyboard (in *italic* in the example below) is saved in `dirconv.v5d`)

- ENTER FM synchronous FILE NAME (without .lfi) ?
*CEXP.1.CSEG.001* <- the input file must be splitted in .des and .lfi
- Verbosity level ?
*5*
- File 2D (xz): L2D=T or F ?
*F*
- Vertical grid: type K,Z or P ?
*Z*
- Type of vertical grid: given by linear FUNCTN (default) or MANUALly ?

*FUNCTN*
```
- Vertical grid:  min, max, int (m for Z, hPa for P)?
```
*1500 9000 3000*
```
- Enter 3D variables to CONVERT (1/1 line, end by END):
```
*THM*
```
- next 3D field or END ?
```
*POVOM*
```
- next 3D field or END ?
```
*END*
```
- Enter 2D variables to CONVERT (1/1 line, end by END):
```
*ZS*
```
- next 2D field or END ?
```
*END*
```
2 fields (3D), and 1 fields (2D) written in CEXP.1.CSEG.001d.V5D
```

### 6.2.5 CONVLFI program

The MesoNH program **CONVLFI** allows conversion onto GRIB (the horizontal grid is either the native MesoNH grid (Arakawa C-grid) of the field, the MesoNH mass grid (Arakawa A-grid), the vertical grid is either the native K levels or pressure levels), or conversion onto Vis5D (the horizontal grid is the MesoNH mass grid (A-grid), the vertical grid is either the native K levels without orography, altitude or pressure levels).

The conversion is done with the Meso–NH procedure **prepmodel** used with the **CONVLFI** program and the `CONVLFI1.nam` namelist file. Up to 24 FM files can be treated identically in a single prepmodel job.

A) In the file **prepmodelrc**, the input and output host, directories and login control variables refer to the input and output files as usual. The other control variables to initialize specifically in this file are:

- MAINPROG=CONVLFI
- LOAD_OPT='location_of_v5d_library'
- OUTHOST=name_workstation (for example)
  this allows future use of `vis5d` or `metview` on your local host.

B) In the `CONVLFI1.nam` namelist file, the user must indicate the format type wanted, the number and type of vertical levels, the type of horizontal interpolation on a lat/lon domain as well as the name of the variables to write into the output file:

1. Namelist NAM_OUTFILE:

| Fortran name | Fortran type | default value |
|---|---|---|
| CMNHFILE | array of character (len=28) | none |
| COUTFILETYPE | character (len=3) | none |
| NVERB | integer | 5 |
| LAGRID | logical | .TRUE. |
| CLEVTYPE | character (len=1) | 'P' if COUTFILETYPE='GRB'<br>'K' if COUTFILETYPE='V5D' |
| CLEVLIST | character (len=6) | 'FUNCTN' |
| XVLMIN | real | 10000. if COUTFILETYPE='GRB' |
| XVLMAX | real | 100000. if COUTFILETYPE='GRB' |
| XVLINT | real | 10000. if COUTFILETYPE='GRB' |
| LLMULTI | logical | .TRUE. |

- CMNHFILE: name of the input FM file (from an initialization sequence, or a model simulation, or after diagnostics computation).
- COUTFILETYPE: type of the output file, appended to CMNHFILE to generate the name of the output file.
  - 'V5D'
  - 'GRB'
- NVERB: verbosity level
  - 0 for minimum of prints
  - 5 for intermediate level of prints
  - 10 for maximum of prints.
- LAGRID: switch to interpolate fields on an Arakawa A-grid (mass grid),
  forced to .TRUE. if Vis5D file or horizontal interpolation.
- CLEVTYPE: type of vertical levels in output file,
  - 'P' pressure levels
  - 'Z' z levels (only used for COUTFILETYPE='V5D')
  - 'K'
    if COUTFILETYPE='GRB': native vertical grid of Meso-NH (no interpolation, height specified in GRIB message is the one of the grid without orography),
    if COUTFILETYPE='V5D': native vertical grid of Meso-NH (fields are interpolated on the levels of the lowest point of the domain).
- CLEVLIST: how vertical levels are specified
  - 'MANUAL' number and list of levels specified in the $1^{st}$ free-format part,
  - 'FUNCTN' using a linear function, with the next 3 parameters.
- XVLMIN: minimum value for the vertical grid
  (in m for CLEVTYPE = 'Z', in Pa for CLEVTYPE = 'P'),
- XVLMAX: maximum value for the vertical grid (''),
- XVLINT: increment value for the vertical grid ('').
- LLMULTI: switch to produce a multigrib file (.T.) or monogrib files (.F.), only used for COUTFILETYPE='GRB' (each monogrib file name is composed with the date, the variable name and the level).

