EnKF Ensemble Kalman Filter

User's Guide Version 1.1

- Compatible with GSI community release v3.5

August 2016

Hui Liu
National Center for Atmospheric Research (NCAR)

Ming Hu
National Oceanic and Atmospheric Administration (NOAA)/Earth System Research Laboratory
Cooperative Institute for Research in Environmental Sciences (CIRES)

Hui Shao, Don Stark, Kathryn Newman
NCAR

Jeff Whitaker
NOAA/Earth System Research Laboratory

DTC Developmental Testbed Center

Acknowledgement

We thank the National Oceanic and Atmospheric Administration (NOAA) Hurricane Forecast Improvement Program (HFIP) and Office of Oceanic and Atmospheric Research (OAR) for their support of this work. This work is also facilitated by the National Center for Atmospheric Research (NCAR). NCAR is supported by the National Science Foundation (NSF).

Forward

This User's Guide for the community ensemble Kalman filter (EnKF) data analysis system is particularly geared for beginners. It describes the fundamentals of using EnKF, including basic skills of installing, running, diagnosing, and tuning EnKF. EnKF version (v) 1.1 was released in July 2016. This version of code is compatible with the Gridpoint Statistical Interpolation (GSI) analysis system community release v3.5.

This User's Guide includes six chapters and one appendix:

Chapter 1 provides a background introduction of the EnKF operational and community system, EnKF review committee, and data types that can be used in this version.
Chapter 2 contains basic information about how to get started with EnKF, including system requirements; required software (and how to obtain it); how to download EnKF; and information about compilers, libraries, and how to build the code.
Chapter 3 focuses on the input files needed to run EnKF and how to configure and run GSI observer and EnKF through a sample run script. This chapter also provides an example of a successful EnKF run.
Chapter 4 includes information about diagnostics and tuning of the EnKF system through EnKF standard output and namelist variables.
Chapter 5 illustrates how to setup and run the GSI observer and EnKF for a regional configuration and a global configuration, as well as how to diagnose the results.
Chapter 6 introduces EnKF theory and the main structure of the code.
Appendix A describes the contents of the EnKF namelist.

This document is updated annually. For the latest version of this document and annual released code, please visit the EnKF User's Website:

`http://www.dtcenter.org/EnKF/users/index.php`

Please send questions and comments to the EnKF help desk:

enkf-help@ucar.edu

This document and the annual EnKF releases are made available through a community EnKF effort led by the Developmental Testbed Center (DTC), in collaboration with EnKF developers. To help sustain this effort, we encourage for those who use the community released EnKF, the EnKF helpdesk, the EnKF User’s Guide, and the other DTC EnKF services, please refer to this user’s guide in their work and publications.

Citation:

Liu, H., M. Hu, D. Stark, H. Shao, K. Newman, and J. Whitaker, 2016: Ensemble Kalman Filter (EnKF) User's Guide Version 1.1. Developmental Testbed Center. Available at http://www.dtcenter.org/EnKF/users/docs/index.php, 80 pp.

Contents

1 Overview

1.1 EnKF History and Background

The ensemble Kalman filter (EnKF) is a Monte-Carlo algorithm for data assimilation that uses an ensemble of short-term forecasts to estimate the background-error covariance in the Kalman filter. Each ensemble member is cycled through the data assimilation system and updated by EnKF. The EnKF code was developed by the National Oceanic and Atmospheric Administration (NOAA) Earth System Research Lab (ESRL) in collaboration with the research community. It contains two separate algorithms for calculating an analysis increment, a serial Ensemble Square Root Filter (EnSRF) algorithm described by [6] and a Local Ensemble Kalman Filter (LETKF) algorithm described by [4]. The parallelization scheme used by the EnSRF algorithm is based on that used in the Data Assimilation Research Testbed (DART) toolkit developed at the National Center for Atmospheric Research (NCAR) and described by [1]. The LETKF code was contributed by Yoichiro Ota of the Japanese Meteorological Agency (JMA) while he was a visitor at the National Centers for Environmental Prediction (NCEP).

The EnKF code became an operational data assimilation system at NCEP in May 2012, providing ensemble update to the Global Forecast System (GFS) hybrid Ensemble-Variational (EnVar) data assimilation system. It can work with both global forecast models (e.g., GFS) and regional forecast models (e.g., the Hurricane Weather Research and Forecasting (WRF) (HWRF) model, the North American Mesoscale (NAM) system, the Advanced Research WRF (ARW) system).

1.2 EnKF Becomes Community Code

The Developmental Testbed Center (DTC), in collaboration with major development groups, began transforming the EnKF operational system into a community system in 2014, following the same protocol as the GSI community effort ([5], `http://www.dtcenter.org/com-GSI/users/`). Consequently, the EnKF code and its user support are managed by the DTC, along with the GSI community system.

The DTC complements the development groups in providing EnKF documentation, porting EnKF to multiple platforms, and testing EnKF in an independent and objective environment, while still maintaining functionally equivalent to operational centers. Working with code developers, the DTC is maintaining a community GSI/EnKF repository, which is equivalent to the operational developmental repository, and facilitates community users to develop EnKF. Based on the repository, the DTC releases EnKF code annually with GSI. The first community version of the EnKF system was released on July 31, 2015. This user's guide describes the second release of EnKF (v1.1) in July 2016. The DTC provides user support through the EnKF Helpdesk (enkf-help@ucar.edu), tutorials, and workshops. More information about the EnKF community services can be found at the DTC EnKF webpage (`http://www.dtcenter.org/EnKF/users/`).

1.2.1 EnKF Code Management and Review Committee

The EnKF code development and maintenance are administrated by the Data Assimilation Review Committee (DARC). DARC was originally formed as the GSI Review Committee in 2010. The committee was reformed in 2014 to include members representing the EnKF development and applications. Such a combination enhanced collaboration of development groups in both variational and ensemble data assimilation communities. Currently, DARC contains members from NCEP's Environmental Modeling Center (EMC), the National Aeronautics and Space Administration (NASA) Goddard Global Modeling and Assimilation Office (GMAO), NOAA/ESRL, the National Center for Atmospheric Research (NCAR) Mesoscale & Microscale Meteorology Laboratory (MMM), the National Environmental Satellite, Data, and Information Service (NESDIS), the Unite States Air Force (USAF), the University of Maryland, and the DTC (chair).

DARC primarily steers distributed GSI/EnKF development and community code management and support. The responsibilities of the committee are divided into two major aspects: coordination and code review. The purpose and guiding principles of the review committee are as follows:

Coordination and Advisory -

- Propose and shepherd new development
- Coordinate on-going and new development
- Establish and manage a code review and transition process
- Community support recommendation

Code Review -

- Establish and manage a unified coding standard followed by all GSI/EnKF developers
- Review proposed modifications to the code trunk
- Make decisions on whether code change proposals are accepted or denied for inclusion in the repository and manage the repository
- Oversee the timely testing and inclusion of code into the repository

1.2.2 Community Code Contributions

EnKF is a community data assimilation system, open to contributions from scientists and software engineers from both the operational and research communities. DARC oversees the code transition from prospective contributors. This committee reviews proposals for code commits to the GSI/EnKF repository and monitors that coding standards and tests are being fulfilled. Once the committee reaches approval, the contributed code will be committed to the GSI/EnKF code repository and available for operational implementation and public release.

To facilitate this process, the DTC is providing code transition assistance to the general research community. Prospective contributors of code to the EnKF system should contact the DTC EnKF helpdesk (enkf-help@ucar.edu) for the preparation and integration of their code. It is the contributor's responsibility to ensure that a proposed code change is correct, meets the EnKF coding standards, and its expected impact is documented. The DTC will help the contributors run the regression tests and merge the code with the top of the repository trunk. Prospective contributors can also apply to the DTC visitor program for their EnKF research and code transition. The visitor program is open to applications year-round. Please check the visitor program webpage (`www.dtcenter.org/visitors/`) for the latest announcement of opportunity and application procedures.

1.3 About This EnKF Release

This user's guide was composed for the EnKF community release version(v) 1.1. This version of EnKF is compatible with the GSI community release v3.5. Please note the major focuses of the DTC are currently on testing and evaluation of EnKF for regional numerical weather prediction (NWP) applications though the instructions and cases for EnKF global applications are available with this release.

Running this EnKF system requires running GSI a prior for its observation operators. Therefore, the GSI User's Guide is referred throughout this documentation. This GSI user's guide can be obtained at the GSI user's webpage (`http://www.dtcenter.org/com-GSI/users/docs/index.php`).

1.3.1 What Is New in This Release Version

Major updates to this version of EnKF are code optimization, including bug fixes and code cleanup. Added features include new capabilities to update multiple-time background ensembles and use of the NCEP I/O library *nemio*. The observation types assimilated by EnKF were also updated as part of the GSI v3.5 updates.

1.3.2 Observations Used by This Version

EnKF is using the GSI system as the observation operator to generate observation innovations. Therefore, the observation types assimilated by EnKF are the same as GSI. This version of EnKF has been tested to work with the community GSI release v3.5. It can assimilate, but is not limited to, the following types of observations:

Conventional observations (including satellite retrievals):

- Radiosondes
- Pilot ballon (PIBAL) winds
- Synthetic tropical cyclone winds
- Wind profilers: USA, Jan Meteorological Agency (JMA)
- Conventional aircraft reports
- Aircraft to Satellite Data Relay (ASDAR) aircraft reports
- Meteorological Data Collection and Reporting System (MDCRS) aircraft reports
- Dropsondes
- Moderate Resolution Imaging Spectroradiometer (MODIS) IR and water vapor winds
- Geostationary Meteorological Satellite (GMS), JMA, and Meteosat cloud drift IR and visible winds
- European Organization for the Exploitation of Meteorological Satellites (EUMETSAT) and GOES water vapor cloud top winds
- GEOS hourly IR and cloud top wind
- Surface land observations
- Surface ship and buoy observation
- Special Sensor Microwave Imager (SSMI) wind speeds
- Quick Scatterometer (QuikSCAT), the Advanced Scatterometer (ASCAT) and Oceansat-2 Scatterometer (OSCAT) wind speed and direction
- RapidScat observations
- SSM/I and Tropical Rainfall Measuring Mission (TRMM) Microwave Imager (TMI) precipitation estimates
- Doppler radial velocities
- Velocity-Azimuth Display (VAD) Next Generation Weather Radar ((NEXRAD) winds
- Global Positioning System (GPS) precipitable water estimates
- Solar Backscatter Ultraviolet (SBUV) ozone profiles, Microwave Limb Sounder (MLS) (including NRT) ozone, and Ozone Monitoring Instrument (OMI) total ozone
- Sea surface temperature (SST)
- Tropical Cyclone Vitals Database (TCVital)

- Doppler wind Lidar
- Flight level and Stepped Frequency Microwave Radiometer (SFMR) High Density Observation (HDOB) from reconnaissance aircraft
- Tall tower wind

Satellite radiance/brightness temperature observations (instrument/satellite ID):

- SBUV: *NOAA-17, NOAA-18, NOAA-19*
- High Resolution Infrared Radiation Sounder (HIRS): *Meteorological Operational-A (MetOp-A), MetOp-B, NOAA-17, NOAA-19*
- GOES imager: *GOES-11, GOES-12*
- Atmospheric IR Sounder (AIRS): *aqua*
- AMSU-A: *MetOp-A, MetOp-B, NOAA-15, NOAA-18, NOAA-19, aqua*
- AMSU-B: *MetOp-B, NOAA-17*
- Microwave Humidity Sounder (MHS): *MetOp-A, MetOp-B, NOAA-18, NOAA-19*
- SSMI: *DMSP F14, F15, F19*
- SSMI/S: *DMSP F16*
- Advanced Microwave Scanning Radiometer for Earth Observing System (AMSR-E): *aqua*
- GOES Sounder (SNDR): *GOES-11, GOES-12, GOES-13*
- Infrared Atmospheric Sounding Interferometer (IASI): *MetOp-A, MetOp-B*
- Global Ozone Monitoring Experiment (GOME): *MetOp-A, MetOp-B*
- Ozone Monitoring Instrument (OMI): *aura*
- Spinning Enhanced Visible and Infrared Imager (SEVIRI): *Meteosat-8, Meteosat-9, Meteosat-10*
- Advanced Technology Microwave Sounder (ATMS): *Suomi NPP*
- Cross-track Infrared Sounder (CrIS): *Suomi NPP*
- GCOM-W1 AMSR2
- GPM GMI
- Megha-Tropiques SAPHIR
- Himawari AHI

2

Software Installation

2.1 Introduction

The DTC community EnKF is a community distribution of NOAA's operational ensemble Kalman filter. The community EnKF expands the portability of the operational code by adding a flexible build system and providing example run scripts that allow the system to run on many common platforms. The current version of the community EnKF builds and runs on most standard Linux platforms using the Intel, PGI, and GNU compilers.

This chapter describes how to build and install the DTC community EnKF on your computing resources. These instructions apply only to the DTC community EnKF. The source code for the community EnKF release is identical to the particular revision of NCEP's trunk code frozen for community release. The only difference from the NCEP trunk code, is the addition of the more general community build system in order to support a wider variety of computing platforms.

The EnKF build process consists of the four general steps necessary to build GSI.

- Obtain the source code: combined GSI/EnKF, and WRF.
- Build the WRF model (see the WRF users guide).
- Set the EnKF code defaults (see the EnKF users guide)
- Build the GSI and EnKF model (see the GSI users guide).

Section 2.2 describes how to obtain the source code. Section 2.3 presents an outline of the build process. Sections 2.4 and 2.5 cover the system requirements (tools, libraries, and environment variable settings) and currently supported platforms in detail. Section 2.6 discusses what to do if you have problems with the build and where to get help.

2.2 Obtaining the Source Code

The community EnKF code and the GSI code are released as a combined source code package. The current EnKF release is v1.1 and is paired with the community GSI release version 3.5. The community EnKF release is available from the DTC community EnKF users website;

`http://www.dtcenter.org/EnKF/users/index.php`

The community GSI/EnKF release includes the source code for both the EnKF v1.1 and the GSI v3.5 models, as well as an integrated build system, utilities, and documentation necessary to build and run the EnKF.

To download the source code from the either the GSI or the EnKF website, select the Download tab along with the GSI/EnKF System subtab on the vertical menu located on the left side of the main page. New users must first register before downloading the source code. Returning users only need to enter their registration email address to log in. After accessing the download page, select the link to the `comGSIv3.5_EnKFv1.1` tarball. Please only use the source code provided with the `comGSIv3.5_EnKFv1.1` tarball. Do not mix and match this tarball with other versions of the community GSI code or supplemental libraries, as this will lead to unpredictable results.

The community EnKF version 1.1 comes in a tar file named `comGSIv3.5_EnKFv1.1.tar`. The tar file may be unpacked by using the standard UNIX commands:

```
gunzip comGSIv3.5_EnKFv1.1.tar.gz
tar -xvf comGSIv3.5-EnKFv1.1.tar
```

This creates the top level GSI directory `comGSIv3.5_EnKFv1.1/`. After downloading the source code, and prior to building, the user should check the known issues link on the download page of DTC website to determine if any bug fixes or platform specific customizations are needed.

2.3 Compiling EnKF

The EnKF code is built automatically when GSI is built (details in the GSI User's Guide Chapter 2). It makes use of the `configure.gsi` file produced for the GSI build. Therefore to build EnKF, simply build GSI. The default EnKF build settings produce a regional version of EnKF, called wrf_enkf. This version of EnKF assumes that a WRF style I/O is being used.

This section provides a quick outline of the steps necessary to build the EnKF code from the release distribution. Typically, EnKF will build *straight out of the box* on any system that successfully builds GSI. Should the user experience any difficulties with the default build, check the build environment against the requirements described at the end of section 2.4.

1. Set the environment for the compiler: If not already done so, set the necessary paths for using your selected compiler, such as loading the appropriate modules or modifying the path.
2. If not already done, build and install a recent version of the WRF model. The WRF build is currently needed for the WRF I/O libraries and should use the same compiler as used for the EnKF and GSI builds.
3. Build GSI (see chapter 2 of the GSI users guide for more details)
   a. Set the environment variables (see chapter 2.4.2 of the GSI users guide)
   b. Run the configure script located at the main GSI system directory.
   c. Select the EnKF configuration (the default is regional, see section 2.3.1)
   d. Run the compile script
   e. Confirm that GSI has successfully built.
   f. Change into the directory `src/main/enkf`
   g. Confirm that the EnKF executable resides in the directory (default executable is `wrf_enkf`).

Other I/O configurations for EnKF are available, and must be selected at build time. The current choices are regional, global or NMMB. The choice of I/O configuration is specified in the file `src/main/enkf/Makefile.conf`. Section 2.3.1 provides a full explanation on specifying the I/O configuration.

2.3.1 Versions of EnKF

The EnKF code has three build time configurations for the I/O; regional, global, and NMMB. The EnKF analysis is identical in each case, only the capability to digest model input differs. The regional version can only digest WRF formatted I/O files. The global version can only digest spectral input from the NCEP global model. Lastly, the NMMB version can only digest NMMB files.

Before initiating the compile command, the user must select which EnKF configuration is to be built by manually editing the file src/main/enkf/Makefile.conf. Examining lines 45 to 50 of Makefile.conf shows three pairs of build flags.

```
45  # FFLAGS_F90 = -DGFS
46  # EXE_FILE = global_enkf i
47  FFLAGS_F90 = -DWRF
48  EXE_FILE = wrf_enkf
49  # FFLAGS_F90 = -DNMMB
50  # EXE_FILE = nmmb_enkf
```

By default the build system is set to build the regional configuration. This sets the Fortran preprocessor flag to *-DWRF*, and the executable name to *wrf_enkf*. Other EnKF configurations are selected by commenting out lines by adding # symbols and activating lines by removing # symbols. Once the desired configuration is selected by editing the file *Makefile.conf*, the executable may be built by running the the top level GSI configure/compile commands.

2.4 System Requirements and External Libraries

The EnKF source code is written in FORTRAN 90 and requires some flavor of MPI and OpenMP for the distributed memory parallelism. Lastly the I/O relies on the NetCDF I/O libraries. The build system relies on standard make.

The basic requirements for building are:

- FORTRAN 95+ compiler
- MPI v1.2+
- OpenMP
- NetCDF V3.6.3 or V4.2+
- LAPACK and BLAS mathematics libraries, or equivalent
- WRF V3.5+

Because all but the last of these tools and libraries are typically the purview of system administrators to install and maintain, they are lumped together here as part of the basic system requirements.

2.5 Compilers Tested for Release

Version 1.1 of the DTC community EnKF system has been successfully tested on a variety of Linux platforms with many versions of the Intel, PGI, Gnu Fortran compilers.

The following Linux compiler combinations have been fully tested:

	Fortran compiler version	C compiler version
Intel only	ifort 16.0.1, 15.0.1, 13.0.1, 12.1.5, 12.1.4	icc
Intel & gcc	ifort 16.0.1, 15.0.1, 13.0.1, 12.1.5, 12.1.4	gcc 4.8.2, 4.4.7
PGI only	pgf90 16.1, 15.10, 15.7, 15.1, 14.10, 14.9, 14.7, 13.9, 13.3	pgcc
PGI & gcc	pgf90 16.1, 15.10, 15.7, 15.1, 14.10, 14.9, 14.7, 13.9, 13.3	gcc 4.8.2
GNU only	gfortran 6.3.0, 5.3.0	gcc 6.3.0, 5.3.0

Unforeseen build issues may occur when using older compiler and library versions. As always, the best results come from using the most recent version of compilers.

2.6 Getting Help and Reporting Problems

Should the user experience any difficulty building EnKF on their system, please first confirm that both the WRF model and the GSI have successfully built. Should the EnKF build fail, but the other two succeed, feel free to contact the community EnKF support at *enkf-help@ucar.edu*

for assistance.

At a minimum, when reporting issues building the code, please include a copy of the EnKF build log.

3 Running EnKF

3.1 Input Data Required to Run EnKF

This chapter discusses the process of running EnKF cases. It includes:

1. Discussions of the input data required to run EnKF.
2. A detailed explanation of how to run both a regional and global EnKF with the released run scripts.
3. Introduction to the files in a successful regional and global EnKF run directory

3.1.1 Input Data Required to Run EnKF

In most cases, three types of input data, ensemble mean and members, observations, and fixed files, must be available before running EnKF.

1. Ensemble mean and members
The ensemble members and ensemble mean of certain regional and global ensemble forecast systems are used as the background for the EnKF analysis. When this EnKF system is used for regional analysis with WRF ensembles, the ensemble members and ensemble mean follow the naming convention:

```
firstguess.mem001
firstguess.mem002
... ...
```

```
firstguess.ensmean
```

Please note that the number of allocated computer cores to run EnKF must be larger than the ensemble size. The ensemble members can be generated using various methods, such as:

- Using global/regional ensemble forecasts.
- Ensemble forecasts generated using multi-physics, multi-models, or adding random perturbations drawn from climatology.
- In cycling assimilation, using ensemble forecasts initialized from previous ensemble analyses generated by EnKF.

This version EnKF can use any of the following ensemble files as the background:

- ARW NetCDF forecast
- NMM NetCDF forecast
- GFS forecast files

2. Prepare observation ensemble priors (observation innovation)
In addition to the ensemble backgrounds on model grids, the ensemble priors of all observations (observation innovation for all ensemble members) are also needed to run EnKF. The observation ensemble priors are generated by running GSI observation forward operators with the ensemble members as backgrounds (without doing actual GSI analyses). In this release, the GSI v3.5 run script includes options for generating the observation ensemble priors (details in section 3.2). The observation ensemble priors files should follow the following naming conventions:

- For conventional observations:

  ```
  diag_conv_ges.mem001
  diag_conv_ges.mem002
  ... ...
  diag_conv_ges.ensmean
  ```

- For radiance observations:

  ```
  diag_instrument_Satellite.mem001 (e.g. diag_hirs4_n19.mem001)
  diag_ instrument_Satellite.mem002 (e.g. diag_hirs4_n19.mem002)
  ... ...
  diag_ instrument_Satellite.ensmean (e.g. diag_hirs4_n19.ensmean)
  ```

These diag files contain a lot of information about each observation. For more details on the content of diag files, please refer to the GSI User's Guide Appendix A.2. The preparation of observations for EnKF assimilation is done within GSI, including quality control of observations, selection of observation types for assimilation, and observation error tuning. In the default namelist situation, NO additional online quality control of observations is performed in the EnKF analysis step (although there is an option to do the similar quality control of observations as the GSI variational scheme.)

3. Fixed files
EnKF uses the the same fixed files as GSI to setup the analysis configurations. Detailed

explanation of all fixed files provided in the community GSI system is in the GSI user's Guide, Chapter 3. The following is a list of fixed files needed for EnKF analyses:

- for observation control:

  ```
  convinfo - conventional data (prepufr) info file
  ozinfo - ozone retrieval info file
  satinfo - satellite channel info file
  ```

- when satellite radiance data are assimilated, the following files are needed to do the adaptive radiance bias correction:

  ```
  satbias_in - satellite bias correction coefficient file
  satbias_pc - satellite bias correction coefficient file for passive channels
  ```

Note that this version EnKF uses adaptive bias correction. The bias correction coefficients are in a single file that combined satellite angle dependent and mass bias correction coefficients. See GSI User's Guide for more detail. When the namelist parameter *readin_localization* is set to true, file *hybens_locinfo* is needed, in which customized localization values varying by model level are contained.

3.2 EnKF and GSI Observer Run Scripts

In this release version, four sample run scripts for EnKF applications are under directory `comGSIv3.5_EnKFv1.1/run`:

- *run_gsi_regional.ksh* for running regional GSI to generate the observation ensemble priors. Referred to as the GSI observer run scripts.
- *run_gsi_global.ksh* is GSI observer run script for global applications (to generate the observation ensemble priors).
- *run_enkf_wrf.ksh* for running regional EnKF
- *run_enkf_global.ksh* for running global EnKF

These run scripts are introduced in detail in the following sections. Also provided are two scripts for generating the GSI and EnKF namelist:

- *comgsi_namelist.sh* generates GSI namelist on the fly (called by GSI observer run scripts).
- *comgsi_namelist_gfs.sh* generates GSI namelist on the fly (called by GSI observer run scripts) for global GSI applications.
- *enkf_wrf_namelist.sh* generates EnKF namelist on the fly (called by EnKF run script) for regional EnKF applications.

