



National Centers for Environmental Prediction

NCEP Central Operations
Environmental Modeling Center

Software Version Numbering



Disclaimers

The information contained in this document is subject to change without notice.

Typographical Conventions

This document uses the following typographical conventions:

Bold – Command and option names appear in **bold** type in definitions and examples.

- Directories, files, partitions, and volumes also appear in bold.
- Interface controls (check boxes, radio buttons, fields, folders, icons, list boxes, items inside list boxes, multicolumn lists, menu choices, menu names, and tabs)
- Keywords and parameters in text

Italic – Variable information appears in *italic* type. This includes user-supplied information on command lines.

- Citations (titles of books, diskettes, and CDs)
- Emphasis of words
- Words defined in text

Monospace – Screen output and code samples appear in monospace type.

- Examples and code examples, for example, `this is a line of code`
- File names, programming keywords, and other elements that are difficult to distinguish from surrounding text
- Message text and prompts addressed to the user
- Text that the user must enter
- Values for arguments or command options

Who Should Read This Document?

This guide is designed to benefit the following professionals:

System/Subsystem Owners and Team Members

This document will help the system owners apply the rules for version numbering consistently across all operational software.

Configuration Managers

This document will help the configuration managers enforce the rules for version numbering consistently across all operational software.

Document History

Paper copies are valid only on the day they are printed. Contact the author if you are in any doubt about the accuracy of this document.

Revision History

Revision Number	Revision Date	Summary of Changes	Author
1	5/10/2010	Initial Document	Scott Jacobs
2	6/29/2010	Add Software Units and Initial Version Numbers	Scott Jacobs
3	7/2/2010	Update Mesoscale Branch information	Geoff DiMego
4	7/2/2010	Update Global Branch information	John Ward
5	7/6/2010	Update NCO library information	Scott Jacobs
6	7/7/2010	Update Marine Branch information	Hendrik Tolman
7	7/7/2010	Re-order sections	Scott Jacobs
8	7/28/2010	Sponsor corrections and suggestions added	Scott Jacobs
9	7/29/2010	Added "Major" Systems to the software unit tables	Scott Jacobs

Approvals

This document requires following approvals:

Name	Title
Ben Kyger	Director, NCEP Central Operations
Stephen J. Lord	Director, Environmental Modeling Center

Table of Contents

1.	Introduction.....	5
1.1	Purpose of this document.....	5
1.2	Identification	5
1.3	Points of Contact.....	5
2.	Version Numbering Scheme.....	6
2.1	Overall Scheme.....	6
2.2	Major Version	6
2.3	Minor Version	7
2.4	Maintenance Version.....	7
2.5	Summary	7
2.6	Updating the Scheme.....	8
3.	System and Subsystem Versions.....	9
3.1	Production Suite Major Systems	9
3.2	Libraries and Utilities	10
3.3	Mesoscale Modeling Branch	10
3.4	Marine Modeling and Analysis Branch	11
3.5	Global Climate and Weather Modeling Branch.....	12
3.6	Miscellaneous Software or Support Units.....	12
3.7	Maintaining Version Numbers	13
4.	System and Subsystem Unit Descriptions.....	14
4.1	Libraries and Utilities	14
4.2	Mesoscale Modeling Branch	16
4.3	Marine Modeling and Analysis Branch	33
4.4	Global Climate and Weather Modeling Branch.....	38
4.5	Miscellaneous Software or Support Units.....	39

1. Introduction

1.1 Purpose of this document

This document covers the scheme used to number the software units used in production on the NCEP Central Computer Systems (CCS). This document will also identify the software units and define the boundaries of each unit.

1.2 Identification

- This document applies to all software units currently used in operations and to any future units developed by EMC or NCO.
- The numbering scheme presented in this document is an amalgamation of many schemes used in various software development fields. This document attempts to combine these into a single scheme appropriate to model and library development at NCEP.

1.3 Points of Contact

- Project Manager
 - Scott Jacobs
- Team Members
 - Jeff Ator
 - Chad Cary
 - Geoff DiMego
 - Chris Magee
 - Dan Starosta
 - Hendrik Tolman
 - John Ward

2. Version Numbering Scheme

2.1 Overall Scheme

There are a wide variety of numbering schemes used in software development projects. They are all variations on the same basic concept: keep track of different versions of a piece of software. The vast majority of the version numbering schemes are sequence based. A sequence of numbers, separated by a delimiter, is used to convey the significance of the changes between releases. The left most value is changed for the most significant software modifications. Changes to the subsequent values indicate decreasing significance.

Many version numbering schemes are quite complex. One goal with the NCEP scheme is to keep it simple. Therefore, the NCEP scheme is defined as:

```
major.minor.maintenance
```

The NCEP scheme is also defined as only using numeric values for each sequence number. Some version numbering schemes allow for the use of “a” and “b” to denote alpha and beta versions of the software. This usage makes these schemes more complex. Therefore, the use of letters in the NCEP version numbering is not allowed.

For planning purposes, the next version number of a software unit should be defined early in the development process. That is, when a requirement, or set of requirements, forces a change in the software unit, the scope of the development needed should be determined. At this time, the scope will also dictate whether the released software will be a *major*, *minor* or *maintenance* release. If, during development, it becomes apparent that the scope was not properly estimated, the version number may change in the planning documents.

Each of the following sections defines the criteria for increasing the value of each sequence number.

2.2 Major Version

What constitutes a Major Version? Generally, use of the term *major* is very subjective. This section attempts to define the causes of incrementing the major value in the version sequence.

For the numerical prediction systems, changes to the resolution, the underlying model, the model physics or the data assimilation would necessitate incrementing the major version number. These changes have wide-spread effects on the entire model system being modified. The scope of the change must also play a role in determining the version number. For an entire model system, if only one subsystem is changing, the major number for the encompassing system may not change. However, if many subsystems are modified, then the major version value should increase.

The common libraries version numbers are mainly driven by the scope of the changes to the library routines. For example, if a new edition of GRIB is introduced by the WMO, the changes necessary to implement the new edition would be very extensive, touching most, if not all, of the library functions. This scenario would result in incrementing the major version number.

Table format changes require an increase in the major version of the table or of the software unit used to access the table. This type of change could be related to a new international standard or an internal NCEP requirement to modify an existing table to store new or different information.

Customer service should always be a focus for any software unit that creates products. Therefore, any time a software change causes a change to the output content or format, the change constitutes a major version.

2.3 Minor Version

Similar to the major value, *minor* is a subjective term. A minor version is defined as a change involving selected areas of a model or changes for IT reasons. The scope of the changes is also less far-reaching than a major update.

Where a major table update might be a complete format change, a minor update may only be modification to, addition of or deletion of a few entries.

Data format changes to existing data sets fall into the category of minor versions. For example, the data type is already being ingested into the models, but the data supplier announces a change to the format. The decoder would need to be changed in this case, but the changes may be of limited scope.

From a customer perspective, a minor version should present no differences in the content or format of the output.

When a major update is released, the value of the minor version should be reset to zero.

2.4 Maintenance Version

A maintenance version has the least scope. This level of update is reserved for bug fixes or changes to a single routine or file. The maintenance number will usually be increased for solutions to operational problems that cannot wait for a minor or major version.

When a major or minor update is released, the value of the maintenance version should be reset to zero.

2.5 Summary

Major Version

- Resolution changes
- Underlying model replacement and/or changes
- Major scientific technology upgrade, e.g., semi-lagrangian or physics updates
- Large scope, many subsystems involved
- Update to an international standard
- Table format changes
- Customers can expect to see differences in the output

Minor Version

- Model Physics changes
- IT-related changes, modifications that do not affect the model or its output, but how the model is executed or how data moves into or out of the model
- Medium scope, limited number of subsystems or routines
- Table entry updates
- Customers should expect to see no significant differences in the output or performance

Maintenance Version

- Bug fixes
- Solutions to operational problems
- Small scope, only a few files are affected by the change
- Updates to data files with an implicit dynamic nature, such as buoy files

2.6 Updating the Scheme

There will undoubtedly be circumstances pertaining to version numbering that the initial team did not foresee. Therefore, this document will be reviewed on a periodic basis to ensure that all software units and situations involving updates to those units are accounted for in the numbering scheme.

The concept of using *major.minor.maintenance* for numbering all software units is currently the best approach devised by the team. If, however, a situation occurs in the future that requires more granularity in the tracking of versions, the scheme will need to be reviewed and modified. More likely, the definitions of *major*, *minor* and *maintenance* will undergo modifications and clarifications in the coming years. The definitions of the scope of these levels of changes will be reviewed periodically.

The initial review period shall be quarterly. A team will be convened to determine if there are needed modifications to this document. The team's term will be limited to two weeks and will result in an updated document or a statement indicating that an update was not necessary.

After one year, the review cycle will also be reviewed.

3. System and Subsystem Versions

This chapter identifies the current and planned versions as of the writing of this document. It also describes the location and process for updating this information.