2. Free-format part: (number and list of vertical levels)
   This part is only used if CLEVLIST='MANUAL':
   (a) first the number of vertical levels,
   (b) then the list of levels, by increasing values in m if CLEVTYPE = 'Z', or decreasing values in Pa if CLEVTYPE = 'P'
3. Free-format part: (variable names) This part indicates the record name of the variables of the input file to write in the output file. It is specified in two parts:
   (a) between the keywords BEGIN_3D and END_3D: the name of the 3D fields, following by their grib code if COUTFILETYPE='GRB' (separed by tabular character).
   (b) between the keywords BEGIN_2D and END_2D: the name of the 2D fields, following by their grib code, and possibly level indicator and level value if COUTFILETYPE='GRB' (separed by tabular character).

**N.B.:** do not forget the comment line after the keyword BEGIN_3D and BEGIN_2D.

C) Example of namelist file CONVLFI1.nam

- to convert into a Vis5d file:

```
&NAM_OUTFILE  CMNHFILE(1)='T1E20.2.09B24.002',
              CMNHFILE(2)='T1E20.2.09B24.003',
              COUTFILETYPE='V5D',
              CLEVTYPE='Z', CLEVLIST='MANUAL',
              LAGRID=T, NVERB=10 /
15
30.
100.
250.
500.
1000.
1500.
2000.
2500.
3000.
3500.
4000.
4500.
5000.
6000.
8000.

BEGIN_3D
#variables 3D (MesoNH field name)
UM
VM
WM
THM
END_3D
BEGIN_2D
```

```
#variables 2D (MesoNH field name)
ZS
END_2D
```

- to convert into a GRIB file:

```
&NAM_OUTFILE  CMNHFILE(1)='T1E20.2.09B24.002',
              CMNHFILE(2)='T1E20.2.09B24.003',
              COUTFILETYPE='GRB',
              CLEVTYPE='P', CLEVLIST='FUNCTN',
              XVLMAX=100000., XVLMIN=10000., XVLINT=10000.,
              LAGRID=T, NVERB=5 /

BEGIN_3D
#variables 3D (MesoNH field name, grib parameter indicator)
UM  33
VM  34
WM  40
THM 13
END_3D
BEGIN_2D
#variables 2D (MesoNH field name, grib parameter indicator)
ZS 8
END_2D
next lines are ignored
codes example:
MSLP    1
ACPRR   61
INPRR   59
PABSM   1
ALT 6
TEMP    11
REHU    52
RVM 53
RCM 153
RRM 170
RIM 178
RSM 171
RGM 179
RHM 226
RARE    230
HHRE    231
VVRE    232
VDOP    233
POVOM   234
```

## 6.3 Short description of the program

Two main tasks are performed by the program:

1. After the specification of the name of the input file, a 'light' initialization subroutine `init_for_convlfi.f90` is called to initialize the I/O interface, the geometry, dimensions, grids, metric coefficients, times, and to read pressure field.

   According the output grids choosen, extra arrays are allocated for interpolations.

2. Then fields are treated one after another: first 3D fields, then 2D fields.

   In the case of GRIB conversion, fields are interpolated and written one after another (subroutine `code_and_write_grib.f90` called for each horizontal level of each field).

   For Vis5D conversion, fields are interpolated and written all together (subroutine `code_and_write_vis5d.f90` called at the end).

Using a 'light' initialization routine and reading fields name from standard input allows the conversion program not to be dependant of a MesoNH version or program.

## 6.4 Some tips to use Vis5D

See the complete guide for using Vis5D: file README.ps in the Vis5D package.

### 6.4.1 Utilities

(section 5 of README.ps)

- `v5dinfo filename`: shows summary of the v5d file: number and name of the variables, size of the 3-D grid, number of time steps, vertical grid definition and projection definition.

- `v5dstats filename`: shows statistics of the v5d file: minimum value, maximum value, mean value, standard deviation of each variable.

- `v5dedit filename`: edits the header of the v5d file and allows to change it: variables names, variables units, times and dates, projection, vertical coordinate system, low levels.
  *Useful to set the variable's units since they are not set by the program CONVLFI.*

- `v5dappend [-var] filename1 ... targetfile`: joins v5d files together: *useful since the* **prepmodel** *job generates a separate v5d file for each timestep*, `var` indicates list of variables to omit in the target file, the dimensions of 3-D grids must be the same in each input file.

### 6.4.2 Options

(section 6.1 of README.ps)

To call Vis5D: `vis5d file1 [options] file2 [options] ...`
Options can be be specified here when calling, or by pressing the DISPLAY button of the main control panel and then the 'Options' menu.

