NASA/TM–2017-104606 / Vol. 48

Technical Report Series on Global Modeling and Data Assimilation, Volume 48

Randal D. Koster, Editor

Description of the GMAO OSSE for Weather Analysis Software Package: Version 3

Ronald M. Errico, Nikki C. Privé, David Carvalho, Meta Sienkiewicz, Amal El Akkraoui, Jing Guo, Ricardo Todling, Will McCarty, William M. Putman, Arlindo da Silva, Ronald Gelaro, Isaac Moradi

National Aeronautics and
Space Administration

Goddard Space Flight Center
Greenbelt, Maryland 20771

July 2017

NASA/TM–2017-104606/Vol. 48

Technical Report Series on Global Modeling and Data Assimilation, Volume 48

Randal D. Koster, Editor

Description of the GMAO OSSE for Weather Analysis Software Package: Version 3

Ronald M. Errico
Morgan State University, Baltimore, MD

Nikki C. Privé
Morgan State University, Baltimore, MD

David Carvalho
Universities Space Research Association, Columbia, MD

Meta Sienkiewicz
Science Systems Applications Inc., Lanham, MD

Amal El Akkraoui
Science Systems Applications Inc., Lanham, MD

Jing Guo
Science Systems Applications Inc., Lanham, MD

Ricardo Todling
Goddard Space Flight Center, Greenbelt, MD

Will McCarty
Goddard Space Flight Center, Greenbelt, MD

William M. Putman
Goddard Space Flight Center, Greenbelt, MD

Arlindo da Silva
Goddard Space Flight Center, Greenbelt, MD

Ronald Gelaro
Goddard Space Flight Center, Greenbelt, MD

Isaac Moradi
Earth System Science Interdisciplinary Center, College Park, MD

National Aeronautics and
Space Administration

Goddard Space Flight Center
Greenbelt, Maryland 20771

July 2017

Trade names and trademarks are used in this report for identification only. Their usage does not constitute an official endorsement, either expressed or implied, by the National Aeronautics and Space Administration.

Level of Review: This material has been technically reviewed by technical management

Available from

NASA STI Program
Mail Stop 148'
NASA Langley Research Center
Hampton, VA 23681–2199

National Technical Information Service
5285 Port Royal Road
Springfield, VA 22161
703–605–6000

This document is available for downloaded from the NASA Technical Reports Server (NTRS).
https://ntrs.nasa.gov

Abstract

The Global Modeling and Assimilation Office (GMAO) at the NASA Goddard Space Flight Center has developed software and products for conducting observing system simulation experiments (OSSEs) for weather analysis applications. Such applications include estimations of potential effects of new observing instruments or data assimilation techniques on improving weather analysis and forecasts. The GMAO software creates simulated observations from nature run (NR) data sets and adds simulated errors to those observations. The algorithms employed are much more sophisticated, adding a much greater degree of realism, compared with OSSE systems currently available elsewhere. The algorithms employed, software designs, and validation procedures are described in this document. Instructions for using the software are also provided.

Contents

A Subroutine Calling Sequences

List of Tables

List of Figures

Chapter 1

Introduction

Observing System Simulation Experiments (OSSEs) are applications of data assimilation systems (DAS) conducted in an entirely simulated environment. Rather than using real observations of the real atmosphere that have real errors, OSSEs employ simulated observations of simulated atmospheres that have simulated errors. For this reason, they are not restricted to assimilating only existing observation types but can incorporate future envisioned ones as well. Also, unlike the real atmosphere, for which the true state is never known precisely, the simulated atmosphere that provides the truth for the analysis is known perfectly in an OSSE. These two properties underlie the OSSEs usefulness.

When developing a new observing system, many competing designs usually present themselves, each with its own costs and benefits. In an OSSE, simulated data having the planned spatial and temporal distributions and error characteristics of selected system designs are assimilated along with simulated data representing other expected observations. The resulting data assimilation products are then examined with regard to the particular design's impacts. In this way, OSSEs can estimate some of the expected benefits of each design or system before funds are expended actually deploying the instruments. Once a flexible, validated OSSE framework is available, little cost or time is required to conduct the required comparative investigations. This traditionally has been the motivation for OSSEs, since development and deployment of new observing systems that are generally satellite-based are very expensive and require long lead times.

Modern DAS are very complex and fairly accurate. Root–mean–squared errors of temperature analyses, for example, are currently less than about 0.7K. Given their complexity, evaluation of data assimilation systems is difficult, particularly in terms of understanding the mechanisms responsible for various results. Statistics of analysis error can only be inferred indirectly using generally unsupported assumptions. Determination of realizations of actual analysis error is virtually impossible. In the OSSE context, however, the simulated analysis error can be determined perfectly given that the simulated truth is known. This determination can be exploited to examine the behavior of a data assimilation systems more directly and thoroughly. This is a second general application of OSSEs.

All OSSEs are intended to estimate corresponding results that would be produced in a real context. As for any simulation, the utility of an OSSE therefore depends on how well it can reproduce the relevant behavior of a DAS that is applied to corresponding real observations. Any OSSE framework must accordingly be first tested by conducting experiments that closely mimic real DAS experiments. In such a test, a variety of metrics produced by the OSSE are compared to those produced in the established DAS experiment. This is called validating the OSSE.

Although OSSEs have been performed since the early 1980s, their validation has been minimal. That step in their application has too often been effectively neglected. Consequently, OSSEs have been poorly regarded by many DAS experts. Proper validation, however, will be the hallmark of the GMAO OSSE framework. This should render OSSE results acceptable and useful to the DAS and observing system communities.

The GMAO OSSE framework began as a NASA contribution to the Joint OSSE effort that was an informal collaboration between ECMWF, NCEP, NASA, Simpson Weather, NOAA/ETL, KNMI, and others between 2006 and 2009. As a primary contribution, ECMWF produced what is called the EC nature run (NR) that was a 13–month simulation of nature using a 2005 version of their numerical weather prediction (NWP) model. That NR data set was used by the GMAO until 2015, when the GMAO produced its own NR at higher temporal and horizontal resolution, incorporating more atmospheric fields.

This report focuses on the general design and validation of the GMAO OSSE rather than on applications to specific new instruments or assimilation techniques. Specifically, it describes details of the third version of this framework. In the following, the term GOWASP–3 denotes the GMAO OSSE for Weather Analysis Software Package version 3, G5DAS denotes version 5.14 of the

NCEP/GMAO Grid–point Statistical Interpolation (3–D VAR) system, and G5NR–1 is the GMAO GEOS–5 Nature Run version 1 run on the cubed sphere at resolution C1440L72 (approximately 7 km horizontal grid spacing).

Chapter 2

Simulation of Nature

An OSSE requires three components in addition to a DAS: (1) a simulation of nature, (2) a simulation of observations of that nature, and (3) a simulation of observation (instrument plus representativeness) error. Although the third is rarely mentioned by others writing about OSSEs, the theory of data assimilation clearly states that errors in the DAS products depend on errors in the DAS observations and data assimilation model. Additional diagnostic software is also required to aid the validation and error tuning processes.

The simulation of nature, termed a nature run (NR), is generally an integration of a high–resolution climate simulation model or a long–term run of a NWP model. This provides atmospheric fields that evolve according to a formulation of atmospheric dynamics and physics. These fields can be provided at time intervals as short as the model time–step.

The first widely used NR was produced by ECMWF in 2006. It had a resolution of T511L91, which corresponds to a roughly 40 km horizontal grid. The fields were provided every 3 hours for the simulated period 1 May 2005 thru 31 May 2006. This period was chosen in the hopes that the simulation would produce a large number of tropical Atlantic cyclones since a record–breaking number of such cyclones occurred during the corresponding real time period. Fields of aerosols or chemical constituents (aside from ozone) were not included in this NR.

The GMAOs nature run (G5NR-1) was produced in 2014. It has a horizontal grid spacing of approximately 7 km, with a vertical grid of 72 levels in addition to the surface. It covers the period 15 May 2005 through 16 June 2007. This replicates the ECMWF period but also includes an additional year so that some interannual variability can be examined. Besides being at higher horizontal resolution and covering a longer period, the G5NR–1 also has more frequent output data (half hourly rather than 3 hourly) and includes additional output fields such as concentrations for 16 aerosol types. This NR is described in Putman et al. (2014).

Besides its general formulation for the earth's atmosphere and its initial atmospheric and land conditions, which are derived for the start date from a real analysis, the only constraints on the G5NR–1 weather with regard to what occurred at the corresponding time in nature are the sea surface temperature and sea ice extent that are prescribed from real observations. Otherwise, due to the chaotic nature of atmospheric dynamics and physics, there is no correspondence between the NR simulated and real–time weather except in terms of their climatologies; e.g., that the extra–tropics is typically colder and more windy in the winter than in summer. For this reason, the NR cannot be validated by simply comparing simulated and real weather on a given date. Instead, all validation metrics must be based on corresponding statistics produced from separate NR and nature data sets. Several appropriate validation statistics for the G5NR–1 are presented in Gelaro et al. (2015).

The G5NR–1 fields are defined on constant η–surfaces; i.e., on constant p surfaces for $p < 150$ hPa, on a constant $\sigma = p/p_s$ surface for the lowest level, and on a hybrid of the two coordinates at levels between. Specifically, the p values at interfaces between the data levels are defined by

$$p_{k+\frac{1}{2}} = a_{k+\frac{1}{2}} + b_{k+\frac{1}{2}} p_s \ , \qquad (2.0.1)$$

where $k = 0, \ldots, 72$ and a, b are provided values. Subscripts that are an integer plus $\frac{1}{2}$ refer to values defined at layer interfaces and integral values refer to layer–mean or mid–layer values. Larger values of k refer to locations lower in the atmosphere (associated with greater values of p). The values $p_{\frac{1}{2}}$ and $p_{72+\frac{1}{2}}$ refer to the model top and surface, respectively.

We define the p at $k = 1, \ldots, 72$ data levels (i.e., between interfaces) using the formulation consistent with its assignment in some components of the NR model, as described in the following equation:

$$p_k = \left[\frac{p^{\kappa}_{k+\frac{1}{2}} - p^{\kappa}_{k-\frac{1}{2}}}{\kappa \left[\ln(p_{k+\frac{1}{2}}) - \ln(p_{k-\frac{1}{2}}) \right]} \right]^{\frac{1}{\kappa}} \ , \qquad (2.0.2)$$

where $\kappa = R/C_p$, $R = 287.06$ J kg^{-1} K^{-1} is the gas constant for dry air, and $C_p = 1005.7$ J kg^{-1} K^{-1}

is the specific heat capacity of dry air at constant pressure.. This yields a formulation of hydrostatic geopotential height that is consistent with its calculation in the G5NR–1:

$$z_{k-\frac{1}{2}} = z_{k+\frac{1}{2}} + \frac{1}{g} R T_k \left(1 + \epsilon q_k\right) \left[\ln(p_{k+\frac{1}{2}}) - \ln(p_{k-\frac{1}{2}})\right] \ , \tag{2.0.3}$$

$$z_k = z_{k+\frac{1}{2}} + \frac{1}{g} R T_k \left(1 + \epsilon q_k\right) \left[\ln(p_{k+\frac{1}{2}}) - \ln(p_k)\right] \ , \tag{2.0.4}$$

where $g = 9.80665$ m s^{-2} is the acceleration of gravity and $\epsilon = 0.622$ is the ratio of the molecular weight of water vapor to that of dry air.

Although the G5NR–1 is run using a cubed–sphere grid, it also produces output on a Cartesian (0.0625 degree) grid, with 5760 longitudes and 2881 latitudes (equally spaced and including the poles). Horizontal interpolations from model to output grids are bi-linear on eta surfaces. Fields are available every 30 minutes. Additional data sets contain fields with lower (0.5 degree) horizontal and temporal (hourly) resolutions, but these are not used to simulate observations. (These reduced–resolution data sets are produced using a mass conserving mapping; i.e., as integrals over overlaying high–resolution grid boxes.) When creating the observations, reading the high–resolution NR fields is the most computationally expensive component of the GOWASP–3 software. Several aspects of the GOWASP–3 design are specifically intended to mitigate this expense.

Although the G5NR–1 has been extensively validated, users may need to perform further validation using additional metrics specific to their needs. Realistic simulation of a specific observing system does not require a uniform degree of realism across all facets of the modeled atmosphere. For example, it is not so important that clouds be simulated well if an observation is unaffected by them. In contrast, observations that either are intended to measure clouds or are strongly affected by them will require a high degree of realism in the simulation of the clouds. For some observing systems, the G5NR–1 data sets and G5DAS may be inadequate.

In general, a higher spatial resolution for a NR implies the need for a higher frequency of field output. Otherwise, temporal interpolation error would swamp spatial interpolation error, reducing the total effective resolution of the NR data set. (An exception may occur for some observation types whose simulations do not require temporal interpolation.) The temporal frequency that should accompany a particular spatial resolution can be determined by estimating when magnitudes of spatial and temporal interpolation errors become similar, as described in Privé and Errico (2016). Costs of data

I/O and storage can quickly become prohibitive as higher spatial resolutions are requested. For the G5NR–1, an output frequency of 15 minutes would have been desirable if it had been feasible.

Chapter 3

Simulation of Observations

The GOWASP–3 observation classes include most of those assimilated operationally by the GMAO sometime during 2015. These include

- Conventional observations, including from rawindsondes, dropsondes, atmospheric profilers, aircraft, ships, buoys, and ocean–surface scatterometers;
- Infrared radiances from HIRS4, AIRS, IASI, and CRIS;
- Microwave radiances from AMSUA, MHS, ATMS, SSMIS, and GMI;
- Cloud–track or moisture–gradient track SATWINDS (also known as atmospheric motion vectors or AMVs) from geostationary or polar–orbiting imagers;
- Bending angles from GPSRO occultations.

Excluded are some observation classes that had small estimated impacts on real analysis and forecasts. These classes include GOES radiances, OMI ozone, and MLS temperatures. Tropical cyclone bogus observations (TCVITALS) are also excluded. Except for the last, these neglected observations have been shown to have small impacts operationally, although this judgment may reflect not so much their intrinsic utility as how their impacts were measured (e.g., based on global–mean metrics) or how they were used by the G5DAS.

The simulated observations are assigned observing times within the 2–year period of the NR, even if such assignments are outside a period for which the real observation were available. Essentially, a temporal offset is specified for the simulations, so that the temporal distributions of the simulated observations are realistic (e.g., with respect to satellite–viewing swaths for subsequent 6–hour assimilation periods). For GOWASP–3 simulated observations the OSSE periods 1 June 2005 — 31 May 2006 and 1 June 2006 – 31 May 2007 both correspond to the real data period 1 June 2015 – 31 May 2016; e.g simulated observations for both 6 June 2005 and 6 June 2006 are based on the real observation locations for 6 June 2015. Note that this results in an unrealistic discontinuity of satellite observation paths at the 1 June 2006 simulation time that should, however, not affect any important OSSE statistics.

Unless otherwise noted for particular observation types, horizontal interpolations of the G5NR–1 gridded fields to observation locations are bi-linear on (hybrid) η–surfaces and are linear in time between 0.5–hour, 0.0625–degree, archived data sets. Vertical interpolations are generally linear in $\log(p)$, and those for q are linear in terms of corresponding relative humidity. For fields that are time averaged (e.g., accumulated precipitation) rather than instantaneous, no time interpolation is performed; instead the time–averaged NR field defined at the time closest to the observation time is used.

There are several critical properties of the simulated observations whose realism can strongly affect the DAS results. For each observation class, these include

- Numbers of observations;
- Spatial and temporal distributions of observations;
- Relationships between observations and the fields to be determined;
- Characteristics of instrument and representativeness errors, including biases, variances, correlations, and their relationships to local synoptic conditions.

The realism of these properties specifically concerns the sets of QC–accepted observations since only these observations can affect DAS results.

The simulated observations are written to a file in BUFR format that is designed to look like the original file that contained the real observations of the same class. We only write those BUFR variables that are actually read by the version of GSI on which our OSSE software development was based. That version successfully reads and interprets all of the observational data in the simulation files. Some variables that are in the original BUFR files but are not presently used by the G5DAS therefore may be missing from the OSSE BUFR files. (One important example is discussed next). Some other variables that are not presently used by the G5DAS may be included in the file, but without knowing how they would otherwise be used, their simulation may not be testable yet, and minimal effort has been expended on their validation or creation. Only the simulated variables actually used have been validated to any degree.

When the GMAO OSSE development was begun, the standard deviation of the observational error utilized by the G5DAS for each observation was determined from tabulated values appearing in files independent of the BUFR data. In later versions of the G5DAS, for some observation types these are instead read from the BUFR files. For most of these types, the values in the BUFR files are actually the same as those obtained from the tables. Since the latest version of the G5DAS optionally allows for the values to be read from the error–table files as before, we have kept to the original procedure. Users of the G5DAS with this version of our BUFR files must therefore remember to activate this option when running the G5DAS. In later versions of our software, we will include the error standard deviations in the created BUFR files. This will first require simulating the algorithm used to define the varying error statistics.

For most applications, a validated and meaningful OSSE requires that simulated instrument plus representativeness errors be added to the simulated observations. This is accomplished by first creating observations without such explicitly added errors. If ingested by the G5DAS, these observations still introduce some implicit representativeness error, however, either because they are produced on the G5NR–1 grid rather than the G5DAS grid or because the radiances have some cloud, precipitation, or surface contamination, as discussed in subsequent sections. The observations with no added error are archived for users who prefer to add their own errors.

3.1 Simulation of Radiances

In order to obtain realistic spatial and temporal distributions of radiance observations for observation classes that already exist, we begin by using available BUFR files for corresponding real data. For non–existing data classes that have orbit and scan characteristics sufficiently similar to those for an existing class, the required distributions can be modified from those provided by an existing BUFR file. Otherwise, an orbit and scan model must be introduced. In this document, we will be concerned only with existing data classes.

3.1.1 Partial radiance thinning

The G5DAS thins most radiance observation classes having high spatial densities in order to limit computational demand and reduce the presence of observations having large spatially–correlated errors. This is particularly true for radiances. The G5DAS thins based on several criteria, including spatial location and time as well as estimated data quality. Some GOWASP–3 radiance observations are also thinned when created so as to reduce computational demand. The GOWASP–3 thinning, however, is less severe than performed by the G5DAS. The G5DAS is therefore still able to choose a best observation among what it is offered, obviating the need for the GOWASP–3 thinning to precisely mimic what is done in the G5DAS.

Thinning is performed separately for each data class and observing platform. For each class and platform, the globe is divided into trapezoids of approximately equal area. The G5DAS uses a similar approach. Since our intention is to generally use smaller trapezoids than used in the G5DAS, it is unnecessary to use an identical algorithm. The desired sizes can be specified separately for each observation class.

From among all the observation locations that fall within a single thinning box, a single one is chosen for retention for each combination of satellite instrument and observing platform. Information from multiple platforms or instruments may therefore be retained in each box if more than one satellite or instrument has views within the same box. Observation values will be computed for a specified set of channels for that instrument on that satellite platform at that location.

The GOWASP–3 also uses several criteria to choose a likely best observation from among those that fall within a box. In this context, however, since no simulated observation values yet exist, the choice cannot be based on the same information as used within the G5DAS. Instead, the thinning is based on some specified NR fields at the observation locations, specifically, on fields that especially may affect the quality of subsequently produced data. For IR radiances, this includes fractions of high, medium, and low level clouds. For MW radiances, this includes surface properties that may

Table 3.1.1: Values of parameters used by the algorithm employed by GOWASP-3 to thin radiance observations.

Instrument	HIRS	AIRS	IASI	CRIS	AMSUA	MHS	GMI	ATMS	SSMI
Box Size (km)	60	60	60	60	60	60	60	none	none
% retained	37	9	21	70	67	10	1	100	100
Weights:									
Fraction Land	0.1	0.1	0.1	0.1	0.8	0.8	0.8		
Fraction Land Ice	0.1	0.1	0.1	0.1	0.8	0.8	0.8		
Fraction Sea Ice	0.1	0.1	0.1	0.1	0.8	0.8	0.8		
Preip Rate Anvil					0.5	0.5	0.5		
Preip Rate Convec					0.3	0.3	0.3		
Preip Rate LScale					0.2	0.2	0.2		
Cloud Frac High	0.7	0.7	0.7	0.7					
Cloud Frac Mid	0.5	0.5	0.5	0.5					
Cloud Frac Low	0.3	0.3	0.3	0.3					

affect emissivity or precipitation that may affect scattering.

For each observation location, a thinning penalty value is determined based on a sum of weighted field values. For IR observations, for example, this is a linear combination of the cloud fractions for three levels of clouds, with the fractions of higher–level clouds weighted more heavily since those clouds can have a worse effect on observation retention by the G5DAS QC. For MW observations, locations over water are favored since the emissivity model is generally most accurate over water. The location assigned the smallest penalty within each penalty box is chosen. If two locations are assigned the same penalty value, the one whose time is closest to the central (synoptic) analysis time is chosen. There is no choosing based on proximity to the center of a box since there is no set relationship between centers of the GOWASP–3 and G5DAS thinning boxes.

For GOWASP–3, the sizes of the thinning boxes, the fields and weights used for thinning, and the fractions of retained observations are presented in Table 3.1.1. Box sizes are specified in terms of a length of one of the sides. Note that no thinning is performed for ATMS and SSMIS since, for these classes, the G5DAS will perform an averaging of observations as a means of noise reduction in addition to a thinning. This presumably requires the densities of real and simulated observations to be the same for the algorithm to function similarly.

3.1.2 Consideration of effects of clouds on IR radiances

Transmittance of IR radiation through the atmosphere is strongly affected by clouds. When the GMAO OSSE development effort began, the modeling of scattering, absorption, and transmittance of radiation by clouds was still in its infancy, especially regarding the development of computational algorithms fast enough to produce the hundreds of millions of observations required for the OSSE in a reasonable time. Even though better algorithms are now available, their usefulness in the OSSE context remains problematic since the cloud distributions in the NR are not entirely realistic and the ranges (in contrast to single prescribed values) of cloud parameters characterizing hydro–meteor size and shape distributions are unavailable.

For most radiance observation classes, the G5DAS currently assimilates only what it believes to be radiances unaffected by clouds. For any cloud–affected radiances not eliminated by data thinning or quality control, presumably either the clouds are optically thin or are far enough below the atmospheric layers from where the observed radiation is effectively emitted. Differences between the accepted observed radiances and corresponding, CRTM-determined cloud–free radiances that are due to cloud effects yield a portion of representativeness error (i.e., specifically, due to an invalid cloud–free assumption in the observation operator). Thus, even if an accurate radiative transfer model is used to simulate the effects of clouds on radiance observations from the nature run, most of that extra effort will simply be discarded as the G5DAS detects large differences between the observation value provided by the OSSE and its cloud–free calculation determined by the CRTM applied to the G5DAS background. Those observations only weakly affected by clouds will pass the quality checks, affecting the distribution of errors in the observations as considered by the G5DAS.

The effects of a thick cloud can easily be modeled, since in this case it may be considered a black body. Thus, a thick elevated cloud appears the same as an elevated surface as far as IR is concerned. IR channels that normally peak lower in the atmosphere will therefore appear much colder. Channels that normally peak much above the cloud top level, however, will remain unaffected by the elevated surface. In the current OSSE observation simulation software, the effects of clouds on IR radiation therefore are introduced by simply setting the cloud–top temperatures to the atmospheric temperatures at their elevations, and informing the radiative transfer model that the effective radiative surface is at that level. Thus, the gross effects of clouds are modeled without using a radiative transfer model that explicitly considers clouds. The use of this gross modeling is primarily to obtain a realistic count of cloud–free observations as a function of radiance channel and consistent with the distribution of clouds in the nature run. Effects of thin clouds on the radiances are handled by appropriately tuning the simulation of added representativeness plus instrument errors.

At this time, the distributions of cloud–related fields provided in the nature run (specifically profiles of liquid and ice water contents and cloud fractions) have not been sufficiently validated regarding their effects on IR radiances, especially in the presence of only thin clouds. Although examination of time and zonal mean fields of some measures of cloud content in the nature run is useful, their agreement with nature does not ensure that realistic cloud effects will be obtained when they are considered by a radiative transfer model that includes clouds, even if the transfer model is a good one. While we believe that the cloud–related fields in the nature run are much more realistic than in former nature runs, we expect that some important aspects may be unrealistic, especially regarding the prevalence of high thin clouds.

In order to expedite the development work in light of the above issues, the GOWASP–3 software includes a simple tunable scheme to incorporate the effects of clouds into the IR simulated observations. This scheme uses a random function to determine whether radiances are cloud affected, where the probability that they are so affected is a function of the fractional cloud cover at 3 levels, as provided by the nature run data set.

The cloud presence algorithm is as follows:

- Denote $j = 1, 2, 3$ for high, medium and low clouds, respectively.
- For each j determine a probability P_j that a cloud of that type is within the instrument field of view and is significantly affecting some observed radiance channels. P_j is simply defined as a convenient piece–wise linear function of the corresponding cloud fraction f_j provided by the NR:

$$P_j = \begin{cases} 0 & \text{if } f_j \le a_j, \\ (f_j - a_j)/(b_j - a_j) & \text{if } a_j < f_j < b_j, \\ 1 & \text{if } f_j \ge b_j, \end{cases} \tag{3.1.1}$$

  where $0 \le a_j < b_j \le 1$ are tuned parameters that vary with j and instrument type.
- For each j, draw a random number $0 \le r_j \le 1$ from a uniform distribution on that interval.
- A significant cloud is deemed present if $r_j < P_j$.

- Find the smallest value of j for which a significant cloud has been deemed present at this location. If found, then the effective radiative surface for this observation location is assigned to be the cloud top pressure $p_c = \sigma_j p_s$, where p_s is the true surface pressure at that location and σ_j is another tuning parameter.

Three levels of clouds are considered; low, medium, and high (height) clouds. In the nature run data set, these correspond to pressure ranges $p > 0.8p_s$, $0.8p_s \geq p \geq 0.45p_s$ and $0.45p_s > p$, respectively. The presence of each type of cloud is determined by the algorithm previously described. This particular form for the probability functions was chosen because it is both simple and tunable. The three tunable parameters, a, b, and σ are specified for each instrument type: an instrument with a smaller viewing footprint has a greater chance of encountering a hole in the clouds than one with a larger footprint. The cloud top pressure is specified as a fraction of surface pressure ($\sigma = p/p_s$) so that low clouds can be present for $p < 500$ hPa even over high topography, such as in Tibet.

The probability function used by this algorithm is piece–wise linear. If the cloud fraction for a particular level is less than or equal to parameter a, then the field of view is defined as free of clouds at that level. If it is greater than or equal to parameter b, then it is definitely cloud contaminated. If neither of these conditions holds, then the probability P of a cloud being present is between 0 and 1. In this case, whether a contaminating cloud is declared present is determined by drawing a random number $0 < x < 1$ from a uniform probability distribution (using a standard FORTRAN call to a random number generator) and comparing it with P: If $x < P$, then such a cloud is present; otherwise not. The statistics of this procedure are such that, e.g., 20% of all the cases when $P = 0.20$ are expected to be declared as cloud affected.

What matters is where the cloud tops are, so first this procedure is performed for high clouds, then for middle, and last for low. If a cloud is declared, then σ is specified as given in the table for that level cloud and clouds at lower levels are not considered. If no radiatively significant clouds are declared present, $\sigma = 1$ is specified, indicating that the effective radiative surface is the true surface. Since the presence of lower clouds is conditioned on clouds above being absent, the probabilities for middle and low clouds should actually be considered as probabilities that are conditioned on there being no significant clouds above at that location. This conditional aspect should be considered when specifying the parameters for the probability functions.

In this procedure there are 9 parameters that can be adjusted for each satellite instrument type.

Table 3.1.2: Values of a_j , b_j and σ_j used to determine cloud "contamination" of otherwise clear-sky radiances for the indicated IR instruments

Instrument	HIRS	AIRS	IASI	CRIS
High Levels				
Field Used	CLDHGH	CLDHGH	CLDHGH	CLDHGH
a_1	0.0	0.0	0.0	0.0
b_1	0.1	0.1	0.3	0.1
σ_1	0.35	0.35	0.35	0.35
Mid Levels				
Field Used	CLDMID	CLDMID	CLDMID	CLDMID
a_2	0.0	0.0	0.0	0.0
b_2	0.02	0.4	0.8	0.04
σ_2	0.65	0.65	0.65	0.65
Low Levels				
Field Used	CLDLOW	CLDLOW	CLDLOW	CLDLOW
a_3	0.0	0.0	0.0	0.0
b_3	0.004	0.004	0.8	0.004
σ_3	0.85	0.85	0.85	0.85

We have tried varying them to see what impact they have on the G5DAS QC, and the variations appear to have the expected effects. As we demand more from the validations, however, it may be necessary to use different functional parameters in different latitude bands. Such additions to the software capability, if needed, will only be introduced in later versions of the simulation software. Parameter values used to create GOWASP–3 simulated observations are presented in Table 3.1.2.

This procedure for creating cloud–affected radiances that will be discarded by G5DAS QC is validated by comparing monthly counts of observations for real and OSSE data sets as functions of instrument and channel number. (An example of such a validation that was produced using an earlier version of GOWASP applied to the ECMWF NR is presented in Errico et al. (2013).) The tuning parameter values in Table 3.1.2 were determined by iterating some G5DAS OSSEs until corresponding values for most instruments and channels were within approximately 10% of each other. It should be noted that the parameters for IASI differ greatly from those for other types principally because IASI includes use of a cloud–affected quality flag in the G5DAS thinning algorithm, effectively applying the cloud screening twice (a speculation by the authors).

Acceptability of the general procedure and of the particular values listed in Table 3.1.2 also depends on the realism of other characteristics of the locations of QC–accepted observations. For example,

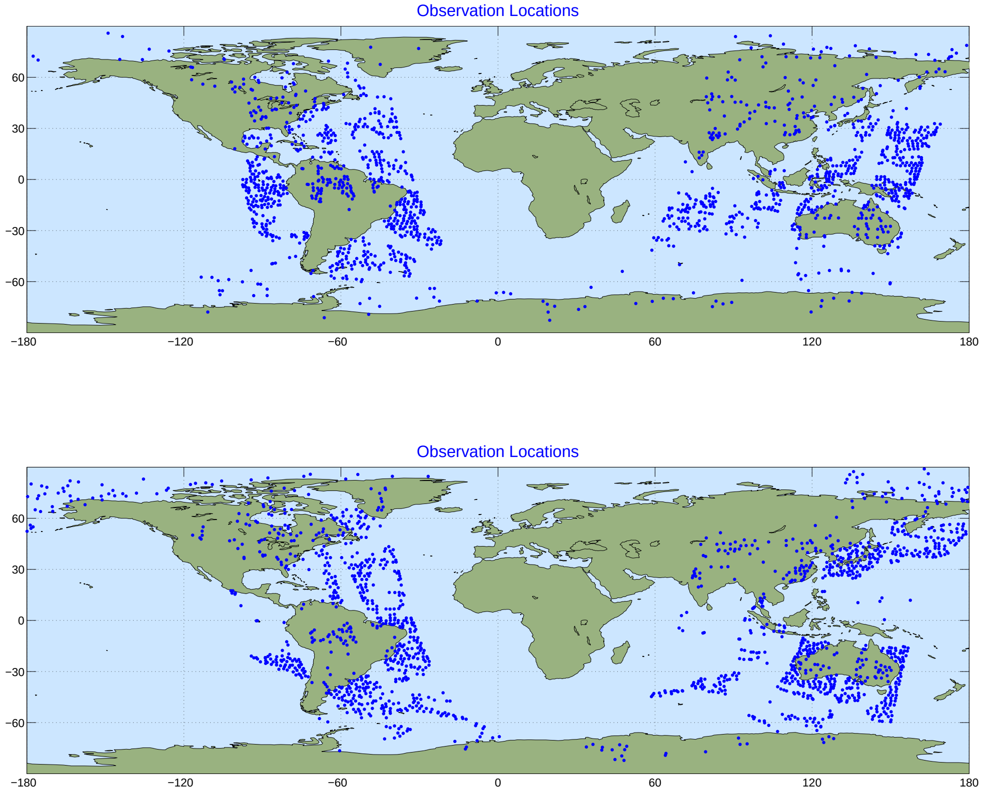


Figure 3.1: Locations of AIRS channel 256 observations accepted by G5DAS quality control for the 6–hour assimilation window centered on 20 July 1800 UTC for real (top) and simulated (bottom) observations. Real data are from 2015 and OSSE data are derived from them but reassigned to 2006. Channel 256 corresponds to channel 100 in the 281 NCEP subset.

not only the mean counts but also the appearance of spatial gaps in the observing swaths should be similar. Since the locations of clouds within any specific analysis period may differ in the NR and real data sets, however, it is not the precise locations of these gaps but instead their qualitative nature that is relevant (e.g., the numbers, shapes, and sizes of typical gaps). No attempt has been made to quantify these characteristics for the GOWASP–3 tuning. Such validation has instead been limited to simple visual inspection of plots of QC–accepted locations.

An example of the spatial distributions of QC–accepted observations for one instrument, channel, and pair of 6–hour analysis periods for real and corresponding OSSE data sets is presented in Fig. 3.1. The data are for channel 256 of AIRS on the NASA AQUA satellite, which has a weighting function that peaks near the earth's surface unless clouds are present. It is therefore a channel radiance that is often cloud–affected. The analysis periods shown are centered on 20 July 1800 UTC for 2015 and 2006, respectively, with the former being the observing period used to determine locations for the latter. Locations of the viewing swaths in the two figures are therefore guaranteed to be very similar, with differences only due to differing data selection by both thinning and QC algorithms. Counts of the two sets of QC–accepted observations for this period are similar: 1426 for the real and 1518 for the OSSE. Note that qualitatively, the general nature of the gaps in the swaths are similar, suggesting that the procedure that introduces clouds to affect QC is working as designed.

The results produced by the GOWASP radiance cloud–contamination algorithm depend on the table of parameters for both thinning and adding clouds and also on the spatial distribution of cloud fractions. In particular, they depend on the actions of several tandem stochastic algorithms and on the joint distributions of several (e.g., cloud–fraction) fields, rendering the results difficult to anticipate. To aid the tuning of these algorithms it therefore is useful to examine at least some important NR statistics missing from the G5NR validation.

One example is the distribution of cloud fractions (f) used in the radiance thinning and cloud–effect algorithms. In Fig. 3.2 the fractions of global area containing binned ranges of high, mid, and low cloud fractions averaged 4 times daily over July 2006 of the G5NR are displayed. Each bin includes areas of all grid boxes for which $0.1(n-1) \leq f < 0.1n$ with bins $n = 1, 2, \ldots, 10$. The value $f = 1$ is also included in bin 10, although a negligible number of grid points have that precise value. In contrast, most of the grid boxes falling in bin 1 have $f = 0$. Note that with most bins populated by only portions of the globe, the cloud–effect algorithm is expected to be fairly insensitive to many ranges of changes of a_i, b_i; e.g., some parameter changes will only cause decisions to be altered at a few locations.

3.1.3 Consideration of effects of clouds and precipitation on MW radiances

Some MW channels are also affected by clouds or precipitation. Although the CRTM utilized by both GOWASP–3 and G5DAS can incorporate such effects, further work is required to use it in a way that produces realistic errors due to uncertainties in the way in which these effects are computed. For example, the CRTM assumes specific cloud or rain drop size distributions and

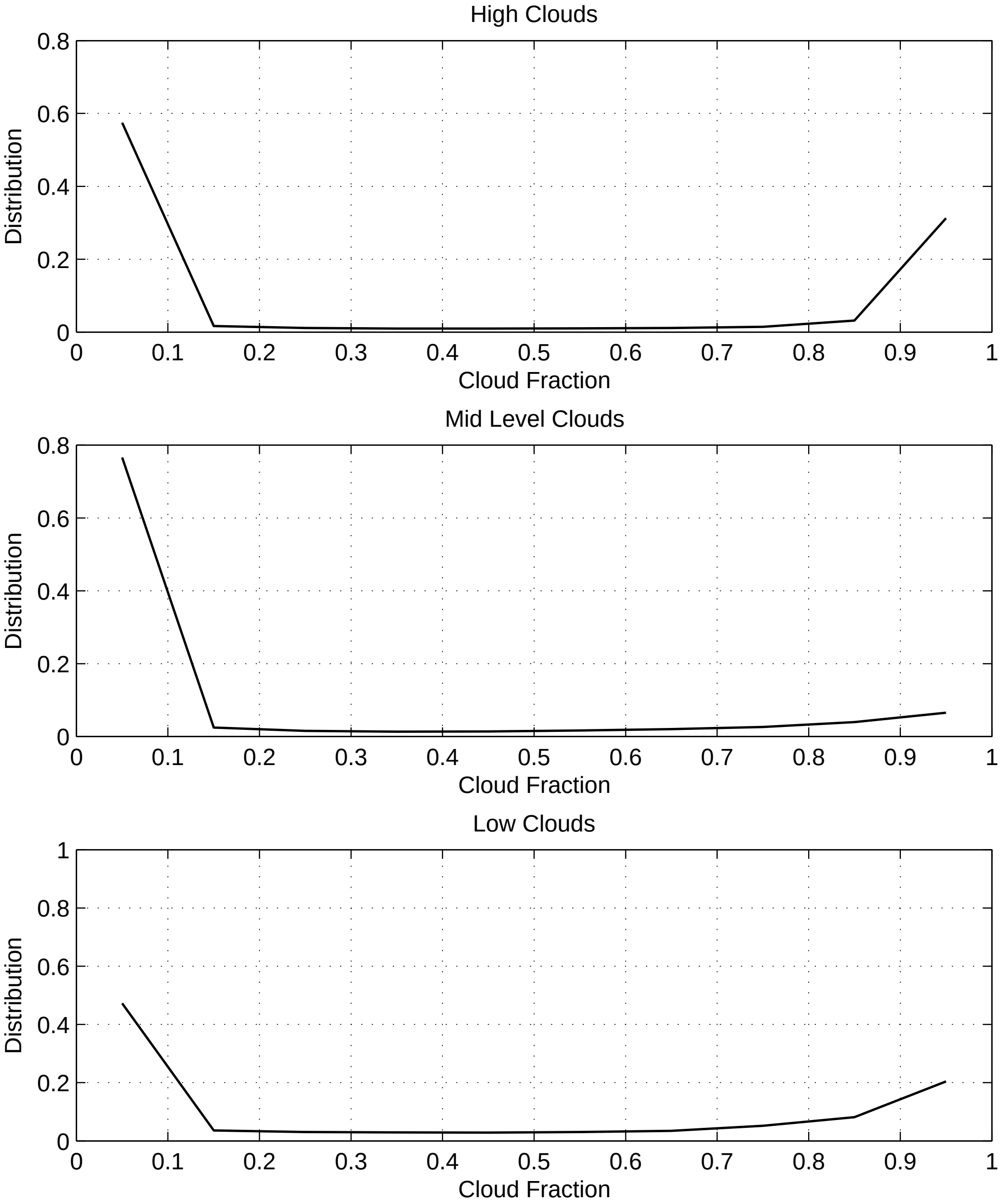


Figure 3.2: Binned distributions of values of high– mid– and low–level cloud fractions in the G5NR for every 6 hours during July 2006. The 10 bins are equally spaced, with the abscissa indicating values at the center of each bin. Cloud fraction values 0 and 1 are contained in the first and last bin, respectively. An area weighting is used in the calculation so that the distribution values are fractions of the earth's surface containing the cloud fraction values falling within each particular bin.

