NCMRWF

UMRider

Version 1.0.2

https://github.com/NCMRWF/UMRider

Arulalan . T
Project Scientist - C
26-Apr-2016

[Abstract: This document contains the UMRider usage details. UMRider is a utility to convert NCMRWF-UK MetOffice Unified Model outputs fileformat from fieldfiles to grib2 through parallel python. Through STASH code and cf_standard_name of variable we are extracting and converting to WMO-NCEP grib2 standard by setting correct GRIB2 Param Code (Discipline, Category, Number, type Of First Fixed Surface) and followed by creating Grads Control File by using g2ctl.pl scripts. Also created the NCMRWF Grib2 Local Table (which I created to produce grib2 files for 9 variables). Acknowledgement: I thank Mr. M.N.Raghavendra Sreevathsa , Scientist-E who made the serial python code initially in simple way to convert pp0 to grib2.]

ncum post operational v1.0.2

Source code : https://github.com/NCMRWF/UMRider

Releases : https://github.com/NCMRWF/UMRider/releases/tag/v1.0.0 on 28-Mar-2016

UMRider has 3 modules in **g2utils** and 6 python scripts in **g2scripts** to create analysis/forecast grib2 files along with corresponding bsub bash scripts.

um2grib2.py module has capability to convert pp fieldsfiles format to grib2 through iris in embarrassing parallel mode.

UMRider supported analysis files (short forecast) are

1. 'qwqg00.pp0'
2. 'umglca_pb???'
3. 'umglca_pd???'
4. 'umglca_pe???'
5. 'umglca_pf???'
6. 'umglca_pi???'

UMRider supported forecast files (long forecast) are

1. 'umglaa_pb???'
2. 'umglaa_pd???'
3. 'umglaa_pe???'
4. 'umglca_pf???'
5. 'umglca_pi???'

Replace ??? with corresponding hours like 024, 048, …, 240.

024 file contains either twenty four 1-hourly data or eight 3-hourly data and similarly for other files.

UMRider supports to create either 3-hourly grib2 files or 6-hourly (either 6^{th} hour instantaneous or average/ accumulation of two 3-hourly data/six 1-hourly data) or 24-hourly (either 24^{th} hour instantaneous or average/ accumulation of eight 3-hourly data/twenty four 1-hourly data). User will be able to control what they want in out grib2 file either 3-hourly data or 6-hourly data or 24-hourly data.

For example UMRider has configured for 6-hourly grib2 files, then

- It creates 4 analysis 6 hourly files (during 00, 06, 12, 18 UTC cycles)
- It creates 40 forecast 6 hourly files (000, 006, 012, ..., 234, 240 during 00 and 12 UTC cycles)
- Finally it creates ctl, idx files for all 44 grib2 files by using g2ctl.pl
- Also all 44 grib2 files variables are in same order (defined by user)

In this version v1.0.2 user will be able to control the following 30 arguments by exporting **setup.cfg** file into their environment variable called **UMRIDER_SETUP**

1. inPath
2. outPath
3. tmpPath
4. anl_step_hour
5. anl_aavars_reference_time
6. anl_aavars_time_bounds
7. fcst_step_hour
8. start_long_fcst_hour
9. end_long_fcst_hour_at_00z
10. end_long_fcst_hour_at_12z
11. latitude
12. longitude
13. targetGridResolution
14. pressureLevels
15. soilFirstSecondFixedSurfaceUnit
16. startdate
17. enddate
18. loadg2utils
19. overwriteFiles
20. anlOutGrib2FilesNameStructure
21. fcstOutGrib2FilesNameStructure
22. createGrib2CtlIdxFiles

23. convertGrib2FilestoGrib1Files
24. grib1FilesNameSuffix
25. removeGrib2FilesAfterGrib1FilesCreated
26. createGrib1CtlIdxFiles
27. debug
28. setGrib2TableParameters
29. wgrib2Arguments
30. callBackScript

Also in this version v1.0.2 user will be able to control the variables by exporting vars.cfg file into their environment variable called **UMRIDER_VARS** . User defined variables and ordered will be retained in the out grib2 files (but pressure level variables comes first and followed by non-pressure level variables)

User will be able to control would they like to get 3-hourly or 6-hourly or 24-hourly forecast files.

