

Reporte WRF

Joao Fabián

Instalación

La instalación del modelo WRF, del sistema de procesamiento WPS, y de los softwares de post-procesamiento, son realizados en distribuciones de Linux. Todas las secuencias y comandos mostrados en este reporte deben realizarse en el terminal de Linux, a no ser que se especifique lo contrario.

Se deben crear dos directorios en la *carpeta de usuario* o *home directory*. Un directorio será llamado *Build_WRF*, en este se montarán los archivos necesarios del modelo *WRF*. El otro directorio será llamado *TESTS*, y en este se montarán archivos que permitirán realizar pruebas para confirmar que el software adicional requerido funciona correctamente.

Software adicional requerido

Es necesario instalar: *gfortran*, *csh*, *m4* y *build-essential*.

```
1 | $ sudo apt-get install gfortran csh m4 build-essential
```

Prueba del entorno del sistema

Se deben tener instalados compiladores de *Fortran* (gfortran), *C++* (gpp) y *C* (gcc), actualizados en sus versiones más recientes.

Es necesario descargar los archivos que servirán para hacer las pruebas.

```
1 | $ cd ~/TESTS
```

```
2 $ wget http://www2.mmm.ucar.edu/wrf/OnLineTutorial/  
   compile_tutorial/tar_files/Fortran_C_tests.tar  
3 $ tar -xvf Fortran_C_tests.tar
```

Son siete pruebas que se encuentran disponibles. Estas se correrán una por una.

1. Prueba para *Fortran* de formato fijo:

```
1 $ gfortran TEST_1_fortran_only_fixed.f  
2 $ ./a.out
```

En caso de que la prueba sea exitosa, en el terminal se mostrará:

```
1 SUCCESS test 1 fortran only fixed format
```

2. Prueba para *Fortran* de formato libre:

```
1 $ gfortran TEST_2_fortran_only_free.f90  
2 $ ./a.out
```

En caso de que la prueba sea exitosa, en el terminal se mostrará:

```
1 SUCCESS test 2 fortran only free format
```

3. Prueba de *C*:

```
1 $ gcc TEST_3_c_only.c  
2 $ ./a.out
```

En caso de que la prueba sea exitosa, en el terminal se mostrará:

```
1 SUCCESS test 3 C only
```

4. Prueba de *Fortran* llamando a una función en *C*:

```

1 $ gcc -c -m64 TEST_4_fortran+c_c.c
2 $ gfortran -c -m64 TEST_4_fortran+c_f.f90
3 $ gfortran -m64 TEST_4_fortran+c_f.o TEST_4_fortran+
   c_c.o
4 $ ./a.out

```

En caso de que la prueba sea exitosa, en el terminal se mostrará:

```

1 C function called by Fortran
2 Values are xx = 2.00 and ii= 1
3 SUCCESS test 4 fortran calling c

```

5. Prueba de *cs**h*:

```

1 $ cs h TEST_csh.csh

```

En caso de que la prueba sea exitosa, en el terminal se mostrará:

```

1 SUCCESS cs h test

```

6. Prueba de *Perl*:

```

1 $ ./TEST_perl.pl

```

En caso de que la prueba sea exitosa, en el terminal se mostrará:

```

1 SUCCESS perl test

```

7. Prueba de *sh*:

```

1 $ ./TEST_sh.sh

```

En caso de que la prueba sea exitosa, en el terminal se mostrará:

```

1 SUCCESS sh test

```

Construcción de librerías

Se crea el directorio *LIBRARIES*:

```
1 $ mkdir ~/Build_WRF/LIBRARIES
```

Se descargan los archivos comprimidos de las librerías: *MPICH*, *NetCDF*, *JasPer*, *libpng* y *zlib*:

```
1 $ cd ~/Build_WRF/LIBRARIES
2 $ wget http://www2.mmm.ucar.edu/wrf/OnLineTutorial/
  compile_tutorial/tar_files/mpich-3.0.4.tar.gz
3 $ wget http://www2.mmm.ucar.edu/wrf/OnLineTutorial/
  compile_tutorial/tar_files/netcdf-4.1.3.tar.gz
4 $ http://www2.mmm.ucar.edu/wrf/OnLineTutorial/
  compile_tutorial/tar_files/jasper-1.900.1.tar.gz
5 $ http://www2.mmm.ucar.edu/wrf/OnLineTutorial/
  compile_tutorial/tar_files/libpng-1.2.50.tar.gz
6 $ http://www2.mmm.ucar.edu/wrf/OnLineTutorial/
  compile_tutorial/tar_files/zlib-1.2.7.tar.gz
```

Configuración de NetCDF

Se abre el archivo *.bashrc* que se encuentra en el *home directory*:

```
1 $ sudo nano ~/.bashrc
```

Así, al final del archivo *.bashrc* se deben agregar las líneas:

```
1 # WRF environment variables
2 export DIR=/home/lmc/Build_WRF/LIBRARIES
3 export CC=gcc
4 export CXX=g++
5 export FC=gfortran
6 export CFLAGS=-m64
7 export F77=gfortran
8 export FFLAGS=-m64
```

Se corre *.bashrc* y se descomprime el archivo correspondiente a *NetCDF*:

```

1 $ source ~/.bashrc
2 $ cd ~/Build_WRF/LIBRARIES
3 $ tar -zxvf netcdf-4.1.3.tar.gz

```

Se corre el archivo de configuración de *NetCDF*:

```

1 $ cd ~/Build_WRF/LIBRARIES/netcdf-4.1.3
2 $ ./configure --prefix=$DIR/netcdf --disable-dap --
  disable-netcdf-4 --disable-shared
3 $ make
4 $ make install