Table 3.1.3: Values of a_j , b_j and σ_j used to determine precipitation "contamination" of otherwise clear-sky radiances for the indicated MW instruments

Instrument	AMSUA	MHS	GMI	ATMS	SSMIS
High Levels					
Field Used	PRECANV	PRECANV	PRECANV	PRECANV	PRECANV
a_1	0.0010	0.0002	0.0010	0.0010	0.0010
b_1	0.0010	0.0002	0.0010	0.0010	0.0010
σ_1	0.30	0.30	0.30	0.30	0.30
Mid Levels					
Field Used	PRECCON	PRECCON	PRECCON	PRECCON	PRECCON
a_2	0.0006	0.0002	0.0006	0.0006	0.0006
b_2	0.0006	0.0002	0.0006	0.0006	0.0006
σ_2	0.60	0.60	0.60	0.60	0.60
Low Levels					
Field Used	PRECLSC	PRECLSC	PRECLSC	PRECLSC	PRECLSC
a_3	0.0003	0.0002	0.0003	0.0003	0.0003
b_3	0.0006	0.0002	0.0003	0.0003	0.0003
σ_3	0.80	0.80	0.80	0.80	0.80

snow or ice shapes that, although perhaps typical or average in some sense, may belie the range of uncertainty that may exist in particular instances. For those MW channels possibly affected by clouds or precipitation that, however, are considered within the G5DAS as being unaffected by such things, incorporation of scattering by cloud or precipitation within the GOWASP–3 algorithm will yield a form of representativeness error rather than a useful signal.

Although the GOWASP–3 software allows clouds and precipitation to influence MW radiances computed within the CRTM, the GOWASP–3 data sets have been produced without using this option. Instead, to introduce a rudimentary contamination due to precipitation, a method analogous to that used to contaminate IR radiances by cloud effects in the GOWASP–3 data sets is used. This follows the same algorithm, except precipitation at the surface due to stratiform, convective, and anvil processes are used in place of fractions of low, medium, and high cloud cover, respectively. Values of a, b, and σ naturally differ from their IR counterparts. These are provided in Table 3.1.3. Although this technique seems to add some realism in the G5DAS QC, it is insufficient at creating realistic observation counts for low–peaking channels. This suggests that missing from the GOWASP–3 is sufficient contamination of surface emissivity that suffers in the OSSE because the G5DAS and GOWASP3 use identical algorithms for its determination.

3.1.4 Consideration of surface emissivity uncertainty

In GOWASP–3 we assume that observations of surface–affected microwave channels should exhibit large errors over land and ice due to both imperfectly modeled emissivity and imperfectly known and highly variable surface conditions. Since the same radiative transfer code is being used to model surface emissivity in both simulation and assimilation contexts, however, the first source of error is absent. The second source of error may also be diminished due to the ways in which surface properties are distinguished. Without these sources of error, the realism of errors implicit in the simulation of MW observations may be severely compromised because the quality of such observations is otherwise generally high. Impacts of these observations on DAS background fields also tend to be high since short–term forecasts are especially sensitive to temperature analysis near 700 hPa as revealed by forecast sensitivity calculations made with adjoint models. Yet, without sufficient guidance concerning the nature of the errors due to surface emissivity modeling, it is difficult to simulate them.

Although apparently deficient, there are some sources of emissivity uncertainty that affect some GOWASP–3 radiances. Over water, the emissivity values computed by the CRTM in the G5DAS and GOWASP–3 are almost identical when the same atmospheric profiles are introduced, the only slight exception concerning how the surface winds are precisely computed. Over land, however, the land types in the two contexts are not determined from the same data sets at the same resolutions. Also, the treatment of fractional coverage by land, water, ice, and snow is handled differently. These differences in surface treatment are intentional since there should be large error due to uncertainty in the emissivity. The same differences are applied when computing IR emissivities, although resulting differences in computed emissivity will be much less than for MW. We hope, however, that a better treatment of emissivity uncertainty for MW channels will be introduced in a later version of GOWASP.

3.1.5 Consideration of biases in radiance errors

There are several sources of bias in real radiance observations from satellites. Some of these concern the instruments; e.g. an instrument antenna detecting interference from its satellite platform. This is inferred especially from an asymmetric scan angle bias, since depending on which way the antenna is pointed, it sees a different portion of the platform. Biases can also result from the forward model, including the surface emissivity model. Biases may be difficult to determine if, although systematic, they depend on the synoptic state being observed. Some real radiance biases have been estimated to be rather large. They must therefore be removed prior to a DAS attempting to extract usable information from the observed radiances.

For GOWASP–3, the sources for creating biases mentioned above, except for surface emissivity over land or ice, are absent. There is no simulated satellite platform, and the forward model (the CRTM) is at most a different version of the same algorithm and software as used in the G5DAS. Since our hope and expectation is that observations over land or ice strongly affected by surface emissivity will be discarded by the quality control, as they are in real assimilations, most of that bias should be removed. Thus, there is little substantial bias introduced in the simulations and thus there is not much need to use radiance bias correction in the G5DAS when assimilating these observations. Nonetheless we have run the G5DAS with its bias correction algorithm in both OSSE and real data contexts. We note that this requires substantial spin–up of bias correction coefficients, since the e–folding time for relaxing bias correction coefficients in our version of the G5DAS is 1 month. The spin–up process is accelerated by starting the OSSE with 0–valued coefficients. The spun–up bias coefficients are generally smaller than those for corresponding real observations.

Obviously, one source of likely error in the OSSE that is unrealistically absent is small yet significant error in the radiance bias correction algorithm. This can be simulated by adding some sort of bias to the observations that may have any characteristics an experimenter cares to incorporate. For example, biases that are derived from the DAS bias correction model can be added easily. Presumably, however, these would then also be effectively removed by the bias correction. There therefore seems little point in adding such biases unless an experimenter has a specific test of the bias correction in mind. In that case, the biases that are added should be carefully designed to test the specific hypothesis proposed; e.g., effects of biases not described by the DAS algorithm. So biases can be added, but otherwise there seems to be no need to do so for general experiments.

The G5DAS attempts to estimate observation biases by examining observation innovations (also known as observation increments). Essentially, its algorithm assumes that innovation biases are strongly dominated by observation biases. Since the GOWASP–3 observations have no added bias, any biases determined by the G5DAS applied to those observations will instead be dominated by biases of background error. In that case, the G5DAS bias correction will remove biases it should actually keep as corrections. Thus, running the G5DAS OSSE with bias correction does the opposite of what is desired and acts as a source of DAS error. Presumably, some of the bias removed when assimilating real data also is due to background error bias and therefore also results in an erroneous correction. For the real observations, however, most of the bias correction is expected to be appropriate.

3.2 Rawindsondes and Dropsondes

In GOWASP–3, the locations and times of rawindsonde or dropsonde launches are determined by those of real observations at a corresponding real time. The pressure level at which an ascending sonde stops reporting or a descending one begins reporting are also determined by corresponding real observations. All other aspects of the observation are determined by the NR fields. In particular the sondes drift horizontally with the NR winds while ascending or descending.

The sondes are assumed to ascend or descend at a rate of 5 ms^{-1}. The drift algorithm considers 30 s intervals, determining the NR wind speed and direction at the end of each interval and advecting the sonde during the next interval using that wind. The values of T, q, u, v, p, z determined at the end of each interval are considered as the raw sounding. From this sounding, observation reports at mandatory and significant levels are determined. Values for T and q at the surface are set to the 2m values provided by the G5NR–1 and those for u, v are set to the provided 10m values.

The sonde BUFR reports are taken from the raw soundings similar to the way in which real reports are determined. Values for mandatory levels are determined by spatial and temporal interpolation. Values of T, q, p, z at the start and end of the sounding are also included. Other values included are local maxima in the raw sounding as well as values that cannot be well approximated by linear interpolation from successive values. Values for T at the level closest to 105 hPa and at a level determined to be the tropopause are also included, as required by the WMO reporting standard. Since the NR has coarse vertical resolution, the GOWASP–3 sonde algorithm does not yield as many significant level values as real soundings often do. Also, it is apparent that many real BUFR reports contain more levels than the official WMO guidelines warrant, indicating that some data providers use a different algorithm to select reported data from raw soundings. The counts of GOWASP–3 radiosonde observation values are therefore 75% of real counts.

Horizontal interpolations are bi-linear on NR η–surfaces and linear in time between 0.5 hour archived data sets. Vertical interpolations are generally linear in $\log(p)$. Vertical interpolations of q are linear in terms of corresponding relative humidity. Vertical interpolation of winds between levels in a raw sounding is in terms of wind speed and direction rather than u, v components.

Quality flags in the GOWASP–3 BUFR reports on each level are set based on corresponding real reports. If a real report has missing values for some fields at some levels, the simulated reports will all have the same missing values at similar levels. This helps preserve some characteristics of the simulated sonde BUFR reports with respect to subsequent QC in the G5DAS.

The G5DAS expects that, under particular conditions, the temperature observations within the BUFR data files will actually be corresponding values of virtual temperature T_v. Those conditions are: (1) the report contains a valid moisture observation at the same location, as required to transform between T_v and T; (2) the observation is at a level $p > 300$ hPa.

3.3 Single–level Reports

In GOWASP–3, the vertical locations of all single–level conventional observations are determined from the pressure values included in the real observation reports, with the exception being observations reported at the surface. If the report for a corresponding real observation includes height as well as pressure information, simulated heights are determined hydrostatically from profiles of T, q, and p_s, horizontally interpolated to the observation location. The hydrostatic calculation is performed assuming that T is constant within each atmosphere layer of the NR fields. If no height is provided for the corresponding real observation, the simulated height is recorded as a missing value.

Station elevations of surface observations are generally recorded as the topographic height interpolated from locations in the NR data set. Exceptions are station elevations recorded as 0 in the real observation files. (Since conversions between FORTRAN type variables or data packing may change an exact zero to something slightly different, the code checks whether the nearest integer is zero.) So, for example, the station elevation of any SHIP report recorded as 0 in the NCEP .prepbufr file is preserved. The station elevations of scatterometer winds are recorded as 10m, as expected by the G5DAS. The observation pressures for SSMI surface winds are recorded as 1013 mb, as they appear in the .prepbufr file for real data.

Single–level observations that have station elevations recorded as the observation height, such as for SATWNDS or AIRCAR, are noted by comparing the observation height and station elevations read in. If the read values are identical (as rounded to the nearest integer), then their agreement is preserved in the written data set by copying the observation height determined from the nature run to the station elevation written in the observation header. This specification of the station elevation, however, results in missing values being provided in the header when originally acceptable real observations fall below the surface of the nature run, as based on their reported pressures.

3.4 Satellite Winds

Real wind reports that are obtained by tracking images of clouds or water vapor gradients (termed SATWINDS or AMVs for "atmospheric motion vectors") depend on the presence of trackable features in satellite images. Presently available nature runs, including G5NR–1, have insufficient resolution, however, to allow the locations of simulated SATWINDs to be determined by a similar feature–tracking approach. This is not simply due to the not–so–small grid spacing of the NR but also to the excessive horizontal smoothing or dissipation that most models employ to control various computational problems, so that the effective resolution is generally several times greater than the grid spacing. Fortunately, for simulating existing wind observations it is unnecessary to use the same algorithm as for real observations if the goal is specifically to produce a realistic response of the DAS to the observations. For that purpose, it is only required that the spatial and temporal distributions of observations are realistic in whatever aspects are important. As long as this realism is sufficiently achieved, extraneous details of how that realism is achieved are unimportant.

A key to the above disclaimer is that the simulated observation distributions must be sufficiently realistic. This requirement can vary as differing degrees of realism are desired. For GOWASP–3, this desire includes the aspect that real feature–tracked winds are only determined where trackable features exist; i.e. where either individual clouds or strong horizontal gradients of water vapor exist. This aspect can impact the response of a DAS because where such features exist is often where more interesting, rapidly changing, and unpredictable weather occurs.

For the above reason, the locations of GOWASP–3 SATWINDS are determined based on cloud fractions and water–vapor gradients provided by the G5NR–1. Rather than attempting to track such features that may have insufficient realism, however, the cloud fractions f at model levels and the magnitudes of horizontal gradients $w_g = |\nabla w|$ of layer–integrated water vapor w are used to define probabilities P that trackable features exist at each considered location. Then random numbers r are generated from a uniform distribution on the interval $0 < r < 1$ and compared with values of P. If $r < P$, an observation of the considered type is deemed to exist; otherwise not. (This procedure is similar to that for determining if clouds or precipitation are affecting radiance observations, as described previously in this chapter.) The set of locations considered and the functions $P(f)$ and $P(w_g)$ are tuned to yield feature–associated, spatially and temporally realistic distributions of simulated SATWIND observations that are accepted by G5DAS QC and thinning; i.e., the tuning need only simulate the distribution of real observations actually used by the DAS.

The locations considered by the GOWASP–3 algorithm are a subset of the points on the G5NR–1 0.0625 degree data set grid. Every n–th grid point in the latitudinal and longitudinal directions is considered, starting with a point that depends on sub–periods considered within the 6–hour observation data–set window. This starting point varies with time so that created observations rarely occur at the same location at different times in the 6–hour window. The skipping of points in the longitudinal direction is adjusted with latitude so that the distance between points remains approximately the same as the meridians converge at higher latitudes. Within each defined sub period, all observations are specified to occur at the same time. In GOWASP–3, $n = 4$, the sub periods are half–hourly, and the time within each such period is set to be its end time (since NR data are only available half–hourly).

For an observation to exist at any considered point, it also must be in the viewing region of some imaging instrument. These can be either from geostationary or polar–orbiting satellites. The former include GOES–E, GOES–W, Meteosat–1, Meteosat–2, and Himawari–8, whose locations are considered fixed to what they were on a date in early 2016. Each of their viewing regions include all NR grid points lying within 60–degree maximum viewing angles from the satellites' nadirs. Since the reporting frequency for these satellites vary, whether their reports are simulated for a particular sub period is determined by whether there are more than 200 real reports during the corresponding real time period. For the polar orbiters, the longitudinal spreads of real observation locations for each data subtype (G5DAS **kx** value) within each corresponding considered sub period poleward of 60°N or 60°S are first determined. Simulated observations for polar–orbiters are then considered only in those longitudinal ranges.

Vertically, GOWASP–3 SATWIND observations are only permitted to exist at model levels. Since multiple layers of clouds may exist at a single G5NR horizontal grid location, up to 5 levels of reported clouds are permitted at any viewing time and location. Maximum random overlap is assumed, meaning that in each horizontal NR grid box, vertically adjacent clouds are considered to have maximum overlap but vertically separate clouds are considered to be randomly positioned compared with any above or below. The possibility of higher clouds is considered first, with the probability of clouds below being seen depending on an assigned opacity of the higher clouds. For moisture tracked winds, the height of the observation is assigned to be that of the tropopause at that grid-point. For each location so identified, observed winds are taken from the NR fields. At this stage, no height assignment error is introduced into the observations. Parameters that define the functions $P(f)$ and $P(w_g)$ have been determined by trial and error so that the counts and spatial distributions are similar for real and simulated observations.

Several other conditions on SATWIND observation locations are imposed to make the spatial distributions realistic. For simulations of visible or IR observed clouds, these are:

- Only cloud layers with a maximum cloud fraction between 0.2 and 0.8 are considered to have possible tracking features.
- There are no cloud layers with fractions equal to 1.0 above a possible trackable layer; i.e., the trackable layer is not totally obscured by clouds above.
- Smaller probabilities of observations being present are set for simulations associated with geostationary satellites over land.
- Higher probabilities are assigned to $0.4 \leq f \leq 0.6$.
- Only locations where wind speeds are greater than 3 ms^{-1} are considered.

Observations based on simulated water vapor gradients specifically consider integrated precipitable water, as determined by

$$w = \frac{1}{g} \int_{p_1}^{p_2} q \; dp \; , \qquad (3.4.1)$$

where $p_1 = 10000$ Pa and $p_2 = 60000$ Pa, with w therefore in units of mm water per m^2 surface area. Rather than actually computing its horizontal gradients, w_g is defined as the difference between the maximum and minimum values of w in a 5×5 grid–point region centered on the considered location, without reference to the grid spacing. The applied conditions are:

- Only consider possible observations where either $w > 0.6$ or $w_g > 0.8$.
- Only consider locations where wind speeds are greater than 3 ms^{-1}.
- The probability of an observation existing is approximately proportional to w_g.

For other details, the module ***m_satwind.f90*** should be consulted.

3.5 GPS Radio Occultation

The GOWASP–3 simulated GPSRO observations are only in the form of bending angles. The viewing latitudes, longitudes, heights (impact parameters), bearings and times are determined by real observations at corresponding times. For each such impact parameter, the GOWASP–3 viewing planes are centered at the corresponding location with the corresponding bearing. Values of fields within each plane are determined by interpolation from the NR. The grid spacing for this plane is 7 km and its total span is 500 km. A single observation time is used for all impact parameters in a common report.

Unlike for radiances, for which GOWASP–3 and G5DAS use the same CRTM, different GPSRO packages are used. Specifically, GOWASP–3 uses the Radio Occultation Processing Package (ROPP) provided by the Satellite Application Facilities (SAF) of EUMETSAT. Below 10 km, its observation operator considers NR fields in a 2–d viewing plane to determine bending angle rather than assuming there are no horizontal gradients of the fields. This is unlike the calculations in the G5DAS that consider no horizontal gradients at any pressure level. The two schemes also calculate the Abel integral differently. Such differences yield an implicit form of representativeness error in the simulated observations, as discussed in the chapter on simulating observation errors.

3.6 Validation of Observations

The simulation of observations from the NR has been validated using several metrics. These concern comparing results produced in an applied OSSE with those produced when assimilating a corresponding set of real observations. We have found that using 10–day averaging periods for making these comparisons is sufficient to determine if corresponding metric values are appropriately close.

The first metrics compared quantitatively are the numbers of QC–accepted observations evaluated for different observation types as functions of radiance channel or atmospheric layer. For each radiance instrument, these counts vary with channel due to QC rejections based on suspected gross errors arising from erroneous surface emissivity, cloud, precipitation, or other effects. Observations in channels that effectively measure lower in the atmosphere tend to be rejected more than those measuring above because the latter are not as affected by clouds far below. Matching this QC effect is important because the numbers of observations assimilated should affect most other data impact metrics. With differing regions of the atmosphere providing different potential impacts, obtaining similar real and OSSE distributions with respect to channel is particularly important. For GOWASP–3, counts for simulated and real data are typically within 10% of each other, the one exception being MW instrument channels that are strongly sensitive to surface emissivity. For the latter the GOWASP–3 counts are as much as 20% higher than those for corresponding real observations.

A subjective comparison of the geographic locations of satellite wind and radiance observations is also made. Real and OSSE observations should not be necessarily co–located at corresponding times since the real and NR atmospheres will generally differ regarding where there are trackable clouds or water vapor gradients and where there are radiance-scattering clouds or precipitation. Some characteristics of the spatial distributions of retained observations, however, should be similar. Besides data counts, the spacing, general locations (e.g., satellite–observing tracks), and "gappiness" should be similar. The latter refers to gaps in the satellite tracks for channels affected by clouds: typical ranges of sizes or shapes of such gaps should be similar. Subjectively, this appears to be the case with GOWASP–3 observations with one exception being water–vapor tracked winds. For them, the G5NR–1 vertically–integrated water vapor fields are too smooth, and trackable gradients occur almost exclusively along fronts, unlike for wind produced from real atmospheric images.

The means and standard deviations of observation innovations (i.e., these statistics of GOWASP–3 observation values minus their corresponding values computed in the G5DAS during assimilation) were also examined. Since the simulated observations are without added simulated instrument error and are likely missing some representativeness error, and since an OSSE ingesting these too accurate observations should yield too accurate background fields, standard deviations of OSSE innovations should be smaller than those of real observations. Differences in innovation means may not compare as simply but should not be radically different in magnitude. This comparison assumes that radiances for the DAS applied to real observations have been bias corrected.

For some data classes, comparisons were also made by using GOWASP–3 software to compute observations from the same background fields as used by the observation operators in the G5DAS. In some such tests, observations were specified at only points on the background grid and at times when the background was available. This thereby precluded spatial and temporal interpolation. Radiances over water, where emissivity differences should be negligible, were identical up to the precision of the data in the BUFR files. For GPSRO, differences were significant since the G5DAS and GOWASP-3 compute bending angles using different algorithms.

Also compared were various metrics for OSSEs conducted using GOWASP–3 observations versus using data sets provided by JCSDA researchers. The latter were produced by different software using different data selection algorithms; e.g, satellite wind observations were located where trackable observations existed in reality and cloud contaminated radiances were located where clouds were in reality, irrespective of where these features were in the G5NR–1. Although there are some noteworthy differences due to the ways in which data locations are chosen and radiance contamination is introduced, the investigated metrics were very similar. Although a more complete comparison may yield more noteworthy differences, the examination thus far suggests the GOWASP–3 data sets have been created as intended.

Chapter 4

Simulation of Observation Errors

Most metrics used to assess real atmospheric data assimilation or forecast systems are determined by the various types of errors inherent in those systems. These include observation instrument errors, observation representativeness errors, forecast model formulation errors, errors introduced by the data assimilation algorithm (such as phase errors introduced by an incremental analysis update scheme or bias correction errors due to a wrong partitioning of bias between observation and background values), and modifications of these errors due to the influence of dynamical and physical processes acting upon them. Validation of an OSSE therefore depends on how well all of these errors are simulated. It is not simply about simulating observations, but more specifically about simulating the instrument plus representativeness errors that impact analysis and forecasts.

4.1 Consideration of Representativeness Error

It is critically important to understand how observation representativeness error manifests itself in an OSSE context. The following extends the standard notation of Ide et al. (1997):

- $\mathbf{z}$ denotes either the real atmospheric temporally– and spatially–continuous fields or simulations of those fields provided by the NR as a finite set of gridded values or spectral coefficients.
- $\mathbf{x}^t = \mathbf{X}(\mathbf{z})$ denotes the representation of $\mathbf{z}$ on the analysis grid at a discrete set of times. The superscript t denotes truth. When $\mathbf{z}$ denotes the real atmosphere, $\mathbf{x}^t$ is generally unknown

and therefore considered as only "postulated", but for the NR it can be precisely defined by some up–scaling or generalized interpolation algorithm. The representation $\mathbf{x}^t$ also may consist of a different set of fields than denoted by $\mathbf{z}$; e.g., a simplified description of a larger set of atmospheric constituent concentration fields.

- $\mathbf{x}$ denotes an estimate of $\mathbf{x}^t$, such as that to be determined by the DAS.
- $\mathbf{x}^b$ denotes a prior (or background) estimate of $\mathbf{x}$, typically produced by extrapolating past analyzed observations to the present using a short–term forecast model, as produced within a cycling DAS.
- y^o denotes the value of an observation by an imperfect instrument.
- $y^t = H_z(\mathbf{z})$ denotes what an imagined perfect instrument would provide as an observation value based on the real relationship between the complete, true state, $\mathbf{z}$ and what is observed in terms of the complete and true geometry and physics of the relationship (e.g., considering spatial integration of radiance by the receiving antenna and radiative emissions, transfer, and scattering by all of the atmospheric and surface constituents).
- $y = H(\mathbf{x})$ represents an observation as determined from an algorithm applied to $\mathbf{x}$ by the DAS. H generally includes spatial and temporal interpolation functions applied to the DAS fields represented by $\mathbf{x}$ and may also include computationally fast (and therefore approximated) radiative transfer schemes applied to the interpolated values.
- $\epsilon^i = y^o - y^t$ denotes the error of the observing instrument. For real observations, because the truth is never known (it can only be postulated), ϵ^i is never known although its statistical properties may be estimated by some means under some conditions. Since the OSSE employs no actual instruments, ϵ^i must be explicitly simulated. Its values can therefore be precisely known.
- $\epsilon^r = H(\mathbf{x}^t) - H_z(\mathbf{z})$ denotes the error of observation representativeness. This is specifically the difference between what the DAS would determine as the observation value if its representation of the truth was determined directly by that truth (instead of only estimated by an imperfect data assimilation algorithm) and what the true observation would be given a perfect instrument and a perfect relationship between the complete atmospheric state and what is physically observed. These values differ because H is only an approximation to H_z and it only operates on the approximation $\mathbf{x}^t$ of the true state $\mathbf{z}$. For real observations, ϵ^r is also only a postulated value; i.e., although its statistics may be estimated, its realizations are generally unknowable.

In the OSSE context, a portion of ϵ^r is implicitly created when observations are simulated because generally the algorithms used to simulate and assimilate are not identical; for some observation types, different algorithms may be employed for H and H_z, and even when appearing the same (e.g., both use bi-linear interpolation operators), the operations are applied to different grids with different representations $\mathbf{x}$ and $\mathbf{z}$. Since the algorithms and grids employed in the assimilation and simulation are most likely more similar to each other than those for the assimilation are faithful to the real relationships in nature, typical magnitudes of this implicit portion of simulated ϵ^r likely underestimate typical magnitudes of corresponding real ϵ^r. Thus, additional simulated ϵ^r must be added to the GOWASP–3 observations if the observation errors are to be realistic.

Although ϵ^r results from modeling an observation (in terms of H), it is considered as one of the two contributors to observation error. This consideration is a consequence of how errors due to imperfect H enter the fundamental data fitting metric (i.e., the cost function) in the DAS algorithm. The other contributor to observation error is the instrument error ϵ^i. It is the (total) observation error $\epsilon^o = \epsilon^i + \epsilon^r$ that must be simulated. Specifically, its statistics in the OSSE and real contexts should match. Note that for most observations, values of ϵ^r typically dominate those of ϵ^i; i.e., inaccuracies of observations are due more to how they are inaccurately related to what is being analyzed than to instrument inaccuracies.

In a DAS applied to real observations, H_z is a fixed relationship provided by nature. In that context, making H more realistic implies reducing typical (e.g., root mean square) values of ϵ^r. In contrast, in the OSSE context for which the H employed by the DAS is kept fixed, making the H_z more realistic than H actually increases the typical magnitudes of ϵ^r. In the OSSE context therefore, adding more realism to the simulation of observations from the NR does not increase the utility of the observation signal but instead increases its noise. This will act to pull the analysis further from the truth. Furthermore, if the H_z is sufficiently complicated, non–Gaussian error statistics may be introduced, including possibly gross errors that should be appropriately treated by the DAS QC procedures. Great care must be taken, therefore, when using a H_z very different from the usually simple H employed by a DAS.

4.2 Simulation of Added Observation Error

The GOWASP–3 uses one of four formulas for creating observation errors to add. The simplest is for errors that are uncorrelated. A second is for observation errors whose correlations are only considered to be vertical. It is intended particularly for GPSRO or conventional sounding observations (i.e., radiosonde or dropsonde reports). A third is for observation errors modeled as combinations of possibly uncorrelated and horizontally correlated errors. This is intended for some conventional observations, including AMVs and most other single–level reports. It is also intended for radiance observations with no inter–channel correlated errors. The fourth formula is for radiance observation errors that are inter–channel and perhaps horizontally correlated. This is intended particularly for hyper–spectral instruments.

The most general formula for creating simulated error is the fourth:

$$\epsilon(\lambda_k, \theta_k, z_k) = \epsilon_f \left[\sqrt{\nu_k}\ \gamma_k + \sqrt{1-\nu_k}\ \beta_k\right] \ . \tag{4.2.1}$$

where γ_k and α_k are computed from

$$\gamma_k = \sum_i h_i(\lambda_k, \theta_k) s_i \phi_i(z_k) \ , \tag{4.2.2}$$

$$\beta_k = \sum_i \alpha_i(\lambda_k, \theta_k) s_i \phi_i(z_k) \ . \tag{4.2.3}$$

Here,

- ϵ is the simulated error at longitude λ_k, latitude θ_k, and pressure level, height level or channel number index z_k,
- ϵ_f is a scalar factor that introduces a simple, location–independent rescaling of the sizes of all simulated errors for a given observation type,
- $\nu_k = \nu(z_k)$ is the fraction of total simulated error variance associated with the horizontally correlated part of the error (in GOWASP–3, ν_k is only permitted to be a function of z_k),
- h_i are a discrete set of random 2–dimensional fields defined on the sphere for specified shapes and parameters of correlation functions (in GOWASP–3, the functional forms of these shapes must be identical for all i except for a single length-scale parameter that may depend on i),

- α_i are a set of uncorrelated random numbers, each drawn from a distribution with mean 0 and variance 1,
- s_i are square roots of the eigenvalues of a prescribed, expected, vertical or channel, error covariance matrix defined for a discrete set of pressure or height levels or channel numbers. (When (4.2.1) is applied in GOWASP–3, this matrix must be independent of horizontal location.)
- $\phi_i(z_k)$ are a set of normalized eignvectors (or principle components, PCs) corresponding to the s_i.

Note that (4.2.1) is not completely general because s_i, ϕ_i and ν_k are independent of horizontal location, imposing a form of separability on the allowed error covariances. The GOWASP–3 algorithms for creating h_i, s_i and ϕ_i are described later in this chapter.

The third formula is of the general form

$$\epsilon(\lambda_k, \theta_k, z_k) = \epsilon_f \left[\sqrt{\nu_k}\ \gamma_k + \sqrt{1 - \nu_k}\ \alpha_k \right] \ . \tag{4.2.4}$$

It is like (4.2.1), except that the horizontally uncorrelated part has no vertical or channel correlation. The α_k are therefore like the former α_i except directly applied to level or channel z_k.

The second formula is of the general form

$$\epsilon(\lambda_k, \theta_k, z_k) = \epsilon_f \sum_i \alpha_i s_i \phi_i(z_k) \ . \tag{4.2.5}$$

It only considers possible vertical correlations. Implicitly, it also includes an uncorrelated part when defined by a suitable prescribed covariance matrix. Unlike for (4.2.1), however, this covariance matrix may vary with horizontal location in GOWASP–3 since spatial separability is less of an issue.

The first formula allows no correlation. It is simply

$$\epsilon(\lambda_k, \theta_k, z_k) = \epsilon_f \alpha_k \ . \tag{4.2.6}$$

As for the earlier formulas, the set of α are independent for different observation classes or subtypes. (While this independence is very reasonable for instrument errors, it is much less so for representativeness errors, since different instruments can see the same subgrid fields or use similar forward observation operators.)

More general forms of simulated errors than those prescribed for GOWASP–3 may be desired. Some possibly desired aspects are described in later sections of this chapter. Difficulties in creating more general algorithms lie not only in issues of computational efficiency but also in the tuning of their parameters.

The covariance between a pair of observations produced using (4.2.4) is

$$\begin{aligned}\langle\epsilon(\lambda_j,\theta_j,z_j)\epsilon(\lambda_k,\theta_k,z_k)\rangle &= \epsilon_f^2\delta_{j,k}\left(1-\nu_k\right)\sigma_k^2 \\ &+ \epsilon_f^2\sqrt{\nu_j\nu_k}\sum_i\phi_i(z_j)\phi_i(z_k)s_i^2\langle h_i(\lambda_j,\theta_j)h_i(\lambda_k,\theta_k)\rangle\ ,\end{aligned}\tag{4.2.7}$$

where angle brackets denote expectations, $\delta_{j,k}$ is the Kronecker delta, and it has been assumed that the random values of α and h are independently chosen such that

$$\langle\alpha_j\alpha_k\rangle = \delta_{j,k}\ ,\tag{4.2.8}$$

$$\langle\alpha_j h_i(\lambda_k,\theta_k)\rangle = 0\ ,\tag{4.2.9}$$

$$\langle h_i(\lambda_j\theta_j)h_m(\lambda_k,\theta_k)\rangle = \delta_{i,m}\langle h_i(\lambda_j,\theta_j)h_i(\lambda_k,\theta_k)\rangle\ .\tag{4.2.10}$$

The last condition holds if the random fields applied to distinct vertical or channel PCs are independent. With the h normalized such that the expected variance of the field at any location is 1, then $\langle h_i(\lambda_j,\theta_j)h_i(\lambda_k,\theta_k)\rangle$ is the horizontal correlation (denoted here as $\mathcal{H}$) of PC coefficients. If this horizontal correlation is the same for all PCs, then

$$\begin{aligned}\epsilon(\lambda_j,\theta_j,z_j)\epsilon(\lambda_k,\theta_k,z_k)\rangle &= \epsilon_f^2\delta_{j,k}\left(1-\nu_k\right)\sigma_k^2 \\ &+ \epsilon_f^2\sqrt{\nu_j\nu_k}c_{j,k}\mathcal{H}_i(\lambda_j,\theta_j,\lambda_k,\theta_k)\ ,\end{aligned}\tag{4.2.11}$$

where, given that the PC structures ϕ are orthonormal,

$$c_{j,k} = \sum_i\phi_i(z_j)\phi_i(z_k)s_i^2\tag{4.2.12}$$

is the covariance matrix from which the PCs were derived. Further details of these properties are described later in this chapter.

The covariance between a pair of observations produced using (4.2.1) is

$$\begin{aligned}\epsilon(\lambda_j,\theta_j,z_j)\epsilon(\lambda_k,\theta_k,z_k)\rangle = \epsilon_f^2\sqrt{(1-\nu_j)\,(1-\nu_k)}\delta_{\theta_j,\theta_k}\delta_{\lambda_j,\lambda_k}\sum_i \phi_i(z_j)\phi_i(z_k)s_i^2 \\ + \epsilon_f^2\sqrt{\nu_j\nu_k}\sum_i \phi_i(z_j)\phi_i(z_k)s_i^2\mathcal{H}_i(\lambda_j,\theta_j,\lambda_k,\theta_k)\rangle \ .\end{aligned} \tag{4.2.13}$$

For a pair at the same geographic (latitude and longitude) location but possibly different values of z, this reduces to

$$\langle\epsilon(z_j)\epsilon(z_k)\rangle = c_{j,k}\left[\sqrt{(1-\nu_j)\,(1-\nu_k)} + \sqrt{\nu_j\nu_k}\right] \ . \tag{4.2.14}$$

Thus, the covariances obtained differ from the covariance matrix specified. If the specified matrix elements are appropriately adjusted, however, using the inverse of the factor in brackets, then the desired covariance will be obtained. Note that

$$\langle\epsilon^2(\lambda_j,\theta_j,z_j)\rangle = c_{i,i} \ , \tag{4.2.15}$$

even without such adjustment.

4.3 Consideration of Correlated Observation Errors

The G5DAS and most other current DAS are designed to especially filter random uncorrelated errors from observations. This they do very effectively as long as the spatial densities of relatively independent observations remain high. Most correlated errors, however, will be retained by the system. The uncorrelated errors therefore have little effect on the analysis but the correlated ones do. For an OSSE to realistically mimic the retained errors, it is therefore imperative that the simulated observation errors include any significant correlations found in real observation errors.

As part of the GOWASP–3 validation, various kinds of correlations of observation innovations

$$d = y^o - H(\mathbf{x}^b) \tag{4.3.1}$$

between observations of the same type are compared for DAS applied to real and OSSE contexts. This includes correlations between channels for hyper-spectral IR brightness temperatures (T_B), horizontal and vertical correlations for satellite wind observations, vertical correlations for conventional sonde observations and GPSRO bending angles, and horizontal correlations of T_B for some MW observations. These correlations of d arise due to both correlations of observation errors and correlations of background error. If corresponding correlations do not match in the real and OSSE contexts, then either the observation or background error correlations are dissimilar in the two contexts or the ratios of observation and background error variances are dissimilar. The latter can affect the innovation correlations by affecting the mix of observation and background correlations contributing to them.

When it can be reasonably assumed that the background error variances and correlations in the OSSE and real contexts are similar, then any large differences between corresponding innovation correlations can be attributed to unrealistic statistics for the OSSE simulated observation errors. Once the observation error variances are tuned so that the innovation variances sufficiently match, observation error correlations can be introduced so that the innovation correlations also match. If the background error statistics poorly match, however, compensating for them by instead adjusting the observation error variances or correlations may be unproductive for simultaneously matching other aspects of the DAS behavior, since the discrepancy's cause is thereby ascribed to the wrong source.

4.3.1 Calculation of vertically correlated errors

Let $c_{k,l} = \langle \epsilon(z_k)\epsilon(z_l) \rangle$ be the covariance between errors at a set of K levels $\mathbf{z} = z_1, z_2, \ldots, z_K$, where angle brackets denote the mean of a set of realizations (i.e., an ensemble mean). This c may also be expressed as

$$c_{k,l} = \sigma_k r_{k,l} \sigma_l \ , \tag{4.3.2}$$

where $-1 \le r \le 1$ is the error correlation between 2 levels and the σ_k are the error standard deviations on those levels. For a Gaussian–shaped, vertical correlation function,

$$r_{k,l} = \exp\left[-0.5(z_k - z_l)^2 / L^2\right] \ , \tag{4.3.3}$$

where L is a correlation length. For $|z_k - z_l| = L$, the correlation is $1/\sqrt{e} = 0.6$.

Next consider the matrix $\mathbf{\Phi}$ whose columns $\phi_i(z_k)$ $(i = 1, \ldots, K)$ are ortho–normal eigenvectors of the matrix $\mathbf{C}$ whose elements are the $c_{k,l}$. These ϕ_i are also called principal components (PCs) or, in meteorology, empirical orthogonal functions (EOFs). The orthonormalization is such that

$$\delta_{i,j} = \sum_k \phi_i(z_k)\phi_j(z_k) \ , \tag{4.3.4}$$

where δ here is the Kronecker delta. The orthogonality of the eigenvectors is guaranteed since C is a symmetric matrix but the normalization constant 1 is required in how these ϕ are to be applied. For each i there is also a s_i that is the square root of the corresponding eigenvalue. If $\mathbf{S}$ denotes the diagonal matrix whose elements are the s_i^2,

$$\mathbf{C\Phi} = \mathbf{\Phi S} \ . \tag{4.3.5}$$

The ortho–normality may be expressed as

$$\mathbf{I} = \mathbf{\Phi}^T \mathbf{\Phi} \ , \tag{4.3.6}$$

where $\mathbf{I}$ is the multiplicative identity matrix.

Errors with the desired correlations and variances may be created from

$$\epsilon(z_k) = \sum_{i=1}^{K} \alpha_i s_i \phi_i(z_k) \ , \tag{4.3.7}$$

where the α_i are random coefficients drawn from some distribution having mean 0 and variance 1. In GOWASP–3, this is a truncated normal (Gaussian) distribution, but it need not be. (The truncation removes $|\alpha|$ greater than several standard deviations.) Note that the choice of random distribution for α can differ from the shape of the vertical correlation function, although in GOWASP–3 it is also a Gaussian.

For atmospheric observations that include the pressure level rather than height at which the observation is made, the correlations are still considered as functions of something like distance. Specifically, a vertical separation $D \propto \log(p_k/p_l)$ is considered. This is not the same as using a hydrostatically defined distance, since computing that distance would require profiles of T and q that are not generally available in the program ***create_error***.

