Chapter 10: Utilities and Tools

Table of Contents

Introduction

This chapter contains a number of short utilities to read and manipulate WRF-ARW data.

Also included in this chapter are references to some basic third party software, which can be used to view/change input and output data files.

read_wrf_nc

This utility allows a user to look at a WRF netCDF file at a glance.

What is the difference between this utility and the netCDF utility ncdump?

- This utility has a large number of options, to allow a user to look at the specific part of the netCDF file in question.
- The utility is written in Fortran 90, which will allow users to add options.
- This utility can be used for both WRF-ARW and WRF-NMM cores.□□
  It can be used for geogrid, metgrid and wrf input / output files.□□
  Only 3 basic diagnostics are available, pressure / height / tk, these can be activated with the -diag option *(these are only available for wrfout files)*

Obtain the **read_wrf_nc** utility from the WRF Download page (https://www2.mmm.ucar.edu/wrf/users/download/get_sources_pproc_util.html)

Compile

The code should run on any machine with a netCDF library *(If you port the code to a different machine, please create a post in the WRF & MPAS-A Support Forum to let the WRF development team know).*

To compile the code, use the compile flags at the top of the utility.

e.g.*, for a LINUX machine you need to type:*

```
pgf90  read_wrf_nc.f  -L/usr/local/netcdf/lib
   -lnetcdf  -lm  -I/usr/local/netcdf/include
   -Mfree  -o read_wrf_nc
```

If successful, this will create the executable: `read_wrf_nc`

Run

```
./read_wrf_nc  wrf_data_file_name  [-options]

options : [-h / help] [-att] [-m] [-M z] [-s]
          [-S x y z] [-v VAR] [-V VAR] [-w VAR]
          [-t t1 [t2]] [-times]
          [-ts xy X  Y  VAR VAR ....]
          [-ts ll  lat lon VAR VAR ....]
          [-lev z] [-rot] [-diag]
          [-EditData VAR]
```

Options:	*(Note: options [-att] ; [-t] and [-diag] can be used with other options)*
-h / help	Print help information.
-att	Print global attributes.
-m	Print list of fields available for each time, plus the min and max values for each field.
-M **z**	Print list of fields available for each time, plus the min and max values for each field. The min and max values of 3d fields will be for the **z** level of the field.
-s	Print list of fields available for each time, plus a sample value for each field. Sample value is taken from the middle of model domain.
-S **x y z**	Print list of fields available for each time, plus a sample value for each field. Sample value is at point **x y z** in the model domain.
-t **t1 [t2]**	Apply options only to times **t1** to **t2**. t2 is optional. If not set, options will only apply to **t1**.

-times	Print only the times in the file.
-ts	Generate time series output. A full vertical profile for each variable will be created. -ts xy X Y VAR VAR ….. will generate time series output for all VAR's at location X/Y -ts ll lat lon VAR VAR ….. will generate time series output for all VAR's at x/y location nearest to lat/lon
-lev **z**	Work only with option **–ts** Will only create a time series for level **z**
-rot	Work only with option **–ts** Will rotate winds to Earth coordinates
-diag	Add if you want to see output for the diagnostics temperature (K), full model pressure and model height *(tk, pressure, height)*
-v **VAR**	Print basic information about field **VAR**.
-V **VAR**	Print basic information about field **VAR**, and dump the full field out to the screen.
-w **VAR**	Write the full field out to a file **VAR.out**
	Default Options are [-att –s]

SPECIAL option: -EditData VAR

This option allows a user to **read** a WRF netCDF file, **change** a specific field, and **write** it BACK into the WRF netCDF file.
This option will CHANGE your CURRENT WRF netCDF file so TAKE CARE when using this option.
ONLY one field at a time can be changed; therefore, if you need 3 fields changed, you will need to run this program 3 times, each with a different "VAR"
IF you have multiple times in your WRF netCDF file – **by default ALL times** for variable "VAR" WILL be changed. *If you only want to change one time period, also use the "-t" option.*

HOW TO USE THIS OPTION:

Make a COPY of your WRF netCDF file before using this option

EDIT the subroutine USER_CODE

ADD an IF-statement block for the variable you want to change. This is to prevent a variable getting overwritten by mistake.