```

Se debe modificar otra vez el archivo *.bashrc*:

```

1 $ sudo nano ~/.bashrc

```

Al final del archivo *bashrc* se deben agregar las siguientes líneas:

```

1 export PATH=$DIR/netcdf/bin:$PATH
2 export NETCDF=$DIR/netcdf

```

Se corre *.bashrc*:

```

1 $ source ~/.bashrc

```

Configuración de MPICH

Se descomprime el archivo correspondiente a *MPICH*:

```

1 $ cd ~/Build_WRF/LIBRARIES
2 $ tar -zxvf mpich-3.0.4.tar.gz

```

Se corre el archivo de configuración de MPICH:

```

1 $ cd ~/Build_WRF/LIBRARIES/mpich-3.0.4
2 $ ./configure --prefix=$DIR/mpich
3 $ make
4 $ make install

```

Se abre el archivo *.bashrc* que se encuentra en el *home directory*:

```
1 $ sudo nano ~/.bashrc
```

Al final del archivo *.bashrc*, se agrega la línea:

```
1 export PATH=$DIR/mpich/bin:$PATH
```

Se corre *.bashrc*:

```
1 $ source ~/.bashrc
```

Configuración de zlib

Se descomprime el archivo correspondiente a *zlib*:

```
1 $ cd ~/Build_WRF/LIBRARIES
2 $ tar -zxvf zlib-1.2.7.tar.gz
```

Se abre el archivo *.bashrc* que se encuentra en el *home directory*:

```
1 $ sudo nano ~/.bashrc
```

Al final del archivo *.bashrc* se agregan las líneas:

```
1 export LDFLAGS=-L$DIR/grib2/lib
2 export CPPFLAGS=-I$DIR/grib2/include
```

Se corre *.bashrc*:

```
1 $ source ~/.bashrc
```

Se corre el archivo de configuración de *zlib*:

```
1 $ cd ~/Build_WRF/LIBRARIES/zlib-1.2.7
2 $ ./configure --prefix=$DIR/grib2
3 $ make
4 $ make install
```

Configuración de libpng

Se descomprime el archivo correspondiente a *libpng*:

```
1 $ cd ~/Build_WRF/libraries
2 $ tar -zxvf libpng-1.2.50.tar.gz
```

Se corre el archivo de configuración de *libpng*:

```
1 $ cd ~/Build_WRF/LIBRARIES/libpng-1.2.50
2 $ ./configure --prefix=$DIR/grib2
3 $ make
4 $ make install
```

Configuración de JasPer

Se descomprime el archivo correspondiente a *Jasper*

```
1 $ cd ~/Build_WRF/LIBRARIES
2 $ tar -zxvf jasper-1.900.1.tar.gz
```

Se corre el archivo de configuración de *JasPer*:

```
1 $ cd ~/Build_WRF/LIBRARIES/jasper-1.900.1
2 $ ./configure --prefix=$DIR/grib2
3 $ make
4 $ make install
```

Prueba de compatibilidad de librerías con WRF y WPS

Se descargan los archivos de pruebas:

```
1 $ cd ~/TESTS
2 $ wget http://www2.mmm.ucar.edu/wrf/OnLineTutorial/
   compile_tutorial/tar_files/Fortran_C_NETCDF_MPI_tests
   .tar
3 $ tar -xvf Fortran_C_NETCDF_MPI_tests.tar
```

Son dos pruebas, y se realizarán una por una:

1. Prueba de Fortran + C + NetCDF:

```
1 $ cd ~/TESTS
2 $ cp ${NETCDF}/include/netcdf.inc .
3 $ gfortran -c 01_fortran+c+netcdf_f.f
4 $ gcc -c 01_fortran+c+netcdf_c.c
5 $ gfortran 01_fortran+c+netcdf_f.o 01_fortran+c+
   netcdf_c.o -L${NETCDF}/lib -lnetcdff -lnetcdf
6 $ ./a.out
```

En caso de que la prueba sea exitosa, el terminal mostrará:

```
1 C function called by Fortran
2 Values are xx = 2.00 and ii = 1
3 SUCCESS test 1 fortran + c + netcdf
```

2. Prueba de Fortran + C + NetCDF + MPI:

```
1 $ cd ~/TESTS
2 $ cp ${NETCDF}/include/netcdf.inc .
3 $ mpif90 -c 02_fortran+c+netcdf+mpi_f.f
4 $ mpicc -c 02_fortran+c+netcdf+mpi_c.c
5 $ mpif90 02_fortran+c+netcdf+mpi_f.o 02_fortran+c+
   netcdf+mpi_c.o -L${NETCDF}/lib -lnetcdff -lnetcdf
6 $ mpirun ./a.out
```

En caso de que la prueba sea exitosa, el terminal mostrará:

```
1 C function called by Fortran
2 Values are xx = 2.00 and ii = 1
3 status = 2
4 SUCCESS test 2 fortran + c + netcdf + mpi
```

Construcción de WRF V4

Se descarga WRF:


```
1 $ cd ~/Build_WRF
2 $ wget http://www2.mmm.ucar.edu/wrf/src/WRFV4.0.TAR.gz
3 $ tar -zxvf WRFV4.0.TAR.gz
```

Se corre el archivo de configuración de WRF:

```
1 $ cd ~/Build_WRF/WRF
2 $ ./configure
3 $ 34
```

Se escoge la opción **34** debido a que la computadora presenta varios núcleos, y permite trabajar en paralelo, además, se trabajará con *gfortran/gcc*.

Una vez que la configuración se complete, debe haberse generado un archivo llamado *configure.wrf* en el mismo directorio.