An example of vertical covariances and principal components appears in Fig. 4.1. It has been obtained using p–dependent standard deviations of added observation errors for wind components tuned for SATWINDs obtained from the Himawari geostationary satellite. These standard deviations appear in the left panel. Correlations assume a Gaussian structure in hydrostatic height (not pressure), with a correlation length of 350 m, independent of height. Covariances with errors at 700, 500, and 300 mb appear in the center panel. For PCs ordered by decreasing explained variances, the normalized structures for PC numbers 1, 4, and 17 appear in the right panel. The variances associated with these PCs are 42.9, 36.1, and 14.9 $\mathrm{m^2s^{-2}}$, respectively. Note that for the

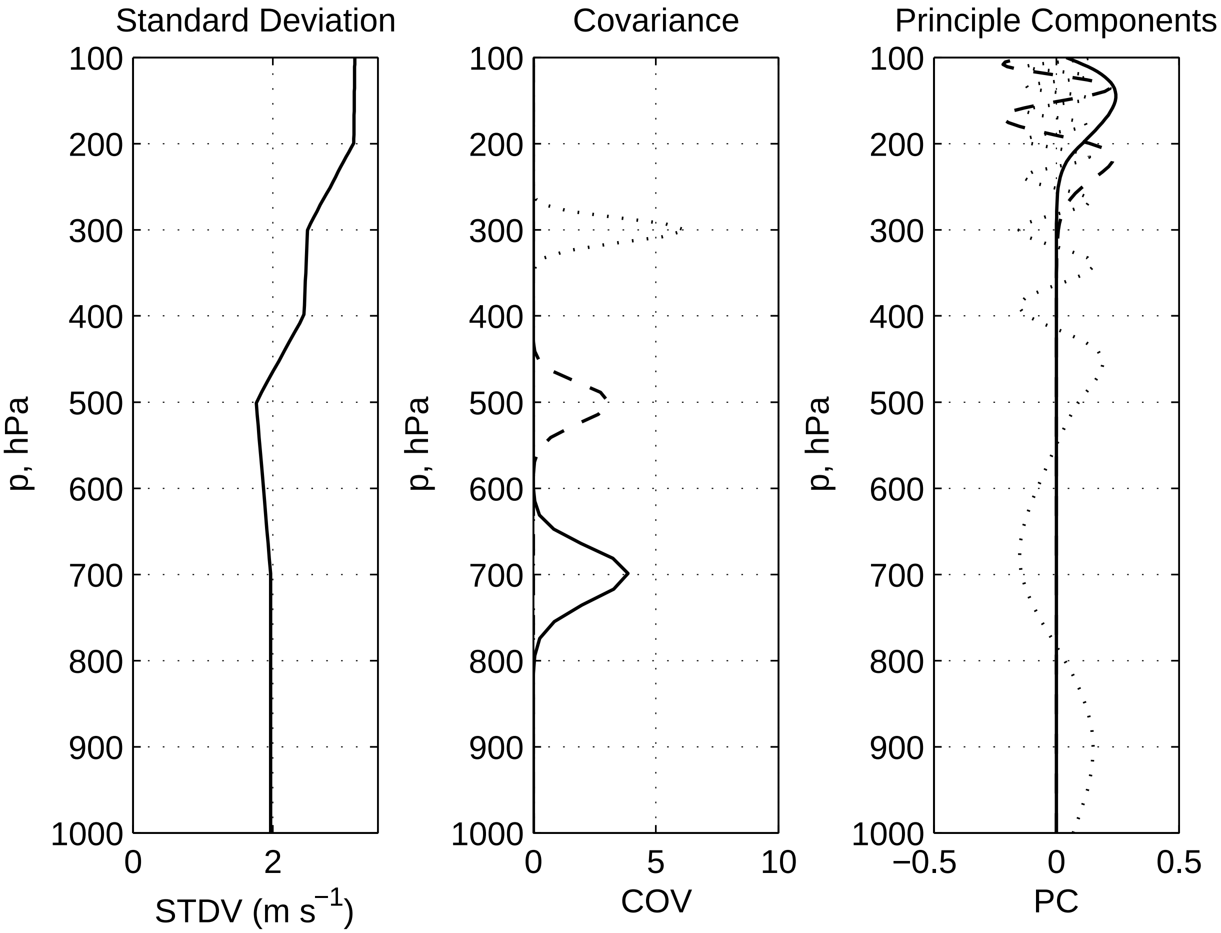


Figure 4.1: An example vertical covariance function and some of its principal components. Left panel: Error standard deviation (units ms^{-1}) as a function of pressure. Center panel: Covariances with values at 700 (solid line), 500 (dashed line), and 300 mb (dotted line). Right panel: Normalized structures for the 1st, 4th, and 17th leading PCs (solid, dashed, and dotted curves, respectively).

leading PC (i.e., number 1), the structure does not change sign with height, unlike for the other PCs.

4.3.2 Calculation of channel correlated errors

The creation of channel correlated errors follows the procedure for creating vertical correlations, except indexes for vertical levels are replaced by channel numbers. Although errors are created for all channels in an instrument's data set, only those actually assimilated by the G5DAS have correlations between pairs of channels. This is primarily because currently we have no way to estimate the correlations for unused channels in the context of real observations but also because there is no need to consider unused ones in a G5DAS application.

4.3.3 Calculation of horizontally correlated errors

The procedure used for generating errors correlated in the vertical is quite general. Its applicability to some other seemingly equivalent contexts, however, is computationally limited. In particular, attempting to use it for constructing horizontal correlations would require determination of PCs of huge matrices. Fortunately a simpler procedure can be employed in the latter case if the horizontal correlations are designed to be homogeneous and isotropic, meaning their statistics are to be independent of location and separation direction, respectively.

For GOWASP–3, horizontally–correlated observation errors (or horizontally–correlated PC coefficients for vertically or channel correlated errors) are created by first generating random correlated fields $h_j = h(\lambda_j, \theta_j, z_j)$ on some high–resolution global grid, where λ is longitude, θ is latitude, and z denotes either a vertical level, a channel index, or an index for coefficients of PCs. (Use of the latter is described in the next section.) Corresponding to each θ_j is a

$$\mu_j = \sin\theta_j \ . \tag{4.3.8}$$

The horizontal distance D between a pair of points (λ_j, θ_j) and (λ_k, θ_k) is measured along a great circle connecting the points:

$$D = 2a \arcsin\left[\sin^2\left(\frac{\theta_j - \theta_k}{2}\right) + \sin^2\left(\frac{\lambda_j - \lambda_k}{2}\right)\cos\theta_j\cos\theta_k\right]^{\frac{1}{2}} \ , \tag{4.3.9}$$

where a here is the earth's radius. Other formulas can be used for this purpose, but this is sufficient for the intended purpose as long as the desired correlation length scales are not too small (e.g., not less than 10 km).

The random fields $h(\lambda_j, \theta_j, z_j)$ are computed such that they are independent either for different radiance channels, heights, or PC coefficients. Correlations between points having indexes j and k are therefore

$$C_{j,k} = \langle h_j h_k \rangle \delta(z_j, z_k) \ , \tag{4.3.10}$$

where δ here is the Kronecker delta conditioned on whether $z_j = z_k$. When $\delta = 1$, horizontal homogeneity and isotropy imply that $C_{j,k}$ only varies with D, although parameters defining that function may depend on the value of z_k; i.e., they may vary with level or channel or PC coefficient index. Hereafter, this dependence on z_k will not be explicitly noted.

The h can be constructed using

$$h(\lambda_j, \theta_j) = \sum_{n=0}^{N} \sum_{m=-n}^{n} \alpha_{m,n} P_n^m(\mu_j) \exp im\lambda_j \ , \tag{4.3.11}$$

where the factor i here denotes $\sqrt{-1}$, α are complex random coefficients, $P_n^m(\mu_j)$ are associated Legendre polynomials of degree n and order m, and N is a spectral truncation parameter. Each product

$$\Psi_n^m(\lambda_j, \mu_j) = P_n^m(\mu_j) \exp im\lambda_j \tag{4.3.12}$$

denotes a particular spherical harmonic function. In what follows, it is assumed that the P_n^m and Ψ_n^m are normalized such that

$$1 = \int_{-1}^{1} [P_n^m(\mu)]^2 \ d\mu \ , \tag{4.3.13}$$

$$1 = \int_{-1}^{1} \int_{0}^{2\pi} \Psi_n^m(\lambda, \mu) \Psi_n^{m*}(\lambda, \mu) \ d\lambda \ d\mu \ , \tag{4.3.14}$$

for each m, n, where the asterisk denotes complex conjugate.

The α are random complex coefficients whose statistics are determined by the desired horizontal correlation function. For the h to be real–valued,

$$\alpha_{-m,n} = \alpha^*_{m,n} \ , \tag{4.3.15}$$

and therefore only values for $m \geq 0$ need be specified. Note that for $m = 0$, $\Im(\alpha) = 0$. Homogeneity and isotropy imply that the real ($\Re(\alpha)$) and imaginary ($\Im(\alpha)$) components of each independent $\alpha_{m,n}$ should be independent random draws from a distribution with mean 0 and variance dependent on n only:

$$V = \begin{cases} V_n & \text{for } m = 0 \text{ or } m = N, \\ 0.5V_n & \text{otherwise.} \end{cases} \tag{4.3.16}$$

The condition for $m = N$ is required if the random field is discretized such that $2N$ is the number of longitudes, in which case the coefficients for $m = N$ must be real–valued. For each k, $\tilde{V}_n = (2n + 1)V_n$ describes the 2–dimensional power spectrum for the random field. Note that V_n is independent of m and that $2n+1$ is the number of existing zonal wavenumbers (positive, negative and zero valued) for a given value of n. It is also the number of independent real and imaginary components of $\alpha_{m,n}$ that are not identically 0.

It remains to specify the desired V_n for the desired correlation $C(D)$. Their relationship is determined by inserting (4.3.11) into (4.3.10) to obtain

$$C_{j,k} = \sum_{n=0}^{N} \sum_{m=-n}^{n} \sum_{n'=0}^{N} \sum_{m'=-n'}^{n'} \langle \alpha_{m,n} \alpha^*_{m',n'} \rangle \Psi_n^m(\lambda_j, \mu_j) \Psi_{n'}^{m'*}(\lambda_k, \mu_k) \ . \tag{4.3.17}$$

For C to be only dependent on the separation D and not on horizontal location, it is necessary that $\langle \alpha_{m,n} \alpha^*_{m',n'} \rangle$ vanish unless both $m = m'$ and $n = n'$. This leads to the simplification

$$C_{j,k} = \sum_{n=0}^{N} \sum_{m=-n}^{n} V_n \Psi_n^m(\lambda_j, \mu_j) \Psi_n^{m*}(\lambda_k, \mu_k) \tag{4.3.18}$$

which further simplifies to

$$C(D) = \sum_{n=0}^{N} \sqrt{2n+1} V_n P_n^0(D) \tag{4.3.19}$$

after applying the“addition formula for spherical harmonics”

$$P_n^0(D) = \frac{1}{\sqrt{2n+1}} \sum_{m=-n}^{n} \Psi_n^m(\lambda_j, \mu_j) \Psi_n^{m*}(\lambda_k, \mu_k) \ . \tag{4.3.20}$$

The addition formula appears differently here than in some textbooks because of the non–standard but appropriate normalization (4.3.13)– (4.3.14). Utilization of the orthonormality of the P_n^0 leads to

$$V_n = \frac{1}{\sqrt{2n+1}} \int_{-1}^{1} C(D(\mu)) P_n^0(\mu) d(\mu) \tag{4.3.21}$$

since P_n^0 is independent of λ and thus

$$D(\mu) = a \left(\frac{\pi}{2} - \arcsin(\mu) \right) \ . \tag{4.3.22}$$

In other words the desired variances of the random spherical harmonic coefficients are simply determined by projecting the desired correlation in terms of distance onto the Legendre polynomials of degree 0.

Before being scaled by a desired error variance, the V_n from (4.3.21) are renormalized by applying

$$S = \sum_{n=0}^{N} (2n+1) V_n \ , \tag{4.3.23}$$

$$V_n(\text{normalized}) = V_n / S \ . \tag{4.3.24}$$

The expected spatial mean of the variance of the corresponding random field will be 1 if this normalized V_n is used in (4.3.16) without further rescaling.

The GOWASP–3 considers 4 different shapes of horizontal correlation functions although it is general enough to consider almost any appropriate function after simple modification. Three current choices are:

$$C(D) = \begin{cases} \exp\left(-0.5\frac{D^2}{L^2}\right) & \text{for a Gaussian shape,} \\ \exp\left(-\frac{|D|}{L}\right) & \text{for an exponential shape,} \\ \left[1+\frac{|D|}{L}+\frac{D^2}{3L^2}\right]\exp\left(-\frac{|D|}{L}\right) & \text{for a TOAR shape.} \end{cases} \tag{4.3.25}$$

TOAR refers to a third-order auto-regressive function. The fourth choice is for white noise in 2 dimensions, meaning no correlation. For this choice, $V_n = 1$ in place of (4.3.21), meaning that the expected variances of all the moduli of random spectral coefficients are identical.

If the random fields are for vector wind fields rather than for scalar fields such as temperature, the appropriate spherical harmonic functions are for the scalar fields of vorticity and divergence from which wind fields will be determined. For specified correlations intended to apply to the winds, however, the power spectra for the scalar fields must therefore be altered by using a W_n in place of V_n, with

$$W_n = \frac{n^2+n}{a^2} V_n \; . \tag{4.3.26}$$

The factor multiplying V_n converts the power, and hence variance, of the vorticity and divergence fields to that for wind fields having horizontal mean variance of 1. A normalization is applied as in (4.3.23)–(4.3.24) but with W_n replacing V_n.

Once the h are created, errors of observations or error PC coefficients are drawn from them by interpolating the values of h to the observation locations. The resulting PC coefficient or observation error values will have the desired horizontal correlations. It is critically important that the h be defined on a grid whose spacing is much less than the prescribed correlation length L. Otherwise, the interpolation will be between effectively uncorrelated values, masking the desired correlation and diminishing the error variance.

As an example of the previous point, consider linear interpolation between values of a random field at two locations. If those random values are uncorrelated or the correlation distance is much shorter than the separation distance between them, then the expected variance V for the interpolated value is

$$\begin{aligned} v &= \langle [wh_1 + (1-w)h_2]^2 \rangle \ , \\ &= w^2 \langle h_1^2 \rangle + 2w(1-w)\langle [h_1 h_2] \rangle + (1-w)^2 \langle h_2^2 \rangle \ , \\ &= (1 - 2w + 2w^2)\langle h^2 \rangle \ , \end{aligned} \tag{4.3.27}$$

where $0 \leq w \leq 1$ is a weight and $\langle h^2 \rangle$ is the common variance of h at the points. For a point midway between the points of h, $w = \frac{1}{2}$ and $v = \frac{1}{2}\langle h^2 \rangle$. So, the variances of observation errors drawn from the random fields can be less than the specified variances of those fields. In contrast, as the correlation between the 2 points of h approaches 1, $v \rightarrow \langle h^2 \rangle$ for all values of w.

4.3.4 Calculation of errors correlated both horizontally and by channel or level

Observation errors can also be generated that are both horizontally correlated and either channel or vertically correlated by combining the aforementioned algorithms. There are some limitations, however, involving a form of separability. Namely, these random fields are created by restricting

$$f_k(\lambda, \mu) = \sum_{l=1}^{K} h_l(\lambda, \mu) s_l \phi_l(z_k) \ , \tag{4.3.28}$$

where h are random horizontal fields and l is a PC index rather than a height or channel index. Note that these h are defined for the K values of l, not k. The resulting horizontal correlations of f may therefore vary with k. For a prescribed set of $\phi_l(z_k)$, however, how to prescribe the correlation lengths for h_ℓ to obtain desired horizontal correlations in f_k is beyond the scope of this algorithm.

Horizontal correlations were first used at the GMAO to simulate SATWIND observation errors without simultaneously considering vertical correlations. Since many observations were located between levels on which h were defined, values of random error were vertically interpolated between defined levels of h. This resulted in error variances being notably less than those prescribed for the h. In fact, when observation error variances were examined as a function of pressure, pronounced peculiar structures were obtained, with maxima at the levels specified for h and minima midway between them. The reason for this was explained at the end of the previous section.

For observation error values that will be interpolated both vertically and horizontally between values of a prescribed f, it is critical that f be created with simultaneous horizontal and vertical correlations if peculiar vertical structures of variance are to be avoided. The grids on which f are defined must have sufficient horizontal and vertical resolution to resolve the prescribed correlations. Since the interpretation of correlation length L can vary with the shape of the correlation function, no specific relationship between L and grid spacing Δx or Δz is offered here. For Gaussian–shaped correlations, having both $\Delta x < L_h/5$ and $\Delta z < L_v/5$ appears sufficient. It is best, however, if the resolution is tested by creating some f and examining variances for samples of interpolated values to see if any obtained structural peculiarities are acceptable.

4.4 Examples of Random Correlated Fields

Examples of random correlated fields are shown in this section. These are produced using the algorithm described in this chapter assuming homogeneous and isotropic statistics with mean 0, variance 1, and Gaussian functions of separation distance for the prescribed correlations. Rather large values of correlation distance parameter L were used to produce these plots in order to make the correlations more visible.

Four random scalar fields appear in Fig. 4.2. Figures on the left used $L = 2000$ km and on the right $L = 500$ km. Fields on the top and bottom were produced using different random seeds and thus display different realizations produced from the same applied statistics. Note that while it may appear that the fields are zonally smoother at high latitudes, that is only due to the poleward convergence of the meridians.

Examples of random correlated vector fields appear in Fig. 4.3. Variances for divergent and rotational components of the wind were specified equally. Left and right are again for $L = 2000$ km and 500 km, respectively. The field of u components of the vectors are shown above those for the corresponding v fields. These corresponding u and v fields constitute a single realization. Note that near the poles, the u and v components are no longer independent since at the pole they must have a zonal wavenumber 1 structure and be exactly 90° out of phase with each other, as indeed shown here.

Power spectra for the second set of scalar random fields appear in Fig. 4.4 as a function of “total wavenumber” n. These were all produced using an expected total variance of 1. Panels show spectra for both randomly determined coefficients and their assigned expectations. Two of the panels are for Gaussian–shaped correlations with length scales 2000 km and 500 km and a third is for white noise. Note the shift in the peak of the spectrum between the first two panels. Also note that the expected spectra for white noise is proportional to $2n + 1$, which is the number of spectral coefficients (independent values of zonal wavenumbers m) for each value of n.

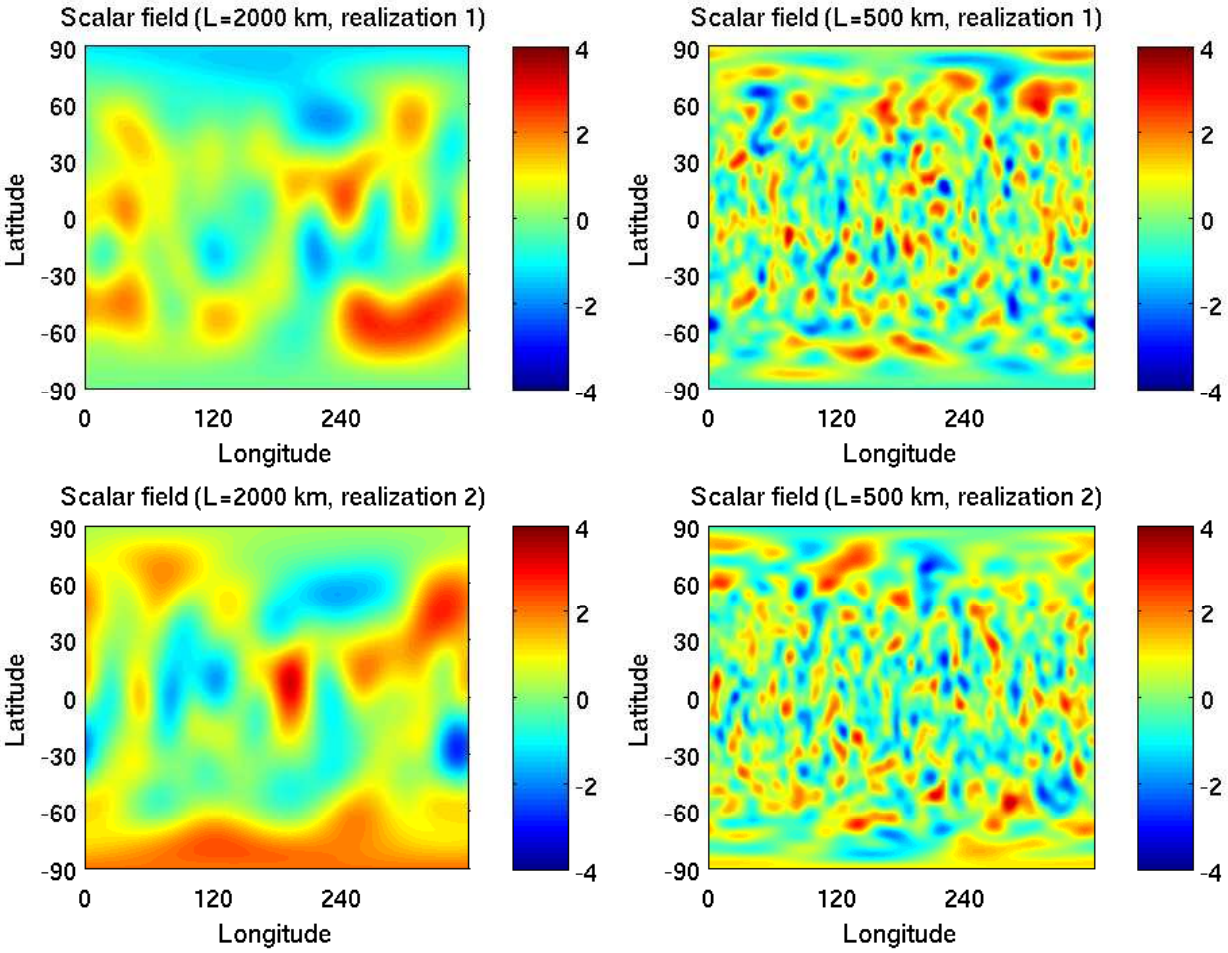


Figure 4.2: Four examples of isotropic, homogeneous, random scalar global fields using a Gaussian–shape for the correlations, a mean 0 and a variance 1. Left are for a specified correlation distance of 2000 km; right for 500 km. The top and bottom rows are for 2 different realizations, altered by changing the random seed.

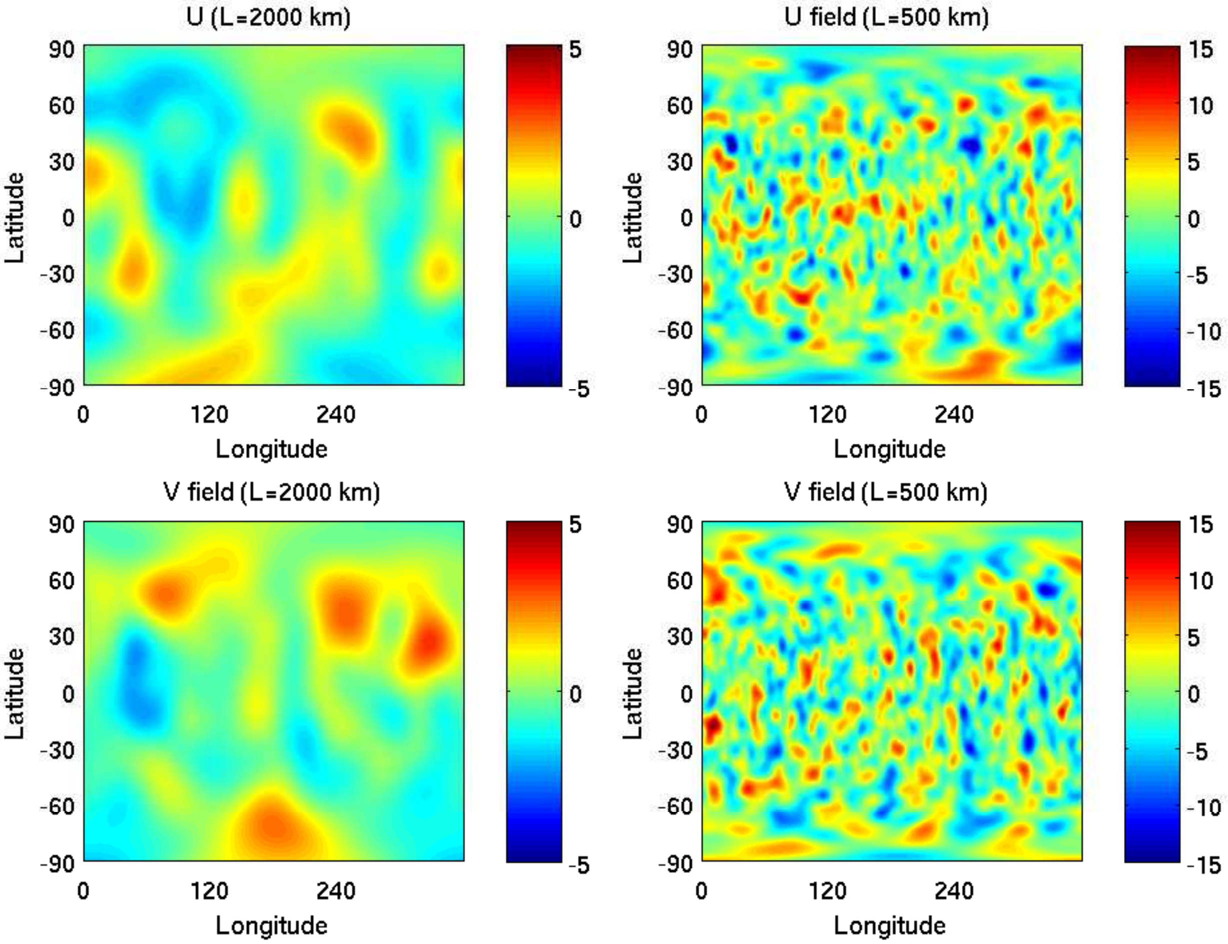


Figure 4.3: Two examples of isotropic, homogeneous, random vector global fields using a Gaussian–shape for the correlations, a mean 0 and a variance 1. Top and bottom are for u and corresponding v components, respectively. Left are for a specified correlation distance of 2000 km; right for 500 km.

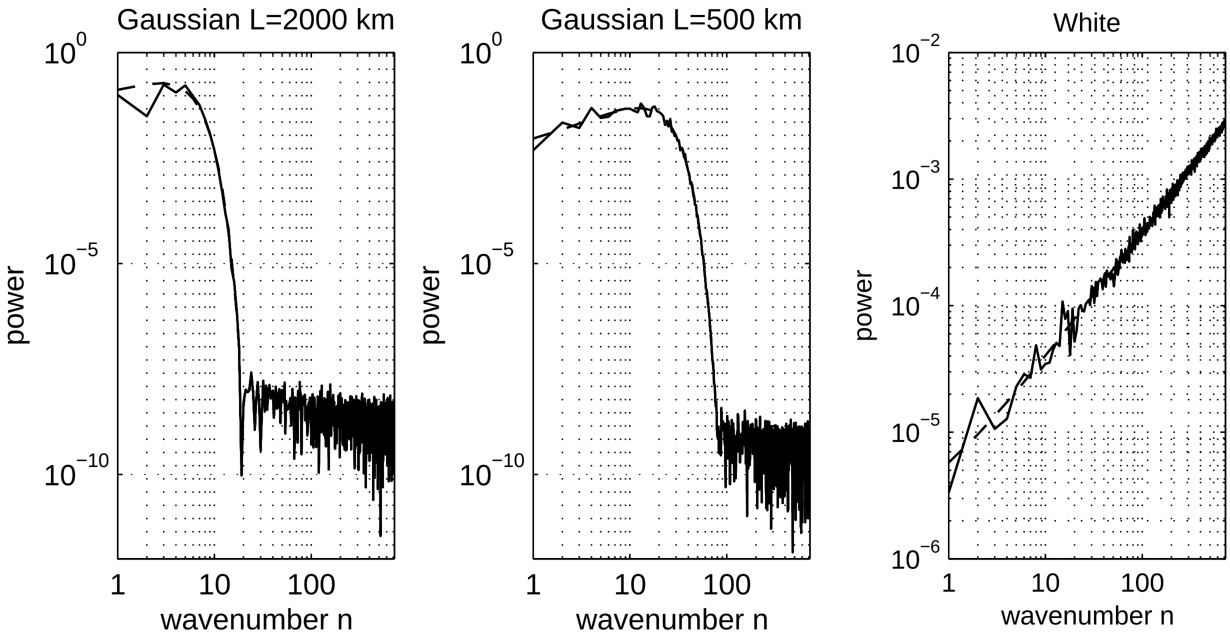


Figure 4.4: Power spectra for random scalar fields as functions of 2-dimensional (total) wave–number n. Solid lines are spectra determined from random coefficients. Dashed lines are for the corresponding assigned expected power for a total variance of 1. Left and center panels are for Gaussian shaped correlations with length scales of 2000 and 500 km, respectively. The right panel is for white noise (i.e., no correlation).

Chapter 5

User's Guide

Before reading this section, users should carefully read the description of the algorithms employed to simulate observations and their errors. The current section does not explain how software must be modified to incorporate new observation types.

Note that the terms "observation type" and "observation class" are used interchangeably here. The former is the term typically used in GOWASP FORTRAN comments and the latter in the G5DAS. Generally, separate types or classes are provided in separate observation data files. The term "subtype" refers either to a particular type of conventional observation or to an instrument on a particular satellite that is provided along with other subtypes in the same BUFR file.

Control of applications is exercised through the use of 3 input mechanisms.

- Program arguments: These are passed to the programs as specified in the application scripts. They include values for such variables as names of input, output and resource files, dates and times for the assimilation period, observation class name, and a testing flag.
- Resource files: These are text files that determine which NR fields and files are to be used, whether those fields should be spatially or temporally interpolated, what observation subtypes should be simulated, instructions for data thinning or contamination (e.g., by cloud or precipitation scattering of radiation), parameters that define the vertical levels in η–coordinates, and parameters for simulating observation errors.

- Binary input files: These are used by the software that simulates observation errors. Specifically, they contain information regarding desired channel or horizontal correlations the errors should exhibit.

Simulations of existing data types also require input BUFR files for observations existing at a corresponding real time. These are only used to provide observation locations or spatial and temporal distributions and other characteristics (such as scanning geometry for radiance instruments), but not observation values themselves.

The period of dates and times for the simulated data will be identical to the period of dates and times appearing in the file names of NR fields used. It will also be the period expected by the DAS for its OSSE observation data and restart input files. The period for the corresponding real data may be different, although it must agree with the dates and times in the BUFR files of real observations accessed. Time correspondence between the simulated and real time periods is set in the application scripts. The periods need not agree in month, day, year or hour. Recommended, however, is that they differ only in year. As the synoptic times for the simulated observations are incremented in the scripts, the corresponding times for the real observation BUFR files are incremented by the same amount.

Some applications require the use of random numbers. Reproducability is ensured by explicitly setting a random seed when the program begins. Generally, that seed is created by combining values from several integers. These include: the year, month, day, and hour of the synoptic simulation period, an integer that may vary with observation class, and an integer common to all classes but that may vary with experiment.

5.1 Applications

The GOWASP–3 applications may be placed into one of three groups. Group 1 includes applications that simulate all observation classes except for radiances. Group 2 includes all applications required to simulate radiances. Group 3 includes only the application that generates simulated observation errors.

Group 1 consists of the following applications:

- ***create_conv*** creates simulated conventional data, including all in–situ measurements but also some retrievals based on remote sensing by weather radars, scatterometers, and other instruments. In–situ observations include those by radiosondes, dropsondes, surface stations (land, ship, or buoy), and aircraft. Conventional data can also include atmospheric motion vectors (AMVs or SATWIND) although these are better simulated using other software designed for that specific purpose. The specific subtypes to be simulated are specified in a resource file. An existing BUFR file of type **.prepbufr** is read to specify the locations and times of all observations, although for sondes, this specifies only the initial (launch) locations and times, since their subsequent trajectories are determined by the NR fields.

- ***create_gpsro*** creates simulated GPSRO occultation bending angles.

- ***create_satwind*** creates simulated AMV observations as currently obtained from geostationary satellites and polar orbiters. This uses a probabilistic formulation to create observations whose location characteristics mimic those of real observations without attempting to track features in images created from the NR data sets.

- ***create_gp_raobs*** creates simulated radiosonde observations over the entire globe with a user–selected spatial density. This is primarily intended to provide a means of spinning up DAS restarts by using dense and accurate observations to force analysis from arbitrary backgrounds to states that resemble the NR fields. No advection of sondes is performed for this application.

Group 2 consists of 5 applications used for simulating radiance observations.

- ***create_rad_list*** reads real observation locations, times, scanning angles, etc. from observation header records within reports in existing BUFR files. Output are binary files of this information, excluding any actual radiance observation values, plus any NR field values at the observation locations that the user has specified for this application (generally, only values of 2–d fields used by the GOWASP–3 radiance data thinning algorithm). This data is ordered into a succession of time slots within each data assimilation period. The data may also be thinned using criteria specified by the user. For simulating observations that do not yet exist, it is best that the function of this application be mimicked, with equivalent files produced.

- ***create_rad_prof*** reads binary output from ***create_rad_list*** and extracts values from the NR fields for a user–specified list of field names. Output is a binary file of the binary input information along with values of the 2–d and vertical columns (profiles) of 3–d NR fields defined at the observation locations. The output data is ordered as for the input data.

- ***create_rad_reord*** takes the output from ***create_rad_prof*** and reorders it by observation subtype, usually referring to satellite platform. In all other aspects, the output file contains the same information as the input one.

- ***create_rad_bufr*** reads the reordered profile data file created by ***create_rad_reord***, passes it to the CRTM, and outputs simulated radiance data in BUFR format. The output files are in the same format as the BUFR files input to ***create_rad_list***. Values for any BUFR data variables not actually referenced by the G5DAS BUFR reader, however, will be absent for the simulated observations.

- ***create_rad_merge*** merges 2 separate files created by ***create_rad_reord*** from the same input list of observations. This permits the later inclusion of additional fields, such as cloud or aerosol profiles, without having to re–read NR files for field profiles previously extracted. Any redundancy in the two files to be merged is removed.

Group 3 consists of the the single application ***create_error***. This applications adds simulated observation errors to an existing set of observations. All observation classes are considered. Input and output observations are BUFR files. Various files containing observation–error simulation parameters are also input.

5.2 Use of Resource Files

There are several types of resource files. Some of them have common variables that require specification. Some have common formats. For a complete and detailed description of the formats and variables, the respective routines that read these files must be examined. Most users, however, should find sufficient explanation here to enable them to specify what they need if an existing file is used as a template. The resource file ***sat_info.rc*** is discussed in a separate section.

Each resource file begins with a single line that specifies the name **file_format** followed by a pair of integers that denote the file's format. Currently, these integers may be simply both set to 1. In future versions of the software it is possible that the expected file format may change, but having the file's format denoted first may permit legacy files to remain readable. There must be at least 1 blank space between the character string and each of the integers.

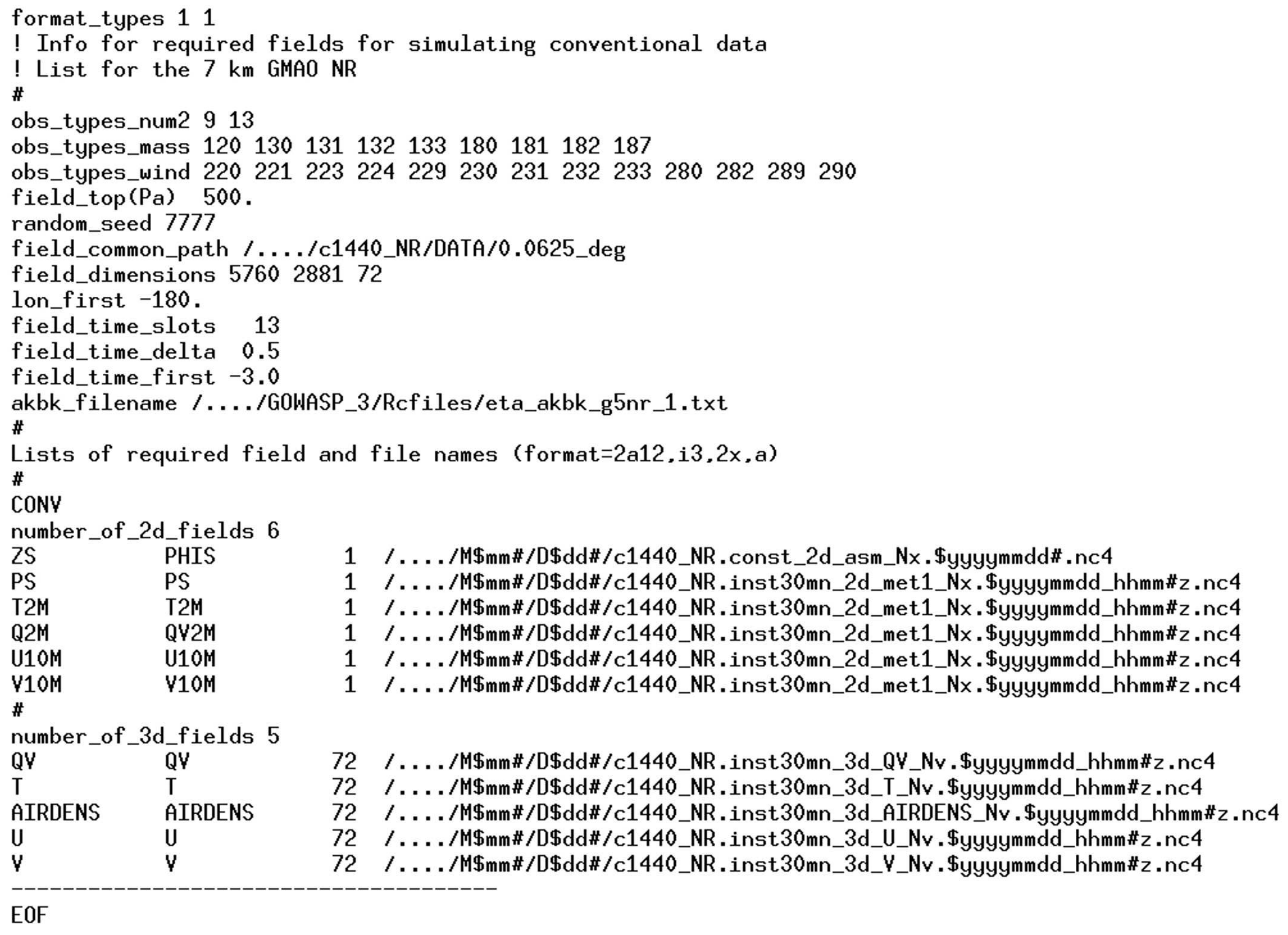
This first line is followed by up to 20 lines of arbitrary length containing user–provided comments. These are simply appended optional notes to the file that will not affect the reading of any required or applied information. The final comment line must begin with the character #.

Further information in the file may refer to all its intended applications or only to specific observation classes. In particular, differing radiance instrument classes may require their own specific instructions, as will be described. Following the header comments, however, are lines specifying values for variables that are common to all observation classes. These may be in any order but each must begin with the character name for that variable and spaces must occur between each value provided on a line. If a required variable is absent, the application will either terminate or supply a default value that will be indicated in the script's log file. This section of the resource file ends with a line having # as the first character.

Three examples of resource files used for creating observations are shown in figures: Fig. 5.1 displays an example of the file ***field_list_conv.rc***; Fig. 5.2 displays an example of the file ***rad_thin.rc***; and Fig. 5.3 displays an example of the file ***rad_prob.rc***. In the figures pertaining to radiances, only a few of the radiance types are shown so that each figure can be fit on a single page. Otherwise such files generally contain the entire list of radiance observation types.