3.1 Production Suite Major Systems

The software units in this section are major model systems currently in operational use at NCEP. Each uses numerous software units identified in the later sections. Each of these major systems has its own version number and is composed of software units with their own version numbers. As noted in section 3.7, the information about software units and version numbers should be posted to a web site that will be easier to update and maintain than this document.

Software Unit	Current Version	Next Version
GFS (model)	9.0.0	9.1.0
GFS (analysis)	9.0.0	9.1.0
GEFS – NAEFS	9.0.0	9.1.0
NAM (model)	2.5.2	3.0.0
NAM (analysis)	2.5.2	3.0.0
SREF	5.0.0	6.0.0
Rapid Refresh	5.5.5 (RUC)	1.0.0 (RR)
RTMA (CONUS, Alaska, Hawaii, Puerto Rico, Guam)	2.1.0	2.2.0
RTOFS (model)	1.0.0	
RTOFS (analysis)	1.0.0	
Waves (global, hurricane, nests)	2.0.0	
Waves (ensemble)	2.0.0	
Hurricane (WRF)	4.0.0	
Hurricane (GFDL)	8.0.0	
Nested Window Model	5.0.0	5.1.0
AQ (CMAQ)	4.6.0	
AQ (HYSPLIT)	5.0.0	5.1.0
CFS	2.0.0	
GODAS	3.0.0	

3.2 Libraries and Utilities

Software Unit	Current Version	Next Version
BUFRLIB	20080528	10.0.0
Decoders	5.11.4	6.1.0
NAWIPS	6.1.1	6.1.2
W3LIB	1.8.0	2.0.0
G2LIB (GRIB2 Fortran library)	1.2.0	2.0.0
G2CLIB (GRIB2 C library)	1.2.0	2.0.0
Cnvgrib	1.2.0	2.0.0
Verification	1.0.0	1.1.0
CPG (Selected Cities Product)	1.0.0	1.1.0
Model Analysis and Guidance (MAG) web site	1.1.1	1.2.0
NOMADS	1.0.0	1.1.0
MADIS	Under Development	1.0.0

3.3 Mesoscale Modeling Branch

Software Unit	Current Version	Next Version
NAM Data Assimilation System (NDAS)	2.5.2	3.0.0
North American Mesoscale (NAM) system	2.5.2	3.0.0
Downscaled GFS with NAM Extension (DGEX)	2.4.2	3.0.0
High Resolution Window	5.0.0	5.1.0
Short Range Ensemble Forecast (SREF)	5.0.0	6.0.0
Rapid Update Cycle (RUC)	5.5.5	1.0.0 (RUC becomes the Rapid Refresh)
Air Quality Forecasting System (AQFS)	4.5.0	
aqm_prep_5xwrf	45.0.0	
aqm_premaq_5xwrf	45.0.0	
aqm_fcst_5xwrf	4.5.0	
aqm_post_5xwrf	45.0.0	
Community Multiscale Air Quality (CMAQ)	4.6.0	
aqm_prep_5xpmwrf	46.0.0	
aqm_premaq_5xpmwrf	46.0.0	
aqm_fcst_5xpmwrf	46.0.0	
aqm_post_5xpmwrf	46.0.0	
Hawaii CMAQ	46.0.0	
aqm_prep_Hlwrf	46.0.0	
aqm_premaq_Hlpmwrf	46.0.0	
aqm_fcst_Hlpmwrf	46.0.0	
aqm_post_Hlpmwrf	46.1.0	
Alaska CMAQ	46.0.0	
aqm_prep_Akwrf	46.0.0	
aqm_premaq_Akpmwrf	46.1.0	

aqm_fcst_Akpmwrf	4.6.0	
aqm_post_Akpmwrf	46.1.0	
Hybrid Single-Particle Lagrangian Integrated Trajectory (HYSPLIT)	5.0.0	5.1.0
Cyclone Tracker - Regional	2.3.1	2.3.2
Cyclone Tracker - Global	3.3.1	3.4.0
TrakVerif	2.1.1	2.2.0
Verification	1.0.1	1.1.0
Precipitation processing and analysis	1.1.1	1.2.0
Real Time Mesoscale Analysis (RTMA)	2.1.0	2.2.0
Radar Obs processing	1.0.0	1.1.0
All Obs Processing	1.0.0	1.1.0

3.4 Marine Modeling and Analysis Branch

Software Unit	Current Version	Next Version
Gulf Stream Finder	2.0.0	2.0.1
gsf_parse.pl	1.0.0	
Sea Ice Drift Model (SID)	2.0.0	2.0.1
SID input averaging	1.0.0	1.0.1
SID fix files	1.0.0	
Sea Ice Concentration Analysis (SIC)	3.1.0	
SIC - SSMI	3.0.1	
SIC - AMSR	1.0.0	
SIC - multisatellite blending	1.0.0	
SIC - sea ice grid resolution reduction	1.0.0	
SIC - XPM graphics	1.0.0	
SIC - output into GRIB	1.1.0	
SIC - SST filtering	1.1.0	
SIC - Polar to LatLon	1.1.0	
SIC - Land mask - high res	3.1.0	
SIC - Land mask - low res	3.0.0	
Global Marine Visibility and Vessel Icing	5.0.0	
MMABlib	2.0.0	
RTOFS-Atlantic	2.4.1	
HY-HWRF	1.0.0	
RTOFS-Global	1.0.0	
Global multi-grid wave model	2.0.0	
Multi-grid hurricane wave model	1.0.0	
Global wave ensemble	2.0.0	
Great Lakes NAM wave model	2.0.0	2.1.0
Great Lakes NDFD wave model	1.0.0	2.0.0
WAVEWATCH III ¹	3.14.0	3.14.1

¹ WAVEWATCH III codes are used in all wave model suites, and will follow the major-minor version numbering of releases for this package, with the maintenance version number identifying bug fixes to the established release versions. All other wave model codes are numbered with their main suite numbering.

	Legacy codes	2.22.0	Discontinued
SST Analysis - RTG low res		1.6.0	
SST Analysis - RTG high res		2.7.0	
SST Analysis - daily OI		1.1.1	
SST Analysis - weekly OI		2.0.0	
SST Analysis - quarter degree OI		3.0.0	

3.5 Global Climate and Weather Modeling Branch

Software Unit	Current Version	Next Version
global_postevents	9.0.0	9.1.0
global_mrf_look_alike	9.0.0	9.1.0
global_sfchdr	9.0.0	9.1.0
global_satcount	9.0.0	9.1.0
global_chgres	9.0.0	9.1.0
global_chgres_thread	2.0.0	
global_sighdr	9.0.0	9.1.0
global_gsi	3.0.0	
global_fcst	9.0.0	9.1.0
global_cycle	9.0.0	9.1.0
global_angupdate	3.0.0	
ncep_post	3.0.0	3.1.0
overparm_grib	9.0.0	9.1.0
genpsiandchi	9.0.0	9.1.0
relocate_mv_nvortex	5.0.0	
global_chg_igen	9.0.0	9.1.0
global_sig_igen	9.0.0	9.1.0

3.6 Miscellaneous Software or Support Units

Software Unit	Current Version	Next Version
anomgb	8.0.0	
grbindex		
cnvgrb	8.0.0	
copygb	8.0.0	
ndate		
supvit		
gettrk		
wgrib		
wgrib2		
BACIO - Byte-addressable C I/O	1.3.0	2.0.0
IPLIB		
SpectralLIB		

3.7 Maintaining Version Numbers

The version numbers on the above software units will be reviewed when work is planned for the units. That is, as new development is scheduled for a specific item, the version number for that work will be set. This version number will be used to refer to the proposed software changes during planning and scheduling. If, during development, the version number needs to be updated, the project sponsor must approve any modifications.

The above tables are valid as of the writing of this document. Future information about the software units, including version numbers, will be maintained on a central web site. (The URL for the web site will be established at a later time, and this document will be updated.)

4. System and Subsystem Unit Descriptions

The following sections describe each of the software units in more detail. The content was created by the team to give background information on each of the software units. The information found in this section was used to create the tables in Section 3.

This is the first attempt at bringing all of this information into a single document. In the future, the information will be maintained on a single or multiple web sites. Once the sites have been identified or created, the URLs will be added to this document to refer the background information.