Puesto que la configuración está completa, se pasa a compilar, seleccionando el caso de acuerdo al propósito para el que será usado el modelo *WRF*. En este caso, se compila *WRF* para casos reales:

```
1 $ cd ~/Build_WRF/WRF
2 $ ./compile em_real >& compile.log &
3 $ tail -f compile.log
```

Para corroborar que la compilación fue exitosa, se revisa el directorio *WRF*:

```
1 $ cd ~/Build_WRF/WRF
2 $ ls -las main/*.exe
```

La compilación se habrá completado correctamente si en el terminal se observan los ejecutables:

```
1 main/ndown.exe
2 main/real.exe
3 main/tc.exe
4 main/wrf.exe
```

Construcción de WPS V4

El programa *WPS* debe construirse debido a que se busca trabajar con casos reales.

Se descarga WPS:

```
1 $ cd ~/Build_WRF
2 $ wget http://www2.mmm.ucar.edu/wrf/src/WPSV4.0.TAR.gz
3 $ tar -zxvf WPSV4.0.TAR.gz
4 $ cd WPS
5 $ ./clean
```

Se abre el archivo *.bashrc*:

```
1 $ sudo nano ~/.bashrc
```

Al final del archivo se agregan las líneas:

```
1 export JASPERLIB=$DIR/grib2/lib
2 export JASPERINC=$DIR/grib2/include
```

Se corre *.bashrc*:

```
1 $ source ~/.bashrc
```

Se corre el archivo de configuración de *WPS*:

```
1 $ cd ~/Build_WRF/WPS
2 $ ./configure
3 $ 1
```

Se selecciona la opción **1** debido a las características de la computadora, además de estar usando *gfortran*. Si la configuración se completa correctamente, debe aparecer un mensaje en el terminal que terminará con:

```
1 This installation NetCDF is 64-bit
2 C compiler is 64-bit
3 Fortran compiler is 64-bit
```

El sistema *WPS* debe direccionarse hacia las librerías *I/O* de *WRF*, donde se encuentran los ejecutables *metgrid.exe* y *geogrid.exe*:

```
1 $ cd ~/Build_WRS/WPS
2 $ sudo nano configure.wps
```

En el archivo, especificar la ruta hacia el directorio *WRF*:

```
1 WRF_DIR = ../WRF
```

Se compila *WPS*:

```
1 $ cd ~/Build_WRS/WPS
2 $ ./compile >& compile.log&
3 $ tail -f compile.log
```

Para corroborar que la compilación fue exitosa, revisar el directorio *WPS*:

```
1 $ cd ~/Build_WRS/WPS
2 $ ls -las *.exe
```

La compilación será correcta si en el terminal se muestran los ejecutables:

```
1 geogrid.exe
2 metgrid.exe
3 ungrib.exe
```

Para inicializar casos reales, se deben crear la locación física del dominio en el gobo, y la información estática para tal locación. Esta información debe ser descargada:

```
1 $ cd ~/Build_WRF
2 $ wget http://www2.mmm.ucar.edu/wrf/src/wps_files/
   geog_10m.tar.gz
3 $ tar -zxvf geog_10m.tar.gz
4 $ mv geog WPS_GEOG
```

Se abre el archivo *namelist.WPS*:

```
1 $ cd ~/Build_WRF/WPS
```

```
2 $ nano namelist.wps
```

Se modifica el directorio correcto en el archivo *namelist.wps*:

```
1 geog_data_path= '/home/lmc/Build_WRF/WPS_GEOG/'
```

Post - Procesamiento

Para poder leer los archivos generados por *WRF* usando el core *ARW*, se usa el programa *ARWPost*, además, se usa *GrADS* para poder hacer un plot de los resultados obtenidos.

Descargar ARWPost:

```
1 $ cd ~/Build_WRF
2 $ wget http://www2.mmm.ucar.edu/wrf/src/ARWpost_V3.tar.gz
3 $ tar -zxvf ARWpost_V3.tar.gz
```

Asumiendo que ya se configuró NetCDF, se configura ARWpost:

```
1 $ cd ~/Build_WRF/ARWpost
2 $ ./configure
3 $ 3
4 $ cd src
```

Se abre el archivo *Makefile*:

```
1 $ cd ~/Build_WRF/ARWpost
2 $ sudo nano Makefile
```

En el archivo *Makefile* se modifica la línea correspondiente a *ARWpost.exe*:

```
1 ARWpost.exe: $(OBS)
2             $(FC) $(FFLAGS) $(LDFLAGS) -o $$ $(OBS)
3             \
                -L$(NETCDF)/lib -lnetcdf
                -lnetcdff -I$(NETCDF
                )/include -lnetcdf
```

Además, se abre el archivo *configure.arwp*:

```
1 $ cd ~/Build_WRF/ARWpost
2 $ sudo nano configure.arwp
```

En el archivo *configure.awps* se modifican las líneas correspondientes a *CFLAGS* y *CPP*:

```
1 CFLAGS          =          -fPIC -m64
2 CPP              =          /lib/cpp -P -traditional
```

Se compila *ARWpost*:

```
1 $ cd ~/Build_WRF/ARWpost
2 $ ./compile
```

Se revisan los archivos generados, para comprobar si la compilación se realizó de manera correcta:

```
1 $ cd ~/Build_WRF/ARWpost
2 $ ls -ls *.exe
```

La compilación será correcta si el terminal muestra el ejecutable:

```
1 ARWpost.exe
```

Para poder hacer un plot de los archivos que serán generados a través de ARWpost, es necesario instalar el programa GrADS:

```
1 $ sudo apt-get install grads
```