For REAL data arrays, work with the array "data_real" and for INTEGER data arrays, work with the array "data_int".

Example 1:
If you want to change all (all time periods too) values of U to a constant 10.0 m/s, you would **add** the following IF-statement:

```
  else if ( var == 'U') then
    data_real = 10.0
```

Example 2:
If you want to change a section of the LANDMASK data to SEA points:

```
  else if ( var == 'LANDMASK') then
    data_real(10:15,20:25,1) = 0
```

Example 3:
Change **all** ISLTYP category 3 values into category 7 values (NOTE this is an INTEGER field):

```
  else if ( var == 'ISLTYP') then
    where (data_int == 3 )
      data_int = 7
    end where
```

Compile and run the program.
You will be asked if this is really what you want to do.
ONLY the answer "yes" will allow the change to take effect.

iowrf

This utility allows a user to do some basic manipulation on WRF-ARW netCDF files.

- The utility allows a user to thin the data; de-stagger the data; or extract a box from the data file.

Obtain the **iowrf utility** from the WRF Download page: (https://www2.mmm.ucar.edu/wrf/users/download/get_sources_pproc_util.html).

Compile

The code should run on any machine with a netCDF *library (If you port the code to a different machine, please please create a post in the WRF & MPAS-A Support Forum to let the WRF development team know).*

To compile the code, use the compile flags at the top of the utility.

e.g., *for a LINUX machine you need to type:*

```
pgf90 iowrf.f  -L/usr/local/netcdf/lib  -lnetcdf  -lm
      -I/usr/local/netcdf/include  -Mfree  -o iowrf
```

If successful, this will create the executable: `iowrf`

Run

```
./iowrf  wrf_data_file_name  [-options]

options : [-h / help] [-thina X] [-thin X] [-box {}]
          [-A] [-64bit]
```

-thina **X**	Thin the data with a ratio of 1:X Data will be averaged before being fed back
-thin **X**	Thin the data with a ratio of 1:X No averaging will be done
-box **{}**	Extract a box from the data file. X/Y/Z can be controlled independently. e.g., -box **x** 10 30 **y** 10 30 **z** 5 15 -box **x** 10 30 **z** 5 15 -box **y** 10 30 -box **z** 5 15
-A	De-stagger the data – no thinning will take place
-64bit	Allow large files (> 2GB) to have read / write access

p_interp

This utility interpolates WRF-ARW netCDF output files to user-specified pressure levels. Several new capabilities have been supported in **p_interp** since October 2010. These includes:

- The ability to output fields needed to create *met_em* files, which can be used as input to *real.exe.* This output can be used to change the vertical resolution of WRF input files. Output from **p_interp** can also be used as input to TC bogusing or OBSGRID.
- A new namelist option is included to split input files containing multiple times into multiple output files, each with a separate time.
- **p_interp** can be compiled and ran in parallel to improve the time needed to processes large input files.
- Output from **p_interp** can now also be read directly by MET (*http://www.dtcenter.org/met/users/index.php*), removing the requirement to first run WPP before WRF-ARW data can be processed by the MET toolkit.

Obtain the **p_interp utility** from the WRF Download page: (https://www2.mmm.ucar.edu/wrf/users/download/get_sources_pproc_util.html).

