NCEP's UNIFIED POST PROCESSOR (UPP)

Presented by Kate Fossell

January 24, 2018

Developmental Testbed Center

Outline

- Overview
- Components and Functions
- Sample fields generated
- Installing UPP
- Running *unipost*
  - Required input files
  - Controlling output generation
    - grib1 and grib2 formats
- Running *copygb*
  - Specifying target grid
- Visualization

Developmental Testbed Center

UPP Overview

- UPP is one of the many post processing packages available
- NCEP Developed & Supported Operationally
  - GFS, GEFS, NAM, SREF, RAPR, HRRR, HWRF, etc.
- NCAR Supports community code for WRF Post Processing

Why would you want to use UPP?

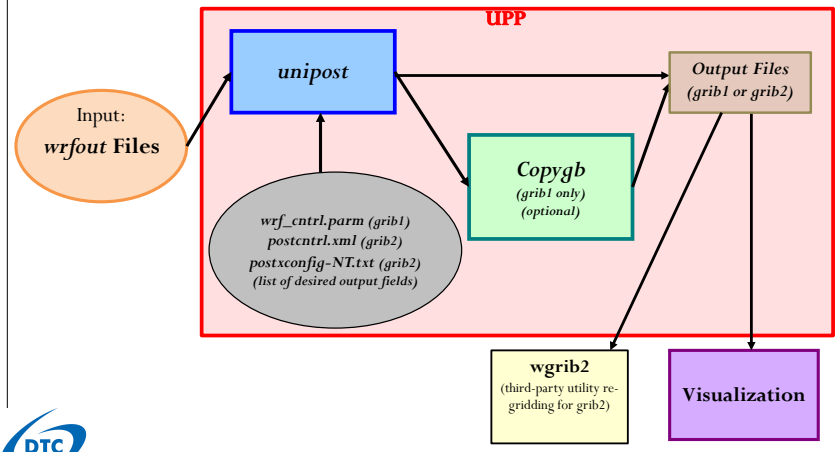
- Generates output in GRIB1 and GRIB2 format.
- Produces hundreds of products like those used operationally on same operational grids.
- Enables product generation on any output grid.

  E.g. MET: Regrid model data to match a observational grid for verification
- Processes model output from the WRF-ARW dynamical core
- Produces requested diagnostics and fields, but does not plot or visualize data.
- MPI parallelized code
- UPP Supports WRFV3.9 new vertical hybrid coordinate

Developmental Testbed Center

Components of the UPP

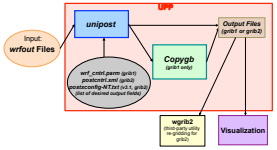
UPP has two components: 1) unipost 2) copygb (grib1 only)



Developmental Testbed Center

Unipost Functions & Features

- Performs **vertical** interpolation from model levels/surfaces onto isobaric, height, and other levels/surfaces
- Computes **diagnostic** fields
- Destaggers wind onto mass points
- An MPI-parallel code



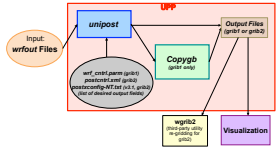
Copygb (grib1 format only) Functions & Features

- Performs ***optional*** **horizontal** interpolation to a defined output grid – grib1 format only

i.e. Creates an output grid different than the model integration domain

- e.g. Convert to operational grid: 221 (NAM, RAP, SREF)
- e.g. Lambert ⟶ Lat-Lon
- e.g. convert to observational grid

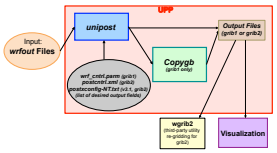
wgrib2 Third-party utility for re-gridding grib2



Ingesting WRF model output

Input: ***wrfout* Files**

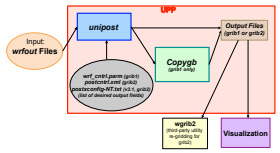
- The unipost ingests WRF model output in netCDF using the WRF I/O package.
  - One time per output file is best w/ sample UPP run scripts (frames_per_outfile=1 in WRF model namelist).
  - By default UPP tries to read a set list of fields in wrfout files.
    - Should contain necessary fields for basic diagnostics
    - Could impact UPP if you change registry or wrfout fields



Fields generated by the UPP

Output Files (Grib1 or Grib2)

