

How to make an unexpensive installation of the MM5 Modelling Suite

T. Javier Robles Prado

December 16, 2005

Contents

0.1	Getting the software	2
0.2	Previous Steps	3
0.3	TERRAIN	5
0.4	REGRID	10
0.5	Objective Analysis: LITTLE_R and RAWINS	13
0.5.1	LITTLE_R	13
0.5.2	RAWINS	14
0.6	INTERPF	17
0.7	MM5	18
0.8	INTERPB	20
0.9	Viewing results	21
0.9.1	Vis5d	21
0.9.2	GrADS	28
0.10	Using more processors: MPP	35
0.10.1	Setting mpich	35
0.10.2	Compiling MM5-MPP	37
0.11	About this document	41

PREFACE

The MM5 mesoscale model is a limited-area, nonhydrostatic, terrain-following sigma-coordinate model designed to simulate or predict mesoscale atmospheric circulation. It works using several programs written mainly in Fortran to pre-process inputs and post-process outputs. It has been developed at Penn State and NCAR and has received contributions from users worldwide.

The MM5 modeling system is freely provided and supported by the Mesoscale Prediction Group in the Mesoscale and Microscale Meteorology Division, NCAR.

MM5 code is quite portable and runs on many Unix flavors. In this document, I am going to describe the installation process of the model on a Linux box with the Intel Fortran Compiler. You can get a lot of Linux distributions for free, and Intel provides its Fortran Compiler for free for non-commercial uses. At this moment, it is not possible to compile MM5 with GNU Fortran compiler.

This document does not describe the installation of Linux, and you should be familiar with Unix commands to read it. If you have never used an Unix machine before, you should read something about it before trying to install MM5.

0.1 GETTING THE SOFTWARE

There is some software you need to install and run MM5. First of all, you need a Linux box. There are several distributions of Linux, if you are familiar with one of them you should choose that particular one. For this document, I am going to use Mandriva Linux 2006, which you can download for free from <ftp://www.mandrivalinux.com>. Mandriva is a nice distribution for new and not so new Linux users. It is designed to be easy to use and has one of the most powerful package tool, *urpmi*, which, if it is properly configured, can get almost all the software you need and install it in one command line.

Although this text is based on a installation in Mandriva Linux, you should not need to do too many changes to work with your favourite Linux distribution.

Once you have Linux properly installed, it is time to get a Fortran Compiler. Currently MM5 can be compiled with the Portland Group Fortran Compiler and with the Intel Fortran Compiler. Intel provides a free license for non-commercial uses of its compilers, so I will use it. You have to register to download the software and to obtain a valid license key. If you need more information about this, you can visit <http://www.intel.com>.

The next thing you should do is to download NCAR Graphics. Although it is not strictly necessary, they are needed to see TERRAIN and other plot files. You can find NCAR Graphics and download them for free at <http://ngwww.ucar.edu>.

At this point, you can download the programs which compound the MM5 modelling system. You can go to <ftp://ftp.ucar.edu/mesouser/MM5V3> and download these programs: TERRAIN, REGRID, LITTLE_R, INTERPF and MM5. These are the programs you need to run the *Storm of the Century Case* which comes as an example to do a simulation. In this text I will cover some other programs such as GrADS or Vis5d you might find useful.

0.2 PREVIOUS STEPS

This section covers the installation of the Fortran compiler, NCAR Graphics and some other steps which will help you later.

Install Intel Fortran Compiler is easy. You only have to untar the file you downloaded and run the `install.sh` script. The installer will ask you some questions (including the License Key which you should have received) and then will install the software. The default installation directory is `/opt` which is not bad.

Once you have installed the compiler it is time to do some changes in your `/.bashrc` file. I assume you are using Bash shell because it is the most used shell in Linux distributions. `/.bashrc` is a file which is executed each time you open a new terminal. I will use this file to modify some environment variables such as `PATH` or `LD_LIBRARY_PATH`, to include Intel Fortran binaries and libraries. So, you can add these lines at the end of your `/.bashrc` file.

```
export PATH=$PATH:/opt/intel/fc/9.0/bin
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/opt/intel/fc/9.0/lib
```

To test this, you can open a new terminal and type:

```
$ ifort -v
Version 9.0
```

Now, let's install NCAR Graphics. The installation is also easy. Once you have untarred the file you downloaded, you have to enter `INSTALL` directory and run `INSTALL` script. You will have to answer some questions including the directory where you want to install the software. The default directory is `/usr/local/ncarg`, so you will need to have `root` permissions to create it and to install there. If you do not have `root` permissions, ask to your system administrator or install NCAR in other location where you have permissions.

After that, modify your `/.bashrc` file to include NCAR paths and NCAR variables:

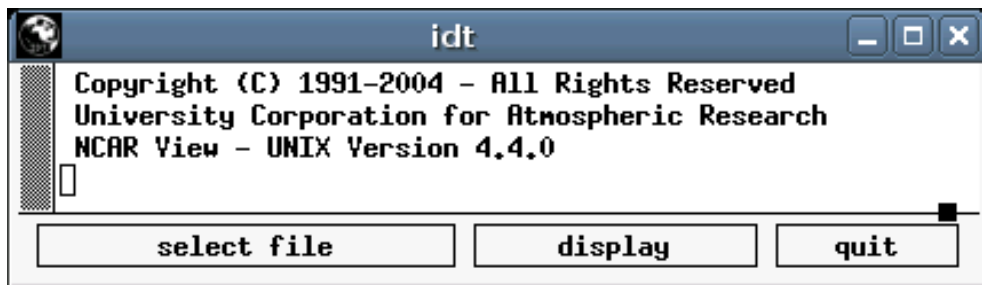


Figure 1: Idt running

```
export PATH=$PATH:/opt/intel/fc/9.0/bin:/usr/local/ncarg/bin
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/opt/intel/fc/9.0/lib:/usr/local/ncarg/lib
export NCARG_ROOT=/usr/local/ncarg
```

Now if you open a new terminal and type `idt`, you will see something like figure 1.

At this moment you are ready to start with the first program in the MM5 suite: `TERRAIN`, so create a new directory called `mm5` and go on.

0.3 TERRAIN

This is the first program you need to run in order to simulate a meteorological situation with the MM5 model. With this program you define the geographical area in which you are interested, and some other areas you could be interested.

First of all, untar the TERRAIN files in the `mm5` directory you created and let's have a look to the files of the TERRAIN directory:

```
$ ls
CHANGES*      con.tbl*  lsco.tbl*  lvco.tbl*   map.tbl*    src/
confiP.tbl*    Data/    luco.tbl*  Makefile*   psids*      Templates/
confi.tbl*     Diff/    lvcl.tbl*  maparea.tbl* raobsta.ieee*
confiT.tbl*    ezids*   lvc2.tbl*  mapfi.tbl*  README*
```

We are interested in `Makefile` which is a file that tells `make` how to build the program. A `Makefile` has usually two parts. In the first one, there are some variables defined, in the second, there are `targets` which build the `TERRAIN` program for each platform supported. We are interested in the `intel` target:

```
intel:
    echo "Compiling for Linux using INTEL compiler"
    ( $(CD) src ; $(MAKE) all          \
    "RM              = $(RM) "         \
    "RM_LIST         = $(RM_LIST) "     \
    "LN              = $(LN) "         \
    "MACH            = SGI "           \
    "MAKE            = $(MAKE) "       \
    "CPP             = /lib/cpp"       \
    "CPPFLAGS        = -I. -C -traditional -D$(NCARGGRAPHICS) "\
    "FC              = ifort "         \
    "FCFLAGS         = -I. -w90 -w95 -convert big_endian "\
    "LDOPTIONS       = -i_dynamic"     \
```

```
"CFLAGS          = -I." \
"LOCAL_LIBRARIES= -L$(NCARG_ROOT)/lib -L/usr/X11R6/lib \
    -lncarg -lncarg_gks -lncarg_c -lX11 \
    -L/usr/lib/gcc-lib/i386-redhat-linux/3.3.3 -lg2c" );\
( $(RM) terrain.exe ; $(LN) src/terrain.exe . ) ;
```

Note that the code listed above is not exactly the same you will find in your Makefile. I have modified it to improve formatting and to see it clearer. If you have followed the steps listed in the previous chapter, there is only one change to do. The location of the library `lg2c` can vary from one installation to another. Even it is quite probable that it is not present on your system. To get this library you will need to install the `gcc-gfortran` package¹. In Mandriva is easy if you have `urpmi` properly installed and configured. Just type `urpmi gcc-gfortran` and it will do the hard work. If you are using a different distribution and this does not work for you, try to install GNU Fortran compiler, which includes this library.