This **v1.0.2** supports for following productions

1. NCUM Global Post Production
2. NCUM Indian Region Post Production
3. NCUM Global OSF (Ocean State Forecast) Model Input Production
4. NCUM Global HYCOM Model Input Production
5. NCUM Global VSDB Input Production

NCUM Global Post Production				
Inpath :	**/gpfs3/home/umfcst/NCUM/fcst/**			
Outpath :	**/gpfs3/home/umfcst/NCUM/post/**		Resolution : **0.25X0.25 degree**	
No of Variables :	**63** (63 – analysis & 61 – forecast)		Out data frequency : **6-hourly**	
Infile Type	UTC	No of grib2 outfiles	No of parallel processors used	Time taken to complete (app.)
Analysis File	00	1	6	4 minutes
Analysis File	06	1	4	3 minutes
Analysis File	12	1	4	3 minutes
Analysis File	18	1	4	3 minutes
Forecast Files	00	40	40	14 minutes
Forecast Files	12	20	20	11 minutes

Table 1 : Details of NCUM Global Post Production which has been implemented at Bhaskara, NCMRWF

NCUM Indian Region Post Production				
Inpath :	**/gpfs3/home/umfcst/NCUM/fcst/**			
Outpath :	Suggest outpath		Resolution : **0.25X0.25 degree**	
No of Variables :	**63** (63 – analysis & 61 – forecast)		Out data frequency : **6-hourly**	
Infile Type	UTC	No of grib2 outfiles	No of parallel processors used	Time taken to complete (app.)
Analysis File	00	1	6	3 minutes
Analysis File	06	1	4	2.5 minutes
Analysis File	12	1	4	2.5 minutes
Analysis File	18	1	4	2.5 minutes
Forecast Files	00	40	40	11 minutes
Forecast Files	12	20	20	8 minutes

Table 2 : Details of NCUM Indian Region Post Production which will be implemented at Bhaskara, NCMRWF. All variables are same as Table1 except the latitude (20S t0 60N), longitude (20E to 140E) extracted from Global output.

NCUM Global OSF Model Input Production				
Inpath :	**/gpfs3/home/umfcst/NCUM/fcst/**			
Outpath :	Suggest outpath		Resolution : **0.25X0.25 degree**	
No of Variables :	**14**		Out data frequency : **6-hourly**	
Infile Type	UTC	No of grib2 outfiles	No of parallel processors used	Time taken to complete (app.)
Analysis File	00	1	6	30 seconds
Analysis File	06	1	4	30 seconds
Analysis File	12	1	4	30 seconds
Analysis File	18	1	4	30 seconds
Forecast Files	00	40	40	2 minutes
Forecast Files	12	DO NOT REQUIRED AT THE MOMENT		

Table 3 : Details of NCUM Global OSF Model Input Production which will be implemented at Bhaskara, NCMRWF.

NCUM Global HYCOM Model Input Production				
Inpath :	**/gpfs3/home/umfcst/NCUM/fcst/**			
Outpath :	Suggest outpath		Resolution : **0.25X0.25 degree**	
No of Variables :	**13**		Out data frequency : **3-hourly**	
Infile Type	UTC	No of grib2 outfiles	No of parallel processors used	Time taken to complete (app.)
Analysis File	00	2	6	30 seconds
Analysis File	06	2	4	30 seconds
Analysis File	12	2	4	30 seconds
Analysis File	18	2	4	30 seconds
Forecast Files	00	56	56	2 minutes
Forecast Files	12	DO NOT REQUIRED AT THE MOMENT		

Table 4 : Details of NCUM Global HYCOME Model Input Production which will be implemented at Bhaskara, NCMRWF.

