Documentation of the Data Assimilation Coding Environment (DACE)

Roland Potthast

March 4, 2019

Contributor's List:

Many thanks to all the people who have put a lot of work into the DACE coding environment and its documentation. Particular thanks is given to Andreas Rhodin, who has been responsible for the code over a period of 15 years and has been the main designer of its structure and components.

Roland Potthast, Andreas Rhodin, Harald Anlauf, Christoph Schraff, Thorsten Steinert, Annika Schomburg, Stefanie Hollborn, Elisabeth Bauernschubert, Anne Walter, Michael Bender, Alexander Creß, Robin Faulwetter, Christina Köpken-Watts, Martin Lange, Gerhard Paul, Hendrik Reich, Klaus Stephan, Olaf Stiller, Silke May, Marc Pondrom, Ulrich Blahak, Lisa Neef, Christian Welzbacher, Lilo Bach, Sven Ulbrich, Kobra Khosravian, Martin Weissmann, Heiner Lange, Tijana Janjic-Pfander, Heinz-Werner Bitzer, Rory Gray, Africa Perianez, Jaison Ambadan, Eric Lange, Stefan Declair, Detlef Pingel, Oliver Schmid, Nicolas Fischer

About this Documentation

This *Documentation of the DACE Data Assimilation Coding Environment* has two aims:

1. To provide new users an introduction to the scientific background of Data Assimilation (DA) and its implementation within the Data Assimilation Coding Environment DACE
2. To serve as a work of reference for the developers and users of the respective subsystems.

The documentation is organized as follows:

Part I **Data Assimilation and the Cycled NWP System** displays the basic ideas of data assimilation and how it is organized at DWD. This part of the documentation serves as an introduction to the further parts that are linked in there.

Part II **Scientific Documentation** describes the algorithms of the different data assimilation methods.

Part III **User Guide** shows the user how to install and execute the system's components. Herein, the namelists are described and guidance is provided how to run cycled data assimilation.

Part IV **Implementation** describes the codes and modules.

The sequence of these Parts is such that a continous reading of the document should be consistently possible. The Parts contain reading guidelines for themselves.

The typical reader will probably skip back and forth through the document – in that case, *forward-links* from the scientific part to the implementation and user guide are provided, and occasionally *backward-links*, especially in the case of namelist variables in the user guide that act as switches for certain algorithmic features.

It is important to acknowledge that this documentation is always *work-in-progress*, because the features and algorithms of the DWD DA System are always in development.

If the reader has the feeling that something is missing or badly/wrongly documented, he/she should not hesitate to contact the authors or to actively contribute to this documentation, following the writing guidelines (Chapter 29) in the Appendix.

Contents

Todo list

Part I

Data Assimilation and the Cycled NWP System

Chapter 1

Introduction

1.1 The Development Process

Developing data assimilation (DA) methods for use in an operational environment needs a mature and well-documented *development process*. It includes the development of particular system parts in an research environment, testing and calibration in a basic cycling system and their integration into the operational environment. We employ the following four levels of system development.

1. **Individual Code Development.** Individual developers and researchers work on their part of the code. Initial testing is carried out for an individual module by using well-prepared input in a test environment, usually a part of the local file system on a development machine or a high-performance cluster (when testing parallelization).

2. **Basic Cycling.** The *basic cycle* is an environment for testing and calibrating the whole system which consists of model runs and different data assimilation modules. The basic cycling is described in Section 3. The basic cycling system is highly *portable* and suitable for use in different computing environments. It runs based on input files and will write the output into particular directories of a file system.

3. **Preoperational Testing in NUMEX.** The NUMEX system is a testing environment of the cycled NWP system in a quasi-operational setup. It employs *data bases* for storing files, such that large-scale experiments over extended periods of time and in high-resolution can be carried out. NUMEX has all features which the operational system has, and it is able to run any part of the system in an operational mode, including particular cut-off times for data when the system is run under tight time-constraints.

   Full description of NUMEX somewhere?

4. **Operational Assimilation and Forecasting.** The operational system is run under high demands on *availability* and *fault tolerance*. Forecasts need to be available for key national customers both within Deutscher Wetterdienst (DWD) as well as externally. Weather services in about 40 States worldwide take the global data as boundary conditions for their regional weather forecasting models.

Development based on the NWP system and for the NWP system takes place in very different ways. The use of a cycled test environment is of particular importance for all of them, as soon as the development wants to go beyond single *case studies*. Development consists in:

1. developing new *model components*, either as part of the *dynamical core* or *model physics*;
2. developing new components of the core data assimilation module, for example a new *bias correction* or *adaptive determination* of *system parameters*;
3. integrating *new observations* into the system, which usually includes several development steps:
   (a) developing fast *observation operators*, which simulate an observation based on the model of the atmospheric state,
   (b) *integrating* the observation operator into either the model or the data assimilation system or both (depending on which method to be used in data assimilation),
   (c) developing an appropriate *preprocessing* of the observation data, including for example
      i. *remapping* and *conversion* of data
      ii. *quality control*
      iii. *bias correction*
      iv. gaining additional information by calculating *retrievals*
      v. cloud detection and correction schemes
   (d) *calibrating* the cycled system with the observations and their assimilation interacting with the model dynamics;
4. developing new *postprocessing* methods such as *model output statistics* or *verification tools*, calculating various scores and norms which help to understand and improve the use of the analysis and forecast and to guide further system development.

@Roland: You planned a figure here, but there was none (HL).

1.2 Model Hierarchy

The DWD Data Assimilation System is built to provide a flexible integrated data assimilation environment to serve both the global model as well as high-resolution convective data assimilation. Modeling takes place on different spatial and temporal scales:

1. On synoptic scales, the global ICON is being developed and operationally run by DWD. In the past, for the global model GME the horizontal resolution has been increased about every couple of years, reaching 20 km mesh size in 2013. The global ICON model 13 km horizontal mesh size is envisioned in 2014.

2. On the European scale in 2013 we have run the COSMO model in its version COSMO-EU with a horizontal resolution of 7 km. From 2015, a nested ICON model is planned to run, with 6 km resolution in Europe.

3. Currently, the COSMO model is run on a horizontal resolution of 2.8 km over Germany as COSMO-DE. The plan is to move towards a 2.2 km resolution and extended area in 2015. MeteoSwiss is working on the ensemble data assimilation system for its 1 km COSMO NWP system over the Alpine region.

Each NWP model is linked to its particular *data assimilation system*, which need to be adapted to the particular *spatial and temporal scale* of meteorological processes and the particular *measurements* which are available on each scale.

1. On the global scale we provide a Hybrid Variational Ensemble Kalman Filter (VarEnKF), which consists of an Ensemble Kalman Filter (EnKF) and a 3-dimensional Variational Data Assimilation system.

2. The same system is run on the European scale for the nested part of the ICON model system. For COSMO-EU *dynamical forcing* or *nudging* has been employed until 2014. *Boundary conditions* for these methods are provided by the global model.

3. The high-resolution COSMO system COSMO-DE and COSMO-DE-EPS has used *nudging* up to 2014. An EnKF with an additional *deterministic analysis* has been developed and is to become operational in 2015.

   Boundary conditions for the regional NWP system are provided by the global model and its nested European version.

Chapter 2

DA Concepts

The task of data assimilation is to *determine the state of the atmosphere* as a key input for the forecasts which are obtained by running the NWP model forward in time. To this end

- *measurement data* $\mathbf{y}_k \in \mathbb{R}^m$ with dimension $m \in \mathbb{N}$ of the measurement space from the time period $(t_{k-1}, t_k]$, $k \in \mathbb{N}$, between the last analysis time t_{k-1} and the current time t_k are *assimilated* and
- an *analysis* $\mathbf{x}_k^a \in \mathbb{R}^n$ with dimensions n of the state space is generated,

where the superscript a refers to *analysis* and $k \in \mathbb{N}$ is the time index.

Data assimilation uses several different *analysis modules* which are taken in turns with *model runs*. This is called *cycling*, compare Figure 2.1.

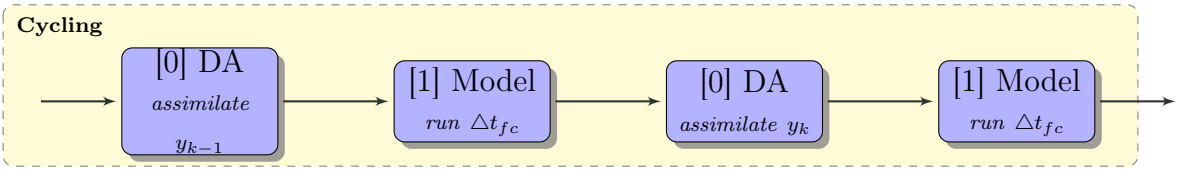


Figure 2.1: Cycling refers to the use of both the *model dynamics* and *measurement data* to estimate the current state of the atmosphere. The forecast interval is written as $\triangle t_{fc} = t_k - t_{k-1}$.

The basic idea is

1. to start with some previous analysis $\mathbf{x}_{k-1}^a$ at time t_{k-1},
2. to use the model to propagate the state $\mathbf{x}_{k-1}^a$ forward in time to time t_k. We call the propagated state *the background* at time t_k and denote it by $\mathbf{x}_k^b$.
3. Then, the analysis modules are run, using both the background $\mathbf{x}_k^b$ and the data $\mathbf{y}_k$ to calculate a new analysis $\mathbf{x}_k^a$, valid at time t_k.
4. Then, this elementary cycle is repeated.

The main modules which are part of the data assimilation cycle are shown in Figure 2.2. The situation is slightly more involved than indicated in the points 1.-4. above:

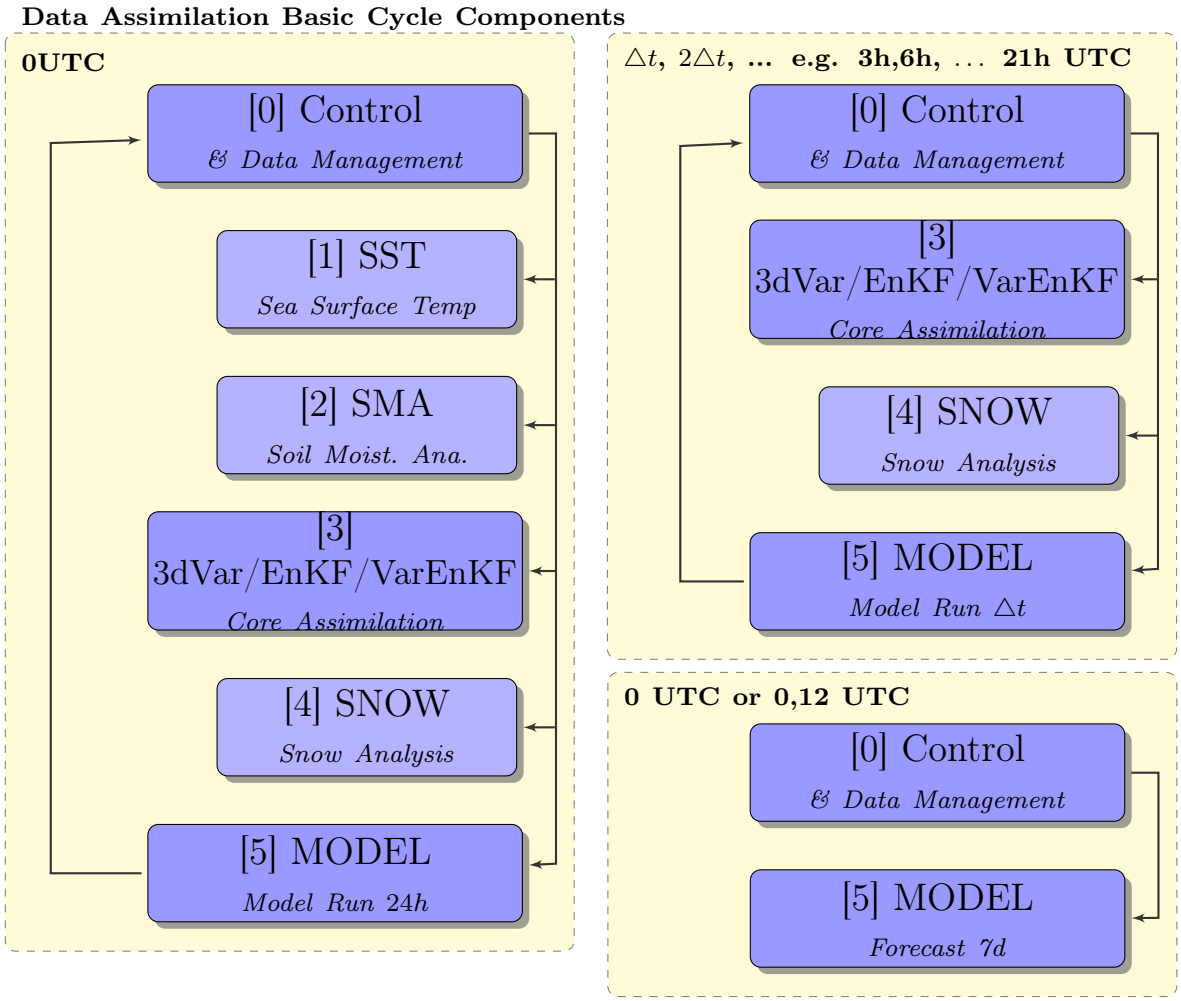


Figure 2.2: We show the components of the data assimilation basic cycle. Once a day at 0 UTC the modules shown in the left column are run. At other analysis times, only snow analysis SNOW [4] and the core module 3dVAR/EnKF/VarEnKF [3] is carried out. Once a day a forecast run of 24h is needed as input for the SMA [2] of the next day.

1. the analysis modules
   (a) *sea surface temperature* SST/ICE [1] and
   (b) *soil moisture analysis* SMA [2]

   are run only once per day, while the
   (a) *snow analysis* SNOW [4] and the
   (b) *DA core module* 3dVAR/EnKF/VarEnKF [3]

   are run in every analysis step, usually every three hours for the global model,
2. the *soil moisture analysis* SMA needs more forecast data than just the background and measurement data. It needs a forecast run MODEL [5] over the past 24 hours with appropriate field output.

3. At all other analysis times $t_k \neq 0$ UTC, the model MODEL [5] is run in increments of time $\triangle t$.

In Figure 2.2 the situation is shown by the two areas which display the run of the modules for the global data assimilation at 0 UTC and at $\triangle t$, $2\triangle t$,... UTC.

Forecast runs as shown in the box at the bottom right of Figure 2.2 can be run any time starting from an analysis.

@Roland: Maybe display operational Runclock of COSMO/GME here as an illustration?

2.1 Module Workbench

There are several different modules running in a data assimilation cycle. Usually, assimilation of soil moisture, for example, is not carried out within the core data assimilation component, but by the separate module SMA taking into account the particular situation of the earth system. The basic ingredients of these models can be classified as visualized in Figure 2.3.

- We need to take into account the *background* BG, i.e. the state of our system as calculated and forecasted from previous analysis steps.
- Of course, *observations* OBS are used, where in addition to *raw observations* we may also assimilate *retrievals* which are calculated as *analysis fields* by other processing centers, for example the SNOW analysis or SST analysis provided by NCEP.
- We need parameter files PAR ranging from climatological field to particular masks (land-sea-mask), bias correction data, particular covariance fields (B-Matrix) or external parameter fields as vegetation indices or physical constants.
- The control of the module takes place by *control files*, which usually include a NAMELIST providing all the input parameters of the module.

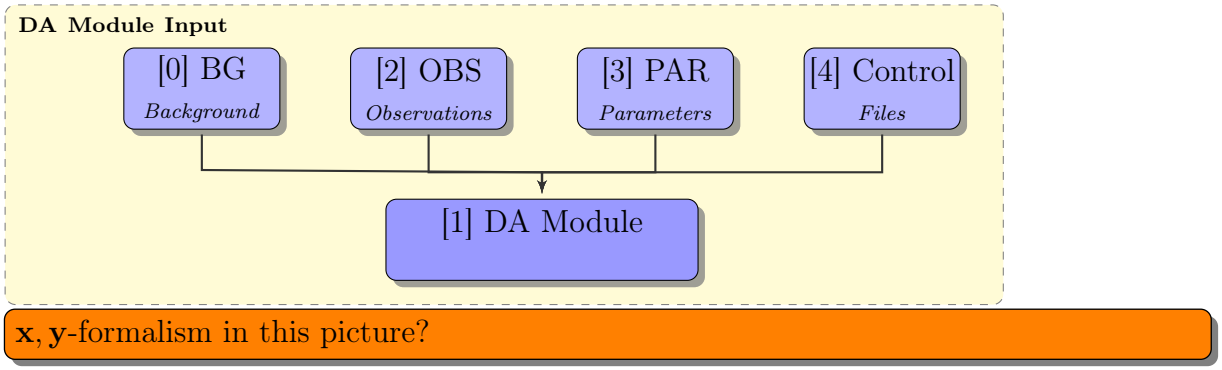


Figure 2.3: We show a classification of different types of input for the data assimilation modules, ranging from *background states* BG via *observations* OBS and *parameter files* PAR to the *control files*, e.g. the NAMELIST.

2.2 Assimilation Modules

The core data assimilation module takes all atmospheric *observations* and the *background* states to calculate the atmospheric *analysis*, the best estimate for the current atmospheric state.

The core module offers different algorithms to carry out this assimilation task. In particular, the module includes

1. a 3-dimensional Variational Data Assimilation (3dVar),
2. a Local Ensemble Transform Kalman Filter (LETKF) (often referred to as Ensemble Kalman Filter (EnKF)),
3. a Hybrid Variational Ensemble Kalman Filter (VarEnKF).

The scientific description of these algorithms is carried out in Chapters 7, 9 and 10. Here, it is sufficient to know that the assimilation methods calculate the analysis based on some version of the norm difference

$$J^b(\mathbf{x}) := \|\mathbf{x} - \mathbf{x}^b\|^2 \tag{2.1}$$

of a state to the background as well as the difference

$$J^o(\mathbf{x}) := \|\mathbf{y} - H(\mathbf{x})\|^2 \tag{2.2}$$

with some particular norm $\|\cdot\|$ between the observations $\mathbf{y} \in \mathbb{R}^m$ and the model equivalent $H(\mathbf{x})$ in observation space, where the *observation operator*

$$H : \mathbb{R}^n \to \mathbb{R}^m \tag{2.3}$$

calculates simulated observations or *model equivalent*, respectively, based on the model state $\mathbf{x} \in \mathbb{R}^n$.

2.2.1 Core Modules: 3dVar, EnKF and VarEnKF

Here, we first look at the conceptional differences between the three algorithms.

1. 3dVar is a variational algorithm, which calculates the analysis $\mathbf{x}^a$ for one atmospheric state, based on the background state $\mathbf{x}^b$ via an *iterative minimization approach* based on the *conjugate gradient method* (CG).
2. The EnKF is an *ensemble* data assimilation method. It needs an ensemble of states $\mathbf{x}^{b(\ell)}$, $\ell = 1, ..., L$, as input, and generates an *analysis ensemble* $\mathbf{x}^{a(\ell)}$, $\ell = 1, ..., L$, as output by a combination of an *optimization method* in ensemble space with an *algebraic algorithm* to generate the analysis ensemble.
3. The VarEnKF is a hybrid method, which takes a background ensemble $x^{b(\ell)}$, $\ell = 1, ..., L$, and a deterministic state $\mathbf{x}^b$ as input and generates an analysis ensemble $\{\mathbf{x}^{a(\ell)} : \ell = 1, \ldots, L\}$ and a deterministic analysis $\mathbf{x}^a$ as output. The method couples the 3dVar with an EnKF, using information from both schemes in each of its modules.

When speaking about *the background*, this can refer either to the deterministic background $\mathbf{x}^b$ of 3dVar or the the background ensemble $\{\mathbf{x}^{b(\ell)},\ \ell = 1,\ldots,L\}$ of the EnKF or to the joint ensemble $\{\mathbf{x}^b, \mathbf{x}^{b(\ell)},\ \ell = 1,\ldots,L\}$ of a deterministic state plus an ensemble of states.

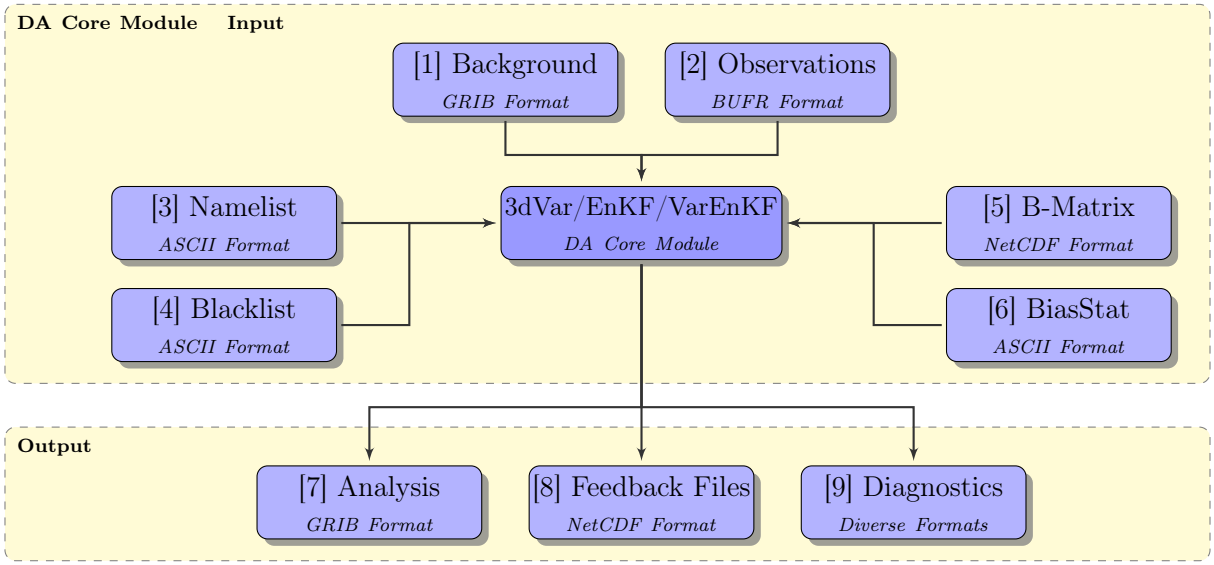


Figure 2.4: The figure displays the data processing concept of the core data assimilation module 3dVar/EnKF/VarEnKF (specific flowcharts for the different methods are provided in Chapter 14).

2.2.2 I/O Concept and File formats

Before we go into more details about the particular input and output, we summarize the basic concept of observation and state I/O.

1. Atmospheric *states* $\mathbf{x} \in \mathbb{R}^n$ are usually stored in GRIB file format.
2. *Observations* are stored in BUFR format and converted into NetCDF before used by the data assimilation modules.
3. The *model equivalents* $H(\mathbf{x}^b)$ and $H(\mathbf{x}^a)$ are stored in the form of so-called *feedback files* in NetCDF format. These files contain
   (a) the original observations $\mathbf{y}$,
   (b) the model equivalents of these observations $H(\mathbf{x}^b)$ and
   (c) *quality flags* and *quality information* which has been derived by the analysis module when processing the data and comparing observations to the model equivalents.

All programs read and write files to the local file system, which are then copied into some *data base* by the *control module*, compare Control [4] in Figure 2.3. The data base can be

some *archive*, as used for the *operational NWP*, or it can consist of a larger *file system*, where particular analysis or forecast files are stored for further use and evaluation.

The concept of data processing of the core data assimilation module is shown in Figure 2.4. The main types of input and output files are shown by [1] to [9].

1. The *background* [1] will be provided in the form of one or several files in GRIB format. Fields based on different model grids may need additional information for further processing.

   @Roland: for example, the model grid information of the ICON model is ... – Missing sentence!

2. *Observations* [2] $\mathbf{y}^o$ are provided in the form of several files in BUFR format. They are usually converted into NetCDF format, before they are read by the core module.

   @Roland: I completed this list, preliminarily. Please read it. (HL).

3. FORTRAN *Name lists* [3] are used to communicate the user-settings to the binaries of the DA Core Modules.

4. Certain observations can be excluded using explicit *Blacklists* [4].

5. In case of 3dVar, the background error covariance matrix $\mathbf{B}$ can be provided externally as a NetCDF file [5].

6. Information about observation biases can be communicated using text lists [6].

7. The Core DA Module produces an *Analysis* [7] state $\mathbf{x}^a$ or an ensemble of analysis states $\{\mathbf{x}^{a(\ell)} : \ell = 1, \ldots, L\}$ in the GRIB format.

8. Observations $\mathbf{y}^o$ and their model equivalents $H(\mathbf{x})$ are written together into *feedback files* [8] in the NetCDF-format in order to evaluate *offline* diagnostics or verifications in observation space.

9. Some *diagnostics* [9] are performed *online* within the DA Core Modules. Their output can be in diverse formats, including graphical plots.

2.3 Supplemental Modules

In order to produce both synoptic and convective scale analyses, the core-modules of Section 2.2 need external fields on the lower boundaries consisting of sea surface temperature (SST), sea ice coverage (ICE), snow coverage (SNOW) and soil moisture (SMA) analysis. This section gives a first glance at how these supplemental modules produce the needed analysis fields. A full documentation is provided in Chapter 11.

2.3.1 Sea Surface Temperature (SST) and ICE Analysis

The *sea surface temperature analysis* SST and *ice analysis* ICE employs different data sources, including

1. the NCEP SST analysis;
2. observations from *ships* and *buoys* over several days before the analysis time;
3. *sea ice concentration* or *sea ice fraction* as provided by
   (a) the Federal Maritime and Hydrographic Agency of Germany (BSH)
   (b) and NCEP MMAB.
4. climatological fields as derived from ECMWF climatology.

@Roland by TJ ECMWF reanalysis ERA... ?

@Roland by HL: I commented out some incomplete stuff here, please look into the .tex-File

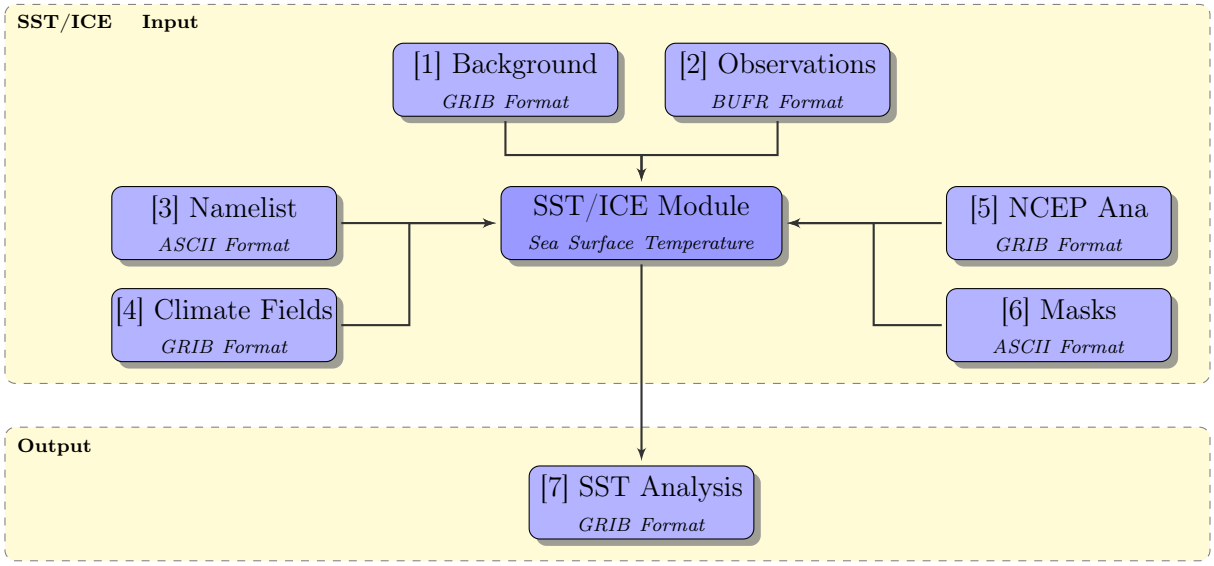


Figure 2.5: Input and Output of the SST/ICE module.

@Roland by HL: Also here, I commented out some incomplete stuff, please look into the .tex-File

2.3.2 Snow Analysis

Snow analysis is carried out by a local averaging and interpolation of observed values, where the external analysis and a background field is used in data-poor regions. The *snow analysis* is based on

1. synoptic observations (SYNOP)
2. NCEP Snow and Ice Analysis[1] as displayed in Figure 2.7 (a) and Figure 2.7 (b). NCEP uses satellite data from
   (a) POES AVHRR
   (b) AMSU
   (c) GOES Imager
   (d) GMS
   (e) MSG Seviri
   (f) DMSP SSM/I

   We use NCEP *snow height* and *snow cover* fields (contained in a file `noaasn`) as input to the DWD snow analysis module.
3. monthly snow depth climatology from ECMWF and
4. *model background* forecasted from previous analysis steps.

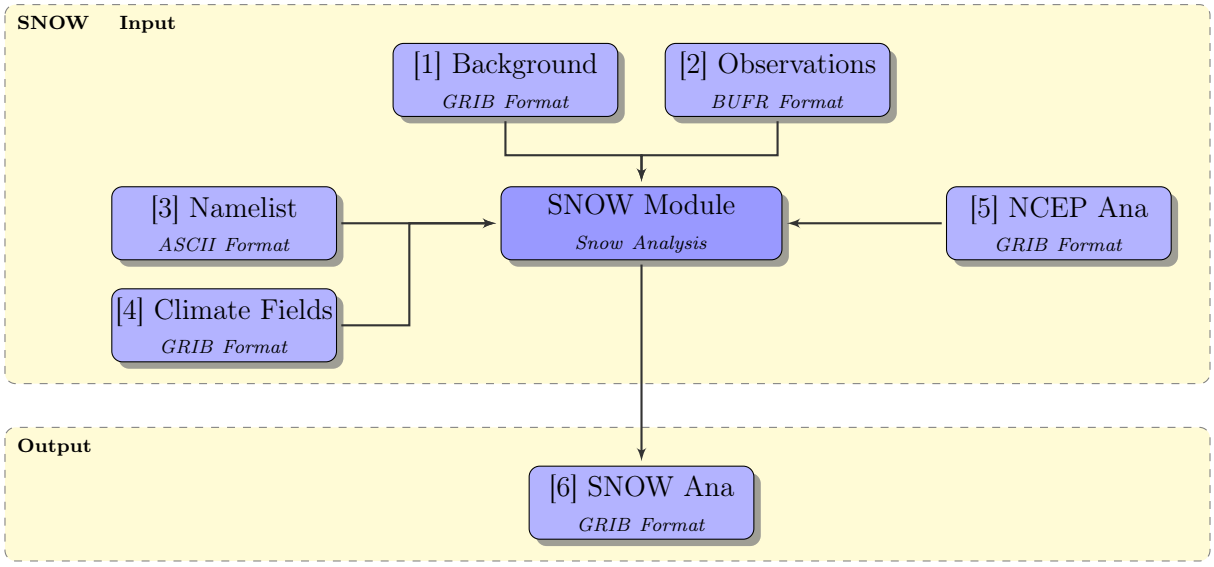


Figure 2.6: The figure displays the input and output fields of the SNOW module.

The snow analysis generates a *snow depth* (also called *snow height*). Input and output of the SNOW module are shown in Figure 2.6.

[1] http://www.ospo.noaa.gov/Products/land/snow.html

2.3.3 Soil Moisture Analysis (SMA)

Soil moisture is well-known to be an important ingredient of the atmospheric model, with strong influence on the lower part of the atmosphere and long memory effects. *Soil moisture analysis* refers to the part of the data assimilation system which adapts soil moisture to observations.

At the current state, soil moisture analysis is carried out in its own module, using mainly the near surface variables of two-meter temperature T2m and relative humidity Rh2m. The soil moisture analysis consists of two main parts:

1. *First*, an analysis of the two-meter temperatures *2Tm Analysis* is carried out. This is a global interpolation of observations from
   (a) synop stations
   (b) ships
   (c) drifting and stationary buoys.

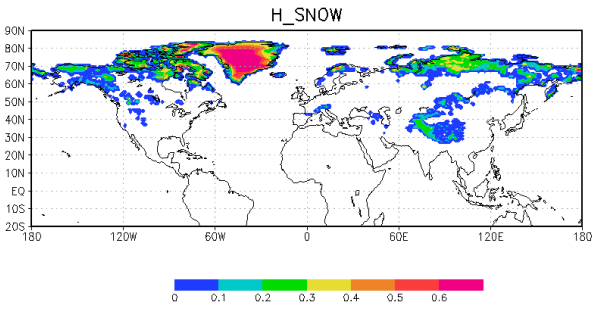
   @Roland: Ships and Buoys for SOIL moisture analysis?
2. *Second*, in the `SMA Module` the *humidity in the different soil layers* is adapted to the two-meter temperature using the difference between model T2m and the analysed T2m, based on *surface flux distributions*.

The `T2m DA Module` as visualized in Figure 2.10 cycles through the past 24h with cycle $\triangle t = 3h$. It takes the background of the T2m temperature [1] and assimilates the corresponding observations [2] of type (a) to (c) into an analysis field, the *T2m Analysis* [5]. Together with the relevant external parameter fields [4], these field are collected into a GRIB file and given as input to the `SMA Module`.

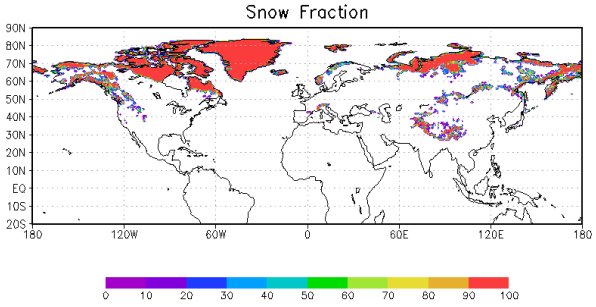
The `SMA Module` calculates increments of the *soil moisture* based on a *24h forecast* [2] which is calculated from the 0 UTC atmospheric analysis of the previous day. Since the sensitivity of the 2m temperatures to soil moisture is largest around noon, the module compares the local temperature of the forecast run with the corresponding analysis temperature from *T2m Analysis*. To this end the module cycles over *spherical wedges*

$$S(\varphi_1, \varphi_2) := \Big\{ x = r(\varphi, \theta) \in \mathbb{S} : \quad \varphi_1 \leq \varphi \leq \varphi_2 \Big\}, \tag{2.4}$$

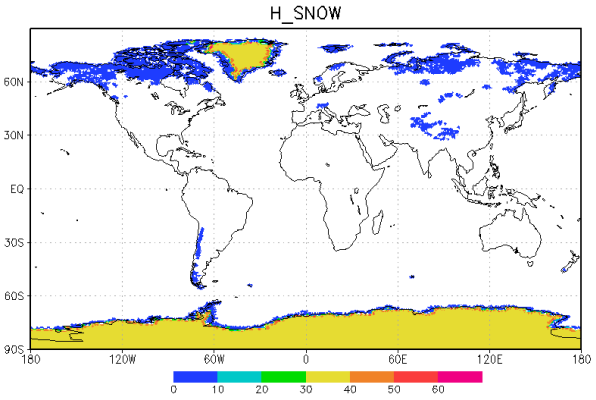
with wedge size $\pi/4$ (or $30°$) calculating a local soil increment, compare Figure 2.8, where one of the wedges is displayed.



(a)



(b)



(c)

Figure 2.7: (a) The figure visualized the NCEP snow height given in meters. Data here are given in the northern hemisphere. (b) NCEP *snow fraction* is shown in %. (c) The figure displays the *snow height* `h_snow` as generated by the SNOW module. The fields are from June 1^{st}, with summer on the northern hemisphere.

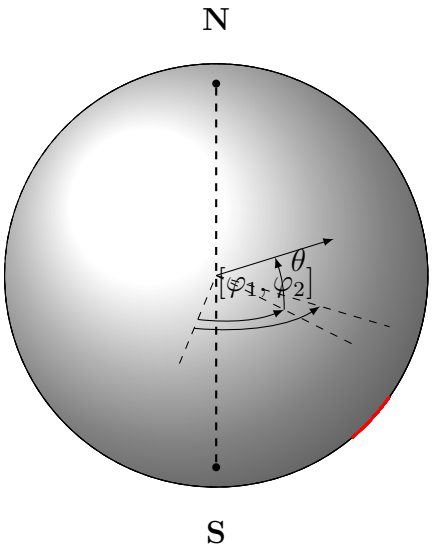


Figure 2.8: The figure shows one of the wedges which is used by the SMA to calculate soil moisture increments. The wedge is used for noon local time, when the sensitivity of the atmospheric states with respect to soil moisture is largest.

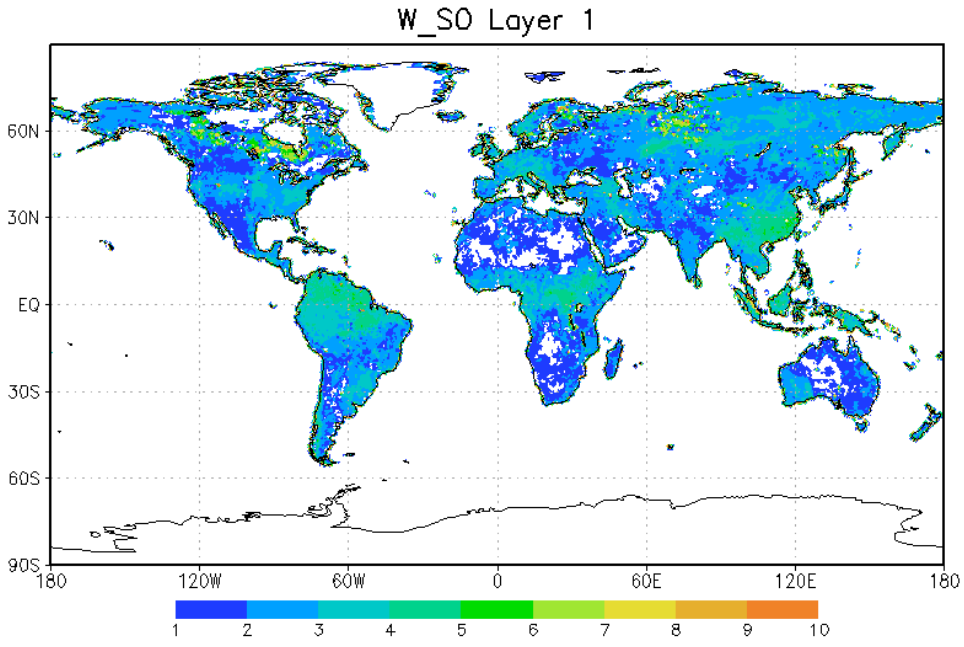


Figure 2.9: The figure shows the soil moisture in the first layer of size 1cm=10mm as generated by the SMA module, with values between 0mm and 10mm. Values above sea are set to zero and displayed in white. Here, the figure is generated by interpolation from the ICON model grid to a lat-long grid and display by `grads`.

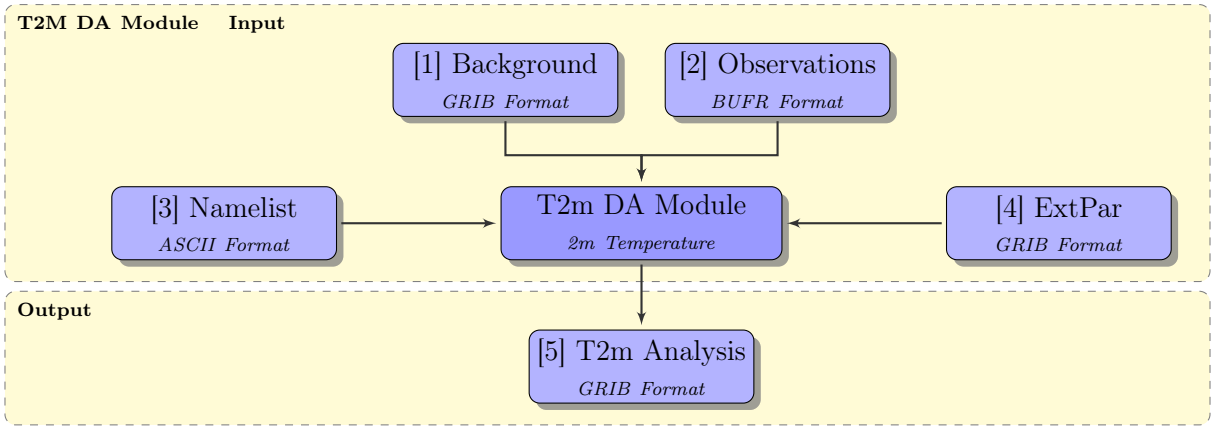


Figure 2.10: The `T2m DA Module` calculates an analysis of the 2m-temperature on the model grid for a period of 24h, using the background of the model forecasts and surface relevant observations.

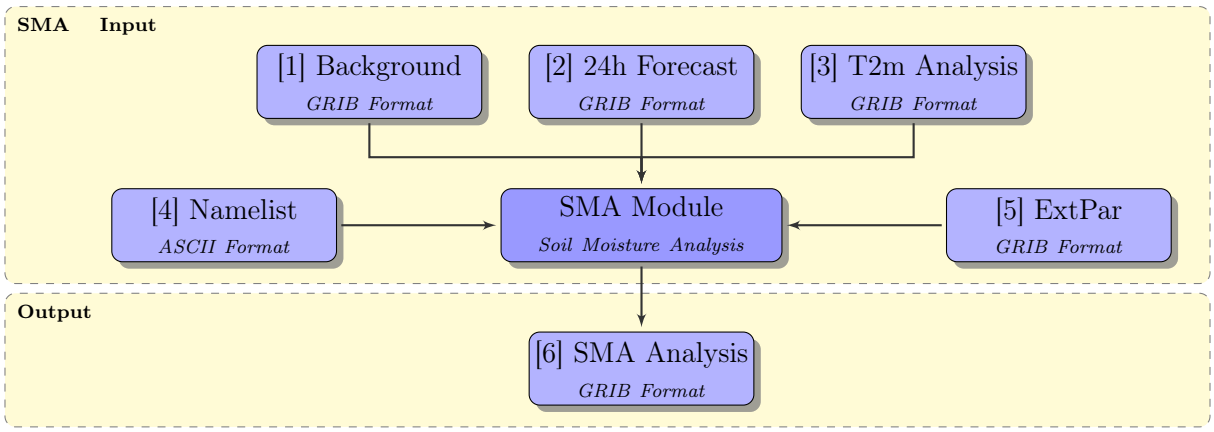


Figure 2.11:

2.4 Observations

As mentioned in 2.2, observations are the key to the determination of the current atmospheric state. This Section shortly explains how the model is compared to the data (Section 2.4.1), which observations are assimilated (Section 2.4.2) and how their quality is monitored and assessed (Section 2.4.3).

2.4.1 Observation Operators

Observation operators are the important ingredient of any data assimilation system. For some given measurements $\mathbf{y} \in \mathbb{R}^m$ with $m \in \mathbb{N}$, the *observation operator* calculates the *model equivalent* or *simulated measurement* $H(\mathbf{x}) \in \mathbb{R}^m$, where $\mathbf{x} \in \mathbb{R}^n$ is the model state. For example, consider two extremes:

- If we have *one* temperature measurement only, i.e. $\mathbf{y} \in \mathbb{R}$ in some point $\mathbf{r}$, then H interpolates the model state $\mathbf{x}$ from the model grid into its value $H(\mathbf{x})$ at $\mathbf{r}$.
- When we measure *radiances*, then H can include a full *radiative transfer model* as implemented in the RTTOV code. It simulates the radiation coming from a column of the atmosphere when a temperature profile, a humidity profile, cloud cover and several further fields are provided.

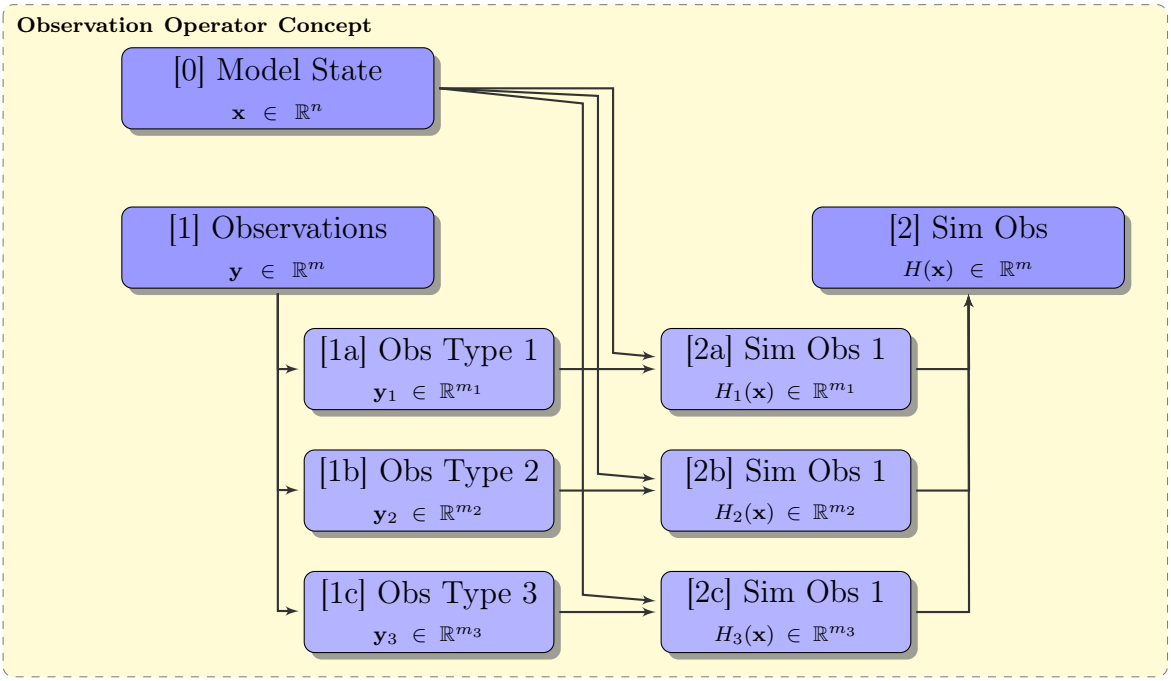


Figure 2.12: Observation operators calculate *model equivalents* $H(\mathbf{x})$, from the given model states $\mathbf{x} \in \mathbb{R}^n$.

Observation operators are used in different ways in the assimilation algorithms.

1. The three-dimensional variational assimilation actively minimizes the functional

$$J(\mathbf{x}) = \|\mathbf{x} - \mathbf{x}^b\|^2_{B^{-1}} + \|\mathbf{y} - H(\mathbf{x})\|^2_{R^{-1}}, \tag{2.5}$$

   where R and B are the *covariance matrices* reflecting the data error distribution and the background error distribution. To carry out the minimization, we need both the linearization $\mathbf{H}$ of the operator H and its *adjoint* $\mathbf{H}^*$.

2. The EnKF uses the *obs minus first guess*

$$\mathbf{y} - H(\mathbf{x}) \tag{2.6}$$

   to calculate the *analysis ensemble* when a *background ensemble* and observation data $\mathbf{y}$ are given at some point t_k in time. The EnKF does not need the adjoint operator.

In all cases we need to be able to calculate the model equivalent $H(\mathbf{x})$ for given observations $\mathbf{y}$. A full documentation of a data assimilation system needs to describe all observation operators which are integrated into the system.

2.4.2 Observation Types

A short overview about the observation types used by DWD is given in the following Sections. More details about *observation operators* are provided in Chapter 6.

SYNOPS, Ship and Buoy reports

SYNOP data are the classical weather measurements obtained from automated and manual land stations, buoys and ships. Meteorological quantities provided by SYNOP data are temperature, moisture, cloud state, dew point, wind speed and direction, visibility, pressure, weather state, precipitation and snow.

DWD operationally assimilates classical SYNOP data and buoy reports in all of its NWP systems (the global GME model, the European COSMO-EU and the central European COSMO-DE).

Radiosondes, Dropsondes (TEMP), PILOTs and ASAP (Ships)

Observations of upper regions of the atmosphere are made by weather balloons (TEMP) released from surface level or dropped from planes, measuring temperature, pressure, humidity.

Wind Observations in the Atmosphere by Satellites, Passive Methods (AMVs)

Using the displacement vectors of clouds the wind speed and wind direction is calculated. The technique is known as AMV (Atmospheric Motion Vectors). Geostationary satellites provide observations with high time resolution.

AMV winds are operationally assimilated at DWD within its global model GME.

Wind Observations in the Atmosphere from Satellites, Active Methods

With a launch planned for 2014, a polar orbiting satellite equipped with a Lidar system (ADM (Atmospheric dynamics mission)-AEOLUS) will reconstruct wind information from the Doppler shift of backscattered signals in the line of sight within the complete atmosphere. DWD is represented in the Aeolus working groups and works on the assimilation of the corresponding data in cooperation with DLR.

Aircraft Measurements (AIREPS, AMDAR, ACARS, ASDAR)

Airplanes provide measurements of temperature, pressure, wind and humidity (experimental). DWD operationally assimilates all the available aircraft data into all three models GME, COSMO-EU and COSMO-DE.

Ground-Based Radar Stations

Reflectivity measurements by ground-based radar stations provide information about precipitation and winds. This information is usually calculated from measurements with particular products, currently precip-scans of DWD and temperature, pressure, humidity.

The assimilation of radial winds is performed for COnsortium for Small-scale MOdelling (COSMO)-model.

Radar Wind Profiler

Radar wind profiler are Doppler radars using lower transmit frequencies than classical weather radars, which allows getting echo returns from both the clear and the cloudy atmosphere. Clear air echoes are caused by the ubiquitous fluctuations of the refractive index of air. The vertical data coverage depends on the used wavelength, for the DWD 482 MHz systems (62 cm wavelength), measurements are typically confined to a height range between 0.5 and 16 km agl. Radar wind profiler networks are for example run by the EUMETNET Programme E-PROFILE http://www.eumetnet.eu/e-profile.

Satellite-Based Active Radar

Using active radar (scatterometer) satellites measure the intensity of radiation scattered back from the sea surface. It depends on the roughness of the surface, which is again a function of wind speed close to the surface (10m). Using measurements from different observation angles it is possible to also determine the wind direction by advanced scatterometer observations (ASCAT). DWD operationally assimilates ASCAT measurements within the global model GME, tests are being run with COSMO-EU.

Radio Occultations

Radio Occultations (RO) is a remote sensing measurement technique to probe the planetary atmosphere using radio signals which pass from some sender to some receiver, where usually both sender and receiver are outside of the atmosphere under consideration. Usually one employs simplified models of ray bending for analysis. The basic physical idea is to determine approximate values of the refractive index at the bending points, the limb-sounding being sensitive to humidity and temperature.

Global Navigation Satellite System (GNSS)

Global navigation systems like GPS (USA), Galileo (EU), GLONASS (Russia) or COMPASS (China) are sending signals, which are received by ground based stations. These signals are delayed due to atmospheric parameters; the path through the atmosphere differs from the geometrical path. The delay is determined by measuring the time difference between sending and receiving the signal, and it is proportional to the integrated water vapour (IWV) of the ray's path through the atmosphere.

Within its COSMO-DE NWP system DWD is ready for assimilating this type of data and experiments are running.

Microwave and Infrared Sounding Instruments on-board of Satellites

Satellite borne sounder instruments are not able to measure meteorological quantities directly, rather they measure the upwelling radiation emitted by the Earth's atmosphere

and surface in narrow spectral bands (so-called channels). The radiation measured by a satellite instrument depends in a complex way on the vertical distribution of temperature and the atmospheric composition, on hydrometers and on the Earth's surface properties. In order to assimilate such data in NWP models, the complicated relationship between the observed radiation and the meteorological variables (the state vector of the NWP model) is simplified and modeled numerically in an efficient way. At the moment these measurements are the only way to obtain near-real-time information on the atmosphere with global coverage and high horizontal resolution.

2.4.3 Quality Control

Measurements of atmospheric fields are necessarily imperfect and tainted by errors. Before incorporating the data in the assimilation algorithms, it has to pass quality checks (a full description is provided in Section 8.1):

- Parts of data which are known to be bad can be *blacklisted*.
- Data is *selected* by its geographical distribution (e.g land/sea).
- A *first guess check* is performed wherein the departure of the observation from the model equivalent is compared with their respective prescribed standard deviation values.
- Observations with an observation error that is too large are excluded.
- Spatially and temporally dense observation sets tend to be correlated and are therefore redundant. *Observation thinning* is performed to remove the redundant information as far as possible.

The probability density function for observation errors (PDF) is not necessarily Normal: outliers may exist and nonlinear observation operators such as radiances will produce non-Gaussian estimates, making the *cost function* (which is to be minimized) also non-Gaussian. In the case of the DA schemes that use a variational solving method (3dVar and VarEnKF), it is possible to use apply a *Variational quality control* (Section 8.2) to account for this issue.

Chapter 3

DA Basic Cycling

3.1 The Global Basic Cycling Environment

An introduction on how cycled data assimilation is performed with the DWD systems is provided here. The User Guide explains it in Detail in Chapter 15.

3.1.1 The Design of Basic Cycling

The *basic cycling system* is a development environment for data assimilation. As described in Section 2, data assimilation cannot determine the atmospheric state at a given point in time from data alone.

- The process of short-range model predictions and the fusion of data with the model *first guess* or *background*, respectively, is an indispensable tool for generating the estimate for the state of the atmosphere and earth's surface.
- The *cycling process* is of key importance for the quality and behavior of the whole NWP system. Many parts of the data assimilation system interact with the model dynamics, such as *bias correction* algorithms. Further, various parts of the model dynamics react on the increments which are generated by the data assimilation components, such that the calibration of the entire cycled system is more than half of the work which is carried out to develop and run data assimilation modules.

3.1.2 Technical perspective on Cycling

The idea of the basic cycle is to have a tool for development and debugging of each of its modules. To this end, a careful treatment of the input and output of all modules and its file processing is realized.

Figure 3.1 examplifies the basic idea of a cycling environment. In a fixed interval Δt_{fc}, the data assimilation is performed using model background files and observation files. Then, a model forecast is started to the next assimilation time (here: 3 hours later) where the assimilation is repeated.

The observation operator is applied differently for global and local assimilation methods: The global 3dVar, LETKF and VarEnKF algorithms apply the operator directly onto the

model forecast(s) within the Core DA module. In the local Kilometre-Scale Ensemble Data Assimilation (KENDA)-LETKF, the observations are processed by the ensemble of COSMO members and communicated to the Core DA together with the mapped model equivalents by using feedback files.

From certain analysis points (e.g. every 6 hours), a model forecast run is started from which the actual forecast products are derived which are disseminated by the weather service.

The model forecasts in between start from the analysis states of the assimilation and make use of surface boundary data from the SMA, SST, ICE and SNOW modules – in the local model case also lateral boundary conditions from global model forecasts.

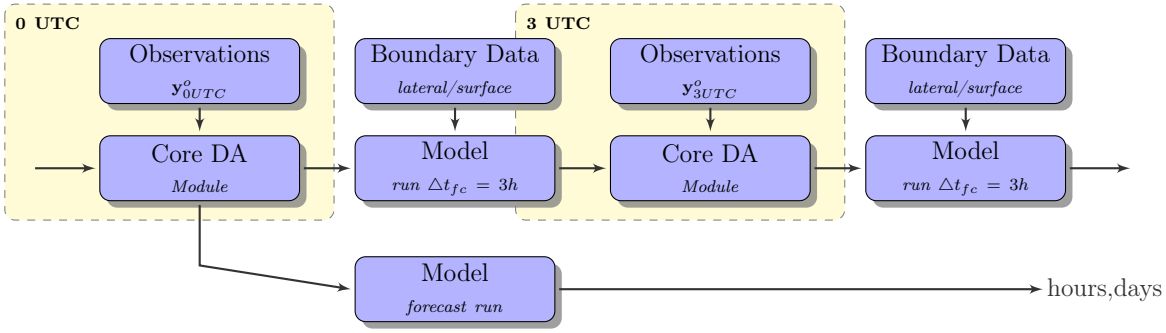


Figure 3.1: Example for a cycling system where an analysis is produced every 3 hours, and a forecast run is started from the analysis at 0 UTC.

3.2 Ensemble data assimilation

Technically, the cycling of ensemble data assimilations like LETKF or VarEnKF is similar to the cycling of a deterministic assimilation method as 3dVar.

The biggest difference is that instead of 1 deterministic model run, the forecast model has to be started L times from ensemble initial conditions, producing an ensemble forecast (eventually using ensemble boundary conditions). A detailed view on Kilometre-Scale Ensemble Data Assimilation (KENDA) is provided in the User Guide in Section 15.3.

Part II

Scientific Documentation

Chapter 4

Overview

The simulation of the dynamics of the atmosphere is one of the key ingredients of the art of modern weather prediction. To this end, global and regional systems have been developed in all operational centers around the world. Besides numerical simulations of the atmosphere, measurements give us a distinct second source of information to determine the current state of the atmospheric system. Data assimilation is a way of combining these two sources of information, observations and the prediction in order to obtain improved representation of the atmosphere for the numerical model that we are using. Only with a precise knowledge of the current atmospheric state we can hope to achieve reliable predictions for this state in the future.

Data assimilation algorithms usually start with some knowledge about the atmospheric state which is obtained from earlier measurements and forecasts. This state is called the background or *forecast* state $\mathbf{x}^b$. Using measurement data $\mathbf{y}$ at fixed time k we change the background state to a new improved estimate for the true state of the atmosphere. The estimate is called the *analysis* $\mathbf{x}^a$.

Today, a variety of measurements are available, which can be used to infer knowledge about the current atmospheric state. This includes conventional measurements of temperature, atmospheric pressure, humidity and wind either by ground-based stations or by weather balloons (radio-sondes), as well as, a variety of remote sensing data. This includes radar measurements of precipitation or wind speed, scatterometers which can infer knowledge about the wind speed at the ocean surface and various radiometers which provide knowledge about temperature and humidity in the atmosphere by measurements of infrared or near-visible radiation of atmospheric gases.

In this document we describe a class of data assimilation systems as they are are developed and used operationally at the DWD (German Weather Service). These include ensemble data assimilation systems, the three-dimensional variational methods as well as hybrid variational-ensemble filter methods.

This Part II of the document is organized to provide an understanding of the different elements that go into the assimilation algorithms and of the algorithms themselves:

1. Chapter 5 provides a description of the representation of the forecast and analysis state $\mathbf{x}^b$ and $\mathbf{x}^a$, in terms of supported atmospheric models, prognostic variables, and grids used for discretization.

2. As the next ingredient, the observations $\mathbf{y}$ and the observation operators $H(\mathbf{x})$ are described in Chapter 6. There the overview of different observing systems is provided, with measurements they produce and how the model state is projected onto these measurements using observation operators. This includes in-situ observations achieved from radiosondes and airplanes, surface observations from stations, ships and buys, satellite radiances, GPS-products and radar data.

3. With $\mathbf{x}^b$, $\mathbf{y}$ and $H(\mathbf{x})$ as input, $\mathbf{x}^a$ can be computed. The first assimilation algorithm described in this document is 3-dimensional Variational Data Assimilation (3dVar) (Chapter 7): The key idea is to reformulate the assimilation task as a large-scale optimization problem which is carried out independently at every analysis time step under consideration. Section 7.1 gives a derivation of the optimization functional $J(\mathbf{x})$ starting with a *Bayesian approach* and searching the analysis state for a maximum likelihood estimator. The minimization of the functional $J(\mathbf{x})$ is carried out by an iterative Newton method and a *conjugate gradient scheme* to solve the systems which arise within this framework. Section 7.2 is devoted to the construction of the background error covariance matrix $\mathbf{B}$ used for the analysis-computation in 3dVar.

4. Chapter 8 contains the methods for the *treatment of observations*:

   - Before entering the assimilation process, quality control and thinning of observations have to be performed. Quality control usually starts with generic quality checks, assuming for example that measurements are not likely to be completely different from a calculated *model equivalent*. Then, the reduction of the amount of data is carried out by *thinning* algorithms. We describe the methods in Section 8.1.
   - Specific to 3dVar, stochastic and variational algorithms provide more advanced tools to deal with data quality than simple quality control checks. To take gross errors into account, we employ a non-Gaussian description of observation errors. As a consequence less weights than in the Gaussian formulation are employed for large distances of measurements to the analysis. This is described in *Variational Quality Control* (Section 8.2).
   - Specific to 3dVar, Section 8.3 summarizes details on the assimilation of humidity. It summaries humidity sensitive observations, describes the treatment of the stratosphere and of cloud liquid water and ice content, and introduces the transformation to the generalized humidity variable used as control variable by the background error covariance model.

5. In Chapter 9, the Ensemble Kalman Filter (EnKF) is introduced, wherein an ensemble of the forecast model is used in a Monte Carlo approach to sample, estimate and propagate the flow dependent background error covariances (here referred to as the matrix $\mathbf{P}^b$). The implementation of the local and highly parallel LETKF-algorithm is documented as well as the strategies on *covariance localization* and (adaptive) *covariance inflation*. As the solutions supported by a sampled $\mathbf{P}^b$ can be non-physical, constraints like *positivity*, *saturation adjustment* and *hydrostatic balancing of analysis increments* are presented. To account for different characteristics

of observations, a *multistep analysis* scheme and a method for *adaptive localization* are derived.

6. The variational 3dVar scheme has the advantage of a well-developed background error covariance model, but it is static. On the other hand, the EnKF provides flow-dependent background error covariances, but these are tainted by sampling error. The strengths of these two schemes are combined in a Hybrid Variational Ensemble Kalman Filter (VarEnKF) (Chapter 10) that uses a mix of the static $\mathbf{B}$ matrix of 3dVar and the flow-dependent $\mathbf{P}^b$-estimate of the EnKF in order to generate stable analysis that incorporate more information from the observations than the pure 3dVar could.

7. All NWP models need lower boundary conditions of sea surface temperature, sea ice coverage, soil moisture and snow coverage. These are provided by the supplemental modules SST, ICE, SMA and SNOW, documented in Chapter 11.

Chapter 5

Forecast Model State

The background and analysis states $\mathbf{x}^b$ and $\mathbf{x}^a$ introduced in Chapter 2 usually consists of all prognostic variables of the atmospheric model, discretized on the respective grid. The formulation of the 3dVar-PSAS (Physical Space Assimilation System), EnKF and VarEnKF does not depend on a specific grid so that different models can be handled. However, specification of the observation operators and appropriate covariances of data assimilation system depend on the grid, and on the scales that the particular model can resolve.

Section 5.1 gives an overview of the amount of support provided for different models. The prognostic model variables are described in Section 5.2. The control variables needed for implementation of the 3dVar are introduced in Section 5.3. The supported horizontal and vertical grids are further described in Sections 5.4 and 5.5.

5.1 Supported Models

GME
: GME is the operational global forecast model of DWD. It is a hydrostatic grid-point model formulated on an icosahedral horizontal Arakawa-A grid and a vertical hybrid pressure coordinate system. The GME is fully supported by the 3dVar, EnKF and VarEnKF (under development).

HRM
: The HRM is the former mesoscale forecast model of DWD. It is a hydrostatic grid-point model formulated on a rotated regular Arakawa-C grid and a vertical hybrid pressure coordinate system. The HRM is fully supported by the 3dVar.

COnsortium for Small-scale MOdelling (COSMO)
: COSMO is the operational regional forecast model of DWD and the COSMO consortium. At DWD it is used in two different configurations, covering the European (COSMO-EU) and the German (COSMO-DE) area. COSMO is a non-hydrostatic grid-point model formulated on a rotated regular Arakawa-C grid in vertical z-coordinates. COSMO is supported by the EnKF (Chapter 9).

ICON

ICON is the future non-hydrostatic global forecast model of DWD. It is formulated on an horizontal icosahedral staggered grid with an option for local refinments. ICON is the prospective successor of the GME and COSMO-EU. The adaption of the 3dVar to ICON and the formulation of a hybrid 3dVar/EnKF is under development.

IFS

The IFS is the operational global forecast model of ECMWF. It is a hydrostatic spectral model formulated on a vertical hybrid pressure coordinate system. The transformation from IFS model grid to the horizontal Gaussian grid is handled within 3dVar algorithm. This capability is used for diagnostic purposes, i.e. for running the 3dVar observation operators on IFS analyses or forecasts.

Further information may be found in the model documentation for GME, COSMO, HRM, or IFS.

5.2 Prognostic Model Variables

Prognostic variables uniquely define the state of the forecast model and, based on the model equations, its future state. The initial values of these variables have to be provided as for the integration of the dynamical model by the assimilation system.

The result of 3dVar, EnKF and VarEnkf is the analysis of the atmospheric temperature, wind and humidity fields. Other components of the NWP model are analysed by separate components of the analysis system, introduced in Chapter 11 i.e. the sea surface temperature (SST) and sea ice analysis, the snow analysis, and the soil moisture analysis.

The number and kind of prognostic variables depends on the formulation of the model dynamics (hydrostatic or non-hydrostatic) and the vertical coordinate used (pressure or geometrical height). Below we describe the set of variables used by the 3dVar for GME, HRM, and IFS and by the ensemble Kalman filter for COSMO and ICON.

GME, HRM, IFS

These models are based on the hydrostatic equations in a vertical pressure coordinate system. Prognostic variables analysed by the 3dVar are surface pressure p_s, temperature T, and horizontal wind components u and v. The vertical wind component (ω in pressure coordinates) is a diagnostic quantity which is derived by the continuity equation. Furthermore water vapour content, i.e. specific humidity q is analysed.

The moisture variables cloud liquid water content q_{cl}, cloud ice content q_{ci}, prognostic rain q_r, and snow q_s are not analysed in the 3dVar-algorithm as their relationships to the other dynamical quantities are strongly nonlinear and the time scale of their dynamical evolution is much shorter. However, they are used in the hydrostatic equation to derive geopotential height. Merely cloud water content is set to zero if analysed relative humidity falls below 90 % in order to keep these prognostic quantities consistent.

In addition the diagnostic variables surface temperature T_s and sea ice coverage are required by the 3dVar as certain observation operators (satellite radiances) depend on the surface characteristics.

COSMO

The additional degrees of freedom of the non-hydrostatic COSMO model are reflected by further prognostic variables: vertical velocity w and pressure p. Internally, p is described by the perbation pressure p' with respect to a reference pressure p_0: $p = p_0 + p'$.

COSMO contains the moisture variables specific humidity q, cloud liquid water content q_{cl}, cloud ice content q_{ci}, prognostic rain q_r, snow q_s and graupel q_g.

ICON

Missing: ICON prognostic variables

5.3 Control Variables in 3dVar

Specific to the 3dVar algorithm (the EnKF does not use control variables), the internal representation of the deviations from the background atmospheric state $\delta\mathbf{x} = \mathbf{x} - \mathbf{x}^b$ differs from that of the dynamical model because i) correlations between the variables in this representation are small so that they may be approximated as zero entries in the matrix $\mathbf{P}^b$ and ii) background error distributions are more Gaussian so that the assumption of a quadratic cost functional J_b is more appropriate. Specifically the representation is as follows:

Wind field

Instead of the horizontal wind components u and v the velocity potential χ and stream function ψ are used for internal representation. The wind components are derived from these integral quantities by differentiation in the horizontal direction:

$$\begin{aligned} u &= d/dx\,\chi - d/dy\,\psi \\ v &= d/dy\,\chi + d/dx\,\psi \end{aligned} \tag{5.1}$$

Mass field

Instead of surface pressure p_s and temperature T geopotential height h is used. This quantity is related to the density of the air ρ by the hydrostatic equation:

$$\frac{\partial p}{\partial h} = -g\rho \tag{5.2}$$

taking a constant value for gravity acceleration g. Air density relates to temperature by the equation of state:

$$\rho = \frac{p}{R\,t_v} \tag{5.3}$$

For moist air, virtual temperature T_v differs from T and is given by

$$t_v = (1 + (R_v/R - 1)\, q - q_x)\, t \tag{5.4}$$

depending on temperature T, specific humidity q and the liquid and ice content of clouds and precipitation $q_x = q_{cl} + q_{ci} + q_r + q_s$. R and R_v denote the gas constants of dry air and water vapour.

Humidity

Instead of specific humidity q relative humidity over water rh_w, or more specifically generalised humidity gh is used. The transformation to generalised humidity is made because relative humidity is restricted to values $rh_w \geq 0$ (and, if no super-saturation is allowed, to $rh_w \leq 1$ as well). A Gaussian error distribution cannot be assumed in this case. Generalised humidity is chosen to be identical to relative humidity for $rh_w > 3\%$. Below this value a nonlinear relationship is used so that rh_w remains positive for any value of gh.

The background error covariances are either prescribed analytically, or represented in discretized form. In the latter case the underlying grid may differ from the model grid. More details on the representation of the background error covariances in terms of gh, h, ψ, and χ are given in Chapter 7.2.

In addition to the model state variables additional control variables are used in the 3dVar:

Dummy sink variables η

The purpose of these control variables is to account for errors of representativeness which cannot be taken into consideration by an increase of the observational error. In this case the observation operator also depends on these additional control variables: $H(\mathbf{x}, \eta)$. Dummy sink variables are currently used by the satellite radiance observation operator to model the uncertainties of surface temperature as well as atmospheric temperature and humidity profiles above model top.

Persistent external parameters

The purpose of these control variables is to estimate external parameters within the variational scheme, which account for a persistent bias of the observations or model. They will be used for the bias correction coefficients of the satellite radiance observations and of individual stations or platforms (SYNOP surface pressure and aircraft temperatures).

5.4 Horizontal Grid

A variety of horizontal grids is supported by the assimilation algorithms:

Icosahedral Grid

The grid-points of the GME are located at the vertices of triangles. In general each grid-point has 6 neighbours (but only 5 for 20 exceptional ones). Every point on the sphere is

surrounded by 3 model grid-points which enter the horizontal interpolation. For details refer to the GME documentation.

Reference to GME-Doku

Reduced Gaussian Grid

The native grid-point representation of the spectral IFS is a Gaussian grid, i.e. a latitude-longitude grid with non-equidistant spacing in the meridional direction. The specific non-equidistant spacing has advantages for the Legendre transformation used in the spectral formulation of the dynamical core. The IFS furthermore uses a reduced grid with less zonal points near the poles than at the equator. Fields on a reduced grid are interpolated on input and hold on a regular grid within the 3dVar.

Rotated Regular Grid

The regional COSMO and HRM models are formulated on a regular latitude-longitude grid. Their model grid may be rotated with respect to the geographical coordinates so that the model domain is located near the equator of the model grid, so that deviations from a Cartesian coordinate system are small.

Arakawa-C grid

In the COSMO and HRM model the grid-points of the wind components are shifted by half a grid cell in zonal direction for u and in meridional direction for v. This has advantages for the numerical representation of certain operators (advection, continuity equation). This kind of model grid is called Arakawa-C grid. Within the EnKF, all fields are interpolated to the same grid-points (Arakawa-A grid) on input. Analysis increments are then derived on the A-grid and re-interpolated to the C-grid. The final analysis is calculated on the model C-grid.

5.5 Vertical Grid

Hybrid Pressure Coordinates

GME, HRM, and IFS use hybrid pressure coordinates. In the upper troposphere pressure p is the vertical coordinate, whereas near the surface coordinate lines follow the orography. The pressure values of a vertical grid point k are calculated as $p_k = a_k + b_k\, p_s$ for given surface pressure p_s and fixed constants a_k, b_k with $a_k = 0$ and $b_k = 1$ at the surface and $b_k = 0$ at model top.

Chapter 6

Observation Operators

The observation operator $H(\mathbf{x})$ derives the equivalents to the observations $\mathbf{y}^o$ from the model state $\mathbf{x}$.

In the algorithm for the global models in the DWD DA system (3dVar, global Local Ensemble Transform Kalman Filter (LETKF), VarEnKF), H is applied directly before the assimilation on the background fields. The respective observation operators are described in Section 6.1.

Furthermore, the variational 3dVar-algorithm (Chapter 7) uses the tangent linear operator which calculates the product of the Jacobian of H with a vector ($\mathbf{H} \cdot \mathbf{x}'$), and the adjoint operator which calculates the product of the transposed Jacobian ($\mathbf{H}^T \cdot \mathbf{y}'$, or $\mathbf{y}' \cdot \mathbf{H}$) (see Section 6.1.6).

In the cycling of the EnKF for the local models (Chapter 9), the observation operators are applied within the ensemble members of the COSMO model itself at model runtime. A description of the COSMO-Operators is provided in Section 6.2.

6.1 Observation operators for global models

In this section we describe the different observation types used within the algorithms for global models (3dVar, global LETKF, VarEnKF) and give a short description of the observation operators.

For point-observations, these operators perform bi-linear horizontal interpolation from the model grid to the location of the observation.

In Section 6.1.2 and 6.1.3 we describe in-situ observations and surface observations, which require comparatively easy observation operators. The more complex observations from satellite radiances and radio occultation are described in section 6.1.4 and 6.1.5.

6.1.1 Physical quantities used within the 3dVar

Check whether this section is relevant for the application of global observation operators or only for 3dVar

During observation processing different representations of atmospheric state variables are used (cf. table 6.1):

	Prognostic variables (GME, HRM)	Control variables	Interpolated quantities	Assimilated variables	Reported data
humidity	q	gh	rh_w	rh_w	rh, q, m, t_d
wind	(u, v)	(ψ, χ)	(u, v)	(u, v)	$(ff, dd), (u, v)$
mass	(t, p_s)	h	h, t_v	(t, h_s)	(t, p_s)

Table 6.1: Physical values within GME, 3dVar and observations.

Prognostic model variables

The state variables temperature t, surface pressure p_s, specific humidity q, and wind components u, v of the atmospheric forecast model were described in section 5.2.

Control variables of the 3dVar

The representation of the state variables in terms of generalised humidity gh, geopotential height h, streamfunction ψ and velocity potential χ used by the background error covariance model of the 3dVar was briefly introduced in section 5.3. More details will be given in chapter 7.2.

Variables used by the interpolation operators

The observation operators consist of two parts:

1. The interpolation operator H_I which interpolates the model quantities to the location of the observation.
2. Tte actual observation operator H_O which derived the observed quantities.

All Observation operators use a common set of interpolation operators H_I. Interpolation is performed in terms of virtual temperature t_v or geopotential height h, wind components u, v, and relative humidity over water rh_w.

Reported quantities

The reported quantities, i.e. the quantities given in the input (BUFR) files, are not necesarily identical to the raw observations. For in-situ observations frequently reported quantities are: wind components u and v or wind speed ff and direction dd, and relative humidity rh, specific humidity q, mixing ratio m, or dew point temperature t_d.

Assimilated quantities

In general the set of assimilated quantities $\mathbf{y}^o$ which appears in the observational cost function J_o (7.11),(7.12) should be the same as the observed set. In that case the variables are most likely uncorrelated so that the specification of observational errors is simplified. Nevertheless for in-situ observations we use a common set of variables independently from the observed one, namely temperature t, geopotential height h_s (for a given pressure), wind components u, v, and relative humidity rh_w.

6.1.2 In-situ Observations

In-situ observations directly measure the atmospheric quantities temperature t, pressure p, wind components u and v (or alternatively wind speed ff and direction dd) and humidity q within the atmosphere. Furthermore some products (AMV, SATEM) derived from remote sensing satellite observations (provided by the data providers) are handled similarly by the data assimilation system. The native or pre-processed in-situ observations used are listed below.

In-situ Observation Types

The mnemonics TEMP, PILOT, AIREP, SATOP, SATEM are used within the 3dVar to distinguish different observation types. They are borrowed from the World Meteorological Organisation (WMO) alpha-numerical codes used for reporting the respective observations. The mnemonics AIREP and SATOB are used here in a more general way to denote all aircraft and all AMV observations, and not only that specified by the original code.

Radiosondes, Dropsondes, Ships [1] Observations of upper regions of the atmosphere are made by weather balloons released from surface level or dropped from planes. A weather balloon carries instruments to send information back to the ground stations, usually temperature, pressure, humidity. To obtain wind information, they can be tracked by radar or GPS navigation signals. Measurements of radiosondes launched from ships are managed by the Automated Shipboard Aerological Programme (ASAP) of Eumetnet.

TEMP : Observed quantities provided and used within the assimilation are temperature t, wind components u, v and humidity q as a function of the vertical pressure coordinate p. In addition to temperature radiosondes report geopotential height h, which is not assimilated as it is a redundant quantity derived by the hydrostatic equation from the observed air density $\rho\,(t, q, p)$.

PILOT : Similar to TEMP, but only wind (ff, dd) is reported.

Aircrafts [2] Observations from an in-flight aircraft. These reports describe weather conditions at flight level in the upper layers of the atmosphere or profiles measured in the starting or landing phase.

---

[1] Radiosondes, Dropsondes, Ships:

1. WMO station list
2. EUMETNET Automated Shipboard Aerological Programme (ASAP)

[2] Aircrafts:

1. WMO AMDAR
2. EUMETNET Aircraft Meteorological Data Relay (AMDAR)

AIREP : Meteorological observations are wind, air temperature, cloud amount, cloud cover, cloud height (cloud base and cloud top), and special phenomena. Temperature (t) and wind (u, v) observations are currently assimilated. Usage of humidity (q) is investigated.

Airplane measurements are transmitted by AIREP messages for particular degrees of longitude or continuously by AMDAR (Aircraft Meteorological Data Relay) transmission. There are two ways of data communication used here, standard VHF radio signal (ACARS, Aircraft Communications Addressing and Reporting System) and satellites (ASDAR, Aircraft to Satellite Data Relay). The AMDAR transmission provides information about vertical profiles close to airports, since it is also available from starting and landing airplanes.

Atmospheric Motion Vectors (AMV) [3] Using the displacement vectors of clouds the wind speed and wind direction is calculated. The technique is known as AMV (Atmospheric Motion Vectors). Geostationary satellites provide observations with high time resolution between +/-60 degrees latitude, polar orbiting satellites provide similar data for the polar regions. The use of different spectral channels leads to wind speed reconstructions in different heights. AMV wind vectors are operationally assimilated at DWD from the geostationary satellites GOES-11, GOES-12, Meteosat-7, Meteosat-9, MTSAT-1R, MTSAT-2R, as well as polar orbiting satellites from Metop, the NOAA series, AQUA and TERRA. The geostationary satellite FY-2D is currently being monitored for possible future operational use. Spectral channels in the visible, infrared and microwave range are employed. Particular instruments or products are SEVIRI (Meteosat), AVHRR (Metop, NOAA) and MODIS (Aqua, Terra).

SATOB : Wind vectors (u, v) derived from drift of clouds or humidity patterns in satellite observations. These products are provided by the data providers (EUMETSAT, NESDIS).

Atmospheric thickness Satellite temperature and humidity soundings that are converted to atmospheric thickness. Assimilation of SATEM reports is implemented only for test purposes as direct assimilation of radiance data is preferred in variational data assimilation schemes.

SATEM : Atmospheric thickness $\triangle h$ (i.e. geopotential height difference between two pressure levels provided by EUMETSAT or NESDIS.

[3] AMVs:

1. KENNETH HOLMLUND, The Utilization of Statistical Properties of Satellite-Derived Atmospheric Motion Vectors to Derive Quality Indicators, Weather and Forecasting 1998
2. NIELS BORMANN, SAMI SAARINEN, GRAEME KELLY, AND JEAN-NOEL THEPAUT: The Spatial Structure of Observation Errors in Atmospheric Motion Vectors from Geostationary Satellite Data, Monthly weather review, Vol. 131, 2003.
3. "What is AVHRR?"

In-situ Observation Operators

The operation operator for in-situ data consists of:

1. Vertical interpolation H_v to the pressure level of the observation.
   Interpolation is performed in terms of temperature t, generalised humidity gh and wind components u, v. Linear interpolation is used in case of humidity, spline interpolation in case of the other quantities.

2. Horizontal interpolation H_h to the location of the observation.
   Bi-linear interpolation is used for all quantities.

3. Calculation of the observed quantity (H_o).
   This is the identity operator in case of temperature and wind. gh is converted to relative humidity rh (over water). If other moisture variables are reported (in general dew point temperature t_d), they are converted to rh_w before assimilation.

6.1.3 Surface Observations

Surface observations are observations from weather stations or or satellites close to the surface, e.g. at 2 or 10 meter height. Contrary to in-situ observations surface observations are horizontally interpolated only. Some quantities (surface pressure) require additional reduction or extrapolation to account for the heigt difference between model orography and observation.

Surface Observation Types

SYNOPs, Ship and Buoy reports [4] SYNOP data are the classical weather measurements obtained from automated and manual land stations and ships. They are provided by public and private measurement networks worldwide. Synoptic measurements are compiled using the particular SYNOP format for transmission. Meteorological quantities provided by SYNOP data are temperature, moisture, cloud state, dew point, wind speed and direction, visibility, pressure, weather state, precipitation and snow state and dynamics. Buoys provide measurements of pressure, wind and sea surface temperature (SST), transmitted by satellite.

- **SYNOP** : Reports by manned and unmanned weather stations or ships, containing temperature t_{2m} and humidity t_{d2m} (usually at 2 m height) wind u_{10m}, v_{10m} (at 10m height) and surface pressure p_s.
- **DRIBU** : Drifting buoys, measuring temperature t_{2m}, wind u, v, humidity q_{2m}, and surface pressure p_s.

Pseudo Pressure Observations [5] The term "PAOB" is an abbreviation derived from "paid observation" which was pointing to sea level pressure values derived by manual

[4] SYNOPs, Ship and Buoy reports:
Federal Meteorological Handbook, Synoptic Codes

[5] PAOBs:
The history of PAOBs in the Australian Bureau of Meteorology

analysis (later by fitting algorithms employed by numerical systems) to provide data for areas in the southern hemisphere which were not covered otherwise. Today PAOBs are automatically extracted from digital output of onscreen analysis.

PAOB : Pseudo surface pressure p_s derived from satellite data. Disemination of PAOB reports was stopped at 2010/08/18 .

Satellite-Based active radar [6] Using active radar (scatterometer) satellites measure the intensity of radiation scattered back from the sea surface. Backscattered radar intensity depends on the roughness of the surface, which is again a function of wind speed close to the surface (10m). Using measurements from different observation angles it is possible to also determine the wind direction (with some ambiguities) by advanced scatterometer observations (ASCAT).

SCATT : Scatterometer winds: 10m wind vectors over sea, u_{10m}, v_{10m}. As wind direction may be ambiguous, 2 or 4 wind vectors are reported. Currently we use the vector flagged as the most probable choice by the data provider.

Surface Observation Operators

The different quantities are derived as follows:

t_{2m} : Near surface temperature is not assimilated because it strongly depends on the surface characteristics and is not representative for the temperature in the atmosphere.

u_{10m}, v_{10m} : Ten meter wind is horizontally interpolated from the values of the lowest model level (which is located at 10 m height). Near surface winds are used over sea only (and from land observations below 150m height in the tropics).

rh_{2m} : Relative humidity currently is horizontally interpolated from the respective values at the lowest model level.

p_s : Surface pressure is reported either at station height, or as mean sea level pressure, or reduced to standard geopotential height levels. Model surface pressure is interpolated or extrapolated to the respective height. Technically, not pressure (at given height) is assimilated, but geopotential height at the respective pressure level.

6.1.4 Satellite Radiances - Passive Sounding Instruments onboard of Satellites (RAD)

Satellite borne sounder instruments[7] are not able to measure meteorological quantities directly, rather they measure the upwelling radiation emitted by the Earth's atmosphere and surface in narrow spectral bands (so-called channels). The radiation measured by a

[6] Scatterometer:
http://www.esa.int/esaLP/SEMBWEG23IE_LPmetop_0.html

[7] Satellite Radiances:
http://www.eumetsat.int/Home/Main/Satellites/Metop/Instruments/index.htm

satellite instrument depends in a complex way on the vertical distribution of temperature and the atmospheric composition, on hydrometeors and on the Earth's surface properties. In order to assimilate such data in NWP models, the complicated relationship between the observed radiation and the meteorological variables (the state vector of the NWP model) is simplified and modeled numerically in an efficient way. At the moment these measurements are the only way to obtain near-realtime information on the atmosphere with global coverage and high horizontal resolution. In the variational scheme, satellite observed radiances (reported as brightness temperature t_b) are assimilated directly.

Passive Sounding Instruments

AMSU-A, HIRS The HIRS (High-resolution Infra-Red Sounder) and AMSU-A (Advanced Microwave Sounding Unit A) instruments mainly measure the emission in channels that are dominated by well mixed gases with a known mixing ratio (carbon dioxide and oxygen). Therefore, these measurements contain information on the vertical temperature distribution. Currently only the AMSU-A microwave instrument on the polar orbiting satellites is used operationally for cloud-free pixels located over sea. Assimilation of HIRS data is under preparation.

AMSU-B, MHS The AMSU-B (Advanced Microwave Sounding Unit B) and MHS (Microwave Humidity Sounder) channels are chosen in water vapour emission spectral lines, such that these measurements contain information on the vertical distribution of water vapour in the higher layers of the atmosphere. Assimilation of the moisture sensitive AMSU-B and MHS instruments is under preparation.

IASI Due to the small number of channels the vertical resolution of these classical sounders is quite small. The IASI (Infrared Atmospheric Sounding Interferometer) instrument has several thousands of channels, and therefore provides information on the atmospheric temperature and atmospheric composition with a much better vertical resolution and accuracy. The assimilation of IASI data is under preparation.

Observation Operator for Satellite Radiances

Missing: Assignment of nominal height for satellite observations (radiances!) as done in code.

The observation operator $H(x)$ is the radiative transfer model which calculates the outgoing radiation at the top of the atmosphere in the observed spectral interval from the atmospheric temperature and humidity profile as well as surface temperature. We use the RTTOV package, a fast radiative transfer model for passive infrared and microwave satellite radiometers, spectrometers and interferometers, which has been developed to fit the time constraints for repeated application of the observation operator in iterative data assimilation algorithms. The tangent linear and adjoint operator is provided as well. The RTTOV package has been developed within the EUMETSAT NWP SAF[8] and is licensed

[8]EUMETSAT NWP SAF: Activity that exists to co-ordinate research and development efforts among the SAF (Satellite Application Facility) partners to improve the interface between satellite data and NWP for the benefit of EUMETSAT member states.

by the Met Office. We use Version 7 of the RTTOV code, but will switch to Version 9 in the near future.

Before entering the 3dVar scheme the data are pre-processed as follows:

1. The field of views (FOVs) of the AMSU-A and AMSU-B instrument are aggregated (interpolated) to the location of the HIRS instrument FOV.
2. Only cloud free pixels located over sea are selected.
3. Bias correction is applied.
4. A 1D-Var retrieval is performed.

This preprocessing package was designed to be used with the OI scheme (Optimum Interpolation, the predecessor of the 3dVar). Some functionalities are obsolete now. The retrieval of the 1D-Var is not used but radiances are assimilated directly. The preprocessing package is currently under revision and will be tailored to the requirements of the 3dVar.

Within the 3dVar satellite radiance data is processed as follows:

1. Interpolation from the model grid to the fixed pressure levels used by the RTTOV operator is performed by the same algorithms as used for the in-situ data (i.e. TEMPs, cf. section 6.1.2).
2. The operational GME 30 L60 (60 level version) reaches up to 5 hPa. As the RTTOV operator requires input at even higher levels, forecasts from the IFS model are used as the background on top of the GME model.
3. Some of the input variables passed to the RTTOV operator (sea surface temperature, IFS profile above GME top) are not derived from the GME prognostic variables and therefore not fitted to the observations by adjusting the GME background field in the variational scheme. However these data are uncertain and should be allowed to vary as well. For this reason additional control variables (dummy sink variables) are introduced to model the uncertainties. An error of 1 K for SST and 4.5 K for temperature derived from the IFS data is assumed. No correlations are considered for these data.
4. Thinning is performed to account for horizontal correlations in the radiance observations which are not explicitly modelled (cf. section 8.1.2). Currently a thinning radius of 240 km is used.

6.1.5 GNSS Radio Occultations (GPSRO)

Radio Occultations (RO)[9] is a remote sensing measurement technique to probe the planetary atmosphere using radio signals which pass from some sender to some receiver, where

[9] Radio Occultations:

1. E. R. Kursinski et al., "Observing Earth's atmosphere with radio occultation measurements using the Global Positioning System", J. Geophys. Res., 102, D19, 23,429-23,465, 1997

usually both sender and receiver are outside of the atmosphere under consideration. In the case of GNSS radio occultations the sender is a GNSS satellite (e.g. GPS) in a high orbit, the receiver is a satellite in a Low-Earth-Orbit (LEO) at an altitude of typically 700-800 km.

One usually employs simplified models of ray bending for analysis. The basic physical idea is to determine approximate values of the refractive index at the bending points. In the Earth's atmosphere we obtain a sensitivity of the refractive index with respect to humidity and temperature.

Near Real-Time data of GPS Radio Occultations used at DWD are GRACE-A, GRACE-B, TerraSAR-X and TanDEM-X as processed by the GFZ at Potsdam, FORMOSAT-3/COSMIC processed by UCAR, and data from the GRAS instrument on EUMETSAT's polar orbiting satellites (Metop-A and Metop-B).

Observed Quantities

From the point of view of a Low-Earth-Orbit satellite the GNSS satellites will ceaselessly rise above, or set behind, the horizon of the Earth. During these so-called "radio occultations", where the GNSS and the LEO satellite are just able to "see" each other through the atmosphere, the GNSS signals will be slightly delayed and their ray path slightly bent on the way through the ionosphere (twice) and the atmosphere. On the LEO this is observed as differences in the phase and amplitude of the received signals. From this phase differences a vertical profile of the ray bending angle α in a single point on the earth's surface can be derived. The refractive index, which depends on temperature and humidity can be calculated from the bending angle.

Observation Operator for Radio Occultations

The observation operator transforms the model variables to the observations in the following way:

1. Calculate the refractive index in each point of the model grid from temperature and humidity
2. Perform a vertical interpolation of the refractive index
3. Use a numerical integration method on a fine grid to obtain the bending angle α (Abel transform)

The signal has to pass the ionospere, which has the largest impact on the ray bending. This impact is estimated in a seperate model and has the largest effect on the measurement error.

---

2. Eumetsat GRAS-SAF, "The Radio Occultation Method"
3. GRACE Radio Occultation Survey
4. UCAR Radio Occultation Survey

6.1.6 Tangent-linear and adjoint operators

Missing: Description of tangent-linear and adjoint operators of 3dVar!

Definition of the adjoint

Automatic or algorithmic differentiation

6.1.7 GNSS Slant Total Delay Operator

Ground based GNSS[10] makes use of GNSS data collected by large GNSS networks. GNSS processing algorithms provide the optical path length for each individual satellite-receiver link, i. e. information about the atmospheric state integrated along the signal path. The slant total delay (STD) is the atmospheric contribution to the optical path length. These quantities are called delays or execess path lengths as the atmosphere delays the signal as compared to undisturbed propagation in vacuum. Zenith total delays (ZTDs) are the correspondig delays in zenith direction and are equivalent to STDs with elevations of 90°. However, ZTDs are usually processed in a different way involving all available observations within a given period which are combined to a hypothetical observation in zenith direction.

ZTD data can be obtained from several network providers or GNSS processing centers in the world. The data are usually available in near real-time or as real-time stream. Near real-time data are processed in temporal batches, e. g. one hour, and provided with a certain delay, e. g. 75 minutes. ZTD data for Europe can be obtained from E-GVAP which is an association of several European GNSS processing centers and weather services. Operational STD data can currently[11] obtained only from the GFZ in Potsdam. The GFZ provides near real-time STD data with a delay of about 75 minutes and a focus on Germany.

The observed STD is an integral of the atmospheric refractive index along the signal path from the GNSS satellite to the GNSS receiver at the ground. The refractive index is related to the pressure, temperature and humidity fields and the length of the signal path inside the model ranges from the vertical extension of the model (ZTD, 90° elevation) up to several hundred km or even more than 1000 km, depending on the elevation and the height of the model top layer. STD observations are therefore neither localized nor related to one specific meteorological quantity.