Next step consists on search the location of the `libg2c` file. You can type something like this:

```
$ find /usr -name "*libg2c*"
/usr/lib/libg2c.so.0.0.0
/usr/lib/libg2c.so.0
```

As you see there is not a `libg2c.so` file which is the file that will be looked for. Just create a symbolic link (you will need to be root):

```
$ ln -s /usr/lib/libg2c.so.0 /usr/lib/libg2c.so
```

Now it is time to modify the Makefile to indicate the righth location of `lg2c`:

```
intel :
    echo "Compiling for Linux using INTEL compiler"
    ( $(CD) src ; $(MAKE) all          \
    "RM          = $(RM)"              \
```

¹If you are using previous versions of Mandriva or Mandrake Linux, the package is `gcc-g77`

```

"RM_LIST      = $(RM_LIST) " \
"LN           = $(LN) " \
"MACH         = SGI " \
"MAKE        = $(MAKE) " \
"CPP         = /lib/cpp " \
"CPPFLAGS    = -I. -C -traditional -D$(NCARGGRAPHICS) "\
"FC          = ifort " \
"FCFLAGS     = -I. -w90 -w95 -convert big_endian "\
"LDOPTIONS   = -i_dynamic " \
"CFLAGS      = -I. " \
"LOCAL_LIBRARIES= -L$(NCARG_ROOT)/lib -L/usr/X11R6/lib \
               -lncarg -lncarg_gks -lncarg_c -lX11 \
               -L/usr/lib -lg2c " );\
( $(RM) terrain.exe ; $(LN) src/terrain.exe . ) ;

```

Now, type `make intel` to compile the code and be sure it finishes with no errors. If so, type `make terrain.deck`. After that, two new files will be created: `terrain.deck` and `terrain.deck.intel`. TERRAIN program is able to download the data it need, but according to my own experience it is better you download all the Terrain files. They can be download from ftp://ftp.ucar.edu/mesouser/MM5V3/TERRAIN_DATA. In my case, I have created a directory called `/mnt/data/terrain_data`. A fast way to download and unzip the files could be:

```

$ cd /mnt/data/terrain_data
$ wget ftp://ftp.ucar.edu/mesouser/MM5V3/TERRAIN_DATA/*
$ for x in `ls -l *.gz`; do gunzip $x; done

```

Once you have downloaded and gunzipped terrain files, it is time to modify `terrain.deck.intel` to configure the path where we have terrain files:

```

set ftpdata = false
#
# Set the following for ftp'ing 30 sec
# elevation data from USGS ftp site

```

```
#
```

```
set Where30sTer = /mnt/data/terrain_data
```

At the moment you are ready to run `terrain.deck.intel`. Note that this will compile again the terrain code. When it finishes, have a look at the last two lines of the `terrain.printout` file. It should be something like:

```
$ tail -2 terrain.print.out
== NORMAL TERMINATION OF TERRAIN PROGRAM ==
99999
```

If not, look at the file to find more information. Common problems are:

- You do not have the terrain data files needed.
- Your domain configuration is not correct

If you are going to use more detailed terrain files than the default ones, you should create symlinks in your `TERRAIN/Data` for those files. So, if you have your terrain files in `/mnt/data/terrain_data` you could do:

```
$ ln -s /mnt/data/terrain_data/* TERRAIN/Data/
```

The default `terrain.deck.intel` domain configuration is the *Storm of the Century Case*. So if you type:

```
$ idt TER.PLT
```

you will see something like the figure 2.

After a successful execution of terrain program, you will have a `TERRAIN_DOMAIN` file for each domain you have defined. This files will be the input to other programs such as `regridder` and `MM5`.

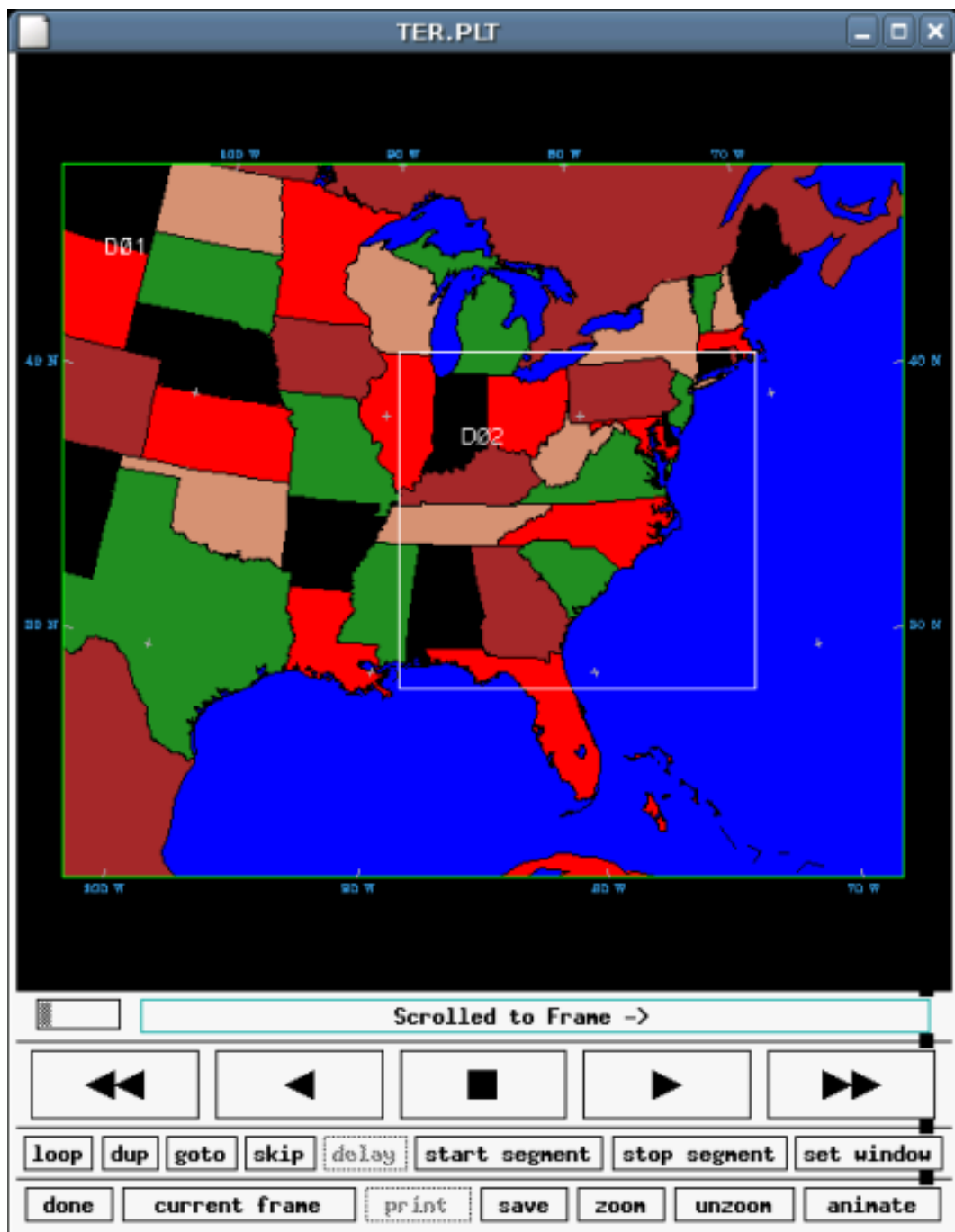


Figure 2: TER.PLT

0.4 REGRID

Once you have run successfully TERRAIN program it is time to run REGRID. REGRID is composed by two programs: `pregrid` and `regridder`. Untar the REGRID.TAR.gz file and type `make intel`. You do not have to make any changes here.