NCUM Global VSDB Input Production				
Inpath :	**/gpfs3/home/umfcst/NCUM/fcst/**			
Outpath :	Suggest outpath		Resolution : **0.25X0.25 degree**	
No of Variables :	**6**		Out data frequency : **24-hourly**	
Infile Type	UTC	No of grib1 outfiles	No of parallel processors used	Time taken to complete (app.)
Analysis File	00	1	2	30 seconds
Analysis File	06	DO NOT REQUIRED AT THE MOMENT		
Analysis File	12	DO NOT REQUIRED AT THE MOMENT		
Analysis File	18	DO NOT REQUIRED AT THE MOMENT		
Forecast Files	00	7	8	2 minutes
Forecast Files	12	DO NOT REQUIRED AT THE MOMENT		

Table 5 : Details of NCUM Global VSDB Input Production which will be implemented at Bhaskara, NCMRWF.

UMRider Setup Configure Options

setup configure file: Used to setup indata path, outdata path, temporary path to run the um2grb2 python parallel scripts which will create analysis and forecast files.

Author : Arulalan <arulalan@ncmrwf.gov.in>

Updated : 09-Feb-2016

BEGIN OF UMRIDER SETUP CONFIGURE FOR um2grb2 SCRIPTS

1. *inPath* model pp filedsfiles path

   inPath = /gpfs3/home/umfcst/NCUM/fcst/

2. *outPath* model grib2 files path

   outPath = /gpfs3/home/umfcst/NCUM/post/

3. *tmpPath* working directory (used to create temporary log files)

   tmpPath = /gpfs4/home/umtid/tmp/um2grb2/logs/

4. *anl_step_hour* analysis step/interval hours. By default it takes 6 hour which mean um2grb2 produce 6 hourly instantaneous and/or 6 hourly average and/or 6 hourly accumulation values analysis files. If user specfied as 3 then it will extract only 3 houlry instantaneous fields. By default model produced 3 hourly average/accumulation.

   anl_step_hour = 6

5. *anl_aavars_reference_time* takes either **'analysis'** or '**shortforecast'**. When some variables are taken from previous cycle short-forecast (average/accumulation) vars, the reference time need to be set as either current **'analysis'** reference cycle (utc) or previous cycle's **'shortforecast'** reference time. '**shortforecast'** gives exactly based on which utc that variable has processed, whereas **'analysis'** shift reference time utc as actual analysis utc time. Note : This option applicable only to average/accumulation vars in analysis grib2 files. By default it takes 'shortforecast' argument.

   anl_aavars_reference_time = analysis

6. *anl_aavars_time_bounds* takes either **'True'** or '**False'**. By default, True keeps the analysis time bounds, reference time bounds and False removes it (so that it become instantaneous instead of average/accumulation vars). False will be applicable only if *anl_aavars_reference_time* arg passed as **'analysis'**. Note : This option applicable only to average/accumulation vars in analysis grib2 files.

   anl_aavars_time_bounds = True

7. *fcst_step_hour* long forecast step/interval hours . By default it takes 6 hour which mean um2grb2 produce 6 hourly instantaneous and/or 6 hourly average and/or 6 hourly accumulation values. If user specfied as 3 or 24 then it will extract only 3 or 24 houlry

instantaneous fields and for calculate average/accumulation for 24 hourly. By default model produced 3 hourly average/accumulation.# Note: the average and accumulation supports only for either 3 or 6 or 24 hours!!!

fcst_step_hour = 6

8. long forecast start hour. By default it takes 6 hour which mean um2grb2 produce grib2 files from 06-th hour forecasts. If user wants from different hours, then they can specify it ! It should be multiples of '*fcst_step_hour*' (see above option)!