3.2.1 General Introduction to the Run Scripts

These run scripts provide the run time environment necessary for running the GSI and EnKF executables. They all have similar steps, as follows:

1. Request computer resources to run GSI/EnKF.
2. Set environmental variables for the machine architecture.
3. Set experimental variables (such as experiment name, analysis time, background, and observation).
4. Check the definitions of required variables. Generate a run directory for GSI/EnKF
5. Copy the GSI/EnKF executable to the run directory.
6. Copy/Link the background file/ensemble files to the run directory.
7. Link observations to the run directory.
8. Link fixed files (statistic, control, and coefficient files) to the run directory.
9. Generate namelist for GSI/EnKF.
10. Run the GSI/EnKF executable.
11. Post-process: save analysis results, generate diagnostic files, clean run directory.

In the GSI User's Guide, three sections explain the first three steps in detail:

- Section 3.2.2.1: Setting up the machine environment (step 1)
- Section 3.2.2.2: Setting up the running environment (step 2)
- Section 3.2.2.3: Setting up an analysis case (step 3)

For this documentation, the first 2 steps will be skipped and the 3rd step in the GSI observer and EnKF run scripts will be discussed. The community GSI analysis run script and the GSI observer run script are the same, with the GSI observer capability controlled by flags that turn off the minimization, select appropriate namelist options and enable looping through all the ensemble members to generate the ensemble observation priors for each member, including the ensemble mean.

3.2.2 GSI Observer Run Scripts

Setting up a case

This section discusses variables specific to the user's case, such as analysis time, working directory, background and observation files, location of fixed files and CRTM coefficients, and the GSI executable file. The script looks like:

```
#####################################################
# case set up (users should change this part)
#####################################################
#
# ANAL_TIME= analysis time  (YYYYMMDDHH)
# WORK_ROOT= working directory, where GSI runs
# PREPBURF = path of PreBUFR conventional obs
# BK_FILE  = path and name of background file
# OBS_ROOT = path of observations files
# FIX_ROOT = path of fix files
# GSI_EXE  = path and name of the gsi executable
  ANAL_TIME=2014061700
  HH=‘echo $ANAL_TIME | cut -c9-10‘
  WORK_ROOT=run/testarw
  OBS_ROOT=data/20140617/obs
  PREPBUFR=${OBS_ROOT}/nam.t${HH}z.prepbufr.tm00.nr
  BK_ROOT=data/20140617/2014061700/arw
  BK_FILE=${BK_ROOT}/wrfinput_d01.${ANAL_TIME}
```

```
CRTM_ROOT=data/CRTM_2.2.3
GSI_ROOT=code/comGSIv3.5_EnKFv1.1
FIX_ROOT=${GSI_ROOT}/fix
GSI_EXE=${GSI_ROOT}/run/gsi.exe
GSI_NAMELIST=${GSI_ROOT}/run/comgsi_namelist.sh
```

The options ANAL_TIME, WORK_ROOT, PREPBURF, BK_FILE, OBS_ROOT, FIX_ROOT, GSI_EXE are all the same settings as the GSI analysis configuration. Two options: BK_ROOT , GSI_ROOT, are the root directories for ensemble members and the GSI system. These exist to make links to the background and GSI system easy and shorter. The new option: GSI_NAMELIST is needed because the namelist section was taken out of the run scripts in this release as a separate file to improve the structure and readability of the run scripts. Users can find the namelist files for both GSI and EnKF in the same directory as the run scripts. Please note the option BK_FILE is pointing to the ensemble mean. The next part of this block has additional options to specify other important aspects of the GSI observer.

```
 #---------------------
# bk_core= which WRF core is used as background (NMM or ARW or NMMB)
# bkcv_option= which background error covariance and parameter will be used
#              (GLOBAL or NAM)
# if_clean = clean  : delete temperal files in working directory (default)
#            no     : leave running directory as is (this is for debug only)
  bk_core=ARW
  bkcv_option=NAM
  if_clean=clean
# if_observer = Yes  : only used as observation operater for enkf
# no_member     number of ensemble members
# BK_FILE_mem   path and base for ensemble members
  if_observer=No   # Yes, or, No -- case sensitive !
  no_member=20
  BK_FILE_mem=${BK_ROOT}/wrfarw.mem
```

The options *bk_core*, *bkcv_option*, and, *if_clean* are the same as in the GSI analysis run scripts. The new option if_observer indicates if this GSI run is for the generation of the observation ensemble priors or for a regular GSI run. The new option *no_member* specifies the number of ensemble members that are need to be calculated for the observation ensemble priors. This should also be the ensemble number in the EnKF analysis. Option *BK_FILE_mem* is the path and the name of the ensembles without the ensemble member ID appended. The scripts will add the ensemble member ID as a three digital number, such as 000, 001,. . . .

Loop through ensemble members

As mentioned previously, the GSI ensemble observer run scripts are the same as the GSI analysis run scripts released with the community GSI. Since the observer only generates diag files, which includes useful information on the observation innovation, the GSI outer loop number for the observer should be set to 0 to skip all minimization iterations.

The contents of the run scripts can be divided into two parts, those before the following comments and those after:

```
##################################################
# start to calculate diag files for each member
##################################################
```

Before this comment, the scripts have the same functionality as when running a GSI analysis, except that the background options in the scripts for the observer funcationality are set for the ensemble mean. Additionally, the namelist is built with the two following options set, which skips the minimization and saves all observation processing from the ensemble mean:

```
if [ ${if_observer} = Yes ] ; then
  nummiter=0
  if_read_obs_save='.true.'
  if_read_obs_skip='.false.'
else
```

Please refer to the GSI user's guide for detailed explanation of the remainder of this portion of the run scripts. The second portion of the script loops through each member to calculate the observation ensemble priors based on the GSI run environments setup by the first portion.

Listed below is an annotated version of the 2nd part of the GSI observer run script with explanations on each function block.

```
if [ ${if_observer} = Yes ] ; then
```

This 2nd part of the script only runs if option *if_observer* is set to "Yes". The diag files from the ensemble mean need to be saved first with the following commands:

```
  string=ges
  for type in $listall; do
    count=0
    if [[ -f diag_${type}_${string}.${ANAL_TIME} ]]; then
       mv diag_${type}_${string}.${ANAL_TIME} diag_${type}_${string}.ensmean
    fi
  done
  mv wrf_inout wrf_inout_ensmean
```

The following section builds the namelist for ensemble members. Please note two options need to be set different between mean and members:

```
# Build the GSI namelist on-the-fly for each member
  nummiter=0
  if_read_obs_save='.false.'
  if_read_obs_skip='.true.'
. $GSI_NAMELIST
cat << EOF > gsiparm.anl

 $comgsi_namelist

EOF
```

The option *if_read_obs_save* and *if_read_obs_skip* switch from "True" and "False", respectively for the mean to "False" and "True", respectively for the ensemble members.

This saves all observation processing information (including bias correction, thinning, etc) from the ensemble mean and save the same information for the ensemble members to keep observations constant.

The script loops through each ensemble member (from member 001 to *no_member*) to create the diag files for each member:

```
# Loop through each member
  loop="01"
  ensmem=1
  while [[ $ensmem -le $no_member ]];do

     rm pe0*

     print "\$ensmem is $ensmem"
     ensmemid=`printf %3.3i $ensmem`
```

After a member is processed, the script removes the old ensemble member and links to the new member before rerunning the calculation:

```
# get new background for each member
     if [[ -f wrf_inout ]]; then
       rm wrf_inout
     fi

     BK_FILE=${BK_FILE_mem}${ensmemid}
     echo $BK_FILE
     ln -s $BK_FILE wrf_inout
```

Run the GSI observer for this member:

```
#  run  GSI
     echo ' Run GSI with' ${bk_core} 'for member ', ${ensmemid}

     case $ARCH in
        'IBM_LSF')
           ${RUN_COMMAND} ./gsi.exe < gsiparm.anl > stdout_mem${ensmemid} 2>&1  ;;

        * )
           ${RUN_COMMAND} ./gsi.exe > stdout_mem${ensmemid} 2>&1 ;;
     esac

#  run time error check and save run time file status
     error=$?

     if [ ${error} -ne 0 ]; then
       echo "ERROR: ${GSI} crashed for member ${ensmemid} Exit status=${error}"
       exit ${error}
     fi

     ls -l * > list_run_directory_mem${ensmemid}
```

Generate diag files for this member:

```
# generate diag files

     for type in $listall; do
           count=`ls pe*${type}_${loop}* | wc -l`
        if [[ $count -gt 0 ]]; then
           cat pe*${type}_${loop}* > diag_${type}_${string}.mem${ensmemid}
        fi
     done

# next member
     (( ensmem += 1 ))

  done

fi
```

Since all members are using the same run directory, the run status of each member is overwriten by the following member. The stdout file and the file list in run directory are preserved with the ensemble member ID for debug.

3.2.3 Sample Regional EnKF Run Scripts

As described in section 3.2.1, the regional EnKF run scripts have been designed to have a similar structure to the GSI analysis and observer run scripts. Again, please refer to the GSI User's Guide section 3.2.2.1 and 3.2.2.2 for the first two steps.

The 3rd step is to setup the variables specific to the user's case, such as analysis time, working directory, background and observation files, location of fixed files and CRTM coefficients, and the EnKF executable. Most of the options in this portion are the same as the 3rd step in the GSI observer run scripts, which is discussed in section 3.2.2 of this User's Guide. Users should setup most of the variables in this portion based on the options in the GSI observer run scripts as they are the variables to setup the same things for the GSI and EnKF. The following is a sample script with explanations:

```
#
#####################################################
# case set up (users should change this part)
#####################################################
#
# ANAL_TIME= analysis time  (YYYYMMDDHH)
# WORK_ROOT= working directory, where GSI runs
# PREPBURF = path of PreBUFR conventional obs
# BK_FILE  = path and name of background file
# OBS_ROOT = path of observations files
# FIX_ROOT = path of fix files
# GSI_EXE  = path and name of the gsi executable

  ANAL_TIME=2012102506
  WORK_ROOT=enkf/regional/enkf_arw
  diag_ROOT=enkf/regional/gsidiag_arw
  BK_ROOT=enkf/enkfdata/arw/bk
  BK_FILE=${BK_ROOT}/wrfarw.ensmean
  GSI_ROOT=/enkf/code/comGSIv3.4_EnKFv1.0
  FIX_ROOT=${GSI_ROOT}/fix
  ENKF_EXE=${GSI_ROOT}/src/main/enkf/wrf_enkf
  CRTM_ROOT=CRTM_REL-2.2.3
  ENKF_NAMELIST=${GSI_ROOT}/run/enkf_wrf_namelist.sh
```

Options *ANAL_TIME*, *BK_ROOT*, *BK_FILE*, *GSI_ROOT*, *FIX_ROOT*, *CRTM_ROOT* have the same meanings as the ensemble GSI observer run scripts and should be set to the same values as the ensemble GSI observer run scripts. The option *WORK_ROOT* is the working directory which should have enough space to hold the ensemble members and EnKF analysis results. The option *diag_ROOT* is pointing to the run directory of the GSI observer, where the diag files are generated as data input for the EnKF. The option *ENKF_EXE* points to the EnKF executable, which is under the GSI source code directory in this release. The option *ENKF_NAMELIST* is the path and the EnKF namelist file, which sits outside of the run script as a separate file like the GSI namelist. Users can find the namelist files for both GSI and the EnKF in the same directory as the run scripts.

The next part of this block includes several additional options that specify several aspects of the ensemble members.

```
# ensemble parameters
#
  NMEM_ENKF=20
  BK_FILE_mem=${BK_ROOT}/wrfarw
  NLONS=111
```

```
  NLATS=111
  NLEVS=56
  IF_ARW=.true.
  IF_NMM=.false.
  list="conv amsua_n18 hirs4_n19"
```

Options *NMEM_ENKF*, *BK_FILE_mem* are also in the GSI observer run script and should be set to the same values as the GSI observer run script. Options *NLONS*, *NLATS*, and *NLEVS* specify 3 dimensions (XYZ) of the ensemble grid. Options *IF_ARW* and *IF_NMM* indicates which background, ARW NetCDF ensemble or the NMM NetCDF ensemble, is used in this EnKF run. Option *list* is a list of the observation types that the EnKF will use in the analysis. This list should be based on the diag files generated by the ensemble GSI observer.

At this point, users should be able to run the EnKF for simple cases without changing the rest of the script. However, some advanced users may need to change some of the following blocks for special applications.

```
#####################################################
# Users should NOT change script after this point
#####################################################
```

The next block sets the run command to run EnKF on multiple platforms. The *ARCH* is set at the beginning of the script.

```
case $ARCH in
   'IBM_LSF')
      ###### IBM LSF (Load Sharing Facility)
      RUN_COMMAND="mpirun.lsf " ;;

   'LINUX')
      if [ $GSIPROC = 1 ]; then
         #### Linux workstation - single processor
         RUN_COMMAND=""
      else
         ###### Linux workstation -  mpi run
        RUN_COMMAND="mpirun -np ${GSIPROC} -machinefile ~/mach "
      fi ;;

   'LINUX_LSF')
      ###### LINUX LSF (Load Sharing Facility)
      RUN_COMMAND="mpirun.lsf " ;;

   'LINUX_PBS')
      #### Linux cluster PBS (Portable Batch System)
#      RUN_COMMAND="mpirun -np ${GSIPROC} " ;;
      RUN_COMMAND="mpiexec_mpt -n ${GSIPROC} " ;;

   'DARWIN_PGI')
      ### Mac - mpi run
      if [ $GSIPROC = 1 ]; then
         #### Mac workstation - single processor
         RUN_COMMAND=""
      else
         ###### Mac workstation -  mpi run
         RUN_COMMAND="mpirun -np ${GSIPROC} -machinefile ~/mach "
      fi ;;

   * )
     print "error: $ARCH is not a supported platform configuration."
     exit 1 ;;
esac
```

The following block sets up fixed files and some analysis-time related values:

```
# Given the analysis date, compute the date from which the
# first guess comes.  Extract cycle and set prefix and suffix
# for guess and observation data files
# gdate=`$ndate -06 $adate`
gdate=$ANAL_TIME
YYYYMMDD=`echo $adate | cut -c1-8`
HH=`echo $adate | cut -c9-10`

# Fixed files
# CONVINFO=${FIX_ROOT}/global_convinfo.txt
# SATINFO=${FIX_ROOT}/global_satinfo.txt
# SCANINFO=${FIX_ROOT}/global_scaninfo.txt
# OZINFO=${FIX_ROOT}/global_ozinfo.txt
CONVINFO=${diag_ROOT}/convinfo
SATINFO=${diag_ROOT}/satinfo
SCANINFO=${diag_ROOT}/scaninfo
OZINFO=${diag_ROOT}/ozinfo
# LOCINFO=${FIX_ROOT}/global_hybens_locinfo.l64.txt
```

The next block creates a working directory (`workdir`) in which EnKF will run. The directory should have enough disk space to hold all the files needed for this run. This directory is cleaned before each run, therefore, save all the files needed from the previous run before rerunning EnKF.

```
# Set up workdir
rm -rf $WORK_ROOT
mkdir -p $WORK_ROOT
cd $WORK_ROOT
```

After creating a working directory, copy or link the EnKF executable, ensembles, diag files (observations), bias correction coefficients, and fixed files into the working directory.

```
cp $ENKF_EXE        ./enkf.x

cp $CONVINFO        ./convinfo
cp $SATINFO         ./satinfo
cp $SCANINFO        ./scaninfo
cp $OZINFO          ./ozinfo
# cp $LOCINFO         ./hybens_locinfo

cp $diag_ROOT/satbias_in ./satbias_in
cp $diag_ROOT/satbias_pc ./satbias_pc

# get mean
ln -s ${BK_FILE_mem}.ensmean ./firstguess.ensmean
for type in $list; do
   ln -s $diag_ROOT/diag_${type}_ges.ensmean .
done

# get each member
imem=1
while [[ $imem -le $NMEM_ENKF ]]; do
   member="mem"`printf %03i $imem`
   ln -s ${BK_FILE_mem}.${member} ./firstguess.${member}
   for type in $list; do
      ln -s $diag_ROOT/diag_${type}_ges.${member} .
   done
   (( imem = $imem + 1 ))
done
```

The following script is used to generate the EnKF namelist called enkf.nml in the working directory. Some namelist variables are explained in detail in Section 4.3. Appendix A gives a full list of namelist options.

```
# Build the GSI namelist on-the-fly
. $ENKF_NAMELIST
cat << EOF > enkf.nml

 $enkf_namelist

EOF
```

Copy the ensemble background files to the working directory and rename them as "analysis.${member}" . The EnKF will update those files as the analysis results.

```
# make analysis files
cp firstguess.ensmean analysis.ensmean
# get each member
imem=1
while [[ $imem -le $NMEM_ENKF ]]; do
   member="mem"`printf %03i $imem`
   cp firstguess.${member} analysis.${member}
   (( imem = $imem + 1 ))
done
```

The following block runs EnKF and checks if the EnKF has successfully completed.

```
###################################################
#  run  EnKF
###################################################
echo ' Run EnKF'

${RUN_COMMAND} ./enkf.x < enkf.nml > stdout 2>&1

##################################################################
#  run time error check
##################################################################
error=$?

if [ ${error} -ne 0 ]; then
  echo "ERROR: ${ENKF_EXE} crashed  Exit status=${error}"
  exit ${error}
fi
```

If this point is reached, the EnKF successfully finishes and exits with 0:

```
exit
```

3.3 Understanding Resulting Files in GSI Observer and EnKF Run Directory

To check if the GSI observer and EnKF runs have been successfully finished, it is important to understand the meaning of each file in the run directory.

3.3.1 The GSI Observer Run Directory

After customizing the GSI observer run script to your personal environment, it may be submitted to the batch system just as any other job. Following a successful run, the majority of the files in the GSI observer run directory will be the same as those in a sucessful GSI analysis run directory. The difference for the observer run is that the GSI observer generates more diag and stdout files related to each ensemble member. Below is an example of the files generated in the run directory from a GSI observer run:

```
amsuabufr                      diag_amsua_n18_ges.mem010      fit_t1.2014021300           mhsbufr
amsubbufr                      diag_amsua_n18_ges.mem011      fit_w1.2014021300           obs_input.0001
anavinfo                       diag_amsua_n18_ges.mem012      fort.201                    obs_input.0002
berror_stats                   diag_amsua_n18_ges.mem013      fort.202                    obs_input.0003
convinfo                       diag_amsua_n18_ges.mem014      fort.203                    obs_input.0004
diag_amsua_metop-a_ges.ensmean diag_amsua_n18_ges.mem015      fort.204                    obs_input.0006
diag_amsua_metop-a_ges.mem001  diag_amsua_n18_ges.mem016      fort.205                    obs_input.0010
diag_amsua_metop-a_ges.mem002  diag_amsua_n18_ges.mem017      fort.206                    obs_input.0019
diag_amsua_metop-a_ges.mem003  diag_amsua_n18_ges.mem018      fort.207                    obs_input.0021
diag_amsua_metop-a_ges.mem004  diag_amsua_n18_ges.mem019      fort.208                    obs_input.0022
diag_amsua_metop-a_ges.mem005  diag_amsua_n18_ges.mem020      fort.209                    obs_input.0026
diag_amsua_metop-a_ges.mem006  diag_conv_ges.ensmean          fort.210                    obs_input.0027
diag_amsua_metop-a_ges.mem007  diag_conv_ges.mem001           fort.211                    obs_input.0028
diag_amsua_metop-a_ges.mem008  diag_conv_ges.mem002           fort.212                    obs_input.0029
diag_amsua_metop-a_ges.mem009  diag_conv_ges.mem003           fort.213                    obs_input.0030
diag_amsua_metop-a_ges.mem010  diag_conv_ges.mem004           fort.214                    obs_input.common
diag_amsua_metop-a_ges.mem011  diag_conv_ges.mem005           fort.215                    ozinfo
diag_amsua_metop-a_ges.mem012  diag_conv_ges.mem006           fort.217                    pcpbias_out
diag_amsua_metop-a_ges.mem013  diag_conv_ges.mem007           fort.218                    pcpinfo
diag_amsua_metop-a_ges.mem014  diag_conv_ges.mem008           fort.219                    prepbufr
diag_amsua_metop-a_ges.mem015  diag_conv_ges.mem009           fort.220                    prepobs_prep.bufrtable
diag_amsua_metop-a_ges.mem016  diag_conv_ges.mem010           fort.221                    satbias_ang.out
diag_amsua_metop-a_ges.mem017  diag_conv_ges.mem011           fort.223                    satbias_in
diag_amsua_metop-a_ges.mem018  diag_conv_ges.mem012           fort.224                    satbias_out
diag_amsua_metop-a_ges.mem019  diag_conv_ges.mem013           fort.225                    satbias_out.int
diag_amsua_metop-a_ges.mem020  diag_conv_ges.mem014           fort.226                    satbias_pc
diag_amsua_n15_ges.ensmean     diag_conv_ges.mem015           fort.227                    satbias_pc.out
diag_amsua_n15_ges.mem001      diag_conv_ges.mem016           fort.228                    satinfo
diag_amsua_n15_ges.mem002      diag_conv_ges.mem017           fort.229                    sigf03
diag_amsua_n15_ges.mem003      diag_conv_ges.mem018           fort.230                    stdout
diag_amsua_n15_ges.mem004      diag_conv_ges.mem019           gpsrobufr                   stdout.anl.2014021300
diag_amsua_n15_ges.mem005      diag_conv_ges.mem020           gsi.exe                     stdout_mem001
diag_amsua_n15_ges.mem006      diag_hirs4_metop-a_ges.ensmean gsiparm.anl                 stdout_mem002
diag_amsua_n15_ges.mem007      diag_hirs4_metop-a_ges.mem001  hirs3bufr                   stdout_mem003
diag_amsua_n15_ges.mem008      diag_hirs4_metop-a_ges.mem002  hirs4bufr                   stdout_mem004
diag_amsua_n15_ges.mem009      diag_hirs4_metop-a_ges.mem003  l2rwbufr                    stdout_mem005
diag_amsua_n15_ges.mem010      diag_hirs4_metop-a_ges.mem004  list_run_directory          stdout_mem006
diag_amsua_n15_ges.mem011      diag_hirs4_metop-a_ges.mem005  list_run_directory_mem001   stdout_mem007
diag_amsua_n15_ges.mem012      diag_hirs4_metop-a_ges.mem006  list_run_directory_mem002   stdout_mem008
diag_amsua_n15_ges.mem013      diag_hirs4_metop-a_ges.mem007  list_run_directory_mem003   stdout_mem009
diag_amsua_n15_ges.mem014      diag_hirs4_metop-a_ges.mem008  list_run_directory_mem004   stdout_mem010
diag_amsua_n15_ges.mem015      diag_hirs4_metop-a_ges.mem009  list_run_directory_mem005   stdout_mem011
diag_amsua_n15_ges.mem016      diag_hirs4_metop-a_ges.mem010  list_run_directory_mem006   stdout_mem012
diag_amsua_n15_ges.mem017      diag_hirs4_metop-a_ges.mem011  list_run_directory_mem007   stdout_mem013
diag_amsua_n15_ges.mem018      diag_hirs4_metop-a_ges.mem012  list_run_directory_mem008   stdout_mem014
diag_amsua_n15_ges.mem019      diag_hirs4_metop-a_ges.mem013  list_run_directory_mem009   stdout_mem015
diag_amsua_n15_ges.mem020      diag_hirs4_metop-a_ges.mem014  list_run_directory_mem010   stdout_mem016
diag_amsua_n18_ges.ensmean     diag_hirs4_metop-a_ges.mem015  list_run_directory_mem011   stdout_mem017
diag_amsua_n18_ges.mem001      diag_hirs4_metop-a_ges.mem016  list_run_directory_mem012   stdout_mem018
diag_amsua_n18_ges.mem002      diag_hirs4_metop-a_ges.mem017  list_run_directory_mem013   stdout_mem019
diag_amsua_n18_ges.mem003      diag_hirs4_metop-a_ges.mem018  list_run_directory_mem014   stdout_mem020
diag_amsua_n18_ges.mem004      diag_hirs4_metop-a_ges.mem019  list_run_directory_mem015   wrfanl.2014021300
diag_amsua_n18_ges.mem005      diag_hirs4_metop-a_ges.mem020  list_run_directory_mem016   wrf_inout
diag_amsua_n18_ges.mem006      errtable                       list_run_directory_mem017   wrf_inout_ensmean
diag_amsua_n18_ges.mem007      fit_p1.2014021300              list_run_directory_mem018
diag_amsua_n18_ges.mem008      fit_q1.2014021300              list_run_directory_mem019
diag_amsua_n18_ges.mem009      fit_rad1.2014021300            list_run_directory_mem020
```

This case was a regional analysis with WRF/ARW NetCDF backgrounds. In this case, 20 ensemble members are used to generate the diag files. A brief introduction of the additional files in the GSI observer runs is given below:

- *diag_conv_ges.mem???* - Diag files of conventional observations for ensemble member ???.