4.1 Libraries and Utilities

NCO SIB/SWD “units” to be version-controlled:

- **BUFRLIB** – does not currently contain a version number, but the next version to be implemented later this year will be labeled as version 10.0.0
 - Application codes can determine the current version by calling a defined subroutine within the library
- **decoders** – none of these currently contain a version number; however, they are all built upon NAWIPS and BUFRLIB, each of which have their own version numbers. So we could envision future decoder implementations following the proposed model scheme; e.g. the `decod_dcusnd` decoder X.Y.Z includes NAWIPS X.Y.Z and BUFRLIB X.Y.Z
 - Each decoder could print out its current version to the decoder log when it starts up
 - Several of the decoders are also built upon MADIS and the netCDF libraries, which have their own version numbers and could be handled in the same manner as above for NAWIPS and BUFRLIB. Similarly, the `decod_dcshef` decoder is built upon the OH SHEF parsing library which has its own version number. However, for these “external” libraries, I would propose that we stick with the version numbers assigned by the external developers of those libraries.
 - All of the decoders are also built upon the `DECOD_UT` library, which does not currently have a version number but could be given one and then treated similarly to the above libraries
 - The “scripts” for the decoders are maintained by PMB Dataflow, including the LDM configuration file `pqact.conf`
- **W3LIB** – does not currently contain a version number on the CCS; however, the public copy of this library (on the PMB web site) does have one which could be incorporated back onto the CCS for consistency
 - The version number could then be determined by any application code by calling a new subroutine which would return this information.
 - Since this library is a sort of “catch-all” of subroutines which perform a variety of unrelated functions for a variety of unrelated application programs, I would think that most updates to this library would probably fall into the Y or Z category. More specifically, Y would be updated when we add or delete a new function, and Z would be updated if we modified or fixed a bug in an existing function.

Perhaps if we modified the calling sequence for an existing function then this might qualify as a change in the X category(?)

- **G2LIB, G2CLIB** – do not currently contain version numbers on the CCS; however, the public copies of these libraries (on the PMB web site) do have version numbers which could be incorporated back into the CCS copies for consistency
 - The version number could then be determined by any application code by calling a new subroutine which would return this information.
- **CNVGRIB** – does not currently contain a version number on the CCS; however, the public copy (on the PMB web site) does have a version number which could be incorporated back into the CCS copy for consistency
 - The version number could then be determined by the user if we added a new runtime option (e.g. “-v”)
 - This program is built upon our G2LIB and G2CLIB libraries (which will have their own version numbers assigned by us) as well as the “external” JASPER, PNG and ZLIB libraries (for which we should keep the version numbers assigned by the external developers of these libraries).
 - Model would then be “CNVGRIB X.Y.Z includes G2LIB X.Y.Z and JASPER X.Y.Z, etc.
- **verification codes (/nwprod/sorc/verf*)** - none of these currently contain a version number, but they could be added to the codes
 - These are built upon the W3LIB and BUFRLIB libraries, which will each have their own version numbers.
 - Model would then be “VERF_CQC_BIAS X.Y.Z includes W3LIB X.Y.Z and BUFRLIB X.Y.Z, etc.
 - The version number could be printed to the joblog whenever the job runs
 - The associated scripts and fix files would have their own version numbers
- **CPG** - does not currently contain a version number on the CCS, but this could be added to the main code or even to the run script (see below)
 - The version number could be printed to the joblog whenever the job runs
 - The script for this program is closely-linked to the program in order to create the resulting XML output, so the script and source code could keep the same version numbering scheme
 - This program is built upon the BUFRLIB and NAWIPS (specifically, GEMLIB, CGEMLIB and XML2) libraries, each of which have (or will have) their own version numbers
 - Model would then be “CPG X.Y.Z includes NAWIPS X.Y.Z and BUFRLIB X.Y.Z, etc.
- **IP LIB**
- **Spectral LIB**
- **MAG**
- **NOMADS**
- **MADIS**

4.2 Mesoscale Modeling Branch

The following attempts to identify the major components of the Production Suite owned by Mesoscale Modeling Branch and to list their major components i.e. those pieces that, if changing, would warrant their own RFC. This ends up being on the lengthy side.

3.2.1 NDAS, NAM, and DGEX

NAM Data Assimilation System (NDAS) v2.5.2 from Eric Rogers

WPS (WRF preprocessing system)

Cold start (5 pieces, 2 scripts)

 partialcyc inserts cycled NDAS sfc states in GDAS init file, runs in its own script

 sst2mdl, ice2mdl, snow2mdl interpolate sst, ice & snow to NAM grid

 sfcupdate inserts 3 above into GDAS init file, these 4 run in the same script

MAKBND (2 pieces, 2 scripts)

 mkbnd interpolates global coefs to model boundary & writes them out in binary

 combc reads the binary files and writes them out in the model's BC file format

GSI analysis – NAM

Precip proc. 1: codes that run once/day just before first NDAS forecast in the 12z cycle

 lspasno24h extracts 24-h NDAS precip from GRIB output

 cpc2grbrfc8 convert CPC 1/8 deg daily precip file from binary to GRIB

 pcpbudget (24h NDAS sum - adj. daily gauge anal) added to budget history file

Precip proc. 2: codes that run before every NDAS forecast

 mergest2n4 merge Stage II and Stage IV precip analysis

 pcpprep interpolate merged analysis to model grid

WRF-NMM prediction model

NCEP Post processor

PRDGEN product generator

Sounding codes (just two)

 post0 extracts station profiles for each forecast hour and writes them into binary

 sndp converts the binary file to BUFR

North American Mesoscale (NAM) system v2.5.2 from Eric Rogers

NDAS (see above)

MAKBND (2 pieces, 2 scripts)

 mkbnd interpolates global coefs to model boundary & writes them out in binary

 combc reads the binary files and writes them out in the model's BC file format

GSI analysis - NAM

WRF-NMM prediction model

NCEP Post processor

PRDGEN

Sounding codes (all four)

 post0 extracts station profiles for each forecast hour and writes them into binary

 sndp converts the binary file to BUFR

 stnmlist breaks out BUFR file into station time-series

 collective creates BUFR collectives for shipping to AWIPS

DNG NAM downscales NAM forecasts producing sensible wx fields on NDFD grids

smartinit four different execs for CONUS, Alaska, Hawaii and Puerto Rico
Precip proc. 3: code runs only after the 00z NAM forecast
get1236hpcp computes/extracts NAM 12-36 h QPF for OCONUS precip adj. in NDAS

DGEX (Downscaled GFS with NAM EXtension) v2.4.2 from Eric Rogers

WPS creates restart file from NAM forecast on 32km #221 grid
partialcyc inserts 12km NAM sfc states in restart file
MAKBND (2 pieces, 2 scripts)
 mkbnd interpolates global coefs to model boundary & writes them out in binary
 combc reads the binary files and writes them out in the model's BC file format
WRF-NMM prediction model
NCEP Post processor
PRDGEN

3.2.2 Rapid Update Cycle (RUC)

Rapid Update Cycle (RUC) v5.5.5 from Geoff Manikin

HYBNAMBC boundary condition generator
GETBUFR data processing
HYBFRONT analysis
HYBCSTPR pre-processor
HYBCST prediction model
HYBPOST post processor
HYBSNDP sounding post processor

3.2.3 High Resolution Window (HiResW)

High Resolution Window v5.0.0 from Matt Pyle

This runs two models – the two dynamic core versions of the Weather Research and Forecasting (WRF) modeling system.

WPS (WRF preprocessing system)
WRF-NMM real initialization
WRF-NMM prediction model
NCEP Post processor
PRDGEN
Sounding codes (four)
HREF processing (next imp.)
HREF product gen (next imp.)

WPS (WRF preprocessing system)
WRF-ARW real initialization
WRF-ARW prediction model
NCEP Post processor
PRDGEN
Sounding codes (four)

3.2.4 Short Range Ensemble Forecast (SREF)

Short Range Ensemble Forecast (SREF) v5.0.0 from Jun Du

This is for the future three model system (current uses 4-5 model configurations).

Ens. perturb. prep.
NPS (NEMS Prep. System)
Cold start (pieces)
MAKBND
NEMS-NMMB
NCEP Post
PRDGEN
Sounding codes (4)
Ensemble prdgen

Ens. perturb. prep.
WPS (WRF Prep. System)
Cold start (pieces)
MAKBND
WRF-NMM
NCEP Post
PRDGEN
Sounding codes (4)
Ensemble prdgen

Ens. perturb. prep.
WPS (WRF Prep. System)
Cold start (pieces)
MAKBND
WRF-ARW
NCEP Post
PRDGEN
Sounding codes (4)
Ensemble prdgen

3.2.5 Air Quality

Air Quality Forecasting System (AQFS) using CMAQ (Community Multiscale Air Quality) from Jianping Huang & Jeff McQueen

CONUS CMAQ Operational v4.5.0

aqm_prep_5xwrf (1 exec, 4 scripts) v45.0.0

aqm_prep_5xwrf interpolates meteorological fields from NMM to CMAQ CONUS domain

Scripts: exaqm_5xwrf_prephyb0.sh.sms, exaqm_5xwrf_prephyb1.sh.sms,
exaqm_5xwrf_prephyb2.sh.sms, exaqm_5xwrf_prephyb3.sh.sms

aqm_premaq_5xwrf (1 exec, 1 script) v45.0.0 (un-unified domain system)

aqm_premaq_5xwrf prepares meteorological and emission fields for CONUS CMAQ forecast

Scripts: exaqm_premaq_5xwrf.sh.sms

aqm_fcst_5xwrf (1 exec, 1 script) v4.5.0 version dictated by EPA & NOAA/ARL

aqm_fcst_5xwrf prediction model for surface ozone and PM2.5 for the CONUS domain

Scripts: exaqm_fcst_5xwrf.sh.sms

aqm_post_5xwrf (3 execs, 3 scripts) serial AQM post-processing system v45.0.0 (unoptimized code w/o day 1 daily max) generates AQM CMAQ products in GRIB format, calculates 1-hr and 8-hr max surface ozone, and derives AOD (aerosol optical depth) products for CONUS based on CMAQ outputs.