You should be reading the MM5 tutorial (which you can find in <http://www.mmm.ucar.edu/mm5/On-Line-Tutorial/teachyourself.html>) at the same time you are reading this text. That tutorial gives you more information about you are doing in each step, and will guide you to simulate the *Storm of the Century* case. So, if have read that document you will have downloaded NCEP_ON84.9303 which is an input file for `pregrid`. I have put it in `/mnt/data/pregrid_data`, so I will edit `pregrid.csh` file to indicate the place where the program will look for data:

```
#
# Put your input files for pregrid into
# the directory you specify as DataDir:
#
set DataDir = /mnt/data/pregrid_data
```

Now, you are ready to run `pregrid.csh`:

```
$ ./pregrid.csh
```

If everything went OK, you will see these files in your `pregrid` directory:

```
Doc/          nise/          ON84_SNOW:1993-03-13_00  pregrid.csh*
era/          nnrp/          ON84_SNOW:1993-03-13_12  pregrid_era40_int.csh*
grib.misc/    on84/          ON84_SNOW:1993-03-14_00  pregrid.namelist
Makefile*     ON84:1993-03-13_00  ON84_SST:1993-03-13_00  README_ERA40
navysst/     ON84:1993-03-13_12  ON84_SST:1993-03-13_12  toga/
ncep.grib/    ON84:1993-03-14_00  ON84_SST:1993-03-14_00  util/
```

`pregrid` outputs can be plotted. In the `pregrid` directory you will find an `util` directory. In it,

there is the source of an utility called `plotfmt`. To compile it, you will have to do a pair of changes in the Makefile. Just change the value of `NCARG_LIBS` as follows:

```
NCARG_LIBS=      -L$(NCARG_ROOT)/lib  \
                 -lncarg -lncarg_gks -lncarg_c \
                 -L/usr/X11R6/lib -lX11 -lm  \
                 -L/opt/intel/fc/9.0/lib -L/usr/lib -lg2c
```

Now, type `make plotfmt`, and `plotftm` utility will be compiled. If you have no errors, you will see `plotftm` binary file. Just run it with one of the `pregrid` generated files as parameter to plot it:

```
$ ./plotfmt ../ON84:1993-03-14_00
```

After a while, if there is not any problem, you will see the program stops with a *Graceful Stop* message. A new file called `gmeta` have been created. You can plot it with `idt` program to see something like figure 3.

```
$ idt gmeta
```

Next step is running `regridder`. If you are following the MM5 tutorial and you are simulating *The Storm of The Century* Case you will not need to do any change. So, move to `regridder` directory and type:

```
$ ./regridder
```

If everything went fine, you will have a new file called `REGRID_DOMAIN1` which will be used by other programs in the next steps.

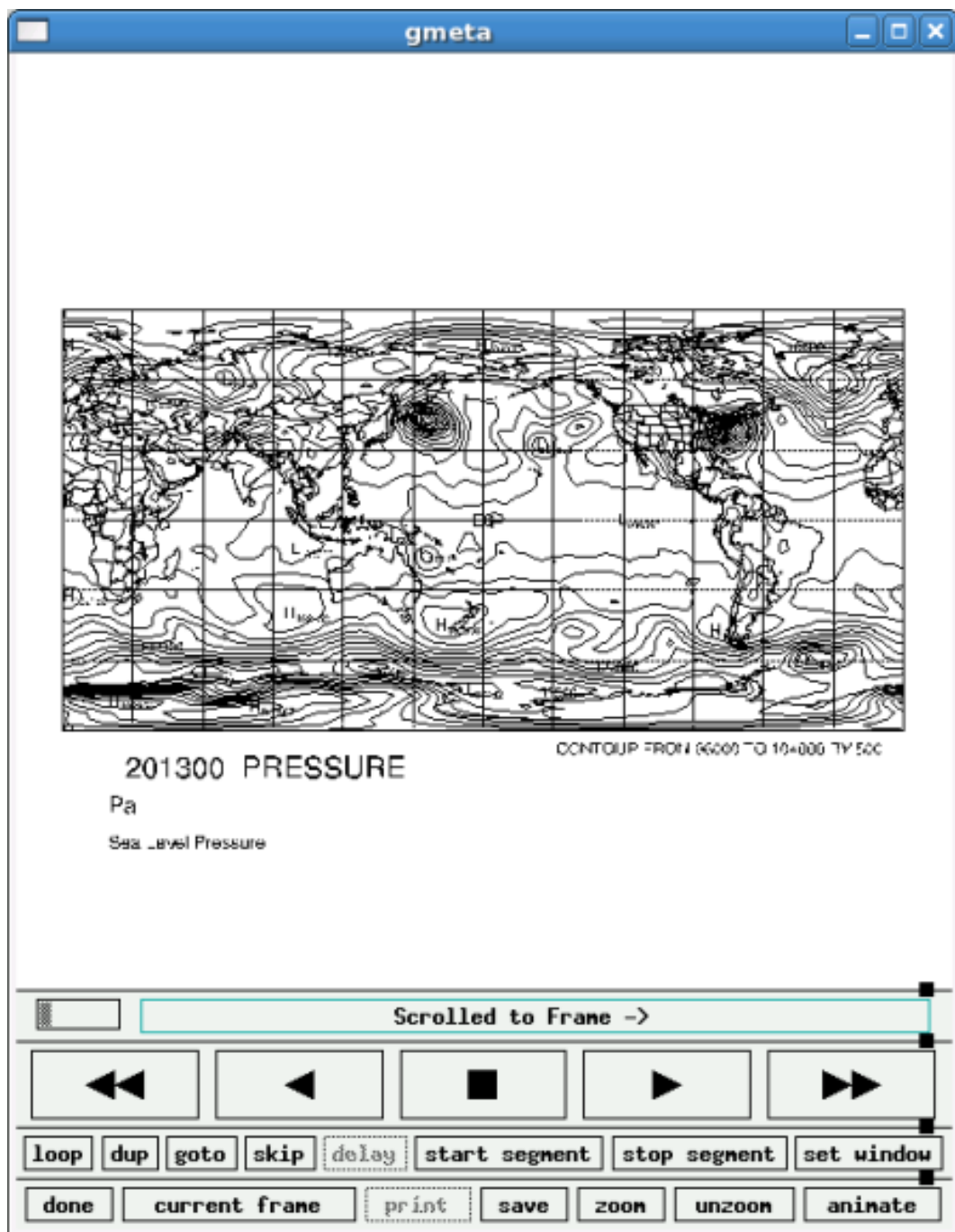


Figure 3: plotfmt plot

0.5 OBJECTIVE ANALYSIS: LITTLE_R AND RAWINS

If you are planning to add Objective Analysis to your simulations, you will need LITTLE_R or RAWINS. Although MM5 documentation says that LITTLE_R should be used because it is easier to use, both of them will be covered in this section.

0.5.1 LITTLE_R

Once you have untared the LITTLE_R.TAR.gz file, it is time to compile it. There is only one change to do to the Makefile. Just search for the string `-L/usr/lib/gcc-lib/i386-redhat-linux/3.3.2` and replace it by `-L/usr/lib` (or the path where `libg2c.so` can be found). Compile LITTLE_R by typing:

```
$ make intel
```

If you do not have any error, you can continue. Download and untar the test data files as described in the MM5 Tutorial. Once you have untared the file, you will see some gzipped files. A little bit of bash scripting is useful to uncompress them:

```
$ for x in `ls -l *.gz`; do gunzip $x; done
```

I will store them in `/mnt/data/little_r_data`, so I will modify some lines in the `namelist.input` to change `Test_Data` by `/mnt/data/little_r_data`. Now run `little_r`:

```
$ ./little_r
```

When it finishes, you will see a `LITTLE_R_DOMAIN1` which will be one of the inputs of the MM5 program.

0.5.2 RAWINS

The compilation of RAWINS has no problems. The modification you have todo in the RAWINS is the same as the described in LITTLE_R (see 0.5.1 subsection). So, the typical `make intel` will work.

