# User Manual for the LMD Martian Atmospheric General Circulation Model

**François FORGET, E. Millour, K. Dassas,**
**Christophe HOURDIN, Frédéric HOURDIN,**
**and Yann WANHERDRICK**
*initial version translated by Gwen Davis*
*(LMD)*

**May 31, 2007**

**Draft**

# Contents

Chapter 1

# Introduction

This document is a user manual for the General Circulation Model of the Martian atmosphere developed by the Laboratoire de Météorologie Dynamique of the CNRS in Paris in collaboration with the Atmospheric and Oceanic Planetary Physics sub-department in Oxford. It corresponds to the version of the model available since November 2002, that includes the new dynamic code lmdz3.3 and the input and output data in NetCDF format. The physical part has been available since June 2001, including the NLTE radiative transfer code valid at up to 120 km, tracer transport, the water cycle with water vapour and ice, the ”double mode” dust transport model, and with optional photochemistry and extension in the thermosphere up to 250km.

A more general, scientific description of the model without tracers can be found in *Forget et al.* [1999].

Chapter 2 of this document, to be read before any of the others, describes the main features of the model. The model is divided into two relatively independent parts: (1) The hydrodynamic code, that is shared by all atmospheres (Earth, Mars, etc.) that integrates the fluid mechanics equations in time and on the globe, and (2) a set of Martian physical parameterizations, including, for example, the radiative transfer calculation in the atmosphere and the turbulence mix in the upper layer.

It is followed by a list of references for anyone requiring a detailed description of the physics and the numerical formulation of the parameterizations of the Martian physical part (Chapter 3).

For your **first contact with the model**, chapter 4 guides the user through a practice simulation (choosing the initial states and parameters and visualizing the output files).

The document then describes the programming code for the model, including a user computer manual for compiling and running the model (Chapter 5).

Chapter 6 describes the input/output data of the model. The input files are the files needed to initialize the model (state of the atmosphere at instant $t0$ as well as a dataset of boundary conditions) and the output files are “historical files”, archives of the atmospheric flow history as simulated by the model, the “diagfi files”, the “stats files”, the daily averages etc. The means to edit or visualize these files (editor “ncdump” and the graphics software “grads”) are also explained.

Chapter 8 explains how to run a simulation including the water cycle. Chapter 9 illustrates how to run the model with the photochemical module.

Finally, chapter 10 will help you to use a 1-dimensional version of the model, which may be a simplier tool for some analysis work.

# Chapter 2

# Main features of the model

## 2.1 Basic principles

The General Circulation Model (GCM) calculates the temporal evolution of the different **variables** (listed below) that control or describe the Martian meteorology and climate at different points of a **3D "grid"** (see below) that covers the entire atmosphere.

From an initial state, the model calculates the evolution of these variables, timestep by timestep:

- At instant $t$, we know variable $X_t$ (temperature for example) at one point in the atmosphere.
- We calculate the evolution (the **tendencies**) $(\frac{\partial X}{\partial t})_1$ , $(\frac{\partial X}{\partial t})_2$ , etc. arising from each physical phenomenon, calculated by a **parameterization** of each of these phenomenon (for example, heating due to absorption of solar radiation).
- At the next time step $t + \delta t$, we can calculate $X_{t+\delta t}$ from $X_t$ and $(\frac{\partial X}{\partial t})$. This is the **"integration"** of the variables in time. (For example, $X_{t+\delta t} = X_t + \delta t(\frac{\partial X}{\partial t})_1 + \delta t(\frac{\partial X}{\partial t})_2 + ...$)

**The main task of the model is to calculate these tendencies $(\frac{\partial X}{\partial t})$ arising from the different parameterized phenomenon.**

## 2.2 Dynamical-Physical separation

In practice, the 3D model operates in two parts:
- one **dynamical part** containing the numerical solution of the general equations for atmospheric circulation. This part (including the programming) is common to the Earth and Martian model, and in general for all atmospheres of the terrestrial type.
- a second **physical part** that is specific to the planet in question and which calculates the forced circulation and the climate details at each point.

The calculations for the dynamical part are made on a 3D grid with horizontal exchanges between the grid boxes, whereas the physical part can be seen as a juxtaposition of atmosphere "columns" that do not interact with each other. (diagram 2.1).

The dynamical and physical parts deal with variables of different natures, and operate on grids that are differently constructed. The temporal integration of the variables is based on different numerical schemes (simple, such as the one above for the physical part, and more complicated, the "Matsuno-Leapfrog" scheme for the dynamical part). The timesteps are also different. The physical timestep is $I_{\text{physiq}}$ times longer than the dynamical timestep, as the solution of the dynamic equations requires a shorter timestep than the forced calculation for the physical part.

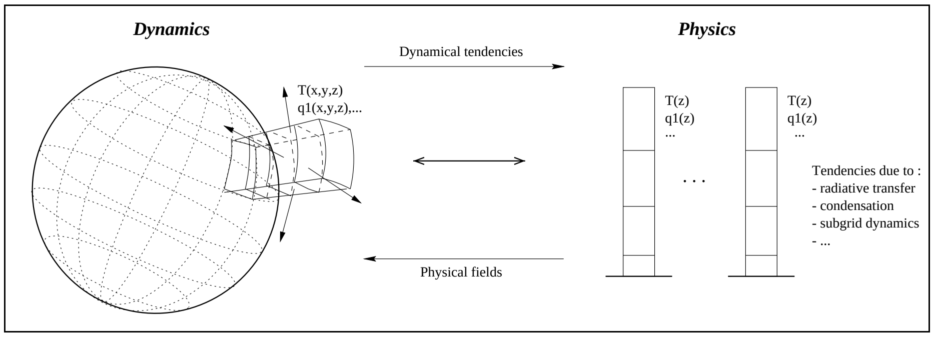


Figure 2.1: Physical/dynamical interface

In practice, the main program that handles the whole model ( `gcm.F`) is located in the dynamical part. When the temporal evolution is being calculated, at each timestep the program calls the following:

1. Call to the subroutine that handles the total tendency calculation $(\frac{\partial X}{\partial t})$ arising from the dynamical part ( `caldyn.F`)

2. Integration of these dynamical tendencies to calculate the evolution of the variables at the following timesteps (subroutine `integrd.F`)

3. Every $I_{\text{physiq}}$ dynamic timestep, a call to the interface subroutine ( `calfis.F`) with the physical model ( `physiq.F`), that calculates the evolution of some of the purely physical variables (ex: Tsurf) and returns the tendencies $(\frac{\partial X}{\partial t})$ arising from the physical part.

4. Integration of the physical variables (subroutine `addfi.F`)

5. Similarly, calculation then integration of the tendencies arising from the horizontal dissipation and the “sponge layer”, etc.

*Remark: The physical part can be run separately for a 1-D calculation for a single column using program* `testphys1d.F`.

## 2.3 Grid boxes:

Examples of typical grid values are 64x48x25, 64x48x32 or 32x24x25. These are the number of points in longitude, latitude and altitude. Thus, we obtain horizontal grid boxes to the order of 200x200 kilometers near the equator.

### 2.3.1 Horizontal grids

Each part uses a different grid. Figure 2.2 shows the numbering of the physical and dynamical grids as well as the different possible positions of the variables on these grids. To identify the coordinates of a variable (at one grid point up, down, right or left) we use coordinates **rlonu , rlatu; rlonv , rlatv** (longitude and latitude, in radians).

For the dynamical grid, we repeat values i=1 at i=IM+1 (periodicity in longitude). As for the values at the poles, they are duplicated IM+1 times. However, on the physical grid, there is only one value at the poles and there is no periodicity in longitude. In practice, the calculations are made for a series of NGRID atmospheric columns with NGRID=IM$\times$(JM-1) +2

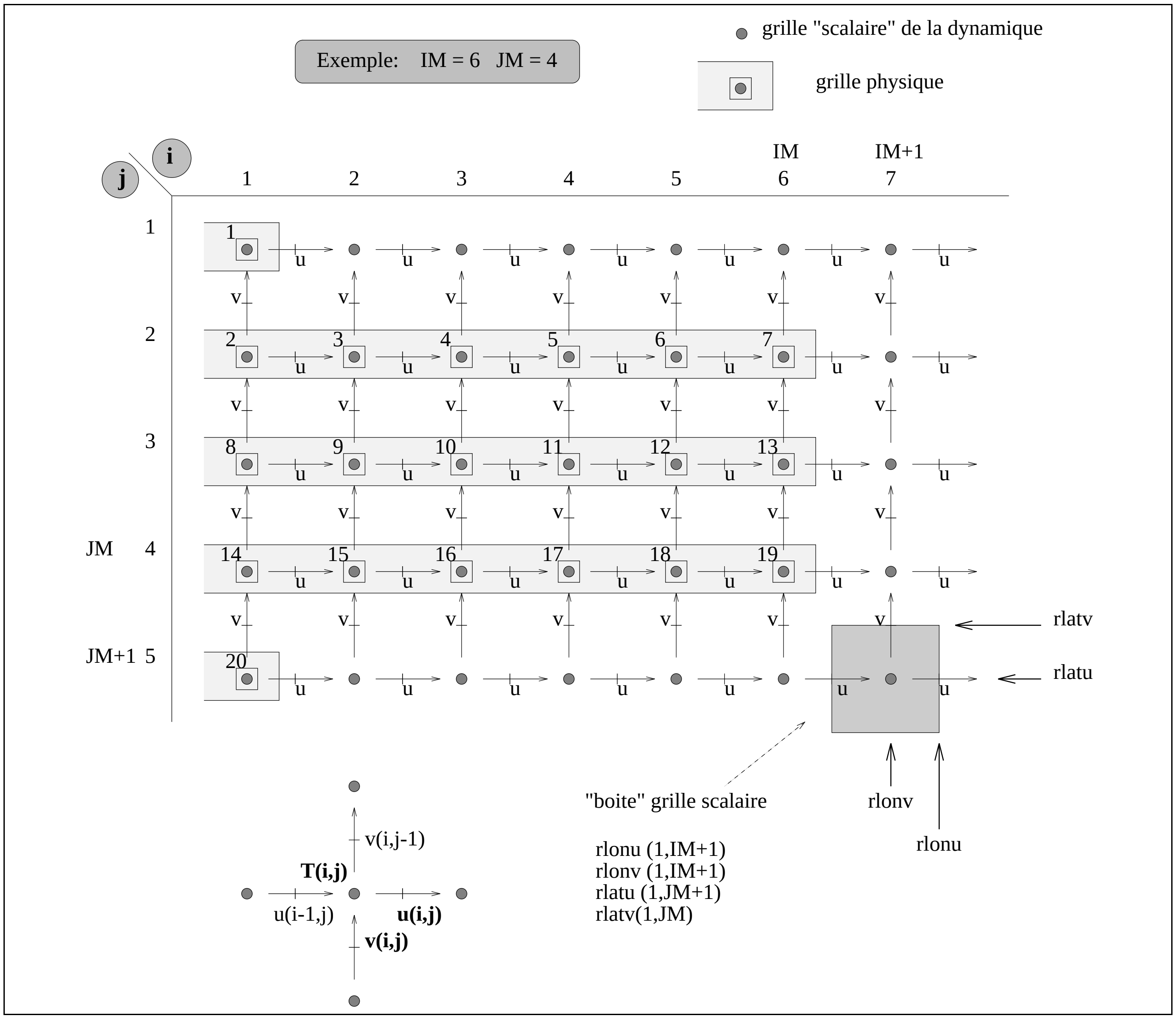


Figure 2.2: Dynamical and physical grids for a 6 $\times$ 7 horizontal resolution. In the dynamical part (but not in the physical part) winds u and v are on a staggered dynamical grid. The other variables are on the dynamical "scalar" grid. The physical part uses the same grid for all the variables, except for the points that are indexed in a single vector containing NGRID=2+(JM-1)$\times$IM points when counting from the north pole. NB. In the Fortran program, the following variables are used: `iim=IM , iip1=IM+1, jjm=JM , jjp1=JM+1`.

### 2.3.2 Vertical grids

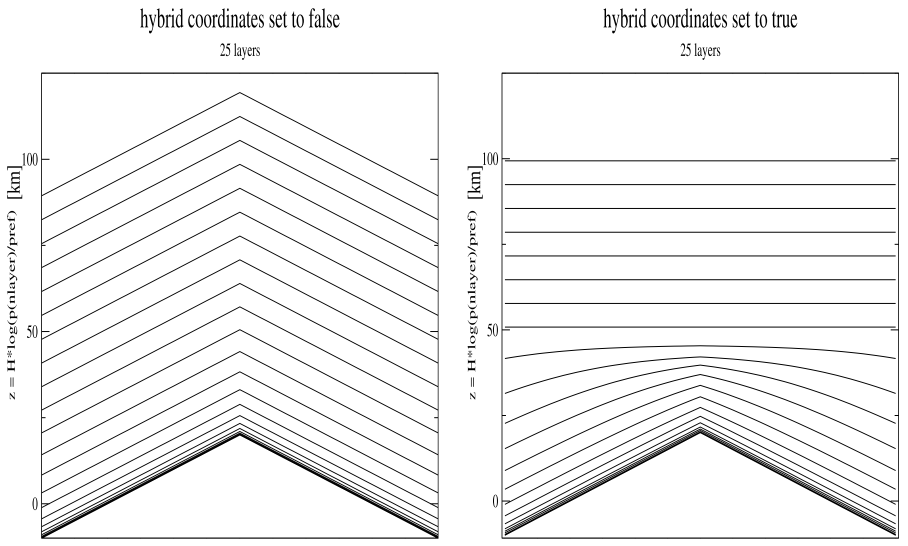

hybrid coordinates set to false
25 layers
hybrid coordinates set to true
25 layers
z = H*log(p(nlayer)/pref) [km]
100
50
0

Figure 2.3: Sketch illustrating the difference between hybrid and non-hybrid coordinates

The GCM was initially programmed using coordinates $\sigma = p/p_s$ (atmospheric pressure over surface pressure ratio) which had the advantage of using a constant domain ($\sigma = 1$ at the surface and $\sigma = 0$ at the top of the atmosphere) whatever the underlying topography. However, it became obvious that these coordinates significantly disturbed the stratospheric dynamical representation as the topography is visible in the coordinate system up to the top of the model. An elegant solution to this problem has been found: using the equivalent hybrid coordinates $\sigma$ nearer the surface and $p$ for higher altitudes. Figure 2.3 illustrates the importance of using these hybrid coordinates compared to the classical coordinates. The distribution of the vertical layers is irregular, to enable greater precision at ground level. In general we use 25 levels to describe the atmosphere to a height of 80 km, 32 levels for simulations up to 120 km, or 50 levels up to thermosphere. The first layer describes the first few meters above the ground, whereas the upper layers describe several kilometers. Figure 2.4 describes the vertical grid variables.