Pruebas

Test 1

Geogrid.exe

En el procesamiento con WPS, se modifica el archivo *namelist.wps*:

```
1 $ cd ~/Build_WRF/WPS
2 $ sudo nano namelist.wps
```

Así, los parámetros en *namelist.wps* son modificados de acuerdo a:

```
1 wrf_core = 'ARW',
2 &share
3   max_dom = 1,
4   start_date = '2011-09-05_12:00:00', '2006-08-16_12
      :00:00',
5   end_date   = '2011-09-10_12:00:00', '2006-08-16_12
      :00:00',
6   interval_seconds = 10800
7 /
8
9 &geogrid
10  parent_id      = 1, 1,
11  parent_grid_ratio = 1, 3,
12  i_parent_start  = 1, 31,
13  j_parent_start  = 1, 17,
14  e_we           = 15, 112,
15  e_sn           = 15, 97,
16  geog_data_res   = '2m', '2m',
17  dx = 3333.33,
18  dy = 3333.33,
19  map_proj = 'lambert',
20  ref_lat  = 45.68,
21  ref_lon  = -111.06,
22  truelat1 = 30.0,
23  truelat2 = 60.0,
24  stand_lon = -111.06,
25  geog_data_path = '/home/marv/Build_WRF/WPS_GEOG/'
26 /
27
28 &ungrib
29  out_format = 'WPS',
30  prefix = 'FILE',
31 /
32
```

```

33 &metgrid
34   fg_name = 'FILE'
35   io_form_metgrid = 2,
36 /

```

Se genera el ejecutable *geogrid.exe*:

```

1 $ cd ~/Build_WRF/WPS
2 $ ./geogrid.exe

```

El terminal indicará el mensaje de confirmación siguiente:

```

1 !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
2 !   Successful completion of geogrid.           !
3 !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

```

ungrib.exe

El dataset usado es el **ds609.2**. Se realiza la descarga de este:

```

1 $ mkdir ~/Build_WRF/data
2 $ mkdir ~/Build_WRF/data/AWIP_datasets
3 $ mkdir ~/Build_WRF/data/AWIP_datasets/ds609.2
4 $ cd ~/Build_WRF/data/AWIP_datasets/ds609.2
5 $ wget https://rda.ucar.edu/data/ds609.2/SFanal/2011/
   G43091.SFanal.201109
6 $ wget https://rda.ucar.edu/data/ds609.2/3Danal/2011/
   G43084.3Danal.201109.16-30
7 $ wget https://rda.ucar.edu/data/ds609.2/3Danal/2011/
   G43083.3Danal.201109.01-15
8 $ mv G43091.SFanal.201109 G43091.SFanal.201109.tar
9 $ mv G43084.3Danal.201109.16-30 G43084.3Danal
   .201109.16-30.tar
10 $ mv G43083.3Danal.201109.01-15 G43083.3Danal
   .201109.01-15.tar
11 $ tar -zxvf G43091.SFanal.201109.tar
12 $ tar -zxvf G43084.3Danal.201109.16-30.tar
13 $ tar -zxvf G43083.3Danal.201109.01-15.tar

```

Se genera el ejecutable *ungrib.exe*:

```
1 $ cd ~/Build_WRF/WPS
2 $ ln -sf ungrib/Variable_Tables/Vtable.AWIP Vtable
3 $ ./link_grib.csh ~/Build_WRF/AWIP_datasets/ds609
  .2/201109
4 $ ./ungrib.exe >& ungrib.log
```

metgrid.exe

Se corre el ejecutable *metgrid.exe*:

```
1 $ cd ~/Build_WRF/WPS
2 $ ./metgrid.exe
```

El terminal indicará el mensaje de confirmación siguiente:

```
1 !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
2 !   Successful completion of metgrid.           !
3 !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
```

real.exe y wrf.exe

Se entra al directorio *test*:

```
1 $ cd ~/Build_WRF/WRF/test/em_real
2 $ ln -sf ~/Build_WRF/WPS/met_em.d01.* .
3 $ sudo nano namelist.input
```

Los parámetros dentro del archivo *namelist.input* se modifica de la forma:

```
1 &time_control
2 run_days           = 3,
3 run_hours          = 0,
4 run_minutes        = 0,
5 run_seconds        = 0,
6 start_year         = 2011, 2000, 2000,
7 start_month        = 09, 01, 01,
```



```

8      start_day          = 06,    24,    24,
9      start_hour         = 12,    12,    12,
10     start_minute       = 00,    00,    00,
11     start_second       = 00,    00,    00,
12     end_year           = 2011, 2000, 2000,
13     end_month          = 09,    01,    01,
14     end_day            = 09,    25,    25,
15     end_hour           = 12,    12,    12,
16     end_minute         = 00,    00,    00,
17     end_second         = 00,    00,    00,
18     interval_seconds   = 10800,
19     input_from_file    = .true.,.true.,.
      true.,
20     history_interval   = 180,    60,    60,
21     frames_per_outfile = 1000, 1000, 1000,
22     restart            = .false.,
23     restart_interval   = 21600,
24     io_form_history     = 2,
25     io_form_restart     = 2,
26     io_form_input       = 2,
27     io_form_boundary    = 2,
28     debug_level        = 0,
29     /
30
31     &domains
32     time_step           = 30,
33     time_step_fract_num = 0,
34     time_step_fract_den = 1,
35     max_dom             = 1,
36     s_we                = 1,
37     e_we                = 15,    112,    94,
38     s_sn                = 1,
39     e_sn                = 15,    97,    91,
40     s_vert              = 1,
41     e_vert              = 60,    28,    28,
42     p_top_requested     = 5000,
43     num_metgrid_levels  = 27,
44     num_metgrid_soil_levels = 4,
45     dx                  = 3333.33, 10000,
      3333.33,