   start_long_fcst_hour = 6

9. maximum long forecast hours at 00utc cycle (*max_long_fcst_hours_at_00z*) produced by NCUM model for 10days forecast 240hour (by default 240 hours)

   end_long_fcst_hour_at_00z = 240

10. maximum long forecast hours at 12utc cycle (*max_long_fcst_hours_at_12z*) produced by NCUM model for 5days forecast 120hour (by default 120 hours)

    end_long_fcst_hour_at_12z = 120

11. *latitude* is required latitude which user wants to extract from the model global data. By default it takes *None* (i.e. extract model global latitudes). User can specify their required *latitude* in tuple. eg 1: *latitude* = (-30, 30) will extract only latitudes from 30S to 30N. eg 2: *latitude* = (90, -90) will extract latitude from 90N to 90S (reverse latitude though UM model produce it from 90S to 90N).

    latitude = None

12. *longitude* is required longitude which user wants to extract from the model global data. By default it takes *None* (i.e. extract model global longitudes) User can specify their required longitude in tuple. For eg : *longitude* = (60, 100) will extract only longitude from 60E to 100E. Note : Model requires longitude should specified based on (0 to 360), and not by (-180 to 180.) In future I may fix it, if user wish to specify longitude by within range of (-180, 180)

    longitude = None

13. *targetGridResolution* is resolution in degree (1 degree = 100km approx) if *targetGridResolution* is set to *None*, then model resolution will be kept in the grib2 file. This must be a *number* or *None*.

    targetGridResolution = 0.25

14. *pressureLevels* is required pressure levels slice / extract only particular set of pressure levels from model pressure levels. User can specify either one or more levels. By default it takes *None*, i.e. it will extract all the model pressure levels.
    eg 1 : pressureLevels = [850] -> extract 850 hPa only
    eg 2 : pressureLevels = [850, 500, 200] -> extract only 850, 500 & 200 hPa levels only.
    Note 1 : These pressure slice levels applicable to all the pressure level variables.
    Note 2 : At the moment pressure levels interpolation is not supported!

    pressureLevels = None

15. *soilFirstSecondFixedSurfaceUnit* takes either '*cm*' or '*mm*'. By default it takes '*cm*' argument (suggested for general purpose/ WRF-Noah supported). For soil moisture/ soil temperature variables depth below land surface, units are initially set to either '*cm*' (centimeter) or '*mm*' (millimeter), and finally converted to '*m*' (meter) in wgrib2. But anyhow if grib2 files will be read by some other utility other than wgrib2, then this first & second fixed surface units plays matter. So suggested units is 'cm'.

    soilFirstSecondFixedSurfaceUnit = cm

16. By default *startdate* takes argument as *YYYYMMDD* (which means it assume today's date), But user can specify the different *startdate* by follwing the same format. for eg, *startdate* = 20151209 # then it will execute the scripts for 09-Dec-2015. *startdate* = *YYYYMMDD* # then it will execute the scripts for today's date.
    Note : However **UMRIDER_STARTDATE** environment variable will override this startdate option.

    startdate = YYYYMMDD

17. By default *enddate* is *None*. User can specify different *enddate* (but > startdate) i.e. for only specfied date / one date, user needs to control only in the *startdate* and *enddate* must be *None*. If user want to execute the um2grb2 conversion for the range of dates, then user need to set the lower *startdate* end higher *enddate*. *enddate* could be even *YYYYMMDD*, but make sure that *startdate* is lower than *enddate*.
    For eg : *startdate* = 20151209 and *enddate* = 20160114 , then um2grb2 conversion program executes from 09-dec-2015 to 14-jan-2016.
    Note : However **UMRIDER_ENDDATE** environment variable will override this enddate option.

    enddate = None

18. Load g2utils from *'system'* python which has installed through setup.py (OR) Load g2utils from 'local' previous directory for the operational purpose, where normal user don't have write permission to change the g2utils! So *loadg2utils* argument should be either *system* (default) or *local*.