## 2.4 Variables used in the model

### 2.4.1 Dynamical part

The dynamical state variables are the atmospheric temperature, surface pressure, winds and tracer concentrations. In practice, the formulation selected to solve the equations in the dynamical part is optimised using the following less "natural" variables:

- **theta** potential temperature, linked to **T** the temperature by $\theta = T(P/Pref)^{-\kappa}$ with $\kappa = R/C_p$ (note that $\kappa$ is called `kappa` in the dynamical code, and `rcp` in the physical code). We take $Pref = 610$ Pa on Mars.

- **ps** surface pressure.

```
DYNAMIQUE                                                  PHYSIQUE
---------                                                  --------
(variables ap bp)                                          (niveaux de pression)


ap(llm+1)=0,bp(llm+1)=0  ***********************           plev(nlayer+1)=0

aps(llm),bps(llm)        .. llm-1 ........  nlayer         play(nlayer)

ap(llm),bp(llm)          ***********************           plev(nlayer)

aps(llm-1),bps(llm-1)    .. llm-1 ........  nlayer-1       play(nlayer-1)

ap(llm-1),bp(llm-1)      ***********************           plev(nlayer-1)
                                  :
                                  :
                                  :
                                  :

aps(2), bps(2)           ..... layer 2 ...........         play(2)

ap(2), bp(2)             ************************          plev(2)

aps(1), bps(1)           ..... layer 1 ...........         play(1)

ap(1)=1                  ******** surface *******          plev(1)= Psurf
```

Figure 2.4: Vertical grid description of the `llm (or nlayer)` atmospheric layers in the programming code (`llm` is the variable used in the dynamical part, and `nlayer` is used in the physical part). Variables `ap, bp` and `ap_s bp_s` indicate the hybrid levels at the interlayer levels and at middle of the layers respectively. Pressure at the interlayer is $Plev(l) = ap(l) + bp(l) \times Ps$ and pressure in the middle of the layer is defined by $Play(l) = aps(l) + bps(l) \times Ps$, (where $Ps$ is surface pressure). Sigma $\sigma$ coordinates are merely a specific case of hybrid coordinates such that $aps = 0$ and $bps = P/Ps$. Note that for the hybrid coordinates, $bps = 0$ above $\sim 50$ km, leading to pressure levels. The user can choose whether to run the model using hybrid coordinates or not by setting variable `hybrid` in run.def to True or False.

- **masse** the atmosphere mass in each grid box.

- **ucov** and **vcov** the covariant meridional and zonal waves. These variables are linked to the ”natural” winds by `ucov = cu * u` and `vcov = cv * v`, where `cu` and `cv` are constants that only depend on the latitude.

- **q01, q02**, etc. the tracer mixing ratio in the atmosphere (typically kg/kg).

**ucov** and **vcov**, “vectorial” variables, are stocked on “scalari” grids u and v respectively, in the dynamical part. (see section 2.2). **theta**, **q01**, **ps**, **masse**, “scalar variables”, are stocked on the “scalar” grid of the dynamical part.

### 2.4.2 Physical part

In the physical part, the state variables of the dynamical part are transmitted via an interface that interpolates the winds on the scalar grid (that corresponds to the physical grid) and transforms the dynamical variables into more “natural” variables. Thus we find winds **u** and **v** (m.s$^{-1}$), temperature **T** (K), the pressure field in the middle of the layers **play** (Pa) and at the level of the interlayers **plev** (Pa), tracers **q01, q02**, etc. (kg/kg).

Furthermore, the physical part handles the evolution of the purely physical state variables:

- **co2ice** $CO_2$ ice on the surface (kg.m$^{-2}$)

- **tsurf** surface temperature (K),

- **tsoil** temperature at different layers under the surface (K),

- **emis** surface emissivity,

- **q2** wind variance, or more precisely the square root of the turbulent kinetic energy.

- **qsurf01, qsurf02**, etc. tracer budget on the surface (kg.m$^{-2}$).

### 2.4.3 How tracers are handled

The model may include different types of tracers:

- dust particles, which may have several modes

- chemical species which depict the chemical composition of the atmosphere

$$\begin{aligned}
&co2 = nqchem\_min\\
&co = nqchem\_min + 1\\
&o = nqchem\_min + 2\\
&o(1d) = nqchem\_min + 3\\
&o2 = nqchem\_min + 4\\
&o3 = nqchem\_min + 5\\
&h = nqchem\_min + 6\\
&h2 = nqchem\_min + 7\\
&oh = nqchem\_min + 8\\
&ho2 = nqchem\_min + 9\\
&h2o2 = nqchem\_min + 10\\
&n2 = nqchem\_min + 11\\
&ar = nqchem\_min + 12
\end{aligned}$$

If you choose not to handle dust particles (general case), then nqchem_min=1

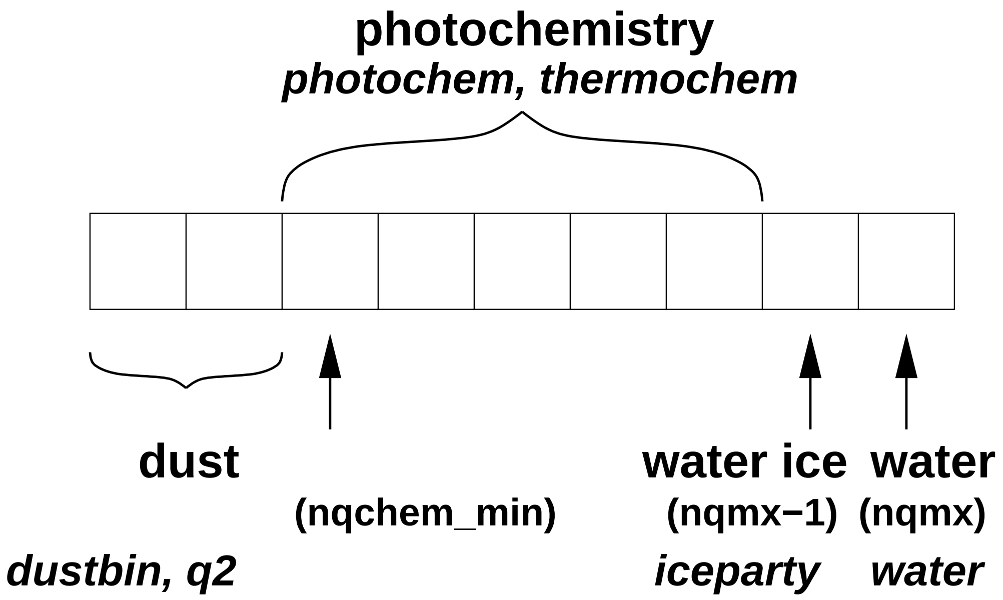


Figure 2.5: tracers management

- water vapor and water ice particles

  h2o corresponds to the nqmx tracer (nqmx is chosen when compiling with the option `-t nqmx`)

  If the option ”ice” is chosen, water ice corresponds to the (nqmx-1)th tracer.

These tracers are managed with one unique vector, by using different parameters. Figure 2.5 describes this vector. The inclusion of these tracers in the calculation can be controled by setting coresponding options in the `callphys.def` file (see Section 6.2.2). The combination of theses options lead to including all or only part of the tracers.

# Chapter 3

# The physical parameterizations of the Martian model: some references

## 3.1 General

The Martian General Circulation Model uses a large number of physical parameterizations based on various scientific theories and some generated using specific numerical methods.

A list of these parameterizations is given below, along with the most appropriate references for each one. Most of these documents can be consulted at: `http://www.lmd.jussieu.fr/mars.html`.

### General references:

Two documents attempt to give a complete scientific description of the current version of the GCM (a version without tracers):

- *Forget et al.* [1999] (article published in the JGR)
- "Updated Detailed Design Document for the Model" (ESA contract, Work Package 6, 1999, available on the web) which is simply a compilation of the preceding article with a few additions that were published separately.

## 3.2 Radiative transfer

The radiative transfer parameterizations are used to calculate the heating and cooling ratios in the atmosphere and the radiative flux at the surface.

### 3.2.1 $CO_2$ gas absorption/emission:

**Thermal IR radiation**

( `lwmain`)

- New numerical method, solution for the radiative transfer equation: *Dufresne et al.* [2005].
- Model validation and inclusion of the "Doppler" effect (but using an old numerical formulation): *Hourdin* [1992] (article).

- At high altitudes, parameterization of the thermal radiative transfer (`nltecool`) when the local thermodynamic balance is no longer valid (e.g. within 0.1 Pa) : Lopez-Valverde et al. [2001] : Report for the ESA available on the web as: “CO2 non-LTE cooling rate at 15-um and its parameterization for the Mars atmosphere”.

**Absorption of near-infrared radiation**

( `nirco2abs`)

- *Forget et al.* [1999]

### 3.2.2 Absorption/emission and diffusion by dust:

**Dust spatial distribution**

( `dustopacity`)

- Vertical distribution and description of “MGS” and “Viking” scenarios in the ESA report *Mars Climate Database V3.0 Detailed Design Document* by Lewis et al. (2001), available on the web.
- For the “MY24” scenario, dust distribution obtained from assimilation of TES data is used (and read via the `readtesassim` routine).

**Thermal IR radiation**

( `lwmain`)

- Numerical method: *Toon et al.* [1989]
- Optical properties of dust: *Forget* [1998]

**Solar radiation**

( `swmain`)

- Numerical method: *Fouquart and Bonel* [1980]
- Optical properties of dust: see the discussion in *Forget et al.* [1999], which quotes *Ockert-Bell et al.* [1997] and *Clancy and Lee* [1991].

## 3.3 Subgrid atmospheric dynamical processes

### 3.3.1 Turbulent diffusion in the upper layer

( `vdifc`)

- Implicit numerical scheme in the vertical: see the thesis of Laurent Li (LMD, Université Paris 7, 1990), Appendix C2.
- Calculation of the turbulent diffusion coefficients: *Forget et al.* [1999].

### 3.3.2 Convection

( `convadj`)
See *Hourdin et al.* [1993]

### 3.3.3 Effects of subgrid orography and gravity waves

( `calldrag_noro`, `drag_noro`)
See *Forget et al.* [1999] and *Lott and Miller* [1997]

## 3.4 Surface thermal conduction

(`soil`)
Thesis of Frédéric Hourdin (LMD, Université Paris 7, 1992) : section 3.3 (equations) and Appendix A (Numerical scheme).

## 3.5 $CO_2$ Condensation

In *Forget et al.* [1998] (article published in Icarus):
- Numerical method for calculating the condensation and sublimation levels at the surface and in the atmosphere ( `newcondens`) explained in the appendix.
- Description of the numerical scheme for calculating the evolution of $CO_2$ snow emissivity (`co2snow`) explained in section 4.1

## 3.6 Tracer transport and sources

- "Van-Leer" transport scheme used in the dynamical part ( `tracvl` and `vlsplt` in the dynamical part): *Hourdin and Armengaud* [1999]
- Transport by turbulent diffusion (in `vdifc`), convection (in `convadj`), sedimentation ( `sedim`), dust lifting by winds ( `dustlift`) : see note "Preliminary design of dust lifting and transport in the Model" (ESA contract, Work Package 4, 1998, available on the web).
- Dust lifting by "Dust devils" ( `dustdevil`) *Rennò et al.* [1998].
- Dust transport by the "Mass mixing ratio / Number mixing ratio" method for grain size evolution: article by F. Forget in progress
- **Watercycle**, see *Montmessin et al.* [2004]
- **Chemistry**, see *Lefèvre et al.* [2004]

## 3.7 Thermosphere

- See *Angelats i Coll et al.* [2005] and *González-Galindo et al.* [2005]

# Chapter 4

# Running the model: a practice simulation

This chapter is meant for first time users of the LMD model. As the best introduction to the model is surely to run a simulation, here we explain how to go about it. All you will need are files necessary to build the GCM (all are in the `LMDZ.MARS` directory) as well as some initial states to initiate simulations (see below).
Once you have followed the example given below, you can then go on to change the control parameters and the initial states as you wish. A more detailed description of the model's organization as well as associated inputs and outputs are given in sections 5 and 6.

## 4.1 Installing the model

- Copy the basic model directory LMDZ.MARS into your files (the contents of this directory are described in chapter 5).

- Set the environment variables for the model:

  **LMDGCM** Directory path where you have stored the model (full path)

  ```
  setenv  LMDGCM $PATH1/LMDZ.MARS
  ```

  **LIBOGCM** Directory path (you should create this directory: `libo` for example; if that directory does not exist then `makegcm` will create it) where the object libraries will be stored when you compile the model with `makegcm`.

  ```
  setenv  LIBOGCM $PATH2/libo
  ```

- Install NetCDF and set attribute environment variables **NCDFINC** and **NCDFLIB**:

  The latest version of the NetCDF package is available on the web at the following address:

  ```
  http://www.unidata.ucar.edu/packages/netcdf/faq.html#howtoget
  ```

  This package contains everything needed to create object library *libnetcdf.a*, the necessary ıinclude files (.h), and to compile the basic NetCDF software, *ncdump and ncgen*.

To ensure that during compilation, the model can find the library and include files that correspond to the type of machine used, you should declare environment variables **NCDFINC** and **NCDFLIB**.

