Formats:

Grib-2

NCEP documentation: documentation for grib version 2

http://www.nco.ncep.noaa.gov/pmb/docs/grib2/grib2_doc.shtml

wgrib2: inventory, decode, manipulate, write grib2 data

http://www.cpc.ncep.noaa.gov/products/wesley/wgrib2.html

introduction: ftp://ftp.cpc.ncep.noaa.gov/wd51we/wgrib2/README

tricks: ftp://ftp.cpc.ncep.noaa.gov/wd51we/wgrib2/tricks.wgrib2

GrADS: visualization and much more

grads home page: http://www.iges.org/grads/

make ctl: http://www.cpc.noaa.gov/products/wesley/g2ctl.html

copygb2: grid interpolation (grib2 -> grib2) (not easy to compile)

http://www.nco.ncep.noaa.gov/pmb/codes/nwprod/util/sorc/copygb2.fd/

cnvgrib: convert from grib1 to grib2 and back again

http://www.nco.ncep.noaa.gov/pmb/codes/GRIB2/

libraries: NCEP has fortran and C libraries

http://www.nco.ncep.noaa.gov/pmb/codes/GRIB2/

other programs (little or no experience)

GDAL: geospatial data abstraction library, grib2 -> GIS formats

http://www.gdal.org/

degrib: NWS code

http://www.nws.noaa.gov/mdl/degrib/

ECMWF: gribex, grib API

http://www.ecmwf.int/products/data/software/

NCL: NCAR command language

http://www.ncl.ucar.edu/

and others

Formats:

Grib-1

ON-388: documentation for grib version 1

http://www.nco.ncep.noaa.gov/pmb/docs/on388/

wgrib: inventory, decode, manipulate grib1 data

http://www.cpc.ncep.noaa.gov/products/wesley/wgrib.html

introduction: ftp://ftp.cpc.ncep.noaa.gov/wd51we/wgrib/readme

tricks: ftp://ftp.cpc.ncep.noaa.gov/wd51we/wgrib/tricks.wgrib

GrADS: visualization and much more

grads home page: http://www.iges.org/grads/

make ctl: http://www.cpc.noaa.gov/products/wesley/grib2ctl.html

copygb: grid interpolation (grib -> grib)

http://www.nco.ncep.noaa.gov/pmb/codes/nwprod/util/sorc/copygb.fd/

http://www.cpc.noaa.gov/products/wesley/copygb.html

cnvgrib: convert from grib1 to grib2 and back again

http://www.nco.ncep.noaa.gov/pmb/codes/GRIB2/

gribpoint.pl: get value at point (requires perl, copygb, wgrib)

ftp://ftp.cpc.ncep.noaa.gov/wd51we/wgrib.scripts/

libraries: many available

other programs (little or no experience)

GDAL: geospatial data abstraction library, grib -> GIS formats

http://www.gdal.org/

degrib: NWS code

http://www.nws.noaa.gov/mdl/degrib/

ECMWF: gribex, grib API

http://www.ecmwf.int/products/data/software/

NCL: NCAR command language

http://www.ncl.ucar.edu/

and many others

Formats:

Netcdf: common in university environments

Most of the Reanalysis producers are operational meteorological centers and need to use WMO-standard grib. So the problem is converting from grib to netcdf.

GIS: common in many environments

grib -> netcdf:

NCL: NCAR command language

http://www.ncl.ucar.edu/

grib2nc: unidata

http://mailman.unidata.ucar.edu/software/decoders/

degrib: NWS code

http://www.nws.noaa.gov/mdl/degrib/

GrADS: with LATS or LATS4D package

http://www.iges.org/grads/

wgrib2: grib2 only

http://www.cpc.ncep.noaa.gov/products/wesley/wgrib2.html

GDAL: geospatial data abstraction library, grib -> GIS formats

http://www.gdal.org/

grib -> GIS formats

GDAL: geospatial data abstraction library, grib -> GIS formats

http://www.gdal.org/

GrADS: need version 2

http://www.iges.org/grads/

degrib: shape files

http://www.nws.noaa.gov/mdl/degrib/

grb2grid: grib1 to ascii raster file

http://www.cpc.ncep.noaa.gov/products/wesley/grb2grid.html

Caution: when converting to other formats, metadata is often lost. For example, the precipitation averaged from the 0-6 hour forecast with an initial time of 00Z, January 1, 2000 may end up with a descriptor of “00Z January 1, 2000 precipitation”.

I have a Grib-1 file, what is it?