- *diag_conv_ges.ensmean* - Diag files of satellite radiance observation for ensemble mean.
- *stdout_mem???* - Standard output from GSI observer run for ensemble member ???.
- *list_run_directory_mem???* - The list of the files in the run directory after the GSI observer is finished for ensemble member ???.

3.3.2 Files in the EnKF Run Directory

After customizing the EnKF run script to your personal environment, it may be submitted to the batch system just as any other job. Upon successful assimilation, the ensemble analyses (both members and ensemble mean), covariance inflation factor to the ensemble analyses, and updated satellite bias correction coefficients (if requested) are output in the run directory. Below is an example of the files generated in the run directory from one of the EnKF regional test cases:

```
analysis.ensmean           diag_amsua_n18_ges.mem003  diag_conv_ges.mem008          firstguess.mem002
analysis.mem001            diag_amsua_n18_ges.mem004  diag_conv_ges.mem009          firstguess.mem003
analysis.mem002            diag_amsua_n18_ges.mem005  diag_conv_ges.mem010          firstguess.mem004
analysis.mem003            diag_amsua_n18_ges.mem006  diag_conv_ges.mem011          firstguess.mem005
analysis.mem004            diag_amsua_n18_ges.mem007  diag_conv_ges.mem012          firstguess.mem006
analysis.mem005            diag_amsua_n18_ges.mem008  diag_conv_ges.mem013          firstguess.mem007
analysis.mem006            diag_amsua_n18_ges.mem009  diag_conv_ges.mem014          firstguess.mem008
analysis.mem007            diag_amsua_n18_ges.mem010  diag_conv_ges.mem015          firstguess.mem009
analysis.mem008            diag_amsua_n18_ges.mem011  diag_conv_ges.mem016          firstguess.mem010
analysis.mem009            diag_amsua_n18_ges.mem012  diag_conv_ges.mem017          firstguess.mem011
analysis.mem010            diag_amsua_n18_ges.mem013  diag_conv_ges.mem018          firstguess.mem012
analysis.mem011            diag_amsua_n18_ges.mem014  diag_conv_ges.mem019          firstguess.mem013
analysis.mem012            diag_amsua_n18_ges.mem015  diag_conv_ges.mem020          firstguess.mem014
analysis.mem013            diag_amsua_n18_ges.mem016  diag_gome_metop-a_ges.ensmean  firstguess.mem015
analysis.mem014            diag_amsua_n18_ges.mem017  diag_gome_metop-b_ges.ensmean  firstguess.mem016
analysis.mem015            diag_amsua_n18_ges.mem018  diag_omi_aura_ges.ensmean      firstguess.mem017
analysis.mem016            diag_amsua_n18_ges.mem019  diag_sbuv2_n16_ges.ensmean     firstguess.mem018
analysis.mem017            diag_amsua_n18_ges.mem020  diag_sbuv2_n17_ges.ensmean     firstguess.mem019
analysis.mem018            diag_conv_ges.ensmean      diag_sbuv2_n18_ges.ensmean     firstguess.mem020
analysis.mem019            diag_conv_ges.mem001       diag_sbuv2_n19_ges.ensmean     ozinfo
analysis.mem020            diag_conv_ges.mem002       diff.nc                        satbias_ang.out
anavinfo                   diag_conv_ges.mem003       enkf.log                       satbias_in
convinfo                   diag_conv_ges.mem004       enkf.nml                       satbias_out.int
diag_amsua_n18_ges.ensmean diag_conv_ges.mem005       enkf.x                         satbias_pc
diag_amsua_n18_ges.mem001  diag_conv_ges.mem006       firstguess.ensmean             satinfo
diag_amsua_n18_ges.mem002  diag_conv_ges.mem007       firstguess.mem001              stdout
```

This case was a regional analysis with WRF/ARW NetCDF backgrounds. In this case, 20 ensemble members are used to estimate ensemble covariance, and both conventional observations (prepbufr) and radiance observations (AMSU-A) are assimilated.

A brief introduction of the files is given in the table below:

stdout	A text output file. This is the most commonly used file to check the analysis processes as well as basic and important information about the analyses. The contents of stdout are explained in detail in Chapter 4 and users are encouraged to read this file to become familiar with the order of EnKF analysis processing.
firstguess. mem001-0??	ensemble members of WRF background, in NetCDF format. This is in the same format as the WRF forecast. The ensemble background of the analysis variables are extracted from the files.
firstguess.ensmean	ensemble mean of WRF background, in NetCDF format.
analysis. mem001-0??:	ensemble analysis if EnKF completes successfully. The format is the same as the background file.
analysis.ensmean	ensemble mean of analyses, in NetCDF format.
diag_conv_ges. mem001-0??:	diag files in binary for conventional and GPS RO observations, which contain the observations and their priors.
covinflate.dat	Three-dimensional multiplicative inflation factor fields (from function inflate_ens in module inflation). These inflation fields can be visualized using plotting software.
*info (convinfo, satinfo,. . .):	info files that control data usage. Please see GSI User guide Chapter 4 for details.
satbias_in	the input coefficients of bias correction for satellite radiance observations.
satbias_pc	the input coefficients of satellite radiance bias correction for passive channels.

4

EnKF Diagnostics and Tuning

This chapter will discuss how to assess whether an EnKF was successful based on the contents of the standard output (stdout). Properly checking the EnKF output will also provide useful information to diagnose potential errors in the system. The chapter begins with an introduction to the content and structure of the EnKF stdout, followed by detailed discussion of tuning options in the namelist. This chapter follows the online exercise for a case at 00z on February 13th, 2014 (case 2014021300). This case uses WRF-ARW NetCDF ensemble files as the background and analyzes several observations typical for operations, including most conventional observation data and select radiance data (AMSU-A , HIRS4). The case was run on a Linux cluster supercomputer, using 32 cores. Users can follow this test to reproduce the following results by visiting:

`http://www.dtcenter.org/EnKF/users/tutorial/index.php`

4.1 Understanding the Standard Output from EnKF (stdout)

Upon completion of an EnKF run, it is always useful to do a quick check of the standard output (stdout), to assess the performance of the EnKF. The stdout file has information about whether the EnKF analysis has successfully completed, if the ensemble spread inflation looks good, and if the background and analysis fields are reasonable. Understanding the contents of this file can also be very helpful for users to find clues in the event of a crash.

The EnKF stdout has following information:

a. namelist configuration
b. background ensemble members of observations and analysis variables

c. statistics of the ensemble prior
d. domain and observation partition
e. statistics of the ensemble analysis
f. spread inflation of the analysis ensemble

The following section contains a detailed description of the content of stdout, explained using the WRF-ARW case: 2014021300. The analysis domain consists of 129 x 70 x 50 grid points. To keep the output concise and make it more readable, redundant content are omitted.

The following indicates the start of the EnKF analysis. It shows how many processors are used to run the analysis and the beginning time of this run:

```
Execute poe command line: poe  ./enkf.x
 running on           32  processors ...


* . * . * . * . * . * . * . * . * . * . * . * . * . * . * . * . * . * . * . * .
     PROGRAM ENKF_ANL HAS BEGUN. COMPILED 2011319.55     ORG: NP25
     STARTING DATE-TIME  JUL 14,2016  00:01:03.956  196  THU   2457584
```

The following lines show how many satellite observation types are set to be read in and if the radiance bias correction is updated before the analysis process:

```
 number of satellite radiance files used          59
 number of satellite ozone files used          7
```

The following lines show the namelist used in the analysis:

```
 namelist parameters:
 --------------------
 &NAM_ENKF
 DATEIN  =  2014021300,
 DATAPATH        = ./                                                                                                                         ,
 IASSIM_ORDER    =           0,
 COVINFLATEMAX   =   100.0000    ,
 COVINFLATEMIN   =   1.000000    ,
 DETERMINISTIC   = T, SORTINC = T,
 CORRLENGTHNH    =   500.0000    ,
 CORRLENGTHTR    =   500.0000    ,
 CORRLENGTHSH    =   500.0000    ,
... ...
 SAVE_INFLATION  = F,
 LETKF_FLAG      = F,
 MASSBAL_ADJUST  = F,
 USE_EDGES       = F,
 EMISS_BC        = T,
 ISEED_PERTURBED_OBS     =           0
 /
 --------------------
```

The following lines show ensemble size, background information, the number of 3D analysis variables, and total number of 2D fields of the 3D variables, plus surface pressure (Ps).

```
         20  members
 number of background forecast times to be updated =           1
 first-guess forecast hours for analysis = 06
          5 3d vars to update
 total of         251  2d grids will be updated (including ps)
```

The actual analysis variables and the background type are printed, as well as the maximum and minimum of the surface pressure as a sanity check:

```
 Updating U, V, T, QVAPOR, PH, and MU for WRF-ARW...
 Surface pressure (spressmn) min/max range:   678.673339843750
   1032.22473144531
```

Next, the contents of convinfo are displayed:

```
READ_CONVINFO: tcp     112    0  1   3.000      0   0   0   75.00     5.000     1.000     75.00     0.000        0  0.000     0.000       0  0.000     0.000      2
READ_CONVINFO: ps      120    0  1   3.000      0   0   0   4.000     3.000     1.000     4.000    0.3000E-03    0  0.000     0.000       0  0.000     0.000      2
 line ignored in convinfo due to use flag  ps             132          0
         -1
......

 line ignored in convinfo due to use flag  gps             44          0
         -1
```

The next 272 lines show the content of ozinfo:

```
OZINFO_READ:  jpch_oz=    272
  1 sbuv6_n14      lev =    1 use = -1 pob =     0.240 gross =   1.000 error =   1.000 b_oz =  10.000 pg_oz =   0.000
  2 sbuv6_n14      lev =    2 use = -1 pob =     0.490 gross =   1.000 error =   1.000 b_oz =  10.000 pg_oz =   0.000
  3 sbuv6_n14      lev =    3 use = -1 pob =     0.980 gross =   1.000 error =   1.000 b_oz =  10.000 pg_oz =   0.000
... ...
271 mls30_aura     lev =   54 use = -1 pob =   999.999 gross =   9.999 error =   9.999 b_oz =  10.000 pg_oz =   0.000
272 mls30_aura     lev =   55 use = -1 pob =   999.999 gross =   9.999 error =   9.999 b_oz =  10.000 pg_oz =   0.000
```

The next 2680 lines show the content of satinfo and starts reading in the radiance bias correction coefficients:

```
RADINFO_READ:  jpch_rad=   2723
   1 amsua_n15      chan= 1 var= 3.000 varch_cld= 20.000 use= 1 ermax=   4.500 b_rad= 10.00 pg_rad=  0.00 icld_det=-2
   2 amsua_n15      chan= 2 var= 2.200 varch_cld= 18.000 use= 1 ermax=   4.500 b_rad= 10.00 pg_rad=  0.00 icld_det=-2
   3 amsua_n15      chan= 3 var= 2.000 varch_cld= 12.000 use= 1 ermax=   4.500 b_rad= 10.00 pg_rad=  0.00 icld_det=-2
   4 amsua_n15      chan= 4 var= 0.600 varch_cld=  3.000 use= 1 ermax=   2.500 b_rad= 10.00 pg_rad=  0.00 icld_det=-2
... ...

2722 ahi_himawari8  chan=15 var= 2.200 varch_cld=  0.000 use= -1 ermax=   2.000 b_rad= 10.00 pg_rad=  0.00 icld_det=-2
2723 ahi_himawari8  chan=16 var= 2.200 varch_cld=  0.000 use= -1 ermax=   2.000 b_rad= 10.00 pg_rad=  0.00 icld_det=-2
```

The majority of next near 3000 lines show the content of radiance bias correction coefficients:

```
 RADINFO_READ:  guess air mass bias correction coefficients below
   1            amsua_n15   -1.96518   0.00000  39.06817  -0.92319   0.40846   0.00000   0.00000   0.00173   3.68970   2.36747   8.86294  -1.84829
   2            amsua_n15   -7.41540   0.00000  80.13801  -0.81847  -1.01321   0.00000   0.00000   0.01427   2.63880   8.31509  22.04145  -1.56477
  ... ...
2679       avhrr3_metop-b    0.00000   0.00000   0.00000   0.00000   0.00000   0.00000   0.00000   0.00000   0.00000   0.00000   0.00000   0.00000
2680       avhrr3_metop-b    0.00000   0.00000   0.00000   0.00000   0.00000   0.00000   0.00000   0.00000   0.00000   0.00000   0.00000   0.00000
```

Among the lines describing the radiance bias correction coefficients, the EnKF also starts to inventory the observation number and types for both conventional and radiance data. Observations of various types are read in (from the `diag**` files) and the number of the observations and time spent reading in the observations are shown:

Start to check the radiance and ozone observations:

```
   1            sbuv2_n16  nread=         0  nkeep=         0  num_obs_tot=         0
   1            amsua_n15  nkeep=         0  num_obs_tot=         0
   2            sbuv2_n17  nread=         0  nkeep=         0  num_obs_tot=         0
   3            sbuv2_n18  nread=         0  nkeep=         0  num_obs_tot=         0
   4            sbuv2_n19  nread=         0  nkeep=         0  num_obs_tot=         0
   5             omi_aura  nread=         0  nkeep=         0  num_obs_tot=         0
   6         gome_metop-a  nread=         0  nkeep=         0  num_obs_tot=         0
   7         gome_metop-b  nread=         0  nkeep=         0  num_obs_tot=         0
          0  ozone obs
   2            amsua_n18  nkeep=      6284  num_obs_tot=      6284
   3            amsua_n19  nkeep=         0  num_obs_tot=      6284
   6           amsua_aqua  nkeep=         0  num_obs_tot=      6284
   7        amsua_metop-a  nkeep=         0  num_obs_tot=      6284
   8            airs_aqua  nkeep=         0  num_obs_tot=      6284
  11        hirs4_metop-a  nkeep=       154  num_obs_tot=      6438
  12              mhs_n18  nkeep=         0  num_obs_tot=      6438
  13              mhs_n19  nkeep=         0  num_obs_tot=      6438
  14          mhs_metop-a  nkeep=         0  num_obs_tot=      6438
  15         goes_img_g11  nkeep=         0  num_obs_tot=      6438
  25            ssmis_f17  nkeep=         0  num_obs_tot=      6438
  45           sndrd1_g15  nkeep=         0  num_obs_tot=      6438
  46           sndrd2_g15  nkeep=         0  num_obs_tot=      6438
  47           sndrd3_g15  nkeep=         0  num_obs_tot=      6438
  48           sndrd4_g15  nkeep=         0  num_obs_tot=      6438
  49         iasi_metop-a  nkeep=         0  num_obs_tot=      6438
  52           seviri_m10  nkeep=         0  num_obs_tot=      6438
  53        amsua_metop-b  nkeep=         0  num_obs_tot=      6438
```

Start to check the conventional observations:

```
 columns below obtype,nread, nkeep
    t       8043       7904
    q       2625       2621
   ps      14133      14133
   uv      21982      21982
  sst          0          0
  gps          0          0
   pw          0          0
   dw          0          0
  srw          0          0
   rw          0          0
  tcp          0          0
```

A summary of the total number of conventional (1^{st} number 46640), ozone (2^{nd} number 0), and radiance observations (3^{rd} number 6438) in diag files is given:

```
 nobs_conv, nobs_oz, nobs_sat =        46640           0        6438
```

Also, the following normalization factors for the radiance bias predictors could be inside the bias correction coefficient lines:

```
 npred  =            12
           1 biasprednorm =    1.000000
           2 biasprednorm =    1.000000
... ...
          11 biasprednorm =    1.000000
          12 biasprednorm =    1.000000
```

Then, the time spent reading in the observations and the total number of observations kept and rejected are shown:

```
max time in mpireadobs  =   0.6105460
total number of obs       53078
      53078 obs kept
          0 total obs rejected
time in read_obs =   1.25072427513078      on proc           0
```

In the stdout, regional averaged (northern hemisphere - NH, southern hemisphere - SH, and tropics - TR) statistics of the ensemble 'priors' fit to all observations are provided, partly for checking if the inflation is appropriate. The details of this part are discussed in next Section 4.2.

```
 innovation statistics for prior:
 conventional obs
 region, obtype, nobs, bias, innov stdev, sqrt(S+R), sqrt(S), sqrt(R):
NH    all ps  14083 -0.591E-02  0.860E+00  0.814E+00  0.601E+00  0.550E+00
TR    all ps     50 -0.629E-01  0.422E+00  0.661E+00  0.367E+00  0.550E+00
NH     all t   7766 -0.228E+00  0.163E+01  0.128E+01  0.480E+00  0.119E+01
TR     all t    138 -0.130E+00  0.118E+01  0.119E+01  0.234E+00  0.117E+01
NH    all uv  21680  0.136E-01  0.295E+01  0.276E+01  0.140E+01  0.237E+01
TR    all uv    302 -0.379E+00  0.251E+01  0.268E+01  0.124E+01  0.238E+01
NH     all q   2553 -0.323E-01  0.146E+00  0.157E+00  0.868E-01  0.131E+00
TR     all q     68 -0.721E-01  0.202E+00  0.164E+00  0.582E-01  0.154E+00
 satellite brightness temp
 instrument, channel #, nobs, bias, innov stdev, sqrt(S+R), sqrt(S), sqrt(R):
          amsua_n18   1   241 -0.199E+00  0.194E+01  0.306E+01  0.176E+01  0.250E+01
          amsua_n18   2   248 -0.194E+00  0.145E+01  0.234E+01  0.806E+00  0.220E+01
          amsua_n18   3   248 -0.620E+00  0.119E+01  0.206E+01  0.497E+00  0.200E+01
          amsua_n18   4   248 -0.189E+00  0.398E+00  0.568E+00  0.140E+00  0.550E+00
          amsua_n18   5   248  0.188E-01  0.337E+00  0.315E+00  0.966E-01  0.300E+00
          amsua_n18   6  1203 -0.396E-01  0.198E+00  0.238E+00  0.610E-01  0.230E+00
          amsua_n18   7  1225 -0.215E+00  0.324E+00  0.235E+00  0.480E-01  0.230E+00
          amsua_n18   8  1155 -0.235E+00  0.369E+00  0.259E+00  0.675E-01  0.250E+00
          amsua_n18  10  1211  0.408E+00  0.491E+00  0.353E+00  0.429E-01  0.350E+00
          amsua_n18  11    12  0.720E+00  0.723E+00  0.404E+00  0.567E-01  0.400E+00
          amsua_n18  15   245 -0.311E+00  0.198E+01  0.380E+01  0.148E+01  0.350E+01
      hirs4_metop-a   3     1  0.115E+01  0.115E+01  0.534E+00  0.631E-01  0.530E+00
      hirs4_metop-a   4    42  0.260E+00  0.521E+00  0.403E+00  0.488E-01  0.400E+00
      hirs4_metop-a   5     7 -0.534E+00  0.558E+00  0.367E+00  0.696E-01  0.360E+00
      hirs4_metop-a   6     7 -0.748E+00  0.782E+00  0.473E+00  0.109E+00  0.460E+00
      hirs4_metop-a   7     9 -0.557E+00  0.815E+00  0.580E+00  0.108E+00  0.570E+00
      hirs4_metop-a   8     9  0.374E+00  0.675E+00  0.100E+01  0.208E-01  0.100E+01
      hirs4_metop-a  10     9  0.199E+00  0.581E+00  0.612E+00  0.120E+00  0.600E+00
      hirs4_metop-a  11     9 -0.380E-01  0.718E+00  0.143E+01  0.770E+00  0.120E+01
      hirs4_metop-a  12    34 -0.988E+00  0.179E+01  0.213E+01  0.141E+01  0.160E+01
      hirs4_metop-a  13     9  0.694E-01  0.330E+00  0.382E+00  0.116E+00  0.364E+00
      hirs4_metop-a  14     9 -0.369E-01  0.257E+00  0.294E+00  0.137E+00  0.260E+00
      hirs4_metop-a  15     9  0.352E-01  0.289E+00  0.281E+00  0.107E+00  0.260E+00
 time in estimate_work_enkf1 =   5.723375361412764E-002  secs
```

Next, the analysis variable fields and the observations are distributed to the different processors. The following lines show the maximum and minimum number of observations and grid on subdomain and the time used to setup those decompositions:

```
 time in estimate_work_enkf1 =   5.723375361412764E-002  secs
 min/max numobs          32        7663
 min/max estimated work       637809      638478
 npts =         9030
 min/max number of points per proc =          243         317
 time to do model space decomp =   1.529277069494128E-003
 nobstot =        53078
 min/max number of obs per proc =         1658        1659
 time to do ob space decomp =   3.935033455491066E-004
 sending out observation prior ensemble perts from root ...
 nobstot*nanals              1061560
 npts*ndim              2266530
 ... took   1.730861375108361E-002  secs
 time in load_balance =  7.929413160309196E-002 on proc           0
```

Then, the ensemble background members of the analysis variables are read in and the maximum and minimum values of the fields at each vertical level are displayed. The maximum and minimum values are useful for a quick confirmation that the background fields have been read successfully. The size of the real array of ensemble perturbations updated on each processor and the time spent to read in and distribute the background ensemble are also shown:

```
anal_chunk size =      1591340
READGRIDDATA_ARW: U          1  -20.55222       20.72632
 ...
READGRIDDATA_ARW: U         50  -12.60845       52.03727
READGRIDDATA_ARW: V         51  -11.67894       21.03603
...
READGRIDDATA_ARW: V
...
READGRIDDATA_ARW: V        100  -17.20951       21.87867
READGRIDDATA_ARW: T        101  -62.00526       13.33912
...
READGRIDDATA_ARW: T        150   469.5381       509.8757
READGRIDDATA_ARW: QVAPOR         151  8.4173851E-05  1.7215064E-02
...
READGRIDDATA_ARW: QVAPOR         200  1.5460877E-06  1.0506745E-05
READGRIDDATA_ARW: PH        201  -14.64355       4.688477
...
READGRIDDATA_ARW: PH        250   7195.078       14206.67
READGRIDDATA_ARW: MU        251  -1326.610       3290.569
time in readgridata on root  0.326851664343849      secs
time to scatter state on root  0.163762195268646      secs
time in read_ensemble =  0.494254242163152      on proc           0
```

In EnSRF, observations can be skipped/not assimilated due to the adaptive observation thinning. The following lines show how many observations are skipped or not assimilated by this thinning. In this case, paoverpb_thresh = 0.99, which lead to 423 of the observations being skipped.

```
assimilate obs in order they were read in
1 timing on proc     0  =    0.51    0.24    0.01    0.10    0.00    0.15    0
      22152  out of       53078 obs skipped,       30926  used
      22736  out of       30926  same lat/long
time to broadcast obfit_post =   5.432746838778257E-004  secs, niter =
          1
```

Next, the update to the analysis variables is performed and the time for the updating is shown:

```
time in enkf_update =   4.79503449611366      on proc           0
```