    loadg2utils = local

19. If *overwriteFiles* option is *'True'* then existing output final files (if any) will be deleted from outPath and re-creating freshly. If *overwriteFiles* option is *'False'* and all output final files are already exists in the outPath, then program will be exited without re-creating the output files. If partially created files exist (like few hours outfiles only exist or intermediate nc files) then by default make *overwriteFiles* option as True (though False as passed to *overwriteFiles* option).

    overwriteFiles = True

20. *anlOutGrib2FilesNameStructure* and *fcstOutGrib2FilesNameStructure* takes list of string naming arguments to construct out file names. um2grb2 will just concatenate the arguments, by replacing 3 predefined naming structure *('*HHH*', *YYYYMMDD*', '*ZZ*')* with its corresponding values/numbers in place of it.

    *'*H*'* - forecast hours

    *'*D*'* - forecast days (applicable only for multiples of 24 hours)

Note : Either '*H*' or '*D*' valid, not both!

*'*YYYYMMDD*'* - forecast reference date

'%d%m%y*'* - Alternate to above *'*YYYYMMDD*'* option. User can specify any acceptable time *strftime* format. um2grb2 will figure it by finding *'%'* symbol.

*'*Z*'* - forecast reference utc time (optional)

If user wants to 3 digit filled hours, then they need to specify as 3 times '*HHH*'. If they specify 2 digit filled hours (say '*HH*' only), but forecast hours have 3 digit, then by default it will assume as 3 digits but for single digit hour, it will fill 0 as prefix to make it as 2 digit. same option for utc '*Z*'.

*'*pXp*'* - latitude x longitude grid resolution

Note : * will not be included in the name of the final out grib2 files.

eg1 : ('um_ana', '_', '*HHH*', 'hr', '_', '*YYYYMMDD*', '_', '*ZZ*', 'Z', '.grib2') this will produce grib2 files as 'um_ana_006hr_20160208_12Z.grib2'

eg2 : ('fcs', '_', '*HH*', 'h', '_z', '*YYYYMMDD*', '.grb2') this will produce grib2 files as 'fcs_06h_z20160208.grb2'

eg3 : ('prg', '*D*', '00z', '*%d%m%y*', '.grb2') will produce grib2 files as ‘prg100z080216.grb2'

eg4 : ('prg', '*D*', '00z', '*%d%m%y*', '_', '*pXp*' '.grb2') will produce grib2 files as 'prg100z080216_0p17x0p17.grb2' in case of *targetGridResolution = None* (i.e modelResolution) or as 'prg100z080216_2p5x2p5.grb2' in case of t*argetGridResolution = 2.5* Defining analysis grib2 fileName structure. Must be in single line.

anlOutGrib2FilesNameStructure = ('um_ana', '_', '*HHH*', 'hr', '_', '*YYYYMMDD*', '_', '*ZZ*', 'Z', '_', '*pXp*', '.grib2')

21. Defining forecast grib2 fileName structure. Must be in single line.

    fcstOutGrib2FilesNameStructure = ('um_prg', '_', '*HHH*', 'hr', '_', '*YYYYMMDD*', '_', '*ZZ*', 'Z', '_', '*pXp*', '.grib2')

22. If *createCtlIdxFiles* is *True* then um2grb2 module will create grads control files and its index files for each and every grib2 files by using g2ctl.pl

    createGrib2CtlIdxFiles = True

23. If *convertGrib2FilestoGrib1Files* is *True* then using *'cnvgrib -g21'* command line um2grb2 module will convert grib2 file to grib1 files. *CAUTION* : it may produce invalid variables names, grib1 param code for few variables which are produced by this um2grib2 conversion tool!!!

    convertGrib2FilestoGrib1Files = False

24. If *grib1FilesNameSuffix* is *'.grib1'*, then grib1 files will endswith *'.grib1'* (default). otherwise will whatever string assigned will be added at the end of grib1 file names. None will add nothing to grib1 file names at the end of it.