Grib is a WMO format for sending gridded data. Grib version 1 is in wide but declining use. Sites have been converting to grib version 2. Once you have a grib file, you want to see what is in it.

wgrib -s: short inventory

```
-sh-3.00$ wgrib -s narr.t09z.awip32.merged
1:0:d=09102809:MSLET:MSL:anl:NAve=0
2:166602:d=09102809:PRMSL:MSL:anl:NAve=0
3:333204:d=09102809:PRES:hybrid lev 1:anl:NAve=0
4:440414:d=09102809:HGT:hybrid lev 1:anl:NAve=0
5:607016:d=09102809:TMP:hybrid lev 1:anl:NAve=0
6:737984:d=09102809:POT:hybrid lev 1:anl:NAve=0
..
422:56074230:d=09102809:WVUFLX:atmos col:0-3hr acc:NAve=0
423:56264588:d=09102809:WVVFLX:atmos col:0-3hr acc:NAve=0
```

-s is the short inventory
column 1 = message (record) number, the file has 423 records (fields)
column 2 = byte location starting from 0
column 3 = analysis time or initial time of the forecast
column 4 = variable name
column 5 = level/layer
column 6 = timing information, anl=analysis, acc=accumulation, ave=average
column 7 = number of fields used to make an average/accumulation

The variable depends on the grib number, the grib table and the originating center. For example, the zonal wind is called by UGRD (NCEP), U (ECMWF), and uvel (CPTEC). All the centers use “33” as the grib number for the zonal wind. However, you shouldn't use “33” as meaning the zonal wind because “33” could also be the “snow density” or something else when a different table is used.

It takes some time to learn the cryptic names. Why did NCEP call the zonal wind UGRD or the snow WEASD? They had their reasons that few understand. Anyways, description of the NCEP names are here, or you could use the -v option.

http://www.nco.ncep.noaa.gov/pmb/docs/on388/table2.html

wgrib -v: verbose inventory

Another way of seeing the description of each variable is by the -v option of wgrib.

```
-sh-3.00$ wgrib -v narr.t09z.awip32.merged | more
1:0:D=2009102809:MSLET:MSL:kpds=130,102,0:anl:winds in grid direction:"Mean sea level pressure (Mesinger method) [Pa]
2:166602:D=2009102809:PRMSL:MSL:kpds=2,102,0:anl:winds in grid direction:"Mean sea level pressure (Shuell method) [Pa]
3:333204:D=2009102809:PRES:hybrid lev 1:kpds=1,109,1:anl:winds in grid direction:"Pressure [Pa]
4:440414:D=2009102809:HGT:hybrid lev 1:kpds=7,109,1:anl:winds in grid direction:"Geopotential height [gpm]
5:607016:D=2009102809:TMP:hybrid lev 1:kpds=11,109,1:anl:winds in grid direction:"Temp. [K]
6:737984:D=2009102809:POT:hybrid lev 1:kpds=13,109,1:anl:winds in grid direction:"Potential temp. [K]
7:880830:D=2009102809:RH:hybrid lev 1:kpds=52,109,1:anl:winds in grid direction:"Relative humidity [%]
..
422:56074230:D=2009102809:WVUFLX:atmos col:kpds=242,200,0:0-3hr acc:winds are N/S:"Water vapor zonal transport (vertical int)[kg/m]
423:56264588:D=2009102809:WVVFLX:atmos col:kpds=243,200,0:0-3hr acc:winds are N/S:"Water vapor meridional transport (vertical int) [kg/m]
```

-v is the verbose inventory
column 1 = message (record) number, the file has 423 records (fields)
column 2 = byte location starting from 0
column 3 = analysis time or initial time of the forecast
column 4 = variable name
column 5 = level/layer
column 6 = variable, level1 level2 integer values, kpds(5)..kpds(7) as used by the NCEP grib library (w3lib)
column 7 = timing information, anl=analysis, acc=accumulation, ave=average
column 8 = wind orientation, see the wind orientation section
column 9 = description of the variable

wgrib -V: really verbose inventory

The really verbose inventory give much more detail about each record. The inventory can be really big, so I usually do the inventory on a few records. One easy way is to use the "-d N" option to dump a selected record. Note that if you don't have write permission in the current directory, you'll have to change the default dump file by "-o file".

```
sh-3.00$ wgrib narr.t09z.awip32.merged -V -d 1
rec 1:0:date 2009102809 MSLET kpds5=130 kpds6=102 kpds7=0 levels=(0,0) grid=221 MSL anl: bitmap: 1648 undef
  MSLET=Mean sea level pressure (Mesinger method) [Pa]
  timerange 0 P1 0 P2 0 TimeU 1  nx 349 ny 277 GDS grid 3 num_in_ave 0 missing 0
  center 7 subcenter 15 process 140 Table 131 scan: WE:SN winds(grid)
  Lambert Conf: Lat1 1.000000 Lon1 -145.500000 Lov -107.000000
     Latin1 50.000000 Latin2 50.000000 LatSP 0.000000 LonSP 0.000000
     North Pole (349 x 277) Dx 32.463000 Dy 32.463000 scan 64 mode 8
  min/max data 96683.1 103519  num bits 13  BDS_Ref 96683.1  DecScale 0 BinScale 0
```