Executable 1: aqm_post_5xwrf Script 1: exaqm_post1_5xwrf.sh.sms

Executable 2: aqm_cmaq_maxi2grib Script 2: exaqm_post1_maxi.sh.sms

Executable 3: aqm_post_5xwrf Script 3: exaqm_post2_5xwrf.sh.sms

CONUS CMAQ (running in NWPARA) v4.6.0

aqm_prep_5xpmwrf (1 exec, 4 scripts) v46.0.0

aqm_prep_5xpmwrf interpolates meteorological fields from NMM to CMAQ CONUS domain

aqm_premaq_5xpmwrf (1 exec, 1 script) v46.0.0 (un-unified domain system)

aqm_premaq_5xpmwrf prepares meteorological and emission fields for CONUS CMAQ forecast

aqm_fcst_5xpmwrf (1 exec, 1 script) v46.0.0

aqm_fcst_5xpmwrf prediction model for surface ozone and PM2.5 for the CONUS domain

aqm_post_5xpmwrf (4 execs, 4 scripts) serial AQM post-processing system v46.0.0 (unoptimized code w/o day 1 daily max) generates AQM CMAQ products in GRIB format, calculates 1-hr and 8-hr max surface ozone, and derives AOD (aerosol optical depth) products for CONUS based on CMAQ outputs.

Exec 1: aqm_post_5xpmwrf Script 1: exaqm_post1_5xwrf.sh.sms

Exec 2: aqm_cmaq_maxi2grib Script 2: exaqm_post1_maxi.sh.sms

Exec 3: aqm_post_5xpmwrf Script 3: exaqm_post2_5xwrf.sh.sms

Exec 4: aqm_rdgrbwt_aot Scripts 4: exaqm_post3_5xpmwrf.sh.sms

Note; CMAQ runs with CB04 mechanism in operations and with CB05 mechanism in parallel for CONUS

HI CMAQ (running in NWPARA) v46.0.0

aqm_prep_HIwrf (1 piece, 4 scripts) v46.0.0

aqm_prep_HIpmwrf interpolates meteorological fields from NMM to CMAQ Hawaii domain

aqm_premaq_HIpmwrf (1 script) v46.0.0 (un-unified domain codes) RFC submitted

aqm_premaq_HIpmwrf prepares meteorological and emission fields for Hawaii CMAQ forecast

aqm_fcst_HIpmwrf (1 script) v46.0.0

aqm_fcst_HIpmwrf prediction model for surface ozone and PM2.5 for the Hawaii domain

aqm_post_HIpmwrf serial AQM post-processing system v46.1.0 (more efficient code w/day1 max)

```
Exec 1: aqm_cmaq2grib, aqm_post_HIpmwrf Script 1:
exaqm_post1_HIpmwrf.sh.sms
Exec 2: aqm_cmaq_maxi2grib, Script 2: exaqm_post1_HImaxi.sh.sms
Exec 3: aqm_cmaq2grib Script 3: exaqm_post2_HIpmwrf.sh.sms
Exec 4: aqm_rdgrbwt_aot_CHA Script 4: exaqm_post3_HIpmwrf.sh.sms
```

AK CMAQ (will be running in parallel soon) v46.0.0

aqm_prep_AKwrf (1 exec, 1 script) v46.0.0

aqm_prep_AKwrf interpolates meteorological fields from NMM to CMAQ Alaska domain
Script: exaqm_ak_prephyb.sh.sms

aqm_premaq_AKpmwrf (1 script) v46.1.0 (Unified system for 3 domains)

aqm_premaq_AKpmwrf prepares meteorological and emission fields for Alaska CMAQ forecast

aqm_fcst_AKpmwrf (1 script) v4.6.0

aqm_fcst_AKpmwrf prediction model for surface ozone and PM2.5 for the Alaska domain

aqm_post_AKpmwrf serial AQM post-processing system v46.1.0 (more efficient code w/day1 max)

```
Exec 1: aqm_cmaq2grib, aqm_post_AKpmwrf Script 1:
exaqm_post1_AKpmwrf.sh.sms
Exec 2: aqm_cmaq_maxi2grib Script 2: exaqm_post1_AKmaxi.sh.sms
Exec 3: aqm_cmaq2grib Script 3: exaqm_post2_AKpmwrf.sh.sms
Exec 4: aqm_cmaq2grib, aqm_rdgrbwt_aot_CHA Script 4:
exaqm_post3_AKpmwrf.sh.sms
```

Air Quality HYSPLIT (Hybrid Single-Particle Lagrangian Integrated Trajectory) – everything v5.0.0 from Geoff Manikin

For the HYSPLIT-smoke jobs

```
NAMS2ARL - generates NAM background meteorology files
XTRCTGRID - creates regional background files for AK/HI and other smaller
regions
HYSPLIT FIRES-gets fires from NESDIS fires loc. file & active smoke from
prev. day
EPM - computes emissions from prescribed fires
HYMODEL - the dispersion model
CONCLOT - generates plots of the smoke
CON2ASC - generates ascii file for EPA
CON2GRIB - generates grib files
```

SMOKE_COPYGB-interpolates to desired grid while maintaining log 10 concentrations

The NAMS2ARL step is run once, and the XTRCTGRID is run for various regions. The remaining steps are each run for the CONUS, Alaska, and Hawaii systems.

These are other hysplit prep jobs that produce background meteorological data from other NCEP operational models for hysplit applications:

GDAS2ARL - converts GDAS analyses to ARL format
GFS2ARL - converts GFS forecasts to ARL format
GFSLR2ARL - converts low resolution GFS forecasts beyond 180 hours to ARL format
RUC2ARL - converts RUC forecasts to ARL format
NAM12ARL - converts NAM forecasts to ARL format
NAMAKARL - convert NAM Alaska forecats to ARL format
HIRESW4ARL - converts hi-res window forecasts to ARL format

other hysplit processing:

HYMODEL - dispersion model
HYMODEL - trajectory model
CONCPLOT - makes plots of concentrations
GELABEL - generates labels for plots sent to ftp server
CONCROP - deletes unneeded zeroes to make concentration file smaller
ASCII2SHP - creates gis files
VOLCPLOT - makes plots of volcanic ash
TRAJPLOT - basic trajectory plotting program

3.2.6 “Genesis” Tracker

Cyclone Tracker from Guang Ping Lou

1. Tracker V2.3.1: Currently running in operations. It is a regional storm tracking system that covers Atlantic basin, west and east Pacific basins, and CONUS.

The verification is not part of operational runs but a separate cron job. The code and scripts are in subversion repository. Steps are:

a) Whenever current cycle model data become available, a SMS script is invoked:

/nwprod/jobs/JTRACKER.sms.prod

b) The J-JOB submits a tracker script:

/nwprod/util/ush/genesis_tracker.sh

c) The script prepares required parameters and interpolates model data if necessary.

d) The script checks if there are any named storms issued by NHC and prepares them by invoking executable /nwprod/util//exec/supvit.

e) After preparation is completed, tracking starts by issuing executable:

/nwprod/util//exec/genesis_gettrk

f) After tracking is done, all track files are copied to appropriate directories and archived.

g) All post processing are done in a cron job and they are not part of operation runs.

The post process includes sorting, plotting, verification, and web displaying.

2. Tracker V3.3.1: In the pipeline (RFCed). It is a global cyclone tracking system that covers pole to pole. The code and scripts are in subversion repository.

The steps are:

- a) Whenever current cycle model data become available, a SMS script is invoked:
JTRACKER.sms.prod
- b) The J-JOB submits a tracker script:
genesis_tracker_poe.sh
This script uses multiple CPUs for ensemble models.
- c) The script prepares required parameters and interpolates model data if necessary.
- d) The script checks if there are any named storms issued by NHC and prepares them by invoking executable "supvit".
- e) The script checks previous 6hr, 12hr cyclone names and prepares them by invoking executable "supvit_gen".
- f) After preparation is completed, tracking starts by issuing executable:
/nwprod/util//exec/genesis_gettrk
- g) After tracking is done, all track files are copied to appropriate directories and archived.
- h) All post processing are done as part of operation runs except web displaying.
The post processing includes pairing, verification, plotting and web displaying.
See below for post processing steps.

3. TrakVerif V2.1.1: In the pipeline (RFCed). It is an automated cyclone verification system that verifies forecast tracks against model analysis tracks for past ten days. It also includes plotting capabilities. This system was submitted and attached to Tracker V3.3.1. The code and scripts are in subversion repository.