The regional averaged statistics of the inflation values are shown for surface pressure:

```
global ps prior std. dev min/max =    21.52301       172.3270
NH mean ps prior standard deviation =    57.82042
NH mean ps posterior standard deviation (before inflation)=    29.20955
NH mean ps posterior standard deviation (after inflation) =    54.58403
NH mean ps inflation =    2.421974
TR mean ps prior standard deviation =    31.09247
TR mean ps posterior standard deviation (before inflation)=    28.31872
TR mean ps posterior standard deviation (after inflation) =    30.79383
TR mean ps inflation =    1.106122
time in inflate_ens =  1.366839744150639E-002 on proc           0
```

After the EnKF analysis, innovation statistics of the ensemble analyses are shown, similar to the ensemble priors'fit to all observations:

```
 innovation statistics for posterior:
 conventional obs
 region, obtype, nobs, bias, innov stdev, sqrt(S+R), sqrt(S), sqrt(R):
NH    all ps  14083  0.503E-03  0.757E+00  0.569E+00  0.146E+00  0.550E+00
TR    all ps     50 -0.562E-01  0.398E+00  0.585E+00  0.198E+00  0.550E+00
NH     all t   7766 -0.780E-01  0.145E+01  0.122E+01  0.263E+00  0.119E+01
TR     all t    138 -0.107E+00  0.115E+01  0.118E+01  0.199E+00  0.117E+01
NH    all uv  21680  0.406E-01  0.267E+01  0.246E+01  0.655E+00  0.237E+01
TR    all uv    302 -0.259E+00  0.236E+01  0.246E+01  0.636E+00  0.238E+01
NH     all q   2553 -0.174E-01  0.114E+00  0.138E+00  0.444E-01  0.131E+00
TR     all q     68 -0.279E-01  0.168E+00  0.159E+00  0.413E-01  0.154E+00
  satellite brightness temp
 instrument, channel #, nobs, bias, innov stdev, sqrt(S+R), sqrt(S), sqrt(R):
          amsua_n18   1   241  0.614E+02  0.915E+02  0.263E+01  0.823E+00  0.250E+01
          amsua_n18   2   248 -0.132E+03  0.171E+03  0.223E+01  0.376E+00  0.220E+01
          amsua_n18   3   248 -0.303E+02  0.398E+02  0.201E+01  0.244E+00  0.200E+01
          amsua_n18   4   248 -0.231E+00  0.580E+01  0.555E+00  0.776E-01  0.550E+00
          amsua_n18   5   248 -0.788E+00  0.199E+01  0.305E+00  0.575E-01  0.300E+00
          amsua_n18   6  1203 -0.138E+01  0.158E+01  0.234E+00  0.423E-01  0.230E+00
          amsua_n18   7  1225 -0.273E+01  0.287E+01  0.232E+00  0.322E-01  0.230E+00
          amsua_n18   8  1155 -0.131E+01  0.186E+01  0.254E+00  0.437E-01  0.250E+00
          amsua_n18  10  1211 -0.179E+01  0.196E+01  0.352E+00  0.374E-01  0.350E+00
          amsua_n18  11    12 -0.130E+01  0.133E+01  0.404E+00  0.537E-01  0.400E+00
          amsua_n18  15   245 -0.163E+03  0.198E+03  0.357E+01  0.693E+00  0.350E+01
      hirs4_metop-a   3     1  0.118E+01  0.118E+01  0.533E+00  0.558E-01  0.530E+00
      hirs4_metop-a   4    42  0.286E+00  0.542E+00  0.403E+00  0.455E-01  0.400E+00
      hirs4_metop-a   5     7 -0.451E+00  0.487E+00  0.364E+00  0.562E-01  0.360E+00
      hirs4_metop-a   6     7 -0.724E+00  0.755E+00  0.468E+00  0.852E-01  0.460E+00
      hirs4_metop-a   7     9 -0.572E+00  0.817E+00  0.577E+00  0.871E-01  0.570E+00
      hirs4_metop-a   8     9  0.122E+01  0.136E+01  0.100E+01  0.205E-01  0.100E+01
      hirs4_metop-a  10     9  0.855E+00  0.100E+01  0.609E+00  0.104E+00  0.600E+00
      hirs4_metop-a  11     9 -0.236E-01  0.543E+00  0.130E+01  0.507E+00  0.120E+01
      hirs4_metop-a  12    34 -0.738E+00  0.153E+01  0.178E+01  0.790E+00  0.160E+01
      hirs4_metop-a  13     9  0.121E+01  0.128E+01  0.375E+00  0.905E-01  0.364E+00
      hirs4_metop-a  14     9  0.776E-01  0.256E+00  0.281E+00  0.106E+00  0.260E+00
      hirs4_metop-a  15     9  0.281E+00  0.384E+00  0.273E+00  0.838E-01  0.260E+00
```

Finally, the minimum and maximum of the analysis increments are shown as below. The analysis increments should be within a reasonable range. The computational time of these steps are also shown:

```
 time to gather state on root  0.101591360988095      secs
 time level           1
 --------------
 ens. mean anal. increment min/max ps  -123.0895       264.4738
 ens. mean anal. increment min/max var          1  -7.882635       7.015289
 ens. mean anal. increment min/max var          2  -7.460623       6.953684
 ens. mean anal. increment min/max var          3  -4.598419       1.846811
 ens. mean anal. increment min/max var          4 -0.2952938      0.4161232
 ens. mean anal. increment min/max var          5  0.0000000E+00  0.0000000E+00
 time to gather ens mean increment on root  2.756185410544276E-002 secs
 time in writegriddata on root  0.255841280566528      secs
 time in write_ensemble =  0.385106393136084      on proc           0
```

The next line indicates the time of the analysis finish:

```
    ENDING DATE-TIME    JUL 14,2016  00:01:11.329  196  THU   2457584
    PROGRAM ENKF_ANL HAS ENDED.
* . * . * . * . * . * . * . * . * . * . * . * . * . * . * . * . * . * . * . * .
```

The next lines show the computer resources used in the analysis and a sign that EnKF has finished, "all done":

```
* . * . * . * . * . * . * . * . * . * . * . * . * . * . * . * . * . * . * . * .
```

```
*****************RESOURCE STATISTICS*******************************
The total amount of wall time                        = 7.431095
The total amount of time in user mode                = 6.480014
The total amount of time in sys mode                 = 0.713891
The maximum resident set size (KB)                   = 122848
Number of page faults without I/O activity           = 46688
Number of page faults with I/O activity              = 0
Number of times filesystem performed INPUT           = 0
Number of times filesystem performed OUTPUT          = 0
Number of Voluntary Context Switches                 = 24121
Number of InVoluntary Context Switches               = 138
*****************END OF RESOURCE STATISTICS*************************

 all done!
```

4.2 Tuning of Inflation and Localization

Proper inflation and localization values need to be determined by experimenting with different inflation values. The goal is to make the total ensemble spreads of priors match the innovations as much as possible. In the tuning process, the vertical and horizontal structure of the match should be carefully examined. The 3D distribution of the inflation values can be plotted offline from the output file `"covinflate.dat"` .

The tuning processes can vary for different resolutions and physics of the models as well as the observation types assimilated. The performance of the tuning may be better examined after multiple assimilation cycles allowing their effects to be accumulated and converged in cycling assimilation. Please also note that proper settings of observation error estimations are also critical for the inflation tuning.

Next, the test case is taken as an example to check if the inflation is appropriate. In the following lines of the standard output (`stdout`), the number (column `nobs`), mean (column `bias`), and standard deviation (column `innov stdev`) of the ensemble background fits to Ps, wind, temperature, and water vapor observations in the NH and TR are shown:

```
 innovation statistics for prior:
 conventional obs
 region, obtype, nobs, bias, innov stdev, sqrt(S+R), sqrt(S), sqrt(R):
NH    all ps  14083 -0.591E-02  0.860E+00  0.814E+00  0.601E+00  0.550E+00
TR    all ps     50 -0.629E-01  0.422E+00  0.661E+00  0.367E+00  0.550E+00
NH     all t   7766 -0.228E+00  0.163E+01  0.128E+01  0.480E+00  0.119E+01
TR     all t    138 -0.130E+00  0.118E+01  0.119E+01  0.234E+00  0.117E+01
NH    all uv  21680  0.136E-01  0.295E+01  0.276E+01  0.140E+01  0.237E+01
TR    all uv    302 -0.379E+00  0.251E+01  0.268E+01  0.124E+01  0.238E+01
NH     all q   2553 -0.323E-01  0.146E+00  0.157E+00  0.868E-01  0.131E+00
TR     all q     68 -0.721E-01  0.202E+00  0.164E+00  0.582E-01  0.154E+00
```

It would be desirable that the total ensemble spreads of prior/background match the innovation standard deviations as close as possible (Equation 15). In practice, one should check if there is any substantial ensemble spread deficiency. In the above sample standard output lines, the ensemble prior spreads are shown in the column $\sqrt(S)$, and the total spreads in the column $\sqrt(S+R)$. Here, $\sqrt(R)$ is the observation error.

The above sample statistics shows that in NH, the total spread of temperature observation priors (0.128E+01) is close to the innovation standard deviation (0.163E+01) of the priors fit to the observations.

4.3 Tuning EnKF through Key Namelist Options

The namelist parameters controlling the EnSRF analysis are set in the namelist sections: `/enkf_nml/`, `/satobs_enkf/`, and `/nam_wrf/` (all in the file `enkf.nml`). To obtain a successful analysis, it is very important to setup/tune properly the namelist variables, particularly those related to inflation and localization, for each specific application configuration of various models and observation types.

4.3.1 Set Up the Analyses Time and Data Location

The analysis time and working location are setup by the following parameters:

datein:	analysis time (YYYYMMDDHH)
datapath:	path to data directory (include trailing slash). In most cases, this should point to the current directory because the run scripts have setup the run environment in the working directory including copying the EnKF executable and data into the working directory.

4.3.2 Set Up Analysis Algorithm

In the current implementation, a few variations/flavors of the EnKF are available, including the ensemble square root Kalman filter (EnSRF, *Whitaker et al.*, 2010, etc), the perturbed observations EnKF, and the local transformed Kalman filter (LETKF). These options are set in the namelists below:

deterministic	= true, use EnSRF without perturbed observations; = false, use perturbed observations version of EnKF.
sortinc	if deterministic = false, re-order observations to minimize regression errors as described in Anderson (2003).
letkf_flag	if true, use the LETKF scheme.
boxsize	observation box size for LETKF (deg)
iassim_order	= 0, assimilation of observations in the order they are read in; = 1 in random order; = 2 in order of predicted posterior variance reduction.

paoverpb_thresh the threshold of the ratio of observation space posterior variance divided by prior variance (<= 1.0), that is, the expected reduction of prior variance by the observations.

Smaller values mean only those observations with stronger impact will be assimilated.

If =1, no thinning of observations is done. Typical values: 0.99 or 0.98.

4.3.3 Set Up the Analyses Variables

The analyses variables for both global and regional models are hardcoded in the EnSRF. There are several options using the following parameters:

```
nvars: number of 3d model variables to update.
```

For hydrostatic global models, there are typically 5, including U, V, T, QVAPOR, Ozone. For non-hydrostatic regional models, like WRF, different combinations of the 3D and 2D analyses variables are set as below:

```
For ARW             (arw =true)
      nvars            =3: U, V, T, and MU
                       =4: U, V, T, QVAPOR, and MU
                       =5: U, V, T, QVAPOR, PH, and MU
                       =6: U, V, W, T, QVAPOR, PH, and MU

For NMM             (nmm =true)
       nvars           =3: U, V, T, and PD
                       =4: U, V, T, QVAPOR, and PD
                       =5: U, V, T, QVAPOR, CWM, and PD
```

The analyses variable list can be adjusted or additional variables, such as moisture related variables, can be added to the list by modifying the subroutines "gridinfo.F90" and "gridio.F90" accordingly.

pseudo_rh: use or not 'pseudo-rh' analysis variable, as in GSI.
cliptracers: if true, tracers are clipped to zero when read in, and just before they are written out.

4.3.4 Set Up the Ensemble Backgrounds

The following parameters define which background fields will be used in the analyses and their dimensions:

regional	= true, perform regional EnSRF analyses using either ARW or NMM inputs as the background. =false, perform a global analysis. If either the parameter nmm or wrf is set to true, it will be set to true.
nmm	if true, background comes from WRF NMM. When using other background fields, set it to false.
wrf	if true, background comes from WRF ARW. When using other background fields, set it to false.
nlons	number of grid points in longitude of the model background
nlats	number of grid points in latitude of the model background
nlevs	number of vertical levels of the model background
nanls	number of ensemble members

4.3.5 Set Up Localization Distances

In the current EnKF implementation, distances for localization can be set separately in the northern hemisphere, tropics and southern hemisphere, and in the horizontal, vertical and time dimensions, and for different observation types using namelist parameters. The length scales should be given in km for the horizontal, hours for time, and scale heights (units of -$\log(P/P_{ref})$) in the vertical.

There are two options to setup the localization distances in horizontal and vertical. These are decided by the namelist variable `"readin_localization"` :

readin_localization:	=.true., customized horizontal and vertical localization values varying with model levels are read in from the external text file `"hybens_locinfo"`. = .false., the horizontal and vertical localization distances are set by the following parameters
corrlengthnh:	length for horizontal localization in km in the northern hemisphere (25N-90N, NH)
corrlengthtr:	length for horizontal localization in km in the tropics (25S-25N, TR)
corrlengthsh:	length for horizontal localization in km in the southern hemisphere (25S-90S, SH)
lnsigcutoffnh:	scale height for vertical localization in -$\log(P/P_ref)$ in NH.
lnsigcutofftr:	scale height for vertical localization in -$\log(P/P_ref)$ in TR.
lnsigcutoffsh:	scale height for vertical localization in -$\log(P/P_ref)$ in SH.

The text file `"hybens_locinfo"` contains vertical profile of horizontal and vertical localization length scales (in e-folding scale). These scales are converted to the full distance width (*1/0.388 in km) of Gaspari-Cohn function, where it goes to zero. The horizontal and vertical scales can be adjusted for each observation types easily in the subroutine `"read_locinfo.f90"` .

For satellite radiance and surface pressure observations, the vertical localization distances are set separately using the following parameters:

```
lnsigcutoffsatnh, lnsigcutoffsattr, lnsigcutoffsatsh; and
lnsigcutoffpsnh, lnsigcutoffpstr, lnsigcutoffpssh
```

There is also one option to setup the time localization window, which is the time away from the analyses time. This is decided by the following namelist variables:

obtimelnh:	time in hours away from the analyses time to move 2800 km at 30 ms-1 in the northern hemisphere
obtimeltr:	same as obtimelnh but for the tropics
obtimelsh:	same as obtimelnh but for the southern hemisphere

The empirical determination/tuning of proper localization distances is important for successful analyses with ensemble data assimilation. In general, large horizontal scale observations, like GPS radio occultation and radiosonde observations, require larger horizontal localization distances to fully take advantage of the observations. On the other hand, high horizontal resolution observations, like radar observations, may need shorter horizontal localization distances. The same is true for the localization distance in vertical.

4.3.6 Set Up Adaptive Posterior Inflation Parameters

The inflation can be set up by the following parameters:

anapertwtnh:	inflation parameter in NH.
anapertwttr:	inflation parameter in TR.
anapertwtsh:	inflation parameter in SH.
The parameters	= 0 means no inflation. = 1 means inflation all the way back to prior spread.

These are the key tuning parameters for the performance of EnSRF. Typical values: 0.95.

The inflation factor fields can be smoothed out using the following parameter:

smoothparm:	parameter for smoothing inflation factor, = -1 for no smoothing. > 0, the estimated inflation factor is smoothed using a Gaussian spectral filter with an e-folding scale of the parameter
latbound:	where the transition latitude starts (=25N or 25S)
delat:	latitude width of transition zone where the inflation parameter is smoothed.

The minimum and maximum inflation values allowed can be controlled by the following parameters:

```
covinflatemin: minimum inflation factor
covinflatemax: maximum inflation factor
```

4.3.7 Satellite Observations Related Parameters

The following parameters are used to setup the adaptive satellite bias correction:

lupd_satbias:	= .true., update the satellite bias correction coefficients using the adaptive bias correction scheme (when the iterations of numiter > 1). = .false., the bias correction coefficients are not updated.
numiter:	number of iterations for state/bias correction coefficients update. (only relevant when satellite radiances assimilated, i.e. nobs_sat>0)
biasvar:	background error variance for radiance bias coefficients (used in radbias.f90). The default is GSI value.
saterrfact:	factor to multiply satellite radiance errors.
sattypes_rad:	list of satellite observation types to be assimilated.
sattypes_oz:	list of ozone observation types to be assimilated.

4.3.8 Observation QC Parameters

In addition to the observation quality flags decided in the GSI observer, the EnSRF itself also conducts a gross check on the observations.

The outlier test computes the difference between the observation value and the prior ensemble mean. It then computes a standard deviation by taking the square root of the sum of the observation error variance and the prior ensemble variance for the observation. If the difference between the ensemble mean and the observation value is more than the specified number of standard deviations, then the observation is tossed. The threshold of the check is set as following:

sprd_tol: tolerance for gross check of observations.

Reject a observation if the prior mean is more than this many $\sqrt{(S+R)}$ from the observation, where S is ensemble prior variance and R is observation error variance. Typical value: 3.0

The following parameters are used to control if the GSI quality control procedure is performed in EnSRF:

varqc:	= .false., no variational quality control performed (default).
huber:	= .true., use huber norm instead of "flat-tail" (default: .false.).
zhuberleft:	departure parameter of huber norm.
zhuberright:	departure parameter of huber norm.

5

Applications for Regional and Global EnKF

In this chaper, the elements from the previous chapters will be applied to demonstrate how to run a regional and global case using the GSI observer and EnKF. These examples are intended to give users a clear idea of how to set up the GSI observer and EnKF for a particular application and properly check the run status and analysis results in order to determine if the run was successful. Note that the regional example focuses on WRF ARW, however WRF NMM and NMMB runs are similar, but require different background ensemble and namelist options. Similarly, the global example features a single global configuration (T254), however users may wish to use a different configuration, again requiring different background ensemble and namelist options.

It is assumed that the reader has successfully compiled GSI and EnKF on a local machine. For **regional case studies**, users should have the following data available:

1. Ensemble for background
    - Ensemble files from WRF-ARW, WRF-NMM, NMM-B forecast files may be used. For this case, WRF-ARW ensemble members will be used, which are generated from the GFS ensemble, following the naming convention: `wrfarw.mem00nn`. Where `nn` corresponds to each ensemble member ("ensmean" appended for ensemble mean).
2. Conventional, GPSRO and radiance data
    - Real time GDAS and NAM PrepBUFR data can be obtained from the server: `ftp://ftpprd.ncep.noaa.gov/pub/data/nccf/com/gfs/prod/`

ftp://ftpprd.ncep.noaa.gov/pub/data/nccf/com/nam/prod/

3. Fixed files
   - Fixed files are located in the `comGSIv3.5_EnKFv1.1/fix` directory

For **global case studies**, users should have the following data available:

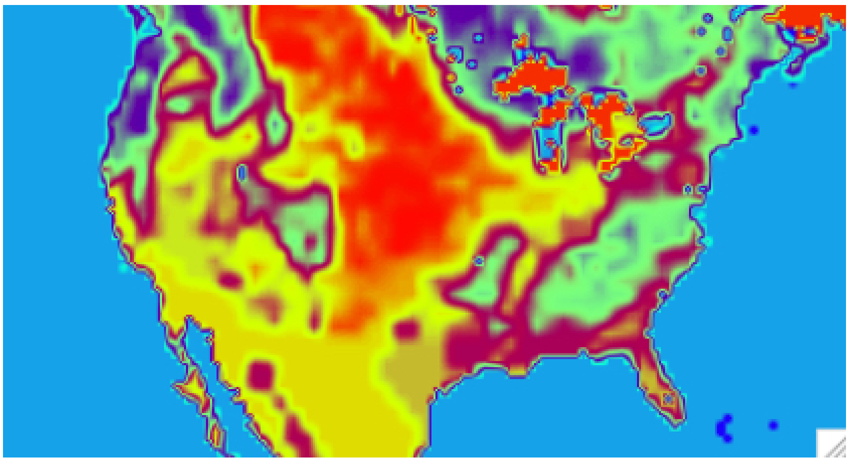
1. Ensemble for background
   - Ensemble files from GFS forecast files may be used. GFS ensemble members corresponding to various spectral resolutions may be used, following the naming convention: `sfg_yyyymmddhh_fhrff_mem0nn`. The "yyyymmddhh" corresponds to the initialization date, "ff" corresponds to the forecast hour, and "nn" corresponds to each ensemble member (`ensmean` appended for ensemble mean).
2. Conventional data
   - Real time GDAS PrepBUFR (and BUFR) data can be obtained from the server: `ftp://ftpprd.ncep.noaa.gov/pub/data/nccf/com/gfs/prod/`
3. Fixed files
   - Fixed files are located in the `comGSIv3.5_EnKFv1.1/fix/global` directory

The following cases will give users an example of how to run the EnKF for a regional case with various data sources, as well as a simple global application case. Users are welcome to download these examples from the EnKF User's webpage (online case for release v1.1) or create a new background ensemble and obtain observation data from the above server.

The background ensemble files and observations used in the regional case studies are as follows:

1. Background files: wrfarw.mem0001 - wrfarw.mem0020, wrfarw.ensmean
   - The horizontal grid spacing is 45-km with 51 vertical sigma levels.

Figure 5.1: Background used for regional case study



2. Conventional, GPSRO and radiance data from 00 UTC 13 February 2014
   - Conventional file:
     - `gdas1.t00z.prepbufr.nr`
   - GPSRO file:
     - `gdas1.t00z.gpsro.tm00.bufr_d`

- Radiance files:
  - gdas1.t00z.1bamua.tm00.bufr_d
  - gdas1.t00z.1bhrs4.tm00.bufr_d
  - gdas1.t00z.1bmhs.tm00.bufr_d

3. Fixed files:
   - Fixed files are located in the GSI/EnKF community code package, under /comGSIv3.5_EnKFv1.1/fix
   - For observation control:
     - convinfo: conventional data (PrepBUFR, GPSRO) info file
     - ozinfo: ozone retrival info file
     - satinfo: satellite radiance info file
   - Adaptive radiance bias correction (although example files are provided in the /fix directory, it is better to obtain bias correction files valid for the proper date. For this case, GFS bias correction coefficients are provided with the case data):
     - satbias_in: satellite bias correction coefficient file
     - satbias_pc: satellite bias correction coefficient file for passive channels.

This case study was run on a linux cluster. GSI no longer (from GSI v3.2) requires byte-swapping to little endian format. BUFRLIB can automatically handle byte order issues.