Compile

The code should run on any machine with a netCDF *library (If you port the code to a different machine, please please create a post in the WRF & MPAS-A Support Forum to let the WRF development team know)*

To compile the code, use the compile flags at the top of the utility.

e.g.*, for a serial compile on a LINUX machine you need to type:*

```
pgf90 p_interp.F90  -L/usr/local/netcdf/lib
      -lnetcdf  -lm  -I/usr/local/netcdf/include
      -Mfree  -o p_interp
```

e.g.*, for a parallel compile on an IBM machine you need to type:*

```
mpxlf_r -qfree=f90 -L/usr/local/netcdf/lib -lnetcdf
      -lm -I/usr/local/netcdf/include -o p_interp
      p_interp.F90 -WF,-D_MPI
```

If successful, this will create the executable: `p_interp`

Edit the Namelist

Edit the associated *namelist.pinterp* file. *(see namelist options below).*

&io	Default value	Description
path_to_input	`./`	Path to input data
input_name	None – must be set in namelist	File name(s) of wrfout files. *Use wild character if more than one file is processed.*
path_to_output	`./`	Path where output data will be written
output_name	`' '`	If no name is specified, the output will be written to *input_name_PLEV*
process	`'all'`	Indicate which fields to process. 'all' fields in wrfout file *(diagnostics PRES, TT, HGT & RH will automatically be calculated)*; 'list' of fields as indicated in 'fields'
fields	`' '`	List of fields to process, if *'list'* is used in parameter *'process'*
debug	`.false.`	Set to `.true.` for more debugging
mpi_debug	`.false.`	Set to `.true.` for additional output that may be helpful when debugging parallel code.
bit64	`.false.`	Allow large files (> 2GB) to have read / write access.
met_em_output	`.false.`	Set to `.true.` to calculate the output fields needed in a *met_em* file. These files are used as input to *real.exe*.
split_output	`.false.`	`.true.` will output each time in the input file to a separate output file.

&interp_in	Default Value	Description

interp_levels	`-99999.`	List of pressure levels to interpolate data to
extrapolate	`0`	`0` - set values below ground and above model top to missing values *(default)* `1` - extrapolate below ground, and set above model top to model top values
interp_method	`1`	`1` - linear in p-interpolation *(default)* `2` - linear in log-p-interpolation
unstagger_grid	`.false.`	Set to `.true.` to unstagger the data on output

If `met_em_output` is set to `.true.` in the namelist, other options also need to be set:

```
split_output   = .true.
unstagger_grid = .false.
extrapolate    = 1
process        = 'all'
```

If you do not set any of the first 3 options as shown above, they will be reset automatically in the code. If `process` is set to `'list'`, the code will stop and the user will have to set `process` to `'all'`.

Also note that **p_interp** will stop if *met_em** files already exist in the `path_to_output` directory. This is to reduce the change of overwriting any *met_em** files created by `metgrid.exe`.

Run

To run **p_interp** compiled with the serial options, type

```
./p_interp
```

For distributed memory systems, some form of **mpirun** will be needed to run the executable. To run **p_interp** *(compiled with parallel options)* interactively, and using *x* processors, the command may look like:

```
mpirun -np x ./p_interp
```

On some systems, parallel interactive jobs may not be an option, in which case the command would be

```
mpirun ./p_interp
```

to run in a batch script. On some IBM systems, the parallel job launcher may be **poe** or **mpirun.lsf,** rather than **mpirun**.

TC Bogus Scheme

The ARW core for the WRF modeling system provides a simple Tropical Cyclone (TC) Bogussing scheme. It can remove an existing tropical storm, and may optionally bogus in a Rankine vortex for the new tropical storm. The input to the program is a single time-period and single domain of **`metgrid`** data, and a few namelist variables from the **`namelist.input`** file that describes the bogus TC's location and strength. The output is also a **`metgrid`**-like file. The scheme is currently only set up to process isobaric data. After running the tc.exe program, the user must manually rename the files so that the **`real.exe`** program can read the modified input.