Steps are:

- a) Pairing forecast tracks with analysis tracks using "verify_pair_tcvit.sh" for tropical named storms and "verify_pair_glbl.sh" for extra tropical storms.
- b) Verify forecast tracks against analysis tracks using "verify_ver.sh" and executable "nhcver_opr_glbl.x".
- c) Getting verified data together and putting them in binary format by using tvercut_opr.sh and executable "wrtdat.x".
- d) plotting out verification results by "tver_plotter_ens.gs".
- e) Plotting tracks on maps by using mgtrak_plots_glbl.sh.

3.2.7 Verification Package

Verification - all to be v1.0.1 from Perry Shafran and Julia Zhu

For the model verifications versus observations:

verf_gridtobs_editbufr - thin prepbufr files based on grid area, time window, and data types selected
verf_gridtobs_prepfits - interpolates model data to observation location and writes out the pairs in a new bufr file.

verf_gridtobs_gridtobs - Job that reads in the output prepfits bufr file and writes out the partial sums needed for verification.

verf_gridtobs_gridtobs_ens (SREF runs only) - Jobs that reads in all the prepfits files for each SREF member and writes out ensemble partial sums needed for verification.

Ozone verification:

verf_gridtobs_editbufr_ozone - thins ozone prepbufr files based on grid area, time window, and data types

verf_gridtobs_editbufr_ozonemax - Job that computes the 1-hr and 8-hr ozone averages and max obs values.

verf_gridtobs_maxgrib_ozone - Job that finds the 1-hr and 8-hr maximum values of ozone in the model.

verf_gridtobs_prepfits_ozone - Job that interpolates model ozone data to the obs location and writes out the partial sums.

verf_gridtobs_gridtobs_ozone - Jobs that writes out the ozone partial sums needed for verification.

PM verification:

verf_gridtobs_editbufr_pm - Job that thins PM prepbufr files.

verf_gridtobs_editbufr_pmmax - Job that computes the 1-hr and 8-hr average and max obs values.

verf_gridtobs_avegrib_pm - Job that computes the 1-hr average values in the model.

verf_gridtobs_maxgrib_pm - Job that computes the 1-hr max values in the model.

verf_gridtobs_prepfits_pm - Job that interpolates model PM data to the obs location and writes out partial sums.

verf_gridtobs_gridtobs_pm - Job that writes out PM partial sums needed for verification.

3.2.8 Precipitation Processing and Analysis

Precipitation processing and analysis – everything to be v1.1.1 from Ying Lin

- Stage II analysis and precipitation RTMA v1.1.1

- 1) nam_combdpa: combine two consecutive hourly DPAs into one file (we need DPAs closest to the top of the hour)
- 2) nam_decode: decode the DPAs (Digital Precipitation Arrays)
- 3) nam_prepmetar: read in METAR gauge input and distribute them under each radar umbrella, for “early” Stage II runs
- 4) nam_prephads: read in HADS gauge input and distribute them under each radar umbrella, for “mid” and “late” Stage II runs
- 5) nam_stageii: perform “Stage II” analysis for each radar umbrella
- 6) nam_mosaic: mosaicking the analysis from nam_stageii into a ConUS product
- 7) nam_remap: map the 4km analysis to a 15km grid
- 8) pcprtma_changepds: change the generating process number from 152 (Stage II) to 109 (RTMA products)
- 9) /nwprod/util/exec/tocgrib2: add a TOC Flag Field separator block and WMO Header in front of each GRIB2 RTMA file for AWIPS
- 10) nam_stage4_acc: compute 6-hourly and 24-hourly accumulations of Stage II (this executable is also used by Stage IV)

- Stage IV analysis v1.1.1

-
- 1) nam_stage4_mosaic: mosaic the regional analyses from the RFCs into a ConUS analysis
 - 2) nam_stage4_acc: compute 24h accumulations from the 6-hourly analyses (this executable is also used by Stage II)
 - 3) nam_stage4_chkst3log: check the stage3 log files and write out a daily summary

- Precipitation verification v1.1.1

- 1) verf_precip_brkout_fcst: extract forecast precipitation from a model output file
- 2) verf_precip_brkout_ndas: extract NDAS precipitation field from a model output file
- 3) verf_precip_pcpconform: process precipitation files from various sources to a standard format (parameter type, precipitation unit, time indicator *etc.*)
- 4) nam_cpc2grbrfc8: convert the CPC 1/8 deg daily gauge analysis from binary to GRIB. Create 1,2,3-day “persistence forecasts”. This executable is also used by NDAS.
- 5) nam_stage4_acc: accumulate forecast and analysis precipitation into time intervals required for the verification. This executable is also used by Stage II and IV.
- 6) verf_precip_diffpcp: give two precipitation accumulations that are valid from T0 to T1 and T0 to T2 ($T2 > T1 > T0$), subtract the former from the latter to get accumulation from T1 to T2.
- 7) verf_precip_verfgen: compute verification statistics
- 8) verf_precip_average: compute the average of two or more model precipitation forecast files (for the MEDLEY forecast – an average of operational models)

3.2.9 Real Time Mesoscale Analysis (RTMA)

See above for the precip analysis component of RTMA

RTMA (Real Time Mesoscale Analysis) – all v2.1.0 from Manuel Pondeca

```
FIRSTGUESS : prepares a GSI-compliant first guess from downscaled model fields
GSIANL     : the GSI analysis step
POST       : post-processor
```

Note: The RTMA is unified such that these steps are identical for all five RTMA implementations (CONUS, Alaska, Hawaii, Puerto Rico and Guam).

3.2.10 Doppler Radar (aka 88D or NEXRAD) Observation Processing & Mosaic

Radar Obs Processing – from Shun Liu

```
wsr88d_level2 : Level-II radar data quality control opnl v1.0.0
radar_reflectivity_mosaic : 3D reflectivity mosaic opnl v1.0.0
radar_reflectivity_ref2grb : reformat binary 3D reflectivity to GRIB RFCed v1.0.0
```

3.2.11 Observations Processing and Quality Control

Obs Processing – use v1.0.0 from Dennis Keyser

Because scripts get RFCed separately, they are listed here along with the applications.

SATELLITE INGEST:

- Job Scripts: (these all set variables to ingest various satellite data types into the /dcom database)
 - o JIAEROSOL.sms.prod (copies aerosol, green fraction vegetation and smoke files; ingests MODIS AOD data into BUFR)
 - o JIAIRS.sms.prod (ingests AIRS, AMSR-E and IASI data into BUFR)
 - o JIAVHRR.sms.prod (ingests AVHRR/GAC data into BUFR)
 - o JIERS.sms.prod (ingests ERS RA and SAP data into BUFR)
 - o JIGOES_RADSND.sms.prod (ingests GOES 1x1 sounder data, GOES 1x1 cloud retrievals and GOES 11x17 imager data into BUFR)
 - o JIGOES_SST.sms.prod (ingests Navy 24-hr avg GOES SST data into BUFR)
 - o JIOZONE14.sms.prod (copies daily BUFR SBUV file)
 - o JIOZONE_ORBIT.sms.prod (ingests SBUV, GOME-2 and OMI data into BUFR)
 - o JIPOES_SST.sms.prod (ingests NAVO and NESDIS POES SST data into BUFR – both provider data and NCEP-generated physical retrievals)
 - o JIQST.sms.prod (ingests ASCAT data into BUFR)
 - o JISMI.sms.prod (ingests SSM/I EDR, SSM/I SDR and SSM/IS UPP data into BUFR)
 - o JISNOW.sms.prod (copies daily NESDIS and USAF snow and ice files)
 - o JISSTFLD.sms.prod (copies daily NESDIS SST files)
 - o JISWD.sms.prod (ingests GOES and MODIS satellite-derived wind data into BUFR)
 - o JITMI.sms.prod (ingests TRMM/TMI data into BUFR)
 - o JITOV.sms.prod (ingests ATOVS product and 1B data into BUFR)
 - o JIWINDSAT.sms.prod (ingests WindSAT data into BUFR)
- Model Scripts: (executed by job scripts)
 - o existday.sh.sms (handles statically-named files, same name every day)
 - o existstore.sh.sms (handles dynamically-named files, files names with date-time stamp in them)
- USH Scripts: (executed by model scripts)
 - o ingest_avhrr.sh (copies green fraction vegetation files)
 - o ingest_check_lapsed_data (checks for lapses in the transfer and processing of data files from remote unix server)
 - o ingest_get (transfers a file from a remote unix machine to the NCEP CCS)
 - o ingest_gross_window_orbits (checks to see if a file on a remote server is within the specified time window)
 - o ingest_process_days (initiates the transfer of one requested, statically-named, file from a remote unix machine)
 - o ingest_process_onetype_newdays (determines whether the current day and time are consistent with the availability of an updated, statically-named, file from a remote unix machine)
 - o ingest_process_onetype_neworbits (determines the existence of yet-to-be processed files in one family of time-stamped files from a remote unix machine)