- **The UPP currently outputs hundreds of possible fields.**
  - Complete list in the Post Processing Utilities Chapter of the user guide
  - Fields are output in Grib1 or Grib2 format
- **Sample fields generated by UPP:**
  1) T, Z, humidity, wind, cloud water, cloud ice, rain, and snow on isobaric levels
  2) SLP + shelter level T, humidity, and wind fields
  3) Precipitation-related fields
  4) PBL-related fields
  5) Diagnostic products (i.e. RH, radar reflectivity, CAPE)
  6) Radiative/Surface fluxes
  7) Cloud related fields
  8) Aviation products
  9) Synthetic satellite products



Outputting fields on different vertical coordinates

- *unipost* outputs on several vertical coordinates:
  - Native model levels
  - 47 isobaric levels: Default: 2, 5, 7, 10, 20, 30, 50, 70, then every 25 hPa from 75-1000 hPa.
  - 15 flight/wind energy levels: 30, 50, 80, 100, ..., 2743, 3658, 4572, 6000 m (above ground or above MSL)
  - 6 PBL layers: each averaged over a 30 hPa deep layer
  - 2 AGL radar levels: 1000 & 4000
- Except for AGL radar and isobaric levels, vertical levels are listed from the ground surface up in *wrf_cntrl.parm (postcntrl.xml)*.

UPP download and compile

UPP Dependencies & Required Libraries

- UPP build relies on the existence of a built WRF source directory. Uses WRF i/o routines.
- UPPV2.1+ depends on WRFV3.5 or later releases.
- UPPV3.0+ depends on WRFV3.7+ for Ferrier Physics
- Libraries required:
  - netCDF
  - JasPer
  - PNG
  - Zlib
  - WRF i/o libs

Downloading the UPP source code

- The UPP source code can be obtained from:
  http://www.dtcenter.org/upp/users/downloads/index.php
  - The latest version available is: UPPV3.2.tar.gz
- Unpack the downloaded file:
  tar –zxvf UPPV3.2.*tar.gz*
- *cd* to newly created UPPV3.2/ directory
  - Important Directories:
    - **scripts/**: sample scripts for running UPP and generating graphics
    - **parm/**: contains the files used to request output fields when running the unipost (i.e. wrf_cntrl.parm, postcntrl.xml)
    - **clean, configure, compile**: scripts used in the build process

Compile source codes

- The build mechanism follows the WRF model build paradigm:

  ./configure : respond to screen prompts about target computing platform

- *./compile >& compile_upp.log*

Developmental Testbed Center

Compile source codes (cont.)

- If compilation is successful, these three executables will be present in bin/ :

  copygb.exe
  ndate.exe
  unipost.exe

- Currently have build options established for IBM and Linux (PGI/Intel/Gnu compilers)

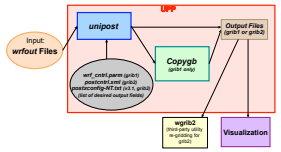
- The arch/configure.defaults file has compilation options for various platforms, and is where new computers or compilers might be added.

Developmental Testbed Center

Running unipost and copygb

Developmental Testbed Center

Running UPP



* Use sample scripts as a template or guide to run UPP *

Run Script: ./run_unipost >& script_output.log &

- run_unipost is a korn shell script that runs UPP end to end: unipost + copygb (if desired)
- User edits model core, paths, date, time, command syntax (serial vs. parallel) in script.
- Links all required files, loops over times/files and processes fields requested fields from wrf_cntrl.parm or postcntrl.xml, runs copygb if requested.
- Unipost.exe output/error messages is redirected to log files, e.g. unipost_d01.00.out. Hint: Look in these files for information about errors.

Unipost Running unipost.exe

** Requires 2 input files to run + a few extra data files **

1) *itag*: 4-5 line text file that details WRF model output to process. Also referred to as the namelist.

```
wrfout_d01_2010-06-27_00:00:00   <-- WRF history filename
netcdf                           <-- WRF output format (netcdf/binary)
grib2                            <-- extra line only if writing GRIB2
2010-06-27_00:00:00              <-- validation time
NCAR                             <-- model name: "NCAR" for WRF-ARW
```

2) *wrf_cntrl.parm (grib1)* : control file specifying fields/levels to output in GRIB1 (*text file*)