```

```

46      dy = 3333.33, 10000,
      3333.33,
47      grid_id = 1, 2, 3,
48      parent_id = 0, 1, 2,
49      i_parent_start = 1, 31, 30,
50      j_parent_start = 1, 17, 30,
51      parent_grid_ratio = 1, 3, 3,
52      parent_time_step_ratio = 1, 3, 3,
53      feedback = 1,
54      smooth_option = 0,
55      sfcp_to_sfcp = .true.
56      /
57
58      &physics
59      physics_suite = 'CONUS'
60      mp_physics = -1, -1, -1,
61      cu_physics = -1, -1, 0,
62      ra_lw_physics = -1, -1, -1,
63      ra_sw_physics = -1, -1, -1,
64      bl_pbl_physics = -1, -1, -1,
65      sf_sfclay_physics = -1, -1, -1,
66      sf_surface_physics = -1, -1, -1,
67      radt = 30, 30, 30,
68      bldt = 0, 0, 0,
69      cudt = 5, 5, 5,
70      icloud = 1,
71      num_land_cat = 21,
72      sf_urban_physics = 0, 0, 0,
73      /
74
75      &fdde
76      /
77
78      &dynamics
79      hybrid_opt = 2,
80      w_damping = 0,
81      diff_opt = 1, 1, 1,
82      km_opt = 4, 4, 4,
83      diff_6th_opt = 0, 0, 0,
84      diff_6th_factor = 0.12, 0.12, 0.12,

```

```

85     base_temp                = 290.
86     damp_opt                 = 3,
87     zdamp                    = 5000., 5000., 5000.,
88     dampcoef                 = 0.2, 0.2, 0.2
89     khdif                    = 0, 0, 0,
90     kvdif                    = 0, 0, 0,
91     non_hydrostatic          = .true., .true., .true.,
92     moist_adv_opt            = 1, 1, 1,
93     scalar_adv_opt           = 1, 1, 1,
94     gwd_opt                  = 1,
95     /
96
97     &bdy_control
98     spec_bdy_width           = 5,
99     specified                 = .true.
100    /
101
102    &grib2
103    /
104
105    &namelist_quilt
106    nio_tasks_per_group = 0,
107    nio_groups = 1,
108    /

```

Para obtener los ejecutables *real.exe* y *wrf.exe*:

```

1 $ ./real.exe
2 $ ./wrf.exe >& wrfoutput.log

```

Se generan archivos de salida correspondientes al procesamiento realizado haciendo uso de *WRF*.

ARWpost

Se ingresa al directorio *ARWpost* y se modifica el archivo *namelist.ARWpost*:

```

1 $ cd ~/Build_WRF/WRF/ARWpost
2 $ sudo nano namelist.ARWpost

```

El archivo debe modificarse de la siguiente forma:

```
1  &datetime
2    start_date = '2000-01-24_12:00:00',
3    end_date   = '2000-01-25_00:00:00',
4    interval_seconds = 10800,
5    tacc = 0,
6    debug_level = 0,
7  /
8
9  &io
10   input_root_name = '~/Build_WRF/WRF/test/em_real/
      wrfout_d01'
11   output_root_name = './test'
12   plot = 'all_list'
13   fields = 'height,pressure,tk,tc'
14   mercator_defs = .true.
15 /
16   split_output = .true.
17   frames_per_outfile = 2
18
19
20   plot = 'all'
21   plot = 'list'
22   plot = 'all_list'
23   ! Below is a list of all available diagnostics
24   fields = 'height,geopt,theta,tc,tk,td,td2,rh,rh2,umet,
      vmet,pressure,u10m,v10m,wdir,wspd,wd10,ws10,slp,
      mcape,mcin,lcl,lfc,cape,cin,dbz,max_dbz,clfr'
25
26
27 &interp
28   interp_method = 0,
29   interp_levels =
      1000.,950.,900.,850.,800.,750.,700.,650.,600.,550.,500.,450.,400.,350
30 /
31   extrapolate = .true.
32
33   interp_method = 0,      ! 0 is model levels, -1 is nice
```

```

height levels, 1 is user specified pressure/height
34
35 interp_levels =
    1000.,950.,900.,850.,800.,750.,700.,650.,600.,550.,
36 500.,450.,400.,350.,300.,250.,200.,150.,100.,
37 interp_levels = 0.25, 0.50, 0.75, 1.00, 2.00, 3.00,
    4.00, 5.00, 6.00, 7.00, 8.00, 9.00, 10.0, 11.0,
    12.0, 13.0, 14.0, 15.0, 16.0, 17.0, 18.0, 19.0,
    20.0,

```

Se corre *ARWpost.exe* para generar un archivo con extensión *.ctl* en el directorio especificado en *namelist.ARWpost*:

```

1 $ cd ~/Build_WRF/WRF/ARWpost
2 $ ./ARWpost.exe

```

Se va al directorio donde se generó el archivo *.ctl* para generar un gráfico a partir de este:

```

1 $ ~/Build_WRF/ARWpost
2 $ grads
3 $ open test.ctl
4 $ d t2

```

El gráfico obtenido a partir de la variable **t2**, se muestra en la Figura 1.
 El gráfico obtenido a partir de la variable **tk**, se muestra en la Figura 2.
 El gráfico obtenido a partir de la variable **p**, se muestra en la Figura 3.
 El gráfico obtenido a partir de la variable **v**, se muestra en la Figura 4.

Figura 1: Variable t2.