-
- ingest_process_orbits (initiates the transfer and processing of time-stamped files from a remote unix machine)
 - ingest_query (determines the availability of a group of files on a remote unix machine)
 - ingest_script_atovs1b.sh (ingests ATOVS product and 1B data into BUFR)
 - ingest_script_omi.sh prod (ingests OMI data into BUFR)
 - ingest_sncvgrib.sh (converts NESDIS IMS 16th mesh Global snow/sea-ice from ascii to grib)
 - ingest_sncvgrib96.sh (converts NESDIS IMS 96th mesh Global snow/sea-ice from ascii to grib)
 - ingest_snodepgr.sh (converts USAF 8th mesh N. Hem. snow-depth/sea-ice from binary to grib)
 - ingest_translate_orbits (runs some type of translation processing on time-stamped data files previously received from a remote unix machine)
 - ingest_transst_physstret.sh (ingests NAVO and NESDIS POES SST data into BUFR – both provider data and NCEP-generated physical retrievals)
 - tranjb (appends NCEP-BUFR file to /dcom database tank)
 - PROGRAMS: (most executed by USH scripts)
 - bufr_satsplit (splits POES SST data into files containing data from separate satellites)
 - bufr_sno16grb (converts NESDIS IMS 16th mesh Global snow/sea-ice from ascii to grib)
 - bufr_sno8grb (converts USAF 8th mesh N. Hem. snow-depth/sea-ice from binary to grib)
 - bufr_sno96grb (converts NESDIS IMS 96th mesh Global snow/sea-ice from ascii to grib)
 - bufr_tranamsua (ingests ATOVS 1B AMSU-A data into BUFR)
 - bufr_tranamsub (ingests ATOVS 1B AMSU-B data into BUFR)
 - bufr_tranavhrr (ingests AVHRR/GAC data into BUFR)
 - bufr_traners2 (ingests ERS RA and SAP data into BUFR)
 - bufr_trangoescld (ingests GOES 1x1 cloud retrievals into BUFR)
 - bufr_trangoessst (ingests Navy 24-hr avg GOES SST data into BUFR)
 - bufr_tranhirs3 (ingests ATOVS 1B HIRS-3/-4 data into BUFR)
 - bufr_tranimgr (ingests GOES 11x17 imager data into BUFR)
 - bufr_tranjb (appends NCEP-BUFR file to /dcom database tank)
 - bufr_tranmhs (ingests ATOVS 1B MHS data into BUFR)
 - bufr_tranmtypsbt (changes the BUFR message type and subtype prior to tranjb)
 - bufr_tranomi (ingests OMI data into BUFR)
 - bufr_tranpoessst (ingests NAVO and NESDIS POES SST data from provider into BUFR)
 - bufr_tranquik (ingests ASCAT data into BUFR)
 - bufr_transatw (ingests GOES and MODIS satellite-derived wind data into BUFR)
 - bufr_transsmi (ingests SSM/I EDR and SSM/I SDR data into BUFR)
 - bufr_transsnd (ingests GOES 1x1 sounder data into BUFR)
 - bufr_trantmi (ingests TRMM/TMI data into BUFR)
 - bufr_tranwindsat (ingests WindSAT data into BUFR)

DUMP PROCESSING:

- Job Scripts: (these all set variables to dump data in the various NCEP networks)
 - o JAIRNOW_DUMP.sms.prod (dumps AIRNOW data)
 - o JCDAS_DUMP.sms.prod (dumps data for CDAS network)
 - o JDUMP_MONITOR.sms.prod (dumps data for monitor DUMP network)
 - o JGDAS_DUMP.sms.prod (dumps data for GDAS network)
 - o JGFS_DUMP.sms.prod (dumps data for GFS network)
 - o JNAM_DUMP.sms.prod (dumps data, excluding NEXRAD, for NAM network)
 - o JNAM_DUMP2.sms.prod (dumps NEXRAD data for NAM network)
 - o JNDAS_DUMP.sms.prod (dumps data, excluding NEXRAD, for NDAS network)
 - o JNDAS_DUMP2.sms.prod (dumps NEXRAD data for NDAS network)
 - o JRTMA_DUMP.sms.prod (dumps data for RTMA network)
 - o JRUC2A_DUMP.sms.prod (dumps data for RUC network)
 - o JSRUC_DUMP.sms.prod (dumps data for RUCS network)
- Model scripts: (executed by job scripts)
 - o exairnow_dump.sh.sms (dumps “airnow” ozone concentration data and then reprocesses it into a PREPBUFR look-a-like file)
 - o exdump_monitor.sh.sms (dumps data for monitor DUMP network)
 - o exglobal_dump.sh.sms (dumps data for monitor CDAS, GDAS and GFS networks)
 - o exnam_dump.sh.sms (dumps data, excluding NEXRAD, for NAM and NDAS networks)
 - o exnam_dump2.sh.sms.old (dumps NEXRAD data for NAM and NDAS networks)
 - o exrtma_dump.sh.sms (dumps data for RTMA network)
 - o exruc2_dump.sh.sms (dumps data for RUC network)
 - o exsfcruc_dump.sh.sms (dumps data for RUCS)
- USH Scripts: (executed by model scripts)
 - o bufr_dump_obs.sh (driver for dumpjb)
 - o dumpjb (performs the data dumping)
- PROGRAMS: (most executed by USH scripts)
 - o bufr_chkbfr (checks a list of bufr files (all in the same group, usually all of the same message type and always all using the same dictionary messages), determining whether they contain any data or not)
 - o bufr_combfr (concatenates individual bufr files into a single BUFR file)
 - o bufr_dcodwindsat (reprocesses dumped WindSAT data - performs a number of q.c. checks - reports passing checks may be superobed)
 - o bufr_dumpmd (dumps data from a database file or files which fall within a user supplied time window, by looking only at the message date in the BUFR section one header records)
 - o bufr_dupair (duplicate checks aircraft data across all “aircft” dump types)
 - o bufr_dupcor (duplicate-checks and time-filters all dump types where this is not done by some other dup-check code, i.e., this is the default code)
 - o bufr_dupmar (duplicate-checks, time-filters, and merges “parts” for single level surface marine, METAR and mesonet reports)

-
- bufr_dupmrg (duplicate-checks, time-filters, and merges “parts” for multi-level RAOB, PIBAL and dropsonde reports)
 - bufr_duprad (duplicate-checks and time-filters WSR-88D NEXRAD radial wind and reflectivity reports)
 - bufr_dupsat (duplicate-checks, time-filters, and geographically files all satellite data types)
 - bufr_dupsst (duplicate-checks and time-filters SST data types)
 - bufr_edtbfr (applies real-time interactive quality control flags, generated from either a "reject" list maintained by NCO or from decisions made by the SDM – applies to surface land, surface marine, upper-air, aircraft 004) or satellite-derived wind reports as they are being dumped)
 - bufr_geofil (geographically filters non-satellite data by either a lat/lon box filter or a lat/lon grid point mask file)
 - bufr_quipc (applies real-time interactive quality control and duplicate flags and corrections generated by NCEP/OPC (using "CREWSS" software) to surface marine ship, buoy, c-man platform or tide gauge data as they are being dumped)
 - bufr_raddate (computes the endpoints of a dump time window given a center point and radius)
 - bufr_supertmi (reprocesses dumped TRMM data - performs a number of q.c. checks - reports passing checks may be superobed)
 - prepobs_prepanow (reprocesses the dump of "airnow" ozone concentration data into a PREPBUFR look-a-like file)
 - wave_dcodquikscat (reprocesses dumped ASCAT data - performs a number of q.c. checks - reports passing checks may be superobed)

DUMP POST-PROCESSING:

- Job Scripts: (these all set variables to perform the postprocessing of dump data in the various NCEP networks)
 - JCDAS_DUMP_POST.sms.prod (postprocesses dumped data for CDAS network)
 - JDUMP_ALERT.sms.prod
 - JGDAS_DUMP_POST.sms.prod (postprocesses dumped data for GDAS network)
 - JGFS_DUMP_POST.sms.prod (postprocesses dumped data for GFS network)
 - JNAM_DUMP_POST.sms.prod (postprocesses dumped data for NAM network)
 - JNDAS_DUMP_POST.sms.prod (postprocesses dumped data for NDAS network)
 - JRTMA_DUMP_POST.sms.prod (postprocesses dumped data for RTMA network)
 - JRUC2A_DUMP_POST.sms.prod (postprocesses dumped data for RUC network)
 - JSRUC_DUMP_POST.sms.prod (postprocesses dumped data for RUCS network)
- Model scripts: (executed by job scripts)
 - exdump_post.sh.sms (if requested, 1) generates combined dump STATUS file; 2) prepares data counts for the SDM/RTDMS; 3) removes or masks restricted data from dump files; 4) converts dump files to unblocked format; 5) lists the contents of dump files; 6) updates dump data count average tables)
- USH Scripts: (executed by model scripts)