-V = more verbose than -v

-d N = dump message N (default output file is dump)
has the side effect of only displaying the inventory of the selected record
1:0 = record number and byte location
MSLET = variable
grid=221 = NCEP grid number
bitmap: 1648 undef = a undefined bitmap is used and 1648 grid points are undefined
nx 349 ny 277: grid size
GDS grid 3 = lambert conformal grid
center 7 = NCEP (see WMO documentation)
subcenter 15 = NARR project
Table 131 = NCEP grib table
scan: WE:SN = how the grid is stored

```
do y = south to north
    do x = west to east
        write   data(x,y)
    enddo
enddo
```

winds(grid) = orientation of the winds. This flag is ignored for scalar fields.
For vector fields, NARR uses, winds(N/S)
example wgrib narr.t09z.awip32.merged -V -d 11
Lambert Conf: .... = all the parameters needed to define the grid
min/max data 96683.1 103519 = minimum and maximum values. The mean sea level pressure is in Pascals, and the minimum values are associated with a deep cyclone off the coast of Korea.
num bits 13 = field values are stored as scaled 13 bit integers.
BDS_Ref 96683.1 = scaled minimum value
DecScale 0 BinScale 0 = the decimal scaling factors are 10**0 and 2**0.
Normally the field values are (i * 2**BinScale + BDS_Ref) * 10**(-DecScale)

Wind Orientation

Vectors such as winds, ocean currents, and surface stresses can be oriented relative to the grid or the North pole. For example, a positive meridional wind can point from the (x,y) grid point to the (x,y+1) grid point (grid orientation) or towards the North pole. The grid orientation makes sense for the model developers and the North pole orientation makes sense for the users. The wind orientation file can be seen in the -v and -V inventories.

At NCEP, most winds are grid oriented. Consequently the zonal wind is called UGRD

which is short for U_GRID. For the latitude-longitude, Mercator and Gaussian grids, the grid and NP orientations are the same and the orientation flag can be ignored. However, for the polar stereographic and Lambert conformal grids, you need to pay attention to the wind orientation. Most of the NCEP produced polar stereographic and Lambert conformal grids will use grid oriented winds, except for NARR. With NARR, we decided to make life easier for the new users and life more difficult for current users of the NCEP regional models.

Some software understands the wind orientation. Copygb works correctly and the more recent versions of grads/grib2ctl/g2ctl work correctly. The quick check is to plot 500 mb heights and winds. If the winds are really ageostrophic near the edges, then your software is not handling the wind orientation correctly.

Grid

The grid information can be obtained by the -V wgrib inventory.

```
-sh-3.00$ wgrib narr.t09z.awip32.merged -V -d 1
rec 1:0:date 2009102809 MSLET kpds5=130 kpds6=102 kpds7=0 levels=(0,0) grid=221 MSL anl: bitmap: 1648 undef
  MSLET=Mean sea level pressure (Mesinger method) [Pa]
  timerange 0 P1 0 P2 0 TimeU 1  nx 349 ny 277 GDS grid 3 num_in_ave 0 missing 0
  center 7 subcenter 15 process 140 Table 131 scan: WE:SN winds(grid)
  Lambert Conf: Lat1 1.000000 Lon1 -145.500000 Lov -107.000000
     Latin1 50.000000 Latin2 50.000000 LatSP 0.000000 LonSP 0.000000
     North Pole (349 x 277) Dx 32.463000 Dy 32.463000 scan 64 mode 8
  min/max data 96683.1 103519  num bits 13  BDS_Ref 96683.1  DecScale 0 BinScale 0
```

Lambert Conf: .... = all the parameters needed to define the Lambert conformal grid.
Consult a text book for the meaning of the parameters.

Here is an example of a file from CFSR.

```
-sh-3.00$ wgrib -d 1 -V pgblnl.gdas.2007010100
rec 1:0:date 2007010100 HGT kpds5=7 kpds6=100 kpds7=1 levels=(0,1) grid=2 1 mb anl:
  HGT=Geopotential height [gpm]
  timerange 10 P1 0 P2 0 TimeU 1  nx 144 ny 73 GDS grid 0 num_in_ave 0 missing 0
  center 7 subcenter 0 process 82 Table 2 scan: WE:NS winds(N/S)
  latlon: lat  90.000000 to -90.000000 by 2.500000  nxny 10512
        long 0.000000 to -2.500000 by 2.500000, (144 x 73) scan 0 mode 128 bdsgrid 1
  min/max data 44192 50495.3  num bits 20  BDS_Ref 4.4192e+06  DecScale 2 BinScale 0
```

The parameters for a latitude-longitude grid are easier to interpret. In this example, the

grid is 144 points in the longitude and 73 points in the latitude. It goes from 90S to 90N by 2.5 degree increments. It goes from 0E to 2.5W in 2.5 degree steps.