For the **regional ARW case**, assume the following locations:

Path to the background ensemble files:

```
/scratch/casedata/arw_2014021300/bk/
```

the path to the observations:

```
/scratch/casedata/arw_2014021300/obs
```

and the GSI release version 3.5/EnKF version 1.1 is located at:

```
/scratch/comGSIv3.5_EnKFv1.1
```

For the global GFS case, assume locations as follows: Path to background ensemble files:

```
/scratch/casedata/enkf_glb_t254/bk
```

the path to the observations:

```
/scratch/casedata/enkf_glb_t254/obs
```

and the GSI release version 3.5/EnKF version 1.1 is located at:

```
/scratch/comGSIv3.5_EnKFv1.1
```

5.1 Running GSI Observer for Regional Applications

5.1.1 Run Script

With both GSI and EnKF compiled and the background ensemble files and observations acquired, the next step is to work with the gsi observer run script, `run_gsi_regional.ksh`. The location of this script is under `comGSIv3.5_EnKFv1.1/run`. This run script is the same as the one used for a GSI analysis run, with a few specific options selected in order to loop through all the ensemble members and generate the ensemble observation priors for each member and the ensemble mean. In addition to the GSI observer specific options, other user-specific modifications need to be made:

- **Set up batch queueing system**
  To run GSI with multiple processors, a job queuing head must be added to the `run_gsi_regional.ksh` script. The set up of the job queue is dependent on the machine and the job control system. Refer to the GSI User's Guide, section 3.2.2, for more examples of the setup section of this script. The following example is setup to run on a Linux cluster with LSF:

```
#BSUB -P ??????             # project code
#BSUB -W 00:30                 # wall-clock time (hrs:mins)
#BSUB -n 4                  # number of tasks in job
#BSUB -R "span[ptile=16]"      # run 16 MPI tasks per node
#BSUB -J gsi                   # job name
#BSUB -o gsi.%J.out          # output file name in which %J is replaced by the job ID
#BSUB -e gsi.%J.err          # error file name in which %J is replaced by the job ID
#BSUB -q small               # queue
```

- **Set up number of processors and the job queue system used** For this example, `LINUX_LSF` and 4 processors are used:

```
GSIPROC=4
ARCH='LINUX_LSF'
```

- **Set up the case data, analysis time, fix files, GSI executable, and CRTM coefficients:**
  Set up analysis time:

```
ANAL_TIME=2014021300
```

  Set up working directory, which will hold the analysis results (all ensemble members will be run in this directory). This directory must have the proper write permissions as well as enough space to hold the output.

```
WORK_ROOT=/scratch/${user}/comGSIv3.5_EnKFv1.1/run/gsidiag_2014021300
```

  Set path to the observation directory and the PrepBUFR file within the observation directory. All observations to be assimilated should reside in this directory.

```
OBS_ROOT=/scratch/${user}/casedata/arw_2014021300/obs
PREPBUFR=${OBS_ROOT}/gdas1.t00z.prepbufr.nr
```

  Set path to background ensemble files:

```
BK_ROOT=/scratch/${user}/casedata/arw_2014021300/bk
```

  Set file for background ensemble mean:

```
BK_FILE=${BK_ROOT}/wrfarw.ensmean
```

Set the GSI system used for this case, including the paths of the fix files and the CRTM coefficents as well as the location of the GSI executable:

```
CRTM_ROOT=/scratch/${user}/CRTM_REL-2.2.3
GSI_ROOT=/scratch/${user}/comGSIv3.5_EnKFv1.1/
FIX_ROOT=${GSI_ROOT}/fix
GSI_EXE=${GSI_ROOT}/run/gsi.exe
GSI_NAMELIST=${GSI_ROOT}/run/comgsi_namelist.sh
```

- **Set which background and background error file to use**

```
bk_core=ARW
bkcv_option=NAM
if_clean=clean
```

This example uses the ARW NetCDF background; therefore `bk_core` is set to `ARW`. The regional background error covariance file is used in this case, as set by `bkcv_option=NAM` . Finally, the run scripts are set to clean the run directory to delete all temporary intermediate files.

- **Choose to run GSI observer, set up background ensemble information**

```
if_observer=Yes
no_member=20
BK_FILE_mem=${BK_ROOT}/wrfarw.mem
```

The option `if_observer=Yes` is the switch that enables `run_gsi_regional.ksh` to run the GSI observer (rather than GSI analysis). In this example, 20 ensemble members are selected with the naming convention: `wrfarw.memnnnn`. Note that memnnnn which is associated with each ensemble member (nnnn), is not included and will be appended later in the script.

- **Link observations**

```
# Link to the prepbufr data
ln -s ${PREPBUFR} ./prepbufr
# Link to the radiance data
ln -s ${OBS_ROOT}/gdas1.t00z.1bamua.tm00.bufr_d amsuabufr
ln -s ${OBS_ROOT}/gdas1.t00z.1bhrs4.tm00.bufr_d hirs4bufr
ln -s ${OBS_ROOT}/gdas1.t00z.1bmhs.tm00.bufr_d mhsbufr
ln -s ${OBS_ROOT}/gdas1.t00z.gpsro.tm00.bufr_d gpsrobufr
```

Past the arch selection, environment variable checks, and creation of working directory, users will find the location where the observations are linked. For this case, we can see that the conventional PrepBUFR observations have been linked, as well as three different satellite radiance BUFR files (AMSU-A, HIRS4, and MHS) and a GPS RO BUFR file. These files will be linked to the working directory and separate observation innovation (diag) files will be generated for each observation.

In the run script, the proper anavinfo file is selected based on the core and background error covariance used for the case:

```
echo ' Use NAM background error covariance'
BERROR=${FIX_ROOT}/${BYTE_ORDER}/nam_nmmstat_na.gcv
```

```
  OBERROR=${FIX_ROOT}/nam_errtable.r3dv
  if [ ${bk_core} = NMM ] ; then
     ANAVINFO=${FIX_ROOT}/anavinfo_ndas_netcdf
  fi
  if [ ${bk_core} = ARW ] ; then
     ANAVINFO=${FIX_ROOT}/anavinfo_arw_netcdf
  fi
  if [ ${bk_core} = NMMB ] ; then
     ANAVINFO=${FIX_ROOT}/anavinfo_nems_nmmb
  fi
```

The anavinfo file used for this case is `anavinfo_arw_netcdf` , because the background is NetCDF ARW and the regional background error covariance has been selected. It is important to check this file, located in the `./fix` directory. The number of vertical levels in `anavinfo_arw_netcdf` must match those for the background. In this example, the case has 50 vertical levels with the default value in `anavinfo_arw_netcdf` is 40. Change these values (`level column`) accordingly:

```
met_guess::
!var     level    crtm_use    desc                 orig_name
  ps        1       -1        surface_pressure      ps
  z         1       -1        geopotential_height   phis
  u        50        2        zonal_wind            u
  v        50        2        meridional_wind       v
  div      50       -1        zonal_wind            div
  vor      50       -1        meridional_wind       vor
  tv       50        2        virtual_temperature   tv
  q        50        2        specific_humidity     sphu
  oz       50        2        ozone                 ozone
  cw       50       10        cloud_condensate      cw
... ...
control_vector::
!var     level  itracer as/tsfc_sdv  an_amp0   source  funcof
 sf       50      0       1.00        -1.0     state    u,v
 vp       50      0       1.00        -1.0     state    u,v
 ps        1      0       0.50        -1.0     state    prse
 t        50      0       0.70        -1.0     state    tv
 q        50      1       0.70        -1.0     state    q
 oz       50      1       0.50        -1.0     state    oz
 sst       1      0       1.00        -1.0     state    sst
 cw       50      1       1.00        -1.0     state    cw
 stl       1      0       1.00        -1.0     motley   sst
 sti       1      0       1.00        -1.0     motley   sst
::
```

5.1.2 Run GSI Observer and Check Run Status

Once the run script is set up properly for the case and the machine, GSI can be run through the run script. The following command will submit the job:

```
$ bsub < run_gsi_regional.ksh
```

While the job is running, move to the working directory and check the details. Given the following working directory setup:

```
WORK_ROOT=/scratch/${user}/comGSIv3.5_EnKFv1.1/run/gsidiag_2014021300
```

Go to directory `WORK_ROOT=/scratch/${user}/comGSIv3.5_EnKFv1.1/` and check the run directory. A directory named `gsidiag_2014021300` should have been created. This directory is the run directory for this GSI observer case study. The directory will be populated with many files:

```
amsua_n18.TauCoeff.bin
ssmi_f15.SpcCoeff.bin
amsua_n18.SpcCoeff.bin
ssmi_f15.TauCoeff.bin
imgr_g11.SpcCoeff.bin
imgr_g11.TauCoeff.bin
```

These files are CRTM coefficients that have been linked to this run directory through the GSI run script. Similar to running the GSI analysis, many other files are linked or copied to this run directory, such as:

gsiparm.anl:	GSI namelist
prepbufr:	PrepBUFR file for conventional observation
convinfo:	data usage control for conventional data
satbias_in:	satellite bias correction coefficient file
satinfo:	data usage and channel control for satellite radiance data
berror_stats:	background error file
errtable:	observation error file

Additionally, for the GSI observer many files are generated as a result of the `lread_obs_save` and `lread_obs_skip` options in the namelist for writing/reading collective observation selection information:

obs_input.nnnn:	During ensemble mean (save=True, skip=False), save all observation preprocessing to this file. Each file (nnnn) is for a different variable type
pennnn.obs_setup:	When looping through ensemble members (save=False, skip=True), all observation preprocessing is skipped, *obs_input.nnnn* files are read and *pennnn.obs_setup* is written for each processor (nnnn) for all observations.

As the GSI observer is running for the ensemble mean as well as looping through each member, files are are generated for each ensemble member, as well as the ensemble mean.

The presence of the standard output files in the directory suggest the GSI observer run scripts have successfully set up a run environment for the GSI observer, properly looping through the ensemble members, and the GSI executable is running on each ensemble member.

pennnn.conv_01:	Files generated after O-B, generated for conventional observations as well as one file per sensor (e.g. AMSU-A n18). These files are trimmed for resulting diag files and cleaned.
list_run_directory:	Directory listing after ensemble mean is run, before directory is cleaned for running the ensemble members.
list_run_directory_memnnn:	As as above, for each ensemble member (nnn).
stdout:	Standard output file for ensemble mean.
stdout_memnnn:	Standard output file for each ensemble member (nnn).

Once GSI has finished running, diag files should be generated for each observation type for each member as well as the ensemble mean:

```
diag_conv_ges.ensmean    diag_conv_ges.mem001
diag_conv_ges.mem002     diag_conv_ges.mem003
diag_conv_ges.mem004     diag_conv_ges.mem005
... ...                  diag_conv_ges.mem020

diag_amsua_metop-a_ges.ensmean diag_amsua_metop-a_ges.mem001
diag_amsua_metop-a_ges.mem002  diag_amsua_metop-a_ges.mem003
diag_amsua_metop-a_ges.mem004  diag_amsua_metop-a_ges.mem005
 ... ...                       diag_amsua_metop-a_ges.mem020

diag_amsua_n18_ges.ensmean      diag_amsua_n18_ges.mem001
diag_amsua_n18_ges.mem002       diag_amsua_n18_ges.mem003
diag_amsua_n18_ges.mem004       diag_amsua_n18_ges.mem005
......
diag_hirs4_n19_ges.ensmean          diag_amsua_n15_ges.mem001
diag_hirs4_n19_ges.mem002           diag_amsua_n15_ges.mem003
diag_hirs4_n19_ges.mem004           diag_amsua_n15_ges.mem005
... ...                             diag_amsua_n15_ges.mem020
```

5.1.3 Check for Successful GSI Completion

The presence of the diag files and standard out files for the ensemble mean and each member indicates that the GSI observer has run without crashing, but does not necessary indicate a successful analysis. It is important to check the stdout files in the run directory to make sure the GSI observer completed each step without any obvious problems. The following are several important areas of the standard out file to check:

1. Read in the anavinfo and namelist

```
gsi_metguess_mod*init_:  2D-MET STATE VARIABLES:
ps
z
gsi_metguess_mod*init_:  3D-MET STATE VARIABLES:
u
v
div
vor
tv
q
```

```
 oz
 cw
.....

 &SETUP
 GENCODE =   78.0000000000000     ,
 FACTQMIN        =  0.000000000000000E+000,
 FACTQMAX        =  0.000000000000000E+000,
.....
```

2. Read in the background field
   The following lines in stdout immediately following the namelist section, indicate that GSI is reading the background fields. Checking that the range of the max and min values will indicate if particular background fields are normal.

```
  dh1  =           3
  iy,m,d,h,m,s=        2014           2          13           0           0
          0
  dh1  =           3
 rmse_var = SMOIS
 ndim1 =           3
 ordering = XYZ
 staggering =  N/A
 start_index =           1           1           1           0
 end_index =         129          70           4           0
 WrfType =          104
 ierr  =           0
... ...
  rmse_var = T ndim1=           3
  WrfType =          104  WRF_REAL=         104 ierr  =            0
  ordering = XYZ staggering =  N/A
  start_index =           1           1           1           0  end_index =
         129          70          50           0
  k,max,min,mid T=           1   313.3391       237.9947       273.2565
  k,max,min,mid T=           2   313.4587       238.4884       273.5590
  k,max,min,mid T=           3   313.6850       239.3906       274.1599
  k,max,min,mid T=           4   314.0594       241.0577       275.0899
  k,max,min,mid T=           5   314.6113       243.6186       276.3394
  k,max,min,mid T=           6   315.3213       247.0504       277.7711
... ...
```

3. Read in observational data
   The `stdout` (ensemble mean) and `stdout_memnnn` (ensemble members) contains distinct differences for the reading in the observational data controlled by the `lread_obs_save` and `lread_obs_skip` options in the namelist

   a. Ensemble mean

```
 read_obs_check: bufr file date is   2014021300 prepbufr q
 read_obs_check: bufr file date is   2014021300 prepbufr ps
 read_obs_check: bufr file uv                   not available satwndbufr
 read_obs_check: bufr file rw                   not available radarbufr
 read_obs_check: bufr file pcp_tmi   trmm       not available tmirrbufr
 read_obs_check: bufr file hirs3     n17        not available hirs3bufr
... ...
  number of extra processors           1
 READ_OBS:  read   33 mhs       mhs_n18           using ntasks=   2   0   1795      0
 READ_OBS:  read   34 mhs       mhs_n19           using ntasks=   2   2    584      0
 READ_OBS:  read   35 mhs       mhs_metop-a       using ntasks=   2   0    174      0
 READ_OBS:  read   36 mhs       mhs_metop-b       using ntasks=   2   2      2      0
... ...
```

```
 READ_BUFRTOVS         : file=mhsbufr        type=mhs        sis=mhs_n19              nread=        0
 ithin= 2 rmesh=  60.000000 isfcalc= 0 ndata=         0 ntask=  2
... ...
 READ_PREPBUFR         : file=prepbufr       type=uv         sis=uv                   nread=    52122
 ithin= 0 rmesh= 120.000000 isfcalc= 0 ndata=     50772 ntask=  1
... ...
 READ_OBS:  write collective obs selection info to obs_input.common
```

For the ensemble mean, the same procedure for observation processing as the GSI analysis is followed (read_obs_check, READ_OBS, READ_BUFRTOVS, READ_GPS, READ_PREPBUFR ...). Finally, it is indicated that the collective observations selection information is written out for use with the ensemble members.

b. Ensemble members
For the stdout_memnnn, the observation processing steps are skipped and observation information is read in from the ensemble mean:

```
OBSERVER_SET: read collective obs selection info from obs_input.common
```

Finally, for both the ensemble mean and members, the following lines will appear:

```
OBS_PARA: ps                     2038      2538      5499      6600
OBS_PARA: t                      3271      4431      7459      8753
OBS_PARA: q                      2823      3917      6110      7187
OBS_PARA: pw                      261       217       226       180
OBS_PARA: uv                     3333      5207      9038      8902
OBS_PARA: gps_ref                 597      1001       210      1085
OBS_PARA: hirs4      metop-a        0         0         0       174
OBS_PARA: amsua      n15          542       540         0         0
OBS_PARA: amsua      n18          606       672         0         0
OBS_PARA: amsua      metop-a        0         0         0       151
OBS_PARA: mhs        n18          730       828         0         0
OBS_PARA: mhs        metop-a        0         0         0       201
```

This table is important to check if the observations have been read in, which types of observations have been read in, and the distribution of observations in each subdomain.

4. Indication that the GSI observer has successfully run:

```
 observer_final: successfully finalized
 glbsoi: complete
[000]gsisub(): : complete.


     ENDING DATE-TIME    JUL 14,2016  23:29:19.825  196  THU   2457584
     PROGRAM GSI_ANL HAS ENDED.
* . * . * . * . * . * . * . * . * . * . * . * . * . * . * . * . * . * . * . * . * . * .
```

After looking over each section of the standard output files, it can be concluded that the GSI observer ran without issues. Note that because the outer loop was set to 0 (miter=0) for the GSI observer, no minimization occurred and therefore no analysis results were produced.

5.2 Running EnKF for Regional Applications

5.2.1 Run Script

Once the GSI observer has been run successfully, the next step is to setup the EnKF run script, `run_enkf_wrf.ksh`. This script is located under `comGSIv3.5_EnKFv1.1/run`. This run script uses the diag files generated by the GSI observer script as observation input, and generates the EnKF analysis. Similar to the GSI observer script, several user-specific modifications need to be made:

- Set up batch queueing system

```
#BSUB -P ????             # project code
#BSUB -W 00:30                 # wall-clock time (hrs:mins)
#BSUB -n 32                  # number of tasks in job
#BSUB -R "span[ptile=16]"      # run 16 MPI tasks per node
#BSUB -J gsi                   # job name
#BSUB -o gsi.%J.out          # output file name in which %J is replaced by the job ID
#BSUB -e gsi.%J.err          # error file name in which %J is replaced by the job ID
#BSUB -q small               # queue
```

- Set up number of processors and the job queue system used:

```
GSIPROC=32
ARCH='LINUX_LSF'
```

One difference from the GSI observer script is that the number of processors used should be greater than at least the number of ensemble members for running the EnKF. In this case we have 20 ensemble members and have requested 32 cores.

- Set up the analysis time, fixed files, EnKF executable

```
ANAL_TIME=2014021300
WORK_ROOT=/scratch/${user}/comGSIv3.5_EnKFv1.1/run/enkf_2014021300
diag_ROOT=/scratch/${user}/comGSIv3.5_EnKFv1.1/run/gsidiag_2014021300
BK_ROOT=/scratch/${user}/casedata/arw_2014021300/bk
BK_FILE=${BK_ROOT}/wrfarw.ensmean
GSI_ROOT=/scratch/${user}/comGSIv3.5_EnKFv1.1
FIX_ROOT=${GSI_ROOT}/fix
ENKF_EXE=${GSI_ROOT}/src/main/enkf/wrf_enkf
CRTM_ROOT=/scratch/${user}/CRTM_2.2.3/
ENKF_NAMELIST=${GSI_ROOT}/run/enkf_wrf_namelist.sh
```

- Set background file and location of the diag files (from GSI observer)

```
diag_ROOT=/scratch/${user}/comGSIv3.5_EnKFv1.1/run/gsidiag_2014021300
BK_FILE=${BK_ROOT}/wrfarw.ensmean
```

Two modifications to note for the EnKF are the `diag_ROOT` and `BK_FILE`. The `diag_ROOT` points to the working directory where the GSI observer was run. This directory contains diag* files which will be linked to the EnKF working directory. `BK_FILE` points to the background ensemble mean.

- Select ensemble parameters

```
NMEM_ENKF=20
BK_FILE_mem=${BK_ROOT}/wrfarw
```

```
NLONS=129
NLATS=70
NLEVS=50
IF_ARW=.true.
IF_NMM=.false.
```

The next section sets information about the ensemble, including the number of members, background ensemble members, domain specifications, dynamical core, and list of observations to be used in the EnKF. Note that the ensemble members (BK_FILE_mem) do not include the 3-digit member number. This will be appended to the name specified for this field later in the script.

- Select observations

```
list="conv amsua_n18 amsua_metop-a"
```

The previous line is contained with the ensemble parameter section. For the observations to be assimilated, be aware that only observations/platforms that have been run through the GSI observer (and therefore a diag* file exists) are valid.

5.2.2 Run EnKF and Check Run Status

Once the run script is set up properly for the case and the machine, EnKF can be run through the run script. The following command will submit the job:

```
$ bsub < run_enkf_wrf.ksh
```

While the job is running, move to the working directory and check the details. Given the following working directory setup:

```
WORK_ROOT=/scratch/${user}/comGSIv3.5_EnKFv1.1/run/enkf_2014021300
```

Go to directory /scratch/${user}/comGSIv3.5_EnKFv1.1/run and check the run directory. A directory named enkf_2014021300 should have been created. This directory is the run directory for this EnKF case study. The directory will be populated with many links:

```
diag_conv_ges.ensmean  diag_conv_ges.mem007  diag_conv_ges.mem014
diag_conv_ges.mem001   diag_conv_ges.mem008  diag_conv_ges.mem015
diag_conv_ges.mem002   diag_conv_ges.mem009  diag_conv_ges.mem016
diag_conv_ges.mem003   diag_conv_ges.mem010  diag_conv_ges.mem017
diag_conv_ges.mem004   diag_conv_ges.mem011  diag_conv_ges.mem018
diag_conv_ges.mem005   diag_conv_ges.mem012  diag_conv_ges.mem019
diag_conv_ges.mem006   diag_conv_ges.mem013  diag_conv_ges.mem020

diag_amsua_n18_ges.ensmean  diag_amsua_n18_ges.mem007  diag_amsua_n18_ges.mem014
diag_amsua_n18_ges.mem001   diag_amsua_n18_ges.mem008  diag_amsua_n18_ges.mem015
diag_amsua_n18_ges.mem002   diag_amsua_n18_ges.mem009  diag_amsua_n18_ges.mem016
diag_amsua_n18_ges.mem003   diag_amsua_n18_ges.mem010  diag_amsua_n18_ges.mem017
diag_amsua_n18_ges.mem004   diag_amsua_n18_ges.mem011  diag_amsua_n18_ges.mem018
```

```
diag_amsua_n18_ges.mem005   diag_amsua_n18_ges.mem012  diag_amsua_n18_ges.mem019
diag_amsua_n18_ges.mem006   diag_amsua_n18_ges.mem013  diag_amsua_n18_ges.mem020

diag_hirs4_metop-a_ges.ensmean  diag_hirs4_metop-a_ges.mem007  diag_hirs4_metop-a_ges.mem014
diag_hirs4_metop-a_ges.mem001   diag_hirs4_metop-a_ges.mem008  diag_hirs4_metop-a_ges.mem015
diag_hirs4_metop-a_ges.mem002   diag_hirs4_metop-a_ges.mem009  diag_hirs4_metop-a_ges.mem016
diag_hirs4_metop-a_ges.mem003   diag_hirs4_metop-a_ges.mem010  diag_hirs4_metop-a_ges.mem017
diag_hirs4_metop-a_ges.mem004   diag_hirs4_metop-a_ges.mem011  diag_hirs4_metop-a_ges.mem018
diag_hirs4_metop-a_ges.mem005   diag_hirs4_metop-a_ges.mem012  diag_hirs4_metop-a_ges.mem019
diag_hirs4_metop-a_ges.mem006   diag_hirs4_metop-a_ges.mem013  diag_hirs4_metop-a_ges.mem020
```

The diag files are linked from the GSI observer working directory. These links are specified in the run script. Note that ozone diag files (gome, omi, sbuv) will appear in the directory without links (zero length files) if ozone diags are not present from the GSI observer.