- or -

postxconfig-NT.txt (grib2) : control file specifying fields/levels to output in GRIB2 (*text file generated from xml postcntrl.xml*)

wrf_cntrl.parm postcntrl.xml postxconfig-NT.txt

3) *extra data files: e.g. eta_micro_lookup.dat, coefficient files for satellite, etc.*

*** In the sample scripts/run_unipost* scripts, these files are automatically generated (itag) or linked (wrf_cntrl.parm & eta_micro_lookup.dat, etc).

unipost control file:

wrf_cntrl.parm postcntrl.xml postxconfig-NT.txt

* file that lists desired fields and levels that unipost reads directly

Grib1 Format:

wrf_cntrl.parm (text file)

Grib2 Format:

v3.1+: postxconfig-NT.txt (text file, but not formatted for easy read/write) generated from postcntrl.xml & post_avblflds.xml (xml files)

unipost control file for **grib1** :

wrf_cntrl.parm (Grib1)

- User controlled and modified text file that lists fields and level(s) of fields to output; each product described by 2 lines (Examples next slides)
- The included parm/*wrf_cntrl.parm* file has entries for most output fields. ** Use this as template! ** (Text file fixed width format)
- The users' guide "Fields produced by *unipost*" (Table 1) more fully explains the character string abbreviations used in the wrf_cntrl.parm file.

unipost control file: *wrf_cntrl.parm*

wrf_cntrl.parm (Grib1)

- Each field described by 2 lines: product description and levels

```
(PRESS ON MDL SFCS  ) SCAL=(6.0)
L=(11000 00000 00000 00000 00000 00000 00000...
(HEIGHT ON MDL SFCS ) SCAL=(6.0)
L=(11000 00000 00000 00000 00000 00000 00000...
```

GRIB packing precision**

Levels to output: Each column represents a single model/isobaric level:
"1" (or "2" - special case) = output, "0" = no output

Product description – unipost code keys on these character strings.

** larger values → more precision, but larger GRIB files.

Examples

wrf_cntrl.parm (Grib1)

➢ Output T every 50 hPa from 50 hPa to 1000 hPa:

```
(TEMP ON PRESS SFCS  ) SCAL=( 4.0)
L=(00000  01001  01…)
      2  5 7 10 20    30 50 70 75 100    125 150
```

*** Isobaric levels increase from left to right:
2, 5, 7, 10, 20, 30, 50, 70, then every 25 hPa from 75-1000 hPa.
(Default/standard – can manually change code for different pressure levels)

Isobaric levels every 50 hPa:

```
L=(00000 01001 01010 10101 01010 10101 01010 10101 01010 10000 00000 00000 00000 00000)
```

Isobaric levels every 25 hPa:

```
L=(00000 01011 11111 11111 11111 11111 11111 11111 11111 10000 00000 00000 00000 00000)
```

Developmental Testbed Center

Examples

wrf_cntrl.parm (Grib1)

➢ Output instantaneous surface sensible heat flux:

```
(INST SFC SENHEAT FX) SCAL=( 4.0)
L=(10000 00000 00000 00000 00000 00000 00000 00000 00000 00000…)
```

➢ Output the U-wind component at the 5 lowest model levels:

```
(U WIND ON MDL SFCS  ) SCAL=( 4.0)
L=(11111 00000 00000 00000 00000 00000 00000 00000 00000 00000…)
```

➢ Output U-wind component at 30, 50, and 80 m AGL:

```
(U WIND AT FD HEIGHT) SCAL=( 4.0)
L=(22200 00000 00000 00000 00000 00000 00000 00000 00000 00000…)
```

For the flight/wind energy level fields:
- "2" requests AGL.
- "1" requests above mean sea level.

Developmental Testbed Center

unipost control file for **grib2** :

postcntrl.xml postxconfig-NT.txt (Grib2)

➢ User controlled xml file that lists desired fields to be output by UPP.

➢ The included parm/*postcntrl.xml* file has examples of the template to follow.

➢ The included parm/*post_avblcntrl.xml* file has a listing of all available field names, shortnames, parameter info.