People need to associate latitude and longitudes with each grid point. For the latitude-longitude grid, it is easy. However, the other grids are more involved. One could use the grid parameters from the grib file to generate the latitude and longitudes for each grid point. However, this can be difficult for most users. So grib files with the latitude and longitude of each grid point are often available. The NARR grib file with the latitude, longitudes, surface elevation and other useful fields is available at

ftp://ftp.cpc.ncep.noaa.gov/wd51we/NARR/rr-fixed.grb

Grib-1 Scan Order

The grib format allows the data to be in C-order (y varies the fastest), fortran-order (x varies the fastest), progressing northward or southward, progressing eastward or westward. In grib-1, data can be saved in one of 8 different scan orders. The two common orders are

WE:SN	scan from west to east (row), rows go from south to north
WE:NS	scan from west to east (row), rows go from north to south

```
(1,4) (2,4) (3,4) (4,4)    ^
(1,3) (2,3) (3,3) (4,3)    |
(1,2) (2,2) (3,2) (4,2)    N   ---E--->
(1,1) (2,1) (3,1) (4,1)    |
```

The gridded data is stored sequentially.

If (WE:SN) the order is
(1,1) (2,1) (3,1) (4,1) (1,2) ... (4,2) (1,3) ..(3,4) (4,4)
If (WE:NS) the order is
(1,4) (2,4) (3,4) (4,4) (1,3) ... (4,3) (1,2) ..(3,1) (4,1)

When you decode data with many programs, including wgrib, you will get a sequence of numbers. You'll need to know the scan order to interpret the numbers.

The grib-2 scan order is more complicated and will be described later.

I have a Grib-2 file, what is it?

Grib version 2 (grib-2) is the current WMO standard, replacing grib version 1(grib-1). Grib-2 fixes many of the small problems in grib-1, adds compression, more metadata and is not backwards compatible with grib-1. It is a more complicated standard and required all the software to be updated or rewritten.

With grib-1 files, we showed how to examine the contents using wgrib. For grib-2, we will using the program wgrib2. Wgrib2 is different from most programs. In most programs, options (-somethings) initialize flags or internal variables. In wgrib2 lingo, these are called "init" options. However, most wgrib2 options are mapped into function calls. For each record processed, the option functions are executed from left to right. For example, the following line will first write the data as text into a.txt (-text a.txt), write the data as binary (-bin a.bin) and finally write the short inventory (-s). This sequence is repeated for each record.

```
wgrib2 file.grb2 -text a.txt -bin a.bin -s
```

wgrib2 -s: short/default inventory

Here is an inventory for a CFSR file.

```
-sh-3.00$ wgrib2 pgblnl.gdas.2007010100.grb2 -s
1:4:d=2007010100:HGT:1 mb:anl:
2:16552:d=2007010100:TMP:1 mb:anl:
3:22064:d=2007010100:RH:1 mb:anl:
4:23631:d=2007010100:SPFH:1 mb:anl:
5:30587:d=2007010100:VVEL:1 mb:anl:
6.1:46414:d=2007010100:UGRD:1 mb:anl:
6.2:46414:d=2007010100:VGRD:1 mb:anl:
7:61900:d=2007010100:ABSV:1 mb:anl:
...
474:5462530:d=2007010100:VWSH:PV=-2e-06 (Km^2/kg/s) surface:anl:
475:5472085:d=2007010100:PRMSL:mean sea level:anl:
```

column 1: message or message.submessage number

In grib-2, a message can have more than one field. In this file, the zonal and

meridional winds are saved in the sixth message. The CFSRR may have the U and V winds in the same message as shown above or in different messages

column 2: the byte location of the grib message

column 3: the analysis or start of forecast time, use -T to see the minutes and seconds

column 4: variable name

column 5: level

column 6 = timing information, anl=analysis, acc=accumulation, ave=average
fcst = forecast

Note: when wgrib2 is parsing the command line, it checks for any "inv" options. If none are found, a "-s" is added to the end of the command line.