The first guess files are also linked into the directory from the run script, pointing to the path of the ensemble mean and ensemble members designated in the setup section.

```
firstguess.ensmean  firstguess.mem007  firstguess.mem014
firstguess.mem001   firstguess.mem008  firstguess.mem015
firstguess.mem002   firstguess.mem009  firstguess.mem016
firstguess.mem003   firstguess.mem010  firstguess.mem017
firstguess.mem004   firstguess.mem011  firstguess.mem018
firstguess.mem005   firstguess.mem012  firstguess.mem019
firstguess.mem006   firstguess.mem013  firstguess.mem020
```

Similar to running the GSI observer, many other static files are linked or copied to this run directory, such as:

enkf.nml:	EnKF namelist
convinfo:	data usage control for conventional data
satbias_in:	satellite bias correction coefficient file
satbias_pc:	satellite bias correction file for passive channels
satinfo:	data usage and channel control for satellite radiance data
ozinfo:	data usage control from ozone data
covinflate.dat:	Three-dimensional multiplicative inflation factor fields
stdout:	EnKF standard output file

The presence of the standard output file in the directory suggest the EnKF run script has successfully set up a run environment for the EnKF, properly linking the first guess and diag files from the GSI observer, and the EnKF executable is running. Once EnKF has finished running, analysis files should be generated for each member as well as the ensemble mean:

```
analysis.ensmean  analysis.mem006  analysis.mem012  analysis.mem018
analysis.mem001   analysis.mem007  analysis.mem013  analysis.mem019
analysis.mem002   analysis.mem008  analysis.mem014  analysis.mem020
analysis.mem003   analysis.mem009  analysis.mem015
analysis.mem004   analysis.mem010  analysis.mem016
analysis.mem005   analysis.mem011  analysis.mem017
```

5.2.3 Check for Successful EnKF Completion

The presence of the analysis files for the ensemble mean and each member as well as the standard out file indicates that the EnKF has run without crashing, but does not necessary indicate a successful analysis. It is important to check the stdout file in the run directory to make sure the EnKF completed each step without any obvious problems. The following are several important areas of the standard out file to check:

1. Check namelist has been properly read in and configuration is correct:

```
 namelist parameters:
 --------------------
 &NAM_ENKF
 DATEIN  =  2014021300,
 DATAPATH        = ./
 IASSIM_ORDER    =           0,
 COVINFLATEMAX   =   100.0000    ,
 COVINFLATEMIN   =   1.000000    ,
 DETERMINISTIC   = T,
... ...
```

2. Check number of ensemble members, as well as the actual analysis variables and the background type. The maximum and minimum values for surface pressure are printed for a sanity check:

```
          20  members
 number of background forecast times to be updated =           1
 first-guess forecast hours for analysis = 06
           5 3d vars to update
 total of         251  2d grids will be updated (including ps)
 using multiplicative inflation based on Pa/Pb
 Vars in Rad-Jacobian (dims)
 --------------------------
 sst                          0
 enkf_main:  number of threads           1
 Updating U, V, T, QVAPOR, PH, and MU for WRF-ARW...
 Surface pressure (spressmn) min/max range:   678.673339843750
   1032.22473144531
```

3. Statistics of the ensemble prior
Following many lines describing the bias correction coefficients, the inventory of the observation number, observation types from the input diag* files, and the time spent reading in each observation type, statistics of the ensemble priors fit to all observations are provided for each region (NH, SH, TR):

```
 innovation statistics for prior:
 conventional obs
 region, obtype, nobs, bias, innov stdev, sqrt(S+R), sqrt(S), sqrt(R):
NH    all ps  14083 -0.591E-02  0.860E+00  0.814E+00  0.601E+00  0.550E+00
TR    all ps     50 -0.629E-01  0.422E+00  0.661E+00  0.367E+00  0.550E+00
NH     all t   7766 -0.228E+00  0.163E+01  0.128E+01  0.480E+00  0.119E+01
TR     all t    138 -0.130E+00  0.118E+01  0.119E+01  0.234E+00  0.117E+01
NH    all uv  21680  0.136E-01  0.295E+01  0.276E+01  0.140E+01  0.237E+01
TR    all uv    302 -0.379E+00  0.251E+01  0.268E+01  0.124E+01  0.238E+01
NH     all q   2553 -0.323E-01  0.146E+00  0.157E+00  0.868E-01  0.131E+00
TR     all q     68 -0.721E-01  0.202E+00  0.164E+00  0.582E-01  0.154E+00
 satellite brightness temp
```

```
instrument, channel #, nobs, bias, innov stdev, sqrt(S+R), sqrt(S), sqrt(R):
         amsua_n18   1   241 -0.199E+00  0.194E+01  0.306E+01  0.176E+01  0.250E+01
         amsua_n18   2   248 -0.194E+00  0.145E+01  0.234E+01  0.806E+00  0.220E+01
         amsua_n18   3   248 -0.620E+00  0.119E+01  0.206E+01  0.497E+00  0.200E+01
         amsua_n18   4   248 -0.189E+00  0.398E+00  0.568E+00  0.140E+00  0.550E+00
         amsua_n18   5   248  0.188E-01  0.337E+00  0.315E+00  0.966E-01  0.300E+00
         amsua_n18   6  1203 -0.396E-01  0.198E+00  0.238E+00  0.610E-01  0.230E+00
         amsua_n18   7  1225 -0.215E+00  0.324E+00  0.235E+00  0.480E-01  0.230E+00
         amsua_n18   8  1155 -0.235E+00  0.369E+00  0.259E+00  0.675E-01  0.250E+00
         amsua_n18  10  1211  0.408E+00  0.491E+00  0.353E+00  0.429E-01  0.350E+00
         amsua_n18  11    12  0.720E+00  0.723E+00  0.404E+00  0.567E-01  0.400E+00
         amsua_n18  15   245 -0.311E+00  0.198E+01  0.380E+01  0.148E+01  0.350E+01
     hirs4_metop-a   3     1  0.115E+01  0.115E+01  0.534E+00  0.631E-01  0.530E+00
     hirs4_metop-a   4    42  0.260E+00  0.521E+00  0.403E+00  0.488E-01  0.400E+00
     hirs4_metop-a   5     7 -0.534E+00  0.558E+00  0.367E+00  0.696E-01  0.360E+00
     hirs4_metop-a   6     7 -0.748E+00  0.782E+00  0.473E+00  0.109E+00  0.460E+00
     hirs4_metop-a   7     9 -0.557E+00  0.815E+00  0.580E+00  0.108E+00  0.570E+00
     hirs4_metop-a   8     9  0.374E+00  0.675E+00  0.100E+01  0.208E-01  0.100E+01
     hirs4_metop-a  10     9  0.199E+00  0.581E+00  0.612E+00  0.120E+00  0.600E+00
     hirs4_metop-a  11     9 -0.380E-01  0.718E+00  0.143E+01  0.770E+00  0.120E+01
     hirs4_metop-a  12    34 -0.988E+00  0.179E+01  0.213E+01  0.141E+01  0.160E+01
     hirs4_metop-a  13     9  0.694E-01  0.330E+00  0.382E+00  0.116E+00  0.364E+00
     hirs4_metop-a  14     9 -0.369E-01  0.257E+00  0.294E+00  0.137E+00  0.260E+00
     hirs4_metop-a  15     9  0.352E-01  0.289E+00  0.281E+00  0.107E+00  0.260E+00
```

This table should be checked in order to determine if the inflation is appropriate. The goal is to make the total ensemble spreads of priors ($\sqrt{S+R}$ match the innovations (innov std) as much as possible. We can see in certain regions, particularly in the NH the differences are noteable. These differences are associated with higher observation error ($\sqrt{R}$). Users should consider tuning of inflation and localization, which is typically determined using cases with multiple assimilation cycles. Refer to section 4.2 of this User's Guide for more information on tuning.

4. Domain and observation partition:

```
npts =         9030
min/max number of points per proc =          243         317
time to do model space decomp =   1.529277069494128E-003
nobstot =        53078
min/max number of obs per proc =          1658        1659
time to do ob space decomp =   3.935033455491066E-004
sending out observation prior ensemble perts from root ...
nobstot*nanals             1061560
npts*ndim              2266530
... took   1.730861375108361E-002  secs
time in load_balance =  7.929413160309196E-002 on proc           0
```

The analysis variables and the observations are distributed to different processors. We can see in this case that the min/max number of points per processor are 243 and 317, respectively. Similarly, we can see that the min/max number of observations per processor are 1658 and 1659, respectively, indicating that the observations are well dispersed among the processors.

```
READGRIDDATA_ARW: U           1  -20.55222        20.72632
READGRIDDATA_ARW: U           2  -20.87878        21.20316
READGRIDDATA_ARW: U           3  -21.61046        22.36177
READGRIDDATA_ARW: U           4  -22.62561        24.08465
```

```
 READGRIDDATA_ARW: U          5  -24.07165       26.47827
 READGRIDDATA_ARW: U          6  -25.89237       32.81841
 READGRIDDATA_ARW: U          7  -27.10455       35.81213
... ...
```

Additionally, check the minimum and maximum values of the fields at each vertical level as a quick sanity check.

5.2.4 Diagnose EnKF Analysis Results

At the bottom of the standard output file, there are several output statistics and tables that are helpful for users to diagnose the quality of the EnKF analysis.

1. Spread inflation of the analysis ensemble

```
 global ps prior std. dev min/max =    21.52301       172.3270
 NH mean ps prior standard deviation =    57.82042
 NH mean ps posterior standard deviation (before inflation)=    29.20955
 NH mean ps posterior standard deviation (after inflation) =    54.58403
 NH mean ps inflation =    2.421974
 TR mean ps prior standard deviation =    31.09247
 TR mean ps posterior standard deviation (before inflation)=    28.31872
 TR mean ps posterior standard deviation (after inflation) =    30.79383
 TR mean ps inflation =    1.106122
 time in inflate_ens =  1.366839744150639E-002 on proc           0
```

This section is important for checking whether the values of the inflation are reasonable by looking at a summary of the maximum and minimum values. Also, viewing the regional averaged statistics before and after inflation. For this case we can see both the NH and TR show larger standard deviation after the inflation, which is more consistent with the prior standard deviation.

2. Spread inflation of the analysis ensemble

```
 innovation statistics for posterior:
 conventional obs
 region, obtype, nobs, bias, innov stdev, sqrt(S+R), sqrt(S), sqrt(R):
NH    all ps  14083  0.503E-03  0.757E+00  0.569E+00  0.146E+00  0.550E+00
TR    all ps     50 -0.562E-01  0.398E+00  0.585E+00  0.198E+00  0.550E+00
NH     all t   7766 -0.780E-01  0.145E+01  0.122E+01  0.263E+00  0.119E+01
TR     all t    138 -0.107E+00  0.115E+01  0.118E+01  0.199E+00  0.117E+01
NH    all uv  21680  0.406E-01  0.267E+01  0.246E+01  0.655E+00  0.237E+01
TR    all uv    302 -0.259E+00  0.236E+01  0.246E+01  0.636E+00  0.238E+01
NH     all q   2553 -0.174E-01  0.114E+00  0.138E+00  0.444E-01  0.131E+00
TR     all q     68 -0.279E-01  0.168E+00  0.159E+00  0.413E-01  0.154E+00
 satellite brightness temp
 instrument, channel #, nobs, bias, innov stdev, sqrt(S+R), sqrt(S), sqrt(R):
          amsua_n18   1   241  0.614E+02  0.915E+02  0.263E+01  0.823E+00  0.250E+01
          amsua_n18   2   248 -0.132E+03  0.171E+03  0.223E+01  0.376E+00  0.220E+01
          amsua_n18   3   248 -0.303E+02  0.398E+02  0.201E+01  0.244E+00  0.200E+01
          amsua_n18   4   248 -0.231E+00  0.580E+01  0.555E+00  0.776E-01  0.550E+00
          amsua_n18   5   248 -0.788E+00  0.199E+01  0.305E+00  0.575E-01  0.300E+00
          amsua_n18   6  1203 -0.138E+01  0.158E+01  0.234E+00  0.423E-01  0.230E+00
          amsua_n18   7  1225 -0.273E+01  0.287E+01  0.232E+00  0.322E-01  0.230E+00
```

```
       amsua_n18   8  1155 -0.131E+01  0.186E+01  0.254E+00  0.437E-01  0.250E+00
       amsua_n18  10  1211 -0.179E+01  0.196E+01  0.352E+00  0.374E-01  0.350E+00
       amsua_n18  11    12 -0.130E+01  0.133E+01  0.404E+00  0.537E-01  0.400E+00
       amsua_n18  15   245 -0.163E+03  0.198E+03  0.357E+01  0.693E+00  0.350E+01
   hirs4_metop-a   3     1  0.118E+01  0.118E+01  0.533E+00  0.558E-01  0.530E+00
   hirs4_metop-a   4    42  0.286E+00  0.542E+00  0.403E+00  0.455E-01  0.400E+00
   hirs4_metop-a   5     7 -0.451E+00  0.487E+00  0.364E+00  0.562E-01  0.360E+00
   hirs4_metop-a   6     7 -0.724E+00  0.755E+00  0.468E+00  0.852E-01  0.460E+00
   hirs4_metop-a   7     9 -0.572E+00  0.817E+00  0.577E+00  0.871E-01  0.570E+00
   hirs4_metop-a   8     9  0.122E+01  0.136E+01  0.100E+01  0.205E-01  0.100E+01
   hirs4_metop-a  10     9  0.855E+00  0.100E+01  0.609E+00  0.104E+00  0.600E+00
   hirs4_metop-a  11     9 -0.236E-01  0.543E+00  0.130E+01  0.507E+00  0.120E+01
   hirs4_metop-a  12    34 -0.738E+00  0.153E+01  0.178E+01  0.790E+00  0.160E+01
   hirs4_metop-a  13     9  0.121E+01  0.128E+01  0.375E+00  0.905E-01  0.364E+00
   hirs4_metop-a  14     9  0.776E-01  0.256E+00  0.281E+00  0.106E+00  0.260E+00
   hirs4_metop-a  15     9  0.281E+00  0.384E+00  0.273E+00  0.838E-01  0.260E+00
```

After the EnKF analysis, it is important to check the innovation statistics, much like the prior fit to all observation.

5.3 Running GSI Observer for Global Applications

5.3.1 Run Script

With both GSI and EnKF compiled (note that a separate global executable must be generated for EnKF) and the background ensemble files and observations acquired, the next step is to work with the gsi observer run script, `run_gsi_global.ksh`. The location of this script is under `comGSIv3.5_EnKFv1.1/run`. This run script is the same as the one used for a GSI global analysis run, with a few specific options selected in order to loop through all the ensemble members and generate the ensemble observation priors for each member and the ensemble mean. For the global examples, users must choose a case that corresponds to the spectral resolution of the background ensemble. For this example, we will be using T254. In addition to the GSI observer specific options, other user-specific modifications need to be made:

- Set up batch queueing system
  To run GSI with multiple processors, a job queuing head must be added to the `run_gsi_global.ksh` script. The set up of the job queue is dependent on the machine and the job control system. Refer to the GSI User's Guide, section 3.2.2, for more examples of the setup section of this script. The following example is setup to run on a Linux cluster with LSF:

```
# LSF batch script to run an MPI application
#BSUB -P ???????            # project code
#BSUB -W 00:30                # wall-clock time (hrs:mins)
#BSUB -n 24                  # number of tasks in job
#BSUB -R "span[ptile=16]"     # run 16 MPI tasks per node
#BSUB -J gsi                  # job name
#BSUB -o gsi.%J.out         # output file name in which %J is replaced by the job ID
#BSUB -e gsi.%J.err         # error file name in which %J is replaced by the job ID
#BSUB -q small              # queue
```

- Set up number of processors and the job queue system used. For this example, `LINUX_LSF` and 24 processors are used:

```
GSIPROC=24
ARCH='LINUX_LSF'
```

- Set up the case data, analysis time, fix files, GSI executable, and CRTM coefficients:

  Set up analysis time and select global case:

```
ANAL_TIME=2014040512
GFSCASE=enkf_glb_t254
```

  Set up working directory, which will hold the analysis results (all ensemble members will be run in this directory). This directory must have the proper write permissions as well as enough space to hold the output.

```
WORK_ROOT=/scratch/${user}/comGSIv3.5-EnKFv1.1/run/gsidiag_${GFS_CASE}
```

  Set path to background ensemble files:

```
BK_ROOT=/scratch/${user}/casedata/enkf_glb_t254/bk
```

  Set path to the observation directory and the PrepBUFR file within the observation directory. All observations to be assimilated should reside in this directory.

```
OBS_ROOT=/scratch/${user}/casedata/enkf_glb_t254/obs
PREPBUFR=${OBS_ROOT}/gdas1.t12z.prepbufr.nr
```

  Set the GSI system used for this case, including the paths of the fix files and the CRTM coefficients as well as the location of the GSI executable:

```
CRTM_ROOT=/scratch/${user}/CRTM_REL-2.2.3
GSI_ROOT=/scratch/${user}/comGSIv3.5_EnKFv1.1/
FIX_ROOT=${GSI_ROOT}/fix
GSI_EXE=${GSI_ROOT}/run/gsi.exe
GSI_NAMELIST=${GSI_ROOT}/run/comgsi_namelist_gfs.sh
```

  Note that the GSI namelist generation script used for the global case (`comgsi_namelist_gfs.sh`) is different from the one used for the regional example.

- Choose to run GSI observer, set up background ensemble information

```
if_observer=Yes  # Yes ,or, No -- case sensitive!!!
no_member=10
BK_FILE_mem=${BK_ROOT}/sfg_2014040506
```

  The option `if_observer=Yes` is the switch that enables `run_gsi_global.ksh` to run the GSI observer (rather than GSI analysis). In this example, 10 ensemble members are selected with the naming convention: `sfg_${GUESS_DATE}_memnnnn` . Note that _memnnnn which is assocated with each ensemble member (nnnn), is not included and will be appended later in the script.

- Set the JCAP resolution for the case:

```
if [[ "$GFSCASE" = "T62" ]]; then
  JCAP=62
  JCAP_B=62
elif [[ "$GFSCASE" = "T126" ]]; then
  JCAP=126
```

```
  JCAP_B=126
elif [[ "$GFSCASE" = "enkf_glb_t254" ]]; then
  JCAP=254
  JCAP_B=254
elif [[ "$GFSCASE" = "T254" ]]; then
  JCAP=254
  JCAP_B=574
elif [[ "$GFSCASE" = "T574" ]]; then
  JCAP=574
  JCAP_B=1534
else
   echo "INVALID case = $GFSCASE"
   exit
fi
  LEVS=64
```

Note that this selection is filled based on selection of the GFSCASE. This example used the `enkf_glb_t254` case. All cases use 64 levels.

Further in the run script `run_gsi_global.ksh`, the resolution parameters are set based on the requested case (resolution). For this case, we are using JCAP=254:

```
elif [[ "$JCAP" = "254" ]]; then
   LONA=512
   LATA=256
   DELTIM=1200
   resol=2
```

- Link satellite bias correction coefficients, background ensemble files, and observations:

```
if [[ "$GFSCASE" = "enkf_glb_t254" ]]; then
  cp $OBS_ROOT/gdas1.t12z.abias      ./satbias_in
  cp $OBS_ROOT/gdas1.t12z.satang     ./satbias_angle

  cp $BK_ROOT/sfcanl_2014040506_fhr03_ensmean  ./sfcf03
  cp $BK_ROOT/sfcanl_2014040506_fhr06_ensmean  ./sfcf06
  cp $BK_ROOT/sfcanl_2014040506_fhr06_ensmean  ./sfcf09

  cp $BK_ROOT/sfg_2014040506_fhr03_mem001    ./sigf03
  cp $BK_ROOT/sfg_2014040506_fhr06_mem001    ./sigf06
  cp $BK_ROOT/sfg_2014040506_fhr09_mem001    ./sigf09
... ...
# Link to the prepbufr data
ln -s ${PREPBUFR} ./prepbufr

# Link to the other observation data

if [[ "$GFSCASE" = "enkf_glb_t254" ]]; then
  obsfile_amua=gdas1.t12z.1bamua.tm00.bufr_d
  obsfile_amub=gdas1.t12z.1bamub.tm00.bufr_d
else
```

Past the arch selection, environment variable checks, and creation of working directory, users will find the location where the observations are linked. For this case, we can see that the conventional PrepBUFR observations have been linked, as well as AMSU-A and AMSU-B satellite radiance BUFR files. These files will be linked to the working directory and separate observation innovation (`diag`) files will be generated for each observation.

5.3.2 Run GSI Observer and Check Run Status

Once the run script is set up properly for the case and the machine, GSI can be run through the run script. The following command will submit the job:

```
$ bsub < run_gsi_global.ksh
```

While the job is running, move to the working directory and check the details. Given the following working directory setup:

```
WORK_ROOT=/scratch/${user}/comGSIv3.5_EnKFv1.1/run/gsidiag_${GFSCASE}
```

Go to directory `/scratch/${user}/comGSIv3.5_EnKFv1.1/` and check the run directory. A directory named `gsidiag_enkf_glb_t254` should have been created. This directory is the run directory for this GSI observer case study. The directory will be populated with many files such as:

```
AerosolCoeff.bin                hirs4_n19.TauCoeff.bin          sndrD1_g14.TauCoeff.bin
ahi_himawari8.SpcCoeff.bin      iasi616_metop-a.SpcCoeff.bin    sndrD1_g15.SpcCoeff.bin
ahi_himawari8.TauCoeff.bin      iasi616_metop-a.TauCoeff.bin    sndrD1_g15.TauCoeff.bin
airs281SUBSET_aqua.SpcCoeff.bin iasi616_metop-b.SpcCoeff.bin    sndrD2_g11.SpcCoeff.bin
... ...
```

These files are CRTM coefficients that have been linked to this run directory through the GSI run script. Similar to running the GSI analysis, many other files are linked or copied to this run directory, such as:

`gsiparm.anl` :	GSI namelist
`prepbufr` :	PrepBUFR file for conventional observation
`convinfo` :	data usage control for conventional data
`satbias_in` :	satellite bias correction coefficient file
`satinfo` :	data usage and channel control for satellite radiance data
`berror_stats` :	background error file
`errtable` :	observation error file

Additionally, for the GSI observer many files are generated as a result of the `lread_obs_save` and `lread_obs_skip` options in the namelist for writing/reading collective observation selection information:

`obs_input.nnnn`:	During ensemble mean (save=True, skip=False), save all observation preprocessing to this file. Each file (nnnn) is for a different variable type
`pennnn.obs_setup`:	When looping through ensemble members (save=False, skip=True), all observation preprocessing is skipped, `obs_input.nnnn` files are read and `pennnn.obs_setup` is written for each processor (nnnn) for all observations.

As the GSI observer is running for the ensemble mean as well as looping through each member, files are for each ensemble member, as well as the ensemble mean:

`pennnn.conv_01` :	Files generated after O-B, generated for conventional observations as well as one file per sensor (e.g. AMSU-A n18). These files are trimmed for resulting `diag` files and cleaned.
`list_run_directory` :	Directory listing after ensemble mean is run, before directory is cleaned for running the ensemble members.
`list_run_directory_memnnn`:	As as above, for each ensemble member (nnn).
`stdout` :	Standard output file for ensemble mean.
`stdout_memnnn` :	Standard output file for each ensemble member (nnn).