-
- bufr_avgdata.sh (generates a table containing the average counts for each BUFR type/subtype dumped in a particular network by cycle over a period of time, normally 30-days)
 - bufr_datacount.sh (produces a data count report from the data dump jobs for all types of data in the current dump – if a data count is deemed deficient, an alert is sent to the SDM and the RTDMS)
 - bufr_remorest.sh (reads through an input dump BUFR file which can contain a mixture of reports which are unrestricted or restricted and either writes out only those reports which are unrestricted or writes out all reports but with masked report id's for those reports which are restricted)
 - PROGRAMS: (executed by USH scripts)
 - bufr_avgdata (generates a table containing the average counts for each BUFR type/subtype dumped in a particular network by cycle over a period of time, normally 30-days)
 - bufr_datacount (produces a data count report from the data dump jobs for all types of data in the current dump – if a data count is deemed deficient, an alert is sent to the SDM and the RTDMS)
 - bufr_listdumps (lists contents of a time-windowed BUFR data dump file)
 - bufr_remorest (reads through an input dump BUFR file which can contain a mixture of reports which are unrestricted or restricted and either writes out only those reports which are unrestricted or writes out all reports but with masked report id's for those reports which are restricted)

TROPICAL CYCLONE QC and RELOCATION:

- Job Scripts: (these all set variables to perform tropical cyclone q.c. and relocation in the various NCEP networks)
- JGDAS_TROPCY_QC_RELOC.sms.prod (performs tropical cyclone vitals Q.C. followed an attempt to relocate tropical cyclones in the Global guess fields in the GDAS network)
 - JGFS_TROPCY_QC_RELOC.sms.prod (performs tropical cyclone vitals Q.C. followed an attempt to relocate tropical cyclones in the Global guess fields in the GFS network)
 - JNAM_TROPCY_QC_RELOC.sms.prod (performs tropical cyclone vitals Q.C. followed an attempt to relocate tropical cyclones in the Global guess fields in the NAM network)
 - JNDAS_TROPCY_QC.sms.prod (performs tropical cyclone vitals Q.C. in the NDAS network)
 - JNDAS_TROPCY_RELOC.sms.prod (attempts to relocate tropical cyclones in the Global guess fields in the NDAS network)
- Model scripts: (executed by job scripts)
 - extropcy_qc_reloc.sh.sms (performs tropical cyclone vitals Q.C. followed by GBL guess relocation in various networks)
- USH Scripts: (executed by model scripts except where noted)
 - syndat_getjbul.sh (executed by syndat_qctropcy.sh – merges JTWC tropical cyclone records which can be split into two parts)

-
- syndat_qctropcy.sh (Q.C.'s tropical cyclone vitals files made by TPC and other tropical centers)
 - tropcy_relocate.sh (attempts to relocate tropical cyclones in the Global guess fields)
 - PROGRAMS: (executed by USH scripts)
 - gettrk (I am not responsible for this program)
 - relocate_mv_nvortex (I am not responsible for this program)
 - supvit (I am not responsible for this program)
 - syndat_getjtbl (merges JTWC tropical cyclone records which can be split into two parts)
 - syndat_qctropcy (Q.C.'s tropical cyclone vitals files made by TPC and other tropical centers)

PREPBUFR PROCESSING

- Job Scripts: (these all set variables to perform PREPBUFR processing and Q.C. in the various NCEP networks)
 - JCDAS_PREP1.sms.prod (performs PREPBUFR processing in the CDAS network – no Q.C. is performed)
 - JCDAS_PREP2.sms.prod (performs PREPBUFR processing in the CDAS network – only Q.C. is performed)
 - JGDAS_PREP.sms.prod (performs PREPBUFR processing and Q.C. in the GDAS network)
 - JGFS_PREP.sms.prod (performs PREPBUFR processing and Q.C. in the GFS network)
 - JNAM_PREP.sms.prod (performs PREPBUFR processing and Q.C. in the NAM network)
 - JNDAS_PREP.sms.prod (performs PREPBUFR processing and Q.C. in the NDAS network)
 - JRTMA_PREP.sms.prod (performs PREPBUFR processing in the RTMA network – no Q.C. is performed)
 - JRUC2A_PREP.sms.prod (Part 1: performs PREPBUFR processing and Q.C. in the RUC network; Parts 2-?: I am not responsible for these)
- Model scripts: (executed by job scripts)
 - exglobal_makeprepbufr.sh.sms (performs PREPBUFR processing and Q.C. in the CDAS, GDAS and GFS networks)
 - exnam_makeprepbufr.sh.sms (performs PREPBUFR processing and Q.C. in the NAM and NDAS networks)
 - exrtma_makeprepbufr.sh.sms (performs PREPBUFR processing and Q.C. in the RTMA network)
 - exruc2_prep.sh.sms (Part 1: performs PREPBUFR processing and Q.C. in the RUC network; Part 2-?: I am not responsible for these)
- USH Scripts: (executed by model scripts)
 - prepobs_acarsqc.sh (performs MDCRS aircraft quality control checking in the PREPBUFR processing)

-
- prepobs_cqcbufr.sh (performs rawinsonde upper-air complex quality control checking of heights and temperatures in the PREPBUFR processing; also performs radiation corrections)
 - prepobs_cqcvad.sh (performs VAD wind complex quality control checking in the PREPBUFR processing)
 - prepobs_makeprepbufr.sh (driver script to perform PREPBUFR processing and Q.C. in the various NCEP networks)
 - prepobs_oiqcbufr.sh (performs an OI-based quality control on all data in the PREPBUFR processing, GBL networks only)
 - prepobs_prepacqc.sh (performs non-MDCRS aircraft quality control checking in the PREPBUFR processing)
 - prepobs_prevents.sh [encodes the background (first guess) and observational errors into the PREPBUFR reports, interpolated to obs locations, reforms rudimentary Q.C., CDAS network only]
 - prepobs_profqc.sh (performs wind profiler quality control checking in the PREPBUFR processing)
 - prepobs_syndata.sh [generates synthetic cyclone bogus near tropical storms and appends them to a PREPBUFR file (always in NAM/NDAS, only for weak storms in GFS/GDAS) ; may also flag mass pressure reports "near" tropical storms; may also flag dropwinsonde wind reports "near" tropical storms)
 - PROGRAMS: (executed by USH scripts)
 - prepobs_acarsqc (performs MDCRS aircraft quality control checking in the PREPBUFR processing)
 - prepobs_cqcbufr (performs rawinsonde upper-air complex quality control checking of heights and temperatures in the PREPBUFR processing; also performs radiation corrections)
 - prepobs_cqcvad sh (performs VAD wind complex quality control checking in the PREPBUFR processing)
 - prepobs_listheaders (generates a list of all unique headers in a PREPBUFR file matching input list, reorders the file into the input list order – used as part of multi-tasking of PREPBUFR processing into 12 parts that all run simultaneously)
 - prepobs_monoprepbufr (merges multiple PREPBUFR files into a monolithic PREPBUFR file, groups like-named table a messages together in the expected order – used as part of multi-tasking of PREPBUFR processing into 12 parts that all run simultaneously)
 - prepobs_mpcopybufr [reads through user-specified number of input BUFR data dump files and copies selected messages from each into unique output BUFR data dump subset files (1-1 input/output) - used as part of multi-tasking of PREPBUFR processing into 12 parts that all run simultaneously]
 - prepobs_oiqcbufr (performs an OI-based quality control on all data in the PREPBUFR processing, GBL networks only)
 - prepobs_prepacqc (performs non-MDCRS aircraft quality control checking in the PREPBUFR processing)

-
- prepobs_prepdata (reads in dump files and converts them to PREPBUFR, performs rudelmentart Q.C., performs the function of PREPOBS_PREVENTS in non-CDAS networks)
 - prepobs_prevents [encodes the background (first guess) and observational errors into the PREPBUFR reports, interpolated to obs locations, reforms rudimentary Q.C., CDAS network only]
 - prepobs_profqc (performs wind profiler quality control checking in the PREPBUFR processing)

PREPBUFR POST-PROCESSING

- Job Scripts: (these all set variables to perform the postprocessing of PREPBUFR data in the various NCEP networks)
 - JCDAS_PREP1_POST.sms.prod prod (postprocesses non-QC'd PREPBUFR data for CDAS network)
 - JCDAS_PREP2_POST.sms.prod (postprocesses QC'd PREPBUFR data for CDAS network)
 - JGDAS_PREP_POST.sms.prod prod (postprocesses QC'd PREPBUFR data for GDAS network)
 - JGFS_PREP_POST.sms.prod prod (postprocesses QC'd PREPBUFR data for GFS network)
 - JNAM_PREP_POST.sms.prod prod (postprocesses QC'd PREPBUFR data for NAM network)
 - JNDAS_PREP_POST.sms.prod prod (postprocesses QC'd PREPBUFR data for NDAS network)
 - JRTMA_PREP_POST.sms.prod prod (postprocesses PREPBUFR data for RTMA network)
 - JRUC2A_PREP_POST.sms.prod prod (postprocesses QC'd PREPBUFR data for RUC network)
- Model scripts: (executed by job scripts)
 - exprep_post.sh.sms [runs various post-analysis processing steps on the PREPBUFR files: if requested 1) removes or masks restricted data; 2) identifies "TimeTwin" duplicate reports (GDAS only) - I am not responsible for this; 3) generates a table of GDAS received, selected, and assimilated data counts (18Z GDAS only) - I am not responsible for this]
- USH Scripts: (executed by model scripts)
 - bufr_remorest.sh (reads through an input PREPBUFR file which can contain a mixture of reports which are unrestricted or restricted and either writes out only those reports which are unrestricted or writes out all reports but with masked report id's for those reports which are restricted)
- PROGRAMS: (executed by USH scripts)
 - bufr_remorest (reads through an input PREPBUFR file which can contain a mixture of reports which are unrestricted or restricted and either writes out only those reports which are unrestricted or writes out all reports but with masked report id's for those reports which are restricted)

4.3 Marine Modeling and Analysis Branch

Ocean models in NCO production suite.