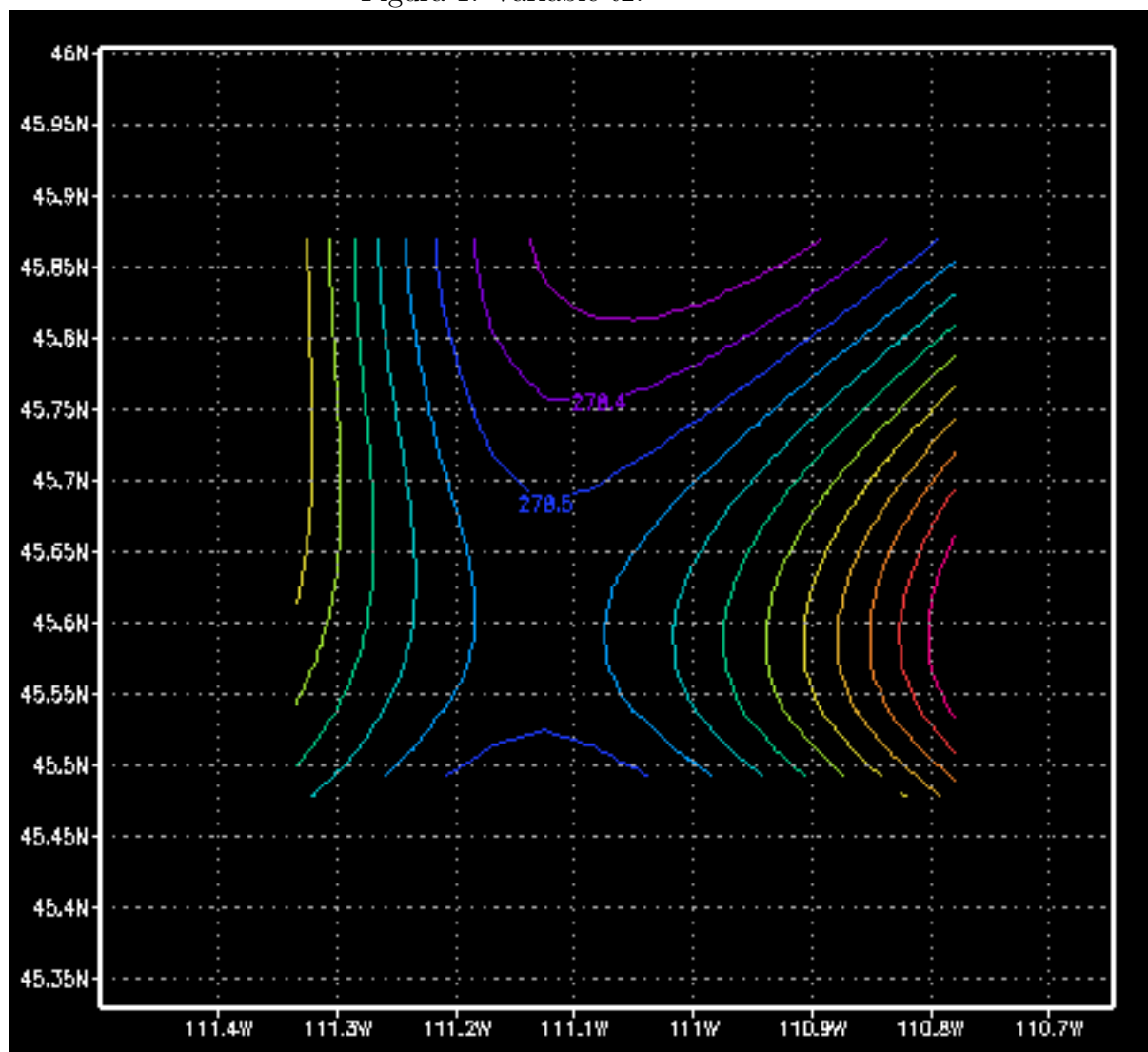


Figura 2: Variable tk.