If you are planning to run RAWINS in the SOC case you will have to download some data files as described in the MM5 Tutorial. Now, type:

```
$ make rawins.deck
```

Two new files, `rawins.deck` and `rawins.deck.intel` will be created. Next step consists on editing the `rawins.deck.intel` file to indicate where data resides. I have modified the value of these variables:

```
#      locations for datasets
#
set InDatg  =  ../REGRID/regridder/REGRID_DOMAIN1
#
#
#      Find MSS names for observations from the following catalogs:
#      upper air: /fs/othrorgs/home0/mesouser/catalog/catalog.raob
#      surface: /fs/othrorgs/home0/mesouser/catalog/catalog.sfc
#
if ( $INOBS == ARCHIVE ) then
  set IfUni = F
  set InRaobs  =  ( /mnt/data/rawins_data/RAOBS_A )
  if ( $SFCsw == SFC ) then
    set InSfc6h  =  ( /mnt/data/rawins_data/SFC6HR_A )
    set InSfc3h  =  ( /mnt/data/rawins_data/SFC6HR_A )
  endif
else
  set IfUni = T
  set upa_unidate = ( xxxxxxxx )          # YYMMDDHH
  set sfc_unidate = ( xxxxxxxx )          # YYMMDDHH
```

```

endif
#
#   _____ RAWINS PARAMETER STATEMENT _____
#
if ( -e src/paramdim.incl ) rm src/paramdim.incl
cat > src/paramdim.incl << EOF
C
C  IMX,JMX, MUST CORRESPOND TO THE DIMENSIONS IN THE INPUT FILE.  THESE
C  WILL BE THE EXPANDED DIMENSIONS IF THIS IS THE COARSE GRID, AND THE
C  EXPANDED OPTION WAS SELECTED.
C
C  LMX MUST BE GREATER THAN OR EQUAL TO THE MAXIMUM NUMBER OF LEVELS
C  (PRESSURE LEVELS + SURFACE).
C
C          PARAMETER(IMX=35,JMX=41,LMX=22)
C_____
EOF

[ ... ]

IPLOT=T,F,F,F,F,F,F,F,F,F,F,    # PLOT VERTICAL RAOBS (As directed
#                                  # by ISKEWT).

```

Noy type `make clean` and after that, run `rawins.deck.intel`. If it executes without problems, you will see some `*_DOMAIN1` files which should be useful for you later. If you want to plot RAWINS output, you can use `idt` to see `SND.PLT`. You will see something like figure 4:

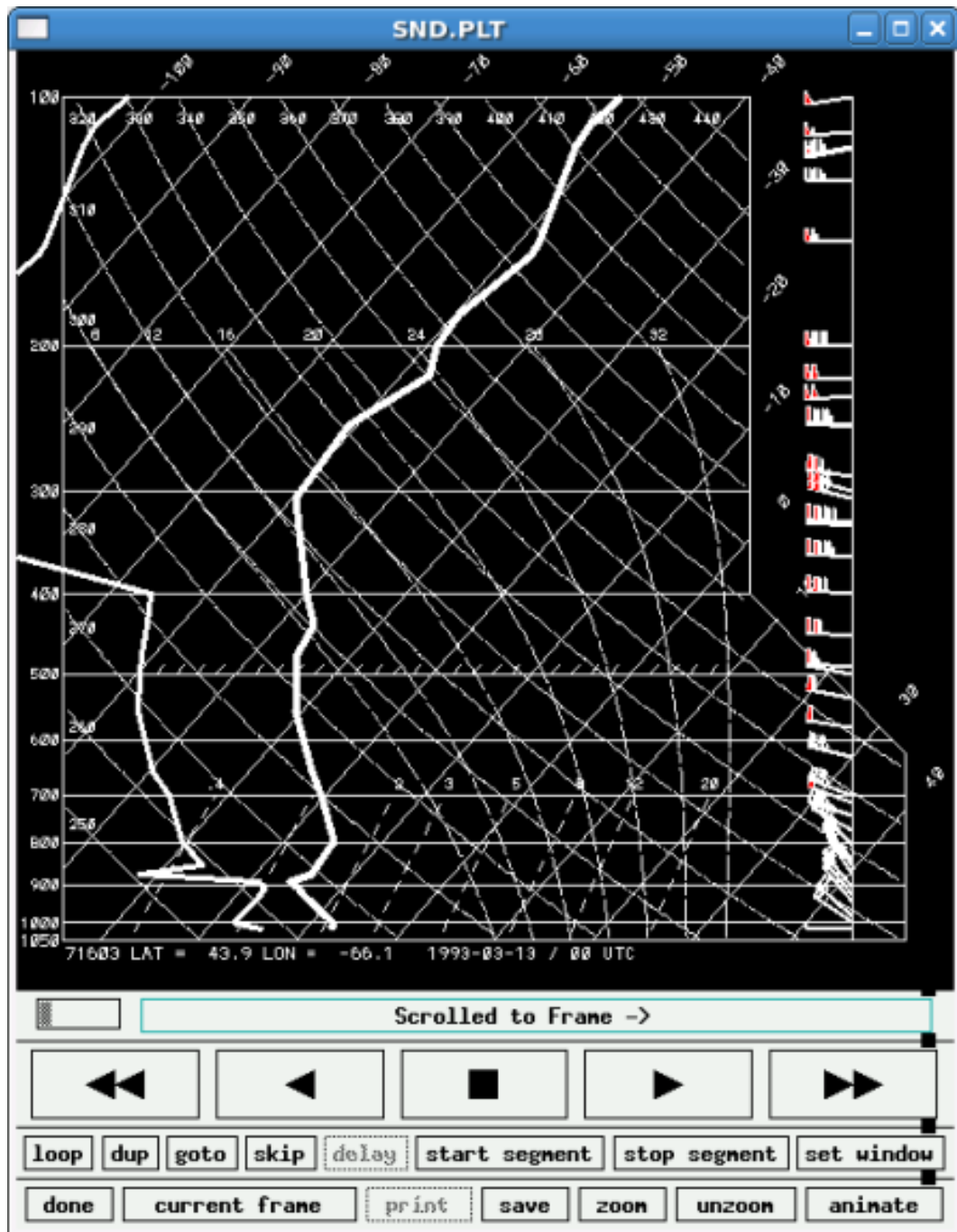


Figure 4: SND.PLT

0.6 INTERPF

INTERPF performs interpolations in different pressure levels. To compile INTERPF is really easy. You do not have to do any changes to the `Makefile` file, so simply type `make intel`.

If you are simulating *SOC* Case, you will not need to change `namelist.input` file, so type:

```
$ ./interpf
```

After that, these files will be created: `MMINPUT_DOMAIN1`, `LOWBDY_DOMAIN1` and `BDYOUT_DOMAIN1` and will be used as inputs for the MM5 program.

0.7 MM5

Now it is time to compile and run a simulation. The first step to do it is to ensure that you have done the previous steps with no errors. If so, you will have the input files needed. Go to your `MM5/Run` directory and create symbolic links to these files²:

```
$ ln -s ../../INTERPF/MMINPUT_DOMAIN1 .
$ ln -s ../../INTERPF/BDYOUT_DOMAIN1 .
$ ln -s ../../INTERPF/LOWBDY_DOMAIN1 .
$ ln -s ../../TERRAIN/TERRAIN_DOMAIN2 .
```

Now, go back to the `MM5` directory and have a look to the `configure.user` file. Look for the `3i2.PC_INTEL (LINUX/INTEL)` section and uncomment those options. You will have something like:

```
#-----
# 3i2. PC_INTEL (LINUX/INTEL)
#-----
RUNTIME_SYSTEM = "linux"
FC = ifort
FCFLAGS = -I$(LIBINCLUDE) -O2 -tp p6 -pc 32 -convert big_endian
CPP = /lib/cpp
CFLAGS = -O
CPPFLAGS = -I$(LIBINCLUDE)
LDOPTIONS = -O2 -tp p6 -pc 32 -convert big_endian
LOCAL_LIBRARIES =
MAKE = make -i -r
```

Type `make` to compile it. Remember that if you change any value in the `configure.user` file, you will have to recompile the model. Once it has been compiled, type `make mm5.deck`. A new `mm5.deck` file will be created. If you are running the example case, you will not have to modify anything, so run it:

²I prefer linking files than copying them, if you do not, it is up to you

```
$ ./mm5.deck
```

Now go to your Run directory and type:

```
$ ./mm5.exe
```

Depending on your machine, go for a cup of coffee or to sleep. In my case, I am running the example simulation on a dual Intel Xeon machine³, and the simulation has taken five minutes.

Congratulations! You have run your first simulation!

³Note that if you have a smp machine, only one processor will be used unless you have compiled a MPP model.

0.8 INTERPB

If you remember, `INTERPF` interpolates pressure levels to normalize them. The outputs of the MM5 model use these interpolated and normalized pressure levels. So, if you want to work with real pressure levels, you will have to use `INTERPB`.

Download `INTERPB` from the same place where you downloaded the others program of the MM5 modelling suite. Uncompress it and type `make intel` to compile it.

Edit the `namelist.input` file to determine the dates and the interval for the program. Once you are ready, run `interpb`.

0.9 VIEWING RESULTS

Previous chapters are completely useless if you are not able to view the results of your simulation properly. There are many ways to see the outputs of the model, in this text I will cover two of them, which are quite interesting.