An other issue is the signal path inside the atmosphre which is not known in advance but must be estimated from the background model using a raytracer. GNSS data sets provide the coordinates of the GNSS satellite and of the GNSS station, the signal path in between depends on the specific atmospheric state in the vicinity of the connecting line and can be estimated by solving Fermat's principile for a given model state. The resulting

[10] The generic term GNSS (Global Navigation Satellite Systems) refers to all existing navigation satellite systems, i. e. GPS, GLONASS, Compass, Galileo, ...

[11] Summer 2014

bended signal path has a considerable effect on the path delay and cannot be neglected for observations with elevations smaller than $\sim 30°$.

Observation Operator

The STD observation operator consits of two parts. In the first part a connecting line between the station and the satellite is defined and a set of supporting points along this line is chosen. Using the coordinates of these supporting points the interface to the assimilation system collects the required meteorological data and provides them to the second part of the operator. As the GNSS signal can propagate through large parts of the model access to grid nodes with quite large distances to the station is often required. The second part of the STD operator calls a raytracer in order to estimate the bended signal path and integrates along this path to obtain the STD.

The basic steps of the STD operator are:

1. Define a connecting line between the GNSS station and the GNSS satellite and a set of supporting points on this line which will later be used for numerical integration. The point density is scaled with the pressure profile in order to obtain point distances which decrease with increasing height.

2. Collect the temperature, pressure and humidity data on the grid nodes surrounding the supporting poins.

3. Compute the atmospheric refractivity on these nodes using the Smith & Weintraub formula.

4. Interpolate the refractivities at the supporting points.

5. Call the raytracer to estimate the signal path in the atmosphere, i.e. the deviations from the connecting line. The raytracer is implemented as a iterative Newton algorithm which minimizes the optical path length in the atmosphere (Fermat's principle). The iteration is started with the connecting line and makes small modifications in order to approximate the "true" bended signal path. Within this step the refractivities need to be interpolated on the latest approximation of the signal path.

6. The STD is the line integral through the refractivity field along the signal path. This is obtained by numerical integration.

7. Ionospheric effects are neglected.

The STD operater can also be used to assimilate ZTD data as the ZTD is a specific case of a STD with an elevation of 90°. However, due to different GNSS procesing strategies the ZTD errors should be smaller than the corresponding STD errors. The same STD operator code is used in GME/ICON and COSMO, only the interfaces are different.

Incomplete: GNSS Slant Total Delay operator

6.2 Observation Operators in COSMO

COSMO-DE is only used within the EnKF in Kilometre-Scale Ensemble Data Assimilation (KENDA), so most aspects of this section are closely linked to the LETKF-algorithm described in Chapter 9.

When running KENDA, the computation of the model equivalents $H(\mathbf{x}^\ell)$ to the observations $\mathbf{y}^o$ is performed within the single model instances $\{\mathbf{x}^\ell : \ell = 1, \ldots, L\}$ during the forecast step wherein the COSMO-members are forecasting in parallel (Figure 9.2). This is a relict from the deterministic approach of single COSMO-forecasts wherein *nudging* was used to incorporate informations from observations, and it motivates the use of this separate section that deals only with the aspects of the COSMO operators.

The following subsections first explain the spatial and temporal assignments, followed by the observation types used for COSMO. A description of the observation errors is provided in Section 9.3.

6.2.1 Spatial assignment

COSMO-DE (2013) is run at a resolution of $\Delta x = 2.8$ km – therefore it is not considered necessary to interpolate the model values horizontally to the exact location. Using the *nearest-neighbour* technique for assigmnent, a horizontal shift of 1 to 2 grid lengths Δx appears to be acceptable, as grid point models cannot resolve wavelengths $\leq 2\Delta x$.

It is considered more important that the vertical of assignment the observations onto the model grid is appropriate: especially near the surface the badly resolved orography of the model will give representativity errors for surface pressure observations, and the boundary layer in the model can be misrepresented with respect to vertical sounding observations.

Upper air reports

Upper-air reports such as single-level aircraft reports are assigned to the nearest model grid point.

Sea surface reports

Sea surface reports are those which are surrounded by sea grid points or which are specifically labeled, such as drifting buoys or shop. The observations are assigned to the nearest sea grid point.

Land surface reports

Land reports are assigned to the nearest land grid point only if the (geometrical) distance to it is less than or hafl of the latitudinal mesh width Δ in the horizontal and less than 40 m in the vertical. Otherwise, the land point with the minimum vertical distance Δz to the station is selected out of the points within a horizontal radius of $\sqrt{2}\Delta y$ from the observation location.

6.2.2 Temporal assignment

The length of the EnKF forecast window Δt_{fc} can be in the order of one hour down to a few minutes, depending of the frequent availability of observation reports. During Δt_{fc}, the single reports of $\mathbf{y}^o(t)$ are assigned to the closest timestep t within COSMO and their equivalents $H(\mathbf{x}(t))$ is computed.

In the actual assimilation of KENDA (Chapter 9), all pairs of

$$[y_j^o, H(\mathbf{x})_j] \ : j = 1, \ldots, m$$

are currently regarded to be valid at the specific point-in-time when the analysis computation takes place (cf. Figure 9.1 with the naive LETKF recursion and Figure 9.2 with the special LETKF of COSMO-KENDA).

6.2.3 Observation types

The observation types used by COSMO are similar to those used in the observation operators of 3dVar.

In-situ observations

Conventional in-situ observations (cf. Section 6.1.2) include TEMP, PILOT, AIREP, AMDAR, ACARS, SYNOP, SHIP and DRIBU. These, with the addition of wind profilers, directly observe model variables such as horizontal wind, humidity and pressure – therefore, these parts of $\mathbf{y}^o$ only need to be assigned (or interpolated) to a model grid point to obtain $H(\mathbf{x})$.

Remote sensing observations

Current remote sensing observations supported by COSMO contain reflectivity and Doppler wind speed from ground-based radars, satellite radiances (Section 6.1.4) with the addition of visible radiances observed by geostationary satellites, GNSS Radio Occultations (Section 6.1.5) and integrated water vapor (IWV) derived from ground based GPS total zenith delay.

For these types, H consists of algorithms that simulate the observing instruments and systems to compute a model equivalent $H(\mathbf{x})$. These computations usually use the complete 3D model field $\mathbf{x}$, instead of the point-wise assignment in the case of in-situ observations.

Various other observation types (such as lightning observations) could be used, as long as it is possible to construct a senseful observation operator and as long as the observed information is not already redundantly provided by other observation systems.

Chapter 7

Variational Data Assimilation 3dVar

As formulated in Section 2, the goal of data assimilation is to determine the most likely state of the atmosphere using observations and a model background, weighted by the observation error covariances and the background error covariances.

Variational data assimilation is a well-known approach to employ measurement data to control the simulation of atmospheric dynamical systems, compare for example Kalnay [18]. The key idea is to reformulate the assimilation task into a minimization problem either for basic time-steps separately (3dVar) or for the trajectories of the system state in time (4dVar) when the assimilation of measurements within a time window is combined into a large minimization problem. In this Chapter, the 3dVar scheme is documented:

- Section 7.1 sets up the minimization problem of 3dVar and how it is solved here
- Section 7.2 describes the background error covariance model $\mathbf{P}^b_{OI}$ taken over from the former OI scheme. This scheme is superseeded in the meantime by the NMC derived formulation. However the description of horizontal covariances is still used for preconditioning in the CG scheme.
- Section 7.3 describes the background error covariance $\mathbf{P}^b_{NMC}$ model using vertical covariances derived by the NMC method. This formulation is currently used to construct the $\mathbf{B}$-matrix of 3dVar.
- A further option to derive the background error covariance model $\mathbf{P}^b_{EnKF}$ uses localised covariances derived from an short range ensemble forecast. A combination of P^b_{NMC} and $\mathbf{P}^b_{EnKF}$ will be used in the upcoming Veriational ensemble Kalman Filter. This method is described seperatly in Section 10.

Note that the 3dVar-scheme is only applied to the hydrostatic global model GME and the nonhydrostatic ICON model in its global configuration which is close to hydrostatic balance.

For the local COSMO model, the LETKF-scheme is used (Chapter 9) because it is impractical (and maybe even impossible) to construct a proper analytic background error covariance model for nonhydrostatic phenomena.

7.1 3dVar: Formulation of the minimization problem and its solution

In this section, we will first summarize the main stochastic approach based on Bayes formula in Section 7.1.1. We show how maximum-likelihood estimators for the posteriori density distribution of states lead to a general minimization problem. Then, we use an iterative *Newton-Method* involving a linearization of the observation operators and a *quadratic approximation* to the non-quadratic error functionals to minimize the given functional. This method requires the solution of large linear systems as a part of the iterative procedure. We will employ a *congugate gradient method* to iteratively solve the linear system. The derivation of this scheme is worked out in Section 7.1.2

7.1.1 Stochastic Background of the Minimization Problem

The goal of data assimilation is to use measurements $\mathbf{y}^o$ to determine an improved estimate $\mathbf{x}^a$ (called the analysis) for the state of a dynamical system on the basis of some calculated background state $\mathbf{x}^b$ which is derived from previous measurements and simulations. The new analysis $\mathbf{x}^a$ is then used as initial condition for calculating further forecasts.

Often, the analysis is calculated as the maximum likelihood estimate on the a posteriori Bayesian distribution of the system states. This means, the analysis $\mathbf{x}^a$ is the most probable solution (or the most likely state of the dynamical system, e.g. atmosphere), given the background $\mathbf{x}^b$ as prior distribution and the observations $\mathbf{y}^o$:

$$\mathbf{x}^a = \arg\max_{\mathbf{x}} \left(p^{\text{post}} \left(\mathbf{x} | \mathbf{y}^o \right) \right) \tag{7.1}$$

It is convenient to define a cost function

$$J(\mathbf{x}) = -\log \left(p^{\text{post}} \left(\mathbf{x} | \mathbf{y}^o \right) \right) + const. \tag{7.2}$$

Since the logarithm is a monotonic function, this cost function is minimized by the same $\mathbf{x}^a$ that maximizes the likelihood $p^{\text{post}} \left(\mathbf{x} | \mathbf{y}^o \right)$

$$\mathbf{x}^a = \arg\min_{\mathbf{x}} \left(J(\mathbf{x}) \right) \tag{7.3}$$

Applying Bayes' theorem (compare [17]) gives

$$p^{\text{post}} \left(\mathbf{x} | \mathbf{y}^o \right) = \frac{p^{\text{obs}} \left(\mathbf{y}^o | \mathbf{x} \right) p^{\text{prior}} (\mathbf{x})}{p \left(\mathbf{y}^o \right)}. \tag{7.4}$$

The probability $p \left(\mathbf{y}^o \right)$ acts as a normalization constant. Since the minimum of (7.3) is not influenced by this constant, it is not needed to calculate the analysis $\mathbf{x}^a$. The probability $p^{\text{prior}}(\mathbf{x})$ is given by the a priori knowledge, e.g. the background state $\mathbf{x}^b$ and a covariance matrix or in general by more complex non-Gaussian distributions. This leads to

$$p^{\text{post}} \left(\mathbf{x} | \mathbf{y}^o \right) \propto p^{\text{obs}} \left(\mathbf{y}^o | \mathbf{x} \right) p^{\text{prior}} \left(\mathbf{x} \right). \tag{7.5}$$

and

$$J \left(\mathbf{x} \right) = -\log \left(p^{\text{obs}} \left(\mathbf{y}^o | \mathbf{x} \right) \right) - \log \left(p^{\text{prior}} \left(\mathbf{x} \right) \right) + const. \tag{7.6}$$

The maximum likelihood approach is carried out by minimization of the functional $J(x)$ defined in (7.6). It is applicable to any probability density functions $p^{\text{obs}}(\mathbf{y}^o|\mathbf{x})$ and $p^{\text{prior}}(\mathbf{x})$.

Often, the special case of Gaussian probability density functions is used as basic standard model. In this case we have

$$p^{\text{prior}}(\mathbf{x}) = \frac{1}{(2\pi)^{N/2}\left|\mathbf{P}^b\right|^{1/2}} \exp\left(-\frac{1}{2}\left(\mathbf{x}^b - \mathbf{x}\right)^T \mathbf{P}^{b^{-1}}\left(\mathbf{x}^b - \mathbf{x}\right)\right), \tag{7.7}$$

$$p^{\text{obs}}(\mathbf{y}^o|\mathbf{x}) = \frac{1}{(2\pi)^{M/2}\left|\mathbf{R}\right|^{1/2}} \exp\left(-\frac{1}{2}\left(\mathbf{y}^o - H(x)\right)^T \mathbf{R}^{-1}\left(\mathbf{y}^o - H(x)\right)\right), \tag{7.8}$$

where $\mathbf{P}^b$ and $\mathbf{R}$ are the covariance matrices for the prior distribution and the error distribution. In this special case the cost function, given by equation (7.6), with an appropriate choice of the constant *const* is

$$\begin{array}{rcccc} J(\mathbf{x}) & = & \frac{1}{2}[\mathbf{x}^b - \mathbf{x}]^T \mathbf{P}^{b^{-1}}[\mathbf{x}^b - \mathbf{x}] & + & \frac{1}{2}[\mathbf{y}^o - H(\mathbf{x})]^T \mathbf{R}^{-1}[\mathbf{y}^o - H(\mathbf{x})] \\ & = & J_b(\mathbf{x}) & + & J_o(\mathbf{y}), \end{array} \tag{7.9}$$

with

$$J_b(\mathbf{x}) := \frac{1}{2}[\mathbf{x}^b - \mathbf{x}]^T \mathbf{P}^{b^{-1}}[\mathbf{x}^b - \mathbf{x}] \tag{7.10}$$

$$J_o(\mathbf{y}) := \frac{1}{2}[\mathbf{y}]^T \mathbf{R}^{-1}[\mathbf{y}] \tag{7.11}$$

$$\mathbf{y} := \mathbf{y}^o - H(\mathbf{x}) \tag{7.12}$$

We call $J_b(\mathbf{x})$ the *background cost function* and $J_o(\mathbf{y})$ the *observation cost function*. The *observation operator* H calculates the model predictions for the observed quantities. H may be simply an interpolation to the location of the observation (in case of in situ measurements) or a complex operator (in case of remote sensing observations). In general, the operator H will be a nonlinear operator. For linear operators the cost function (7.9) is a quadratic cost function. We will discuss its minimization in the next Section 7.1.2.

Figure 7.1: Flowchart of the iterative Newton scheme.

7.1.2 Minimization by a Newton Scheme with Local Quadratic Approximation

The minimum of (7.6) or (7.9) is sought by an iterative Newton method with quadratic approximations of the cost functionals and a conjugate-gradient (CG) algorithm for the linear solution step.

We first consider the case (7.9) of Gaussian distributions (with quadratic cost functions except for the nonlinearity of the observation operator). We employ *first order optimality conditions*, i.e. we study the gradient $\partial J/\partial \mathbf{x}$ of the cost function J with respect to the control variables $\mathbf{x}$

$$\frac{\partial J}{\partial \mathbf{x}} = \mathbf{H}_x{}^T \mathbf{R}^{-1}[H(\mathbf{x}) - \mathbf{y}^o] + \mathbf{P}^{b\,-1}[\mathbf{x} - \mathbf{x}^b] \tag{7.13}$$

$\mathbf{H}_x$ is the Jacobi matrix $\partial H/\partial \mathbf{x}$ to H, linearised at the location $\mathbf{x}$. The product of $\mathbf{H}$ with a vector to the right is usually called the tangent linear operator[1].

At the minimum of J its gradient is zero. For linear observation operators ($\mathbf{H}\,\mathbf{x} = H(\mathbf{x})$) setting (7.13) to zero immediately leads to a set of linear equations

$$\left(\mathbf{H}^T \mathbf{R}^{-1} \mathbf{H} + \mathbf{P}^b\right) \mathbf{x} = \left(\mathbf{H}^T \mathbf{R}^{-1} \mathbf{y}^o + \mathbf{P}^b \mathbf{x}^b\right) \tag{7.14}$$

which can be solved for the analysis state $\mathbf{x}^a$[2]. In general, H is non-linear and we will use a non-quadratic observational cost function J_o in the framework of Variational Quality Control (cf. Section 8.2). In this case, the first order optimality condition leads to

$$0 = \frac{\partial J}{\partial \mathbf{x}}(\mathbf{x}) = \mathbf{H}_x^T \frac{\partial J_o}{\partial \mathbf{y}}(H(\mathbf{x}) - \mathbf{y}^o) + \ \mathbf{P}^{b\,-1}[\mathbf{x} - \mathbf{x}^b] \tag{7.15}$$

We solve this nonlinear set of equations by the iterative Newton's scheme. Newton's method implies a linearization of equation (7.15) at the current estimate $\mathbf{x}_i$. This is done by linearizing the observation operator at $\mathbf{x}_i$ and approximating the observational cost function quadratically at $\mathbf{y}_i = H(\mathbf{x}_i) - \mathbf{y}^o$. The linearized equation then is solved to obtain an updated estimate $\mathbf{x}_{i+1}$.

Linearising H at $\mathbf{x}_i$ and denoting the resulting Jacobian by $\mathbf{H}_i$ yields:

$$\frac{\partial J}{\partial \mathbf{x}} \approx \mathbf{H}_x^T \frac{\partial J_o}{\partial \mathbf{y}}(\mathbf{H}_i(\mathbf{x} - \mathbf{x}_i) + H(\mathbf{x}_i) - \mathbf{y}^o) + \ \mathbf{P}^{b\,-1}[\mathbf{x} - \mathbf{x}^b] \tag{7.16}$$

Defining $\mathbf{y}_i = H(\mathbf{x}_i) - \mathbf{y}^o$ and using the Taylor expansion

$$J_o(\mathbf{y}) \ = \ J_o(\mathbf{y}_i) \ + \ [\mathbf{y} - \mathbf{y}_i]^T \left.\frac{\partial J_o}{\partial \mathbf{y}}\right|_{y_i} + \frac{1}{2}\,[\mathbf{y} - \mathbf{y}_i]^T \left.\frac{\partial^2 J_o}{\partial \mathbf{y}^2}\right|_{y_i} [\mathbf{y} - \mathbf{y}_i] \tag{7.17}$$

[1] These operators are either programmed explicitly or can be derived from the program code of H by automatic or algorithmic differentiation.

[2] In the inverse problems literature this is known as *Tikhonov regularization*, compare [9]

we obtain a quadratic approximation of the observation cost function . We denote the gradient by $\mathbf{j}_{oi}$ and the Hessian matrix by $\mathbf{R}_i^{-1}$:

$$\mathbf{j}_{oi} := \left.\frac{\partial J_o}{\partial \mathbf{y}}\right|_{y_i}, \qquad \mathbf{R}_i^{-1} := \left.\frac{\partial^2 J_o}{\partial \mathbf{y}^2}\right|_{y_i}. \tag{7.18}$$

With approximation (7.17) and notation (7.18), the gradient of J_o can be written as:

$$\frac{\partial J_o}{\partial \mathbf{y}} \approx \mathbf{j}_{oi} + \mathbf{R}_i^{-1}[\mathbf{y} - \mathbf{y}_i] \; . \tag{7.19}$$

Substitution in Equation (7.16) yields:

$$\frac{\partial J}{\partial \mathbf{x}} \approx \mathbf{H}_x^T \, \mathbf{R}_i^{-1} \, [\, \mathbf{H}_i(\mathbf{x} - \mathbf{x}_i) + \mathbf{R}_i \, \mathbf{j}_{oi} \,] \; + \; \mathbf{P}^{b^{-1}} \, [\mathbf{x} - \mathbf{x}^b] \tag{7.20}$$

Finally we replace $\mathbf{H}_x$ by $\mathbf{H}_i$ and split $(\mathbf{x} - \mathbf{x}_i)$ into $(\mathbf{x} - \mathbf{x}^b) + (\mathbf{x}^b - \mathbf{x}_i)$:

$$\frac{\partial J}{\partial \mathbf{x}} \approx \mathbf{H}_i^T \, \mathbf{R}_i^{-1} \, \mathbf{H}_i[\mathbf{x} - \mathbf{x}^b] \; + \mathbf{H}_i^T \, \mathbf{R}_i^{-1} \, [\mathbf{H}_i(\mathbf{x}^b - \mathbf{x}_i) + \mathbf{R}_i \, \mathbf{j}_{oi} \,] \; + \; \mathbf{P}^{b^{-1}} \, [\mathbf{x} - \mathbf{x}^b] \tag{7.21}$$

Setting (7.21) to zero and solving for $\mathbf{x}$ yields the preliminary improved estimate $\widetilde{\mathbf{x}_{i+1}}$:

$$\widetilde{\mathbf{x}_{i+1}} - \mathbf{x}^b = [\mathbf{H}_i^T \, \mathbf{R}_i^{-1} \, \mathbf{H}_i + \mathbf{P}^{b^{-1}}]^{-1} \, \mathbf{H}_i^T \, \mathbf{R}_i^{-1} \, [\, \mathbf{H}_i(\mathbf{x}_i - \mathbf{x}^b) - \mathbf{R}_i \, \mathbf{j}_{oi} \,] \tag{7.22}$$

The matrix $\mathbf{P}^b$ is of size n_m^2 where $n_m \approx 10^7$ is the number of model variables. It is thus much too large to be represented explicitely. For that reason we do not perform the minimization in this form. Algebraic manipulation of equation (7.22) leads to a transformation from model to observation space. We obtain an equation that is easier to solve:

$$\widetilde{\mathbf{x}_{i+1}} - \mathbf{x}^b = \mathbf{P}^b \mathbf{H}_i^T \, [\mathbf{H}_i \mathbf{P}^b \mathbf{H}_i^T \; + \; \mathbf{R}_i]^{-1} \, [\, \mathbf{H}_i(\mathbf{x}_i - \mathbf{x}^b) - \mathbf{R}_i \, \mathbf{j}_{oi} \,]. \tag{7.23}$$

We split this equation in two parts, using $\mathbf{z}_{i+1} = \left(\mathbf{P}^b \mathbf{H}_i^T\right)^{-1} \left(\widetilde{\mathbf{x}_{i+1}} - \mathbf{x}^b\right)$:

$$\left(\mathbf{H}_i \mathbf{P}^b \mathbf{H}_i^T + \mathbf{R}_i\right) \mathbf{z}_{i+1} \;\; = \;\; \mathbf{H}_i \left(\mathbf{x}_i - \mathbf{x}^b\right) - \mathbf{R}_i \mathbf{j}_{oi} \tag{7.24}$$

$$\widetilde{\mathbf{x}_{i+1}} \;\; = \;\; \mathbf{x}^b + \mathbf{P}^b \mathbf{H}_i^T \mathbf{z}_{i+1}, \tag{7.25}$$

and obtain a preliminary improved estimate $\widetilde{\mathbf{x}_{i+1}}$. We solve equation (7.24) with a preconditioned conjugate gradient method.

To ensure convergence and to further improve this preliminary estimate a line search in the direction $(\widetilde{\mathbf{x}_{i+1}} - \mathbf{x}_i)$ is performed by minimizing the equation

$$J_{ls}\,(\alpha) := J\,(\alpha(\widetilde{\mathbf{x}_{i+1}} - \mathbf{x}_i) + \mathbf{x}_i) \tag{7.26}$$

for α. The final updated estimate is

$$\mathbf{x}_{i+1} = \alpha\,(\widetilde{\mathbf{x}_{i+1}} - \mathbf{x}_i) + \mathbf{x}_i. \tag{7.27}$$

The complete sequence (cf. figure 7.1) of the Newton scheme with embedded conjugent gradient minimisation and line search is as follows:

(S1) The background $\mathbf{x}^b$ is chosen as the first guess $\mathbf{x}_0$ of the iteration process.

(S2) For the current estimate $\mathbf{x}_i$ the terms required for the linearisation (7.18) are calculated: $\mathbf{H}_i$, $\mathbf{R}_i$, $\mathbf{j}_{oi}$.

(S3) For preconditioning of the conjugate gradient scheme we have to calculate an approximation to the inverse of $\left(\mathbf{H}_i\mathbf{P}^b\mathbf{H}_i^T + \mathbf{R}_i\right)$.

Preconditioning is a very important step to get a fast algorithm: With the conjugate gradient method a n-dimensional linear system can be solved exactly in n steps. However, in case of a well conditioned problem a good approximation to the exact solution can be obtained after a few steps already. As the matrix $\left(\mathbf{H}_i\mathbf{P}^b\mathbf{H}_i^T\right)$ is usually ill-conditioned, we will use a preconditioning method for solving (7.24).

We approximate $\widetilde{\mathbf{H}_i\mathbf{P}^b\mathbf{H}_i^T} :\approx \mathbf{H}_i\mathbf{P}^b\mathbf{H}_i^T$, with $\widetilde{\mathbf{H}_i\mathbf{P}^b\mathbf{H}_i^T}$ consisting of the block diagonal of $\mathbf{H}_i\mathbf{P}^b\mathbf{H}_i^T$ only, with an approximate size of 1000 x 1000 each. The preconditioning matrix $(\widetilde{\mathbf{H}_i\mathbf{P}^b\mathbf{H}_i^T} + \mathbf{R})^{-1}$ is calculated using a Cholesky decomposition.

(S4) Equation (7.24) is solved by the preconditioned conjugate gradient scheme.

(S5) The preliminary improved estimate $\widetilde{\mathbf{x}_{i+1}}$ is calculated from equation (7.25).

(S6) The updated estimate $\mathbf{x}_{i+1}$ is calculated by minimizing equation (7.26) for α (line search).

(S7) Finally the convergence of the Newton scheme is checked and, if required, the loop is continued with step **(S2)**.

7.2 OI Background error covariance model

This section describes the separable formulation of background error covariances by means of analytical functions, taken over from the former OI scheme. This scheme is superseeded in the meantime by the NMC derived formulation.

In the preceding Sections we explained all parts of the 3dVar cost function except the background error covariance matrix which describes the correlations between quantities used within the analysis. The independent state variables on which the analysis is based and for which covariances need to be known are given by

1. the *geopotential height* h
2. *vectorial wind*, expressed by
   the velocity potential χ
   the stream function ψ
3. and the *relative humidity* rh.

This means we consider four scalar functions which are defined on the global or local grids under consideration. Grid points are refered to by the indices i or j in the index set $\mathcal{J} = \{1, 2, 3, ..., N\}$ when N grid points are given. Here, we use the notation employed by ECMWF, where standard deviation for a quantity y is denoted by E_y. Correlation between y and y' are written as $< y, y' >$.

The following approach is based on both extensive statistical evaluations which have been carried out at DWD and other centers and on the modelling of particular relations between the variables. Let $\mathbf{B}_{y,y'}$ denote the matrix of covariances between y_i and y'_j where i, j runs over the grid points under consideration, i.e.

$$\mathbf{B}_{y,y'} := \left(< y_i - \overline{y}_i, y'_j - \overline{y'}_j > \right)_{i,j \in \mathcal{J}}, \tag{7.28}$$

where $\overline{y_i}$ denotes the expectation value of y_i. Clearly, the matrices $\mathbf{B}_{y,y'}$ need to be symmetric and $\mathbf{B}_{y,y'} = \mathbf{B}_{y',y}$. We assume that h, χ and rh are uncorrelated, i.e. $\mathbf{B}_{h,\chi} = 0$, $\mathbf{B}_{h,rh} = 0$ and $\mathbf{B}_{\chi,rh} = 0$. Further, we will approximate the correlations between ψ and χ and ψ and rh by zero, i.e. $\mathbf{B}_{\psi,\chi} = 0$ and $B_{\psi,rh} = 0$. Then, for the complete covariance matrix $\mathbf{B}$ we obtain the form

$$\mathbf{B} = \begin{pmatrix} \mathbf{B}_{h,h} & \mathbf{B}_{h,\psi} & 0 & 0 \\ \mathbf{B}_{\psi,h} & \mathbf{B}_{\psi,\psi} & 0 & 0 \\ 0 & 0 & \mathbf{B}_{\chi,\chi} & 0 \\ 0 & 0 & 0 & \mathbf{B}_{rh,rh} \end{pmatrix} \tag{7.29}$$

In the following parts, we describe the setup of the matrices $\mathbf{B}_{h,h}$, $\mathbf{B}_{h,\psi}$, $\mathbf{B}_{\psi,\psi}$, $\mathbf{B}_{\chi,\chi}$ and $\mathbf{B}_{rh,rh}$. There are several options how to carry out the setup of these matrices, depending on the error statistics which are used and the particular approximation which is chosen. A part of the documentation is based on the approach which is in detail documented by ECMWF [8]. However, DWD today is using an new version of this based on error statistics which have been evaluated with the GME model.

7.2.1 A Separable Formulation for $\mathbf{B}_{h,h}$

A separable Ansatz is chosen for the height error correlations

$$< h_i, h_j >= E_{h_i,\lambda}(z_i) E_{h_j,\lambda}(z_j) c_{h_i,h_j}(z_i, z_j, a_h) F_h(r_{i,j}, L_h), \tag{7.30}$$

depending on the distance on the sphere $r_{i,j}$ and the vertical coordinate $z = ln(p)$ (with p given in hPa). Here, height covariances are fully determined by specifying the *horizontal correlation function* $F_h(r_{i,j}, L_h)$, and the *vertical correlation function* $c_{h_i,h_j}(z_i, z_j, a_h)$. The horizontal and vertical length scales L_h and a_h are prescribed as functions of geographical latitude λ. Standard deviations are specified as polynomials in $z = \ln p$:

$$E_{h_i}(z) = \sum_{n=0}^{6} e_{h,n}\, z^n, \tag{7.31}$$

where the coefficients $e_{h,n}$ are prescribed in dependence of latitude.

7.2.2 The Matrices $\mathbf{B}_{\psi,\psi}$ and $\mathbf{B}_{\chi,\chi}$ (Wind)

As a consequence of the well-known Helmholtz decomposition applied to a vector field on the sphere, the wind field v consists of a divergent and non-divergent part, described by velocity potential χ and stream-function ψ:

$$v = \vec{\nabla}\, \chi + \mathbf{k} \times \vec{\nabla}\, \psi \tag{7.32}$$

The covariances of the wind field v are derived from the covariances of stream-function ψ and velocity potential χ which are modeled in the same way as the height covariances (7.30):

$$< \psi_i, \psi_j >= E_{\psi_i} E_{\psi_j} c_{\psi_i,\psi_j} F_\psi(r_{i,j}, L_\psi) \tag{7.33}$$

$$< \chi_i, \chi_j >= E_{\chi_i} E_{\chi_j} c_{\chi_i,\chi_j} F_\chi(r_{i,j}, L_\chi) \tag{7.34}$$

As mentiones above, we assume that cross covariances between stream-function and velocity potential $\mathbf{B}_{\psi,\chi}$ or between velocity potential and height $\mathbf{B}_{h,\chi}$ are zero.

For the total wind prediction error E_v we calculate from its divergent and non-divergent part

$$E_v^2 = E_{v_\psi}^2 + E_{v_\chi}^2 + \underbrace{B_{\psi,\chi}}_{=0} = E_{v_\psi}^2 + E_{v_\chi}^2. \tag{7.35}$$

The contribution from each component is prescribed by specifying the parameter ν:

$$E_{v_\chi}^2 = \nu E_v^2 \quad \text{or} \quad E_{v_\psi}^2 = (1-\nu) E_v^2 \tag{7.36}$$

Correlations for the wind components u^l longitudinal and u^t transverse to the line between the observations under consideration are derived from the correlations of velocity potential and stream-function:

$$< u_i^l, u_j^l >= -c_{\psi_{i,j}} \frac{1}{r_{i,j}} \frac{\partial}{\partial r_{i,j}} F_\psi(r_{i,j}) - c_{\chi_{i,j}} \frac{\partial^2}{\partial r_{i,j}^2} F_\chi(r_{i,j}) \tag{7.37}$$

$$< u_i^t, u_j^t >= -c_{\psi_{i,j}} \frac{\partial^2}{\partial r_{i,j}^2} F_\psi(r_{i,j}) - c_{\chi_{i,j}} \frac{1}{r_{i,j}} \frac{\partial}{\partial r_{i,j}} F_\chi(r_{i,j}) \tag{7.38}$$

$$< u_i^l, u_j^t >=< u_i^t, u_j^l >= 0 \tag{7.39}$$

7.2.3 Horizontal Height - Wind Correlations: $\mathbf{B}_{\psi,h}$

If the geostrophic relationship were fulfilled exactly, the following relationships would hold:

$$\psi_i = \frac{g}{f} h_i, \tag{7.40}$$

with the standard gravity $g = 9.80665 \frac{m}{s^2}$ and XXX f, consequently:

$$c_{\psi_{i,j}} = c_{h_{i,j}}, \;\; F_\psi(r_{i,j}) = F_h(r_{i,j}), \;\; E_{\psi_i} = \frac{g}{f} E_{h_i}. \tag{7.41}$$

The geostrophic relationship only holds approximately and is only valid in the extratropics. Thus we demand:

$$< \Phi_i, \psi_j >= \mu E_{\Phi_i} E_{\psi_j} c_{\Phi_i,\psi_j} F(r_{i,j}) \tag{7.42}$$

The correlation between the transverse wind component and height then is:

$$< u_i^t, z_j >= -\mu\sqrt{1-\nu} c_{\Phi_i,\psi_j} \frac{\partial F(r_{i,j})}{\partial r_{i,j}} \tag{7.43}$$

7.2.4 Horizontal Correlations: Bessel function expansion

The horizontal correlation functions F_h are described by a Bessel function expansion:

$$F_h(r_{i,j}) = \sum_{n=0}^{8} A_n \, J_0(k_n \, r_{i,j}/D) \tag{7.44}$$

The coefficients A_n in the Physical Space Assimilation System (PSAS)-scheme take similar values as in the OI-scheme (denoted by $\tilde{A}_n$). The wave numbers k_n are defined by zero derivative boundary conditions $\partial J_0(k_n \, x)/\partial x = 0|_{x=r_{i,j}/D=1}$. The value of $\tilde{A}_0$ is adjusted, so that $F_h(r_{i,j}) = 0$ for $r_{ij} = D$. Finally the coefficients are rescaled so that $F_h(0) = 1$. Table 7.1 shows the coefficients (A_n) used here in comparison to the $\tilde{A}_n$ used in the OI scheme.

In the OI a horizontally constant mode (with different large scale vertical correlation matrix M_{LS}) was used. This is not the case in the PSAS scheme because for reasons of efficiency the correlations shall be zero for large separations $r_{i,j} > D$

The horizontal length scale L_h is defined as

$$L_h^2 = -\left(\frac{F_h}{\nabla^2 F_h}\right)_{r=0}. \tag{7.45}$$

It is related to D by

$$L_h^2 = D^2 \frac{\sum A_n}{\sum k_n^2 A_n} \tag{7.46}$$

n	$\tilde{A}_n$	A_n	k_n
0	0.15	0.062	
1	0.30	0.309	3.832
2	0.28	0.289	7.016
3	0.14	0.144	10.173
4	0.08	0.082	13.324
5	0.05	0.052	16.471
6	0.03	0.031	19.616
7	0.02	0.021	22.760
8	0.01	0.010	25.904

Table 7.1: Bessel function coefficients

For the actual coefficients we get $L_h = 0.10439D$ or $D = 9.5793L_h$.

The horizontal correlation function and its derivatives (required for the height-wind and wind-wind correlations) is shown in Figure 7.2.

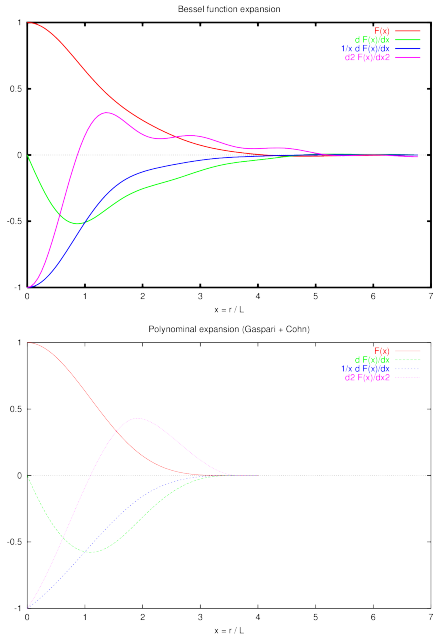


Figure 7.2: Horizontal correlation function and its derivatives.

...

7.2.5 Horizontal Correlations: Polynomial expansion

As an alternative to the Bessel function expansion, the compact polynomial expansion of Gaspari and Cohn is implemented [11].

$$F(r_{i,j}) = \begin{cases} -\frac{1}{4}r_{i,j}^5 + \frac{1}{2}r_{i,j}^4 + \frac{5}{8}r_{i,j}^3 - \frac{5}{3}r_{i,j}^2 + 1 & \text{if } 0 \le r_{i,j} \le 1, \\ +\frac{1}{12}r_{i,j}^5 - \frac{1}{2}r_{i,j}^4 + \frac{5}{8}r_{i,j}^3 + \frac{5}{3}r_{i,j}^2 - 5r_{i,j} + 4 - \frac{2}{3}\frac{1}{r_{i,j}} & \text{if } 1 < r_{i,j} \le 2, \\ 0 & \text{if } r_{i,j} > 2. \end{cases} \tag{7.47}$$

7.2.6 Height - Height Covariances : Vertical Correlations

The vertical correlation matrices are specified as functional relationships $c_{h_i,h_j}(x_i - x_j)$ in dependence on a transformed coordinates x:

$$c_{h_i,h_j}(x_i - x_j) = \frac{1}{e-1} \exp \frac{a_h}{a_h + (x_i - x_j)^2} - \frac{1}{e-1} \tag{7.48}$$

depending on the transformed coordinate x defined as a polynomial in $z = ln(p)$:

$$x(z) = \sum_{n=0}^{6} x_{h,n}\, z^n \tag{7.49}$$

The coefficients $x_{h,n}$ and vertical length scale a_h are prescribed as a function of latitude.

7.2.7 Thickness - Height Covariances

To derive the covariances for thickness, the equation $\bigtriangledown h_{1,2} = h_1 - h_2$ must be obeyed. Thus the covariance between thickness and any other variable a can be calculated from the height covariances:

$$< \bigtriangledown h_{1,2}, a > = < h_1, a > - < h_2, a > . \tag{7.50}$$

h_1 and h_2 are background values of height at two levels z_1 and z_2 separated by a small difference dz.

The height - height covariances are:

$$< h_i, h_j > = E_{h_i}(z_i) E_{h_j}(z_j) c_{h_i h_j}(z_i, z_j) F_h(r_{i,j}) \tag{7.51}$$

Thus the thickness - height covariances are:

$$< \frac{\partial h_i}{\partial z_i}, h_j > = \frac{\partial}{\partial z_i} < h_i, h_j > =$$

$$\left[E'_{h_i}(z_i) E_{h_j}(z_j) c_{h_i h_j}(z_i, z_j) + E_{h_i}(z_i) E_{h_j}(z_j) c'_{h_i h_j}(x_i(z_i), x_j(z_j)) x'_i(z_i) \right] F_h(r_{i,j}) \tag{7.52}$$

with the derivatives

$$c'(x_i, x_j) = -c'(x_j, x_i) = \frac{\partial c_{h_i h_j}}{\partial x_i} = -2\left(\frac{a_h(x_i - x_j)}{(a_h + (x_i - x_j)^2)^2}\right)\left[\frac{1}{e-1}\exp\frac{a_h}{a_h + (x_i - x_j)^2}\right] \tag{7.53}$$

$$x'(z) = \frac{\partial x}{\partial z} = \sum_{n=1}^{6} n\, x_{h,n}\, z^{n-1} \tag{7.54}$$

$$E'_{h_i}(z) = \frac{\partial E_{h_i}}{\partial z} = \sum_{n=1}^{6} n\, e_{h,n}\, z^{n-1} \tag{7.55}$$

7.2.8 Thickness - Thickness Covariances

Taking the derivative of (7.53) with respect to x_j we get:

$$\begin{aligned} c'' &= \frac{\partial^2 c}{\partial x_i\, \partial x_j} = \frac{\partial c'}{\partial x_j} \\ &= \left[2\frac{a_h}{(a_h + (x_i - x_j)^2)^2} - 4\frac{a_h^2(x_i - x_j)^2}{(a_h + (x_i - x_j)^2)^4} - 8\frac{a_h(x_i - x_j)^2}{(a_h + (x_i - x_j)^2)^3}\right] \\ &\quad \left[\frac{1}{e-1}\exp\frac{a_h}{a_h + (x_i - x_j)^2}\right] \end{aligned} \tag{7.56}$$

Thus the thickness-thickness covariance is:

$$\begin{aligned} < \frac{\partial h_i}{\partial z_i}, \frac{\partial h_j}{\partial z_j} > &= \big[E'_{h_i}(z_i)E'_{hj}(z_j)c_{h_i h_j}(x_i, x_j) \\ &\quad + E'_{h_i}(z_i)E_{h_j}(z_j)c'(x_j, x_i)x'(z_j) + E_{h_i}(z_i)E'_{h_j}(z_j)c'(x_i, x_j)x'(z_i) \\ &\quad + E_{h_i}(z_i)E_{h_j}(z_j)c''(x_i, x_j)x'(z_i)x'(z_j)\big] \end{aligned} \tag{7.57}$$

for $i = j$ we get $c = 1$, $c' = 0$ and $c'' = 2e/(a_h(e-1))$. Consequently the thickness variance is:

$$< \frac{\partial h_i}{\partial z_i}, \frac{\partial h_i}{\partial z_i} >= \left[E'_{h_i}(z_i)E'_{hi}(z_i) + \frac{2e}{a_h(e-1)}E_{h_i}(z_i)E_{h_i}(z_i)x'(z_i)x'(z_i)\right] \tag{7.58}$$

and the error (standard deviation) is:

$$E_{\frac{\partial h_i}{\partial z_i}} = \left[E'^2_{h_i}(z_i) + \frac{2e}{a_h(e-1)}E^2_{h_i}(z_i)x'^2(z_i)\right]^{1/2} \tag{7.59}$$

Temperature correlations are equivalent to thickness correlations and temperature errors are related to thickness errors by

$$E_T = \frac{g}{R}E_{\frac{\partial h_i}{\partial z_i}} \tag{7.60}$$

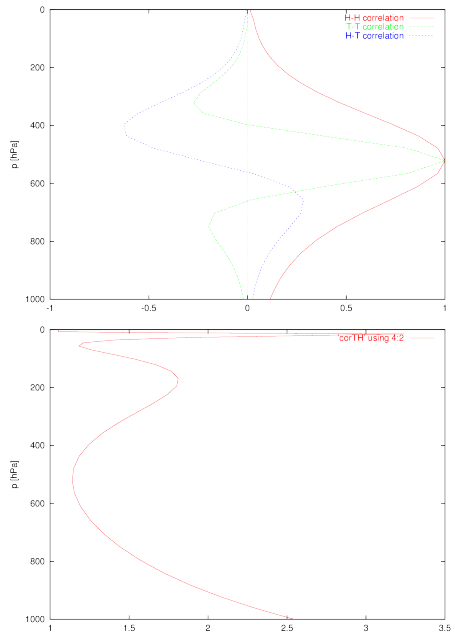


Figure 7.3: Vertical background error correlations of height-height, temperature-temperature and height-temperature (top), and errors for temperature (bottom) . These temperature-temperature correlations and variances are used in the 1DVAR routine for the RTTOVS operator.

with gravity acceleration g and gas constant R. The correlations of height-height, temperature-temperature and height-temperature, as well as the errors for temperature are shown in Figure 7.3.

In the PSAS radio-soundings are currently treated as observations of geopotential height, making use of the height-height correlations specified, although the derived thickness-thickness (temperature-temperature) covariances allow a direct use of the temperature observations. The thickness-thickness covariances are required for the RTTOVS operator for satellite observed radiances and for the post multiplication step.

7.2.9 Implementation of Covariance Calculations

For each location, depending on the observed quantities, the following terms are calculated:

observation		term, equation							
		E_h (7.31)	E'_h (7.55)	x_h (7.49)	x'_h (7.54)	E_v	x_v	E_{rh}	x_{rh}
height	h	X		X					
thickness	$\bigtriangledown h$	X	X	X	X				
wind	u, v	X		X		X	X		
humidity	rh							X	X

7.2.10 Horizontal variation of covariance parameters

The following parameters of the covariance model are prescribed as a function of latitude (Table 7.2, Figure 7.4.

variable	source file	values	parameter
		climatological values	
L_h	clerr	1000 km	length scale for height and wind.
L_q	clerr	300 km	length scale for humidity.
μ	clerr	-0.950 ... 0.950	geostrophic factor.
a_h	clerr	1.089 ... 1.233	height vertical transformation coefficient.
a_v	clerr	0.969 ... 1.331	wind vertical correlation coefficient.
a_q	clerr	0.974 ... 0.975	relative humidity vertical correlation coefficient.
		forecast values	
L_h	fgerr	400 ... 600 km	length scale for height and wind.
L_q	fgerr	300 km	length scale for humidity.
μ	fgerr	-0.900 ... 0.900	geostrophic factor.
a_h	fgerr	0.953 ... 1.062	height vertical transformation coefficient.
a_v	fgerr	0.887 ... 1.063	wind vertical correlation coefficient.
a_q	fgerr	0.887 ... 0.894	relative humidity vertical correlation coefficient.
		constant parameters	
ν	hardcoded	0.1	

Table 7.2: Latitude dependent covariance model parameters.

7.2.11 Vertical variation of covariance parameters

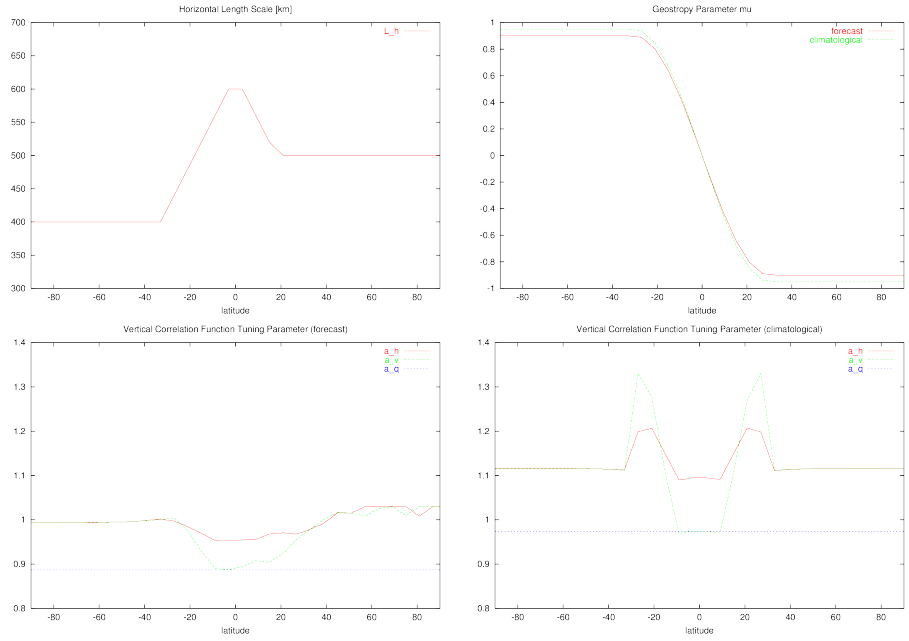
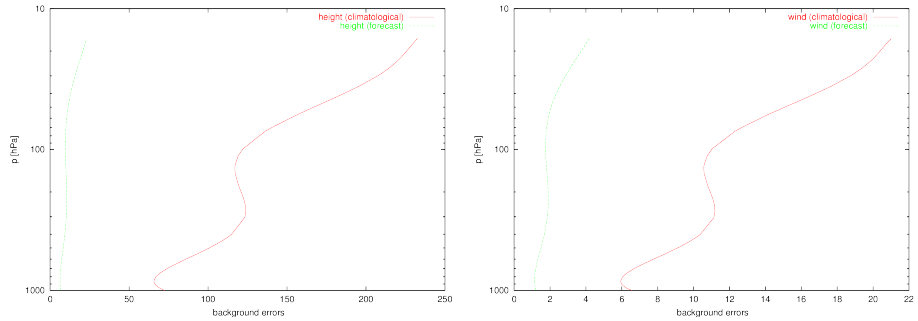
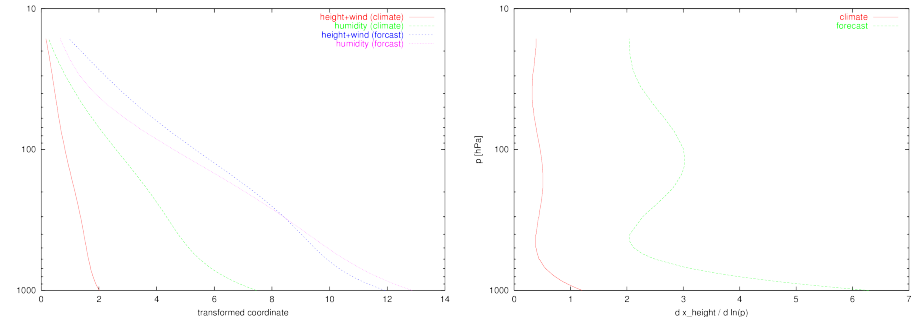


Figure 7.4: Latitude dependent covariance model parameters.

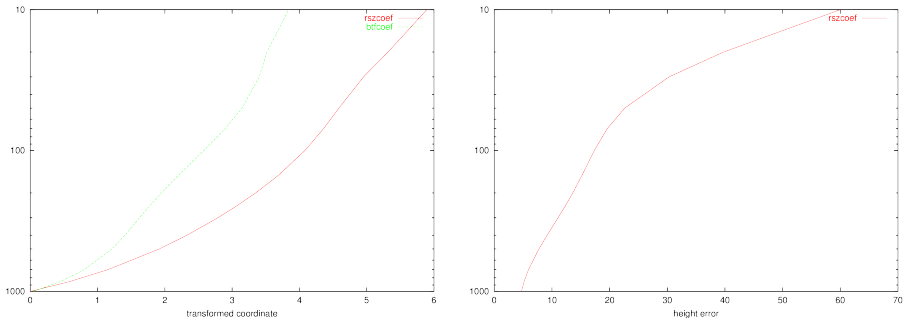


Height background error —- Wind background error



transformed coordinate x —- $\frac{\partial x}{\partial ln(p)}$

Figure 7.5: Height dependent covariance model parameters.


10
100
1000
0
1
2
3
4
5
6
transformed coordinate
rszcoef
bfcoef
10
100
1000
0
10
20
30
40
50
60
70
height error
rszcoef

7.3 NMC Background Error Covariance Model

This section describes the formulation of the background error covariances model capable to represent NMC derived covariances. The current implementation still follows the separability assumption and is able to handle both analytical and NMC derived background correlations as well as a mixture of vertical NMC and horizontal analytic covariances. Technically the advantage of this approach is that it scales linearly with the number of observations or grid-points, not quadratically.

7.3.1 Wavelet Transformed Background Error Covariance Matrices

The wavelet transformed background error covariance matrix in wavelet transformed space $\widetilde{\mathbf{P}^b}$ is a sparse matrix as long as small and insignificant matrix elements are neglected. The goal of this approach is to store matrices derived by the NMC method in a suitable sparse representation.

We have:

$$\mathbf{P}^b = \mathbf{W}\,\widetilde{\mathbf{P}^b}\,\mathbf{W}^T \tag{7.61}$$

with $\mathbf{W}$ being the 3-dimensional wavelet transformation (synthesis). We represent the wavelet transformed covariance matrix by a factorization $\widetilde{\mathbf{P}^b} = \mathbf{L}\,\mathbf{L}^T$ so that positive-definiteness of the matrix is ensured.

In our approach $\widetilde{\mathbf{P}^b}$ is represented in coordinates λ, ϕ and $\log p$. This regular horizontal grid facilitates deriving zonally averaged climatological covariances. In order to calculate covariances between arbitrary locations of observations or model grid-points respective interpolation operators I will be used. In addition to interpolation from the above grid to the desired locations these operators provide vertical or horizontal differentiation to derive physical quantities (temperature from geopotential height or wind components from stream function and velocity potential). Then the representation of $\mathbf{P}^b$ between arbitrary locations is:

$$\mathbf{P}^b = I\,\mathbf{W}\,\mathbf{L}\,\mathbf{D}\,\mathbf{L}^T\,\mathbf{W}^T\,I^T \tag{7.62}$$

In order to represent multivariate correlations $\mathbf{L}$ separates in matrix blocks:

$$\mathbf{L} = \begin{pmatrix} \mathbf{L}_h & & & \\ \mathbf{L}_{h\psi} & \mathbf{L}_{\psi_u} & & \\ . & . & \mathbf{L}_\chi & \\ . & . & . & \mathbf{L}_q \end{pmatrix} \tag{7.63}$$

In this representation $\mathbf{L}_h$ describes the covariances of geopotential height, $\mathbf{L}_\chi$ covariances of velocity potential and $\mathbf{L}_q$ that of moisture. $\mathbf{L}_{h\psi}$ represents cross-correlations between geopotential height and stream function and $\mathbf{L}_{\psi_u}$ covariances of the part of stream function that is not correlated with geopotential height. If required further cross correlation terms can be added to the lower triangle of the block matrix.

In an elaborate formulation the cross correlation term $L_{h\psi}$ is a sparse matrix in wavelet representation. In our current more simple formulation we will still use a balance operator in spatial representation describing a geographically varying geostrophic coupling.

7.3.2 Detailed description of the separable operator approach

Equation (7.63) is fully general and does not assume separability in the horizontal and vertical direction. However we start with a formulation where the separable analytical approach described in Section 7.2 can be replaced step by step with matrices derived from the NMC method.

First the vertical covariances $c_{i,j}$ in equations (7.30,7.33,7.34), will be replaced by an explicit matrix representation:

$$c_{i,j} = I_{v\,i}\, E_i(\mathbf{r}_i)\, \mathbf{W}_v\, \mathbf{L}_v(\lambda_i)\, \mathbf{L}_v(\lambda_j)^T\, \mathbf{W}_v^T\, E_j(\mathbf{r}_j)\, I_{v\,j}^T \tag{7.64}$$

Here I_i describes the interpolation from the vertical grid of $\tilde{\mathbf{P}}^b$ in wavelet representation (typically 64 levels in $\log p$) to the level of the model grid or observation, E_i is a correction term for the error (square root of the variance), depending on horizontal location $\mathbf{r}_i$, $\mathbf{W}_v$ the vertical wavelet transformation and $\mathbf{L}_v$ a factorization of the wavelet-transformed vertical correlation matrix derived by the NMC method for the respective geographical latitude λ_i.

If the horizontal covariance function $\mathbf{F}$ in (7.30,7.33,7.34) is represented as a matrix as well, with coefficients depending only on the horizontal distance $(\mathbf{r}_i - \mathbf{r}_j)$, and being non-zero only for corresponding vertical indices of the matrices $\mathbf{L}$ (in wavelet representation), we formally end up with the operator description:

$$\mathbf{P}^b_{i\,j} = I_{v\,i}\, E_i(\mathbf{r}_i)\, \mathbf{W}_v\, \mathbf{L}_v(\lambda_i)\, \mathbf{F}(\mathbf{r}_i - \mathbf{r}_j) \mathbf{L}_v(\lambda_j)^T\, \mathbf{W}_v^T\, E_j(\mathbf{r}_j)\, I_{v\,j}^T \tag{7.65}$$

Actually the operator $\mathbf{F}_h(\mathbf{r}_i - \mathbf{r}_j)$ is represented by a sparse covariance matrix in wavelet representation, fitted by the NMC method, leading to:

$$\mathbf{P}^b_{i\,j} = I_i\, E_i\, \mathbf{W}_v\, \mathbf{L}_v(\lambda_i)\, \mathbf{W}_h\, \mathbf{L}_h\, \mathbf{L}_h^T\, \mathbf{W}_h^T\, \mathbf{L}_v^T(\lambda_j)\, \mathbf{W}_v^T\, E_j\, I_j^T \tag{7.66}$$

$\mathbf{W}_h$ is the 2-dimensional horizontal wavelet transformation and $\mathbf{L}_h$ a factorization of the horizontal covariance matrix. Formulation (7.66) now allows a variation of the parameters of the horizontal covariance matrix with geographical location (latitude). If the horizontal correlations $\mathbf{F}_h$ depend on the vertical index (in wavelet representation), it is mostly equivalent to formulations proposed in [5]. There the vertical covariances are not described by a wavelet representation but by EOFs of the vertical covariance matrix. This approach to a large extent also fulfills the requirements for a full 3-dimensional non-separable formulation. Merely $\mathbf{W}_v\, \mathbf{L}_v(\lambda_i)\, \mathbf{W}_h\, \mathbf{L}_h$ has to be replaced by 3-d matrices $\mathbf{W}\,\mathbf{L}$.

By splitting up horizontal and vertical interpolation and explicitly introducing balance and transposition operators (7.66) yields:

$$\mathbf{P}^b_{i\,j} = E_i\, I_{vi}\, I_{hi}\, \mathbf{K}\, \mathbf{W}_v\, \mathbf{L}_v(\lambda_i)\, \mathbf{T}\, \mathbf{W}_h\, \mathbf{L}_h\, \mathbf{L}_h^T\, \mathbf{W}_h^T\, \mathbf{T}^T\, \mathbf{L}_v^T(\lambda_j)\, \mathbf{W}_v^T\, \mathbf{K}^T\, I_{hj}^T\, I_{vj}^T\, E_j \tag{7.67}$$

In detail the operators are as follows:

Multiplication with standard deviations (E)

Multiplication with the errors E (standard deviations) yields the representation of covariance functions from the correlation functions.

Interpolation operators (I_h, I_v)

The interpolation operators interpolate from the regular (or Gaussian) grid ($\log p, \lambda, \phi$) of the $\widetilde{\mathbf{P}^b}$ representation to the location of the observation or the model grid. In case of temperature and wind they also perform vertical and horizontal differentiation to derive these quantities from geopotential height, stream function and velocity potential (later on denoted as $\mathbf{D}_v$, $\mathbf{D}_x$ and $\mathbf{D}_y$).

Balance operator (K)

For modeling multivariate covariances the correlated quantities (stream function ψ and geopotential height h) are represented as linear combinations of uncorrelated quantities (ψ_u and h). Currently the balance operator $\mathbf{K}$ is a factor depending on latitude only. In a more elaborate representation it could be a (wavelet transformed) matrix and incorporate more correlated quantities (compare with (7.63)).

vertical wavelet transformation ($\mathbf{W}_v$)

In the current implementation assuming separability the vertical wavelet transformation $\mathbf{W}_v$ is not urgently required. In that case $\mathbf{L}_v$ would be represented in physical space. Only if the horizontal correlation functions shall depend on vertical scales or the number of vertical grid-points becomes large it will be needed.

Square root of vertical covariance matrix ($\mathbf{L}_v$)

Vertical correlations are modeled by means of a representation of the vertical covariances in physical or wavelet transformed space.

Transposition (T)

The operators described so far ($E\, I_v\, I_h\, \mathbf{K}\, \mathbf{W}_v\, \mathbf{L}_v$) act on columns at the location of the observations (or adjacent grid columns where $\widetilde{\mathbf{P}^b}$ is represented). Thus they are calculated by the processors which hold the respective observations.

The operators described below ($\mathbf{W}_h\, \mathbf{L}_h$) will act in a horizontal plane. In the parallel environment we distribute the uncorrelated parameters (h, ψ_u, χ, q) and levels of the $\widetilde{\mathbf{P}^b}$ representation over processors. The redistribution is performed by the transposition operator $\mathbf{T}$.

Horizontal wavelet transformation ($\mathbf{W}_h$)

The horizontal wavelet transformations act in the planes described above.

Square root of horizontal covariance matrices ($\mathbf{L}_h$)

Multiplication of vectors with these sparse matrices is performed by the respective operators implemented in the data assimilation code. So far no advantage is taken from the horizontal homogeneity of the matrices.

The operator approach is able to represent the OI covariance model as well as the covariance model derived by the NMC method. In the former case (wavelet-)representations of the respective analytical covariance model are used. Currently we use a mixed mode

where vertical covariance functions are taken from the NMC method but horizontal correlations mimic the former analytical covariance model.

Explicitly splitting up (7.67) for the different quantities of the multivariate formulation yields the explicit expression (7.68) for the square root of $\mathbf{P}^b$, first for the analytic (OI) covariance model . The $\zeta_{..}$ represent normally distributed uncorrelated deviations. The $\delta..$ indicate the resulting correlated multivariate deviations. $\mathbf{D}_x$, $\mathbf{D}_y$, $\mathbf{D}_v$ are the interpolation operators including horizontal or vertical differentiation.

$\mu(\lambda)$, ν and L_h are the prescribed coefficients of the OI covariance model, describing geostrophic balance, partitioning of rotational and divergent wind, and horizontal length scale. The respective terms in (7.68) are the explicit representation of the balance operator $\mathbf{K}$.

$$
\begin{array}{lcllllll}
\delta z & = & E_z & & \mathbf{I}_h & \mathbf{I}_v & & \mathbf{W}_v\mathbf{L}_{vz} \quad \mathbf{W}_h\mathbf{L}_{hz} \quad \zeta_z \\
\delta t & = & E_t & & \mathbf{I}_h & \mathbf{D}_v & & \mathbf{W}_v\mathbf{L}_{vz} \quad \mathbf{W}_h\mathbf{L}_{hz} \quad \zeta_z \\
\delta u & = & E_u & L_h & (-\mathbf{D}_y) & \mathbf{I}_v & \sqrt{1-\nu} \quad \mu(\lambda) & \mathbf{W}_v\mathbf{L}_{vz} \quad \mathbf{W}_h\mathbf{L}_{hz} \quad \zeta_z \\
 & + & E_u & L_h & (-\mathbf{D}_y) & \mathbf{I}_v & \sqrt{1-\nu} \quad \sqrt{1-\mu^2} & \mathbf{W}_v\mathbf{L}_{v\psi_u} \quad \mathbf{W}_h\mathbf{L}_{h\psi_u} \quad \zeta_{\psi_u} \\
 & + & E_u & L_h & \mathbf{D}_x & \mathbf{I}_v & \sqrt{\nu} & \mathbf{W}_v\mathbf{L}_{v\chi} \quad \mathbf{W}_h\mathbf{L}_{h\chi} \quad \zeta_\chi \\
\delta v & = & E_v & L_h & \mathbf{D}_x & \mathbf{I}_v & \sqrt{1-\nu} \quad \mu(\lambda) & \mathbf{W}_v\mathbf{L}_{vz} \quad \mathbf{W}_h\mathbf{L}_{hz} \quad \zeta_z \\
 & + & E_v & L_h & \mathbf{D}_x & \mathbf{I}_v & \sqrt{1-\nu} \quad \sqrt{1-\mu^2} & \mathbf{W}_v\mathbf{L}_{v\psi_u} \quad \mathbf{W}_h\mathbf{L}_{h\psi_u} \quad \zeta_{\psi_u} \\
 & + & E_v & L_h & \mathbf{D}_y & \mathbf{I}_v & \sqrt{\nu} & \mathbf{W}_v\mathbf{L}_{v\chi} \quad \mathbf{W}_h\mathbf{L}_{h\chi} \quad \zeta_\chi \\
\delta rh & = & E_{rh} & & \mathbf{I}_h & \mathbf{I}_v & & \mathbf{W}_v\mathbf{L}_{v\,rh} \quad \mathbf{W}_h\mathbf{L}_{h\,rh} \quad \zeta_{rh}
\end{array}
\tag{7.68}
$$

The NMC method in addition to the correlations also estimates variances or standard deviations $\sigma_{..}$. Furthermore the length scales are not prescribed but implicitly provided by the correlations. Furthermore the latitudinal dependence of $\mu(\lambda)$ is not prescribed but estimated from the data. Consequently the balance operator is based on these quantities and takes the different explicit representation given below:

$$
\begin{array}{lcllllll}
\delta z & = & E_z & & \mathbf{I}_h & \mathbf{I}_v & & \mathbf{W}_v\mathbf{L}_{vz} \quad \mathbf{W}_h\mathbf{L}_{hz} \quad \zeta_z \\
\delta t & = & E_t & & \mathbf{I}_h & \mathbf{D}_v & & \mathbf{W}_v\mathbf{L}_{vz} \quad \mathbf{W}_h\mathbf{L}_{hz} \quad \zeta_z \\
\delta u & = & E_u & \sigma_u^{-1} & (-\mathbf{D}_y) & \mathbf{I}_v & \sigma_\psi \quad \mu(\lambda) & \mathbf{W}_v\mathbf{L}_{vz} \quad \mathbf{W}_h\mathbf{L}_{hz} \quad \zeta_z \\
 & + & E_u & \sigma_u^{-1} & (-\mathbf{D}_y) & \mathbf{I}_v & \sigma_{\psi_u} & \mathbf{W}_v\mathbf{L}_{v\psi_u} \quad \mathbf{W}_h\mathbf{L}_{h\psi_u} \quad \zeta_{\psi_u} \\
 & + & E_u & \sigma_u^{-1} & \mathbf{D}_x & \mathbf{I}_v & \sigma_\chi & \mathbf{W}_v\mathbf{L}_{v\chi} \quad \mathbf{W}_h\mathbf{L}_{h\chi} \quad \zeta_\chi \\
\delta v & = & E_v & \sigma_v^{-1} & \mathbf{D}_x & \mathbf{I}_v & \sigma_\psi \quad \mu(\lambda) & \mathbf{W}_v\mathbf{L}_{vz} \quad \mathbf{W}_h\mathbf{L}_{hz} \quad \zeta_z \\
 & + & E_v & \sigma_v^{-1} & \mathbf{D}_x & \mathbf{I}_v & \sigma_{\psi_u} & \mathbf{W}_v\mathbf{L}_{v\psi_u} \quad \mathbf{W}_h\mathbf{L}_{h\psi_u} \quad \zeta_{\psi_u} \\
 & + & E_v & \sigma_v^{-1} & \mathbf{D}_y & \mathbf{I}_v & \sigma_\chi & \mathbf{W}_v\mathbf{L}_{v\chi} \quad \mathbf{W}_h\mathbf{L}_{h\chi} \quad \zeta_\chi \\
\delta rh & = & E_{rh} & & \mathbf{I}_h & \mathbf{I}_v & & \mathbf{W}_v\mathbf{L}_{v\,rh} \quad \mathbf{W}_h\mathbf{L}_{h\,rh} \quad \zeta_{rh}
\end{array}
\tag{7.69}
$$

Technically switching between the OI and NMC model is achieved by assigning different values to certain components in the description of the covariance operator `covm`:

Variable	OI model	NMC model
`covm% L_h`	L_h	1
`covm% c_h_psi`	μ	$\sigma_\psi\ \mu$
`covm% c1_h_psi`	$\sqrt{1-\mu^2}$	σ_{ψ_u}
`covm% sqnu`	$\sqrt{\nu}$	σ_χ
`covm% sq1nu`	$\sqrt{1-\nu}$	1
`covm% sdev_u`	1	σ_u
`covm% sdev_v`	1	σ_v

7.3.3 Implementation of Background Error Covariace Matrices

This section descripes the implementation of the various B-matrix approaches which may be used in the 3dVar and VarEnKF. The routines to multiply a vector with the $\mathbf{P}^b$ matrix are located in module `mo_bg_err_2d`. Main entries are:

Name	Notation	Left hand side (output)	Right hand side (input)
`apply_B_oo`	$\mathbf{H}I\mathbf{P}^bI^T\mathbf{H}^T$	observation space	observation space
`apply_B_io`	$I\mathbf{P}^bI^T\mathbf{H}^T$	interpolation space	observation space
`apply_B_oi`	$\mathbf{H}I\mathbf{P}^bI^T$	observation space	interpolation space
`apply_B_mi`	$I'\mathbf{P}^bI^T$	model grid	interpolation space
`apply_B_ii`	$I\mathbf{P}^bI^T$	interpolation space	interpolation space

Routines `apply_B_oo`, `apply_B_io`, and `apply_B_oi` finally call `apply_B_ii` and in addition apply the Jacobi matrix $\mathbf{H}$ of the observation operator. Routines `apply_B_mi` and `apply_B_ii` are further described below. In dependence on namelist settings they use either $\mathbf{P}^b_{OI}$ or a combination of $\mathbf{P}^b_{NMC}$ and $\mathbf{P}^b_{EnKF}$.

$\mathbf{P}^b_{NMC}$ – PSAS

Multiplication of $\mathbf{P}^b$ by a vector as required by the PSAS algorithm of the 3dvar or VarEnKF is performed by subroutine `apply_B_ii`. This routine calls the specific routine `apply_B_2d` for NMC covariances and further branches into a number of nested calls to lower level routines:

<table>
<tr><td colspan="15">B_ii</td></tr>
<tr><td colspan="15">B_ii_2d</td></tr>
<tr><td colspan="6">LvWvKIhIv</td><td>T</td><td colspan="2">WhLh</td><td colspan="2">WhLh_t</td><td>T_t</td><td colspan="6">LvWvKIhIv_t</td></tr>
<tr><td colspan="3">IhIv</td><td rowspan="2">K</td><td rowspan="2">$\mathbf{W}_v$</td><td rowspan="2">$\mathbf{L}_v$</td><td rowspan="2">T</td><td rowspan="2">$\mathbf{W}_h$</td><td rowspan="2">$\mathbf{L}_h$</td><td rowspan="2">$\mathbf{L}_h^T$</td><td rowspan="2">$\mathbf{W}_h^T$</td><td rowspan="2">$\mathbf{T}^T$</td><td rowspan="2">$\mathbf{L}_v^T$</td><td rowspan="2">$\mathbf{W}_v^T$</td><td rowspan="2">$\mathbf{K}^T$</td><td colspan="3">IhIv_t</td></tr>
<tr><td>E_i</td><td>I_{vi}</td><td>I_{hi}</td><td>I^T_{hj}</td><td>I^T_{vj}</td><td>E_j</td></tr>
</table>

The data flow in between the nested calls is shown below. Input variables to a given routine are shown in green, output variables in red and meta data parameters in **black**:

```
B_ii              (  obs,                                  x,  y,  e_fi                 )
  B_ii_2d         (                                        x,  y,  e_fi,  sink          )
    LvWvKIhIv_t   (                  xw,                   x,      e_fi,  sink          )
      IhIv_t      (                       xv,              x,      e_fi,  sink          )
    T_t           (           xh,    xw,                                          io    )
    WhLh_t        (      xd,  xh,                                                       )
    WhLh          (      xd,  xh,                                                       )
    T             (           xh,    xv,                                          io    )
    LvWvKIhIv     (                  xw,                   y,  e_fi,  sink          )
      IhIv        (                       xv,              y,  e_fi,  sink          )
```

Description of parameters:

Name	Type	Description
x	t_vector	input vector (gradient of cost function) in interpolation space (control variables interpolated to the location of the observations).
xv	real(wp) (:,:,:)	gradient/increment of prognostic variables at the locations of the columns of the NMC covariance grid.
xw	real(wp) (:,:,:)	gradient/increment of vertical 'modes' (multiplied by the square root of vertical covariance matrix) at the locations of the columns of the NMC covariance grid.
xh	real(wp) (:,:,:)	same as xv, but re-distributed to the processors holding planes of the different control variables.
xd	real(wp) (:,:,:)	same as xh, but horizontally wavelet transformed coefficients. (Actually the transformed control variables with unit B matrix.)
y	t_vector	result (analysis increment) in interpolation space.
obs	t_obs_set	observation meta data.
e_fi	t_vector	background error in interpolation space.
sink	t_vector	temporary to hold gradient/increment of sink variables. Sink variables are handled locally by each observation report. The respective gradients/increments are extracted from the input vector x, bypassed, and included again in y.
io	t_intop	coefficients (taken from intop_obs, module mo_bg_err_io) for interpolation from the grid of the NMC covariances to the locations of the observations.

$\mathbf{P}^b_{NMC}$ – Postmultiplication

Multiplication of $\mathbf{P}^b$ by a vector in the Post multiplication of the 3dvar or VarEnKF is performed by subroutine `apply_B_mi`. The algorithm calls the specific routine `apply_B_mi_2d` for NMC covariances and branches a number of nested calls to lower level routines similar to `apply_B_ii_2d` described above. Merely the final transposition and interpolation step is not targeting to the observations but to the model grid.

B_mi																				
B_mi_2d																				
L_m											L_i_t									
LvWvKIhIv_m					T	WhLh			WhLh_t		T_t	LvWvKIhIv_t								
IhIv_m			LvWvK_m												IhIv					
E_i	I_{vi}	I_{hi}	$\mathbf{K}$	$\mathbf{W}_v$	$\mathbf{L}_v$	$\mathbf{T}$	$\mathbf{W}_h$	$\mathbf{L}_h$	$\mathbf{L}_h^T$	$\mathbf{W}_h^T$	$\mathbf{T}^T$	$\mathbf{L}_v^T$	$\mathbf{W}_v^T$	$\mathbf{K}^T$	I_{hj}^T	I_{vj}^T	E_j			

The data flow in between the nested calls is shown below. Input variables to a given routine are shown in green, output variables in red and meta data parameters in **black**:

```
B_mi                ( a_m,  cbg,  obs,                                  x,  lnewpl,  e_fi              )
  B_mi_2d           ( a_m,  cbg,                                        x,  lnewpl,  e_fi,             )
    L_i_t           (                    xd,                            x,          e_fi,             )
      LvWvKIhIv_t   (                                  xw,  xv,         x,          e_fi,             )
        IhIv_t      (                                       xv,         x,          e_fi,             )
      T_t           (                         xh,  xw,                                        io      )
      WhLh_t        (                    xd,  xh,                                                     )
    L_m             ( a_m,  cbg,         xd,                                 lnewpl,                 )
      WhLh          (                    xd,  xh,                                                     )
      T             (                         xh,  xw,                                        io_cbg  )
      LvWvKIhIv_m   ( a_m,  cbg,               xw,                         lnewpl,          io_cbg  )
        LvWvK_m     (                        xw,  xv,                                       io_cbg  )
        IhIv_m      ( a_m,  cbg,                  xv,                    lnewpl,          io_cbg  )
```

The adjoint calculations correspond to those in subroutine `B_ii_2d` used in the PSAS solver. Merely in the forward calculations the final transposition and interpolation is not targeting to the observations but to the model grid.

Name	Type	Description
x	t_vector	input vector (gradient of cost function) in interpolation space (control variables interpolated to the location of the observations).
xv	real(wp) (:,:,:)	gradient/increment of prognostic variables at the locations of the columns of the NMC covariance grid.
xw	real(wp) (:,:,:)	gradient/increment of vertical 'modes' (multiplied by the square root of vertical covariance matrix at the locations of the columns of the NMC covariance grid.
xh	real(wp) (:,:,:)	same as xv, but re-distributed to the processors holding planes of the different control variables.
xd	real(wp) (:,:,:)	same as xh, but horizontally wavelet transformed coefficients. (Actually the transformed control variables with unit B matrix.)
a_m	t_cols	result (analysis increment) on the model grid.
obs	t_obs_set	observation meta data.
cbg	t_cols	reference state (first guess). The derived type holds information on the target and destination gridpoints (processor, grid indices) for transposition and interpolation purposes.
e_fi	t_vector	background error in observation space.
lnewpl	logical	flag for steering of interpolation.
io	t_intop	coefficients for interpolation from the NMC covariance grid to the locations of observations (taken from intop_obs, module mo_bg_err_io).
io_cbg	t_intop	coefficient for interpolation from the NMC covariance grid to the model grid (derived on the fly from the reference state info `cbg`.

$\mathbf{P}^b_{EnKF}$ – PSAS

Multiplication of $\mathbf{P}^b$ by a vector as required by the PSAS algorithm of the 3dvar or VarEnKF is performed by subroutine `apply_B_ii`. The routine calls the specific routine `apply_B_ii_ensb` for EnKF covariances and further branches into a number of nested calls to lower level routines:

<table>
<tr><td colspan="12">B_ii</td></tr>
<tr><td colspan="12">B_ii_ensb</td></tr>
<tr><td colspan="2">IhIv_ensb</td><td>get_cols_adj</td><td colspan="7">varenkf</td><td>get_cols</td><td colspan="2">IhIv_ensb_t</td></tr>
<tr><td>I_v</td><td>I_h</td><td>$\mathbf{T}_{og}$</td><td>$\hat{\mathbf{X}}$</td><td>$\mathbf{S}_v$</td><td>$\mathbf{S}_h$</td><td>$\mathbf{T}$</td><td>$\mathbf{S}_h^T$</td><td>$\mathbf{S}_v^T$</td><td>$\hat{\mathbf{X}}^T$</td><td>$\mathbf{T}_{og}^T$</td><td>I_h^T</td><td>I_v^T</td></tr>
</table>

The data flow in between the nested calls is shown below. Input variables to a given routine are shown in green, output variables in red and meta data parameters in **black**:

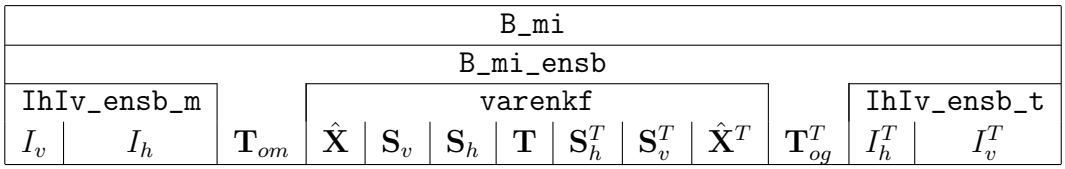
```
B_ii              (  obs,                                          x,  y,  e_fi  )
  B_ii_ensb       (                                                x,  y         )
    IhIv_ensb_t   (       Benkf%io                      xv,        x,            )
    get_cols_adj  (       Benkf%io%mc,        z,        cols,  ids               )
    varenkf       (       Benkf,        a,    z,                                 )
    get_cols      (       Benkf%io%mc,  a,          cols,  ids                   )
    IhIv_ensb     (       Benkf%io                      xv,              y       )
```

Description of parameters passed:

Name	Type	Description
x	t_vector	input vector (gradient of cost function) in interpolation space (control variables interpolated to the location of the observations).
xv	real(wp) (:,:,:)	gradient/increment of prognostic variables at the locations of the columns of the EnKF covariance model grid (but still on the processors which handle the observations).
cols	t_cols	same data as xv but copied to different data structure suitable for sending/receiving to the destination processor..
z	t_atm	gradient of prognostic variables, same data as cols but redistributed according to the ensemble grid decomposition.
a	t_atm	increment of prognostic variables.
y	t_vector	result (analysis increment) in interpolation space.
obs	t_obs_set	observation meta data (not used here).
Benkf	t_Benkf	variational ensemble B matrix meta data.
Benkf%io	t_intop	interpolation coefficients from the ensemble grid to the model grid locations.
Benkf%io%mc	t_mcols	model column meta data for transposition from the ensemble grid to the model grid decomposition.
ids	integer	bit flags to specify required parameters used by the observations.
e_fi	t_vector	background error in interpolation space (not used here).

$\mathbf{P}^b_{EnKF}$ – Postmultiplication

Multiplication of $\mathbf{P}^b$ by a vector in the Post multiplication of the 3dvar or VarEnKF is performed by subroutine `apply_B_mi`. The algorithm calls the specific routine `apply_B_mi_ensb` for EnKF covariances and branches into a number of nested calls to lower level routines similar to `apply_B_ii_ensb` described above. Merely the final transposition and interpolation is not targeting to the observations but to the model grid.



The data flow in between the nested calls is shown below. Input variables to a given routine are shown in green, output variables in red and meta data parameters in **black**:

```
B_ii              (  a_m,  cbg,                                          x,  lnewpl,  e_fi  )
  B_ii_ensb       (  a_m,  cbg,                                          x,  lnewpl         )
    IhIv_ensb_t   (           Benkf%io                      xv,          x,                 )
    get_cols_adj  (           Benkf%io%mc,          z,      cols,  ids                     )
    varenkf       (           Benkf,          a,    z,                                     )
...
```

Description of parameters passed:

Name	Type	Description
x	t_vector	input vector (gradient of cost function) in interpolation space (control variables interpolated to the location of the observations).
xv	real(wp) (:,:,:)	gradient/increment of prognostic variables on the columns of the EnKF covariance model grid (but still on the processors which handle the observations).
cols	t_cols	same data as xv but copied to different data structure suitable for sending/receiving to/from the destination processor.
z	t_atm	gradient of prognostic variables, same data as cols but redistributed according to the ensemble grid decomposition.
a	t_atm	increment of prognostic variables.
a_m	t_cols	result (analysis increment) on the model grid.
cbg	t_cols	reference state (first guess). The derived type holds information on the target and destination gridpoints (processor, grid indices) for transposition and interpolation purposes.
Benkf	t_Benkf	variational ensemble B matrix meta data.
Benkf%io	t_intop	coefficients for interpolation from the EnKF ensemble grid to the locations of the observations.
Benkf%io%mc	t_mcols	model column meta data for transposition from the EnKF ensemble grid to the decomposition of the observations.
ids	integer	bit flags to specify required parameters used by the observations.
lnewpl	logical	flag for interpolation steering.
e_fi	t_vector	background error in observation space (not used here).

Chapter 8

Observation Treatment

Now that the observation operator and observation types (Section 6) and the first of the data assimilation schemes (Section 7) have been described, a look is taken at how real-world observation quantities are treated in the assimilation:

- Section 8.1 describes the preprocessing of the data with the main goals of *quality control* and *data-thinning*.
- Section 8.2 (3dVar-specific) describes the procedure of *Variational Quality Control*.
- Section 8.3 (3dVar-specific) describes how humidity is assimilated using a transformed control variable
- Section 8.4 describes the preprocessing of satellite observations.

8.1 Quality Control and Thinning of Data

The data assimilation algorithms derive the optimum analysis based on the background (previous forecast) and the observations, taking into account the error estimates of both background and observations. The uncertainty of the observations is specified by the observation error covariance matrix $\mathbf{R}$. However, observational errors in general are not normally distributed as presumed by a quadratic observational cost function. Instead, observations have outliers and their error distributions may have fat tails. If observations with outliers (i.e with actual errors which do not obey to the assumptions made in the specification of $\mathbf{R}$) are used, they will degrade the analysis. A number of checks (comparison of observed values with background or climatological values, test for consistency) are applied in order to identify and reject outliers, so that they are not used in the subsequent analysis.

In general the observational errors are specified by a diagonal covariance matrix $\mathbf{R}$, i.e. observational errors are assumed to be uncorrelated. In principle error correlations may be specified by using a non-diagonal $\mathbf{R}$. However it is not straightforward to accurately determine the correlation pattern of observations. Thus, in practice, correlated (i.e. redundant) observations are accounted for by a spatial thinning algorithm.

The quality control checks and the thinning algorithms are described in this Section. In general only gross errors can be identified by this kind of tests. Remaining deviations from a normal distribution are handled by using a non-quadratic observational cost function in the variational scheme. This mechanism, called Variational Quality Control (VQC), is further described in Section 8.2.

Observation processing is performed as follows:

1. The observation input files are read. Some of the observational quality control (which does not rely on any comparison of observed to modelled data) is performed immediately so that observations are not processed unnecessarily.

2. The observations are redistributed over the available processors, observation operators are applied to the model background, and quality control checking and thinning is performed as described below.

3. The observations which will be used in the analysis are redistributed over the available processors and the analysis is derived.

4. Optionally the observation operators are re-run on the complete set of observations (as used in step 2.) for diagnostics.

Finally, each observations takes one of the following states:

`ACCEPTED` : The observation was used in the assimilation step (passed all quality checks) and finally received a VQC-weight larger than 0.5 .

`ACTIVE` : The observation was used in the assimilation step (passed all quality checks) and finally received a VQC-weight smaller than 0.5 .

`REJECTED` : The observation has failed at least one of the quality checks and was therefore not used in the assimilation.

`PASSIVE` : The observation was not intended to be used in the analysis but quality checks were applied (and passed successfully) for monitoring purposes. This is the usual procedure to assess the quality of new observation types.

`PASSIVE-REJECTED` : The observation was not intended to be used in the analysis and has failed at least one of the quality checks.

`DISMISSED` : The observation was rejected without being written to any of the monitoring files.

Observations are hold within reports (radiosonde ascend, SYNOP report, satellite field of view) which are generally related to a single horizontal coordinate but may (SYNOP report, aircraft report) or may not (radiosonde ascend, satellite field of view, radio occultation) be confined to a single vertical location. Each report consists of one or more quantities (temperature, humidity, wind, bending angle,...) observed at the same or at different levels. Some of the tests described below refer to single observed quantities, others to the report as a whole.

	Report Statistics				Single Observations	
status	`DISMISSED`	`PASSIVE`	`REJECTED`	`ACTIVE`	monitored	used
SYNOP	8 068	0	17	7 591	16 610	16 373
AIREP	19 204	70	55	10 427	31 579	29 204
SATOB	192 065	0	1 682	9 369	22 102	18 738
DRIBU	2 055	0	10	622	706	694
TEMP	117	1	3	626	101 298	28 071
PILOT	27	9	29	165	5 432	1 446
PAOB	82	0	27	276	303	276
SCATT	105 361	2 082	413	12 978	30 946	25 956
RAD	126 762	139 145	130	3 834	2 466 973	33 123
GPSRO	5	9	0	248	8 868	8 065
total	453 746	141 316	2 366	46 136	2 684 817	161 946

Table 8.1: Operational monitoring and usage of data in the pre-operational suite of the global data assimilation at 2010-07-20 00 UT. 'Monitored' observations have states of `PASSIVE-REJECTED`, `PASSIVE`, `REJECTED`, `ACTIVE` or `ACCEPTED`, 'used' observations have states of `ACTIVE` or `ACCEPTED`.

A report obtains the highest status value of all its observation (i.e. if all but one observations of a TEMP report were `REJECTED` and one stays `ACTIVE`, the status of the report remains `ACTIVE`). Table 8.1 shows the amount of data typically monitored and used during a 3 hours time window (at 00 UT) in the global assimilation system.

8.1.1 Generic Quality Checks

Generic checks applied to all observation types are described here. However, values of the quality check bounds are specific to the respective observation type.

Blacklist

Individual observation platforms (radiosonde or SYNOP stations, ships, aircrafts, ...) may have persistent errors or quality problems for various reasons (defective or inaccurate instrument, misspecified coordinates or station height, ...). In order to identify these situations the observations are continuously monitored and problematic stations are put on the so called 'blacklist'. The blacklist may apply to specific observed quantities (temperature, wind, humidity) and height ranges only. This kind of monitoring is not performed at DWD, instead the blacklist from ECMWF is used.

Data Selection

Criteria for data selection (geographic area, height range, underlying surface type, ...) may be specified individually for observation types as certain measurements are known to be inaccurate or not representative under certain conditions. More specific criteria may be applied to certain observation types due to their individual nature (e.g. cloud check for radiances).

First Guess Check

Within this Section, the observed value is denoted as o, the forecasted value as fg, and the specified observational and background error as e_o and e_{fg}, respectively. Prescribed values for bounds to be used are denoted as σ_{fg}, σ_{o_1}, ...

The first guess check rejects observations, those deviation from the first guess is larger than σ_{fg} times the expected standard deviation $(e_{fg}^2 + e_o^2)^{1/2}$. Furthermore the observed values have to be confined by prescribed climatological or physically meaningful bounds.

Specific checks are applied to reject small observed wind vectors if the first guess wind speed is large in order to reject misspecified aircraft and pilot reports.

Observation Error Check

If the nominal observational error is too large , the observation will be rejected, because it does not add sufficient information to the analysis. Observations are rejected, if either $e_o > \sigma_{o_1}$, $e_o/|o| > \sigma_{o_2}$ or $e_o/e_{fg} > \sigma_{o_3}$.

The specified observational error has to account for both the measurement error and the error of representativeness. If the nominal observational error is considerably smaller than the background error, this is an indication that the error of representativeness has not been properly considered. Thus e_o is bound below to a value of $\sigma_{o_4} \cdot e_{fg}$.

8.1.2 Thinning of Data

A virtual grid is defined with equidistant pressure levels in the vertical and a hexagonal (GME) grid in the horizontal. Depending on the desired density of data a suitable vertical spacing and horizontal resolution parameter (ni) is chosen. Each observation report is assigned to the nearest grid-point and for each grid-point only one report is kept. For observation reports not related to a vertical location (as surface observations or satellite observed radiances) only horizontal thinning is applied.

A number of rules is evaluated in order to identify the report to be kept for a given grid-point. If two reports are weighted equally next rule is applied in turn. The criteria can be specified individually for each observation type. The default sequence is as follows:

1. Status of the report (active, rejected,...)
2. Preference (individually specified for each observation type)
3. Minimum temporal distance to the analysis time
4. Quality of the data (individually specified for each observation type)
5. Minimum spatial distance to the virtual grid-point
6. Amount of data in the report
7. Sequence in the input file (to obtain a unique order if all other tests fail)

8.2 Variational Quality Control

This Section is specific to the 3dVar algorithm (Chapter 7) and can be skipped by users that are only interested in the pure EnKF (Chapter 9).

Unreliable observations are rejected by the first guess check as described in the previous section 8.1, before they enter the assimilation step. It is not recommended to make the first guess check too restrictive in order to keep the ability to correct bad forecasts. Instead observations should be checked by comparing them not only to the background, but to other independent observations as well. This is implicitly done by the Variational Quality Control described below.

In the context of Variational Quality Control more realistic distributions for the pdf of the observational error are used, taking into account outliers. These non-Gaussian distributions result in a non-quadratic cost function J_o. In section 7.1.2 we have already introduced non-quadratic observational cost functions and presented a method to solve the non-quadratic minimization problem. We have seen that this results in a generalized observation error covariance matrix $\mathbf{R}$, cf. equation (7.18).

In section 8.2.1 we will introduce three different formulations for the observation's probability density function and briefly discuss their properties. The handling of correlated, uncorrelated and wind observations is sketched in section 8.2.3. Section 8.2.4 gives a detailed description in the special case of a superposition of a Gaussian and a flat distribution.

8.2.1 Observation's Probability Density Function – Different Formulations

Different formulations of the probability distribution of observations can be used. They reflect different assumptions on the deviations of observational errors from a Gaussian distribution resulting in different formulations of the cost function with different convergence properties of the minimization algorithm. Here we show the uncorrelated case, where R denotes the variance. The parameter σ describes the shape of the pdf and is further specified in section 8.2.2.

Gaussian+Flat In this formulation the pdf is described by a superposition of a Gaussian and a flat distribution. The underlying idea is that the observation is either good, corresponding to a normal distribution, or bad without any information content at all, corresponding to a flat distribution. This formulation is described in more detail in section 8.2.4. A plot of this probability density function can be found in figure 8.1, the respective cost function is given in figure 8.2.

Huber In this approach the pdf is described by a Gaussian distribution in the inner part ($-\sigma < x < \sigma$), and by an exponential decay of the distribution in the tails

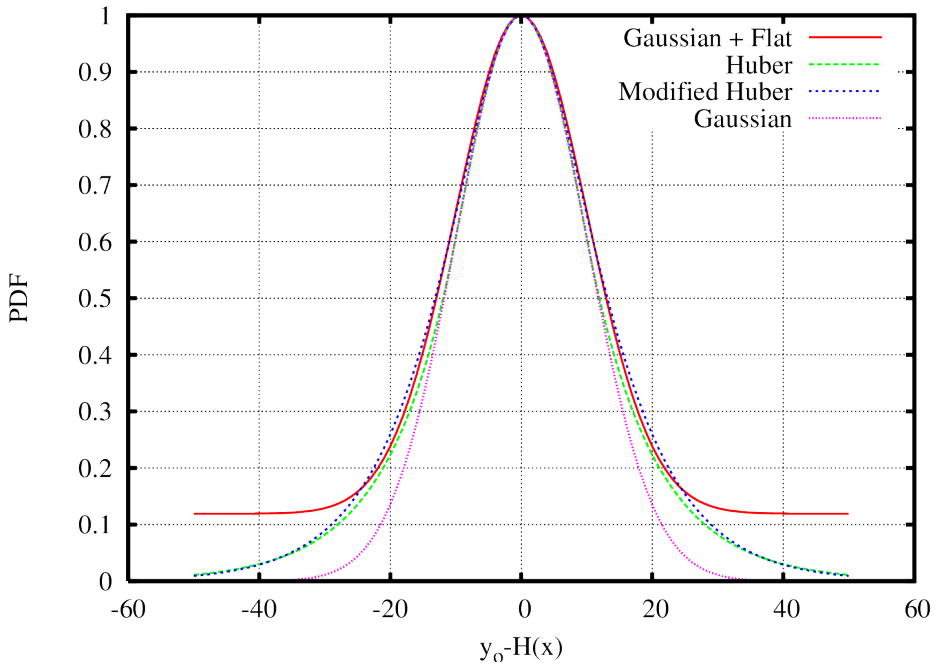


Figure 8.1: Probability density function for the three different formulations compared to the Gaussian distribution. The similarity of Huber and modified Huber function is clearly visible.