**NCDFLIB** Name of the directory containing the object library (libnetcdf.a) and **NCDFINC** name of the directory containing the NetCDF include files (netcdf.inc) for the different platforms.

```
setenv NCDFINC $PATH3/include
setenv NCDFLIB $PATH3/lib
```

- Install GrAdS

For people working at LMD, thanks to the brilliant Laurent Fairhead, all you need to do to access GrAdS and its environment, is to add the following to your *.cshrc* (this should already be in *.env_soft*):

```
source /distrib/local/grads/grads.env
```

- Copy compilation script **makegcm** (into $HOME/bin for example). This script is available in

```
$PATH1/LMDZ.MARS/
```

(or ask the LMD).

- Finally, make sure that you have access to all the executables needed for using the model and remember to set the corresponding pathes.

  - UNIX function ımake
  - Fortran compiler f90
  - ncdump
  - grads

## 4.2 Compiling the model

- Example 1: Compiling the Martian model at grid resolution 64x48x25 for example, type (in compliance with the manual for the makegcm function given in section 5.4)

```
makegcm -d 64x48x25 -p mars gcm
```

You can find executable **gcm.e** (the compiled model) in the directory where you ran the makegcm command.

- Example 2: Compiling the Martian model with 2 tracers (water vapour and ice):

```
makegcm -d 64x48x25 -t 2 -p mars gcm
```

- Example 3: Compiling the the Martian model to check for table overflow (debugging: warning, the model compiles very slowly!):

```
makegcm -d 64x48x25 -p mars -O "-C" gcm
```

## 4.3 Input files (initial state )

**-** In directory

```
 $PATH1/LMDZ.MARS/deftank
```

you will find parameter file **run.def** (described in section 6.2) needed for the model. For the Martian model, you will also need files **callphys.def** and **z2sig.def**. The parameters are attributed using the usual values, and we recommend running the first simulation without changing them.

**-** Copy files **start.nc** and startfi.nc (described in section 6.2) to the same directory. You can extract such files from **start_archive** ‘banks of initial states’ (i.e. files which contain collections of initial states from stndard scenarios and which can thus be used to check if the model is installed correctly) stored on the LMD website at `http://www.lmd.jussieu.fr/~forget/datagcm/Starts`. See section 4.8 for a description of how to proceed to extract **start** files from **start_archives**.

## 4.4 Running the model

Move program **gcm.e** and the input files into the same directory, then type:

```
gcm.e
```

You might also want to keep all messages and diagnotics written to standard output (i.e. the screen). You should then redirect the output to some file, e.g. `gcm.out`:

```
gcm.e > gcm.out
```

## 4.5 Visualizing the output files

If you have never used the graphic software **GrAds**, we strongly recommend spending half an hour to familiarize yourself with it by following the demonstration provided for that purpose. The demo is fast and easy to follow and you will learn the basic commands. To do this read file

```
/distrib/local/grads/sample
```

For example, to visualize files `diagfi.nc` and `stats.nc`
NetCDF files `diagfi.nc` and `stats.nc` can be accessed directly using GrAdS thanks to utility program gradsnc, (the user does not need to intervene).

To visualize the temperature card in the 5th layer using file `diagfi.nc` for example:

**-** Visualization using GrAdS:

`grads` *return*

*return* (opens a landscape window)

`ga-> sdfopen diagfi.nc`

`ga-> query file` (displays info about the open file, including the name of the stored variables. Shortcut: *q file*)

`ga-> set z 5` (fixes the altitude to the 5th layer)

`ga-> set t 1` (fixes the time to the first stored value)

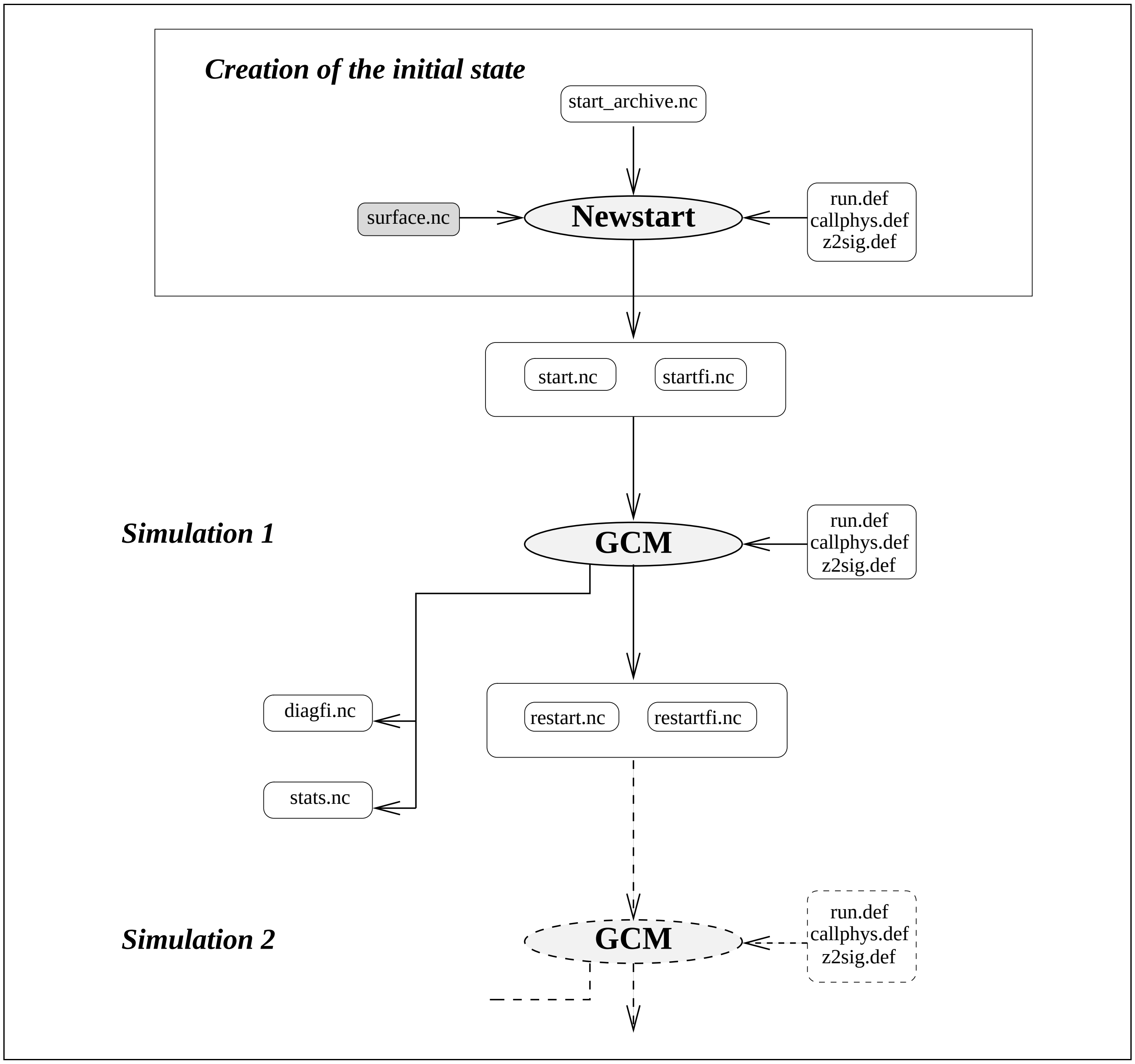


Figure 4.1: Input/output data

`ga-> query dims` (indicates the fixed values for the 4 dimensions. Shortcut: *q dims*)

`ga-> display temp` (displays the temperature card for the 5th layer and for the first time value stored. Shortcut: *d T*)

`ga-> clear` (clears the display. Shortcut: *c*)

`ga-> set gxout shaded` (not a contour plot, but a shaded one)

`ga-> display temp`

`ga-> set gxout contour` (returns to contour mode to display the levels)

`ga-> display temp` (superimposes the contours if the clear command is not used)

## 4.6 Resuming a simulation

At the end of a simulation, the model generates **restart** files that contain the final state of the model. as shown in figure 4.1, these files (of same format as the start files) can be used as initial states for a new simulation.

The **restart** files just need to be renamed:

```
mv restart.nc start.nc
mv restartfi.nc startfi.nc
```

and running a simulation with those will in fact resume it from where the previous run had ended.

## 4.7 Chain simulations

In practice, we recommend running a chain of simulations lasting several days or longer (or hundreds of days at low resolution).

To do this, a script named `run0` is available in

```
$PATH1/LMDZ.MARS/deftank
```

Utilization:

- Set the length of each simulation in `run.def`
- Set the maximum number of simulations in the header of `run0`
- Copy start files `start.nc startfi.nc` and rename them `start0.nc startfi0.nc`.
- Type: `run0`

`run0` runs a series of simulations that generate the indexed output files (e.g. `start1, startfi1, diagfi1`, etc.) including files `lrun1, lrun2`, etc. containing the redirection of the display and the information about the run.

*NOTE:* to restart a series of simulations after a first series (for example, starting from `start5 and startfi5`), just write the index of the initial files (e.g. 5) in the file named `num_run`. If `num_run` exists, the model will start from the index written in `num_run`. If not it will start from, `start0 and startfi0`.

*NOTE*: A script is available for performing annual runs with 12 seasons at $30^o$ solar longitude as it is in the database (script `run_mcd`, also found in directory `deftank`). This script functions with script run0. Just set the number of simulations to 1 in run0. Then copy run.def into run.def.ref and set nday to 9999 in this file. To start from startN.c, edit the file run_mcd and comment (with a #) the N months already created and describe N in `num_run`. Then run `run_mcd`.

## 4.8 Creating and modifying initial states

### 4.8.1 Using program “newstart”

Several model parameters (for example, the dust optical depth) are stored in the initial states (NetCDF files `start` and `startfi` ). To change these parameters, or to generally change the model resolution, use program **newstart**.

This program is also used to create an initial state. In practice, we usually reuse an old initial state, and modify it using `newstart` .

Like the GCM, program **newstart** compiles to the required grid resolution. For example:

```
makegcm -d 64x48x25 -p mars newstart
```

Then run

```
newstart.e
```

The program then gives you two options:

```
 A partir de quoi souhaitez vous creer vos etats initiaux ?
     0 - d un fichier start_archive
     1 - d un fichier start
```

- - Option “1” allows you to read and modify the information needed to create a new initial state from the files `start.nc, startfi.nc`
- - Option “0” allows you to read and modify the information needed to create a new initial state from file `start_archive.nc` (whatever the `start_archive.nc` grid resolution is).

If you use tracers, make sure that they are taken into account in your start files (either start or start_archive).

For Option “0”, the `start_archive.nc` you can find in `$PATH1/LMDZ.MARS/deftank` contains 12 martian seasons.

Then reply to the questions in the scroll menu. These questions allow you to modify the initial state for the following parameters.

```
 First set of questions:
 Modifications of variables in tab_cntrl:
 ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
   day_ini : Jour initial (=0 a Ls=0)
   z0 :  surface roughness (m)
   emin_turb :  energie minimale
   lmixmin : longueur de melange
   emissiv : Emissivite du sol martien
   emisice : Emissivite des calottes
   albedice : Albedo des calotte
   iceradius : mean scat radius of CO2 snow
   dtemisice : time scale for snow,
     . ' metamorphism
        tauvis : profondeur optique visible,
     . ' moyenne
   obliquit : planet obliquity (deg)
   peri_day : perihelion date (sol since Ls=0)
   periheli : min. sun-mars dist (Mkm)
   aphelie  : max. sun-mars dist (Mkm)
```

```
 Second set of questions :
   flat : no topography ("aquaplanet")
   bilball : albedo, inertie thermique uniforme
   coldspole : sous sol froid et haut albedo au pole sud
   q=0 : traceurs a zero
   ini_q : traceurs initialises pour la chimie
   ini_q-H2O : idem, sauf le dernier traceur (H2O)
   ini_q-iceH2O : idem, sauf ice et H2O
   watercapn : H20 ice sur la calotte permanente nord
   watercaps : H20 ice sur la calotte permanente sud
   wetstart  : start with a wet atmosphere
   iqset     : give a specific value to tracer iq
   isotherm : Temperatures isothermes et vents nuls
   co2ice=0 : elimination des calottes polaires de CO2
   ptot : pression totale
```

Program **newstart.e** creates files `restart.nc` , `restartfi.nc` , that you generally need to rename (for instance rename them in start0.nc and startfi0.nc if you want to use run0 or run_mcd, starting with season 0 - rename them in start.nc and startfi.nc if you just want to perform one run with gcm.e).

### 4.8.2 Creating the initial start_archive.nc file

Archive file `start_archive.nc` is created from files `start.nc`, `startfi.nc` by program **start2archive**. Program **start2archive** compiles to the same grid resolution as the start.nc and startfi.nc grid resolution. For example:

```
makegcm -d 64x48x25 -p mars start2archive
```

Then run `start2archive.e`

You now have a `start_archive.nc` file for one season that you can use with newstart. If you want to cumulate other initial states for other seasons, rerun start2archive.e with the start.nc and startfi.nc corresponding to your new season. The new initial state will automatically be added to the `start_archive.nc` file present in your directory.

### 4.8.3 Changing the horizontal or vertical grid resolution

For example, to create initial states at grid resolution $32\times24\times25$ from NetCDF files `start` and `startfi` at grid resolution $64\times48\times32$ :

- Create file `start_archive.nc` with **start2archive.e** compiled at grid resolution $64\times48\times25$ using **old file** `z2sig.def` **used previously**
- Create files `newstart.nc`, `newstartfi.nc` with **newstart.e** compiled at grid resolution $32\times24\times25$, using **new file** `z2sig.def`

If you want to create starts files with tarcers for 50 layers using a start_archive.nc file done for 32 layers, do not forget to use the `ini_q` option in newstart in order to correctly initialize tracers value for layer 33 to layer 50. You just have to answer yes to the question on thermosphere initialization if you want to initialize the thermosphere part only (l=33 to l=50), and no if you want to initialize tracers for all layers (l=0 to l=50).