The presence of the standard output files in the directory suggest the GSI observer run scripts have successfully set up a run environment for the GSI observer, properly looping through the ensemble members, and the GSI executable is running on each ensemble member. Once GSI has finished running, `diag` files should be generated for each observation type for each member as well as the ensemble mean:

```
diag_conv_ges.ensmean  diag_conv_ges.mem004  diag_conv_ges.mem008
diag_conv_ges.mem001   diag_conv_ges.mem005  diag_conv_ges.mem009
diag_conv_ges.mem002   diag_conv_ges.mem006  diag_conv_ges.mem010
diag_conv_ges.mem003   diag_conv_ges.mem007
diag_amsua_metop-a_ges.ensmean  diag_amsua_metop-a_ges.mem006
diag_amsua_metop-a_ges.mem001   diag_amsua_metop-a_ges.mem007
diag_amsua_metop-a_ges.mem002   diag_amsua_metop-a_ges.mem008
diag_amsua_metop-a_ges.mem003   diag_amsua_metop-a_ges.mem009
diag_amsua_metop-a_ges.mem004   diag_amsua_metop-a_ges.mem010
diag_amsua_metop-a_ges.mem005
diag_amsua_n18_ges.ensmean  diag_amsua_n18_ges.mem004  diag_amsua_n18_ges.mem008
diag_amsua_n18_ges.mem001   diag_amsua_n18_ges.mem005  diag_amsua_n18_ges.mem009
diag_amsua_n18_ges.mem002   diag_amsua_n18_ges.mem006  diag_amsua_n18_ges.mem010
diag_amsua_n18_ges.mem003   diag_amsua_n18_ges.mem007
diag_amsua_n19_ges.ensmean  diag_amsua_n19_ges.mem004  diag_amsua_n19_ges.mem008
diag_amsua_n19_ges.mem001   diag_amsua_n19_ges.mem005  diag_amsua_n19_ges.mem009
diag_amsua_n19_ges.mem002   diag_amsua_n19_ges.mem006  diag_amsua_n19_ges.mem010
diag_amsua_n19_ges.mem003   diag_amsua_n19_ges.mem007
```

5.3.3 Check for Successful GSI Completion

The presence of the `diag` files and standard out files for the ensemble mean and each member indicates that the GSI observer has run without crashing, but does not necessary indicate a successful analysis. It is important to check the stdout files in the run directory to make sure the GSI observer completed each step without any obvious problems. The following are several important areas of the standard out file to check:

1. Read in the `anavinfo` and namelist

```
 READ_FILES:  analysis date,minutes         2014           4           5
          12           0    19070640
```

```
 gsi_metguess_mod*init_:  2D-MET STATE VARIABLES:
 ps
 z
 gsi_metguess_mod*init_:  3D-MET STATE VARIABLES:
 u
 v
... ...

 &SETUP
 GENCODE =   78.0000000000000     ,
 FACTQMIN        =  0.000000000000000E+000,
 FACTQMAX        =  0.000000000000000E+000,
```

2. Read in the background field
   The following lines in the stdout immediately following the namelist section, indicate that GSI is reading the background fields. Checking that the range of the max and min values will indicate if particular background fields are normal.

```
GESINFO:  jcap_b= 254, levs= 64, latb=  256, lonb=  512, ntrac=  3, ncldt=  1, idvc=  2, nvcoord=  2,
idvm=  0, idsl=  0, idpsfc=  0, idthrm=  0
   k,ak,bk,ck,tref=  1   0.00000000000       1.00000000000       0.00000000000       300.000000000
   k,ak,bk,ck,tref=  2   0.00000000000      0.994671165943       0.00000000000       300.000000000
   k,ak,bk,ck,tref=  3  0.574999988079E-03  0.988626599312       0.00000000000       300.000000000
   k,ak,bk,ck,tref=  4  0.574100017548E-02  0.981742262840       0.00000000000       300.000000000
   ... ...
```

3. Read in observational data

   The `stdout` (ensemble mean) and `stdout_memnnn` (ensemble members) contains distinct differences for the reading in the observational data controlled by the `lread_obs_save` and `lread_obs_skip` options in the namelist.

   c. Ensemble mean

```
 read_obs_check: bufr file date is   2014040512 prepbufr t
 read_obs_check: bufr file date is   2014040512 prepbufr uv
 read_obs_check: bufr file date is   2014040512 prepbufr q
 read_obs_check: bufr file date is   2014040512 prepbufr ps
... ...
  number of extra processors           13
 READ_OBS:  read    1 ps         ps                  using ntasks=   1   0      0      0
 READ_OBS:  read    2 t          t                   using ntasks=   1   1      0      0
 READ_OBS:  read    3 q          q                   using ntasks=   1   2      0      0
 READ_OBS:  read    4 pw         pw                  using ntasks=   1   3   2738      0
... ...
 READ_PREPBUFR         : file=prepbufr        type=pw         sis=pw                  nread=       345
 ithin= 0 rmesh=1450.000000 isfcalc= 0 ndata=       345 ntask=  1
 READ_BUFRTOVS         : file=amsuabufr       type=amsua      sis=amsua_n18           nread=    202575
 ithin= 2 rmesh=1500.000000 isfcalc= 0 ndata=       675 ntask=  1
 ... ...
 READ_OBS:  write collective obs selection info to obs_input.common
```

   For the ensemble mean, the same procedure for observation processing as the GSI analysis is followed (read_obs_check, READ_OBS, READ_BUFRTOVS, READ_GPS, READ_PREPBUFR, ...). Finally, it is indicated that the collective observations selection information is written out for use with the ensemble members.

   d. Ensemble members
   For the `stdout_memnnn`, the observation processing sets are skipped and observation information is read in from the ensemble mean:

```
OBSERVER_SET:  read collective obs selection info from obs_input.common
```

Finally, for both the ensemble mean and members, the following lines will appear:

```
OBS_PARA: ps                     24       400      1649      2577        58       245      1496       506
                 82       402   616       251        36        67       328       864        44       224
               5210      1397    69       414       506       629
OBS_PARA: t                      48       403      3406      4875       174       488      4128      1825
                 84       724  2702      1146         4       215       882      2218       151       913
              11864      3167   160      1622      1308      2381
OBS_PARA: q                      16       370      3025      4749       154       383      3211      1737
                 51       594  2126      1102         0       161       697      1951       140       850
              10310      2974   155      1525      1104      1506
OBS_PARA: pw                      0         0         0         0         0         0         0         0
                  0         0     3         0         0         0        42        25         0         0
                251        32     0         0         0         4
OBS_PARA: uv                     52       410      4459      5664       239      1023      4994      1835
                 86      1553  3792      1316         3       184       877      2242       180      1576
              11854      3595   264      3186      2231      2895
OBS_PARA: amsua     n15           0         0         0         0         7         9         0         0
                  1         0     0         0         5         0         0         0         3        10
                  7         0     0         0         0         0
OBS_PARA: amsua     n18           3         1         0         0         5         2         0         0
                  1         0     0         0         5         6         2         0         2         5
                  6         8     0         0         0         0
OBS_PARA: amsua     n19           0         0         3         7         0         0         0         0
                  3         1     0         0         2         9        10         6         0         0
                  0         3     0         0         0         3
OBS_PARA: amsua     metop-a       0         0         0         4         0         0         0         2
                  0         4     9         5         0         0         0         0         0         0
                  0         0     0         6         9         6
OBS_PARA: amsua     metop-b       0         0         0         0         1         0         0         0
                  6         9     0         0         3         0         0         0         4         0
                  0         0     4        11         7         0
```

This table is important to check if the observations have been read in, which types of observations have been read in, and the distribution of observations in each subdomain.

4. Indication that the GSI observer has successfully run:

```
 glbsoi: complete
[000]gsisub(): : complete.


     ENDING DATE-TIME    JUL 15,2016  15:48:52.880  197  FRI   2457585
     PROGRAM GSI_ANL HAS ENDED.
* . * . * . * . * . * . * . * . * . * . * . * . * . * . * . * . * . * . * . * . * . * .
```

After looking over each section of the standard output files, it can be concluded that the GSI global observer ran without issues. Note that because the outer loop was set to 0 (`miter=0`) for the GSI observer, no minimization occurred and therefore no analysis results were produced.

5.4 Running EnKF for Global Applications

5.4.1 Run Script

Once the GSI global observer has been run successfully, the next step is to setup the EnKF run script, `run_enkf_global.ksh`. This script is located under `comGSIv3.5_EnKFv1.1/run`. This run script uses the `diag` files generated by the GSI global observer script as observation input, and generates the global EnKF analysis. Similar to the GSI observer script, several user-specific modifications need to be made:

- Set up batch queueing system

```
#BSUB -P ??????              # project code
#BSUB -W 00:30                  # wall-clock time (hrs:mins)
#BSUB -n 80                    # number of tasks in job
#BSUB -R "span[ptile=16]"       # run 16 MPI tasks per node
#BSUB -J gsi                    # job name
#BSUB -o gsi.%J.out           # output file name in which %J is replaced by the job ID
#BSUB -e gsi.%J.err           # error file name in which %J is replaced by the job ID
#BSUB -q small                # queue
```

One difference from the GSI observer script is that the number of processors used should be greater than at least the number of ensemble members for running the EnKF. In this case we have 10 ensemble members and have requested 80 cores due to the large global domain.

- Set up the analysis time, global case, fixed files, EnKF executable

```
ANAL_TIME=2014040512
GUESS_TIME=2014040506
GFSCASE=enkf_glb_t254
WORK_ROOT=/scratch/${user}/comGSIv3.5_EnKFv1.1/run/enkf_${GFSCASE}
BK_ROOT=/scratch/${user}/casedata/enkf_glb_t254/bk
GSI_ROOT=/scratch/${user}/comGSIv3.5_EnKFv1.1
FIX_ROOT=${GSI_ROOT}/fix
ENKF_EXE=${GSI_ROOT}/src/main/enkf/global_enkf
CRTM_ROOT=/scratch/${user}/CRTM_2.2.3
ENKF_NAMELIST=${GSI_ROOT}/run/enkf_gfs_namelist.sh
```

- Set location of the diag files (from GSI observer)

```
DIAG_ROOT=/scratch/${user}/comGSIv3.5_EnKFv1.1/run/gsidiag_${GFSCASE}
```

One modifications to note for the EnKF is the `DIAG_ROOT`. The `DIAG_ROOT` points to the working directory where the GSI observer runs. This directory contains `diag*` files which will be linked to the EnKF working directory.

- Select ensemble and case parameters

```
NMEM_ENKF=10
LEVS=64
NVARS=5
... ...
elif [[ "$GFSCASE" = "enkf_glb_t254" ]]; then
  JCAP=254
  JCAP_B=254
... ...
  elif [[ "$JCAP" = "254" ]]; then
   LONA=512
   LATA=256
   DELTIM=1200
   resol=2
```

This section sets information about the number of background ensemble members, domain specifications. This information mirrors that of the `run_gsi_global.ksh` script.

- Read in namelist

```
#Build EnKF namelist on-the-fly
. $ENKF_NAMELIST
cat << EOF > enkf.nml
```

```
 $enkf_namelist

EOF
```

Similar to the regional enkf (run_enkf_wrf.ksh), the global EnKF builds the namelist from an outside script for namelist generation (enkf_gfs_namelist.sh).

- Select observations

  ```
  list="conv amsua_metop-a amsua_n18 amsua_n15"
  ```

  The previous line is contained within the ensemble looping section. For the observations to be assimilated, be aware that only observations/platforms that have been run through the GSI observer (and therefore a diag* file exists) are valid. For this example case, we will only run EnKF for conventional observations.

5.4.2 Run EnKF and Check Run Status

Once the run script is set up properly for the case and the machine, EnKF can be run through the run script. The following command will submit the job:

```
$ bsub < run_enkf_global.ksh
```

While the job is running, move to the working directory and check the details. Given the following working directory setup:

```
WORK_ROOT=/scratch/${user}/comGSIv3.5_EnKFv1.1/run/enkf_${GFSCASE}
```

Go to directory /scratch/${user}/comGSIv3.5_EnKFv1.1/ and check the run directory. A directory named enkf_enkf_glb_t264 should have been created. This directory is the run directory for this EnKF case study. The directory will be populated with many links:

```
diag_conv_ges.ensmean  diag_conv_ges.mem004  diag_conv_ges.mem008
diag_conv_ges.mem001   diag_conv_ges.mem005  diag_conv_ges.mem009
diag_conv_ges.mem002   diag_conv_ges.mem006  diag_conv_ges.mem010
diag_conv_ges.mem003   diag_conv_ges.mem007
diag_amsua_metop-a_ges.ensmean  diag_amsua_metop-a_ges.mem006
diag_amsua_metop-a_ges.mem001   diag_amsua_metop-a_ges.mem007
diag_amsua_metop-a_ges.mem002   diag_amsua_metop-a_ges.mem008
diag_amsua_metop-a_ges.mem003   diag_amsua_metop-a_ges.mem009
diag_amsua_metop-a_ges.mem004   diag_amsua_metop-a_ges.mem010
diag_amsua_metop-a_ges.mem005
diag_amsua_n18_ges.ensmean  diag_amsua_n18_ges.mem004  diag_amsua_n18_ges.mem008
diag_amsua_n18_ges.mem001   diag_amsua_n18_ges.mem005  diag_amsua_n18_ges.mem009
diag_amsua_n18_ges.mem002   diag_amsua_n18_ges.mem006  diag_amsua_n18_ges.mem010
diag_amsua_n18_ges.mem003   diag_amsua_n18_ges.mem007
diag_amsua_n19_ges.ensmean  diag_amsua_n19_ges.mem004  diag_amsua_n19_ges.mem008
diag_amsua_n19_ges.mem001   diag_amsua_n19_ges.mem005  diag_amsua_n19_ges.mem009
diag_amsua_n19_ges.mem002   diag_amsua_n19_ges.mem006  diag_amsua_n19_ges.mem010
diag_amsua_n19_ges.mem003   diag_amsua_n19_ges.mem007
```

The diag files are linked from the GSI global observer working directory. These links are specified in the run script. Note that ozone diag files (gome, omi, sbuv) will appear in the directory without links (zero length files) if ozone diags are not present from the GSI observer.

```
sfg_2014040512_fhr06_ensmean  sfg_2014040512_fhr06_mem004  sfg_2014040512_fhr06_mem008
sfg_2014040512_fhr06_mem001   sfg_2014040512_fhr06_mem005  sfg_2014040512_fhr06_mem009
sfg_2014040512_fhr06_mem002   sfg_2014040512_fhr06_mem006  sfg_2014040512_fhr06_mem010
sfg_2014040512_fhr06_mem003   sfg_2014040512_fhr06_mem007
```

The first guess files are also linked into the directory from the run script, pointing to the path of the ensemble mean and ensemble members designated in the setup section.

Similar to running the GSI observer, many other static files are linked or copied to this run directory, such as:

enkf.nml:	EnKF namelist
convinfo:	data usage control for conventional data
satbias_in:	satellite bias correction coefficient file
satbias_angle:	satellite bias correction angle file
satinfo:	data usage and channel control for satellite radiance data
ozinfo:	data usage control from ozone data
covinflate.dat:	Three-dimensional multiplicative inflation factor fields
stdout:	EnKF standard output file

The presence of the standard output file in the directory suggests the EnKF run script has successfully set up a run environment for the EnKF, properly linking the first guess and diag files from the GSI observer, and the EnKF executable is running. Once EnKF has finished running, analysis files should be generated for each member:

```
sanl_2014040512_mem001  sanl_2014040512_mem005  sanl_2014040512_mem009
sanl_2014040512_mem002  sanl_2014040512_mem006  sanl_2014040512_mem010
sanl_2014040512_mem003  sanl_2014040512_mem007
sanl_2014040512_mem004  sanl_2014040512_mem008
```

5.4.3 Check for Successful EnKF Completion

The presence of the EnKF analysis files for each member as well as the standard out file indicates that the EnKF has run without crashing, but does not necessary indicate a successful analysis. It is important to check the stdout file in the run directory to make sure the EnKF completed each step without any obvious problems. The following are several important areas of the standard out file to check:

1. Check namelist has been properly read in and configuration is correct:

```
namelist parameters:
--------------------
```

```
&NAM_ENKF
DATEIN  =  2014040512,
DATAPATH        = ./
IASSIM_ORDER    =           0,
COVINFLATEMAX   =   100.0000    ,
COVINFLATEMIN   =   1.000000    ,
DETERMINISTIC   = T,
SORTINC = T,
CORRLENGTHNH    =   2000.000    ,
CORRLENGTHTR    =   2000.000    ,
CORRLENGTHSH    =   2000.000    ,
... ...
```

2. Check analysis time, number of ensemble members, as well as the actual analysis variables and the background type. The maximum and minimum values for surface pressure are printed for a sanity check:

```
analysis time 2014040512
         10  members
number of background forecast times to be updated =           1
first-guess forecast hours for analysis = 06
          5 3d vars to update
total of         321  2d grids will be updated (including ps)
using multiplicative inflation based on Pa/Pb
Vars in Rad-Jacobian (dims)
--------------------------
sst                           0
enkf_main:  number of threads           1
ensemble mean first guess surface pressure:
  516.109537854119        1051.19857562343
```

3. Statistics of the ensemble prior
After many lines describing the bias correction coefficents, the inventory of the observation number, observation types from the input diag* files, and the time spent reading in each observation type, statistics of the ensemble priors fit to all observations are provided for each region (NH, SH, TR):

```
 innovation statistics for prior:
 conventional obs
 region, obtype, nobs, bias, innov stdev, sqrt(S+R), sqrt(S), sqrt(R):
NH    all ps  12059 -0.700E-01  0.122E+01  0.151E+01  0.499E+00  0.143E+01
TR    all ps   2184  0.402E+00  0.118E+01  0.141E+01  0.476E+00  0.133E+01
SH    all ps    849 -0.102E+00  0.125E+01  0.148E+01  0.563E+00  0.137E+01
NH     all t  20598  0.929E-01  0.165E+01  0.121E+01  0.387E+00  0.115E+01
TR     all t   6126  0.218E+00  0.153E+01  0.113E+01  0.418E+00  0.105E+01
SH     all t   1169  0.141E+00  0.172E+01  0.113E+01  0.417E+00  0.105E+01
NH    all uv  49010  0.199E+00  0.312E+01  0.250E+01  0.951E+00  0.231E+01
TR    all uv  20980  0.651E-01  0.335E+01  0.265E+01  0.127E+01  0.232E+01
SH    all uv   4682 -0.153E+00  0.375E+01  0.247E+01  0.114E+01  0.219E+01
NH     all q   7936 -0.335E-01  0.181E+00  0.207E+00  0.534E-01  0.200E+00
TR     all q   2951 -0.119E-01  0.180E+00  0.214E+00  0.770E-01  0.200E+00
SH     all q    543 -0.163E-01  0.212E+00  0.213E+00  0.730E-01  0.200E+00
```

This table should be checked in order to determine if the inflation is appropriate. As mentioned in the regional EnKF example, the goal is to make the total ensemble spreads of priors ($\sqrt{(S+R)}$ match the innovations (innov std) as much as possible. Because this is a global run using only conventional observation data, we can see that all regions (NH,TR,SH) are listed and there are no statistics for radiances. We can see certain obtypes, particularly in the uv has fairly close values. On the other hand, there are still regions/obtypes that have noteable differences, mainly associated with larger observation errors ($\sqrt{(R)}$). Users should consider tuning of inflation and localization, which is typically determined using cases with multiple assimilation cycles. Refer to section 4.2 of this User's Guide for more information on tuning.

4. Domain and observation partition:

```
npts =       131072
min/max number of points per proc =         1598        1710
time to do model space decomp =   1.697483193129301E-002
nobstot =       129087
min/max number of obs per proc =         1613        1614
time to do ob space decomp =   5.538591649383307E-004
sending out observation prior ensemble perts from root ...
nobstot*nanals              1290870
npts*ndim             42074112
... took   6.416106596589088E-003  secs
```

The analysis variables and the observations are distributed to different processors. We can see in this case that the min/max number of points per processor are 1598 and 1710, respectively. Similarly, we can see that the min/max number of observations per processor are 1613 and 1614, respectively, indicating that the observations are well dispersed among the processors.

```
min/max pressi           1   516.1096       1051.199
min/max pressi           2   513.3593       1045.597
min/max pressi           3   510.2454       1039.249
min/max pressi           4   506.7440       1032.063
min/max pressi           5   502.8375       1023.943
... ...
```

Additionally, check the minimum and maximum values of the fields at each vertical level as a quick sanity check.

5.4.4 Diagnose EnKF Analysis Results

At the bottom of the standard output file, there are several output statistics and tables that are helpful for users to diagnose the quality of the EnKF analysis.

5. Statistics of the ensemble analysis

```
global ps prior std. dev min/max =   7.2904140E-02   3.055338
NH mean ps prior standard deviation =   0.5657563
NH mean ps posterior standard deviation (before inflation)=   0.3681831
NH mean ps posterior standard deviation (after inflation) =   0.5323756
NH mean ps inflation =    1.653412
SH mean ps prior standard deviation =   0.6475344
SH mean ps posterior standard deviation (before inflation)=   0.6009755
SH mean ps posterior standard deviation (after inflation) =   0.6399752
SH mean ps inflation =    1.086266
TR mean ps prior standard deviation =   0.4878409
TR mean ps posterior standard deviation (before inflation)=   0.4240963
TR mean ps posterior standard deviation (after inflation) =   0.4770048
TR mean ps inflation =    1.170121
time in inflate_ens =  5.236316868104041E-002 on proc           0
```

This section is important for checking whether the values of the inflation are reasonable by looking at a summary of the maximum and minimum values. Also, viewing the regional averaged statistics before and after inflation. For this case we can see for all regions, larger standard deviation after the inflation are reported, which is more consistent with the prior standard deviation.

6. Spread inflation of the analysis ensemble

```
 innovation statistics for posterior:
 conventional obs
 region, obtype, nobs, bias, innov stdev, sqrt(S+R), sqrt(S), sqrt(R):
NH    all ps  12059  0.538E-01  0.912E+00  0.144E+01  0.200E+00  0.143E+01
TR    all ps   2184  0.243E+00  0.102E+01  0.136E+01  0.285E+00  0.133E+01
SH    all ps    849 -0.755E-01  0.978E+00  0.142E+01  0.388E+00  0.137E+01
NH     all t  20598  0.590E-01  0.147E+01  0.117E+01  0.196E+00  0.115E+01
TR     all t   6126  0.177E+00  0.135E+01  0.107E+01  0.222E+00  0.105E+01
SH     all t   1169  0.100E+00  0.148E+01  0.108E+01  0.246E+00  0.105E+01
NH    all uv  49010  0.765E-01  0.269E+01  0.236E+01  0.478E+00  0.231E+01
TR    all uv  20980  0.116E-02  0.274E+01  0.240E+01  0.605E+00  0.232E+01
SH    all uv   4682 -0.981E-01  0.303E+01  0.227E+01  0.591E+00  0.219E+01
NH     all q   7936 -0.333E-01  0.166E+00  0.202E+00  0.268E-01  0.200E+00
TR     all q   2951  0.369E-02  0.148E+00  0.203E+00  0.370E-01  0.200E+00
SH     all q    543 -0.188E-01  0.187E+00  0.204E+00  0.393E-01  0.200E+00
```

After the EnKF analysis, it is important to check the innovation statistics, as previously discussed in the regional test case.

6

EnKF Basic Concepts and Code Structure

This chapter briefly describes basic concepts and the main code structure used in the current implementation of the NOAA EnKF in the form of EnSRF. Please note there are also other EnKF algorithms provided in this EnKF system. We are working on documenting the other algorithms and will complete the User's Guide in the future.

6.1 Basic Concepts (in the Form of EnSRF)

6.1.1 Analysis Variables

In theory, EnKF can use any of the model prognostic variables as analysis variables as long as there exist meaningful/clear correlations between the variables and observations. Typically, for hydrostatic global models, horizontal wind components, temperature, water vapor, surface pressure, and ozone are used as analysis variables. For non-hydrostatic meso-scale models, like WRF, the vertical component of wind, rain/cloud related water content variables, and surface variables could be used as additional analysis variables.

6.1.2 Update of Analysis Variables

The minimum error-variance estimate of the analyzed variables $\boldsymbol{X}^a$ is given by the traditional Kalman filter update equation,

$$\boldsymbol{X}^a = \boldsymbol{X}^b + \boldsymbol{K}(\boldsymbol{y}^o - \boldsymbol{H}\boldsymbol{X}^b) \tag{6.1}$$

$$\boldsymbol{K} = \boldsymbol{P}^b\boldsymbol{H}^T(\boldsymbol{H}\boldsymbol{P}^b\boldsymbol{H}^T + \boldsymbol{R})^{-1} \tag{6.2}$$

Where

$\boldsymbol{X}^b$ an m-dimensional background model forecast (i.e., prior)
$\boldsymbol{X}^a$ an m-dimensional analyses at model grids (i.e., posterior)
$\boldsymbol{y}^o$ a p-dimensional set of observations
$\boldsymbol{H}$ the operators that convert the model state to the observation space
$\boldsymbol{P}^b$ the mm-dimensional background-error covariance matrix
$\boldsymbol{R}$ the pp-dimensional observation-error covariance matrix
$\boldsymbol{K}$ the Kalman gain

Here, the Kalman gain is a function of the multivariate covariances of the model state variables and observations and the operator matrix that relates the model state to the observations. This update process is basically similar to a simple optimal interpolation (OI) scheme, where the Kalman gain is set to static.

The update equations (6.1) and (6.2) can be solved using ensemble technique. The Kalman gain can be estimated and propagated using a set of ensemble forecasts. Expressing the model state vector of the analysis variables as an ensemble mean (denoted by an overbar) and a deviation from the mean (denoted by a prime), the update equations for EnSRF ([6]) are written as:

$$\overline{\boldsymbol{X}}^a = \overline{\boldsymbol{X}}^b + \boldsymbol{K}(\boldsymbol{y}^o - \boldsymbol{H}\overline{\boldsymbol{X}}^b) \tag{6.3}$$

$$\boldsymbol{X}'^a = \boldsymbol{X}'^b - \tilde{\boldsymbol{K}}\boldsymbol{H}\boldsymbol{X}'^b \tag{6.4}$$

$$\tilde{\boldsymbol{K}} = \alpha\boldsymbol{K} \tag{6.5}$$

$$\alpha = \left[1 + \sqrt{R/(\boldsymbol{H}\boldsymbol{P}^b\boldsymbol{H}^T + R)}\right]^{-1} \tag{6.6}$$

Where $\boldsymbol{K}$ is the Kalman gain defined by (6.2) and estimated using the ensemble method described in the following section. $\tilde{\boldsymbol{K}}$ is the gain used to update ensemble deviations from the ensemble mean. Here, the observational error covariance is assumed uncorrelated, that is, $\boldsymbol{R}$ is diagonal. Then observations can be assimilated serially, one at a time, so that the analysis after assimilation of the Nth observation becomes the background estimate for assimilating the (N+1)th observation. For an individual observation, R and $\boldsymbol{H}\boldsymbol{P}^b\boldsymbol{H}^T$ are scalars, $\boldsymbol{K}$ and $\tilde{\boldsymbol{K}}$ are vectors of the same dimension as the model state vector (before applying localization). Both $\boldsymbol{K}$ and $\tilde{\boldsymbol{K}}$ are calculated from the prior ensemble of the observation being assimilated and each of the analyses variables on the model grids individually.