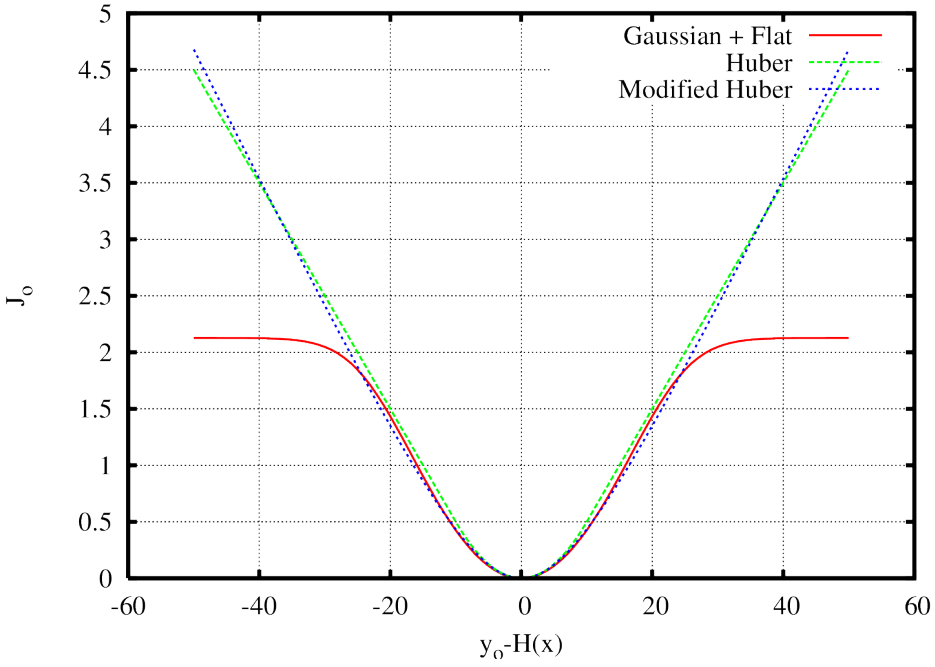


Figure 8.2: The observation cost functions corresponding to the three different formulations.

$(x < -\sigma, x > \sigma)$. This corresponds to a quadratic cost function with linear tails:

$$J_{qc}(x) = \begin{cases} \frac{1}{2}\frac{x^2}{R} & -\sigma \leq x \leq \sigma \\ \frac{1}{2}\frac{\sigma^2}{R} + (x-\sigma)\frac{\sigma}{R} & x > \sigma \\ \frac{1}{2}\frac{\sigma^2}{R} - (x+\sigma)\frac{\sigma}{R} & x < \sigma \end{cases} \tag{8.1}$$

This distribution is also called 'Huber norm', although it is not a norm in the mathematical sense. However, for small deviations x the formulation is similar to the L_2 norm whereas for large x it comes close to the L_1 norm, which is a very robust formulation for data with outliers. In the L_1 case the result of the analysis is the median of the distribution. A plot of this probability density function can be found in figure 8.1, the respective cost function is given in figure 8.2.

Modified Huber In this formulation the cost function is described by a branch of a hyperbel. σ is a parameter, further described in the context of weights later on.

$$J_{qc}(x) = \sqrt{0.7\frac{\sigma}{\sqrt{R^3}}}\sqrt{x^2 + 0.7\sigma\sqrt{R}} - 0.7\frac{\sigma}{\sqrt{R}} \tag{8.2}$$

For small deviations it approximates a quadratic, and for large x a linear function. This choice is very similar to the previous one. Thus we call it the modified Huber function. A plot of this probability density function can be found in figure 8.1, the respective cost function is given in figure 8.2.

Figure 8.1 shows the probability density function for the three different formulations compared to the Gaussian distribution. The similarity of Huber and modified Huber function is clearly visible.

The corresponding observation cost functions are given in figure 8.2. While the cost function is constant for large $\mathbf{y}^o - H(\mathbf{x})$ in the case of the Gaussian+Flat distribution, it grows linearly in case of the Huber and modified Huber function.

The gradient and the second derivative of the observation cost function are given in figure 8.3 and 8.4. Both, Gaussian+Flat and modified Huber function are continous, while the Huber function is discontinous. In case of the modified Huber function the second derivative is strictly positive.

Using pdf's with fat tails (Gaussian+Flat) results in a smaller absolute first derivative of J_o for large deviations $\mathbf{y}^o - H(\mathbf{x})$ (cf. figure 8.3). In the minimum of the total cost function ($J_b + J_o$ for all observations) the gradients from all contributions sum up to zero. Thus applying variational quality control results in a smaller weight for the observations with large deviations to the analysis. Thus we define the weight as the derivative of the cost function divided by the derivative of the corresponding quadratic function for the same x. The weight is displayed in figure 8.5. The extend of the tails of the pdf is controlled by the parameter σ, defined as the deviation $\mathbf{y}^o - H(\mathbf{x})$ for which the weight is 1/2. This definition applies to all formulations of the VQC.

In formulation two and three the analysis is still influenced by large outliers as a non-zero VQC weight is maintained in any case. On the contrary, large outliers are fully neglected in formulation one.

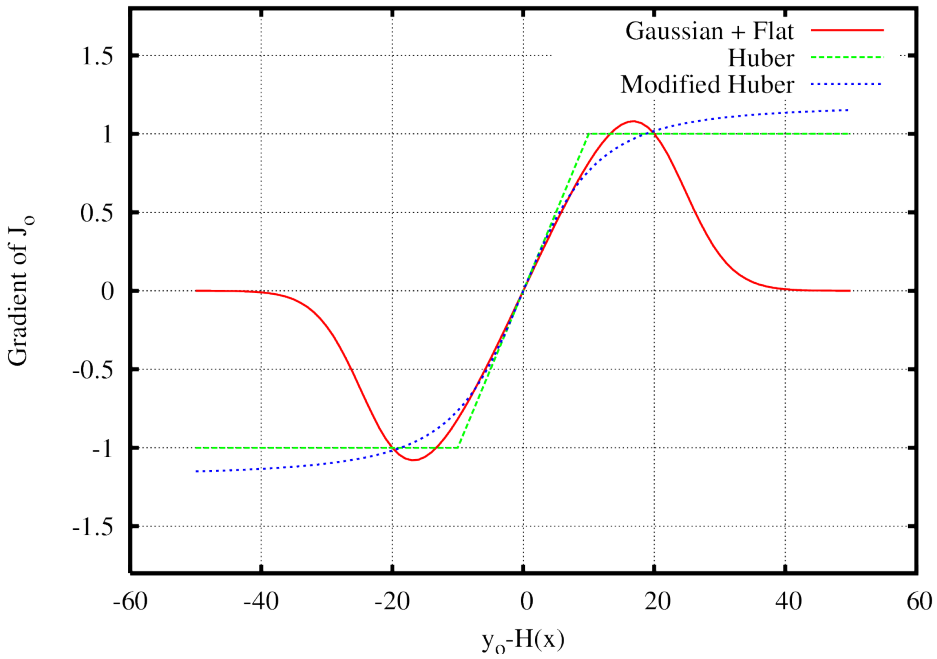


Figure 8.3: Gradient of the observation cost function for the three formulations.

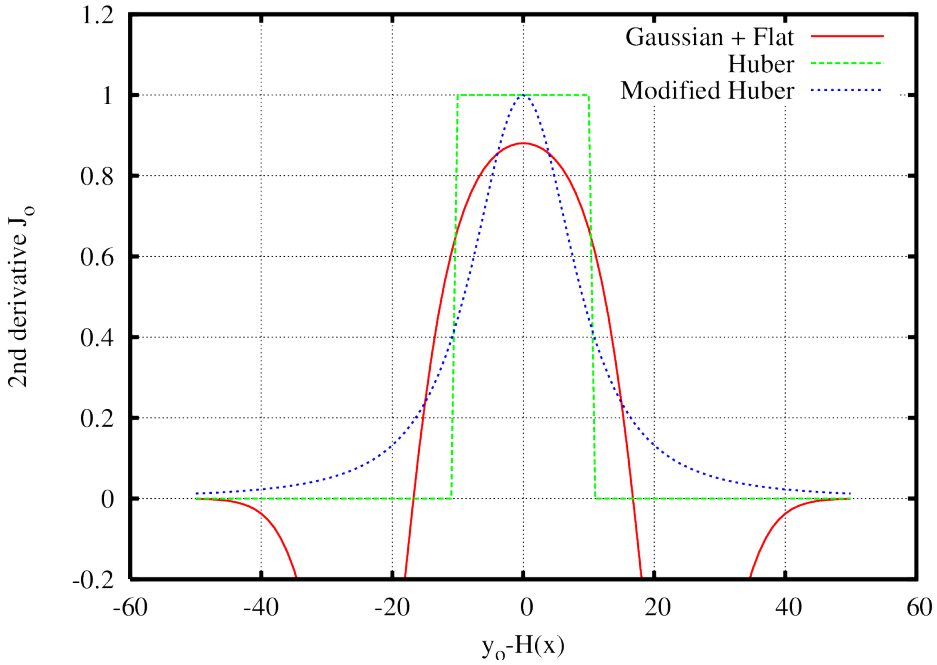


Figure 8.4: Second derivative of the observation cost function for the three formulations. While the Gaussian+Flat and the modified Huber version are continuous, the Huber version is discontinuous. Another important fact is the positivity of the modified Huber function.

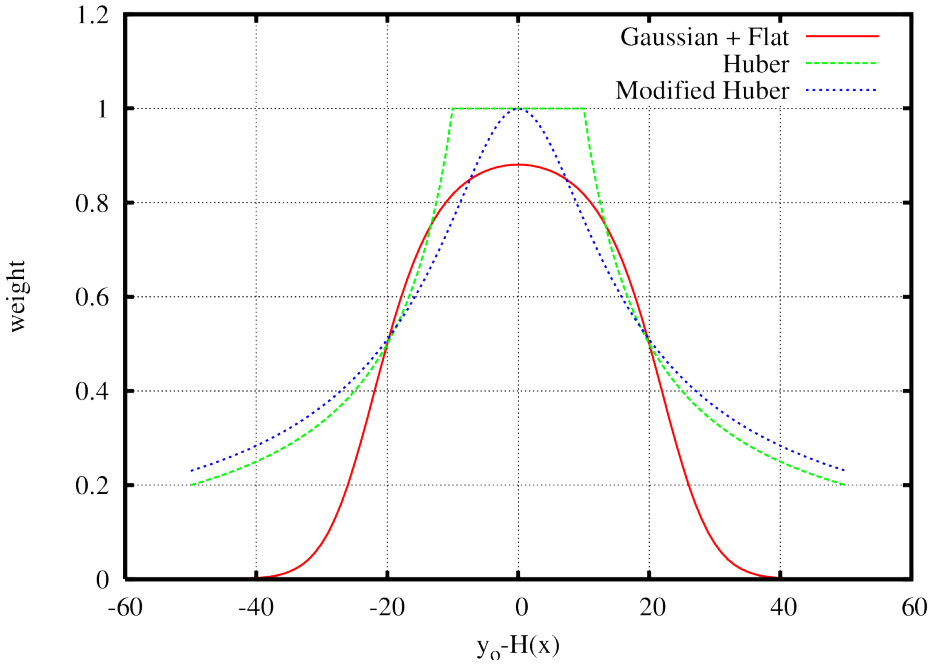


Figure 8.5: VQC weight for three different approaches.

The different formulations of VQC pdf and cost function have a large impact on the convergence properties of the minimisation algorithm. Formulation 1 (Gaussian+Flat) shows negative values of the second derivative of the observation cost function. This has the following consequences:

1. The total cost function may have multiple local minima.
2. The total cost function may have a negative second derivative. The preconditioned conjugate gradient algorithm (cf. Section 7.1.2) requires a quadratic local approximation to the true cost function with positive second derivatives (Hessian). As this is not guaranteed a less accurate approximation has to be used.

As a consequence the Huber function results in better convergence properties of the minimisation algorithm than the Gaussian+Flat formulation. The properties of the modified Huber functions are even better (as the second derivative is continuous and strictly positive). Thus the latter formulation is currently used operationally.

8.2.2 Parameterization of VQC

In order to specify the shape of the probability density distribution and the related cost function for the different formulations of VQC two parameters are required:

1. The nominal observational error σ_o in the case that no VQC is applied. For uncorrelated observations this parameter corresponds to the square root of the diagonal elements of the observation error covariance matrix $\mathbf{R}$. The parameter determines the second derivative of the pdf and cost function at $x = 0$.

2. The parameter ς_{vqc} determines the shape of the tails of the pdf. This parameter is defined so that for $x = \varsigma_{vqc} \cdot \sigma_o$ the weight of the observations is 0.5. The parameters of the different formulations are related to ς_{vqc} by:

Gaussian + Flat:	$\gamma = \exp\left(\frac{1}{2}\varsigma_{vqc}\right)$
Huber:	$\sigma = \frac{1}{2}\ \varsigma_{vqc}\ \sigma_o$
Modified Huber:	$\sigma = \frac{3}{4}\ \varsigma_{vqc}\ \sigma_o$

All figures of the previous section were generated with a choice of $\varsigma_{vqc} = 2$ and $\sigma_o = 10$. Thus all lines of figure 8.5 meet in the points (-20,0.5) and (20,0.5).

8.2.3 Correlated, Uncorrelated and Wind Observations

For independent, uncorrelated observations the joint pdf is the product of the two pdf's: $p(x_1, x_2) = p(x_1) \cdot p(x_2)$. The corresponding cost function $f(x_1, x_2) = f(x_1) + f(x_2)$ is shown in Figure 8.8 for the Gaussian+Flat formulation.

The assumption of independent, uncorrelated observations is not applicable to the observation of the two components of the horizontal wind vector. For this case a formulation with rotational symmetry is required. Thus, for wind observations, we use the same formulation of the cost function, but relate it to the absolute value of the wind vector deviation only: $f(x_1, x_2) = f(\sqrt{x_1^2 + x_2^2})$. In this case both components of the wind observation obtain the same VQC weight.

Despite wind observations, correlated observations are currently not used. A method to treat correlated observations is described in Section 8.2.4 for the case of the Gaussian+Flat formulation. No attempt has been made so far to implement the use of correlated observations in the framework of the Huber norm.

8.2.4 Detailed Description for Gaussian + Flat Distribution

In this section the modified cost function for the first formulation, the superposition of Gaussian and flat distribution, is calculated as an example. First and second derivative, which are needed to solve the optimization problem, are calculated as well.

Uncorrelated Data

In variational quality control a non-Gaussian probability density function $p^{\mathtt{QC}}$ is used for modeling the observation errors in order to formally include the existence of outliers in the formulation of the minimization problem. As proposed by [2] we assume a superposition of a Gaussian distribution N and a flat distribution F:

$$p^{\mathtt{QC}} = (1 - A)N + AF \tag{8.3}$$

A is the a priori probability that the observation is incorrect.

In the case of uncorrelated data, the covariance matrix R is a diagonal matrix. Only

diagonal terms contribute.

$$N_{\hat{y},\sigma_o}(y) = \frac{1}{\sigma_o\sqrt{2\pi}} \exp\left[-\frac{1}{2}\left(\frac{y-\hat{y}}{\sigma_o}\right)^2\right] \quad \text{with} \quad \hat{y} = H(x) \tag{8.4}$$

$$F_{\hat{y},d,\sigma_o}(y) = \begin{cases} \dfrac{1}{D} = \dfrac{1}{2d\sigma_o}, & \text{if} |y-\hat{y}| < D/2 \\ \\ 0 & \text{otherwise.} \end{cases} \tag{8.5}$$

y is the observed quantity and $\hat{y}$ its model equivalent. The distribution of incorrect data is assumed to be a flat distribution in a vicinity of $d \times \sigma_0$ around the observed value.

Note that other choices for $p^{\mathtt{QC}}$ are possible and that the distinction between correct and incorrect data is not mandatory. Other distributions $p^{\mathtt{QC}}$ are described in 8.2.1.

The observational cost function corresponding to a given probability density distribution p is given by

$$J_o = -\ln p + c \tag{8.6}$$

In general, c is chosen so that $J_o(\hat{y} = y) = 0$. A pure Gaussian error distribution corresponds to a quadratic cost function, as introduced in Section 7.1. The distribution given in Eq. (8.4) corresponds to

$$J_o(y)^{\mathtt{N}} = \frac{1}{2}\left(\frac{y-\hat{y}}{\sigma_o}\right)^2 \tag{8.7}$$

Its gradient is

$$\frac{\partial J_o^{\mathtt{N}}}{\partial \hat{y}} = -\frac{1}{\sigma_o}\left(\frac{y-\hat{y}}{\sigma_o}\right) \tag{8.8}$$

Non-Gaussian error distributions correspond to non-quadratic cost functions. The distribution (8.3) corresponds to:

$$J_o^{\mathtt{QC}} = -\ln\left[\frac{\gamma + \exp(-J_o^N)}{\gamma + 1}\right] \quad \text{with} \quad \gamma = \frac{A\sqrt{2\pi}}{(1-A)2d} \tag{8.9}$$

The gradient of this cost function is:

$$\frac{\partial J_o^{\mathtt{QC}}}{\partial \hat{y}} = w^{\mathtt{QC}} \frac{\partial J_o^{\mathtt{N}}}{\partial \hat{y}} \tag{8.10}$$

The term $w^{\mathtt{QC}}$ is the a posteriori probability that the observation is correct:

$$w^{\mathtt{QC}} = 1 - \frac{\gamma}{\gamma + \exp(-J_o^{\mathtt{N}})} \tag{8.11}$$

The second derivative of the cost function (8.9) is:

$$\frac{\partial^2 J_o^{\mathtt{QC}}}{\partial \hat{y}^2} = w^{\mathtt{QC}} \frac{1}{\sigma_o^2}\left(1 - \gamma \frac{(y-\hat{y})^2}{\sigma_o^2} \exp\left[\frac{1}{2}\frac{(y-\hat{y})^2}{\sigma_o^2}\right]\right) \tag{8.12}$$

In order to apply variational quality control, the parameters of the error model (A, d, and σ_o in Eq.(8.9)) must be specified. In praxis, the exact values of A and d as defined in equations (8.3) and (8.5) are not very relevant. Indeed the maximum size if outliers, $d \cdot \sigma_o$, is merely introduced to make F integrable. Instead of A and d the parameter γ is prescribed by specifying a threshold value σ_{rej} for $\hat{y} - y$, beyond which the observation is rejected with an a priory probability of P_{rej}:

$$\left(\frac{\sigma_{rej}}{\sigma_o}\right)^2 = 2 \ln \frac{P_{rej}}{(1 - P_{rej})\gamma} \tag{8.13}$$

The cost function and a posteriori probability for correct data is shown in Figure 8.6. The second derivative of the cost function (Hessian) and approximations to it are shown in Figure 8.7.

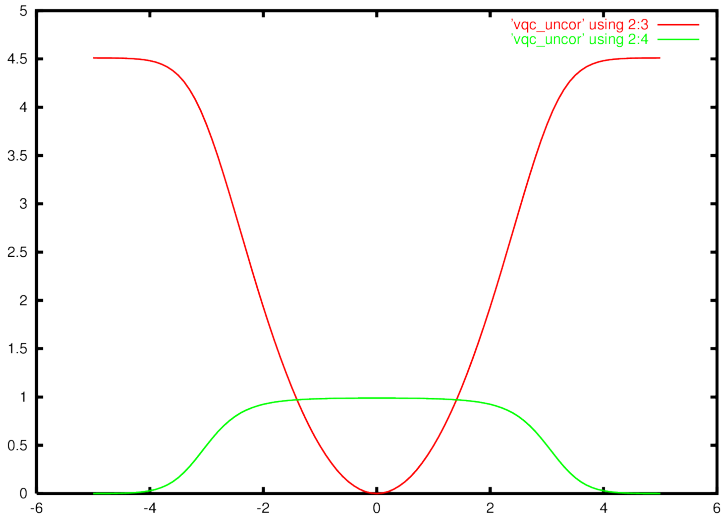


Figure 8.6: Cost function $J^{\mathtt{QC}}$ (red) and a posteriori probability for correct data $w^{\mathtt{QC}}$ (green) as a function of $(\hat{y} - y)/\sigma_o$ for a rejection limit of $\sigma_{rej} = 3\sigma_o$ and $P_{rej} = 0.5$.

For uncorrelated observations, the joint probability distribution is given by the product of the marginal distributions of the individual probability distributions. Consequently the cost function (cf. Figure 8.8) is the sum of the individual cost functions according to equation (8.9).

Wind Data

Wind observations consist of its u and v-component. The components are assumed to be uncorrelated. However if a wind component has gross errors, the whole observation, i.e. both of its components are rejected. This is described by the probability distribution $p_{uv}^{\mathtt{QC}}$:

$$p_{uv}^{\mathtt{QC}} = (1 - A_{uv}) N_u N_v + A_{uv} F_u F_v \tag{8.14}$$

with Normal distributions N_u, N_v and flat distributions F_u, F_v defined similar to equations (8.4), (8.5), but with y, $\hat{y}$ replaced by u, $\hat{u}$ or v, $\hat{v}$, respectively. The expressions

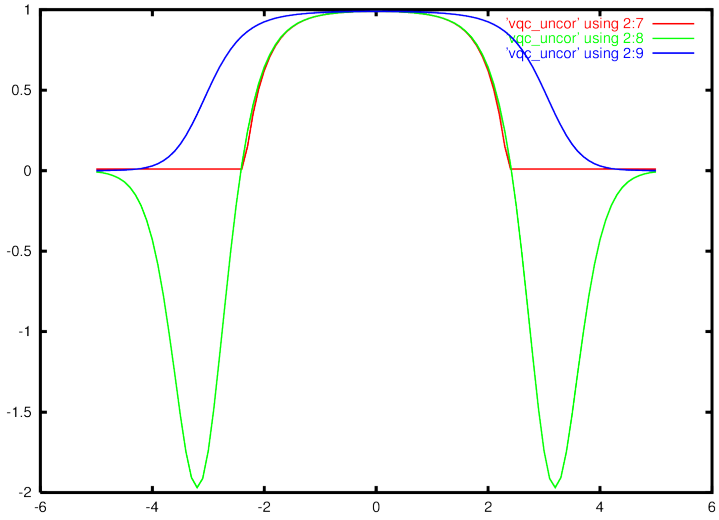


Figure 8.7: Second derivative of the cost function $\frac{\partial^2 J_o^{\mathtt{QC}}}{\partial(\hat{y}-y)^2}$ (green) and its approximation (limited to values $> 0.01/\sigma_o$) used in the Newton algorithm (red) as a function of $(\hat{y} - y)/\sigma_o$. The approximation $w^{\mathtt{QC}}/\sigma_o^2$ used by [7] is shown in blue.

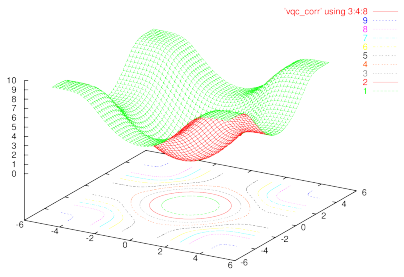


Figure 8.8: As figure 8.6, but for 2 uncorrelated observations

for the cost function (Figure 8.9) and its gradient remain as in equations (8.9) and (8.10) with $J_o^{\mathtt{N}}$ replaced by $J_{uv}^{\mathtt{N}}$

$$J_{uv}^{\mathtt{N}} = J_u^{\mathtt{N}} + J_v^{\mathtt{N}} \ , \tag{8.15}$$

γ replaced by γ_{uv}

$$\gamma_{uv} = \frac{A_{uv} 2\pi}{(1 - A_{uv}) 2d_u 2d_v} \ , \tag{8.16}$$

and $w^{\mathtt{QC}}$ replaced by $w_{uv}^{\mathtt{QC}}$

$$w_{uv}^{\mathtt{QC}} = 1 - \frac{\gamma_{uv}}{\gamma_{uv} + \exp(-J_{uv}^{\mathtt{N}})} \ . \tag{8.17}$$

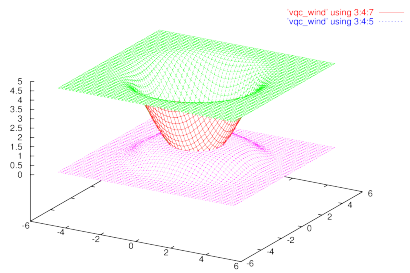


Figure 8.9: Cost function $J_{uv}^{\mathtt{QC}}$ (green) and a posteriori probability $w_{uv}^{\mathtt{QC}}$ for correct wind data (magenta).

The second derivatives of the cost function in the longitudinal direction to $\hat{\mathbf{y}} - \mathbf{y}$ is the same as those given in equation (8.12) (green graph in Figure 8.7), with $\mathbf{y}$ being the observed wind vector and $\hat{\mathbf{y}}$ the model equivalent. In the transversal direction it is (blue graph in Figure 8.7):

$$\frac{\partial^2 J_{uv}^{\mathtt{QC}}}{\partial \hat{y}_t^2} = w_{uv}^{\mathtt{QC}} \frac{1}{\sigma_o^2} \tag{8.18}$$

The cost function for wind data (Figure 8.9) is considerably different from a case there probabilities for gross error of the components are independent (Figure 8.8).

Correlated Data

The more complicated case with correlated data has implemented and will therefore be described, but it is currently not used.

In this section, we use normalized observation increments

$$\mathbf{u} = \mathbf{S}^{-1}(\hat{\mathbf{y}} - \mathbf{y}) \tag{8.19}$$

and observation error correlations $\mathbf{C}$ instead of covariances $\mathbf{R}$. $\mathbf{S}$ is the diagonal matrix consisting of the observation error standard deviations σ_o.

In case of no gross errors, the probability density distribution is described by a multivariate normal distribution:

$$N_n(\mathbf{u}) = ((2\pi)^n|\mathbf{C}|)^{-\frac{1}{2}} \exp\left(\frac{1}{2}\mathbf{u}^T\mathbf{C}^{-1}\mathbf{u}\right) \tag{8.20}$$

with $|\mathbf{C}|$ being the determinant of $\mathbf{C}$.

In case of observations $u_{j_1}, u_{j_2}, \ldots$ rejected due to gross errors, the distribution of the remaining correct observations $u_{i_1}, u_{i_2}, \ldots$ is described by the marginal probability density function:

$$N_{i_p,k}(\mathbf{u}) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \ldots N_n \; du_{j_1}, du_{j_2}, \ldots \tag{8.21}$$

The index k counts the total number $(0 \ldots n)$ of rejected observations and i_p runs over the different combinations $\frac{n!}{k!(n-k)!}$ of accepted or rejected observations for given k.

$N_{i_p,k}$ is given by a multivariate normal distribution as well:

$$N_{i_p,k}(\mathbf{u}) = \left((2\pi)^k|\mathbf{C}_{i_p,k}|\right)^{-\frac{1}{2}} \exp\left(\frac{1}{2}\mathbf{u}_{i_p,k}^T\mathbf{C}_{i_p,k}^{-1}\mathbf{u}_{i_p,k}\right) \tag{8.22}$$

with $\mathbf{C}_{i_p,k}$ being the observation error correlation matrix consisting of the rows and columns of $\mathbf{C}$ corresponding to the accepted observation increments $\mathbf{u}_{i_p,k}$ only.

Consequently the probability distribution of correlated observations with gross errors is given by the sum of distributions with different combinations of rejected data, weighted by their a priory probabilities:

$$p^{\mathtt{QC}} = \sum_{i_p,k} A_{i_p,k} N_{i_p,k} F_{i_p,k} \tag{8.23}$$

with $N_{i_p,k}$ given by equation (8.22), $F_{i_p,k}$ being the product of the flat distributions of the rejected observations according to equation (8.5), and $A_{i_p,k}$ the a priori probability of the combination of accepted or rejected data. If the a priory probabilities for rejection of the different components are independent, $A_{i_p,k}$ is the product of these independent probabilities:

$$A_{i_p,k} = \prod(1 - A_{i_1})(1 - A_{i_2}) \cdots \prod A_{j_1} A_{j_2} \ldots \tag{8.24}$$

with $A_{i_1}, A_{i_2}, \ldots$ being the a priory probability of gross error in the $n - k$ accepted observations and $A_{j_1}, A_{j_2}, \ldots,$ of the k rejected observations of the combination denoted by i_p, k. For dependent probabilities of rejections different a priory probabilities must be prescribed.

Again, instead of using the a priory probabilities A_i, A_j we can use parameters γ_j related to certain rejection limits, similar to equation (8.13). Then the correlation function (8.23) becomes:

$$p^{\mathtt{QC}} = \sum_{i_p,k} p_{i_p,k}^{\mathtt{QC}} = \frac{\sum_{i_p,k} \gamma_{i_p,k} |\mathbf{C}_{i_p,k}|^{-\frac{1}{2}} \exp\left(\frac{1}{2}\mathbf{u}^T\mathbf{C}_{i_p,k}^{-1}\mathbf{u}\right)}{\sum_{i_p,k} \gamma_{i_p,k}} \tag{8.25}$$

The cost function (Figure 8.10) of the distribution and its first and second derivatives are given by the expressions:

$$J^{\mathtt{QC}} = -\ln(p^{\mathtt{QC}}) \tag{8.26}$$

$$\frac{dJ^{\mathtt{QC}}}{d\mathbf{u}} = \frac{1}{p^{\mathtt{QC}}} \sum_{i_p,k} p^{\mathtt{QC}}_{i_p,k} \mathbf{C}^{-1}_{i_p,k} \mathbf{u} \tag{8.27}$$

$$\frac{d^2J}{d\mathbf{u}^2} = \frac{1}{p^{\mathtt{QC}}} \sum_{i_p,k} p^{\mathtt{QC}}_{i_p,k} (\mathbf{C}^{-1}_{i_p,k} - [\mathbf{C}^{-1}_{i_p,k}\mathbf{u}][\mathbf{C}^{-1}_{i_p,k}\mathbf{u}]^T) \ . \tag{8.28}$$

The probability for correct observations is:

$$w^{\mathtt{QC}} = \frac{1}{p^{\mathtt{QC}}} \sum_{i_p} p^{\mathtt{QC}}_{i_p,k} \delta_{i_p,k} \tag{8.29}$$

there $\delta_{i_p,k}$ denotes the vector with components of 1 for accepted data and 0 for rejected data.

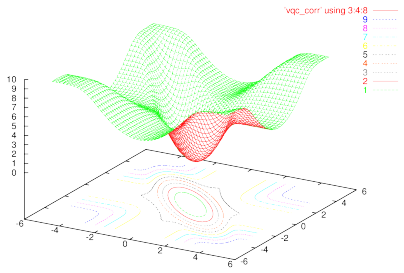


Figure 8.10: Cost function $J^{\mathtt{QC}}_{uv}$ for correlated data (correlation coefficient $\rho = -0.8$).

Practical Implementation

The evaluation of the observational cost function for n correlated data is very expensive because 2^n evaluations (including inversions of $n \times n$ matrices) of all the combinations of accepted and rejected must be performed. There exist different strategies to cope with the problem as discussed below. The first 3 items were also discussed by [2].

1. Evaluating all terms
   This choice is too expensive in general and can only be pursued for a small number of correlated observations.

2. Omitting some terms
   Some of the combinations may be omitted. In particular all combinations with $k > k_{max}$ may be omitted (that means all terms with more than k_{max} observations rejected) but the term with $k = n$ (all data rejected) kept. This has been done for

the wind data with $kk_{max} = 1$. This implies that the a priory probabilities for gross errors are not independent any more and especially the value of the probability $A_{i_p,n}$ (all data rejected) must be reconsidered.

3. Diagonalizing the problem
   If gross errors themselves are correlated in the same way as random errors, i.e. according to the matrix $\mathbf{C}$ the matrix may be diagonalized and variational quality control may be formulated in the diagonalized vector space. There the 'observations' are treated as uncorrelated. [2] argue that it is unphysical to assume that gross errors have the same correlations as random errors. On the other hand correlated observations occur for two reasons:

   (a) The observation operator may lead to correlated errors of representativeness.

   (b) The observations are not independent but derived from primary observations which may be assumed to be uncorrelated. In our case the geopotential observations are derived from temperature observations using the hydrostatic equation and some equation of state.

   Especially in the latter case gross errors cannot be assumed to be uncorrelated.

4. Calculating only the most important terms
   In general only a small number of terms should contribute significantly to the sums (8.25) to (8.29). Some algorithm may be sought to evaluate only the terms with large $p^{\mathtt{QC}}_{i_p,k}$. An algorithm was tested which for given k evaluates only those combinations which are reached from the most probable combination for $k-1$ by a limited number of changes to the sets of accepted and rejected observations. However this strategy was not fully satisfactory because considerably different values for $J_o^{\mathtt{QC}}$ could be obtained with different parameter settings.

5. Searching for other formulations of $J_o^{\mathtt{QC}}$
   It may be possible to find a simpler formulation for $J_o^{\mathtt{QC}}$ which has similar characteristics.

Currently only geopotential observations within one radio sounding are assumed to be correlated, but as there is no preselection of levels up to now, up to 80 correlated observations occur within one sounding. Evaluation of 2^{80} terms is not feasible and thus the following strategy has be pursued up to now for variational quality control (the parameters may be changed by namelist group /VARQC/ .):

$n \leq 10$:
All combinations are considered for.

$n \leq 20$:
All combinations with less than 6 rejected observations are calculated. Combinations with higher number of rejections are evaluated only partially (strategy 4 above).

$n \leq 30$:
All combinations with less than 4 rejected observations are calculated. Combinations with higher number of rejections are evaluated only partially.

$n \leq 50$:
All combinations with less than 3 rejected observations are calculated. Combinations with higher number of rejections are evaluated only partially.

$n \leq 100$:
All combinations with less than 2 rejected observations are calculated. Combinations with higher number of rejections are evaluated only partially.

$n > 100$:
All combinations with less than 2 rejected observations are calculated. Combinations with higher number of rejections are not considered.

8.3 Assimilation of humidity data

This Section is specific to the 3dVar algorithm (Chapter 7) and can be skipped by users that are only interested in the pure EnKF (Chapter 9).

8.3.1 Humidity sensitive observations

TEMP and aircraft data

Information on humidity is gained by direct in-situ observations from radiosondes (TEMPS) and aircraft observations. As radiosonde humidity observations have large biases in the upper troposphere, they are currently used only below 300 hPa. Aircrafts were only recently equipped with accurate humidity sensors (in the US and very view in Europe). Assimilation of aircraft humidity data is under investigation.

SYNOP data

Humidity observations from SYNOP, SHIP and BUOY reports are assimilated as relative humidity data. The model equivalent is taken from the lowest model level (10 m) without further adjustment to the station height.

AMSU-B and MHS data

Assimilation of humidity sensitive satellite radiances measured by the AMSU-B and MHS instruments is under preparation.

GNSS radio occultations

In the lower troposphere refractivity as observed by GNSS radio occultations depends on a linear combination of temperature and humidity.

8.3.2 Humidity in the stratosphere

Humidity is not assimilated in the stratosphere. Instead specific humidity is set to a constant climatological value of $q = 4. \cdot 10^{-6}$ Kg/Kg. The tropopause height is analysed in the same way as beforehand in the Optimum Interpolation system. In addition to that, analysis increments are forced to zero above the 250 hPa level.

8.3.3 Generalised humidity control variable

The background error model assumes a quadratic cost function J_b and a Gaussian background error distribution. This assumption cannot hold for a quantity as humidity which is confined to a finite interval ($q \geq 0$). For this reason a transformation to a generalised humidity variable is performed. For values larger than $rh_0 = 3\%$ generalised humidity gh is equal to relative humidity rh_w over water. Below rh_0 relative humidity differs from generalised humidity. There are 2 options for the choice of generalised humidity in this range:

1. exponential dependence of relative humidity on generalised humidity.

2. quadratic dependence of relative humidity on generalised humidity.

The transformation to generalised humidity has the following effects:

- The control variable gh is allowed to take any value within the variational analysis process. Positive definiteness of the derived relative humidity is ensured in any case.
- In the vicinity of the lower bound ($0 < rh \leq rh_0$) the background error of relative humidity is reduced. This is reflected by the Jakobian $d\,rh/d\,gh$.

This behavior is sketched in Figures 8.11 and 8.12.

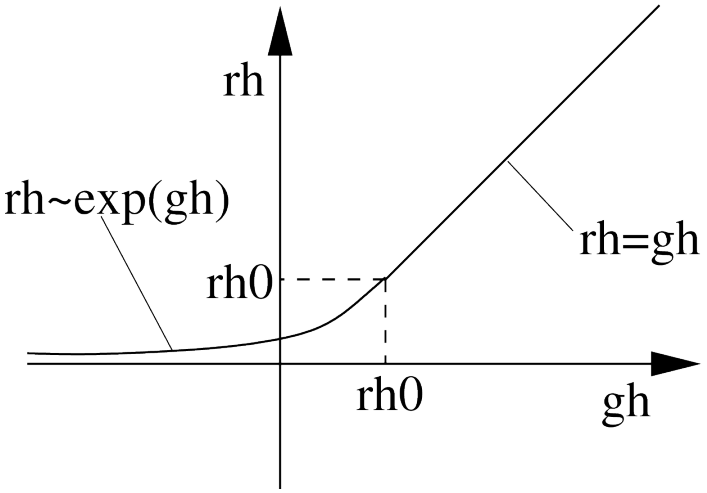


Figure 8.11: Generalised humidity variable gh

Here the transformation to generalised humidity is mainly performed for technical reasons, i.e. in order to ensure that analysed humidity remains positive. No attempt has been made so far to adapt parameters intentionally in order to match a given distribution of background errors.

A similar mechanism may be activated to prevent super-saturation ($rh > 1$) as well by a respective re-definition of generalised humidity in the vicinity of $h \approx 100\%$. In case of the exponential dependence of relative humidity on generalised humidity rh_0 is not allowed to become zero because gh would become $-\infty$. For this reason the first guess relative humidity is constrained to be larger than a small positive threshold value rh_0. Further details can be found in the user guide 16.23.

8.3.4 Cloud water and ice

Cloud liquid water and ice content (as well as prognostic rain and snow) is not assimilated in the 3dVar. Merely cloud water content is forced to zero if analysed relative humidity falls below 90%.

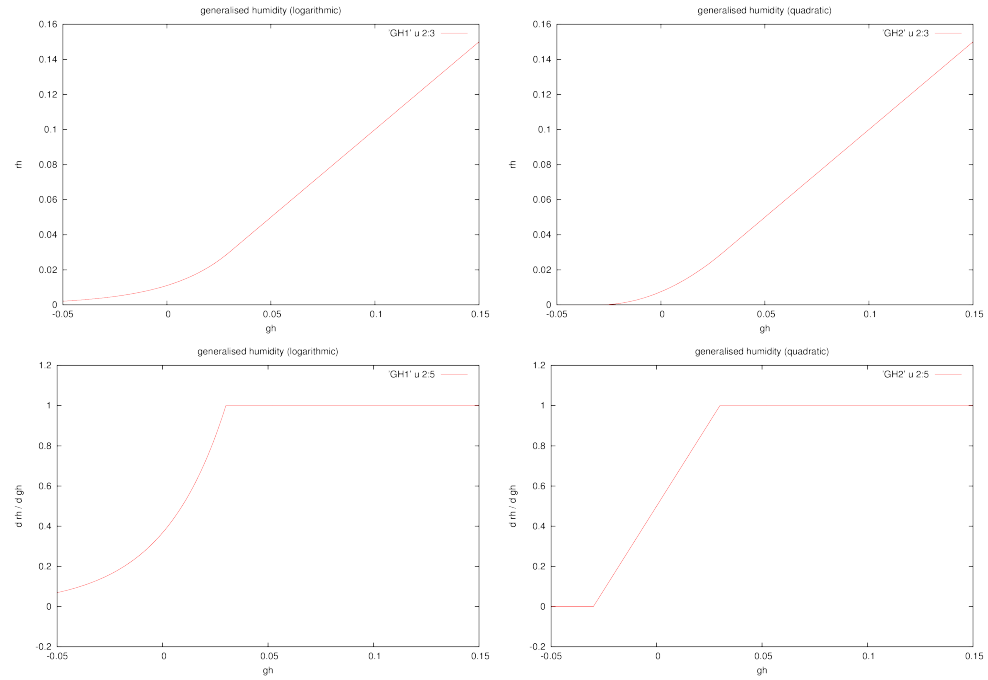


Figure 8.12: Relative humidity rh as a function of generalised humitity gh for exponential (top left) and quadratic formulation (top right). The respective Jakobians $d\,rh/d\,gh$ are shown below.

8.4 Preprocessing Module (satpp)

Level	Description
0	Raw Data
1	Data extracted by instrument, at full instrument pixel resolution, with Earth-location and calibration information
1a	Instrument counts with ancillary information
1b	Instrument counts with quality control and with Earth-location and calibration information appended but not applied
1c	Brightness temperatures (IR) or reflectance factor (VIS) of instrument pixels with Earth-location and calibration information
1d	Same as level 1c, with cloud flag (only for sounding data)
2	Geophysical value (temperature, humidity, radiative flux ...) at instrument pixel resolution

Figure 8.13: WMO Description of Levels for Satellite Data

DWD has developed a tool for *preprocessing of satellite data* (satpp). The main processing steps of this module are:

1. Data files are converted from BUFR to NetCDF.
2. The NetCDF data files with Level 1b data are red and *converted* to Level 1d data, compare Table 8.13.
3. Quality checks are carried out on the data sets, including the test of *cut-off times*, *scanline status*, *Field of View (FOV) and Calibration tests*, and diverse instrument specific tests.
4. Erroneous data are *corrected* as far as possible.
5. Mapping procedures are carried out to map datasets onto each other.
6. The results of the preprosessing is saved in a common format in NetCDF.

Chapter 9

Ensemble Kalman Filter (EnKF/LETKF)

In Chapter 7 the 3dVar algorithm was introduced with the goal to determine the state of the atmosphere with the highest probability, using a static error covariance matrix $\mathbf{B}$ (Section 7.2).

In this Chapter, the EnKF/LETKF is introduced. EnKF-type filters use an ensemble of atmospheric realizations to sample-estimate and predict the background error covariance matrix $\mathbf{P}^b$. This matrix is fully flow-dependent which is an advantage over the static $\mathbf{B}$, and it also contains correlations between model variables that do not need to be prescribed analytically (e.g. the geostrophic wind balance contained in $\mathbf{B}$). Therefore the EnKF-method is most useful where such balance relations cannot be formulated – which is the case in a nonhydrostatic local model such as COSMO-DE. Here the EnKF is used as the core of the COSMO-KENDA-System, and also for the global LETKF (GME/ ICON).

The first type of EnKF for atmospheric applications was proposed by [10]. It was further worked out by [14] that ones needs to use perturbed observations to generate an analysis ensemble in this EnKF type. The second type of EnKF algorithms are the *deterministic filters* that typically use a *matrix square root* to generate the analysis ensemble.

As a natural preconditioning, $\mathbf{P}^b$ can be transformed into an ensemble space (instead of the model phase space) – in this case, one usually speaks of an *Ensemble Transform Kalman Filter* (ETKF, [3]). To get any of these ensemble type Kalman Filter working with a small ensemble size, covariance localization needs to be applied – which coins the term LETKF.

The deterministic and transformed EnKF-algorithm in the DWD-DA-system is based on the Local Ensemble Transform Kalman Filter (LETKF) formulated in [16]. When the terms "EnKF" and "LETKF" are used throughout this documentation, they usually refer to the specific algorithm which is described in this Chapter.

The cycling filter recursion of the LETKF includes three steps (cf. Fig 9.1):

1. **Analysis step:** computation of the analysis from current observations and a background ensemble (Section 9.1)

2. **Resampling step:** resampling to produce the analysis ensemble (Section 9.1)

3. **Forecast step:** propagation of the analysis ensemble members to obtain a new forecast/background ensemble (Section 9.1.4)

Therefore, during ensemble data assimilation the forecast and the forecast ensemble are updated with measurements to produce an analysis and an analysis ensemble. The analysis ensemble is propagated to the next analysis time with the full nonlinear NWP model. Once this new forecast ensemble is computed, one cycle of the recursion is completed and the process repeats at the next analysis time.

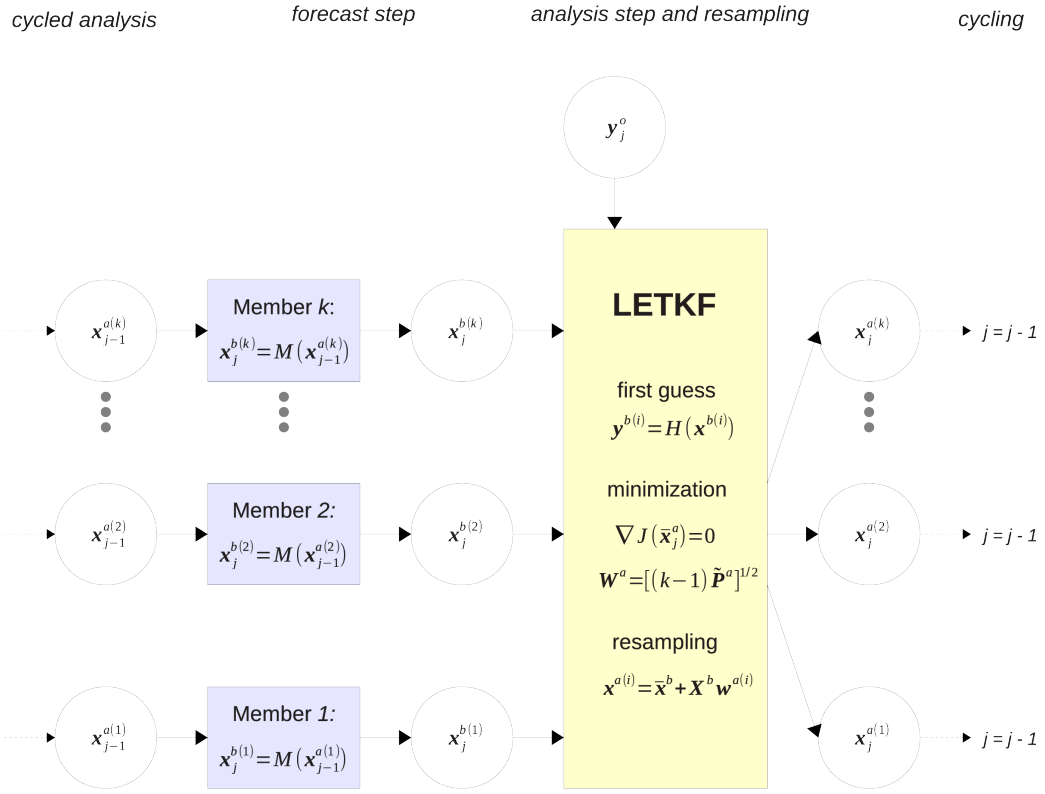


Figure 9.1: LETKF filter recursion. The index j counts the cycled analyses.

In ensemble Kalman filtering, the uncertainty is quantified only in the low-dimensional subspace spanned by the ensemble with $L-1$ degrees of freedom. The forecast/background ensemble $\{\mathbf{x}^{b(\ell)} : \ell = 1, \ldots, L\}$ typically consists of $L \geq 32$ members. At the times the analysis is calculated, the number of observations m is much larger than L, and the model has n degrees of freedom, so the ratio $L \ll m \ll n$ holds.

This limitation can be overcome by localizing the covariances and performing locally independent analyses (Section 9.2.1), so effectively more degrees of freedom are available with respect to the whole model domain. Because the dimensionality of atmospheric error covariances tends to be locally much lower than the possible degrees of freedom of the

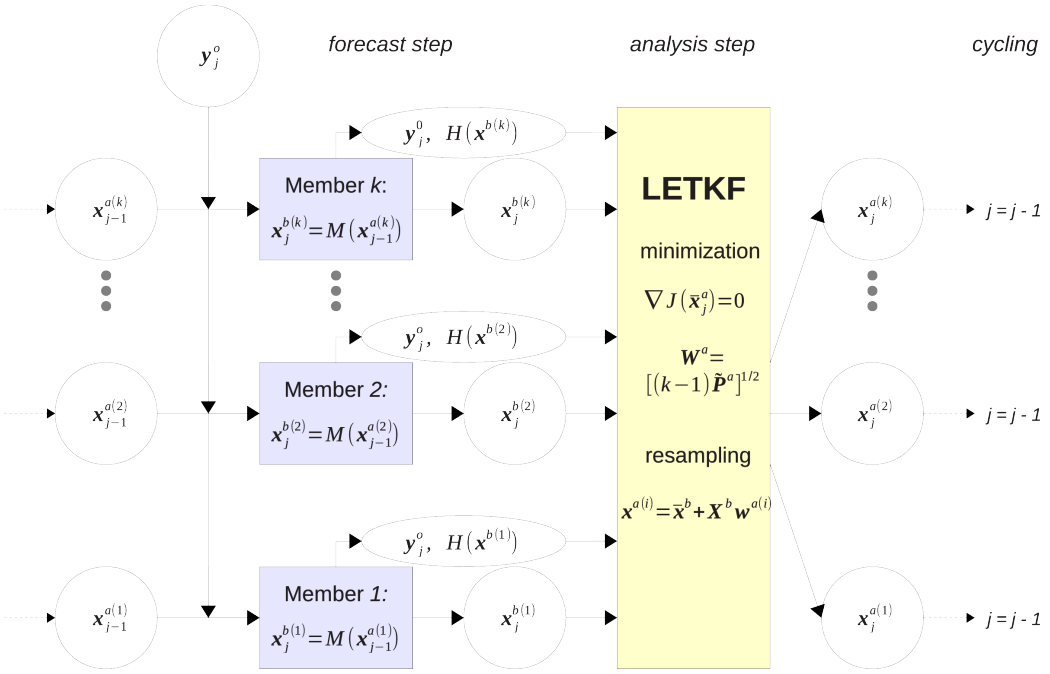


Figure 9.2: LETKF filter recursion for KENDA (using COSMO with internal operators)

model [21], the local $L-1$ degrees of freedom of the covariance localization can be sufficient to explore the possible states of the full model. Nevertheless, the background ensemble will usually predict too little variance, so it is necessary to apply inflation (Section 9.2.4).

Section 9.1 describes how the analysis is produced by the LETKF algorithm, including detailed description of each recursion step illustrated in Fig. 9.1.

Sections 9.2.1 and 9.2.4 describe the techniques of localization and inflation, that address the problem of calculations of background error covariances with small ensemble size. Section 9.2.6 describes balancing operations on the raw filter solution and reasoning behind their implementations.

9.1 Analysis of the Local Ensemble Transform Kalman Filter (LETKF)

The analysis should represent the state of the atmosphere with the highest probability, estimated by the background ensemble, weighted with its background error covariance, and by the observations, weighted with their observation error covariance. The entries of the observation error covariance matrix $\mathbf{R}$ are the variances and covariances of the measurement errors and errors of representativity. The background error covariance in ensemble methods is calculated from the ensemble.

Assuming a normal distribution of states[1], the most likely background state is simply

[1]This assumption holds rather well for wind or temperature, but not for example for precipitation or vertical wind speeds in convective storms.

given by the mean $\bar{\mathbf{x}}^b$ of the background ensemble

$$\bar{\mathbf{x}}^b = L^{-1} \sum_{\ell=1}^{L} \mathbf{x}^{b(\ell)} \tag{9.1}$$

The background error covariance $\mathbf{P}^b$ of this multidimensional Gaussian is sampled by the ensemble:

$$\mathbf{P}^b = (L-1)^{-1} \sum_{\ell=1}^{L} [\mathbf{x}^{b(\ell)} - \bar{\mathbf{x}}^b][\mathbf{x}^{b(\ell)} - \bar{\mathbf{x}}^b]^T \tag{9.2}$$

(9.2) can be written as

$$\mathbf{P}^b = (L-1)^{-1} \mathbf{X}^b (\mathbf{X}^b)^T \tag{9.3}$$

where the background perturbation matrix $\mathbf{X}^b$ is a $n \times L$ matrix with columns of $\mathbf{x}^{b(\ell)} - \bar{\mathbf{x}}^b$ and spans the L-dimensional space S of the background ensemble perturbations. By definition, the sum of the columns of $\mathbf{X}^b$ is zero, so they are necessarily linearly dependent. Matrices $\mathbf{P}^b$ and $\mathbf{X}^b$ are thus at most of rank $L-1$.

9.1.1 The cost function in ensemble space

The quadratic cost function $J(\mathbf{x})$ is minimized for the analysis $\mathbf{x} = \bar{\mathbf{x}}^a$:

$$J(\mathbf{x}) = [\mathbf{x} - \bar{\mathbf{x}}^b]^T (\mathbf{P}^b)^{-1} [\mathbf{x} - \bar{\mathbf{x}}^b] + [\mathbf{y}^o - H(\mathbf{x})]^T \mathbf{R}^{-1} [\mathbf{y}^o - H(\mathbf{x})] \tag{9.4}$$

As $\mathbf{P}^b$ is only of rank $L-1$, its inverse is well defined in only a $L-1$ subspace of the n-dimensional model space. The natural preconditioning for the minimization of (9.4) involves a change of variables from state $\mathbf{x}$ of size $n \times 1$ to a vector $\mathbf{w}$ of size $L \times 1$. Based on this reasoning, we regard $\mathbf{X}^b$ as a linear transformation from a L-dimensional space $\tilde{S}$ onto a L-dimensional space S and perform the analysis for a L-dimensional vector $\mathbf{w}$ in $\tilde{S}$. The corresponding analysis is given by $\mathbf{x} = \bar{\mathbf{x}}^b + \mathbf{X}^b \mathbf{w}$, so $\mathbf{X}^b \mathbf{w}$ describes a linear combination of ensemble perturbations that is added to the ensemble mean to obtain the solution $\mathbf{x}$.

If $\mathbf{w}$ is a Gaussian random vector with mean $\bar{\mathbf{w}} = \mathbf{0}$ and covariance $\tilde{\mathbf{P}}^f = (L-1)^{-1}\mathbf{I}$, then $\mathbf{x} = \bar{\mathbf{x}}^b + \mathbf{X}^b \mathbf{w}$ is Gaussian with mean $\bar{\mathbf{x}}^b$ (9.1) and covariance $\mathbf{P}^b$ (9.3). This motivates the cost function

$$\tilde{J}(\mathbf{w}) = (L-1)\mathbf{w}^T \mathbf{w} + [\mathbf{y}^o - H(\bar{\mathbf{x}}^b + \mathbf{X}^b \mathbf{w})]^T \mathbf{R}^{-1} [\mathbf{y}^o - H(\bar{\mathbf{x}}^b + \mathbf{X}^b \mathbf{w})] \tag{9.5}$$

in $\tilde{S}$ which relieves us from computing $\mathbf{P}^b$ explicitly. The solution $\mathbf{w} = \bar{\mathbf{w}}^a$ that minimizes (9.5) corresponds to $\bar{\mathbf{x}}^a = \bar{\mathbf{x}}^b + \mathbf{X}^b \bar{\mathbf{w}}^a$ that minimizes (9.4).

9.1.2 Observation operator in LETKF (linear approximation)

$H(\bar{\mathbf{x}}^b + \mathbf{X}^b \mathbf{w})$ can not be computed if H is nonlinear. Therefore, we need to linearly approximate H: First we compute an ensemble of model equivalents $\{\mathbf{y}^{b(\ell)} : \ell = 1, \ldots, L\}$

$$\mathbf{y}^{b(\ell)} = H(\mathbf{x}^{b(\ell)}) \tag{9.6}$$

With their mean $\bar{\mathbf{y}}^b$ and the $n \times L$ matrix $\mathbf{Y}^b$ (whose ℓ-th column is $\mathbf{y}^{b(\ell)} - \bar{\mathbf{y}}^b$) we make the linear approximation

$$H(\bar{\mathbf{x}} + \mathbf{X}^b\mathbf{w}) \approx \bar{\mathbf{y}}^b + \mathbf{Y}^b\mathbf{w} \tag{9.7}$$

This means that a deviation of one member from the mean in observation space (i.e. $\mathbf{y}^{b(\ell)} - \bar{\mathbf{y}}^b$ via $\mathbf{Y}^b$) is approximated to correspond linearly to the resulting deviation of the same member from the mean in model space (i.e. $\mathbf{x}^{b(\ell)} - \bar{\mathbf{x}}^b$ via $\mathbf{X}^b$)[2].

9.1.3 Analysis equation / Resampling for analysis ensemble

While still in $\tilde{S}$, we substitute (9.7) in (9.5) to gain the final cost function

$$\tilde{J}^*(\mathbf{w}) = (L-1)\mathbf{w}^T\mathbf{w} + [\mathbf{y}^o - \bar{\mathbf{y}}^b - \mathbf{Y}^b\mathbf{w}]^T\mathbf{R}^{-1}[\mathbf{y}^o - \bar{\mathbf{y}}^b - \mathbf{Y}^b\mathbf{w}] \tag{9.8}$$

with the analysis equations

$$\bar{\mathbf{w}}^a = \tilde{\mathbf{P}}^a(\mathbf{Y}^b)^T\mathbf{R}^{-1}(\mathbf{y}^o - \bar{\mathbf{y}}^b) \tag{9.9}$$

$$\tilde{\mathbf{P}}^a = [(L-1)\mathbf{I} + (\mathbf{Y}^b)^T\mathbf{R}^{-1}\mathbf{Y}^b)]^{-1} \tag{9.10}$$

and their retransformations [3] in model space

$$\bar{\mathbf{x}}^a = \bar{\mathbf{x}}^b + \mathbf{X}^b\bar{\mathbf{w}}^a \tag{9.11}$$

$$\mathbf{P}^a = \mathbf{X}^b\tilde{\mathbf{P}}^a(\mathbf{X}^b)^T \tag{9.12}$$

Now that the cost function is solved for $\bar{\mathbf{x}}^a$, we need to gain an analysis ensemble $\{\mathbf{x}^{a(\ell)} : \ell = 1, \ldots, L\}$ with mean $\bar{\mathbf{x}}^a$ and covariance $\mathbf{P}^a = (L-1)^{-1}\mathbf{X}^a(\mathbf{X}^a)^T$.

Therefore, we rewrite (9.10) as $\tilde{\mathbf{P}}^a = (L-1)^{-1}\mathbf{W}^a(\mathbf{W}^a)^T$ to perform the transformation $\mathbf{X}^a = \mathbf{X}^b\mathbf{W}^a$. The columns of $\mathbf{X}^a$ have the same properties as those of $\mathbf{X}^b$. We gain $\mathbf{W}^a$ by taking the symmetric square root of $\tilde{\mathbf{P}}^a$ as

$$\mathbf{W}^a = [(L-1)\tilde{\mathbf{P}}^a]^{1/2} \tag{9.13}$$

The use of matrix square root makes the LETKF a deterministic EnKF-algorithm (opposed to a stochastic EnKF with perturbed observations) and ensures that the column-sum of $\mathbf{X}^a$ is zero as desired, and that $\mathbf{W}^a$ depends continuosly on $\tilde{\mathbf{P}}^a$. Also, the mean-square distance between $\mathbf{I}$ and $\mathbf{W}^a$ is minimized, so the analysis perturbations are as close as possible to the background perturbations – a crucially important feature for nonlinear atmospheric models that are very sensitive to small changes.

The ℓ-th column of the computed $\mathbf{W}^a$ can be written as

$$W^{a(\ell)} = \mathbf{w}^{a(\ell)} - \bar{\mathbf{w}}^a, \ \ell = 1, \ldots, L \tag{9.14}$$

[2] For a linear H this approximation is exact, while for a nonlinear H (e.g. observations of radar reflectivity) the approximation only holds for small deviations and may cause unbalanced increments in model space if the analysis is fit rigorously towards the observations.

[3] not used in the algorithm, but mentioned for the sake of completeness

By adding $\bar{\mathbf{w}}^a$ to we the columns of $\mathbf{W}^a$ we therefore obtain $\{\mathbf{w}^{a(\ell)} : \ell = 1, \ldots, L\}$ and can then compute the final analysis ensemble members in model space as

$$\mathbf{x}^{a(\ell)} = \bar{\mathbf{x}}^b + \mathbf{X}^b \mathbf{w}^{a(\ell)}, \ \ell = 1, \ldots, L \tag{9.15}$$

The terms $\{\mathbf{w}^{a(\ell)} : \ell = 1, \ldots, L\}$ are called *analysis weight vectors*, the terms $\{\mathbf{X}^b \mathbf{w}^{a(\ell)} : \ell = 1, \ldots, L\}$ are called *ensemble increments*.

9.1.4 Forecast step

Cycling interval

As seen in Figure 9.1, the single members of $\{\mathbf{x}^{a(\ell)} : \ell = 1, \ldots, L\}$ are used as initial conditions for the forecast model and integrated forward through the forecast interval Δt_{fc} from the analysis time $k-1$ to the next assimilation time k where the forecasts represent the new background ensemble $\{\mathbf{x}^{b(\ell)} : \ell = 1, \ldots, L\}$:

$$\mathbf{x}_k^{b(\ell)} = M(\mathbf{x}_{k-1}^{a(\ell)}), \ \ell = 1, \ldots, L \ , \ k \in N \tag{9.16}$$

The cycling interval Δt_{fc} is a characteristic parameter of the EnKF and should be chosen long enough for the ensemble PDF to explore new dynamical developments of the system without diverging from the mean state too strongly. Δt_{fc} can also be referred to as *analysis interval* because at the end an analysis takes place, as *forecast interval* because the forecast model propagates the ensemble through that time-interval, or simply as the *assimilation interval* of the cycling.

Ensemble boundary conditions

In case of the local LETKF (COSMO-KENDA), ensemble initial (IC) and boundary conditions (BC) are essential in order to provide a range of possibilites (spread) for the synoptic scales and the mesoscale. A global ensemble of GME members, cycled in an LETKF system, provides these ensemble IC and BC (cf. Section 9.1.5).

Generally: ICON is probably not well-mentioned in the EnKF Chapter

9.1.5 Generation of initial ensemble (global model)

Generation of initial ensemble: Complete?

It is necessary to have an initial ensemble at time t_0 to spin up before the very first assimilation. In the DWD-DA-system, this is done with an ensemble of the global model GME. A 3dVar analysis state $\mathbf{x}^a_{3dVar}$ is used as the mean and the 3dVar-$\mathbf{B}$-matrix (Section 7.2) is used as the covariance of the initial random ensemble sample $\{\mathbf{x}_{t_0}^{init(\ell)} : \ell = 1, \ldots, L\}$:

$$\mathbf{x}_{t_0}^{init(\ell)} \in N(\mathbf{x}^a_{3dVar}, \mathbf{B}) \tag{9.17}$$

The matrix square root of $\mathbf{B}$, multiplied by a random noise vector $\mathbf{b}$, is added to the 3dVar analysis

$$\mathbf{x}_{t_0}^{init(\ell)} = \mathbf{x}^a_{3dVar} + f_{gen} \ \mathbf{B}^{1/2}\mathbf{b} \ , \ell = 1, \ldots, L \ , \ \mathbf{b} \in N(\mathbf{0}, \mathbf{I}) \tag{9.18}$$

where f_{gen} is a factor between 0 and 1 that controls the amplitude of the perturbations.

This initial ensemble is then consistent with the dynamics and can be spun up (integrated forward in time) to serve as a background-ensemble for the GME-LETKF assimilation, or as initial states and boundary conditions for the COSMO-KENDA ensemble.

9.1.6 Deterministic update

In the case of COSMO, ensemble forecasts of convective systems are produced. These need extensive postprocessing in order to be evaluated. As a simple alternative, it is possible to let a deterministic instance $\mathbf{x}^{det}$ of the model run parallelly to the EnKF-cycling and to update it via

$$\mathbf{x}^{a,det} = \mathbf{x}^{b,det} + \mathbf{K}[\mathbf{y}^o - H(\mathbf{x}^{b,det})] \tag{9.19}$$

with the *Kalman gain* $\mathbf{K}$:

$$\mathbf{K} = \mathbf{X}^b[(L-1)\mathbf{I} + \mathbf{Y}^{bT}\mathbf{R}^{-1}\mathbf{Y}^b]^{-1}\mathbf{Y}^{bT}\mathbf{R}^{-1}\mathbf{X}_b\left[(k-1)\mathbf{I} + \mathbf{Y}_b^T\mathbf{R}^{-1}\mathbf{Y}_b\right]^{-1}\mathbf{Y}_b^T\mathbf{R}^{-1}$$

The deterministic analysis is obtained on the same (coarse) grid as the ensemble is running on; the analysis increments are interpolated to the full model resolution of the deterministic run.

HL: It could make sense to reformulate the equations of this deterministic update in terms of the EnKF-update above. I could not infer this backwards from the code.

9.2 Background error covariance in EnKF

Contrary to the prescribed background error covariance of 3dVar (Section 7.2), the EnKF uses a background error covariance matrix $\mathbf{P}^b$ (9.2) that is sampled and propagated in time by the background ensemble.

The aspect of localization is discussed already here, although the LETKF localizes $\mathbf{R}$ and not $\mathbf{P}^b$ – but the sampling error, which is aimed to be reduced by localization, arises from the estimate of $\mathbf{P}^b$.

9.2.1 Local Analysis and Localization

As $L \ll m$, $\mathbf{P}^b$ is strongly rank deficient and contains extensive sampling error: Spatial correlations are not always trustworthy, especially with long distances between intermittent atmospheric phenomena such as convective cells.

To overcome this limitation, one possibility would be to localize $\mathbf{P}^b$ in phase space by setting spatially distant covariance-entries of $\mathbf{P}^b$ to zero, but this method is not performed here. Instead, local analyses are computed parallely by minimizing (9.8) for every model gridpoint of $\mathbf{x}$. If done this way, the model grid is also the *analysis grid*. As shown in Section 9.2.2, a coarser analysis grid can also be chosen for the minimization.

Performing the local analyses, only nearby observations are taken into account, so a localized observation error covariance matrix $\mathbf{R}_{loc}$ is used for every analysis grid point. After combining the local analyses (which were localized in observation space) back into the phase space of $\mathbf{x}$, the result is an effective localization in phase space, although the localization radius of $\mathbf{R}$ should be chosen shorter than for $\mathbf{P}^b$ – for a comprehensive comparison see [12].

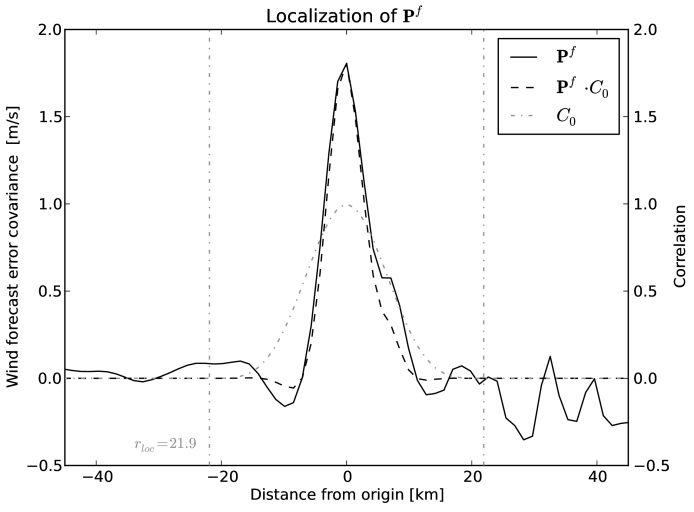


Figure 9.3: Ensemble-sampled $\mathbf{P}^b$, localized by C_0 (9.21).

Horizontal Localization

For every point $i = 1, \ldots, m$ of the analysis grid, a local ensemble of model equivalents $\{\mathbf{y}_i^{b(\ell)} : \ \ell = 1, \ldots, L \ , \ i = 1, \ldots, m\}$ with corresponding local observations $\mathbf{y}_i^o$ is formed: So, for every point i of the analysis grid, the corresponding entries in the inverse observation error covariance matrix $\mathbf{R}^{-1}$ are multiplied with a distance dependent correlation function C_0 (9.21) to gain a localized set of $\{\mathbf{R}_{loc,i}^{-1} : i = 1, \ldots, m\}$. This means that the single observations $\mathbf{y}_i^o$ will contribute to the local analysis at i with an *observation-weight* ≤ 1 determined by (9.21).

The following derivation is adapted from [11] in the formulation of [13].

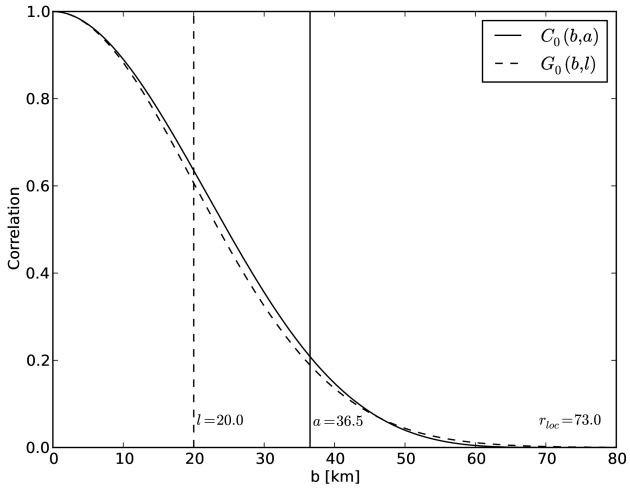


Figure 9.4: Function $C_0(b, a)$ of Gaspari Cohn 1999 compared to Gaussian $G_0(b, l)$. l_h was chosen as 20 km.

(9.21) is a polynomial approximation to a Gaussian

$$G_0(b, l) = \exp\left(\frac{-b^2}{2l^2}\right) \tag{9.20}$$

where b is the spatial distance between the single analysis grid point and the location of one single observation. l is the *length scale* where $G_0 = \exp(-\frac{1}{2}) \approx 0.61$. From hereon, the term *length scale* always refers to the "Gaussian" length scale l. G_0 is approximated by

$$C_0(b, a) = \begin{cases} -\frac{1}{4}(\frac{b}{a})^5 + \frac{1}{2}(\frac{b}{a})^4 + \frac{5}{8}(\frac{b}{a})^3 - \frac{5}{3}(\frac{b}{a})^2 + 1 & \text{if } 0 \leq b \leq a, \\ \frac{1}{12}(\frac{b}{a})^5 - \frac{1}{2}(\frac{b}{a})^4 + \frac{5}{8}(\frac{b}{a})^3 + \frac{5}{3}(\frac{b}{a})^2 - 5(\frac{b}{a}) + 4 - \frac{2}{3}(\frac{b}{a})^{-1} & \text{if } a < b \leq 2a \\ 0 & \text{if } b > 2a. \end{cases} \tag{9.21}$$

(cf. Figure 9.4). For corresponding $C_0(b, a)$ and $G_0(b, l)$, the parameter a relates to the

length scale l as

$$a = l\sqrt{\frac{10}{3}} \tag{9.22}$$

The radius $b = 2a$ where (9.21) goes to zero is here referred to as *localization radius* r_{loc} (or *support radius of localization*). The entries of the reduced $\mathbf{R}^{-1}_{loc,i}$ that is used in one local analysis (9.8) at a grid point $i = 1, \ldots, m$ are given by

$$R^{-1}_{loc,ji} = R^{-1}_{jj} C_0(b_{ji}, a), \;\; j = 1, \ldots, m \tag{9.23}$$

where b_{ji} is the spatial distance between the analysis point i and observation point j.

The input-parameter l that results in $r_{loc} = 2l\sqrt{10/3}$ is typically chosen ample enough to overlap with neighboring analysis grid points in order to obtain a spatially smooth field of local analysis weights $\{w_i^{a(\ell)} : \ell = 1, \ldots, L; i = 1, \ldots, n\}$. Observations that are further away than r_{loc} are simply neglected, while the observation-weights of the contributing observations are very close to a Gaussian. The horizontal length scale is referred to as l_h and is typically specified in kilometers.

Example: $l_h = 20\,\text{km}$ is chosen and the observation is 30 km away from the analysis point. The horizontal observation-weight is given by $C_0(30\,\text{km}, 20\,\text{km} \cdot \sqrt{10/3}) \approx 0.36$ (cf. Figure 9.4).

The localization in observation space usually takes place both in the horizontal and in the vertical. In the latter case, the observation-weight C_0 is the product of the horizontal observation-weight $C_{0,hor}$ and the vertical observation-weight $C_{0,vert}$:

$$C_0 = C_{0,hor} C_{0,vert} \in [0, ..., 1] \tag{9.24}$$

Vertical Localization

In this implementation of the LETKF, the vertical coordinate of observations is their pressure p. Using the pressure-distance of an observation to the analysis point would result in irregular spacings above and beneath the reference. Therefore, the logarithm $\ln(p)$, which scales approximately linear with height, is used as the argument for (9.21).

The parameter a of (9.21) is derived from the specified vertical localization length scale l_v, both in the units of $\ln(p)$ (typically in the order of 0.3 to 0.5). For a specific Δp beneath the reference pressure p_{ref}, l_v is given by

$$l_v = \ln\left(\frac{p_{ref}}{p_{ref} - \Delta p}\right) \tag{9.25}$$

The pressure distance Δp beneath a specific p_{ref} with specified l_v is given by

$$\Delta p = p_{ref}[1 - \exp(l_v)] \tag{9.26}$$

Example: For an analysis point at $p_{ref} = p_{ana} = 500\,\text{hPa}$ and $l_v = 0.3$, $\Delta p = 175\,\text{hPa}$, so an observation at $p_{obs} = 675\,\text{hPa}$ is assigned a observation-weight of $C_{0,vert}(\ln(675) - \ln(500), 0.3 \cdot \sqrt{10/3}) \approx \exp(-1/2) \approx 0.6$

Adaptive Horizontal Localization

The ideal localization length scale should minimized the analysis error. Physically, it depends e.g. on the weather situation, the observation density and other non-constant parameters.

One simple adaptive method

@Hendrik: Was there a reference planned here?

is to keep the number of *effective observations* fixed, and vary the localization radius r_{loc}. The *effective number of observations* $N_{eff,i}$ at an analysis point $i = 1, \ldots, n$ is defined as the sum of the observation weights $C_{0,j}$, $j = 1, \ldots, m_i$ (9.21) that fall into the local subdomain of i:

$$N_{eff,i} = \sum_{j=1}^{m_i} C_{0,j}(b_{j,i}) \ , \ i = 1, \ldots, n \ ; \ b_{j,i} \leq r_{loc} \tag{9.27}$$

where r_{loc} is determined by the length scale l_h as in (9.22). Up to now, adaptive localization is only implemented in horizontal direction. In order to perform the adaptive localization, one has to a define minimum and maximum length scale l_h^{min} and l_h^{max}; within this range the actual length scale l_h is variied to achieve the prescribed number of *effective observations* N_{eff} within r_{loc}. As the LETKF with L ensemble members computes the update in an L-dimensional subspace, it makes no sense to use a number m of observations with $m \gg L$. Thus, the ideal number of effective observations in this implementation only depends on ensemble size and should be choosen as $N_{eff} = \alpha L$ with $\alpha \approx 1.5 - 3$.

9.2.2 Coarse Analysis Grid

In the raw LETKF algorithm, the cost function (9.8) is minimized for every single model gridpoint of $\mathbf{x}$. As the optimal linear combination of members indicated by the field of $\{\mathbf{w}^{a(\ell)} : \ell = 1, \ldots, L\}$ usually varies smoothly between spatial locations of the domain, the field of the local analysis weight vectors $\{w_i^{a(\ell)} : \ell = 1, \ldots, L; i = 1, \ldots, n\}$ can be computed on a coarser analysis grid $\mathbf{x}^{ana}_{coarse}$ as $\{w_i^{a(\ell)} : \ell = 1, \ldots, L; i = 1, \ldots, n^{ana}_{coarse}\}$ and then be spatially interpolated onto the fine model grid (see [23]).

Horizontally coarse

COSMO has a horizontally regular grid. The coarse analysis grid is also horizontally regular and coarsed by a prescribed reduction factor $f_{hor,r}$. The computational effort for the analysis is therefore reduced by a factor of $f^2_{hor,r}$.

The GME has an icosahedral diamond-grid that covers the globe. The number n_i of diamonds (c.f. GME-documentation) to be used in the analysis can be chosen as the parameter for the horizontally coarse grid.

HL: What about ICON?

Vertically coarse

The number of vertical levels in the reduced grid is given by n_{zr}. The structure of the vertical coordinates of the reduced grid depends on the logarithm of the pressure. It is constructed as follows:

The maximum pressure p_{max} is given by $p_{max}(\bar{\mathbf{x}}^b)$ on the lowest full model level. If $p_{max}(\bar{\mathbf{x}}^b)$ is smaller than $p_{max}^{const} = 1050\,\text{hPa}$, $p_{max} = p_{max}^{const}$ is chosen.

The minimum pressure p_{min} is given by $p_{min}(\bar{\mathbf{x}}^b)$ on the highest full model level. If $p_{min}(\bar{\mathbf{x}}^b)$ is larger than $p_{min}^{const} = 40\,\text{hPa}$, $p_{min} = p_{min}^{const}$ is chosen.

The vertical localization length scale can be specified for the model top l_v^{top} and surface l_v^{surf} in the scale of $\delta[\ln(p)]$.

If $l_v^{surf} = l_v^{top}$, the pressure levels of the reduced grid are chosen to be linear in $\ln(p)$ by the recursion

$$p_0 = p_{min} \tag{9.28}$$

$$p_{i+1} = \ln(p_{min}) + i * \delta[\ln(p)] \tag{9.29}$$

with $\delta[\ln(p)] = \{[\ln(p_{max}) - \ln(p_{min})]/(n_{zr} - 1)\}$ here.

Usually it makes sense to choose $l_v^{surf} < l_v^{top}$ in order to localize on finer scales near the surface. Then, the pressure levels of the reduced grid are chosen correspondingly to a l_v that varies linearly in $\ln(p)$. The recursion is described in the subroutine `set_lv` in `mo_t_enkf`.

HL: How to handle references to such subroutines? Write the full recursion as equations here?

9.2.3 Updated and Non-Updated variables

In the analysis update (9.15) the analysis increments $\{\mathbf{X}^b\mathbf{w}^{a(\ell)} : \ell = 1, \ldots, L\}$ are added to the background mean $\bar{\mathbf{x}}^b$ to obtain the analysis members $\{\mathbf{x}^{a(\ell)} : \ell = 1, \ldots, L\}$.

This updates those prognostic model variables (Section 5.2) which should be affected by the observation-derived analysis. Constant fields, e.g. surface parameters, are not updated.

If the model variables of $\bar{\mathbf{x}}^a$ are updated independently of the observation types (and operators) that are used, some variables may be updated only through correlations given by the ensemble perturbations contained in $\mathbf{P}^b$. This is the case if these variables are not mapped in any observation operator $H(\mathbf{x})$ into observation space to be compared with actual observations.

To be safe, specific variables of $\mathbf{x}$ can be excluded from the update, e.g. the Rain, Snow and Graupel mixing ratios q_r, q_s, q_g, if they are have no observed counterpart (e.g. radar reflectivity of precipitation).

9.2.4 (Adaptive) Covariance Inflation

The flow-dependent background error covariance $\mathbf{P}^b$ is estimated from the ensemble perturbations. It usually underestimates the true background error, partly due to the limited

number of ensemble members and partly due to the presence of model errors. This lack of spread causes the analysis scheme to overestimate the influence of the background ensemble while the influence of the observations is possibly underestimated. An inferior analysis estimate is the consequence, and during cycled assimilation the background ensemble might eventually diverge from the true state.

Multiplicative inflation

Multiplicative covariance inflation [1] is a simple technique to deal with this problem. A *covariance inflation factor* $\rho > 1$ can be chosen by which the analysis perturbation of $\mathbf{W}^a$ are explicitly multiplied

$$\mathbf{W}^a_{infl} = \rho \mathbf{W}^a \tag{9.30}$$

before adding $\bar{\mathbf{w}}^a$ to $\mathbf{W}^a_{infl}$ in order to obtain $\{\mathbf{w}^{a(\ell)}_{infl} : \ell = 1, \ldots, L\}$ through (9.14) which is then used for the update of the member states in model space as in (9.15). This method enhances the spread of the analysis ensemble posterior to the computation of the analysis.

As an alternative to (9.30), one can apply ρ in the computation of $\tilde{\mathbf{P}}^a$. (9.10) then changes to

$$\tilde{\mathbf{P}}^a_{infl} = \left[\frac{(L-1)}{\rho} \mathbf{I} + (\mathbf{Y}^b)^T \mathbf{R}^{-1} \mathbf{Y}^b) \right]^{-1} \tag{9.31}$$

This method enhances the spread of the background ensemble (contained in $\mathbf{P}^b$) prior to the computation of the analysis.

A typical value could be $\rho = 1.05$, but tuning for an optimal value is inevitable, costly and mostly impossible due to the changing number of available observations.

Adaptive multiplicative inflation

The underestimation of variance might not be homogeneous throughout the forecast domain. Therefore it makes sense to estimate and apply a locally dependent $\rho(\mathbf{x})$ in model space. Following [6] and [19], an appropriate inflation factor $\tilde{\rho}$ can be estimated using statistics of increments in observation space:

$$\tilde{\rho} = \frac{\mathbf{d}^T_{o-b} \mathbf{d}_{o-b} - Tr(\mathbf{R})}{Tr(\mathbf{H}\mathbf{P}^b\mathbf{H})} \tag{9.32}$$

This equation computes a scalar $\tilde{\rho}$-estimate for the whole domain. In the following derivation, it is used as a template for a local estimate $\tilde{\rho}(\mathbf{x}^{ana}_{coarse})$ on the coarse analysis grid of the present LETKF-implementation.

Let the vector $\mathbf{d}^2_{o-b}$ denote the squared deviation of the mean of the members' first guesses $\bar{\mathbf{y}}^b$ from the observations $\mathbf{y}^o$ with

$$d^2_{o-b,j} = \left(\frac{y^o_j - \bar{y}^b_j}{\sigma_j} \right)^2, \; j = 1, \ldots, m \tag{9.33}$$

These terms are normalized by the attributed observation errors σ_j in order to get rid of the respective units (e.g. K or m/s) and the different orders of magnitude in the numerical values of different observation types.

Then, let the vector $\mathbf{p}^2$ denote the local traces of the background error covariance in observation space (analytically given by $\mathbf{HP}^b\mathbf{H}$, but here with the same linear approximation as in (9.7)):

$$p_j^2 = \frac{1}{L-1}\sum_{\ell=1}^{L}\left(\frac{y_j^{b(\ell)} - \bar{y}_j^b}{\sigma_j}\right)^2, \; j = 1, \ldots, m \tag{9.34}$$

The σ-normalization is performed as in (9.33). As $\mathbf{R}$ is taken to be diagonal, it can be expressed by a vector $\mathbf{r}^2$ with

$$r_j^2 = \left(\frac{\sigma_j}{\sigma_j}\right)^2 = 1, \; j = 1, \ldots, m \tag{9.35}$$

Now the mismatch between the specified $\mathbf{R}$ and the forecasted $\mathbf{P}^b$, which turns up in $\mathbf{d}_{o-b}^2$, can be expressed in observation space as

$$\tilde{\rho}(\mathbf{y}) = \frac{\mathbf{d}_{o-b}^2 - \mathbf{r}^2}{\mathbf{p}^2} \tag{9.36}$$

with the element-wise computation

$$\tilde{\rho}_j = \frac{d_{o-b,j}^2 - r_j^2}{p_j^2}, \; j = 1, \ldots, m \tag{9.37}$$

$\tilde{\rho}(\mathbf{y})$ is then the estimate for the local covariance inflation of every observation point $j \in \mathbf{y}$. As the local analyses are performed on a (coarse) analysis grid (Section 9.2.2), the estimated inflation needs to be known at every analysis grid point as $\tilde{\rho}(\mathbf{x}_{coarse}^{ana})$.

Therefore, all $\mathbf{d}_{o-b}^2$, $\mathbf{r}^2$ and $\mathbf{p}^2$ that fall into the vicinity of one analysis grid point are summed up, observation-weighted by their respective localized contribution $C_{0_{ji}}$ (cf. (9.24)):

$$d_{o-b,i}^2 = \sum_{j=1}^{m} d_{o-b,j}^2 C_{0_{ji}}, \; i = 1, \ldots, n_{coarse}^{ana} \tag{9.38}$$

$$r_i^2 = \sum_{j=1}^{m} 1 \cdot C_{0_{ji}}, \; i = 1, \ldots, n_{coarse}^{ana} \tag{9.39}$$

$$p_i^2 = \sum_{j=1}^{m} p_j^2 C_{0_{ji}}, \; i = 1, \ldots, n_{coarse}^{ana} \tag{9.40}$$

The final estimate $\tilde{\rho}(\mathbf{x}_{coarse}^{ana})$ can now be computed as

$$\tilde{\rho}_i = \frac{d_{o-b,i}^2 - r_i^2}{p_i^2}, \; i = 1, \ldots, n_{coarse}^{ana} \tag{9.41}$$

which can be traced back to (9.32).

This estimation is performed for every analysis grid point and then interpolated onto the model grid to obtain a $\tilde{\rho}(\mathbf{x})$, before computing the analysis with an inflated $\mathbf{P}^a_{infl}$ as in (9.31) or to apply the update using the inflated $\mathbf{W}^a_{infl}$ as in (9.30).

Because sampling error is an issue with the small number of members L that is used here, the ρ-field of the covariance inflation should be temporally smoothed to avoid discontinuities. A weighting factor α is chosen to combine the estimated $\tilde{\rho}_k$ at the current analysis time k and the ρ_{k-1} that was used in the previous analysis cycle to get the ρ_k to be applied at the current analysis time k:

$$\rho_k = \alpha\tilde{\rho}_k + (1-\alpha)\rho_{k-1} \tag{9.42}$$

Additive Inflation

Additive Inflation: Complete?

In the present implementation of the LETKF, the analysis computation accounts for background and observation errors, but not for the inherent model error. In the case of the global LETKF, it is possible to use the background error of the 3dVar-$\mathbf{B}$-matrix (Section 7.2) as a proxy for the model error: After or before the generation of the analysis ensemble $\{\mathbf{x}^{a(\ell)} : \ell = 1, \ldots, L\}$, random noise with mean $\mathbf{0}$ and covariance $\mathbf{B}$ is added to the members. This is similar to the generation of an initial global ensemble (Section 9.1.5).

$$\mathbf{x}^\ell = \mathbf{x}^\ell + f_{add}\mathbf{B}^{1/2}\mathbf{b}\ , \ell = 1, \ldots, L\ ,\ \mathbf{b} \in N(\mathbf{0}, \mathbf{I}) \tag{9.43}$$

where f_{add} is a factor between 0 and 1 that controls the amplitude of the added perturbations.

9.2.5 Multistep analysis

Multistep analysis is also known as *successive, serial* or *batch assimilation*

@Hendrik: Was there a reference planned here?

. It means that the observations are assimilated in two or more steps, using a subset of the observations in each of these steps. The analysis obtained in the previous step serves as the first guess for the next step. If also the $\mathbf{P}^b$ matrix is updated and no localization is applied, it can be shown that the last analysis is identical to an analysis obtained by assimilating all observations in one step simultaneously.

So far, *multistep analysis* was used for computational or algorithmic reasons. In the COSMO-LETKF, there are various motivations to use *multistep analysis*:

- to account for local / nonlocal observations (e.g. Radiances)
- in relation with *adaptive localization*: to account for different observation densities (conventional / radar). When using *adaptive localization* and assimilating different kinds of observations with different observation densities simultaneously, an adaptive localization method would choose a very small localization radius only slaved to the

dense parts of the observation network. This would result in missing information from the coarser observation types at many gridpoints.

- to respect different observed scales (synoptic / convective scale) and observation errors

It appears senseful to assimilate nonlocal and sparse observations ahead of local and dense observations. A final strategy for different observation types has yet to be developed. For example, nonlocal observations such as radiances could be assimilated in a second step without vertical localization.

The *multistep analysis* is technically implemented in COSMO-LETKF and will be tested with radar data.

Theoretical background

The proof of the following derivations can be found in

@Hendrik: Reference to inverse-problems paper?

For the standard Kalman Filter with analysis $\mathbf{x}^a$, and the multistep Kalman Filter with analysis $\mathbf{x}^{a,\xi}$ and multiple steps $\xi = 1, ..., q$, we have for the final analysis q

$$\mathbf{x}^a = \mathbf{x}^{a,q} \quad \text{and} \quad \mathbf{B}^a = \mathbf{B}^{a,q}.$$

For the covariance matrices $\mathbf{P}^a := (\mathbf{X}^a)(\mathbf{X}^a)^T$ generated by the classical EnKF and the covariance matrix $\mathbf{P}^{a,q} := (\mathbf{X}^{a,q})(\mathbf{X}^{a,q})^T$ generated by the multi-step EnKF we have

$$\mathbf{P}^a = \mathbf{P}^{a,q}.$$

We define

$$\mathbf{A}_1 := (\mathbf{X}^b)^T(\mathbf{H}^{(1)})^T(\mathbf{R}^{(1)})^{-1}\mathbf{H}^{(1)}\mathbf{X}^{(b)}$$

and

$$\mathbf{A}_2 := (\mathbf{X}^{(b)})^T(\mathbf{H}^{2)})^T(\mathbf{R}^{(2)})^{-1}\mathbf{H}^{(2)}\mathbf{X}^{(b)}.$$

Assume that the observation operators $\mathbf{H}^{(1)}$ and $\mathbf{H}^{(2)}$ for two different sets of measurements satisfy $\mathbf{A}_1\mathbf{A}_2 = \mathbf{A}_2\mathbf{A}_1$. Then the analysis ensemble generated by the multi-step EnKF with square root filter is identical to the analysis ensemble generated by the classical EnKF.

Note: This assumption only holds for linear observation operators - which is not the general case in COSMO, especially with satellite radiances and radar data (9.7). A sensible strategy has to be developed for the operational system.

9.2.6 Physical corrections and balances

The forecast state of a model run that is spun up sufficiently can be assumed to be dynamically consistent, meaning that no unphysical phenomena are contained as far as the model physics are concerned.

In contrast to the 3DVAR analysis where stability and balance are prescribed (Section 7.2), this dynamical consistency of the forecast is altered in an EnKF system: the EnKF-analysis update is based on a flow dependent error covariance that is represented by a finite

ensemble size. A linear combination of dynamically consistent states is not necessarily a dynamically consistent and balanced state[12], especially in the case on the nonlinear atmospheric dynamics.

To prevent such inconsistencies, the analysis states are adjusted posterior to the minimization of (9.8) by different means:

Positive humidity

Mixing ratios of atmospheric water mass (vapor q, cloud droplets q_{cl}, cloud ice particles q_{ci}, rain q_r, snow q_s and graupel q_g) are necessarily positive. As there is no constraint on positivity of these mass fields in the computation of $\{\mathbf{x}^{a(\ell)} : \ell = 1, \ldots, L\}$, these fields have to be postprocessed in order to provide physically senseful initial conditions for the forecast model.