# Chapter 5

# Programming organization and compilation

The LMD model is organized in a basic directory. This directory is associated with environment variable **LMDGCM**. The example below shows the attribution of this variable using Cshell in UNIX for the model reference version at the LMD.

```
setenv LMDGCM /users/lmdz/SOURCES/LMDZ.MARS
```

Here is a brief description of the directory contents:

```
LMDZ.MARS/


  libf/  All the model FORTRAN Sources (.F for cpp)
         and  include files (.h) organised in sub-directories
        (physics (phymars), dynamics (dyn3d), filters (filtrez)...)

   data_mars_gcm/ All the surface data (topography, albedo,
                  thermal inertia) and initialization files
                  for the chemistry and the troposphere.

   create_make_gcm   Executable used to create the makefile.
                     This command is run automatically by
                     "makegcm" (see below).

   deftank/          A collection of parametrization files required
                     for the run (examples of run.def...)
```

## 5.1 Organization of the model source files

The model source files are stored in various sub directories in directory **libf**. These sub-directories correspond to the different parts of the model:

**grid:** mainly made up of ”dimensions.h” file, which contains the parameters that define the model grid, i.e. the number of points in longitude (IIM), latitude (JJM) and altitude (LLM), as well as the number of tracers (NQMX) advected in the dynamical part (for example, 2 for water vapour and liquid in the basic version of the Earth model).

**dyn3d:** contains the dynamical subroutines.

**bibio:** contains some generic subroutines used by all the parts.

**phymars:** contains the Martian physical sources.

**filtrez:** contains the longitudinal filter sources applied in the upper latitudes, where the Courant-Friedrich-Levy stability criterion is violated.

**aeronomars:** contains the Martian chemistry and thermosphere sources.

## 5.2 Programming

The model is programmed in **FORTRAN-77** and **FORTRAN-90**.

- The program sources are written in **“file.F”** files. The extension .F is the standard extension for a FORTRAN file. These files must be preprocessed (by a**C preprocessor** such as (cpp)) before compilation. Most FORTRAN compliers recognize this extension automatically. A source file generally corresponds to a FORTRAN program or a subroutine, or sometimes to a small group of coherent subroutines.
- Constants are placed in COMMON declarations, located in the common “include” files **”file.h”**
- In general, variables are passed from subroutine to subroutine as arguments (and never as COMMON).
- In some parts of the code, for “historical” reasons, we apply the following rule: in any subroutine, the variables (ex: `name`) passed as an argument by the calling program are given the prefix `p` (ex: `pname`) while the local variables are given the prefix `z` (ex: `zname`). As a result, several variables change their prefix (and thus their name) when passing from a calling subroutine to a called subroutine.

## 5.3 Model organization

Figure 5.1 describes the main subroutines called by physiq.F.

## 5.4 Compiling the model

The model is compiled using the UNIX utility program `make`. The file ”makefile”, which describes the compilation process, is created automatically by the script

```
create_make_gcm
```

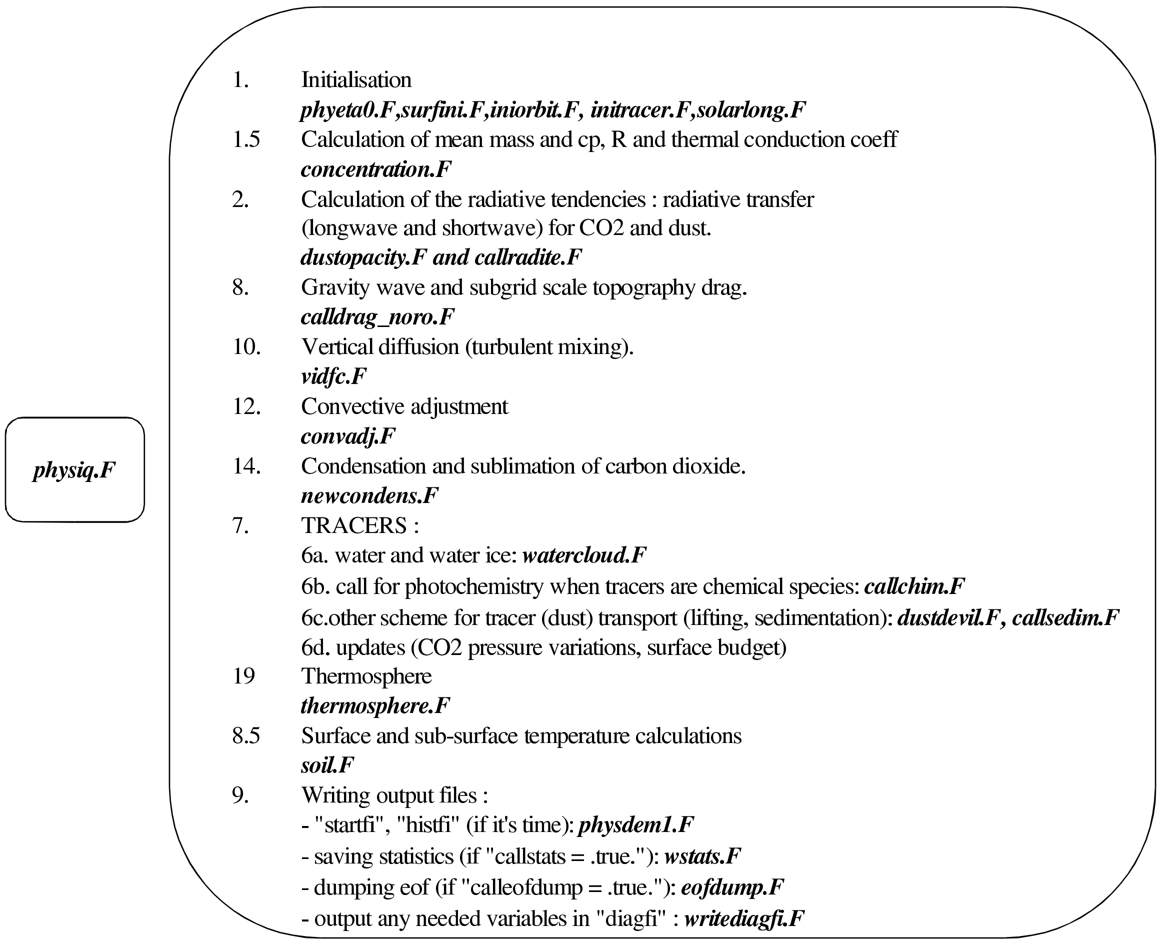
This utility program recreates the ”makefile” file when necessary, for example, when a source file has been added or removed since the last compilation.

**None of this is visible to the user. To compile the model just run the command**

```
makegcm
```

with adequate options (e.g. `makegcm -d 62x48x32 -p mars gcm`), as discussed below and described in section 4.2. The `makegcm` command compiles the model by calling the ”make” utility. A detailed description of how it functions is given in the help manual below (which will also be given by the `makegcm -h` command).
Note that before compiling the GCM with `makegcm` you should have set the environment variable **LIBOGCM** to a path where intermediate objects and libraries will be generated. In Cshell under Unix, this would be achieved by the following command line:

```
setenv LIBOGCM /users/me/put/GCM/objects/there
```



**physiq.F**

1. Initialisation
   ***phyeta0.F,surfini.F,iniorbit.F, initracer.F,solarlong.F***

1.5 Calculation of mean mass and cp, R and thermal conduction coeff
   ***concentration.F***

2. Calculation of the radiative tendencies : radiative transfer
   (longwave and shortwave) for CO2 and dust.
   ***dustopacity.F and callradite.F***

8. Gravity wave and subgrid scale topography drag.
   ***calldrag_noro.F***

10. Vertical diffusion (turbulent mixing).
   ***vidfc.F***

12. Convective adjustment
   ***convadj.F***

14. Condensation and sublimation of carbon dioxide.
   ***newcondens.F***

7. TRACERS :
   6a. water and water ice: ***watercloud.F***
   6b. call for photochemistry when tracers are chemical species: ***callchim.F***
   6c.other scheme for tracer (dust) transport (lifting, sedimentation): ***dustdevil.F, callsedim.F***
   6d. updates (CO2 pressure variations, surface budget)

19 Thermosphere
   ***thermosphere.F***

8.5 Surface and sub-surface temperature calculations
   ***soil.F***

9. Writing output files :
   - "startfi", "histfi" (if it's time): ***physdem1.F***
   - saving statistics (if "callstats = .true."): ***wstats.F***
   - dumping eof (if "calleofdump = .true."): ***eofdump.F***
   - output any needed variables in "diagfi" : ***writediagfi.F***

Figure 5.1: Organigram of subroutine function physiq.F

## Help manual for the makegcm function

```
By default, the makegcm command:
---------------------------------

1. compiles a series of subroutines located in the $LMDGCM/libf
 sub-directories.
The subroutines are then stored in the FORTRAN libraries at $LIBOGCM.

2. next, makegcm compiles program prog.f located by default at
$LMDGCM/libf/dyn3d and makes the link with the libraries.

Variable '$LMDGCM' must be initialized in your .cshrc file or directly
 in the makegcm file.

The makegcm command is used to control the different versions of the model
 in parallel, compiled using the compilation options
and the various dimensions, without having to recompile the whole model.

The FORTRAN libraries are stored in directory $LIBOGCM.


OPTIONS:
--------

The following options can either be defined by default by editing the
makegcm "script", or in interactive mode:

-d imxjmxlm  where im, jm, and lm are the number of longitudes,
             latitudes and vertical layers respectively.

-t ntrac   Selects the number of tracers advected by the dynamical part.
           In current versions of the Earth model,
           ntrac=2 for water vapour and liquid, for example.

             Options -d and -t overwrite file
             $LMDGCM/libf/grid/dimensions.h
             which contains the 3 dimensions of the
             horizontal grid
             im, jm, lm plus the number of tracers passively advected
             by the dynamic ntrac,
             in 4 PARAMETER FORTRAN format
             with a new file:
             $LMDGCM/libf/grid/dimension/dimensions.im.jm.lm.tntrac
             If the file does not exist already
             it is created by the script
             $LMDGCM/libf/grid/dimension/makdim

-p PHYS    Selects the physical parameterization set
           you require to compile the model.
           The model is then compiled using the physical
           parameterization sources in directory:
            $LMDGCM/libf/phyPHYS

-g grille  Selects the grid type.
           This option overwrites file
           $LMDGCM/libf/grid/fxyprim.h
           with file
           $LMDGCM/libf/grid/fxy_grille.h
           the grid can take the following values:
           1. reg - the regular grid
           2. sin - to obtain equidistant points in terms of sin(latitude)
           3. new - to zoom into a part of the globe

-O "optimisation fortran" where the fortran optimizations are the
            command f90 options.

-include path
```

```
           Used if the subroutines contain #include files (ccp) that
           are located in directories that were not referenced by default.

-adjnt     Compiles the adjoint model to the dynamical code.

-filtre  filter
           To select the longitudinal filter in the polar regions.
           "filter" corresponds to the name of a directory located in
           $LMDGCM/libf. The standard filter for the model is "filtrez"
           which can be used for a regular grid and for a
           grid with longitudinal zoom.

-link "-Ldir1 -lfile1 -Ldir2 -lfile2 ..."
           Adds a link to FORTRAN libraries
           libfile1.a, libfile2.a ...
           located in directories dir1, dir2 ...respectively
           If dirn is a directory with an automatic path
           (/usr/lib ... for example)
           there is no need to specify  -Ldirn.

-64        Compilation 64 bits for Sun.

Author: Frederic Hourdin  (hourdin@lmd.jussieu.fr)
```

# Chapter 6

# Input/Output

## 6.1 NetCDF format

GCM input/output data are written in **NetCDF** format (Network Common Data Form). NetCDF is an interface used to store and access geophysical data, and a library that provides an implementation of this interface. The NetCDF library also defines a machine-independent format for representing scientific data. Together, the interface, library and format support the creation, access and sharing of scientific data. NetCDF was developed at the Unidata Program Center in Boulder, Colorado. The freely available source can be obtained from the Unidata website.

`http://www.unidata.ucar.edu/packages/netcdf/index.html`

A data set in NetCDF format is a single file, as it is self-descriptive.

### 6.1.1 NetCDF file editor: ncdump

The editor is included in the NetCDF library. By default it generates an ASCII representation as standard output from the NetCDF file specified at the input.

#### Main commands for ncdump

*ncdump diagfi.nc*

dump contents of NetCDF file ”diagfi.nc” to standard output (i.e. the screen).

*ncdump -c diagfi.nc*

Displays the **coordinate** variable values (variables which are also dimensions), as well as the declarations, variables and attribute values. The values of the non-coordinate variable data are not displayed at the output.

*ncdump -h diagfi.nc*

Shows only the informative header of the file, which is the declaration of the dimensions, variables and attributes, but not the values of these variables. The output is identical to that in option **-c** except for the fact that the coordinated variable values are not included.

*ncdump -v var1,...,varn diagfi.nc*

The output includes the specific variable values, as well as all the dimensions, variables and attributes. More that one variable can be specified in the list following this option. The list must be a simple argument for the command, and must not contain any spaces. If no variable is specified, the command displays all the values of the variables in the file by default.

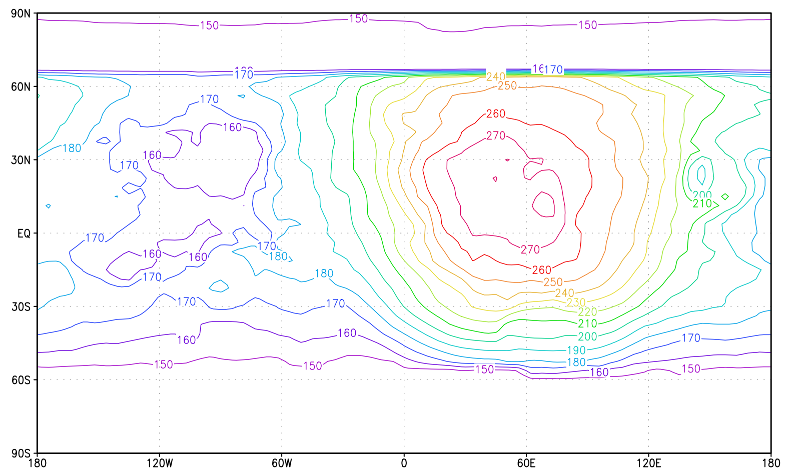


Figure 6.1: Example of temperature data at a given time using GrADS visualization

### 6.1.2 Graphic visualization of the NetCDF files using GrAds

GrAdS (The Grid Analysis and Display System) is a graphic software developed by Brian Doty at the "Center for Ocean-Land-Atmosphere (COLA)".