Namelist Options

The namelist information for the TC scheme is located in an optional namelist record **`&tc`**. Only a single domain is processed. Users with multiple domains should horizontally-interpolate the generated meteorological fields to the fine-grid domains. Alternatively, users may run the **`tc.exe`** program on separate **`metgrid`** output files for different domains, though this is not recommended.

insert_bogus_storm	logical, insert a bogus storm
remove_storm	logical, removes an existing storm
num_storm	integer, number of storms to bogus, currently must be set to 1
latc_loc	real, latitude of bogus storm (+ north, - south)
lonc_loc	real, longitude of bogus storm (+ east, - west)
vmax_meters_per_second	real, maximum observed sustained wind speed (m/s)
rmax	real, radius from the cyclone center to where the maximum wind speed occurs (m)
vmax_ratio	real, scale factor for model's Rankine vortex

Note: If **`insert_bogus_storm`** is set to *true* then **`remove_storm`** should be set to *false*. If **`remove_storm`** is set to *true* then **`insert_bogus_storm`** should be set to *false*.

The value for **`vmax_ratio`** should be about 0.75 for a 45-km domain and about 0.90 for a 15-km domain (use these values to interpolate other for other resolutions). This is a representativeness scale factor. The observed maximum wind speed is not appropriate for an entire grid cell when the domain is fairly coarse.

For example, assume that a cyclone report came in with the storm centered at 25^{o} N and 75^{o} W, where the maximum sustained winds were observed to be 120 kts, with the maximum winds about 90 km from the storm center. With a 45-km coarse grid model domain, the **`namelist.input`** file would be:

```
&tc
insert_bogus_storm = .true.
remove_storm = .false.
latc_loc = 25.0
lonc_loc = -75.0
vmax_meters_per_second = 61.7
rmax = 90000.0
vmax_ratio = 0.75
/
```

Program tc.exe

The program **tc.exe** is automatically built along with the rest of the ARW executables. This, however, is a serial program. For the time being, it is the best to build this program using serial and no-nesting options.

Running tc.exe

1) Run all of the WPS programs as normal (**geogrid**, **ungrib**, and **metgrid**).
2) As usual, link-in the metgrid output files into either the **test/em_real** or the **run** directory.
3) Edit the **namelist.input** file for usage with the **tc.exe** program. Add-in the required fields from the **&tc** record, and only process a single time period.
4) Run **tc.exe**
5) Rename the output file, **auxinput1_d01_<date>** to the name that the **real.exe** program expects, **met_em.d01.<date>**, note that this will overwrite your original **metgrid.exe** output file for the initial time period.
6) Edit the **namelist.input** file to process all of the time periods for the **real.exe** program.

v_interp

This utility can be used to add vertical levels in WRF-ARW netCDF input. An example of the usage would be one-way nesting, via the program `ndown`. Since the program `ndown` does not do 'vertical nesting' prior to Version 3.2, namely adding vertical levels, this program can be used after running `ndown` to achieve the same results. Starting from Version 3.2, vertical levels may be added in the program `ndown`, via the namelist option '`vert_refine_fact`', which allows one to refine vertical levels by an integer factor.

The **v_interp** utility program can be obtained from the WRF Download page: (https://www2.mmm.ucar.edu/wrf/users/download/get_sources_pproc_util.html).

Compile

The code should be easily built and ran on any machine with a netCDF library. To compile the code, use the compile flags shown at the top of the utility program.

e.g.*, for a LINUX machine and pgf90 compiler, one may type:*

```
pgf90 v_interp.f  -L/usr/local/netcdf/lib -lnetcdf \
      -I/usr/local/netcdf/include  \
      -Mfree  -o v_interp
```

If successful, this will create the executable: `v_interp`

Run

Edit the namelist file `namelist.v_interp` *(see namelist options below)* for the number of new vertical levels (`nvert`) and the new set of levels (`nlevels`). To find out the existing model levels, check the original WRF `namelist.input` file used to create the input files, or type the following:

```
ncdump –v ZNW wrfinput_d01
```

The executable takes two arguments on the command line:

```
./v_interp file file_new
```

where `file` is the input file you want to add the vertical levels to, and `file_new` is the output file that contains more vertical levels. To run the program for `wrfinput` file, type

```
./v_interp wrfinput_d01 wrfinput_d01_new
```

For the `wrfbdy` file, type

```
./v_interp wrfbdy_d01 wrfbdy_d01_new
```

namelists:

&newlevels	
nvert	Number of new vertical levels (staggered)
nlevels	Values of new model levels

Program Notes:

When adding vertical levels, please keep the first- and the last-half levels the same as in the input file, itself. A problem may occur if levels are added outside the range.