Please note EnSRF is a revised EnKF that eliminates the necessity to perturb the observations. Therefore, the equation (6.4) does not contain the equivalent observation term to the one in the equation (6.3). However, this EnKF system itself provides an option to perturb the observations as well. In the future, we will add the description of the EnKF algorithm using perturbed observations into this user's guide.

In the EnKF framework, there is no need to compute and store the full matrix $\boldsymbol{P}^b$. Instead, $\boldsymbol{P}^b\boldsymbol{H}^T$ and $\boldsymbol{H}\boldsymbol{P}^b\boldsymbol{H}^T$ are estimated statistically from an ensemble of model forecasts/background. Specifically, these two quantities are obtained as:

$$\boldsymbol{P}^b\boldsymbol{H}^T = \overline{\boldsymbol{X}'^b(\boldsymbol{H}\boldsymbol{X}'^b)^T} = \sum_{i=1}^{n} \boldsymbol{X}_i'^b(\boldsymbol{H}\boldsymbol{X}_i'^b)^T/(n-1) \tag{6.7}$$

$$\boldsymbol{H}\boldsymbol{P}^b\boldsymbol{H}^T = \overline{\boldsymbol{H}\boldsymbol{X}'^b(\boldsymbol{H}\boldsymbol{X}'^b)^T} = \sum_{i=1}^{n} \boldsymbol{H}\boldsymbol{X}'^b(\boldsymbol{H}\boldsymbol{X}'^b)^T/(n-1) \tag{6.8}$$

where $\boldsymbol{n}$ is the ensemble size of model forecasts; $\boldsymbol{i}$ is the index of each individual ensemble member. The expected analyses error covariance at the model grids after assimilation is given by

$$\boldsymbol{P}^a = (\boldsymbol{I} - \boldsymbol{K}\boldsymbol{H})\boldsymbol{P}^b(\boldsymbol{I} - \boldsymbol{K}\boldsymbol{H})^T \tag{6.9}$$

6.1.3 Updates of Observation Priors

In a serial assimilation of observations, the model first-guess or backgrounds at model grids are updated by one single observation at a time. For the assimilation of next observation, the first-guess of the next observation (observation priors) needs to be re-computed using the updated model background and a forward observation operator. This process is straightforward while running on a single processor computer, but is not efficient in a parallel computing environment, particularly when the first-guess of the observations are all pre-calculated.

Alternatively, the first-guess of the next observation can also be updated by the observation being assimilated, similar to the update to the model variables, so that re-computing the first-guess using the observational operators is not needed (see [1]). After the update of the first guess of model variables, the Nth observation being assimilated is also used to update the first-guess of the (N+1)th and next observations within the localization distance. This process can be expressed in the following update equations, similar to the equations (6.3)-(6.6):

$$\overline{\boldsymbol{Z}}^a = \overline{\boldsymbol{Z}}^b + \boldsymbol{K}_z(y^o - \boldsymbol{H}\overline{\boldsymbol{X}}^{\boldsymbol{b}}) \tag{6.10}$$

$$\boldsymbol{Z}'^a = \boldsymbol{Z}'^b - \alpha\boldsymbol{K}_z\boldsymbol{H}\boldsymbol{X}'^b \tag{6.11}$$

$$\boldsymbol{K}_z = \boldsymbol{Z}'^b(\boldsymbol{H}\boldsymbol{X}'^b)^T(\boldsymbol{H}\boldsymbol{P}^b\boldsymbol{H}^T + \boldsymbol{R})^{-1} \tag{6.12}$$

Where

y^o is the observation being assimilated.
$\boldsymbol{Z}^b$ is the first-guess of the next unassimilated observation
$\boldsymbol{Z}^a$ is the updated first-guess of the next unassimilated observation
$\boldsymbol{K}_z$ is the Kalman gain between the first-guess of the observation being assimilated and next unassimilated observation

6.1.4 Assimilation Order and Adaptive Thinning of Observations

In realistic assimilation systems, observations may have a non-linear relationship with the analysis variables. In addition,sampling errors are common due to the limited ensemble sizes. As a result, the achieved analysis can depend on the order of the observations assimilated. There are three options for choosing the orders of the observations for assimilation:

1. Assimilate observations in the order they are read in (default). This seems a reasonable choice for assimilating "BEST" observation types first (like radiosonde winds)
2. Shuffle the observations randomly before assimilating
3. Assimilate in order of increasing predicted observation analysis variance relative to the prior

For the third option, the predicted observational analysis variance against the first-guess in the observational space is defined as (for details, see [8] and [7]):

$$\boldsymbol{HP^aH^T}/\boldsymbol{HP^bH^T} = \boldsymbol{R}/(\boldsymbol{HP^bH^T} + \boldsymbol{R}) \tag{6.13}$$

Note that the predicted variance is based on the observation priors'variance as if the observation is assimilated alone, and does not include the effect of the assimilation of other observations.

The observations can be further thinned adaptively. It is done via the updated estimation of the predicted analysis variance of next observation. If the predicted analysis variance for one observation is very close to the prior of this observation, the impact of this observation is expected to be very small and, therefore, it can be skipped. The threshold is set by the namelist parameter *paoverpb_thresh*.

6.1.5 Ensemble Spread Inflation

EnSRF uses a multiplicative inflation to inflate analyses/posterior ensemble spread back to the one of first-guess. The amount of inflation is given at each analysis grid point by:

$$\begin{aligned}
\sigma^b &= \sqrt{\sum_{i=1}^{n}(\boldsymbol{X'}^b_i)^2/(n-1)} \\
\sigma^a &= \sqrt{\sum_{i=1}^{n}(\boldsymbol{X'}^a_i)^2/(n-1)} \\
r &= \left(\beta\frac{\sigma^b-\sigma^a}{\sigma^a}+1\right) \\
r\mathbf{X}'^a_i &\rightarrow \mathbf{X}'^a_i(\text{inflated})
\end{aligned} \tag{6.14}$$

where σ^b is the prior/first-guess ensemble standard deviation; σ^a is posterior/analyses ensemble standard deviation (before inflation); r is the inflation factor applied to each ensemble member deviation from the ensemble mean; β is a tunable namelist parameter (*analpertwt*) defined in module params: If $\beta = 1$, ensemble is inflated so posterior standard deviation becomes the same as prior; If $\beta = 0$, there is no inflation.

For a given value of β, the inflation factor is proportional to the amount of ensemble spread reduction by assimilation of observations, normalized by the analyses ensemble spread. As a result, the inflation is in general larger where observations are denser or have larger impact. The actual inflation factor can be quite different for each variable and cross vertical and horizontal grids. If the smoothing namelist parameter $(smoothparm) > 0$, the estimated inflation factor is smoothed using a Gaussian spectral filter with an e-folding scale of $smoothparm$. The minimum and maximum values allowed can be controlled by namelist parameters. In additions, extra inflation can be obtained by adding random noise from a climatology distribution of the model errors ([7]). The amount of the random noises to be added can be controlled through namelist parameters as well.

The total amount of inflation from these two inflation schemes should meet the following relationship ([3]), as close as possible:

$$\left\langle (\boldsymbol{y}^o - \boldsymbol{H}\overline{\boldsymbol{X}}^b)(\boldsymbol{y}^o - \boldsymbol{H}\overline{\boldsymbol{X}}^b) \right\rangle = (\boldsymbol{H}\boldsymbol{P}^b\boldsymbol{H}^T + \boldsymbol{R}) \tag{6.15}$$

Satisfaction of this equation ensures that the total ensemble spreads (ensemble spreads plus observational error covariance, right side of the equation) is a reasonable estimation of the RMS errors of observation priors against observations (left side of the equation). This relationship justifies that the ensemble system works reasonably well. Moreover, the ensemble spreads should be tuned appropriately relative to observation errors (assuming these errors are correctly set). Substantially smaller ensemble spreads could lead to underweight of observations. As a result, EnKF may ignore the observations for assimilation and/or lead to an over-divergent ensemble.

6.1.6 Covariance Localization

To reduce the impact of spurious ensemble covariance on the update of both analyses variables at model grids and the first-guess of observations (observation priors), localization is applied to the covariance (Kalman gains) in the equations (6.7) and (6.12) in both horizontal and vertical. The function of [2] is used in horizontal localization. It uses a 5th order compact polynomial and the impact of observations is gradually reduced to zero at the specified cutoff distance. The scale height $-\log{(P/P_{ref})}$ is used in vertical localization.

The localization distance is an important tunable parameter for a successful analysis of EnSRF. Tuning the localization distance depends on several factors, including the ensemble size used, model grid resolutions, weather scenario, etc. In general, a larger ensemble size allow a longer localization distance. But assimilation with a smaller ensemble size may benefit from a reduced localization distance to reduce the impact of spurious covariance. In addition, assimilation of higher-resolution observations (e.g., radar data) may require a much shorter localization distance.

6.1.7 Adaptive Radiance Bias Correction with EnSRF

An adaptive radiance bias correction procedure is used for the satellite radiance data assimilation in EnSRF. The first-guess of the radiance observations are updated by the radiance observations using the assimilation procedure outlined in the section (6.2.2). The updated innovations (O-B) are then used to update the coefficients of the radiance bias correction scheme. The first-guess of the radiance observations are updated again using the updated bias correction coefficients. This process may be repeated/iterated multiple times until the updated first-guess of the radiance observations and bias correction coefficients converge.

6.2 EnSRF Code Structure and Key Functions

6.2.1 Main Code Tree

The code structure for main code *enkf_main.f90*:

Step	Code	Explanation
initialize	call mpi_initialize	initialize MPI
	call init_rad	Initial radinfo variables
	call read_namelist	read namelist (enkf.nml)
	call mpi_initialize_io	initialize MPI communicator for IO tasks
	call init_rad_vars	Initialize derived radinfo variables
prepare data	call getgridinfo	read horizontal grid information and pressure fields from forecast ensemble mean file
	call readobs	Read observations, observation priors for each ensemble member (from diag** files generated by GSI forward operators). initial screening.
	call print_innovstats	print innovation statistics for prior
	if (readin_localization) call read_locinfo	read in vertical profile of horizontal and vertical localization length scales, set values for each obs
	call load_balance	do load balancing (partitioning of grid points, observations among processors)
	call read_ensemble	read in prior ensemble members, distribute pieces to each task

analysis	*if(letkf_flag) then*
	call letkf_update
	else
	call enkf_update
		Update state variables, observation priors, and radiance bias correction coefficients with EnSRF or the EnKF with perturbed observations

write results	call inflate_ens	Inflate posterior ensemble using the multiplicative inflation scheme
	call print_innovstats	print innovation statistics for posterior
	call radinfo_write	write out bias coeffs on root
	call write_ensemble	write out analysis ensemble
clean up	call obsmod_cleanup call gridinfo_cleanup call statevec_cleanup call loadbal_cleanup call mpi_cleanup	

Please note additive inflation may be added to the posterior ensemble offline.

6.2.2 Driver of Serial Ensemble Square Root Filter

If using the EnKF algorithm (if $letkf\_flag = .false.$), the priors of model analyses variables, observation priors, and bias correction coefficients are updated by subroutine $enkf\_update$ in $enkf.f90$. The code structure of $enkf\_update$ is:

Step	Code

1:
Determine the order of all observations for assimilation according to the namelist choice
	<pre>if (iassim_order == 1) then
	! create random index array so obs are assimilated
	! in random order.
	else if (iassim_order .eq. 2) then
	! assimilate obs in order of increasing HPaHT/HPbHT
	else
	! assimilate obs in order they were read in
	end if</pre>

2 begin outer loop	*do niter=1,numiter*

2.1
reset ob error to account for gross errors
	<pre> if (niter > 1 .and. varqc) then
	    if (huber) then ! "huber norm" QC
	    else ! "flat-tail" QC.
	    endif
	  else
	      oberrvaruse(nob) = oberrvar(nob)
	  end if</pre>

2.2: assimilation loop over all observations, one observation at a time	<pre> obsloop: do nobx=1,nobstot .... end do obsloop ! loop over obs to assimilate</pre>
2.3	<pre> ! make sure posterior perturbations still have ! zero mean ! distribute the O-A stats to all processors ! satellite bias correction update</pre>
2: end outer loop	*enddo ! niter loop*

The details of the step 2.2 is listed below:

Step	Code in Step 2.2

1:
which observation to assimilate next
	if (iassim_order == 2) then
	else
	nob = indxassim(nobx)
	endif

2. calculate : $\boldsymbol{HP^bH^T}$ $(\boldsymbol{HP^bH^T}+\boldsymbol{R})^{-1}$ $\boldsymbol{R}/(\boldsymbol{HP^bH^T}+\boldsymbol{R})$	hpfht = sum(anal_obchunk(:,nob1)**2)*r_nanalsm1 hpfhtoberrinv=one/(hpfht+oberrvaruse(nob)) paoverpb = oberrvar(nob)/(hpfht + oberrvar(nob))

3.
calculate Equation 6.6
	if (deterministic) then
	! EnSRF.
	obganl = -anal_obtmp/(one+sqrt(oberrvaruse(nob)*hpfhtoberrinv))
	else
	! perturbed obs EnKF.
	end if

4.Calculate Kalman gain (*kfgain*) (K, Equation (6.5)) and add the analyses increments of the observation being assimilated to the ensemble mean (*ensmean_chunk* and perturbations of analyses variables at model grids (*anal_chunk*) (Equations (6.3) and (6.4))	if (nf2 > 0) then
	do ii=1,nf2 ! loop over nearby horiz grid points
	do nb=1,nbackgrounds ! loop over background time levels
	do nn=nn1,nn2
	nnn=index_pres(nn)
	if (taperv(nnn) > zero) then
	! gain includes covariance localization update all time levels
	kfgain=taperv(nnn)*sum(anal_chunk(:,i,nn,nb)*anal_obtmp)
	! update mean.
	ensmean_chunk(i,nn,nb) = ensmean_chunk(i,nn,nb) + kfgain*obinc_tmp
	! update perturbations.
	anal_chunk(:,i,nn,nb) = anal_chunk(:,i,nn,nb) + kfgain*obganl(:)
	end if
	end do
	end do ! end loop over background time levels.
	end do ! end loop over nearby horiz grid points
	end if ! if .not. lastiter or no close grid points

5. Calculate the Kalman gain *kfgain* (Kz: Equation (6.12)) and add the analyses increments of the observation being assimilated to the nearby observation priors *ensmean_obchunk, and anal_obchunk* (Equations (6.10) and (6.11)).	if (nf > 0) then
	do nob1=1,nf
	! gain includes covariance localization.
	kfgain = taper_disob(nob1)* &
	taper(lnsig*lnsiglinv)*taper(obt*obtimelinv)* &
	sum(anal_obchunk(:,nob2)*anal_obtmp)*hpfhtcon
	! update mean.
	ensmean_obchunk(nob2) = ensmean_obchunk(nob2) + kfgain*obinc_tmp
	! update perturbations.
	anal_obchunk(:,nob2) = anal_obchunk(:,nob2) + kfgain*obganl
	end do

A

Content of Namelist

The following are lists and explanations of the EnKF namelist variables. Users can also check file *params.f90* for the details.

Section **nam_enkf**

Variable Name	Description	Data Type	Default
datein	Analysis date in YYYYMMDDHH	integer	0
datapath	path to data directory (include trailing slash)	Character (len=500)	""
iassim_order	= 0 for the order they are read in, =1 for random order = 2 for order of predicted posterior variance reduction (based on prior)	integer	0
covinflatemax	maximum inflation	real(r_single)	1.e30
covinflatemin	minimum inflation	real(r_single)	1.0
deterministic	if true, use EnSRF w/o perturbed obs. if false, use perturbed obs EnKF.	logical	true
sortinc	if false, re-order obs to minimize regression errors as described in Anderson (2003).	logical	true
corrlengthnh	length for horizontal localization (in km) in north hemisphere	real(r_single)	2800
corrlengthtr	length for horizontal localization (in km) in tropic	real(r_single)	2800
corrlengthsh	length for horizontal localization (in km) in south hemisphere	real(r_single)	2800

Variable Name	Description	Data Type	Default
varqc	Turn on varqc	logical	false
huber	use huber norm instead of "flat-tail"	logical	fales
nlons	number of lons	integer	0
nlats	Number of lats	integer	0
smoothparm	smoothing parameter for inflation (-1 for no smoothing)	real(r_single)	-1
readin_localization	If true, read in localization length scales from an external file	logical	false
zhuberleft	Parameter for "huber norm" QC	real(r_single)	1.e30
zhuberright	Parameter for "huber norm" QC	real(r_single)	1.e30
obtimelnh	observation time localization in hours over north hemisphere	real(r_single)	25.925
obtimeltr	observation time localization in hours over tropic	real(r_single)	25.925
obtimelsh	observation time localization in hours over south hemisphere	real(r_single)	25.925
reducedgrid	Do smooth in a reduced grid with a variable number on longitudes per latitude. The number of longitudes is chosen so that the zonal grid spacing is approximately the same as at the equator	logical	false
lnsigcutoffnh	length for vertical localization in ln(p) over north hemisphere for conventional observation	real(r_single)	2.0
lnsigcutofftr	length for vertical localization in ln(p) over tropic conventional observation	real(r_single)	2.0
lnsigcutoffsh	length for vertical localization in ln(p) over south hemisphere for conventional observation	real(r_single)	2.0
lnsigcutoffsatnh	length for vertical localization in ln(p) over north hemisphere for satellite radiance observation	real(r_single)	-999.0
lnsigcutoffsattr	length for vertical localization in ln(p) over tropic satellite radiance observation	real(r_single)	-999.0
lnsigcutoffsatsh	length for vertical localization in ln(p)over south hemisphere for satellite radiance observation	real(r_single)	-999.0
lnsigcutoffpsnh	length for vertical localization in ln(p) over north hemisphere for surface pressure observation	real(r_single)	-999.0
lnsigcutoffpstr	length for vertical localization in ln(p) over tropic surface pressure observation	real(r_single)	-999.0
lnsigcutoffpssh	length for vertical localization in ln(p) over south hemisphere for surface pressure observation	real(r_single)	-999.0

Variable Name	Description	Data Type	Default
analpertwtnh	adaptive posterior inflation parameter over north hemisphere: 1 means inflate all the way back to prior spread	real(r_single)	0.0
analpertwtsh	adaptive posterior inflation parameter over tropic: 1 means inflate all the way back to prior spread	real(r_single)	0.0
analpertwttr	adaptive posterior inflation parameter over south hemisphere: 1 means inflate all the way back to prior spread	real(r_single)	0.0
sprd_tol	tolerance for background check: observations are not used if they are more than sqrt(S+R) from mean, where S is ensemble variance and R is observation error variance.	real(r_single)	9.9e31
nlevs	total number of levels	integer	0
nanals	number of ensemble members	integer	0
nvars	number of 3d variables to update. For hydrostatic models, typically 5 (u,v,T,q,ozone).	integer	5
saterrfact	factor to multiply sat radiance errors	real(r_single)	1.0
univaroz	If true, ozone observations only affect ozone	logical	true
regional	If true, analysis is for regional	logical	false
use_gfs_nemsio	If true, GFS background is in NEMS format	logical	false
paoverpb_thresh	if observation space posterior variance divided by prior variance less than this value, observation is skipped during serial processing. 1.0 = don't skip any obs	(r_single)	1.0
latbound	definition of tropics and mid-latitudes (for inflation).	real(r_single)	25.0
delat	width of transition zone	real(r_single)	10.0
pseudo_rh	use 'pseudo-rh' analysis variable, as in GSI	logical	false
numiter	number of times to iterate state/bias correction update. (only relevant when satellite radiances assimilated, i.e. nobs_sat>0)	integer	1.0

Variable Name	Description	Data Type	Default
biasvar	background error variance for rad bias coeffs (used in radbias.f90). Default is (old) GSI value. if negative, bias coeff error variace is set to - biasvar/N, where N is number of obs per instrument/channel. if newpc4pred is .true., biasvar is not used - the estimated analysis error variance from the previous cycle is used instead (same as in the GSI).	real(r_single)	0.1
lupd_satbiasc	if performing satellite bias correction update	logical	true
cliptracers	if true, tracers are clipped to zero when read in, and just before they are written out.	logical	true
simple_partition	partition obs for enkf using Graham's rule	logical	true
adp_anglebc	turn off or on the variational radiance angle bias correction	logical	false
angord	order of polynomial for angle bias correction	Integer	0
newpc4pred	controls preconditioning due to sat-bias correction term	logical	
nmmb	If true, ensemble forecast is NMMB	logical	false
iau		logical	false
nhr_anal	background forecast time for analysis	integer	6
letkf_flag	If true, do LETKF	logical	false
boxsize	Observation box size for LETKF (deg)	real(r_single)	90.0
massbal_adjust	mass balance adjustment for GFS	logical	false
use_edges	logical to use data on scan edges (.true.=to use)	logical	true
emiss_bc	If true, turn on emissivity bias correction	logical	false

Section **nam_wrf**

Variable Name	Description	Data Type	Default
arw	regional dynamical core ARW	logical	false
nmm	regional dynamical core NMM	logical	true
doubly_periodic		logical	true

Section **satobs_enkf**

Variable Name	Description	Data Type	Default
sattypes_rad	strings describing the satellite data type (which form part of the diag* filename).	character(len=20) array (nsatmax_rad)	’""
dsis	strings corresponding to sattypes_rad which correspond to the names in the NCEP global_satinfo file.	character(len=20) array (nsatmax_rad)	""

Section **ozobs_enkf**

Variable Name	Description	Data Type	Default
sattypes_oz	strings describing the ozone satellite data type (which form part of the diag* filename)	character(len=20) array (nsatmax_oz)	""

Bibliography

[1] J. Anderson and N. Collins. Scalable implementations of ensemble filter algorithms for data assimilation. *Journal of Atmospheric and Oceanic Technology*, 24:1452–1463, 2007.

[2] G. Gaspari and S. E. Cohn. Construction of correlation functions in two and three dimensions. *Quarterly Journal of the Royal Meteorological Society*, 125.

[3] P. L. Houtekamer and H. L. Mitchell. Ensemble kalman filtering. *Quarterly Journal of the Royal Meteorological Society*, 131(613):3269–3289, 2005.

[4] B. R. Hunt, E. J. Kostelich, and I. Szunyogh. Efficient data assimilation for a spatiotemporal has: a local ensemble transform kalman filter. *Physica D*, 230:112–126, 2007.

[5] H. Shao, J. Derber, X.-Y. Huang, M. Hu, K. Newman, D. Stark, M. Lueken, C. Zhou, L. Nance, Y.-H. Kuo, and B. Brown. Bridging research to operations transitions: Status and plans of community gsi. *Bulletin of the American Meteorological Society*, 2016.

[6] J. Whitaker and T. Hamill. Ensemble data assimilation without perturbed observations. *Mon. Wea. Rev.*, 130:1913–1924, 2002.

[7] J. Whitaker and T. Hamill. Evaluating methods to account for system errors in ensemble data assimilation. *Mon. Wea. Rev.*, 140:3078–3089, 2012.

[8] J. Whitaker, T. Hamill, X. Wei, Y. Song, and Z. Toth. Ensemble data assimilation with the ncep global forecast system. *Mon. Wea. Rev.*, 436:463–482, 2008.