For of every member of $\{\mathbf{x}^{a(\ell)} : \ell = 1, \ldots, L\}$ the following checks are performed subsequently:

$$\begin{aligned}
\mathbf{q}_{cl}^{a(\ell)} &= \begin{cases} \mathbf{q}_{cl}^{a(\ell)}, & \text{where } \mathbf{q}_{ci}^{a(\ell)} \geq 0 \\ \mathbf{q}_{cl}^{a(\ell)} + \mathbf{q}_{ci}^{a(\ell)}, & \text{where } \mathbf{q}_{ci}^{a(\ell)} < 0 \end{cases} \\
\mathbf{q}_{ci}^{a(\ell)} &= \begin{cases} \mathbf{q}_{ci}^{a(\ell)}, & \text{where } \mathbf{q}_{ci}^{a(\ell)} \geq 0 \\ 0, & \text{where } \mathbf{q}_{ci}^{a(\ell)} < 0 \end{cases} \\
\mathbf{q}^{a(\ell)} &= \begin{cases} \mathbf{q}^{a(\ell)}, & \text{where } \mathbf{q}_{cl}^{a(\ell)} \geq 0 \\ \mathbf{q}^{a(\ell)} + \mathbf{q}_{cl}^{a(\ell)}, & \text{where } \mathbf{q}_{cl}^{a(\ell)} < 0 \end{cases} \\
\mathbf{q}_{cl}^{a(\ell)} &= \begin{cases} \mathbf{q}_{cl}^{a(\ell)}, & \text{where } \mathbf{q}_{ci}^{a(\ell)} \geq 0 \\ 0, & \text{where } \mathbf{q}_{cl}^{a(\ell)} < 0 \end{cases}
\end{aligned} \tag{9.44}$$

This means that negative analysis-values of cloud particles are evaporated "instantly" and turned into their predecessor in the hierarchy water vapor – cloud water – cloud ice.

After that, the following positivities is set:

$$\begin{aligned}
\mathbf{q}_g^{a(\ell)} &= \mathbf{0}, \quad \text{where } \mathbf{q}_g^{a(\ell)} < \mathbf{0} \\
\mathbf{q}_s^{a(\ell)} &= \mathbf{0}, \quad \text{where } \mathbf{q}_s^{a(\ell)} < \mathbf{0} \\
\mathbf{q}_r^{a(\ell)} &= \mathbf{0}, \quad \text{where } \mathbf{q}_r^{a(\ell)} < \mathbf{0} \\
\mathbf{q}^{a(\ell)} &= \mathbf{0}, \quad \text{where } \mathbf{q}^{a(\ell)} < \mathbf{0}
\end{aligned} \tag{9.45}$$

While (9.44) conserves the mass sum of q, q_{cl} and q_{ci}, (9.45) actually adds water mass to $\{\mathbf{x}^{a(\ell)} : \ell = 1, \ldots, L\}$. Nevertheless, this constraint is necessary in order to obtain correct initial conditions.

Saturation adjustment

In the analysis, oversaturation may occur and disturb the model physics. To compensate, the saturation water vapor pressure e^a over water is computed at points using the Magnus

formula. All the following computations are performed only where it is warmer than $-40°$ C (i.e. $T > 233.15$ K).

$$e^a(T) = 610.78 \ \exp\left(\frac{17.27\ (T - 273.15)}{T - 35.86}\right) \text{Pa} \tag{9.46}$$

From e^a, the saturation mixing ratio q^a_{sat} is derived:

$$q^a_{sat}(T) = \frac{\frac{R}{R_d} \cdot e^a(T)}{p^a - \left(1 - \frac{R}{R_d}\right) \cdot e^a(T)} \tag{9.47}$$

To compensate for possible subsaturation, first all the cloud water is added towards the water vapor in all members independently:

$$\text{where } (T > 233.15 \text{ K}) \ \begin{cases} \mathbf{q}^{a(\ell)} & = \mathbf{q}^{a(\ell)} + \mathbf{q}^{a(\ell)}_{cl} \\ \mathbf{q}^{a(\ell)}_{cl} & = \mathbf{0} \end{cases} \tag{9.48}$$

Following this, occuring supersaturations are compensated:

$$\text{where } (T > 233.15 \text{ K}) \text{ and } (\mathbf{q}^{a(\ell)} > \mathbf{q}^{a(\ell)}_{sat}) \begin{cases} \mathbf{q}^{a(\ell)}_{cl} & = \mathbf{q}^{a(\ell)} - \mathbf{q}^{a(\ell)}_{sat} \\ \mathbf{q}^{a(\ell)} & = \mathbf{q}^{a(\ell)}_{sat} \end{cases} \tag{9.49}$$

This saturation adjustement conserves the mass of q and q_{cl}.

The following 3 paragraph should be moved into the implementation part and then be deleted here.

There are two subroutines which influence the humidity variables, `apply_hum_bounds` and `sat_ad`. The first one corrects negative values of the humidity variables, the second one performs corrections of q and qcl in the case of sub/supersaturation. In the following we briefly describe both subroutines.

`apply_hum_bounds` first checks whether negative values of qci occur. If this is the case, qcl is reduced by this value, and qci is set to zero. The same procedure is now repeated for qcl (subtracting negative values from q, setting qcl to zero if necessary). In the last step, negative values of q, qr, qs, qh are set to zero. rh is limited to values between 0 and 1.1.

`sat_ad` performs a very simple saturation adjustement. This is done by first computing the saturation humidity qv_{sat}. In a second step, q and qcl are adjusted according to qv_{sat}: in the case of subsaturation e.g. qcl is added to q until qcl is zero or the saturation humidity is reached.

Hydrostatic balancing of pressure increments (COSMO)

In the case of COSMO, the pressure is a prognostic variable and formulated as $p = p_0 + p'$, so it is represented by the pressure perturbation p' from a hydrostatic reference pressure p_0. COSMO is a non-hydrostatic model and is used to forecast convective systems associated with vertical movement of air parcels. The abundance of inertia gravity waves (IGW) is therefore necessary and desired, but IGW may be spuriously triggered by unbalanced

analysis increments of $\Delta p' = p'^a - p'^b$ and then propagate, interfere and deteriorate the forecast.

To prevent this, $\Delta p'$ is hydrostatically balanced before it is added to the background: Assume a non-hydrostatic and dynamically consistent background state containing an updraft of a convective system with positive w. This is dynamically associated to a larger p' above the updraft than in the horizontal environment, and a smaller p' below the updraft. Now assume observations that cause an amplification of the vertical wind speed w through analysis increments Δw. Usually, w is not directly observed, so Δw is determined through $\mathbf{P}^b$ (9.3), as is $\Delta p'$. Because the dynamical consistency of Δw and $\Delta p'$ is not guaranteed, it is safer to compute hydrostatically balanced increments $\Delta p'_{hydbal}$ (9.52) in order to prevent some of the inevitable spurious IGW.

The following derivation is adapted from the COSMO documentation *Part III: Data Assimilation*: The hydrostatic equation between background and analysis in finite differences of the COSMO formulation on model levels (counted downwards by index k) reads

$$\frac{p'^a_{k+1} - p'^a_k}{\frac{1}{2}\left(dp_{0_{k+1}} + dp_{0_k}\right)} = \frac{dp_{0_k}}{dp_{0_{k+1}} + dp_{0_k}} \left\{ \frac{T_{0_{k+1}}}{T^b_{k+1} p_{0_{k+1}}} p'^a_{k+1} - \frac{T^a_{k+1} - T_{0_{k+1}}}{T^a_{k+1}} - q^{a,virt}_{k+1} \right\} + \frac{dp_{0_{k+1}}}{dp_{0_{k+1}} + dp_{0_k}} \left\{ \frac{T_{0_k}}{T^b_k p_{0_k}} p'^a_k - \frac{T^a_k - T_{0_k}}{T^b_k} - q^{a,virt}_k \right\} \quad (9.50)$$

where $q^{a,virt} = (R_v/R_d - 1) \cdot q^a - q^a_{cl} - q^a_r - q^a_s$, and T^b, T^a are the background and analysis temperature. $(R_v/R_d - 1)$ contains the gas constants of dry air and water vapor, q^a is the analysis water vapor mixing ratio, q^a_{cl}, q^a_r, q^a_s describe the mixing ratios of cloud water, rain and snow. dp_{0_k} describes the vertical variation of the reference pressure p_0 on the k-th model half level as

$$dp_{0_k} = (p_0)_{k+1/2} - (p_0)_{k-1/2}$$

A background buoyancy term B_k on level k is defined as

$$B_k = R \rho_{0_k} T^b_k \quad (9.51)$$

where ρ_{0_k} is the density of the COSMO reference atmosphere (in the Fortran code: `zfbuyot` $= B_k$ and `zfbuyob` $= B_{k+1}$).

The final pressure increment $\Delta p'_{hydbal}$ (hydrostatically balanced with respect to temperature T) is then computed upwards from the lowest model level (largest k) by the recursion

$$\Delta p'_{k,hydbal} = \left(2 + \frac{dp_{0_k}}{B_k}\right)^{-1} \cdot \left\{ \Delta p'_{k+1} \cdot \left(2 - \frac{dp_{0_{k+1}}}{B_{k+1}}\right) + \frac{T^a_k dp_{0_{k+1}}}{T^b_k} + dp_{0_k} \Delta q^{a,virt}_{k+1} + \frac{T^a_{k+1} dp_{0_k}}{T^b_{k+1}} + dp_{0_{k+1}} \Delta q^{a,virt}_k \right\} \quad (9.52)$$

where $\Delta q^{a,virt} = (R_v/R_d - 1) \cdot \Delta q^a - \Delta q^a_{cl} - \Delta q^a_r - \Delta q^a_s$.

9.3 Observation error covariance

Section 9.2 already discussed the properties of the background error covariance $\mathbf{P}^b$, implicitly contained in the minimization of the background part of (9.8), and the localization of $\mathbf{R}$.

This section discusses the actual entries of the observation error covariance matrix $\mathbf{R}$. For simplicity, the present implementation of the LETKF assumes and uses a diagonal $\mathbf{R}$ whose entries are the variances $\{sigma^2_{obs_{jj}} \ : \ j = 1, \ldots, m\}$. This means that only the variance of the observation errors are taken into account and error correlations (contained in the off-diagonal covariances) are disregarded.

9.3.1 Error Models

Observation errors consist of three main partitions:

1. **Measurement error:** For in-situ measurements like temperature or pressure, this part is usually small and possible to determine, while for remote sensing it is more difficult.

2. **Representativity error:** A single observation may represent a scale that is smaller than the model resolution, e.g. a turbulent wind gust of $\mathcal{O}$(10-100 m) observed by a radio sonde compared to the model resolution of $\mathcal{O}$(1 km).

3. **Operator error:** Although not an error of the observation itself, the computations of the observation operator may be simplified or tainted by assumptions that make the model equivalent $H(\mathbf{x})$ differ from $\mathbf{y}^o$, even if $\mathbf{x}$ was perfect.

Here the error models of the observation operators of shortly discussed, first those that are applied in COSMO, then those of the operators contained in 3dVar which are also used for the VarEnKF (Chapter 10).

COSMO-Operators

When the observation reports are read in the by COSMO members (using the COSMO-module `data_obs_cdfin`), every single observation data point $\mathbf{y}_j$ is assigned a standard deviation σ_{obs} from an error table which depends on the type and height of the observation. Usually, the assumed observations errors are larger near the surface and the stratosphere than within the stratified mid-tropospheric levels.

GME/ICON

Missing: Error models/tables of GME/ICON

9.3.2 (Adaptive) R-correction

The abundance of observations is spatially inhomogeneous, so typically there are regions within the analysis model domain that have a more dense observation network. The

presence of many observations causes a large contribution of the observational part of (9.8) in the computation of the analysis solution $\bar{\mathbf{x}}^a$. The background ensemble gets little weight in such a situation, and the ensemble spread may be overly diminished, while $\bar{\mathbf{x}}^a$ gets pulled fiercely towards the observed values.

By transforming the observational part of the costfunction into the ensemble space, it can be determined if the background ensemble perturbations are sufficient in order to span the possible states of the observations.

This is often not the case if there is a large number of observations or if the observation error in $\mathbf{R}$ is chosen to small. Therefore, the entries of the $\mathbf{R}$-matrix can be corrected (typically enlarged) to make the computation of the analysis ensemble consistent.

Incomplete: Description of (adaptive?) R-correction

Chapter 10

Hybrid Variational Ensemble Kalman Filter (VarEnKF)

VarEnKF Chapter: Complete?

The LETKF described in the previous section uses ensembles of short range forecasts and analyses in order to represent the uncertainty of the background and analysis in the cycled analysis system. The ability to provide a flow dependent estimate of the model forecast and to make use of it in the analysis system itself is the most important advantage of Ensemble data assimilation systems and the motivation to use them in operational weather forecasting.

A number of compromises have to be made regarding ensemble size, resolution, model balances, and consistency in localization of non-local observations in order to achieve a computationally efficient assimilation system. It is difficult to beat current deterministic variational analysis systems by pure ensemble Kalman filter formulations and in general they are not replaced but merely augmented by EnDA systems. Combined hybrid systems exchange information for instance by using the variance of ensemble forecast to adjust the variance of the background error covariance matrix of the variational scheme (but keeping the static correlations) or by adjusting the ensemble mean to the deterministic analysis.

It is possible to use the complete localized ensemble background error covariance matrix in a variational context as proposed by [4]. The static 3DVAR background error covariance matrix is replaced or augmented by the flow dependent ensemble B-matrix. Thereby the 3DVAR gains characteristics of a 4DVAR scheme, namely consideration of temporal background error covariances consistently with the model dynamics. This setup will be used in the global data assimilation system for ICON and is described in the following subsections.

10.1 Extension of the 3DVAR background error covariance matrix

The background error covariance matrix $\mathbf{P}^b$ used in the variational scheme is replaced by a combination of the static covariance matrix $\mathbf{P}^b_{NMC}$ as used formerly in the 3DVAR and

the ensemble covariance matrix $\mathbf{P}^b_{EnKF}$:

$$\mathbf{P}^b = \beta\,\mathbf{P}^b_{NMC} + \alpha\,\mathbf{P}^b_{EnKF} \tag{10.1}$$

Without localization $\mathbf{P}^b_{EnKF}$ is given by (9.3) and by using $\mathbf{x} = \bar{\mathbf{x}}^b + \mathbf{X}^b\mathbf{w}$ the cost function of the EnKF is given by (9.5) now depending on the k (ensemble size) dimensional vector of control variables $\mathbf{w}$. For this reason this augmentation of the static $\mathbf{P}^b_{NMC}$ with $\mathbf{P}^b_{EnKF}$ is also known as the method of additional control variables.

In the variational PSAS scheme the background error covariance matrix is given by operator representations which allow to calculate the product of the background error covariance matrix $\mathbf{P}^b$ with a vector. The representation of $\mathbf{P}^b_{NMC}$ was described in Section 7.3 and the representation of a localised $\mathbf{P}^b_{EnKF}$ is described in the subsequent Section 10.2.

10.2 Localization by the Schur product

The localization approach follows the original proposal by [15] to multiply the ensemble background covariances in physical space (i.e. at each model grid-point) element by element (Schur product $\circ$) with a localization matrix $\mathbf{C}$:

$$\mathbf{P}^b = \mathbf{C} \circ \mathbf{P}^b_{EnKF} \tag{10.2}$$

Considering (9.3) we write

$$\mathbf{P}^b = (L-1)^{-1}\mathbf{C} \circ [\mathbf{X}^b(\mathbf{X}^b)^T] \tag{10.3}$$

In the variational optimization algorithm we have to look for an efficient representation for multiplying the matrix $\mathbf{P}^b$ (10.3) with some vector $\mathbf{z}$. If we denote the spatial indices of the matrix with i and j, and the ensemble index with ℓ, elements of (10.2) are written as:

$$p^b_{ij} = (L-1)^{-1} \sum_{\ell=1}^{L} c_{ij}\; x_i^{b(\ell)}\, x_j^{b(\ell)} \tag{10.4}$$

We can write:

$$p^b_{ij} = (L-1)^{-1} \sum_{\ell} \sum_{i'} \delta_{ii'} \sum_{j'} \delta_{jj'} \sum_{\ell'} \delta_{\ell\ell'}\, c_{i'j'}\; x_i^{b(\ell)}\, x_j^{b(\ell')} \tag{10.5}$$

and re-arrange:

$$p^b_{ij} = (L-1)^{-1} \sum_{\ell} \sum_{i'} \sum_{\ell'} \sum_{j'} (x_i^{b(\ell)}\delta_{ii'})\,(c_{i'j'}\delta_{\ell\ell'})\,(x_j^{b(\ell')}\delta_{jj'}) \tag{10.6}$$

Finally (10.3) can be formally written as the product of 3 matrices:

$$\mathbf{P}^b = (L-1)^{-1}\hat{\mathbf{X}}^b\;\hat{\mathbf{C}}\;(\hat{\mathbf{X}}^b)^T \tag{10.7}$$

Equation (10.6) provides the recipe to multiply $\mathbf{P}^b$ with a vector $\mathbf{z}$ in model representation:

1. Multiply with $(\hat{\mathbf{X}}^b)^T$: The vector $\mathbf{z}$ is a gradient vector defined for each of the model variables analysed in the variational scheme (temperature, wind components, humidity) at each model grid-point. For each of the L ensemble members l the vector $\mathbf{z}$ is multiplied grid-point by grid-point and variable by variable with the deviation $[\mathbf{x}^{b(\ell)} - \bar{\mathbf{x}}^b]$. The results for all variables are summed up. The final result of the operation is a set of L gradient fields defined on the model grid.

2. Multiply with $\hat{\mathbf{C}}$: multiply each of the resulting L vectors with the localization matrix $\mathbf{C}$, resulting in L fields on the (ensemble) model grid.

3. Multiply with $\hat{\mathbf{X}}^b$: multiply each of the L vectors element by element with the respective ensemble deviation $[\mathbf{x}^{b(\ell)} - \bar{\mathbf{x}}^b]$ and sum over the L products. The result has the same dimension as $\mathbf{z}$, being defined for each model variable at each grid-point.

It can be seen that most of the computational effort has to be spent for multiplying L vectors with the matrix $\mathbf{C}$. An efficient implementation for this operation is discussed in the following section.

10.3 Representation of the localization matrix C

As discussed in Section 9 the elements c_{ij} of the localization matrix $\mathbf{C}$ are chosen as a function of the spatial distance of the location of the respective grid-points i and j. Localisation functions have similar shape as Gaussian functions and are chosen here as the Gaspary & Cohn function in multidimensional space. This ensures that the resulting matrix is positive definite and matrix elements are zero for distances larger than a given cutoff radius. However for smaller radii the matrix remains dense and matrix vector multiplications would require a relatively large number of multiplications. Thus we will aim for another representation which is computational efficient and approximates a Gaussian function as well. We can exploit the following points:

1. Multiplication of the matrix is basically the application of a smoothing operator or convolution with a Gaussian like kernel.

2. To ensure that the resulting operator is positive definite we will factorize it as $\mathbf{C} = \mathbf{S}\mathbf{S}^T$ with $\mathbf{S}$ being the square root of $\mathbf{C}$. Again $\mathbf{S}$ can be represented by a smoothing operator (with lengthscale smaller by a factor $2^{-1/2}$).

3. As the localization length scale is considerably larger than the grid spacing, the smoothing operator can be efficiently implemented on a hierarchical grid. $\mathbf{S}$ need neither be the symmetric square root nor be a square matrix at all, but instead a matrix of size n times n_c, with n being the number of model grid-points and n_c the number of grid-points of a coarser grid, with grid-spacing of order of the localization length scale.

4. As localisation in vertical and horizontal direction is seperable, we can split up the matrix $\mathbf{S}$ in a vertical and horizontal operator $\mathbf{S} = \mathbf{S}_v\mathbf{S}_h$.

Thus the ensemble background error covariance matrix has the following form:

$$\mathbf{P}^b_{EnKF} = (L-1)^{-1}\,\hat{\mathbf{X}}\,\mathbf{S}_v\,\mathbf{S}_h\,\mathbf{S}_h^T\,\mathbf{S}_v^T\,\hat{\mathbf{X}}^T \tag{10.8}$$

10.4 Detailed description of the operators applied

Equation (10.8) describes the representation of $\mathbf{P}^b_{EnKF}$ on the model grid (in the resolution of the ensemble). If used in the PSAS algorithm we have to use $\mathbf{H}\,P^b_{EnKF}\,\mathbf{H}^T$, i.e. we have to consider interpolation operators I to the locations of the observations. Furthermore in the MPI parallel environment transposition operators $\mathbf{T}$ which redistribute data in between processors are needed. Taking this into account the background background error covariance matrix between observation locations has the following form:

$$I\mathbf{P}^b_{EnKF}\,I^T = (L-1)^{-1}\,I_v\,I_h\,\mathbf{T}_{og}\,\hat{\mathbf{X}}\,\mathbf{S}_v\,\mathbf{S}_h\,\mathbf{T}\,\mathbf{S}_h^T\,\mathbf{S}_v^T\,\hat{\mathbf{X}}^T\;\mathbf{T}_{og}^T\,I_h^T\,I_v^T \tag{10.9}$$

Multiplication of the matrix $\mathbf{P}^b_{EnKF}$ with a vector z is achieved by the following sequence of operations:

1. Multiplikation with $I_h^T\,I_v^T$: (the adjoint of the interpolation operators.) The elements of the vector $\mathbf{z}$ are defined at the pressure levels of the observations for the adjoint variables of geopotential height, virtual temperature, (generalised) relative humidity, and wind components. The interpolation operators $I_v\,I_h$ are mainly the same as those described in Section 7.3 for the NMC background error correlation model. However here they do not include any spatial differentiation as the EnKF background error is represented in physical quantities, not streamfunction or velocity potential.

2. Multiplikation with $\mathbf{T}_{og}^T$: The interpolation operators (and their adjoints) act at the locations of the observation and they are evaluated on the processors which hold the observations. For the (ensemble) model grid a different paralelisation strategie is followed. The transposition operator $\mathbf{T}_{og}^T$ redistributes the model columns accordingly.

3. Multiplikation with $(\hat{\mathbf{X}}^b)^T$: The vector at the right hand side is defined for each model variable (temperature, wind components, humidity) at each model grid-point. For each of the L ensemble members l the vector is multiplied grid-point by grid-point and variable by variable with the deviation $[\mathbf{x}^{b(\ell)} - \bar{\mathbf{x}}^b]$. The results for all variables are summed up.

   The operations of this step are performed on the original model grid. As model columns are distributed over all processor elements computations are distributed over processors accordingly.

   The final result of the operation is a set of L fields (for each ensemble member). The above operations for each of the L ensemble members are independent from each other, so that the operations of this step 3, as well as those of the subsequent steps 4 to 9 can be performed sequentially.

4. Multiplikation with $\mathbf{S}_v^T$: This is a 1-dimensional smoothing operator in vertical direction. It is evaluated on a grid with linear spacing in $log(p)$ corresponding to the choice of the vertical localisation parameter. As motivated previously the result is computed on a coarser grid with spacing corresponding to the vertical localisation length scale. Matrix elements between coarse grid-points (here index j) and model grid-points (index i) are calculated as $s_{ij} = N_i\, cg(\triangle_{ij})$. Here $cg(\triangle_{ij})$ is the Gaspari&Cohn function depending on the 'distance' in $log(p)$ with vertical localisation length scale multiplied by $2^{-1/2}$. The factor N_i is a normalisation factor ensuring that $\sum_j (N_i\, cg(\triangle_{ij}))^2 = 1$. This ensures that diagonal elements of $\mathbf{S}_v\mathbf{S}_v^T$ have value 1, i.e. correct normalisation for a correlation function.

   Because the coarse grid spacing is chosen of order of the localisation length scale and the support of the Gaspari&Cohn function is of order 6 (-3...3) the number of nonzero coefficients of $\mathbf{S}_v$ per model gridpoint i is of order 6. consequently the number of operations involved is about 6 times the number of model gridpoints. The size of the result vector is $m_h\, m_v$ there n_h is the number of horizontal grid-points and m_v is the number of vertical modes (vertical model extend divided by the localisation length scale). The latter is of order 10 which is considerably smaller than the number of vertical grid-points.

5. Multiplikation with $\mathbf{S}_h^T$: This is a 2-dimensional smoothing operator in horizontal direction. As motivated previously the result is computed on a coarser grid with spacing corresponding to the vertical localisation length scale. Different approaches are used for regular regional lat-lon grids (COSMO) and irregular global grids (GME, ICON):

   COSMO

   On the regular (rotated) lat-lon grid the horizontal localisation operator can be split into seperable operators $\mathbf{S}_h^T = \mathbf{S}_x^T\mathbf{S}_y^T$. This option is currently not implemented.

   GME/ICON

   For global irregular grids a GME grid is taken for the coarse grid. The GME resolution parameter ni is chosen so that the grid spacing is equal or smaller to the localisation length scale. Again coefficients take the form $s_{ij} = N_i\, cg(\triangle_{ij})$ with proper normalisation N_i. Now of order 30 coefficients are required for each horizontal model grid-point i. For a horizontal localisation length scale of 300 km ni becomes 24 corresponding to 6250 horizontal gridpoints.

6. Transposition $\mathbf{T}$: As model columns are distributed over all processor elements, the computations in the vertical (step 4, $\mathbf{S}_v^T$) are evaluated on different processors. The previous step 5 ($\mathbf{S}_v^T$) actually consists of smoothing the results horizontally on a coarse grid, i.e. of multiplying the right hand side with some predefined coefficients and sum them up. This requires input from different processors. However, as the result grid is coarse (and thus the amount of resulting data is manageable) we perform the summation of the terms independently on each processor, and finally in this transposition step $\mathbf{T}$ take the sum over the individual results of each processor and again redistribute it to all processors.

7. Multiplikation with $\mathbf{S}_h$: This step corresponds to the multiplication with the transposed of step 5.

8. Multiplikation with $\mathbf{S}_v$: This step corresponds to the multiplication with the transposed of step 4.

9. Multiplikation with $\hat{\mathbf{X}}^b$: The previous step provides the ensemble member weights on the model grid. It is multiplied with the respective ensemble deviation and summed up to finally yield the analysis increment. This step is the last one of steps 3 to 9 which is performed sequentially for each ensemble member.

10. Multiplikation with $\mathbf{T}_{og}$: This transposition operator redistributes the model columns to the processors which handle the observations.

11. Multiplikation with $I_v\, I_h$: Apply the interpolation operators from the (ensmble) model grid to the location of the observations.

In the postmultiplication step the result of the multiplication of a vector with $\mathbf{P}^b_{EnKF}$ is not interpolated to the locations of the observations but to the grid of the deterministic analysis. In this case (10.9) is applied in the same way, merely in the final steps 10 and 11 ($I_v\, I_h\, \mathbf{T}_{og}$) the transposition and interpolation operators act in between the ensemble grid and the deterministic analysis grid.

Some computational optimisations are possible in the above procedure (10.9): The columns of the ICON grid are distributed arbitrarily over processors. Thus we are free to allocate the columns of the ensemble deviations $\mathbf{X}$ on the same processors as used for handling the observations (and dublicate them if the same grid column is required for observations on different processors). Then the transpositions $\mathbf{T}_{og}$ and $\mathbf{T}^T_{og}$ become obsolete.

10.5 Prospects for Multiscale Localisation

Schur product multiplication of $\mathbf{P}^b_{EnKF}$ with a localisation matrix $\mathbf{C}$, is an ad hok approach to suppress stochastical noise due to the limited ensemble size for matrix elements which are known to be small, here correlations for large spatial distances. However application of the Schur product in model representation is a somehow arbitrary choice and other options are possible. Localisation may be applied in transformed representation.

In the ensemble framework this corresponds to

1. derive the ensemble deviations in transformed representation by applying the transformation $\mathbf{T}^{-1}$:
$$\widetilde{\mathbf{X}} = \mathbf{T}^{-1}\mathbf{X}$$

2. apply the Schur product using a localisation matrix $\widetilde{\mathbf{C}}$ in transformed representation:
$$\mathbf{P}^b_{EnKF} = \mathbf{T}\,\widetilde{\mathbf{P}^b_{EnKF}}\,\mathbf{T}^T = \mathbf{T}\left(\widetilde{\mathbf{C}} \circ (\widetilde{\mathbf{X}}\widetilde{\mathbf{X}}^T)\right)\mathbf{T}^T$$

For specific choices of the transformation $\mathbf{T}$ this has the following effects:

Spectral transform

If $\mathbf{T}$ is a spectral transform, localisation on $\widetilde{\mathbf{P}^b_{EnKF}}$ corresponds to the suppression of correlations between different scales. This is equivalent to a spatial smoothing of the correlations. The method has been applied when deriving the climatological static background correlations $\mathbf{P}^b_{NMC}$ for the 3dVar from the NMC method by using the FFT transformation for zonal averaging of the correlations.

Wavelet transform

If $\mathbf{T}$ is a wavelet transformation, localisation on $\widetilde{\mathbf{P}^b_{EnKF}}$ corresponds to the suppression of correlations between different scales and different locations. This kind of multiscale localisation becomes important especially for assimilation in regional models or refined regions there multiple scales with different correlation length scales are involved, for instance convective and synoptic scales.

The technique to apply the localisation matrix on a coarser grid as described in the previous section can be extended to localisation in wavelet representation using a hierarchical model grid if we define the transformations $\mathbf{T}$ and $\mathbf{T}^{-1}$ as follows:

Inverse transformation $\mathbf{T}^{-1}$:

1. repeatedly smooth the ensemble deviation $\mathbf{X}$ and represent it on a coarser grid.
2. repeatedly re-interpolate the smoothed fields from the coarser to the next finer grid and subtract it from the ensemble deviations represented on the latter.

This procedure seperates the ensemble deviations on different scales. Localisation using different length scales may be applied to each of them.

Forward transformation T: repeatedly re-interpolate the smoothed fields from the coarser to the next finer grid and add it to the ensemble deviations represented on the latter scale.

The proposed method does not provide an orthogonal wavelet transformation (which would imply $\mathbf{T}^T = \mathbf{T}^{-1}$). However, this is not mandatory for this application. The method is known as the lifting scheme and especially applicable to irregular grids on the sphere.

10.6 Usage of the VarEnKF

The following namelist parameters are relevant for steering the VarEnKF:

Namelist Group	Parameter	Value for VarEnKF	Description
/RUN/	method	'VARENKF'	Switches on VarEnKF.

In the global data assimilation cycle the deterministic 3dVar and ensemble LETKF analyses are derived in the same call of the data assimilation binary. The sequence of operations depends on the method specified in the namelist group /RUN/: PSAS denotes a deterministic 3dVar only. PSAS+LETKF denotes the LETKF ensemble analysis with

preceding 3dVar which provides quality control and bias correction of observations. for VARENKF in addition the ensemble background error covariance matrix is used in the variational scheme. In detail the following sequence of operations is performed:

method			task
PSAS	PSAS+LETKF	VARENKF	
X	X	X	read observations and deterministic forecast
	X	X	read forecast ensemble
		X	set up ensemble B matrix to be used in the variational scheme.
X	X	X	monitoring of observations vs. deterministic forecsat. FG-check, observational QC.
X	X	X	deterministic variational analysis, including variational quality control of observations.
	X	X	LETKF ensemble analysis.

Chapter 11

Supplemental Modules SST, ICE, SMA, SNOW

In Section 2.3, the concept of the four modules was introduced which produce analysis fields for

- Sea Surface Temperature (SST) and sea Ice coverage (ICE) (both in Section 11.1)
- the Soil Moisture Analysis (SMA) (Section 11.2)
- the Snow coverage (SNOW) (Section 11.3)

In this Chapter, the analysis methods of these modules are documented. They stem from the COSMO package [22], which is called *LM* in this Chapter.

HL: This documentation of SST, ICE, SNOW, SMA is simply COPIED from the COSMO-Docu Part III (Data Assimilation). It needs a real good overhaul to make it consistent with the rest of this docu. Also, someone needs to tell Christoph Schraff that this DWD-DA-Docu has now taken the job of documenting these 4 modules!

11.1 SST and ICE: Analysis of Sea Surface Temperature and Ice Cover

11.1.1 Overview

Over water, the sensible and latent heat fluxes at the surface strongly depend on the surface temperature. The correctness of its specification is a prerequisite for realistic simulations of cyclonic development, particularly in baroclinic regions. In addition, the location of the sea ice boundary is also very important since strong vertical heat fluxes can be induced when cold air masses are advected from ice-covered areas to the open water.

Therefore, sea surface temperature and the location of the sea ice boundary are analyzed 2-dimensionally in another separate procedure. Typically, this analysis is performed once a day at 0 UTC as part of the data assimilation.

The object of the sea surface temperature analysis for LM is to capture not only the slowly varying large-scale temperature patterns, but also smaller-scale phenomena such as cold anomalies due to upwelling currents or the relatively rapid warming in shallow coastal areas in periods of strong solar irradiation. However, diurnal or very small-scale variations should be filtered out because sea surface temperature is held constant during the forecast.

11.1.2 Analysis Method

Firstly, the extent of the sea ice cover is analyzed. For this purpose, external analyses are directly interpolated to the model grid. Namely for the Baltic Sea, a weekly analysis from the Federal Maritime and Hydrographic Agency of Germany (BSH) with a latitudinal–longitudinal resolution of 0.1 by $0.1\overline{6}$ degrees is used. For other areas, a daily global analysis (Grumbine, 1996) from the Ocean Modelling Branch of NCEP based on SSMI satellite data is available at a resolution of 0.5 by 0.5 degrees and could be used if required. Sea ice temperature is currently derived from ECMWF climatology, however it is planned to compute this quantity by means of a one-layer sea ice model incorporated in the LM in the foreseeable future.

Sea surface temperature (SST) is analyzed by means of a correction scheme. The analysis is determined by adding weighted observation increments to a first guess or background field in the environs of the observations. As a first guess field, the interpolated SST analysis of GME is deployed for which the first guess field is given by a 0.5 by 0.5 degree SST analysis of NCEP. The latter analysis does not only use in-situ observations but also bias-corrected satellite data (Reynolds and Smith, 1994). Hence, the SST analysis of LM does also benefit from these satellite data.

The observational data set used for the SST analysis of LM comprises of all the ship and buoy data from within the previous 5 days. As a quality control, the data are checked against the first guess and against observations from the other stations in the vicinity. For the analysis at each grid point, the first guess value is corrected by a weighted mean of all the observation increments. The individual weights depend on the distance between

analysis and observation time, on the observation type, and on the spatial distance of the observation from the target point according to Eq. (11.18) with $\Delta z = 0$ and $s = 200\,\mathrm{km}$ ($s = 430\,\mathrm{km}$ for the SST analysis of GME).

11.2 SMA: Variational Soil Moisture Analysis

11.2.1 Overview

2–m temperature observations are assimilated by the variational soil moisture analysis, as presented in the following. Soil water content influences screen–level values for temperature and relative humidity during clear–sky days. An inaccurate specification can result in forecast errors up to several degrees centigrade. Since direct measurements of soil moisture contents are rarely available, an indirect determination is necessary. This is done by a variational method: The optimal soil moisture contents minimize a cost functional that expresses the differences between model derived and observed screen–level temperature and humidity[1] (cf. Mahfouf et al., 1991, Callies et al., 1998, Rhodin et al., 1999, and Bouyssel et al., 1999).

Since, basically, the soil moisture fields are adapted so that the model screen–level forecasts approximate the observed values, the retrieved soil moisture fields depend essentially on the used forecast model, especially on the applied soil and boundary layer parameterizations. Errors of the model forecasts are reflected in errors of the retrieved values. However, the resulting forecasts are improved and the computed heat and moisture fluxes at the bottom of the atmosphere model have a good chance to be improved as well.

Since the soil–atmosphere coupling is strongest with high radiative impact, observations close to noon are assimilated. The diurnal variation of 2–m temperature and 2–m relative humidity have to be provided by the physical parameterizations.

The soil moisture analysis is applied once per day in order to provide improved soil moisture fields to be used by the forecast that starts at the following day. Because the soil–atmosphere coupling is not always strong enough to derive sufficient information on the soil moisture contents to compute them in a reliable way, a Kalman filter cycled analysis is applied that incorporates a background state along with background error estimates. In case of low radiative impact, the retrieved moisture fields remain close to the background. High impact, on the other hand, results in improved moisture fields and reduced background error estimates.

The variational Kalman–filter analysis scheme that is documented in this section requires one additional forecast run for each analyzed soil moisture layer, but does not depend on a specific soil model nor on a certain boundary layer model and is applicable for general numerical models and technical environments.

11.2.2 Variational Analysis

The variational analysis scheme derives improved soil moisture contents by minimizing a cost functional. The minimization problem is of high dimension in general; the moisture contents of each horizontal grid column of every soil layer has to be retrieved. However, since the screen–level values for temperature and relative humidity are mainly vertically

[1] Currently only 2–m temperature observations are assimilated; 2–m relative humidity observations can be included in the analysis in a straight forward way as soon as the model forecast values are in comparative quality.

coupled to the soil moisture contents of the same grid column (at least in case of *clear–sky conditions* with strong soil moisture–atmosphere coupling), a horizontal decoupling is assumed that reduces the high–dimensional minimization problem to a large series (one for each horizontal grid point) of low–dimensional (the number of analyzed soil layers) minimizations[2]. In this way the computational requirements are essentially reduced.

In the following the formulation of the variational soil moisture analysis scheme is given for an arbitrary horizontal grid point and for the assimilation of 2–m temperature observations only.

Let η and η^b denote vectors of dimension n^{soil} containing the moisture contents of the analyzed soil layers and their background states, respectively. The vectors T^o and $T(\eta)$ of dimension n^{obs} contain analyses (based on synoptic observations) and model forecasts, respectively, of 2–m temperature for specified observation times. The cost functional $\mathcal{J}$ to be minimized at each analysis step (i. e. daily) reads

$$\mathcal{J}(\eta) = \mathcal{J}^o(\eta) + \mathcal{J}^b(\eta) \tag{11.1}$$

with the observation term

$$\mathcal{J}^o(\eta) = \frac{1}{2}\Big(T^o - T(\eta)\Big)^T \mathbf{R}^{-1} \Big(T^o - T(\eta)\Big) \tag{11.2}$$

and the background term

$$\mathcal{J}^b(\eta) = \frac{1}{2}(\eta - \eta^b)^T \mathbf{B}^{-1} (\eta - \eta^b) \quad , \tag{11.3}$$

with $ADP \leq \eta_j \leq PV\,, \quad j = 1, \ldots, n^{soil}$. The components of η (indicated with lower indices) are limited by air dryness point (ADP) and pore volume (PV) of the actual soil type.

Matrix $\mathbf{R} \in \mathbb{R}^{n^{obs} \times n^{obs}}$ denotes the observation error covariance matrix and $\mathbf{B} \in \mathbb{R}^{n^{soil} \times n^{soil}}$ the background error covariance matrix. Both matrices $\mathbf{R}$ and $\mathbf{B}$ are symmetric and positive definite for physical reasons. Matrix $\mathbf{R}$ is assumed constant and diagonally dominant. At the start of the cycled soil moisture analysis scheme matrix $\mathbf{B}$ is initialized with estimated error variances and error covariances of the first guess soil moisture fields that are used as initial background η^b. Further on, the background values η^b and the background error covariance matrix $\mathbf{B}$ are provided within the cycled Kalman filter analysis scheme, see Section 11.2.2.

Minimization of $\mathcal{J}$ results in the analyzed soil moistures η^a:

$$\mathcal{J}(\eta^a) \leq \mathcal{J}(\eta) \quad \text{for all} \quad \eta \neq \eta^a \quad . \tag{11.4}$$

The minimization is performed under the assumption that the linearization given in Section 11.2.2 is valid; a unique minimum is guaranteed.

[2] Experiments have been carried out that confirm the assumption of horizontal decoupling for grid sizes of 14 km and 7 km.

Minimization

Although the soil moisture – 2–m temperature dependency is nonlinear in general, linearization around the background state provides good approximations as long as the retrieved values are not too far from the background state.

Linearization of the model 2–m temperature $T(\eta)$ around η^b gives

$$T(\eta) \doteq T(\eta^b) + \mathbf{\Gamma}\,(\eta - \eta^b) \quad , \tag{11.5}$$

where the Jacobian $\mathbf{\Gamma} \in \mathbb{R}^{n^{obs} \times n^{soil}}$ is approximated by one–sided finite differences,

$$\Gamma_{i,j} := \min\left(\frac{T_i(\eta^j) - T_i(\eta^b)}{\eta^j_j - \eta^b_j} \;,\; 0 \right) \quad , \tag{11.6}$$

with $i = 1, \ldots, n^{obs}$ and $j = 1, \ldots, n^{soil}$. The approximation (11.6) requires the routine forecast based on the background moisture contents η^b and n^{soil} *additional* forecast runs with varied soil moisture contents η^j. The partial derivatives are known to be negative for physical reasons.[3] The components of vector η^j are computed by

$$\eta^j_k = \begin{cases} \eta^j_j & \text{for} \quad k = j \\ \eta^b_j & \qquad\quad k \neq j \end{cases} \quad , \quad k = 1, \ldots, n^{soil} \quad , \tag{11.7}$$

where the varied soil moisture content η^j_j is altered depending on air dryness point (ADP) and field capacity (FC) of the soil model in order to reduce the influence of the soil type of the actual horizontal grid point. The size of the alteration $\Delta\eta$ is given by

$$\Delta\eta = (FC - ADP)\,\Delta\epsilon \quad , \tag{11.8}$$

the fraction $\Delta\epsilon \in]0, \frac{1}{2}[$ is a tuning parameter. The direction of the variation is chosen according to the forecast error as long as the background η^b is not too close to the limits ADP and FC, i. e. :

$$\eta^j_j = \begin{cases} \min(\eta^b_j, FC) + \Delta\eta & \text{for} \begin{cases} \sum_{i=1}^{n^{obs}} T_i(\eta^b) > \sum_{i=1}^{n^{obs}} T^o_i \text{ and } \eta^b_j + \Delta\eta < FC \\ \text{or} \\ \eta^b_j - \Delta\eta \leq ADP \end{cases} \\ \min(\eta^b_j, FC) - \Delta\eta & \text{for} \begin{cases} \sum_{i=1}^{n^{obs}} T_i(\eta^b) \leq \sum_{i=1}^{n^{obs}} T^o_i \text{ and } \eta^b_j - \Delta\eta > ADP \\ \text{or} \\ \eta^b_j + \Delta\eta \geq FC \end{cases} \end{cases} \tag{11.9}$$

Although soil moisture influences evapotranspiration and 2–m temperature only between ADP and FC (as defined by the applied soil model of the LM[4]), higher moisture

[3] Slightly varying cloud covers in the different forecast runs can result in erroneous positive finite differences, that are eliminated by the minimization.

[4] Similar boundaries should exist for other soil models.

values up to PV are allowed to reduce the instant impact on the soil model and to provide more realistic soil moisture values (e. g. in case of heavy precipitation).

In order to provide reasonable analysis increments for background values η^b that exceed FC, it is necessary to derive non–vanishing soil moisture – 2–m temperature dependencies. The minimization of η_j^b by FC in Equation (11.9) assures that the varied soil moisture contents reside in the sensitive range between ADP and FC even if η_j^b exceeds FC.

Using the linearization (11.5) the gradient of the cost function can be analytically expressed as

$$\nabla \mathcal{J}(\eta^b) = -\mathbf{\Gamma}^T \mathbf{R}^{-1} \left(T^o - T(\eta^b) - \mathbf{\Gamma} \left(\eta - \eta^b \right) \right) + \mathbf{B}^{-1} \left(\eta - \eta^b \right) \quad . \tag{11.10}$$

For the low–dimensional minimization problem it is highly efficient to solve

$$\nabla \mathcal{J}(\eta^a) \stackrel{!}{=} 0 \tag{11.11}$$

directly[5]. Little calculus gives the minimum η^a of the cost function as

$$\eta^a = \eta^b + \left(\mathbf{\Gamma}^T \mathbf{R}^{-1} \mathbf{\Gamma} + \mathbf{B}^{-1} \right)^{-1} \mathbf{\Gamma}^T \mathbf{R}^{-1} \left(T^o - T(\eta^b) \right) \quad . \tag{11.12}$$

Equation (11.12) is the formula that is implemented to actually compute η^a.

Worth to mention that the applied minimization by linearization and direct solution is no degradation in accuracy of the retrieved soil moisture fields.

The minimization of the cost functional $\mathcal{J}$ is performed under the constraint that the analyzed values η^a are in the range

$$ADP \leq \eta_j^a \leq PV \quad , \quad j = 1, \ldots, n^{soil} \quad . \tag{11.13}$$

If the global minimum of the cost functional resides outside this valid range, the minimum at the boundaries is computed to result in the analyzed values. The algorithm that computes the minimum considering the boundary side constraints is rather technical and will not be reported here[6].

Kalman–filter Cycling

The soil moisture analysis is performed daily for 0 UTC. For the start of the cycled soil moisture analysis scheme the background error covariance matrix $\mathbf{B}$ is initialized with estimated error variances and covariances $\mathbf{B}^0$ of first guess moisture fields that are used as initial background η^b.

The background state $(\eta^b)^{t+1}$ and the background error covariance matrix $(\mathbf{B})^{t+1}$ for the following day are provided in a Kalman–filter cycled analysis (the valid times of the variables η^a, η^b, $\mathbf{A}$, and $\mathbf{B}$ are indicated from now on by upper indices outside brackets).

[5] Because of the linearization a unique minimum is guaranteed.

[6] A simpler approach would be to compute the global minimum and restrict the moisture values in both soil layers independently to the valid range. Experiments showed only slight differences in the analyses.

An increase of confidence in the retrieved soil moisture values due to the assimilated screen–level observations as well as a decrease due to the model error of the soil model are taken into account.

The background soil moisture contents for the following day $(\eta^b)^{t+1}$ are computed as

$$(\eta^b)^{t+1} = (\eta^a)^t + \left(M_t^{t+1}\left((\eta^b)^t\right) - (\eta^b)^t\right) \quad , \tag{11.14}$$

where $M_t^{t+1}\left((\eta^b)^t\right)$ are the 24 h model values that result from the routine forecast that is started at 0 UTC with the background fields $(\eta^b)^t$. Changes in soil moisture contents by precipitation and evapotranspiration during the 24 hours are taken into account in this way without the requirement of another additional forecast run.

The confidence in the retrieved values $(\eta^a)^t$ is given by the analysis error covariance matrix $\mathbf{A} \in \mathbb{R}^{n^{soil} \times n^{soil}}$,

$$(\mathbf{A})^t = \left(\nabla^2 \mathcal{J}\right)^{-1} = \left(\mathbf{\Gamma}^T \mathbf{R}^{-1} \mathbf{\Gamma} + \left((\mathbf{B})^t\right)^{-1}\right)^{-1} \quad , \tag{11.15}$$

which is the inverse of the Hessian of $\mathcal{J}$ (e. g. Tarantola, 1987). If there is little dependence of the moisture contents on the 2–m temperatures ($\mathbf{\Gamma} \approx 0$), then $(\mathbf{A})^t$ almost equals the background error covariance matrix $(\mathbf{B})^t$. The larger the dependence is, the smaller the estimated analysis errors become.

An auxiliary new background error covariance matrix $(\widetilde{\mathbf{B}})^{t+1}$ is computed by

$$(\widetilde{\mathbf{B}})^{t+1} = \mathbf{M}\,(\mathbf{A})^t\,\mathbf{M}^T + \mathbf{Q} \quad , \tag{11.16}$$

where matrix $\mathbf{M} \in \mathbb{R}^{n^{soil} \times n^{soil}}$ is an estimation of the tangent linear of the forecast operator M_t^{t+1}. Matrix $\mathbf{Q} \in \mathbb{R}^{n^{soil} \times n^{soil}}$ expresses the assumed error of M_t^{t+1}. This additive term reduces the sensitivity of the background to past observations and is important to keep the retrieved moisture contents variable in long–term cycled analyses.

In case of weak soil–atmosphere coupling ($\mathbf{\Gamma} \approx 0$) for a sequence of days, the background error covariance matrices $(\mathbf{B})^{t+1}, (\mathbf{B})^{t+2}, \ldots$ increase linearly with $\mathbf{Q}$, which reflects the reduced confidence in the retrieved moisture fields and larger variations become possible in subsequent analyses. The error variances and error covariances of $\mathbf{Q}$ are the main tuning parameters of the cycled assimilation scheme.

The background error covariance matrix $(\mathbf{B})^{t+1}$ is limited by $\mathbf{B}^{max}$ in order to prevent the background errors from unlimited growth that could affect the stability of the soil moisture analysis in case of longer periods with no soil moisture impacts (e. g. snow). Strictly speaking: the variances (diagonal elements of $(\widetilde{\mathbf{B}})^{t+1}$) are limited by $\mathbf{B}^{max}$ and the covariances (non–diagonal elements) are adjusted in order to result in the same correlation coefficients, i. e. the coefficients of $(\mathbf{B})^{t+1}$ are computed as

$$b_{i,j} = \begin{cases} \min(\tilde{b}_{i,j}, b_{i,j}^{max}) & \text{for} \quad i = j \\ \tilde{b}_{i,j} \sqrt{\frac{b_{i,i}\, b_{j,j}}{\tilde{b}_{i,i}\, \tilde{b}_{j,j}}} & \text{for} \quad i \neq j \end{cases} \quad , \quad i,j = 1, \ldots, n^{soil} \quad .$$

Free Parameters

The tuning parameters of the variational soil moisture analysis scheme are the matrices $\mathbf{R}$, $\mathbf{Q}$, $\mathbf{B}^0$, $\mathbf{B}^{max}$, and $\mathbf{M}$ as well as the fraction $\Delta\epsilon$. (They are controlled by namelist parameters of the model `SMA`.)

11.3 SNOW: Snow Analysis

11.3.1 Overview

Knowledge of the distribution of snow is necessary for the determination of surface albedo and surface fluxes in weather prediction models. The presence of a snow cover reduces dramatically the absorption of short-wave radiance at the ground and hence strongly influences the local near-surface temperature. Moreover, the distribution of snow cover can also have a significant large-scale impact as it may affect the characteristics of air masses.

Therefore, snow depth is analyzed 2-dimensionally in a separate procedure and transformed into the model variable snow water content in order to be used in the LM. Typically, this is done at a frequency of once every 6 hours during the data assimilation cycle. Note that in the context of short-range numerical weather prediction, it is less important to determine exactly the snow depth than the horizontal extent of the snow-covered areas.

The object of the snow depth analysis is to find a proper value for each model grid point which is representative of the grid box and the height of the model orography. This height is a mean value of the real orography within the grid box.

11.3.2 Input Data

There are three sources of information that are used in the snow depth analysis:

1. surface-level synoptic observations (SYNOP);

2. model 'forecast' values of snow water content transformed into snow depth from the LM nudging run for data-pour areas where the total influence (weight) of the SYNOP observations is below a given threshold;

2. monthly snow depth climatology (from ECMWF) for defining permanently ice-covered glacial areas.

Depending on the information content, different data are extracted from SYNOP reports:

- Total snow depth. If this is not reported then also:

- 6-hourly precipitation sum. This is converted into a snow depth increment provided that the 2-m temperature T_{2m} (observed or, if it is not observed, model-derived) is below 0° C and the present or past weather observations ww indicate snowfall. If 6-hourly precipitation is also missing, then also:

- Present and past weather observations. In case of reported snowfall, a snow depth increment is calculated using an empirical function of ww. Other weather observations result in zero snow depth increments.

11.3.3 Quality Control

At first, observed snow depth is subject to a plausibility check. It is rejected if it exceeds an acceptance limit $d_{sn}^{al} = 1.5\,[m] \cdot (1 + z_{ob}/800\,[m])$ which depends on station height z_{ob}.

Then, a first guess quality control check is performed. Here, the previous snow depth analysis $d_{sn}^{a_{(t-1)}}$ is regarded as a first guess for truth, and the observation is rejected if it deviates from that guess by more than a threshold value d_{sn}^{thr} given by

$$d_{sn}^{thr} \;=\; 0.8\,[m] \cdot \left(1 + \frac{z_{ob}}{2000\,[m]}\right) \cdot \max\left(0, \min\left(1, \frac{287.16\,[K] - T_{2m}}{10\,[K]}\right)\right) \tag{11.17}$$

11.3.4 Analysis Method

The analysis method used for snow depth is based on a simple weighted averaging of observed values and does not use a background field except in data-pour regions. The individual weight w_k of an observation k at a target grid point depends both on the horizontal distance Δr and the vertical distance Δz between these two points,

$$w_k \;=\; \max\left(\frac{s^2 - \Delta r^{\,2}}{s^2 + \Delta r^{\,2}}\;,\; 0\right) \;\cdot\; \max\left(\frac{z_s^2 - \Delta z^{\,2}}{z_s^2 + \Delta z^{\,2}}\;,\; 0\right) \tag{11.18}$$

The weight is zero for observations which are further away from the target point than the radii of influence s and z_s. The horizontal radius s is set to 120 km in data-dense areas and to 200 km elsewhere, whilst the vertical radius of influence $z_s = 0.4 \cdot z + 180\,\mathrm{m}$ increases with increasing height z of the target point.

Using Eq. (11.18), the weighted averages of snow depth observations $\overline{d_{sn}}^{ob}$ and of snow depth increments $\overline{\Delta d_{sn}}^{ob}$ derived from precipitation and weather observations are calculated as well as the sum of weights $W^d = \sum w_k^d$ resp. $W^{\Delta d} = \sum w_k^{\Delta d}$. The total weights W^d and $W^{\Delta d}$ are measures for the local data density of the two types of observations.

As these data densities may vary strongly in space and time, the way to determine the final analyzed snow depth depends on them. If the sum of weights of snow depth data is greater than a prescribed value W_{suf} then the analyzed snow depth d_{sn}^a at that grid point is set simply to the weighted mean value of observed snow depth, i.e.

$$d_{sn}^a \;=\; \overline{d_{sn}}^{ob} \;=\; \sum_k w_k \cdot d_{sn}^k \;/\; W^d \qquad , \qquad W^d \;\geq\; W_{suf} \;\equiv\; 2 \tag{11.19}$$

If this condition is not satisfied but the mean of the two data densities is greater than W_{suf}, i.e. if $\frac{1}{2}\cdot(W^d + W^{\Delta d}) \geq W_{suf} > W^d$, then (only) the weighted mean of snow depth increments $\overline{\Delta d_{sn}}^{ob}$ is additionally taken into account. The weighted mean increment is added to the previous analysis $d_{sn}^{a_{(t-1)}}$, and a possible snow melt Δd_{sn}^{melt} is subtracted. The result is combined with the mean weighted observed snow depth so that

$$d_{sn}^a \;=\; \frac{W^d}{W_{suf}} \cdot \overline{d_{sn}}^{ob} \;+\; \left(1 - \frac{W^d}{W_{suf}}\right) \cdot \left(d_{sn}^{a_{(t-1)}} + \overline{\Delta d_{sn}}^{ob} - \Delta d_{sn}^{melt}\right) \tag{11.20}$$

Whenever the mean of the two data densities is also smaller than W_{suf}, the third data source, i.e. the 6-hour model forecast of snow water content converted into snow depth

d_{sn}^{fc}, is also used. Then, the analyzed value is given by

$$d_{sn}^{a} = \frac{W^d}{W_{suf}} \cdot \overline{d_{sn}}^{ob} + \frac{W^{\Delta d}}{2\,W_{suf}} \left(1 - \frac{W^d}{W_{suf}}\right) \cdot \overline{d_{sn}}^{inc} + \left(1 - \frac{W^{\Delta d}}{2\,W_{suf}}\right)\left(1 - \frac{W^d}{W_{suf}}\right) \cdot d_{sn}^{fc} \quad (11.21)$$

where $\overline{d_{sn}}^{inc} = d_{sn}^{a(t-1)} + \overline{\Delta d_{sn}}^{ob} - \Delta d_{sn}^{melt}$.

At grid points where climatological data indicate permanent ice cover the snow depth is set to 100 m . In a final check, it is ensured that the analysis increment at any grid point does not exceed a limit that depends on height and temperature.

Part III

User Guide

User Guide: Overview

This Part III *User Guide* explains how to install, run and cycle the three core data assimilation algorithms of 3-dimensional Variational Data Assimilation (3dVar) (Chapter 7), the Ensemble Kalman Filter (EnKF) in the formulation of the Local Ensemble Transform Kalman Filter (LETKF) (global LETKF and local Kilometre-Scale Ensemble Data Assimilation (KENDA), Section 9) and the Hybrid Variational Ensemble Kalman Filter (VarEnKF) (Section 10).

Historically, the 3dVar-algorithm is the basis of the DWD-Data-Assimilation package. The central binary which contains all three core algorithms is therefore called `var3d` – so, whenever in this user guide `var3d` is mentioned, it refers to the central software and *not necessarily to the algorithm of 3dVar*.

Chapter 12 **Installation**
: explains how to download and install the `var3d` DA software.

Chapter 13 **How to start the DA software**
: shows how to start the `var3d`-binary using the namelist `/RUN/`.

Chapter 14 **DA Software: Perform the Assimilation Methods**
: shows how to perform the five different DA methods:

  14.1 3dVar for global model (GME/ICON)

  14.2 LETKF for global model (GME/ICON)

  14.3 generation of global ensemble (GME/ICON)

  14.4 LETKF for local model (COSMO-KENDA)

  14.5 VarEnKF for global model (GME/ICON)

  The Sections explain which namelists, switches and basic parameters need to be set in order to perform these data assimilation algorithms.

Chapter 15 **Cycled Data Assimilation**
: describes how the concept of *cycled data assimilation* is performed in the DWD DA suite. Section 15.1 focuses on global 3dVar-cycling, 15.2 on global LETKF-cycling and 15.3 on the local KENDA-LETKF-cycling.

Chapter 16 **Namelist Groups**
: serves as a *reference for all the possible namelists* used in Chapter 14, often with back-links to the scientific docu for specific switches.

Chapter 17 **Observation Handling**
is 3dVar-specific and shows how to handle observation processing.

Chapter 18 **1D-Var**
is 3dVar-specific and shows how to set up observation retrievals.

Chapter 19 **Computation Environment**
gives information about the DWD NUMEX system (Section 19.1) and the utility- and test-programs that are contained in the `var3d`-package (Section 19.2).

Chapter 20 **File formats**
gives detailed information about the file formats used in `var3d`.

Chapter 21 **Program output (stdout)**
explains the control-output of the `var3d` software and its different modules and namelist sections.

Chapter 12

Installation

12.1 Source code management

Program development is done using the subversion (SVN) repository at the Max-Planck-Institute in Hamburg (same SVN server as for ICON).

Subversion is a source code revision system which allows to work the code concurrently by different persons, keep track of all changes and merge the changes applied by different developers. This note describes how to create a personal working copy of the code maintained in the repository and how to compile it for running a test-case.

The most important SVN commands are:

`svn checkout https://svn.zmaw.de/svn/osas/trunk/3dvar`
: First checkout of the DWD data assimilation system sources. SVN user-id and password are requested. The command will create a directory `3dvar/` with subdirectories holding the source code, makefiles, and configuration files.
The SVN repository is hosted by the ZMAW in Hamburg. Direct access to the repository is restricted by the firewall and granted only for exclusive IP adresses. See Section 12.1.1 for details.

`svn update [-r` *revision*`] [files]`
: This command will update the files (if given) or the content of the current directory (and subdirectories), taking into account the changes made by others since the last `checkout` or `update` command was issued. The `-r` flag switches the working directory to the specified revision.

`svn commit -m'`*comment*`' [files]`
: Changed files are written back to the repository by the `commit` command. If the files were committed by others since the last `checkout` or `update` was issued, they must be updated beforehand.

`svn diff [files]`
: Shows the changes between the working copy and the repository. Note that the `emacs` editor provides a mode for this comparison as well.

`svn log [file]`
Shows the log file on the revisions checked in (revision number, date, comment given by `svn commit -m'...'`, ...).

12.1.1 Access restrictions

The SVN repository is hosted by the ZMAW in Hamburg. The password can be changed interactively by using the WEB interface:

```
https://svn.zmaw.de
...
-> SVN access management
```

Direct access to the repository is restricted by the firewall and granted only for exclusive IP adresses. Some details for access from different sites are given below.

MPIfM / ZMAW sites :

No further precautions are required.

DWD sites :

In order to pass the DWD firewall the following lines must be included in /.subversion/servers :

```
...
[global]
http-proxy-host = ofsquid.dwd.de
http-proxy-port = 8081
...
```

Users with SSH access to ZMAW sites :

Access may be achieved by setting up an ssh tunnel:

1. Set up the SSH tunnel in a seperate window:
   `ssh -L 8888:svn.zmaw.de:443 m214030@login1.zmaw.de`
   Enter ZMAW user-id and password.
2. Access the SVN as described in Section 12.1. In the checkout command the repository has to be specified as follows:
   `svn checkout https://localhost:8888/svn/osas/trunk/3dvar`

12.2 Compilation

A Makefile is required for compilation in the source directory `3dvar/`. In order to create the Makefile the script `./Make` has to be called. It calls the script `./configure` with suitable options.

Different platforms are pre-configured. For program development and testing we recommend the gfortran compiler. For operational production code the NEC compiler (SX9) and the sun compiler (hpc-Linux-cluster) are used.

For these platforms the configuration steps are in detail:

1. Within `3dvar/`, edit the header of `./Make` (as suggested by the comments). `./Make` is pre-configured for the gfortran compiler. A different script (`Make-sx9`) is provided for the NEC SX9.

2. Make sure to acess the correct compiler. For program development on hpc we use a non-standard compiler. Create the following symbolic link in your bin-directory to access that compiler:
   ```
   cd  /bin
   ln -s /e/uhome/hanlauf/X86_64/lib/gcc-4.3/bin/gfortran
   ```

3. Call `./Make` :
   ```
   cd 3dvar/
   ./Make
   ```

4. call `make` (or `make -j8` for parallel compilation on 8 processors) to compile.

Library paths and compiler options are specified in `config/mh-`*platform*. If your platform is not supported yet by respective entries in `Make` and `config/mh-`*platform* these files must be modified accordingly.

For temporary changes the top level Makefile (`3dvar/Makefile`) may be changed, which is restored by(`.Make`) .

A HTML-reference of the source files (`3dvar/html/index.html`) is created by typing `make index` .

12.3 External libraries

For I/O purposes and parallelization the 3D-Var package requires the following external libraries:

GRIB
: The GRIB library is used to read and write the gridded model data. The CGRIBEX library of the Max-Planck Institute for Meteorology is used. This library uses the same interface as the GRIBEX and PBIO routines from ECMWF, but can handle the GME icosahedral grid. This library supports GRIB version 1. In the future GRIB version 2 will be handled using a different library (ECMWF GRIB API)

NetCDF
: The NetCDF library is currently used to read (and eventually write back feedback files) for observational data which is not coded in BUFR format (currently RTTOV 1D-Var data).

MPI (Message Passing Interface)
The parallelization is based on the Message Passing Interface. Depending on the platform, it may be necessary to link explicitely to the MPI library (for instance openMPI), as long as the the library is not implicitly linked by the compiler or program is not compiled for a single processor version (compiler option -DNOMPI).

The following libraries are optional and may be enabled/disabled by respective options in the `Make` script:

BUFR
The DWD BUFRX library was used to read the observational data. Currently the observations are converted to NetCDF files (by bufrx2netcdf, also from the BUFRX package) and then read by the NetCDF routines. Thus linking to the BUFRX library is obsolete and currently optional.

GRIP API Currently we upgrade from GRIB version 1 to GRIB version 2. The **GRIB API** library supports both versions and will replace the **CGRIBEX** library in the future.

The paths of the NetCDF- and MPI-libraries are set in the configuration files `3dvar/config/mh-`*platform*.

The content of the following external libraries are provided within the source code tree in order faciliate moving to other platforms and testing. Thus these sources may be compiled and are provided as a library from the source code. Alternatively the respective libraries may be linked externally:

BLAS, **LAPACK**
Linear algebra routines (Directory 3dvar/blas, 3dvar/lapack). For many platforms optimised external libraries exist and should be used.

RTTOV7
Fast radiative transfer observation operator version 7 developed by the **NWP SAF** (3D-Var Directory 3dvar/rttov7). This version was used in the past and will become obsolete in the future.

RTTOV10
Fast radiative transfer observation operator version 9 developed by the **NWP SAF** (3D-Var Directory 3dvar/rttov10).

12.4 DWD VCS

For the generation of operational binaries the assimilation code is also maintained in the DWD VCS.

Recipe to access the VCS:

```
setenv ADM [/e]/rhome/for0adm/vcscmd
mkdir ..../3dvar
cd    ..../3dvar
$ADM/workbench 3dvar
```

doku: $ADM/html/index.html

Three scripts handle the transfer of source files from CVS to VCS and back:
3dvar/scripts/vcs-from-cvs
3dvar/scripts/vcs-edit
3dvar/scripts/vcs-to-cvs

Chapter 13

How to start the DA software

After the succesful compilation (Chapter 12), the executable `var3d` is found in the directory `3dvar/build/`*platform*`/bin/var3d`, or shorter *var3d-path*`/var3d`. It is the central program to be executed.

13.1 Starting `var3d`

Section 13.1.1 describes the execution commands for `var3d`. The namelist `/RUN/` is fundamentally needed, as it sets up the different methods. It is documented in Section 16.1 and needs to be set up in any case.

13.1.1 Execution commands

`var3d` is compiled against the parallel computing library of the *Message Passing Interface* (MPI) (cf. Chapter 12.3) which is also used for the execution. The following commands may be used to run the executable:

Linux: *mpirun-path*`/mpirun -np` *n* *var3d-path*`/var3d`
Here *n* is the number of processors invoked which should be equal to the product of `nproc1` times `nproc2` specified in the namelist-group `/RUN/`.

IBM (cos5): llsubmit LL
In the run script LL, the number of nodes times number of processes per node again must match the values of the namelist parameters.

13.1.2 Starting Parameters

In general, it makes sense to start working from an existing set of namelists, like a template or one that has been used by somebody else succesfully – with the same version of `var3d`, as parameters may change!

The namelist `/RUN/` sets the basic parameters for the execution of `var3d`. The following settings in the namelist are the most fundamental to set:

- `method` selects the assimilation algorithm (Chapter 14)
- `model` selects the forecast model
- `nproc1` times `nproc2` sets the number of parallel processors
- `run_type` determines whether the assimilation is performed or the program is only tested
- `time_ana` sets the analysis point-in-time
- `time_ref` delivers the reference point-in-time from which the background forecast had been started
- `data`, `iopath`, `input`, `output`, `obsinput`, `aux` set the input and output-directories for the files
- `center` and `subcenter` are written into the GRIB-records (to match the WMO-GRIB-tables)

Chapter 14

DA Software: Perform the Assimilation Methods

The `var3d` software offers five wmethods of data assimilation to be performed:

Section 14.1
3dVar for global model (GME/ICON)

Section 14.2
LETKF for global model (GME/ICON)

Section 14.3
generation of global ensemble (GME/ICON)

Section 14.4
LETKF for local model (COnsortium for Small-scale MOdelling (COSMO)-KENDA)

Section 14.5
VarEnKF for global model (GME/ICON)

The following subsections describe the basic settings of how these methods are called, which namelists they use, what algorithms they perform, and their I/O.

When namelists and their parameters are mentioned, the user is *highly advised to look into the descriptive tables of the respective namelists* (Chapter 16)

As soon as testcases and template-namelists exist for these methods, descriptions need to be copied here!

With everyone of these algorithms, the assimilation computation is performed at a specific point in time t_k. The execution of this computation is described here – the *cycled data assimilation* is described in Chapter 15.

14.1 3dVar for global model

The 3dVar algorithm is the basis of the `var3d` software. It is applicable to the global model GME and the global version of ICON.

The minimization algorithm and the background error covariance model for the **B**-matrix of 3dVar are documented in Chapter 7.

14.1.1 3dVar – Settings

The following settings activate 3dVar:

Namelist `/RUN/`:

- `method = 'PSAS'`
- `model = 'GME'` or `'ICON'`
- `run_type = 2`

14.1.2 3dVar – Namelists

This section provides a description of the namelists that are necessary for the usage of 3dVar algorithm.

Processing of observations

`var3d` reads in the observations and applies the observation operators for global models that are documented in Section 6.1.

General data selection parameters as described in Section 17 may be specified globally or individually for the different report types (`TEMP, SYNOP, TOVS, ...`) in namelist `/REPORT/`*. More complex or specific rules can be set up in `/RULES/`*. Some other observation processing parameters are specified in `/OBSERVATIONS/`. Parameters applicable to specific observation types only may be set in one of the namelist groups `/SYNOP_OBS/`, `/TEMP_OBS/`, `/AMV_OBS/`, `/AIREP_OBS/`, `/TOVS_OBS/`, or `/GPS_RO/` and `/RO_VDA/`*. Name and locations of blacklisting files are specified in namelist `/BLACKLIST/`*. Artificial observations may be generated using namelist `/DEF_OBS_NML/`*.

Modeling of background errors

Documented in Section 7.2, parameters of the background error covariance model are specified in `/PSCMODEL/` (correlation function parameters) and `/CNTRLVAR/` (control variable transformations). Details for the humidity analysis are set in `/HUM_ANA/`. The new wavelet based **B**-model is specified in `/BG_ERROR_OPERATOR/`. Observation error correlation model parameters are specified in `/OBSERR/`. Parameters of the Variational Quality Control algorithm are set in `/VARQC/`. Parameters for the analysis error calculation are set in `/ANAERROR/`.

Solver parameters

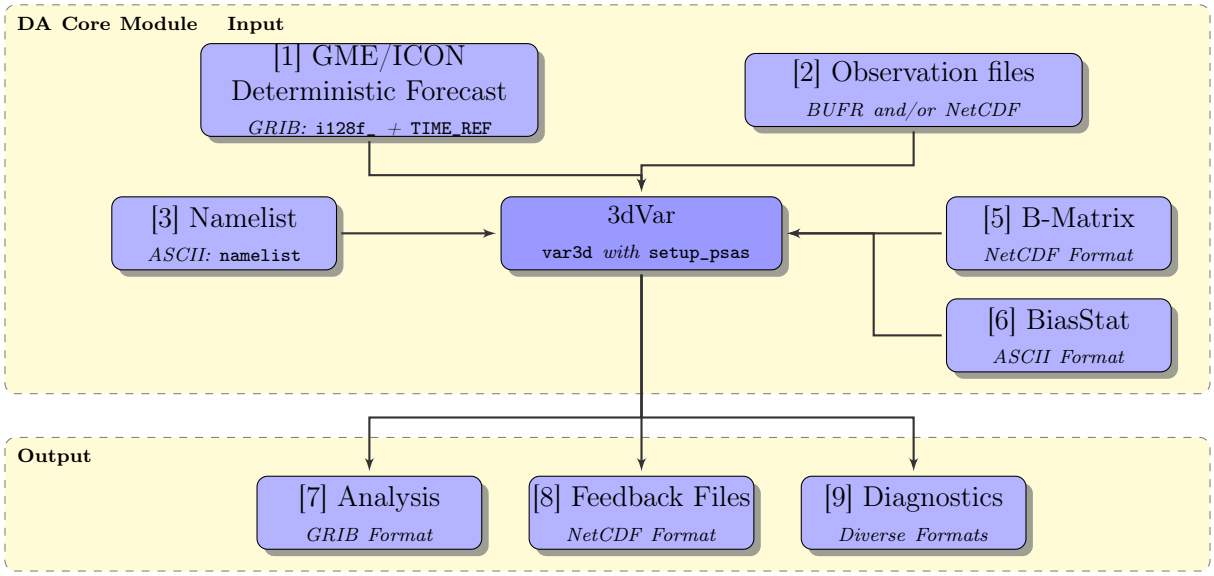
Parameters of the optimization algorithm (number of iterations, convergence criteria) are specified in namelist `/PSAS/`. Here also parts of the program may be switched off (analyses error estimation, post multiplication) and additional output may be invoked.

Debugging and monitoring

Monitoring of the convergence process is controlled in `/PSAS_MONITOR/`. Tests of the consistency of tangent tangent linear operators with their nonlinear counterparts is invoked in `/TEST_OBS_OPR/`.

14.1.3 3dVar – Input/Output

Here the input and output of one 3dVar analysis computation at analysis time t_k. Figure 14.1 provides a flowchart of the I/O concept, described in the lists below.



3dVar-one-assim figure is incomplete!

Figure 14.1: The figure displays the data processing concept of `var3d` in the 3dVar mode for one analysis computation at analysis time t_k (`TIME_ANA`). The previous analysis had been at t_{k-1} (`TIME_REF`).

3dVar Input

The following list describes the input of 3dVar, following Figure 14.1:

Missing: Description of 3dVar input. Could be derived from the testcase in Section 14.1.4

3dVar Output

The following list describes the output of 3dVar

Missing: Description of 3dVar output. Could be derived from the testcase in Section 14.1.4

14.1.4 3dVar – Testcase

In the sub-directory `3dvar/run`, files for running a 3dvar test-case are provided:

`namelist-ref13`:	namelist file for program steering
`MPIRUN-gfortran`:	script to run the 3dvar interactively
`D13`	script to compare the output with a reference data set
`psas.info-ref13`:	reference data set

Perform the following steps to run the test-case:

1. Copy the steering file `namelist-ref13` to the standard input file `namelist`:

   ```
   cp namelist-ref13 namelist
   ```

2. Provide an output directory for the test-case (create a directory or provide a symbolic link):

   ```
   mkdir output-ref13
   ```

3. Run the test-case on 4 processors, redirect the output to file `OUT`:

   ```
   ./MPIRUN-gfortran 4 | tee OUT
   ```

4. Compare the observation output of the program with a reference data set:

   ```
   ./D13
   ```

A description of the standard program output (file `OUT`) is provided in Section 21.

A description on the observation output (`psas.info`, feedback file in ASCII format) is provided in Section 20.2.

Input data

Constant data

Files `btfcoef_f`, `clerr_f`, `fgerr_f` and `rszcoef_f` with climatological background error coefficients and `ww15mgh.grd` with worldwide 15 minute gridded geoid heights (required for the GPS ray-tracing operator) are provided in the `3dvar/data/` directory from the CVS repository. These files are stored in compressed format so that decompression (`gzip -d *.gz`) is required prior to the first model run.

The RTTOV package needs coefficient files located in the directory there the analysis code is run (`3dvar/run/`). The script `rttlink` establishes the required symbolic links.

Variable data

The following input files are required for each analysis:

namelist:
Namelist file (cf. section 16) (ASCII), to be provided in the current directory.

*io/input/***i128f_***yyyymmddhh*:
Model forecast starting from the previous analysis (first guess) (GRIB).

*io/input/***i_ga_err_***yyyymmddhh* :
Estimated statistical error of the previous analysis (GRIB).

*io/input/***blk_***yyyymm***0100**:
Blacklist (ASCII).

*io/***mld_file.z**:
Observations (BUFR).

The directories *io*, *input* as well as the date of the previous analysis *yyyymmddhh* are specified in the namelist-group `/RUN/`.

Executing the var3d code

HL: Parts of this Section are used in the new Chapter 13 – when it is finished, this Section can be deleted/altered

The executable is located in `3dvar/build/`*platform*`/bin/var3d`. For testing purposes it is called from the directory `3dvar/run`. The following commands may be used to run the executable:

Linux: MPIRUN *n*
Here *n* is the number of processes invoked which should be equal to the product of `nproc1` times `nproc2` specified in the namelist-group `/RUN/` (cf. section 16).

IBM (cos5): llsubmit LL
In the run script LL, the number of nodes times number of processes per node again must match the values of the namelist parameters.

Output data

For cyclic data assimilation the following output files are written:

n*xxxxxx.xxxxxx***.o**:
standard output (ASCII).

*io/output/***i128a_ga_***yyyymmddhh*:
analysis (GRIB)

*io/output/***i_ga_err_***yyyymmddhh*:
analysis error (GRIB)

For diagnostics and monitoring the following output files are written optionally:

*io/output/***i128a_ga_***yyyymmddhh***_inc**:
analysis increment (GRIB)

*io/output/***i128a_ga_***yyyymmddhh***_incp**:
analysis increment on pressure levels (GRIB)

*io/output/aux/***fg_err.grads**:
analysis error (GRADS)

*io/output/aux/***fg_err.grads.ctl**:
analysis error (GRADS)

*io/output/aux/***nlpcg.info**:
cost function and gradient as a function of iteration (ASCII, cf. Section 20.1)

*io/output/aux/***psas.info**:
observation diagnostics (ASCII, cf. Section 20.2)

*io/output/aux/***spots.ctl**:
diagnostics on observations (GRADS)

*io/output/aux/***spots.dat**:
diagnostics on observations (GRADS)

The directory names *io*, *output*, and *aux*, as well as the date of the previous analysis *yyyymmddhh* are specified in the namelist-group `/RUN/`.

14.2 LETKF for global model

Performing the LETKF-algorithm is an option of the `var3d` software. An assimilation computation using it can be performed a) for an ensemble of the global models GME and ICON, and b) for the local COSMO model. Here case a) is described[1], shortly called *global LETKF*.

The LETKF algorithm is documented in Chapter 9.

In the case of global LETKF, the assimilation is performed in two steps:

1. First, a deterministic 3dVar-analysis (exactly as described in Section 14.1) is computed with a deterministic forecast run in parallel, including first guess checks (Section 8.1) and Variational Quality Control (Section 8.2). The idea behind this is to make use of the sophisticated observation treatment of the 3dVar algorithm. The treated observations are saved internally and handed over to the next step:

2. Secondly, a pure LETKF-analysis is performed with an ensemble of the global model, using the preselected and preprocessed observations of the beforehandedly performed 3dVar analysis.

The following Sections only describe the application of the pure LETKF step.

Missing: Flowcharts for all the 5 methods!

14.2.1 global LETKF – Settings

The following settings activate the LETKF for the global model:

Namelist `/RUN/`:

- `method = 'PSAS+LETKF'`
- `model = 'GME'` or `'ICON'`
- `run_type = 2`

14.2.2 global LETKF – Namelists

This section provides a description of the namelists that are used for the global LETKF.

Processing of observations

The application of observation operators is performed analogously to 3dVar (observation namelists in Section 14.1.2), but now for every member. The settings in the observation namelists are only relevant for the beforehanded 3dVar-computation which provides the LETKF with preselected and preprocessed observations.

[1] case b) is in KENDA Section 14.4

Modeling of background errors

As documented Section 9.1, the background error covariance $\mathbf{P}^b$ is computed using only the ensemble perturbations. As documented in Section 9.2, the namelist `/ENKF/` sets up options for the background error covariances:

- the ensemble size L as parameter `k_enkf`
- the horizontal and vertical covariance localization using the parameters `lh`, `lv_surf`, `lv_top`
- the multiplicative covariance inflation using `rho`, `apply_rho`, and `adap_rho` (the latter with dependent settings)
- variables to be transformed in the analysis (`par_trans`) or passed through from the forecast ensemble (`par_fce`)

Model errors

Using the background error covariance matrix $\mathbf{B}$ of 3dVar as a proxy for *model error*, the namelist `/ENKF/` sets up the amplitude of the added noise (*additive inflation*) with parameter `mf`. It is added either to the background or the analysis ensemble, controlled by `moderr_fc`.

Solver parameters

The LETKF-computation is performed using the algorithm documented in Section 9.1.

The namelist `/ENKF/` sets up:

- the coarse grid number of diamonds n_i as parameter `rni`
- the number of vertical analysis grid levels n_{zr} as parameter `nzr`
- the analysis to be performed on common pressure levels with `fix_plev`

Physical corrections

For the post-processing, `/ENKF/` activates the positivity of humidity using `q_bound` and saturation adjustment using `sat_adj`.

@Andreas and Hendrik: Please check if this makes sense for the global LETKF!

14.2.3 global LETKF – Input/Output

Here the input and output of one global LETKF analysis step at analysis time t_k. Figure 14.2 provides a flowchart of the I/O concept, described in the lists below.

global LETKF Input

The following list describes the input of the global LETKF, following Figure 14.3:

[1] Prior 3dVar analysis

As mentioned above, `var3d` first computes a full deterministic 3dVar-analysis (Section 14.1, Figure 14.1) to preprocess the observations. Then, `var3d` keeps running and passes the proprocessed observations $\mathbf{y}^o$ in the specific fortran datatype containers [3] to the routine `letkf_driver` of `var3d` which applies $H(\mathbf{x}^\ell)$ for all members and then performs the global LETKF.

Description of 3dVar flowchart and I/O is incomplete.

[2] global Forecast Ensemble

An L-sized ensemble of global GME or ICON members has produced forecasts from t_{k-1} (`TIME_REF`) to t_k (`TIME_ANA`). These forecast states (valid at t_k) are provided as GRIB files in the `input` directory in the filename form

`gff` + `TIME_REF` + `.xxx`

where `xxx` is the number of the respective ensemble member, e.g. `001` or `032`.

[3] Preprocessed observations

These observations (value, location, observation error) had been read and preprocessed in [1].

[4] Adaptive Inflation Estimate from t_{k-1}

If the adaptive covariance inflation is activated by the parameter `adap_rho`, the inflation estimate ρ_{k-1} from the previous analysis at t_{k-1} is read from the `input` directory as the binary file

`rho_local.dat`

where `local` refers to the fact that ρ is computed locally for every analysis grid point.

What happens at the very first assimilation where there is no previous rho-estimate?

[5] Namelists

Placed in the run-directory where `var3d` is executed, the ASCII-file

`namelist`

contains the namelist groups that contain the specific settings for the global LETKF (Section 14.2.2 and Chapter 16).

global LETKF Output

The following list describes the output of the global LETKF (all files are saved in the `output` directory), following Figure 14.2:

[6] global Analysis Ensemble

The LETKF produces a L-sized analysis ensemble at t_k as initial conditions for the next GME/ICON-forecast step. These GRIB files are saved in the filename form

`gaf` + `TIME_ANA` + `.xxx`

where `xxx` is the number of the respective ensemble member, e.g. `001` or `032`.

[7] Forecast Ensemble Mean and Spread

Before computing the analysis, the mean and the spread of the forecast ensemble of [1] are saved as
`gff` + `TIME_REF` + `.mean/spread`
in two separate files for `mean` and `spread`.

[8] Analysis Ensemble Mean and Spread

The mean and the spread of the analysis ensemble of [6] are saved as
`gaf` + `TIME_ANA` + `.mean/spread`
in two separate files for `mean` and `spread`.

[9] Adaptive Inflation Estimate at t_k

Depending on [4], the new covariance inflation estimate ρ_k from the current analysis is saved as the binary file
`rho_local.dat`

[10] Output of $\mathbf{W}^a$ matrix

If `gp_mat` is active, the analysis weight matrix $\mathbf{W}^a$ is saved for certain analysis grid points selected by `gp_mat` as the binary file
`matrix.dat.iii`
where `iii` describes the indices of the chosen analysis gridpoints.

Name convention for iii? See letkf_driver!

[11] Diagnostics

Various diagnostics of RMSE and spread and usage of observations are possible.

Missing: Description of LETKF-diagnostics and produced output-files!

[12] Control Output

Running on a parallel computer, `var3d` produces text output to the terminal via `stdout`. This is output typically caught by the queuing system of the cluster and saved into the ASCII-file
`var3d.out`

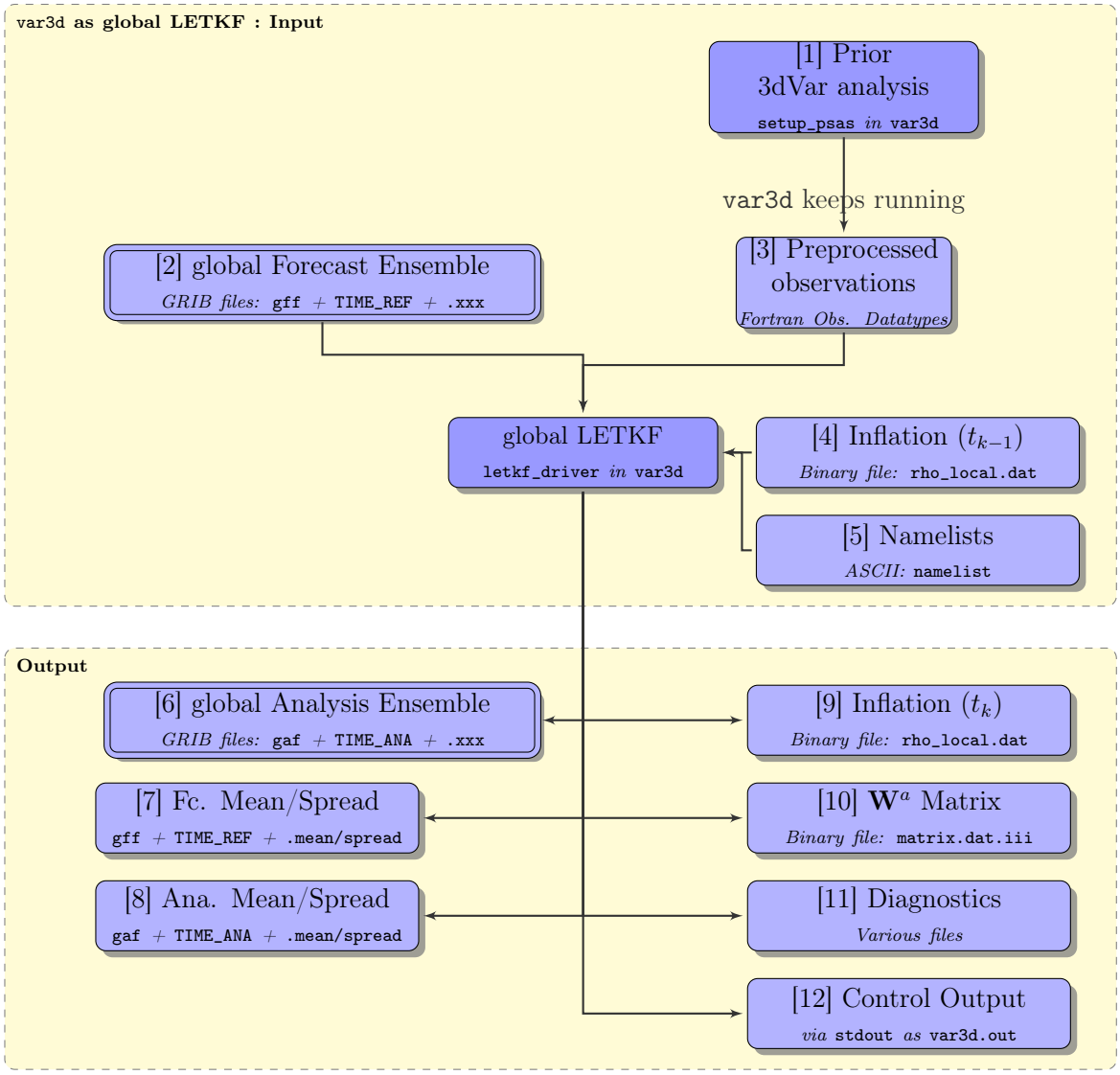


Figure 14.2: The figure displays the data processing concept of `var3d` in the KENDA mode for one analysis computation at analysis time t_k (`TIME_ANA`). The previous analysis had been at t_{k-1} (`TIME_REF`). Boxes with double borders denote an *ensemble of files*.

14.3 generation of global ensemble

Missing: Userguide for generation of global ensemble

14.4 KENDA: LETKF for local model COSMO

This Section describes how the LETKF is set up and performed for the local non-hydrostatic model COSMO. The cycling system which is built around this assimilation is KENDA.

The LETKF algorithm is documented in Chapter 9.

The following Sections only describe the application of the pure LETKF step.

Missing: Flowcharts for all the 5 methods!

14.4.1 KENDA – Settings

The following settings activate KENDA:

Namelist `/RUN/`:

- `method = 'LETKF'`
- `model = 'COSMO'`
- `run_type = 2`

14.4.2 KENDA – Namelists

This section provides a description of the namelists that are used for KENDA:

Processing of observations

The application of observation operators is performed within the members of the COSMO ensemble (Section 6.2).

The communication between the COSMO members and `var3d` is done using feedback files that contain the model equivalents $H(\mathbf{x}^\ell)$ and $\sigma_{obs,fof}$ (Section 6.2). They are selected using `fof_prefix` in the namelist `/ENKF/`.

The namelist `/FOF_INPUT/` allows to manipulate $\sigma_{obs,fof}$ while `/OBSERR/` allows access to the built-in error tables of the module `mo_obs_err`.

The namelist `/RADAR_OBS/` gives control over observations processed by the radar operator in COSMO.

The namelists `/REPORT/` and `/RULES/` give control over the usage of different observation locations, types and single measurements.

Modeling of background errors

As documented Section 9.1, the background error covariance $\mathbf{P}^b$ is computed using only the ensemble perturbations. The namelist `/ENKF/` sets up options for the background error covariances (Section 9.2):

- ensemble size L as parameter `k_enkf`
- horizontal and vertical covariance localization using the parameters `lh`, `lv_surf`, `lv_top`

- adaptive localization using `adap_loc` (with dependent parameters)
- multiplicative covariance inflation using `rho`, `apply_rho`, and `adap_rho` (the latter with dependent parameters)
- variables to be transformed in the analysis (`par_trans`) or passed through from the forecast ensemble (`par_fce`)

Solver parameters

The LETKF-computation is performed using the algorithm documented in Section 9.1.

The namelist `/ENKF/` sets up:

- horizontal coarse grid factor $f_{hor,r}$ as parameter `rf`
- number of vertical analysis grid levels n_{zr} as parameter `nzr`
- (non-)application of analysis weights at COSMO boundaries using `ensbc_weights`
- usage of a deterministic update additional to the ensemble update using `det_run`
- output of $\mathbf{W}^a$-weights on analysis gridpoints using `gp_mat` on indices `gp_ind`

Physical corrections

For the post-processing, `/ENKF/` activates:

- positive humidity using `q_bound`
- saturation adjustment using `sat_adj`
- hydrostatic balancing of analysis increments using `hyd_bal`

14.4.3 KENDA – Input/Output

Here the input and output of one KENDA analysis computation at analysis time t_k. Figure 14.3 provides a flowchart of the I/O concept, described in the lists below.

KENDA Input

The following list describes the input of the LETKF of KENDA, following Figure 14.3:

[1] COSMO Forecast Ensemble

An L-sized ensemble of COSMO members has produced forecasts from t_{k-1} (`TIME_REF`) to t_k (`TIME_ANA`). These forecast states (valid at t_k) are provided as GRIB files in the `input` directory in the filename form
`lff` + `TIME_REF` + `.xxx`
where `xxx` is the number of the respective ensemble member, e.g. `001` or `032`.