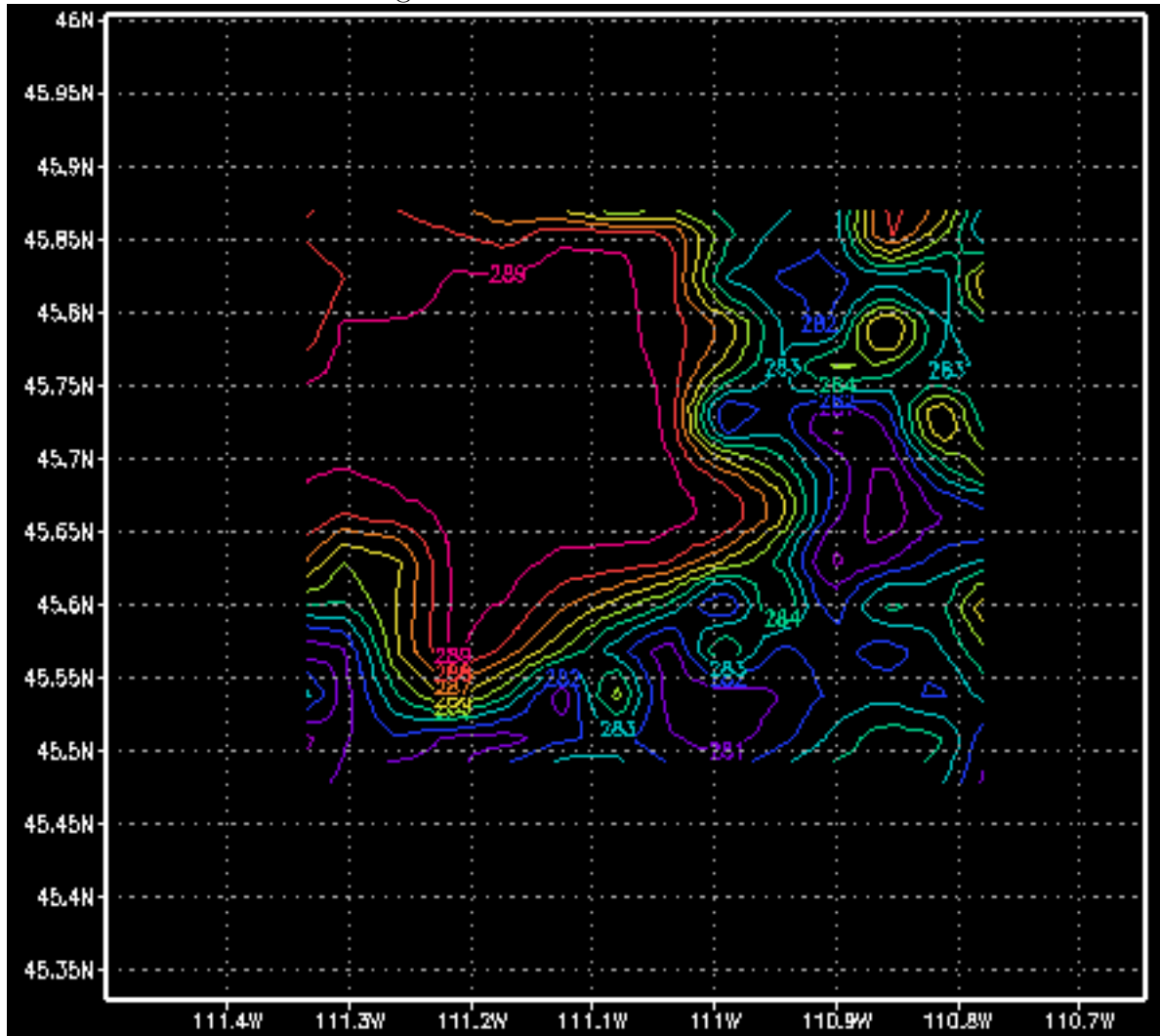


Figura 3: Variable p.