wgrib2 -v -s: full variable names

The variable names are still cryptic in grib2. A translation table is given by increasing the verbosity to level 1.

```
-sh-3.00$ wgrib2 pgblnl.gdas.2007010100.grb2 -v
1:4:d=2007010100:HGT Geopotential Height [gpm]:1 mb:anl:
2:16552:d=2007010100:TMP Temperature [K]:1 mb:anl:
3:22064:d=2007010100:RH Relative Humidity [%]:1 mb:anl:
...
```

Increasing the verbosity to level 2 prints the date in "GrADS" format and give a numeric values of the level/layer information.

```
-sh-3.00$ wgrib2 pgblnl.gdas.2007010100.grb2 -v2 | head
1:4:00Z01jan2007:HGT Geopotential Height [gpm]:lvl1=(100,100) lvl2=(255,missing):1 mb:anl:
2:16552:00Z01jan2007:TMP Temperature [K]:lvl1=(100,100) lvl2=(255,missing):1 mb:anl:
3:22064:00Z01jan2007:RH Relative Humidity [%]:lvl1=(100,100) lvl2=(255,missing):1 mb:anl:
4:23631:00Z01jan2007:SPFH Specific Humidity [kg/kg]:lvl1=(100,100) lvl2=(255,missing):1 mb:anl:
```

Grid and Scan information

wgrib2 is improved over wgrib when dealing with the grid and scan order. For the commonly used grids, wgrib2 now knows the latitude and longitude of each grid point. Consequently a big stumbling block for some users has been eliminated. As for the scan order, grib-2 supports 16 different ways ordering the data (scan modes). The new modes are hard to describe but are used at the National Digital Forecast Database (NDFD)

because they improved the efficiency of certain compression schemes. It is too much to expect the users to deal with hard-to-describe scan orders, so wgrib2 converts the data into WE:SN order by default.

To get the grid information, the -V option can be used. In the following example,

```
-sh-3.00$ wgrib2 pgblnl.gdas.2007010100.grb2 -V -d 6.2
6.2:46414:vt=2007010100:1 mb:anl:VGRD V-Component of Wind [m/s]:
   ndata=10512:undef=0:mean=0.279861:min=-114.9:max=102.7
   grid_template=0:
      lat-lon grid:(144 x 73) units 1e-06 input WE:NS output WE:SN res 48
      lat 90.000000 to -90.000000 by 2.500000
      lon 0.000000 to 357.500000 by 2.500000 #points=10512

    field 1: 6.2 = message 6, submessage 2
    field 2: 46414 = byte location
    field 3: vt = verification time
    ndata = number of grid points
    undef = number of grid points with undefined values
    mean = average value of the grid point (not area weighted)
    min/max = minimum and maximum grid point value
    input WE:NS = the grib file has the data stored in WE:NS order
    output WE:SN = the data has been decoded into a WE:SN order
        If the data were to be written to a binary/text file, it will be
        written in a WE:SN order (different from wgrib). The output
        order can be changed using the -order option. Note than many
        of the options only work in WE:SN order.
```

Wgrib2, by default, writes non-grib data in a WE:SN order. Some people like a WE:NS order or even the original order. Wgrib2 can handle this by changing by using the -order option. The following examples write a text file in raw and we:ns order.

```
-sh-3.00$ wgrib2 pgblnl.gdas.2007010100.grb2 -order raw -s -scan -text text.dat -d 1
1:4:d=2007010100:HGT:1 mb:anl::scan=0 input=WE:NS output=raw
```

```
-sh-3.00$ wgrib2 pgblnl.gdas.2007010100.grb2 -order we:ns -s -scan -text text.dat -d 1
1:4:d=2007010100:HGT:1 mb:anl::scan=0 input=WE:NS output=WE:NS
```

As mentioned before, you should try to keep the output order in we:sn because many

wgrib2 functions require the data to be in that order.

Viewing grib-1 and grib-2 data using GrADS

There are many ways to visualize grib-1 and grib-2 data. You can convert the data into another format and use your favorite visualization program or you can use a program that can read the grib data directly. The advantages of keeping the data in grib format are efficiency. A grib-1 format is typically one-third the size of netcdf-3 database. Grib-2 uses compression which makes the file 20% to 50% the size of the grib-1 file. Since reanalysis datasets are often very big, compression is necessary. Of the programs that can directly read grib, GrADS is very powerful, open source and is available on many machines. Other programs can be used to view grib data; proponents of other systems are everywhere, including NCEP. Nevertheless, GrADS gets the job done and handles all the NCEP grib files.