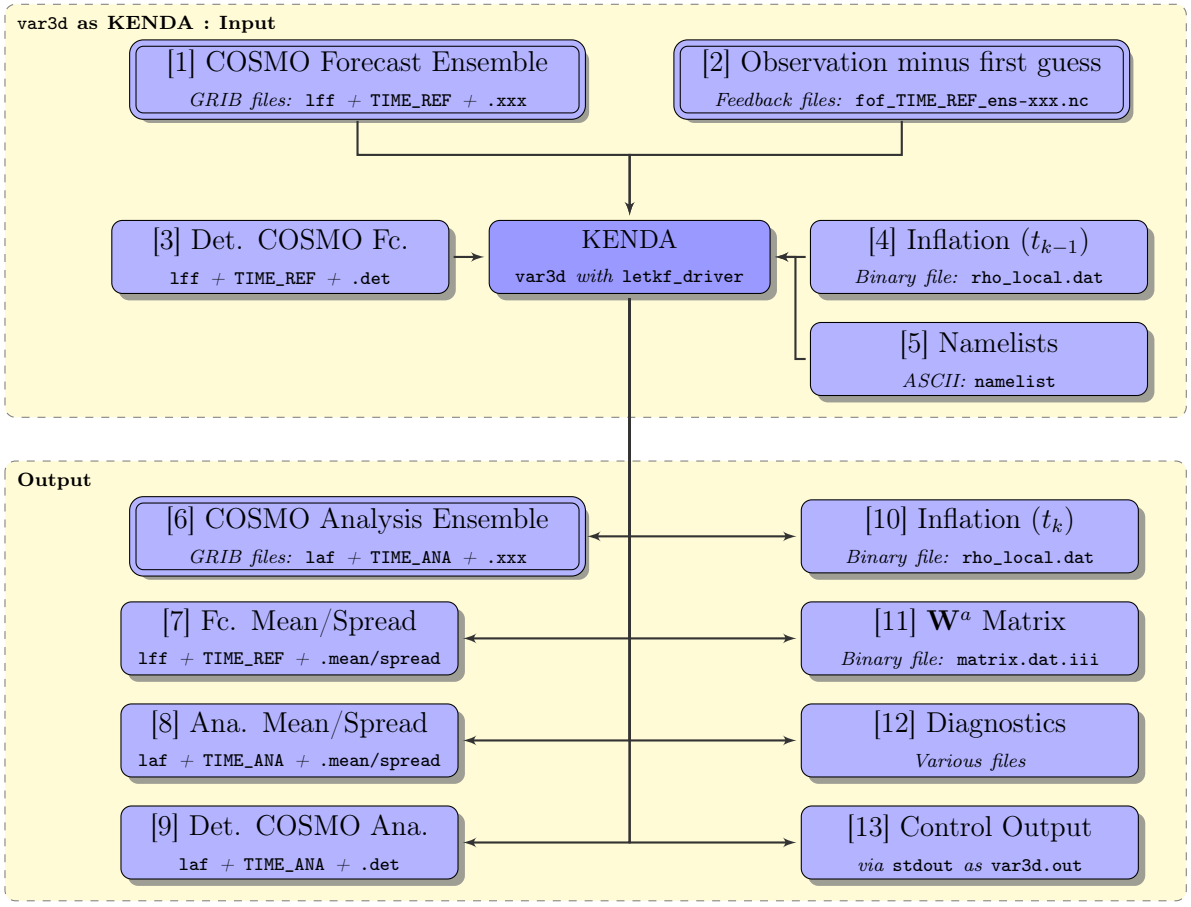


Figure 14.3: The figure displays the data processing concept of `var3d` in the KENDA mode for one analysis computation at analysis time t_k (`TIME_ANA`). The previous analysis had been at t_{k-1} (`TIME_REF`). Boxes with double borders denote an *ensemble of files*.

[2] Observation minus first guess

In every COSMO-member ℓ of [1], the observation operator was applied to get the first guesses $H(\mathbf{x}^\ell)$, mapping onto the space of the observations $\mathbf{y}^o$. The pairs $[H(\mathbf{x}^\ell), \mathbf{y}^o]$ (together with observation time, location, and the observation error from COSMO) are contained in L feedback files in the `input` directory in the filename form

`fof_` + `TIME_REF` + `_ens` + `xxx.nc`

where `xxx` is the number of the respective ensemble member. It is possible that there are feedback files with different prefixes (due to different or external operators). These can be listed in the parameter `fof_prefix` to read all the available data.

[3] Deterministic COSMO Forecast

For the deterministic COSMO-update of KENDA (optional, controlled by `det_update`), the deterministic forecast from t_{k-1} (`TIME_REF`) to t_k (`TIME_ANA`) is provided in the `input` directory in the GRIB-file

`lff` + `TIME_REF` + `.det`

[4] Adaptive Inflation Estimate from t_{k-1}

If the adaptive covariance inflation is activated by the parameter `adap_rho`, the inflation estimate ρ_{k-1} from the previous analysis at t_{k-1} is read from the `input` directory as the binary file

`rho_local.dat`

where `local` refers to the fact that ρ is computed locally for every analysis grid point.

What happens at the very first assimilation where there is no previous rho-estimate?

[5] Namelists

Placed in the run-directory where `var3d` is executed, the ASCII-file

`namelist`

contains the namelist groups that contain the specific settings for KENDA (Section 14.4.2 and Chapter 16).

KENDA Output

The following list describes the output of KENDA (all files are saved in the `output` directory), following Figure 14.3:

[6] COSMO Analysis Ensemble

The LETKF produces a L-sized analysis ensemble at t_k as initial conditions for the next COSMO-forecast step. These GRIB files are saved in the filename form

`laf` + `TIME_ANA` + `.xxx`

where `xxx` is the number of the respective ensemble member, e.g. `001` or `032`.

[7] Forecast Ensemble Mean and Spread

Before computing the analysis, the mean and the spread of the forecast ensemble of [1] are saved as

`lff` + `TIME_REF` + `.mean/spread`

in two separate files for `mean` and `spread`.

[8] Analysis Ensemble Mean and Spread

The mean and the spread of the analysis ensemble of [6] are saved as

`laf` + `TIME_ANA` + `.mean/spread`

in two separate files for `mean` and `spread`.

[9] Deterministic COSMO Analysis

The deterministic analysis at t_k is saved in the GRIB-file

`laf` + `TIME_ANA` + `.det`

[10] Adaptive Inflation Estimate at t_k

Depending on [4], the new covariance inflation estimate ρ_k from the current analysis is saved as the binary file

`rho_local.dat`

[11] Output of $\mathbf{W}^a$ matrix

If `gp_mat` is active, the analysis weight matrix $\mathbf{W}^a$ is saved for certain analysis grid points selected by `gp_mat` as the binary file
`matrix.dat.iii`
where `iii` describes the indices of the chosen analysis gridpoints.

Name convention for iii? See letkf_driver!

[12] Diagnostics

Various diagnostics of RMSE and spread and usage of observations are possible.

Missing: Description of LETKF-diagnostics and produced output-files!

[13] Control Output

Running on a parallel computer, `var3d` produces text output to the terminal via `stdout`. This is output typically caught by the queuing system of the cluster and saved into the ASCII-file
`var3d.out`

14.5 VarEnKF for global model

Chapter 15

Cycled Data Assimilation

As introduced in Chapter 2, data assimilation is performed in a *cycling* manner where an assimilation step is followed by a model forecast step with a forecast time interval Δt_{fc} ranging from 3 hours in the global 3dVar-cycling down to 15 minutes or less for the local KENDA-cycling system.

The following sections are building on the description of the point-in-time analysis computations documented in Chapter 14:

Section 15.1
: cycling for the global 3dVar analyses using deterministic forecasts of the GME or ICON model.

Section 15.2
: cycling for the global LETKF analyses ensembles of the GME or ICON model.

Section 15.3
: cycling for local model LETKF analyses of KENDA using ensembles of the COSMO model.

It should be noted that the cycling descriptions provided here refer to standalone-cycling scripts used at the DWD. The user is free to construct own scripts for data management, namelist templates and binary calls, depending on the architecture of the computational cluster that is used.

15.1 Global 3dVar Cycling

Userguide for 3dVar-Cycling: Complete?

The basic design of the *basic cycling system* for the global 3dVar data assimilation system is shown in Figure 15.1.

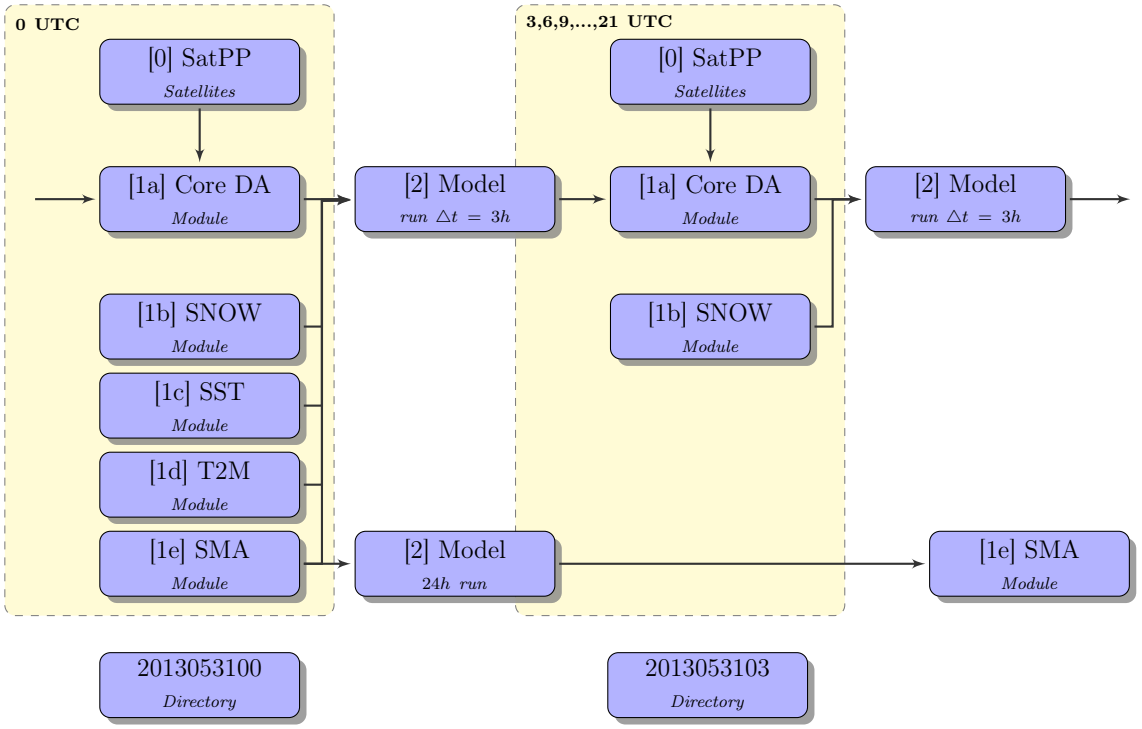


Figure 15.1: The figure displays the design of the basic cycling environment for the global 3dVar data assimilation system. The modules [1b] to [1d] are run only at 0UTC, the *snow analysis* [1a] is run at every cycling time, currently at 3,6,9,... UTC. A 24h model run is needed for the SMA which runs at every 0 UTC.

The idea of the basic cycle is to have a tool for development and debugging of each of its modules. To this end, a careful treatment of the input and output of all modules and its file processing is realized. The basic ideas are best explained by going into a little more detail.

1. All files for each assimilation column are stored in one directory with a name reflecting the analysis date and time. We employ the *name convention* `YYYYMMDDHH` with `YYYY` containing the year in four digits, `MM` the month with two digits, `DD` the day in two digits and `HH` the hour. For faster update rates in KENDA, minutes `mm` and seconds `ss` can also be added.

2. The cycling is managed by a loop which we also refer to as `level 0`, realized by a script `run_cycle`. It takes basic parameters such as the `startdate`, the `cycling interval` and the `enddate` and cycles by adding the cycling interval to the current *analysis date* `ADATE` in each step and carrying out the assimilation for `ADATE`.

3. At each analysis time, the different modules shown in a column of Figure 15.1 for assimilation are called, where we use the order SST [1c], T2M [1d], SMA [1e], Core DA [1a], the SNOW module [1b] and the *model run* [2]. We refer to this succession of modules as *level 1*.

4. The runs generate diverse output, an output directory is generated containing *log files* and *feedback files*, which contain the observations and corresponding model equivalents for further evaluation and verification.

5. Some modules are carried out at every analysis step, for example the snow analysis [1b]. Others as SST [1c], T2M [1d] and SMA [1e] are only run at 0 UTC. Further, the SMA module [1e] needs a run of the model over the past 24 hours as input. This is shown in Figure 15.1 in the bottom line with an arrow from [2] Model to [1e] SMA.

An example of the list of data files when cycling ICON is given as follows:

```
LL                              ... shell script for running job
NAMELIST_ICON_output_atm        ... ICON namelist
NAMELIST_icon_0018              ... ~
an_R02B06.2013050412            ... the core analysis file
bias_AIREP.201305041200         ... bias correction file
bias_RAD.201305041200           ... ~
end_LL                          ... job output
end_LL.icon                     ... ~
er_R02B06.2013050412            ... B Matrix for 3dVar
fc_R02B06.2013050415            ... first guess
fc_R02B06_ll.2013050415         ... lat-long version of first guess
icon_master.namelist            ... ?
ini_R02B06_ll.2013050412        ... ?
namelist                        ... 3dVar namelist
nml.atmo.log                    ... log for namelist
run_out.snw                     ... log for snow analysis
snw_R02B06.2013050412           ... snow analysis
stdout_snow                     ... job output snow analysis
```

The above files are generated for the date `2013050412`, i.e. for 12 UTC on the 4th of May, 2013. In this case only snow analysis files `snw_R02B06.2013050412` are available, compare Figure 15.1. Forecast fields can be identified by `fc` (`ll` for lat-long output grid), analysis fields by `an`, several further auxilliary files will be discussed later.

The *basic cycle* environment uses the following software concept:

1. The scripting environment is contained in some directory, e.g. *DA4ICON*, which contains subdirectories and files

```
README              ... generic information
bin                 ... binaries
config              ... functions and shell variable setup
data                ... constants, external parameters, climate fields
initdata            ... initial data for starting cycling
```

```
mld                 ... observations stored locally
scripts             ... scripts for cycling
status              ... status files while cycling
templates           ... namelist templates for cycling
```

2. The different scripts run on different loop levels are contained in `scripts`. Scripts include for example

```
cat_smainput
copy_feedback
getconst_sma
rerun_icon_fg
run_3dvar
run_ICON_climate
run_ICON_coldstart
run_cycle
run_icon
run_icon_sfc2mga
run_icon_sma
run_icon_snow
run_icon_sst
```

   where the modules shown in Figure 15.1 can be identified by their short names `sma`, `snow` and `sst`. Here, `3dvar` is the core DA module for the ICON deterministic analysis.

3. The scripts carry out the following basic steps.

   (a) They organize the input files by setting up a list of shell variables, unzip or untar observation files if necessary, copy and cat input files where needed, and generate (if needed) output directories.

   (b) They copy and fill the namelists for each module under consideration using the *namelist templates* which are provided in the directory `templates` and the shell variables set up before. Here, a standard approach with *identifiers* in the templates is used, where identifiers of variables are replaced by the actual variable values. The replacement is done by the unix `sed` editor and is carried out by shell-functions from the `config` directory.

   (c) Then, the actual module is run.

   (d) Afterwords, the output files are stored and cleaning up takes place.

4. Every experiment generates unique work directories, e.g. at `/e/gtmp/$USER`, which are used by the different scripts. They will be deleted when the same part of the cycle runs again in a subsequent step. Output data is moved to the output directory `YYYYMMDDHH` as described above.

15.2 global LETKF cycling

Missing: Userguide for global LETKF cycling

15.3 KENDA-LETKF cycling

Userguide for KENDA Cycling: Complete?

The Core Module for Ensemble Kalman Filter Analyses is the program `var3d` in the EnKF-mode. The settings and features of the EnKF are controlled by the namelist `/ENKF/` (see link for the full description!).

This Chapter describes how the LETKF of the COSMO-KENDA system is set up and used.

15.3.1 Workflow of KENDA

Figure 9.2 displays the cycling as it is implemented in COSMO-KENDA:

1. **Analysis Step (time-index $j = j$)**

   - A file-set of an initial background ensemble $\mathbf{x}^{b(i)}$ (`lff`, GRIB) is provided together with an ensemble of initial *feedback files* (`fof`, NetCDF) which contain the observations $\mathbf{y}^o$ and the first guesses of the members $H(\mathbf{x}^{b(i)})$. In the 0-th cycle, these need to be provided externally. Note that $\mathbf{y}^o$ is redundantly contained in every feedback-file.
   - $\mathbf{x}^{b(i)}$ and $[\mathbf{y}^o, H(\mathbf{x}^{b(i)})]$ are read in by the LETKF (which is the program `var3d` in the LETKF-mode) which computes and saves the analysis-ensemble $\mathbf{x}^{a(i)}$ (`laf`, GRIB).

2. **Forecast Step (cycled time-index $j = j + 1$)**

   - An ensemble of COSMO-DE reads in the recently produced $\mathbf{x}^{a(i)}$ together with the externally provided boundary conditions x^{bound} (`lbf`, GRIB) and the observations $\mathbf{y}^o$ (NetCDF).
   - COSMO computes an ensemble of background forecasts $\mathbf{x}^{b(i)}$ for the next analysis step. The observation operator H is contained in COSMO and maps the observations onto the forecast (per member), producing the feedback files.

3. Next analysis time. Back to Analysis Step 1.

15.3.2 Ensemble data assimilation

Ensemble data assimilation EDA for the global model is following the lines of the deterministic global cycle (Chapter 15. In principle, Figure 15.1 also displays the concept of *basic cycling* for the ensemble system. For Ensemble data assimilation

- the module `[1a] Core DA` consists of the EnKF/LETKF or a *Hybrid Method* VarEnKF; for its scientific description we refer to Section 4.6.

- The model run `[2] Model` over $\triangle t$ consists of L independent model runs, where L is the number of ensemble members under consideration, which is for example $L = 40$ for the EnKF or $L = 41$ for the VarEnKF. Note that for the VarEnKF there are $L - 1$ members with *lower* resolution which constitute the EnKF part and there is one member with *higher* resolution, on which the variational component of the method is built.

The scripting environment is built up analogously to the description on Page 201. As described in Points 1 and 2, the scripts are contained in a central directory `scripts`. Data are stored in the `data` directory. Every run generates unique *working directories* in a temporary file space (compare Point 4), where the input and output of both the different data assimilation modules and the model run are read and written, for example

```
/e/gtmp/$USER/working_directory/modelrun_001
.....
/e/gtmp/$USER/working_directory/modelrun_040
```

for running the first to fourtieth ensemble member.

15.3.3 KENDA Basic Cycling

The cycling environment for the high-resolution regional model with the KENDA for the COSMO model system is basically following the philosophy which we use for the global ICON 3dVar and for the global ICON VarEnKF.

For a *regional* model like COSMO the cycling is slightly more complicated due to the fact that we need to provide and manage the boundary conditions which are needed for each *model run*. In particular, for the ensemble data assimilation we need to provide an *ensemble of boundary conditions* for the short-range forecast runs of the model.

1. The **concept** of the *KENDA basic cycle* employs different directories

   ```
   bcdata              ... provides the global BC data (gfff fieles)
   bcfata_forecast     ... ~ for forecast
   data                ... to store files for later use
   data_forecast       ... ~ for forecast
   feedback            ... collects all feedback files
   obs                 ... provides the observations as tar-files
   rundir              ... all working directories
   ```

   The basic idea here is that the individual module runs use a *working directory* in `rundir` to read and store their data. Then, the managing script moves files which are to be stored to the `data` directories and removes whatever is not meant to be kept.

   COSMO needs the initial conditions (names as `laf`-files) and the boundary conditions (with name convention `lbff`-files).

2. Initially, the **boundary data** are stored in subdirectories of `bcdata` as `giff-` or `gfff`-files, respectively. For the boundary conditions it is important to note that traditionally, the fields have been saved in a 3h period. Thus, for a particular analysis date, we need to determine the corresponding boundary conditions date `DATE_BD`, which is the starting time of the 3h period of the current analysis date. The boundary data are stored in subdirectories with name convention `YYYYMMDDHHmmss`.

   These files contain a selected region of a global model which is chosen to be able to interpolate these low-resolution boundary condition files to the high-resolution COSMO model grid. The conversion is carried out by the module `INT2LM` (compare `[0] INT2LM` in Figure 15.2), which generates the actual boundary files on the COSMO grid, which are much larger files with the `lbff` name convention. This conversion is done on the fly before the files are needed for an individual COSMO run.

3. For the individual *model runs* (see `[1] Model` in Figure 15.2) we employ a central *working directory* `cosmo` in `rundir`, which contains

```
det                ... directory for the deterministic run
ens001             ... for ensemble member 001
ens002             ... for ensemble member 002
...                ...
ens040             ... for ensemble member 040
```

   and further files which contain

   - the *observations* in NetCDF formats
   - the blacklist in ASCII format.

   for example:

```
blklsttmp               ... blacklist file
cdfin_acars.nc          ...
cdfin_amdar.nc          ... AMDAR airplane data
cdfin_buoy.nc           ... Buoys measurements
cdfin_radar_vad.nc      ... Radar measurements
cdfin_rass.nc           ...
cdfin_ship.nc           ...
cdfin_synop.nc          ...
cdfin_temp.nc           ...
cdfin_wprof.nc
```

   Note that these files and the content of the directories has temporary meaning for one analysis time only and is overwritten in each analysis step.

4. A **satellite preprocessing step** is needed for all KENDA versions where satellite data are assimilated. Here, the preprocessing of data is carried out by a module `SatPP`, compare `[2] SatPP` in Figure 15.2.

5. The **EnKF module**, compare `[3a] EnKF` in Figure 15.2, uses a central *working directory* `letkf` in the `rundir` directory for its input and output. The working directory has subdirectories `input` and `output`, which are prepared by the `run_letkf` script. All job output and log files are written into the working directory, here a name convention `LETKF.eID` or `LETKF.oID` with some job scheduling number ID is used.

6. **Storage.** Due to limited storage space, usually one has to be very careful with storing data during cycling. For the *KENDA Basic Cycling* we store data in the `data` directory in subdirectories following the name convention `YYYYMMDDHHmmss` described in Item 1. of Section 15.1. This could easily be replaced by storage in some *data base*.

The scripting of KENDA basic cycle provides file input via *symbolic links* (soft links under UNIX/LINUX), which avoids copying a large quantity of data.

Currently, the surface analysis modules [3b] SNOW, [3c] SST, [3d] T2M Analysis and [3e] SMA (soil moisture analysis) are not yet included in the basic cycle, but are planned for the near future.

The basic design of the *KENDA basic cycle* for is shown in Figure 15.2.

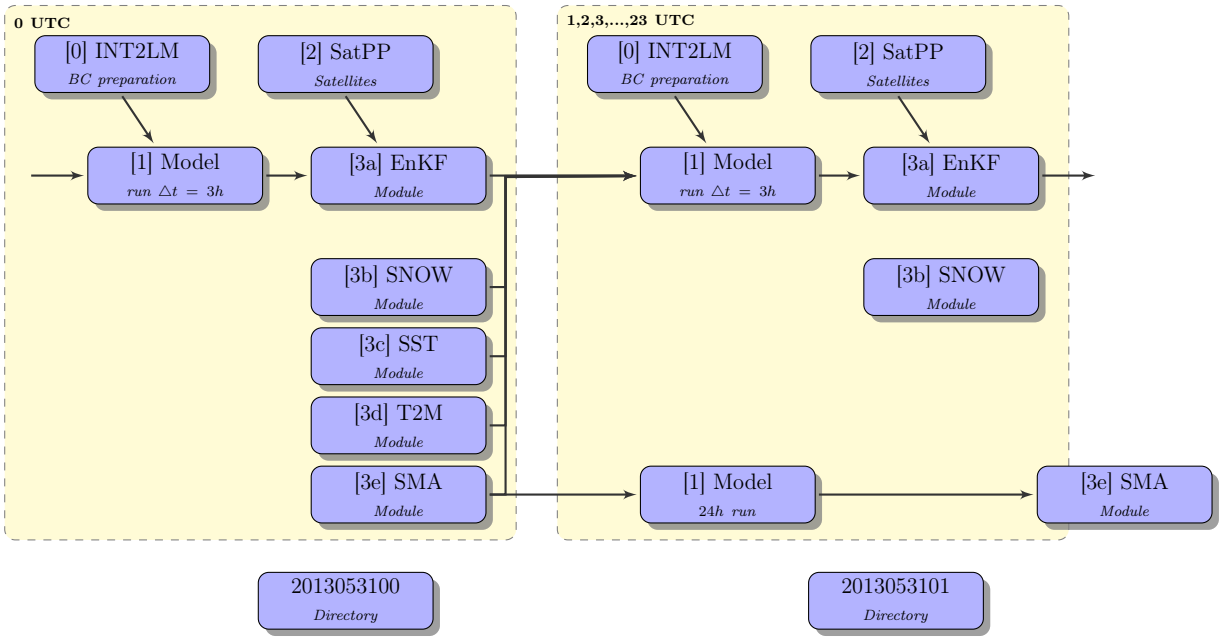


Figure 15.2: The figure displays the design of the KENDA basic cycling environment. INT2LM [0] prepares boundary condition for the model runs [1], which are providing input in the form of `feedback files` for the EnKF [3a]. The use of satellite data needs the preprocessing module SatPP [2]. The surface analysis modules [3b] to [3e] are carried out in analogy to the global data assimilation.

Chapter 16

Namelist Groups

This Chapter serves a reference to fortran namelists that are used by the binary `var3d` and for its various modules and assimilation methods.

The namelist input must be provided in a file `'namelist'` in the directory there the `var3d` program is started. As an extension to the standard namelists may appear in arbitrary order, may be omitted, or may be called repeatedly. The latter feature is useful if a similar set of parameters has to be specified for a number of targets (for instance different observation types). The implementation of namelist groups which make use of this extensions in a parallel environment is described in Section 22.1.5. An overview of the namelist groups is given below. A * indicates that the namelist may be called repeatedly:

Note: Where the namelists are described here, mostly the code-copy of the namelist-definitions is OLD! Need to check and copy new code for all namelist!

16.1 /RUN/ (Job control parameters)

`mo_run_params.f90`

```
!-------------------
! general parameters
!-------------------DA3
character(len=10)  :: method       = 'PSAS' ! 'PSAS'  (default 3dvar mode)
                                            ! '3DVAR' (currently not implemented)
                                            ! 'PSAS+LETKF'
                                            ! 'LETKF'
                                            ! 'ENVAR', 'ENVAR+LETKF', 'VARENKF'
                                            ! 'GMESTAT', 'MEC'
character(len=8)   :: model        = 'ICON' ! or 'COSMO', 'IFS','GME'
integer            :: nproc1       = -1     ! number of PEs in direction 1
integer            :: nproc2       = -1     ! number of PEs in direction 2
integer            :: nproc_repro  = -1     ! fict. PEs for repeatable runs
logical            :: barrier      = .true. ! call mpibarrier for diagnostics
```

```
logical            :: check_sync   = .false.! check syncronisation(stop_time)
integer            :: p_readgrib   = -1     ! PE used to read GRIB
integer            :: p_readbufr   = -1     ! PE used to read BUFR
integer            :: npe_read_obs = -1     ! number of PEs to read obsv
integer            :: nex          =  0     ! experiment number
integer            :: run_type     =  3     ! haupt=0, (vor=1), ass=2, test=3
character(len=8)   :: runtype      =  'forecast' ! set to '' to read analyses
integer            :: expseed      =  0     ! nondefault random number seed
character(len=256) :: read_fields  = ' '    ! fields to read from forecast
character(len=256) :: sst_fields   = ' '    ! read from SST analysis (00UTC)
character(len=256) :: opt_fields   = ' '    ! read optionally
character(len=256) :: urun_fields  = ' '    ! unspecified runtype
character(len=256) :: pass_fields  = ' '    ! pass to analysis (relabel)
character(len=128) :: dealloc_fields= ' '   ! do not write to analysis
character(len=128) :: gribout_24bit = ' '   ! fields to write with 24 bits
integer            :: interp_strato= 0      ! interpolate stratosphere
logical            :: abort_strato =.false. ! abort if file is not present
logical            :: read_obs_first=.false.! read obs. before background
!------
! dates
!------
type(t_time) ,save :: ana_time              ! analysis time
type(t_time) ,save :: ana_date              ! analysis date (0 UTC)
type(t_time) ,save :: fc_ref_time           ! reference time (forecast start)
type(t_time) ,save :: fc_time               ! forecast interval
integer            :: fc_hours       = -1   ! forecast interval (hours)
integer            :: yyyymmddhh_ana =  0   ! analysis time
integer            :: yyyymmddhh_ref =  0   ! time of forecast start
integer(i8)        :: time_ana       =  0   ! analysis time  (ccccmmddhhmmss)
integer(i8)        :: time_ref       =  0   ! forecast start (ccccmmddhhmmss)
integer(i8)        :: runtime        =  0   ! program run time YYYYMMDDHHMM
!----------------
! directory paths
!----------------
character(len=256) :: data           = ''   ! constant input data
character(len=256) :: iopath         = ''   ! path for input/output dirs
character(len=256) :: input          = ''   ! input             directory
character(len=256) :: output         = ''   ! output            directory
character(len=256) :: obsinput       = ''   ! observation input directory
character(len=256) :: aux            = ''   ! additional output directory
!-----------------
! input file names
!-----------------
character(len=128) :: fg_file        = ''   ! forecast file
character(len=128) :: blacklists(5)  = ''   ! blacklists
```

```
character(len=128) :: oldanerr_file  = ''   ! analysis error
character(len=128) :: file_strato    = ''   ! stratosphere file name
character(len=128) :: invar_det      = ''   ! invariant fields
character(len=128) :: grid_file      = ''   ! grid metadata (ICON)
!----------------------------
! output file characteristics
!----------------------------
character(len=128) :: ana_file       = ''   ! analysis
character(len=128) :: ana_err_file   = ''   ! analysis error
character(len=128) :: out_file       = ''   ! protocol
character(len=128) :: fdbk_basename  = ''   ! feedback file
character(len=128) :: ready_det      = ''   ! ready file, determ. analysis
integer            :: grib_edition   =  2   ! GRIB edition to write (-1,1,2)
integer            :: grib_library   =  2   ! API: 1=(C)GRIBEX  2=GRIB2-API
!---------------------------------
! identification of center, process
!---------------------------------
integer            :: center       =  78 ! DWD
integer            :: subcenter    = 255 ! none
integer            :: proc_ana_err =   0 ! process: analysis error
integer            :: proc_ana     =   0 ! process: analysis
integer            :: proc_ana_ens =   0 ! process: analysis ensemble
integer            :: range_ana    =   0 ! time range parameter for analysis
integer            :: ensemble_id  =   1 ! ensemble id for GRIB output
!-----------------------------------------------------
! switch behaviour of analysis system at a given time
!-----------------------------------------------------
integer            :: ga3_biasc_rad   = 2099010100 ! radiance online biascor.
integer            :: ga3_biasc_airep = 2013021409 ! aircraft bias correction
integer            :: ga3_biasc_synop = 2099010100 ! SYNOP    bias correction
integer            :: ga3_biasc_gpsgb = 2099010100 ! GPSGB    bias correction
integer            :: ga3_biasc_scatt = 2099010100 ! SCATT    bias correction
type(t_time) ,save :: date_biasc_rad
type(t_time) ,save :: date_biasc_airep
type(t_time) ,save :: date_biasc_synop
type(t_time) ,save :: date_biasc_gpsgb
integer            :: flag_biasc_rad   = -1        ! radiance online biascor.
integer            :: flag_biasc_airep = -1        ! aircraft bias correction
integer            :: flag_biasc_synop = -1        ! SYNOP    bias correction
integer            :: flag_biasc_gpsgb = -1        ! GPSGB    bias correction
integer            :: flag_biasc_scatt = -1        ! SCATT    bias correction
```

Missing: Description-Table for Namelist `/RUN/`

16.2 /REPORT/ (selection of observations)

This namelist is defined in subroutine (module mo_obs_tables) :

```
character(len=12) :: check
character(len=12) :: type
character(len=12) :: use
integer           :: max_proc
integer           :: max_act
integer           :: ni
real(wp)          :: height_t
real(wp)          :: height_b
real(wp)          :: lat_nb
real(wp)          :: lat_sb
integer           :: time_b
integer           :: time_e
real(wp)          :: min_dist  ! min. distance between observations [km]
real(wp)          :: sgm_dism  ! first guess check bound: dismiss      data
real(wp)          :: sgm_pass  ! first guess check bound: passive      data
real(wp)          :: sgm_act0  ! first guess check bound: inactive     data
real(wp)          :: sgm_act0i ! first guess check bound: vqcontrolled data
real(wp)          :: sgm_act   ! first guess check bound: vqcontrolled data
real(wp)          :: sgm_act1i ! first guess check bound: vqcontrolled data

namelist /REPORT/ check, type, use, max_act, max_proc, ni, min_dist,  &
                  height_t, height_b, lat_nb, lat_sb, time_b, time_e, &
                  sgm_dism, sgm_pass, sgm_act0, sgm_act0i, sgm_act,   &
                  sgm_act1i
```

This namelist group may appear several times in the namelist file. The parameters specified apply either to all observation types or to specific types only if the variable **type** is specified. If the variable **check** is given, **use** specifies the effect of this check, otherwise it determines the status of the observations if no check is flagged. Default values are defined in the table `rept_char` in module .

Examples:

The following sequence will dismiss all observation types besides SYNOPs and TEMPs for assimilation. SATOBs will be monitored.

```
&REPORT               use='DISMISS' \
&REPORT type='SYNOP' use='ACTIVE'  \
&REPORT type='TEMP'  use='ACTIVE'  \
&REPORT type='SATOB' use='PASSIVE' \
```

The following sequence specifies that the bound of the first guess check which causes a SYNOP to be excluded from the assimilation (and monitored only will be 3 times the (squared) sum of first guess and observational error:

```
&REPORT type='SYNOP' sgm_pass=3. \
```

The following sequence specifies that the time range for SYNOPs used for assimilation is +− one hour. Furthermore the effect of the time range check

type	`character(*)`	Name of a Report Type as defined in Table 17.1. If this variable is present, the parameters given below only apply to this Report Type, otherwise they will apply to all Report Types.
check	`character(*)`	Mnemonic of a check as defined in Table **??**. If this variable is present the **use**-flag only applies to this check (and the given Report Type), otherwise it will apply to all checks.
use	`character(*)`	One of `'dismiss'`, `'active'`, or `'passive'`. This flag determines how to proceed with a report if the condition of one of the checks is fulfilled.
count	`integer`	Maximum number of reports to process. This parameter is provided to enable program development and testing with a small number of observations.
dist_h	`real`	Minimum horizontal distance between two reports of the same type (in km) used in the (old) thinning procedure (Section 8.1.2).
height_t	`real`	Top and ...
height_b	`real`	bottom bound of observations to be used, generally given in hPa.
lat_nb	`real`	Northern and ...
lat_sb	`real`	southern bound of observations to be used, given in degrees.

16.3 /RULES/ (complex data selection)

This namelist is defined in subroutine (module mo_obs_rules):

Missing: Documentation of namelist /RULES/

```
integer           :: type
integer           :: bf_type
integer           :: bf_subt
integer           :: db_kz (nkz)

real(wp)          :: lat (2)
real(wp)          :: lon (2)
logical           :: xlonlat      ! exclude area
real(wp)          :: plim(2)

integer           :: use
integer           :: verb
integer           :: msl
```

```
  type(t_set)       :: t
  type(t_set)       :: gp
  type(t_set)       :: uv
  type(t_set)       :: q
  type(t_set)       :: p
  type(t_set)       :: c (nc)

type t_rules
  character(len=40) :: comment = ''
  integer     :: type          = iud   ! 3D-Var-type
  integer     :: bf_type       = iud   ! BUFR-message-type
  integer     :: bf_subt       = iud   ! BUFR-message-subtype
  integer     :: db_kz (nkz)   = iud   ! Databank-'Kennziffer'

  real(wp)    :: lat (2)       =  (/ rud, -rud /) ! Latitude  bounds
  real(wp)    :: lon (2)       =  (/ rud, -rud /) ! Longitude bounds
  real(wp)    :: plim(2)       =  (/ rud, -rud /) ! Pressure  bounds
  logical     :: xlonlat       =  .false.

  integer     :: use           = iud   ! how to use (USE_NOT, _ASS, _MON)
  integer     :: verb          = iud   ! verbosity flag (diagnostic printout)
  integer     :: msl           = iud   ! reduce pressure to mean sea level

  type(t_set) :: t                     ! temperature specific settings
  type(t_set) :: gp                    ! geopotential
  type(t_set) :: uv                    ! wind
  type(t_set) :: q                     ! humidity
  type(t_set) :: p                     ! pressure
  type(t_set) :: c (nc)                ! channels, constituents ...
end type t_rules

type t_set
  integer     :: use         = iud     ! how to use this parameter
  integer     :: m_rej       = iud     ! for variational quality control
  real(wp)    :: sgm_o       = rud     ! observation error
  real(wp)    :: sgm_fg      = rud     ! first guess check bound
  real(wp)    :: sgm_vq      = rud     ! variational quality control bound
end type t_set


  namelist /RULES/ comment,                                  &
                   type, bf_type, bf_subt, db_kz,            &
                   lat, lon, plim, xlonlat,                  &
                   use, verb, msl, t, gp, uv, q, p, c
```

```
comment = 'added by namelist /RULES/'
```

16.4 /OBSERVATIONS/ (selection of observations)

This namelist is defined in .
Most entries be superseded by namelist /REPORT/ *in the future.*

Missing: Description of namelist /OBSERVATIONS/

.

mo_t_obs.f90

```
!=========
! namelist
!=========
logical           :: read_bufr     = .false. ! read BUFR   files
logical           :: read_NetCDF   = .false. ! read NetCDF files
logical           :: read_cosmo    = .false. ! read COSMO observation files
character(len=64) :: obs_path      = ''      ! path to read observation files
character(len=64) :: bufr_files(63)= ''      ! names of BUFR files
character(len=64) :: obs_files (63)= ''      ! observation file names (NetCDF)
character(len=64) :: fdb_files (63)= ''      ! feedback (input) files
integer           :: bufr_verb     =  0      ! verbose flag for BUFR decoding
logical           :: bufr_pause    = .false.
integer           :: netcdf_verb   =  2      ! verbosity level of NetCDF decoding
logical           :: derive_dbkz   = .false. ! derive DBKZ if not present
logical           :: fix_no_obs    = .true.  ! set F to revert no_obs-check
logical           :: int_vh        = .true.  ! interpolation: 1.vert.,2.hor.
logical           :: int_nn        = .false. ! hor. interp.: nearest neighb.
logical           :: vint_lin_t    = .false. ! linear vert.intp. for temp.
logical           :: vint_lin_z    = .false. ! linear vert.intp. for geop.
logical           :: vint_lin_uv   = .false. ! linear vert.intp. for wind
logical           :: vint_lin_tov  = .false. ! linear vert.intp. for RTTOV
integer           :: nwv_rad       = -1      ! radiance vertical interp. flag
integer           :: ndv_rad       =  3      ! vertical differentiation flag
integer           :: pcc_conv      =  1      ! per cent confidence usage flag
integer           :: corme_conv    = -1      ! correction message handling:
                                             ! -1 NetCDF=off BUFR=off
                                             !  0 NetCDF=on  BUFR=off
                                             !  1 NetCDF=on  BUFR=on
                                             !  2 handle corme, ABCD separately
```

16.5 /BLACKLIST/ (blacklist file name and path)

This namelist is defined in module :

Missing: Description of namelist /BLACKLIST/

```
mo_blacklist.f90

! !NAMELIST: /BLACKLIST/
!
! !DESCRIPTION:
!
! By namelist group {\bf /BLACKLIST/} the path and name of the blacklist
! files are specified by variables {\bf path} and {\bf name},
! respectively. The namelist group may be called repeatedly for
! different blacklisting files. In this case the path defined in
! previous calls remains valid. The logical variable {\bf default}
! may be set to {\bf .false.} to suppress reading of the default
! blacklist (\/io\/input\/blacklist\_yyyymmddhh).
!
! !DEFINITION:
!
  character(len=128) :: file
  character(len=128) :: path
  logical            :: default

  namelist /BLACKLIST/ path, file, default
!
!EOP

!!P.G.U P.G.O P.W.U P.W.O P.T.U P.T.O P.D.U P.D.O

!0010     1  1100     0     0     0     0     0     0     0
!04045    1  1100     0     0     0     0     0     0     0
!0XRA6    1  1100     0     0     0     0     0     0     0

!......   2     0     0    59    10    59    10    59    10
!034BATBA 2     0     0     0     0  1100     0     0     0
!0EVEIEBA 2     0     0     0     0  1100     0     0     0
!ABX...   2     0     0  1100     0     0     0     0     0
!ACHUIEBA 2     0     0     0     0  1100     0     0     0
```

16.6 /DEF_OBS_NML/ (artificial observations)

Missing: Description of namelist /DEF_OBS_NML/

This namelist is defined in subroutine (module mo_obs_nml) :

```
real(wp)         :: lon   ! latitude  [degree]
real(wp)         :: lat   ! longitude [degree]
real(wp)         :: p     ! pressure  [hPa]
character(len=8) :: rtype ! report type
character(len=2) :: otype ! 'h', 't', 'rh', 'u', 'v'
real(wp)         :: value ! gpm   K    1    m/s  m/s
integer          :: nlat  ! number of latitudes
namelist  /DEF_OBS_NML/ lon, lat, p, otype, rtype, value, nlat

  lon   =   0._wp
  lat   =  50._wp
  p     = 500._wp ! hPa
  otype = ''
  rtype = 'TEMP'
  value = 0._wp   ! K
  nlat  = 0
```

16.7 /TEMP_OBS/ (TEMP observation operator)

Missing: Description of namelist /TEMP_OBS_NML/

This namelist is defined in module :

```
!------------------
! Namelist TEMP_OBS
!------------------
logical   :: use_t      = .false.
logical   :: use_h      = .true.
logical   :: use_u      = .true.
logical   :: use_v      = .true.
logical   :: use_q      = .true.
real(wp)  :: top_q      = 300._wp ! top level for humidity
observations (hPa)
logical   :: mainpl     = .false. ! use main pressure levels only
logical   :: prt_data   = .false.
real(wp)  :: vmainpl (20) = -1._wp  ! main pressure level values
[hPa]
namelist  /TEMP_OBS/ use_t, use_h, use_u, use_v, use_q, mainpl,
vmainpl, &
```

```
                    top_q, prt_data
```

16.8 /SYNOP_OBS/ (SYNOP observation operator)

Missing: Description of namelist /SYNOP_OBS/

This namelist is defined in module :

```
!------------------
! Namelist SYNOP_OBS
!------------------
logical  :: use_t2   = .false. ! 2 m temperature
logical  :: use_rh   = .false. ! 2 m humidity
logical  :: use_vs   = .true.  ! 10 m wind over sea
logical  :: use_vl   = .false. ! 10 m wind over land
logical  :: use_p    = .false. ! p(h) is observable
logical  :: use_h    = .true.  ! h(p) is observable
logical  :: use_msl  = .true.  ! reduce p,h to mean sea level
logical  :: prt_data = .false. ! print data
namelist  /SYNOP_OBS/ use_t2, use_rh, use_vs, use_vl, use_p, use_h,
use_msl,&
                      prt_data
```

16.9 /TOVS_OBS/ (TOVS observation operator)

This namelist is defined in module

```
!==================
! Namelist TOVS_OBS
!==================
!------
! Input
!------
character(len=64) :: netcdf_path     = ''       ! path to read input files
character(len=32) :: feedbk_files(5) = ''       ! names of input/output files
!---------------
! Data selection
!---------------
integer           :: max_scan        = 1000     ! max. number of scans
integer           :: ionly (60)      = -1       ! channal indices to use
integer           :: inot  (60)      = -1       ! channal indices not to use
integer           :: skip_spots      = 0        ! no.spots to skip in input
logical           :: flg_external    = .false.  ! set channels by namelist
```

```
integer           :: flg_prc         = 15       ! 1=data,2=min,3=sur,8=cld
integer           :: flg_sur         =  0       ! 1dvar surface flag
integer           :: flg_cld         =  0       ! 1=clr,2=cld,4=mwclr,8=mwclo
!------------
! Computation
!------------
logical           :: use_1dvar_fg    = .false.  ! use first guess of 1dvar
integer           :: rttov_version   = 7        ! rttov version to use
real(wp)          :: min_err         = 0._wp    ! minimum observation error
real(wp)          :: p_rhtop         = 10._wp   ! no rh ass. above this lev.
real(wp)          :: e_bg_ts_sea     = 1._wp    ! surf.temp. bg.error sea
real(wp)          :: e_bg_ts_land    = 3._wp    ! surf.temp. bg.error land
real(wp)          :: e_bg_ts_ice     = 1._wp    ! surf.temp. bg.error ice
real(wp)          :: e_bg_t_top      = 3.0_wp   ! bg. error top levels t
real(wp)          :: e_bg_lnq_top    = 1.0_wp   ! bg. error top levels q
logical           :: lhum_dum_ana    = .true.   ! dummy humidity above an.lev
!------------
! Constraints
!------------
integer           :: chk_bound   (4) =  2       ! check bounds (:2=t-+;3:=q)
integer           :: rep_bound       =  0       ! report 0-2:none,brief,det.
!------------
! Diagnostics
!------------
integer           :: monitor_prof    = 0        ! monitor profiles (fg-scan)
integer           :: mon_ana_prof    = 0        ! monitor profiles (an-scan)

namelist /TOVS_OBS/ max_scan, netcdf_path, feedbk_files,              &
                    use_1dvar_fg, rttov_version, min_err, p_rhtop,    &
                    ionly, inot, skip_spots, flg_external,            &
                    e_bg_ts_sea, e_bg_ts_land, e_bg_ts_ice,           &
                    e_bg_t_top,e_bg_lnq_top,                          &
                    flg_prc, flg_sur, flg_cld,                        &
                    chk_bound, rep_bound, lhum_dum_ana, monitor_prof, &
                    mon_ana_prof
```

Parameter	Default	Description
		Input
netcdf_path	*input*	Path to the input files. Default is *input* as specified in namelist /RUN/.
feedbk_files	”,”,”, ..	Names of the input files (currently feedback files from the 1D-Var).
		Data Selection

flg_prc	15		1dvar processing flag: 0= observation is accepted and processed by 1D-Var Any value different to 0 indicates that the observation is rejected by 1D-Var. In particular: 1=data did not pass first-guess check, 2=1DVar did not converge for this data, 3=surface, 8=cloudy observation For a comparison of the 3D-Var and 1D-Var results, only 1D-Var observations with the 1D-Var precessing flag flg_prc=0 are processed in the 3D-Var
flg_sur	0		1D-Var surface flag, indicating the surface type of the observation: 1=sea, 2=ice, 4=land, 8=highland, 16=mismatch
flg_cld	0		1D-Var cloud flag: 1=clear, 2=IR cloudy, 3=MW clear, 4=MW cloudy
			Computation
e_bg_ts_sea	1.	K	Background error of surface temperature over sea. The surface temperature has to be provided for the RTTOV-operator and is treated as a dummy variable within 3DVar: The initial value is given by a 1D-Var background value superposed with a normal distribution of width e_bg_ts_sea. Its intermediate value is altered by the variational algorithm, but the resulting value is not fed back to the analysis.
e_bg_ts_land	3.	K	Background error of surface temperatures as above, but over land
e_bg_ts_ice	1.	K	Background error of surface temperatures as a above, but over ice
e_bg_t_top	3.	K	Background error of temperature in RTTOV pressure levels above the highest pressure level of the GME. To provide the RTTOV operator with values in these levels, the temperature is treated as a dummy variable distributed around a 1D-Var background with a width e_bg_t_top. These temperature values are not fed back to the analysis.
e_bg_lnq_top	1.		Background error of specific humidity in RTTOV pressure levels that are not assimilated in 3D-Var (top levels). The specific humidity is treated as a dummy variable. The initial values spread around a 1D-Var background with width e_bg_lnq_top. To avoid the problem of negative humidities, the error is specified in ln(q).
			Constraints

chk_bound	2	Check if the RTTOV input variables are within specified bounds: chk_bound(1): flag for temperature, lower bound chk_bound(2): flag for temperature, upper bound chk_bound(3): flag for humidity, lower bound chk_bound(4): flag for humidity, upper bound 0 = no check 1 = check (and report) 2 = check and constrain values (change variable) 3 = check and extrapolate (not yet implemented) 4 = check and abort
rep_bound	0	Report about result of check whether the RTTOV input variables b are within specified bounds: 0 = no report 1 = brief report for each profile 2 = detailed report for each variable and level messages are written to stderr .
		Diagnostics
mon_ana_prof	0	Switch on monitoring of profiles in the analysis step. The parameter is a bit-mask: 1: Output to ASCII-file. 2: Output to NetCDF-file. 4: Switch on some more expensive diagnostics. 8: Switch on diagnostics on **H** (observation operator) and **K** (gain matrix).

monitor_prof	0	Switch on monitoring of profiles in the first-guess step: 1: Output to ASCII-file. 2: Output to NetCDF-file.

16.9.1 Monitoring of profiles

The input parameters of the RTTOV opservation operator are monitored if the namelist variables `mon_ana_prof` or `monitor_prof` are set.

ASCII Output:

The following files are written:

intm.info	Monitoring in the first-guess step.
intp.info	Monitoring in the analysis step.
intp-H.info	Monitoring in the analysis step: linearised observation operator **H**.
intp-K.info	Monitoring in the analysis step: gain matrix **K**.

Netcdf-Output:

The following files are written:

monRTOVP.nc	Monitoring in the first-guess step.
cofRTOVP.nc	Monitoring in the analysis step.

The content of the NetCDF-file is:

name	flag	description
profile		profile index variable.
level		level index variable, 1 to 43
channel		channel index variable.
len8		8 byte character index variable.
len16		16 byte character index variable.
parameter (parameter, len16)		parameter index variable. Values:
	2	'fg ': first-guess
	2	'fg_error ': first-guess error
	2	'obs_error ': not used
	2	'fg_1dvar ': first-guess from the 1dvar feedback-
	2 A	'd_Jobs/d_x ': gradient of the cost function
	2 A	'ana ': analysis
	2 A	'ana_1dvar ': analysis from the 1dvar feedback-fil
	2 A	'a-fg_aprox_1dvar': analysis - first-guess, approximated
	2 A	'a-fg_aprox_ginv ': not used
	2+4 A	'a-fg_aprox_psas ': analysis - first-guess, approximated
id(profile)	2	observation id in 3dvar
lat(profile)	2	latitude of observation
lon(profile)	2	longitude of observation
statid(profile, len8)	2	station id in character form
ts(profile, parameter)	2	surface temperature (K)
ps(profile, parameter)	2	surface pressure (Pa)
t(profile, parameter, level)	2	temperature (K)
q(profile, parameter, level)	2	specific humidity (kg/kg)
H_ts(profile, channel)	2+8 A	surface temperature **H**-operator
H_ps(profile, channel)	2+8 A	surface pressure **H**-operator
K_ts(profile, channel)	2+8 A	surface temperature gain-matrix **K**
K_ps(profile, channel)	2+8 A	surface pressure gain-matrix **K**
H_t(profile, level, channel)	2+8 A	temperature **H**-operator
H_q(profile, level, channel)	2+8 A	specific humidity **H**-operator
K_t(profile, level, channel)	2+8 A	temperature gain-matrix **K**
K_q(profile, level, channel)	2+8 A	specific humidity gain-matrix **K**

Column 'flag' denotes the bits to be set in 'mon_ana_prof' in order to optain the respective output. 'A' means that this quantity is available only in the analysis step.

The gain matrix is calculated as $\mathbf{K} = \mathbf{B}\mathbf{H}^T(\mathbf{H}\mathbf{B}\mathbf{H}^T+\mathbf{R})^{-1}$, with the matrices **B**, **H**, and **R** being restricted to the subspace of the current field of view. The gain matrix written to the file is multiplied by the innovation vector (obs-fg) from the right hand side, so that its rows correspond to the analysis increment in a 1dvar-approximation. Thus the rows sum up to the same value as given in 'a-fg_aprox_1dvar'. Because the observation operator and the gain matrix require a large amount of space in the output file, their derivation is enabled seperately by setting the respective bit in 'mon_ana_prof.

'a-fg_aprox_psas' is the analysis increment in the space of the arguments of the observation operators, obtained directly from $\mathbf{H}^T(\mathbf{H}\mathbf{B}\mathbf{H}^T + \mathbf{R})^{-1}(obs - fg)$, without in-

terpolation to the model grid. Because the calculation of this quantity is expensive, it is enabled seperately by setting the respective bit in `'mon_ana_prof`.

Information on the observations themselfes (brightness temperatures) is present in the feedback-file `cofTOVS.nc` and in `cofTOVS_pp.nc` (or `monTOVS.nc` and `monTOVS_pp.nc`). The latter is derived from the former by the program `post_psas`.

16.10 /THINNING/ (thinning of observations)

This namelist is defined in subroutine (module mo_thinning):

Observational thinning is done as follows: For all observations of a certain type (classified by the CMA observation type and optionally by a list of CMA code types) thinning is done independently from other types. For each observation the next gridpoint (in a horizontal icosahedral grid specified by the partition parameter **ni** and a vertical grid with aequidistand spacing **dlev**) is sought. All observations attributed to the same grid point are ranked according to a number of criteria. The observation with the highest ranking will be chosen and the others will be rejected.

Each criterium is represented by a number (in general between 0 and 1) there higher values indicate a better quality of the data. For each criterium a lower bound is specfied; data with a lower value of the respective criterium is rejected in any case. For each criterium a weight is specified as well to be used in the rules described below. Default values of weights and bounds are given in the table below:

criterium	weight	bound	description
`center`	1.	0.	Horizontal distance parameter to the nearest grid point. A value of 1 indicates that the observation is exactly at that grid point. A value of 0 indicates that the observation is just in between two grid points.
`time`	1.	-3.	Time difference to the analysis time (absolute value in hours). The default value of the bound parameter of -3 denotes that all observations which exceed an assimilation winwow of $+-3$ hours are rejected.
`sequence`	1.	0.	This is just the sequence number of the observation in the input file. This criterium is generally applied as the last one in order to allow a unique decision between observations not ditinguished by any other criterium.
`data`	1.	0.	Number of independent observations provided by a report. For instance a SYNOP report providing pressure and wind information will be ranked higher than a report with pressure information only. (Note that thinning is performed on reports (TEMP,SYNOP), not on observation types (pressure,wind) so far).
`quality`	1.	-1.	Observation type specific quality parameter. its value is generally derived from the input (BUFR) file information provided by the producer of the data. The default value of -1. asures that reports without quality index (in this case the parameter is set to -0.01) are used as well.
`vertical`	1.	0.	Vertical distance parameter to the nearest gridpoint. A value of 1 indicates that the observation is exactly at that grid point. A value of 0 indicates that the observation is just in between two grid points.
`status`	1.	0.	Status (eg. dismissed,rejected,active) of the report set by the 3D-Var checks. This criterium makes sure that reports rejected by obs-fg or other checks is not used any more.
`pref_satids`	1.	0.	Preferences for specific satellite ids.
`pref_center`	1.	0.	Preferences for generating centers.
`pref_retrv`	1.	0.	Preferences for specific retrieval methods.

There are three namelist entries specifying preferences concerning satellites, processing center or retrieval methods: `pref_satids`, `pref_center`, `pref_retrv`. These entries must be given as pairs (satid, preference). For instance in order to favour NOAA-18 with respect to NOAA-19 and both of them against other satellites write:

```
pref_satids =  209 2 ! NOAA-18
               223 1 ! NOAA-19
```

A higher number for 'preference' gives a higher preference for this 'satid'. 'satid's not in the list will get a preference of zero. The preferences (for satids, center, retrv) will be added.

In order to rank the reports within each grid box a number of rules is used. Each rule consists of a number of criteria which are added with the respective weights. If a rule gives the same ranking for a pair of reports the next rule is used unless a decision can be

made. The default sequence of rules are as follows:

rule	criteria
1	'status'
2	'preference'
3	'quality'
4	'time'
5	'center' 'vertical'
6	'data'
7	'sequence'

By default first the best ranking of the 3D-Var checks (status) is selected. Afterwards any preferences concerning satellites, generating centers or retrieval methods are considered, followed by the best ranking derived from external quality information. The centering acording to the analysis time and to the underlying grid is taken into account next. Finally the amount of data in the report and its sequence number in the input file is checked. The latter rule cannot be overwritten by the namelist.

The meaning of the namelist variables is as follows:

obstype	`string`	CMA observation type this rule applies to. (cf. Table 17.1)
codetype	`integer array`	List of CMA code types this rule applies to. If not specified the rule applies to all code types of the observation type specified. (cf. Table 17.2)
dbkz	`integer array`	List of DWD data base id's this rule applies to. If not specified the rule applies to all id's of the observation type specified. (cf. Table 17.2)

ni	`integer`	Resolution (GME partitioning) parameter. The default is the resolution used for the analysis. The distance between observations is approximately 7054 km/ni. The min., max. distance is (cf. GME documentation).
		`16: 440.5 - 526.0 km`
		`24: 293.7 - 346.9 km`
		`32: 223.3 - 263.2 km`
		`48: 146.8 - 173.5 km`
		`64: 110.1 - 131.6 km`
		`96: 73.4 - 86.8 km`
		`128: 55.1 - 65.8 km`
		`192: 36.7 - 43.4 km`
		`256: 27.6 - 32.9 km`
		`384: 18.4 - 21.7 km`
		`512: 13.8 - 16.5 km`
		`768: 9.2 - 10.9 km`

dlev	`real`	Vertikal distance between observations (hPa).
comment	`string`	Arbitrary comment to appear in the printout.
crit1	`string,real,real`	Triplet: 'name' weight bound. This parameters allows to adjust the parameters weight and bound for each criterium.
crit2..7		as **crit1** for other criteria.
rule1	`string array`	List of criteria used in a rule.
rule2..5		as **rule1** for other rules.

As an example the following namelist entries are given:

```
&THINNING
  obstype = 'AIREP'
/
&THINNING
  obstype = 'SATOB'
  ni      = 64
  dlev    = 40
  crit1   = 'center'   1. 0.2
  rule1   = 'quality'
  rule3   = 'none'
/
```

The above namelist activates the thinning procedure for AIREPs with the default parameters:

```
obstype  = AIREP
codetype =   -1   0   0   0   0   0   0   0   0   0
ni       =  192
nlev     =    1
dlev(hPa)=    0

criterium   weight     bound
center        1.00      0.00
time          1.00     -3.00
sequence      1.00      0.00
data          1.00      0.00
quality       1.00      0.00
vertical      1.00      0.00
status        1.00      0.00

rule  1 :  status
rule  2 :  time
rule  3 :  quality
rule  4 :  center   vertical
rule  5 :  data
rule  6 :  sequence
```

Thinning is activated for the AMVs (SATOBs) with some non-default parameters: The thinning distance corresponds to ni=64, i.e. ≈120 km, and a vertical distance of 40 hPa. All data whose 'enter'-parameter is less than 0.2 is rejected. This ensures a minimum horizontal distance of $24 = 0.2 \times 120$ km. The 'status' rule (rule 1) is replaced by the 'quality' rule (formerly rule 3).

```
obstype  = SATOB
codetype =   -1   0   0   0   0   0   0   0   0   0
ni       =    64
nlev     =   26
dlev(hPa)=   40

criterium   weight     bound
center        1.00      0.20
time          1.00     -3.00
sequence      1.00      0.00
data          1.00      0.00
quality       1.00      0.00
vertical      1.00      0.00
status        1.00      0.00
```

```
rule  1 :  quality
rule  2 :  time
rule  4 :  center   vertical
rule  5 :  data
rule  6 :  sequence
```

16.11 /AMV_OBS/ (AMV observation operator)

Missing: Description of namelist /AMV_OBS/

16.12 /AIREP_OBS/ (AIREP observation operator)

Missing: Description of namelist /AIREP_OBS/

This namelist is defined in module :

```
!-------------------
! Namelist AIREP_OBS
!-------------------
logical  :: black_t_nop = .false. ! blacklist t-obs. if pressure is missing
logical  :: use_regnum  = .true.  ! use registration number, not flight no.
logical  :: require_q   = .false. ! require q for sufficient data
logical  :: reject_q_0  = .true.  ! reject zero value humidity observations
namelist  /AIREP_OBS/ black_t_nop, use_regnum, require_q, reject_q_0
```

16.13 /GPS_RO/ (GPS Radio Occultations)

This namelist is defined in module :

```
!---------
! namelist
!---------
real(wp) :: ray_incr (10) = 0._wp   ! increments between rays to pick up
real(wp) :: ray_levs (10) = 0._wp   ! levels to define increments
logical  :: smooth_eps    = .true.  ! smooth observed  bending angles
logical  :: skip_lowest   = .false. ! skip lowest ray
logical  :: skip_invalid  = .true.  ! skip invalid rays in subsequent iter.
logical  :: corr_geoid    = .true.  ! correct for geoid in interpolation
namelist /GPS_RO/ ray_levs, ray_incr, smooth_eps, skip_lowest, skip_invalid, &
                           corr_geoid
```

Namelist /GPS_RO/ is used to influence the behavior of the GPS Radio occultation operator:

The parameters denote:

ray_incr (10)	*type:* real(wp) ,*default*=0._wp Increments between rays to use (km). Zero denotes no further rays above that height.
ray_levs (10)	*type:* real(wp) ,*default*=0._wp Lower bounds (km) for increments.
smooth_eps	*type:* logical ,*default*=.true. Smooth observed bending angles. If set to .true., bending angles are smoothed with a window size according to the selected spacing.
skip_invalid	*type:* logical ,*default*=.true. If .true., rays which are invalid during one iterations are not used any more in subsequent iterations.
corr_geoid	*type:* logical ,*default*=.true. If .true., the geoid anomalies are accounted for.

For example the following sequence selects rays with a spacing of 300 m above 2 km, 500 m above 5 km, and 1000 m above 10 km height. No rays are selected below 2 km height and above 20 km height. Height refers to values of impact parameter minus local curvature radius.

```
&GPS_RO
  ray_levs = 2.0, 5.0, 10., 20.  ! levels  (km)
  ray_incr = 0.3, 0.5,  1.,  0.  ! spacing (km) above given levels
/
```

16.14 /RO_VDA/ (GPS Radio Occultation data files)

This namelist is defined in subroutine (module mo_occ) :

```
!-------------------
! namelist /RO_VDA/
!-------------------
integer                   :: ierr
character(len=64)         :: file ! vda file names
namelist /RO_VDA/  file
```

Currently input file names (vda-format provided by the CT algorithm) are explicitely specified by namelist group /RO_VDA/:

```
character(len=128) :: path ! path for reading vda files
character(len=64)  :: file ! vda file names
namelist /RO_VDA/ path, file
```

The namelist group is read repeatedly for different input files. A given path remains valid for subsequently read filenames.

Example:

```
&RO_VDA path='../../champ/vda/'/                          ! directory to use
&RO_VDA file='WO-AI-2-PD+2001_148_00_0001_cro1_001.dat'/  ! files to read
&RO_VDA file='WO-AI-2-PD+2001_148_00_0002_cro1_001.dat'/  ! ...
```

16.15 /STD_OBS/ (GNSS slant delay operator)

Empty: GNSS Slant Total Delay operator namelist

This namelist is defined in module STD_operator :

```
!-----------------------------
! namelist STD_OBS
!-----------------------------
logical            :: read_ascii   = .false.! read ascii file
integer ,parameter :: mo           = 12     ! max number of observation
                                            ! input files
integer ,parameter :: mf           = 256    ! max length of file names
character (len=10) :: RefTimeStr = ''       ! Reference time (analysis)
                                            ! yyyymmddhh
integer            :: STDperiod    =  0     ! STD period, minutes
integer            :: NStepVertMod = 60     ! Number of vertical points
                                            ! inside the model
integer            :: NStepVertTop = 80     ! Number of vertical points
                                            ! above the model
real (wp)          :: Hmax    = 150000.     ! Maximum hight for STD
                                            ! integration in m
real               :: HScaleP = 6500.       ! air pressure scale hight
                         ! (for scaling the point density along the slant)
real               :: HScaleP2 = 25000.     ! air pressure scale hight
real (wp)          :: k1 = 77.60_wp         ! refractivity coefficients
real (wp)          :: k2 = 70.40_wp         ! refractivity coefficients
real (wp)          :: k3 = 3.739E5_wp       ! refractivity coefficients
character(len=mf)  :: SlantPath = ''        ! Path to slant data
character(len=mf)  :: std_obs_file (mo) ='' ! STD observation input files
character(len=mf)  :: HORIFile = 'stako_gps_SNX_G_12_HORI'
                                            ! GNSS station file
character(len=mf)  :: DomesFile = 'smark_domes_SNX_RTT'
                                            ! GNSS station locations
```

Parameter	Default	Description
NStepVertMod	60	Number of supporting points used for vertical integration inside the model, should be $\geq$ the number of vertical grid nodes of the model. The number of supporting points on the signal path is scaled with the elevation: Nmod = NStepVertMod / $(2 \sin \varepsilon)$.
NStepVertTop	80	Number of supporting points used for vertical integration above the model top. NStepVertTop can usually be smaller than NStepVertMod as the contributions from the upper atmosphere is small. Like NStepVertMod this is scaled with the elevation: Nup = NStepVertTop / $(2 \sin \varepsilon)$. The total number of supporting points on the signal path is Ntot = 1 + (NStepVertMod + NStepVertTop) / $(2 \sin \varepsilon)$.
Hmax	150000	The STD is integrated from the GNSS station up to Hmax in m. 100000 $\leq$ Hmax $\leq$ 200000 (100 km ... 200 km)
HScaleP	6500	Scale hight in m of a hypothetical pressure profile. This profile is used to scale the density of supporting points with height. Smaller numbers lead to more points in the lower atmosphere. HScaleP is used for the signal path inside the model.
HScaleP2	25000	Like HScaleP but used above the model top.
$\mathbf{k}_1$	77.60	Refractivity coefficient k_1 [K hPa−1] used in the Smith & Weintraub formula or the Thayer formular, default according to Bevis, 1994.
$\mathbf{k}_2$	70.40	Refractivity coefficient k_2 [K hPa−1], default according to Bevis, 1994.
$\mathbf{k}_3$	$3.739 \cdot 10^5$	Refractivity coefficient k_3 [K^2 hPa−1], default according to Bevis, 1994.

The remaining namelist parameters are closely related to the preliminary data format and will not be described here.

Refractivity coefficients according to differnt authors and their impact on a hypothetical ZTD (last column):

Quelle	k_1 [K hPa^{-1}]	k_2 [K hPa^{-1}]	k_3 [K^2 hPa^{-1}]	ZTD [m]
Smith & Weintraub	77.607	71.60	$3.747 \cdot 10^5$	2.417459855
Thayer	77.604	64.79	$3.776 \cdot 10^5$	2.417989384
Hasagawa	77.60	69.40	$3.701 \cdot 10^5$	2.414152786
Bevis	77.60	70.40	$3.739 \cdot 10^5$	2.416587827
Foelsche	77.650	65.99	$3.777 \cdot 10^5$	2.419539541
Rueger	77.689	71.2952	$3.75463 \cdot 10^5$	2.420170997

The refractivity coefficients suggested by Bevis seem to provide the best results and have been chosen as default.

Add references

16.16 /PSAS/ (PSAS specific parameters)

Physical Space Assimilation System (PSAS)

Missing: Description of namelist /PSAS/

This namelist is defined in module :

```
!=========
! namelist
!=========
integer          :: bar_it    =   0      ! barrier outer subiterations
integer          :: bar_i     =   1      ! barrier inner subiterations
real(wp)         :: ckobg     = 0.
character(len=8) :: prec      = 'left'   ! precond:'left','right','both','no'
character(len=8) :: ptype     = 'old'    ! precond: 'OLD', 'JAC', 'GS', 'BGS'
logical          :: lnonl     = .true.   ! nonlinear CG
logical          :: linv      = .false.  ! return inverse matrix instead of R
!---------------
! general set up
!---------------
integer          :: nboxes    =    0     ! number of boxes
integer          :: nobs_box  =  500     ! approx. number of observations/box
integer          :: nobs_fg   = 5000     ! approx.number obs/box in fg-scan
integer          :: idtwin    = 0        ! 1:1, random
logical          :: freeio    = .false.  ! no computations on I/O PE
logical          :: fg_scan   = .true.   ! run first guess scan
logical          :: ana_scan  = .true.   ! run analysis
integer          :: fgaprx    = PB_DIAG  ! approximation of Pb in fg scan
integer          :: anaprx    = PB_FULL  ! approximation of Pb in psas scan
logical          :: sort_boxes= .true.   ! sort boxes addording to distance
character(len=12):: lstat_ana = ''       ! lowest status to keep
integer          :: istat_ana = STAT_ACTIVE_0
logical          :: init_seed = .false.  ! init seed for random vectors
!----------------------
! control of inner loop
!----------------------
character(len=8) :: solver    = 'nlpcg'  ! solver 'numrec' or 'smlib'
character(len=8) :: solver1   = 'newton' ! solver 'newton', 'nlpcd'
integer          :: min_it    =   1      ! minimum number of cg iterations
integer          :: max_it    =  10      ! maximum number of cg iterations
real(wp)         :: gtol      = 1.e-6_wp ! stop criterium
real(wp)         :: gtolo     = 1.e-6_wp ! stop criterium outer loop
logical          :: scal      = .true.   ! scaling
```

```
!----------------------
! control of outer loop
!----------------------
integer,parameter:: mio       = 300      ! max number of outer iterations
integer          :: nitout    =  10      ! number of outer iterations
logical          :: fg_b     (mio)       ! start with first guess from bg
logical          :: fg_o     (mio)       ! start with first guess from obs
logical          :: new_R    (mio)       ! recalculate R
integer          :: new_H    (mio) = 0   ! recalculate H (and HPH)
integer          :: new_Hls  (mio) = 0   ! recalculate H in the line search
logical          :: new_R_H  (mio)       ! recalculate preconditioner
logical          :: linesrch (mio)       ! perform line search
real(wp)         :: vqcf     (mio)       ! variational quality control flag
logical          :: xlnsrch   = .true.   ! flag for more exact line search
integer          :: maxlnsrch = 5        ! stop if more steps in line search
!-----------------------
! control of line search
!-----------------------
real(wp)         :: omega     = 1._wp
!-------------------------------
! control of post-multiplication
!-------------------------------
integer          :: ni_ana    = 0       ! analyses grid resolution
logical          :: l_int_fg  = .true.  ! interpolate analysis to fg resolt.
integer          :: pes_post  = 1       ! stride for Pes in post multiplic.
logical          :: no_post   = .false. ! skip post-multiplication
logical          :: no_ps_inc = .false. ! no analysis increment on ps
logical          :: no_tv_inc = .false. ! no analysis increment on tv
logical          :: no_rh_inc = .false. ! no analysis increment on rh
logical          :: no_uv_inc = .false. ! no analysis increment on uv
integer          :: no_topinc = 0       ! no. top-levels with zero increment
integer          :: lhskep    = 0       ! keep lhs in post-multiplication
logical          :: lnewpl    = .false. ! analysis increment on new p.levels
logical          :: lvint     = .true.  ! vertical interpolation of bg to "
!------------
! diagnostics
!------------
integer          :: lplot     = 0       ! write plot (GRADS/GMT)-Files
logical          :: ldiag     = .true.  ! print diagnostics
logical          :: grads_er  = .true.  ! write errors to GRADS file
logical          :: out_anin  = .false. ! write analysis increment
logical          :: out_aninp = .false. ! write analysis increment on p-levs.
logical          :: out_add   = .false. ! write additional fields to analysis
integer          :: test_ev   = 0       ! test matrices B,R,B+R for pos.def.
integer          :: fix_ev(3) =(/2,0,2/)! fix matrices B,R,B+R
```

```
integer          :: verb_ev   = 2       ! verbosity in pos.def. test
logical          :: tune_RB   = .false. ! call tune_matrix
!------------
! Debug flags
!------------
logical  :: luni     = .false. ! univariate correlations
logical  :: samg     = .false. ! write SAMG (Algebraic Multigrid) file
logical  :: samg_B   = .false. ! write SAMG file  (B only)
logical  :: samg_R   = .false. ! write SAMG file  (R only)
logical  :: norm_samg= .false. ! normalize matrices written to SAMG file
logical  :: test_samg= .false. ! reread SAMG-file and test for pos.def.
logical  :: mleak    = .false. ! mark timing output for memory leak search
integer  :: nsamg    = huge(1) ! number of rows/columns to write
integer  :: lerror_tq= 4       ! errors in tq_tvgh: 1..3:message, 4:abort
logical  :: tight    = .true.  ! .true. for tight memory usage

namelist /PSAS/ nboxes, nobs_box, solver, solver1,  min_it, max_it,         &
                nobs_fg, bar_it, bar_i, gtol, gtolo, ldiag, linv,           &
                samg, samg_B, samg_R, norm_samg, test_samg, nsamg,          &
                ckobg, idtwin, fgaprx, anaprx,                              &
                lhskep, lplot, ni_ana, pes_post, l_int_fg, prec, ptype,     &
                omega, lnonl, scal, nitout, no_topinc,                      &
                fg_b, fg_o, new_R, new_H, new_Hls, new_R_H, vqcf, linesrch, &
                xlnsrch, maxlnsrch, no_post, freeio, lnewpl, lvint,         &
                no_ps_inc, no_tv_inc, no_rh_inc, no_uv_inc, tmu, tight,     &
                grads_er, out_anin, out_aninp, out_add, test_ev, fix_ev,    &
                verb_ev, fg_scan, ana_scan, lerror_tq, mleak, sort_boxes,   &
                lstat_ana, init_seed, tune_RB
```

16.17 /PSAS_MONITOR/ (monitor convergence)

Missing: Description of namelist /PSAS_MONITOR/

This namelist is defined in module :

```
!-----------------------------------------
! parameters set by namelist /PSAS_MONITOR/
!-----------------------------------------
logical :: mon_lnsrch = .false. !monitor gradients in line search
real(wp) :: delta_gev  = 0.01_wp ! increment for gradient evaluation
real(wp) :: delta_fev  = 0.10_wp ! increment for function evaluation
namelist /PSAS_MONITOR/ mon_lnsrch, delta_gev, delta_fev
```

16.18 /TEST_OBS_OPR/ (test tangent linear operator)

Missing: Description of namelist /TEST_OBS_OPR/

This namelist is defined in module :

```
!   Namelist /TEST_OBS_OPR/        : provide parameters for the test
!   Subroutine read_nml_testobsopr : read this namelist
!   Subroutine test_obsopr         : to test operators
!
!
! Usage:
!
!   Enable via namelist /TEST_OBS_OPR/.
!   Examples:
!
!   mode     = 'on'          : Use one random increment dx
!              'scale'       : Use scaling factors 1.e0 .. 1.e-12
!              'operator'    : Seperately test all operators  (TEMP, SYNOP, ..)
!              'quantity'    : Seperately test all quantities (tv,rh,u,v,..)
!              'iarg'        : Seperately test each argument
!   operator = 'TEMP','SYNOP': Test the given operators  only
!   quantity = 'u','v'       : Test the given quantities only
!   scale    = 1.e-5         : Use this scaling factor instead of 1.e-3
!   iarg     = 3             : Test 3rd argument of selected operator/quant.
!
!
! Output (example):
!
!   mode      : on                          ! selected mode
!   scale     :   1.0000000000000000E-03    ! selected scale factor
!   operator  : TOVS                        ! selected operator
!   quantity  : tv                          ! selected quantity
!
!   arguments : 87                          ! size of argument vector
!    (active) : 43                          ! no of arguments disturbed
!   results   : 24                          ! size of result vector
!
!   |dx|    =    0.7185369056023909        ! norm of random increment (scaled)
!   |du|    =    1.0485202182687827        ! norm of result vector (fin. diff.)
!   |dy|    =    1.0507179991226590        ! norm of result vector (analytic)
!   |du-dy| =    0.1438600848470013        ! difference (fin. diff. - analytic)
!
! The vector |du-dy| should be several orders of magnitude smaller than
! |du|, |dy|. This example test failed.
```

```
!---------------------------
! Namelist (module) variables
!---------------------------
integer ,parameter :: mo           = 10      ! max no operators, quantities
character(len=5)   :: operator(mo) = ''      ! operators to check ('TEMP',..)
character(len=2)   :: quantity(mo) = ''      ! quantity  to check ('tv',  ..)
real(wp)           :: scale        = 1.e-3_wp! scaling of the increment
integer            :: iarg         = 0       ! argument index
integer            :: ires         = 0       ! result index
character(len=8)   :: mode         = ''      ! test mode
integer            :: seed (10)    = 0       ! seed for random number gener.
integer            :: test_H       = 0       ! 1=fg.test, 2=test if new H

namelist /TEST_OBS_OPR/ scale, mode, operator, quantity, iarg, ires, seed, &
                        test_H
logical          :: namelist_read = .false.
```

16.19 /VARQC/ (Variational Quality Control)

This namelist is defined in module :

```
!=================
! Namelist /VARQC/
!=================
!-------------------------------------------------------------------------
! Namelist /VARQC/ is read by subroutine 'read_vqc_nml'. (Default values
! are set by subroutine 'set_vqc_def'. The namelist provides parameters
! specifying the general behaviour of the variational quality control
! routine, mainly the approximation strategies followed by subroutine
! 'vqc_corr':
!
! 'iprint'  determines the level of printout (0: none, 1:short, 2: long).
! 'w_min'   is the minimum eigenvalue of the returned Hessian.
! 'iappr'   determines the approximation of the returned inverse Hessian
!           0: The inverse Hessian is not calculated
!           1: an approximation is used (allways positive semidefinit)
!           2: The exact Hessian (despite constraints on the eigenvalues
!              is returned. Eigenvalues may be constrained as
!              specified by parameter 'w_min'.
!           3: As 2, but eigenvalues of R^(-1)+B^(-1) are constrained.
! 'vqc_par' Approximation strategies for subroutine 'vqc_corr'
!-------------------------------------------------------------------------
integer          ,save :: iprint      ! print flag
```

```
real(wp)          ,save :: w_min       ! minimum singular value in Hessian
integer           ,save :: iappr       ! approximation level for R
type (t_vqc_par) ,save :: vqc_par(10) ! set of strategies

real(wp)          :: svqc     = 3._wp    ! variational quality control sigma
integer           :: mvqc     = 3        ! max number of rejected observation
real(wp)          :: gvqc     = 0._wp
logical           :: cvqc     = .true.   ! correlated observations in VQC


namelist /varqc/ w_min, iappr, vqc_par, iprint, &
                 svqc, mvqc, cvqc
```

/VARQC/		Variational-Quality-Control Namelist-Group
variable	default	description
iprint	0	level of printout: 0: none 1: short 2: long
w_min	10^{-2}	minimum eigenvalue of the returned Hessian.
iappr	2	approximation level for $\mathbf{R}$: 0: $\mathbf{R}$ is always given for $y_i - y^o = 0$. 1: an approximation is used. 2: the exact Hessian (despite constraints on the eigenvalues) is returned.
vqc_par	see below	approximation strategies for subroutine ṽqc_corr.
cvqc	.true.	account for correlated observations in VQC.
svqc	3.	default cutoff bound (in terms of σ_o.
mvqc	3	default maximum number of rejected correlated observations.

The namelist /VARQC/ allows to modify the technical parameters of the Variational Quality Control algorithm (Section 8.2). Observation type specific parameters shall be set in the namelist group /RULES/. For the case that no observation type specific parameters are given, the values of svqc and mvqc as provided here take effect.

In each outer iteration i of the PSAS algorithm, i.e. for each guessed minus observed value $y_i - y^o$ of an observed quantity, the VQC algorithm determines the contribution to the observational cost function J_o as well as the first and second derivative of the contribution. The matrix of the second derivatives is the so called Hessian matrix. In case of fully rejected observations (large $y_i - y^o$) the derivatives approach zero.

The inverse of the Hessian matrix is required in the PSAS optimization algorithm (denoted as $\mathbf{R}_i$). In order to remain invertible the eigenvalues of the returned Hessian are restricted to remain higher than a minimum value w_min. In addition, the Hessian (respectively $\mathbf{R}_i$) may be calculated in different approximations (parameter iappr.) Changing these two parameters may have an effect of the efficiency of the optimization algorithm

but should not alter the final solution.

For correlated data (up to now only the geopotential height reports within each TEMP observations are treated as correlated) for all combinations of accepted and rejected data (levels) the observational cost function and its derivatives must be determined and added with respect to their individual weights. In general (for a large number of correlated observations) calculation of all combinations is not feasible and only the most probable terms of the sum are considered. The strategy for finding these terms is determined by the parameter `vqc_par`, which is an array of derived data type `t_vqc_par`. The components of this data type are as follows:

`/VARQC/`		Variational-Quality-Control Namelist-Group
variable	default	description
dim_lim	0	limit on the number of dimensions (data).
n_full	0	up to this number of rejected all combinations are calculated.
c_start	”	start value for exploration.
c_final	”	exploration strategy in final phase.
n_lev_s	0	number of levels to explore in start-phase.
n_lev_f	0	number of levels to explore in final-phase.

For instance, to evaluate all combinations of accepted and rejected observations (up to 1000 correlated observations) the parameter `vqc_par` should be specified as follows:

```
vqc\_par(1)\%dim\_lim = 1000
vqc\_par(1)\%n\_full  = 1000
vqc\_par(2)\%dim\_lim = 0
```

The default setting is as follows:

	dim_lim	n_full	c_start	n_lev_s
vqc_par(1)%	10	10		
vqc_par(2)%	20	5	'mostprob'	2
vqc_par(3)%	30	3	'mostprob'	2
vqc_par(4)%	50	2	'mostprob'	1
vqc_par(5)%	100	1	'mostprob'	1
vqc_par(6)%	1000	1		

For up to 10 correlated observations all ($2^{10} = 1024$) combinations of accepted and rejected observations are computed. For up to 20 correlated observations all combinations with less than or equal 5 rejected observations are calculated. For more than 5 rejected observations only a subset of the combinations is accounted for in the sum. For higher numbers of correlated observations the number of combinations is further reduced. For more than 100 correlated observations only one of them (or all) may be rejected.

16.20 /PSCMODEL/ (background error model)

This namelist is defined in module :

```
!--------------------
```

```
! namelist /PSCMODEL/
!--------------------
integer,parameter:: IFC_CL = 1        !   climatological error
integer,parameter:: IFC_CFC= 2        !   climatological fc error
integer,parameter:: IFC_AFC= 3        !   actual         fc error
integer          :: ifc    = IFC_AFC  ! flag for fg-error type
character(len=8) :: model  = 'dwd-oi' ! horizontal correlation model
logical          :: rhcexp = .true.   ! RH horiz. corr. = exp(-.5(L/L_q)**2)
integer          :: irad   = -1       ! radian or secant is argument to c.f.
real(wp)         :: L_h    =  0._wp   ! horizontal length scale for height
real(wp)         :: L_q    =  0._wp   ! horizontal length scale humidity
real(wp)         :: a_h    =  0._wp   ! vertical tuning parameter height
real(wp)         :: a_q    =  0._wp   ! vertical tuning parameter humidity
real(wp)         :: a_x    =  0._wp   ! vertical tuning parameter vel.pot.
real(wp)         :: t_h    =  0._wp   ! linear transformation coef. height
real(wp)         :: t_q    =  0._wp   ! linear transformation coef. humidity
real(wp)         :: t_x    =  0._wp   ! linear transformation coef. vel.pot.
real(wp)         :: e_h    = -1._wp   ! height error
real(wp)         :: e_q    = -1._wp   ! relative humidity error
real(wp)         :: e_v    = -1._wp   ! wind error
real(wp)         :: rnu    = 0.1_wp   ! e2_x / e2_v
real(wp)         :: cut    =  0._wp   ! cutoff radius (multiple of L)
logical          :: test_bl= .false.  ! flag to test blockdiagonal of HBHt
integer          :: fixbc  = 1        ! fix bessel coefs.: d2c/dr2|r=1 = 0
real(wp)         :: modcvx = 0.08_wp  ! modified vertical corr. for div.wind
real(wp)         :: a      = 0._wp    ! scale parameters for cov. models
real(wp)         :: nw     = na       ! number of coefficients for wind corr.
logical          :: thc    = .false.  ! transform horizontal coordinates
real(wp)         :: lsvv   = 0._wp    ! largescale contrib. to wind-wind cor.
real(wp)         :: scale  = 1._wp    ! scaling factor for background error
```

Parameter	Default	Description
ifc	`3`	1: use *climatological* errors. 2: use *climatological forecast* errors 3: use *actual forecast* errors, modified due to the previous analysis error estimate.
model	`'dwd-oi'`	Horizontal correlation model for geopotential to. Valid choices are: `'dwd-oi', 'c5pr_12', 'c5pr_inf', 'csxoar1', 'csxoar2', 'gauss'`
rhcexp	`.true.`	Apply exponential horizontal covariance model for relative Humidity ($C = \exp(-0.5(L/L_q)2)$). Else the same function as for geopotential is applied.
irad	`cf.text`	The radiand (`irad=1`) or secant (`irad=0`) of the distance on the sphere is used as argument to the horizontal correlation function. A default value of 0 is used for the models `'c5pr_XX'` and `'csxoarX'`, 0 elsewhere.
rnu	`0.1`	Contribution of divergent wind variance to total wind variance (E_x^2/e_v^2)
cut	`0`	Cutoff radius (given as a multiple of the horizontal length scale L)
fixbc	`0`	1,2: Adjust Bessel function coefficients so that the second derivative vanishes for the cutoff radius. 0 for no adjustment.
nw	`8.(all)`	Number of coefficients to use for the mass-wind correlations. The value for the DWD-OI was 5.5 .
a	`0.`	Additional parameter for the horizontal covariance models `'csxoar1'` and `'csxoar2'`.
modcvx	`0.08`	parameter of the vertical correlation function for wind.
lsvv	`0.`	additional large scale contribution to wind-wind correlation. (required for stable operation of the global model in the tropics.)
		Debugging and test flags
L_h	`0.`	Value of constant horizontal length scale for mass and wind (m). 0 for variable length scale
L_q	`0.`	Value of constant horizontal length scale for relative humidity (m). 0 for variable length scale
a_h	`0.`	Value of constant vertical mass correlation parameter a. 0 if variable.
a_q	`0.`	Value of constant vertical relative humidity correlation parameter a. 0 if variable.
a_x	`0.`	Value of constant vertical wind correlation parameter a. 0 if variable.
t_h	`0.`	Factor for vertical linear coordinate transformation of mass. 0 for polinominal expansion.
t_q	`0.`	Factor for vertical linear coordinate transformation of relative humidity. 0 for polinominal expansion.
t_x	`0.`	Factor for vertical linear coordinate transformation of wind. 0 for polinominal expansion.
e_h	`-1.`	Value of constant geopotential height error (m). -1 if variable.
e_q	`-1`	Value of constant relative humidity error (0 ... 1). -1 if

Valid choices for **model**

'dwd-oi'	Bessel function expansion as in DWD OI.
'c5pr_12'	compactly supported 5th order piecewise rational function $C_0(a = 1/2)$, Gaspari and Cohn, QJRMS (1999),125,723-757
'c5pr_inf'	compactly supported function $C_0(a = \infty)$.
'csxoar1'	compactly supporting function approximating SOAR, Gneiting, QJRMS (1999),125,2449-2464
'csxoar2'	compactly supporting function approximating TOAR, Gneiting

16.21 /BG_ERROR_OPERATOR/ (wavelet based background error model)

Missing: Description of namelist /BG_ERROR_OPERATOR/

This namelist is defined in module :

```
!============================
! namelist /BG_ERROR_OPERATOR/
!============================
!-----------------------------
! number of gridpoints, bounds
!-----------------------------
integer           :: nz        =  64   ! number of vertical levels
integer           :: ny        =   0   ! number of latitudinal  gridpoints
integer           :: nx        =   0   ! number of longitudinal gridpoints
real(wp)          :: pbot      = 1070. ! bottom (surface) pressure (hPa)
!----------------------------------
! specification of covariance model
!----------------------------------
integer           :: valid     = 0     ! 1:content is valid 2:use cov.model
integer           :: repr_2dh  = 0     ! hor. operator repr: 1=test, 2=use
character(len=128):: file      = ''    ! input file name, NMC, vert.cov.
character(len=128):: file_2dh  = ''    ! input file name, NMC, hor. cov.
integer           :: lclim     = 2     ! 1:clim.err.  valid 2:use clim.err.
real(wp)          :: scale_rh  = 1._wp ! scaling factor for rel.hum. fg.error
!--------------------------------
! covariance matrix manipulations (partly not implemented)
!--------------------------------
logical           :: modify_cor= .true.! modify correlations, else variances
real(wp)          :: lat_homog = 99._wp! latitudinal homogeneous covariances
real(wp)          :: smooth_h  = 0._wp ! smooth horizontally
real(wp)          :: smooth_v  = 0._wp ! smooth vertically
!---------
! details
!---------
```

```
character         :: transform = 'w'   ! 'w'avelet, 'n'one
character         :: sqr       = 's'   ! 's'ymmetric, 'c'holesky
integer           :: nwv       = 4     ! no.points in vertical interpolation
integer           :: base      = WV_CUB_LIFT_INT ! wavelet basis
logical           :: h2psi     = .true.! derive streamf.    cov. from height
logical           :: h2t       =.false.! derive temperature.cov. from height
logical           :: sqrtcor   =.false.! derive sqrt from corr.,not cov.
logical           :: sparse    =.false.! sparse repres. of vertical matrices
integer           :: n_av_pole = -1    ! average anal.incr. at poles (no.pts)
integer           :: zonfilt(3)= (/2,2,3/) !no.Fourier coefs.in zonal filter
integer           :: n_zonfilt = 2     ! no of latitudes near poles to filter
!-----------------------------
! First Guess error estimation
!-----------------------------
integer           :: fger_iter = 25    ! no. iterations for bg-error estimate
logical           :: fger_est  =.false.! estimate first guess error
logical           :: fger_Li1  =.false.! test apply_L_i with e_fi=1, abort
logical           :: fger_seed = .true.! generate seed (for parallel run)
logical           :: fger_write=.false.! write fg error to file
!-------
! tests
!-------
logical           :: test_bg   =.true. ! compare operator impl. with explicit
real(wp)          :: test_bg_b = 1.e-5 ! bound for failure of test_bg
integer           :: test_adj  = 2     ! compare linear vs adjoint
logical           :: ltrace    =.false.! trace minval,maxval,variance
!----------
! debugging
!----------
integer           :: horz_inp  = 2     ! 1:next-neighbour, 2:bi-linear
integer           :: vert_inp  = 2     ! 1:next-neighbour, 2:interpolate
logical           :: l_L_h     = .true.! apply L    matrix in the horizontal
logical           :: l_W_h     = .true.! apply W    matrix in the horizontal
logical           :: unit_v    =.false.! apply unit matrix in the vertical

namelist /BG_ERROR_OPERATOR/ nx, ny, nz, pbot, file, transform, sqr, valid, &
                             base, nwv, h2psi, h2t, sqrtcor, sparse, lclim, &
                             scale_rh, test_bg, repr_2dh, file_2dh,         &
                             modify_cor, lat_homog, smooth_h, smooth_v,     &
                             test_adj, horz_inp, l_L_h, l_W_h, vert_inp,    &
                             test_bg_b, unit_v, ltrace,                     &
                             fger_iter, fger_est, fger_seed, fger_write,    &
                             fger_Li1, n_av_pole, zonfilt
```

16.22 /ANAERROR/ (analysis error calculation)

This namelist is defined in subroutine in module mo_anaerr.

```
!------------------
! Namelist Variables
!------------------
integer  :: anaerr    =  1 ! calculate analysis error
real(wp) :: lat1           ! first latitude         [degree]
real(wp) :: lon1           ! first longitude        [degree]
real(wp) :: latd           ! latitudinal  increment [degree]
real(wp) :: lond           ! longitudinal increment [degree]
integer  :: nx             ! no. grid points in x-direction
integer  :: ny             ! no. grid points in y-direction
real(wp) :: plevs(50)      ! pressure levels        [hPa]

namelist /ANAERROR/ anaerr, lat1, lon1, latd, lond, nx, ny
```

Namelist /ANAERROR/ is used to switch on or off analysis error calculations. Furthermore the domain and grid-spacing of the analysis error grid may be specified. If a previous analysis error was read (cf. parameter ifc in namelist /PSCMODEL/) the default grid specifications are taken from that file. Otherwise they are derived from the domain of the first guess fields, with a spacing of 6. degrees for the global domain and 1. degree for a local area model.