One of the best tool to work with scientific data is `Vis5d`. It is very useful if you want to work with the results of a simulation in a interactive way. But if you are planning to automatize processes to get some information from your simulations, `GrADS` is your choice.

0.9.1 `Vis5d`

Installation

`Vis5d` is a project currently developed under the name of `Vis5d+` which you can find at <http://vis5d.sourceforge.net/>. Although you can compile and build this last version and use it with no problems, in this text I am going to use a previous version which can work much easier in most machines.

So, go to <http://www.ssec.wisc.edu/~billh/vis5d.html> and download both `vis5d-5.1.tar.Z` and `vis5d-data.tar.Z`. Uncompress and untar both files:

```
$ zcat vis5d-5.1.tar.Z | tar -xv
$ zcat vis5d-data.tar.Z | tar -xv
```

Go into `vis5d-5.1` directory and type:

```
$ make linux-x
```

to compile it. Note that this compilation will not use 3-D hardware rendering but will work on most machines. If you have hardware acceleration properly configured in your system, you should try `make linux-opengl` instead, or if you have a NvidiaTM Graphics Card `make linux-nvidia`.

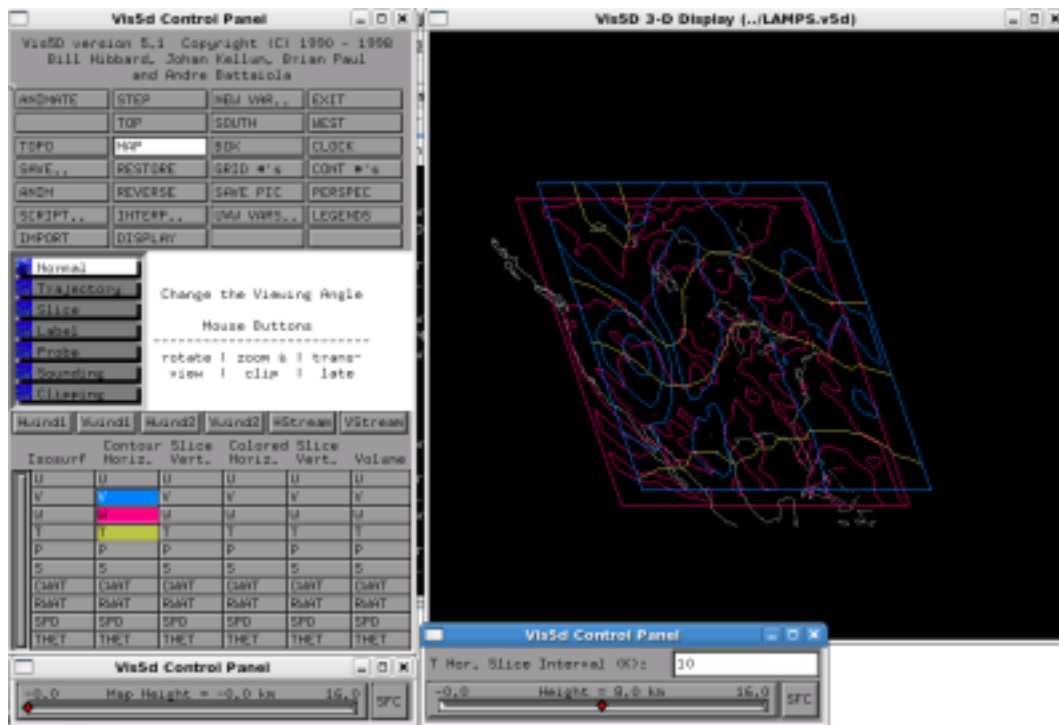


Figure 5: Vis5d in action

Once it has been compiled, you can try it with the data you downloaded. So, if the data files are in the parent directory from your vis5d binary file, type:

```
$ ./vis5d -map ../OUTLSUPW -topo ../EARTH.TOPO ../LAMP5.v5d
```

After playing for a while with the different options, you could see something like figure 5.

It is a good idea to store the vis5d directory in a standard place. I would suggest you storing your vis5d-5.1 directory in /usr/local and then create a symlink called vis5d to vis5d-5.1. It is just an advice, you can follow or not.

You should add the path of your vis5d binary file to your PATH environment.

Using TOVIS5D

TOVIS5D is a tool able to convert MM5 outputs to v5d files suitable for vis5d. You can download it from <ftp://ftp.ucar.edu/mesouser/MM5V3/Util>.

Once you have uncompressed and untarred it, it is time to make some changes in the Makefile file. Find linux section and replace the whole section for this:

```
linux :
    cd src/ ; $(MAKE) target \
    "FC = ifort " \
    "FCFLAGS = -free -DLINUX -I. -convert big_endian " \
    "CCFLAGS = -g -DLITTLE -DUNDERScore -c" \
    "LIBS = "
    $(RM) tovis5d ; $(LN) src/tovis5d .
```

Type make linux to compile. Once it is compiled, and assuming that the path of MMOUT_DOMAIN1 is correct, type:

```
$ ./tovis5d.csh ../MM5/Run/MMOUT_DOMAIN1
```

This will create an user.in file. This is the file you should edit for special options. Once you have looked at it, type:

```
$ ./tovis5d ../MM5/Run/MMOUT_DOMAIN1
```

A new file called vis5d.file will be created. This is a Vis5d file, so you can use vis5d to plot it.

Using MM5p_to_v5d

MM5p_to_v5d is much more useful than TOVIS5D but it requires some previous steps not to be done with TOVIS5D.

First of all, you can download MM5p_to_v5d from *Red Ibérica MM5* web page, which you can find at <http://redibericamm5.uib.es/>. You have to download three files:

- MM5p_to_v5d.f: http://redibericamm5.uib.es/algoritmos/grupo_01/g01_al04_convert/MM5p_to_v5d.f
- MM5p_to_v5d.f.m: http://redibericamm5.uib.es/algoritmos/grupo_01/g01_al04_convert/MM5p_to_v5d.f.m
- pars.MM5p: http://redibericamm5.uib.es/algoritmos/grupo_01/g01_al04_convert/pars.MM5p

You should create a new directory called MM5p_to_v5d and place them there. Once you have moved the files, rename MM5p_to_v5d.f.m to Makefile. It is much clearer.

Edit the Makefile file, and edit the highlighted variables:

```
# makefile for nos_to_v5d.f conversion program

# The input to the conversion program is your 5-D grid format.
# The output is a v5d file.

# By default, the name of the conversion program is 'foo2_to_v5d'.
You should
# probably use a better name. Assign that name to PROGRAM here:

PROGRAM = MM5p_to_v5d

# If DEC or Linux, (Little-endian), add -DLITTLE to CFLAGS
# If AIX, remove the -DUNDERSCORE
CFLAGS = -c -g -DUNDERSCORE -DLITTLE
FFLAGS = -c -g -convert big_endian
CC = cc
```

```

F77=ifort
LIBS = -lm -i_dynamic -Vaxlib

OBJECTS = $(PROGRAM).o binio.o v5d.o

$(PROGRAM): $(OBJECTS)
    $(F77) $(OBJECTS) $(LIBS) -o $@

$(PROGRAM).o: $(PROGRAM).f
    $(F77) $(FFLAGS) -I/usr/local/vis5d/src $(PROGRAM).f

binio.o: /usr/local/vis5d/src/binio.c
    $(CC) $(CFLAGS) /usr/local/vis5d/src/binio.c -o binio.o

v5d.o: /usr/local/vis5d/src/v5d.c
    $(CC) $(CFLAGS) /usr/local/vis5d/src/v5d.c -o v5d.o

```

Note that the Makefile refers to some files that should reside in /usr/local/vis5d. If you followed my advice (see 0.9.1) you do not have to edit the path of the files. If you did not follow my advice, you will have to modify the paths in the Makefile and in the source code of MM5p_to_v5d.

Type `make` to compile the program.