GrADS and grib

GrADS uses a 4 dimensional model: zonal direction, meridional direction, vertical direction and time. Grib files have many time quantities: analysis/forecast start time, forecast valid time, start of an averaging/accumulation period and end of averaging/accumulation period. For example, a grib message could be the 12Z 1/1/2000 temperature forecast for a forecast run initialized at 00z 1/1/2000. The grib message has two time coordinates but GrADS only uses 1 time coordinates. GrADS gets around this problem by allowing you to select the analysis/initial forecast time or the verification time as the time coordinate. Here are steps for making the GrADS control and index files.

Grib-1 Analysis:

```
-sh-3.00$ grib2ctl.pl narr.t09z.awip32.merged >tmp.ctl
-sh-3.00$ gribmap -0 -i tmp.ctl
```

Grib-1 Forecasts, using the verification time:

```
-sh-3.00$ grib2ctl.pl -verf narr.t09z.awip32.merged >tmp.ctl
-sh-3.00$ gribmap -i tmp.ctl
```

Grib-2 Analysis:

```
-sh-3.00$ g2ctl -0 narr.t09z.awip32.merged.grb2 >tmp.ctl
-sh-3.00$ gribmap -0 -i tmp.ctl
```

Grib-2 Forecasts, using the verification time:

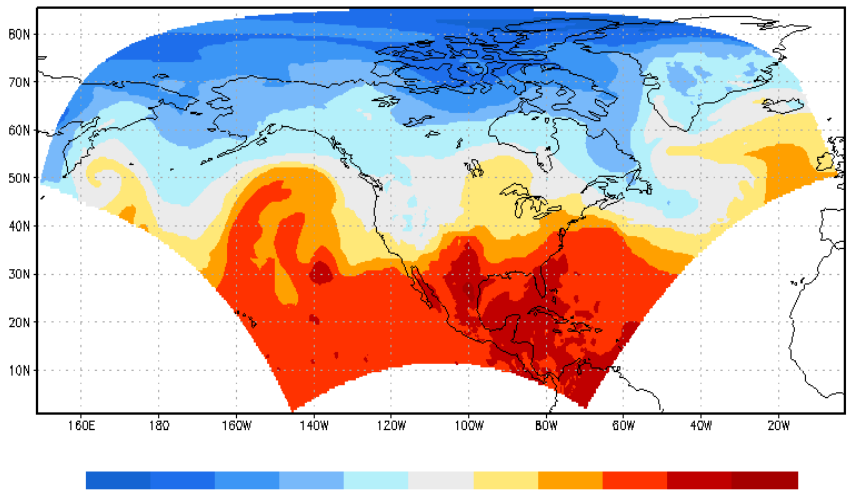
```
-sh-3.00$ g2ctl narr.t09z.awip32.merged.grb2 >tmp.ctl
-sh-3.00$ gribmap -i tmp.ctl
```

Once you have made the control and index files, you run GrADS. (Version 2 is need for grib-2 support.)

Sample session: plotting the 850 mb temperature

```
-sh-3.00$ grads
 (text deleted)
Landscape mode? ('n' for portrait):
GX Package Initialization: Size = 11 8.5
ga-> open tmp.ctl
 (text deleted)
ga-> q file
 (text deleted)
ga-> set lev 850
LEV set to 850 850
ga-> set gxout shaded
ga-> d tmpprs
Notice:  Automatic Grid Interpolation Taking Place
Contouring: 250 to 295 interval 5
ga-> cbar
```

More information on GrADS can be obtained from http://www.iges.org/grads/.



GIS

Some people want GIS compatible data. GDAL (Geospatial Data Abstraction Library) can read grib data and write a number of GIS format. You can also use GrADS v2 to write GIS formatted data. There are several applications that can convert latitude-longitude into ascii raster data. Unfortunately they will not work for the NARR data which is on a conformal grid. The grb2grid script can regrid the NARR data to a latitude-longitude grid and then write out a ascii raster file.

Netcdf

Netcdf reanalysis files are available from NCAR (Boulder CO) and ERSL (Boulder CO) including R1, R2, NARR and CFSR (future). You can also convert grib files into netcdf using previously listed programs.

Grib-1 to Raw Numbers

Sooner or later, you'll want some raw numbers. You can write fortran or C programs that call a grib library. An easier method is to dump the data into a file (binary or text) and then read that file. In this example, wgrib will be used to create a CSV file of 2 meter temperatures.