Parameter	Default	Unit	Description
anaerr	1		1: calculate analysis error 0: do not calculate analysis error
lat1	-87. / *	degree	First latitude value
lon1	0. / *	degree	First longitude value
latd	6. / 1.	degree	Latitude increment
lond	6. / 1.	degree	Longitude increment
nx	30 / *	degree	number of latitudes
ny	60 / *	degree	number of longitudes
plevs	10,50,100,200,300,500,1000	hPa	pressure levels

16.23 /CNTRLVAR/ (control variable transformation)

This namelist is defined in module .

```
real(wp) :: rh0     = 0.03_wp    ! lower bound for linear range (rel.hum)
real(wp) :: rh0fg   = 0.01_wp    ! lower bound for first guess rel.humidity
real(wp) :: q0fg    = 1.25e-6_wp ! lower bound for first guess spc.humidity
real(wp) :: rhmax   = 100._wp    ! upper bound for relative humidity
real(wp) :: rh1     = 100._wp    ! upper bound for linear range (rel.hum)
real(wp) :: rh1fg   = 100._wp    ! upper bound for first guess rel.humidity
integer  :: version = 1          ! 1:exponential; 2:quadratic
namelist /CNTRLVAR/ rh0, rh0fg, q0fg, rhmax, rh1, rh1fg, version
```

In order to constrain analysed humidity to values larger than zero a generalised transformed conrol variable is used in the variational scheme. Above a threshold `rh0` generalised humidity is equal to relative humidity. Below it is modified so that generalised humidity is allowed to take any finite value whereas relative humidity is constrained to values ≥ 0. A similar mechanism may be activated to prevent super-saturation ($rh > 1$) as well by a respective re-definition of generalised humidity in the vicinity of $h \approx 100\%$. Details are given in the scientific documentation 8.3.3.

Parameter	Default	Unit	Description
version	1	1 or 2	Choice of transformation from generalised humidity 1) $rh \approx \exp(gh)$ 2) $rh \approx gh^2$
rh0	0.03	0...1	Lower bound of region with $rh = gh$. Set to 0 to disable generalised humidity transformation for small humidities.
rh0fg	0.01	0...1	First guess value of rh is constrained to be larger than this value. (for version=1 only)
q0fg	1.25e-6	Kg/Kg	First guess value of q is constrained to be larger than this value. (for version=1 only)
rhmax	100.	0...1	Upper bound of rh. Set to ≈ 1 to prevent supersaturation.
rh1	100.	0...1	Upper bound of region with $rh = gh$. Set to $<$ `rh1` to enable generalised humidity transformation for values close to supersaturation.
rh1fg	100.	0...1	First guess value of rh is constrained to be smaller than this value. (for version=1 only)

16.24 /HUM_ANA/ (humidity analysis)

Missing: Description of namelist /HUM_ANA/

This namelist is defined in module :

```
!--------------------
! namelist parameters
!--------------------
real(wp) :: strat_q     = 4.0e-6_wp ! stratospheric humidity
real(wp) :: hum_ana_top = 25000._wp ! top of humidity analysis
real(wp) :: e_hum_top   = 0.01_wp   ! nominal FG error above hum_ana_top
real(wp) :: ln_hum_ana_top

namelist /HUM_ANA/ strat_q, hum_ana_top, e_hum_top
```

16.25 /OBSERR/ (observation error model)

Missing: Description of namelist /OBSERR/

This namelist is defined in module :

```
!--------------------------------------
! namelist parameters (free atmosphere)
!--------------------------------------
character(len=8) :: obstype         ! observation type ('TEMP','PILOT',etc.)
integer          :: codetype (ncde) ! code type        (numeric)
character(len=8) :: quantity        ! quantity        ('uv','tv', 'rh')
character(len=8) :: table           ! 'OI' or 'IFS'
real(wp)         :: scale(nlev)     ! rescaling factor
real(wp)         :: err  (nlev)     ! externally specified obs. errors


!-------------
! set defaults
!-------------

obstype  = ''
codetype = -1
quantity = ''
table    = ''
scale    = -1._wp
scale(1) =  1._wp
err      = -HUGE (0._wp)
```

16.26 /PROFILES/ (write profiles of atmospheric parameters)

Missing: Namelist PROFILES

Missing: Description of namelist /PROFILES/

16.27 /ENKF/ (Ensemble Kalman Filter specific namelist)

This namelist is defined in module mo_letkf:

```
!==================================
! Variables in Namelist Group /ENKF/
```

```
!==================================
integer            :: k_enkf    = 0       ! ensemble size
logical            :: gen_ens   = .false. ! generate random forecast ensemble

!------------------------------------------------------------------------
! steering of atmospheric parameters
! (which of them take from ensemble, mean, deterministic forecast etc.)
!------------------------------------------------------------------------
integer            :: src_fc    = 0  ! fc  from 0:mean     1:psas fc
character(len=256) :: par_read  = ' '! parameters to read (fc) 'ps t q u ..'
character(len=256) :: par_fc_en = ' '! parameters to take from fc ensemble
character(len=256) :: par_fc_mn = ' '! parameters to take from forecast mean
character(len=256) :: par_fc_det= ' '! parameters to take from determ.run
character(len=256) :: par_trans = ' '! parameters to transform 'ps t q u..'
character(len=256) :: par_ana   = ' '! parameters to take from analysis
character(len=256) :: par_fce   = ' '! parameters to take from fc ensemble
character(len=256) :: par_fcm   = ' '! parameters to take from forecast mean
character(len=256) :: par_write = ' '! parameters to write
character(len=256) :: par_diagn = ' '! parameters for diagnostics
integer            :: det_run   = 0  ! deterministic run

!-----------------------------
! input / output specification
!-----------------------------
character(len=256) :: path_diag = ''                    ! path for diagnostics
character(len=256) :: ane_fname = 'gaf_ANA_TIMEMM_.ens'! an. ensemble basename
character(len=256) :: fce_fname = 'gff_FCR_TIMEMM_.ens'! fc. ensemble basename
character(len=256) :: invar_ens = ''                    ! invariant field file
character(len=256) :: ana_file  = 'gaf_ANA_TIMEMM_'     ! det. an. filename
character(len=256) :: ana_read  = ''        ! analysis to compare with
character(len=256) :: grid_file = ''        ! grid metadata (ICON)
character(len=64)  :: fof_prefix(mfof) = '' ! prefix for 'fof' files
logical            :: rename_diag=.false. ! temporary: rename rho_local,diag..

!-----------------------
! coarse grid resolution
!-----------------------
integer            :: rf        = 1  ! horizontal coarsening factor
integer            :: rni       = 1  ! reduced ni-resolution (GME only)
integer            :: nzr       = 1  ! number of height levels (coarse grid)

!-------------
! localisation
!-------------
real(wp)           :: lh        = 0._wp  ! hor.lengthscale for obs.weights(km)
```

```
real(wp)          :: lh_min    = 0._wp  ! hor.lengthscale for obs.weights(km)
real(wp)          :: lh_max    = 0._wp  ! hor.lengthscale for obs.weights(km)
real(wp)          :: lv        = 0._wp  ! ver.lenghtscale for obs.weights
real(wp)          :: lv_surf   = 0._wp  ! min.ver.lenghtscale for obs.weights
real(wp)          :: lv_top    = 0._wp  ! max.ver.lenghtscale for obs.weights
real(wp)          :: lvrad     = 0._wp  ! ver.lenghtscale for radiances
logical           :: adap_loc = .false. ! use adapt. localization
real              :: nobs_gp = 100.0    ! numb. of eff. obs/gridp.(adap loc)

!-------------------------------------------
! additive model error (from 3D-Var B-matrix)
!-------------------------------------------
real(wp)          :: mf        = 0.25   ! B-amplitude for model error
real(wp)          :: mf0       = 0.25   ! B-amplitude for model error (mean)
real(wp)          :: mfgen     = 0.5    ! B-amplitude for ensemble generation
integer           :: m_flag    = 0      ! flag for model error to mean
integer           :: moderr_fc = 0      ! 1:add mod err to fg, 0: to ana

!----------
! inflation
!----------
real(wp)          :: rho        = 1._wp  ! covariance inflation factor
logical           :: adap_rho   = .false.! adaptive inflation
integer           :: apply_rho  = 1      ! apply inflation 1:in LETKF 2:on W
real(wp)          :: adap_rho_l = 0.9    ! lower bound for adap rho
real(wp)          :: adap_rho_u = 10.0   ! upper bound for adap rho
real(wp)          :: rho_max(4) = 10._wp ! max. rho (at 1,10,100,1000 hPa)
real(wp)          :: kappa      = 1.05   ! factor for growth of forecast. cov. infl
real(wp)          :: va         = 1.0    ! error off (analysed) rho-estimate
real(wp)          :: alpha      = 0.0    ! weight of new adaptive computed rho

!---------------------
! adaptive model error
!---------------------
logical           :: adap_R   = .false. ! use adap. R-corr.(local, ens space)
real(wp)          :: adap_R_l   = 0.5   ! lower bound for adap R
real(wp)          :: adap_R_u   = 10.0  ! upper bound for adap R

!------------
! diagnostics
!------------
integer           :: grid_diag  = 0     ! diagn. on coarse grid 0: no , 1: yes
integer           :: rms_diag  = 0      ! rms-stat 0: no, 1: for fg 2: for ana
integer           :: diag_type  = 0     ! rms-stat for 0: det-run 1: mean
integer           :: gp_mat = 0         ! grid point output (W-matrices)
```

```
integer            :: gp_ind(4,ngp_max) = -1 ! W-matrix output grid indices (coarse
integer            :: norm_flag = 0      ! 0:standard norm, 1: energy norm
logical            :: write_mean =.true. ! write mean and spread (GRIB-files)
logical            :: write_gain =.true. ! write gain matrix (i.e. W Y)
logical            :: out_prof   =.false.! printout of profiles

!--------------
! miscellaneous
!--------------
integer            :: ensbc_weights = 0  ! apply weights at boundaries (ens bc only)
integer            :: fix_plev  = 0        ! interpolate to common p-levels
logical            :: q_bound   = .true.   ! apply bounds to humidityvariables
logical            :: no_ps_inc = .false.  ! no surface pressure increment
logical            :: no_tv_inc = .false.  ! no tv increment
logical            :: no_rh_inc = .false.  ! no rh increment
logical            :: no_uv_inc = .false.  ! no uv increment
logical            :: sat_ad    = .false.  ! satur. adj. of ana-incr
logical            :: hyd_bal   = .false.  ! hydrost. balanc. of ana_incr
integer            :: bal_var   = 0        ! 0: balance p; 1 balance t
```

The namelist `/ENKF/` sets up the option for the Ensemble Kalman Filter (Section 9) of KENDA. The following table describes the entries and links them to specific Sections in the scientific documentation.

ENKF-Namelist description needs to be completed!

ENKF-Namelist: Check for undocumented features!

`/ENKF/`-Namelist (1/5): **LETKF-Algorithm Settings**

variable	*default*	*description*
`k_enkf`	0	number of ensemble members (L in Chapter 9)
`gen_ens`	.false.	instead of assimilation, generate random forecast ensemble (only GME, see Section 9.1.5)
`rf`	1	horizontal coarsening factor $f_{hor,r}$ for regular analysis grid (Section 9.2.2)
`rni`	1	number of diamonds n_i: reduced resolution for GME (**not implemented yet!** – analysis weights computed on full horizontal resolution)
`nzr`	1	number of analysis grid height level (n_{zr} in Section 9.2.2)

`lh`	0. km	horizontal lengthscale for **R**-localisation (l_h in Section 9.2.1) what happens if you choose lh=0? -> Hendrik checkt Code
`lv`	0. ln(p)	vertical lengthscale for **R**-localisation (l_v)
`lv_surf`	0. ln(p)	value of `lv` at model surface height (l_v^{surf})
`lv_top`	0. ln(p)	value of `lv` at model top height (l_v^{top})
`lv_rad`	0. ln(p)	vertical lengthscale for radiance observations (cf. also Section 9.2.5)
`adap_loc`	.false.	Use adaptive horizontal localization (only COSMO) (cf. Section 9.2.1) – overwrites effect of `lh`
`nobs_gp`	100.	`adap_loc`: N_{eff} number of effective observations per grid point: Eq. (9.27)
`lh_min`	0.	`adap_loc`: l_h^{min} lower bound for `lh`
`lh_max`	0.	`adap_loc`: l_h^{max} upper bound for `lh`
`rho`	1.	Covariance inflation factor (ρ in Section 9.2.4)
`apply_rho`	1	apply inflation 1: on background ensemble perturbations - Eq. (9.31) 2: on analysis ensemble perturbation weights - Eq. (9.30)
`adap_rho`	.false.	adaptive covariance inflation (cf. Section 9.2.4)
`adap_rho_l`	0.9	`adap_rho`: lower bound for ρ
`adap_rho_u`	10.	`adap_rho`: upper bound for ρ
`rho_max`	(10.,10.,10.,10.)	`adap_rho`: max ρ at 1,10,100,1000 hPa
`kappa`	1.05	`adap_rho`: factor for growth of fc. cov. infl. err (deprecated)
`va`	1.05	`adap_rho`: error of (analysed) rho-estimate (deprecated)
`alpha`	0.	`adap_rho`: weight of new adaptive rho
`adap_R`	.false.	use adaptive **R**-correction (cf. Section 9.3.2) adaptive R-correction: not documented!
`adap_R_l`	0.5	`adap_R`: lower bound
`adap_R_u`	10.	`adap_R`: upper bound

`ensbc_weights`	0	apply additional relaxing weights [1. to 0.] in COSMO-boundary zone (only for COSMO-KENDA, see routine `weights_ensbc`) 0: no 1: yes ensbc_weights: not documented. see routine `weights_ensbc` in `mo_letf`
`fix_plev`	0	interpolate members to common pressure-levels before assimilation (specific to global LETKF for hydrostatic GME/ICON) fix_plev not documented!
`q_bound`	.true.	apply positivity constraints on humidity variables (Section 9.2.6)
`sat_adj`	.false.	apply saturation adjustment (Section 9.2.6)
`hyd_bal`	.false.	hydrostatic balancing of pressure increments (only COSMO, Section 9.2.6)
`bal_var`	0	`hyd_bal`: variable to balance hydrostatically: 0: pressure 1: temperature (not implemented)
`no_ps_inc`	.false.	`gen_ens`: no increments on surface pressure (only during noise-generation for GME-LETKF)
`no_tv_inc`	.false.	no increments on virtual temperature (–""–)
`no_rh_inc`	.false.	no increments on relative humidity (–""–)
`no_uv_inc`	.false.	no increments on horizontal winds (–""–)
`det_run`	0	use deterministic run 0 (`DET_NONE`): none 1 (`DET_UPDATE`): update det. run with EnKF-gain (only COSMO, Section 9.1.6) 2 (`DET_HYBRID`): VarEnKF (only GME, Section 10) 3 (`DET_3DVAR`): independent 3dVar analysis (only GME, Section 7)

`/ENKF/`-Namelist (2/5): **Additive Model Error (only global LETKF)**

`mf`	0.25	**B**-amplitude for model error proxy (f_{add} in Eq. (9.43))
`mf0`	0.25	**B**-amplitude for model error (mean) mf0 fuer twin-experimente. Noch undokumentiert!
`mfgen`	0.5	`gen_ens`: **B**-amplitude for ensemble generation (f_{gen} in Eq. (9.18)

m_flag	0	flag for model error to mean m_flag fuer twin-experimente. Noch undokumentiert!
moderr_fc	0	mf: add model error to 0: forecast ensemble 1: analysis ensemble

/ENKF/-Namelist (3/5): **Steering of atmospheric parameters**

src_fc	0	fc from 0:mean, 1:psas fc src_fc undokumentiert
par_read	" "	Parameters to read from GRIB-files COSMO: "pp t u v w q qcl qci tsurf z0 plcov lai rootdp vio3 hmo3 t_s t_snow w_i qv_s w_snow t_so w_so freshsnw for_e for_d rho_snow" + par_read GME: "ps t tsurf q u v qcl qci" + par_read ICON: "pf t tsurf q u v qcl qci" + par_read
par_fc_en	" "	par_read: Parameters to take from forecast ensemble COSMO: "pp t q u v w qcl qci tsurf t_s t_snow t_so w_so w_i w_snow" + par_fc_en GME (also apply H): "ps t tsurf q u v qcl qci" + par_fc_en ICON (also apply H): "pf t tsurf q u v qcl qci" + par_fc_en
par_fc_mn	" "	par_read: Parameters to take from forecast mean COSMO: None + par_fcm GME / ICON (also apply H): None + par_fcm
par_fcm	" "	Parameters to take from forecast mean COSMO: None + par_fcm GME / ICON: None + par_fcm UNKLAR: welcher dieser beiden Parameter ist gueltig? Welcher macht was???
par_fc_det	" "	par_read: Parameters to take from deterministic run (only COSMO with det-update)

`par_trans`	`" "`	Parameters to update (transform) in analysis (LETKF and VarEnKF, Section 9.2.3)
		COSMO:
		`"pp t q u v w qcl qci" + par_trans`
		GME:
		`"ps geoh tsurf q u v qcl qci" + par_trans`
		ICON:
		`"pf t tsurf q u v qcl qci" + par_trans`

`par_ana`	`" "`	Parameters to take from analysis par_ana possibly deprecated

`par_fce`	`" "`	Parameters to non-update (take from forecast ensemble and pass) in analysis (Section 9.2.3)
		par_fce not documented
		COSMO:
		`"qr qs qg" + par_fce`
		GME / ICON:
		`"qr qs" + par_fce`

`par_write`	`" "`	Parameters to write in output GRIB-files
		COSMO:
		`"pp t u v w q qcl qci tsurf`
		`z0 plcov lai rootdp vio3 hmo3`
		`t_s t_snow w_i qv_s w_snow t_so w_so`
		`freshsnw for_e for_d rho_snow" + par_write`
		GME:
		`"ps t q u v qcl qci" + par_write`
		ICON:
		`"pf t q u v qcl qci" + par_write`

`par_diagn`	`" "`	Parameters for diagnostics (see subtable (5/5) below)
		COSMO:
		`"pp t u v w q qcl qci tsurf qr qs qg`
		`z0 plcov lai`
		`rootdp vio3 hmo3 t_s t_snow w_i qv_s`
		`w_snow t_so w_so`
		`freshsnw for_e for_d rho_snow" + par_diagn`
		GME:
		`"ps t tsurf q u v qcl qci" + par_diagn`
		ICON:
		`"pf t tsurf q u v qcl qci" + par_diagn`

`/ENKF/`-Namelist (4/5): **Input/Output**

`path_diag`	`" "`	path for diagnostics

fce_fname	specific	forecast ensemble file basename COSMO-KENDA: lff_FCR_TIMEMMSS_ GME/ICON-(Var)EnKF: gff_FCR_TIME_
ane_fname	specific	analysis ensemble file basename COSMO-KENDA: laf_ANA_TIMEMMSS_ GME/ICON-(Var)EnKF: gaf_ANA_TIME_
ana_file	specific	deterministic analysis file basename COSMO-KENDA: laf_ANA_TIMEMMSS_ GME/ICON-(Var)EnKF: gaf_ANA_TIME_
invar_ens	" "	invariant field file (ICON?) invar_ens: UNDOKUMENTIERT
ana_read	" "	3dVar analysis to compare with (**deprecated**) ana_read: Deprecated parameter. Delete here when removed from fortran-code
grid_file	" "	grid metadata (ICON)
fof_prefix	" "	prefix for additional feedback files apart from default fof-files recommended value: "fof" takes stringlist as value, e.g. "fof rad"
rename_diag	.false.	(deprecated)

/ENKF/-Namelist (5/5): **Diagnostics**

grid_diag	0	diagnose on coarse grid 0: no 1: yes
rms_diag	0	RMSE-Statistics in observation space (**deprecated**) 0: no 1: for first guess 2: for analysis rms_diag: Deprecated parameter. Delete here when removed from fortran-code
diag_type	0	(**deprecated**) RMSE-Statistics for 0: deterministic run 1: mean diag_type: Deprecated parameter. Delete here when removed from fortran-code

gp_mat	0	binary grid point output of $\mathbf{W}^a$ 0: no 1: yes
gp_ind	-1	gp_mat: $\mathbf{W}^a$-output grid indices (coarse grid)
norm_flag	0	(possibly deprecated) 0: standard norm 1: energy norm
write_mean	.true.	write ensemble mean and spread as GRIB-files (forecast and analysis)
write_gain	.true.	write gain matrix ($\mathbf{W}^a\mathbf{Y}^b$) as NetCDF-file
out_prof	.false.	printout of profiles (cf. Section 16.26)

16.28 /RADAR_OBS/ (Radar observation operator)

This namelist is defined in module mo_radar .

```
  !---------------------
  ! Namelist /RADAR_OBS/
  !---------------------
  integer  :: use_refl   = STAT_ACTIVE ! radar reflectivity usage flag
  integer  :: use_radvel = STAT_ACTIVE ! radial velocity    usage flag
  integer  :: iprintout  =   0         ! steering of printout
  logical  :: split_rprt = .true.      ! temporarily split reports
  logical  :: join_rprt  = .true.      ! join reports again
  logical  :: dealias_fg = .true.      ! dealias radial wind (by first guess)
  real(wp) :: chk_alias  = 2._wp       ! check dealiasing (compare to spread)
  real(wp) :: ofg_alias  = 0.7_wp      ! check dealiasing (compare to o-fg)
  !-----------------------------------------------
  ! +++ CURRENTLY not used: min/max_range/xdist +++
  !-----------------------------------------------
! real(wp) :: min_range  =   0._wp     ! minimum range                 (km)
! real(wp) :: max_range  = 999._wp     ! maximum range                 (km)
! real(wp) :: min_hdist  =   0._wp     ! minimum hor.  distance between rays
! real(wp) :: min_vdist  =   0._wp     ! minimum vert. distance between rays
! real(wp) :: min_rdist  =   0._wp     ! minimum distance within a ray  (km)
```

The namelist group defines parameters specific to the volume radar observation operator. This namelist and observation operator is used only in the COSMO-LETKF, not in the 3dVar.

/RADAR_OBS/-Namelist

variable	*default*	*description*

use_refl	STAT_ACTIVE	use radar reflectivity (flag)
use_radvel	STAT_ACTIVE	use radar radial velocity (flag)
iprintout	0	printout of radar-obs-processing (0: no / 1: yes)
split_rprt	.true.	temporarily split reports
join_rprt	.true.	temporarily join reports again radar: split_rprt und join_rprt undokumentiert
dealias_fg	.true.	dealias radial wind (by first guess) radar: dealiasing (und checks!) undokumentiert!
chk_alias	2.	check dealiasing (compare to spread)
ofg_alias	0.7	check dealiasing (compare to obs minus first guess)

16.29 /FOF_INPUT/ (Interpretation of fof-file input from COSMO)

This namelist is defined in module mo_fdbk_in :

```
!---------------------
! namelist /FOF_INPUT/
!---------------------
logical  :: flag_dataset    = .false. ! set FL_DATASET for rejected data
logical  :: rm_fg_check     = .false. ! remove first guess check
real(sp) :: ps_obs_error    = 0.      ! <0: factor on, >0: value of PS obs.
error
real(sp) :: uv_obs_error    = 0.      ! <0: factor on, >0: value of wind
obs.error
real(sp) :: radvel_obs_error= 0.      ! <0: factor on, >0: value of radial wind
oe
real(sp) :: t_obs_error     = 0.      ! <0: factor on, >0: value of
temp.obs.error
real(sp) :: rh_obs_error    = 0.      ! <0: factor on, >0: value of wind
obs.error
real(sp) :: rh2m_obs_error  = 0.      ! <0: factor on, >0: value of
temp.obs.error
```

The namelist group defines how the ensemble feedback files in the COSMO-LETKF are interpreted. This namelist and observation operator is used only in the COSMO-LETKF, not in the 3dVar.

`/FOF_INPUT/`-Namelist

variable	*default*	*description*
`ps_obs_error`	0.	if $\neq$ 0.: set value for *surface pressure* observation error standard deviation $\sigma_{obs,used}$ for $\mathbf{R}$ (Section 9.3) using method a) or b):
		a) `ps_obs_error > 0.`: set absolute standard deviation $\sigma_{obs,used}$ = `ps_obs_error`
		b) `ps_obs_error < 0.`: factor on the $\sigma_{obs,fof}$ of the feedback file: $\sigma_{obs,used} = \sigma_{obs,fof} \cdot (-$ `ps_obs_error` $)$
`uv_obs_error`	0.	*horizontal wind observation* error: as for `ps_obs_error`. also:
`radvel_obs_error`	0.	*radial velocity* (from radar observations)
`t_obs_error`	0.	*temperature*
`rh_obs_error`	0.	*relative humidity*
`rh2m_obs_error`	0.	*relative humidity at surface*
`flag_dataset`	.false.	for debugging: set all checks to `FL_dataset`, so that any degradation of the state variable present in the input file may be traced back
`rm_fg_check`	.false	reset the effect of the first guess check rm_fg_check: the effect of WHICH first guess check is reset??

Chapter 17

Observation Handling

Observation handling chapter: Maybe put it into better context?

17.1 Classification of Observations

In order to classify observation types the following identifiers are specified in the observational data structures: `observation-type`, `data-base-identifier`, `BUFR-type`, `BUFR-subtype`, `code-type`, and `module-type`.

ID	obstype	module type	Comment
1	SYNOP	SYNOP/COSMO	
2	AIREP	AIREP/COSMO	Aircraft observations
3	SATOB	AMV	Atmospheric wind vectors.
4	DRIBU	SYNOP	
5	TEMP	TEMP/COSMO	
6	PILOT	TEMP/COSMO	
7	SATEM	–	
8	PAOB	SYNOP	Southern Hemispheric bogus observations, available until 2009
9	SCATT	SYNOP	Scatterometer data.
10	RAD	TOVS	Satellite radiance observations.
11	GPSRO	GPSRO	GNSS Radio occultations (bending angles).
12	GPSGB	GPSGB/COSMO	GNSS Ground based observations.
13	RADAR	RADAR	Volume radar observations.
14	POWER	–	Renewable energy data (wind, solar power).

Table 17.1: (CMA) Report Type Identifiers used in the DWD 3dVar.

In the 3dVar observations are classified according to their `observation-type` (cf. Table 17.1) and `data-base-identifier` ('Datenbankkennzahlen'). Observation type identifiers 1 to 7 are identical with whose formally used in the Optimum Interpolation scheme. At ECMWF (and therefore in the ECMWF ODB tables) observations are classified according

to `observation-type` and `code-type`. Observation types 1-9 correspond with whose used in the 3dVar. In addition, observations may be classified according to their BUFR type (WMO specification) and BUFR subtype. Finally a `module-type` is defined, specifying which (Fortran-90-)module holds the observation type specific code. This information is only used within the 3dVar. Some observation types may be associated alternatively with the module COSMO. In this case the formulation of the COSMO model (nudging or KENDA/LETKF) is used for verification purposes (MEC). These formulation does not provide the adjoint code and thus cannot be used in the variational scheme.

Tables of values of the identifiers are defined within the 3dVar in the tables and modules: : `observation-types`; : `data-base-identifier`; : `BUFR-type`; : `BUFR-subtype` and and `code-type` as well as mappings between `data-base-identifier`, `BUFR-subtype` and `code-type`.

DWD dbkz	BUFR type	BUFR subtype	CMA obstype	CMA codetype	3Dvar module	description
0	0	1	1	11	1	surface, SYNOP Sect.1-4 manual+PAST
1	0	140	1	140	1	surface, METAR
2	0	1	1	11	1	surface, SYNOP Sect.5
3	0	-	-	-	0	climatol.data, SYNOP, DWD only
4	0	-	-	-	0	surface, WEHI (DWD)
5	0	1	1	11	1	surface, SYNOP Sect.5, from 1.4.2001
9	0	-	-	-	0	surface, PSEUDO-SYNOP (from TEMP)
11	0	-	-	-	0	agr.met. AGRO
12	0	-	-	-	0	agr.met. PHAEN
13	0	-	-	-	0	agr.met. PHYTOPAT
16	10	-	-	-	0	radioact. RADI
17	10	-	-	-	0	radioact. RADA
32	0	-	-	-	0	verification, SYNOP or SHIP obs-ana
33	0	-	-	-	0	verification, Pseudo-SYNOP - ana
64	0	-	-	-	0	climatol.data, CLIMAT
65	0	-	-	-	0	climatol.data, EILKLIMA
66	0	-	-	-	0	climatol.data, CLIMAT Sect. 1-4
67	0	-	-	-	0	climatol.data, CLIMAT Sect. 5
128	0	3	1	14	1	surface, SYNOP Sect 1-3 autom.
256	1	9	1	21	1	SHIP manuell
320	1	-	-	-	0	climatol.data, CLIMAT SHIP
384	1	13	1	24	1	SHIP, autom.
385	1	21	4	165	1	BUOY
386	253	164	8	180	1	PAOB
400	1	-	1	-	0	verification, DRIBU
512	2	91	6	32	2	upper-air, PILOT Part A
513	2	91	6	32	2	upper-air, PILOT Part B
514	2	91	6	32	2	upper-air, PILOT Part C
515	2	91	6	32	2	upper-air, PILOT Part D
520	2	101	5	35	2	upper-air, TEMP Part A
521	2	101	5	35	2	upper-air, TEMP Part B
522	2	101	5	35	2	upper-air, TEMP Part C
523	2	101	5	35	2	upper-air, TEMP Part D
528	4	-	2	41	2	upper-air, CODAR
529	4	144	2	144	2	upper-air, AMDAR
530	4	142	2	-	2	upper-air, AIREP
531	3	-	-	-	0	upper-air, TEMP from SATEM
532	4	145	2	145	2	upper-air, ACARS Lufthansa
533	4	145	2	145	2	upper-air, ACARS USA
534	4	145	2	145	2	upper-air, ACARS Europe (Bracknell)
535	4	145	2	145	2	upper-air, ACARS other
536	2	101	5	35	2	upper-air, merged TEMP, PILOT
537	2	-	-	-	0	Pseudo TEMPS, Pseudo TEMPS, land
544	2	-	-	-	0	verification, TEMP obs-ana
545	2	-	-	-	0	verification, PILOT obs-ana
546	2	-	-	-	0	verification, TEMP obs-ana
547	4	-	-	-	0	verification, Aircraft obs-ana
553	2				0	wind profiler, USA

17.2 Steering of Observation Handling

General parameters (area, hight and time range, bounds for first guess check, etc.) which are applicable to each observation type are specified by namelist `/REPORT/` (Section 16.2). More complex rules may be specified by namelist `/RULES/` (Section 16.3). Thinning is steered by `/THINNING/`. Observation type specific Parameters are set by the namelists `/SYNOP_OBS/` (16.8), `/TEMP_OBS` (16.7), `/GPS_RO/` (16.13), `/RO_VDA/` (16.14), `/TOVS_OBS` (16.9), `/AMV_OBS` (16.11),`/AIREP_OBS` (16.12). Data type specific settings are given below.

17.2.1 SATOB (AMV) - parameter settings

name	sat.id.	code	dbkz	thinning (h)	(v)	QI	area	time	method
Eumetsat			1704	ni=48: ca. 160 km	40 hPa	0.5		+− 1 : 30h	
GOES			1705	ni=48: ca. 160 km	40 hPa	0.5		+− 1 : 30h	
Modis			1706	ni=128: ca. 60 km	40 hPa	%		+− 1 : 30h	
				*2	*2	*2		*1	

specified by (namelist)
*1 /REPORT/
*2 /THINNING

QI is the required quality index (per cent confidence) reported in the AMV BUFR file.

Computational method (method) is:

1 Wind derived from cloud motion observed in the infrared c
2 Wind derived from cloud motion observed in the visible cha
3 Wind derived from motion observed in the water vapour ch
4 Wind derived from motion observed in a combination of s channels
5 Wind derived from motion observed in the water vapour cl in clear air
6 Wind derived from motion observed in the ozone channel
7 Wind derived from cloud motion observed in water vapour c

17.3 Bookkeeping

17.4 Processing

17.4.1 Input

17.4.2 First Guess Check

17.4.3 Analysis

17.4.4 Feedback

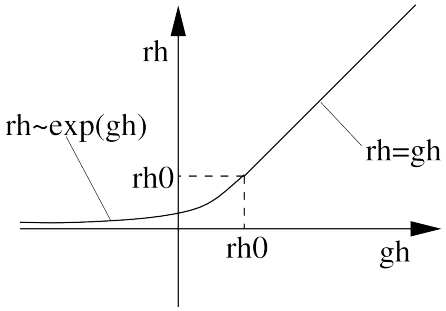
17.4.5 Specific Observation Operators

TOVS (Satellite Radiances)

TOVS (Satellite Radiances)

The processing of the TOVS satellite radiances is performed by the RTTOV-7 fast radiative transfer model. The model performs simulations of radiances for satellite infrared and microwave nadir scanning radiometers. The simulated radiance depends on atmospheric profiles of temperature and gas concentrations and on surface and cloud properties, which are provided for in put as arrays (state vector). For RTTOV-7, the only variable gas components are water vapor and ozone. The coefficients and the profiles are specified on 43 pressure levels. The RTTOV-7 model also provides the Jacobean, i.e. the gradient of the radiance with respect to the state vector variables for the values of the input state vector values. It describes the change of the radiance of each individual channel to be measured at the top of the atmosphere with respect to a unit perturbation of any level of the input profile or the surface and cloud parameters. The value of the gradient indicates for each channel in which atmospheric levels changes in the atmospheric quantity affects the radiance most. The RTTOV model can simulate both clear sky and cloudy radiance, using an approximate form of the radiative transfer equation (Actually, all ATOVS data are supposed to be clear-sky data). The essential quantities to be approximated within the radiative transfer equation are the transmittances and the emissivities. The transmittances are obtained from the variables of the input profile vector by linear regression in optical depth. The emissivities, two two fast surface emissivity routines are implemented: ISEM for infrared channels and FASTEM-1 and FASTEM-2 for the microwave channels, with FASTEM-2 making a better correction for the reflected radiation at the surface. The FASTEM model requires the 2m vector wind speed to be specified in the state vector. At the current state of implementation, the wind speed is set to zero in all channels.

Generalized humidity



Negative values of the relative humidity are a problem that might arise within the 3dVar. Besides describing an unphysical situation, it generally causes a failure of the subroutines that convert the 3dVar physical units to the units required by the RTTOV package. This problem is avoided by introduction of a generalized humidity, which is allowed to have negative values. For a value of relative humidity larger than a certain value rh0 (to be specified in the namelist /CNTRLVAR/) relative humidity equals generalized humidity. In case the generalized humidity is smaller than rh0 or even negative, the corresponding value of the relative humidity is adapted exponentially, allowing positive values only. The transition between the linear and exponential dependence is done in a smooth way.
The correspondence between relative humidity and generalized humidity is unique, which allows a transformation of the humidity variables in both ways: To be consistent, the first guess relative humidity profiles are converted to generalized humidity by subroutine gh_rh in mo_cntrlvar. Thus, even when the original profile has positive humidity values only, the resulting profile of generalized humidity may have negative values. Within the variational algorithm of 3dvar itself, negative values of the generalized humidity cause no difficulties. The transformation of the generalized humidity to relative humidity is performed just before the conversion of the units of the atmospheric variables (i.e. conversion of relative humidity and virtual temperature to specific humidity and temperature) in mo_tovs. For the transformation to relative humidity in particular of first guess profiles the lower bound rh0 of the linear range can additionally be specified, either as relative or specific humidity (rh0fg, q0fg). This option is necessary, as the relative humidity in the tropics can be quite small. In order not to unnecessarily distort the profile, the smallest value of the lower bound in terms of relative or specific humidity is applied.

Dummy variables

The RTTOV-operator requires atmospheric columns of specific humidity and temperature up to a pressure level of $0.1hP$. This pressure level is beyond the upper edge of the GME model ($10hPa$) and also higher than the levels for which the humidity is assimilated within the 3dVar ($\approx 250hPa$). Therefore input values for the RTTOV-routine have to be specified for the temperature between $10hPa$ and $0.1hP$ as well as for the specific humidity between $250hPa$ and $0.1hP$. This specification is done as dummy variables: For the pressure levels mentioned above, 1DVAR profiles of temperature and specific humidity are used as a background. The variable var% av which contains the 3dVar profiles of virtual temperature and relative humidity for the lower pressure levels is loaded with a Gaussian distribution around this background. The width of this distribution equals the background error of temperature and specific humidity, respectively. Its values can be specified in the namelist /TOVS_OBS/ and are actually estimated from the

background error covariance matrix as used by the UK-Met-office. For the pressure levels simulated by dummy variables, the 3dVar background error covariance matrix **B** is a unit diagonal. To provide a background error covariance matrix in physical space $\mathbf{HBH}^t$ with an appropriate specification of the background errors of the dummy variables, the corresponding entries of the tangential operator **H** are multiplied with the background errors of temperature and specific humidity, respectively. The specification of background errors for the specific humidity has a peculiarity: To circumvent the problem of negative humidity, the error is specified to be in $\ln(q)$, resulting in an asymmetric distribution for q which suppresses values smaller than the 1D-Var background. A third quantity treated as a dummy variable is the surface temperature. Its background value is provided by the 1D-Var and background errors for different types of surfaces (land, sea, ice) can be specified in the namelist TOVS_OBS. The values of the dummy variables are modified consistently with the regular, non-dummy values within the variational algorithm of the 3dVar, but are not fed back to the analysis.

Change of variables

For the forward calculation of radiances the RTTOV-7 package uses atmospheric columns of temperature and specific humidity. The units of the corresponding control variables of the 3dVar, virtual humidity and relative (resp. generalized) humidity therefore have to be converted to RTTOV units. The transformation has to be applied both to the atmospheric columns that are the input of the RTTOV package as well as to the tangent-linear observation operator as output. For pressure levels with no dummy atmospheric variables present and for the surface variables (t_2m, q_2m,tv_2m,rh_2m) the transformation both of the columns and the tangent-linear is performed by the subroutine tq_tvgh in mo_cntrlvar. In the top atmosphere pressure levels, both specific humidity and temperature are dummy variables, therefore a conversion of units is not necessary in these levels. For certain intermediate pressure levels the specific humidity is the only atmospheric dummy variable with only the virtual temperature to be transformed. The conversion is performed by the function t_tv_q (forward calculation) and by the subroutine t_tv_q_adj (tangent linear calculation). For the determination of the tangent linear operator in 3dVar units one has to take the conversion of physical units into account. Therefore, the Jacobean provided as output by RTTOVK is to be multiplied by the appropriate derivatives. At this point, it has to be taken into account that the background error for the specific humidity is specified in $\ln(q)$, whereas the tangent linear matrix **H** contains derivatives with respect to q.

Selection of channels

The selection of TOVS instruments and channels to be processed within 3dVar can be done in different ways:

- instrument channel number
- ECMWF channel number, identifying all channels on a given platform
- selection of the ECMWF channels by 1D-Var

- selection of the 1D-Var channels within 3dVar (specified in namelist /TOVS_OBS/)

HIRS channel number	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
ECMWF channel number	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1DVar	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
example: selection from 1DVar	-	1	2	3	4	5	6	7	-	8	9	10	11	12	13

MSU channel number	2	3	4
ECMWF channel number	22	23	24
1DVar	14	15	16
example: selection from 1DVar	-	-	-

SSU channel number	1	2	3
ECMWF channel number	25	26	27
1DVar	19	20	21
example: selection from 1DVar	-	-	-

AMSU-A channel number	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
ECMWF channel number	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42
1DVar	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36
example: selection from 1DVar	-	-	14	15	16	17	18	19	20	21	22	23	24	-	-

AMSU-B channel number	1	2	3	4
ECMWF channel number	43	44	45	46
1DVar	37	38	39	40
example: selection from 1DVar	-	-	-	-

The above example selection corresponds to the entry

```
ionly = 2,3,4,5,6,7,8,10,11,12,13,14,15,24,25,26,27,28,29,30,31,32,33,34
inot  = 1,9,35,36,37,38,39,40
```

in the namelist /TOVS_OBS/.

Routine RTTOVI

The package RTTVI is the initialization routine for the radiance forward and tangent-linear operators RTTOV and RTTVK. It is called only once for all platforms, satellites and instruments. A few essential parameters of the RTTOV-routines are set in the module RTTOV7_MOD_CPARAM in 3dvar/rttov7/rttov7src/ This includes

parameter	value	content
jplev	43	number of pressure levels
jpch	2378	maximal number of tovs channels
jpchpf	jppf*jpchus	maximal number of channels*profiles
jpnsat	9	maximal number of sensors to be used
jpnav	4	maximal number of profile variables (temperature, specific humidity, ozone, liquid water)
jpnsav	5	number of surface air variables (2m temperature, 2m specific humidity, surface pressure, 2m u wind, 2m v wind)
jpnssv	6	number of skin variables (radiative skin temperature + 5 FASTEM-2 land coefficients)
jpncv	2	number of cloud variables (cloud top pressure, cloud fractional cover)

The type of radiance measurement is specified by the numbers platform id, satellite id, and sensor id. The platform id labels the different satellite types (e.g. NOAA) that can be simulated by RTTOV. The satellite id specifies the individual satellites of a type (e.g. NOAA 16), the instrument id the measurement instrument itself (e.g. HIRS). To produce a series of NRTTOVID radiance simulations of different instruments, the first NRTTOVID entries of the input vectors PLATFORM, SATELLITE and INSTRUMENT have to be specified in the desired way. The sizes if the input arrays are determined by jpnsat, jpch (both read from RTTOV7_MOD_CPARAM), jppf, nlev (to be specified, actual values are jppf=1, nlev=jplev=43). The output of RTTVI consists of:

- lower and upper bounds for the profiles of temperature, specific humidity and ozone
- number of profiles to be processed in one call of RTTOV simultaneously
- number of sensors available
- number of pressure levels of the input profiles
- number of channels available
- number of profile variables, surface skin and cloud variables

The field IVCH(jpch,jpnsat) is initialized to zero on input. As output, IVCH(jpch,jpnsat) contains the respective valid channels numbers for each of the sensors 1...jpnsat. The actual values of the optional parameter NIU1 are initialized to zero. Nonzero values of this parameter define the Fortran unit numbers of through which the coefficient files are read in. Actually, the ozone profile used in 3dVar is an updated reference profile (Roger Saunders Jan 2000) and is read from rt_coef_fmt.dat in /3dvar/data/rttov6/rttovdata/. Regarding the profile of the specific humidity, the upper limits of the first 18 pressure levels have been modified in order to guarantee the correct evaluation of the tangent linear operator. For further details of RTTVI, see RTTOV-7 users guide.

Routine RTTOV

The routine RTTOV determines the top atmosphere simulated radiances and brightness temperatures, given the profiles and surface values of several atmospheric variables and parameters. The routine has to be called for every sensor in the sequence used when calling the initialization routine RTTVI before. The input sequence number KSAT refers to this sequence (KSAT=1,2,3,...). Other input parameters are

- number KNPF of profiles to be processed within one call of RTTOV (actual value KNPF=1)
- solar angles
- surface type index
- arrays of profiles, surface 2m values, surface skin values, and clouds

The surface skin array contains the radiative skin temperature (actually modeled by a dummy variable with different background errors for sea, land and ice) and five FASTEM-2 land coefficients. The values assigned to these coefficients correspond to the surface type 'summer land surface', 'open grass' as listed in Table 3, RTTOV-7 Science and Validation Report. The cloud array is not determined by model variables, but set to default values: Cloud top pressure (hPa): PCV(1,:) = 500 and cloud fractional cover : PCV(2,:) = 0 (prossing clear sky observations only). The output includes brightness temperatures, radiances, surface to space transmittance and radiances related to overcast data (overcast cloudy radiance, overcast radiance from cloud top, and layer to space transmittances). At the present state of development, only cloud-free observations are processed, therefore the RTTOV quantities related to overcast profiles is not relevant at the moment.

The input parameter PEMIS describes the surface emissivity. Its value is set to zero for infrared channels (for the actual setting: HIRS channels with the RTTOV series number KSAT=1), and PEMIS=-1 for microwave channels (for the actual setting: AMSU-A and AMSU-B channels with the RTTOV series number KSAT=2 and 3, respectively). These values for PEMIS cause the use of ISEM coefficients for the calculation of infrared channels and the use of FASTEM-2 coefficients for the calculation of microwave data, as described in Table 4 of the RTTOV-7 Users Guide.

Routine RTTVK

The routine RTTVK, too, simulates the top atmosphere radiances and brightness temperatures, additionally it contains the tangent linear operator to determine the gradients of the brightness temperature or radiance in each instrument channel with respect to the input atmospheric, 2m- ,surface and cloud variables. The interfaces basically resembles the interface of the routine RTTOV, but is augmented by the output arrays of the matrix of derivatives. An additional switch (KINRAD) determines whether this array refers to radiances or brightness temperatures. The input values of the surface emissivity array PEMIS_D are the same as for the forward routine RTTOV. The array of the gradients with respect to the surface emissivity PEMIS is initialized to zero.

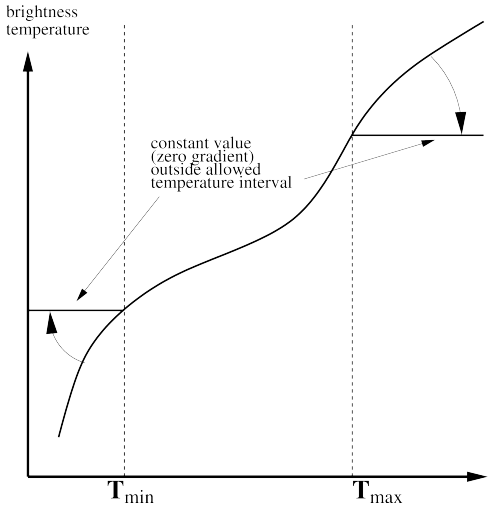
Interpolation of surface and 2m variables

The input of the RTTOV routine includes the 2m values of temperature, specific humidity and the surface pressure. The value of the surface pressure at the observation points is actually read from 1D-Var feedback file. The 2m temperature and 2m specific humidity are interpolated linearly if the surface pressure is sandwiched by two pressure level of the RTTOV: The ratio of the upper and lower pressure differences is the weight for the linear combination of the temperature and humidity values of the corresponding two pressure levels. In case of a surface pressure higher than that of the lowest RTTOV model level, the corresponding values of the lowest RTTOV pressure level are taken as approximations to the surface values.

The surface temperature is modeled as a dummy variable around a 1D-Var background with an background error depending on the type of surface (sea, land, ice). The size of these errors can be specified in the namelist /TOVS_OBS/. Within RTTOV, the 2m vector wind speed (u,v) is used by the FASTEM-1/2 model to simulate the microwave sea surface emissivity. In case this emissivity is not computed (which is the actual setting), the wind vector is set to zero as input to RTTOV.

Bounds of variables (incl. additional limits for q)

The transmittance calculations of the RTTOV-7 routine are valid only if the values of the input profile are within a certain range: The profiles of temperature, specific humidity and ozone are limited by lower and upper bounds. These bounds are specified for each pressure level and are provided as output of the initialization routine RTTVI (OTMIN, OTMAX, OQMIN, OQMAX, OOZMIN, OOZMAX, respectively). Before calling the forward and tangent linear operator with the actual profiles of temperature, specific humidity and ozone, the lower and upper bounds of the respective profile are checked in the subroutine rttov_bounds in mo_rttov. In case a value exceeds the lower or upper bounds, it is reset to the corresponding limit, and the resulting H matrix is modified by assuming zero gradient for the corresponding pressure level.



The upper limit of the specific humidity, however, has to be handled more strict than recommended by the RTTOV-7 manual. Gradient tests of the tangent linear observation operator with a constant profiles of the specific humidity scanning the recommended range show inaccurate results for the Jacobian for values of the specific humidity considerable less the given upper limit. This deviation affects the channels HIRS 7, 8, 10, 11, 12, 13, 14 and AMSU-A 10, 11, 12, 13, 14 and the pressure levels 1 to 18. To guarantee the correct calculation of the tangent model, the upper limit of the specific humidity (specified in call_rttvi in mo_rttov after calling the initialization

routine RTTVI) is changed in the following way:

level number	pressure (hPa)	Qmax kg/kg (RTTOV-7)	Qmax kg/kg(corrected)
1	0.1	4.38e-05	1.0e-05
2	0.3	4.65e-05	1.0e-05
3	0.7	4.61e-05	0.9e-05
4	1.4	4.51e-05	0.9e-05
5	2.6	4.29e-05	0.9e-05
6	4.4	4.26e-05	0.9e-05
7	7.0	4.36e-05	0.9e-05
8	10.4	4.35e-05	0.9e-05
9	14.8	4.01e-05	0.9e-05
10	20.4	4.03e-05	1.0e-05
11	27.3	4.18e-05	1.0e-05
12	35.5	3.62e-05	1.4e-05
13	45.3	3.43e-05	1.4e-05
14	56.7	3.33e-05	1.5e-05
15	70.0	3.23e-05	1.4e-05
16	85.2	3.01e-05	1.4e-05
17	102.1	2.90e-05	1.4e-05
18	122.9	3.58e-05	1.6e-05

Use of 1dvar profiles

For consistency checks with the 1D-Var the 1D-Var background read from the 1D-Var feedback file can be used instead of the interpolated GME columns by setting the flag use_1dvar_fg in namelist /TOVS_OBS/ to .TRUE. By doing this, the consistent simulation of radiances within the 1D-Var and 3dVar can be checked.

Background error fields

Horizontal plots of the background error in the physical space of TOVS radiance data can be generated by setting the value of nlat in namelist DEF_OBS_NML to a nonzero value. The value of nlat is the number of grid points of the field in N-S direction, i.e. nlat=18 generates the background error field on a 10 by 10 degree grid. The output file is a grads data file fg_err_tovs.grads and a grads control file fg_err_tovs_all.grads.ctl (path is specified in code). In the data file, the background error values related to the different TOVS channels are arranged as the z levels of the grads data field. The contribution of individual background error components can be estimated by all other background error components to zero value.

Gradient test

To perform the variational algorithm within the 3dVar, it is necessary to correctly calculate the nonlinear background error matrix in the physical space. For the case of TOVS radiance data, this is accomplished by the tangent linear observation operator(Jacobi

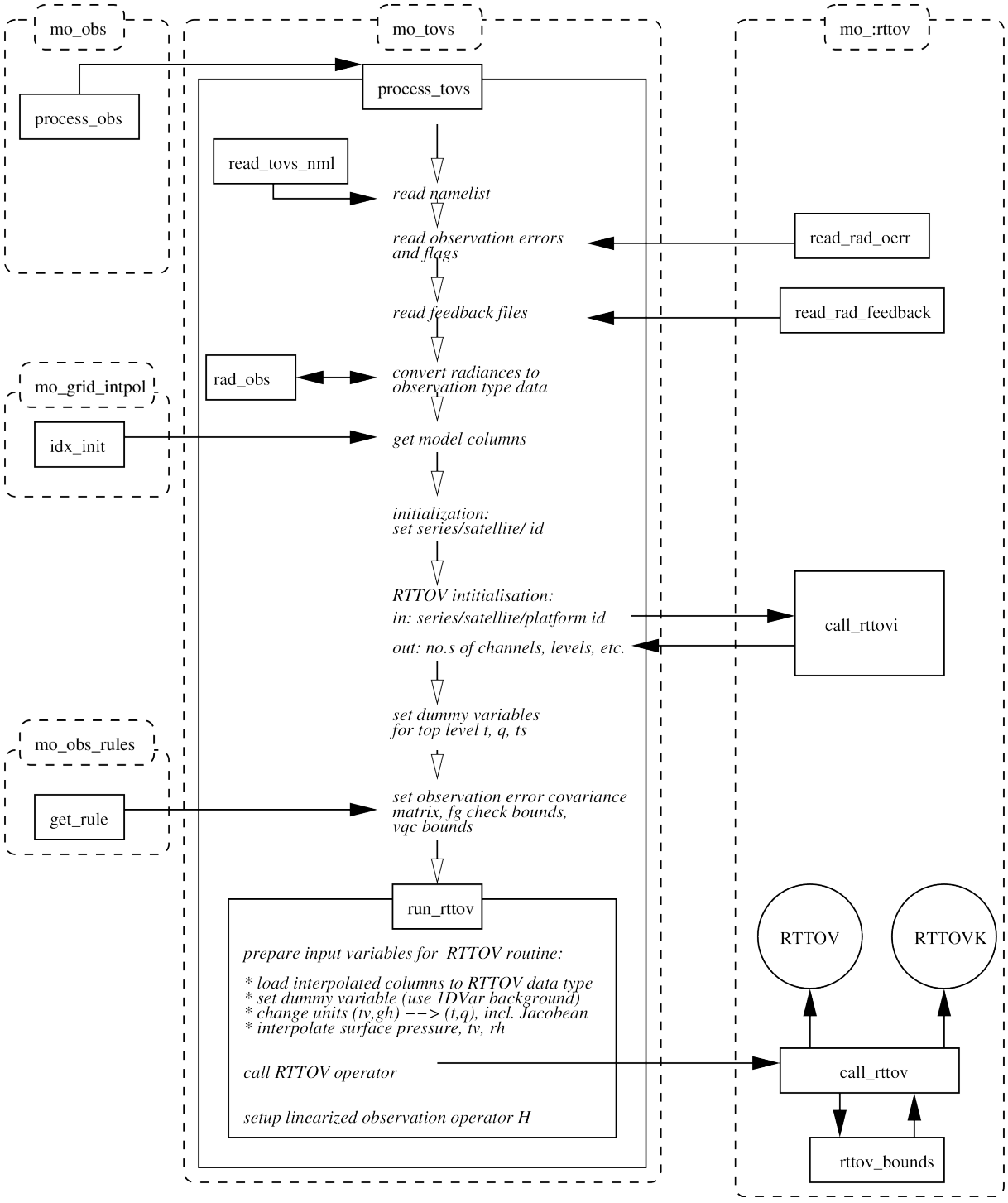
matrix). Its performance can be tested by comparison with the gradient obtained by finite differences. The finite differences increment is a normal distributed random number scaled by the background error times a small scaling factor, to be specified as 'scale' in the namelist /TEST_OBS_OPR/. The scaling factor can either be kept as a fixed value (mode = 'on') or it can be scaled to small values iteratively (mode = 'scale'). If no value for the parameter scale in /TEST_OBS_OPR/ is given, a scale factor of $1.e-3$ is used. The operators (SYNOP, TEMP, TOVS, GPSRO, AMV, OP_X, ..) and quantities (tv,rh,h,u,v,hs,..) to be tested are to be specified. With the entry mode='operator' and mode='quantity' in the namelist, the gradient test is performed to all of them. To test the tangent operator for a particular operator or quantity, an assignment like e.g. operator = 'TEMP' or quantity = 'u' has to be done in the namelist. To test individual arguments of a given operator and quantity, e.g. the value of TEMP virtual temperature tv of a particular pressure level, the specification mode = 'iarg' in the namelist is possible. The number of the argument to be tested (the pressure level in the example) is then to be given as value of the parameter iarg in the namelist. As output the routine returns

- the norm of the random increment vector $|dx|$,
- the norm of the resulting finite difference gradient vector in physical space $|du|$
- the norm of the analytically determined gradient vector in physical space $|dy|$
- the norm of the difference vector of the two gradient vectors $|du - dy|$

The result of the gradient test is considered to be positive in case $|du-dy|$ is several orders of magnitude smaller than $|du|$ and $|dy|$. In case mode='scale' is set, a series of output data with decreasing scale factors is produced. If the analytical gradient is calculated correctly, the series of output values for $|du - dy|$ decreases until it reaches a value small compared to $|du|$ and $|dy|$,

until it reaches a value determined by the numerical inaccuracy.

Sketch of information flow



Assignment of tasks to types of observations for processing:

Task	function	SYNOP	TEMP	TOVS	GPSRO	AMV	OP_X	
TSK_INIT	initialize modules: Set and allocate module variables. Read files and namelists.	x	x	x		x		
TSK_READ	read observations			x				
TSK_SETUP_DIMS	setup dimensions of PSAS-space: Determine number of independent observations. Determine type of independent observations.							
TSK_SETUP_COLS	setup columns:Determine size of interpolation space. Determine type of interpolated quantities. Determine model columns required by the operator.	x	x	x				
TSK_SETUP_FUL0	setup description of PSAS-space: Allocate interpolation space. Determine levels of interpolated quantities. Set R (observational error covariance).		x	x				
TSK_SETUP_FULL	setup description of PSAS-space: Allocate interpolation space. Determine levels of interpolated quantities. Set R (observational error covariance).	x	x	x		x		
TSK_QC_CONS	quality control:consistency					x		
TSK_QC_BLACK	quality control:blacklisting					x		
TSK_QC_CLIM	quality control:climatological					x		
TSK_QC_FC	quality control:obs-fc					x		
TSK_QC_ANA	quality control:obs-ana					x		
TSK_Y	Run nonlinear forward model							
TSK_J	evaluate J, dJ/dx,.	x	x	x		x		
TSK_H	run tangent linear operator	x	x	x		x		
TSK_K	Set H (tangent-linear observation operator)		x	x				1
TSK_XK	broadcast H	x	x	x		x		3
TSK_FC	evaluate J, dJ/dx,.., forecast					x		6

Documentation, RTTOV-6

Further documentation: RTTOV-7 Users Guide, RTTOV-7 Technical Report, RTTOV-7 Science and Validation Report, www.metoffice.com/research/interproj/nwpsaf/rtm/index.html for profile datasets, code and coefficient files updates, documentation, bug reports Version 6 of the RTTOV operator is also implemented and is applied by setting rttov_version = 7 in namelist TOVS_OBS.

17.5 Single Observation Experiments

Currently there is no mechanism to generate artificial observations of arbitrary type at arbitrary locations. The simplest approach for single observation experiments is to use real data, restrict the data usage in the assimilation procedure to one (or a few) observations, and to modify the observed value if required. Suitable namelist settings for these purposes are described below.

17.5.1 Restriction of observation usage

It is recommended to only read the observational data actually used and remove unused data from the lists of observation input files. These lists are passed in namelist group /OBSERVATIONS/:

bufr_files = list of BUFR input file names.
obs_files = list of NetCDF input files converted from BUFR format by the bufrx2netcdf tool.
fdb_files = list of feedback (input) files (from COSMO or a previous run of the assimilation system).

The list of satellite radiance data from SATPP is provided in namelist group /TOVS_OBS/:

feedbk_files = satellite radiance data from SATPP.

There are 2 namelist groups for the steering of observational use:

/REPORT/ works on the observation type level.
/RULES/ works on the single observation level (temperature / wind / pressure level). If

these namelist groups appear multiple times, the latter overwrite any settings of the prior ones. (There are already a lot of them in the default setup).

In order to select a certain observation type (lets say TEMP), a namelist group /REPORT/ which disables all observations can be added after the other groups of this type. Then another one which only enables the TEMP observation type has to be added:

```
&REPORT              use='forget'   ! don't process any obstype
&REPORT type='TEMP' use='active'   ! use TEMP reports
```

Afterwards it is recommended to disable all observed quantities:

```
&RULES
  comment     = 'disable all observed quantities'
  type        =  2    ! modtype TEMP
  gp%use      =  4    ! DISMISS geopotential height observations
   t%use      =  4    ! DISMISS temperature         observations
  uv%use      =  4    ! DISMISS wind                observations
   q%use      =  4    ! DISMISS humidity            observations
   p%use      =  4    ! DISMISS pressure            observations
   o%use      =  4    ! DISMISS other               observations
/
```

Temperature observations within selected pressure bounds may be enabled again:

```
&RULES
  comment     = 'enable temperature within pressure bounds'
  type        = 2         ! modtype TEMP
  plim        = 499 501   ! pressure level range (hPa)
  t%use       = 11        ! ACTIVE
/
```

Finally all observations outside a lat-lon box may be disabled:

```
&RULES
  comment = 'exclude everything outside this box'
  xlonlat = T          ! exclude observations outside this box
  lon     = 36.74     36.76
  lat     = -1.31     -1.29
  use     = 'dismiss'
/
```

Be aware that not the observation type, but the module type is currently used to identify observations in namelist group /RULES/.

Further entries for steering of radiance data on the level of single instruments and channels can be found in the namelist group `/TOVS_OBS_CHAN/`.

17.5.2 Modification of observation input

For single observation studies or double twin experiments it is often recommended not to use the values actually observed, but specific deviations from the true observations or the first guess. This can be achieved by entries '...%prc' in the namelist group /RULES/. Valid entries are:

PRC_ADD_1	=	1	add 1 to the observation
PRC_ADDFG	=	2	add the first guess to the observation..
PRC_ADDOBSER	=	4	add random (normally distributed) error (with standard deviation of the nominal observational error).
PRC_ADDOBSE2	=	8	add random (normally distributed) error (with standard deviation of the nominal observational error divided by sqrt(2)).
PRC_OBS_FG	=	16	set observation to the first guess.
PRC_ADD_R	=	32	add the nominal observational error to the observation.
PRC_SUB_R	=	64	subtract the nominal observational error to the observation.

Different values can be added bit-wise. So '...%prc = 48' would set the observation to the first guess value plus the nominal observational error.

Example:

```
&RULES
  comment     = 'set artificial TEMP temperature observations (fg + obs.error)'
  type        = 2          ! modtype TEMP
  t%prc       = 48         ! ACTIVE
/
```

17.5.3 Diagnostics

For single observation experiments it is desirable to inspect analysis increments as they are proportional to the the background error covariances between observations and background state. This can be done on the bases of the model grid output with respective visualisation tools (plot_grib). In addition it is possible to interpolate first guess, analysis, and analysis increment to specified pressure levels or to extract profiles from the respective model fields on the fly after the assimilation step.

For the global (variational) assimilation system the relevant variables in namelist group /PSAS/ are:

logical	out_aninp	write analysis increment on pressure levels.
logical	out_anp	write analysis on pressure levels.
logical	out_fgp	write first guess on pressure levels.
integer	lev_aninp(50)	pressure levels (in hPa).

Profiles are specified in namelist /PROFILES/:

real	lat_lon	up to 20 latitude / longitude pairs.
integer	ih	horizontal interpolation flag: 1 = nearest neighbour 2 = linear interpolation (default) 3 = area average

In case of area averaging (up to 10) pairs of latitude / longitude pairs must be specified.

In case of the LETKF writing of profiles must be explicitly activated by setting 'out_prof' in namelist /ENKF/ to true. Namelist /PROFILES/ and 'out_aninp' from

/PSAS/ are used as well.

Chapter 18

1D-Var

1D-Var chapter: Maybe put it into better context?

Currently the 3dVar uses a separable background error covariance model. The covariances consists of a product of a horizontal correlation and a vertical covariance matrix. By using only the vertical covariances, observations at different horizontal locations become uncorrelated and the program can be regarded as a set of independent 1D-Var assimilations for eachreport (eg. TEMP report or satellite field of view).

The method may be used for the development of observation operators, as a 1D-Var assimilation allows to assess the responce of the analysis system to do the observations at a given field of view independently from other observations. It is valuable for testing the introduction of new additional control variables (as cloud top, cloud temperature and emissivity models) for adjustable parameters.

In a future setup the 3dVar may be modified so that a 1D-Var is preceeding the 3-dimensioal assimilation. This would allow to obtain first guess values (not necesaryly background values) for additional control parameters for strongly nonlinear processes.

18.1 1D-Var Setup

The 1dvar test mode is activated my setting parameter `var1d` in namelist `/BG_ERROR_OPERATOR/` to .true. . This implicitly modifies the following parameters:

Switch off horizontal covariances

Horizontal covariances have to be represented by the old, strictly separable covariance model. Only here the horizontal covariances can be set to zero by the `var1d` flag.

```
&BG_ERROR_OPERATOR
  ...
  repr_2dh = 0          ! 0: old covariance model
/                       ! 2: new horizontal wavelet based model
```

Horizontal covariances in the preconditioning are switched off by setting the respective namelist variable:

```
&PSAS
  ...
  anaprx = 3   ! 3=PB_1DVAR: no correlations between reports
/
```

Switch off analysis error estimation

Estimation of the analysis error does not make sense in a 1D-Var context and should be switched off:

```
&ANAERROR
  anaerr = 0        ! switch off analysis error estimation
/
```

Skip the post-multiplication

In the 3dVar PSAS scheme optomisation is performed in observational space. The final projection of the analysis increments on the model grid is performed in the post-multiplication step. This does not make sense in a 1D-Var context and should be switched off:

```
&PSAS
  ...
  no_post = T  ! skip post multiplication
/
```

Chapter 19

Computation Environment

This Chapter serves a collection for the description of

19.1 the NUMEX computation environment at DWD

19.2 the Utilities and Test programs within the `var3d`-software package

19.1 DWD Experimentiersystem

DWD Experimentiersystem: Add information about standalone version of the DWD Da Suite?

The interfaces the 3dVar is very similar to the OI. For that reason setting up a 3dVar assimilation cycle is very much similar to an OI cycle. After setting up an assimilation cycle (with mox), the 'Steuerdatei' has to be changed as indicated below:

Run script

The run script `ga3_mwh` is prepared to run the 3dvar instead of the OI. The following variables must be changed to activate the 3dvar and pass the number of tasks (processors, here 96) to use:

`ga3_mwh 2 parm - ntasks_3dvar 96` *instead of* `ntasks`.

`ga3_mwh 2 parm - mode 3dvar` *instead of* `mode ana`.

Constant data

The 3dVar requires several files with constant data. They are passed in the directory `/uhome/arhodin/NUMEX/const/3dvar`. The following line has to be added:

```
all - var - DIR_3DVAR_CONST
cos5:/uhome/arhodin/NUMEX/const/3dvar
```

Namelist files

The facilities of the Experimentiersystem to change namelist variables does not work with the 3dVar namelists (because in the 3dVar namelist groups may appear more than once). Thus changed namelist-files (and for completeness databank request files) must be provided in a dedicated directory (for instance for experiment 5017 in `/uhome/arhodin/NUMEX/5017/const/`:

`ga3_mwh - var - ROU_CONST cos5:/uhome/arhodin/NUMEX/5017/const`

The original (unchanged) files may be copied from ....

Binary

The 3dVar is not yet in the DWD version control system. For that reason the binary (`var3d`) must be provided explicitely, for example:

`ga3_mwh - var - MODUL_3DVAR /uhome/arhodin/NUMEX/5017/bin/var3d`

Feedback files

Feedback information is currently not encoded in BUFR format. Instead a different format (NetCDF, cf. Section 20.3) is used as a replacement for the cof-files and BUFR feedback files.

The verification mode is not yet implemented.

An example script to acess the output of the 3dVar has beed placed in the subdirectory `3dvar/scripts/` of the source tree:

```
#!/usr/bin/csh

#===============================================================================
#
# example script to extract 3D-Var output from experiment-database:
#
#   oa_ga3_mwh.YYYYMMDDHH            Job output
#   ass_ga3_mwh_cofXXXXX.YYYYMMDDHH  NetCDF feedback files for obstype XXXXX
#   ass_ga3_mwh_diag.tar.YYYYMMDDHH  Diagnostic output. Content:
#     monXXXXX.nc                    Monitoring files      for obstype XXXXX
#
#===============================================================================

#-------------------------------------------------
# set experiment number, assimilation date and hour
#-------------------------------------------------

set exp  = 4859
set date = 20040901

foreach hour (00 03 06 09 12 15 18 21)

#-------------
# extract data
#-------------

exp_rou_info     -e $exp ${date}${hour}               # get info

exp_rou_get      -e $exp ${date}${hour} ga3_mwh       # get analysis stdout

exp_cof_rou_info -e $exp ${date}${hour}               # get info on cof-files

exp_cof_rou_get  -e $exp ${date}${hour} ga3_mwh ass   # get cof-files

end
```

19.2 Utility and Test Programs

Besides the 3dVar program itself a number of utility programs have been written (source code in 3dvar/prog). The purpose of these programs is either to provide utilities to pre- or postprocess and investigate input or output of the 3dVar program or to test program components (modules) of the 3dVar. Programs marked as `test` are not intended to be matured applications and may undergo frequent changes if required. Programs marked as `obsolete` are currently not used in the context of the 3dVar.

File	Purpose	Description
var3d	3dVar	3dVar main program.
statistics	Utility	Post-processing-Routine to prepare statistics from feedback fi (NetCDF or ASCII format).
print_grad	Utility	Gather information on the convergence of the PSAS-3dVar fr the Experiment output and write it to a file suitable for plotti Author: Oliver Schmid.
	Utility	Convert the 3dVar feedback files (NetCDF format) to the form (ASCII) file format convention.
bufr2cof	Utility	Convert the OI feedback files (BUFR format) to 3dVar feedba files (ASCII format). Used to compare the usage of reports in t OI and 1D-Var, respectively.
plot_grib	Utility	Plot or investigate GRIB files.
post_grib	Utility	Convert GRIB-files.
post_solver	Utility	Post-processing program to monitor the performance of the solver the set of nonlinear PSAS-equations. (Files solver.inf, solver.da *This program must be rewritten because the file format changed NetCDF.*
post_lnsrch	Utility	Post-processing program to monitor the line search (files solver. solver.dat)
test_bufr	Utility	Investigate the content of a BUFR (observation input) file. T program makes use of the Fortran 90 interface defined in the 3dV
tune_errors	Utility	Plot artificial TOVS data. Author: Detlef Pingel.
interpolate_soil_flake	Utility	Interpolate surface and soil variables to a different (lower) mo resolution with special treatment for soil and flake variabl (19.2.2)
diffgrib	Test	Compare two GRIB-files
test_fg_cov	Test	Test and plot various aspects of the covariance model.
vqc	Test	Test the VQC (Variational Quality Control) implementation.
post_psas	Test	Empty program template for arbitrary purposes.
test_physics	Test	Empty program template for arbitrary purposes.
my_psas_driver	Obsolete	Apply the SAMG algorithm to the DWD-3Dvar B+R matrix. A thor: Tanja Clees, Fraunhofer Gesellschaft.
post_samg	Obsolete	Convert 3dVar feedback files (ASCII-format) to the SAMG (Frau hofer Gesellschaft) format.
test_samg	Obsolete	Test various aspects of the B matrix (SAMG file format)
printgauaw	Obsolete	Print Gaussian grid latitudes (used for spectral models).
scalprod	Obsolete	Test scalar product definitions in spectral space.

The configure script currently does not recognize new programs in this directory. If a program is added the Makefile has to be adapted by hand. Modify the file `3dvar/obj-Makefile` *for that purpose.*

By default, all programs (executables) are linked. If only the 3dVar or a subset of the utilities is required the Makefile (`3dvar/build/`*Platform/*`obj/Makefile`*) may be modified accordingly (e.g. add a line prog = ../bin/var3d).*

19.2.1 Test of the covariance model - test_fg_cov

This program performs tests on the covariance model. Parameters are read from a file `'nml_test_fg'` in the actual directory. This file contains namelist groups `/RUN/` to specify the analysis date (`yyyymmddhh_ana`) and a path (`input`) to the analysis error file (of the previous forecast) as well as a namelist group `/PSCMODEL/` to modify parameters of the covariance model.

Subsequent tests are steered interactively by reading from standard input. One out of 4 tests is chosen by typing one of the characters:

v: Calculate vertical correlations .

h: Calculate horizontal correlations .

H: Calculate horizontal correlations (2-D-fields on pressure levels) .

c: The horizontal correlations are calculated directly as a function of the distance normalized by the length scale L_h .
The values are written to a file suitable for plotting by gnuplot. In detail the columns of the table denote:

1. row index.
2. r: distance normalized by the length scale L_h.
3. c: correlation function for geopotential height.
4. dc/dr: first derivative (geop.-wind correlation, analytically).
5. $-\frac{1}{r}dc/dr$ (analytically).
6. $-d^2c/dr^2$: second derivative (transversal wind-wind correlation, analytically).
7. dc/dr (finite differences).
8. $-\frac{1}{r}dc/dr$ (finite differences).
9. $-d^2c/dr^2$ (finite differences).

19.2.2 Interpolate soil and surface - interpolate_soil_flake

Interpolates surface and soil variables to a different (in general lower) model resolution with special treatment for soil and flake variables.

The program is not parallelised an expects a namelist file 'namelist.interpolate_soil_flake' in the working directory. Example:

```
&interpolate
  file_s     = 'fg_2015051500_r3b07.grb'       ! source GRIB file name
  file_d     = 'fg_2015051500_r2b06.soil'      ! output GRIB file name
  ifile_s    = 'invar_0026v5_R3B7N8_DOM01.grb' ! invariant file (source)
  ifile_d    = 'invar_0024v4_R2B6N7_DOM01.grb' ! invariant file (output)
  gfile_s    = 'icon_grid_0026_R03B07_G.nc'    ! grid-file      (source)
```

```
  gfile_d    = 'icon_grid_0024_R02B06_G.nc'    ! grid-file      (output)
  grid_d     = 'icon'                          ! destination grid (icon, gme, latlon)
  var        = 'T_SO W_SO W_SO_ICE Z0 T_G T_ICE H_ICE FR_ICE QV_S W_I W_SNOW RHO_SNOW
                H_SNOW FRESHSNW T_SNOW T_MNW_LK T_WML_LK H_ML_LK T_BOT_LK C_T_LK'
                                               ! variables to interpolate
  fr_lake    = 0.5 0.5                         ! lower bound for fr_lake (source, destinatio
! nx         =                                 ! resolution parameter for grid_d='latlon'
! ni         =                                 ! resolution parameter for grid_d='gme'
/
```

This example namelist was used to interpolate ICON soil fields from the operational deterministic resolution (13 km) to that of the ensemble data assimilation system (40 km).

The File 'file_s' holds the fields to be interpolated. Results will be stored in file 'file_d'.

Files 'ifile_s' and 'ifile_d' hold invariant fields for source and destination resolution, required by the interpolation routine, i.e. HHL (for ICON / COSMO), FR_LAND, HSURF, FR_LAKE, SOILTYP, and DEPTH_LK. 'ifile_s' is not required if the respective fields are contained in 'file_s'.

'gfile_s' und 'gfile_d' are the grid description files for ICON (not required for other models).

'var' is the list of fields to interpolate (GRIB short-names or 3dvar internally used names). The following groups of variables are handled seperately:

Surface fields `( T_SO W_SO W_SO_ICE )`

For each destination grid-point a point of the surrounding neighbours (3 to 4 depending on the grid) on the source grid with corresponding surface type is sought. If found, the nearest grid-point is used.

If no corresponding grid-point is found the closest land point (excluding rock or ice) is taken. In the case of `W_SO` the soil moisture index (normalised by pore volume and air dryness point) is interpolated. In case of `W_SO_ICE` interpolation is performed, normalising by `W_SO`.

Flake model fields `( T_MNW_LK T_WML_LK H_ML_LK T_BOT_LK C_T_LK )`, **land surface fields** `( FRESHSNW T_SNOW RHO_SNOW W_SNOW H_SNOW Z0 T_G QV_S W_I )`, and **sea surface fields** `( T_ICE H_ICE FR_ICE )`

For each destination grid-point a corresponding point of the surrounding neighbours (3 to 4 depending on the grid) on the source grid is sought. Points falling within the same category of 'sea', 'land' or 'lake' are regarded as corresponding. Points are classified as 'lake' if the invariant field 'FR_LAKE' is larger than rhe respective threshold specified in the namelist (should be 0.05 with and 0.5 without tile approach). In the category 'lake' the point with the most similar depth (DEPTH_LK) is chosen. For other categories the closest point is taken.

If no corresponding grid-point is found the closest point of the same categorie is used.

For the 'flake' fields a consistency check (taking into account DEPTH_LK) is performed.

Remaining (atmospheric) fields

For all remaining fields horizontal bi-linear interpolation is used, keeping the vertical model level.

19.3 Utility Scripts

Chapter 20

File formats

20.1 nlpcg.info (convergence of CG algorithm)

This output file (ASCII) monitors the convergence of the Newton / CG algorithm in observation space:

column	name	description
1	`it`	total iteration number.
2	`inl`	nonlinear iteration number.
3	`il`	linear iteration number.
4	`r.dot.y`	this scalar product should be zero.
5	`J_b`	background cost function.
6	`J_o`	observational cost function.
7	`J`	total cost function (`J_b + J_o`).
8	`J_cg`	cost function minimized by the CG algorithm.
9	`delta_J`	increment of `J` in this step.
10	`delta_J_cg`	increment of `J_cg` in this step.
11	`resid`	norm of the residuum of the CG algorithm.
12	`dJ/dx`	norm of the gradient of `J`.
13		mark `'#ITNL'` to mark starting conditions of inner loop.

20.2 psas.info (feedback file - ASCII)

This file holds a subset of the NetCDF feedback file information on the analysis step in ASCII format. Writing of this file can be enabled by setting the parameter `info` in namelist `/PSAS/`. A similar file `moni.info` holds information on the monitoring step.

The ASCII content can also be derived by the utility program .

```
Usage:
  netcdf2asci <<EOF
  filename.nc
  EOF
```

Each line of the ASCII table gives information on a single observed quantity. The columns denote:

```
entry          example  description

n                    3  consecutive observation count
obs_nr               1  consecutive report count
box                -99  preconditioning box in 3D-Var/PSAS
name             2AAY7  station name
otyp                 1  observation type   (classification of observation)
bft                  1  BUFR type          (classification in BUFR input file)
sbt                  1  BUFR subtype       (classification in BUFR input file)
varno                U  variable number    (specification of observed quantity)
lat              56.50  latitude  [degree] (location of observation)
lon               2.10  longitude [degree] (location of observation)
lev             101960.  level [unit depends on observation type] (height of observatio
fc           5.7766190  forecasted quantity
o-fc        -6.6622247  observation minus forecast
a-fc        -0.1571989  analysis minus observation (preliminary value in 3D-VAR PSAS)
af_f        -0.1283722  analysis minus observation (final value)
bcor         0.0000000  bias correction (currently for radiances and aircrafts)
e_bg         1.3741624  assumed background error
e_o          2.5200000  prescribed observation error
w_qc         0.3988838  weight assigned in the Variational Quality Control
date   20130109000000  observation date YYYYMMDDHHMMSS
obs_id          374877  unique report Id used within the assimilation system
dbkz               256  'Datenbankkennzahl' (classification of the entry in the DWD da
code                21  code type (classification of observation in addition to otyp)
state           ACTIVE  status of the observation in the assimilation (active, passive
check             RULE  reason for above classification (first guess check, blacklist,
mdlsfc              10  model surface characteristic
pcc                  0  additional classification of the observation (type dependent)
```

20.3 cofXXXXX.nc (feedback file - NetCDF)

Different feedback files are written for each of the observation types (cf. Table 17.1, `XXXXX` in the file name is replaced by `SYNOP, TEMP`, ...). The feedback file is writen in NetCDF. Its structure reflects the internal structure of the 3dVar and that of a future ODB feedback file format.

Arrays in the NetCDF file are either of dimension `n_hdr` (entries of table 'hdr' for each report) or `n_body` (entries of table body for eeach observed/assimilated quantity).

header-table entries

name	type	description	units
i_body	int	index to the data of the report. This and the next entry are used to associate 'hdr'-table entries with 'body'-table entries.	
l_body	short	number of data items of the report.	

obstype	short	(CMA) observation type.	Table 17.1
codetype	short	(CMA) observation code type.	Table 17.2
dbkz	short	DWD 'Datenbankkennzahl'.	Table 17.2
bufrtype	short	WMO BUFR type.	Table 17.2
subtype	short	ECMWF BUFR subtype.	Table 17.2
time	short	observation - analysis time.	minutes
instype	int	station type or satellite instrument.	
retrtype	int	station retrieval type.	
lsf	int	station land/sea flag.	
clf	int	station cloud flag.	
pro	int	station processing center.	
lat	float	latitude of observation.	degree
lon	float	longitude of observation.	degree
statid	char(8)	station id in character form.	
r_state	byte	3dVar report state. Values are given in the attribute 'flag_meanings'. Valid values are at least: PASSIVE, REJECTED, ACTIVE, ACCEPTED.	
r_check	byte	Check that caused the above state. Values are given in the attribute 'flag_meanings'. Valid values are: BLACKLIST, FG, ...	
r_flags	int	This is a bit-fields of flags with the same meaning as r_check indicating all triggered checks.	

body-table entries

level	float	observation level (in general pressure).	(hPa)
obt	int	observed quantity type, currently 3dVar specific, conventions of the ECMWF ODB shall be used in the future.	SI units
bg	double	first guess (background).	as bg
o-bg	float	observation - background.	as bg
a-bg	float	analysis - background.	as bg
e_bg	float	background error.	as bg
e_o	float	observational error.	as bg
w_qc	float	VCQ weight.	as bg
af_f	float	final analysis - forecast.	as bg
state	byte	as r_state, but referring to the individual observation.	
check	byte	as r_check, but referring to the individual observation.	
flags	byte	as r_flags, but referring to the individual observation.	

Example (monSYNOP.nc):

```
netcdf cofSYNOP {
dimensions:
n_hdr = 40 ;
n_body = 118 ;
len8 = 8 ;
variables:
```

```
int i_body(n_hdr) ;
i_body:long_name = "index to the data of the report" ;
short l_body(n_hdr) ;
l_body:long_name = "number of data items of the report" ;
short obstype(n_hdr) ;
obstype:long_name = "(CMA) observation type" ;
short codetype(n_hdr) ;
codetype:long_name = "(CMA) observation code type" ;
short bufrtype(n_hdr) ;
bufrtype:long_name = "BUFR type" ;
short subtype(n_hdr) ;
subtype:long_name = "BUFR subtype" ;
short dbkz(n_hdr) ;
dbkz:long_name = "DWD data base id" ;
short time(n_hdr) ;
time:units = "minute" ;
time:long_name = "observation - analysis time" ;
int instype(n_hdr) ;
instype:long_name = "station type or sat. instrument" ;
int retrtype(n_hdr) ;
retrtype:long_name = "station retrieval type" ;
int lsf(n_hdr) ;
lsf:long_name = "station land/sea flag" ;
int clf(n_hdr) ;
clf:long_name = "station cloud flag" ;
int pro(n_hdr) ;
pro:long_name = "station processing center" ;
float lat(n_hdr) ;
lat:units = "degree" ;
lat:long_name = "latitude of observation" ;
float lon(n_hdr) ;
lon:units = "degree" ;
lon:long_name = "longitude of observation" ;
char statid(n_hdr, len8) ;
statid:long_name = "station id in character form" ;
byte r_state(n_hdr) ;
r_state:long_name = "3D-Var report state" ;
r_state:flag_meanings = "ABORT FORGET MERGED DISMISS NOTUSED USED PASSIVE REJECTED ACTIVE_O A
byte r_check(n_hdr) ;
r_check:long_name = "3D-Var report check" ;
r_check:flag_meanings = "NONE NOIMPL OBSTYP SUBTYP CORR CORRERR MERGE DBLSTAT DBLERR DBLCRD I
int r_flags(n_hdr) ;
r_flags:long_name = "3D-Var report flags" ;
float level(n_body) ;
level:long_name = "observation level" ;
int obt(n_body) ;
obt:long_name = "observed quantity type" ;
double bg(n_body) ;
```

```
bg:long_name = "first guess (background)" ;
float o-bg(n_body) ;
o-bg:long_name = "observation - background" ;
float a-bg(n_body) ;
a-bg:long_name = "analysis - background" ;
float e_bg(n_body) ;
e_bg:long_name = "background error" ;
float e_o(n_body) ;
e_o:long_name = "observational error" ;
float w_qc(n_body) ;
w_qc:long_name = "VCQ weight" ;
float af_f(n_body) ;
af_f:long_name = "final analysis - forecast" ;
byte state(n_body) ;
state:long_name = "3D-Var datum state" ;
byte check(n_body) ;
check:long_name = "3D-Var datum check" ;
int flags(n_body) ;
flags:long_name = "3D-Var datum flags" ;

// global attributes:
:experiment = 0 ;
:runtype = 3 ;
:run_time = "2004120114" ;
:analysis_time = "2004100100" ;
:reference_time = "2004093021" ;
:forecast_hours = 3 ;
}
```

Chapter 21

Program output (stdout)

program-output chapte: Need to check how up-to-date it is!

An excerpt of the program output is presented below. It stems from Experiment 5836 (conventional observations and ATOVS), analysis time 2006-07-20 18 UT.

21.1 Namelist processing

First a couple of namelist groups are read. Resulting parameter settings are written to stdout.

21.1.1 Namelist /RUN/

This namelist specifies analysis date, paths and file names, as well as the partitioning in a parallel environment.

```
0:------------------------------------------------------------------------
0:
0:  General Parameters (namelist /RUN/) :
0:
0:  run time (gmt) = 200610130401
0:  host           = p010et01
0:  user           = loadl
0:
0:  nex            =       5836
0:  run_type       =          2
0:  runtype        = forecast
0:  method         = PSAS
0:  interactive    = F
0:  nproc1         =          8
0:  nproc2         =         12
0:
0:  reference time = 2006072015
0:  analysis  time = 2006072018
0:  forecast hours =          3
0:
0:  data   path    = ./
0:  input  path    = ./
0:  output path    = ./
0:  aux    path    = ./
0:
0:  fg_file        = ./i_ifg_grb
0:  oldanerr       = ./i_ga3_err
0:  ana_err_file   = ./o_ga3_err
0:  ana_file       = ./o_ian_grb
0:  output         = ./out_2006072018
0:  pass_fields    = T
0:
0:------------------------------------------------------------------------
```

21.1.2 Namelist /BLACKLIST/

According to the specification in namelist group /BLACKLIST/ a blacklist is read from a file. The number of entries is reported for each observation type.

```
0:------------------------------------------------------------------------
0:
0:  Reading Blacklists
0:
0: reading: ./blklsttmp
0: 1228  lines read.
0:
0:  type entries
0:     1     765
0:     4     134
0:     5     203
0:     6     126
0:
0:------------------------------------------------------------------------
```

21.1.3 Namelist /REPORT/

Report type specific parameters as specified by namelist groups /REPORT/ are printed. In particular report types may be used in the analysis step (ACTIVE), in the monitoring step (PASSIVE) or be dismissed (DISMISS), (FORGET) at all. The anticipated effects of the different quality and consistency checks are reported, as well as spatial bounds and other conditions used for data selection.

```
0:------------------------------------------------------------------------
0:
0: read namelist /REPORT/:
0:
0:  type = TOVS       ,use = active
0:  type = SYNOP      ,use = active
0:  type = DRIBU      ,use = active
0:  type = PAOB       ,use = active
0:  type = SCATT      ,use = passive
0:  type = TEMP       ,use = active
0:  type = PILOT      ,use = active
0:  type = SATOB      ,use = active
0:  type = AIREP      ,use = active
0:
0: final settings:
0: type     : SYNOP    AIREP    SATOB    DRIBU    TEMP     PILOT    SATEM    PAOB     SCATT    GPSRO    unused   TOVS
0: max_proc : ******   ******   ******   ******   ******   ******   ******   ******   ******   ******   ******   ******
0: max_act  : ******   ******   ******   ******   ******   ******   ******   ******   ******   ******   ******   ******
0: ni       :      0        0        0        0        0        0        0        0        0        0        0        0
0: deriv_p  :      0        1        0        0        0        1        0        0        0        0        0        0
0: height_t :      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0
0: height_b :   1100.0   1100.0   1100.0   1100.0   1100.0   1100.0   1100.0   1100.0   1100.0   1100.0   1100.0   1100.0
0: lat_nb   :     90.0     90.0     90.0     90.0     90.0     90.0     90.0    -20.0     90.0     90.0     90.0     90.0
0: lat_sb   :    -90.0    -90.0    -90.0    -90.0    -90.0    -90.0    -90.0    -90.0    -90.0    -90.0    -90.0    -90.0
0: lon_eb   :    180.0    180.0    180.0    180.0    180.0    180.0    180.0    180.0    180.0    180.0    180.0    180.0
0: lon_wb   :   -180.0   -180.0   -180.0   -180.0   -180.0   -180.0   -180.0   -180.0   -180.0   -180.0   -180.0   -180.0
0: time_b   :     1630     1630     1630     1730     1630     1630     1630     1630     1630     1630     1730     1630
0: time_e   :     1930     1930     1930     1830     1930     1930     1930     1930     1930     1930     1830     1930
0: fr_land  :      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0
0: fr_sea   :      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.5      0.0      0.0      0.0      1.0
0: fr_noice :      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0
0: checks   :
0: NONE     : ACTIVE   ACTIVE   ACTIVE   ACTIVE   ACTIVE   ACTIVE   FORGET   ACTIVE   PASSIVE  FORGET   ABORT    ACTIVE
0: NOIMPL   : ABORT    ABORT    ABORT    ABORT    ABORT    ABORT    ABORT    ABORT    ABORT    ABORT    ABORT    ABORT
0: OBSTYP   : DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS
0: SUBTYP   : REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED
0: CORR     : DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS
0: CORRERR  : DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS
0: MERGE    : MERGED   MERGED   MERGED   MERGED   MERGED   MERGED   MERGED   MERGED   MERGED   MERGED   MERGED   MERGED
0: DBLSTAT  : DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS
0: DBLERR   : DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS
0: DBLCRD   : DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS
0: INSDAT   : DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS
0: TIME     : FORGET   FORGET   FORGET   FORGET   FORGET   FORGET   FORGET   FORGET   FORGET   FORGET   FORGET   FORGET
0: AREA     : REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED
0: HEIGHT   : DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS
0: THIN     : REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED
0: RULE     : REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED
0: MAXPR    : DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS
0: MAXACT   : REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED
0: SURF     : REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED
0: DATASET  : REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED
```

```
 0: CLOUD    : DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS
 0: FG       : REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED
 0: OBS_ERR  : DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS  DISMISS
 0: BLACKLIST: REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED
 0: WHITELIST: ACTIVE_1 ACTIVE_1 ACTIVE_1 ACTIVE_1 ACTIVE_1 ACTIVE_1 ACTIVE_1 ACTIVE_1 ACTIVE_1 ACTIVE_1 ACTIVE_1 ACTIVE_1
 0: ARTIFICIA: PASSIVE  PASSIVE  PASSIVE  PASSIVE  PASSIVE  PASSIVE  PASSIVE  PASSIVE  PASSIVE  PASSIVE  PASSIVE  PASSIVE
 0: QI       : REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED REJECTED
 0: NOTUSED  : PASSIVE  PASSIVE  PASSIVE  PASSIVE  PASSIVE  PASSIVE  PASSIVE  PASSIVE  PASSIVE  PASSIVE  PASSIVE  PASSIVE
 0: FINAL    : ACCEPTED ACCEPTED ACCEPTED ACCEPTED ACCEPTED ACCEPTED ACCEPTED ACCEPTED ACCEPTED ACCEPTED ACCEPTED ACCEPTED
 0: DOMAIN   : FORGET   FORGET   FORGET   FORGET   FORGET   FORGET   FORGET   FORGET   FORGET   FORGET   FORGET   FORGET
 0:
 0:--------------------------------------------------------------------------
```

21.1.4 Namelist /RULES/

More specific parameter settings which do not refer to the complete report but only to some of its parts (for instance observation types t,u,v,..) are specified by namelist groups /RULES/. The settings for the example run are given below.

Each 'rule' is reported in the program output. It consists of a selection part which determines weather the rule shall be applied to a specific observation, and a specification part for selecting the parameters. If more than one rule is applicable to an observation the settings of the last rule remain valid.

```
 0:--------------------------------------------------------------------------
 0:
 0: Rules for observation processing
 0:
 0:  1 Filter: forbid everything
 0:
 0:    3D-Var-type          = *****
 0:    BUFR-message-type    = *****
 0:    BUFR-message-subtype = *****
 0:    Databank-Kennzahl    =
 0:
 0:    Optional Filter:
 0:    Latitude  bounds     =  ********** **********
 0:    Longitude bounds     =  ********** **********
 0:    Exclude   area       =  F
 0:    Pressure  bounds     =  ********** **********
 0:    Pressure  below surf.=  **********
 0:    Pressure  above surf.=  **********
```

The selection part of the first rule above does not hold any valid criterium (indicated by ****). Thus it applies to all reports which are not covered by any other of the subsequent rules.