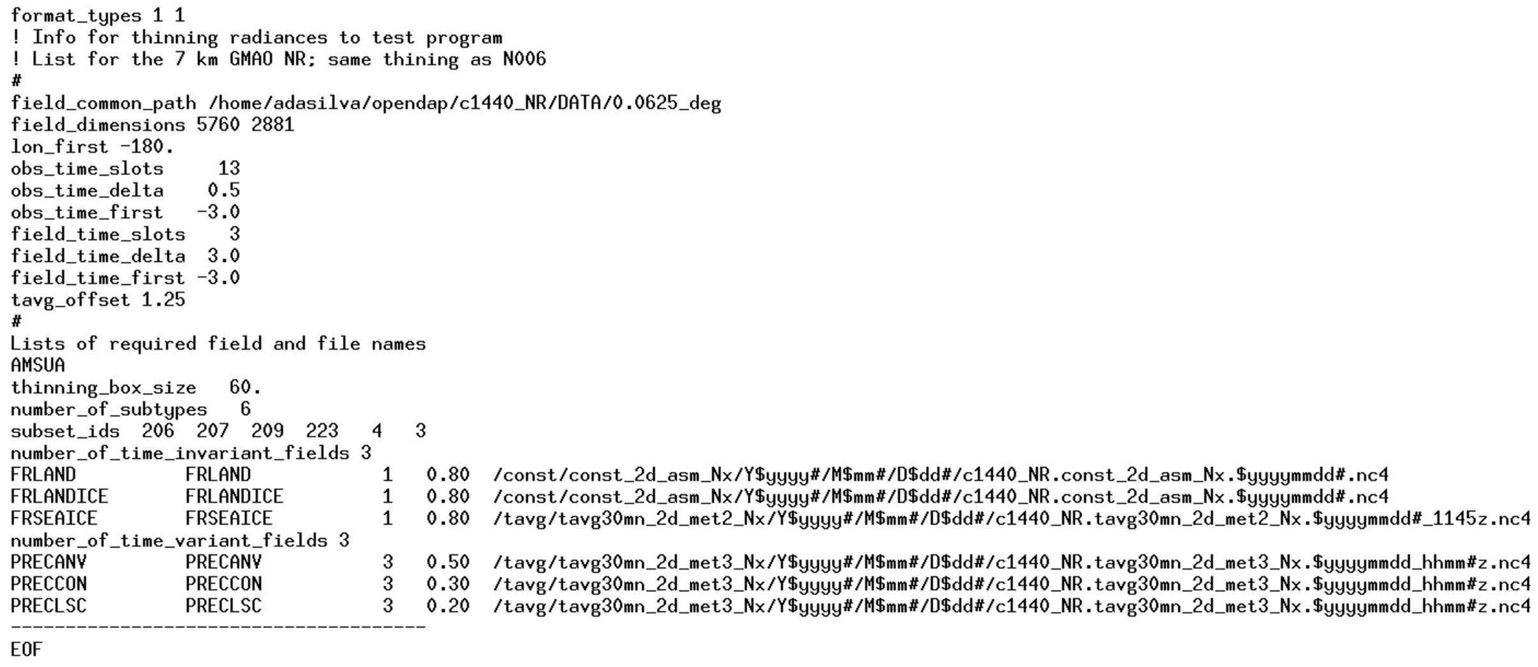
Variables that may appear in this common set are:

- **field_common_path**: This is a character variable of length 1–240 providing the portion of the directory path name that is common to all the NR fields to be read in an application. There is no default value.
- **field_dimensions**: This requires 3 integers denoting the numbers of grid points in longitudinal and latitudinal directions and the number of atmospheric levels for 3–d fields, listed in that order. These must agree with the field dimensions recorded in the files of NR fields.
- **lon_first**: This is a required real variable that denotes the longitude of the first point in each array of NR field values to be read. It is assumed that the same mapping applies to all 2–D and 3–D field arrays.



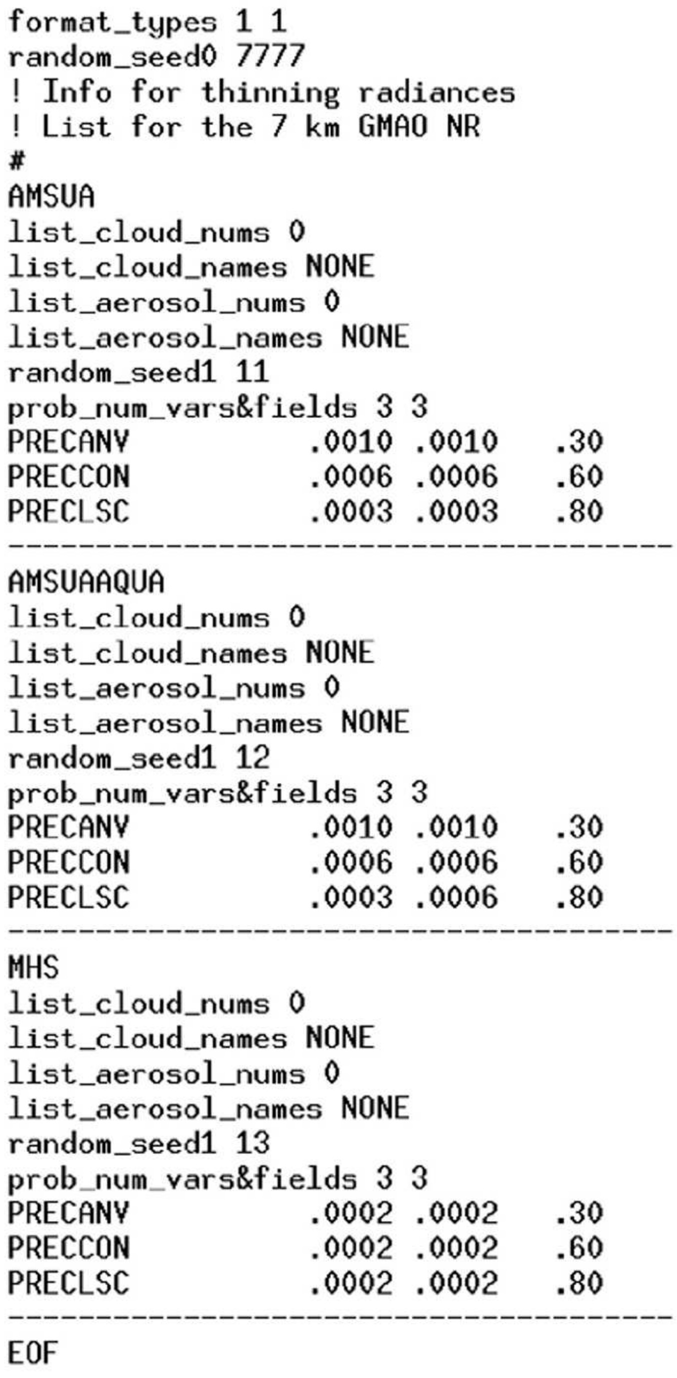
```
format_types 1 1
! Info for required fields for simulating conventional data
! List for the 7 km GMAO NR
#
obs_types_num2 9 13
obs_types_mass 120 130 131 132 133 180 181 182 187
obs_types_wind 220 221 223 224 229 230 231 232 233 280 282 289 290
field_top(Pa)  500.
random_seed 7777
field_common_path /..../c1440_NR/DATA/0.0625_deg
field_dimensions 5760 2881 72
lon_first -180.
field_time_slots   13
field_time_delta  0.5
field_time_first -3.0
akbk_filename /..../GOWASP_3/Rcfiles/eta_akbk_g5nr_1.txt
#
Lists of required field and file names (format=2a12,i3,2x,a)
#
CONV
number_of_2d_fields 6
ZS          PHIS          1  /..../M$mm#/D$dd#/c1440_NR.const_2d_asm_Nx.$yyyymmdd#.nc4
PS          PS            1  /..../M$mm#/D$dd#/c1440_NR.inst30mn_2d_met1_Nx.$yyyymmdd_hhmm#z.nc4
T2M         T2M           1  /..../M$mm#/D$dd#/c1440_NR.inst30mn_2d_met1_Nx.$yyyymmdd_hhmm#z.nc4
Q2M         QV2M          1  /..../M$mm#/D$dd#/c1440_NR.inst30mn_2d_met1_Nx.$yyyymmdd_hhmm#z.nc4
U10M        U10M          1  /..../M$mm#/D$dd#/c1440_NR.inst30mn_2d_met1_Nx.$yyyymmdd_hhmm#z.nc4
V10M        V10M          1  /..../M$mm#/D$dd#/c1440_NR.inst30mn_2d_met1_Nx.$yyyymmdd_hhmm#z.nc4
#
number_of_3d_fields 5
QV          QV           72  /..../M$mm#/D$dd#/c1440_NR.inst30mn_3d_QV_Nv.$yyyymmdd_hhmm#z.nc4
T           T            72  /..../M$mm#/D$dd#/c1440_NR.inst30mn_3d_T_Nv.$yyyymmdd_hhmm#z.nc4
AIRDENS     AIRDENS      72  /..../M$mm#/D$dd#/c1440_NR.inst30mn_3d_AIRDENS_Nv.$yyyymmdd_hhmm#z.nc4
U           U            72  /..../M$mm#/D$dd#/c1440_NR.inst30mn_3d_U_Nv.$yyyymmdd_hhmm#z.nc4
V           V            72  /..../M$mm#/D$dd#/c1440_NR.inst30mn_3d_V_Nv.$yyyymmdd_hhmm#z.nc4
--------------------------------------
EOF
```

Figure 5.1: Example of a ***field_list_conv.rc*** file. The file name has been abbreviated using ellipsis in order to fit names on the page.



```
format_types 1 1
! Info for thinning radiances to test program
! List for the 7 km GMAO NR; same thining as N006
#
field_common_path /home/adasilva/opendap/c1440_NR/DATA/0.0625_deg
field_dimensions 5760 2881
lon_first -180.
obs_time_slots     13
obs_time_delta    0.5
obs_time_first   -3.0
field_time_slots    3
field_time_delta  3.0
field_time_first -3.0
tavg_offset 1.25
#
Lists of required field and file names
AMSUA
thinning_box_size   60.
number_of_subtypes   6
subset_ids  206  207  209  223   4   3
number_of_time_invariant_fields 3
FRLAND          FRLAND            1   0.80  /const/const_2d_asm_Nx/Y$yyyy#/M$mm#/D$dd#/c1440_NR.const_2d_asm_Nx.$yyyymmdd#.nc4
FRLANDICE       FRLANDICE         1   0.80  /const/const_2d_asm_Nx/Y$yyyy#/M$mm#/D$dd#/c1440_NR.const_2d_asm_Nx.$yyyymmdd#.nc4
FRSEAICE        FRSEAICE          1   0.80  /tavg/tavg30mn_2d_met2_Nx/Y$yyyy#/M$mm#/D$dd#/c1440_NR.tavg30mn_2d_met2_Nx.$yyyymmdd#_1145z.nc4
number_of_time_variant_fields 3
PRECANV         PRECANV           3   0.50  /tavg/tavg30mn_2d_met3_Nx/Y$yyyy#/M$mm#/D$dd#/c1440_NR.tavg30mn_2d_met3_Nx.$yyyymmdd_hhmm#z.nc4
PRECCON         PRECCON           3   0.30  /tavg/tavg30mn_2d_met3_Nx/Y$yyyy#/M$mm#/D$dd#/c1440_NR.tavg30mn_2d_met3_Nx.$yyyymmdd_hhmm#z.nc4
PRECLSC         PRECLSC           3   0.20  /tavg/tavg30mn_2d_met3_Nx/Y$yyyy#/M$mm#/D$dd#/c1440_NR.tavg30mn_2d_met3_Nx.$yyyymmdd_hhmm#z.nc4
--------------------------------------
EOF
```

Figure 5.2: Example of a ***rad_thin.rc*** file showing an abbreviated list of observation types.



```
format_types 1 1
random_seed0 7777
! Info for thinning radiances
! List for the 7 km GMAO NR
#
AMSUA
list_cloud_nums 0
list_cloud_names NONE
list_aerosol_nums 0
list_aerosol_names NONE
random_seed1 11
prob_num_vars&fields 3 3
PRECANV          .0010 .0010   .30
PRECCON          .0006 .0006   .60
PRECLSC          .0003 .0003   .80
------------------------------------
AMSUAAQUA
list_cloud_nums 0
list_cloud_names NONE
list_aerosol_nums 0
list_aerosol_names NONE
random_seed1 12
prob_num_vars&fields 3 3
PRECANV          .0010 .0010   .30
PRECCON          .0006 .0006   .60
PRECLSC          .0003 .0006   .80
------------------------------------
MHS
list_cloud_nums 0
list_cloud_names NONE
list_aerosol_nums 0
list_aerosol_names NONE
random_seed1 13
prob_num_vars&fields 3 3
PRECANV          .0002 .0002   .30
PRECCON          .0002 .0002   .60
PRECLSC          .0002 .0002   .80
------------------------------------
EOF
```

Figure 5.3: Example of a ***rad_prob.rc*** file showing an abbreviated list of observation types.

- **field_top(Pa)**: For conventional and AMV (SATWIND) observations, this instructs that 3–d NR fields are not to be read for data levels above this pressure level. This reduces required memory for storing fields since observations of these classes will always be below some level. Default for this value is that for the highest NR 3–D field level. Otherwise this is a real–valued variable in units of Pascals.

- **akbk_filename**: Full path and name of the text file containing values of $a_{k+\frac{1}{2}}$ and $b_{k+\frac{1}{2}}$ that define the vertical η–coordinates at NR field layer interfaces.

- **random_seed**: For any application that uses generated random numbers, this is one of the integers used to set the seed. This value will be used for all observation subtypes in this application. It is intended to allow the seed to be simply changed for differing experiments if desired. If no value is provided, the value 1111 is set as a default.

- **field_stride_ij**: This is followed by two integers that denote that only every i–th longitude and j–th latitude should be considered when simulating observations. This is only used for applications where the observation locations are not provided a–priori, specifically for ***create_gp_raobs*** and ***create_satwind***.

- **obs_dtypes**: This is used only by ***create_satwind***. It is a single character of length 4 that contains only the letters T or F. The first is T if observations are to be simulated for geostationary satellites. The second is T if observations are to be simulated for polar orbiters. The third is T if observations are to be simulated for VIS/IR images. The fourth is T if observations are to be simulated for water vapor images. Otherwise the values should be F. The default value is "TTTT" indicating that all SATWIND subtypes are to be simulated.

The following three variables are only required if the application is ***create_conv***:

- **obs_types_num2**: This is followed by two integers that denote the number of conventional observation subtypes, in terms of distinct **kx** values that should be simulated by ***create_conv***. The first integer denotes the number of "mass" (i.e., T, q, or p_s) subtypes ($100 \leq \mathbf{kx} \leq 199$), the second of wind subtypes ($200 \leq \mathbf{kx} \leq 299$).

- **obs_types_mass**: The set of **kx** values (i.e., GSI indexes) for "mass" subtypes to be simulated. If the number of mass observation subtypes was previously set to 0, this variable name should still be provided but the list of values following it can be absent.

- **obs_types_wind**: The set of **kx** values (i.e., GSI indexes) for “wind” subtypes to be simulated. If the number of wind observation subtypes was previously set to 0, this variable name should still be provided but the list of values following it can be absent.

The selection of NR file times to be read and used is determined by specifying some parameters that also appear in this common list:

- **field_time_slots**: This integer indicates the number of NR field times to be read for simulating observations within a single DAS analysis period.
- **field_time_delta**: This is the number of hours (a real positive number) between NR field times to be read.
- **field_time_first**: This denotes the first time to be read (a real number in units of hours) relative to the synoptic time for the data assimilation period. This is generally -3.0 for data assimilation periods 6 hours long.
- **tavg_offset**: For the G5NR field data sets, some fields are time averaged. The time stamp in their file names reflects this by indicating the time at the center of the averaging period. The software needs to know if such a time offset is required to determine the file names for such fields. If the offset is to be used for a given file, this number of hours (real–valued) will be added to the NR data time requested.

If t_s denotes the synoptic time that is usually the middle of the data assimilation period, then field times

$$t_n = t_s + (\textbf{field\_time\_first}) + (n-1)(\textbf{field\_time\_delta})$$

will be read for $n = 1, \ldots, (\textbf{field\_time\_slots})$. Generally, fields for pairs of time t_n, t_{n+1} are read together so that both are simultaneously available in memory for time interpolation between them. Such a time pair is called a “time slot” in the GOWASP–3 software. (Using this definition, **field_time_slots** actually denotes 1 plus the number of time slots in an analysis period.) Files of required NR fields for all requested t_n must be available.

For fields that are denoted as time mean fields in the resource file, no temporal interpolation will be performed. Instead, the NR field data used for each time slot will be chosen from the pair of times

$$t_n^a = t_n + (\textbf{tavg\_offset}) \tag{5.2.1}$$

$$t_n^b = t_{n+1} - (\textbf{tavg\_offset}) \tag{5.2.2}$$

for each $n = 1, \ldots, (\textbf{field\_time\_slots}) - 1$. For some time offsets, t_n^a and t_n^b may be the same time. If not, the file for the field time closest to the observation time will be used. Data files for all the referenced times must be available.

If **field_time_delta** $= 0.5$ hours is specified, (**tavg_offset**) $= 0.25$ hours will refer to existing times for time–averaged G5NR fields, with $t_n^a = t_n^b$. For **field_time_delta** $= 3$ hours, however, $t_n^a = t_n^b$ would require (**tavg_offset**) $= 1.5$ hours, and this would not refer to any existing time for a time–averaged G5NR field. In the latter case, some other time–offset would be required, resulting in two distinct field times being considered within each time slot.

The file times to be used to extract NR field profiles for radiance calculations are not specified in the resource file read by ***create_rad_prof*** since, for computational efficiency, the observation location information must be sorted into time slots prior to its execution. This sorting is therefore specified in the resource file used by ***create_rad_list*** using the three variables

- **obs_time_slots**: Like **field_time_slots** except referring to the times to be used in the application ***create_rad_prof***.
- **obs_time_delta**: Like **field_time_delta** except referring to the times to be used in the application ***create_rad_prof***.
- **obs_time_first**: Like **field_time_first** except referring to the times to be used in the application ***create_rad_prof***.

These **obs_time** values will be passed to the ***create_rad_prof*** application in the header within the binary output created by ***create_rad_obs_list***. These values may differ from the corresponding **field_time** values used in ***create_rad_obs_list***. All files that will thereby be referenced must exist. A suitable value of the variable **tavg_offset** must be specified in the resource file read by ***create_rad_prof***.

The final section of the resource file contains information specific to distinct observation classes or instructions for obtaining specific NR fields. Information for each class ends with a line beginning with 3 or more successive characters "-" and begins with the name for an allowed observation class, as indicated in the application scripts. The format for this information is as follows:

- The first variable is a character of length 1–16 that is the name of the field in the GOWASP–3 software.
- The second variable is a character of length 1–16 that is the name of the corresponding field in the NR file.
- The next 1 or 2 variables are numbers that are either (a) an integer denoting the number of levels for the field to be read, (b) an integer denoting how the field is to be interpolated, or (c) a weight denoting how the field should be used to create a penalty for performing data thinning. Which are required depends on the application. The choices for (b) are values equal to

  - 1: interpolate both horizontally and temporally;
  - 2: interpolate temporally but not horizontally;
  - 3: interpolate horizontally but not temporally;
  - 4: interpolate neither horizontally nor temporally.

  If no interpolation is performed, the nearest NR grid point and field time is used.
- The last variable is a template for the NR file name that contains the field to be read.

The list of file name templates is divided into two sets. Except for the application ***create_rad_list*** the 2–d and 3–d fields are listed separately. Each of the separate lists is preceded by a line that states "number of ?d fields" followed by an integer indicating the number of fields in the set. For ***create_rad_list***, the two sets are "number_of_time_invariant_fields" and "number_of_time_variant_fields".

The file template provided here may contain fixed or variable characters. Only the part of the file name that is not common to other NR files should be specified here. (See description of the variable **field_common_path**.) The variable characters are denoted by a $ sign followed by a label followed by a # sign. When a specific file name is created, the beginning and ending signs and label will be replaced by a computed value, such as a time stamp. The list of labels recognized by the GOWASP–3 software is:

- **ffff**: denotes the part of the file name that specifies the name of the field.
- **yyyy**: denotes the part of the file name that specifies the 4–digit year.
- **yy**: denotes the part of the file name that specifies the final 2–digits of the year.
- **mm**: denotes the part of the file name that specifies the 2–digit month, with any leading 0 required.
- **dd**: denotes the part of the file name that specifies the 2–digit day, with any leading 0 required.
- **hh**: denotes the part of the file name that specifies the 2–digit hour (00 – 23), with any leading 0 required.
- **hhmm**: denotes the part of the file name that specifies the 2–digit hour followed by 2–digit minute (00 – 59).
- **yyyymmdd**: denotes the part of the file name that specifies a concatanated value for **yyyy**, **mm** and **dd**.
- **yyyymmdd_hh**: denotes the part of the file name that specifies a concatanated value for **yyyymmdd** with an underscore followed by a value for **hh**.
- **yyyymmdd_hhmm**: denotes the part of the file name that specifies a concatanated value for **yyyymmdd_hh** with a 2–digit minute (00 – 59).

For application ***create_rad_list***, for each distinct observation type, there are three additional variables that must be specified. These are:

- **thinning_box_size**: This denotes the real–valued width of a thinning box in units of km, similar to the specification in G5DAS. A value of zero means no thinning will be performed.
- **number_of_subtypes**: An integer greater than 0 indicating the number of observation subtypes to be simulated as denoted in the following variable.
- **subset_ids**: A list of **number_of_subtypes** integers separated by spaces, indicating the specific observation subtypes to be simulated. For most observation types, these refer to satellite–platform WMO identification numbers. For AIRS and AMSUAAQUA these refer to the instrument identification number.

5.2.1 Resource file for ***create_rad_bufr***

The resource file for application ***create_rad_bufr*** differs from that of the others. No NR files are read for this application so no NR descriptive information is supplied. After the format specification and comment lines, there are the separate variable sets for each radiance observation class. This is followed by

- **list_cloud_nums**: A single integer denoting the number of NR hydrometeor fields for which scattering is to be considered by the radiative transform operator (CRTM). If 0, no scattering by hydrometeors will be considered.
- **list_cloud_names**: A list of **list_cloud_nums** character field names separated by blanks. If **list_cloud_nums** is 0, the name 'NONE" should be specified. Any fields listed here must be included in the input profile data file; otherwise the application will abort.
- **list_aerosol_nums**: Like **list_cloud_nums**, except for aerosols. Use of these is not yet implemented in the CRTM version used by GOWASP–3.
- **list_aerosol_names**: Like **list_cloud_names**, except for aerosols. Use of these is not yet implemented in the CRTM version used by GOWASP–3.
- **random_seed1**: This is an integer that provides a portion of the random seed that depends on observation class.
- **prob_num_vars&fields**: This is followed by 2 integers. The first denotes the number J of named fields to be used when determining a probability a radiance is to be affected by clouds or precipitation. The second denotes the number of parameters to be read for each such field that will be used to determine probabilities associated with the given field. This latter must be 3 in the present version of GOWASP–3.

The list of J field names that follow each line of **prob_num_vars&fields** are the fields from which the probabilities of radiances being affected are determined. ($J = 3$ in the example in Chapter 3.) The 3 real values on each line are respective values of a_j, b_j, σ_j associated with this specific field. Order of the list of fields is important since each field is considered in succession, conditioned on results obtained for the previous field.

5.2.2 Resource file *sat_info.rc*

The resource file ***sat_info.rc*** is used to relate or define various parameters, names, and indices associated with distinct radiance instruments on distinct satellite platforms. It is used by ***create_rad_bufr***, ***create_error***, and several programs used for tuning observation errors. All table values are not used by all instrument and platform pairs. An example of the file ***sat_info.rc*** is displayed in Fig. 5.4.

Default values are defined as place holders for table values that are not actually used by any of the programs.

- **dtype**: This is the observation data type as used by the GOWASP–3. Only those data types may be used. New radiance data types should be specified as GENRADTXT. Notice that all types are currently in upper case.
- **platform**: This is the satellite name as it appears in the GOWASP–3 files of satellite data error standard deviations.
- **instr**: The instrument name. This should be the same as used to define the CRTM spectral coefficient file names and the file names in the G5DAS observation diagnostics files.
- **sat**: The satellite name. This should be the same as used to define the CRTM spectral coefficient file names and the file names in the G5DAS observation diagnostics files.
- **spec_file_name**: This is the instrument name that is used to build part of the name for the CRTM file of spectral coefficients. It is required, except for class GENRADTXT for which its value is provided in the observation data text file itself.
- **nchan**: The number of channels for this instrument in the BUFR or GENRADTXT files. When those files are used, the number of channels is actually obtained from them, but for the radiance error tuning programs, it is obtained from this resource file. For these, it should correspond to the numbers in the corresponding BUFR files, that may include only a subset of the complete set of instrument channels (e.g., 281 for AIRS channels currently utilized by G5DAS).
- **siid**: The integer identification number for this instrument corresponding to the BUFR value SIID. This is not required for data type GENRADTXT.

```
dtype            platform         instr            sat              spec_file_name       nchan siid  said  ifov0 fovstep    fovstart   fovrato    fovadjust
AIRS             AQUA             airs             aqua             airs281_aqua           281   420   784    45  1.1       -48.9             1.0         0.
AMSUA            N15              amsua            n15              amsua_n15               15   570   206    15  3.3333333 -48.333333        1.0         0.
AMSUA            N16              amsua            n16              amsua_n16               15   570   207    15  3.3333333 -48.333333        1.0         0.
AMSUA            N17              amsua            n17              amsua_n17               15   570   208    15  3.3333333 -48.333333        1.0         0.
AMSUA            N18              amsua            n18              amsua_n18               15   570   209    15  3.3333333 -48.333333        1.0         0.
AMSUA            N19              amsua            n19              amsua_n19               15   570   223    15  3.3333333 -48.333333        1.0         0.
AMSUA            METOP-A          amsua            metop-a          amsua_metop-a           15   570     4    15  3.3333333 -48.333333        1.0         0.
AMSUA            METOP-A          amsua            metop-b          amsua_metop-b           15   570     3    15  3.3333333 -48.333333        1.0         0.
AMSUAAQUA        AQUA             amsua            aqua             amsua_aqua              15   570   784    15  3.3333333 -48.333333        1.0         0.
ATMS             NPP              atms             npp              atms_npp                22   621   224    48      1.110    -52.725 1.13639870         0.
CRIS             NPP              cris             npp              cris_npp               399   620   224    15       999.       999. 0.87997285         0.
GMI              GPM              gmi              gpm              gmi_gpm                 13   519   288    -1       999.       48.5        1.0         0.
HIRS4            N18              hirs4            n18              hirs4_n18               20   607   209    28       1.80      -49.5 1.13639870         0.
HIRS4            N19              hirs4            n19              hirs4_n19               20   607   223    28       1.80      -49.5 1.13639870         0.
HIRS4            METOP-A          hirs4            metop-a          hirs4_metop-a           20   607   4      28       1.80      -49.5 1.13639870         0.
HIRS4            METOP-B          hirs4            metop-b          hirs4_metop-b           20   607   3      28       1.80      -49.5 1.13639870         0.
IASI             METOP-A          iasi             metop-a          iasi616_metop-a        616   221   4      60      3.334     -48.33        1.0      0.625
IASI             METOP-B          iasi             metop-b          iasi616_metop-b        616   221   3      60      3.334     -48.33        1.0      0.625
MHS              N18              mhs              n18              mhs_n18                  5   203   209    45  1.1111111 -49.444444        1.0         0.
MHS              N19              mhs              n19              mhs_n19                  5   203   223    45  1.1111111 -49.444444        1.0         0.
MHS              METOP-A          mhs              metop-a          mhs_metop-a              5   203   4      45  1.1111111 -49.444444        1.0         0.
MHS              METOP-B          mhs              metop-b          mhs_metop-b              5   203   3      45  1.1111111 -49.444444        1.0         0.
SSMIS            F17              ssmis            f17              ssmis_f17               24   908   285    -1         0.       53.0        1.0         0.
GENRADTXT        N11              msu              n11              msu_n11                  4     1   100    -1       999.       999.        1.0         0.
EOF              X                X                X                X                        0     0     0     0         0.         0.         0.         0.
```

Figure 5.4: Example of a ***sat_info.rc*** file.

- **said**: The integer identification number for this satellite corresponding to the BUFR or GENRADTXT value SAID.
- **ifov0**: This is the field of view index corresponding to a near nadir scan position. A value less than 0 denotes an instrument with circular or no scan, or one for which all the required observation geometry is explicitly provided in the observation data files, with no other calculation or parameters required.
- **fovstep**: This is the angular viewing separation (degrees) between scan positions.
- **fovstart**: This is the first viewing angle (degrees) in the instrument scan.
- **fovrato**: This is either a ratio of mean distances between the satellite and earth's center and between mean sea level and earth's center, or the reciprocal of that ratio, depending on the observation class (compare its use with classes HIRS2, MSU, and CRIS). Default value is 1.
- **fovadjust**: This provides an adjustment required for specifying the starting angle in a scan. Default value is 0.

For further details of the use of these parameters, the module ***m_crtm_interface*** should be consulted. For observation data class **GENRADTXT**, the viewing geometry is completely specified in its observation data files.

5.2.3 Required fields for various applications

The following list presents required fields for indicated applications. For almost all of them, the field names in the program match corresponding ones in the G5NR files. For the exceptions, the name in the G5NR file appears in parentheses following its corresponding name in the GOWASP–3 software. Units of the field values must be those in the G5NR. Order does matter for these except for separation into sets of 2-D and 3–D fields, and for ***create_rad_list***, into time invariant and time varying fields.

- For ***create_conv***: ZS (PHIS), PS, T2M, QV2M. U10M, V10M, QV, T, AIRDENS, U, V.
- For ***create_gpsro***: PHIS, PS, QV, T.

- For ***create_satwind***: FROCEAN, PHIS, PS, QV, CLDFR (CLOUD), Z, U, V.

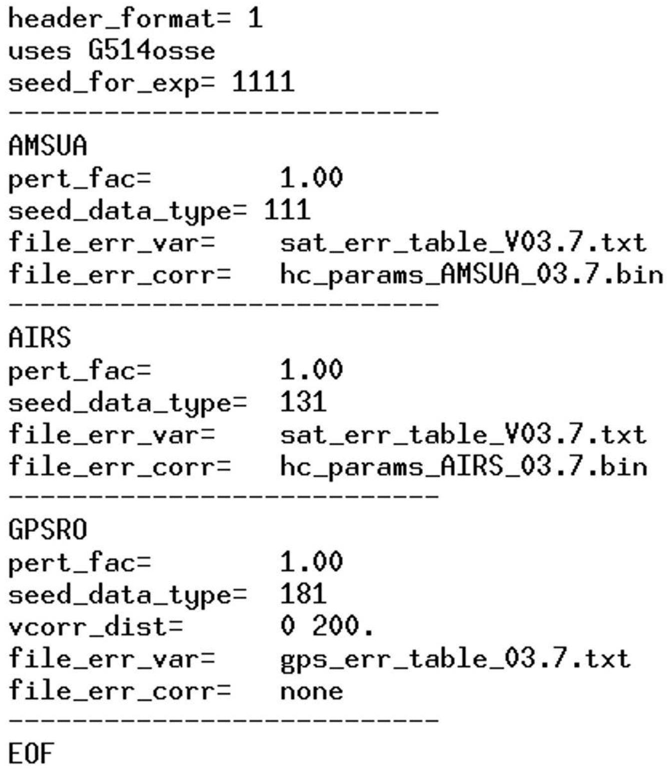
For radiances, all fields required by the CRTM must be provided. These are FRLAND, FRLANDICE, FRSEAICE, TS, PS, U10M, V10M, T, QV. Optionally, the fields vegtype, vegfrac, SNODP, TPSNOW, SFMC, FRSNO, O3, QL, QR, QS, QI may also be used. If vegtype is not provided, the land surface will be considered grassland for radiance calculations. If vegfrac is not provided, the fraction of land covered by vegetation will be set to 0.9. It is not necessary that all the 2–D fields be provided in the resource file list for ***create_rad_prof*** since values for any fields appearing in the resource file used by ***create_rad_list*** will be passed to ***create_rad_prof***. No fields are read in ***create_rad_bufr***. For ***create_rad_list***, the list of fields that must be included are only those that are to be used to thin the set of simulated observations compared to the number in the input real–data BUFR files. These must be 2–D fields.

5.2.4 Resource file for ***create_error***

The resource file for ***create_error*** differs from the others. It begins with a format line and optional comment lines as for all other resource files. Then, there is one line with the one variable **seed_for_exp** that requires a single integer for setting a part of the random seed that will be common for all observation data types and times. This is followed by a line beginning with three dashes that acts as a spacer. An example of the file ***error.rc*** is displayed in Fig. 5.5.

Next are separate sections for each observation class. Each begins with a valid **datatype** name and ends with a line beginning with 3 dashes. For each class, the following variables require specification:

- **pert_fac**= a required real value (ϵ_f) that provides a common multiplier to all errors produced for this data class. Generally, these errors are determined from random numbers, coefficients, or fields, whose statistics depend on values provided in error tables and other files that may be functions of pressure, height, channel, or data subtype. This variable allows the sizes of these errors to change without altering the correlations or ratios of variances between levels or channels. Typically, this value is 1, indicating that the standard deviations read from the error tables are used without modification.
- **seed_data_type**= an integer specifying the part of the random seed that varies with observation class.



```
header_format= 1
uses G514osse
seed_for_exp= 1111
--------------------------
AMSUA
pert_fac=        1.00
seed_data_type= 111
file_err_var=    sat_err_table_V03.7.txt
file_err_corr=   hc_params_AMSUA_03.7.bin
--------------------------
AIRS
pert_fac=        1.00
seed_data_type=  131
file_err_var=    sat_err_table_V03.7.txt
file_err_corr=   hc_params_AIRS_03.7.bin
--------------------------
GPSRO
pert_fac=        1.00
seed_data_type=  181
vcorr_dist=      0 200.
file_err_var=    gps_err_table_03.7.txt
file_err_corr=   none
--------------------------
EOF
```

Figure 5.5: Example of an ***error.rc*** file showing an abbreviated list of observation types.

- **file_err_var**= the required name of the file that contains the standard deviations of observation errors as functions of observation subtype and channel or level. This file must be in the same directory as this resource file.
- **file_err_corr**= an optional binary file that contains horizontal correlation parameters and possibly channel or level correlation matrices. If no such correlations are to be applied, this name should be set to “none”.

For GPSRO or CONV observation types, the variable **vcorr_dist** is also required. The first integer following the equal sign denotes the number of real values to follow next. For GPSRO this integer is 1, and the next value is a single, real–valued, vertical correlation length that denotes the distance (in meters) over which a Gaussian–shaped correlation falls to a value 0.1. For CONV, this integer is 4, followed by real–valued parameters from which vertical correlations of T, q, u and v will be determined. These parameters denote the ratio of any two pressure levels for which the correlation of respective errors is designed to be 0.1 assuming that the correlation shape is Gaussian in $\log(p)$.

5.3 Script Variables and Program Arguments

Most of the scripts and programs require that 2 times are specified. One denotes the first simulated, synoptic analysis period in the format yyyymmddhh. This is used to determine the times of the NR fields to be used. In the G5DAS, this refers to the center time of an analysis window. The other time denotes the time for corresponding real observational data that is used to determine simulated observation distributions in time and space. It is specified using the same format. Both times are incremented by the same number of hours (6 for G5DAS) as observations for successive analysis periods are simulated.

Output will be generated for the number of analysis periods specified by the UNIX variable **ntimes**. All the GOWASP–3 applications consider only one analysis period per execution. Data for multiple periods are obtained by embedding the executable in a UNIX loop.

Most of the GOWASP–3 applications also require an argument **test_print**. In production mode, this should be set to “F” to limit the amount of information printed to the log file. The alternative “T” indicates that much intermediary information will be printed to the log file, often including samples of the observation headers and values produced. While the application software was being tested, this information was extremely useful, so rather than removing their print statements, the ability to reactivate them has been retained by employing this switch. In particular **test_print**= “T” can be set to compare results of previous and modified or ported software.

For the radiance applications, the class of radiance observation must be specified. Generally, this refers to the instrument. The one exception is AMSUA on the AQUA satellite that has its own BUFR format and file at the GMAO. The choices of instrument class are currently: AIRS, AMSUAAQUA, AMSUA, HIRS4, MHS, IASI, ATMS, GMI, CRIS, and SSMIS. Note that they all are specified in upper case. All the applications consider only one of these types per execution, except for ***create_rad_prof*** that, if **instr_in1**= “T” will consider all types in a single execution. This option is computationally more efficient because all NR files are not then read repeatedly for each observation class.

In ***create_satwind***, the argument **count_file** is the name of a text file to which summary tables output to the log file are copied. These tables denote the numbers of simulated observations produced for indicated SATWIND subtypes and pressure layers. This makes it easier to pass these tables to other software for determining time–mean diagnostic distributions. If no copy is required, the user should specify the value “none”.

In ***create_gp_raobs***, the variable **dtype** must be set to one of GPWIND, GPMASS or GPBOTH. The first indicates that only radiosonde wind observations will be simulated at some selected spacing of NR grid points over the entire globe, the second only mass (i.e., T, q, and p_s) observations, and the third, both wind and mass observations.

5.4 Other Input Files

Aside from binary files used to pass data between various ***create_rad*** applications, the only other input files required are for ***create_error***. These include text files of tables of standard deviation values and binary files containing parameters for defining various error correlations.

5.4.1 Tables of error standard deviations

The table of standard deviations used to produce errors for CONV, SATWIND, and PREPBUFR data types was originally derived from the G5DAS prepobs_errtable.global file. Standard deviation values, however, may have been changed during the process of tuning the simulated errors. The format begins with an index 100 ≤**kx**≤ 299 denoting the observation subtype. For each subtype, 33 lines of values follow. The first value in each line is the pressure (mb) at which the error standard deviations on that line apply. The remaining 5 columns are respectively for T (K), relative humidity (fraction), wind (ms^{-1}), ps (mb), and a field not referenced. If values are not referenced for a particular subtype, they are set to 0.1×10^{10}. Actual standard deviations used by GOWASP–3 are interpolated from these p–levels to the observation levels. Note that the values in this table are only used to simulate the errors but generally not to specify what the G5DAS assumes for those errors. Also note that these standard deviations will be further multiplied by the value of **pert_fac** for data type CONV appearing in the ***create_error*** resource file.

The table of standard deviations used to produce errors for GPSRO is provided in its own file. This begins with a line with the single value 1, indicating there is only 1 subtype considered. This is then followed by 100 lines of values, numbered from 1 to 100. After each line index, there are 2 values. The first refers to the GPSRO "impact parameter" that is approximately an observation height (m) measured from earth's center. The second is a factor used to determine error. These factors are interpolated from the given levels to the observation levels. The error standard deviation used is the product of such an interpolated factor and the simulated observation value. Also note that these standard deviations will be further multiplied by the value of **pert_fac** for data type GPSRO appearing in the ***create_error*** resource file.

The table of standard deviations for radiance observations as a function of channel, satellite, and instrument appear in a table formatted similarly to an earlier version of the ***gmao_global_satinfo.rc*** file used by the G5DAS. The only values that are used by ***create_error*** are the number of header (i.e., comment) lines and groups (total number of radiance subtypes, N_g) in line 1, the successive lines of group indexes $1, \ldots, N_g$, the header line for each group containing the instrument name, satellite platform, and number of channels (starting in columns 2, 19, and 38, respectively), and the standard deviations of the errors to be added, in units of brightness temperature (K). Other values in the table are not relevant.

5.4.2 Parameters for error correlations

Error vertical–correlation parameters for radiosondes, dropsondes, and GPSRO bending angles are specified in the resource file for ***create_error***. This is appropriate because these observations do not occur at fixed levels and the correlations are determined separately as each observation report is considered. In contrast, for channel correlations or combined horizontal and vertical correlations, the required matrices determined from the correlations are best pre–computed for prescribed levels or channels. These are then passed to ***create_error*** through a binary file.