Making a dump of the 2 m temperature

a) Seeing what is in the file:

```
-sh-3.00$ wgrib -s narr.t09z.awip32.merged
:0:d=09102809:MSLET:MSL:anl:NAve=0
:166602:d=09102809:PRMSL:MSL:anl:NAve=0
..
```

b) Seeing what temperature fields are available

```
-sh-3.00$ wgrib -s narr.t09z.awip32.merged | grep ':TMP:'
5607016:d=09102809:TMP:hybrid lev 1:anl:NAve=0
..
288:37967126:d=09102809:TMP:2 m above gnd:anl:NAve=0
..
```

c) Making a list of fields to decode (TMP 2 meter)

```
-sh-3.00$ wgrib -s narr.t09z.awip32.merged | grep ':TMP:2 m above gnd:'
288:37967126:d=09102809:TMP:2 m above gnd:anl:NAve=0
```

d) Finally making a text file of the 2 m temperatures

```
-sh-3.00$ wgrib -s narr.t09z.awip32.merged | grep ':TMP:2 m above gnd:' |  \
        wgrib  -s narr.t09z.awip32.merged -i -text -o t2m.txt
288:37967126:d=09102809:TMP:2 m above gnd:anl:NAve=0
-sh-3.00$ cat t2m.txt
349 277                         : nx ny of the grid
299.908                         : temperature in K of 1st grid point
299.908                         : temperature of 2nd grid point
...
```

-s = small inventory
-i = read an inventory and decode data
-text = write the decoded data in text format
-o = name of output file

Now that you've found the 349*277 temperature values, you may wonder which one corresponds to your home town.

```
-sh-3.00$ wgrib rr-fixed.grb -s | grep ':ELON:' | \
        wgrib rr-fixed.grb -s -text -i -o lon.txt
19:2384176:d=79110800:ELON:sfc:anl:NAve=0
-sh-3.00$ cat lon.txt
349 277                         : nx ny
214.499                         : longitude of 1st grid point in degrees
214.687
...

-sh-3.00$ wgrib rr-fixed.grb -s | grep ':NLAT:' | \
        wgrib rr-fixed.grb -s -text -i -o lat.txt
20:2577616:d=79110800:NLAT:sfc:anl:NAve=0
-sh-3.00$ cat lat.txt
349 277                 : nx ny
0.999                   : latitude in degrees of the 1st grid point
1.105
..
```

At this point you have three files, t2m.txt, lat.txt and lon.txt. More work needs to be done before you start doing useful work but you've decoded the grib file. Note, I saved the data as text files. Saving the data as binary (-bin) is preferred because of the greater precision and faster processing.

Grib-1 to Comma Separated Values

Making CSV files can be a problem for some people. If you can make the raw numbers, latitude and longitudes (see previous section), you could write a program to make the CSV file. Depending on your skill set, it could be simple or very difficult. You could also write a GrADS script to read the grib files and write a CSV file. However, people who can't use the first method will have problems writing the GrADS script.

Grib-2 to Raw Numbers

The situation with grib-2 is better than with grib-1. You still can do the text/binary dump of the file.

```
-sh-3.00$ wgrib2  narr.t09z.awip32.merged.grb2  |  grep ":TMP:2 m above ground" | \
     wgrib2  narr.t09z.awip32.merged.grb2 -i -text t2m.txt
258:17810770:d=2009102809:TMP:2 m above ground:anl:
```

The grib-2 extraction is very similar to the grib-1 version. The differences are

"2 m above gnd" is now "2 m above ground"
"-text -o t2m.txt" is now "-text t2m.txt"
-s is now the default when no other inventory options are present

wgrib2 has the new feature that

```
wgrib2 FILE | egrep RegExpression | wgrib2 FILE -i -text out.txt
```

can be written as

```
wgrib2 FILE -match RegExpression -text out.txt
```

It's less typing, Windows-friendly and runs faster.

Grib-2 to Comma Separated Values (CSV)

wgrib2 can create CSV files with the contributions of Karl Pfeiffer (Naval Postgraduate School), who contributed the georeferencing package, and others who have updated the module and contributed bug fixes.

CSV for a single or a few points

The -lon option will print out the value of the grid point closest to the specified longitude latitude point.

```
-sh-3.00$ wgrib2 pgblnl.gdas.2007010100.grb2 -s -lon 200 -30
1:4:d=2007010100:HGT:1 mb:anl::lon=200.000000,lat=-30.000000,val=48163.2
2:16552:d=2007010100:TMP:1 mb:anl::lon=200.000000,lat=-30.000000,val=269.4
3:22064:d=2007010100:RH:1 mb:anl::lon=200.000000,lat=-30.000000,val=0.0003
4:23631:d=2007010100:SPFH:1 mb:anl::lon=200.000000,lat=-30.000000,val=2.87e-06
5:30587:d=2007010100:VVEL:1 mb:anl::lon=200.000000,lat=-30.000000,val=-0.0002418
6.1:46414:d=2007010100:UGRD:1 mb:anl::lon=200.000000,lat=-30.000000,val=-62.6
6.2:46414:d=2007010100:VGRD:1 mb:anl::lon=200.000000,lat=-30.000000,val=-6.7
```

...