➢ New V3.1+: the included parm/*postxconfig-NT.txt* file is read by unipost and includes fields from default postcntrl.xml and can be used directly

Developmental Testbed Center

unipost control file: *post_avblflds.xml*

postcntrl.xml postxconfig-NT.txt (Grib2)

- **Lists all available fields and details for grib2 tables/output**
- **Does not need to be modified unless adding new variables or modifying from default.**

```
<param>
   <post_avblfldidx>2</post_avblfldidx>
   <shortname>TMP_ON_HYBRID_LVL</shortname>
   <pname>TMP</pname>
   <fixed_sfc1_type>hybrid_lvl</fixed_sfc1_type>
   <scale>4.0</scale>
 </param>

 <param>
   <post_avblfldidx>70</post_avblfldidx>
   <shortname>DPT_ON_SPEC_PRES_ABOVE_GRND</shortname>
   <pname>DPT</pname>
   <fixed_sfc1_type>spec_pres_above_grnd</fixed_sfc1_type>
   <scale>3.0</scale>
 </param>
```

- **UPP ID**
- **Character name describing the product/field**
- **Field type abbreviation used by grib2 libraries**
- **Vertical coordinate type**
- **Grib precision packing**

Developmental Testbed Center

unipost control file: *postcntrl.xml*

postcntrl.xml
postxconfig-NT.txt
(Grib2)

- User modified xml file to list all desired grib2 fields to be output by UPP
- Use provided file as a guide

```
<param>
    <shortname> TMP_ON_SPEC_HGT_LVL_ABOVE_GRND_2m</shortname>
    <scale>4.0</scale>
</param>

<param>
    <shortname> UGRD_ON_ISOBARIC_SFC</shortname>
    <scale> 4.0 </scale>
    <level> 50000. 70000. 85000. 92500. 100000. </level>
</param>
```

*** Formatting important, use provided file as a guide**

- Character name describing the product/field
- Grib precision packing
- Vertical coordinate levels desired

Developmental Testbed Center

New for V3.1+ when outputting grib2 format:

postcntrl.xml
postxconfig-NT.txt
(Grib2)

* unipost requires to read a text file called: **postxconfig-NT.txt**
(operationally motivated to speed up unipost)

Additional pre-processing step required to convert xml fie to a flat file:

ONLY IF : user wants to add/modify fields or levels of fields –
must modify postcntrl.xml and then convert xml to text file:

1) >> cd UPPV3.1/parms
2) *edit the postcntrl.xml to add/remove fields/levels*
3) >> make

- make calls a perl program that does that conversion based on postcntrl.xml and post_avblflds.xml
- Detailed instructions in the UPP Users Guide
- output is " **postxconfig-NT.txt** "

ELSE: Can use default postxconfig-NT.txt with default fields and levels
(new steps only necessary if requesting new fields to output)

Developmental Testbed Center

Regridding UPP grib output (optional):

- **grib1** : *Copygb* *(grib1 format only)* (included in UPP package)
- **grib2** : *wgrib2* (third-party; examples in Users Guide and on online tutorial)

Developmental Testbed Center

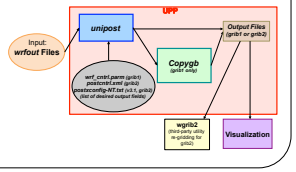
Copygb
(grib1 format only)

target grid definition

- The generic command to run copygb and horizontally interpolate onto a new grid is:

```
copygb.exe –xg"${grid}"  in.grb   out.grb
```

- Two options on how to specify the target $grid:
  1. Pre-defined NCEP standard grid number
  2. User-defined grid definition



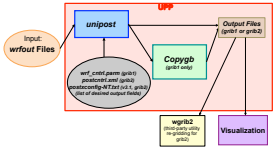
Developmental Testbed Center

Run *copygb* – Option 1

Copygb *(grib1 format only)*

- Interpolate to a pre-defined NCEP standard grid (restrictive but simple)
  - For example, to interpolate onto NCEP grid 212:
    copygb.exe –xg212 in.grb out.grb

Descriptions of NCEP grids are available online:
http://www.nco.ncep.noaa.gov/pmb/docs/on388/tableb.html



Run *copygb* – Option 2a

Copygb *(grib1 format only)*

- Create a user-defined ***Lambert Conformal*** grid by specifying a full set of grid parameters (complicated but flexible).

map type (3=LC) | # of points | 255 indicates user-defined grid | SW corner (millidegrees) | Proj cent lon (millidegrees)

```
copygb.exe –xg"255 3 NX NY STARTLAT STARTLON 8 CENLON DX DY
    0 64 TRUELAT1 TRUELAT2 " in.grb out.grb
```