grib1FilesNameSuffix = '.grib1'

25. If *removeGrib2FilesAfterGrib1FilesCreated* is *True*, then grib2 files will be deleted and kept only grib1 files. By default *False*.

    removeGrib2FilesAfterGrib1FilesCreated = False

26. If *createCtlIdxFiles* is *True* then um2grb2 module will create grads control files and its index files for each and every grib1 files by using grib2ctl.pl

    createGrib1CtlIdxFiles = False

27. This *debug* option should be either *True* or *False*. This will just print extra information like variables details, shape, execution process, etc.,

    debug = False

28. This *setGrib2TableParameters* option takes list of tuples which may contain WMO-Grib2 table parameters and its value. This means, the grib2 table parameter options will be overwritten as per user's setting in this option.
    eg1 : *setGrib2TableParameters* = [('centre', 28), ('subCentre', 0)]
    The above two options will be set to out grib2 files.
    eg2 : *setGrib2TableParameters* = [('shapeOfTheEarth', 0)]
    The above option will be set to out grib2 files.
    CATUION : User must be aware on what are they setting in this option and its causes in out grib2 files! By default this option takes None.

    setGrib2TableParameters = None

29. After successfully created the grib2 (final ordered variables) file, wgrib2 command will be executed with the 'wgrib2Arguments' options. Pygrib / IRIS / UMRider is able to write grib2 file with "grid_simple" packing algorithm, whereas wgrib2 able to convert packing from "grid_simple" to "grid_complex_spatial_differencing" by setting -set_grib_type complex2 option in it. The second type packing reduces file size 1/3 compare to first type packing. And further can be reduced the file size, by passing -set_bin_prec 12 (compatible same as ECMWF) which reduces the floating points precision (which further reduces the file size 1/5 th of original of first packing). By default *wgrib2Arguments* takes **"-set_grib_type complex2 -grib_out"** as argument. User can override this option by including extra wgrib2 arguments Or ***None*** (wgrib2 will not be executed). -grib_out is important argument (to be compressed, set precsion, etc) Ref Links : http://www.cpc.ncep.noaa.gov/products/wesley/wgrib2/speed.html http://www.cpc.ncep.noaa.gov/products/wesley/wgrib2/set_bin_prec.html

    wgrib2Arguments = -set_bin_prec 12 -set_grib_type complex2 -grib_out

30. This *callBackScript* option takes any user defined script (any script)! User should provide absolute or relative path of their script and make sure that script is self-executable with shebang and executable permission! After successfully created out grib2 files, this *callBackScript* will be executed with possibly command line keyword arguments as follows

KWargs : (*date*, *outpath*, *oftype*, *utc*) where

'date' -> out files processed date,
'outpath' -> out files path,
'oftype' -> 'analysis' or 'forecast'
'utc' -> UTC cycle value in string ('00' or '06' or '12' or '18')
For eg : callBackScript = imd_mfi_rename_g2files_and_put_into_ftp.sh

callBackScript = None

######### END OF UMRIDER SETUP CONFIGURE FOR um2grb2 SCRIPTS #########

UMRider Vars Configure Options

---

vars configure file: Used for the purpose of um2grb2 conversion of only needed NCUM model out variables. um2grb2 python parallel scripts will create analysis and forecast files, by conveting to gri2 file only for the following cf_standard_name and varSTASH coded vars.