One of its functions is to enable data stored in NetCDF format to be visualized directly. In figure 6.1 for example, we can see the GrADS visualization of the temperature data at a given moment. However, unlike NetCDF, GrADS only recognizes files where all the variables are stored on the same horizontal grid. These variables can be in 1, 2, 3 or 4 dimensions (X,Y,Z and t).

GrADS can also be obtained on the WWW.

`http://grads.iges.org/grads/`

## 6.2 Input

The (3D version of the) GCM requires the input of two initialization files (in NetCDF format):
**-start.nc** containing the initial states of the dynamic variables.
**-startfi.nc** containing the initial states of the physics variables.
Note that collections of initial states can be retreived at:
`http://www.lmd.jussieu.fr/˜forget/datagcm/Starts`
Extracting `start.nc` and `startfi.nc` from these archived requires using program `newstart`, as described in section 4.8.

To run, the GCM also requires the three following parameter files (ascii text files):
**-run.def** the parameters of the dynamical part of the program, and the temporal integration of the model.
**-callphys.def** the parameters for calling the physical part.
**-z2sig.def** the parameters for the vertical distribution of the layers.
Examples of these parameter files can be found in the `LMDZ.MARS/deftank` directory.

### 6.2.1 run.def

A typical run.def file is given as an example below. The choice of variables to be set is simple (e.g. nday integration time), while the others do not need to be changed for normal use. However the following explanations may be helpful:

- **day_step**, the number of steps per day is governed by the stability criterion called "CFL", which depends on the horizontal resolution of the model. On Mars, in theory, the GCM can run with `day_step`=400 using the 64×48 grid, but model stability improves when this figure is higher: `day_step`=960 is recommended using the 64×48 grid. According to the CFL criterion, `day_step` should vary in proportion with the resolution: for example `day_step`=480 using the 32×24 horizontal resolution. `day_step` must also be divisible by `iperiod`.

- **tetagdiv, tetagrot, tetatemp** control the dissipation intensity. It is better to limit the dissipation intensity (tetagdiv, tetagrot, tetatemp should not be too low). However the model diverges if tetagdiv, tetagrot, tetatemp are too high, especially if there is a lot of dust in the atmosphere.
  Example used with nitergdiv=1 and nitergrot=niterh=2 :
  - using the 32×24 grid tetagdiv=6000 s ; tetagrot=tetatemp=30000 s
  - using the 64×48 grid: tetagdiv=3000 s ; tetagrot=tetatemp=9000 s

- **idissip** is the time step used for the dissipation. If idissip is too short, the model will lose time in the calculations. But if idissip is too long, the dissipation will not be parametrized correctly and the model will be more likely to diverge. A check must be made, in order of size that: idissip $<$ tetagdiv×daystep/88775 (idem for tetagrot and tetatemp). This is now tested automatically during the run.

- **iphysiq** corresponds to the physical timestep. In practice, we normally set this time to the order of half an hour. We therefore set `iphysiq`= `day_step`/48

*Contents of run.def:*

```
-----------------------------------------------------------------------
Parametres de controle du run:
------------------------------

- Nombre de jours d'integration
     nday
 9999

- nombre de pas par jour (multiple de iperiod) ( ici pour  dt = 1 min )
 day_step
 480

- periode pour le pas Matsuno (en pas)
  iperiod
 5

- periode de sortie des variables de controle (en pas)
  iconser
 120

- periode d'ecriture du fichier histoire (en jour)
    iecri
 100

- periode de stockage fichier histmoy (en jour)
 periodav
 60.

- periode de la dissipation (en pas)
  idissip
 1

- choix de l'operateur de dissipation (star ou  non star )
 lstardis
 T

- avec ou sans coordonnee hybrides
 hybrid
 T

- nombre d’iterations de l’operateur de dissipation   gradiv
nitergdiv
 1

- nombre d’iterations de l’operateur de dissipation  nxgradrot
nitergrot
 2

- nombre d’iterations de l’operateur de dissipation  divgrad
   niterh
 2

- temps de dissipation des plus petites long.d ondes pour u,v (gradiv)
 tetagdiv
 3000.

- temps de dissipation des plus petites long.d ondes pour u,v(nxgradrot)
 tetagrot
 9000.

- temps de dissipation des plus petites long.d ondes pour  h ( divgrad)
 tetatemp
 9000.

- coefficient pour gamdissip
  coefdis
 0.

- choix du shema d’integration temporelle (Matsuno ou Matsuno-leapfrog)
  purmats
 F

- avec ou sans physique
   physic
 T

- periode de la physique (en pas)
  iphysiq
 10

- choix d’une grille reguliere
  grireg
  T

- frequence (en pas) de l’ecriture du fichier diagfi
 ecritphy
 120

- longitude en degres du centre du zoom
   clon
 63.

- latitude en degres du centre du zoom
   clat
 0.

- facteur de grossissement du zoom,selon longitude
  grossismx
 1.

- facteur de grossissement du zoom ,selon latitude
```

```
 grossismy
 1.

-  Fonction  f(y)  hyperbolique  si = .true.  , sinon  sinusoidale
  fxyhypb
 F

- extension en longitude  de la zone du zoom  ( fraction de la zone totale)
   dzoomx
 0.

- extension en latitude de la zone  du zoom  ( fraction de la zone totale)
   dzoomy
 0.

-  raideur du zoom en  X
    taux
 2.

-  raideur du zoom en  Y
    tauy
 2.

-  Fonction  f(y) avec y = Sin(latit.) si = .TRUE. ,  Sinon  y = latit.
  ysinus
  F

- Avec sponge layer
  callsponge
  T

- Sponge:  mode0(u=v=0), mode1(u=umoy,v=0), mode2(u=umoy,v=vmoy)
  mode_sponge
  2

- Sponge:  hauteur de sponge (km)
  hsponge
  130

- Sponge:  tetasponge (secondes)
  tetasponge
  50000

c   *****************************
c
c      Autre reglage possible :
c
cc  F ) deltay : deplacement ( en degres ) en  y  de la zone du zoom
cc                                          dans  *** inigeom.F ***
```

### 6.2.2 callphys.def

```
General options
~~~~~~~~~~~~~~~
tracer    (Run with or without tracer transport ?)
F
diurnal   (Diurnal cycle ?  if diurnal=F, diurnal averaged solar heating)
T
season    (Seasonal cycle ? if season=F, Ls stays constant like in "start")
T
lwrite    (want some more output on the screen ?)
F
stats     (Saving statistics in file "cumul" ?)
T
calleofdump (Saving EOF profiles in file "profiles" for Climate Database ?)
```

```
F
Dust scenario. Used if the dust is prescribed (i.e. if tracer=F or active=F)
~~~~~~~~~~~~~
iaervar  (=1 Dust opt.deph read in startfi; =2 Viking scenario; =3 MGS scenario)
4        (=4 Mars Year 24 from TES assimilation)
iddist  (Dust vertical distribution: =0: old distrib. (Pollack90)
3       (=1: top set by "topdustref"; =2: Viking scenario; =3 MGS scenario )
topdustref (Dust top altitude (km). Matter only if iddist=1)
55.
Physical Parameterizations :
~~~~~~~~~~~~~~~~~~~~~~~~~~
callrad   (call radiative transfer ?)
T
callnlte (call NLTE radiative schemes ?   matter only if callrad=T)
T
callnirco2 (call CO2 NIR absorption ?   matter only if callrad=T)
T
calldifv  (call turbulent vertical diffusion ?)
T
calladj   (call convective adjustment ?)
T
callcond  (call CO2 condensation ?)
T
callsoil  (call thermal conduction in the soil ?)
T
calllott  (call Lott’s gravity wave/subgrid topography scheme ?)
T
Radiative transfer options :
~~~~~~~~~~~~~~~~~~~~~~~~~~
iradia    (the rad.transfer is computed every "iradia" physical timestep)
1
callg2d   (Output of the exchange coefficient mattrix ? for diagnostic only)
F
rayleigh  (Rayleigh scattering : should be =F for now)
F
Tracer (dust water, ice and/or chemical species) options (use if tracer=T) :
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
dustbin    (DUST: Transported dust ? (if >0, uses q(1) to q(dustbin))
0
active     (DUST: Radiatively active dust ? (uses q(1) to q(dustbin))
F
doubleq    (DUST: needs dustbin=1, use mass (q(1)) and number (q(2)) mixing ratio to predict dust size ?)
F
lifting    (DUST: lifted by GCM surface winds ?)
F
dustdevil  (DUST: lifted by dust devils ?)
F
scavenging (DUST: Scavenging by CO2 snowfall ?)
F
sedimentation (DUST/WATERICE: Gravitationnal sedimentation ?)
F
iceparty   (WATERICE: includes water ice: q(nqmx-1)). "water" must be T !
F
activice   (WATERICE: Radiatively active transported atmospheric water ice ?)
F
water      (WATER: Compute water cycle using q(nqmx) )
F
caps       (WATER: current permanent caps at both poles. T IS RECOMMENDED
T          (caps=T means Ncap is a source of water, S pole is a cold trap)
photochem  (PHOTOCHEMISTRY: chemical species included)
F
Thermospheric options (relevant if tracer=T) :
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
callthermos  (call thermosphere ?)
F
thermoswater  (WATER: included without cycle only if water=F)
F
```

```
callconduct  (call thermal conduction ?     matter only if callthermos=T)
F
calleuv  (call EUV heating ?                matter only if callthermos=T)
F
callmolvis  (call molecular viscosity ?     matter only if callthermos=T)
F
callmoldiff  (call molecular diffusion ?    matter only if callthermos=T)
F
thermochem  (call thermospheric photochemistry ?  matter only if callthermos=T)
F
solarcondate (date for solar flux calculation: 1985 < date < 2002))
1993.4       (Solar min=1996.4 ave=1993.4 max=1990.6)
```

### 6.2.3 z2sig.def

**z2sig.def** (this version for 50 layers between 0 and 400 km):

```
10.00000     H: atmospheric scale height (km) (used as a reference only)
0.0040       Typical pseudo-altitude (m) for 1st layer (z=H*log(sigma))
0.018       ,, ,,  ,, ,, ,, ,,  ,, ,, ,,  2nd layer, etc...
0.0400
0.1000
0.228200
0.460400
0.907000
1.73630
3.19040
5.54010
8.97780
13.5138
18.9666
25.0626
31.5527
38.4369
45.4369
52.4369
59.4369
66.4369
73.4369
80.4369
87.4369
94.4369
101.4369
108.437
115.437
122.437
129.437
136.437
143.437
150.437
157.437
164.437
171.437
178.437
185.437
192.437
```

```
199.437
206.437
213.437
220.437
227.437
234.437
241.437
248.437
255.437
262.437
269.437
276.437
283.437
290.437
297.437
304.437
311.437
318.437
325.437
332.437
339.437
346.437
353.437
360.437
367.437
374.437
381.437
388.437
395.437
```

### 6.2.4 Initialization files: start and startfi

Files `start.nc` and `startfi.nc`, like all the NetCDF files of the GCM, are constructed on the same model (see NetCDF file composition, figure 6.2). They contain:
- a header with a "control" variable followed by a series of variables defining the (physical and dynamical) grid
- a series of non temporal variables that give information about surface conditions on the planet.
- a "time" variable giving the values of the different instants at which the temporal variables are stored (a single time value (t=0) for start, as it describes the dynamical initial states, and no time values for startfi, as it describes only a physical state).
- the model state variables on the grid corresponding to the different instants of the selected simulation (adjustable by a timestep) for the output files being simulated (diagfi, histmoy, etc.).

To visualize the contents of a start file using the ncdump editor:

*ncdump -h start.nc*

```
netcdf start {
dimensions:
        index = 100 ;
        rlonu = 65 ;
        latitude = 49 ;
        longitude = 65 ;
```