MM5p_to_v5d needs to work real pressure levels, not the normalized ones. So you will have to run INTERPB (see 0.8) before running. Now edit `pars.MM5p`:

```

#### USER CHOICES TO PRODUCE V5D file from pressure-surfaces MM5p output
#
*** MM5p output file
'../../INTERPB/MMOUTP_DOMAIN1'
#
*** Requested BEGIN date in format YYYY-MM-DD_hh:mm:ss
1993-03-13_00:00:00

```

```

*** Requested END date in format YYYY-MM-DD_hh:mm:ss
1993_03-14_00:00:00
*** Requested Time INTERVAL (in seconds)
43200
#
*** Compressmode for V5D data (1, 2 or 4 bytes per grid point)
4
#
*** Time-mean fields Tm, Um, Vm, Hm and RHm if selected (0:NO, 1:YES) ?
0
*** BEGIN, END and INTERVAL to compute time-mean fields
2001-11-07_00:00:00
2001-11-15_00:00:00
43200
#
*** Select fields
VARIABLE |0:NO 1:YES|FILTERING|
-----|-----|-----|
U      | 1      | 10    |
V      | 1      | 10    |
W      | 1      | 5     |
T      | 1      | 10    |
H      | 1      | 10    |
RH     | 1      | 10    |
Q      | 0      | 0     |
CLW    | 1      | 1     |
RNW    | 1      | 1     |
ICE    | 0      | 0     |
SNOW   | 0      | 0     |
GRAUPEL| 0      | 0     |
NCI    | 0      | 0     |
RAD TEND| 0      | 0     |
PP     | 0      | 0     |

```

PSTARCRS		0		0	
RAIN CON		1		0	
RAIN NON		1		0	
TERRAIN		1		0	
MAPFACCR		1		0	
MAPFACDT		0		0	
CORIOLIS		0		0	
RES TEMP		0		0	
LATITCRS		1		0	
LONGICRS		1		0	
LAND USE		0		0	
SNOWCOVR		0		0	
GROUND T		1		1	
TSEASFC		1		1	
PBL HGT		0		0	
REGIME		0		0	
SHFLUX		1		5	
LHFLUX		1		5	
UST		0		0	
SWDOWN		0		0	
LWDOWN		0		0	
SWOUT		0		0	
LWOUT		0		0	
SOIL T 1		0		0	
SOIL T 2		0		0	
SOIL T 3		0		0	
SOIL T 4		0		0	
SOIL T 5		0		0	
SOIL T 6		0		0	
T2		0		0	
Q2		0		0	
U10		0		0	
V10		0		0	

PSFC		0		0	
PSEALVLC		1		10	
PSEALVLD		0		0	
LATITDOT		0		0	
LONGIDOT		0		0	
RH SFC		0		0	
-----		-----		-----	

To run MM5p_to_v5d:

```
$ ./MM5p_to_v5d pars.MM5p out.v5d
```

If you have no errors, `out.v5d` will be an input file for `vis5d` as you can see in figure 6.

There are similar programs for `REGRID` and `RAWINS`.

0.9.2 GrADS

GrADS is an interactive desktop tool that is used for easy access, manipulation, and visualization of earth science data. GrADS means Grid Analysis and Display System.

You can get GrADS and more information and documentation for this software from <http://grads.iges.org/grads/grads.html>. If you want to give a try to GrADS, you should download the binaries of the last version.

GrADS does not need any installation, so to uncompress the file in a directory present in your path is enough.

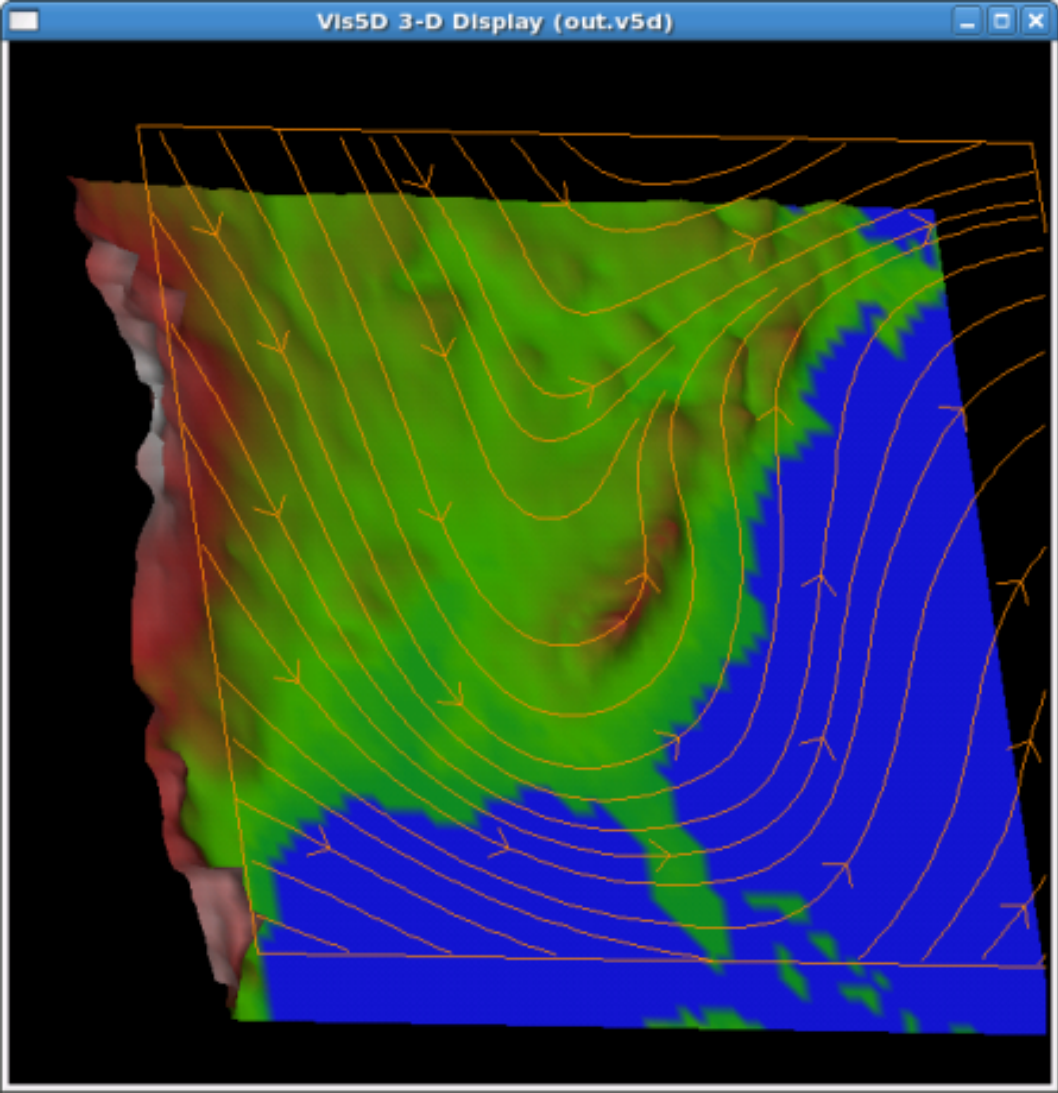


Figure 6: MM5p_to_v5d

MM5toGrADS

MM5toGrADS is a tool able to convert MM5 outputs in data suitable for GrADS. First of all, download MM5toGrADS from <ftp://ftp.ucar.edu/mesouser/MM5V3>. Uncompress it and type `make intel` to compile it.

Once you have compiled it, edit `mm5_to_grads.csh` to type the correct path for your `MMOUT` file. After that, run `mm5_to_grads.csh`. Note that a `namelist.input` exists, where you can modify some options of the outputs.