Author : Arulalan <arulalan@ncmrwf.gov.in>

Updated : 01-Feb-2016

######### BEGIN OF UMRIDER VARS CONFIGURE FOR um2grb2 SCRIPTS ########

Pressure Level Variable names & STASH codes

('geopotential_height', 'm01s16i202')
('x_wind', 'm01s15i243')
('y_wind', 'm01s15i244')
('upward_air_velocity', 'm01s15i242')
('air_temperature', 'm01s16i203')
('relative_humidity', 'm01s16i256')
('specific_humidity', 'm01s30i205')

Non Pressure Level Variable names & STASH codes

('tropopause_altitude', 'm01s30i453')
('tropopause_air_temperature', 'm01s30i452')
('tropopause_air_pressure', 'm01s30i451')
('surface_air_pressure', 'm01s00i409')
('air_pressure_at_sea_level', 'm01s16i222')
('surface_temperature', 'm01s00i024')
('relative_humidity', 'm01s03i245')
('specific_humidity', 'm01s03i237')
('air_temperature', 'm01s03i236')
('dew_point_temperature', 'm01s03i250')
('atmosphere_convective_available_potential_energy_wrt_surface', 'm01s05i233')

```
('atmosphere_convective_inhibition_wrt_surface', 'm01s05i234')
('high_type_cloud_area_fraction', 'm01s09i205')
('medium_type_cloud_area_fraction', 'm01s09i204')
('low_type_cloud_area_fraction', 'm01s09i203')
('cloud_area_fraction_assuming_random_overlap', 'm01s09i216')
('cloud_area_fraction_assuming_maximum_random_overlap', 'm01s09i217')
('atmosphere_cloud_liquid_water_content', 'm01s30i405')
('atmosphere_cloud_ice_content', 'm01s30i406')
('atmosphere_mass_content_of_water', 'm01s30i404')
('atmosphere_mass_content_of_dust_dry_aerosol_particles', 'm01s30i403')
# STASH is None, because atmosphere_precipitable_water_content will be calculated by using
# atmosphere_mass_content_of_water -
atmosphere_mass_content_of_dust_dry_aerosol_particles
# - atmosphere_cloud_liquid_water_content - atmosphere_cloud_ice_content
('atmosphere_precipitable_water_content', 'None')

('x_wind', 'm01s03i209')
('y_wind', 'm01s03i210')
('visibility_in_air', 'm01s03i247')
('precipitation_amount', 'm01s05i226')
('stratiform_snowfall_amount', 'm01s04i202')
('convective_snowfall_amount', 'm01s05i202')
('stratiform_rainfall_amount', 'm01s04i201')
('convective_rainfall_amount', 'm01s05i201')
('rainfall_flux', 'm01s05i214')
('snowfall_flux', 'm01s05i215')
('precipitation_flux', 'm01s05i216')
('fog_area_fraction', 'm01s03i248')
('toa_incoming_shortwave_flux', 'm01s01i207')
('toa_outgoing_shortwave_flux', 'm01s01i205')
('toa_outgoing_shortwave_flux_assuming_clear_sky', 'm01s01i209')
('toa_outgoing_longwave_flux', 'm01s02i205')
('toa_outgoing_longwave_flux_assuming_clear_sky', 'm01s02i206')
('surface_upward_latent_heat_flux', 'm01s03i234')
('surface_upward_sensible_heat_flux', 'm01s03i217')
('surface_downwelling_shortwave_flux_in_air', 'm01s01i235')
('surface_downwelling_longwave_flux', 'm01s02i207')
('surface_net_downward_longwave_flux', 'm01s02i201')
('surface_net_downward_shortwave_flux', 'm01s01i202')

# for the following two vars STASH is None, because it will be calculated by using
# net and down shortwave/longwave flux
('surface_upwelling_shortwave_flux_in_air', 'None')
('surface_upwelling_longwave_flux_in_air', 'None')

('atmosphere_boundary_layer_thickness', 'm01s00i025')
('atmosphere_optical_thickness_due_to_dust_ambient_aerosol', 'm01s02i422')
('moisture_content_of_soil_layer', 'm01s08i223'),  # 4 layers

# single layer, this must be after 4 layers as in order
('soil_moisture_content', 'm01s08i208'),       # single layer
```

though moisture_content_of_soil_layer and volumetric_moisture_of_soil_layer
has same STASH code, but we must include seperate entry here.
('volumetric_moisture_of_soil_layer', 'm01s08i223'), # 4 layers