For the wrfbdy file, please keep the input file name as wrfbdy_* since the program keys-in on the file name in order to do the interpolation for special boundary arrays.

proc_oml.f

This utility may be used to process 3D HYCOM (http://www.hycom.org) ocean model temperature data in netCDF format to produce initial ocean mixed layer depth field (`H0ML`) for use in a WRF simulation that uses the simple ocean mixed layer model option (`omlcall = 1`, and `oml_hml0 < 0`). The program estimates two fields from the HYCOM data: 1) effective mixed layer depth based on the idea of ocean heat content (`H0ML`); and 2) mean ocean temperature in the top 200 m depth (`TMOML`). This is used as the lower limit for cooling SST's in the wake of a hurricane.

To download the **proc_oml.f** utility, please see:
http://www2.mmm.ucar.edu/wrf/users/hurricanes/util.html

Compile

To compile the code, use the compile flags shown at the top of the utility program.
For example, for a LINUX machine and pgf90 compiler one may type:

```
pgf90 proc_oml.f  -L/usr/local/netcdf/lib -lnetcdf \
      -I/usr/local/netcdf/include -Mfree  -o proc_oml.f
```

If successful, this will create the executable: `proc_oml`

Run

To run the program, type

```
./proc_oml ocean-data-file.nc yyyymmddhh
```

where '`ocean-data-file.nc`' is the HYCOM ocean data file, and `yyyymmddhh` is the 10-digit date when the data is valid for (e.g. `2005082700`). Successfully running the program will produce an output file, `MLD`, which is in intermediate format as if it were produced by the `WPS/ungrib` program.

To use this field in `WPS/metgrid`, add it to '`constant_name`' as below:

```
constant_name = 'MLD',
```

V3.2 `WPS/metgrid` has the additional fields in METGRID.TBL for proper horizontal interpolation. For more information, please refer to the following presentation, at
http://www2.mmm.ucar.edu/wrf/users/tutorial/hurricanes/AHW_nest_ocean.pdf

Tools

Below is a list of tools that are freely available, and can be used very successfully to manipulate model data (both WRF model data, as well as other GRIB and netCDF datasets).

Converting Graphics

ImageMagick

ImageMagick is a software suite to create, edit, and compose bitmap images. It can read, convert and write images in a variety of formats (over 100) including DPX, EXR, GIF, JPEG, JPEG-2000, PDF, PhotoCD, PNG, Postscript, SVG, and TIFF. Use ImageMagick to translate, flip, mirror, rotate, scale, shear and transform images, adjust image colors, apply various special effects, or draw text, lines, polygons, ellipses and B_zier curves.

The software package is freely available from, http://www.imagemagick.org. Download and installation instructions are also available from this site.

Examples of converting data with ImageMagick software:

```
convert  file.pdf     file.png
convert  file.png     file.bmp
convert  file.pdf     file.gif
convert  file.ras     file.png
```

ImageMagick cannot convert ncgm (NCAR Graphics) file format to other file formats.

Converting ncgm (NCAR Graphics) file format

NCAR Graphics has tools to convert ncgm files to raster file formats. Once files are in raster file format, ImageMagick can be used to translate the files into other formats.

For *ncgm* files containing a single frame, use ***ctrans***.

```
ctrans  -d sun  file.ncgm  file.ras
```

For *ncgm* files containing multiple frames, first use ***med*** (metafile frame editor) and then *ctrans*. ***med*** will create multiple single frame files called *medxxx.ncgm*

```
med -e '1,$ split $' file.ncgm
ctrans  -d sun_ med001.ncgm > med001.ras
```

Basic Unix Commands

The WRF model is run on any Unix/Linux machine. Some basic Unix commands are required to work in this environment. There are numerous web sites one can visit to learn more about basic and advanced Unix commands. A couple of basic Unix commands are listed below, as well as some web sites where users can obtain more information.