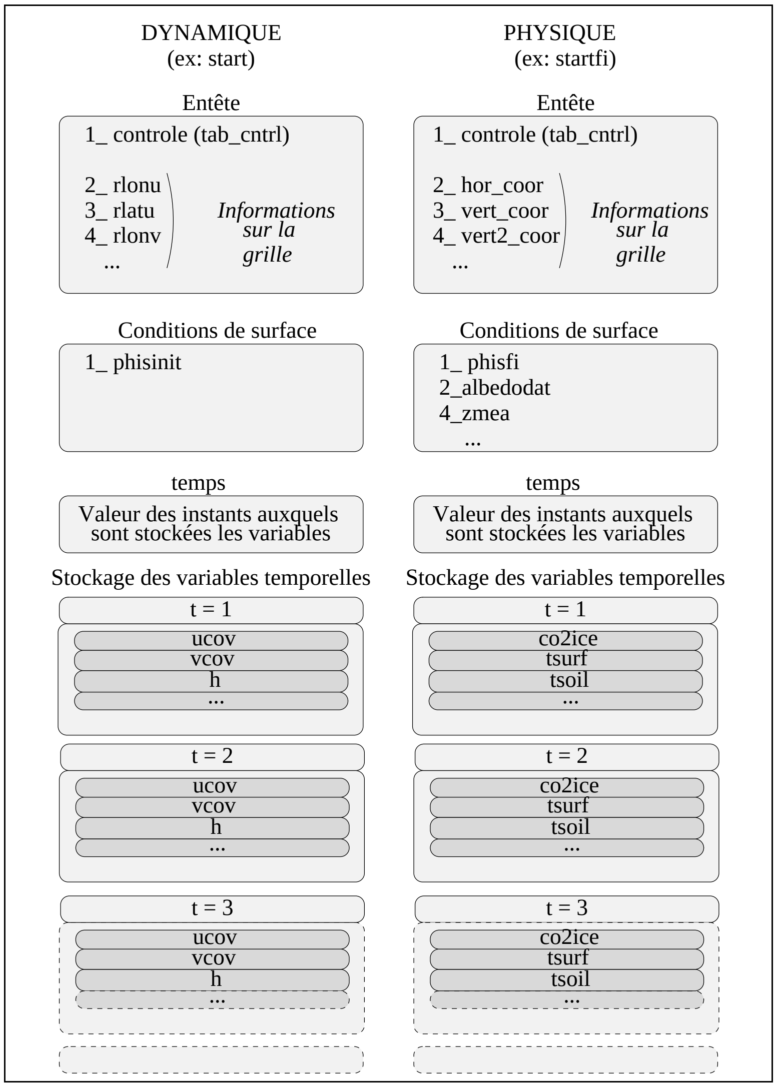


Figure 6.2: Organization of NetCDF files

```
        rlatv = 48 ;
        altitude = 25 ;
        interlayer = 26 ;
        Time = UNLIMITED ; // (1 currently)
variables:
        float controle(index) ;
                controle:title = "Parametres de controle" ;
        float rlonu(rlonu) ;
                rlonu:title = "Longitudes des points U" ;
        float rlatu(latitude) ;
                rlatu:title = "Latitudes des points U" ;
        float rlonv(longitude) ;
                rlonv:title = "Longitudes des points V" ;
        float rlatv(rlatv) ;
                rlatv:title = "Latitudes des points V" ;
        float ap(interlayer) ;
                ap:title = "Coef A: hybrid pres levels" ;
        float bp(interlayer) ;
                bp:title = "Coef B: hybrid sigma level" ;
        float aps(altitude) ;
```

```
                aps:title = "Coef AS: hybrid pressure in midlayer" ;
        float bps(altitude) ;
                bps:title = "Coef BS: hybrid sigma midlayer" ;
        float presnivs(altitude) ;
        float latitude(latitude) ;
                latitude:units = "degrees_north" ;
                latitude:long_name = "North latitude" ;
        float longitude(longitude) ;
                longitude:long_name = "East longitude" ;
                longitude:units = "degrees_east" ;
        float altitude(altitude) ;
                altitude:long_name = "pseudo-alt" ;
                altitude:units = "km" ;
                altitude:positive = "up" ;
        float cu(latitude, rlonu) ;
                cu:title = "Coefficient de passage pour U" ;
        float cv(rlatv, longitude) ;
                cv:title = "Coefficient de passage pour V" ;
        float aire(latitude, longitude) ;
                aire:title = "Aires de chaque maille" ;
        float phisinit(latitude, longitude) ;
                phisinit:title = "Geopotentiel au sol" ;
        float Time(Time) ;
                Time:title = "Temps de simulation" ;
                Time:units = "days since    0-00-00 00:00:00" ;
        float ucov(Time, altitude, latitude, rlonu) ;
                ucov:title = "Vitesse U" ;
        float vcov(Time, altitude, rlatv, longitude) ;
                vcov:title = "Vitesse V" ;
        float teta(Time, altitude, latitude, longitude) ;
                teta:title = "Temperature" ;
        float q01(Time, altitude, latitude, longitude) ;
                q01:title = "Traceurs q01" ;
        float q02(Time, altitude, latitude, longitude) ;
                q02:title = "Traceurs q02" ;
        float masse(Time, altitude, latitude, longitude) ;
                masse:title = "C est quoi ?" ;
        float ps(Time, latitude, longitude) ;
                ps:title = "Pression au sol" ;

// global attributes:
                :title = "Dynamic start file" ;
}
```

List of contents of a startfi file:

*ncdump -h startfi.nc*

```
netcdf startfi {
dimensions:
        index = 100 ;
        physical_points = 3010 ;
        subsurface_layers = 10 ;
variables:
        float controle(index) ;
                controle:title = "Parametres de controle" ;
        float longitude(physical_points) ;
                longitude:title = "Longitudes de la grille physique" ;
        float latitude(physical_points) ;
                latitude:title = "Latitudes de la grille physique" ;
        float area(physical_points) ;
                area:title = "Aire des mailles" ;
        float phisfi(physical_points) ;
                phisfi:title = "Geopotentiel au sol" ;
        float albedodat(physical_points) ;
                albedodat:title = "Albedo du sol nu" ;
        float inertiedat(physical_points) ;
```

```
                inertiedat:title = "Inertie thermique du sol" ;
        float ZMEA(physical_points) ;
                ZMEA:title = "Relief moyen" ;
        float ZSTD(physical_points) ;
                ZSTD:title = "Ecartype du relief" ;
        float ZSIG(physical_points) ;
                ZSIG:title = "Relief: parametre sigma" ;
        float ZGAM(physical_points) ;
                ZGAM:title = "Relief: parametre gamma" ;
        float ZTHE(physical_points) ;
                ZTHE:title = "Relief: parametre theta" ;
        float co2ice(physical_points) ;
                co2ice:title = "CO2 ice cover" ;
        float tsurf(physical_points) ;
                tsurf:title = "Surface temperature" ;
        float tsoil(subsurface_layers, physical_points) ;
                tsoil:title = "Soil temperature" ;
        float emis(physical_points) ;
                emis:title = "Surface emissivity" ;
        float q2(subsurface_layers, physical_points) ;
                q2:title = "pbl wind variance" ;
        float qsurf01(physical_points) ;
                qsurf01:title = "tracer on surface" ;
        float qsurf02(physical_points) ;
                qsurf02:title = "tracer on surface" ;

// global attributes:
                :title = "Fichier demarrage physique" ;
}
```

## Physical and dynamical headers

There are two types of headers: one for the physical headers, and one for the dynamical headers. The headers always begin with a “control’ variable (described below), that is allocated differently in the physical and dynamical parts. The other variables in the header concern the (physical and dynamical) grids. They are the following:

the horizontal coordinates
- **rlonu**, **rlatu**, **rlonv**, **rlatv** for the dynamical part,
- **lati**, **long** for the physical part,

the coefficients for passing from the physical grid to the dynamical grid
- **cu**,**cv** only in the dynamical header

and finally, the grid box areas
- **aire** for the dynamical part,
- **area** for the physical part.

## Surface conditions

The surface conditions are mostly given in the physical NetCDF files by variables:
- **phisfi** for the initial state of surface geopotential,
- **albedodat** for the bare ground albedo,
- **inertiedat** for the surface thermal inertia,
- **zmea**, **zstd**, **zsig**, **zgam** and **zthe** for the subgrid scale topography.

For the dynamical part:
- **physinit** for the initial state of surface geopotential

Remark: variables **phisfi** and **physinit** contain the same information (surface geopotential), but physfi gives the geopotential values on the physical grid, while physinit give the values on the dynamical grid.

## Physical and dynamical state variables

To save disk space, the initialization files store the variables used by the model, rather than the "natural" variables.

For the dynamical part:

- **ucov** and **vcov** the covariant winds
  These variables are linked to the "natural" winds by
  `ucov = cu * u` and `vcov = cv * v`

- **teta** the potential temperature,
  or more precisely, the potential enthalpy linked to temperature **T** by $\theta = T\left(\frac{P}{Pref}\right)^{-K}$

- **q01, q02, etc.** the tracers,

- **ps** surface pressure.

- **masse** the atmosphere mass in each grid box.

"Vectorial" variables **ucov** and **vcov** are stored on "staggered" grids u and v respectively (in the dynamical part) (see section 2.2).
Scalar variables **h**, **q01**, **ps**, **masse** are stored on the "scalar" grid of the dynamical part.

For the physical part:

- **co2ice** surface dry ice,

- **tsurf** surface temperature,

- **tsoil** temperatures at different layers under the surface,

- **emis** surface emissivity,

- **q2** wind variance,
  or more precisely, the square root of the turbulent kinetic energy.

- **qsurf01, qsurf02, etc...** the surface "tracer" budget (kg.m$^{-2}$),

All these variables are stored on the "physical" grid (see section 2.2).

## The "control" variable

Both the physical and dynamical headers of the GCM NetCDF files start with a **controle** variable. This variable is a table with 100 reals (the table called "tab_cntrl" in the program), which contains the program control parameters. These parameters differ in the physical and dynamical parts, and examples of both are listed below. The contents of table `tab_cntrl` can also be seen by typing the command "*ncdump -v controle*".

**The "control" variable in the header of a dynamical NetCDF file: start**

```
-------------------------------------------------------
tab_cntrl(1)  = FLOAT(iim)      ! nombre de points en longitude
tab_cntrl(2)  = FLOAT(jjm)      ! nombre de points en latitude
tab_cntrl(3)  = FLOAT(llm)      ! nombre de couches
tab_cntrl(4)  = FLOAT(idayref)  ! jour 0
tab_cntrl(5)  = FLOAT(anneeref) ! annee 0
tab_cntrl(6)  = rad     ! rayon de mars(m) ~3397200
tab_cntrl(7)  = omeg    ! vitesse de rotation (rad.s-1)
tab_cntrl(8)  = g       ! gravite (m.s-2) ~3.72
tab_cntrl(9)  = cpp
tab_cntrl(10) = kappa   ! = r/cp  ~0.256793 (=rcp dans physique)
tab_cntrl(11) = daysec ! duree du sol (s)  ~88775
tab_cntrl(12) = dtvr   ! pas de temps de la dynamique (s)
tab_cntrl(13) = etot0  ! energie totale
tab_cntrl(14) = ptot0  ! pression totale
tab_cntrl(15) = ztot0  ! enstrophie totale
tab_cntrl(16) = stot0  ! enthalpie totale
tab_cntrl(17) = ang0   ! moment cinetique
tab_cntrl(18) = pa
tab_cntrl(19) = preff  ! pression de reference ~670

tab_cntrl(20)  = clon       ! longitude en degres du centre du zoom
tab_cntrl(21)  = clat       ! latitude en degres du centre du zoom
tab_cntrl(22)  = grossismx  ! facteur de grossissement du zoom,selon longitude
tab_cntrl(23)  = grossismy  ! facteur de grossissement du zoom ,selon latitude

tab_cntrl(25) = dzoomx    ! extension en longitude de la zone du zoom
tab_cntrl(26) = dzoomy    ! extension en latitude de la zone du zoom
tab_cntrl(28) = taux      ! raideur du zoom en x
tab_cntrl(29) = tauy      ! raideur du zoom en y
```

### The ”controle” variable in the header of a physical NetCDF file: startfi.nc

```
_______________________________________________________
c Info sur la grille physique
      tab_cntrl(1) = float(ngridmx)  ! nombre de points de la grille physique
      tab_cntrl(2) = float(nlayermx) ! nombre de couches
      tab_cntrl(3) = float(idayref)  ! jour 0

c Info sur la Planete Mars pour la dynamique et la physique
      tab_cntrl(5) = rad      ! rayon de mars(m) ~3397200
      tab_cntrl(6) = omeg     ! vitesse de rotation (rad.s-1)
      tab_cntrl(7) = g        ! gravite (m.s-2) ~3.72
      tab_cntrl(8) = mugaz    ! Masse molaire de l'atm (g.mol-1) ~43.49
      tab_cntrl(9) = rcp      !  = r/cp  ~0.256793 (=kappa dans dynamique)
      tab_cntrl(10) = daysec  ! duree du sol (s)  ~88775

      tab_cntrl(11) = dtphys  ! pas de temps de la physique
      tab_cntrl(12) = 0.
      tab_cntrl(13) = 0.

c Info sur la Planete Mars pour la physique uniquement
      tab_cntrl(14) = year_day  ! duree de l'annee (sols) ~668.6
      tab_cntrl(15) = periheli  ! dist.min. soleil-mars (Mkm) ~206.66
      tab_cntrl(16) = aphelie   ! dist.max. soleil-mars (Mkm) ~249.22
      tab_cntrl(17) = peri_day  ! date du perihelie (sols depuis printemps)
      tab_cntrl(18) = obliquit  ! Obliquite de la planete (deg) ~23.98

c Couche limite et Turbulence
      tab_cntrl(19) = z0        ! surface roughness (m) ~0.01
      tab_cntrl(20) = lmixmin   ! longueur de melange ~100
      tab_cntrl(21) = emin_turb ! energie minimale ~1.e-8

c propriete optiques des calottes et emissivite du sol
      tab_cntrl(22) = albedice(1)  ! Albedo calotte nord ~0.5
      tab_cntrl(23) = albedice(2)  ! Albedo calotte sud ~0.5
      tab_cntrl(24) = emisice(1)   ! Emissivite calotte nord ~0.95
      tab_cntrl(25) = emisice(2)   ! Emissivite calotte sud ~0.95
      tab_cntrl(26) = emissiv      ! Emissivite du sol martien ~.95
      tab_cntrl(31) = iceradius(1) ! mean scat radius of CO2 snow (north)
      tab_cntrl(32) = iceradius(2) ! mean scat radius of CO2 snow (south)
      tab_cntrl(33) = dtemisice(1) ! time scale for snow metamorphism (north)
      tab_cntrl(34) = dtemisice(2) ! time scale for snow metamorphism (south)

c Proprietes des poussiere aerosol
      tab_cntrl(27) = tauvis      ! profondeur optique visible moyenne

      tab_cntrl(28) = 0.
      tab_cntrl(29) = 0.
      tab_cntrl(30) = 0.
-------------------------------------------------------
```

## 6.3 Output

### 6.3.1 NetCDF restart files - restart.nc and restartfi.nc

These files are of the exact same format as `start.nc` and `startfi.nc`

### 6.3.2 NetCDF file - diagfi.nc

NetCDF file `diagfi.nc` stores the instantaneous physical variables throughout the simulation at regular intervals (set by the value of parameter `ecritphy` in parameter file "run.def").