Format details of this binary file are not described in this documentation. Interested users should consult the subroutine ***error_table_read_corr_params*** in the module ***m_obs_error_table*** and chapter 7 for such details or simply use some binary file already prepared. Separate files are required for each observation data type since consistency will be checked with the argument supplied to the application.

The variables that are expected in this binary file are as follows:

- **itypes_corr**: This is the number of observation data subtypes for which separate sets of parameters are specified in the file. For PREPBUFR, CONV or SATWIND data, these are the numbers of sets of **kx** values that will use distinct random fields. For most radiance classes, these are the numbers of sets of satellite platform identification numbers that will use distinct random fields. For AIRS and AMSUAAQUA classes, **itypes_corr** equals 1, since there is only 1 subtype for each of these classes.

- **dtype_file**: This is the character name of a valid observation class recognized by the software. This must agree with the name provided to ***create_error*** as an argument. It essentially provides a means of checking that a valid binary file for this data class is being used.

- **file_format**: This is an integer denoting the format of this file, similar to the use of such a variable in other resource files. This is currently an unused variable.

- **version_number**: This is a character label denoting a version label assigned for the parameters in the file, such as an iteration number within the tuning process. It is used for reference only and is not used by any calculation.

- **c_info**: This is a character string providing comments within the file. It is used for reference only and is not used by any calculation.

- **c_file_orig**: This is a character string providing a name of a file from which parameters in this file have been derived. It is used for reference only and is not used by any calculation.
- **nlevels**: This is the number of pressure levels, channels, or principal components described by the random fields to be produced.
- **pmax,pmin**: This defines a range of pressures over which random fields will be produced.
- **l_vc_corr**: This is a logical flag denoting whether the file contains matrices describing correlations between either vertical levels or channels.
- **vlengths_i**: This is an array of values describing vertical correlation lengths (m). This is only used if random fields are to be both horizontally and vertically correlated. They therefore do not concern radiance errors. The length parameter here is the distance over which the correlation drops to $1/\sqrt{e}$. Separate lengths can be defined for separate sets of subtypes.
- **ks_list**: This is the list of **ks** or **kx** identification numbers denoting the observation subtypes in this particular subset.
- **rf_nmax**: This denotes the spectral truncation that will define the resolution of the random horizontal fields to be created. The spacing between latitudes for the grid of random fields will be approximately 180/(**rf_nmax**+1) degrees. In GOWASP–3, this truncation value should be less than 800 because for larger values, round–off errors in the computation of the Legendre polynomials are possible.
- **corr_shapes**: This denotes the particular shape of the horizontal correlation: GAUSS, EXP, TOAR, or WHITE for this subtype set.
- **frac_corr**: For each channel or pressure level, this denotes the fraction of error (specifically, ν in chapter 4) that is to be correlated. Separate sets of values are specified for each set of subtypes.
- **hcorr_lengths**: For each channel or pressure level, this denotes the horizontal correlation length parameter (L; units of km). Separate sets of values are specified for each set of subtypes. The meaning of this variable differs with the shape of correlation function selected. (See (4.3.25).)
- **cov_matrix**: This is the matrix of vertical level or channel correlations that will be used to create the correlated portion of errors.

- **e_values**: This is an array of eigenvalues of the corresponding covariance matrix.
- **e_vects**: This is an array of eigenvectors (principal components) of the corresponding covariance matrix.

Chapter 6

Explanation of Software

The GOWASP–3 software has been primarily designed for use by GMAO staff and collaborators. Earlier versions were intended for a broader community of users, but efforts to organize that community effectively failed. In the present version, therefore, portability and transparency were not a priority. Of course they were considered to some degree, however, since software maintenance is crucial.

The design of all complex software is affected by trade–offs in clarity, ease of maintenance, ease of modification, ease of use, computational efficiency, and varieties of applications and options offered. This does not imply that multiple goals or constraints can never be simultaneously satisfied, but rather that different priorities usually lead to very different designs. As an example, using a variable name like **field1** may make an instruction general, but then the field to which it actually refers will be unclear without following assignments and labels backward in execution. Using general names, however, can remove redundancy and facilitate software maintenance and ease its modification. For GOWASP–3 no formal software design priorities were established at inception. Instead, as the software was built and applied, it was continuously modified to improve efficiency, clarity, generality, or ease of maintenance.

GOWASP–3 is written using standard FORTRAN 90. Some applications use FORTRAN Message Passing Interface (MPI). Some use components of a version of the Earth System Modeling Framework (ESMF) employed at the GMAO, specifically components referenced by the Shared Memory (Shmem) application. These latter are tools designed to allow FORTRAN arrays to be distributed in memory across processors but to be referenced by simply specifying array indexes without issuing MPI commands or being concerned about on which processor an array element resides. This makes the software clearer, but its portability requires that a compatible version of ESMF be available on the host computer.

Use of Shmem requires that execution is conducted on a single computer node. The number of global, high–resolution, G5NR–1 3–D fields that can reside in RAM simultaneously is therefore limited. The GOWASP–3 software has been carefully designed to accommodate this limitation on computers available to GMAO researchers, as explained later in this chapter. For use with other computers or higher–resolution NR data sets, Shmem may need replacement by more explicit and general MPI commands.

Following a GMAO standard, FORTRAN source files are denoted by file type **.f90** unless they contain MPI instructions, in which case they are labeled as type **.F90**. Names of FORTRAN module source files all have prefix **m_**. All FORTRAN variable names have explicit type specifications. Most variables are stored as 32 bit variables. Notable exceptions are for some variables within the BUFR libraries and for some used within modules used by ***create_error*** to create horizontally–correlated random fields for which 64–bit arithmetic performs better.

The software consists of approximately 57,000 lines of FORTRAN, although a large percentage is in the form of comments. Few strict coding rules were followed, although some rules are apparent. These include naming conventions for public variables in modules and spacing rules in do–loop and if–block structures. Although it has been developed by scientists rather than programmers, clarity was required to facilitate maintenance and continue development.

Partial sequences of subroutine calls during GOWASP–3 program executions are listed in appendix A. These are intended to give a sense of the work flow within the programs. The names of the files containing each subroutine are also listed.

All the files containing program drivers, modules and their embedded subroutines, and subroutines external to modules are listed in appendix B. Also included there are brief descriptions of the function of each subroutine.

GOWASP–3 is not intended to be used as a “black box” or “push button” tool. Instead, users are expected to understand what the software does. Since years have been invested in its development, it is reasonable that users spend days in mastering its application.

6.1 Software Drivers

There are distinct program drivers and UNIX scripts for different applications. This separation of applications renders each program more clearly than if all were combined. Computational aspects that are common to several applications are placed in libraries.

For radiance simulation, there are 5 distinct drivers and UNIX (c–shell) scripts. These are separated to enhance computational efficiency. For example, if different observation types require different sets of named fields, multiple executions of ***create_rad_prof*** may be most efficient, but all would require the same results from ***create_rad_list***. Or, if the effects of clouds and precipitation on radiances are being tuned, requiring multiple executions of ***create_rad_bufr***, it is most efficient to execute ***create_rad_prof***, containing the expensive NR data reads, only once.

The reading of NR fields is directed within the drivers or modules called by drivers. In some drivers, the FORTRAN names for the read fields are generic (e.g., **field1** or **fld_1t1**) but for other drivers they may be specific (e.g., **fld_u** or **fld_z**). For the generic labels, there is generally a corresponding index to an array of field names that will distinguish specific named fields.

Since G5NR–1 has 13 data times within a 6–hour assimilation window, depending on the application, 39–91 3–D fields, in addition to several 2–D fields, need to be read for simulations within each window. Reading a single 3–D field at a single data time is computationally expensive (at the GMAO, typically 1 minute CPU on a single processor). The GOWASP–3 is designed to significantly but incompletely mitigate this expense. In particular, distinct 3–D fields are read using separate processors. Also, when drawing atmospheric profiles from the NR for later computation of radiances, pairs of times for each user–specified temporal sub–interval of the window are each optionally read once for all the radiance data types. Although this design renders the software much more efficient than for earlier versions, further improvements are possible.

6.2 Software Libraries

GOWASP–3 contains 3 libraries. One is for modules utilizing MPI or Shmem that are common to many applications. A second is for modules and subroutines used to read or write observation data files. A third is for other “basic” modules or subroutines that are common to many applications. These all appear in libraries so that they are not redundant within several applications. Their modifications may therefore be rendered in single files although if subroutine arguments change, all their calls from other applications must also be changed, unless any additional arguments are treated as optional.

The “basic” library modules include those for denoting FORTRAN kinds of real variables, for providing values of some common physical parameters, for reading and interpreting resource files that describe the set of required NR fields or parameters for simulating errors, and for incrementing and computing various dates and times. Other basic subroutines include those for specifying file names given generic formats, for determining indexes of requested field names within a list of names, and for interpolating or deriving fields within atmospheric columns. The latter includes computation of hydrostatic heights from temperatures and pressures.

The library used for reading and writing observational data primarily concerns BUFR–formatted files. The only exception is a routine for reading a generic text file for radiance observations. Presently, these are the only formats of observational data files readable by the G5DAS. This library also contains modules that define the variables and arrays required to communicate the observational data throughout the GOWASP–3 for the various classes of data that may be considered.

6.3 Software Modules

GOWASP–3 makes extensive use of FORTRAN modules. All components are designated as private unless specifically designated as public. The names of public module variables generally have common prefixes associated with the module name. This thereby aids recognition of module variables when they appear in other software components. Uses of module components are explicitly indicated using the **only** option in the **use** statements.

6.4 Software Use of MPI

Several applications use explicit MPI commands, apart from those implicitly in the Shmem routines. In particular, such commands are used to allow the CRTM to simultaneously compute radiances on different processors for corresponding different profiles. They are also used for distributing calculations performed when creating horizontal or 3–D correlated random fields.

Only very small files, such as the resource files, are read by every processor. All NR data files and most other large files are read or written using a single processor. When data for multiple 3–D NR fields are required simultaneously, separate processors are used to simultaneously read each named field at each time requested.

6.5 Software Use of Shmem

Unlike in the G5DAS, some current GOWASP–3 applications require consideration of values of NR fields many grid points distant from an observation location when that observation is simulated. Since simulated sondes drift with the NR winds, they may traverse many grid points during ascent or descent. The GPSRO observations are created by considering signals propagating through wide atmospheric planes rather than simply columns. Also, future versions of GOWASP may simulate radiance observations by explicitly computing radiances spatially averaged over viewing footprints. With Shmem, accessing the required field values is trivial. The more common practice of tiling the data among processors using MPI would instead require many MPI instructions and possibly very large halo (overlap) regions.

The total RAM required by GOWASP–3 applications must not exceed that available on a single node when using ESMF Shmem. Currently at the GMAO, this limit is approximately 28 GB. For computers with less RAM per node, it may be impossible to execute some GOWASP–3 applications without altering the software. For simulation of radiances, required alterations may be minor, but for CONV or SATWIND it may be necessary to replace Shmem.

The G5NR–1 3-D data files each contain 1 named field at one time and comprise 0.88 GB of packed data or 4.8 GB of (unpacked) 4–byte data if all levels are included. Thus, no more than four 3–D fields may be resident simultaneously if memory is also required for other large arrays. For 2 times to be present for each field so that time interpolation may be performed, this limit means only 2 distinctly named pairs of fields can be present simultaneously, although memory remains available for some additional pairs of named 2–D fields. For applications requiring no time interpolations or only a subset of atmospheric levels, more named fields can be resident simultaneously. In particular, for CONV or SATWIND data types for which only the lowest 70% of levels are required, three 3–D fields at two times may be resident. Without the absence of 30% of the levels, the CONV and SATWIND applications that require 6 named fields to be simultaneously present would not function on a single node with the high–resolution G5NR–1 on computers available to the GMAO.

When using Shmem, it is possible that previously allocated memory and resident data are encountered when executing a program anew. It therefore is necessary to execute a shmem cleaning program (***RmShmKeys***) prior to executing any program that invokes Shmem routines. The clean up is also applied after such a program is executed, although this is unnecessary. When such a program is inside a UNIX loop, the Shmem cleaning occurs within each iteration of the loop. This clean up is external to any GOWASP–3 application program execution.

6.6 Software Use of BUFR

The G5DAS reads almost all observations from files in BUFR format. GOWASP–3 therefore does likewise. Although formats for the same observation types are identical, only those BUFR variables that are actually referenced by the G5DAS BUFR read subroutines are actually read or written by the corresponding GOWASP–3 subroutines. Currently, the only observation type not read from a BUFR file is GENRADTXT, which is intended for simplified implementation of new radiance data types.

The BUFR read subroutines in G5DAS not only read the files but also perform data consistency checks, thinning, and some quality control. The corresponding GOWASP–3 subroutines instead only read and write BUFR files. Consistency checks, thinning, and quality control are performed elsewhere.

6.7 Software Use of CRTM

The JCSDA CRTM is used to create radiances and brightness temperatures. This considers viewing locations and geometries as well as collections or vertical profiles of atmospheric fields.

Radiance observations are in the form of either radiances, scaled radiances, or brightness temperatures, according to how the observation class is represented in the G5DAS. The GOWASP–3 applications must be appropriately directed to produce the correct representation and to transform between representations as required. In particular, all explicitly simulated observations errors are determined as brightness temperatures, so transformations to and from radiances are required for some classes. The CRTM is also used to perform such transformations.

Unlike its use in G5DAS, each call to the CRTM in GOWASP–3 treats a single observation location. The treatment of clouds or precipitation as elevated black–bodies results in varying numbers of atmospheric levels above the effective radiative surface. For the CRTM to ingest many profile locations in a single call, the numbers of levels must be identical. Since this is not necessarily the case in GOWASP–3, CRTM parallelization is achieved by calling the CRTM from different processors, one call for each observation location.

Whether scattering or absorption by hydro–meteors or aerosols is considered is controlled by user–specified variables in a resource file. Names for the radiatively relevant fields must be specified. Their availability in the data set of profiles is then confirmed, and their data indexes in the array of profiles are determined. If hydro–meteors and aerosols are not to be considered, then their profiles may be absent. Currently, no aerosols are considered by GOWASP because it is missing a required table of humidity–dependent absorption coefficients. In any case, the use of aerosols or hydro–meteors should include a currently missing stochastic component to ensure realistic inaccuracies of the modeling of their radiative effects.

The ***m_crtm_interface.f90*** module is the interface between GOWASP–3 and the CRTM. Besides calling the CRTM to produce radiances, it also calls CRTM setup routines, including routines that read CRTM files of spectral parameters for specific instruments and platforms. Additionally, it checks parameters for each observation viewing geometry to ensure consistency. These checks are identical to those performed in the G5DAS BUFR–read subroutines for the same observation classes.

The CRTM creates radiances or brightness temperatures for all channels present in the BUFR files for that observation class. For some hyper–spectral instruments, this may be a subset of existing channels, and even all those in the BUFR subset may not actually be used by the G5DAS. For those not used due to elimination by either observation thining or quality control, the quality of their CRTM simulation remains unknown in the OSSE context because of the absence of reliable independent data with which to compare.

6.8 Software Use of Scripts

All GOWASP–3 UNIX scripts are written in c–shell. Their use and descriptions appear in other chapters of this document.

All the scripts use an application called ***addtime***. This is used to increment an input time stamp by an input number of hours. The format for the time stamp is **yyyymmddhh** where **yyyy** is the year, **mm** is the month, **dd** is the day, and **hh** is the hour. Leading zeros for month, day, or hour must be present. The update algorithm works for all years between 1901 and 2099, inclusive.

During the script work flow, the existence of some input files denoted in an executable's argument list is checked. If a required one is nonexistent, the executable is not invoked, an error counter is incremented, and an error notice is printed to the script's default log file. After the executable is invoked, the existence of any output file denoted in the argument list is checked. If an expected output file does not exist, an error counter is incremented and an error message logged; otherwise a counter of successful executions is incremented. Values for these counters are printed to the log file when the script is completed.

Chapter 7

Procedure for Tuning errors

All metrics used to measure impacts of observations in an OSSE context essentially concern measuring errors (i.e., analysis or forecast minus truth) or other differences (e.g., background minus analysis). All these differences are functions of system errors. Such errors include observation errors (due to both instrument and representativeness), forecast model errors (due to formulations of numerics, parameterizations, discretization and boundary conditions), and errors introduced by the DAS algorithm itself (e.g., by noise filters or bias correction). Furthermore, the appearance of these errors are significantly modified by the assimilation algorithm and forecast model. Once the nature run, assimilation model, and DAS are chosen, however, an investigator only is able to adjust the added observation error statistics in order to obtain results that appear valid. In principle, adjusting errors in the assimilation model is also feasible, such as by tuning stochastic physics, but this requires much more consideration about how the model is being artificially made worse, assuming that it has already been well tuned. Tuning of added observation errors therefore should be attempted first.

Ideally, all statistics that could be measured during assimilation of real observations would match corresponding ones in the OSSE context. This is unrealistic because real model error is likely significantly dissimilar to the corresponding simulated model errors (created by differences between assimilation and NR model formulations). Also, important subtleties in real observation errors may be absent from the simulated ones added in the OSSE context.

The statistics to attempt to match first are those for

$$\mathbf{d} = \mathbf{y}^o - \mathbf{H}(\mathbf{x}^b) \ , \tag{7.0.1}$$

where $\mathbf{d}$ is the column vector whose components are the innovations (or increments) for a set of observations. The innovations are the values most directly affected by the added observation errors and thus the easiest to tune.

If we postulate true values of what is measured by the observations of the atmospheric fields, then the innovations also satisfy

$$\mathbf{d} = \epsilon^o - \epsilon^{ob} \ , \tag{7.0.2}$$

where ϵ^o here is the sum of observation instrument and representativeness errors and ϵ^{ob} is the error in the simulated observation value determined from the DAS background fields due to errors in those fields, assuming that a perfect relationship between fields and observations is used (The additional error due to the DAS not using a perfect relationship is the representativeness error component lumped into ϵ^o). Thus, we see that the statistics of $\mathbf{d}$ are determined by those of the differences between corresponding errors of observations and errors of estimates of those observations due to errors in the DAS background states. In particular, the biases are related by

$$\langle \mathbf{d} \rangle = \langle \epsilon^o \rangle - \langle \epsilon^{ob} \rangle \ , \tag{7.0.3}$$

where the angle brackets denote ensemble, temporal, or spatial mean values. Also, if ϵ^o and ϵ^{ob} are uncorrelated, then the covariances satisfy

$$\mathbf{C}^d = \mathbf{C}^o + \mathbf{C}^b \ , \tag{7.0.4}$$

where

$$\mathbf{C}^d = \langle \mathbf{d}\mathbf{d}^T \rangle \ , \tag{7.0.5}$$

$$\mathbf{C}^o = \langle \epsilon^o \epsilon^{oT} \rangle \ , \tag{7.0.6}$$

$$\mathbf{C}^b = \langle \epsilon^{ob} \epsilon^{obT} \rangle \ . \tag{7.0.7}$$

In DAS literature, the latter two covariances are associated with the notations $\mathbf{R}$ and $\mathbf{HBH}^T$, respectively, but are equivalent only if $\mathbf{R}$ and $\mathbf{B}$ are the true observational and background error covariances, not approximated as in any DAS, and if the observation operators relating the background fields to observation values are all linear and thus expressible by a matrix $\mathbf{H}$.

From (7.0.3) it is clear that from examination of innovations only, the observation error biases cannot be determined because the contributions by $\langle \epsilon^o \rangle$ and $\langle \epsilon^{ob} \rangle$ cannot be distinguished. Similarly,

$\mathbf{C}^o$ cannot be determined from $\mathbf{C}^d$ alone. Further information or assumptions about $\langle \epsilon^{ob} \rangle$ and $\mathbf{C}^b$ are required.

7.1 Tuning of Biases of Observation Errors

There are at least two sources of error biases in a DAS: one is biased observation errors and the other is biased model formulation errors. Modern DAS attempt to remove most of the former, particularly for radiance types. Thus carefully tuning such biases in an OSSE may be irrelevant when they are subsequently effectively removed by the DAS. In the OSSE context, model formulation biases reflect differences between the NR and assimilation models rather than between nature and the assimilation model. If both the NR and assimilation models are considered somewhat equally realistic, there is no reason to expect biases of the model formulation errors to be the same in the real and OSSE contexts, except regarding their "typical" magnitudes.

Without formulating some specific hypothesis to investigate, it is notably unclear what or why biases should be matched. Tuning of biases is therefore not recommended unless a specified issue concerning biases is being particularly investigated. Instead, it is simply recommended that biases of observation innovations be monitored to see that values have similar qualitative magnitudes in real and OSSE contexts. Specifically, corresponding bias–corrected values should be compared since only these have any impact on the analysis. Also, the magnitudes of biases in the analysis increments or forecast errors should be monitored to see if some aspect appears so qualitatively dissimilar in real and OSSE contexts that the utility of the OSSE results becomes questionable. In this regard, note that, given the differing sources of biases in the two contexts, there is no reason that biases should have either the same signs or structures. Sufficient is that, for example, if magnitudes of the biases are "small" compared to the corresponding standard deviations in the real data context, they are analogously small in the OSSE context

7.2 Algorithm for Tuning of Variances of Observation Errors

Since correlations are defined as covariances normalized by variances and since the GOWASP–3 parameters for adding observation errors concern variances and correlations, it is best to first tune the variances of added errors before tuning any of their correlations. These variances are the diagonal elements of $\mathbf{C}^o$. The diagonal elements of these matrices will be denoted by, e.g., V^o.

Since $\mathbf{C}^b$ is a function of both forecast model and analysis error statistics and the latter is expected to depend on observational error statistics, $\mathbf{C}^b$ is formally an implicit functions of $\mathbf{C}^o$. If $\mathbf{C}^b$ has a strong and highly nonlinear dependence on $\mathbf{C}^o$, tuning $\mathbf{C}^o$ could be very difficult. Fortunately, for variances at least, this is typically not the case.

Denote variances pertaining to the assimilation of real data using a subscript R (for "Real") as the desired target values. Denote variances pertaining to OSSE tuning experiments by a subscript integer i denoting a tuning–experiment iteration number. Then, consider what is directly known or measurable. Both V_R^d and V_i^d are measurable and generally provided as standard diagnostic output from a DAS. Both V_R^o and V_R^{ob} are generally unknown since, since although $\mathbf{R}$, $\mathbf{H}$ and $\mathbf{B}$ must be specified within a DAS, these are at best approximations for determining the true V_R^o and V_R^{ob}. For $\mathbf{R}$, this may be a rather gross approximation, as when realistic error variances are inflated by factors of 2–3 to account for neglected error correlations. Although in the OSSE experiment V_i^{ob} can be determined in principle but perhaps with some difficulty, the variance of added observation error is known precisely because it is explicitly prescribed. If the variance of any implicit observation representativeness error (e.g., the portion of representativeness error introduced because the DAS background and NR grids differ) is included in V_i^{ob}, the latter can be obtained as a residual:

$$V_i^{ob} = V_i^d - V_i^o \ . \tag{7.2.1}$$

It is also true that

$$V_{i+1}^o = V_{i+1}^d - V_{i+1}^{ob} \ . \tag{7.2.2}$$

In (7.2.2), replace V_{i+1}^d by V_R^d as the desired target value for the next tuning iteration. Then assume that V_i^{ob} changes less from one iteration to the next as V_i^o is changed so that $V_i^{ob} \approx V_{i+1}^{ob}$. Then, combining (7.2.1) and (7.2.2),

$$V_{i+1}^o \approx V_R^d - V_i^d + V_i^o \tag{7.2.3}$$

provides the observation error variances to be used for the next tuning experiment. There is nothing that ensures $V_{i+1}^o > 0$ although that is typically the case because it is expected to represent missing errors. If a negative value is obtained, however, it is set to 0 or some small value.

In practice, only 1–2 tuning iterations are required if added observation errors are uncorrelated and the DAS $\mathbf{R}$ is diagonal. In this case, most of the observation error is filtered when the DAS constructs the analysis, and thus V_i^{ob} varies negligibly between iterations. Thus, (7.2.3) becomes a nearly exact relationship.

Note that by specifying V_{i+1}^o such that $V_R^d = V_i^d$, the observation error variance will be modified to compensate for any unrealism in V_i^{ob} compared to V_R^{ob}. So, e.g., if forecast model error variance in the OSSE is unrealistically low, the tuned V_{i+1}^o will be be unrealistically increased to compensate for the diminished V_i^{ob} that is actually due to the lack of model error. In such a case, the investigator must decide whether obtaining a realistic V_i^d is more desirable than obtaining a realistic V_{i+1}^o after considering what effects this decision may have on other OSSE metrics.

7.3 Algorithm for Tuning of Correlations of Observation Errors

Since a DAS designed to filter uncorrelated observation errors, by assuming a diagonal $\mathbf{R}$ as in the G5DAS, may be expected to effectively filter such errors, realistically modeling observation error components that are spatially or channel correlated may be more critical for validating OSSE metrics. Simply, since these errors are not effectively filtered, they are the ones remaining to potentially affect results. This was indeed seen in earlier OSSE work at the GMAO where it was noticed that it was necessary to add correlated observation errors to better validate variances of analysis increments (i.e., differences between analysis and background fields) produced during assimilation.

In the GOWASP–3 experiments to date, spatial correlations of added observation error have been tuned by considering diagnostics and procedures similar to those presented in Hollingsworth and Lonnberg (1986). Here, these employ sets of averaging “bins”. For correlations between radiance channels, these are simply defined by pairs of integer indexes n, m referring to channel numbers. For spatial correlations, however, the bins are defined by ranges of pressure or separation distances rather than channel indexes.

For determining isotropic horizontal correlations, the averaging bins are defined by ranges of great–circle, horizontal separation distances D between a pair of observations; e.g., bin $k = 1, \ldots, N_h$ includes all observation pairs considered for which $D_{k-1} < D \leq D_k$ for some specified set of D_k. With a $D_0 = 0$, an additional bin with index $k = 0$ is defined for pairs with $D = 0$ (i.e., each observation paired with itself) that will be used to determine the innovation variances. Such ranges are required when observations are irregularly spaced, as most are, since (1) determining correlations requires numerical averaging over many sample products and (2) every observation pair may have a unique value of D. The ranges should be wide enough to include a sufficiently large sample of pairs so that the statistics are reliable but short enough so that the pairs within it may be expected to have similar correlation values.

For vertical correlations, the set of bins may be defined by dividing the vertical range of observation heights into N_v equally–spaced vertical bins. Since in the G5DAS, vertical locations are defined in terms of p rather than z, these are actually equally spaced in $T\Delta \log(p)$, where T is prescribed using a standard atmosphere). Then pairs of observations from the same soundings (so horizontal locations vary relatively little for nearby levels) are assigned to pairs n, m where now the indexes refer to vertical bin indexes.

Next, all the pairs of observations within each bin are considered. Note that any single observation is generally included in several pairs (i.e., it is paired with several other distinct observations). For investigating channel correlations, all pairs of channels are considered, but only for observations at the same scan location. For investigating vertical correlations, all pairs of observations within each individual sounding are considered. For investigating horizontal correlations, only observations with the same channel number (in the case of radiances) or near the same pressure levels (in the case of conventional observations) are paired. These constraints are applied so that these individual aspects of the correlations (i.e., either inter–channel, vertical or horizontal) are examined separately.

For each considered pair, products of the innovations $d_i d_j$ are computed. These are then summed with products from other observation pairs (i.e., for pairs at other horizontal locations) that fall within the same bin. The number N_c of such pairs summed within each bin is then used to compute an average of the products within that bin (Specifically, the sum is divided by $N_c - 1$, as required when computing variances. These averages are the estimated covariances. They are then normalized by the square root of the product of variances of the innovations used within each bin (i.e., for the two channels associated with a channel bin pairing, or the two vertical levels in the case of vertical correlations, or the channel or p-level observations in the case of horizontal correlations) to determine correlations from the covariances.

As an example, consider determination of horizontal correlations among all the pairs of AMSUA channel 8 observations that passed quality control in the G5DAS during a 2–week period in a particular July. These can be denoted here by $d_{i,m}$ where i is associated with geographic location and m with (6–hour) assimilation time window.

- Define 24 bins of horizontal great–circle separation distances in addition to the bin for $D = 0$ by defining $\Delta D = 40$ km and $D_k = k\Delta D$.
- For each observation pair occurring in the same time window, compute the distance D between them. So, if at each time there are I_m observations, there are $I_m(I_m + 1)/2$ distinct pairs, after removing redundancies by considering the pairs (i, m) (j, m) the same as (j, m) (i, m). Ignore pairs with very large separations (here, $D > 960$ km). Then determine the sums

$$S_k = \sum_m \sum_{i=1}^{I_m} \sum_{j=i}^{I_m} d_{i,m} d_{j,m} \delta_k(D) \ , \tag{7.3.1}$$

  where $\delta_k(D)$ is 1 or 0 depending on whether the separation distance D between the locations of the i–th and j–th observation pair lies within bin k.
- Divide each S_k by $N_c - 1$. Then divide those results by the averaged value by S_0. If the sample size is large enough, this result is a reliable estimate of the isotropic horizontal correlation. It is isotropic because only separation distance, not direction, has been considered.

Note that when comparing horizontally separated observations explicitly, there may also be implicit temporal separations. For nearby radiances observed by the same instrument, the temporal separation will typically be small since they will occur within the same observation swath. The only temporal aspect considered by these algorithms is that only observation pairs within the same assimilation period are used for the calculations. Thus, any temporal correlations within an analysis period are aliased onto the estimated spatial correlations.

The value of ΔD should be set sufficiently large so that enough observation pairs fall within each bin that the sample size can provide a reliable estimate. It also should be sufficiently small so that the correlations lumped within the same bin may be reasonably expected to be similar. Since spatial thinning is applied to radiance observations within the G5DAS, the sample sizes for pairs with small separation distances will generally be quite small and their estimated correlations generally unreliable. The same problem applies to radiosonde observations.

The same calculation should be performed for both real assimilated observations and OSSE assimilated observations. These can differ between the two contexts because either the background error fields have different correlations or the observation error correlations differ. If it is suspected that the latter explains a greater portion of the discrepancy, then some portion of the added observation error should be generated with correlations.

Since (7.2.3) is derived from (7.0.4), a similar algorithm can be used to tune observation error correlations to be added. If the symbol V is replaced by C_k denoting the covariance determined within the k–th bin (i.e., k can refer to a horizontal distance bin or a pair of vertical bin or channel indices), the next iterate of the missing covariance becomes

$$C^o_{k,i+1} \approx C^d_{k,R} - C^d_{k,i} + C^o_{k,i} \tag{7.3.2}$$

where $C^o_{k,i}$ is the previous covariance of added observation error. This latter is provided by the observation error model shape and parameters used to create the added observation errors in the previous OSSE tuning exercise.

For covariances between channels or vertical levels, the set of $C_{k,i+1}$ determined by (7.3.2) comprise a matrix where the single index k refers to matrix elements. The form of (7.3.2) does not ensure, however, that this matrix is truly a covariance. In particular, there is nothing to ensure that the correlations defined by this matrix are bounded between ± 1. For this reason, this estimated covariance is adjusted to ensure that the absolute values of the correlations are bounded by some value less than but close to 1. This adjustment preserves the variances along the matrix diagonal. In GOWASP–3, the number of matrix elements adjusted are printed to the log file produced by the script performing the tuning.

For covariances between channels, the added error covariance model may be defined simply as the matrix estimated from (7.3.2). For spatial covariances, however, it is necessary to use (7.3.2) to instead estimate parameters to be applied to an error model, such as those described in chapter 4. This is accomplished by fitting the various error models to the correlations obtained from (7.3.2).

For horizontal correlations, the three correlation functions defined by (4.3.25) are considered, although others can be added easily. Each of these is defined by only 2 parameters. One is the fraction of total error variance associated with the correlated part of the error. The other is the length scale L that has different meanings for each shaped function. For the first parameter, values $\nu = n/100$ for $n = 1, 2, \ldots, 100$ are considered. For the second, $L = 10n$ km for $n = 1, 2, \ldots, 60$.

Larger values of L are not considered because then weak covariances are more likely explained by weak biases or background errors than by random instrument or representativeness errors.

For each functional shape $F(L)$, a measure of fit is determined as

$$S(\nu, L) = \left[\left(\sum_{k=1}^{N_h} w_k \right)^{-1} \sum_{k=1}^{N_h} w_k \left(\nu F[\frac{1}{2}(D_k + D_{k+1}), L] - C_0^{-1} C_k \right)^2 \right]^{\frac{1}{2}} \tag{7.3.3}$$

where the index k denotes horizontal distance bins with $\frac{1}{2}(D_k + D_{k+1})$ being the center value of each range, C_0 denotes the variance (covariance at separation distance 0) such that the normalized C_k are therefore correlations, and w_k are weights

$$w_k = \begin{cases} 0 & \text{if } C_k < 0.05 \text{ or sample size too small,} \\ 1 & \text{otherwise.} \end{cases} \tag{7.3.4}$$

The condition on w_k removes distant or statistically unreliably estimated covariances from the metric. The pair of tested values of ν and L for which S is a minimum defines the best fit.

Although the algorithm for determining error variances to add converges quickly, this procedure applied to covariances may not since correlated components of the added errors are not filtered by the current G5DAS. The off–diagonal components of C_i^{ob} therefore may not be sufficiently stationary between iterations, affecting the efficiency or even stability of iterating (7.3.2). If the algorithm does perform well enough, it may work best applied to the correlations of errors between channels for hyper–spectral radiance observations. Other possible difficulties are described later in this chapter.

Two examples of binned and fitted results for horizontal correlations of observation innovations are presented in Fig. 7.1. The top panel is for Himawari VIS/IR SATWIND u at 850 hPa, and the bottom for AMSUA–A channel 6 on METOP–A. All bins have widths of 40 km. The innovations are from an assimilation of real observations during July 2015. Binned values appear as stars. Blue and red curves are for exponential and Gaussian functions, respectively, that fit the starred values best for separations $120 \leq D \leq 940$ km using weights that are constant with respect to distance. Note that for AMSU–A, a correlation equal to 1 at $D = 0$ is implied, as otherwise shown explicitly for Himawari. For the presented AMSU–A innovations, the best fit for the exponential was obtained using $L = 660$ and $f = 0.38$ and for the Gaussian using $L = 580$ and $f = 0.28$. For the presented Himawari innovations, the best fit for the exponential was obtained using $L = 280$ and $f = 0.75$ and for the Gaussian using $L = 300$ and $f = 0.46$. Note that the latter binned

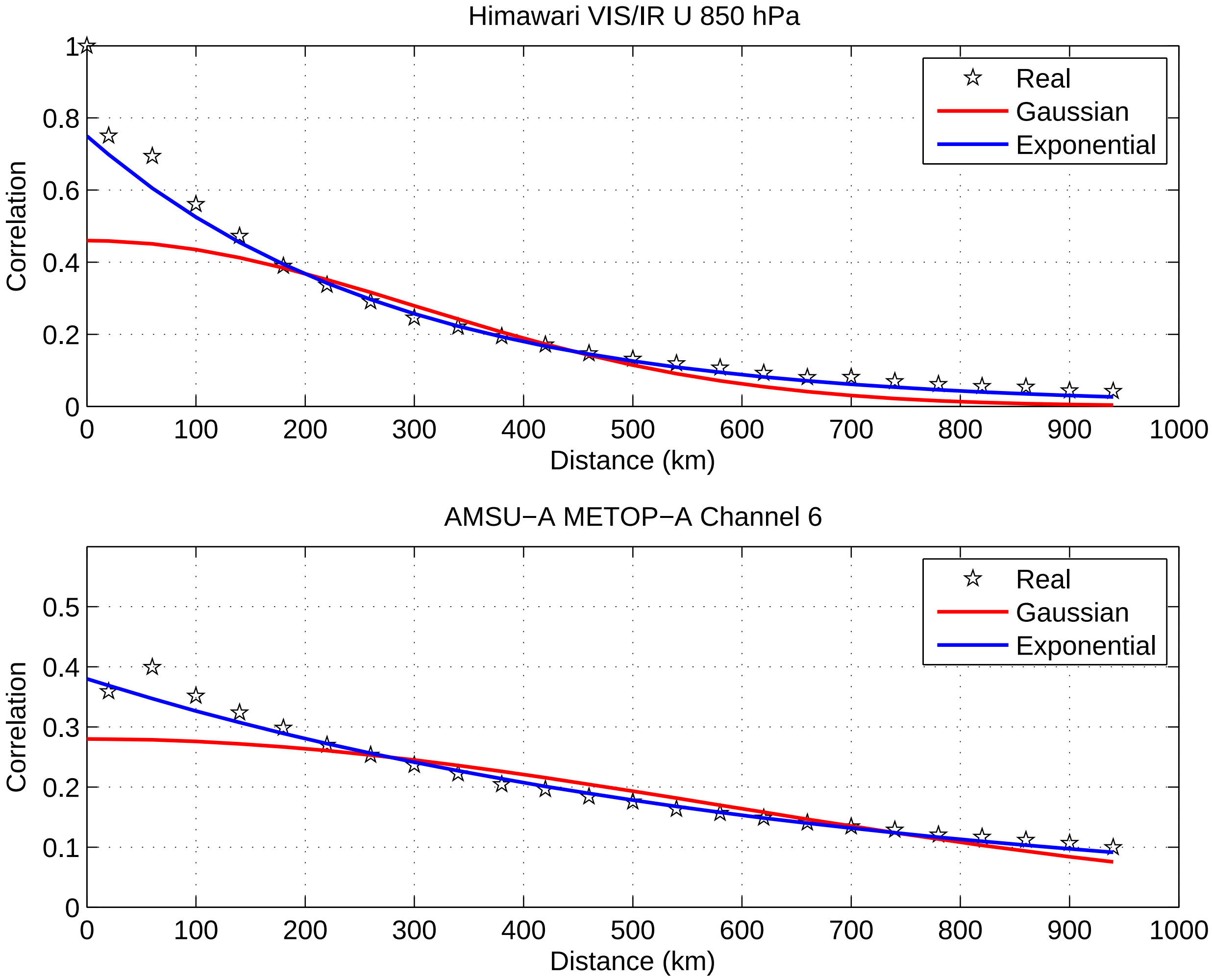


Figure 7.1: Binned and fitted horizontal correlations of observation innovations for Himawari VIS/IR SATWIND u at 850 hPa (top) and for AMSUA–A channel 6 on METOP–A (bottom) as functions of separation distance. Binned values for real observations during July 2015 appear as stars. Blue and red curves are for exponential and Gaussian functions, respectively, that fit starred-values best.

correlations have been computed as if they were isotropic (i.e., ignoring direction) that is likely not valid when considering wind components.

7.4 Users Guide for Observation Error Tuning

The GOWASP–3 includes programs to assist in tuning parameters used to generate simulated observation errors to add. These programs may be divided into 4 groups, each depending on results produced by the previous group.

7.4.1 Compute innovation statistics

The first group contains programs that compute statistics of observation innovations required for observation monitoring and error tuning. Their input are diagnostic G5DAS *.ods files that contain a subset of information derived from the G5DAS *.diag files. The programs use G5DAS library scripts and modules to read these G5DAS diagnostic files.

All the statistics are both time and horizontal mean values. Only observations that were actually used by the assimilation are considered, as indicated by quality flags assigned in the *.ods files. Statistics can also be computed for observation minus analysis values in observation space, analysis minus background values in observation space, or observation values themselves.