mkdir / rmdir	To make *(mkdir)* or remove *(rmdir)* directories.
cd	To change to a new directory.
ls	List the files and directories in a directory .
ls -l	Lists your files in 'long format', which contains lots of useful information, e.g. the exact size of the file, who owns the file and who has the right to look at it, and when it was last modified.
ls –lrt	Lists your files in 'long format', in order of time stamp, and reverse order.
rm	Remove files.
more	Shows the first part of a file, just as much as will fit on one screen. Just hit the space bar to see more or **q** to quit.
cat	Shows the entire file on the screen.
head	Shows the first couple of lines of a file on screen.
tail	Shows the last couple of lines of a file on screen.
grep	Find lines that match patterns in files.
mv	Rename or move a file.
cp	Copy a file to a different name or location.
pwd	Shows the directory path you are currently in.
ln -sf	Makes a symbolic *(-s)* link *(ln)* of a file. The file will appear to be in two locations, but is only physically in one location. *(The –f option ensures that if the target file already exists, then it will first be unlink so that the link may occur correctly.)*
vi / emacs	File editors. For new users, ***emacs*** may be an easier editor to work with, as ***vi*** requires some extra understanding to navigate between the *command* and *insert* modes, whereas ***emacs*** functions more like a conventional editor.

http://mally.stanford.edu/~sr/computing/basic-unix.html
http://pangea.stanford.edu/computing/unix/shell/commands.php
http://en.wikipedia.org/wiki/List_of_Unix_utilities
http://www.cs.colostate.edu/helpdocs/vi.html

Design WRF model domains

WPS/util/plotgrids.ncl
Is an NCL script, which can either plot the domain on screen, or create a variety of different output types (pdf, ps, ncgm). This script must be ran in the same directory where the `namelist.wps` resides. This script only works with NCL version 6.1.0 or newer. If you still have an older version of NCL you can still use the plotgrids_old.ncl script.

Read more about this utility in Chapter 3 of this Users Guide.

Display ungrib (intermediate) files

WPS/util/plotfmt.ncl
Is an NCL script that can be used to display intermediate files created by `WPS/ungrib.exe`.

If you have created intermediate files manually, it is a very good practice to use this utility to display the data in your files first before running `WPS/metgrid.exe`.
***Note**: If you plan on manually creating intermediate files, refer to https://www2.mmm.ucar.edu/wrf/OnLineTutorial/Basics/IM_files/index.php for detailed information about the file formats and sample programs.*

This script reads intermediate files and can output the graphics in a variety of different output formats (on the screen, pdf, ps, ncgm). The script requires NCL version 6.2.0 or newer. An input file much be supplied, e.g:

```
ncl plotfmt.ncl 'filename="FILE:2005-06-01_00"'
```

WPS/util/int2nc.exe, can be used to convert intermediate files created by `WPS/ungrib.exe` into netCDF files.

WPS/util/plotfmt_nc.ncl
Is an NCL script, which can plot the netCDF output files created by int2nc.exe. This script must be run in the same directory where the netCDF files reside. The file to be plotted should be entered on the command line, e.g.,

```
ncl plotfmt_nc.ncl 'inputFILE="FILE:2005-06-01_00.nc"'
```

Read more about these utilities in **Chapter 3** of this Users Guide.

netCDF data

netCDF stands for **ne**twork **C**ommon **D**ata **F**orm.
Most of the information below can be used for WRF netCDF data, as well as other netCDF datasets.
netCDF is one of the current supported data formats chosen for WRF I/O API.

Advantages of using netCDF?
Most graphical packages support netCDF file formats
netCDF files are platform-independent (big-endian / little-endian)
A lot of software already exists that can be used to process/manipulate netCDF data

Documentation:
http://www.unidata.ucar.edu/ (General netCDF documentation)
https://www.unidata.ucar.edu/software/netcdf/docs-fortran/ (NETCDF User's Guide for FORTRAN)

Utilities:
ncdump
This is part of the netCDF libraries. Reads a netCDF file and prints information about the dataset. e.g.