**Any variable from any sub-routine of the physical part can be stored by calling subroutine** `writediagfi`

Visualization of the contents of a diagfi file using ncdump:

*ncdump -h diagfi*

```
netcdf diagfi1 {
dimensions:
Time = UNLIMITED ; // (366 currently)
index = 100 ;
rlonu = 65 ;
latitude = 49 ;
longitude = 65 ;
rlatv = 48 ;
interlayer = 26 ;
altitude = 25 ;
variables:
float Time(Time) ;
Time:long_name = "Time" ;
Time:units = "days since 0000-00-0 00:00:00" ;
float controle(index) ;
controle:title = "Parametres de controle" ;
float rlonu(rlonu) ;
rlonu:title = "Longitudes aux points u" ;
float latitude(latitude) ;
latitude:units = "degrees_north" ;
latitude:long_name = "North latitude" ;
float longitude(longitude) ;
longitude:long_name = "East longitude" ;
longitude:units = "degrees_east" ;
float altitude(altitude) ;
altitude:long_name = "pseudo-alt" ;
altitude:units = "km" ;
altitude:positive = "up" ;
float rlatv(rlatv) ;
rlatv:title = "Latitudes aux points v" ;
float aps(altitude) ;
aps:title = "hybrid pressure at midlayers" ;
aps:units = "h" ;
float bps(altitude) ;
bps:title = "hybrid sigma at midlayers" ;
bps:units = "h" ;
float ap(interlayer) ;
ap:title = "hybrid pressure at interlayers" ;
ap:units = "h" ;
float bp(interlayer) ;
bp:title = "hybrid sigma at interlayers" ;
bp:units = "h" ;
float cu(latitude, rlonu) ;
cu:title = "Coefficients de passage cov <--> nature" ;
float cv(rlatv, longitude) ;
cv:title = "Coefficients de passage cov <--> nature" ;
float aire(latitude, longitude) ;
aire:title = "Aires des mailles" ;
float phisinit(latitude, longitude) ;
phisinit:title = "Geopotentiel au sol init" ;
float mtot(Time, latitude, longitude) ;
mtot:title = "total mass of water vapor" ;
mtot:units = "kg/m2" ;
float icetot(Time, latitude, longitude) ;
icetot:title = "total mass of water ice" ;
icetot:units = "kg/m2" ;
float rice(Time, altitude, latitude, longitude) ;
rice:title = "ice radius" ;
rice:units = "meter" ;
float rave(Time, latitude, longitude) ;
rave:title = "Mean ice radius" ;
rave:units = "meter" ;
float co2ice(Time, latitude, longitude) ;
co2ice:title = "co2 ice thickness" ;
co2ice:units = "kg.m-2" ;
float tsurf(Time, latitude, longitude) ;
tsurf:title = "Surface temperature" ;
tsurf:units = "K" ;
float ps(Time, latitude, longitude) ;
ps:title = "surface pressure" ;
ps:units = "K" ;
float temp(Time, altitude, latitude, longitude) ;
temp:title = "temperature" ;
temp:units = "K" ;
float tau(Time, latitude, longitude) ;
tau:title = "tau" ;
tau:units = "t" ;
float q02(Time, altitude, latitude, longitude) ;
q02:title = "mix. ratio" ;
q02:units = "kg/kg" ;
float qsurf02(Time, latitude, longitude) ;
qsurf02:title = "qsurf" ;
qsurf02:units = "kg.m-2" ;
float q01(Time, altitude, latitude, longitude) ;
q01:title = "mix. ratio" ;
q01:units = "kg/kg" ;
float qsurf01(Time, latitude, longitude) ;
qsurf01:title = "qsurf" ;
qsurf01:units = "kg.m-2" ;
}
```

The structure of the file is thus as follows:

- the dimensions
- variable “time” containing the time of the timestep stored in the file (in Martian days since the beginning of the run)
- variable “control” containing many parameters, as described above.
- from “ rhonu” to ’phisinit”: a list of data describing the geometrical coordinates of the data file, plus the surface topography
- finally, all the 2D or 3D data stored in the run.

The frequency at which variables are stored in file `diagfi.nc` is set by parameter `ecritphy` in file “run.def” (see section 6.2.1).

### 6.3.3 Stats files

As an option (adjustable in “callphys.def”), the model can accumulate any variable from any subroutine of the physical part by calling subroutine `wstat`

This save is performed at regular intervals 12 times a day. An average of the daily evolutions is calculated (for example, for a 10 day run, the averages of the variable values at 0hTU, 2hTU, 4hTU,...24hTU are calculated).

These data are stored in statistics file `stats.nc`.

# Chapter 7

# Zoomed simulations

The LMD GCM can use a zoom to enhance the resolution locally. In practice, one can increase the latitudinal resolution on the one hand, and the longitudinal resolution on the other hand.

## 7.1 To define the zoomed area

The zoom is defined in `run.def`. Here are the variables that you want to set:

- EAST longitude en degres du centre du zoom `clon`: zoom center (degree)
- latitude en degres du centre du zoom `clat` : zoom center (degree N)
- facteur de grossissement du zoom,selon longitude `grossismx`. *Typically 1.5, 2 or even 3 (see below)*
- facteur de grossissement du zoom,selon latitude `grossismy`. *Typically 1.5, 2 or even 3 (see below)*
- longitude en degres du centre du zoom `clon`
- Fonction f(y) hyperbolique si = .true. , sinon sinusoidale `fxyhypb`: **must be set to ”T” for a zoom**, *whereas it must be F usually*
- extension en longitude de la zone du zoom `dzoomx`. This is the total longitudinal extension of the zoomed region (degree).
  *It is recommended that* `grossismx` $\times$ `dzoomx` $< 200^o$
- extension en longitude de la zone du zoom `dzoomy`. This is the total longitudinal extension of the zoomed region (degree).
  *It is recommended that* `grossismy` $\times$ `dzoomy` $< 100^o$
- raideur du zoom en X `taux`. 2 is for a smooth transition in longitude, more means sharper transition.
- raideur du zoom en Y `taux`. 2 is for a smooth transition in latitude, more means sharper transition.

## 7.2 Making a zoomed initial state

One must start from an initial state archive `start_archive.nc` obtained from a previous simulation (see section 4.8) Then compile and run `newstart.e` **using the `run.def` file designed for the zoom**.

After running `newstart.e`. The zoomed grid should visualize using grads, for instance. Here is a grads script that can be used to map the grid above a topography map:

```
set mpdraw off
set grid off
sdfopen restart.nc
set gxout grid
set digsiz 0
set lon -180 180
d ps
close 1
*** replace the path to surface.nc in the following line:
sdfopen  /u/forget/WWW/datagcm/datafile/surface.nc
set lon -180 180
set gxout contour
set clab off
set cint 3
d zMOL
```

## 7.3 Running a zoomed simulation and stability issue

- **dynamical timestep** Because of their higher resolution, zoomed simulation requires a higher timestep. Therefore in `run.def`, the number of dynamical timestep per day `day_step` must be increased by more than `grossismx` or `grossismy` (twice that if necessary). However, you can keep the same physical timestep (48/sol) and thus increase `iphysiq` accordingly (`iphysiq = day_step/48`).
- It has been found that when zooming in longitude, on must set `ngroup=1` in `dyn3d/groupeun.F`. Otherwise the run is less stable.
- The very first initial state made with `newstart.e` can be noisy and dynamically unstable. It may be necessary to strongly increase the intensity of the dissipation and increase `day_step` in run.def for 1 to 3 sols, and then use less strict values.
- If the run remains very unstable and requires too much dissipation or a too small timestep, a good tip to help stabilize the model is to decrease the vertical extension of your run and the number of layer (one generally zoom to study near-surface process, so 20 to 22 layers and a vertical extension up to 60 or 80 km is usually enough).

# Chapter 8

# Water Cycle Simulation

In order to simulate the water cycle with the LMD GCM:

- In `callphys.def`, set tracer to true: `tracer=T`. Use the same options as below for the Tracer part, the rest does not change compared to the basic `callphys.def`. The important parameters are `water=T`, to include water vapor tracer, `iceparty=T`, to use water ice tracer, and `sedimentation=T` to allow sedimentation of water ice clouds.

```
Tracer (dust, water, water ice and/or chemical species)
options (relevant if tracer=T) :
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
dustbin    (DUST: Transported dust ? (if >0, uses q(1) to q(dustbin))
0
active     (DUST: Radiatively active dust ? (uses q(1) to q(dustbin))
F
doubleq    (DUST: needs dustbin=1, use mass (q(1)) and number (q(2))
 mixing ratio to predict dust size ?)
F
lifting    (DUST: lifted by GCM surface winds ?)
F
dustdevil  (DUST: lifted by dust devils ?)
F
scavenging (DUST: Scavenging by CO2 snowfall ?)
F
sedimentation (DUST/WATERICE: Gravitationnal sedimentation ?)
T
iceparty   (WATERICE: Water cycle includes water ice mixing ratio q(nqmx-1))
T
activice   (WATERICE: Radiatively active transported atmospheric water ice ?)
F
water      (WATER: Compute water cycle using q(nqmx) )
T
caps       (WATER: put the current permanent caps at both poles)
T
photochem  (PHOTOCHEMISTRY: chemical species included)
F
Thermospheric options (relevant if tracer=T) :
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
callthermos  (call thermosphere ?)
F
thermoswater  (WATER: included without cycle    only if water=F)
F
callconduct  (call thermal conduction ?     matter only if callthermos=T)
F
calleuv  (call EUV heating ?                matter only if callthermos=T)
F
callmolvis  (call molecular viscosity ?     matter only if callthermos=T)
```

```
F
callmoldiff  (call molecular diffusion ?    matter only if callthermos=T)
F
thermochem  (call thermospheric photochemistry ?  matter only if callthermos=T)
F
solarcondate (date for solar flux calculation)
1998.
```

- **Compilation** You need to compile with at least 2 tracers. If you don't have dust (`dustbin=0`) or other chemical species (`photochem=F`), compilation is done with the command lines:

  ```
  makegcm -d 64x48x25 -t 2 -p mars newstart
  ```

  ```
  makegcm -d 64x48x25 -t 2 -p mars gcm
  ```

- **Run**

  Same as usual. Just make sure that your start files contains the initial states for water, with an initial state for water vapor and water ice particles.

# Chapter 9

# Photochemical Module

The LMD GCM now includes a photochemical module, which allows to compute the atmospheric composition.

- 14 chemical species are included: $CO_2$ (background gas), CO, O, $O(^1D)$, $O_2$, $O_3$, H, $H_2$, OH, $HO_2$, $H_2O_2$, $N_2$, Ar (inert) and $H_2O$.
- In `callphys.def`, set tracer to true `tracer=T`. Use the same options as below for the Tracer part, the rest does not change compared to the basic `callphys.def`. You need to set `photochem=T`, and to include the water cycle (`water=T`, `iceparty=T`, `sedimentation=T`; see Chapter 8), because composition is extremely dependent on the water vapor abundance.

```
Tracer (dust, water, water ice and/or chemical species)
options (relevant if tracer=T) :
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
dustbin    (DUST: Transported dust ? (if >0, uses q(1) to q(dustbin))
0
active     (DUST: Radiatively active dust ? (uses q(1) to q(dustbin))
F
doubleq    (DUST: needs dustbin=1, use mass (q(1)) and number (q(2))
 mixing ratio to predict dust size ?)
F
lifting    (DUST: lifted by GCM surface winds ?)
F
dustdevil  (DUST: lifted by dust devils ?)
F
scavenging (DUST: Scavenging by CO2 snowfall ?)
F
sedimentation (DUST/WATERICE: Gravitationnal sedimentation ?)
T
iceparty   (WATERICE: Water cycle includes water ice mixing ratio q(nqmx-1))
T
activice   (WATERICE: Radiatively active transported atmospheric water ice ?)
F
water      (WATER: Compute water cycle using q(nqmx) )
T
caps       (WATER: put the current permanent caps at both poles)
T
photochem  (PHOTOCHEMISTRY: chemical species included)
T
Thermospheric options (relevant if tracer=T) :
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
callthermos  (call thermosphere ?)
F
thermoswater  (WATER: included without cycle    only if water=F)
F
callconduct  (call thermal conduction ?     matter only if callthermos=T)
F
```

```
calleuv  (call EUV heating ?                matter only if callthermos=T)
F
callmolvis  (call molecular viscosity ?     matter only if callthermos=T)
F
callmoldiff  (call molecular diffusion ?    matter only if callthermos=T)
F
thermochem  (call thermospheric photochemistry ?  matter only if callthermos=T)
F
solarcondate (date for solar flux calculation)
1998.
```

- You will need the up-to-date file `jmars.yyyymmdd` (e.g. `jmars.20030707`), which contains the photodissociation rates. You should find it in the `data_mars_gcm` directory of your arborescence.

- **Compilation**

  You need to compile with 15 tracers (if you don't have dust, `dustbin=0`): 13 chemical species (q01 to q13: co2, co, o, o(1d), o2, o3, h, h2, oh, ho2, h2o2, n2, ar) + water ice (last but one tracer) + water vapor (last tracer). Compilation is done with the command lines:

  ```
  makegcm -d 64x48x25 -t 15 -p mars newstart

  makegcm -d 64x48x25 -t 15 -p mars gcm
  ```

- **Run**

  Same as usual. Just make sure that your start files contains the correct number of tracers. If you need to initialize the composition, you can run newstart and use the options

  - ini_q: the 15 tracers are initialized, including water ice and vapor.
  - ini_q-h2o: the 13 chemical species are initialized, water ice is put to zero, and water vapor is kept untouched.
  - ini_q-iceh2o: the 13 chemical species are initialized, water ice and vapor are kept untouched.

  The initialization is done with the files `atmosfera_LMD_may.dat` and `atmosfera_LMD_min.dat`, that should also be found in the `data_mars_gcm` directory.

- **Outputs**

  The outputs can be done from the `aeronomars/calchim.F` routine for the 14 chemical species. The variables put in the diagfi.nc and stats.nc files are labeled (where *name* is the name of the chemical species, e.g. co2):

  - n_*name*: local density (in molecule cm$^{-3}$, 3-dimensional field)
  - c_*name*: integrated column density (in molecule cm$^{-2}$, 2-dimensional field)

# Chapter 10

# 1D version of the Mars model

The physical part of the model can be used to generate realistic 1-D simulations (one atmosphere column). In practice, the simulation is controlled from a main program called `testphys1d.F` which, after initialization, then calls the master subroutine of the physical part `physiq.F` described in the preceding chapters.

## 10.1 Compilation

- For example, to compile the Martian model in 1-D with 25 layers, type (in compliance with the makegcm function manual described in section 5.4)

```
makegcm -d 25 -p mars testphys1d
```

You can find executable **testphys1d.e** (the compiled model) in the directory from which you ran the makegcm command.