Main ocean model systems:

- RTOFS-Atlantic: After April 2010 Jason-2 upgrade, we should be on version number 2.4.1, with the initial implementation in December 2005 as version 1.0.0 and the major SSH assimilation upgrade in June 2007 as version 2.0.0. Since June 2007, we have had 4 minor upgrades up to April 2010.
- HY-HWRF: This system is yet to be implemented, and as the first of this kind should be version 1.0.0 (HYCOM in coupled model). The overall model version of the coupled system is to be determined by the HWRF group.
- RTOFS-Global: This system is yet to be implemented, and as the first of this kind should be version 1.0.0.

All the above ocean models have different HYCOM executables. Version numbers of these should be consistent with the version numbering of releases of this community model.

Scripts (“ex” scripts and “ush” scripts): If and how these scripts need to be version numbered depends critically on the way in which the model systems are stored and maintained in the file system and in svn. If each major system is stored in its own directory with its own (implicit or explicit) version number, then there is little reason to add individual version numbers to individual scripts, with the exception of utility scripts that are used across model systems. If the present file system with one “ush” directory for all systems is retained, then ALL SCRIPTS will need to be version numbered, as this will be the only way to keep track of them.

- Case I: existing directory structure: all “ex” and “ush” scripts will get a version number, tentatively associated with the version number of the main system they are associated with, but not critical. Shared “ush” scripts will get their own version number, tentatively starting with 1.0.0.
- Case II: New storage structure: Only shared “ush” scripts need version number, tentatively starting with 1.0.0. Version number of model system will be proxy for all other scripts (“ex” and “ush”).

Fixed files: The fixed files for the ocean models are very large and as such should be stored on HPSS. One way to “tag” these files would be to provide version numbers to these files and to retain README files which provide a listing of these fixed files.

Parm files: These can be stored with the scripts of the model systems or with the source code for the corresponding executables. In either case they will not require separate version numbers.

It is important to note that the choices of directory structure, using version numbers in names and so on are closely related to the features of particular revision control software (SVN, CVS, MKS, etc), making software management easy, robust and user-friendly.

Wave models in NCO production suite.

Main wave model systems:

- Global multi-grid wave model; after May 2010 upgrade we should be on version number 2.0.0, as added grids represent first major upgrade since 2007 initial implementation.
- Multi-grid hurricane wave model is in queue to be implemented, and as the first of this kind should be 1.0.0.
- Global wave ensemble is in second major version 2.0.0.
- Great Lakes NAM driven wave model is at version 2.0.0 planned upgrade for this year should be 2.1.0.
- NDFD driven Great Lakes model is at 1.0.0. Next planned version upgrade together with hurricane model would become 2.0.0, if only bug fix goes in, it becomes 1.0.1.
- Legacy wave model systems (NAH, NPH, NWW3) should disappear with multi-grid hurricane model implementation and implementation of data assimilation and their versioning number is irrelevant.

All wave models share various executables from the publicly released WAVEWATCH III[®] model, and use many other FORTRAN codes that may or may not be shared. Note that not all systems change the executables in lockstep, so that several version of the executable may need to be maintained.

- WAVEWATCH III codes should be renamed in NCO ops to the names used in the public model release, and “x” and “y” version number should coincide with the major

and minor version numbers of WAVEWATCH III. That would make the version number of the present WAVEWATCH III codes 3.14.0 (2.22.0 for legacy codes).

- All other FORTRAN executables should be numbered, but with no clear preference for numbering (might as well be 1.0.0)

Scripts (“ex” scripts and “ush” scripts): Same considerations as for ocean model above.

Fixed and other files. See above; if the present data structure is retained, EVERYTHING will need a version number. If not, the following approach is preferred.

- Buoy files are shared by multiple model systems, and require their own version number. Since we are on the second reincarnation of these files, they need to start with version number 2.0.0.
- Template model input files can be stored with the scripts of the model systems or with the source code for the corresponding executables. In either case they will not require separate version numbers.
- Masks and parm files can be stored with the scripts for the model systems and then also will not need a separate version number.

SST Analysis systems

There are 4, soon to be 5, free-standing sea surface temperature analysis systems. Two are the RTG (Real Time Global) and 2 (to be 3) are OI (optimal interpolation) family.

All 5 systems rely strongly on both their fixed fields and control scripts, such that the fixed fields should be contained with the executables. And control scripts merit their own versions. Switches of satellite systems (as for the soon-to-be-implemented switch from NOAA-18 to NOAA-19) can be carried out solely by modification to control scripts.

- RTG low resolution analysis currently (5/13/2010) operational should be versioned 1.6.0
- RTG high resolution analysis currently operational should be labelled version 2.7.0
- The daily OI analysis currently operational should be version 1.1.1

-
- The weekly OI analysis when implemented operationally should be version 2.0.0
 - The quarter degree OI analysis when implemented should be versioned 3.0.0

Data file (climatologies, correlation scales, ...) and component executables and control scripts will warrant their own sequence numbers, but these are still to be determined.

Gulf Stream Finder – RTOFS-GSF

The current operational Gulf Stream Finder is version 2.0.0. When the current RFC is implemented, it will be version 2.0.1 – the change to the system is a code maintenance fix, with a catch up for an error in management of files during operational system failover.

This system has no fixed files outside of the MMABlib. No reason to version number the operational script. There is a utility script, `gsf_parse.pl`, which should be version 1.0.0.

Sea Ice Drift Model in the NCO production suite

The sea ice drift model is simple, with only a few ancillary fields and only 2 executables.

- The drift model itself is now version 2.0.0, and will be 2.0.1 when the current RFC is implemented (support for the T574 gfs).
- The input averaging is now version 1.0.0 and will be 1.0.1 when the

current RFC is implemented (support for T574 GFS)

- Fixed files are version 1.0.0 and need no real attention or versioning – they never change, and have no reason to change in the future.

Sea Ice Concentration analysis in the NCO production suite

The overall sea ice concentration analysis suite is now at version 3.1.0

- SSMI concentration analysis program is version 3.0.1
- AMSR concentration analysis program is version 1.0.0
- Multisatellite blending program is version 1.0.0
- Sea ice grid resolution reduction programs are version 1.0.0 – all are built from the same source, with different compile-time options.
- XPM graphics programs are version 1.0.0 – all are built from the same source, with different compile-time options.
- Programs to put the output into grib and wmo format are version 1.1.0
- SST filtering is version 1.1.0
- Polar stereographic to lat-long grid conversion is 1.1.0

Land mask files require version numbers as well, due to both internal and downstream effects. The current high resolution global grid is version 3.1.0, while the polar stereographic grids (both low and high resolution) and global low resolution grid is version 3.0.0.

Global Marine Visibility and Vessel Icing

These are two separate legacy systems. Both have been in use for a long time, and are simple systems. Both may be versioned at 5.0.0 currently.

MMABlib

Library for MMAB functions and the MMAB C++ class library.

With the implementation of the RFC in support of the T574 GFS, this will be version 2.0.0.

The library itself has some fixed fields and data files, which are versioned with the library as a whole.

There are no scripts associated with the library.

4.4 Global Climate and Weather Modeling Branch

- global_postevents
- global_mrf_look_alike
- global_sfchr
- global_satcount
- global_chgres
- global_chgres_thread
- global_sighdr
- global_gsi
- global_fcst
- global_cycle
- global_angupdate
- ncep_post
- overparm_grib
- genpsiandchi
- relocate_mv_nvortex
- global_chg_igen
- global_sig_igen

4.5 Miscellaneous Software or Support Units

- anomgb
- grbindex
- cnvgrb
- copygb
- ndate
- supvit
- gettrk
- wgrib
- wgrib2
- BACIO library
 - The bacio library is a general utility library for byte-addressable C i/o. The current operational version is 1.3.0, and the version to be implemented with GSI/GFS will be version 2.0.0 – adding capabilities for managing files over 2 Gb in size.