If the execution has no errors, you will see a pair of new files whose name depends on what you typed in the `mm5_to_grads.csh`. Here comes a little lesson about GrADS usage. For details, you should read its documentation.

```
$ gradsc -l
```

```
Grid Analysis and Display System (GrADS) Version 1.9b4
Copyright (c) 1988-2005 by Brian Doty and IGES
Center for Ocean-Land-Atmosphere Studies (COLA)
Institute for Global Environment and Society (IGES)
GrADS comes with ABSOLUTELY NO WARRANTY
See file COPYRIGHT for more information
```

```
Config: v1.9b4 32-bit little-endian readline lats athena printim
```

```
Issue 'q config' command for more information.
```

```
GX Package Initialization: Size = 11 8.5
ga-> open test.ctl
Scanning description file: test.ctl
Data file test.dat is open as file 1
LON set to -109.46 -60.4066
LAT set to 21.17 50.3588
```

```
LEV set to 0.995 0.995
```

```
Time values set: 1993:3:13:0 1993:3:13:0
```

```
Notice: Implied interpolation for file test.ct1
```

```
Interpolation will be performed on any data displayed from this file  
ga-> q file
```

```
File 1 : MM5 data
```

```
Descriptor: test.ct1
```

```
Binary: test.dat
```

```
Type = Gridded
```

```
Xsize = 122 Ysize = 73 Zsize = 23 Tsize = 5
```

```
Number of Variables = 20
```

```
rc 0 99 accum conv pcn (cm)  
rn 0 99 accum non-c pcn (cm)  
ter 0 99 ter elevation (m)  
xlat 0 99 cross lat (degree)  
xlon 0 99 cross lon (degree)  
lu 0 99 land use  
t2m 0 99 2 m temperature (K)  
q2m 0 99 2m mix ratio (kg/kg)  
u10 0 99 10 m u wind (m/sec)  
v10 0 99 10 m v wind (m/sec)  
u 23 99 u wind (m/s)  
v 23 99 v wind (m/s)  
w 23 99 vertical vel (m/s)  
t 23 99 temperature (C)  
q 23 99 mixing ratio (kg/kg)  
clw 23 99 cloud water (kg/kg)  
rnw 23 99 rain water (kg/kg)  
h 23 99 geopot height (m)  
td 23 99 dewpoint temp (C)  
rh 23 99 rel humidity (%)
```

```
ga-> display u
```

```
ga-> clear
ga-> q time
Time = 00Z13MAR1993 to 00Z13MAR1993 Sat to Sat
ga-> set time 12Z13MAR1993
Time values set: 1993:3:13:18 1993:3:13:18
ga-> set lev 0.8
ga-> set gxout shaded
ga-> display h
ga-> clear
ga-> set gxout stream
ga-> display u;v
ga-> clear
ga-> set gxout contour
ga-> d h
Notice: Automatic Grid Interpolation Taking Place
Contouring: 1400 to 3600 interval 200
```

In figure 7 and 8 you can see two plots from GrADS.

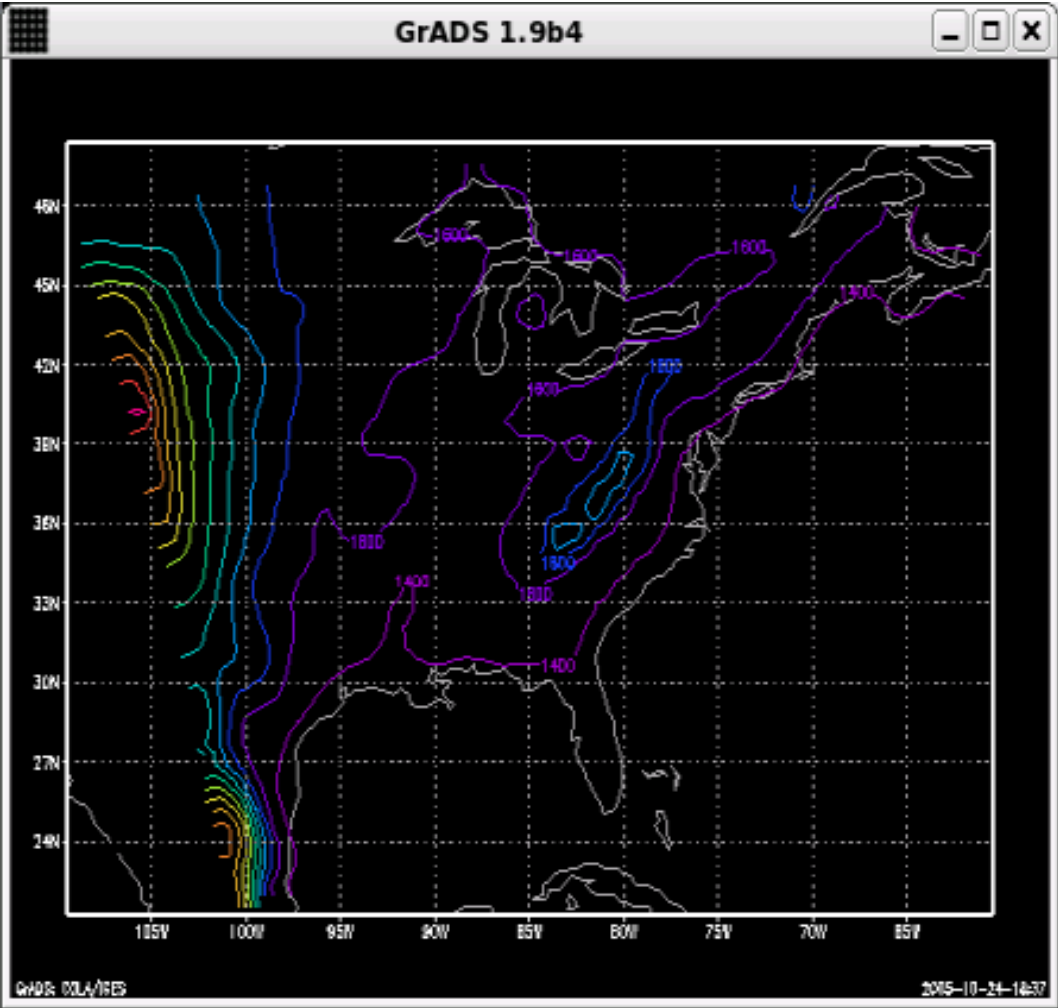


Figure 7: GrADS

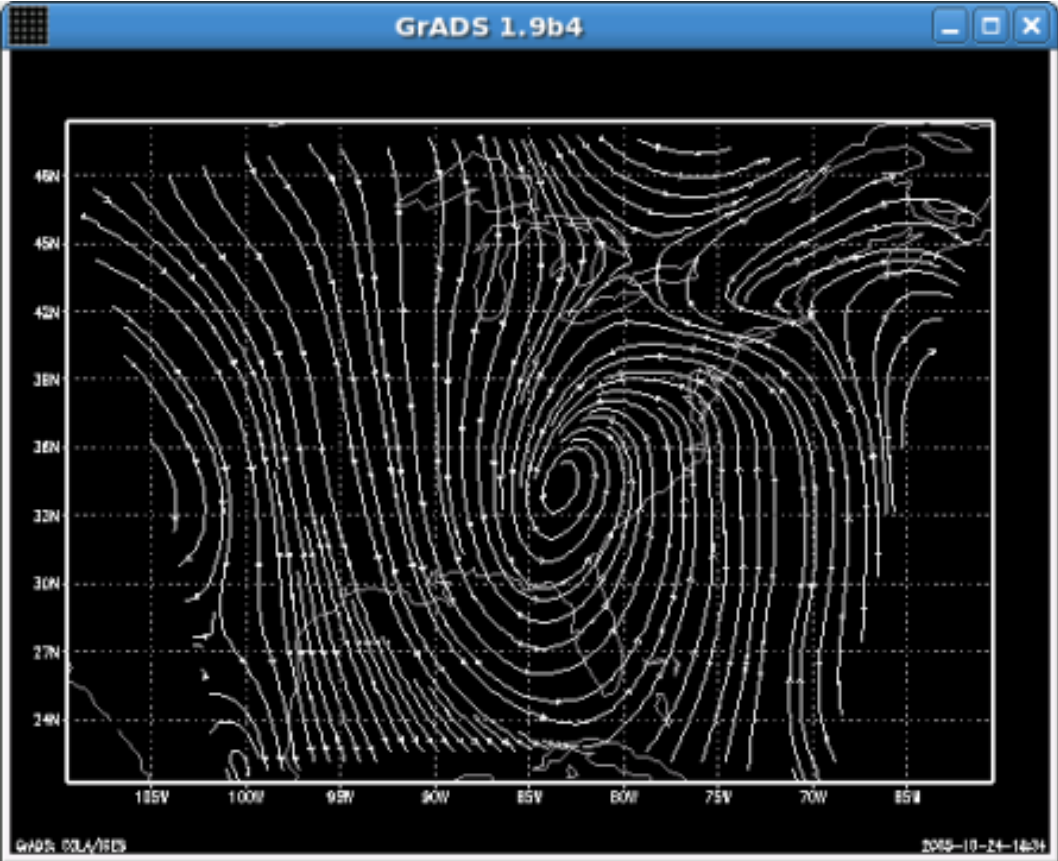


Figure 8: GrADS

0.10 USING MORE PROCESSORS: MPP

If you have a modern computer, the *SOC* example case will not take you more than a few minutes. That is an example case, so its domains are not very detailed and the simulation length is very short. Real simulations can take many hours even days. So it is very important you get all the horsepower you can. If you are running a simulation on a SMP machine and type `top` you will notice that there is a process called `mm5.exe` burning one of your processors. By default, MM5 will only run on a single processor. If you want to use all the nodes of your Linux PC Cluster, or all the processors of your SMP machine you will need to build again the model with the MPP extension.