## 10.2 1-D runs and input files

The 1-D model does not use an initial state file (the simulation will be long enough to obtain a balanced state). Thus, to generate a simulation simply type:

```
> testphys1d.e
```

The following files are available in the same directory

```
$PATH1/LMDZ.MARS/deftank
```

(copy them into your working directory first):
- **callphys.def** : controls the options in the physical part, just like for the 3D GCM.
- **z2sig.def** : controls the vertical discretization (no change needed, in general), functions as with the 3D GCM.
- **testphys1d.def** : controls the 1-D run.

The last file is specific to the 1-D model. It contains the following:

```
0       ! Initial day (martian sol; = 0 at Ls=0)
0       ! initial local time  (between 0 and 24)
48      ! number of time-step per day.
1       ! Run duration (sols)
700.    ! psurf: Surface pressure (Pa)
0.2     ! tauref: reference dust opacity at 700 Pa (true tau~tauref*psurf/700)
30.     ! latitude  (degrees)
```

```
0.2    ! ground albedo  (under the ice)
400    ! ground thermal inertia  (SI)
10.    ! composante vers l'est du vent geostrophique (u)
0.     ! composante vers le nord du vent geostrophique (v)
0.     ! initial CO2 ice mass (kg.m-2)
T      ! Hybrid vertical coordinate ? T=Hybrid , F=sigma

Initial temperature profile
~~~~~~~~~~~~~~~~~~~~~~~~~~~
5       ! ichoice :  Kind of initial profile (see below)
200     ! tref : reference temperature (K) (used if ichoice=1)
0       ! isin (if isin=1, add a perturbation to profile)
26.5220 ! pic  pic perturbation gauss pour ichoice = 6 ou 7
10      ! largeur de la perturbation gauss pour ichoice = 6 ou 7
30.     ! hauteur de la perturbation gauss pour ichoice = 6 ou 7

c=======================================================================
c   differents profils d'atmospheres. T=f(z)
c         ichoice=1   Temperature constante T=tref
c         ichoice=2   profil Savidjari (comme Seiff mais avec dT/dz=cte)
c         ichoice=3   Lindner (profil polaire)
c         ichoice=4   inversion
c         ichoice=5   Seiff  (profile standard base sur les entrees Viking)
c         ichoice=6    T constante + perturbation gauss (level)
c         ichoice=7    T constante + perturbation gauss (km)
c         ichoice=8   Read in an ascii file "profile"
```

## 10.3 Output data

During the entire 1D simulation, you can obtain output data for any variable from any physical subroutine by using subroutine `writeg1d`. This subroutine creates file `g1d.nc` that can be read by GRADS. This subroutine is typically called at the end of subroutine `physiq` .

Example of a call to subroutine `writeg1d` requesting temperature output: ( `ngrid` horizontal point, `nlayer` layers, variable `pt` called "T" in K units):

```
CALL writeg1d(ngrid,nlayer,pt,'T','K')
```

## Appendix A

# GCM Martian Calendar

For Mars, dates and seasons are expressed in Solar Longitude ($L_s$, in degrees or in radians) counting from the northern hemisphere spring equinox. In the GCM, time is counted in Martian solar days, or “sols” (1 sols = 88775 s) from the northern spring equinox. The following table gives the correspondence between sols and $L_s$, calculated for the GCM using one Martian year = 669 sols exactly.

| sol | $L_s$ | | sol | $L_s$ | | sol | $L_s$ | |
|---|---|---|---|---|---|---|---|---|
| 0. | 360.000 | Spring equinox N | 240. | 111.455 | | 480. | 247.408 | |
| 5. | 2.550 | | 245. | 113.816 | | 485. | 250.666 | |
| 10. | 5.080 | | 250. | 116.190 | | 490. | 253.925 | |
| 15. | 7.590 | | 255. | 118.578 | | 495. | 257.182 | |
| 20. | 10.081 | | 260. | 120.981 | | 500. | 260.435 | |
| 25. | 12.554 | | 265. | 123.400 | | 505. | 263.683 | |
| 30. | 15.009 | | 270. | 125.835 | | 510. | 266.924 | |
| | | | | | | 514.76 | 270. | Winter solstice N |
| 35. | 17.447 | | 275. | 128.287 | | 515. | 270.156 | |
| 40. | 19.869 | | 280. | 130.756 | | 520. | 273.377 | |
| 45. | 22.275 | | 285. | 133.243 | | 525. | 276.587 | |
| 50. | 24.666 | | 290. | 135.750 | | 530. | 279.783 | |
| 55. | 27.043 | | 295. | 138.275 | | 535. | 282.965 | |
| 60. | 29.407 | | 300. | 140.821 | | 540. | 286.130 | |
| 65. | 31.758 | | 305. | 143.388 | | 545. | 289.277 | |
| 70. | 34.096 | | 310. | 145.975 | | 550. | 292.406 | |
| 75. | 36.423 | | 315. | 148.585 | | 555. | 295.515 | |
| 80. | 38.739 | | 320. | 151.217 | | 560. | 298.604 | |
| 85. | 41.046 | | 325. | 153.872 | | 565. | 301.671 | |
| 90. | 43.343 | | 330. | 156.550 | | 570. | 304.715 | |
| 95. | 45.631 | | 335. | 159.251 | | 575. | 307.737 | |
| 100. | 47.912 | | 340. | 161.977 | | 580. | 310.735 | |
| 105. | 50.186 | | 345. | 164.727 | | 585. | 313.709 | |
| 110. | 52.453 | | 350. | 167.502 | | 590. | 316.658 | |
| 115. | 54.714 | | 355. | 170.301 | | 595. | 319.583 | |
| 120. | 56.970 | | 360. | 173.126 | | 600. | 322.483 | |
| 125. | 59.222 | | 365. | 175.975 | | 605. | 325.358 | |
| 130. | 61.471 | | 370. | 178.850 | | 610. | 328.207 | |
| . | | | 371.99 | 180. | Autumn equinox N | . | | |
| 135. | 63.716 | | 375. | 181.750 | | 615. | 331.032 | |
| 140. | 65.959 | | 380. | 184.675 | | 620. | 333.831 | |
| 145. | 68.201 | | 385. | 187.624 | | 625. | 336.606 | |
| 150. | 70.442 | | 390. | 190.598 | | 630. | 339.356 | |
| 155. | 72.683 | | 395. | 193.596 | | 635. | 342.082 | |
| 160. | 74.925 | | 400. | 196.618 | | 640. | 344.783 | |
| 165. | 77.168 | | 405. | 199.662 | | 645. | 347.461 | |
| 170. | 79.413 | | 410. | 202.729 | | 650. | 350.116 | |
| 175. | 81.661 | | 415. | 205.818 | | 655. | 352.748 | |
| 180. | 83.912 | | 420. | 208.927 | | 660. | 355.357 | |
| 185. | 86.167 | | 425. | 212.056 | | 665. | 357.945 | |
| | | | | | | 669. | 0. | Spring equinox N |
| 190. | 88.427 | | 430. | 215.203 | | 670. | 0.512 | |
| 193.47 | 90. | Summer solstice N | | | | | | |
| 195. | 90.693 | | 435. | 218.368 | | 675. | 3.057 | |
| 200. | 92.965 | | 440. | 221.549 | | 680. | 5.583 | |
| 205. | 95.245 | | 445. | 224.746 | | 685. | 8.089 | |
| 210. | 97.532 | | 450. | 227.955 | | 690. | 10.577 | |
| 215. | 99.827 | | 455. | 231.177 | | 695. | 13.046 | |
| 220. | 102.131 | | 460. | 234.409 | | 700. | 15.498 | |
| 225. | 104.446 | | 465. | 237.650 | | 705. | 17.933 | |
| 230. | 106.770 | | 470. | 240.898 | | 710. | 20.351 | |
| 235. | 109.107 | | 475. | 244.151 | | 715. | 22.755 | |
| 240. | 111.455 | | 480. | 247.408 | | 720. | 25.143 | |

# Appendix B

# Utilities

A few post-processing tools, which handle GCM outputs (files `diagfi.nc` and `stats.nc`) are available on the web at:
`http://www.lmd.jussieu.fr/˜forget/datagcm/Utilities/`
The directory contains compiled executables (`*.e` files) of the tools decribed below, along with some examples of input instruction (`*.def` files) and a `README`.
There is also a `SOURCES` directory which contains the (Fortran) sources of the codes, if you should need to recompile them on your platform.

## B.1 concatnc

This program concatenates consecutive output files (`diagfi.nc` or even `stats.nc` files) for a selection of variable, in order to obtain one single big file. The time dimension of the output can be ”sols” or ”Ls” (note that in that latter case, Ls values won't be evenly distributed, and software like Grads may not be able to use and plot the data).
To obtain an evenly sampled ”Ls” timescale, you can use the `lslin.e` program (described below).
The output file created by `conctanc.e` is `concat.nc`

## B.2 lslin

This program is designed to interpolate data given in irregular Solar Longitude (Ls) into an evenly sampled linear time coordinate (usable with Grads). Input Netcdf files may be `diagfi.nc` or `concat.nc` files and the resulting output file is `lslin.nc` lslin also create a `lslin.ctl` file that can be read directly by grads (`>xdfopen lslin.ctl`) to plot in Ls coordinate to avoid some problem with grads when Grads think that ”the time interval is too small”...

## B.3 localtime

The `localtime.e` program is designed to re-interpolate data in order to yield values at the same given local time (useful to mimic satellite observations, or analyse day to day variations at given local time).
Input files may be of `diagfi.nc`, `stats.nc` or `concat.nc` type and the output file name is build from the input one, to which `_LT.nc` is appened (e.g. if the input file is `myfile.nc` then output file will be `myfile_LT.nc`).

## B.4 zrecast

With this program you can recast atmospheric (i.e.: 4D-dimentional longitude-latitude-altitude-time) data from GCM outputs (e.g. as given in `diagfi.nc`, `concat.nc` and `stats.nc` files) onto either *pressure* or *altitude above areoid* vertical coordinates.
Since integrating the hydrostatic equation is required to recast the data, the input file must contain surface pressure and atmospheric temperature, as well as the ground geopotential.
If recasting data onto *pressure* coordinates, then the output file name is given by the input file name to which `_P.nc` will be appened. If recasting data onto *altitude above areoid* coordinates, then a `_A.nc` will be appened.

# Bibliography

[1] R. T. Clancy and S. W. Lee. A new look at dust and clouds in the Mars atmosphere: Analysis of emission-phase function sequences from global Viking IRTM observations. *Icarus*, 93:135–158, 1991.

[2] J.-L. Dufresne, R. Fournier, C. Hourdin, and F. Hourdin. Net Exchange Reformulation of Radiative Transfer in the $CO_2$ 15-$\mu$m Band on Mars. *Journal of Atmospheric Sciences*, 62:3303–3319, 2005.

[3] F. Forget. Improved optical properties of the Martian atmospheric dust for radiative transfer calculations in the infrared. *Geophys. Res. Lett.*, 25:1105–1109, 1998.

[4] F. Forget, F. Hourdin, R. Fournier, C. Hourdin, O. Talagrand, M. Collins, S. R. Lewis, P. L. Read, and J.-P. Huot. Improved general circulation models of the Martian atmosphere from the surface to above 80 km. *J. Geophys. Res.*, 104:24,155–24,176, 1999.

[5] F. Forget, F. Hourdin, and O. Talagrand. $CO_2$ snow fall on Mars: Simulation with a general circulation model. *Icarus*, 131:302–316, 1998.

[6] Y. Fouquart and B. Bonnel. Computations of solar heating of the Earth's atmosphere: A new parametrization. *Contrib. Atmos. Phys.*, 53:35–62, 1980.

[7] C. Hourdin, J.-L. Dufresnes, R. Fournier, and F. Hourdin. Net exchange reformulation of radiative transfer in the $CO_2$ 15$\mu$m band on mars. Article in preparation, 2000.

[8] F. Hourdin. A new representation of the $CO_2$ 15 $\mu$m band for a Martian general circulation model. *J. Geophys. Res.*, 97(E11):18,319–18,335, 1992.

[9] F. Hourdin and A. Armengaud. Test of a hierarchy of finite-volume schemes for transport of trace species in an atmospheric general circulation model. *Mon. Wea. Rev.*, 127:822–837, 1999.

[10] F. Hourdin, P. Le Van, F. Forget, and O. Talagrand. Meteorological variability and the annual surface pressure cycle on Mars. *J. Atmos. Sci.*, 50:3625–3640, 1993.

[11] S. Lefèvre, S. Lebonnois, F. Montmessin, and F. Forget. Three-dimensional modeling of ozone on Mars . *Journal of Geophysical Research (Planets)*, 109:E07004, 2004.

[12] S. R. Lewis, M. Collins, P. L. Read, F. Forget, F. Hourdin, R. Fournier, C. Hourdin, O. Talagrand, and J.-P. Huot. A climate database for Mars. *J. Geophys. Res.*, 104:24,177–24,194, 1999.

[13] F. Lott and M. Miller. A new sub-grid scale orographic drag parametrization: its formulation and testing. *Q. J. R. Meteorol. Soc.*, 123:101–128, 1997.

[14] F. Montmessin, F. Forget, P. Rannou, M. Cabane, and R. M. Haberle. Origin and role of water ice clouds in the Martian water cycle as inferred from a general circulation model. *Journal of Geophysical Research (Planets)*, 109(E18):10004–+, 2004.

[15] M. E. Ockert-Bell, J. F. Bell III, C. McKay, J. Pollack, and F. Forget. Absorption and scattering properties of the Martian dust in the solar wavelengths. *J. Geophys. Res.*, 102:9039–9050, 1997.

[16] N. O. Rennò, M. L. Burkett, and M. P. Larkin. A simple thermodynamical theory for dust devils. *J. Atmos. Sci.*, 55:3244–3252, 1998.

[17] O. B. Toon, C. P. McKay, T. P. Ackerman, and K. Santhanam. Rapid calculation of radiative heating rates and photodissociation rates in inhomogeneous multiple scattering atmospheres. *J. Geophys. Res.*, 94:16,287–16,301, 1989.