The relevant scripts within this first group are

- ***countobs_rad.j***: Produces means, standard deviations, and counts as functions of instrument channel.
- ***countobs_conv.j***: Produces means, standard deviations, and counts as functions of a set of 12 broad pressure layers for distinct fields and observation subtypes. The pressure layers are defined within the program. They are contiguous so that each observation falls within one of the layers.
- ***hcorr_rad.j***: Produces means, standard deviations, counts, and horizontal covariances as functions of instrument channel for user–specified ΔD and N_h.

- ***hcorr_conv.j***: Produces means, standard deviations, counts, and horizontal covariances within a set of 10 narrow pressure layers for distinct fields and observation subtypes for user–specified ΔD, N_h, and pressure–layer thicknesses.

- ***chcorr_rad.j***: Produces means, standard deviations, counts, and the matrix of covariances between channels.

- ***vcorr_conv_gps.j***: Produces means, standard deviations, counts, and vertical spatial covariances for radiosonde and GPSRO soundings. It does not do so for dropsondes because the sample size over a month would be too small for obtaining reliable correlation estimates.

Each of these scripts should be run on 10–30 days of input data for both real and OSSE assimilations so that the results can be compared to both verify the OSSE and to iteratively tune the observation errors for the OSSE. Some of the less obvious arguments required by the programs run by these scripts are

- **ibins**: When computing horizontal correlations, this is the number $(1 + N_h)$ of bins of horizontal separation distance ranges to consider. Note that this number includes the bin for distance 0. When computing vertical correlations for radiosondes or GPSRO observations, this is the number of contiguous layers of pressure (in the case of sondes) or height (in the case of GPSRO) within which each observation location will be assigned.

- **x_max**: The maximum horizontal separation distance (D_{N_h+1}) to consider. The width ΔD of the bins of horizontal distances is determined by this value divided by that of **ibins** minus 1.

- **delp**: The width of the pressure layers (in millibars) to consider when computing horizontal correlations for conventional observations. The central values of the 10 pressure layers are prescribed in the programs called. This value should be small enough that the separations between observations may be considered as almost entirely horizontal but large enough that many observation pairs fall within each separate layer.

- **kt**: For conventional observations, this is an integer that denotes the field type (T, u, v, q, p_s).

- **kx**: For conventional observations, this is an integer that denotes the observation subtype. These are described on an NCEP web page listing PREPBUFR subtypes used by G5DAS: http://www.emc.ncep.noaa.gov/mmb/data_processing/prepbufr.doc/table_2.htm.

- **sat_info_file**: The same satellite information resource file used by ***create_rad_bufr*** and ***create_error***.

- **sfc_type**: Allows counting of observations only over sea, only over land and ice, or over all surface types.

- **comp_diag**: This causes computation of statistics for observation minus forecast ("omf"; i.e., observation innovations), observation minus analysis ("oma"), observations themselves ("obs"), or analysis increments ("amb") in observation space. The same types of statistics are computed for each choice.

Innovation counts, means and standard deviations determined by ***countobs_rad.j***, ***hcorr_rad.j***, ***chcorr_rad.j*** will be identical if the statistics are computed for the same region. Observation error variances estimated using results from any of these scripts should be identical. For conventional observations, however, the statistics determined by ***countobs_conv.j*** and ***hcorr_conv.j*** may differ because the first considers contiguous pressure layers while the second considers separated narrow layers.

7.4.2 Compute observation error parameters from innovation statistics

This group includes 3 scripts:

- ***eparams_conv_hcorr.j***: Computes tables of observation error standard deviations and best–fit correlation parameters ν and L as functions of pressure level for selected horizontal correlation shapes. Input are files produced by ***hcorr_conv.j*** although if no correlations are to be fitted, files produced by ***countobs_conv.j*** can be input instead (so that only new fits of variances will be produced).

- ***eparams_rad_hcorr.j***: Computes tables of observation error standard deviations and best–fit correlation parameters ν and L as functions of channel number for selected horizontal correlation shapes. Input are files produced by ***hcorr_rad.j*** although if no correlations are to be fitted, files produced by ***countobs_rad.j*** can be input instead (so that only new fits of variances will be produced).

- ***eparams_rad_chcorr.j***: Computes a binary file of the difference between 2 (e.g., real and OSSE) computed innovation, inter-channel covariance matrices. Also computes a table of fitted observation error standard deviations. Input files are those produced by ***chcorr_rad.j***.

These are intended to be run for all appropriate files in directories previously created by scripts in the first group. They require information about the OSSE and target (i.e., real observation result to be matched) directories and experiment names (the latter are part of the input and output file names), the ***sat_info.rc*** file, and the path and files defining the parameters used to create any observational errors added to the observation data set utilized by the G5DAS OSSE under consideration. These latter are used to create V_i^o and $C_{k,i}^o$. Also, an argument **corr_func_new** (equal to GAUSS, EXP, TOAR, or WHITE) selects the functional shape that should be fitted to determine correlation parameters.

The output from this group of programs should be examined to ascertain what shape of horizontal correlation function would be best. A table of fits to 3 functions are included in the output for comparison. If a different shape than the one specified by **corr_func_new** is to be used by the subsequent tuning procedure steps, than these programs need to be re-run with the new shape specified since only the portion of output for that selected shape will be read by subsequent tuning programs. At this point is is also useful to examine the output for possible peculiarities that may occur in the fitting algorithm; e.g., due to small sample sizes or weak correlation values. At this time, there is no specified examination procedure to recommend. It remains an art.

The values of ν_k estimated for the horizontal correlations are determined for each channel or specified pressure layer. Those for channels are applied to the same channels when simulating observations. Those for pressure layers are interpolated or extrapolated to a set of pressure levels that will encompass all the observations of that type. The values of L_k estimated for the horizontal correlations are also determined for each channel or pressure layer. If they are used to create errors that are also channel or vertical correlated, however, they are instead applied to the the PCs of the channel or vertical covariances. This is a limitation of the separability (between horizontal and other correlations) of the modeled error in GOWASP–3.

The ***eparams_**** programs print a flag indicating the quality of the calculated estimates. If either the real or OSSE observation counts are considered too small to create reliable covariances, the flag **QVCT** is printed. If either V_R^d or V_i^d is exceptionally small, the flag **QV00** is printed. If the estimated $V_i^{ob} < 0$, the flag **QVBN** is printed. This condition can occur when the OSSE innovation variances exceed those of the target even with no observation errors added. If the last condition is not satisfied but $V_R^d - V_i^{ob} < 0.25V_i^o$, then the flag **QVBS** is printed. This condition can occur, e.g., when the added observation errors are modified as by noise filtering in GSI for ATMS or SSMIS such that the error table value of V_i^o does not indicate the real net error. The absence of a printed flag indicates that no obvious problem was detected.

The programs in this group do not function properly for ATMS or SSMIS observation types. For these types, the G5DAS performs a noise-reduction calculation by computing weighted averages of nearby observations. The effective errors of these "super observations" are significantly reduced compared to the simulated errors added. How much depends on details of the G5DAS data selection and weighting function as well as on correlations of the simulated errors added. Thus, the assumption by ***eparams_rad_hcorr.j*** or ***eparams_rad_chcorr.j*** that the effective magnitude of previously added errors is that found in the previous table of observation error values is invalid and a non–automated tuning procedure is currently required.

7.4.3 Create observation error standard deviation tables

This group includes 2 scripts:

- ***new_R_table_conv.j***: Computes the table of PREPBUFR observation error standard deviations to be read by ***create_error.j***. Input are files produced by ***eparams_conv_hcorr.j***.
- ***new_R_table_rad.j***: Computes the table of radiance observation error standard deviations to be read by ***create_error.j***. Input are files produced by ***eparams_rad_hcorr.j*** or ***eparams_rad_chcorr.j***.

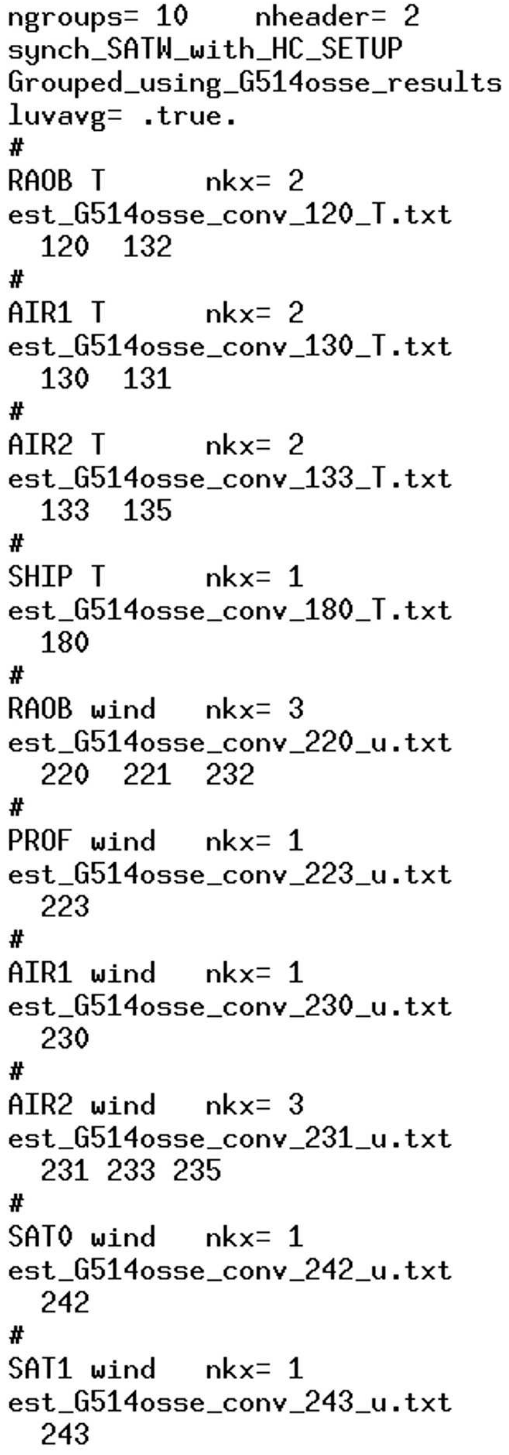
For ATMS and SSMIS, the script ***new_R_table_rad.j*** will not function properly for reasons stated in the previous section. For these data types, therefore, the error table entries must be manually changed. A user must examine the innovation statistics of the real and OSSE innovations and make a best guess at what appropriate standard deviations and correlation parameters may be. For example, if $V_R^d > V_i^d$, then the error table value of $\sqrt{V_i^o}$ should be increased. Note that change should be performed after ***new_R_table_conv.j*** is executed because otherwise the manually changed values may be over–written.

The script ***new_R_table_conv.j*** requires a resource file (such as ***new_R_table_conv.rc***) to specify from what file of estimated error standard deviations the new table values should be copied for each conventional observation data sub-type (**kx** value). An example of this resourece file appears in 7.2. Its format is:

- Line 1: Two integers specifying the number (I_g) of groups of observations sub-types to be subsequently read from this file and the number (I_h) of header comment records to be read next. Each integer must be preceded by a character string and a blank
- Lines 2–($I_h + 1$): Lines of comments that the user may include as a reminder of how or why this table was created.
- Line $I_h + 2$: The value of the variable **luvavg** set to **.true.** if estimated variances for u and v errors from separate input files are to be averaged when winds are considered, and **.false.** otherwise.
- Line $I_h + 3$: The character "#" denoting a separation line between groups.
- Line $I_h + 4$: A line of 2 character strings separated by blanks that denotes the name of the subgroup being described, followed by the character string "nkx=" and a blank followed by an integer I_s. The three character strings are printed to the log file but not otherwise used by the program. The integer I_s denotes the number of **kx** values subsequently listed for this subgroup.
- Line $I_h + 5$: A character string denoting the name of the file of estimated error statistics from which to obtain the table values to create. These files must be in the format of the files output from the second group of tuning scripts. If the file name ends in "_u.txt" or "_v.txt", the corresponding file for the other wind component will also be used, so that the desired variances are determined from an average of estimates for both wind components.
- Line $I_h + 6$: A list of I_s integer values for PREPBUFR data type index **kx** to be included in this group.

Lines like $I_h + 3$ through $I_h + 6$ are repeated for each group. For each group, the values in the table to be created will be identical for all the **kx** values listed for that group.

The ability to group observations permits statistics from one observation subtype to be used to estimate those for another. For example, since there are so few dropsondes, statistics computed for



```
ngroups= 10    nheader= 2
synch_SATW_with_HC_SETUP
Grouped_using_G514osse_results
luvavg= .true.
#
RAOB T      nkx= 2
est_G514osse_conv_120_T.txt
  120  132
#
AIR1 T      nkx= 2
est_G514osse_conv_130_T.txt
  130  131
#
AIR2 T      nkx= 2
est_G514osse_conv_133_T.txt
  133  135
#
SHIP T      nkx= 1
est_G514osse_conv_180_T.txt
  180
#
RAOB wind   nkx= 3
est_G514osse_conv_220_u.txt
  220  221  232
#
PROF wind   nkx= 1
est_G514osse_conv_223_u.txt
  223
#
AIR1 wind   nkx= 1
est_G514osse_conv_230_u.txt
  230
#
AIR2 wind   nkx= 3
est_G514osse_conv_231_u.txt
  231 233 235
#
SAT0 wind   nkx= 1
est_G514osse_conv_242_u.txt
  242
#
SAT1 wind   nkx= 1
est_G514osse_conv_243_u.txt
  243
```

Figure 7.2: Example of a ***new_R_table_conv.rc*** file, showing an abbreviated list of observation types.

the far more numerous radiosondes can be used in their place. The same is true for some AMV subtypes.

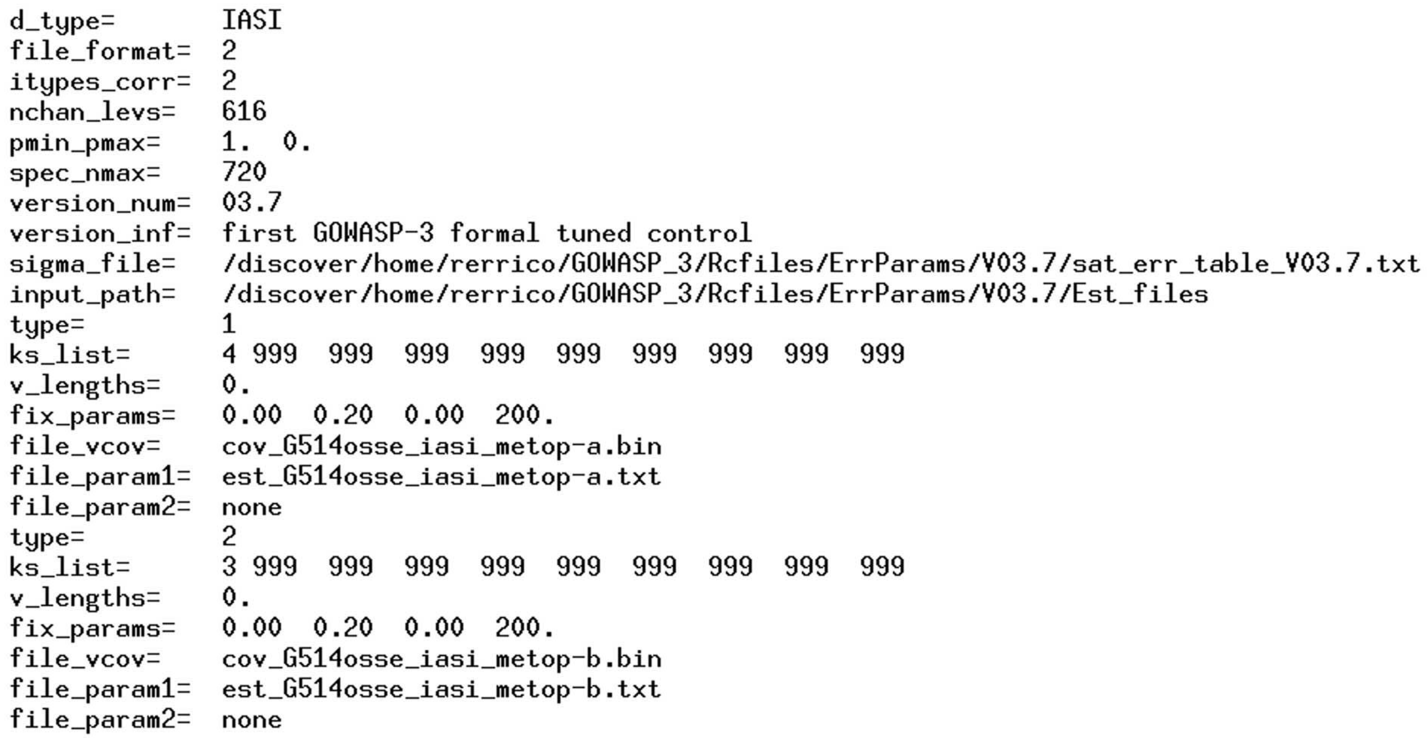
7.4.4 Create files of observation error correlation parameters

This group includes 1 script and 2 types of resource files to be specified.

The script is ***corr_params_input.j***. It uses input from files created by both the observation error correlation fitting scripts and and new R table scripts to create binary files required for the input of observation error correlation files to be read by ***create_error.j*** when creating and adding simulated errors for a subsequent OSSE. The instructions for creating these binary files are specified in a set of ***hc_setup_*.txt*** files, one file for each observation data type for which added observation errors are to have correlations. If correlations are not to be considered for a data type, then its resource file is not required. An example ***hc_setup_*.txt*** is shown in 7.3 for observation type IASI.

Each ***hc_setup_*.txt*** begins with ten lines that must occur in a specific order. Each line begins with a specific character string ending with an equal sign followed by a space. The user–provided values that follow on each line are of specific FORTRAN variable type but otherwise unspecified format. These ordered lines are

- **d_type**: A character string denoting one of the observation data types allowed in GOWASP. See chapter 5 for a list of possible values.
- **file_format**: An integer denoting the file format. This permits some legacy files to be read even if the file format is changed. For GOWASP–3, use the value 2 here.
- **itypes_corr**: An integer denoting the number of distinct groups of subtypes that are to share the same set of correlation parameters.
- **nchan_levs**: An integer denoting the number of channels if this data type is a radiance, or the number of vertical pressure or height levels on which to define random fields if this is a conventional or GPSRO observation. For the non radiance types, chapter 4 should be consulted before specifying this number.
- **pmin_pmax**: Two real values denoting the minimum and maximum (in that order) of a range of pressure levels (in millibars) over which a set of horizontal random fields will be computed



```
d_type=       IASI
file_format=  2
itypes_corr=  2
nchan_levs=   616
pmin_pmax=    1.  0.
spec_nmax=    720
version_num=  03.7
version_inf=  first GOWASP-3 formal tuned control
sigma_file=   /discover/home/rerrico/GOWASP_3/Rcfiles/ErrParams/V03.7/sat_err_table_V03.7.txt
input_path=   /discover/home/rerrico/GOWASP_3/Rcfiles/ErrParams/V03.7/Est_files
type=         1
ks_list=      4 999  999  999  999  999  999  999  999  999
v_lengths=    0.
fix_params=   0.00  0.20  0.00  200.
file_vcov=    cov_G514osse_iasi_metop-a.bin
file_param1=  est_G514osse_iasi_metop-a.txt
file_param2=  none
type=         2
ks_list=      3 999  999  999  999  999  999  999  999  999
v_lengths=    0.
fix_params=   0.00  0.20  0.00  200.
file_vcov=    cov_G514osse_iasi_metop-b.bin
file_param1=  est_G514osse_iasi_metop-b.txt
file_param2=  none
```

Figure 7.3: Example of a ***hc_setup.txt*** file, here specifically for observation type IASI

.

for conventional or GPSRO data types. For radiance data types, the first value should simply be larger than the second, indicating that no range of pressure levels is required.

- **spec_nmax**: The spectral truncation to be used for defining horizontal random fields. For its requirements, chapter 4 should be consulted. In GOWASP–3, the value should be less than 900 due to imprecision in its spectral transform calculations.

- **version_num**: A version number assigned to the output files created. This should be readable as a real value. It is not used in any aspect of calculations to be performed but is for user reference only.

- **version_inf**: A character string that contains some reference notes that the user would like associated with these files. It is not used in any aspect of calculations to be performed.

- **sigma_file**: The file path and name containing the appropriate table of standard deviations for errors to be added for this data type. Generally, this is a file created by ***new_R_table_conv.j*** or ***new_R_table_rad.j***.

- **input_path**: The file path specifying the directory holding the files containing the fitted correlation parameters that are to used by this program. These are the files produced by either ***eparams_rad_hcorr.j*** and ***eparams_rad_chcorr.j***, or by ***eparams_conv_hcorr.j***.

The above list is followed by **itypes_corr** groups of 7 lines each containing instructions for each group of observation sub-types. For radiance types, these sub-types are typically groups of satellite platforms for this instrument that are to be treated using the same random fields. For conventional observations these groups are denoted by sets of different PREPBUFR **kx** values. These lines, in required order, are

- **type**: An integer from the sequence 1 to **itypes_corr** denoting the group. This value is for reference only and is not used by the program.

- **ks_list**: A list of 10 integers, separated by blanks, denoting the observation data sub-types to be included in this group. For conventional data sub-types, this should be a list of **kx** values. For radiance data types, except for AIRS or AMSUAAQUA, this should be a list of satellite identification numbers, specifically BUFR or GENRADTXT **SAID** numbers. For AIRS or AMSUAAQUA, this should be the BUFR **SIID** numbers for these instruments (420 and 570, respectively). If less than 10 sub-types are to be included for this group, these should be listed first, with any remaining values set to 0 if the list is for conventional data subtypes or 999 for radiance sub-types.

- **v_lengths**: A real number denoting the vertical correlation length (units of km) to be used for this group. This only applies to conventional or GPSRO observations. Otherwise set its value to 0.

- **fix_params**: A set of 4 real numbers (s_1, s_2, s_3, s_4) used for possibly modifying the correlation parameters read by this program for this group. Specifically it will modify the parameters as

$$\nu_k(out) = \nu_k(\mathrm{i}n)s_1 + s_2 \ , \qquad (7.4.1)$$

$$L_k(out) = L_k(\mathrm{i}n)s_3 + s_4 \ , \qquad (7.4.2)$$

  where the index k refers to values for different channels or pressure levels. Note that the same modification values applies to all indexes k for this group. The set of values 1., 0., 1., 0. results in no modifications to the input correlation parameters.

- **file_vcov**: The name of a file containing the matrix to be used to specify channel covariances for creating correlated radiance observation errors. If no channel correlations are to be modeled or this is not for radiance data types, this should be specified as "none". If a file is named, it must be in the directory denoted by the value specified for **input_path**. The file expected is one created by ***eparams_rad_chcorr.j***

- **file_param1**: The name of a file containing the estimated horizontal correlation parameters to be input for this group. If no horizontal correlations are to be considered, this should be specified as "none". If a file is named, it must be in the directory denoted by the specified value for **input_path**. The file expected is one created by ***eparams_rad_hcorr.j*** or ***eparams_conv_hcorr.j***.

- **file_param2**: This is like **file_param1**, allowing a second file also to be input. If not "none". then the parameter values used will be an average of those from the 2 files. Specifically, this is intended for groups of parameters estimated for separate wind components, so that the resulting correlation parameters are based on both.

After the program is run as instructed by these setup files, an **error.rc** file must be composed to point to the created binary files and the desired files of error tables.

7.4.5 Treatment of combined horizontal and vertical or channel correlations

When correlations are both horizontally correlated and either vertically or channel correlated, complications arise. Primarily, in this case the horizontal correlations need to be modeled in terms of correlations of coefficients of principal components of the vertical or channel covariance matrix. The present tuning algorithm instead determines horizontal correlations of innovations only for individual channels or narrow pressure layers. Since realistic observations of radiances or AMVs do not generally exist for all channels or pressure layers due to, for example, cloud effects on observation quality or presence, it is not possible to project such observations onto these principal components. Thus, horizontal correlations of PC coefficients cannot be determined for some types of real data. Thus, the target statistics cannot be directly determined.

The horizontal covariance between innovations at the same channel or level may be expressed as a weighted sum of horizontal covariances between PC coefficients. The weights are squares of structural elements of the PCs. Shapes of the two sets of horizontal correlations can therefore differ greatly. Relationships between parameters characterizing the two sets of correlations may be even more complicated.

Great simplification occurs if the horizontal correlations between PC coefficients are specified as all identical. In that case, the horizontal correlations for all channels or vertical levels are also all identical and the same as for the coefficients. For this reason it is therefore recommended that observation types whose simulated errors are to be both horizontally and either channel or layer correlated be modeled using horizontal functions that are level, channel, and PC independent. This applies not just to chosen shape but to function parameters as well.

To follow this recommendation using GOWASP–3, it is easiest to choose some mean or representative values from the tables of error fitting parameters by simple visual inspection. The chosen shape and ν, L_h parameters should then be set in the ***hc_setup_*.txt*** file for this observation type by specifying its 4 **fix_params** values as as $0, \nu, 0, L_h$. This will set parameters for all channels or layers identically. The tuning software will continue to work properly because the shape and parameters for PC coefficients or individual channels or levels will be the same. In particular, $C^o_{k,i}$ for use in (7.3.2) will be determined properly.

Following this recommendation means that notable variations in correlations for innovations of real data may not be reproduced well for the simulated innovations with added errors. For example, if real innovations display different functional shapes or length scales at different pressure levels, the simulated ones may not. If the mismatch between real and OSSE correlations is too great due to this approximation, differing values may be specified but then tuning must be performed using a different algorithm or guesswork and not trusting the automated programs provided within GOWASP. In particular, $C^o_{k,i}$ required by (7.3.2) will not be properly computed.

7.5 Issues to Consider

In order for the variances of observation innovations for real and OSSE contexts to match, for most observation types it is trivial to simply add and tune random uncorrelated error. Correlated parts of the error are more difficult to define, estimate, and simulate although their effects may be greater. In particular, they require definitions of shapes and two or more parameters, such as length scales and fractions of error variance for the correlated parts. Their computationally reasonable simulation may also require unwarranted assumptions about statistical homogeneity and isotropy.

One observation type for which innovation variances may be difficult to match are conventional surface pressure observations. For some unexplained reason, even without simulated errors added, variances for these innovations can be larger for the OSSE than for the real context. Such results have been obtained both at the GMAO and NCEP using either the G5NR–1 or 2006 ECMWF NR.

In the context of real observations, without further assumptions, the portions of contributions by background errors and observation errors to the observation innovation variances remain ambiguous. Even if tuning succeeds in matching corresponding innovation variances in real data and OSSE contexts, the mix of the two contributions may be incorrect. If the OSSE has diminished background error variance, as may happen if model formulation error is too weak in the OSSE, tuning to match innovation variances would produce observation error variances that are larger than real data ones. If the OSSE DAS responds differently to observation and background errors, the response will be unrealistic regardless of the match. It may even be undesirable to match innovation variances if it is known that the OSSE background error variances are unrealistically weak.

The question of whether or how closely to match particular statistics is not limited to variances. The same question can be asked regarding correlations and covariances. The question may also become less clear as more statistical details or the numbers of tuning parameters are increased.

If it is desired to add spatially correlated observation errors so that spatial covariances of observation innovations match, there are several issues to consider. One that is not so clear is that matching covariances at some separation distances may be more important than at some others. If there are extremely few observation pairs at close separations, matching covariances at short separations may be irrelevant. Similarly, matching at long separations at which a DAS assumes background error correlations are effectively zero may also be unimportant.

Assigning realistic fractions of variances of correlated to total observation errors also presents the difficulty of estimating those fractions from the observation innovations. These latter fractions are estimated by extrapolating spatial covariances to 0 separation distances from pairs of innovations having small separation distances. For real observations, covariances at small non–zero separations are generally unreliable due to poor sample sizes. Thus, covariances measured at longer separations must be used for the extrapolation, requiring a longer distance to extrapolate. The extrapolations are also very sensitive to the choice of analytical extrapolation function chosen for the fitting. So, although the procedure is conceptually simple, it is not so in practice.

Appendix A

Subroutine Calling Sequences

For each main program in the GOWASP–3 package, an abbreviated list of subroutines called is provided here. This includes those called from within called subroutines themselves, as indicated by the line indentations. The subroutines are listed in the order of their first appearances in the calling routines. Calls to BUFLIB, MPI, NETCDF, Shmem, or standard FORTRAN routines or functions are not listed.

If the call to a subroutine appears multiple times within a calling routine, it it is only listed once, but marked with an "M" in the second column. If the call appears only once, but is within a loop that may cause repeated calls, it is marked with an "L". If it appears multiple times and within one or more loops, it is marked with an "ML". Subroutines that are called to read resource files, allocate primary arrays, open but not read files, or determine computational parameters are marked with an "S". Ones used for "clean up only" (e.g., deallocate arrays) are marked with a "C". Those for only printing some information to the log file are marked with a "P".

The location and name of the file containing each subroutine is listed in the final column.

A.1 Program create_conv

This program creates simulated conventional observations prior to adding any explicitly simulated errors. Conventional observations include all those provided and currently used in the NCEP

PREPBUFR file, particularly those from radiosondes, dropsondes, aircaft, station data, ships, buoys, profilers, radars, and scatterometers. Generally, however, simulated AMVs (SATWIND) are created by a different program. (The program create_conv can be used for such a purpose, but then the AMV observation locations would be where real observations are located as indicated in the input PREPBUFR file for real observations, irrespective of whether trackable cloud or water vapor features exist in the NR fields.) The program resides in the directory ***Sim_conv***.

```
mpi_die                           M   Lib_shmem1/m_die.F90
nr_fields_setup                   S   Lib_basic1/m_nr_fields_info.f90
  nr_fields_read_rc               S   Lib_basic1/m_nr_fields_info.f90
    find_name_2                       Lib_basic1/find_name_subs.f90
    read_akbk                         Lib_basic1/read_akbk.f90
  nr_fields_print_info            P   Lib_basic1/m_nr_fields_info.f90
conv_names_setup                  S   Lib_obsrw1/m_conv_names.f90
  find_name                           Lib_basic1/find_name_subs.f90
shmem_fields_setup                S   Sim_conv/m_shmem_fields.F90
conv_obs_setup                    S   Sim_conv/m_conv_obs.F90
conv_rw_setup                     S   Lib_obsrw1/m_bufr_conv.f90
  conv_rw                             Lib_obsrw1/m_bufr_conv.f90
conv_obs_read_bufr                    Sim_conv/m_conv_obs.F90
  conv_rw                             Lib_obsrw1/m_bufr_conv.f90
conv_obs_determine_nobs           S   Sim_conv/m_conv_obs.F90
date_and_time                     ML  Lib_basic1/m_time_compute.f90
shmem_fields_read                 ML  Sim_conv/m_shmem_fields.F90
conv_obs_in_tslot                 L   Sim_conv/m_conv_obs.F90
time_compute_new_cdtime           ML  Lib_basic1/m_time_compute.f90
  time_compute_unpack                 Lib_basic1/m_time_compute.f90
  time_compute_add                    Lib_basic1/m_time_compute.f90
  time_compute_pack                   Lib_basic1/m_time_compute.f90
conv_obs_process                  ML  Sim_conv/m_conv_obs.F90
  sonde_drift_wind                    Sim_conv/m_conv_types.f90
    get_interp_horiz_index        ML  Lib_basic1/interpolate_subs.f90
    get_interp_time_weights       ML  Lib_basic1/interpolate_subs.f90
    horiz_interp_2dfld            ML  Sim_conv/m_shmem_fields.F90
    interp_time                   ML  Lib_basic1/interpolate_subs.f90
    compute_pk                    ML  Lib_basic1/interpolate_subs.f90
    get_interp_vert_index         ML  Lib_basic1/interpolate_subs.f90
    polar_winds                   L   Sim_conv/m_conv_types.f90
      get_interp_horiz_index      L   Lib_basic1/interpolate_subs.f90
      horiz_interp_3dfld          ML  Sim_conv/m_shmem_fields.F90
      horiz_interp_2dfld          ML  Sim_conv/m_shmem_fields.F90
      interp_time                 ML  Lib_basic1/interpolate_subs.f90
      interp_vert                 ML  Lib_basic1/interpolate_subs.f90
    sonde_drift_location          L   Sim_conv/sonde_subs.f90
    horiz_interp_3dfld            ML  Sim_conv/m_shmem_fields.F90
    hydros_z                      L   Lib_basic1/interpolate_subs.f90
    interp_hydros_z               L   Lib_basic1/interpolate_subs.f90
    interp_hydros_p               L   Lib_basic1/interpolate_subs.f90
    interp_vert                   ML  Lib_basic1/interpolate_subs.f90
  sonde_drift_mass                    Sim_conv/m_conv_types.f90
    get_interp_horiz_index        L   Lib_basic1/interpolate_subs.f90
    get_interp_time_weights       L   Lib_basic1/interpolate_subs.f90
    horiz_interp_2dfld            ML  Sim_conv/m_shmem_fields.F90
    interp_time                   ML  Lib_basic1/interpolate_subs.f90
    compute_pk                    L   Lib_basic1/interpolate_subs.f90
    get_interp_vert_index         L   Lib_basic1/interpolate_subs.f90
    horiz_interp_3dfld            ML  Sim_conv/m_shmem_fields.F90
    interp_vert                   ML  Lib_basic1/interpolate_subs.f90
  multi_level_report                  Sim_conv/m_conv_types.f90
    get_interp_horiz_index            Lib_basic1/interpolate_subs.f90
    get_interp_time_weights           Lib_basic1/interpolate_subs.f90
    horiz_interp_2dfld            ML  Sim_conv/m_shmem_fields.F90
    interp_time                   ML  Lib_basic1/interpolate_subs.f90
    compute_pk                        Lib_basic1/interpolate_subs.f90
    horiz_interp_3dfld            ML  Sim_conv/m_shmem_fields.F90
    hydros_z                          Lib_basic1/interpolate_subs.f90
    get_interp_vert_index         ML  Lib_basic1/interpolate_subs.f90
    interp_hydros_z               L   Lib_basic1/interpolate_subs.f90
    interp_hydros_p               L   Lib_basic1/interpolate_subs.f90
    interp_vert                   ML  Lib_basic1/interpolate_subs.f90
  surface_level_report                Sim_conv/m_conv_types.f90
    get_interp_horiz_index            Lib_basic1/interpolate_subs.f90
    get_interp_time_weights           Lib_basic1/interpolate_subs.f90
    horiz_interp_2dfld            M   Sim_conv/m_shmem_fields.F90
    interp_time                   M   Lib_basic1/interpolate_subs.f90
  single_level_report                 Sim_conv/m_conv_types.f90
    get_interp_horiz_index            Lib_basic1/interpolate_subs.f90
    get_interp_time_weights           Lib_basic1/interpolate_subs.f90
    horiz_interp_2dfld                Sim_conv/m_shmem_fields.F90
    interp_time                   M   Lib_basic1/interpolate_subs.f90
    compute_pk                    M   Lib_basic1/interpolate_subs.f90
    horiz_interp_3dfld                Sim_conv/m_shmem_fields.F90
    hydros_z                          Lib_basic1/interpolate_subs.f90
    get_interp_vert_index         M   Lib_basic1/interpolate_subs.f90
    interp_hydros_z                   Lib_basic1/interpolate_subs.f90
    interp_hydros_p               M   Lib_basic1/interpolate_subs.f90
    interp_vert                   M   Lib_basic1/interpolate_subs.f90
conv_obs_output                   M   Sim_conv/m_conv_obs.F90
  sondes_output                   L   Sim_conv/sonde_subs.f90
    sonde_report_wind                 Sim_conv/sonde_subs.f90
      siglevs_sort                M   Sim_conv/sonde_subs.f90
      siglevs_mandatory               Sim_conv/sonde_subs.f90
      siglevs_linear              M   Sim_conv/sonde_subs.f90
        siglevs_sort                  Sim_conv/sonde_subs.f90
    sonde_report_mass                 Sim_conv/sonde_subs.f90
      siglevs_sort                M   Sim_conv/sonde_subs.f90
      transform_q_rh              M   Lib_basic1/interpolate_subs.f90
      siglevs_mandatory               Sim_conv/sonde_subs.f90
      siglevs_trop                    Sim_conv/sonde_subs.f90
        siglevs_slope             M   Sim_conv/sonde_subs.f90
      siglevs_linear              M   Sim_conv/sonde_subs.f90
        siglevs_sort                  Sim_conv/sonde_subs.f90
    siglevs_qc                        Sim_conv/sonde_subs.f90
      siglevs_range_check             Sim_conv/sonde_subs.f90
    conv_rw                           Lib_obsrw1/m_bufr_conv.f90
    conv_print_sample             P   Lib_obsrw1/m_bufr_conv.f90
  non_sonde_output                L   Sim_conv/sonde_subs.f90
    siglevs_qc                        Sim_conv/sonde_subs.f90
    conv_rw                           Lib_obsrw1/m_bufr_conv.f90
    conv_print_sample             P   Lib_obsrw1/m_bufr_conv.f90
  summary_counts                  P   Sim_conv/sonde_subs.f90
shutdown                          C   Sim_conv/create_conv.F90
  shmem_fields_clean              C   Sim_conv/m_shmem_fields.F90
  conv_obs_clean                  C   Sim_conv/m_conv_obs.F90
```

A.2 Program create_gpsro

This program creates simulated GPSRO bending angles using software provided by the Radio Occultation Meteorology (ROM) Satellite Applications Facility (SAF) of EUMETSAT. It resides in the directory ***Sim_gpsro***.

```
mpi_die                           M   Lib_shmem1/m_die.F90
nr_fields_setup                   S   Lib_basic1/m_nr_fields_info.f90
  nr_fields_read_rc               S   Lib_basic1/m_nr_fields_info.f90
    find_name_2                       Lib_basic1/find_name_subs.f90
    read_akbk                         Lib_basic1/read_akbk.f90
  nr_fields_print_info                Lib_basic1/m_nr_fields_info.f90
gpsro_names_setup                 S   Lib_obsrw1/m_gpsro_names.f90
  find_name                           Lib_basic1/find_name_subs.f90
gpsro_fields_setup                S   Sim_gpsro/m_gpsro_fields.F90
gpsro_obs_setup                   S   Sim_gpsro/m_gpsro_obs.F90
gpsro_ropp_setup                  S   Sim_gpsro/m_gpsro_ropp.f90
  gpsro_ropp_alloc_strucs         S   Sim_gpsro/m_gpsro_ropp.f90
gpsro_rw_setup                    S   Lib_obsrw1/m_bufr_gpsro.f90
gpsro_obs_read_bufr                   Sim_gpsro/m_gpsro_obs.F90
  gpsro_rw                        L   Lib_obsrw1/m_bufr_gpsro.f90
    gpsro_rw_bang                 M   Lib_obsrw1/m_bufr_gpsro.f90
    time_compute_unpack           M   Lib_basic1/m_time_compute.f90
    time_compute_dhours               Lib_basic1/m_time_compute.f90
    gpsro_check_obs                   Lib_obsrw1/m_bufr_gpsro.f90
    time_compute_add                  Lib_basic1/m_time_compute.f90
gpsro_obs_determine_nobs              Sim_gpsro/m_gpsro_obs.F90
```

```
date_and_time                     ML  Lib_basic1/m_time_compute.f90
gpsro_fields_read                 ML  Sim_gpsro/m_gpsro_fields.F90
  read_data_2d                    M   Sim_gpsro/m_gpsro_fields.F90
  read_data_3d                    M   Sim_gpsro/m_gpsro_fields.F90
gpsro_obs_in_tslot                L   Sim_gpsro/m_gpsro_obs.F90
time_compute_new_cdtime           M   Lib_basic1/m_time_compute.f90
  time_compute_unpack                 Lib_basic1/m_time_compute.f90
  time_compute_add                    Lib_basic1/m_time_compute.f90
  time_compute_pack                   Lib_basic1/m_time_compute.f90
gpsro_obs_process                 L   Sim_gpsro/m_gpsro_obs.F90
  gpsro_ropp_sequence                 Sim_gpsro/m_gpsro_ropp.f90
    gpsro_ropp_plane_grid             Sim_gpsro/m_gpsro_ropp.f90
      ropp_fm_2d_plane                Sim_gpsro/ropp_fm_2d_plane.f90
    gpsro_ropp_plane_values           Sim_gpsro/m_gpsro_ropp.f90
      get_interp_time_weights         Lib_basic1/interpolate_subs.f90
      get_interp_horiz_index      L   Lib_basic1/interpolate_subs.f90
      horiz_interp_2dfld          ML  Sim_gpsro/m_gpsro_fields.F90
      interp_time                 ML  Lib_basic1/interpolate_subs.f90
      horiz_interp_3dfld          ML  Sim_gpsro/m_gpsro_fields.F90
      compute_pk                  L   Lib_basic1/interpolate_subs.f90
      hydros_phis_tq              L   Lib_basic1/interpolate_subs.f90
    gpsro_ropp_set_impp               Sim_gpsro/m_gpsro_ropp.f90
    ropp_fm_bangle_2d                 Sim_gpsro/ropp_fm_bangle_2d.f90
      ropp_fm_compress_2d             Sim_gpsro/ropp_fm_compress_2d.f90
      ropp_fm_alpha2drk               Sim_gpsro/ropp_fm_alpha2drk.f90
    gpsro_ropp_get_bbang              Sim_gpsro/m_gpsro_ropp.f90
gpsro_obs_output                  L   Sim_gpsro/m_gpsro_obs.F90
  gpsro_rw                        L   Lib_obsrw1/m_bufr_gpsro.f90
shutdown                          C   Sim_gpsro/create_gpsro.F90
  gpsro_fields_clean              C   Sim_gpsro/m_gpsro_fields.F90
  gpsro_obs_clean                 C   Sim_gpsro/m_gpsro_obs.F90
  gpsro_ropp_clean                C   Sim_gpsro/m_gpsro_ropp.f90
```