0.10.1 Setting `mpich`

MPICH is a freely available, portable implementation of MPI, the Standard for message-passing libraries.

MPI stands for Message Passing Interface. The goal of MPI, simply stated, is to develop a widely used standard for writing message-passing programs. As such the interface should establish a practical, portable, efficient, and flexible standard for message passing.

You must have `mpich` installed on your system in order to compile MM5 Model with MPP extensions. Although you can obtain packages for your distribution, it is better you compile it manually because we are not going to use the default Fortran Compiler (GNU Fortran).

So, download `mpich`. You can do it from <http://www-unix.mcs.anl.gov/mpi/mpich/>. Once you have downloaded and unpackaged it is time to read documentation to know what kind of options you have to specify. A brief guide to do this could be:

- Type

```
$ ./configure --help
```

and read *carefully* the available options.

- Once you have read the output and you know what options you should set it is time to run configure setting those options. In my case, I am running a dual processor (SMP) Linux machine. These processors can communicate between them using shared memory. so these are my options for configure:

```
$ ./configure --with-arch=LINUX \  
              --with-device=ch_shmem \  
              -cc=gcc -fc=ifort \  
              --disable-short-longs
```

Note that if you have a cluster, you will need to use another device option, such as p4. If you are running a cluster of SMP machines, consult the `mpich` documentation to set right options.

- Check the `Makefile` to see if the `NOF77` option is set to 0. If not, your compiler have not passed configure tests. Be sure your `PATH` environment variable is correct.
- Compile `mpich` by typing `make`.
- Test it! Type `make testing` and look at if there is no problems. The cause of your problems can easily be you have not `rsh` installed or working. I am not going to describe the process of installing `rsh` on your system because it varies from one distribution to other. In Mandriva Linux, the `rsh-server` works as a `xinetd` service, in other distributions could be an `inetd` service or a standalone server. A quick and fast (and probably wrong) guide suitable for Mandriva. Install (with `urpmi` of course) `rsh-server` and `rsh`, use `chkconfig` to activate it as an `xinetd` service and restart `xinetd` by typing `service xinetd restart`. Now in your home directory, create a file called `.rhosts` with lines of this type:

```
hostname user
```

where `hostname` is the name of the host which can execute remote commands⁴ and `user` is the user allowed to do it. An easy way if you have an SMP machine to test if `rsh` works for your current user is to type something like:

```
$ rsh 'hostname' echo "Hello World"  
Hello World
```

⁴the output of `hostname` command will help you

If you get errors, look for information about configuring `rsh` or ask to your system administrator.

`rsh` is not considered a safe application, so machines with `rsh` enabled should be monitored to avoid security problems. This is just an advice.

After you have `rsh` running, type `make testing`. It should work.

- Time to install `mpich`. I will use a non-standard location, to do this type:

```
$ make install PREFIX=/opt/mpich-1.2.7
```

Once it has finished, I will create a symlink:

```
$ ln -s /opt/mpich-1.2.7 /opt/mpich
```

These steps allow to change among different versions and compilations of `mpich` by changing only a symlink.

Add to your environment variables the `mpich` interesting paths (`mpich/bin` to `$PATH` and `mpich/lib` to `$LD_LIBRARY_PATH`).

- Run some examples. Go to your `mpich/examples` and type `make all` to compile some sample programs. In my case, I will type:

```
$ mpirun -arch LINUX -np 2 cpi
Process 1 on suzuki
Process 0 on suzuki
pi is approximately 3.1416009869231241, Error is 0.0000083333333309
wall clock time = 0.000165
```

0.10.2 Compiling MM5-MPP

Download MPP from <ftp://ftp.ucar.edu/mesouser/MM5V3> and untar it in you MM5 directory which you could rename to MM5-MPP.

Edit the `configure.user` file. You should look at section 7, where MPP options resides. The other options (except Physics section) should be commented. Go to 7g2 section, called *Linux PCs. Need INTEL and MPICH.* and edit those options. These are my settings:

```
#
# -----
# 7. MPP options
#
# For general information and updated "helpdesk" information see
#   http://www.mmm.ucar.edu/mm5/mpp
#   http://www.mmm.ucar.edu/mm5/mpp/helpdesk
#
# -----
#
# Presently, of the MPP platforms only the "sp2"
# is supplied with the "make deck" capability.
#
# MPP Software Layer
MPP_LAYER=RSL
#MPP_LAYER=NNTSMS
#
# PROCMIN_NS – minimum number of processors allowed in N/S dim
#
PROCMIN_NS = 1
#
# PROCMIN_EW – minimum number of processors allowed in E/W dim
#
PROCMIN_EW = 1
#
# ASSUME_HOMOGENOUS_ENVIRONMENT – on a machine with a heterogeneous
# mix of processors (different speeds) setting this compile time
# constant to 0 (zero) allows the program to detect the speed of each
# processor at the beginning of a run and then to attempt to come up
# with an optimal (static) mapping. Set this to 0 for a heterogeneous
```

```
# mix of processors, set it to 1 for a homogeneous mix. Unless you
# are certain you have a heterogeneous mix of processors, leave this
# set to 1. Currently, this option is ignored on platforms other
# than the IBM SP.
```

```
#
```

```
ASSUME_HOMOGENEOUS_ENVIRONMENT = 1
```

```
#
```

```
[...]
```

```
#
```

```
# 7g2. Linux PCs. Need INTEL and MPICH.
```

```
#
```

```
RUNTIME_SYSTEM = "linux"
```

```
MPP_TARGET=$(RUNTIME_SYSTEM)
```

```
### edit the following definition for your system
```

```
LINUX_MPIHOME = /opt/mpich
```

```
MFC = $(LINUX_MPIHOME)/bin/mpif77
```

```
MCC = $(LINUX_MPIHOME)/bin/mpicc
```

```
MLD = $(LINUX_MPIHOME)/bin/mpif77
```

```
FCFLAGS = -O2 -convert big_endian -pc32
```

```
LDOPTIONS = -O2 -convert big_endian -pc32
```

```
LOCAL_LIBRARIES = -L$(LINUX_MPIHOME)/lib -lmpich -lmpich
```

```
MAKE = make -i -r
```

```
AWK = awk
```

```
SED = sed
```

```
CAT = cat
```

```
CUT = cut
```

```
EXPAND = /usr/bin/expand
```

```
M4 = m4
```

```
CPP = /lib/cpp -C -P -traditional
```

```
CPPFLAGS = -traditional -DMPI -Dlinux
```

```
CFLAGS = -DMPI -I/opt/mpich -I/opt/mpich/include
```

```
ARCH_OBJS = milliclock.o
IWORDSIZE = 4
RWORDSIZE = 4
LWORDSIZE = 4
```

To compile it type `make mpp`. I had some errors in the compilation. To solve them, I edited `MPP/RSL/RSL/rsl.h` file and commented the line:

```
//typedef int MPI_Fint;
```

After that, the model compiled with no problems.

In your `Run` directory you will have a binary file called `mpm.exe`. To execute it, you should type something like:

```
$ mpirun -v -arch LINUX -np 2 mm5.mpp
```

where `arch` indicates the type of your machine and `np` states the number of processors you want to use.

0.11 ABOUT THIS DOCUMENT

This document covers the basic installation of the MM5 modelling suite under common PCs running Linux. Intel Fortran Compiler has been chosen because it is free for non-commercial uses.

As it is said in this text, you should be reading the MM5 Online Tutorial at the same time you read this. The Online Tutorial gives you an overview of each program, telling you what they do and how they must be used. This guide will only help you to compile and use those programs, but if you want to know what is happening you will want to have a look to MM5 documentation.

This document has been written by Javier Robles from the University of León. You can contact him for comments and suggestions at dfqjrp_AT_unileon_DOT_es. You can visit my personal web page (in Spanish) for more information about me: <http://www.milugar.net> Feedback is welcome!

This work is licensed under the Creative Commons Attribution-NonCommercial-ShareAlike 2.5 License. To view a copy of this license, visit <http://creativecommons.org/licenses/by-nc-sa/2.5/> or send a letter to Creative Commons, 543 Howard Street, 5th Floor, San Francisco, California, 94105, USA.