Proj true latitudes (millidegrees) | horizontal spacing (meters)

```
copygb –xg"255 3 185 129 12190 -133459 8 -95000 40635 40635
    0 64 25000 25000" in.grb out.grb
```


Run *copygb* – Option 2b

Copygb *(grib1 format only)*

- Create a user-defined ***Polar Stereographic*** *grid* by specifying a full set of grid parameters (complicated but flexible).

map type (5=STR)

```
copygb.exe –xg"255 5 NX NY STARTLAT STARTLON 8 CENLON DX DY
    0 64" in.grb out.grb
```

Center flag (0=NH ; 128=SH)

```
copygb –xg"255 5 580 548 10000 -128000 8 -105000 15000 15000
    0 64" in.grb out.grb
```


Run *copygb* – Option 2c

Copygb *(grib1 format only)*

- Create a user-defined ***Latitude-Longitude*** *grid* by specifying a full set of grid parameters (complicated but flexible).

map type (0=LTLN)

```
copygb.exe –xg"255 0 NX NY STARTLAT STARTLON 136 ENDLAT ENDLON
    DLAT DLON 64" in.grb out.grb
```

grid spacing (millidegrees) | NE lat (millidegrees) | NE lon (millidegrees)

```
copygb –xg"255 0 401 401 10000 -130000 136 50000 -90000
    100 100 64" in.grb out.grb
```

Visualization of UPP output

- Gempak - sample run script with UPP package
- GrADS - sample run script with UPP package
- NCL
- Python
- Ncview - (after conversion to netcdf)
- Matlab
- IDL
- Many Others

Visualization: GEMPAK

- The GEMPAK utility "nagrib" reads GRIB files from any non-staggered grid and generates GEMPAK-binary files that are readable by GEMPAK plotting programs
- GEMPAK can plot horizontal maps, vertical cross-sections, meteograms, and sounding profiles.
- Package download and user guide are available online:

  http://www.unidata.ucar.edu/software/gempak/index.html
- A sample script named *run_unipostandgempak* is included in scripts/ that can be used to run *unipost*, *copygb*, and then plot various fields using GEMPAK.
- Further details on this script and using GEMPAK are available in the user's guide.

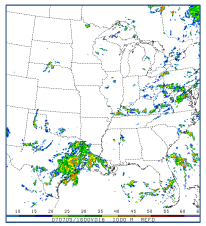
Visualization: GrADS

- GrADS also has utilities to read GRIB1 and GRIB2 files on any non-staggered grids and generate GrADS "control" files.
- The utilities grib2ctl (grib1), g2ctl (grib2), and gribmap are available via:
  http://www.cpc.ncep.noaa.gov/products/wesley/grib2ctl.html
  http://www.cpc.ncep.noaa.gov/products/wesley/g2ctl.html
- Package download and user guide for GrADS are available online:
  http://cola.gmu.edu/grads/
- A sample script named *run_unipostandgrads* is included in scripts/ that can be used to run *unipost*, *copygb*, and then plot various fields using GrADS.

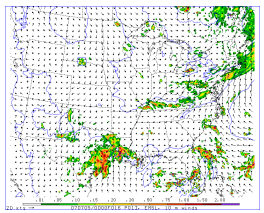
Example of Forecasts Plotted in:

Gempak

Derived Radar Reflectivity

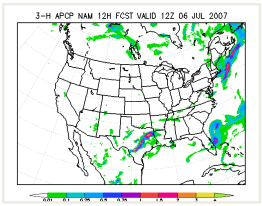


Observed Reflectivity

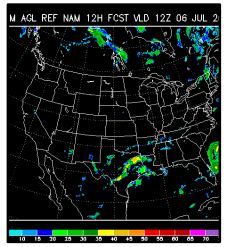


GrADS

Accumulated Precipitation



Observed Reflectivity



Future plans

- Continue adding new products to the released UPP code as they are developed, and expand code portability.
- Improved documentation for users to add custom fields

Helpful Links:

UPP Users Website:
http://www.dtcenter.org/upp/users/index.php

New UPP Online Tutorial
https://dtcenter.org/upp/users/support/online_tutorial/UPPv3.2/index.php

UPP Users' Guide available at:
http://www.dtcenter.org/upp/users/docs/user_guide/V3/upp_users_guide.pdf

UPP FAQ's Page:
http://www.dtcenter.org/upp/users/overview/upp_faqs.php

UPP Questions Contact: upp-help@ucar.edu

Questions???