A.3 Program create_satwind

This program creates simulated SATWINDS (AMVs) as would be obtained from existing polar orbiters and geostationary satellites. Rather than attempt to use coarse NR fields as images, however, observation locations are based on probabilities of an observation existing given cloud fractions and integrated precipitable water gradients. Distributions of observations in time for various satwind types are determined from examining time distributions for corresponding real observations. This program resides in the directory ***Sim_satwind***.

```
mpi_die                               Lib_shmem1/m_die.F90
nr_fields_setup                   S   Lib_basic1/m_nr_fields_info.f90
  nr_fields_read_rc               S   Lib_basic1/m_nr_fields_info.f90
    find_name_2                       Lib_basic1/find_name_subs.f90
    read_akbk                         Lib_basic1/read_akbk.f90
  nr_fields_print_info                Lib_basic1/m_nr_fields_info.f90
satwind_locations_read            S   Sim_satwind/m_satwind_locations.f90
  satwind_locations_set           S   Sim_satwind/m_satwind_locations.f90
  satwind_locations_hdr           L   Sim_satwind/m_satwind_locations.f90
    time_compute_unpack               Lib_basic1/m_time_compute.f90
    time_compute_dhours               Lib_basic1/m_time_compute.f90
time_compute_new_cdtime               Lib_basic1/m_time_compute.f90
  time_compute_unpack                 Lib_basic1/m_time_compute.f90
  time_compute_add                    Lib_basic1/m_time_compute.f90
  time_compute_pack                   Lib_basic1/m_time_compute.f90
read_shmem_data_3d                ML  Sim_satwind/create_satwind.F90
  find_name                           Lib_basic1/find_name_subs.f90
  set_field_file_name                 Lib_basic1/set_file_name.f90
read_shmem_data_2d                ML  Sim_satwind/create_satwind.F90
  find_name                           Lib_basic1/find_name_subs.f90
  set_field_file_name                 Lib_basic1/set_file_name.f90
compute_ipw                       L   Sim_satwind/create_satwind.F90
consider_points                       Sim_satwind/create_satwind.F90
   satwind_all_setup              S   Sim_satwind/m_satwind.f90
     find_name                    M   Lib_basic1/find_name_subs.f90
   compute_lon_stride             L   Lib_basic1/compute_lon_stride.f90
   get_neighbors                  L   Sim_satwind/create_satwind.F90
   satwind_all                    L   Sim_satwind/m_satwind.f90
     satwind_geost                    Sim_satwind/m_satwind.f90
       satwind_gcdist             L   Sim_satwind/m_satwind.f90
       satwind_view_clouds        L   Sim_satwind/m_satwind.f90
         get_solar_elevation          Lib_basic1/sub_zensun.f90
           zensun                     Lib_basic1/sub_zensun.f90
       satwind_view_wvapor        L   Sim_satwind/m_satwind.f90
     satwind_polar                    Sim_satwind/m_satwind.f90
       satwind_polar_loc_table        Sim_satwind/m_satwind.f90
       satwind_view_clouds            Sim_satwind/m_satwind.f90
       satwind_view_wvapor            Sim_satwind/m_satwind.f90
   satwind_assign_bufr            L   Sim_satwind/m_satwind.f90
satwind_count_setup               S   Sim_satwind/m_satwind_count.f90
satwind_count_add                 L   Sim_satwind/m_satwind_count.f90
bufr_satwind_write                L   Sim_satwind/bufr_satwind_write.f90
satwind_count_print               P   Sim_satwind/m_satwind_count.f90
shutdown                          C   Sim_satwind/create_satwind.F90
satwind_locations_clean           C   Sim_satwind/m_satwind_locations.f90
```

A.4 Program create_rad_obs_list

This program reads a BUFR file of radiance data, thins the data set if requested, and writes out a file of observation locations and other header information. It resides in the directory ***Sim_rad***.

```
check_rad_type                    ML  Lib_obsrw1/m_bufr_rad.f90
rad_obs_arrays_setup              S   Lib_obsrw1/m_rad_obs_arrays.f90
open_obs_files                        Lib_obsrw1/open_obs_file.f90
  read_write_gmi_1st_msg              Lib_obsrw1/m_bufr_rad.f90
  read_write_obs_rad              M   Lib_obsrw1/m_bufr_rad.f90
rad_thin_setup                    S   Sim_rad/m_rad_thin.f90
  rad_thin_flds_setup             S   Sim_rad/m_rad_thin_flds.f90
    rad_thin_flds_rc              S   Sim_rad/m_rad_thin_flds.f90
    rad_thin_flds_print_info      P   Sim_rad/m_rad_thin_flds.f90
    rad_thin_flds_read                Sim_rad/m_rad_thin_flds.f90
      set_field_file_name         ML  Lib_basic1/set_file_name.f90
      read_nc4_2dfield            ML  Sim_rad/read_nc4_file.f90
      time_compute_new_cdtime     L   Lib_basic1/m_time_compute.f90
        time_compute_unpack           Lib_basic1/m_time_compute.f90
        time_compute_add              Lib_basic1/m_time_compute.f90
        time_compute_pack             Lib_basic1/m_time_compute.f90
  rad_thin_set_boxes              S   Sim_rad/m_rad_thin.f90
  find_name                       M   Lib_basic1/find_name_subs.f90
read_write_obs_rad                L   Lib_obsrw1/m_bufr_rad.f90
  read_write_obs_aqua                 Lib_obsrw1/m_bufr_rad.f90
  read_write_obs_iasi                 Lib_obsrw1/m_bufr_rad.f90
  read_write_obs_gmi                  Lib_obsrw1/m_bufr_rad.f90
  read_write_obs_cris                 Lib_obsrw1/m_bufr_rad.f90
  read_write_obs_ssmis                Lib_obsrw1/m_bufr_rad.f90
    angles_ssmis                      Lib_basic1/sub_zensun.f90
      zensun                          Lib_basic1/sub_zensun.f90
  read_write_obs_tovs                 Lib_obsrw1/m_bufr_rad.f90
  read_write_obs_genradtxt            Lib_obsrw1/m_bufr_rad.f90
rad_thin_put                      L   Sim_rad/m_rad_thin.f90
  rad_thin_check_hdr                  Sim_rad/m_rad_thin.f90
    time_compute_unpack               Lib_basic1/m_time_compute.f90
    time_compute_dhours               Lib_basic1/m_time_compute.f90
  rad_thin_find_subset
  time_compute_unpack                 Lib_basic1/m_time_compute.f90
  time_compute_add                    Lib_basic1/m_time_compute.f90
  rad_thin_flds_interp                Sim_rad/m_rad_thin_flds.f90
    get_interp_horiz_index        L   Lib_basic1/interpolate_subs.f90
    get_interp_horiz_nearest      ML  Lib_basic1/interpolate_subs.f90
    get_interp_time_nearest       L   Lib_basic1/interpolate_subs.f90
  rad_thin_penalty                    Sim_rad/m_rad_thin.f90
  rad_thin_find_box_id                Sim_rad/m_rad_thin.f90
  rad_thin_box_update                 Sim_rad/m_rad_thin.f90
```

```
print_sample                      PL  Sim_rad/create_rad_obs_list.f90
rad_thin_box_count                    Sim_rad/m_rad_thin.f90
rad_thin_write                        Sim_rad/m_rad_thin.f90
rad_thin_clean                    C   Sim_rad/m_rad_thin.f90
rad_obs_arrays_clean              C   Lib_obsrw1/m_rad_obs_arrays.f90
```

A.5 Program create_rad_profs

This program extracts atmospheric profiles from netcdf files of NR fields for subsequent calculations of radiances. Multiple types (classes) of radiance observations are handled in a single execution. It uses shared memory arrays created by MAPL_ShmemMod; i.e., arrays distributed among all processors and accessible by all using standard FORTRAN without explicitly invoking MPI calls. Files of profiles along with observation header information are output to a binary file. This program resides in the directory ***Sim_rad***.

```
mpi_die                           M   Lib_shmem1/m_die.F90
prof_io_files                         Sim_rad/prof_io_files.f90
  set_field_file_name             M   Lib_basic1/set_file_name.f90
obs_list_types_allocate           S   Sim_rad/m_obs_list.f90
obs_list_setup                    SL  Sim_rad/m_obs_list.f90
  obs_list_read_header            S   Sim_rad/m_obs_list.f90
    obs_list_header_allocate      S   Sim_rad/m_obs_list.f90
  find_name                       M   Lib_basic1/find_name_subs.f90
obs_list_types_put_setup          SL  Sim_rad/m_obs_list.f90
obs_list_print_info               L   Sim_rad/m_obs_list.f90
obs_list_clean                    L   Sim_rad/m_obs_list.f90
obs_list_types_rec_ids                Sim_rad/m_obs_list.f90
nr_fields_setup                   S   Lib_basic1/m_nr_fields_info.f90
  nr_fields_read_rc               S   Lib_basic1/m_nr_fields_info.f90
    find_name_2                       Lib_basic1/find_name_subs.f90
    read_akbk                         Lib_basic1/read_akbk.f90
  nr_fields_print_info                Lib_basic1/m_nr_fields_info.f90
nr_fields_print_info                  Lib_basic1/m_nr_fields_info.f90
obs_list_types_get_setup          ML  Sim_rad/m_obs_list.f90
  obs_list_header_allocate        S   Sim_rad/m_obs_list.f90
write_profiles_header             L   Sim_rad/m_write_profiles.f90
obs_list_clean                    ML  Sim_rad/m_obs_list.f90
time_compute_new_cdtime           ML  Lib_basic1/m_time_compute.f90
obs_list_read_recs                L   Sim_rad/m_obs_list.f90
obs_list_types_put_recs           L   Sim_rad/m_obs_list.f90
rec_list_r_put                    L   Sim_rad/create_rad_profs.f90
read_shmem_data                   ML  Sim_rad/create_rad_profs.f90
```

```
  set_field_file_name                 Lib_basic1/set_file_name.f90
  check                               Sim_rad/create_rad_profs.f90
obs_list_types_get_recs           ML  Sim_rad/m_obs_list.f90
rec_list_r_get                    ML  Sim_rad/create_rad_profs.f90
interpolate_fields_2d             ML  Sim_rad/create_rad_profs.f90
  get_interp_horiz_index              Lib_basic1/interpolate_subs.f90
  get_interp_horiz_nearest        L   Lib_basic1/interpolate_subs.f90
  get_interp_time_nearest         L   Lib_basic1/interpolate_subs.f90
interpolate_fields_3d             ML  Sim_rad/create_rad_profs.f90
  get_interp_horiz_index              Lib_basic1/interpolate_subs.f90
  get_interp_horiz_nearest            Lib_basic1/interpolate_subs.f90
  get_interp_time_nearest             Lib_basic1/interpolate_subs.f90
write_profiles_tv2t               L   Sim_rad/m_write_profiles.f90
write_profiles_o3units            L   Sim_rad/m_write_profiles.f90
write_profiles_recs               L   Sim_rad/m_write_profiles.f90
write_profiles_sample             L   Sim_rad/m_write_profiles.f90
  prof_all_2d3d                       Sim_rad/sub_prof_all_2d3d.f90
write_profiles_close              L   Sim_rad/m_write_profiles.f90
obs_list_clean                    C   Sim_rad/m_obs_list.f90
obs_list_types_clean              C   Sim_rad/m_obs_list.f90
shutdown                          C   Sim_rad/create_rad_profs.f90
```

A.6 Program create_rad_reord

This program reorders profiles from an order based on time slot to one based on observation subtype (e.g., satellite platform). It resides in the directory ***Sim_rad***.

```
read_profiles_setup               ML  Sim_rad/m_read_profiles.f90
  obs_list_read_header                Sim_rad/m_obs_list.f90
write_profiles_header                 Sim_rad/m_write_profiles.f90
read_profiles_cleanup             ML  Sim_rad/m_read_profiles.f90
    obs_list_clean                    Sim_rad/m_obs_list.f90
read_profiles_recs                L   Sim_rad/m_read_profiles.f90
write_profiles_recs               L   Sim_rad/m_write_profiles.f90
```

A.7 Program create_rad_merge

This program merges two profile data sets that have been produced for the same observation set, removing any duplicate information. The orders of observations in the two input data sets must be identical. This program resides in the directory ***Sim_rad***.

```
read_profiles_setup              MS  Sim_rad/m_read_profiles.f90
merge_save_header                    Sim_rad/m_prof_merge.f90
read_profiles_cleanup            MC  Sim_rad/m_read_profiles.f90
  obs_list_clean                 C   Sim_rad/m_obs_list.f90
merge_compare_headers                Sim_rad/m_prof_merge.f90
merge_prof_name_list                 Sim_rad/m_prof_merge.f90
write_profiles_header                Sim_rad/m_write_profiles.f90
merge_allocate_prof_arrays       S   Sim_rad/m_prof_merge.f90
merge_profile_recs                   Sim_rad/m_prof_merge.f90
write_profiles_close             C   Sim_rad/m_write_profiles.f90
```

A.8 Program create_rad_bufr

This program creates radiances from atmospheric profiles (read from a file) at observation locations. For existing observations, these simulated observations are output in a BUFR or GENRADTXT format readable by GSI. This program resides in directory ***Sim_rad***.

```
mympi_setup                      S   Sim_rad/m_mympi.F90
mpi_die                          M   Lib_shmem1/m_die.F90
check_rad_type                       Lib_obsrw1/m_bufr_rad.f90
sat_info_table_read              S   Lib_basic1/m_sat_info_table.f90
read_profiles_setup              S   Sim_rad/m_read_profiles.f90
  obs_list_read_header               Sim_rad/m_obs_list.f90
    obs_list_header_allocate     S   Sim_rad/m_obs_list.f90
rad_obs_arrays_setup             S   Lib_obsrw1/m_rad_obs_arrays.f90
rad_prob_setup                   S   Sim_rad/m_rad_prob.f90
  rad_prob_read_rc                   Sim_rad/m_rad_prob.f90
    find_name_2                  M   Lib_basic1/find_name_subs.f90
  find_name                      M   Lib_basic1/find_name_subs.f90
  rad_prob_print                     Sim_rad/m_rad_prob.f90
rad_index_setup                  S   Sim_rad/m_rad_index.f90
  find_name                      M   Lib_basic1/find_name_subs.f90
```

```
open_obs_files                    S   Lib_obsrw1/open_obs_file.f90
read_write_obs_rad                    Lib_obsrw1/m_bufr_rad.f90
read_profiles_recs                ML  Sim_rad/m_read_profiles.f90
crtm_interface_init               L   Sim_rad/m_crtm_interface.f90
  sat_info_table_get_ic           M   Lib_basic1/m_sat_info_table.f90
  sat_info_table_get_1c           M   Lib_basic1/m_sat_info.f90
  sat_info_table_get_ii               Lib_basic1/m_sat_info_table.f90
  sat_info_table_get_1r           M   Lib_basic1/m_sat_info.f90
  sat_info_table_get_2c               Lib_basic1/m_sat_info_table.f90
time_compute_new_cdtime           L   Lib_basic1/m_time_compute.f90
prof_all_2d3d                     L   Sim_rad/sub_prof_all_2d3d.f90
compute_plevs                     L   Sim_rad/compute_plevs.f90
rad_prob_compute                  L   Sim_rad/m_rad_prob.f90
crtm_interface_comp_rad           L   Sim_rad/m_crtm_interface.f90
  crtm_interface_set_gmi          M   Sim_rad/m_crtm_interface.f90
  crtm_interface_set_clouds           Sim_rad/m_crtm_interface.f90
  crtm_interface_set_aerosols         Sim_rad/m_crtm_interface.f90
copy_rad_obs                      ML  Sim_rad/m_copy_rad_obs.f90
read_write_gmi_1st_msg                Lib_obsrw1/m_bufr_rad.f90
read_write_obs_rad                L   Lib_obsrw1/m_bufr_rad.f90
  read_write_obs_aqua                 Lib_obsrw1/m_bufr_rad.f90
  read_write_obs_iasi                 Lib_obsrw1/m_bufr_rad.f90
  read_write_obs_gmi                  Lib_obsrw1/m_bufr_rad.f90
  read_write_obs_cris                 Lib_obsrw1/m_bufr_rad.f90
  read_write_obs_ssmis                Lib_obsrw1/m_bufr_rad.f90
  read_write_obs_tovs                 Lib_obsrw1/m_bufr_rad.f90
  read_write_obs_genradtxt            Lib_obsrw1/m_bufr_rad.f90
print_sample                      PL  Sim_rad/create_rad_bufr.F90
read profiles_cleanup             C   Sim_rad/m_read_profiles.f90
  obs_list_clean                  C   Sim_rad/m_obs_list.f90
mympi_finalize                    C   Sim_rad/m_mympi.F90
```

A.9 Program create_error

This program creates and adds random observational error to simulated observations. The errors may be correlated in various ways, but only among observations of the same type (class) and subtype. It resides in the directory ***Sim_error***.

```
mpi_die                           M   Lib_shmem1/m_die.F90
sat_info_table_read               S   Lib_basic1/m_sat_info_table.f90
error_table_setup                 S   Lib_basic1/m_obs_error_table.f90
  error_table_read_rc             S   Lib_basic1/m_obs_error_table.f90
    find_name_2                   M   Lib_basic1/find_name_subs.f90
    error_table_default_seed          Lib_basic1/m_obs_error_table.f90
  error_table_read_stdv           S   Lib_basic1/m_obs_error_table.f90
    sat_info_table_get_ii         M   Lib_basic1/m_sat_info_table.f90
    error_table_fix               S   Sim_error/m_obs_error_table.f90
  error_table_read_corr_params    S   Lib_basic1/m_obs_error_table.f90
random_fields_setup               S   Sim_error/m_random_fields.F90
  sh_init_factors                     Sim_error/m_shtrans_rf.f90
    sh_imax                           Sim_error/m_shtrans_rf.f90
    sh_trunc                          Sim_error/m_shtrans_rf.f90
    sh_emns                           Sim_error/m_shtrans_rf.f90
    set_fft                           Sim_error/m_fft.f90
      set99                           Sim_error/m_fft.f90
    sh_init_lats                      Sim_error/m_shtrans_rf.f90
      sh_trig                         Sim_error/m_shtrans_rf.f90
      sh_abfaclats                    Sim_error/m_shtrans_rf.f90
  mpi_die                             Lib_shmem1/m_die.F90
  random_fields_synch_ranf        ML  Sim_error/m_random_fields.F90
  random_fields_spectra           ML  Sim_error/m_random_fields.F90
    random_power_compute              Sim_error/m_random_power.f90
      rp_gauss                        Sim_error/m_random_power.f90
        rp_bssl                       Sim_error/m_random_power.f90
      rp_ems0                         Sim_error/m_random_power.f90
      rp_alp0                     L   Sim_error/m_random_power.f90
    random_gauss_r8               ML  Sim_error/m_random_gauss.f90
  rf_diags_power_calc             L   Sim_error/m_rf_diags_power.f90
    sh_calc_power                     Sim_error/m_shtrans_rf.f90
    rf_diags_power_print          P   Sim_error/m_rf_diags_power.f90
  sh_trans                        L   Sim_error/m_shtrans_rf.f90
    sh_s2fm                           Sim_error/m_shtrans_rf.f90
      sh_s2c_s                    L   Sim_error/m_shtrans_rf.f90
      sh_s2c_v                    L   Sim_error/m_shtrans_rf.f90
      sh_alp                      L   Sim_error/m_shtrans_rf.f90
      sh_c2fm                     L   Sim_error/m_shtrans_rf.f90
    sh_uvcos                          Sim_error/m_shtrans_rf.f90
    run_fft                       L   Sim_error/m_fft.f90
      fft991                          Sim_error/fft99.f90
  sh_clean                        C   Sim_error/m_shtrans_rf.f90
    clean_fft                     C   Sim_error/m_fft.f90
rf_diags_fields_calc                  Sim_error/m_rf_diags_fields.f90
  rf_diags_fields_corrs           L   Sim_error/m_rf_diags_fields.f90
    rf_diags_fields_1_lat         M   Sim_error/m_rf_diags_fields.f90
    rf_diags_fields_get_part      M   Sim_error/m_rf_diags_fields.f90
      rf_diags_fields_1_lat       L   Sim_error/m_rf_diags_fields.f90
        random_fields_get_1_value L   Sim_error/m_random_fields.F90
    rf_diags_fields_print_f       MP  Sim_error/m_rf_diags_fields.f90
    random_fields_get_1_lev       ML  Sim_error/m_random_fields.F90
    rf_diags_fields_int_var           Sim_error/m_rf_diags_fields.f90
    rf_diags_fields_sump          ML  Sim_error/m_rf_diags_fields.f90
    rf_diags_fields_compc         ML  Sim_error/m_rf_diags_fields.f90
    rf_diags_fields_prntc         MP  Sim_error/m_rf_diags_fields.f90
pert_loop_setup                   S   Sim_error/m_pert_loop.f90
  pert_loop_setup_prepbufr        S   Sim_error/m_pert_loop.f90
```

```
    conv_names_setup              S   Lib_obsrw1/m_conv_names.f90
    conv_rw_setup                 S   Lib_obsrw1/m_bufr_conv.f90
  call pert_loop_setup_gpsro      S   Sim_error/m_pert_loop.f90
    gpsro_names_setup             S   Lib_obsrw1/m_gpsro_names.f90
    gpsro_rw_setup                S   Lib_obsrw1/m_bufr_gpsro.f90
  call pert_loop_setup_rad        S   Sim_error/m_pert_loop.f90
    rad_obs_arrays_setup          S   Lib_obsrw1/m_rad_obs_arrays.f90
    open_obs_files                S   Lib_obsrw1/open_obs_file.f90
    find_name                     M   Lib_basic1/find_name_subs.f90
pert_loop_do                          Sim_error/m_pert_loop.f90
  read_write_gmi_1st_msg          M   Lib_obsrw1/m_bufr_rad.f90
  count_subtypes_get_rpts         L   Sim_error/m_count_types.f90
  conv_rw                         ML  Lib_obsrw1/m_bufr_conv.f90
  gpsro_rw                        ML  Lib_obsrw1/m_bufr_gpsro.f90
  read_write_obs_rad              ML  Lib_obsrw1/m_bufr_rad.f90
  pert_loop_copy_values           ML  Sim_error/m_pert_loop.f90
  count_subtypes                  L   Sim_error/m_count_types.f90
  pert_find_type                  L   Sim_error/m_obs_pert.f90
  random_fields_id_type           L   Sim_error/m_random_fields.F90
  rad2bt_cleanup                  DL  Sim_error/m_rad2bt.f90
  rad2bt_setup                    SL  Sim_error/m_rad2bt.f90
  rad2bt_transforms               ML  Sim_error/m_rad2bt.f90
  pert_ps                             Sim_error/m_obs_pert.f90
    pert_interp_stdv              L   Sim_error/m_obs_pert.f90
    random_gauss_r8               L   Sim_error/m_random_gauss.f90
  pert_obs                        L   Sim_error/m_obs_pert.f90
    pert_q2rh                     ML  Sim_error/m_obs_pert.f90
    pert_interp_stdv              L   Sim_error/m_obs_pert.f90
    random_fields_get_values      L   Sim_error/m_random_fields.F90
      random_fields_frac_weights  ML  Sim_error/m_random_fields.F90
    pert_rad_chcorr               L   Sim_error/m_obs_pert.f90
      random_gauss_r8             L   Sim_error/m_random_gauss.f90
    random_gauss_r8               ML  Sim_error/m_random_gauss.f90
    pert_eigen                    ML  Sim_error/m_obs_pert.f90
    pert_normalize_evects         ML  Sim_error/m_obs_pert.f90
    pert_fix_evalues              ML  Sim_error/m_obs_pert.f90
count_subtypes_print              P   Sim_error/m_count_types.f90
obs_error_clean                   C   Sim_error/create_error.F90
  pert_loop_clean                 C   Sim_error/m_pert_loop.f90
    rad_obs_arrays_clean          C   Lib_obsrw1/m_rad_obs_arrays.f90
  error_table_clean               C   Sim_error/m_obs_error_table.f90
  random_fields_clean             C   Sim_error/m_random_fields.F90
```

Appendix B

Lists of Modules and Subroutines

Program drivers, modules and their imbedded subroutines, and other subroutines not defined within modules are listed here. Within each of those sets, the components are listed alphabetically. Brief descriptions of their functions are also provided.

The term "SHMEM" refers to the shared–memory constructs within the Earth System Modeling Framework (ESMF). The term "ROPP" refers to the Radio Occultation Processing Package of Radio Occultation Meteorology (ROM) Satellite Applications Facility (SAF) of EUMETSAT. The term "observation information" refers to meta–data associated with an observation, such as observation time and locations as opposed to values of what the observing instrument is intended to measure. The expression "used by" indicates the program driver for which a module or subroutine is required, which is not necessarily the actual subroutine from which it is called.

B.1 List of Main Programs

- ***create_conv***: Program driver for creating simulated conventional observations.
    - ***shutdown***: Deallocate arrays and terminate MPI in this program.
- ***create_error***: Program driver for creating and adding simulated errors to observations.
    - ***shutdown***: Deallocate arrays and terminate MPI in this program.
- ***create_gpsro***: Program driver for creating simulated GPSRO observations.
    - ***shutdown***: Deallocate arrays and terminate MPI in this program.

- ***create_satwind***: Program driver for creating simulated SATWIND observations from existing viewing platforms, using NR cloud and water vapor fields to determine locations.

  - ***check***: Check status of NETCDF library commands for reading files of NR fields.
  - ***shutdown***: Deallocate some arrays.
  - ***read_shmem_data_2d***: Read 2d field on a NR file.
  - ***read_shmem_data_3d***: Read portion of 3d field on a NR file.
  - ***get_neighbors***: For requested location, get values at neighboring points from global fields.
  - ***compute_ipw***: Compute integrated precipitable water within a specified layer.
  - ***consider_points***: Loop over a subset of horizontal grid points and consider if observations should exist at that location.

- ***create_rad_bufr***: Program driver for creating radiances from profiles (read from a file) of fields at observation locations and outputing them in a BUFR or GENRADTXT format readable by GSI.

  - ***print_sample***: Print sample of output from this program.

- ***create_rad_merge***: Program driver for merging two profile data sets that have been produced for the same observation set, removing duplicate fields.

- ***create_rad_obs_list***: Program driver to read a BUFR or GENRADTXT file of radiance data, possibly thin the data set, and write out a file of just observation locations and other header information.

  - ***print_sample***: Print a sample of output from this program

- ***create_rad_profs***: Program driver for extracting surface data and atmospheric profiles from files of NR fields for subsequent calculations of radiances, allowing mutiple types (classes) of radiance observations to be handled in a single execution.

  - ***check***: Check the return status of a NETCDF command function.
  - ***shutdown***: Deallocate SHMEM arrays.
  - ***read_shmem_data***: Read field on NETCDF file into a SHMEM array.
  - ***interpolate_fields_2d***: Interpolate one or more 2d fields of NR data horizontally and temporally.
  - ***interpolate_fields_3d***: Interpolate a single 3d field of NR data horizontally and temporally.
  - ***rec_list_r_put***: Put all real–valued observation information in all time slots for a single observation type into a common array containing real–valued information for all time slots and all observation types.
  - ***rec_list_r_get***: The reverse of ***rec_list_r_put***, extracting information for a single observation type.

- ***create_rad_reord***: Program driver for reordering profiles from an order based on time slot to an order based on observation subtype (e.g., satellite platform).

B.2 List of Modules and Their Imbedded Subroutines

- ***goss_mpeu***: Used to define some functions and kind parameters used by ***m_mympi***. Used by ***create_rad_bufr***.

- ***m_bufr_conv***: Module to read or write conventional observation data in BUFR format. Used by ***create_conv***, ***create_error***, and ***check_bufr***.

  - ***conv_rw_setup***: Open BUFR files for read and write and set some variables used to read properly.
  - ***conv_rw***: Read or write one conventional observation report (may include values on mutiple pressure levels).
  - ***conv_check_obs***: Check if some parts of observation header appear OK and if the observation subtype there is in the list of subtypes to be considered.
  - ***conv_print_sample***: Print header and observation values for one observation report.
  - ***conv_uv***: Read or write wind observations in NCEP .prepbufr (BUFR) format.
  - ***conv_tq***: Read or write T, q, or p_s observations in NCEP .prepbufr (BUFR) format.
  - ***check_type_conv***: Routine for making data selection of subtypes given the data type requested. (This function statement is not in the module but in the same FORTRAN file containing this module.)

- ***m_bufr_gpsro***: Module to read or write GPSRO observation data in BUFR format. Used by ***create_gpsro***, ***create_error***, and ***check_bufr***.

  - ***gpsro_rw_setup***: Open BUFR file for GPSRO observations and set some variables.
  - ***gpsro_check_observation***: Check that some observation information parameters are in proper ranges.
  - ***gpsro_rw***: Call routine to read, write and check observation for one GPSRO viewing location.
  - ***gpsro_rw_bang***: Use calls to BUFR library to either read or write file of GPSRO bending angle observations.

- ***m_bufr_rad***: Module to read and write radiance observation data in either BUFR or generic radiance text formats. Used by ***create_rad_obs_list***, ***create_rad_bufr***, ***create_error***, and ***check_bufr***.

  - ***read_write_obs_rad*** : Call appropriate routine to read or write BUFR or generic text file of radiance observations.
  - ***read_write_obs_aqua***: Read or write BUFR file of AIRS observations.
  - ***read_write_obs_tovs***: Read or write BUFR file of HIRS2/3/4, AMSUA/B, MSU, MHS, or ATMS observations.
  - ***read_write_obs_iasi***: Read or write BUFR file of IASI observations.
  - ***read_write_obs_cris*** : Read or write BUFR file of CRIS observations.
  - ***read_write_obs_gmi***: Read or write BUFR file of GMI observations.
  - ***read_write_obs_gmi_1st_msg***: Special instructions required for handling GMI BUFR files since, unlike for other radiances, the GMI files contain 2 distinct message types.

  - ***read_write_obs_ssmis***: Read or write BUFR file of SSMIS observations.
  - ***read_write_obs_genradtxt***: Read or write generic text file for radiance observations.
  - ***check_rad_type***: Code for making data selection of subtypes given the data type requested.
  - ***generic_ireadsb***: A generalization of the BUFR library function ireadsb that is suitable for application to either BUFR or generic text files. (This function statement is not in the module but in the same file.)
  - ***generic_ireadmg***: A generalization of the BUFR library function ireadmg that is suitable for application to either BUFR or generic text files. (This function statement is not in the module but in the same file.)

- ***m_conv_names***: Module containing variable names and indexes for observation information, including BUFR header variable names. Used by ***create_conv***, ***create_error***, and ***check_bufr***.

  - ***conv_names_setup***: Specify indexes for conventional observational data array values corresponding to particular information.

- ***m_conv_obs***: Module that uses SHMEM arrays of observation information and observation values. Used by ***create_conv***.

  - ***conv_obs_setup***: Allocate and initialize SHMEM arrays to hold observation information.
  - ***conv_obs_clean***: Deallocate arrays created by ***conv_obs_setup***.
  - ***conv_obs_read_bufr***: Read and save all conventional observations.
  - ***conv_obs_in_tslot***: Count the numbers of observations in each time slot.
  - ***conv_obs_process***: Call appropriate routine to create each type of conventional observation.
  - ***conv_obs_determine_nobs***: Determine the number of observations to be processed.
  - ***conv_obs_output***: Finish processing sonde reports and output all observations.

- ***m_conv_types***: Module for sequencing operations to create observations for each category of conventional observations, as indicated by the public routines herein. Used by ***create_conv***.

  - ***surface_level_report***: Create surface observation (except for aircraft on the ground).
  - ***single_level_report***: Create single–level observation (excluding some surface types).
  - ***multi_level_report***: Create multi–level observation for non–sonde reports.
  - ***sonde_drift_wind***: Create radiosonde, pilot balloon, or dropsonde observations of u, v, z, and p by ascent or descent through the NR atmosphere by considering its wind–blown motion.
  - ***sonde_drift_mass***: Create radiosonde, pilot balloon, or dropsonde observations of T or q at the prescribed locations determined by ***sonde_drift_wind***.
  - ***polar_winds***: Extract a "sample" of u and v winds along a grid latitude adjacent to either pole.

- ***m_copy_rad_obs***: Module containing arrays and variables used to hold observation information for radiance data types. Used by ***create_rad_bufr***.

  - ***rad_obs_arrays_setup*** : Allocate some arrays to hold radiance observation information.

  - ***rad_obs_arrays_clean***: Deallocate arrays declared in ***rad_obs_arrays_setup***

- ***m_count_types***: Module for counting and printing numbers of observations processed for each data subtype. Used by ***create_error*** and ***check_bufr***.

  - ***count_types_setup***: Initialize counting array to 0.
  - ***count_subtypes***: Fill table of subtype counters.
  - ***count_subtypes_get_rpts***: Get number of reports for requested type.
  - ***count_subtypes_print***: Print table of subtype counters.

- ***m_crtm_interface***: Module that interfaces the OSSE observation simulation software with the CRTM, including calls to CRTM initialization routines. Used by ***create_rad_bufr***.

  - ***crtm_interface_init***: Initalize the JCSDA CRTM for a particular radiance instrument on a particular satellite platform.
  - ***crtm_interface_comp_rad***: Fill JCSDA CRTM input values and arrays and call the CRTM to compute brightness temperatures or radiances at a single location for all channels of a single instrument.
  - ***crtm_interface_set_gmi***: Set viewing geometry for GMI using 1 of 2 possible sets of parameters.
  - ***crtm_interface_set_clouds***: Set radiance parameters pertaining to clouds if requested.
  - ***crtm_interface_set_aerosols***: Set radiance parameters pertaining to aerosols if requested.

- ***m_fft***: Module for calling either a fast or slow Fourier transform routine. Used by ***create_error***.

  - ***set_fft***: Set some trigonometric factors used in the Fourier transforms.
  - ***clean_fft***: Deallocate array of trigonometric factors used by Fourier transform.
  - ***run_fft***: Call routine to transform from Fourier coefficients to grid values.

- ***m_gpsro_fields***: Module for reading and using NR fields required by ***create_gpsro***. Used by ***create_gpsro***.

  - ***gpsro_fields_setup***: Allocate space for the global fields using SHMEM.
  - ***gpsro_fields_read***: Call SHMEM reader for requested set of NR fields.
  - ***gpsro_fields_clean***: Deallocate SHMEM arrays.
  - ***horiz_interp_2dfld***: Horizontally interpolate requested 2d fields at 1 or 2 times.
  - ***horiz_interp_3dfld***: Horizontally interpolate requested 3d fields at 1 or 2 times.

- ***m_gpsro_names***: Specify set of variable names and arrays used for GPSRO observational data. Used by ***create_gpsro***, ***create_error***, and ***check_bufr***.

  - ***gpsro_names_setup***: Specify indexes for GPSRO observational data array values corresponding to particular information.

- ***m_gpsro_obs***: Module that defines SHMEM arrays of GPSRO observation information and values and that calls routines to create such simulated observations. Used by ***create_gpsro***.

  - ***gpsro_obs_setup***: Allocate arrays for the observation information and values using SHMEM.

  - ***gpsro_obs_clean***: Deallocate SHMEM arrays holding observations.
  - ***gpsro_obs_read_bufr***: Read all observations in observation BUFR file and determine the time slot in which each observation falls.
  - ***gpsro_obs_in_tslot***: Count the numbers and set indexes of those observations in each time slot.
  - ***gpsro_obs_process***: Call ROPP software sequence to compute bending angles within each occultation observing plane.
  - ***gpsro_obs_determine_nobs***: Determine the number of observations to be processed.
  - ***gpsro_obs_output***: Output all GPSRO observations.

- ***m_gpsro_ropp***: Module to interface with ROPP routines. Used by ***create_gpsro***.

  - ***gpsro_ropp_setup***: Set some variables and allocate memory used by ROPP.
  - ***gpsro_ropp_sequence***: Call sequence of routines to create a propagation plane and bending angles within it.
  - ***gpsro_ropp_plane_values***: Interpolate NR values to a GPS propagation plane.
  - ***gpsro_ropp_plane_grid***: Determine grid locations defining a GPS propagation plane.
  - ***gpsro_ropp_set_impp***: Copy impact parameters to use real observation locations in the vertical.
  - ***gpsro_ropp_get_bbang***: Copy bending angles from ROPP arrays to observation value arrays.
  - ***gpsro_ropp_alloc_strucs***: Allocate some arrays used by ROPP.
  - ***gpsro_ropp_clean***: Deallocate arrays allocated in ***gpsro_ropp_alloc_strucs***.

- ***m_kinds***: Set parameters denoting precisions of real variables. Used by all programs.

- ***m_mympi***: Includes generalized interfaces for MPI commands like gather and scatter allowing for different types of FORTAN variables. Used by ***create_rad_bufr***.

- ***m_nr_fields_info***: Module to read resource file describing field specifications and to set up some parameters. Used by almost all programs.

  - ***nr_fields_setup***: Gather and set information required to read and use NR field information.
  - ***nr_fields_read_rc***: Read resource file describing properties and selection of required NR fields.
  - ***nr_fields_print_info***: Print some information extracted from the resource file that describes field specifications.

- ***m_obs_error_table***: Module for reading and computing error parameters from 3 files: resource file, table of standard deviations, and correlation parameters. Used by ***create_error*** and programs used to tune observation errors.