Options useful to set when calling:
`[-date]` use 'dd month yy' instead of julian 'yyddd' date,

`[-box x y z]` specify the aspect ratio of the 3-D box (default is 2 2 1),
`[-mbs n]` override the assumed system memory size of 32 megabytes (Vis5D tells you value to specify if not enough),
`[-topo file]` use a topography file other than the default EARTH.TOPO

### 6.4.3 Control panel

(section 6.2 of README.ps)
The top buttons control primary functions of Vis5D (see section 6.4.3).
The middle ones control the viewing modes (see section 6.4.3).
The bottom 2-D matrix of buttons contains physical variables on the rows, and types of graphic representation on the columns. To control any type of graphic, click on the button with the left mouse button. A pop-up window appears when clicking with the middle mouse button, and one window to modify colors with the right button (see section 6.4.3).

**Primary functions** (section 6.3 of README.ps)

- SAVE PIC to save the image in a file: first toggle the REVERSE button to reverse black and white, then toggle the SAVE PIC button and choose `xwd` (X Window Dump) format. The file can be visualised with `xv` utility and transformed into `postscript` format.
- GRID#s to display the grid indices instead of latitude, longitude and vertical units along the edges of the box.
- CONT#s, LEGENDS to toggle on or off the isoline values, the colorbar legends.
- BOX, CLOCK to toggle on or off the display of the box and the clock.
- TOP, SOUTH, WEST to set a top (or bottom), a south (or north), a west (or east) view. *Select* SOUTH *to visualise 2D file.*
- SAVE, RESTORE, SCRIPT to save and restore isolines, colors, labels, view (write and read a Tcl script).
- UVW VARS to specify the names of the variables to use to display wind slices and trajectories, several triplets of variables can be used.
- NEW VAR.. to duplicate variables or create new ones by specifying mathematical expressions (formulas use names of existing variables, numbers, arithmetic operations, functions such as $SQRT, EXP, LOG, SIN, COS, TAN, ABS, MIN, MAX$, ex: horizontal wind speed, $spd = SQRT(UM * UM + VM * VM)$ see section 6.13 of README.ps).
- ANIMATE when several time steps: left mouse button: forward, right button: backward, S key: slower, F key: faster.
- STEP when several time steps: left mouse button: one step ahead, middle button: first step, right button: one step back.

- DISPLAY to change the number of displays, the display options (see section 6.4.2), the display parameters (as with the `v5dedit` utility).

**Viewing modes** (section 6.4 of README.ps)
The underlined modes are the most useful (the others are much better displayed with `diaprog` Meso-NH graphics).

- Normal to rotate, zoom and translate the graphics in the 3D window.
- Slice to reposition horizontal and vertical slices.
- Label to create and edit text labels in the 3D window.
- Probe to inspect individual grid values with a cursor moving through the 3D grid.
- Sounding to display a vertical sounding at the location of the moveable cursor.
- Clipping to reposition the six bounding planes of the 3-D box. Select one plane (top, bottom, north, south, west or east) with the middle mouse button, and reposition it with the right mouse button.

**Types of graphic representations** (sections 6.5 to 6.9 of README.ps)
The underlined types are the most useful (the others are much better displayed with `diaprog` Meso-NH graphics).

- Isosurfaces: A 3-D contour surface showing the volume bounding by a particular value of the field (set with the left mouse button). The isosurface is either monocolor or colored according to the values of another variable (right mouse button).
- Slices: Planar cross section (horizontally or vertically) can be moved in this mode. To replace geographic coordinates by grid coordinates, press the "GRID #s" button on the control panel.

  contour line: interval can be changed and min/max values specified in the pop-up window. `-10 (-30,20)` will plot values between -30 and 20 at intervals 10 with negative values dashed. Color can be changed with the right mouse button.

  colored slice: colors can be changed in the pop-up window (with the mouse buttons or arrow keys). Color table is displayed in the 3-D window if the "LEGEND #s" button is selected. To change limits of plotted values, use the keyboard array buttons when in the variable control panel (left and right for limits in the extend of the variable values, up and down for colors inside it).

  wind vector slice: (buttons Hwind1, Vwind1, Hwind2, Vwind2) the scale parameter multiplies the length of vectors drawn (double: 2, half: 0.5), the density parameter controls the number of vectors (between zero and one, 0.5 for one vector of two, 0.25 for one of four).

  wind stream slice: (buttons HStream, VStream) the density parameter controls the number of streamlines (between zero and two).
- Volume rendering: *for powerful workstations..*

### 6.4.4 Advanced use

- generate your own topography file, with the `maketopo.c` program in the `util` directory (see 5 of README.ps).
- Tcl language, to write script (button SCRIPT) or interactively (button INTERP..) (see 6.16 of README.ps).
- external analysis functions written in Fortran, in `userfuncs` directory (see 6.13.3 of README.ps).

## 6.5 State of art

The converter only runs on Linux and VPP. In HP, right compilation options have to be found to use the external library...