`ncdump -h file` *(print header information)*
`ncdump -v VAR file` *(print header information and the full field VAR)*
`ncdump -v Times file` *(a handy way to see how many times are available in a WRF output file)*

ncview
Displays netCDF data graphically. No overlays, no maps and no manipulation of data possible.
http://meteora.ucsd.edu/~pierce/ncview_home_page.html

ncBrowse
Displays netCDF data graphically. Some overlays, maps and manipulation of data are possible.
https://www.pmel.noaa.gov/epic/java/ncBrowse/

read_wrf_nc
A utility to display basic information about WRF netCDF files.

iowrf
A utility to do some basic file manipulation on WRF-ARW netCDF files.

p_interp
A utility to interpolate WRF-ARW netCDF output files to user specified pressure levels.

netCDF operators
http://nco.sourceforge.net/
Stand-alone programs that can be used to manipulate data (by performing grid point averaging / file differencing / file 'appending'). *A couple of available programs are listed below, see the above link for a list of all available programs.*

ncdiff
Difference between two files; e.g.

```
ncdiff  input1.nc input2.nc output.nc
```

ncrcat
Writes specified variables / times to a new file; e.g.

```
ncrcat -v RAINNC wrfout* RAINNC.nc
ncrcat -d Time,0,231 -v  RAINNC  wrfout* RAINNC.nc
```

ncra
Averages variables and writes to a new file; e.g.

```
ncra -v OLR  wrfout* OLR.nc
```

ncks (nc kitchen sink)
Combination of NCO tools all in one (handy: one tool for multiple operations). An especially handy use of this tool is to split large files into smaller files, e.g.

```
ncks -A -F -d Time,1,1 wrfout* -o wrfout_time1.nc
```

GRIB data

Documentation and Decoders

Documentation and decoders for both GRIB1 and GRIB2 can be found here: http://rda.ucar.edu/#!GRIB
Some of the utilities that are worth looking at is the unpackgrib2.c and grib2to1.c code.

GRIB codes

It is important to understand the GRIB codes to know which fields are available in your dataset. For instance, NCEP uses the GRIB1 code 33 for the U-component of the wind, and 34 for the V-component. *Other centers may use different codes, so always obtain the GRIB codes from the center you get your data from.*

GRIB2 uses 3 codes for each field - **product**, **category** and **parameter**. We would most often be interested in **product 0** (*Meteorological products*). **Category** refers to the type of field; e.g., category 0 is temperature, category 1 is moisture and category 2 is momentum. **Parameter** is the field number.
So whereas GRIB1 only uses code 33 for the U-component of the wind, GRIB2 will use 0,2,2, for the U-component, and 0,2,3 for the V-component.

Display GRIB header/field information

GRIB1 data

WPS/util/g1print.exe
wgrib

GRIB2 data

WPS/util/g2print.exe
wgrib2

Both wgrib and wgrib2 are available from the http://rda.ucar.edu/#!GRIB web site.

Convert GRIB data to netCDF format

ncl_grib2nc (http://www.ncl.ucar.edu/Document/Tools/ncl_convert2nc.shtml)

Displaying GRIB files

GRIB data can, amongst other, be displayed with GrADS with the use of the grib2ctl.pl script (http://www.cpc.ncep.noaa.gov/products/wesley/grib2ctl.html) and Panoply (http://www.giss.nasa.gov/tools/panoply/).

Model Verification

MET is designed to be a highly configurable, state-of-the-art suite of verification tools. It was developed using output from the Weather Research and Forecasting (WRF) modeling system, but may be applied to the output of other modeling systems as well.

MET provides a variety of verification techniques, including:

- Standard verification scores, comparing gridded model data to point-based observations
- Standard verification scores, comparing gridded model data to gridded observations
- Object-based verification method, comparing gridded model data to gridded observations

http://www.dtcenter.org/met/users/index.php