  - ***error_table_clean***: Deallocate error table arrays allocated in ***error_table_read_var***.
  - ***error_table_find_corr_id***: Determine the index for the set of horizontal correlation parameters that should be used, if any. Also determine the number of subtypes that will use the same random fields. (The latter is required for rotating the random field to reduce the correlation among different subtypes in each group.)

- ***error_table_find_stdv_id***: Determine the index for the desired subtype in list of subtypes in extracted error tables.
- ***error_table_fix***: Change some units and large values in tables of error standard deviations.
- ***error_table_read_rc***: Read resource file for obtaining error parameters and error table or correlation file names.
- ***error_table_default_seed***: Create a default value of **iseed_dtype** based on the characters in the observation type name.
- ***error_table_read_stdv***: Read table of error standard deviations for a desired data type.
- ***error_table_read_params***: Read file containing parameters and statistics required for specifying the random fields. Separate files are required for each data type. (All conventional observations are handled by a single file with up to 5 different sets of parameter specifications.)

- ***m_obs_list***: Module to set up and read observation list file created by ***create_rad_obs_list***, including file header information and allowing for multiple radiation observation types to be stored simultaneously. Used by ***create_rad_bufr*** , ***create_rad_profs***, and ***create_merge***.

  - ***obs_list_setup***: Create variables and arrays required for storing observation header information for an observation type, based on information in the file header.
  - ***obs_list_read_header***: Read header information from an observation list file.
  - ***obs_list_read_recs***: Read records of data for all observations of a particular observation type and time slot from an observation list file.
  - ***obs_list_print_information***: Print some information read from the header of an observation list file.
  - ***obs_list_clean***: Deallocate arrays used to pass information between this module and other modules for a single observation data type.
  - ***obs_list_header_allocate***: Allocate arrays to contain file header information.
  - ***obs_list_types_allocate***: Allocate array to contain type–dependent observation information local to the module ***m_obs_list***.
  - ***obs_list_types_clean***: The reverse of ***obs_list_types_allocate***.
  - ***obs_list_types_put_setup***: Copy observation list file header information values from a private, type–dependent array to a public array to be used by other modules. Each call to this routine concerns a particular observation type and time slot. Also allocate some arrays based on specific size requirement for this observation type and time slot.
  - ***obs_list_types_get_setup***: The reverse of ***obs_list_types_put_setup***.
  - ***obs_list_types_put_recs***: Copy all observation reports from a private, type–dependent array to a public array to be used by other modules for a particular observation type and time slot.
  - ***obs_list_types_get_recs***: The reverse of ***obs_list_types_put_recs***.
  - ***obs_list_types_rec_ids***: Determine variables to be used as dimensions and indexes for the arrays **rec_list_**∗. These are required to accomodate use of different observation types having different observation meta information and array requirements within a single program execution.

- ***m_obs_pert***: Module to create random perturbations to serve as simulated observation errors. Used by ***create_error***.

  - ***pert_eigen***: Call FORTRAN library routine to compute eigenvalues and eigenvectors of a real symmetric matrix (stored as an array of 8–byte values).

  - ***pert_find_itype***: Find index of desired subtype in list of subtypes found in the observation error tables.
  - ***pert_fix_evalues***: Reset small (relative to largest values) or negative eigenvalues to a small positive value.
  - ***pert_interp_stdv***: Vertically interpolate standard deviations from a table defined on p–levels to an observation level.
  - ***pert_normalize_evects***: Normalize each eigenvector such that the sum of its squared components is 1.
  - ***pert_obs***: Compute and add random perturbations to observations.
  - ***pert_ps***: Compute and add random perturbations to surface pressure.
  - ***pert_print_matrix***: Print matrix.
  - ***pert_print_vector***: Print vector.
  - ***pert_q2rh***: Convert q to relative humidity, or the reverse.
  - ***pert_rad_chcorr***: Create channel correlated errors at a single location.

- ***m_parameters***: Set some physical parameters and mathematical constants. Used by many programs.

- ***m_pert_loop***: Module that loops through messages and reports on a BUFR or generic radiance text file and calls appropriate routines to add simulated errors. Used by ***create_error***.

  - ***pert_loop_setup***: Calls routines to set some variables and open read and write BUFR files for a requested observation type.
  - ***pert_loop_setup_prepbufr***: Set some variables and open read and write BUFR files for conventional observations in PREPBUFR formats.
  - ***pert_loop_setup_gpsro***: Set some variables and open read and write BUFR files for GPSRO observations.
  - ***pert_loop_setup_rad***: Set some variables and open read and write BUFR files for radiance observations.
  - ***pert_loop_do***: Within a do–loop over all observations in a file, sequence calls to read the original observations, create and add perturbations, and write new observations.
  - ***pert_loop_copy_values***: Copy observation values and observation headers between arrays used by ***create_error*** routines and observation read or write routines.
  - ***pert_loop_clean***: Deallocate arrays allocated in ***m_rad_obs_arrays***.

- ***m_prof_merge***: Module containing routines and arrays used to merge 2 observation location profile files containing different sets of fields, removing any duplicates. Used by ***create_rad_merge***.

  - ***merge_profile_recs***: Merge 2 sets of profiles.
  - ***merge_allocate_prof_arrays***: Allocate arrays required to hold profiles for 2 input files and the merged set.
  - ***merge_compare_headers***: Compare observation information headers to ensure that the sets of profiles to merge are for the same observation.
  - ***merge_save_header***: Merge 2 headers of NR field profile files.

  - ***merge_prof_name_list***: Merge field name lists from 2 file headers, removing any duplicates.

- ***m_rad_index***: Module for setting indexes to denote locations of required fields in NR profile arrays input to the CRTM interface ***m_interface_crtm***. Used by ***create_rad_bufr***.

  - ***rad_index_setup***: Set indexes used by ***m_crtm_interface*** to point to particular values held in common arrays, based on names of fields or values in a file of NR extracted profiles.

- ***m_rad_obs_arrays***: Module containing arrays and variables used to hold observation information for radiance data types. Used by ***create_rad_obs_list***, ***create_rad_bufr***, ***create_error***, and ***check_bufr***.

  - ***rad_obs_arrays_setup***: Allocate some arrays to hold radiance observation information.
  - ***rad_obs_arrays_clean***: Deallocate arrays declared in ***rad_obs_arrays_setup***.

- ***m_rad_prob***: Determine effective radiative surface for radiance observations due to the presence of clouds or precipitation by using a probability function based on some NR field value. Also set field names used to compute effects of hydrometeors or aerosols. Used by ***create_rad_bufr***.

  - ***rad_prob_setup***: Set parameters that will be used for determination of cloud, precipitation, or aerosol effects on radiances.
  - ***rad_prob_read_rc***: Read resource file for instructions on computing radiances that depend on instrument type, particularly regarding cloud and precipitation effects.
  - ***rad_prob_print***: Print some information read from the radiance probability resource file.
  - ***rad_prob_compute***: Compute effective radiative surface and an observation quality indicator based on the probability of radiances being obstructed by clouds or other fields.

- ***m_rad2bt***: Module to change between radiances and brightness temperatures using the CRTM. Used by ***create_error***.

  - ***rad2bt_setup***: Set up CRTM to use proper spec and tau coefficients for transforms between radiances and T_B.
  - ***rad2bt_cleanup***: The reverse of ***rad2bt_setup***.
  - ***rad2bt_transforms***: Transforms back and forth between radiances and brightness temperatures.

- ***m_rad_thin***: Module used to store required observation input data headers information and, if requested, to also thin the data. Used by ***create_rad_obs_list***.

  - ***rad_thin_setup***: Set variables and arrays used for thinning observations based on information provided in a thinning resource file.
  - ***rad_thin_set_boxes***: Define locations of thinning boxes and allocate arrays to hold observation information (meta–data).
  - ***rad_thin_clean***: Deallocate the arrays defined in routines ***rad_thin_setup*** and ***rad_thin_set_boxes***.
  - ***rad_thin_put***: Copy required observation header information into thinning box array based on computed penalty.

- ***rad_thin_check_hdr***: Check that some observation header information is within proper limits.
- ***rad_thin_penalty***: Compute observation thinning penalty as a linear function of extracted NR field values.
- ***rad_thin_find_box_id***: Find thinning–box index based on observation location.
- ***rad_thin_box_update***: Place (or over–write) observation information saved in a thinning box based on penalty values and other conditions.
- ***rad_thin_box_count***: Count numbers of thinning boxes that actually have observation information as functions of observation subtype and time slot.
- ***rad_thin_write***: Write thinned observation list to a file ordered by time slot by pulling observation information saved in thinning box arrays.

- ***m_rad_thin_flds***: Module for reading fields used for observation thinning. Used by ***create_rad_obs_list***.

  - ***rad_thin_flds_setup***: Read and print information about fields to use for computing thinning penalty.
  - ***rad_thin_flds_clean***: Deallocate arrays created by ***rad_thin_flds_setup***.
  - ***rad_thin_flds_rc***: Read thinning specification resource file for observation–type dependent thinning parameters and NR field requirements.
  - ***rad_thin_flds_print_information***: Print some information read from the thinning–information resource file.
  - ***rad_thin_flds_read***: Sequence commands to read NR fields required to compute thinning penalty.
  - ***rad_thin_flds_interp***: Interpolate NR fields horizontally and temporally.

- ***m_random_fields***: Module used to create spatially correlated random fields and to get values of those fields at requested locations. Used by ***create_error***.

  - ***random_fields_clean***: Deallocate arrays allocated in ***random_fields_read_params***.
  - ***random_fields_setup***: Setup and create the random, horizontally correlated fields for later extraction of spatially correlated perturbations.
  - ***random_fields_get_values***: Perform horizontal and vertical interpolation of random fields for single–level conventional observations.
  - ***random_fields_get_1_value***: Get single interpolated value at location specified by latitude, longitude, and third index.
  - ***random_fields_get_1_lev***: Get full horizontal field for a single level.
  - ***random_fields_spectra***: Create random spectral coefficients drawn from a distribution such that the expected power spectrum corresponds to that for random fields on the sphere characterized by a selected horizontal correlation function.
  - ***random_fields_synch_ranf***: Synchronize random numbers so all processors create the same sequences after this execution.
  - ***random_fields_frac_weights***: Determine weights for correlated and uncorrelated portions of the perturbations based on the fractions of total variance associated with the correlated part.

- ***m_random_gauss***: Module for drawing random numbers from a Gaussian distribution. Used by ***create_error***.

  - ***random_gauss_r8***: Retrieve an 8–byte random number from a truncated Gaussian distribution with desired mean and standard deviation.

- ***m_random_power***: Determine the expected spherical–harmonic power spectrum for a desired homogeneous, isotropic correlation function on the sphere. Used by ***create_error***.

  - ***random_power_compute***: Determine the expected spherical–harmonic power spectrum a for desired selected homogeneous, isotropic correlation function on the sphere.
  - ***rp_alp0***: Compute Legendre polynomials for a given latitude.
  - ***rp_emns0***: Compute factors required for computing Legendre polynomials.
  - ***rp bssl***: Compute zeros of some Bessel functions used for determining Gaussian latitudes.
  - ***rp_gauss***: Compute Gaussian latitudes and weights used for Gaussian quadrature on a sphere.

- ***m_read_profiles***: Module to read file of interpolated NR profiles created by program ***create_rad_profs***. Used by ***create_rad_reord***, ***create_rad_merge***, and ***create_rad_bufr***.

  - ***read_profiles_setup***: Read header from file of profiles and allocate arrays to read profiles.
  - ***read_profiles_recs***: Read atmospheric profiles for one geographic observing location.
  - ***read_profiles_cleanup***: Deallocate arrays created by ***read_profiles_setup*** and ***obs_list_read_header***.

- ***m_rf_diags_fields***: Module used to compute diagnostics of correlated random fields, for examination or code testing purposes. Used by ***create_error***.

  - ***rf_diags_fields_calc***: Call routine to calculate and print diagnostics of a sample of random fields.
  - ***rf_diags_fields_corrs***: Print a sample of information for testing of random fields if desired.
  - ***rf_diags_fields_int_var***: Integrate the square of a field over the area of the globe.
  - ***rf_diags_fields_get_part***: Get portion of field in desired geographical region.
  - ***rf_diags_fields_print_f***: Print a portion of the random fields.
  - ***rf_diags_fields_1_lat***: Get values at a sample of points at a single latitude.
  - ***rf_diags_fields_prntc***: Print binned NS or EW horizontal correlations as functions of distance and, optionally, print equivalent values for some standard functions.
  - ***rf_diags_fields_sump***: Compute sums of cross products for points separated in N-S and, separately, E-W directions, for all pairs within a portion of the globe.
  - ***rf_diags_fields_compc***: Compute correlations from sums of cross products.

- ***m_rf_diags_power***: Module used to compute diagnostics of power spectra of correlated random fields for examination or code testing purposes. Used by ***create_error***.

  - ***rf_diags_power_calc***: Call routines to calculate and print power spectrum from spherical harmonic coefficients.
  - ***rf_diags_power_print***: Print power spectra and logs of power spectra for up to 7 fields or eta–levels.

- ***m_sat_info_table***: Module used to read satellite radiance information table and to pull requested values from it. Used by ***create_error***, ***create_rad_bufr***, and programs used for observation error tuning.

  - ***sat_info_table_read***: Read the sat info table file and save its values for later reference.
  - ***sat_info_table_get_ic***: Get information from previously read table of instrument/satellite information when requested information is character valued based on matching 1 integer value.
  - ***sat_info_table_get_1c***: Get information from previously read table of instrument/satellite information when requested information is character valued based on matching 1 input name.
  - ***sat_info_table_get_2c***: Get information from previously read table of instrument/satellite information when requested information is character valued based on matching 2 input names.
  - ***sat_info_table_get_1i***: Get information from previously read table of instrument/satellite information when requested information is integer valued based on matching 1 input name.
  - ***sat_info_table_get_2i***: Get information from previously read table of instrument/satellite information when requested information is integer valued based on matching 2 input names.
  - ***sat_info_table_get_1r***: Get information from previously read table of instrument/satellite information when requested information is real valued based on matching 1 input name.
  - ***sat_info_table_get_2r***: Get information from previously read table ofinstrument/satellite information when requested information is real valued based on matching 2 input names.

- ***m_satwind***: Module for subroutines for SATWIND calculations. Used by ***create_satwind***.

  - ***satwind_all_setup***: Determine what types of wind observations are to be considered based on a variable read in a resource file. Also set some required array indexes.
  - ***satwind_all***: Compute visual or IR cloud tracked winds or water vapor feature tracked winds at a single grid point.
  - ***satwind_geost***: Determine header information for SATWIND observations from geostationary satellite platforms for either VIS/IR clouds or water vapor features.
  - ***satwind_polar***: Determine header information for SATWIND observations from polar-orbiting satellite platforms for either VIS/IR clouds or water vapor features.
  - ***satwind_polar_loc_table***: Determine if observation is in the viewing location based on table determined from locations of real observations.
  - ***satwind_view_clouds***: Consider satellite wind observations associated with VIS/IR clouds.
  - ***satwind_view_wvapor***: Consider observations based on variations of layer–integrated precipitable water within the neighborhood of a considered point.
  - ***satwind_findcld***: Find layers with clouds.
  - ***satwind_gcdist***: Calculate great circle distance.
  - ***satwind_assign_bufr***: Copy observation information for one report to an array used by BUFR from an array containing all observation information.

- ***m_satwind_count***: Create and print table of the distribution of SATWIND observations by observation subtype (**kx**) and pressure–level range. Used by ***create_satwind***.

  - ***satwind_count_setup***: Initialize observation counter array and some variables.
  - ***satwind_count_add***: Increment observation counter based on p–level and observation subtype.
  - ***satwind_count_print***: Print table of observation counts as functions of p–layer and **kx** values.

- ***m_satwind_locations***: Module for determining the spatial and temporal distributions of some SATWIND observation subtypes. Used by ***create_satwind***.

  - ***satwind_locations_clean***: Deallocate some arrays used by this module.
  - ***satwind_locations_set***: Set geostationary and polar satellite locations, types, and names.
  - ***satwind_locations_read***: Read SATWIND BUFR file to create table of observation locations and times.
  - ***satwind_locations_hdr***: Read some header information from SATWIND BUFR files.
  - ***check_subset***: Check that am observation subtype is included in a prescribed allowed set.

- ***m_set_unit_nums***: Set some unit numbers for FORTRAN reading or writing. Used by drivers in the ***Sim_rad*** directory.

- ***m_write_profiles***: Module to write a file of atmospheric profiles created by program ***create_rad_profs***. Also used by ***create_rad_reord*** and ***create_rad_merge***.

  - ***write_profiles_header***: Write file header to file of profiles for one observation type.
  - ***write_profiles_recs***: Write observation report header and profile fields for one geographic location.
  - ***write_profiles_sample***: Print profile field values for one observation location.
  - ***write_profiles_close***: Close FORTRAN unit to which profiles have been written.
  - ***write_profiles_tv2t***: Change virtual temperature to temperature.
  - ***write_profiles_o3units***: Change units of ozone from volume mixing ratio to mass mixing ratio.

- ***m_shmem_fields***: Module for reading and using NR fields required by ***create_conv***.

  - ***shmem_fields_setup***: Allocate arrays for the global fields using SHMEM.
  - ***shmem_fields_read***: Read requested set of all 2d or 3d fields required for present purpose, 1 field per processor.
  - ***shmem_fields_clean***: Deallocate SHMEM field arrays.
  - ***read_data_2d***: Read 2d field on NETCDF file
  - ***read_data_3d***: Read all or part of 3d field on NETCDF file.
  - ***horiz_interp_2dfld***: Horizontally interpolate requested 2d fields at 1 or 2 times (Also convert from ϕ_s to z_s).
  - ***horiz_interp_3dfld***: Horizontally interpolate requested 3d fields at 1 or 2 times.

- ***m_shtrans_rf***: Module for projecting spherical harmonic coefficients onto scalar or vector fields that are used to create random fields. Used by ***create_error***.
  - ***sh_clean***: Deallocate arrays used to define spectral truncation and Fourier transform parameters and factors.
  - ***sh_init_factors***: Determine lat–lon grid and all factors used to compute associated Legendre polynomials.
  - ***sh_init_lats***: Determine latitude–dependent factors used for computing values of associated Legendre polynomials
  - ***sh_imax***: Find number of grid longitudes (**imax**) that is the minimum integer that has factors of powers of only 2, 3, or 5 and that is greater than the maximum of 2$*$**mmax**+2 or 16 (**mmax** being the spectral truncation for the random fields).
  - ***sh_abfaclats***: Compute some latitude–dependent factors used to compute associated Legendre polynomials.
  - ***sh_alp***: Compute associated Legendre polynomials for a given zonal wavenumber and latitude.
  - ***sh_emns***: Compute some latitude–independent factors used to compute associated Legendre polynomials.
  - ***sh_trunc***: Determine ordering of associated Legendre polynomials for general pentagonal trunction (of which triangular and rhomboidal are special examples).
  - ***sh_trig***: Determine sines and cosines at edges of latitude bands.
  - ***sh_trans***: Sequence routines to transform from spherical harmonics to global scalar or vector fields.
  - ***sh_s2fm***: Half transform from spherical harmonic coefficients to zonal Fourier coefficients.
  - ***sh_s2c_s***: Copy spectral coefficients for a scalar for one value of zonal wave number.
  - ***sh_s2c_v***: Compute spherical harmonic coefficients for scalar $u\cos(\text{lat})$ or $v\cos(\text{lat})$ fields from spherical harmonic coefficients for stream function and velocity potential.
  - ***sh_c2fm***: Perform half transform from spherical–harmonic coefficients for a scalar field to Fourier coefficients for each grid latitude.
  - ***sh_uvcos***: Divide $u\cos(\text{lat})$ and $v\cos(\text{lat})$ by $\cos(\text{lat})$.
  - ***sh_calc_power***: Calculate power spectra from spherical harmonic coefficients.

- ***m_time_compute***: Module to interpret and modify variables that indicate times associated with observation or field data. Used by several programs.
  - ***time_compute_pack***: Construct date–time character variable from array containing year, month, day, hour, minutes, seconds.
  - ***time_compute_unpack***: Extract array of year, month, day, hour, minutes, seconds from date–time character variable.
  - ***time_compute_dhours***: Compute hours between two sets of date–time arrays.
  - ***time_compute_add***: Add a number of hours to a previous date–time array to create a new date–time array.
  - ***time_compute_new_cdtime***: Add hours to a previous date–time character variable to create a new date–time character variable.
  - ***time_compute_secs_since_yd***: Compute the seconds between a date–time array and a reference time of 1 Jan 1901, 0UTC.

B.3 List of Subroutines Outside of Modules

- ***bufr_satwind_write***: Write satwind observations in .prepbufr (BUFR) format. Used by ***create_satwind***.
- ***compute_lon_stride***: Compute the number of gridpoints at the requested latitude corresponding to the requested stride distance at the equator accounting for convergence of the meridians. Used by ***create_satwind*** and some other programs not part of the regular GOWASP package.
- ***compute_plevs.f90***: This version of subroutine ***compute_plevs*** determines p at layer interfaces and data levels using p_s and the eta–coordinate parameters a_k, b_k for the NR vertical grid. The calculation at data levels uses a formula derived from consideration of the hydrostatic relationship. This specific routine is only used by ***create_rad_bufr***.
- ***compute_press.f90***: This version of subroutine ***compute_plevs*** determines p at layer interfaces and data levels using p_s and the eta–coordinate parameters a_k, b_k for the NR vertical grid. The calculation at data levels uses a simple mean of adjacent interface values. This specific routine is only used by ***create_gpsro***.
- ***fft99.f90***: Collection of Fast Fourier Transform routines orginally from ECMWF written in F77. Used by ***create_error***.
- ***find_name_subs.f90***: Subroutines for finding the index of a particular name within a list of names. Used by all programs.
  - ***find_name***: Find first index of occurance of requested name in a character array of names.
  - ***find_name_2***: Find first index of occurance of requested name in the initial character string (i.e., before any blanks) within a character array.
- ***interpolate_subs.f90***: Routines used for spatial and temporal interpolations, for applying the hydrostatic equation, etc. Used by many programs.
  - ***get_interp_horiz_index***: Compute grid point indexes that surround a specific geographic location and weights for bilinear interpolation.
  - ***get_interp_horiz_nearest***: Redefine horizontal interpolation weights so that the nearest point is given weight 1 and others set to 0.
  - ***get_interp_time_weights***: Determine weights for linear interpolation in time.
  - ***get_interp_time_nearest***: Redefine temporal interpolation weights so that the nearest time is given weight 1 and others set to 0.
  - ***interp_time***: Perform linear interpolation in time.
  - ***get_interp_vert_index***: Determine indexes for grid points above and below some level and determine weights for linear interpolation in z or $\log(p)$.
  - ***interp_vert***: Interpolate vertically between a field at 2 levels.
  - ***hydros_phis_tq***: Compute hydrostatic ϕ at both vertical–grid layer interfaces and data levels using p, T, q.
  - ***hydros_z***: Compute hydrostatic z at vertical–grid layer interfaces using the GEOS-5 layer-mean density field (provided for this purpose).
  - ***interp_hydros_z***: Interpolate z between vertical–grid layer interfaces assuming constant T_v within a layer.

  - ***compute_pk***: Compute p at both vertical grid interfaces and data levels using p_s and the eta–coordinate parameters a_k, b_k.
  - ***interp_rh***: Interpolate q vertically assuming the R_h varies linearly between adjacent data levels.
  - ***transform_q_rh***: Transform between specific and relative humidity.

- ***open_obs_files***: Open observation data files (particularly for radiance observation data types) for reading and/or writing.

- ***prof_io_files***: Construct a list of input observation file names and output profile file names from the date–time and list of observation types.

- ***read_akbk***: Read file of a_k, b_k values for defining eta–coordinates (pressures at interfaces of NR data–layers).

- ***read_nc4_file***: Read a single 2-d field in NETCDF format and check the return status of a NETCDF command.

- ***read_sat_params***: Read file of satellite (scan) parameters for various data types.

- ***set_field_file_name***: Construct a specific file name from a template using a requested field name and time.

- ***sonde_subs.f90***: Routines used to simulate balloon and dropsonde trajectories and observations. Used by ***create_conv***.

  - ***sondes_output***: Create observation reports from raw soundings for RAOB, PIBAL, and DROPSONDE. Output reports to BUFR file.
  - ***non_sonde_output***: Output all requested conventional reports to a BUFR file that are not sondes of some type.
  - ***sonde_report_wind***: Process raw report of sonde wind sounding to create observations on reported levels.
  - ***sonde_report_mass***: Process raw report of sonde mass sounding to create observations on reported levels.
  - ***siglevs_mandatory***: Determine sonde report at mandatory pressure levels from raw sounding.
  - ***siglevs_linear***: Add observations for sounding levels that cannot be reconstructed well by linear interpolation between other observation levels.
  - ***siglevs_sort***: Sort array according to values in field index **nf**.
  - ***siglevs_trop***: Find level of the tropopause.
  - ***siglevs_slope***: Least squares fit to find slope.
  - ***siglevs_qc***: Copy new observation to required arrays and set QC and program codes to be written to BUFR file.
  - ***siglevs_range_check***: Assign quality based on whether value is in range and on original QC mark.
  - ***summary_counts***: For each conventional–observation subtype index **kx**, print a table of numbers of such observations.
  - ***sonde_drift_location***: Determine new location (lat,lon,height) of sonde after short drift interval.

- ***prof_all_2d3d***: Create separate arrays of 2d fields and 3d fields from a single array of profile data.

- ***sub_zensun.f90***: File containing routines to compute solar zenith or elevation and azimuth angles for radiance observations.

  - ***angles_ssmis***: Set up call to subroutine ***zensun*** for getting solar zenith and azimuth angles, as particularly required for SSMIS data.
  - ***get_solar_elevation***: Set up call to subroutine ***zensun*** for solar elevation based on location, date, and UTC time.
  - ***zensun***: Subroutine from GSI code for computing solar position angles as functions of geographic coordinates, date and time.

References

Errico, R. M., R. Yang, N. C. Privé, K.–S. Tai, R. Todling, M. E. Sienkiewicz, J. Guo, 2013: Development and validation of observing system simulation experiments at NASAs Global Modeling and Assimilation Office. *Quart. J. Roy. Meter. Soc.*, **139**, 1162–1178.

Gelaro, R., W. M. Putman, S. Pawson, C. Draper, A. Molod, P. M. Norris, L. Ott, N. Privé, O. Reale, D. Achuthavarier, M. Bosilovich, V. Buchard, W. Chao, L. Coy, R. Cullather, A. da Silva, A. Darmenov, R. M. Errico, M. Fuentes, M.–J. Kim, R. Koster, W. McCarty, J. Nattala, G. Partyka, S. Schubert, G. Vernieres, Y. Vikhliaev, and K. Wargan, 2015. Evaluation of the 7–km GEOS–5 Nature Run. NASA/TM-2014–104606, Vol. 36. Document (88612 kB).

Hollingsworth, A. and P. Lonnberg, 1986: The statistical structure of short–range forecast errors as determined from radiosonde data. Part I: The wind field. *Tellus.* **38A**, 111–136.

Ide, K., P. Courtie, M. Ghil, A.C. Lorenc. 1997. Unified Notation for data assimilation: operational, sequential and variational. *J. Meteorol. Soc. Japan.* **75**: 181–189.

Privé, N.C., R.M. Errico, 2016. Temporal and spatial interpolation errors of high-resolution modeled atmospheric fields. *J. Atmos. Ocean. Tech*, **33**, 303–311.

Putman, W., A. M. da Silva, L. E. Ott, and A. Darmenov, 2014. Model Configuration for the 7–km GEOS–5 Nature Run, Ganymed Release (Non-hydrostatic 7–km Global Mesoscale Simulation). GMAO Office Note No. 5 (Version 1.0), Document (PDF, 6123 kB)

Rienecker MM, Suarez MJ, Todling R, Bacmeister J, Takacs L, Liu H-C, Gu W, Sienkiewicz M, Koster RD, Gelaro R, Stajner I, Nielsen JE. 2008. The GEOS–5 Data Assimilation System –

Documentation of versions 5.0.1, 5.1.0, and 5.2.0. *Technical Report series on global Modeling and Assimilation.* Suarez M, ed. NASA/TM–2008–104606, **27**: 109 pp.

Previous Volumes in This Series

Volume 1
September 1994
Documentation of the Goddard Earth Observing System (GEOS) general circulation model - Version 1
L. L. Takacs, A. Molod, and T. Wang

Volume 2
October 1994
Direct solution of the implicit formulation of fourth order horizontal diffusion for gridpoint models on the sphere
Y. Li, S. Moorthi, and J. R. Bates

Volume 3
December 1994
An efficient thermal infrared radiation parameterization for use in general circulation models
M.-D. Chou and M. J. Suarez

Volume 4
January 1995
Documentation of the Goddard Earth Observing System (GEOS) Data Assimilation System - Version 1
James Pfaendtner, Stephen Bloom, David Lamich, Michael Seablom, Meta Sienkiewicz, James Stobie, and Arlindo da Silva

Volume 5
April 1995
Documentation of the Aries-GEOS dynamical core: Version 2
Max J. Suarez and Lawrence L. Takacs

Volume 6
April 1995
A Multiyear Assimilation with the GEOS-1 System: Overview and Results
Siegfried Schubert, Chung-Kyu Park, Chung-Yu Wu, Wayne Higgins, Yelena Kondratyeva, Andrea Molod, Lawrence Takacs, Michael Seablom, and Richard Rood

Volume 7
September 1995
Proceedings of the Workshop on the GEOS-1 Five-Year Assimilation
Siegfried D. Schubert and Richard B. Rood

Volume 8
March 1996
Documentation of the Tangent Linear Model and Adjoint of the Adiabatic Version of the NASA GEOS-1 C-Grid GCM: Version 5.2
Weiyu Yang and I. Michael Navon

Volume 9
March 1996
Energy and Water Balance Calculations in the Mosaic LSM
Randal D. Koster and Max J. Suarez

Volume 10
April 1996
Dynamical Aspects of Climate Simulations Using the GEOS General Circulation Model
Lawrence L. Takacs and Max J. Suarez

Volume 11
May 1997
Documentation of the Tangent Linear and its Adjoint Models of the Relaxed Arakawa-Schubert Moisture Parameterization Package of the NASA GEOS-1 GCM (Version 5.2)
Weiyu Yang, I. Michael Navon, and Ricardo Todling

Volume 12
August 1997
Comparison of Satellite Global Rainfall Algorithms
Alfred T. C. Chang and Long S. Chiu

Volume 13
December 1997
Interannual Variability and Potential Predictability in Reanalysis Products
Wie Ming and Siegfried D. Schubert

Volume 14
August 1998
A Comparison of GEOS Assimilated Data with FIFE Observations
Michael G. Bosilovich and Siegfried D. Schubert

Volume 15
June 1999
A Solar Radiation Parameterization for Atmospheric Studies
Ming-Dah Chou and Max J. Suarez

Volume 16
November 1999
Filtering Techniques on a Stretched Grid General Circulation Model
Lawrence Takacs, William Sawyer, Max J. Suarez, and Michael S. Fox-Rabinowitz

Volume 17
July 2000
Atlas of Seasonal Means Simulated by the NSIPP-1 Atmospheric GCM
Julio T. Bacmeister, Philip J. Pegion, Siegfried D. Schubert, and Max J. Suarez

Volume 18
December 2000
An Assessment of the Predictability of Northern Winter Seasonal Means with the NSIPP1 AGCM
Philip J. Pegion, Siegfried D. Schubert, and Max J. Suarez

Volume 19
July 2001
A Thermal Infrared Radiation Parameterization for Atmospheric Studies
Ming-Dah Chou, Max J. Suarez, Xin-Zhong Liang, and Michael M.-H. Yan

Volume 20
August 2001
The Climate of the FVCCM-3 Model
Yehui Chang, Siegfried D. Schubert, Shian-Jiann Lin, Sharon Nebuda, and Bo-Wen Shen

Volume 21
September 2001
Design and Implementation of a Parallel Multivariate Ensemble Kalman Filter for the Poseidon Ocean General Circulation Model
Christian L. Keppenne and Michele M. Rienecker

Volume 22
August 2002
A Coupled Ocean-Atmosphere Radiative Model for Global Ocean Biogeochemical Models
Watson W. Gregg

Volume 23
November 2002
Prospects for Improved Forecasts of Weather and Short-term Climate Variability on Subseasonal (2-Week to 2-Month) Time Scales
Siegfried D. Schubert, Randall Dole, Huang van den Dool, Max J. Suarez, and Duane Waliser

Volume 24
July 2003
Temperature Data Assimilation with Salinity Corrections: Validation for the NSIPP Ocean Data Assimilation System in the Tropical Pacific Ocean, 1993–1998
Alberto Troccoli, Michele M. Rienecker, Christian L. Keppenne, and Gregory C. Johnson

Volume 25
December 2003
Modeling, Simulation, and Forecasting of Subseasonal Variability
Duane Waliser, Siegfried D. Schubert, Arun Kumar, Klaus Weickmann, and Randall Dole

Volume 26
April 2005
Documentation and Validation of the Goddard Earth Observing System (GEOS) Data Assimilation System - Version 4
Senior Authors: S. Bloom, A. da Silva and D. Dee
Contributing Authors: M. Bosilovich, J-D. Chern, S. Pawson, S. Schubert, M. Sienkiewicz, I. Stajner, W-W. Tan, and M-L. Wu

Volume 27
December 2008
The GEOS-5 Data Assimilation System - Documentation of Versions 5.0.1, 5.1.0, and 5.2.0
M. M. Rienecker, M. J. Suarez, R. Todling, J. Bacmeister, L. Takacs, H.-C. Liu, W. Gu, M. Sienkiewicz, R. D. Koster, R. Gelaro, I. Stajner, and J. E. Nielsen

Volume 28
April 2012
The GEOS-5 Atmospheric General Circulation Model: Mean Climate and Development from MERRA to Fortuna
Andrea Molod, Lawrence Takacs, Max Suarez, Julio Bacmeister, In-Sun Song, and Andrew Eichmann

Volume 29
June 2012
Atmospheric Reanalyses Recent Progress and Prospects for the Future.
A Report from a Technical Workshop, April 2010
Michele M. Rienecker, Dick Dee, Jack Woollen, Gilbert P. Compo, Kazutoshi Onogi, Ron Gelaro, Michael G. Bosilovich, Arlindo da Silva, Steven Pawson, Siegfried Schubert, Max Suarez, Dale Barker, Hirotaka Kamahori, Robert Kistler, and Suranjana Saha

Volume 30
December 2012
The GEOS-iODAS, Description and Evaluation
Guillaume Vernieres, Michele M. Rienecker, Robin Kovach and Christian L. Keppenne

Volume 31
March 2013
Global Surface Ocean Carbon Estimates in a Model Forced by MERRA
Watson W. Gregg, Nancy W. Casey, and Cécile S. Rousseaux

Volume 32
March 2014
Estimates of AOD Trends (2002–2012) over the World's Major Cities based on the MERRA Aerosol Reanalysis
Simon Provençal, Pavel Kishcha, Emily Elhacham, Arlindo M. da Silva and Pinhas Alpert

Volume 33
August 2014
The Effects of Chlorophyll Assimilation on Carbon Fluxes in a Global Biogeochemical Model
Cécile S. Rousseaux and Watson W. Gregg

Volume 34
September 2014
Background Error Covariance Estimation using Information from a Single Model Trajectory with Application to Ocean Data Assimilation into the GEOS-5 Coupled Model
Christian L. Keppenne, Michele M. Rienecker, Robin M. Kovach, and Guillaume Vernieres

Volume 35
December 2014
Observation-Corrected Precipitation Estimates in GEOS-5
Rolf H. Reichle and Qing Liu

Volume 36
March 2015
Evaluation of the 7-km GEOS-5 Nature Run
Ronald Gelaro, William M. Putman, Steven Pawson, Clara Draper, Andrea Molod, Peter M. Norris, Lesley Ott, Nikki Privé, Oreste Reale, Deepthi Achuthavarier, Michael Bosilovich, Virginie Buchard, Winston Chao, Lawrence Coy, Richard Cullather, Arlindo da Silva, Anton Darmenov, Ronald M. Errico, Marangelly Fuentes, Min-Jeong Kim, Randal Koster, Will McCarty, Jyothi Nattala, Gary Partyka, Siegfried Schubert, Guillaume Vernieres, Yuri Vikhliaev, and Krzysztof Wargan

Volume 37
March 2015
Maintaining Atmospheric Mass and Water Balance Within Reanalysis
Lawrence L. Takacs, Max Suarez, and Ricardo Todling

Volume 38
September 2015
The Quick Fire Emissions Dataset (QFED): Documentation of versions 2.1, 2.2 and 2.4
Anton Darmenov and Arlindo da Silva

Volume 39
September 2015
Land Boundary Conditions for the Goddard Earth Observing System Model Version 5 (GEOS-5) Climate Modeling System - Recent Updates and Data File Descriptions
Sarith Mahanama, Randal Koster, Gregory Walker, Lawrence Takacs, Rolf Reichle, Gabrielle De Lannoy, Qing Liu, Bin Zhao, and Max Suarez

Volume 40
October 2015
Soil Moisture Active Passive (SMAP) Project Assessment Report for the Beta-Release L4_SM Data Product
Rolf H. Reichle, Gabrielle J. M. De Lannoy, Qing Liu, Andreas Colliander, Austin Conaty, Thomas Jackson, John Kimball, and Randal D. Koster

Volume 41
October 2015
GDIS Workshop Report
Siegfried Schubert, Will Pozzi, Kingtse Mo, Eric Wood, Kerstin Stahl, Mike Hayes, Juergen Vogt, Sonia Seneviratne, Ron Stewart, Roger Pulwarty, and Robert Stefanski

Volume 42
November 2015
Soil Moisture Active Passive (SMAP) Project Calibration and Validation for the L4_C Beta-Release Data Product
John Kimball, Lucas Jones, Joseph Glassy, E. Natasha Stavros, Nima Madani, Rolf Reichle, Thomas Jackson, and Andreas Colliander

Volume 43
September 2015
MERRA-2: Initial Evaluation of the Climate
Michael G. Bosilovich, Santha Akella, Lawrence Coy, Richard Cullather, Clara Draper, Ronald Gelaro, Robin Kovach, Qing Liu, Andrea Molod, Peter Norris, Krzysztof Wargan, Winston Chao, Rolf Reichle, Lawrence Takacs, Yury Vikhliaev, Steve Bloom, Allison Collow, Stacey Firth, Gordon Labow, Gary Partyka, Steven Pawson, Oreste Reale, Siegfried D. Schubert, and Max Suarez

Volume 44
2016
Estimation of the Ocean Skin Temperature using the NASA GEOS Atmospheric Data Assimilation System
Santha Akella, Ricardo Todling and Max Suarez

Volume 45
2016
The MERRA-2 Aerosol Assimilation
C. A. Randles, A. M. da Silva, V. Buchard, A. Darmenov, P. R. Colarco, V. Aquila, H. Bian, E. P. Nowottnick, X. Pan, A. Smirnov, H. Yu, and R. Govindaraj

Volume 46
2016
MERRA-2 Input Observations: Summary and Assessment
Will McCarty, Lawrence Coy, Ronald Gelaro, Albert Huang, Dagmar Merkova, Edmond B. Smith, Meta Sienkiewicz, and Krzysztof Wargan

Volume 47
2017
An Evaluation of Teleconnections Over the United States in an Ensemble of AMIP Simulations with the MERRA-2 Configuration of the GEOS Atmospheric Model
A. B. Marquardt Collow, S. P. Mahanama, M. G. Bosilovich, R. D. Koster, and S. D. Schubert