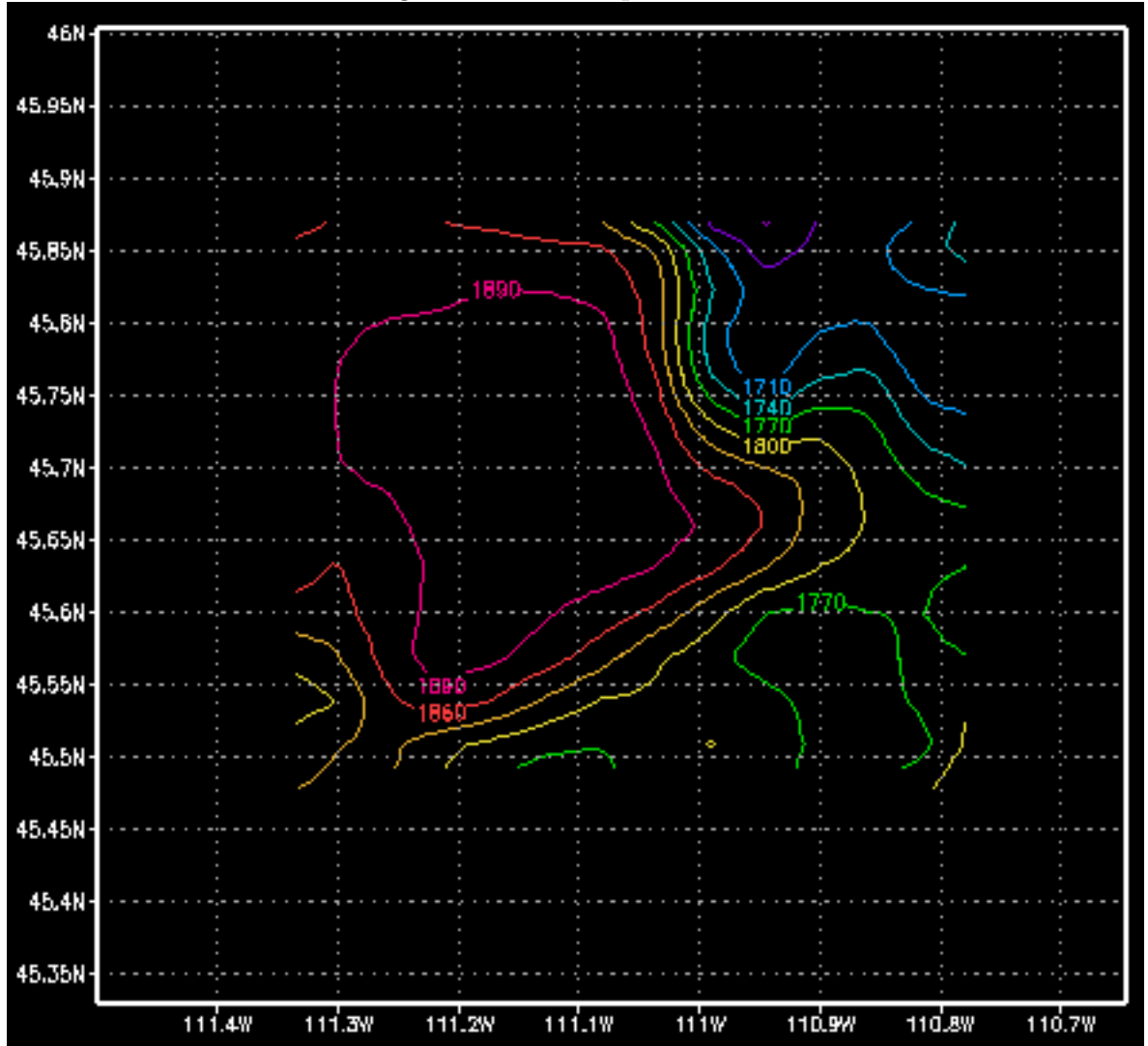
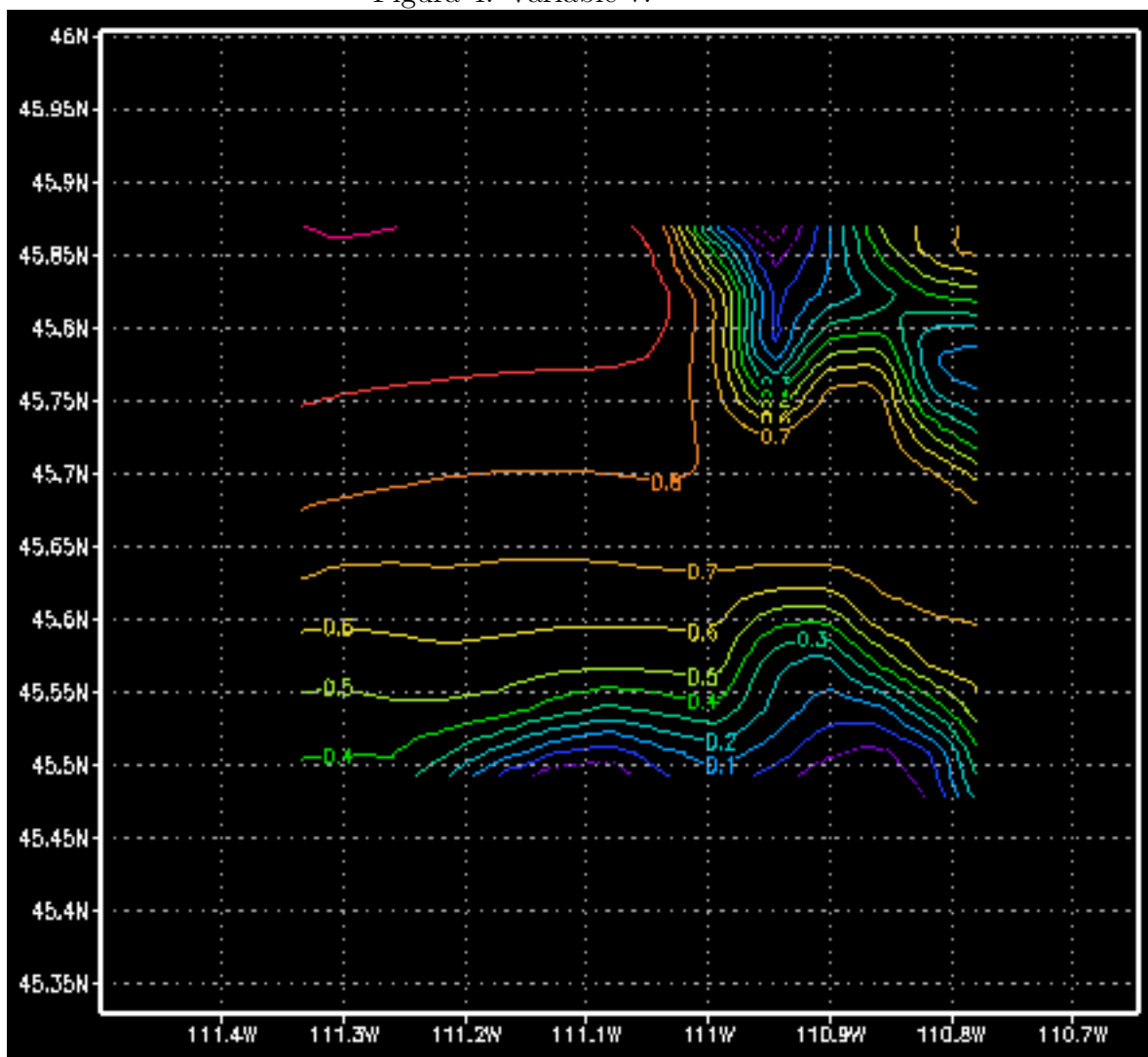


Figura 4: Variable v .



Observaciones

- De acuerdo a los permisos del usuario que instale WRF, WPS y ARW-post; puede ser necesario tener permisos de "superusuario" para algunas tareas. Esta situación se encontró principalmente al momento de correr ejecutables.
- Al momento de ejecutar *geogrid.exe*, dependiendo de la configuración del archivo *namelist.wps*, pueden existir errores que indiquen la carencia de algunos datos. En este caso, los archivos pueden descargarse manualmente desde: http://www2.mmm.ucar.edu/wrf/users/download/get_sources_wps_geog.html
- Al momento de ejecutar *metgrid.exe*, pueden existir errores que indiquen inconsistencias con la configuración de *geogrid.exe* y los parámetros en el archivo *namelist.wps*. Se debe tener en cuenta que la generación de los ejecutables es secuencial, por lo tanto *metgrid.exe* depende directamente de *geogrid.exe*.
- Algunos parámetros elegidos para compilar *WRF* y *WPS* dependen directamente de el equipo en el que se va a trabajar. El listado de opciones es mostrado directamente en el terminal, o puede ser visto en: http://www2.mmm.ucar.edu/wrf/users/docs/user_guide_V3.9/users_guide_chap2.htm