The specification part only holds the flag 'use=4', indicating that the respective observations are not used at all:

```
 0:
 0:    Flags:
 0:    Use flag             =     4
 0:    Verbosity flag       =     9
 0:
 0:    Parameter specific:
 0:             use m_rej      sgm_o   sgm_fg_p  sgm_fg_i0   sgm_fg_a
sgm_fg_i1     sgm_vq frm_vq
 0:         t     4 ***** ********** ********** ********** **********
********** ********** ******
 0:        gp     4 ***** ********** ********** ********** **********
********** ********** ******
 0:        uv     4 ***** ********** ********** ********** **********
********** ********** ******
 0:         q     4 ***** ********** ********** ********** **********
********** ********** ******
 0:         p     4 ***** ********** ********** ********** **********
********** ********** ******
```

Subsequent rules specify valid data base identifiers (Datenbankkennzahlen). The corresponding 3dvar module type is given. The use flag is set to 13 indicating that the observations shall be used for assimilation.

```
   0:
   0:  2 Filter: define "Kennzahlen" for TEMP
   0:
   0:    3D-Var-type          =     2
   0:    BUFR-message-type    = *****
   0:    BUFR-message-subtype = *****
   0:    Databank-Kennzahl    =   512  513  514  515  520  521  522  523  776
777
   0:                             778  779
   0:
   0:    Optional Filter:
   0:    Latitude  bounds     =  ********** **********
   0:    Longitude bounds     =  ********** **********
   0:    Exclude   area       =  F
   0:    Pressure  bounds     =  ********** **********
   0:    Pressure  below surf.=  **********
   0:    Pressure  above surf.=  **********
   0:
   0:    Flags:
   0:    Use flag             =    13
   0:    Verbosity flag       = *****
   0:
   0:    Parameter specific:
   0:            use m_rej      sgm_o   sgm_fg_p  sgm_fg_i0   sgm_fg_a
sgm_fg_i1     sgm_vq frm_vq
   0:
   0:  3 Filter: define "Kennzahlen" for SYNOP
   0:
   0:    3D-Var-type          =     1
   0:    BUFR-message-type    = *****
   0:    BUFR-message-subtype = *****
   0:    Databank-Kennzahl    =     0  128  256  384  385  386 1697
   0:
   . . .
   0:
   0:  4 Filter: define "Kennzahlen" for AMV
   0:
   0:    3D-Var-type          =    16
   0:    BUFR-message-type    = *****
   0:    BUFR-message-subtype = *****
   0:    Databank-Kennzahl    =  1672 1673 1674 1675 1677 1704 1705 1706 1707
   0:
   . . .
   0:
   0:  5 Filter: define "Kennzahlen" for AIREP
   0:
   0:    3D-Var-type          =    64
   0:    BUFR-message-type    = *****
   0:    BUFR-message-subtype = *****
   0:    Databank-Kennzahl    =   529  530  533  534
   0:
   . . .
   0:
   0:  6 Filter: define "Kennzahlen" for TOVS
   0:
   0:    3D-Var-type          =     4
   0:    BUFR-message-type    = *****
   0:    BUFR-message-subtype = *****
   0:    Databank-Kennzahl    =    -1
   0:
   . . .
   0:
   0:  7 Filter: define "Kennzahlen" for GPSRO
   0:
   0:    3D-Var-type          =     8
   0:    BUFR-message-type    = *****
   0:    BUFR-message-subtype = *****
   0:    Databank-Kennzahl    =    -1
   0:
   . . .
```

Providing a data base identifier (Datenbankkennzahl) equal to -1 causes any number to be valid for the respective module type. This is the case for ATOVS data (preprocessed by the 1dvar) and GPS radio occultations. These observations are currently not stored in the DWD data base.

Subsequent rules set parameters specific to certain observation type (t, gp, q, p for temperature, geopotential height, humidity, pressure): usage and quality control flags. Further restrictions may be applied (spatial bounds, pressure bounds) in the specification section.

```
   0:  8 Filter: specific parameters for TEMP gp,rh
   0:
   0:    3D-Var-type          =     2
   0:    BUFR-message-type    = *****
   0:    BUFR-message-subtype = *****
```

```
   0:    Databank-Kennzahl    =
   0:
   0:    Optional Filter:
   0:    Latitude  bounds     =  ********** **********
   0:    Longitude bounds     =  ********** **********
   0:    Exclude   area       =  F
   0:    Pressure  bounds     =  ********** **********
   0:    Pressure  below surf.=    10000.00
   0:    Pressure  above surf.=  **********
   0:
   0:    Flags:
   0:    Use flag             = *****
   0:    Verbosity flag       =     0
   0:
   0:    Parameter specific:
   0:             use m_rej      sgm_o   sgm_fg_p  sgm_fg_i0   sgm_fg_a
sgm_fg_i1     sgm_vq frm_vq
   0:         t    13     3      1.000      5.000 **********      0.000
**********      2.000 ******
   0:        gp    13     3      1.000      5.000 **********      0.000
**********      2.000 ******
   0:         q    13     3      1.000      5.000 **********      0.000
**********      2.000 ******
   0:         p     4 ***** ********** ********** ********** **********
********** ********** ******
   0:
   0:  9 Filter: specific parameters for TEMP wind
   0:
   0:    3D-Var-type          =     2
   . . .
   0:    Pressure  below surf.=     2500.00
   . . .
   0:    Parameter specific:
   0:             use m_rej      sgm_o   sgm_fg_p  sgm_fg_i0   sgm_fg_a
sgm_fg_i1     sgm_vq frm_vq
   0:        uv    13     3      1.000      5.000 **********      0.000
**********      2.000 ******
   0:         p     4 ***** ********** ********** ********** **********
********** ********** ******
   0:
   0: 10 Filter: specific parameters for AMV
   0:
   0:    3D-Var-type          =    16
   . . .
   0:    Parameter specific:
   0:             use m_rej      sgm_o   sgm_fg_p  sgm_fg_i0   sgm_fg_a
sgm_fg_i1     sgm_vq frm_vq
   0:        uv    13     3      1.000      5.000 **********      0.000
**********      2.000 ******
   0:
   0: 11 Filter: specific parameters for AIREP
   0:
   0:    3D-Var-type          =    64
   . . .
   0:    Parameter specific:
   0:             use m_rej      sgm_o   sgm_fg_p  sgm_fg_i0   sgm_fg_a
sgm_fg_i1     sgm_vq frm_vq
   0:         t    13     3      1.000      5.000 **********      0.000
**********      2.000 ******
   0:        uv    13     3      1.000      5.000 **********      0.000
**********      2.000 ******
   0:
   0: 12 Filter: specific parameters for GPSRO
   0:
   0:    3D-Var-type          =     8
   . . .
   0:    Parameter specific:
   0:             use m_rej      sgm_o   sgm_fg_p  sgm_fg_i0   sgm_fg_a
sgm_fg_i1     sgm_vq frm_vq
   0:         o    13     3      1.000      5.000 **********      0.000
**********      2.000 ******
   0:
   0: 13 Filter: added by namelist /RULES/
   0:
   0:    3D-Var-type          = *****
   0:    BUFR-message-type    = *****
   0:    BUFR-message-subtype = *****
   0:    Databank-Kennzahl    =
   0:
   . . .
   0:
   0:    Parameter specific:
   0:             use m_rej      sgm_o   sgm_fg_p  sgm_fg_i0   sgm_fg_a
sgm_fg_i1     sgm_vq frm_vq
   0:         t ***** ***** **********      3.000 ********** **********
**********      2.000 ******
   0:        gp ***** ***** **********      3.000 ********** **********
**********      2.000 ******
   0:        uv ***** ***** **********      3.000 ********** **********
**********      2.500 ******
   0:         q ***** ***** **********      3.000 ********** **********
**********      1.200 ******
```

```
   0:
   0: 14 Filter: specific parameters for SYNOP Land
   0:
   0:    3D-Var-type          =     1
   0:    BUFR-message-type    =     0
   . . .
   0:    Pressure  below surf.=     5000.00
   0:    Pressure  above surf.=     5000.00
   . . .
   0:    Parameter specific:
   0:             use m_rej      sgm_o   sgm_fg_p  sgm_fg_i0   sgm_fg_a
sgm_fg_i1     sgm_vq frm_vq
   0:         t     4 ***** ********** ********** ********** **********
********** ********** ******
   0:        gp    11 ***** ********** ********** ********** **********
********** **********      1
   0:        uv    11 ***** ********** ********** ********** **********
**********      2.000 ******
   0:         q    11 ***** ********** ********** ********** **********
********** ********** ******
   0:         p     4 ***** ********** ********** ********** **********
********** ********** ******
   0:
   0: 15 Filter: specific parameters for SYNOP Sea
   0:
   0:    3D-Var-type          =     1
   0:    BUFR-message-type    =     1
   . . .
   0:    Parameter specific:
   0:             use m_rej      sgm_o   sgm_fg_p  sgm_fg_i0   sgm_fg_a
sgm_fg_i1     sgm_vq frm_vq
   0:         t     4 ***** ********** ********** ********** **********
********** ********** ******
   0:        gp    11 ***** ********** ********** ********** **********
********** **********      1
   0:        uv    11 ***** ********** ********** ********** **********
**********      2.000 ******
   0:         q    11 ***** ********** ********** ********** **********
********** ********** ******
   0:         p     4 ***** ********** ********** ********** **********
********** ********** ******
   0:
   0: 16 Filter: Quality control for AMV
   0:
   0:    3D-Var-type          =    16
   . . .
   0:    Parameter specific:
   0:             use m_rej      sgm_o   sgm_fg_p  sgm_fg_i0   sgm_fg_a
sgm_fg_i1     sgm_vq frm_vq
   0:        uv ***** ***** ********** ********** ********** **********
**********      2.000 ******
   0:
   0:--------------------------------------------------------------------------
```

21.1.5 Namelist /THINNING/

Thinning criteria are specified by namelist /THINNING/, namely horizontal distance parameter (ni), vertical distance (dlev), weights and bounds for different criteria as well as priorities of the criteria.

Thinning rules are related to the reports by matching obstypes, codetypes and data base identifiers.

```
   0:--------------------------------------------------------------------------
   0:
   0:  thinning rules:
   0:
   0:  TOVS
   0:
   0:    obstype  = TOVS
   0:    codetype =    -1    0    0    0    0    0    0    0    0    0
   0:    dbkz     =    -1    0    0    0    0    0    0    0    0    0
   0:    ni       =    48  147.km
   0:    nlev     =     1
   0:    dlev(hPa)=     0
   0:
   0:    criterium   weight     bound
   0:    center        1.00      0.00
   0:    time          1.00     -3.00
   0:    sequence      1.00      0.00
   0:    data          1.00      0.00
   0:    quality       1.00     -1.00
   0:    vertical      1.00      0.00
   0:    status        1.00      0.00
```

```
0:
0:    rule  1 :  status
0:    rule  2 :  time
0:    rule  3 :  quality
0:    rule  4 :  center   vertical
0:    rule  5 :  data
0:    rule  6 :  sequence
0:
0:  AMV: Eumetsat, GOES
0:
0:    obstype  = SATOB
0:    codetype =    -1    0    0    0    0    0    0    0    0    0
0:    dbkz     =  1704 1705    0    0    0    0    0    0    0    0
0:    ni       =    32  220.km
0:    nlev     =    11
0:    dlev(hPa)=   100
0:
0:    criterium   weight     bound
0:    center        1.00      0.00
0:    time          1.00     -3.00
0:    sequence      1.00      0.00
0:    data          1.00      0.00
0:    quality       1.00      0.50
0:    vertical      1.00      0.00
0:    status        1.00      0.00
. . .
0:
0:  AMV: Modis
0:
0:    obstype  = SATOB
0:    codetype =    -1    0    0    0    0    0    0    0    0    0
0:    dbkz     =  1706    0    0    0    0    0    0    0    0    0
0:    ni       =   128   55.km
0:    nlev     =    11
0:    dlev(hPa)=   100
0:
0:    criterium   weight     bound
0:    center        1.00      0.00
0:    time          1.00     -3.00
0:    sequence      1.00      0.00
0:    data          1.00      0.00
0:    quality       1.00     -1.00
0:    vertical      1.00      0.00
0:    status        1.00      0.00
. . .
0:
0:  AIREPs
0:
0:    obstype  = AIREP
0:    codetype =    -1    0    0    0    0    0    0    0    0    0
0:    dbkz     =    -1    0    0    0    0    0    0    0    0    0
0:    ni       =   192   37.km
0:    nlev     =    11
0:    dlev(hPa)=   100
0:
. . .
0:
0:  PAOBs
0:
0:    obstype  = PAOB
0:    codetype =    -1    0    0    0    0    0    0    0    0    0
0:    dbkz     =    -1    0    0    0    0    0    0    0    0    0
0:    ni       =    48  147.km
0:    nlev     =     1
0:    dlev(hPa)=     0
0:
. . .
0:
0:  QuickSCATT
0:
0:    obstype  = SCATT
0:    codetype =    -1    0    0    0    0    0    0    0    0    0
0:    dbkz     =    -1    0    0    0    0    0    0    0    0    0
0:    ni       =    64  110.km
0:    nlev     =     1
0:    dlev(hPa)=     0
0:
. . .
```

21.1.6 Namelists /SYNOP_OBS/ /TOVS_OBS/ /TOVS_OBS_CHAN/ /GPS_RO/ /AMV_OBS/ /AIREP_OBS/

A couple of report type specific namelists are read without further printout.

21.2 Parallel input of observations

21.2.1 Scan of obervation input files

First the meta data of all observation files is scanned. Conventional data is read in BUFR format while ATOVS and GPSRO data is read in NetCDF format. The content of the BUFR files is printed:

```
0:------------------------------------------------------------------------
0:
0: BUFR file : ./mld_file.1
0:
0: type,subt:records  comment
0:    2,   0:   305 ( VERT_SOUND )
0:
0: DB-Kennz.:records  comment
0:       512:    75 - In-Situ Beob.,  PILOT, PART A
0:       513:    90 - In-Situ Beob.,  PILOT, PART B
0:       514:    11 - In-Situ Beob.,  PILOT, PART C
0:       515:    11 - In-Situ Beob.,  PILOT, PART D
0:       520:    26 - In-Situ Beob.,  TEMP,  PART A
0:       521:    28 - In-Situ Beob.,  TEMP,  PART B
0:       522:    24 - In-Situ Beob.,  TEMP,  PART C
0:       523:    24 - In-Situ Beob.,  TEMP,  PART D
0:       776:     6 - In-Situ Beob.,  TEMP,  SHIP, PART A
0:       777:     6 - In-Situ Beob.,  TEMP,  SHIP, PART B
0:       778:     2 - In-Situ Beob.,  TEMP,  SHIP, PART C
0:       779:     2 - In-Situ Beob.,  TEMP,  SHIP, PART D
0:
0:
0: BUFR file : ./mld_file.2
0:
0: type,subt:records  comment
0:    3,   0:  6014 ( VERT_S_SAT )
0:
0: DB-Kennz.:records  comment
0:      1664:  3007 - Satellitenbeob, SATEM, PART A
0:      1666:  3007 - Satellitenbeob, SATEM, PART C
0:
0:
0: BUFR file : ./mld_file.3
0:
0: type,subt:records  comment
0:    2,   0:  7923 ( VERT_SOUND )
0:    4,   4:  2120 ( SINGL_LEV )
0:    4, 255:  1103 ( SINGL_LEV )
0:
0: DB-Kennz.:records  comment
0:       529:  3981 - In-Situ Beob.,  AMDAR
0:       530:  3942 - In-Situ Beob.,  AIREP
0:       533:  2120 - In-Situ Beob.,  ACARS-Daten USA
0:       534:  1103 - In-Situ Beob.,  ACARS-Daten EUROPA (Bracknell)
0:
0:
0: BUFR file : ./mld_file.4
0:
0: type,subt:records  comment
0:    5,   6:    78 ( SINGL_SAT )
0:    5,  10:   455 ( SINGL_SAT )
0:    5,  87:   182 ( SINGL_SAT )
0:
0: DB-Kennz.:records  comment
0:      1704:   573 - Satellitenbeob, AMV,   Eumetsat
0:      1705:   142 - Satellitenbeob, AMV,   GOES (NOAA/NESDIS)
0:
0:
0: BUFR file : ./mld_file.5
0:
0: type,subt:records  comment
0:    5,   6:    73 ( SINGL_SAT )
0:    5,  10:   410 ( SINGL_SAT )
0:    5,  87:   250 ( SINGL_SAT )
0:
0: DB-Kennz.:records  comment
0:      1704:   550 - Satellitenbeob, AMV,   Eumetsat
0:      1705:   183 - Satellitenbeob, AMV,   GOES (NOAA/NESDIS)
0:
0:
0: BUFR file : ./mld_file.6
0:
0: type,subt:records  comment
0:    5,   6:    49 ( SINGL_SAT )
0:    5,  10:   481 ( SINGL_SAT )
0:    5,  87:   292 ( SINGL_SAT )
0:
0: DB-Kennz.:records  comment
```

```
0:      1704:   575 - Satellitenbeob, AMV,   Eumetsat
0:      1705:   247 - Satellitenbeob, AMV,   GOES (NOAA/NESDIS)
0:
0:
0: BUFR file : ./mld_file.7
0:
0: type,subt:records  comment
0:    5,   6:    30 ( SINGL_SAT )
0:    5,  10:   246 ( SINGL_SAT )
0:    5,  87:   220 ( SINGL_SAT )
0:
0: DB-Kennz.:records  comment
0:      1704:   326 - Satellitenbeob, AMV,   Eumetsat
0:      1705:   170 - Satellitenbeob, AMV,   GOES (NOAA/NESDIS)
0:
0:
0: BUFR file : ./mld_file.8
0:
0: type,subt:records  comment
0:    5,   6:    44 ( SINGL_SAT )
0:    5,  10:   316 ( SINGL_SAT )
0:    5,  87:   260 ( SINGL_SAT )
0:
0: DB-Kennz.:records  comment
0:      1704:   425 - Satellitenbeob, AMV,   Eumetsat
0:      1705:   195 - Satellitenbeob, AMV,   GOES (NOAA/NESDIS)
0:
0:
0: BUFR file : ./mld_file.9
0:
0: type,subt:records  comment
0:    5,   6:    54 ( SINGL_SAT )
0:    5,  10:   523 ( SINGL_SAT )
0:    5,  87:   122 ( SINGL_SAT )
0:
0: DB-Kennz.:records  comment
0:      1704:   650 - Satellitenbeob, AMV,   Eumetsat
0:      1705:    49 - Satellitenbeob, AMV,   GOES (NOAA/NESDIS)
0:
0:
0: BUFR file : ./mld_file.10
0:
0: type,subt:records  comment
0:    5,   6:    44 ( SINGL_SAT )
0:    5,  10:   431 ( SINGL_SAT )
0:    5,  87:   104 ( SINGL_SAT )
0:
0: DB-Kennz.:records  comment
0:      1704:   548 - Satellitenbeob, AMV,   Eumetsat
0:      1705:    31 - Satellitenbeob, AMV,   GOES (NOAA/NESDIS)
0:
0:
0: BUFR file : ./mld_file.11
0:
0: type,subt:records  comment
0:    5,   6:   120 ( SINGL_SAT )
0:    5,  10:  1044 ( SINGL_SAT )
0:    5,  87:   251 ( SINGL_SAT )
0:
0: DB-Kennz.:records  comment
0:      1704:  1320 - Satellitenbeob, AMV,   Eumetsat
0:      1705:    95 - Satellitenbeob, AMV,   GOES (NOAA/NESDIS)
0:
0:
0: BUFR file : ./mld_file.12
0:
0: type,subt:records  comment
0:    0,   0:  4598 ( SURF_LAND )
0:
0: DB-Kennz.:records  comment
0:         0:  1243 - Bodenmeldungen, SYNOP, Sect. 1-4, manuell+PAST
0:         5:  1297 - Bodenmeldungen, SYNOP, Section 5
0:         9:     7 - Bodenmeldungen, PSEUDO-SYNOP (aus TEMP)
0:       128:  2051 - Bodenmeldungen, SYNOP, Sect. 1-3, autom.
0:
0:
0: BUFR file : ./mld_file.13
0:
0: type,subt:records  comment
0:    0,   0:  5808 ( SURF_LAND )
0:
0: DB-Kennz.:records  comment
0:         0:  1581 - Bodenmeldungen, SYNOP, Sect. 1-4, manuell+PAST
0:         5:  1846 - Bodenmeldungen, SYNOP, Section 5
0:         9:     6 - Bodenmeldungen, PSEUDO-SYNOP (aus TEMP)
0:       128:  2375 - Bodenmeldungen, SYNOP, Sect. 1-3, autom.
0:
0:
0: BUFR file : ./mld_file.14
0:
0: type,subt:records  comment
0:    0,   0:  5107 ( SURF_LAND )
```

```
 0:
 0: DB-Kennz.:records  comment
 0:         0:  3054 - Bodenmeldungen, SYNOP, Sect. 1-4, manuell+PAST
 0:         5:   773 - Bodenmeldungen, SYNOP, Section 5
 0:         9:    15 - Bodenmeldungen, PSEUDO-SYNOP (aus TEMP)
 0:       128:  1265 - Bodenmeldungen, SYNOP, Sect. 1-3, autom.
 0:
 0:
 0: BUFR file : ./mld_file.15
 0:
 0: type,subt:records  comment
 0:    0,   0: 29427 ( SURF_LAND )
 0:    1,   0:  6246 ( SURF_SEA )
 0:
 0: DB-Kennz.:records  comment
 0:         1: 29424 - Bodenmeldungen, METAR
 0:         4:     3 - Bodenmeldungen, METAR, WEHI
 0:       256:   881 - Bodenmeldungen, SHIP, manuell
 0:       265:     6 - Bodenmeldungen, PSEUDO-SHIP (aus TEMP-SHIP)
 0:       384:  1149 - Bodenmeldungen, SHIP, automatisch
 0:       385:  4210 - Bodenmeldungen, BUOY
 0:
 0:
 0: BUFR file : ./mld_file.16
 0:
 0: type,subt:records  comment
 0:    1,   0:  8730 ( SURF_SEA )
 0:    5,   0:   280 ( SINGL_SAT )
 0:
 0: DB-Kennz.:records  comment
 0:      1697:  8730 - Satellitenbeob, QUICKSCAT (als DRIBU)
 0:      1706:   280 - Satellitenbeob, AMV,   MODIS
 0:
 0:--------------------------------------------------------------------------------
```

21.2.2 Assignment of observations to processors

The total number of records and subsets and their distribution over input files is counted:

```
 0:--------------------------------------------------------------------------------
 0:
 0:  obstype   records  subsets in file no.
  ...987654321
 0:  SYNOP       46939    46939
00000000000000000000000000000000000000000000000001111000000000000
 0:  AIREP       11146    31364
00000000000000000000000000000000000000000000000000000000000000100
 0:  SATOB        6359   192590
00000000000000000000000000000000000000000000000010000111111111000
 0:  DRIBU        4210     4210
00000000000000000000000000000000000000000000000001000000000000000
 0:  TEMP          118      118
00000000000000000000000000000000000000000000000000000000000000001
 0:  PILOT         187      187
00000000000000000000000000000000000000000000000000000000000000001
 0:  SATEM        6014     6014
00000000000000000000000000000000000000000000000000000000000000010
 0:  PAOB            0        0
00000000000000000000000000000000000000000000000000000000000000000
 0:  SCATT        8730     8730
00000000000000000000000000000000000000000000000001000000000000000
 0:  GPSRO           0        0
00000000000000000000000000000000000000000000000000000000000000000
 0:  unused          0        0
00000000000000000000000000000000000000000000000000000000000000000
 0:  TOVS       154258   154258
00000000000000000000000000000000000000000000000111000000000000000
 0:
 0:--------------------------------------------------------------------------------
```

The file names associated with the file numbers are printed:

```
 0:--------------------------------------------------------------------------------
 0:  Observation input files:
 0:
 0:    1 mld_file.1
 0:    2 mld_file.2
 0:    3 mld_file.3
 0:    4 mld_file.4
 0:    5 mld_file.5
 0:    6 mld_file.6
 0:    7 mld_file.7
 0:    8 mld_file.8
 0:    9 mld_file.9
 0:   10 mld_file.10
 0:   11 mld_file.11
```

```
   0:   12 mld_file.12
   0:   13 mld_file.13
   0:   14 mld_file.14
   0:   15 mld_file.15
   0:   16 mld_file.16
   0:   17 sat_206_2006072018-dt.nc
   0:   18 sat_207_2006072018-dt.nc
   0:   19 sat_784_2006072018-dt.nc
   0:--------------------------------------------------------------------------
```

The work of reading and decoding the observations is shared by the processor elements (here 96). Based on the number of records and subsets present in the files, the range of subsets to be read by individual processors is determined:

```
   0:--------------------------------------------------------------------------
   0:
   0:  Distribution of reports for input
   0:
   0:     pe            SYNOP   AIREP   SATOB   DRIBU   TEMP    PILOT   SATEM
PAOB    SCATT   GPSRO   unused  TOVS
   0:
   0:      0  start :        0       0       0       0       0       0       0
    0       0       0       0       0
   0:      0  end   :        0     325    2006      44     118       0      63
    0      87       0       0    1557
   0:      0  used  :        F       T       T       T       T       F       T
    F       T       F       F       T
   0:
   0:      1  start :        0     325    2006      44       0       0      63
    0      87       0       0    1557
   0:      1  end   :        0     640    4012      88       0     187     126
    0     174       0       0    3129
   0:      1  used  :        F       T       T       T       F       T       T
    F       T       F       F       T
   0:
   0:      2  start :        0     640    4012      88       0       0     126
    0     174       0       0    3129
   0:      2  end   :    46939     640    4012      88       0       0     126
    0     174       0       0    3129
   0:      2  used  :        T       F       F       F       F       F       F
    F       F       F       F       F
   0:
   . . .
   0:
   0:     94  start :        0   29578  186558    4092       0       0    5859
    0    8182       0       0  151277
   0:     94  end   :        0   29896  188564    4136       0       0    5922
    0    8270       0       0  152905
   0:     94  used  :        F       T       T       T       F       F       T
    F       T       F       F       T
   0:
   0:     95  start :        0   29896  188564    4136       0       0    5922
    0    8270       0       0  152905
   0:     95  end   :        0   31364  192590    4210       0       0    6014
    0    8730       0       0  154258
   0:     95  used  :        F       T       T       T       F       F       T
    F       T       F       F       T
   0:
   0:--------------------------------------------------------------------------
```

TEMP, PILOT and SYNOP observations are read from one PE only, because merging of reports and processing of correction reports - which is handled by the BUFR input routine - requires the respective obsevations to be present on the same processor.

For each processor a list of the subsets to be read from each file is generated (shown exemplarily for PE 0):

```
   0:--------------------------------------------------------------------------
   0:
   0:   file             SYNOP   AIREP   SATOB   DRIBU   TEMP    PILOT   SATEM
PAOB    SCATT   GPSRO   unused  TOVS
   0:
   0:      1 subsets:        0       0       0       0     118     187       0
    0       0       0       0       0
   0:
   0:      2 offset :        0       0       0       0     118     187       0
    0       0       0       0       0
   0:      2 subsets:        0       0       0       0       0       0    6014
    0       0       0       0       0
   0:
   0:      3 offset :        0       0       0       0     118     187    6014
    0       0       0       0       0
```

```
 0:      3 subsets:        0   31364       0       0       0       0       0
  0       0       0       0       0
 0:
 0:      4 offset :        0   31364       0       0     118     187    6014
  0       0       0       0       0
 0:      4 subsets:        0       0   25246       0       0       0       0
  0       0       0       0       0
 0:
 0:      5 offset :        0   31364   25246       0     118     187    6014
  0       0       0       0       0
 0:      5 subsets:        0       0   25355       0       0       0       0
  0       0       0       0       0
 0:
 0:      6 offset :        0   31364   50601       0     118     187    6014
  0       0       0       0       0
 0:      6 subsets:        0       0   26370       0       0       0       0
  0       0       0       0       0
. . .
 0:     15 offset :    15485   31364  183642       0     118     187    6014
  0       0       0       0       0
 0:     15 subsets:    31454       0       0    4210       0       0       0
  0       0       0       0       0
 0:
 0:     16 offset :    46939   31364  183642    4210     118     187    6014
  0       0       0       0       0
 0:     16 subsets:        0       0    8948       0       0       0       0
  0    8730       0       0       0
 0:
 0:     17 offset :    46939   31364  192590    4210     118     187    6014
  0    8730       0       0       0
 0:     17 subsets:   -46939  -31364 -192590   -4210    -118    -187   -6014
  0   -8730       0       0   63753
 0:
 0:     18 offset :        0       0       0       0       0       0       0
  0       0       0       0   63753
 0:     18 subsets:        0       0       0       0       0       0       0
  0       0       0       0   55453
 0:
 0:     19 offset :        0       0       0       0       0       0       0
  0       0       0       0  119206
 0:     19 subsets:        0       0       0       0       0       0       0
  0       0       0       0   35052
 0:
 0:-------------------------------------------------------------------------
```

The list of files to be opened by each processor is:

```
 0:-------------------------------------------------------------------------
 0:
 0:                         1         2         3         4
 0:               12345678901234567890123456789012345678901234567890
 0:      0      :
TTTTTTTTTTTFFFTTTTTFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
 0:      1      :
TTTTTTTTTTTFFFTTTTTFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
 0:      2      :
FFFFFFFFFFFTTTTFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
 0:      3      :
FTTTTTTTTTTFFFTTTTTFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
 0:      4      :
FTTTTTTTTTTFFFTTTTTFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
 . . .
 0:     94      :
FTTTTTTTTTTFFFTTTTTFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
 0:     95      :
FTTTTTTTTTTFFFTTTTTFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
 0:
 0:-------------------------------------------------------------------------
```

Finally the observations are read by the respective processors. GPSRO data (not used in this example) currently is read only by processor 0.

21.3 Input of atmospheric background fields

The GRIB input file with the background fields (preceeding forecast) is scanned. Its inventory is printed twice, with parameter names given in GME convention and `var3d` convention, respectively.

```
   0:--------------------------------------------------------------------------
   0:
   0:inventory of grib file: ./i_ifg_grb
   0:
   0: rec     adress         name       date     time tab cnt sub prc cde lvt
levelvl runtype   ni grd nlev
   0:   1|         0|PS          |2006-07-20|18:00:00|  2| 78|255|174|  1|  1|
0|  0|forecast|192|192|  1
   0:   2|    745426|T_G         |2006-07-20|18:00:00|  2| 78|255|174| 11|  1|
0|  0|forecast|192|192|  1
   0:   3|   1490852|T_2M        |2006-07-20|18:00:00|  2| 78|255|174| 11|105|
2|  0|forecast|192|192|  1
   0:   4|   2236278|TD_2M       |2006-07-20|18:00:00|  2| 78|255|174| 17|105|
2|  0|forecast|192|192|  1
   0:   5|   2981704|U_10M       |2006-07-20|18:00:00|  2| 78|255|174| 33|105|
10|  0|forecast|192|192|  1
   0:   6|   3727130|V_10M       |2006-07-20|18:00:00|  2| 78|255|174| 34|105|
10|  0|forecast|192|192|  1
   0:   7|   4472556|U           |2006-07-20|18:00:00|  2| 78|255|174| 33|110|
1|  2|forecast|192|192| 40
   0:  47|  34289596|V           |2006-07-20|18:00:00|  2| 78|255|174| 34|110|
1|  2|forecast|192|192| 40
   0:  87|  64106636|T           |2006-07-20|18:00:00|  2| 78|255|174| 11|110|
1|  2|forecast|192|192| 40
   0: 127|  93923676|QV          |2006-07-20|18:00:00|  2| 78|255|174| 51|110|
1|  2|forecast|192|192| 40
   0: 167| 123740716|QC          |2006-07-20|18:00:00|201| 78|255|174| 31|110|
1|  2|forecast|192|192| 40
   0: 207| 153557756|QI          |2006-07-20|18:00:00|201| 78|255|174| 33|110|
1|  2|forecast|192|192| 40
   0: 247| 183374796|FIS         |0001-01-01|00:00:00|  2| 78|255|173|  6|  1|
0|  0|analysis|192|192|  1
   0: 248| 184119894|FR_LAND     |0001-01-01|00:00:00|  2| 78|255|173| 81|  1|
0|  0|analysis|192|192|  1
   0:
   0: rec     adress         name       date     time tab cnt sub prc cde lvt
levelvl runtype   ni grd nlev
   0:   1|         0|ps          |2006-07-20|18:00:00|  2| 78|255|174|  1|  1|
0|  0|forecast|192|192|  1
   0:   2|    745426|tsurf       |2006-07-20|18:00:00|  2| 78|255|174| 11|  1|
0|  0|forecast|192|192|  1
   0:   3|   1490852|t2m         |2006-07-20|18:00:00|  2| 78|255|174| 11|105|
2|  0|forecast|192|192|  1
   0:   4|   2236278|td2m        |2006-07-20|18:00:00|  2| 78|255|174| 17|105|
2|  0|forecast|192|192|  1
   0:   5|   2981704|            |2006-07-20|18:00:00|  2| 78|255|174| 33|105|
10|  0|forecast|192|192|  1
   0:   6|   3727130|            |2006-07-20|18:00:00|  2| 78|255|174| 34|105|
10|  0|forecast|192|192|  1
   0:   7|   4472556|u           |2006-07-20|18:00:00|  2| 78|255|174| 33|110|
1|  2|forecast|192|192| 40
   0:  47|  34289596|v           |2006-07-20|18:00:00|  2| 78|255|174| 34|110|
1|  2|forecast|192|192| 40
   0:  87|  64106636|t           |2006-07-20|18:00:00|  2| 78|255|174| 11|110|
1|  2|forecast|192|192| 40
   0: 127|  93923676|q           |2006-07-20|18:00:00|  2| 78|255|174| 51|110|
1|  2|forecast|192|192| 40
   0: 167| 123740716|qcl         |2006-07-20|18:00:00|201| 78|255|174| 31|110|
1|  2|forecast|192|192| 40
   0: 207| 153557756|qci         |2006-07-20|18:00:00|201| 78|255|174| 33|110|
1|  2|forecast|192|192| 40
   0: 247| 183374796|geosp       |0001-01-01|00:00:00|  2| 78|255|173|  6|  1|
0|  0|analysis|192|192|  1
   0: 248| 184119894|slm         |0001-01-01|00:00:00|  2| 78|255|173| 81|  1|
0|  0|analysis|192|192|  1
   0:
   0:--------------------------------------------------------------------------
```

The files are read and stored into variables of derived data type t_grid for the grid information and t_atm for the atmospheric fields. The content of the variables is printed, showing minimum and maximum values of each parameter and model level:

```
   0:--------------------------------------------------------------------------
   0:
   0: read_atm_state: reading ps
   0: read_atm_state: reading psr
   0: read_atm_state: reading tsurf
   0: read_atm_state: reading t
   0: read_atm_state: reading u
   0: read_atm_state: reading v
   0: read_atm_state: reading q
   0: read_atm_state: reading qcl
   0: read_atm_state: reading qci
   0: read_atm_state: reading t2m
   0: read_atm_state: reading td2m
   0:
   0: (t_atm)
   0: time% (t_time)
```

```
0: time% days          : 2453937
0: time% secs          :         64800
0: time% yyyymmddhhmmss:20060720180000
0: ref_time% (t_time)
0: ref_time% days          : 2453937
0: ref_time% secs          :         54000
0: ref_time% yyyymmddhhmmss:20060720150000
0: grid% (t_grid) :
0: grid% grid  :        192
0: grid% levtyp:        110
0: grid% nx    :        193
0: grid% ny    :        193
0: grid% ngl   :          0
0: grid% nn    :          0
0: grid% nz    :         40
0: grid% rot   : F
0: grid% cyc_x : F
0: grid% poly  : F
0: grid% global: T
0: grid% la1   :          0.00
0: grid% lo1   :          0.00
0: grid% di    :          0.00
0: grid% dj    :          0.00
0: grid% a     :    6371229.00
0: grid% g     :          9.81
0: grid% size  :   14899600
0: grid% dlon  : (1),(n)=  0.00      0.00
0: grid% dlat  : (1),(n)=  0.00      0.00
0: grid% lon(1): (1),(n)=  0.00    -0.628
0: grid% lat(1): (1),(n)=  1.57     0.459
0: grid% rlon  : min,max= -3.14      3.14     |  -180.      180.     [degree]
0: grid% rlat  : min,max= -1.57      1.57     |  -90.0      90.0     [degree]
0: grid% slm   : min,max=  0.00      1.00
0: grid% geosp : min,max= -963.     0.595E+05 |  -98.2     0.607E+04 [m]
0: grid% geo_sh: not associated!
0: grid% geoid : min,max= -107.      85.2
0: grid% k= 1 ak=      0.0000 bk= 0.0000 akf=   1000.0000 bkf= 0.0000
0: grid% k= 2 ak=   2000.0000 bk= 0.0000 akf=   3000.0000 bkf= 0.0000
0: grid% k= 3 ak=   4000.0000 bk= 0.0000 akf=   5000.0000 bkf= 0.0000
0: grid% k= 4 ak=   6000.0000 bk= 0.0000 akf=   7000.0000 bkf= 0.0000
0: grid% k= 5 ak=   8000.0000 bk= 0.0000 akf=   8988.0684 bkf= 0.0002
 . . .
0: grid% k=35 ak=    855.3618 bk= 0.9518 akf=    661.3477 bkf= 0.9597
0: grid% k=36 ak=    467.3335 bk= 0.9676 akf=    338.8637 bkf= 0.9737
0: grid% k=37 ak=    210.3939 bk= 0.9797 akf=    138.1415 bkf= 0.9840
0: grid% k=38 ak=     65.8892 bk= 0.9883 akf=     36.6284 bkf= 0.9911
0: grid% k=39 ak=      7.3677 bk= 0.9940 akf=      3.6838 bkf= 0.9958
0: grid% k=40 ak=      0.0000 bk= 0.9976 akf=      0.0000 bkf= 0.9988
0: grid% k=41 ak=      0.0000 bk= 1.0000
0: lb  :     0     1     1     1
0: ub  :    23    16    40    10
0: size:  940800
0: ps              gg(: min,max=) 0.488E+05 0.104E+06   1   2
0: psr             gg(: min,max=) 0.488E+05 0.104E+06   1   2
0: t               gg(: min,max=)  176.      318.      11   2
0: t                 k= 1      176.      243.
0: t                 k= 2      182.      234.
0: t                 k= 3      183.      231.
0: t                 k= 4      184.      232.
0: t                 k= 5      187.      232.
. . .
0: t                 k=35      221.      316.
0: t                 k=36      217.      317.
0: t                 k=37      214.      317.
0: t                 k=38      210.      317.
0: t                 k=39      208.      318.
0: t                 k=40      207.      318.
0: u               gg(: min,max=) -46.9      96.5      33   2
0: u                 k= 1     -31.0      90.9
0: u                 k= 2     -29.3      79.3
0: u                 k= 3     -29.9      69.6
0: u                 k= 4     -34.0      59.5
0: u                 k= 5     -40.7      55.7
. . .
0: u                 k=35     -33.4      29.3
0: u                 k=36     -30.4      26.2
0: u                 k=37     -27.1      24.0
0: u                 k=38     -25.7      23.1
0: u                 k=39     -24.0      21.4
0: u                 k=40     -21.1      18.8
0: v               gg(: min,max=) -68.3      67.9      34   2
0: v                 k= 1     -18.3      23.1
0: v                 k= 2     -23.3      25.8
0: v                 k= 3     -26.3      25.4
0: v                 k= 4     -22.3      26.8
0: v                 k= 5     -23.2      29.7
. . .
0: v                 k=35     -32.0      40.8
0: v                 k=36     -30.0      36.3
0: v                 k=37     -27.8      27.5
0: v                 k=38     -26.5      26.0
0: v                 k=39     -24.7      24.1
```

```
0: v               k=40     -21.5      21.0
0: q             gg(: min,max=) 0.100E-11 0.259E-01  51   2
0: q               k= 1     0.126E-05 0.402E-05
0: q               k= 2     0.131E-05 0.413E-05
0: q               k= 3     0.918E-06 0.406E-05
0: q               k= 4     0.883E-06 0.422E-05
0: q               k= 5     0.100E-11 0.527E-05
. . .
0: q               k=35     0.100E-11 0.235E-01
0: q               k=36     0.100E-11 0.241E-01
0: q               k=37     0.100E-11 0.245E-01
0: q               k=38     0.100E-11 0.252E-01
0: q               k=39     0.100E-11 0.255E-01
0: q               k=40     0.100E-11 0.259E-01
0: qcl           gg(: min,max=)  0.00     0.161E-02  31 201
0: qcl             k= 1      0.00      0.00
0: qcl             k= 2      0.00      0.00
0: qcl             k= 3      0.00      0.00
0: qcl             k= 4      0.00      0.00
0: qcl             k= 5      0.00      0.00
. . .
0: qcl             k=35      0.00     0.404E-03
0: qcl             k=36      0.00     0.409E-03
0: qcl             k=37      0.00     0.348E-03
0: qcl             k=38      0.00     0.337E-03
0: qcl             k=39      0.00     0.327E-03
0: qcl             k=40      0.00     0.399E-03
0: qci           gg(: min,max=)  0.00     0.814E-03  33 201
0: qci             k= 1      0.00      0.00
0: qci             k= 2      0.00      0.00
0: qci             k= 3      0.00      0.00
0: qci             k= 4      0.00     0.139E-10
0: qci             k= 5      0.00     0.170E-08
. . .
0: qci             k=35      0.00     0.187E-04
0: qci             k=36      0.00     0.173E-04
0: qci             k=37      0.00     0.196E-04
0: qci             k=38      0.00     0.241E-04
0: qci             k=39      0.00     0.193E-04
0: qci             k=40      0.00     0.171E-04
0: tke           gg(: not ALLOCATED!)
0: cl            gg(: not ALLOCATED!)
0: ph            gg(: not ALLOCATED!)
0: pf            gg(: not ALLOCATED!)
0: rh            gg(: not ALLOCATED!)
0: t2m           gg(: min,max=)  200.      319.      11   2
0: rh2m          gg(: not ALLOCATED!)
0: geoh          gg(: not ALLOCATED!)
0: geof          gg(: not ALLOCATED!)
0: tsurf         gg(: min,max=)  192.      321.      11   2
0: td2m          gg(: min,max=)  198.      302.      17   2
0: tv            gg(: not ALLOCATED!)
0: psi           gg(: not ALLOCATED!)
0: chi           gg(: not ALLOCATED!)
0: size =  940800
0:
0:--------------------------------------------------------------------------
```

21.4 Further namelist processing

21.4.1 Namelist /ANAERROR/

Parameters for the analysis error estimation. Currently no Output is written to stdout for namelist /ANAERROR/.

21.4.2 Namelist /PSAS/

Parameters for the solver of the analysis equation. Currently no Output is written to stdout for namelist /PSAS/.

21.4.3 Namelist /VARQC/

Parameters for Variational quality control:

```
0:------------------------------------------------------------------------
0:  Variational Quality Control Parameters
0:
0: w_min     = 0.10E-01 : minimum singular value in Hessian
0: iappr     =        2 : approximation level for R
0: iprint    =        0 : print flag
0: vqc_form  =        4 : formulation of obs-costfunction
0:
```

The value of 'vqc_form=4' denotes a Huber norm like specification of the observational cost function with continuous derivatives. The other parameters have no effect for this choice.

Subsequent parameters are applicable to correlated observations only and currently not used.

```
0:
0: vqc_par( 1) :
0:
0: dim_lim =       10
0: n_full  =       10
0: c_start =
0: c_final =
0: n_lev_s =        0
0: n_lev_f =        0
0:
0: vqc_par( 2) :
. . .
0: vqc_par( 6) :
0:
0:------------------------------------------------------------------------
```

21.4.4 Namelist /HUM_ANA/

Parameters regarding the humidity analysis. Currently no Output is written to stdout for namelist /HUM_ANA/.

21.4.5 Namelist /CNTRLVAR/

Parameters regarding the specification of control variables. No Output is written to stdout for namelist /CNTRLVAR/.

21.4.6 Namelist /TEST_OBS_OPR/

Parameters for testing the tangent linear (Jacobi matrix) of the observation operators by comparing the analytical gradients with finite differences. The finite differences increment is a normal distributed random number scaled by the background error times a small scaling factor. The test was switched off for this experiment (test_H=0).

```
0:------------------------------------------------------------------------
0: Namelist /TEST_OBS_OPR/ read
0:   operator =
0:   quantity =
0:   scale    =  0.1E-02
0:   argument =      0
0:   result   =      0
0:   mode     = on
0:   seed     =   0  0  0  0  0  0  0  0  0  0  0
0:   test_H   =      0
0:------------------------------------------------------------------------
```

21.4.7 Namelist /BG_ERROR_OPERATOR/

Specification of the operator representation of background error covariance matrices. The parameters are written twice: First after processing of the namelist and a second time

after the coefficients were read from an external file. Parameters may be modified due to the specification in the file. In this example the grid and domain sizes (ny: number of meridional points, pbot: pressure at lowest level) changed:

```
 0:------------------------------------------------------------------------
 0:
 0:  reading namelist /BG_ERROR_OPERATOR/
 0:
 0:    nx        =   512
 0:    ny        =     0
 0:    nz        =    64
 0:    pbot      =  1000.
 0:    file      = /uwork0/arhodin/nmc_gme/200507/vertcov
 0:    transform = w
 0:    sqr       = s
 0:    valid     =     4
 0:    lclim     =     2
 0:    base      =    12
 0:    nwv       =     4
 0:    h2psi     = T
 0:    h2t       = T
 0:    sqrtcor   = T
 0:    sparse    = F
 0:
 0:
 0:  cov_from_nmc
 0:
 0:  modified parameters:
 0:
 0:
 0:    nx        =   512
 0:    ny        =   256
 0:    nz        =    64
 0:    pbot      =  1070.
 0:    file      = /uwork0/arhodin/nmc_gme/200507/vertcov
 0:    transform = w
 0:    sqr       = s
 0:    valid     =     4
 0:    lclim     =     2
 0:    base      =    12
 0:    nwv       =     4
 0:    h2psi     = T
 0:    h2t       = T
 0:    sqrtcor   = T
 0:    sparse    = F
```

Some tables with vertical covariance coefficients are printed: Square root of the diagonals of the covariance matrix at ca. 60 degree South.

```
 0: set_bg_err_op: sqrt(diag) of covariance matrices at -61.0588235294117609
 0:
 0:    pressure level        height streamfunction     veloc.pot.
temperature       rel.hum.
 0:  1         10.000         97.435         97.435    1907642.124
2.186         0.014
 0:  2         10.770         93.979         93.979    1778234.747
2.119         0.014
 . . .
 0: 55        548.873         39.772         39.772     622161.377
1.803         0.255
 0: 56        591.132         38.362         38.362     656415.275
1.846         0.265
 0: 57        636.644         37.147         37.147     683479.561
1.869         0.273
 0: 58        685.661         36.143         36.143     682565.230
1.894         0.277
 0: 59        738.451         35.323         35.323     663584.024
1.948         0.282
 0: 60        795.306         34.840         34.840     673482.433
2.034         0.271
 0: 61        856.538         34.765         34.765     666619.355
2.285         0.228
 0: 62        922.484         35.272         35.272     697799.088
2.367         0.165
 0: 63        993.508         36.546         36.546     761425.169
2.037         0.080
 0: 64       1070.000         38.485         38.485     718855.179
2.007         0.070
```

Diagonals of correlation matrices at ca. 60 degree South (should be all 1):

```
 0: set_bg_err_op: diagonals of correlation matrices at -61.0588235294117609
 0:     pressure level        height streamfunction     veloc.pot.
temperature       rel.hum.
```

```
   0:   1         10.000          1.000          1.000          1.000
1.000          1.000
   0:   2         10.770          1.000          1.000          1.000
1.000          1.000
   . . .
   0:  63        993.508          1.000          1.000          1.000
1.000          1.000
   0:  64       1070.000          1.000          1.000          1.000
1.000          1.000
```

Errors at ca. 60 degree South:

```
   0: set_bg_err_op: Errors
   0:
   0:     pressure level              h             rh              t
 xi              v
   0:   1         10.000         97.435          0.014          2.186
1907642.124           5.115
   0:   2         10.770         93.979          0.014          2.119
1778234.747           4.884
   . . .
   0:  55        548.873         39.772          0.255          1.803
622161.377           5.783
   0:  56        591.132         38.362          0.265          1.846
656415.275           5.543
   0:  57        636.644         37.147          0.273          1.869
683479.561           5.354
   0:  58        685.661         36.143          0.277          1.894
682565.230           5.214
   0:  59        738.451         35.323          0.282          1.948
663584.024           5.121
   0:  60        795.306         34.840          0.271          2.034
673482.433           5.099
   0:  61        856.538         34.765          0.228          2.285
666619.355           5.204
   0:  62        922.484         35.272          0.165          2.367
697799.088           5.375
   0:  63        993.508         36.546          0.080          2.037
761425.169           3.855
   0:  64       1070.000         38.485          0.070          2.007
718855.179           3.652
```

Errors at level 56, (approx. 600 hPa):

```
   0: latitudinal dependence at k= 56 591.131701268327447
   0:
   0: set_bg_err_op: Errors:
   0:           latitude              h             rh              t
 xi              v
   0:   1        -90.000         31.102          0.136          1.859
530648.604           4.111
   0:   2        -89.294         31.224          0.136          1.919
537585.310           4.281
   . . .
   0:  37        -64.588         39.591          0.252          1.881
649915.157           5.537
   0:  38        -63.882         39.619          0.256          1.905
650639.298           5.561
   0:  39        -63.176         39.472          0.265          1.915
651764.587           5.570
   . . .
   0: 131          1.765          4.464          0.137          0.652
954960.503           2.151
   0: 132          2.471          4.449          0.142          0.647
954165.232           2.141
   0: 133          3.176          4.453          0.144          0.638
954268.046           2.125
   . . .
   0: 234         74.471         17.801          0.226          1.166
511708.141           2.854
   0: 235         75.176         17.864          0.229          1.163
511240.381           2.848
   0: 236         75.882         17.832          0.230          1.164
509893.169           2.797
   . . .
   0: 255         89.294         13.359          0.205          0.889
505970.011           2.372
   0: 256         90.000         12.877          0.185          0.891
510536.324           2.382
   0:
   0:--------------------------------------------------------------------------
```

21.5 Monitoring step

Monitoring is performed with all data selected so far: Innovations are derived, the first guess check and thinning of data is performed.

21.5.1 Assignment of observations to processors

In order to achieve a better load balance the observations are redistributed over the processors. In this example 396 161 reports were read. Assignment to boxes is performed so that the estimated amount of computing load (here 1 188 483) is distributed evenly over the PEs.

```
 0:--------------------------------------------------------------------------
 0:
 0: set_boxes : estimated total cost   =  1188483
 0: set_boxes : number of spots used   =  396161
 0: set_boxes : number of boxes        =  192
 0:
 0:     box     pe   observations reports
 0:                  proc.   goal   left
 0:      1      0    6192    6190 394097
 0:      2      0   12381   12380 392034
 0:      3      1   18570   18570 389971
 0:      4      1   24762   24760 387907
 . . .
 0:    188     93 1163724 1163723   8253
 0:    189     94 1169913 1169913   6190
 0:    190     94 1176105 1176103   4126
 0:    191     95 1182294 1182293   2063
 0:    192     95 1188483 1188483      0
 0:
 0:--------------------------------------------------------------------------
```

21.5.2 Input of previous analysis error

The estimated analysis error for wind components and geopotential height from the previous assimilation cycle is read from a GRIB file. The fields are stored on a coarse grid with 6 degree spacing on 7 pressure levels. The printout is similar to that of the atmospheric background fields:

```
 0:--------------------------------------------------------------------------
 0:
 0:  reading analysis error:
 0:
 0:  content of grib file ./i_ga3_err
 0:
 0: rec     adress         name       date     time tab cnt sub prc cde lvt
levelvl runtype   ni grd nlev
 0:   1|         0|VAR_U      |2006-07-20|15:00:00|202| 78|255|195| 41|100|
  10|analysis| 60|  0|  7
 0:   2|      3746|VAR_U      |2006-07-20|15:00:00|202| 78|255|195| 41|100|
  50|analysis| 60|  0|  7
 0:   3|      7492|VAR_U      |2006-07-20|15:00:00|202| 78|255|195| 41|100|
 100|analysis| 60|  0|  7
 0:   4|     11238|VAR_U      |2006-07-20|15:00:00|202| 78|255|195| 41|100|
 200|analysis| 60|  0|  7
 0:   5|     14984|VAR_U      |2006-07-20|15:00:00|202| 78|255|195| 41|100|
 300|analysis| 60|  0|  7
 0:   6|     18730|VAR_U      |2006-07-20|15:00:00|202| 78|255|195| 41|100|
 500|analysis| 60|  0|  7
 0:   7|     22476|VAR_U      |2006-07-20|15:00:00|202| 78|255|195| 41|100|
1000|analysis| 60|  0|  7
 0:   8|     26222|VAR_V      |2006-07-20|15:00:00|202| 78|255|195| 42|100|
  10|analysis| 60|  0|  7
 0:   9|     29968|VAR_V      |2006-07-20|15:00:00|202| 78|255|195| 42|100|
  50|analysis| 60|  0|  7
 0:  10|     33714|VAR_V      |2006-07-20|15:00:00|202| 78|255|195| 42|100|
 100|analysis| 60|  0|  7
 0:  11|     37460|VAR_V      |2006-07-20|15:00:00|202| 78|255|195| 42|100|
 200|analysis| 60|  0|  7
 0:  12|     41206|VAR_V      |2006-07-20|15:00:00|202| 78|255|195| 42|100|
 300|analysis| 60|  0|  7
 0:  13|     44952|VAR_V      |2006-07-20|15:00:00|202| 78|255|195| 42|100|
```

```
 500|analysis| 60|  0|  7
   0:  14|     48698|VAR_V       |2006-07-20|15:00:00|202| 78|255|195| 42|100|
1000|analysis| 60|  0|  7
   0:  15|     52444|VAR_FI      |2006-07-20|15:00:00|202| 78|255|195| 40|100|
  10|analysis| 60|  0|  7
   0:  16|     56190|VAR_FI      |2006-07-20|15:00:00|202| 78|255|195| 40|100|
  50|analysis| 60|  0|  7
   0:  17|     59936|VAR_FI      |2006-07-20|15:00:00|202| 78|255|195| 40|100|
 100|analysis| 60|  0|  7
   0:  18|     63682|VAR_FI      |2006-07-20|15:00:00|202| 78|255|195| 40|100|
 200|analysis| 60|  0|  7
   0:  19|     67428|VAR_FI      |2006-07-20|15:00:00|202| 78|255|195| 40|100|
 300|analysis| 60|  0|  7
   0:  20|     71174|VAR_FI      |2006-07-20|15:00:00|202| 78|255|195| 40|100|
 500|analysis| 60|  0|  7
   0:  21|     74920|VAR_FI      |2006-07-20|15:00:00|202| 78|255|195| 40|100|
1000|analysis| 60|  0|  7
   0:
   0:  reading grid information:
   0:
   0:
   0:read_atm_grid: p_readgrib =   0
   0:
   0:
   0:  reading analysis errors:
   0:
   0: (t_atm)
   0: time% (t_time)
   0: time% days          : 2453937
   0: time% secs          :         54000
   0: time% yyyymmddhhmmss:20060720150000
   0: ref_time% (t_time)
   0: ref_time% days          :       0
   0: ref_time% secs          :             0
   0: grid% (t_grid) :
   0: grid% grid  :          0
   0: grid% levtyp:        100
   0: grid% nx    :         60
   0: grid% ny    :         30
   0: grid% ngl   :          0
   0: grid% nn    :          0
   0: grid% nz    :          7
   0: grid% rot   : F
   0: grid% cyc_x : F
   0: grid% poly  : F
   0: grid% global: F
   0: grid% la1   :        -87.00
   0: grid% lo1   :          0.00
   0: grid% di    :          6.00
   0: grid% dj    :          6.00
   0: grid% a     :    6371229.00
   0: grid% g     :          9.81
   0: grid% size  :      12600
   0: grid% dlon  : (1),(n)=  0.00      354.
   0: grid% dlat  : (1),(n)= -87.0      87.0
   0: grid% lon(1): (1),(n)=  0.00      6.18
   0: grid% lat(1): (1),(n)= -1.52      1.52
   0: grid% rlon  : min,max=  0.00      6.18     |   0.00      354.     [degree]
   0: grid% rlat  : min,max= -1.52      1.52     |  -87.0      87.0     [degree]
   0: grid% slm   : not associated!
   0: grid% geosp : not associated!
   0: grid% geo_sh: not associated!
   0: grid% geoid : not associated!
   0: grid% k= 1 ak=      0.0000 bk= 0.0000 akf=   1000.0000 bkf= 0.0000
   0: grid% k= 2 ak=      0.0000 bk= 0.0000 akf=   5000.0000 bkf= 0.0000
   0: grid% k= 3 ak=      0.0000 bk= 0.0000 akf=  10000.0000 bkf= 0.0000
   0: grid% k= 4 ak=      0.0000 bk= 0.0000 akf=  20000.0000 bkf= 0.0000
   0: grid% k= 5 ak=      0.0000 bk= 0.0000 akf=  30000.0000 bkf= 0.0000
   0: grid% k= 6 ak=      0.0000 bk= 0.0000 akf=  50000.0000 bkf= 0.0000
   0: grid% k= 7 ak=      0.0000 bk= 0.0000 akf= 100000.0000 bkf= 0.0000
   0: grid% k= 8 ak=      0.0000 bk= 0.0000
   0: lb  :     1     1     1     1
   0: ub  :    60    30     7     1
   0: size:   37800
   0: ps              gg(: not ALLOCATED!)
   0: psr             gg(: not ALLOCATED!)
   0: t               gg(: not ALLOCATED!)
   0: u               gg(: min,max=) 0.329      7.10      41 202
   0: u                 k= 1     0.439      4.60
   0: u                 k= 2     0.329      2.92
   0: u                 k= 3     0.362      3.61
   0: u                 k= 4     0.600      5.90
   0: u                 k= 5     0.573      7.10
   0: u                 k= 6     0.425      5.79
   0: u                 k= 7     0.399      3.48
   0: v               gg(: min,max=) 0.329      7.07      42 202
   0: v                 k= 1     0.441      4.46
   0: v                 k= 2     0.329      2.91
   0: v                 k= 3     0.361      3.61
   0: v                 k= 4     0.583      5.88
   0: v                 k= 5     0.587      7.07
   0: v                 k= 6     0.423      5.69
```

```
   0: v                k= 7    0.397     3.43
   0: q              gg(: not ALLOCATED!)
   0: qcl            gg(: not ALLOCATED!)
   0: qci            gg(: not ALLOCATED!)
   0: tke            gg(: not ALLOCATED!)
   0: cl             gg(: not ALLOCATED!)
   0: ph             gg(: not ALLOCATED!)
   0: pf             gg(: not ALLOCATED!)
   0: rh             gg(: not ALLOCATED!)
   0: t2m            gg(: not ALLOCATED!)
   0: rh2m           gg(: not ALLOCATED!)
   0: geoh           gg(: not ALLOCATED!)
   0: geof           gg(: min,max=) 0.810     85.1      40 202
   0: geof             k= 1      4.69      85.1
   0: geof             k= 2      2.65      39.3
   0: geof             k= 3      2.57      31.7
   0: geof             k= 4      3.26      38.1
   0: geof             k= 5      2.85      45.9
   0: geof             k= 6      2.36      39.2
   0: geof             k= 7     0.810      32.6
   0: tsurf          gg(: not ALLOCATED!)
   0: td2m           gg(: not ALLOCATED!)
   0: tv             gg(: not ALLOCATED!)
   0: psi            gg(: not ALLOCATED!)
   0: chi            gg(: not ALLOCATED!)
   0: size =   37800
   0:-------------------------------------------------------------------------
```

21.5.3 Derivation of background error

The background error is determined for the observed quantities. Only the diagonal terms of $\mathbf{HBH}^T$ are required for the first guess check to give an estimate of the expected size of the innovations. For technical reasons observations are stored in 'boxes' (4000 to 10000 per box). The progress of the background error calculation is reported and the sparsity pattern of the matrix (here diagonal) is printed. Alltogether 1 707 654 individual observations are processed in the monitoring step of this example. A similar printout will appear later for the partitioning of $\mathbf{HBH}^T$ in the analysis step.

```
   0:-------------------------------------------------------------
   0: set_Pb: block 1 / 192 processed.
   0: set_Pb: block 2 / 192 processed.
   0: set_Pb: block 3 / 192 processed.
   . . .
   0: set_Pb: block 190 / 192 processed.
   0: set_Pb: block 191 / 192 processed.
   0: set_Pb: block 192 / 192 processed.
   0:
   0:  pe   block     m     n   nonzero       total  %         min    max
distance
   0:
   0:   0   1   1  4132  4132      4132    17073424   0.02       0.000
0.000
   0:   0   2   2  4143  4143      4143    17164449   0.02       0.000
0.000
   0:   1   3   3  4160  4160      4160    17413929   0.02       0.000
0.000
   . . .
   0:  94 190 190  4134  4134      4134    17106496   0.02       0.000
0.000
   0:  95 191 191  4127  4127      4127    17032129   0.02       0.000
0.000
   0:  95 192 192  4126  4126      4126    17023876   0.02       0.000
0.000
   . . .
   0:  94       total              8321  -200610268 100.00
   0:  95       total              8253  -200610268 100.00
   0: all       total           1707654  -200610268 100.00
   0:
   0:
   0:
-------------------------------------------------------------------------------
   0:
   0:  Matrix Block encoding:
   0:
   0:      : all elements are zero
   0:    + : full n x m representation
   0:    x : only lower triangle is stored
   0:    - : compressed sparse row storage
   0:    | : compressed sparse column storage
   0:    . : not stored due to symmetry
   0:
```

```
   0: PE
   0:  0 \
   0:  0  \
   0:  1   \
   0:  1    \
   0:  2     \
   0:  2      \
   0:  3       \
   0:  3        \
   0:  4         \
   0:  4          \
   0:  5           \
   0:  5            \
   0:  6             \
   0:  6              \
   0:  7               \
   0:  7                \
   0:  8                 \
   0:  8                  \
   0:  9                   \
   0:  9                    \
   0: 10                     \
   0: 10                      \
   . . .
   0:
-------------------------------------------------------------------------------
   0:
   0:  Matrix Block usage
   0:
   0:    0 :          empty
   0:    1 :  <= 10 % filled
   0:    ...
   0:    9 :  <= 90 % filled
   0:    * :  >  90 % filled
   0:
   0: PE
   0:  0
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000
   0:  0
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000
   0:  1
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000
   0:  1
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000
   0:  2
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000
   0:  2
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000
   0:  3
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000
   0:  3
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000
   0:  4
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000
   0:  4
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000
   0:  5
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000
   0:  5
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000
   . . .
   0:
-------------------------------------------------------------------------------
```

21.5.4 Blacklisting

Blacklisting is applied. Blacklisted observations are reported.

```
0:--------------------------------------------------------------------------------
-
   0:
   0:  process BLACKLIST
   0:
   0:blacklisted   SYNOP    63333    63333
   0:  height           0  110000
   0:blacklisted   PILOT    41715    41715
   0:  wind             0  110000
   0:blacklisted   PILOT    42339    42339
   0:  wind             0  110000
   0:blacklisted   SYNOP    40835    40835
   0:  height           0  110000
   . . .
   0:  height           0  110000
   0:blacklisted   SYNOP    VQGQ4    VQGQ4
   0:  height           0  110000
   0:blacklisted   SYNOP    VRZL3    VRZL3
   0:  height           0  110000
   0:blacklisted   SYNOP    76055    76055
   0:  height           0  110000
   0:blacklisted   SYNOP    9VVN     9VVN
   0:  height           0  110000

0:--------------------------------------------------------------------------------
-
```

21.5.5 Thinning

Thinning is applied. The criteria for thinning are reported as well as the number of processed, accepted and rejected reports.

```
0:--------------------------------------------------------------------------------
-
   0:
   0:  thinning:
   0:
   0:  TOVS
   0:
   0:    obstype  = TOVS
   0:    codetype =    -1    0    0    0    0    0    0    0    0    0
   0:    dbkz     =    -1    0    0    0    0    0    0    0    0    0
   0:    ni       =    48  147.km
   0:    nlev     =     1
   0:    dlev(hPa)=     0
   0:
   0:    criterium   weight     bound
   0:    center        1.00      0.00
   0:    time          1.00     -3.00
   0:    sequence      1.00      0.00
   0:    data          1.00      0.00
   0:    quality       1.00     -1.00
   0:    vertical      1.00      0.00
   0:    status        1.00      0.00
   0:
   0:    rule  1 :  status
   0:    rule  2 :  time
   0:    rule  3 :  quality
   0:    rule  4 :  center   vertical
   0:    rule  5 :  data
   0:    rule  6 :  sequence
   0:
   0:  processed =   80944
   0:  rejected  =   74674
   0:  accepted  =    6270
   0:
   0:  AMV: Eumetsat, GOES
   0:
   0:    obstype  = SATOB
   0:    codetype =    -1    0    0    0    0    0    0    0    0    0
   0:    dbkz     =  1704 1705    0    0    0    0    0    0    0    0
   0:    ni       =    32  220.km
   0:    nlev     =    11
   0:    dlev(hPa)=   100
   . . .
   0:  processed =   83663
   0:  rejected  =   76334
   0:  accepted  =    7329
   0:
   0:  AMV: Modis
   0:
```

```
 0:    obstype  = SATOB
 0:    codetype =    -1    0    0    0    0    0    0    0    0    0
 0:    dbkz     =  1706    0    0    0    0    0    0    0    0    0
 0:    ni       =   128   55.km
 0:    nlev     =    11
 0:    dlev(hPa)=   100
 . . .
 0:  processed =    5633
 0:  rejected  =    2914
 0:  accepted  =    2719
 0:
 0:  AIREPs
 0:
 0:    obstype  = AIREP
 0:    codetype =    -1    0    0    0    0    0    0    0    0    0
 0:    dbkz     =    -1    0    0    0    0    0    0    0    0    0
 0:    ni       =   192   37.km
 0:    nlev     =    11
 0:    dlev(hPa)=   100
 . . .
 0:  processed =   29862
 0:  rejected  =   19880
 0:  accepted  =    9982
 0:
 0:  PAOBs
 0:
 0:    obstype  = PAOB
 0:    codetype =    -1    0    0    0    0    0    0    0    0    0
 0:    dbkz     =    -1    0    0    0    0    0    0    0    0    0
 0:    ni       =    48  147.km
 0:    nlev     =     1
 0:    dlev(hPa)=     0
 . . .
 0:  processed =       0
 0:  rejected  =       0
 0:  accepted  =       0
 0:
 0:  QuickSCATT
 0:
 0:    obstype  = SCATT
 0:    codetype =    -1    0    0    0    0    0    0    0    0    0
 0:    dbkz     =    -1    0    0    0    0    0    0    0    0    0
 0:    ni       =    64  110.km
 0:    nlev     =     1
 0:    dlev(hPa)=     0
 . . .
 0:  processed =       0
 0:  rejected  =       0
 0:  accepted  =       0

0:--------------------------------------------------------------------------------
-
```

21.5.6 Assignment of observations to preconditioning 'boxes'

At the end of the monitoring step the observations (here 120 975 from 35 224 reports) which will be used is the assimilation are redistributed, so that approximately 500 observations are assigned to a 'box' (here 288 on 95 processors) distributed.

```
 0:--------------------------------------------------------------------------------
 0: set_boxes : number of deg. freedom =  120975
 0: set_boxes : number of spots used   =  35224
 0: set_boxes : number of boxes        =  288
 0: set_boxes : size of osas matrix    =  14634950625
 0:     box     pe  observations reports
 0:                 proc.   goal   left
 0:      1      0    421    420  35032
 0:      2      0    841    840  34863
 0:      3      0   1262   1260  34667
 0:      4      1   1681   1680  34512
 0:      5      1   2101   2100  34321
 . . .
 0:    284     94 119297 119294    732
 0:    285     94 119718 119714    617
 0:    286     95 120134 120134    410
 0:    287     95 120554 120554    209
 0:    288     95 120975 120975      0
 0:--------------------------------------------------------------------------------
```

21.5.7 Observation usage report

The number of reports which were processed and finally dismissed during the monitoring step is reported. A similar final printout will be given at the end of the program.

```
  0:--------------------------------------------------------------------------
  0:
  0:    Observation Report Statistics after first guess scan
  0:
  0:   obstype proces.  merged   split   dism. passive reject.  active accept.
 lost description
  0:  dbkz code
  0:
-------------------------------------------------------------------------------
-------------------------------------------------
  0:  SYNOP      46939       0       0   38515       0      40    8384       0
    0 SYNOP
  0:     0  11    5878       0       0    1649       0      18    4211       0
    0 Bodenmeldungen, SYNOP, Sect. 1-4, manuell+PAST
  0:     1 140   29424       0       0   29424       0       0       0       0
    0 Bodenmeldungen, METAR
  0:     5  11    3916       0       0    3916       0       0       0       0
    0 Bodenmeldungen, SYNOP, Section 5
  0:   128  14    5691       0       0    2552       0      21    3118       0
    0 Bodenmeldungen, SYNOP, Sect. 1-3, autom.
  0:   256  21     881       0       0     427       0       1     453       0
    0 Bodenmeldungen, SHIP, manuell
  0:   384  24    1149       0       0     547       0       0     602       0
    0 Bodenmeldungen, SHIP, automatisch
  0:
-------------------------------------------------------------------------------
-------------------------------------------------
  0:  AIREP      30120       0       0      30       0   20108    9982       0
    0 AIREP
  0:   529 144    3981       0       0       0       0    2113    1868       0
    0 In-Situ Beob.,  AMDAR
  0:   530  -1    3942       0       0       0       0    2692    1250       0
    0 In-Situ Beob.,  AIREP
  0:   533 145   14059       0       0       0       0    9297    4762       0
    0 In-Situ Beob.,  ACARS-Daten USA
  0:   534 145    8138       0       0      30       0    6006    2102       0
    0 In-Situ Beob.,  ACARS-Daten EUROPA (Bracknell)
  0:
-------------------------------------------------------------------------------
-------------------------------------------------
  0:  SATOB     192231       0       0       0   30131  152052   10048       0
    0 SATOB (AMV)
  0:  1704  90  135013       0       0       0   26144  104673    4196       0
    0 Satellitenbeob, AMV,   Eumetsat
  0:  1705  90   48270       0       0       0    3987   41150    3133       0
    0 Satellitenbeob, AMV,   GOES (NOAA/NESDIS)
  0:  1706  90    8948       0       0       0       0    6229    2719       0
    0 Satellitenbeob, AMV,   MODIS
  0:
-------------------------------------------------------------------------------
-------------------------------------------------
  0:  DRIBU       1413       0       0     962       0       7     444       0
    0 DRIBU (buoys and scatterometer)
  0:   385 165    1413       0       0     962       0       7     444       0
    0 Bodenmeldungen, BUOY
  0:
-------------------------------------------------------------------------------
-------------------------------------------------
  0:  TEMP         118      87       0       9       0       0      22       0
    0 TEMP
  0:   520  35      26      22       0       0       0       0       4       0
    0 In-Situ Beob.,  TEMP,  PART A
  0:   521  35      28      19       0       0       0       0       9       0
    0 In-Situ Beob.,  TEMP,  PART B
  0:   522  35      24      17       0       6       0       0       1       0
    0 In-Situ Beob.,  TEMP,  PART C
  0:   523  35      24      17       0       2       0       0       5       0
    0 In-Situ Beob.,  TEMP,  PART D
  0:   776  36       6       4       0       1       0       0       1       0
    0 In-Situ Beob.,  TEMP,  SHIP, PART A
  0:   777  36       6       5       0       0       0       0       1       0
    0 In-Situ Beob.,  TEMP,  SHIP, PART B
  0:   778  36       2       2       0       0       0       0       0       0
    0 In-Situ Beob.,  TEMP,  SHIP, PART C
  0:   779  36       2       1       0       0       0       0       1       0
    0 In-Situ Beob.,  TEMP,  SHIP, PART D
  0:
-------------------------------------------------------------------------------
-------------------------------------------------
  0:  PILOT        187      88       0       1       0      24      74       0
    0 PILOT
  0:   512  32      75      38       0       1       0      15      21       0
    0 In-Situ Beob.,  PILOT, PART A
  0:   513  32      90      36       0       0       0       9      45       0
    0 In-Situ Beob.,  PILOT, PART B
```

```
   0:   514  32      11       6       0       0       0       0       5       0
     0 In-Situ Beob.,  PILOT, PART C
   0:   515  32      11       8       0       0       0       0       3       0
     0 In-Situ Beob.,  PILOT, PART D
   0:
-------------------------------------------------------------------------------
-------------------------------------------------
   0:  SATEM          0       0       0       0       0       0       0       0
     0 SATEM (not implemented)
   0:
-------------------------------------------------------------------------------
-------------------------------------------------
   0:  PAOB           0       0       0       0       0       0       0       0
     0 PAOB pressure observations derived from sat.
   0:
-------------------------------------------------------------------------------
-------------------------------------------------
   0:  SCATT       8685       0       0       0    8685       0       0       0
     0 Scatterometer
   0:  1697  64    8685       0       0       0    8685       0       0       0
     0 Satellitenbeob, QUICKSCAT (als DRIBU)
   0:
-------------------------------------------------------------------------------
-------------------------------------------------
   0:  GPSRO          0       0       0       0       0       0       0       0
     0 GPS Radio occultations (bending angles)
   0:
-------------------------------------------------------------------------------
-------------------------------------------------
   0:  unused         0       0       0       0       0       0       0       0
     0 unused
   0:
-------------------------------------------------------------------------------
-------------------------------------------------
   0:  TOVS      154258       0       0      26       0  147962    6270       0
     0 ATOVS
   0:
-------------------------------------------------------------------------------
-------------------------------------------------
   0:  total     433951     175       0   39543   38816  320193   35224       0
     0
   0:
   0:  SYNOP reports DISMISSED due to:
   0:
   0:    SUBTYP         33340
   0:    CORR              61
   0:    CORRERR           52
   0:    DBLSTAT         1508
   0:    DBLERR           515
   0:    INSDAT          1111
   0:    RULE            1928
   0:           total   38515
   0:
   0:  AIREP reports DISMISSED due to:
   0:
   0:    INSDAT            30
   0:           total      30
   0:
   0:  DRIBU reports DISMISSED due to:
   0:
   0:    DBLSTAT           19
   0:    DBLERR             6
   0:    INSDAT           937
   0:           total     962
   0:
   0:  TEMP reports DISMISSED due to:
   0:
   0:    DBLSTAT            6
   0:    INSDAT             3
   0:           total       9
   0:
   0:  PILOT reports DISMISSED due to:
   0:
   0:    INSDAT             1
   0:           total       1
   0:
   0:  TOVS reports DISMISSED due to:
   0:
   0:    INSDAT            26
   0:           total      26
   0:
   0:  SATOB reports PASSIVE due to:
   0:
   0:    NOTUSED        30131
   0:           total   30131
   0:
   0:  SCATT reports PASSIVE due to:
   0:
   0:    NONE            8685
   0:           total    8685
   0:
   0:  SYNOP reports REJECTED due to:
```

```
 0:
 0:    FG               34
 0:    BLACKLIST         6
 0:           total      40
 0:
 0:  AIREP reports REJECTED due to:
 0:
 0:    THIN           19880
 0:    FG               228
 0:           total   20108
 0:
 0:  SATOB reports REJECTED due to:
 0:
 0:    THIN           79248
 0:    SURF            5573
 0:    FG              1470
 0:    QI             65761
 0:           total  152052
 0:
 0:  DRIBU reports REJECTED due to:
 0:
 0:    FG                7
 0:           total       7
 0:
 0:  PILOT reports REJECTED due to:
 0:
 0:    FG                1
 0:    BLACKLIST        23
 0:           total      24
 0:
 0:  TOVS reports REJECTED due to:
 0:
 0:    THIN           74674
 0:    SURF             232
 0:    DATASET        73056
 0:           total  147962
 0:
 0:-------------------------------------------------------------------------
```

21.6 Analysis step

21.6.1 Redistribution of observations

Before starting the analysis step observations are redistributed over boxes and processor elements.

21.6.2 Initialization of the RTTOV package

The RTTOV package is initialized and coefficient files are read.

```
 0:-------------------------------------------------------------------------
 0:
 0: Input to RTTVI / RTTOV7:
 0:
 0:platform     :   1   1   1   1   1   1   1   1   1   9
 0:satellite    :  15  15  15  16  16  16  17  17  17   2
 0:instrument   :   0   3   4   0   3   4   0   3   4   3
 0:numchans     :   0   0   0   0   0   0   0   0   0   0
 0:
 0:    1    1   15    0  noaa-15         hirs
ATOVS_COEFFS_DIR/rtcoef_noaa_15_hirs.dat
 0:    2    1   15    3  noaa-15         amsua
ATOVS_COEFFS_DIR/rtcoef_noaa_15_amsua.dat
 0:    3    1   15    4  noaa-15         amsub
ATOVS_COEFFS_DIR/rtcoef_noaa_15_amsub.dat
 0:    4    1   16    0  noaa-16         hirs
ATOVS_COEFFS_DIR/rtcoef_noaa_16_hirs.dat
 0:    5    1   16    3  noaa-16         amsua
ATOVS_COEFFS_DIR/rtcoef_noaa_16_amsua.dat
 0:    6    1   16    4  noaa-16         amsub
ATOVS_COEFFS_DIR/rtcoef_noaa_16_amsub.dat
 0:    7    1   17    0  noaa-17         hirs
ATOVS_COEFFS_DIR/rtcoef_noaa_17_hirs.dat
 0:    8    1   17    3  noaa-17         amsua
ATOVS_COEFFS_DIR/rtcoef_noaa_17_amsua.dat
 0:    9    1   17    4  noaa-17         amsub
ATOVS_COEFFS_DIR/rtcoef_noaa_17_amsub.dat
 0:   10    9    2    3  eos-2           amsua
ATOVS_COEFFS_DIR/rtcoef_eos_2_amsua.dat
```

```
0:
0:------------------------------------------------------------------------------
0:
0: Output from RTTVI:
0:
0:kerr   :   0
0:kppf   :   1
0:kpnsat :  10
0:kplev  :  43
0:kpch   :2378
0:kpchus :2378
0:kpnav  :   4
0:kpnsav :   5
0:kpnssv :   6
0:kpncv  :   2
0:
0:  profiles:
0:
0:   preslev    otmin    otmax    oqmin    oqmax   oozmin   oozmax    ooref
0:     hPa        K        K      g/kg     g/kg    mg/kg    mg/gk    mg/gk
0:    10.00   162.00   335.50     0.00     0.01     1.01    16.31     9.69
0:    29.00   173.12   335.79     0.00     0.01     1.86    16.85    10.00
0:    69.00   168.91   352.80     0.00     0.01     2.10    17.03    10.12
0:   142.00   160.85   354.40     0.00     0.01     2.11    17.08    10.18
0:   261.00   160.52   349.42     0.00     0.01     2.11    17.11    10.22
. . .
0: 92246.00   195.47   352.15     0.07   240.20     0.00     0.17     0.05
0: 95744.00   188.15   354.71     0.07   269.60     0.00     0.16     0.05
0: 98588.00   154.95   356.59     0.07   278.90     0.00     0.16     0.04
0:100543.00   135.00   357.86     0.07   282.00     0.00     0.16     0.04
0:101325.00   135.00   385.87     0.07   283.80     0.00     0.16     0.04
0:
0:  valid channel numbers
0:
0:sat.idx :       1   2   3   4   5   6   7   8   9  10
0:satellite   :  15  15  15  16  16  16  17  17  17   2
0:instrument  :   0   3   4   0   3   4   0   3   4   3
0:
0:chan.idx:   1   1   1   1   1   1   1   1   1   1   1
0:chan.idx:   2   2   2   2   2   2   2   2   2   2   2
0:chan.idx:   3   3   3   3   3   3   3   3   3   3   3
0:chan.idx:   4   4   4   4   4   4   4   4   4   4   4
0:chan.idx:   5   5   5   5   5   5   5   5   5   5   5
0:chan.idx:   6   6   6   0   6   6   0   6   6   0   6
0:chan.idx:   7   7   7   0   7   7   0   7   7   0   7
0:chan.idx:   8   8   8   0   8   8   0   8   8   0   8
0:chan.idx:   9   9   9   0   9   9   0   9   9   0   9
0:chan.idx:  10  10  10   0  10  10   0  10  10   0  10
0:chan.idx:  11  11  11   0  11  11   0  11  11   0  11
0:chan.idx:  12  12  12   0  12  12   0  12  12   0  12
0:chan.idx:  13  13  13   0  13  13   0  13  13   0  13
0:chan.idx:  14  14  14   0  14  14   0  14  14   0  14
0:chan.idx:  15  15  15   0  15  15   0  15  15   0  15
0:chan.idx:  16  16   0   0  16   0   0  16   0   0   0
0:chan.idx:  17  17   0   0  17   0   0  17   0   0   0
0:chan.idx:  18  18   0   0  18   0   0  18   0   0   0
0:chan.idx:  19  19   0   0  19   0   0  19   0   0   0
0:chan.idx:  20   0   0   0   0   0   0   0   0   0   0
0:chan.idx:  21   0   0   0   0   0   0   0   0   0   0
. . .
0:chan.idx:  40   0   0   0   0   0   0   0   0   0   0
0:------------------------------------------------------------------------------
```

21.6.3 Set up communication patterns for observations

In order to explicitely represent the background error covariance matrix in observation space $\mathbf{HBH}^T$, the observation operator matrix $\mathbf{H}$ has to be applied to both sides of the covariance matrix $\mathbf{B}$. Thus parts of $\mathbf{H}$ must be passed from the processor where H was evaluated to the other processors where it has to be used as well. The communication patterns for that purpose are derived here.

```
0:------------------------------------------------------------------------------
0:
0:  setting up communication flags for observations (H)
0:
0:  pe  box   spot char statid     n   dest...
0:   0    1      4  252 LH3537    29   1   2   3   4   8  10  14  16  23  28
0:   0    1      5  252 10771     29   1   2   3   4   8  10  14  16  23  28
0:   0    1     89  252 EU0109    29   1   2   3   4   8  10  14  16  23  28
0:   0    1     90  252 EU5830    29   1   2   3   4   8  10  14  16  23  28
0:   0    1     91  252 EU5830    29   1   2   3   4   8  10  14  16  23  28
0:   0    1     92  252 LH844     30   1   2   3   4   8  10  14  16  23  28
0:   0    1     93  252 LH844     30   1   2   3   4   8  10  14  16  23  28
```

```
0:   0    1     94  252 LH844     29   1   2   3   4   8  10  14  16  23  28
0:   0    1     95  252 LH844     30   1   2   3   4   8  10  14  16  20  23
0:   0    1     96  252 LH844     30   1   2   3   4   8  10  14  16  20  23
0:   0    1     97  252 LH3383    30   1   2   3   4   8  10  14  16  20  23
0:   0    1     98  252 LH230     29   1   2   3   4   8  10  14  16  23  28
0:   0    1     99  252 LH230     29   1   2   3   4   8  10  14  16  23  28
. . .
0:  95  288    160  244 AQUA      12  34  45  51  55  68  70  74  75  77  81
0:  95  288    161  244 AQUA      13  34  38  45  51  55  68  70  74  75  77
0:  95  288    162  244 AQUA      12  34  45  51  55  68  70  74  75  77  81
0:  95  288    163  244 AQUA      12  34  45  51  55  68  70  74  75  77  81
0:  95  288    164  244 AQUA      12  34  45  51  55  68  70  74  75  77  81
0:  95  288    165  244 AQUA      12  34  45  51  55  68  70  74  75  77  81
0:  95  288    166  244 AQUA      12  34  45  51  55  68  70  74  75  77  81
0:  95  288    167  244 AQUA      12  34  45  51  55  68  70  74  75  77  81
0:  95  288    168  244 AQUA      12  34  45  51  55  68  70  74  75  77  81
0:  95  288    169  244 AQUA      12  34  45  51  55  68  70  74  75  77  81
0:-------------------------------------------------------------------------
```

21.6.4 Derivation of the background error covariance matrix

Now the explicit (sparse) representation of the background error covariance matrix in observation space is calculated. Progress of the calculations is reported.

```
0:-----------------------------------------------------------
0: set_Pb: block 1 / 288 processed.
0: set_Pb: block 2 / 288 processed.
0: set_Pb: block 3 / 288 processed.
0: set_Pb: block 4 / 288 processed.
0: set_Pb: block 5 / 288 processed.
. . .
0: set_Pb: block 284 / 288 processed.
0: set_Pb: block 285 / 288 processed.
0: set_Pb: block 286 / 288 processed.
0: set_Pb: block 287 / 288 processed.
0: set_Pb: block 288 / 288 processed.
```

The number of nonzero coefficients is reported for each matrix block, as well as the minimum and maximum distance between observations related to the left and right hand side of the matrix block, respectively.

```
   0:  pe   block     m     n   nonzero       total  %         min    max
distance
   0:
   0:   0   1   1   421   421    124417      235225  52.89       0.000
403.755
   0:   0   1   2   421   416    112957      218250  51.76       4.195
510.270
   0:   0   2   2   416   416    102270      202500  50.50       0.000
309.997
   0:   0   1   3   421   417    130928      246380  53.14       5.571
556.850
   0:   0   2   3   416   417    117190      228600  51.26     281.609
713.470
   0:   0   3   3   417   417    135887      258064  52.66       0.000
328.115
   . . .

   0:   0   3 264   417   418     92293      235712  39.15    1343.741
4783.369
   0:   0   1 270   421   422        12      238620   0.01    3916.781
4784.968
   0:   0   3 270   417   422       145      249936   0.06    3874.434
4784.820
   0:   0   1 276   421   414     54039      216795  24.93    1673.515
4784.973
   0:   0   3 276   417   414     64185      227076  28.27    1614.673
4784.915
   0:   0   2 287   416   421     20716      194400  10.66    2597.133
4784.987
   0:   1   5   1   419   421    125452      227465  55.15     189.998
763.324
   0:   1   4   2   417   416    102471      200250  51.17     684.698
1244.633
   0:   1   6   2   417   416    114218      223200  51.17     432.716
918.137
   0:   1   5   3   419   417    131205      238252  55.07      11.940
530.105
   0:   1   4   4   417   417    102587      198025  51.81       0.000
456.395
   . . .
   0:  95 288 282   419   416     11963     3967364   0.30    2918.747
```

```
4784.668
   0:  95 287 285   421   420        12     1231200   0.00    3705.332
4784.626
   0:  95 286 286   422   422    177644     7678441   2.31      11.029
1860.389
   0:  95 286 287   422   421     12873     1197072   1.08    3052.477
4784.958
   0:  95 287 287   421   421    146881      186624  78.70       0.507
4670.559
   0:  95 288 288   419   419    166473     3225616   5.16       0.000
2180.171
```

A summary of the total number of coefficients allocated on each processor is reported.

```
   0:   0       total           9158706  1731666313   0.53
   0:   1       total           9662907  1731666313   0.56
   0:   2       total          10023955  1731666313   0.58
   0:   3       total          10960377  1731666313   0.63
   0:   4       total           8692637  1731666313   0.50
   0:   5       total          18198546  1731666313   1.05
   . . .
   0:  91       total           4108753  1731666313   0.24
   0:  92       total           9982533  1731666313   0.58
   0:  93       total           4262991  1731666313   0.25
   0:  94       total           5676948  1731666313   0.33
   0:  95       total           5655390  1731666313   0.33
   0: all       total         953142803  1731666313  55.04
```

The sparsity pattern of the matrix $\mathbf{HBH}^T$ is only reported if the number of boxes does not exceed 256. Thus the printout is suppressed here.

```
   0:  Matrix Block encoding:
   0:
   0:      : all elements are zero
   0:    + : full n x m representation
   0:    x : only lower triangle is stored
   0:    - : compressed sparse row storage
   0:    | : compressed sparse column storage
   0:    . : not stored due to symmetry
   0:------------------------------------------------------------
```

21.6.5 Check the operator implementation of vertical covariance matrices

If both the matrix representation and the operator representation of the vertical covariance matrices are present, the consistency of the two representations is checked by multiplying both of them with a random vector and comparing the results.