single layer, this must be after 4 layers as in order
('volumetric_moisture_of_soil_layer', 'm01s08i208'), # single layer
('soil_temperature', 'm01s03i238')
('sea_ice_area_fraction', 'm01s00i031')
('sea_ice_thickness', 'm01s00i032')

the snowfall_amount might be changed as liquid_water_content_of_surface_snow by convert it into water equivalent of snow amount, before re-ordering itself.
('liquid_water_content_of_surface_snow', 'm01s00i023')

the below one is for land-sea binary mask which should presents only in analysis files. so we must keep this as the last one in the ordered variables!
('land_binary_mask', 'm01s00i030')

the below one is for orography which presents only in analysis 00 file.
so we must keep this as the last one in the ordered variables!
('surface_altitude', 'm01s00i033')

########## END OF UMRIDER VARS CONFIGURE um2grb2 SCRIPTS #############

Future Releases of UMRider

ncum post operational v1.0.2

Currently I am working on umeps products (i.e. NCUM produces 45 ensembles at every 6-hourly and 24-houlry for next 10-days forecasts) and it will be included in the next version UMRider 1.0.2. It will produce all ensembles in single grib2 files (every 6-hourly / 24-hourly grib2 files should contain all its ensemble members).

ncum post operational v1.0.3

Currently I am working on um_reg products (i.e. NCUM_REG produces files at 4.5 km resolution over Indian region) and it will be included in the upcoming version UMRider 1.0.3. NCUM_REG produces in rotated lat,lon grid which needs to convert to regular lat, lon grids and also need to vertical interpolation from hybrid to standard pressure levels.

ncum post operational v1.0.4

User will be able to interact with UMRider via Graphical User Interface to convert any NCUM files to grib2, netCdf, pp files with all user defined UMRider options.

ncum post operational v1.0.5

Need to fix in iris to support the remaining variables which are all ‘unknown’ vars or unknown grib2 param code.

ncum post operational v1.0.6

Command line support to generate bsub scripts, invoke gui to customization and submit job.

ncum post operational v1.1.0

UMRider should supports for all partners of UK-MetOffice-Unified Model, not just limited to NCMRWF’s NCUM.

Acknowledgement

I heartily thank Dr. E. N. Rajagopal (Scientist – G & Head of NCMRWF) who assigned me this task UM to grib2 conversion using Python, initially. Also would like to thank Mr. G.R. Iyengar (Scientist – G) who compared and acknowledged the UMRider testing results made by other scientists.

Most importantly I acknowledge Mr. M.N.Raghavendra Sreevathsa (Scientist – E) 's initial work on iGui code in serial version of python. On top of his code basement, I was able to build UMRider successfully.

I thank Dr. Raghavendra Ashrit (Scientist – E) who tested UMRider out grib2 files by feeding to WRF-Noah Land Surface Model and pointed out the bugs in code.

Also I convey my thank to Dr. Saji Mohandas (Scientist – F) and Dr. John P. George (Scientist – F) for their inputs to construct the UMRider.

I sincerely thank the following scientists of NCMRWF who are all tested the UMRider out grib2 files by visually compared with um subset.ctl tool's grib1 files (subset.ctl script made by Dr. Saji Mohandas, Scientist – F).

- Dr. A.Jayakumar (Scientist – C)
- Mr. M.Momin Imran Ali (Scientist – C)
- Dr. Sumit Kumar (Project Scientist – D)
- Dr. C.J. Johny (Project Scientist – C)
- Mr. Kuldeep Sharma (Project Scientist – C)
- Mr. Abhishek Lodh (Project Scientist – C)

Finally I thank Ms. Shivali Gangwar (IBM Admin & Support) who installed IRIS versions in Bhaskara at NCMRWF and Mr. Nisheedh who running UMRider v1.0.0 in the production of NCMRWF.

Regards,

Arulalan.T
Project Scientist – C
26-Apr-2016