Makeing a CSV file with a little help from sed.

```
-sh-3.00$ wgrib2 pgblnl.gdas.2007010100.grb2 -s -lon 200 -30  | cut -f3- -d: | sed 's/:/,/' | \
        sed 's/:/,"/' | sed 's/::/",/'
d=2007010100,HGT,"1 mb:anl",lon=200.000000,lat=-30.000000,val=48163.2
d=2007010100,TMP,"1 mb:anl",lon=200.000000,lat=-30.000000,val=269.4
d=2007010100,RH,"1 mb:anl",lon=200.000000,lat=-30.000000,val=0.0003
d=2007010100,SPFH,"1 mb:anl",lon=200.000000,lat=-30.000000,val=2.87e-06
...
```

CSV for an Region

The -csv option by Niklas Sondell (Storm Weather Center, Norway) can convert a grib2 file into a CSV file by

```
-sh-3.00$ wgrib2 pgblnl.gdas.2007010100.grb2 -csv monster.csv
...
-sh-3.00$ cat monster.csv
"2007-01-01 00:00:00","2007-01-01 00:00:00","HGT","1 mb",0,-90,50483.7
"2007-01-01 00:00:00","2007-01-01 00:00:00","HGT","1 mb",2.5,-90,50483.7
"2007-01-01 00:00:00","2007-01-01 00:00:00","HGT","1 mb",5,-90,50483.7
...
field 1 = reference time, start of forecast or analysis time
field 2 = reference time, verification time or analysis time
field 3 = variable
field 4 = level
field 5 = longitude
field 6 = latitude
field 7 = value
```

Now monster.csv will contain one line for each grid point value. Considering the 0.5 degree grid has 65K points and the GFS file has 522 fields, you get 34 million lines for one analysis time. Suppose we only wanted the 500 mb height and temperature.

```
-sh-3.00$ wgrib2 pgblnl.gdas.2007010100.grb2 -match "(HGT|TMP)" -match "500 mb" -csv  HGT_TMP.csv
185:1979072:d=2007010100:HGT:500 mb:anl:
186:1995076:d=2007010100:TMP:500 mb:anl:
```

The regular expression “(HGT|TMP)” means match HGT or TMP. Now HGT_TMP.csv is now only 21K bytes is size. Suppose I am interested in the area around DC.

```
-sh-3.00$ wgrib2 pgblnl.gdas.2007010100.grb2 -match "(HGT|TMP)" \
    -match "500 mb" -undefine out-box -80:-60 35:45 -csv dc.csv
```

Now "-undefine out-box -80:-60 35:45" sets all the grid points outside the box of 80W-60W and 35N-45N to undefined. The -csv option doesn't print points with undefined values, so the csv file is now only 90 lines.

Converting between grib-1 and grib-2

You may want to convert from grib-2 to grib-1 because you have some grib-1 programs. You may want to convert from grib-1 to grib-2 because you want a smaller file or you have a grib-2 program. Cnvgrib (NCEP Central Operations) will convert between grib-1 and grib-2 for many of NCEP produced files.

grib1->2: cnvgrib -g12 -p32 grib1.in grib2.out
grib2->1: cnvgrib -g21 grib2.in grib1.out

Grib-2 compression: JPEG2000 Packing

Grib-2 allows several types of packing/compression. One of best compression schemes, JPEG2000, is very slow to uncompress. If you receive a jpeg2000 grib-2 file, you may want to convert it into complex packing because it is faster to read at a reasonable increase in size.

See http://www.cpc.ncep.noaa.gov/products/wesley/wgrib2/small_fast_databases.html

Using the sample files from NARR and CFSR,

		Complex	JPEG2000	
NARR:	read	2.4 sec	23 sec	(9.7x slower)
NARR:	size	27 MB	30 MB	(11% bigger)
CFSR:	read	0.24 sec	3.3 sec	(13.7x slower)
CFSR:	size	5.4 MB	4.4 MB	(20% smaller)

With the sample NARR and CFSR files, reading a jpeg2000 compressed file is 10x slower. JPEG2000 does a better job of compression with the CFSR files but a poorer job with the NARR because the jpeg packing doesn't handle missing values as well as complex packing. Unless disk space is very tight, using jpeg2000 compression is not recommended. You can determine the compression by

-sh-3.00$ wgrib2 narr.t09z.awip32.merged.grb2 -packing -d 1

1:0:packing=grid point data - complex packing and spatial differencing

You can change the packing by,

```
wgrib2 in.grb -set_grib_type X -grib_out out.grb
X=simple, complex1, complex2, complex3, jpeg
Note: this command converts submessages
into messages.
```