The differences are printed in detail for the first box located on processor 0:

```
   0:------------------------------------------------------------------------
   0:
   0:test_B_oo: test the operator implementation of B
   0:
   0:   ib   is    i  typ  x          y_explicit  y_operator  difference
   0:    1    1    1    1  -0.2590933   1.0576991   1.0576991  -0.0000000
   0:    1    1    2    4   1.6015922  28.7493461  28.7493463  -0.0000002
   0:
   0:    1    2    3    1  -1.4989612   1.0577360   1.0577372  -0.0000012
   0:    1    2    4    4   0.1747676  28.7493461  28.7493463  -0.0000002
   0:
   0:    1    3    5    1   0.1192641   1.0577749   1.0577753  -0.0000004
   0:    1    3    6    4  -0.3020232  28.7493461  28.7493463  -0.0000002
   0:
   0:    1    4    7    8   0.4581814  12.5770027  12.5770029  -0.0000002
   0:    1    4    8   16   0.1889846  11.6392758  11.6392753   0.0000005
   0:    1    4    9    2   0.3949742 -11.1691786 -11.1691791   0.0000005
   . . .
   0:    1  157  416    8  -0.3810747  10.5640007  10.5640006   0.0000001
   0:    1  157  417   16   1.6291501  22.8783190  22.8783196  -0.0000006
   0:    1  157  418    2  -0.3620953 -16.0862929 -16.0862939   0.0000010
   0:
   0:    1  158  419    8  -0.5258410  11.2173443  11.2173441   0.0000002
   0:    1  158  420   16  -0.4714969  21.1797884  21.1797882   0.0000001
   0:    1  158  421    2   0.5701788 -12.3612765 -12.3612774   0.0000009
```

In order to be able to track down any problems to certain observation types, maximum absolute values of the differences are reported for all combinations of observation types at the left and right hand side of the matrix, respectively:

```
   0:       all                       all   0.0000063 0.0000028 0.0000031
0.0000015 0.0000017 0.0000024 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
   0:   1 z     geopotential                0.0000027 0.0000028 0.0000005
0.0000000 0.0000005 0.0000005 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
   0:   2 t     temperature                 0.0000030 0.0000008 0.0000030
0.0000010 0.0000006 0.0000006 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
   0:   4 rh    relative humidity           0.0000016 0.0000000 0.0000003
0.0000015 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
   0:   8 u     wind u component            0.0000017 0.0000007 0.0000006
0.0000000 0.0000013 0.0000005 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
   0:  16 v     wind v component            0.0000020 0.0000007 0.0000006
0.0000000 0.0000004 0.0000015 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
   0:  32 zs    geopotential at the surface  0.0000000 0.0000000 0.0000000
0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
   0:  64 rawbt brightness temperature      0.0000000 0.0000000 0.0000000
0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
   0: 128 bangl bending angle               0.0000000 0.0000000 0.0000000
0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
   0: 256 q     specific humidity           0.0000000 0.0000000 0.0000000
0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
   0: 512 sink  dummy (sink) variable       0.0000000 0.0000000 0.0000000
0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
   0:                                 norm   0.0000000 0.0000000 0.0000000
0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
   0:                                          all z         t         rh
    u         v         zs        rawbt     bangl     q         sink
   0:  norm, diff = 0.288E-02 0.241E-01 0.241E-01 0.191E-08
   0:--------------------------------------------------------------------------
```

21.6.6 Positive definiteness check

The smallest eigenvalues of the block diagonals of the matrices $\mathbf{HBH}^T$, $\mathbf{R}$, and $\mathbf{HBH}^T$+$\mathbf{R}$ are calculated. The test is performed for the whole block as well as for their diagonals and for submatrices only consisting of the same type of observations or reports. If negative eigenvalues occur an attempt is made to attribute them to specific reports.

```
0:--------------------------------------------------------------------------
-
   0:
   0:  Check matrix for small eigenvalues
   0:
   0:    matrix = HBHi
   0:   scaling = T
   0:       fix = 1
   0:   verbose = 2
   0:
   0:  test    pe block  dbkz  type   station  count  hits  e-value size file
   0:     4     0     1    -1                      0    65 -0.1E-05  421
   0:     5     0     1    -1 SYNOP                0    39 -0.3E-13  199
   0:     6     0     1    -1 SYNOP                1     7 -0.1E-13  101
   0:     6     0     1    -1 SYNOP                4    20 -0.7E-15   98
   0:     4     0     2    -1                      0   111 -0.2E-05  416
   0:     5     0     2    -1 SYNOP                0    54 -0.1E-12  314
   0:     6     0     2    -1 SYNOP                1     7 -0.1E-13  157
   0:     6     0     2    -1 SYNOP                4    32 -0.7E-15  157
   0:     4     0     3    -1                      0    51 -0.1E-05  417
   0:     5     0     3    -1 SYNOP                0    28 -0.1E-13  150
   0:     6     0     3    -1 SYNOP                1     4 -0.9E-14   75
   0:     6     0     3    -1 SYNOP                4    17 -0.5E-15   75
   . . .
   0:     4    92   277    -1                      0    25 -0.8E-06  422
   0:     5    92   277    -1 SYNOP                0    19 -0.4E-13  146
   0:     6    92   277    -1 SYNOP                1     1 -0.3E-14   51
   0:     6    92   277    -1 SYNOP                4     3 -0.1E-15   35
   0:     6    92   277    -1 SYNOP                8     2 -0.2E-15   30
   0:     6    92   277    -1 SYNOP               16     2 -0.1E-15   30
   0:     5    94   283    -1 SYNOP                0     1 -0.7E-16   32
   0:     5    94   284    -1 SYNOP                0     1 -0.5E-17   12
   0:     4    94   285    -1                      0     1 -0.1E-14  420
   0:     4    95   288    -1                      0     4 -0.8E-06  419
   0:     5    95   288    -1 SYNOP                0     2 -0.3E-13   36
   0:     6    95   288    -1 SYNOP                4     1 -0.7E-16   16
   0:
   0:  test             failed  hits    bound   min.ev
   0:     1 (diagonal) :     0     0  0.0E+00  0.5E-01
   0:     2 (rep./data):     0     2  0.0E+00 -0.1E-14
   0:     3 (report)   :     0     1  0.0E+00 -0.2E-14
   0:     4 (block)    :     3  2973  0.0E+00 -0.2E-05
```

```
   0:     5 (rep.-type):     3  1660  0.0E+00 -0.2E-12
   0:     6 (rep-t/d-t):     6  1129  0.0E+00 -0.5E-13

0:--------------------------------------------------------------------------------
-
   0:
   0:  Check matrix for small eigenvalues
   0:
   0:    matrix = R
   0:   scaling = T
   0:       fix = 2
   0:   verbose = 2
   0:
   0:  test    pe block  dbkz  type   station  count  hits  e-value size file
   0:
   0:  test             failed  hits    bound   min.ev
   0:     1 (diagonal) :     0     0  0.0E+00  0.1E+00
   0:     2 (rep./data):     0     0  0.0E+00  0.1E+00
   0:     3 (report)   :     0     0  0.0E+00  0.1E+00
   0:     4 (block)    :     0     0  0.0E+00  0.1E+00
   0:     5 (rep.-type):     0     0  0.0E+00  0.1E+00
   0:     6 (rep-t/d-t):     0     0  0.0E+00  0.1E+00

0:--------------------------------------------------------------------------------
-
   0:
   0:  Check matrix for small eigenvalues
   0:
   0:    matrix = P+R
   0:   scaling = F
   0:       fix = 2
   0:   verbose = 2
   0:
   0:  test    pe block  dbkz  type   station  count  hits  e-value size file
   0:
   0:  test             failed  hits    bound   min.ev
   0:     1 (diagonal) :     0     0  0.0E+00  0.3E-01
   0:     2 (rep./data):     0     0  0.0E+00  0.2E-01
   0:     3 (report)   :     0     0  0.0E+00  0.2E-01
   0:     4 (block)    :     0     0  0.0E+00  0.2E-01
   0:     5 (rep.-type):     0     0  0.0E+00  0.2E-01
   0:     6 (rep-t/d-t):     0     0  0.0E+00  0.2E-01

0:--------------------------------------------------------------------------------
-
```

The OI covariance model may lead to negative eigenvalues of $\mathbf{HBH}^T$ in the 3dvar. Depending on the parameters passed to the test routines negative eigenvalues are forced to zero.

21.6.7 Iterative solution of the analysis equation

For each iteration of the outer (nonlinear) loop, the set of flags valid for the iteration is printed:

```
0:--------------------------------------------------------------------------------
-
   0:
   0:OSAS outer iteration  1
   0:
   0:  fg_b     =  T
   0:  fg_o     =  F
   0:  linesrch =  F
   0:  xlnsrch  =  T
   0:  new_R_H  =  T
   0:  new_H    =  0
   0:  new_R    =  T
   0:  vqcf     =  1.00
```

Values of cost function, residuum and other parameters are printed for each iteration of the inner (CG) loop. In this example 20 inner CG iterations were performed:

```
   0:#
   0:# it inl  il        r.dot.y            J_b            J_o              J
       J_cg        delta_J     delta_J_cg          resid          dJ/dx
   0:#
   0:   0   1   0         0.0000         0.0000     43000.4897     43000.4897
     0.0000         0.0000         0.0000      1.0000000      0.5097480 #ITNL
   0:   1   1   1        -0.0000     15507.5082    685643.3644    701150.8726
```

```
-109857.1093    658150.3829         0.0000      1.8916764      0.7232529 #ITLI
   0:   2   1   2         0.0000      7827.5341     32617.6091     40445.1432
-117872.8306   -660705.7294     -8015.7213      0.2944196      0.3931095 #ITLI
   0:   3   1   3         0.0000      6143.4829     23014.9456     29158.4286
-118989.7471    -11286.7146     -1116.9165      0.1868543      0.2057325 #ITLI
   0:   4   1   4         0.0000      5521.6601     20709.4739     26231.1340
-119372.3449     -2927.2946      -382.5979      0.1391550      0.1769151 #ITLI
   0:   5   1   5         0.0000      5303.8067     19703.0764     25006.8831
-119573.4530     -1224.2509      -201.1081      0.1148099      0.1592638 #ITLI
   0:   6   1   6         0.0000      5287.4361     19557.7915     24845.2277
-119741.6380      -161.6554      -168.1850      0.1073447      0.1305658 #ITLI
   0:   7   1   7         0.0000      5400.9628     18915.6256     24316.5884
-119869.1407      -528.6392      -127.5027      0.0899827      0.1008851 #ITLI
   0:   8   1   8         0.0000      5548.0405     17944.9072     23492.9477
-119950.5349      -823.6408       -81.3942      0.0644773      0.0773148 #ITLI
   0:   9   1   9         0.0000      5661.8088     17541.9337     23203.7425
-120000.2417      -289.2051       -49.7068      0.0507743      0.0581356 #ITLI
   0:  10   1  10         0.0000      5726.2245     17288.8042     23015.0287
-120034.0979      -188.7138       -33.8562      0.0395045      0.0469073 #ITLI
   0:  11   1  11         0.0000      5739.5505     17229.2286     22968.7792
-120061.1606       -46.2496       -27.0627      0.0348344      0.0467553 #ITLI
   0:  12   1  12         0.0000      5701.7241     17236.3927     22938.1167
-120081.4881       -30.6624       -20.3275      0.0316682      0.0386821 #ITLI
   0:  13   1  13         0.0000      5667.6507     17183.6394     22851.2901
-120093.2989       -86.8266       -11.8108      0.0235509      0.0263236 #ITLI
   0:  14   1  14         0.0000      5642.9435     17193.7465     22836.6900
-120101.9434       -14.6001        -8.6445      0.0211220      0.0291193 #ITLI
   0:  15   1  15         0.0000      5638.6272     17198.1903     22836.8176
-120107.3679         0.1275        -5.4245      0.0206295      0.0210431 #ITLI
   0:  16   1  16         0.0000      5643.2782     17166.4005     22809.6787
-120111.5359       -27.1388        -4.1680      0.0166344      0.0236108 #ITLI
   0:  17   1  17         0.0000      5655.0127     17135.1720     22790.1848
-120115.0434       -19.4940        -3.5075      0.0130784      0.0173357 #ITLI
   0:  18   1  18         0.0000      5669.4972     17110.1773     22779.6745
-120117.6503       -10.5103        -2.6069      0.0105032      0.0123090 #ITLI
   0:  19   1  19         0.0000      5676.0225     17100.0214     22776.0439
-120119.2524        -3.6306        -1.6021      0.0093927      0.0137755 #ITLI
   0:  20   1  20         0.0000      5676.0059     17096.8933     22772.8991
-120120.6147        -3.1448        -1.3623      0.0080518      0.0093643 #ITLI
   . . .
   0:
   0:OSAS outer iteration  2
   0:
   0:  fg_b     =  F
   0:  fg_o     =  F
   0:  linesrch =  T
   0:  xlnsrch  =  T
   0:  new_R_H  =  T
   0:  new_H    =  0
   0:  new_R    =  T
   0:  vqcf     =  1.00
   0:
   . . .
   0:#
   0:# it inl  il        r.dot.y            J_b            J_o              J
       J_cg        delta_J     delta_J_cg          resid          dJ/dx
   0:#
   0:  21   2   0         0.0000      1876.0501     28465.7580     30341.8082
-57460.0913      7568.9090     62660.5233      1.0000000      0.1999451 #ITNL
   0:  22   2   1         0.0000      4113.4333    145896.5726    150010.0058
-70652.5463    119668.1977    -13192.4550      2.3504749      0.2798538 #ITLI
   0:  23   2   2         0.0000      3687.1706     28110.5669     31797.7375
-71347.8000   -118212.2683      -695.2538      0.3994167      0.1465836 #ITLI
   0:  24   2   3         0.0000      3481.6862     26022.9587     29504.6449
-71455.7454     -2293.0927      -107.9454      0.6285719      0.0749381 #ITLI
   0:  25   2   4         0.0000      3378.7376     25361.7333     28740.4709
-71499.4969      -764.1740       -43.7515      0.4226129      0.0518180 #ITLI
   0:  26   2   5         0.0000      3337.6182     25163.0381     28500.6564
-71523.2176      -239.8145       -23.7206      0.3328475      0.0508103 #ITLI
   0:  27   2   6         0.0000      3330.7659     25191.0105     28521.7765
-71542.9686        21.1201       -19.7511      0.3315913      0.0423143 #ITLI
   0:  28   2   7         0.0000      3356.8101     25086.0603     28442.8704
-71561.4976       -78.9061       -18.5289      0.2878550      0.0436108 #ITLI
   0:  29   2   8         0.0000      3386.5639     24945.1423     28331.7062
-71571.6692      -111.1643       -10.1716      0.2273417      0.0330797 #ITLI
   0:  30   2   9         0.0000      3423.8673     24840.0498     28263.9170
-71580.9515       -67.7891        -9.2823      0.1747468      0.0234503 #ITLI
   0:  31   2  10         0.0000      3444.4300     24803.8336     28248.2636
-71587.1830       -15.6534        -6.2315      0.1568574      0.0211011 #ITLI
   0:  32   2  11         0.0000      3450.4639     24781.6411     28232.1050
-71592.7601       -16.1586        -5.5771      0.1369956      0.0204702 #ITLI
   0:  33   2  12         0.0000      3440.3844     24803.4505     28243.8349
-71596.9820        11.7299        -4.2219      0.1448390      0.0180577 #ITLI
   0:  34   2  13         0.0000      3427.8943     24787.3414     28215.2357
-71600.2479       -28.5992        -3.2658      0.1120842      0.0147843 #ITLI
   0:  35   2  14         0.0000      3416.8044     24802.0167     28218.8211
-71602.6262         3.5854        -2.3784      0.1119354      0.0132963 #ITLI
   0:  36   2  15         0.0000      3410.7396     24787.9427     28198.6823
-71604.5588       -20.1388        -1.9325      0.0820701      0.0114533 #ITLI
   0:  37   2  16         0.0000      3411.0075     24783.7963     28194.8038
-71606.2555        -3.8785        -1.6968      0.0727639      0.0106047 #ITLI
   0:  38   2  17         0.0000      3413.8369     24776.0744     28189.9113
```

```
-71607.3005        -4.8925        -1.0449      0.0619101      0.0083276 #ITLI
   0:  39   2  18         0.0000      3418.0782     24766.7894     28184.8676
-71608.1165        -5.0437        -0.8160      0.0476879      0.0076294 #ITLI
   0:  40   2  19         0.0000      3420.9732     24762.7211     28183.6943
-71608.6510        -1.1734        -0.5346      0.0430322      0.0062750 #ITLI
   0:  41   2  20         0.0000      3421.9293     24760.0674     28181.9967
-71609.0463        -1.6976        -0.3952      0.0359152      0.0059763 #ITLI
   . . .
   0:
   0:OSAS outer iteration  9
   0:
   0:  fg_b     =  F
   0:  fg_o     =  F
   0:  linesrch =  T
   0:  xlnsrch  =  T
   0:  new_R_H  =  T
   0:  new_H    =  0
   0:  new_R    =  T
   0:  vqcf     =  1.00
   0:
   0:#
   0:# it inl  il        r.dot.y            J_b            J_o              J
       J_cg        delta_J     delta_J_cg          resid          dJ/dx
   0:#
   0: 168   9   0         0.0000      3353.8763     25195.7510     28549.6273
-63096.5435        -0.0000         0.0687      1.0000000      0.0000033 #ITNL
   0: 169   9   1         0.0000      3353.8739     25195.7534     28549.6273
-63096.5435        -0.0000        -0.0000      0.9874884      0.0000025 #ITLI
   0: 170   9   2         0.0000      3353.8707     25195.7566     28549.6273
-63096.5435        -0.0000        -0.0000      0.8606731      0.0000027 #ITLI
   0: 171   9   3         0.0000      3353.8699     25195.7574     28549.6273
-63096.5435        -0.0000        -0.0000      0.7561378      0.0000021 #ITLI
   0: 172   9   4        -0.0000      3353.8708     25195.7566     28549.6273
-63096.5435         0.0000        -0.0000      0.7847892      0.0000018 #ITLI
   0: 173   9   5         0.0000      3353.8720     25195.7553     28549.6273
-63096.5435         0.0000        -0.0000      0.9499225      0.0000023 #ITLI
   0: 174   9   6        -0.0000      3353.8716     25195.7558     28549.6273
-63096.5435         0.0000        -0.0000      1.1675538      0.0000025 #ITLI
   0: 175   9   7         0.0000      3353.8722     25195.7552     28549.6273
-63096.5435        -0.0000        -0.0000      1.0647133      0.0000021 #ITLI
   0: 176   9   8         0.0000      3353.8710     25195.7563     28549.6273
-63096.5435        -0.0000        -0.0000      0.5931741      0.0000017 #ITLI
   0: 177   9   9         0.0000      3353.8707     25195.7567     28549.6273
-63096.5435        -0.0000        -0.0000      0.4808165      0.0000015 #ITLI
   0: 178   9  10        -0.0000      3353.8702     25195.7571     28549.6273
-63096.5435         0.0000        -0.0000      0.5267271      0.0000017 #ITLI
   0: 179   9  11         0.0000      3353.8703     25195.7571     28549.6273
-63096.5435        -0.0000        -0.0000      0.5202085      0.0000019 #ITLI
   0: 180   9  12         0.0000      3353.8704     25195.7570     28549.6273
-63096.5435        -0.0000        -0.0000      0.4166229      0.0000015 #ITLI
   0: 181   9  13         0.0000      3353.8704     25195.7569     28549.6273
-63096.5435        -0.0000        -0.0000      0.3940269      0.0000012 #ITLI
   0: 182   9  14         0.0000      3353.8707     25195.7566     28549.6273
-63096.5435         0.0000        -0.0000      0.3921252      0.0000010 #ITLI
   0: 183   9  15         0.0000      3353.8711     25195.7563     28549.6273
-63096.5435        -0.0000        -0.0000      0.3807942      0.0000010 #ITLI
   0: 184   9  16         0.0000      3353.8715     25195.7558     28549.6273
-63096.5435        -0.0000        -0.0000      0.3016651      0.0000009 #ITLI
   0: 185   9  17         0.0000      3353.8716     25195.7558     28549.6273
-63096.5435        -0.0000        -0.0000      0.2726596      0.0000008 #ITLI
   0: 186   9  18         0.0000      3353.8713     25195.7561     28549.6273
-63096.5435         0.0000        -0.0000      0.2661211      0.0000007 #ITLI
   0: 187   9  19         0.0000      3353.8707     25195.7566     28549.6273
-63096.5435        -0.0000        -0.0000      0.2241733      0.0000007 #ITLI
   0: 188   9  20         0.0000      3353.8705     25195.7569     28549.6273
-63096.5435         0.0000        -0.0000      0.2346517      0.0000008 #ITLI
   0:
   0:OSAS outer iteration 10
   0:
   0:  fg_b     =  F
   0:  fg_o     =  F
   0:  linesrch =  T
   0:  xlnsrch  =  T
   0:  new_R_H  =  T
   0:  new_H    =  0
   0:  new_R    =  T
   0:  vqcf     =  1.00
   0:
   0: exit outer loop:dJdz < gtolo:  0.79248298089770343 4E-06

0:--------------------------------------------------------------------------------
-
```

In this case the outer loop is terminated because the stopping criterium - based on the residuum (gradient of the cost function) - was met.

21.7 Estimation of analysis error

The analysis error is estimated and written to a GRIB file. A summary of the file content is given in the program output:

```
0:------------------------------------------------------------------------------
-
   0:
   0:  Inventory of analysis error file:
   0:
   0: rec     adress         name       date     time tab cnt sub prc cde lvt
levelvl runtype   ni grd nlev
   0:   1|         0|VAR_U       |2006-07-20|18:00:00|202| 78|255|195| 41|100|
  10|analysis| 60|  0|  7
   0:   8|     26222|VAR_V       |2006-07-20|18:00:00|202| 78|255|195| 42|100|
  10|analysis| 60|  0|  7
   0:  15|     52444|VAR_FI      |2006-07-20|18:00:00|202| 78|255|195| 40|100|
  10|analysis| 60|  0|  7
   0:
   0: (t_atm)
   0: time% (t_time)
   0: time% days          : 2453937
   0: time% secs          :         64800
   0: time% yyyymmddhhmmss:20060720180000
   0: ref_time% (t_time)
   0: ref_time% days          : 2453937
   0: ref_time% secs          :         64800
   0: ref_time% yyyymmddhhmmss:20060720180000
   0: grid% (t_grid) :
   0: grid% grid  :          0
   0: grid% levtyp:        100
   0: grid% nx    :         60
   0: grid% ny    :         30
   0: grid% ngl   :          0
   0: grid% nn    :          0
   0: grid% nz    :          7
   0: grid% rot   : F
   0: grid% cyc_x : F
   0: grid% poly  : F
   0: grid% global: F
   0: grid% la1   :        -87.00
   0: grid% lo1   :          0.00
   0: grid% di    :          6.00
   0: grid% dj    :          6.00
   0: grid% a     :    6371229.00
   0: grid% g     :          9.81
   0: grid% size  :      12600
   0: grid% dlon  : (1),(n)=  0.00      354.
   0: grid% dlat  : (1),(n)= -87.0      87.0
   0: grid% lon(1): (1),(n)=  0.00      6.18
   0: grid% lat(1): (1),(n)= -1.52      1.52
   0: grid% rlon  : min,max=  0.00      6.18     |   0.00      354.     [degree]
   0: grid% rlat  : min,max= -1.52      1.52     |  -87.0      87.0     [degree]
   0: grid% slm   : not associated!
   0: grid% geosp : not associated!
   0: grid% geo_sh: not associated!
   0: grid% geoid : not associated!
   0: grid% k= 1 ak=      0.0000 bk= 0.0000 akf=   1000.0000 bkf= 0.0000
   0: grid% k= 2 ak=      0.0000 bk= 0.0000 akf=   5000.0000 bkf= 0.0000
   0: grid% k= 3 ak=      0.0000 bk= 0.0000 akf=  10000.0000 bkf= 0.0000
   0: grid% k= 4 ak=      0.0000 bk= 0.0000 akf=  20000.0000 bkf= 0.0000
   0: grid% k= 5 ak=      0.0000 bk= 0.0000 akf=  30000.0000 bkf= 0.0000
   0: grid% k= 6 ak=      0.0000 bk= 0.0000 akf=  50000.0000 bkf= 0.0000
   0: grid% k= 7 ak=      0.0000 bk= 0.0000 akf= 100000.0000 bkf= 0.0000
   0: grid% k= 8 ak=      0.0000 bk= 0.0000
   0: lb  :     1     1     1     1
   0: ub  :    60    30     7     1
   0: size:   37800
   0: ps              gg(: not ALLOCATED!)
   0: psr             gg(: not ALLOCATED!)
   0: t               gg(: not ALLOCATED!)
   0: u               gg(: min,max=) 0.329      7.36      41 202
   0: u                 k= 1     0.442      4.62
   0: u                 k= 2     0.329      2.97
   0: u                 k= 3     0.358      3.62
   0: u                 k= 4     0.558      6.14
   0: u                 k= 5     0.617      7.36
   0: u                 k= 6     0.453      5.91
   0: u                 k= 7     0.399      3.55
   0: v               gg(: min,max=) 0.330      7.37      42 202
   0: v                 k= 1     0.442      4.63
   0: v                 k= 2     0.330      2.97
   0: v                 k= 3     0.357      3.62
   0: v                 k= 4     0.566      6.13
   0: v                 k= 5     0.600      7.37
   0: v                 k= 6     0.476      5.92
   0: v                 k= 7     0.384      3.52
   0: q               gg(: not ALLOCATED!)
   0: qcl             gg(: not ALLOCATED!)
```

```
   0: qci             gg(: not ALLOCATED!)
   0: tke             gg(: not ALLOCATED!)
   0: cl              gg(: not ALLOCATED!)
   0: ph              gg(: not ALLOCATED!)
   0: pf              gg(: not ALLOCATED!)
   0: rh              gg(: not ALLOCATED!)
   0: t2m             gg(: not ALLOCATED!)
   0: rh2m            gg(: not ALLOCATED!)
   0: geoh            gg(: not ALLOCATED!)
   0: geof            gg(: min,max=)  1.00     83.3      40 202
   0: geof              k= 1      4.77      83.3
   0: geof              k= 2      2.65      39.2
   0: geof              k= 3      2.58      32.2
   0: geof              k= 4      3.27      40.0
   0: geof              k= 5      2.92      48.4
   0: geof              k= 6      2.34      41.1
   0: geof              k= 7      1.00      33.9
   0: tsurf           gg(: not ALLOCATED!)
   0: td2m            gg(: not ALLOCATED!)
   0: tv              gg(: not ALLOCATED!)
   0: psi             gg(: not ALLOCATED!)
   0: chi             gg(: not ALLOCATED!)
   0: size =   37800
   0:

0:--------------------------------------------------------------------------------
-
```

21.8 Post-multiplication

The progress of the post-multiplication is reported.

```
0:--------------------------------------------------------------------------------
-
   0:
   0:  apply_B_mi newpl =F
   0:
   0: apply_B_mi: box, nbox, ncol 1 96 40
   0: apply_B_mi: pass 2
   0: apply_B_mi: box, nbox, ncol 2 96 40
   0: apply_B_mi: pass 2
   0: apply_B_mi: box, nbox, ncol 3 96 40
   0: apply_B_mi: pass 2
   . . .
   0: apply_B_mi: box, nbox, ncol 94 96 43
   0: apply_B_mi: pass 2
   0: apply_B_mi: box, nbox, ncol 95 96 43
   0: apply_B_mi: pass 2
   0: apply_B_mi: box, nbox, ncol 96 96 45
   0: apply_B_mi: pass 2

0:--------------------------------------------------------------------------------
-
```

21.9 Output of the analysis

A summary of the analysis increment and of the analysis itself is printed in the job output. The analysis is writtem to a GRIB file.

```
0:--------------------------------------------------------------------------------
-
   0:
   0:   analysis increment
   0:
   0: (t_atm)
   0: time% (t_time)
   0: time% days          : 2453937
   0: time% secs          :         64800
   0: time% yyyymmddhhmmss:20060720180000
   0: ref_time% (t_time)
   0: ref_time% days          : 2453937
   0: ref_time% secs          :         64800
   0: ref_time% yyyymmddhhmmss:20060720180000
   0: grid% (t_grid) :
   0: grid% grid  :        192
   0: grid% levtyp:        110
   0: grid% nx    :        193
   0: grid% ny    :        193
```

```
0: grid% ngl   :          0
0: grid% nn    :          0
0: grid% nz    :         40
0: grid% rot   : F
0: grid% cyc_x : F
0: grid% poly  : F
0: grid% global: T
0: grid% la1   :          0.00
0: grid% lo1   :          0.00
0: grid% di    :          0.00
0: grid% dj    :          0.00
0: grid% a     :    6371229.00
0: grid% g     :          9.81
0: grid% size  :   14899600
0: grid% dlon  : (1),(n)=  0.00      0.00
0: grid% dlat  : (1),(n)=  0.00      0.00
0: grid% lon(1): (1),(n)=  0.00    -0.628
0: grid% lat(1): (1),(n)=  1.57     0.459
0: grid% rlon  : min,max= -3.14      3.14     |  -180.      180.     [degree]
0: grid% rlat  : min,max= -1.57      1.57     |  -90.0      90.0     [degree]
0: grid% slm   : min,max=  0.00      1.00
0: grid% geosp : min,max= -963.     0.595E+05 |  -98.2     0.607E+04 [m]
0: grid% geo_sh: not associated!
0: grid% geoid : min,max= -107.      85.2
0: grid% k= 1 ak=      0.0000 bk= 0.0000 akf=   1000.0000 bkf= 0.0000
0: grid% k= 2 ak=   2000.0000 bk= 0.0000 akf=   3000.0000 bkf= 0.0000
0: grid% k= 3 ak=   4000.0000 bk= 0.0000 akf=   5000.0000 bkf= 0.0000
0: grid% k= 4 ak=   6000.0000 bk= 0.0000 akf=   7000.0000 bkf= 0.0000
0: grid% k= 5 ak=   8000.0000 bk= 0.0000 akf=   8988.0684 bkf= 0.0002
. . .
0: grid% k=37 ak=    210.3939 bk= 0.9797 akf=    138.1415 bkf= 0.9840
0: grid% k=38 ak=     65.8892 bk= 0.9883 akf=     36.6284 bkf= 0.9911
0: grid% k=39 ak=      7.3677 bk= 0.9940 akf=      3.6838 bkf= 0.9958
0: grid% k=40 ak=      0.0000 bk= 0.9976 akf=      0.0000 bkf= 0.9988
0: grid% k=41 ak=      0.0000 bk= 1.0000
0: lb  :     0     1     1     1
0: ub  :    23    16    40    10
0: size:  622080
0: ps              gg(: min,max=) -519.      499.       1   2
0: psr             gg(: min,max=) 0.488E+05 0.104E+06   1   2
0: t               gg(: min,max=) -2.24      2.22      11   2
0: t                 k= 1     -1.33      1.34
0: t                 k= 2     -1.21      1.05
0: t                 k= 3     -1.41      2.22
0: t                 k= 4     -1.29      1.61
0: t                 k= 5     -1.51     0.862
. . .
0: t                 k=35     -1.67      1.53
0: t                 k=36     -1.65      1.54
0: t                 k=37     -1.69      1.54
0: t                 k=38     -1.69      1.53
0: t                 k=39     -1.68      1.54
0: t                 k=40     -1.67      1.54
0: u               gg(: min,max=) -11.3      9.25      33   2
0: u                 k= 1     -2.00      1.84
0: u                 k= 2     -1.35      1.87
0: u                 k= 3     -1.73      1.27
0: u                 k= 4     -1.89      2.44
0: u                 k= 5     -3.16      3.79
. . .
0: u                 k=35     -6.00      3.87
0: u                 k=36     -5.92      3.85
0: u                 k=37     -5.85      3.85
0: u                 k=38     -5.81      3.84
0: u                 k=39     -5.77      3.83
0: u                 k=40     -5.75      3.81
0: v               gg(: min,max=) -12.1      9.93      34   2
0: v                 k= 1     -2.56      1.84
0: v                 k= 2     -1.42      1.35
0: v                 k= 3     -1.40      1.77
0: v                 k= 4     -1.58      2.33
0: v                 k= 5     -2.42      2.25
. . .
0: v                 k=35     -7.00      4.41
0: v                 k=36     -6.46      4.29
0: v                 k=37     -6.06      4.20
0: v                 k=38     -5.79      4.12
0: v                 k=39     -5.62      4.07
0: v                 k=40     -5.51      4.03
0: q               gg(: not ALLOCATED!)
0: qcl             gg(: not ALLOCATED!)
0: qci             gg(: not ALLOCATED!)
0: tke             gg(: not ALLOCATED!)
0: cl              gg(: not ALLOCATED!)
0: ph              gg(: not ALLOCATED!)
0: pf              gg(: not ALLOCATED!)
0: rh              gg(: min,max=)-0.385     0.341     255 255
0: rh                k= 1      0.00      0.00
0: rh                k= 2      0.00      0.00
0: rh                k= 3      0.00      0.00
0: rh                k= 4      0.00      0.00
0: rh                k= 5      0.00      0.00
```

```
   . . .
   0: rh                k=35    -0.383     0.165
   0: rh                k=36    -0.384     0.165
   0: rh                k=37    -0.383     0.166
   0: rh                k=38    -0.381     0.165
   0: rh                k=39    -0.380     0.164
   0: rh                k=40    -0.378     0.164
   0: t2m             gg(: not ALLOCATED!)
   0: rh2m            gg(: not ALLOCATED!)
   0: geoh            gg(: not ALLOCATED!)
   0: geof            gg(: not ALLOCATED!)
   0: tsurf           gg(: not ALLOCATED!)
   0: td2m            gg(: not ALLOCATED!)
   0: tv              gg(: not ALLOCATED!)
   0: psi             gg(: not ALLOCATED!)
   0: chi             gg(: not ALLOCATED!)
   0: size =  622080
   0:

0:--------------------------------------------------------------------------------
-
   0:
   0:   analysis
   0:
   0: (t_atm)
   . . .
   0: size =  933120
   0:

0:--------------------------------------------------------------------------------
-
   0:
   0:  Inventory of analysis file:
   0:
   0: rec     adress         name       date     time tab cnt sub prc cde lvt
levelvl runtype   ni grd nlev
   0:   1|         0|PS          |2006-07-20|18:00:00|  2| 78|255|173|  1|  1|
0|  0|analysis|192|192|  1
   0:   2|    745426|T           |2006-07-20|18:00:00|  2| 78|255|173| 11|110|
1|  2|analysis|192|192| 40
   0:  42|  30562466|U           |2006-07-20|18:00:00|  2| 78|255|173| 33|110|
1|  2|analysis|192|192| 40
   0:  82|  60379506|V           |2006-07-20|18:00:00|  2| 78|255|173| 34|110|
1|  2|analysis|192|192| 40
   0: 122|  90196546|QV          |2006-07-20|18:00:00|  2| 78|255|173| 51|110|
1|  2|analysis|192|192| 40
   0: 162| 120013586|QC          |2006-07-20|18:00:00|201| 78|255|173| 31|110|
1|  2|analysis|192|192| 40
   0: 202| 149830626|QI          |2006-07-20|18:00:00|201| 78|255|173| 33|110|
1|  2|analysis|192|192| 40
   0: 242| 179647666|T_2M        |2006-07-20|18:00:00|  2| 78|255|173| 11|105|
2|  0|analysis|192|192|  1
   0: 243| 180393092|TD_2M       |2006-07-20|18:00:00|  2| 78|255|173| 17|105|
2|  0|analysis|192|192|  1

0:--------------------------------------------------------------------------------
-
```

21.10 Report on observations used

Statistics on the usage of observations are printed.

```
0:--------------------------------------------------------------------------------
-
   0:
   0:    Observation Report Statistics
   0:
   0:   obstype proces.  merged   split   dism. passive reject.  active accept.
  lost description
   0:  dbkz code
   0:
--------------------------------------------------------------------------------
-------------------------------------------------
   0:  SYNOP      46939       0       0   38515       0      40    8384       0
     0 SYNOP
   0:     0  11    5878       0       0    1649       0      18    4211       0
     0 Bodenmeldungen, SYNOP, Sect. 1-4, manuell+PAST
   0:     1 140   29424       0       0   29424       0       0       0       0
     0 Bodenmeldungen, METAR
   0:     5  11    3916       0       0    3916       0       0       0       0
     0 Bodenmeldungen, SYNOP, Section 5
   0:   128  14    5691       0       0    2552       0      21    3118       0
     0 Bodenmeldungen, SYNOP, Sect. 1-3, autom.
   0:   256  21     881       0       0     427       0       1     453       0
     0 Bodenmeldungen, SHIP, manuell
   0:   384  24    1149       0       0     547       0       0     602       0
```

```
    0 Bodenmeldungen, SHIP, automatisch
  0:
--------------------------------------------------------------------------------
------------------------------------------------
  0:  AIREP      30120       0       0      30       0   20108    9982       0
    0 AIREP
  0:   529 144    3981       0       0       0       0    2113    1868       0
    0 In-Situ Beob.,  AMDAR
  0:   530  -1    3942       0       0       0       0    2692    1250       0
    0 In-Situ Beob.,  AIREP
  0:   533 145   14059       0       0       0       0    9297    4762       0
    0 In-Situ Beob.,  ACARS-Daten USA
  0:   534 145    8138       0       0      30       0    6006    2102       0
    0 In-Situ Beob.,  ACARS-Daten EUROPA (Bracknell)
  0:
--------------------------------------------------------------------------------
------------------------------------------------
  0:  SATOB     192231       0       0       0   30131  152052   10048       0
    0 SATOB (AMV)
  0:  1704  90  135013       0       0       0   26144  104673    4196       0
    0 Satellitenbeob, AMV,   Eumetsat
  0:  1705  90   48270       0       0       0    3987   41150    3133       0
    0 Satellitenbeob, AMV,   GOES (NOAA/NESDIS)
  0:  1706  90    8948       0       0       0       0    6229    2719       0
    0 Satellitenbeob, AMV,   MODIS
  0:
--------------------------------------------------------------------------------
------------------------------------------------
  0:  DRIBU       1413       0       0     962       0       7     444       0
    0 DRIBU (buoys and scatterometer)
  0:   385 165    1413       0       0     962       0       7     444       0
    0 Bodenmeldungen, BUOY
  0:
--------------------------------------------------------------------------------
------------------------------------------------
  0:  TEMP         118      87       0       9       0       0      22       0
    0 TEMP
  0:   520  35      26      22       0       0       0       0       4       0
    0 In-Situ Beob.,  TEMP,  PART A
  0:   521  35      28      19       0       0       0       0       9       0
    0 In-Situ Beob.,  TEMP,  PART B
  0:   522  35      24      17       0       6       0       0       1       0
    0 In-Situ Beob.,  TEMP,  PART C
  0:   523  35      24      17       0       2       0       0       5       0
    0 In-Situ Beob.,  TEMP,  PART D
  0:   776  36       6       4       0       1       0       0       1       0
    0 In-Situ Beob.,  TEMP,  SHIP, PART A
  0:   777  36       6       5       0       0       0       0       1       0
    0 In-Situ Beob.,  TEMP,  SHIP, PART B
  0:   778  36       2       2       0       0       0       0       0       0
    0 In-Situ Beob.,  TEMP,  SHIP, PART C
  0:   779  36       2       1       0       0       0       0       1       0
    0 In-Situ Beob.,  TEMP,  SHIP, PART D
  0:
--------------------------------------------------------------------------------
------------------------------------------------
  0:  PILOT        187      88       0       1       0      24      74       0
    0 PILOT
  0:   512  32      75      38       0       1       0      15      21       0
    0 In-Situ Beob.,  PILOT, PART A
  0:   513  32      90      36       0       0       0       9      45       0
    0 In-Situ Beob.,  PILOT, PART B
  0:   514  32      11       6       0       0       0       0       5       0
    0 In-Situ Beob.,  PILOT, PART C
  0:   515  32      11       8       0       0       0       0       3       0
    0 In-Situ Beob.,  PILOT, PART D
  0:
--------------------------------------------------------------------------------
------------------------------------------------
  0:  SATEM          0       0       0       0       0       0       0       0
    0 SATEM (not implemented)
  0:
--------------------------------------------------------------------------------
------------------------------------------------
  0:  PAOB           0       0       0       0       0       0       0       0
    0 PAOB pressure observations derived from sat.
  0:
--------------------------------------------------------------------------------
------------------------------------------------
  0:  SCATT       8685       0       0       0    8685       0       0       0
    0 Scatterometer
  0:  1697  64    8685       0       0       0    8685       0       0       0
    0 Satellitenbeob, QUICKSCAT (als DRIBU)
  0:
--------------------------------------------------------------------------------
------------------------------------------------
  0:  GPSRO          0       0       0       0       0       0       0       0
    0 GPS Radio occultations (bending angles)
  0:
--------------------------------------------------------------------------------
------------------------------------------------
  0:  unused         0       0       0       0       0       0       0       0
```

```
     0 unused
   0:
--------------------------------------------------------------------------------
------------------------------------------------
   0:  TOVS      154258       0       0      26       0  147962    6270       0
     0 ATOVS
   0:
--------------------------------------------------------------------------------
------------------------------------------------
   0:  total     433951     175       0   39543   38816  320193   35224       0
     0
   0:
   0:  SYNOP reports DISMISSED due to:
   0:
   0:    SUBTYP         33340
   0:    CORR              61
   0:    CORRERR           52
   0:    DBLSTAT         1508
   0:    DBLERR           515
   0:    INSDAT          1111
   0:    RULE            1928
   0:           total   38515
   0:
   0:  AIREP reports DISMISSED due to:
   0:
   0:    INSDAT            30
   0:           total      30
   0:
   0:  DRIBU reports DISMISSED due to:
   0:
   0:    DBLSTAT           19
   0:    DBLERR             6
   0:    INSDAT           937
   0:           total     962
   0:
   0:  TEMP reports DISMISSED due to:
   0:
   0:    DBLSTAT            6
   0:    INSDAT             3
   0:           total       9
   0:
   0:  PILOT reports DISMISSED due to:
   0:
   0:    INSDAT             1
   0:           total       1
   0:
   0:  TOVS reports DISMISSED due to:
   0:
   0:    INSDAT            26
   0:           total      26
   0:
   0:  SATOB reports PASSIVE due to:
   0:
   0:    NOTUSED        30131
   0:           total   30131
   0:
   0:  SCATT reports PASSIVE due to:
   0:
   0:    NONE            8685
   0:           total    8685
   0:
   0:  SYNOP reports REJECTED due to:
   0:
   0:    FG                34
   0:    BLACKLIST          6
   0:           total      40
   0:
   0:  AIREP reports REJECTED due to:
   0:
   0:    THIN           19880
   0:    FG               228
   0:           total   20108
   0:
   0:  SATOB reports REJECTED due to:
   0:
   0:    THIN           79248
   0:    SURF            5573
   0:    FG              1470
   0:    QI             65761
   0:           total  152052
   0:
   0:  DRIBU reports REJECTED due to:
   0:
   0:    FG                 7
   0:           total       7
   0:
   0:  PILOT reports REJECTED due to:
   0:
   0:    FG                 1
   0:    BLACKLIST         23
   0:           total      24
   0:
```

```
   0:  TOVS reports REJECTED due to:
   0:
   0:    THIN           74674
   0:    SURF             232
   0:    DATASET        73056
   0:           total  147962
   0:

0:--------------------------------------------------------------------------------
-
```

21.11 Report on CPU time resorces

The amount of CPU and wall time required for each step of the computation is printed:

```
0:--------------------------------------------------------------------------------
-
   0:
   0: cpu(min)   (max)  (mean)   wall percent  task
   0:    8.14   12.06    9.65   20.49   47.11  read observations
   0:    0.58    2.57    1.76    3.55   49.62  read inventory
   0:    0.80    0.85    0.83    2.24   36.93  read grid
   0:    3.68    3.87    3.78    7.58   49.82  read background state
   0:    4.42    9.80    7.19   14.40   49.92  set up background error operators
   0:    0.00    0.02    0.01    0.02   31.77  thinning of observations
   0:   14.57   16.04   15.31   30.65   49.94  set up FG boxes
   0:    1.19    1.59    1.39    2.79   49.86  broadcast observations to PEs
(FG)
   0:    0.54    0.70    0.65    1.40   46.31  scatter model columns (FG)
   0:    0.25    0.48    0.30    0.77   38.51  set up FG data structures
   0:    0.06    0.38    0.20    0.80   24.83  interpolate (FG)
   0:    0.56    2.23    1.24    4.55   27.20  run observation operators (FG)
   0:    5.32    6.28    5.80   11.62   49.94  set background error correlations
(FG)
   0:    2.11    4.00    3.00    7.78   38.50  perform FG checks
   0:   19.66   23.95   21.95   46.46   47.24  write monitoring files
   0:   23.18   27.09   25.13   50.37   49.88  redistribute observations
   0:    0.00    0.02    0.01    0.03   39.93  cleanup after FG scan
   0:    0.00    0.01    0.00    0.01   34.37  set up PSAS boxes
   0:    0.22    0.30    0.27    0.58   47.05  scatter model columns
   0:    0.37    0.73    0.50    1.17   42.65  setup PSAS data structure
   0:    0.00    0.02    0.01    0.04   31.77  interpolate
   0:    6.08    6.32    6.23   12.49   49.86  run observation operators (for H)
   0:  130.76  144.44  137.51  275.69   49.88  set background error correlations
   0:   96.35  109.07  102.68  205.71   49.91  test operator implementation of B
   0:    0.01    0.03    0.02    0.05   44.79  calculate preconditioning matrix
   0:    5.73    6.64    6.18   12.38   49.96  Test R,B,R+B for positive
definiteness
   0:    0.01    0.02    0.01    0.03   40.62  Cholesky factorization of R
   0:    0.08    0.10    0.09    0.19   48.63  Cholesky factorization of B+R
   0:    0.00    0.01    0.00    0.01   21.87  interpolate first guess
   0:    0.04    0.07    0.06    0.14   41.22  1st iteration
   0:    0.17    0.18    0.17    0.35   49.97  variational quality control
   0:    0.34    0.36    0.35    0.69   50.66  recalculate R^(-1)
   0:    0.10    0.13    0.11    0.24   46.66  recalculate preconditioner
   0:   10.44   11.19   10.81   21.67   49.90  CG solver
   0:   95.03  101.80   98.40  197.18   49.90  iterations > 1
   0:    3.45    4.26    3.85    7.72   49.91  estimate analysis error
   0:    0.01    0.03    0.02    0.06   38.37  distribute model columns for post
mult.
   0:   30.20   30.70   30.44   60.99   49.92  post multiplication
   0:    0.03    0.07    0.05    0.13   40.14  gather results on model grid
   0:    0.94    1.03    1.00    2.05   48.80  add (interpolate) increment
   0:    0.03    0.05    0.04    0.07   50.30  write summary of GRIB-files
   0:    8.50    9.99    9.33   18.69   49.92  write GRIB-file
   0:    2.95   34.74   20.49   66.20   30.95  write psas.info file
   0:  505.30  553.13  530.68 1090.03   48.68  Total

0:--------------------------------------------------------------------------------
-
```

A good load balance is achieved if the mean CPU time amounts to approximately 50% of the wall time. (Not 100/multi-threading).

Part IV

Implementation

Implementation-Part needs LETKF-modules!

Chapter 22

Overview (Tutorial)

This chapter briefly describes the usage of modules, derived types and procedures of the DWD data assimilation system. More formal coding standards are given in **Coding Standards Datenassimilation**.

22.1 Program Execution in the Parallel Environment

22.1.1 MPI (Message Passing Interface)

The data assimilation, post-processing, and utility programs must run in a parallel environment to fulfil memory and run-time requirements. Parallelisation of the code is done by using the Message Passing Library (MPI). The routines provided the MPI library are encapsulated the module **mo_mpi**. For the specific procedures of the MPI library This module provides generic routines, applicable to variables of different type and arrays of different rank. The generic routines automatically chose the correct MPI data type and buffer size. If a communicator is not passed MPI_COMMON_WORLD is assumed.

In a non-parallel environment the data-assimilation code is compiled with the directive -DNOMPI. In this case the interface of module **mo_mpi** remains the same and calling the procedures has no effect so that in general no further changes to the code are required.

The most important entities provided by module **mo_mpi** are listed below:

subroutines to initialise and shut down the MPI environment	
p_start	initialise the parallel environment
p_stop	finish MPI and clean up all PEs
p_abort	abort the program (with error code /= 0)
generic point to point communication routines	
p_send	MPI send routine
p_recv	MPI receive routine
p_sendrecv	MPI sendrecv routine
p_isend	MPI non-blocking send
p_irecv	MPI non-blocking receive
requests for non blocking calls	
p_wait	waits for an MPI send or receive to complete

p_probe	check incoming messages w/o actually receiving
generic collective communication routines	
p_barrier	all PEs wait until all PEs have reached this point
p_bcast	broadcast data from one to all other PEs
p_bcast_ptr	broadcast a pointer and its content to all other PEs
p_alltoall	all PEs send and receive to/from all PEs
p_gather	one PE gathers data from all PEs
p_allgather	all PEs gather data from all PEs
p_scatterv	one PE scatters vector data to all PEs
p_gatherv	one PE gathers vector data from all PEs
alltoallv_args	set up/check arguments for MPI alltoallv
allgatherv_args	set up/check arguments for MPI allgatherv
gatherv_args	set up/check arguments for MPI gatherv
generic MPI reduction routines (minimum, maximum, ..)	
p_max	calculates maximum value of variable on all PEs
p_min	calculates minimum value of variable on all PEs
p_sum	calculates sum of values of variable on all PEs
p_or	calculates logical .or. of variable on all PEs
p_and	calculates logical .and. of variable on all PEs
processor element identifiers	
p_pe	Id of this PE (in MPI_COMM_WORLD)
p_io	Id of the processor dedicated to do I/O
p_nprocs	number of available processor elements
logical flags	
p_parallel	true for parallel environment
p_parallel_io	true if this PE is p_io
pre-defined MPI communicator group Ids	
MPI_COMM_WORLD	group consisting of all PEs
MPI_COMM_SELF	group consisting of this PE only
MPI_COMM_NULL	group consisting of no PE

22.1.2 Program Start and Stop

When a parallel job is started by the batch system each instance is started independently from each other. In order to communicate the Message Passing interface must be initiated at the beginning of the program and finalised at its end by calling specific MPI routines (MPI_init, ...). In the data assimilation environment these tasks are fulfilled by calling the routines **p_start** and **p_stop** from module mo_mpi. In case of an error routine **finish** from module **mo_exception** may be called to abort the program.

In the following example program instance of the program prints out its processor element number. The program aborts if it is run on 13 processors.

```
program test
  !-------------
  ! Modules used
```

```
  !-------------
  use mo_mpi,       only: p_start,  &! set up the MPI environment
                          p_stop,   &! shut down the mpi_environment
                          p_pe,     &! Id of THIS pe (starting with 0)
                          p_nprocs, &! number of available processor elements
                          p_barrier  ! MPI barrier routine
  use mo_exception, only: finish     ! abort in case of error
  implicit none
  !-------------
  ! program code
  !-------------
  call p_start                       ! set up the MPI environment
  if (p_nprocs == 13) call finish('program test','ERROR: running on 13 PEs')
  print *,'My PE number is',p_pe     ! print out processor Id
  call p_barrier                     ! wait until all PEs are ready
  call p_stop                        ! shut down the mpi_environment
end program test
```

22.1.3 Input and Output in the Parallel Environment

The MPI standard does not guarantee that all processor elements can do I/O. However most platforms allow I/O for all PEs.

For input it is possible to read the same file in parallel if the information is required by more than one processor. However this is not advisable because internal communication using the MPI library is in general more efficient than accessing a file multiple times. Thus files are in general read by one processor and the information is scattered or broadcasted to the others.

A single file cannot easily be written by multiple processors at the same time. Thus it is required to gather the information on one processor and then write it to the file.

Below the approaches chosen for different situations are summarised.

Steering of the program flow by namelist input

Fortran namelist input is the recommended way for steering the program flow in operational applications. We read the namelist input by one dedicated PE and when distribute it via the MPI interface. Details are given is Section 22.1.5.

Interactive steering of the program

For interactive applications namelist input is not appropriate. Parallel interactive input is faciliated using routines from module **mo_dialog** and further described in Section 22.1.6.

Job control output (stdout)

The output sequence in a sequential file from different PEs is unpredictable. Thus Job control output to standard output (stdout, Fortran unit 6) should be done only from processor p_io. Merging of output from different processors is supported by routines from module **mo_p_output** and further described in Section 22.1.7.

Errors and warning messages (stderr)
Urgent output (error and warning messages) may be written from individual processors to standard error (stderr, Fortran unit 0). In contrast to stdout this unit is in general not buffered and appears in the output immediately. As the sequence of the output from the different PEs is unpredictable this method should be used rarely. Output to stderr can be performed using subroutine **message** from module **mo_exception** as well.

Fortran file I/O
Files handled by the Fortran I/O library are identified by Fortran Unit Numbers. In the data assimilation program bookkeeping on used and unused unit numbers is performed by module **mo_fortran_units** and further described in Section 22.1.4.

Atmospheric fields
Atmospheric fields in the data assimilation system are represented by means of variables of derived type **t_atm** (state variables) and **t_grid** defined in modules **mo_atm_state** and **mo_atm_grid**, respectively (cf. Section 22.2). I/O to of these fields to GRIB files is handled by module **mo_grib** (cf. Section 22.2.5). Atmospheric fields on a regular latitude-longitude or Gaussian grid can also be written to a binary file readable by the GRADS plotting utility (cf. Section 28.5.1).

Observational data
Observational data may be held in different representations as described in Section **??**). Input of the original observational data (in BUFR format or converted by BUFR2NetCDF) is done by routines specific to the observation types.

Observational data which has already been processed either by the data assimilation system or by the COnsortium for Small-scale MOdelling (COSMO) model is stored in NetCDF feedback format. This format is further described in **Feedback File Definition**.

22.1.4 Fortran I/O

Files handled by the Fortran I/O library are identified by Fortran Unit Numbers. In the data assimilation program bookkeeping on used and unused unit numbers is performed by entities from module **mo_fortran_units**:

get_unit_number	reserve a unit number
return_unit_number	release a unit number
reserve_unit_number	reserve a specific unit number
print_used_units	used for debugging

A typical code sequence which reads/writes files via the Fortran I/O library in a parallel environment may look like:

```
subroutine sub

use mo_mpi,           only: p_pe,             &! Id of THIS pe (starting with 0)
                            p_io,             &! Id of processor dedicated to do I/
```

```
                              p_nprocs,          &! number of available processor elem
                              p_bcast,           &! generic MPI broadcast routine
                              p_gather            ! generic MPI gather routine
  use mo_fortran_units, only: get_unit_number,   &! reserve a unit number
                              return_unit_number  ! release a unit number
  implicit none

  integer :: unit                                 ! Fortran unit number to use
  real    :: a                                    ! input
  real    :: r (p_nprocs)                         ! result from the different Processo

  if (p_pe == p_io) then                          ! read input on one PE only
    write (6,*) 'enter number'                    ! prompt on standard output
    read  (5,*) a                                 ! read from standard input
  endif

  call p_bcast (a, p_io)                          ! broadcast input from p_io to all P
  a = a * a                                       ! do some calculations
  call p_gather (a, r, p_io)                      ! gather the results on one PE

  if (p_pe == p_io) then                          ! write result on one PE only
    unit = get_unit_number()                      ! reserve a unit number
    open (unit, file='my_results',action='write') ! open output file
    write (unit,*) r                              ! write result to file
    close (unit)                                  ! close file
    call return_unit_number(unit)                 ! release the unit number
  endif

  end subroutine sub
```

22.1.5 Namelist Input

Fortran namelist input is the recommended way for steering the program flow in operational applications.

A namelist input file consists of a number of namelist groups, each of them holding a number of 'variable=value' pairs, for instance:

```
&GROUP_A
  A = 1.
  B = 2.
/
&GROUP_B
  D = 3.
/
```

When the namelist is read, the values of the respective variables in the program are

replaced by the values from the file, if present.

Without any precautions namelist groups must be read in the order of their presence in the input file. Therefore module **mo_namelist** provides a routine to position the input file at the beginning of a specified namelist group, allowing to detect if a namelist group is present or to read a namelist group multiple times:

open_nml	open namelist file
position_nml	position file at start of namelist group
close_nml	close namelist file
nnml	Fortran unit used for namelist input
POSITIONED	return value from position_nml: namelist group found
MISSING	return value from position_nml: namelist group not found

A typical code sequence to read a namelist looks like:

```
subroutine read_namelist

use mo_mpi,       only: p_pe,               &! Id of THIS pe (starting with 0)
                        p_io,               &! Id of processor dedicated to do I/O
                        p_bcast              ! generic MPI broadcast routine
use mo_exception, only: finish               ! abort in case of error
use mo_namelist,  only: open_nml,           &! open the namelist file
                        nnml,               &! Fortran unit used for namelist input
                        position_nml,       &! position file at specific namelist gr
                        POSITIONED,         &! return code for successful positionir
                        MISSING              ! return code for missing namelist grou
implicit none

logical  :: first                            ! rewind namelist file for first try
integer  :: status                           ! return code from 'position_nml'
real     :: a, b, c                          ! declare namelist variables
namelist /GROUP_A/ a, b, c                   ! declare namelist group

first = .true.                               ! set mark for first try to read the na
do                                           ! read namelist group several times
  if (p_pe == p_io) then                     ! read on dedicated PE only
    if (first) call open_nml ('namelist')    ! open namelist file (if not yet done)
    a=-1.; b=-1.; c=-1.                      ! set default values for namelist varia
    call position_nml ('GROUP_A',    &       ! position namelist group
                       rewind=first, &
                       status=status )
    first=.false.                            ! remember that group was already read
    select case (status)
    case (POSITIONED)                        ! read namelist group if found
      read (nnml ,nml=GROUP_A)
    case (MISSING)                           ! do nothing if namelist group not four
      write(6,*) 'GROUP_A: not found'
    case default
```

```
        call finish('read_namelist','GROUP_A') ! abort in case of error
      end select
      write(6,*) 'GROUP_A:',a,b,c              ! printout namelist variables read
    endif
    call p_bcast (status, p_io)                ! broadcast namelist variables if read
    if (status==POSITIONED) then
      call p_bcast (a, p_io)
      call p_bcast (b, p_io)
      call p_bcast (c, p_io)
    else
      exit                                     ! exit the loop if no further namelist
    endif
  end do
  end subroutine read_namelist
```

Running the above code results in the output:

```
 GROUP_A:  1.00000000       2.0000000     -1.00000000
 GROUP_A: not found
 GROUP_A: -1.00000000     -1.00000000     -1.00000000
```

22.1.6 Interactive Steering of the Program

Namelist input as described in the previous system is recommended for steering of batch programs but not suitable for interactive applications. For the latter module **mo_dialog** provides some routines for interactive input in a parallel environment:

ask_menue	menue with different options
ask	get value of a variable
ask_y_n	ask for answer 'y'es or 'n'o

ask is a generic routine for querying for the value of a string, integer, single or double precision real, or vector of integer, real or logical variable. An example program may look like:

```
program test
  use mo_mpi,           only: p_start,        &! set up the MPI environment
                              p_barrier,      &! wait until all PEs reached this point
                              p_stop           ! shut down the mpi_environment
  use mo_dialog,        only: ask_menue,      &! enter menue
                              ask,            &! ask for anything
                              ask_y_n          ! ask for a logical (answer 'y' or 'n')
  implicit none

  logical   :: l
  integer   :: j, i(2)
  character :: c
```

```
  call p_start                                    ! set up the MPI environment
  do
    c =  ask_menue (’’,                       &   ! present the menue
                    ’+ add two numbers ’,     &
                    ’* multiply two numbers’)
    call ask (i,’enter integer numbers’)          ! enter the arguments
    select case (c)
    case (’+’)                                    ! calculations
      j = i(1) + i(2)
    case (’*’)
      j = i(1) * i(2)
    end select
    write (6,*) ’result =’,j                      ! printout (on all PEs !!!)
    call p_barrier                                ! wait until all PEs are ready
    l = ask_y_n (’continue ?’)                    ! ask for continuing
    if (.not.l) exit
  end do
  call p_stop                                     ! shut down the mpi_environment
end program test
```

22.1.7 Job Control Output

The output sequence to a sequential file from different PEs is unpredictable. Thus Job control output to standard output (stdout, Fortran unit 6) should be done only from processor p_io. Merging of output from different processors is supported by routines from module **mo_p_output**. The following entities are accessible from this module:

oline	output line buffer (array of strings)
iol	index of next line to write to buffer
nextline	routine to increment actual buffer line number
flush_buf	routine to gather lines and write buffer
flush_buf_pio	as write_p, but only if p_pe==p_io
add_line	routine to add a line to the output buffer
add_line_pio	add a line to output buffer for p_pe==p_io only

oline is an array (current size 30k) of character strings (length 132) which has to be filled by each pe and later is gathered and printed by PE p_io. A typical example code may look like:

```
  subroutine job_control_output
  use mo_mpi,      only: p_pe       ! processor id
  use mo_p_output, only: oline,    &! output buffer (array of strings)
                         iol,      &! index of next line to write in buffer
                         nextline, &! routine to increment buffer line number
                         flush_buf  ! routine to write buffer to stdout
  integer :: i

  call flush_buf                    ! flush old buffer content
```

```
do i=1, p_pe                        ! write lines from individual processors
  call nextline
  write (oline(iol),*) 'printout: line',i,'from processor',p_pe
end do
call flush_buf                      ! flush buffer content
end subroutine job_control_output
```

Running the above program on 4 processors will result in the following output:

```
printout: line           1 from processor           1
printout: line           1 from processor           2
printout: line           2 from processor           2
printout: line           1 from processor           3
printout: line           2 from processor           3
printout: line           3 from processor           3
```

22.1.8 Report on CPU-time, Wall-time and Memory Requirements

For an assessment of the cpu and wall-time requirements of the different parts of the program may be instrumented. The following utility routines are defined in module **mo_cpu_time**:

stop_time Determine cpu and wall-time for the subsequent part of the program.

print_times Print cpu and wall-time times used by the parts of the program.

Example code:

```
program test
use mo_mpi,      only: p_start,       &! set up the MPI environment
                       p_stop,        &! shut down the mpi_environment
                       p_pe, p_io      ! processor element ids
use mo_cpu_time, only: stop_time,     &! stop time used for a program segment
                       print_times     ! print times used for program segments
implicit none

integer              :: i
integer, allocatable :: k (:), j(:)

call p_start                           ! set up the MPI environment

call stop_time ('work on PE 0')
if (p_pe == p_io) then
  allocate (k (1000000))               ! some work on PE 0 only
  k(1) = 1
  do i = 2, 1000000
    k(i) = k(i-1) + i
  end do
```

```
    write (6,*) 'k =',k(1000000)
  endif

  call stop_time ('work on all PEs')
  allocate (j (1000000))
  j(1) = 1                                ! some work on all PEs
  do i = 2, 1000000
    j(i) = j(i-1) + i
  end do
  if (p_pe==p_io)  write (6,*) 'j =',j(1000000)
  call stop_time ('')                     ! empty comment to trace last segment

  call print_times
  call p_stop                             ! shut down the mpi_environment
end program test
```

Each invocation of **stop_time** gives a report on the resources used in the previous program segment. **print_times** prints a summery (usually at the end of the program). Running on 2 processors the above program produces the following output:

```
 k =  1784293664

 cpu(min)   (max)  (mean)   wall percent minrss maxrss  sumrss  task
    0.00    0.02    0.01    0.02   50.00      3      7      10  work on PE 0

-------------------------------------------------------------------------------
 j =  1784293664

 cpu(min)   (max)  (mean)   wall percent minrss maxrss  sumrss  task
    0.02    0.02    0.02    0.02   95.66      7     11      18  work on all PEs

-------------------------------------------------------------------------------

 cpu(min)   (max)  (mean)   wall percent minrss maxrss  sumrss  task
    0.00    0.02    0.01    0.02   50.00      3      7      10  work on PE 0
    0.02    0.02    0.02    0.02   95.66      7     11      18  work on all PEs
    0.04    0.04    0.04    0.04   97.68      7     11      18  Total
```

The columns of the report have the following meaning:

`cpu(min)` :	minimum cpu time (s) used on any processor
`(max)` :	maximum cpu time (s) used on any processor
`(mean)` :	mean cpu time (s) per processor
`wall` :	wall time (s)
`percent` :	mean cpu time/wall time (%)
`minrss` :	minimum memory (Mbyte) used on any processor
`maxrss` :	maximum memory (Mbyte) used on any processor
`sumrss` :	total memory used (Mbyte)
`task` :	description of program segment (parameter to stop_time)

22.2 Representation of Atmospheric Model Data

Model data is stored in variables of derived type **t_atm** defined in module **mo_atm_state** (cf. Figure 22.1). Multiple instances or arrays of this type are suitable for representing different model states (forecast, analysis) or ensembles of model states. The derived type contains multi-dimensional arrays holding the different model fields (temperature, humidity, wind, ...). It further consists of a pointer referencing variables of type **t_grid** defined in module **mo_atm_grid**. This type holds meta data and coefficients as well as invariant fields (orography, land-sea-mask, ...) for the description of the model discretisation as well as information on the MPI partitioning of the model state.

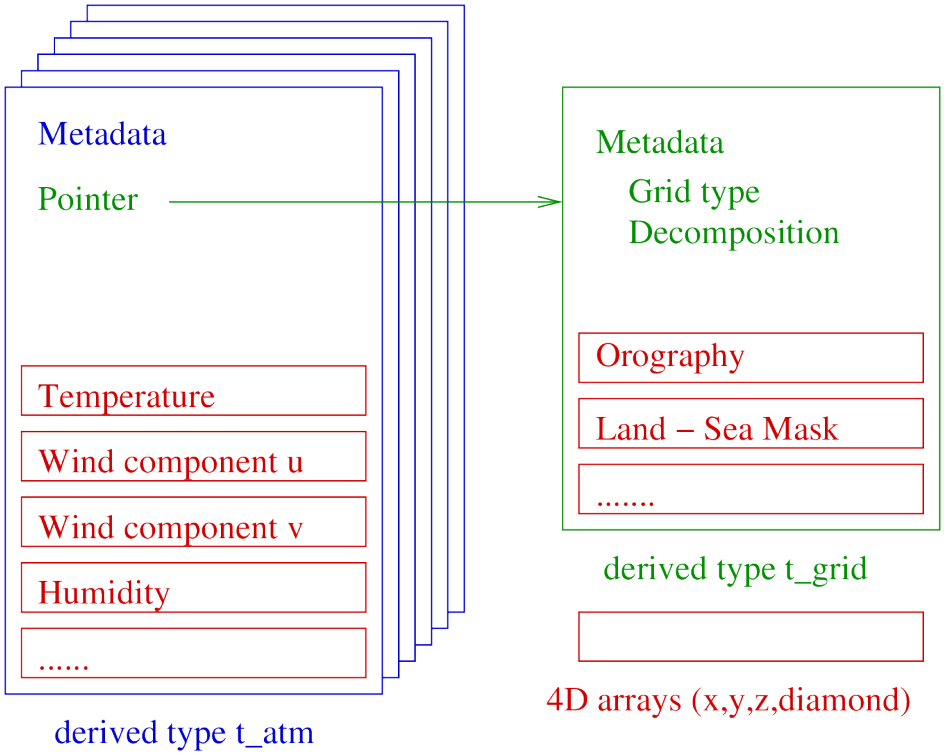


Figure 22.1: Derived types for model states and grid representation. Red: gridded fields. Green: type t_grid holding the specification of the model grid. Blue: type t_atm holding the model state.

22.2.1 Representation of Model Fields in the Parallel Environment

The gridded model fields are represented as arrays of rank 4. The first 3 dimensions correspond to the 2 horizontal coordinates (longitude, latitude in general), and the vertical coordinate. In case of the GME the 4th dimension holds the 10 diamonds, otherwise its size is 1. For surface fields (surface pressure, orography, ...) the size of the 3rd dimension is 1. In a parallel environment the model fields may be horizontally distributed over different processor elements. In this case the horizontal dimensions only run over a subset of the respective grid-points.

The 4-dimensional arrays are directly accessible by pointer components within the derived types **t_atm** and **t_grid**. In addition to that an array component of derived type **t_m** (defined in module **mo_memory**) holds a reference and meta information of each field, e.g. name, GRIB code numbers, ... This allows to loop through all fields which is required for certain operations on the model state and to access the fields by passing their name as a parameter.

22.2.2 Model Grid Description

The derived type **t_grid** (module **mo_atm_grid**) holds information on the following aspects of the model grid:

Horizontal discretization
The type of horizontal grid is indicated by the component **gridtype** (cf. WMO Table 6, module **mo_wmo_tables**). Supported grid-types are regular or rotated latitude-longitude grid (COSMO, HRM), Gaussian grid (IFS, RCHAM), and icosahedral grid (GME).

Vertical discretization
The type of vertical grid is indicated by the component **leveltyp** (cf. WMO Table 3, module **mo_wmo_tables**). Supported level-types are hybrid pressure coordinates (GME, IFS, ECHAM, HRM), or hybrid height (COSMO). The cosmo coordinate system is further indicated by a value of **ivctype**$\neq$0.

Array bounds and partitioning
Array bound values are given by the components **lbg**(1:4) and **ubg**(1:4) (lower and upper bound of each dimension). For individual fields the actual range may differ due to a staggered grid or for soil variables. In a parallel environment only a subset of the horizontal grid-points may be allocated (if **ldec**=.true.) on a given processor. The bounds of the allocated fields on the actual processor are given by the components **lb**(1:4) and **ub**(1:4). Detailed information on the partitioning (neighborhood relationships, etc.) is gigen in the component **dc** of type **t_atm_dec** from module **mo_atm_decomp**.

Invariant fields
Invariant sueface fields related to the coordinate system are accessible:

lsm	land sea mask	
geosp	surface geopotential	(m^2/s^2)
rlon	longitude	(rad)
rlat	latitude	(radians)
geoid	geoid heights (for GNSS radio occultations)	(m)

In case of the COSMO vertical coordinate systen the following invariant fields are provided:

hsurf	geometrical height of surface	(m)
hhl	geometrical height of half levels	(m)
p0	base-state pressure	
dp0	base-state pressure thickness	
rho0	base-state density	

Operations

The following operations are provided by module **mo_atm_grid**:

Grid construction and destruction

construct	Set up the grid description (by optional parameters). In general grids are not set explicitly but read from file (22.2.5).
destruct	Deallocate derived type components.
set_geoid	Set geoid surface field
cosmo_ref_atm	Set fields of COSMO reference atmosphere
coarsen	Derive coarser resolution grid: take each stride.th point.

Printout and plotting

print	Print out the grid meta data.
init_ctl	Set the GRADS meta data from the grid meta data.
to_grads	Dump the invariant fields to a GRADS file.

Interpolation and transformation

phi2phirot	Convert latitude from real geographical system to rotated system
phirot2phi	Convert latitude from one rotated system to another
rlarot2rla	Convert longitude from one rotated system to another
rla2rlarot	Convert longitude from real geographical system to rotated system
hunt	Obtain indices to grid-points for 2d lat-lon coordinates
w_intpol	Return indices and interpolation weights to grid
lin_intpol	Linear interpolation of 2D lat-lon or Gaussian field.
average	Average to coarser grid (nearest neighbour). Loop over invariant fields, pick up selected gridpoints.
lgtd	Perform direct Legendre transform (on geosp)
lgti	Perform inverse Legendre transform

Example code:

```
type (t_grid) , pointer :: grid
```

```
allocate       (grid)
call construct (grid, gridtype = 0,   &
                              nx = 360  )
call print     (grid)
call destruct  (grid)
```

22.2.3 Model States

Model states are represented in the derived type **t_atm** defined in module **mo_atm_state**. The type holds the following information:

Product meta data		
runtype	'forecast', 'analysis', ..)	
runclass	0..3: main-run,pre-assimilation,assimilation,test	
expid	experiment id	
member	ensemble member number	
members	ensemble size	
ensemble_id	ensemble id	
Grid meta data		
time	verification time (start of forecast)	
ref_time	reference time (validity of data)	
grid	pointer to grid description t_grid	
lb (4)	lower bounds of fields (copy from t_grid)	
ub (4)	upper bounds of fields (copy from t_grid)	
Prognostic atmospheric variables		
ps	surface pressure	(Pa)
psr	reference pressure	(Pa)
pp	pressure deviation	
t	temperature	(K)
u	longitudinal wind	(m/s)
v	meridional wind	(m/s)
w	vertical wind	(m/s)
q	specific humidity	(kg/kg)
qcl	specific cloud liquid water	
qci	specific cloud ice	
qr	prognostic rain	
qs	prognostic snow	
qg	prognostic graupel	
Soil variables		
t_so	soil temperature	
w_so	soil humidity	
w_so_ice	soil ice	

Derived variables

clc	cloud cover
ph	half level pressure
pf	full level pressure
rh	relative humidity
t2m	2m temperature
rh2m	2m relative humidity
geoh	geopotential height (half levels)
geof	geopotential height (full levels)
tsurf	surface temperature
fr_ice	sea ice fraction
td2m	2m dewpoint temperature
tv	virtual temperature
psi	streamfunction
chi	velocity potential
vrt	vorticity
div	divergence

Diagnostic variables

vmax_10m	maximum 10m wind speed
tmin_2m	minimum 2m temperature
tmax_2m	maximum 2m temperature
clct	total cloud cover
clcl	low cloud cover
clcm	medium cloud cover
clch	high cloud cover
tot_prec	total precipitation
u_10m	10m wind component
v_10m	10m wind component

Temporary backup for wind components on the C-grid

u_10m_c	10m wind component
v_10m_c	10m wind component
u_c	u component
v_c	v component

Fields to pass trough (COSMO LETKF)

z0	roughness length
soiltyp	soil type
plcov	plant coverage
lai	leaf area index
rootdp	root depth
vio3	vertically integrated O3
hmo3	height of O3 maximum

for_e	Bedeckung Nadelwald
for_d	Bedeckung Laubwald

Fields to pass trough (GME LETKF)

h_ice	ice thickness
t_ice	ice surface temperature
h_snow	snow depth (m)
tqr	integrated rainwater
tqs	integrated snowwater

22.2.4 GRIB file inventory

Model states are generally stored in GRIB files. In order to faciliate the handling of GRIB files module **mo_grib_handling** provides the following entities:

t_inventory	inventory data type
get_inventory	read inventory table
print_inventory	print inventory
p_bcast	broadcast inventory

Example code:

```
use mo_grib_handling, only: t_inventory,   &! Inventory Data Type
                            get_inventory, &! read inventory table
                            print_inventory ! print inventory
implicit none

type(t_inventory), pointer :: inv(:)

nullify (inv)

write(6,*)
write(6,*)' inventory of file gff200908070000.ens.004 (GME ensemble member)'
write(6,*)
call get_inventory      (inv,'gff200908070000.ens.004')
call print_inventory    (inv, first=.true.)

write(6,*)
write(6,*)' inventory of file lbff20090807000000_0 (COSMO deterministic run)'
write(6,*)
call get_inventory      (inv,'lbff20090807000000_0')
call print_inventory    (inv, first=.true.)
```

Example input data may be found at /e/uscratch/arhodin/kenda_bc:

```
lbff20090807000000_0
gff200908070000.ens.004
```

The above program (called with a GME ensemble member and COSMO example file) results in the following printout:

```
 inventory of file gff200908070000.ens.004 (GME ensemble member)

rec    adress ed    name       date     time tab cnt sub prc cde zen lvt levelvl  runtype vvm
  1|        0| 1|PS     |2009-08-07|03:00:00|  2| 78|255|173|  1|  0|  1|  0|  0|forecast| 300
  2|    84957| 1|T      |2009-08-07|03:00:00|  2| 78|255|173| 11|  0|110|  1|  2|forecast| 300
 42|  3483277| 1|U      |2009-08-07|03:00:00|  2| 78|255|173| 33|  0|110|  1|  2|forecast| 300
 82|  6881597| 1|V      |2009-08-07|03:00:00|  2| 78|255|173| 34|  0|110|  1|  2|forecast| 300
122| 10279917| 1|QV     |2009-08-07|03:00:00|  2| 78|255|173| 51|  0|110|  1|  2|forecast| 300
162| 13678237| 1|QC     |2009-08-07|03:00:00|201| 78|255|173| 31|  0|110|  1|  2|forecast| 300
202| 17076557| 1|QI     |2009-08-07|03:00:00|201| 78|255|173| 33|  0|110|  1|  2|forecast| 300
242| 20474877| 1|FR_LAND|0001-01-01|00:00:00|  2| 78|255|173| 81|  0|  1|  0|  0|analysis|   0
243| 20559507| 1|FIS    |0001-01-01|00:00:00|  2| 78|255|173|  6|  0|  1|  0|  0|analysis|   0

 inventory of file lbff20090807000000_0 (COSMO deterministic run)

rec    adress ed    name       date     time tab cnt sub prc cde zen lvt levelvl  runtype vvm
  1|        0| 1|U      |2009-08-07|00:00:00|  2| 78|255|138| 33|  0|110|  1|  2|analysis|   0
 51| 19425300| 1|V      |2009-08-07|00:00:00|  2| 78|255|138| 34|  0|110|  1|  2|analysis|   0
101| 38850600| 1|T      |2009-08-07|00:00:00|  2| 78|255|138| 11|  0|110|  1|  2|analysis|   0
151| 58275900| 1|QV     |2009-08-07|00:00:00|  2| 78|255|138| 51|  0|110|  1|  2|analysis|   0
201| 77701200| 1|QC     |2009-08-07|00:00:00|201| 78|255|138| 31|  0|110|  1|  2|analysis|   0
251| 97126500| 1|PP     |2009-08-07|00:00:00|201| 78|255|138|139|  0|110|  1|  2|analysis|   0
301|116551800| 1|T_SNOW |2009-08-07|00:00:00|201| 78|255|138|203|  0|  1|  0|  0|analysis|   0
302|116940306| 1|QV_S   |2009-08-07|00:00:00|  2| 78|255|138| 51|  0|  1|  0|  0|analysis|   0
303|117328812| 1|W_SNOW |2009-08-07|00:00:00|  2| 78|255|138| 65|  0|  1|  0|  0|analysis|   0
304|117717318| 1|W      |2009-08-07|00:00:00|  2| 78|255|138| 40|  0|109|  1|  0|analysis|   0
355|137531124| 1|QI     |2009-08-07|00:00:00|201| 78|255|138| 33|  0|110|  1|  2|analysis|   0
405|156956424| 1|QR     |2009-08-07|00:00:00|201| 78|255|138| 35|  0|110|  1|  2|analysis|   0
455|176381724| 1|QS     |2009-08-07|00:00:00|201| 78|255|138| 36|  0|110|  1|  2|analysis|   0
505|195807024| 1|HSURF  |2009-08-07|03:00:00|  2| 78|255|137|  8|  0|  1|  0|
0|nudging |   0| 421| 10|  1| -1|  1|  0|ass | 7434
```

The columns have the following meaning:

`rec`	Record number in the GRIB file. Only 1 record is shown for multi-level fields (due to first=.true. in print_inventory)
`adress`	Adress of record in the GRIB file. (Used by GRIB input routines to read selected fields).
`ed`	GRIB edition number (currently 1, GRIB 2 will be implemented soon).
`name`	Name of GRIB field (DWD convention, toggle 3dvar convention by liname=.true. in print_inventory)
`date`	Verification date (toggle to data base time by ldbtime=.true. in print_inventory)
`time`	Verification time (toggle to data base time by ldbtime=.true. in print_inventory)
`tab`	GRIB table number
`cnt`	Generating center, 78 for DWD
`sub`	Generating subcenter 255 for unspecified
`prc`	generating process
`cde`	GRIB code number
`zen`	Zusatzelementnummer (used to identify fields from surface analysis etc.)
`lvt`	Level type (cf. WMO Table 3, module mo_wmo_tables)
`levelvl`	Level value (only 1st record shown for multi-level fields)
`runtype`	Run type, i.e forecast, analysis, nudging, ...
`vvmm`	Forecast interval (hours,minutes)
`ni`	Grid resolution parameter, number of triangles in diamond for GME, number of longitude points for COSMO.
`grd`	Grid type, (cf. WMO Table 6, module mo_wmo_tables)
`nlv`	Number of levels of the field in the file (only 1st record shown)
`mem`	Ensemble size, -1 for deterministic run
`nme`	Number of ensemble members (or deterministic run) found in the file.
`eid`	Ensemble member number (0 for deterministic run)
`clas`	Run class: main (forecast) run, assimilation cycle, ...
`expid`	Experiment number in NUMEX

22.2.5 GRIB I/O

In order to handle GRIB input and output of model fields module **mo_grib** provodes the following entities:

read	generic routine to read model state or grid information
write_grib	generic routine to write model state or grid or single field to GRIB file
read_multi	read a multi-level field
read_single	read a single-level field
SCATTER	flag values
BCAST	flag values

The specific routine **read_atm_grid** of the generic interface **read** to read the grid information takes the following arguments:

`grid`		Variable of derived type to set. The variable must not be initialized (don't call constuct before). It is recommended to use a pointer variable (which must be allocated) to be passed to instances of the atmospheric deriived type later.
`file`		GRIB file name to read. The GDS (Grid Descriptor Section) of a suitable record in the file is evaluated to derive the grid information.
`invt(:)`	optional,pointer	Inventory of the GRIB file. If the variable is given on input, it is not derived from the file conent again. If the pointer is passed but not associated, the inventory is provided on output.
`invar`	optional	Invariant fields File name. If this parameter is given, the invariant fields are read from this seperate file (to be provided only once for GME LETKF).
`nproc1`	optional	Number of processor elements in x-direction if the model fields shall be distributed in a parallel environment.
`nproc2`	optional	Number of processor elements in y-direction if the model fields shall be distributed in a parallel environment. nproc1, nproc2 must be provided so that nproc1*nproc2 equals the total number of processors in the group. Otherwise the complete model fields will be allocated on each processor.
`comm`	optional	MPI communicator group. If the variable is not present MPI_COMM_WORLD is assumed.
`geosp`	optional	If this parameter is passed with value .false. the routine will not abort if the orography invariant field (geosp or hsurf) is missing in the file.
`lsm`	optional	If this parameter is passed with value .false. the routine will not abort if the land-sea-mask invariant field is missing in the file.
`scanmode`	optional	Force a specific scanmode (default:WMO8_J_POSITIVE, cf WMO Table 8 in module mo_wmo_tables)
`member`	optional	Select ensemble member with this number.

The specific routines **read_atm_state** or **read_atm_ens** of the generic interface **read** to read the model or ensemble state take the following arguments:

`atm`		Atmospheric state variable to set. Must be an array in case of an ensemble. The variable must be initialised already (call construct).
`file`		Name of GRIB file to read. In case of an ensemble the name is extended by '.eee' (3 digits ensemble number). In a parallel environment Ensemble members are read in parallel from different processors.
`invt`	optional	Inventory of the GRIB file. May be provided so that it is not set up twice unnecessarily.
`fields`	optional	List of fields to read, separated by blanks.
`runtype`	optional	Run-type to select ('forecast','analysis',...)
`time`	optional	Verification-time to select.
`reftime`	optional	Reference time to select.
`member`	optional	Ensemble member to select.
`optionals`	optional	List of optional fields to read (subset of 'fields'). The routine does not abort if any of these fields is missing in the file.
`unsp_type`	optional	List of fields with unspecified runtype (subset of 'fields'). These fields are read even if their run-type does not correspond with the parameter 'runtype'.

The specific routine **write_grid_grib** of the generic interface **write_grib** writes the invariant fields stored in a variable of type t_grid to a GRIB file and takes the following parameters:

`grid`		Grid variable to write.
`file`		Name of file to write to.
`mode`	optional	Writing mode: 'w' (default for write) or 'a' (append).
`time`	optional	Time to write, default: ????? (for invariant fields).

The optional arguments `grib, pio, ie` are only used if the routine is called from within write_atm_grib (see below).

The specific routines **write_atm_grib** to write a model state and **write_atm_ens** to write an ensemble of model states take the following parameters:

`atm`		Model state to write. Array of model states in case of an ensemble.
`file`		Name of file to write.
`mode`	optional	Writing mode: 'w' (default for write) or 'a' (append).
`grid`	optional	Flag to write grid invariant fields as well (fefault: .false.).
`edition`	optional	GRIB edition (default: 1), Edition 2 is not fully implemented yet.

The specific routine **write_var_grib** is only used internally.

Example code:

```
use mo_atm_state,  only: t_atm,     &! atmospheric state data-type
                         construct, &! initialize model state variable
                         destruct,  &! release model state memory
                         print       ! print model state variable
use mo_atm_grid,   only: t_grid,    &! Type definition for grid information
                         construct, &! initialize model grid variable
```

```
                        destruct    ! release model grid memory
use mo_grib,       only: read,      &! read  model state or grid
                        write_grib  ! write model state or grid
use mo_run_params, only: set_nprocs  ! set processor configuration
implicit none

type (t_grid)      ,pointer :: grid
type (t_atm)                :: state
type (t_atm)                :: ensemble(4)
integer                     :: nproc1, nproc2

!---------------------------------------
! set reasonable processor configuration
!---------------------------------------
call set_nprocs (nproc1, nproc2)
print *,'processor configuration:',nproc1, nproc2

!---------------------------------------
! read and write GME model grid & state
!---------------------------------------
allocate        (grid)
call read       (grid,  'lbff20090807000000_0',&
                        nproc1=nproc1,        &
                        nproc2=nproc2,        &
                        lsm=.false.           )
call construct  (state, grid)
call read       (state, 'lbff20090807000000_0', &
                        fields='u v w t'       )
call write_grib (state, 'lbff20090807000000_copy')
call print      (state,  comment='GME model state')
call destruct   (state)
call destruct   (grid)

!-------------------------------------
! read and write COSMO grid & ensemble
!-------------------------------------
call read       (grid,     'gff200908070000.ens.001',&
                           nproc1=nproc1,           &
                           nproc2=nproc2            )
call construct  (ensemble,  grid)
call read       (ensemble, 'gff200908070000.ens',&
                           fields='u v t ps'   ,&
                           runtype='forecast'   )
call write_grib (ensemble, 'gff200908070000.copy')
call destruct   (ensemble)
```

```
  call destruct   (grid)
  deallocate      (grid)
```

Example input data may be found at /e/uscratch/arhodin/kenda_bc:

```
lbff20090807000000_0
gff200908070000.ens.004
gff200908070000.ens.003
gff200908070000.ens.002
gff200908070000.ens.001
```

Running the program writes the following files:

```
lbff20090807000000_copy
gff200908070000.copy.004
gff200908070000.copy.003
gff200908070000.copy.002
gff200908070000.copy.001
```

Running on 4 PEs the program provides the following output:

```
 processor configuration:           2           2
 processor configuration:           2           2
 processor configuration:           2           2
 processor configuration:           2           2

read_atm_grid: p_readgrib =   0

read_atm_grid: file       =lbff20090807000000_0

 derive vertical coordinates grom GDS:
 center      =          78
 sub_center  =         255
 process     =         138
 local_ident =         254
 repres      =          10
 nv          =          56
       1   100000.00000000
       2      288.14990234
       3       42.00000000
       4    11357.00000000
       5    22000.00000000
...
      52       94.63999939
      53       51.42999268
      54       20.00000000
      55        0.00000000
      56        0.00000000
 COSMO vertical coordinate parameters:
 ke1         =          51
 ivctype     =           2
 irefatm     =           0
 p0sl        =  100000.000000000000
```

```
 t0sl        =   288.14990234375000
 dt0lp       =   42.000000000000000
 vcflat      =   11357.000000000000
 delta_t     =   0.0000000000000000
 h_scal      =   0.0000000000000000
 svc1        =   0.0000000000000000
 svc2        =   0.0000000000000000
 nfltvc      =           0
 read_atm_state: reading u
 read_atm_state: reading v
 read_atm_state: reading w
 read_atm_state: reading t
--------------------------------------------------------------------------------

    GME model state

 (t_atm)
 time% (t_time)
 time% days          : 2455051
 time% secs          :              0
 time% yyyymmddhhmmss:20090807000000
 ref_time% (t_time)
 ref_time% days          : 2455051
 ref_time% secs          :              0
 ref_time% yyyymmddhhmmss:20090807000000
 grid% grid   :         10
 grid% nx     :        421
 grid% ny     :        461
 grid% ngl    :          0
 grid% ni     :          0
 grid% levtyp :        110
 grid% ivctype:          2
 grid% p0sl   :     100000.00000
 grid% t0sl   :        288.14990
 grid% dt0lp  :         42.00000
 grid% vcflat :      11357.00000
 grid% nn     :          0
 grid% nz     :         50
 grid% ns     :          0
 grid% rot    : T
 grid% cyc_x  : F
 grid% poly   : F
 grid% global : F
 grid% arakaw : C
 grid% rlon   : min,max= 0.182E-01 0.346     |   1.04      19.8     [degree]
 grid% rlat   : min,max= 0.780     0.986     |   44.7      56.5     [degree]
 grid% hhl             gg(: min,max=) -5.50     0.220E+05 255 255  1: 51
 grid% p0              gg(: min,max=) 0.341E+04 0.999E+05 255 255  1: 50
 grid% dp0             gg(: min,max=)  109.     0.291E+04 255 255  1: 50
 grid% rho0            gg(: min,max=) 0.810E-01  1.21     255 255  1: 50
 run type       :analysis
 run class      :       0
 experiment Id  :       1
 ensemble member:      -1
```

```
 ensemble size  :      -1
 lb             :     1     1     1     1
 ub             :   210   230    50     1
 size           :     9708300
 t               gg(: min,max=)  211.      301.      11   2  1: 50
 u               gg(: min,max=) -12.7      18.1      33   2  1: 50
 v               gg(: min,max=) -26.3      30.6      34   2  1: 50
 w               gg(: min,max=)-0.609     0.557      40   2  1: 51
 size =     9708300

read_atm_grid: p_readgrib =   0


read_atm_grid: file       =gff200908070000.ens.001

 derive vertical coordinates grom GDS:
 center      =          78
 sub_center  =         255
 process     =         173
 local_ident =         253
 repres      =         192
 nv          =          82
       1        0.00000000
       2     2000.00000000
       3     4000.00000000
       4     6000.00000000
       5     8000.00000000
       6     9976.13671875
       7    11902.14453125
....
      81        0.99763012
      82        1.00000000
 GME/HRM vertical coordinate

  reading ensemble members           1  to           4

 read_atm_state: reading u
 scatter fields
 read_atm_state: reading v
 scatter fields
 read_atm_state: reading t
 scatter fields
 read_atm_state: reading ps
```

Chapter 23

Data structures

23.1 Model grid

23.2 Atmospheric state

The derived type **type t_atm** holds the atmospheric fields :
reference (poibter) to the grid data,
some meta data (time, ensemble number,...
atmospheric fields (4D-pointer components: x,y,z,diamond),
additional table with references to all fields.

set up state
deallocate state
allocate component
deallocate component
printout
state = state; state = number; ...
state + state; ...

Example code:

```
type (t_grid) , pointer :: grid
type (t_atm)            :: atm

allocate       (grid)
call construct (grid, gridtype = 0,   &
                            nx = 360  )

call construct (atm,      gri d= grid )

call allocate  (atm,'t')
atm% t = 273.15_wp
call print     (atm)
atm = 0._wp
call print     (atm)
```

```
call destruct  (atm)
call destruct  (grid)
```

23.3 Handling of pointer components

As long as `allocatable components` of derived types are not supported by all relevant compilers `pointer components` must be used instead.

Deallocation of pointer components is not done automatically and must be done by the programmer.

In the include file 'tr15581.incf' routines for bookkeeping (especially for cases of function results passed to other subroutines) are defined.

23.4 Low level GRIB interface (MPIfM GRIB library interface)

The 3-dimensional Variational Data Assimilation (3dVar) uses the EMOS compatible GRIB library of the MPIfM.

A derived type is defined for a convenient acess to the parameters of the GRIB routines:

```
type t_grib1        ! GRIB record data type
     t_sec0         ! component: Section0
     t_sec1         !            Section1, product definition.
     t_s1_dwd       !            Section1, DWD specific section.
     t_rsec2        !            Section2, real values in grid def.
     t_s2_gauss     !            Section2, Gaussian grid.
     t_s2_latlon    !            Section2, lat/lon grid.
     t_s2_tri       !            Section2, triangular grid.
     t_s2_sph       !            Section2, spherical harmonics.
     t_sec3         !            Section3, bitmap (integer field).
     t_rsec3        !            Section3, bitmap (real field).
     t_sec4         !            Section4, binary data
```

Fortran 90 interface to a subset of the EMOS GRIB and BPIO library:

- Open a GRIB file
- Close a GRIB file
- Read next GRIB product
- Write a block of bytes
- Seeks to a specified location
- Encode/Decode GRIB record
- Set DWD record headers (or raw mode)

23.5 Medium level GRIB interface (GDS setup, Inventory)

data type definition: t_inventory and components:
type t_inventory
routines to derive the inventory from a GRIB file:
derive the inventory of a GRIB file.
printout of the inventory.
write a GRIB file :
open a file, set default grib sections
encode a record and write
close the file
prepare a GRIB record for output :
specify default values
specify dwd local part of PDB
specify dwd local PDB for ensemble prediction
specify center, subcenter, process
specify dwd local part of PDB
specify dwd local PDB for ensemble prediction
specify defaults for ECMWF local extension
specify ECMWF local extension for ensemble fc.
specify reference time
specify verification time
specify leveltype, vertical coordinate params.
specify lat/lon grid
specify Gaussian grid
specify triangular grid
specify code, table, bits
specify level
transfer data
example code:

```
type (t_inventory) :: inv

call get_inventory   (inv, file='gribfilename')
call print_inventory (inv)
```

23.6 High level GRIB interface (interface to dace derived types)

read atmospheric state.
read grid information.
write atmospheric state.
write grid (constant data).
write variable.

example code:

```
type (t_grid) , pointer :: grid
type (t_atm)            :: atm

allocate       (grid)
call read      (grid, file='gribfilename')
call read      (atm,  file='gribfilename', fielts='ps u v t')
call print     (atm)
```

Chapter 24

Parallelization Strategy

Parallelization of the code is based in MPI (Message Passing Interface). All data has to be distributed over the processors.

24.1 Model grid

Atmospheric states (background, analysis), discretized on a model grid, are stored in a data type `t_atm`, defined in module `mo_atm_state`. Information on the grid is hold in data type `t_grid`, defined in module `mo_atm_grid`. Atmospheric 3-dimensional fields are stored in 4-dimensional array components (for instance `atm% t(:,:,:,:)` for temperature), in the same way as in the forecast model GME. The first 2 indices correspond to horizontal position, the third to vertical position, and the fourth index denotes the diamond. Regular global longitude-latitude or Gaussian grids are stored using only the first 3 indices.

The operations applied on this data types in this analysis scheme do not rely on neighborhood relationships, because the analysis is performed in the space of the observations. In a parallel environment, the grid-points are distributed over the processor using a domain decomposition approach. The ranges of the first and second index (horizontal domain) are divided into `nproc1` and `nproc2` (defined in namelist `/RUN/`) subranges, respectively; The data is distributed over `nproc1`×`nproc2` processors.

24.2 Observation operators

For certain operations (e.g. for setting up the forecast error covariance matrix and for the post-multiplication step) basic information on the observation operators must be present on several or on all processors. Therefore the basic information is replicated on all processors. More expensive (in terms of CPU-time and memory) operations (for instance evaluation of nonlinear operators) are performed only on dedicated processors. Up to now, these are the processors which hold the block-diagonal of the respective background error covariance matrix (cf. Section 24.3.1).

The observation operators H consist of a pure interpolation operator H_i of the atmospheric fields to the location of the observation and the operator H_o which calculates the observed quantities. In order to evaluate H_i, the model columns involved must be present

on the respective processor. A data type `t_col` is defined in module `mo_t_col` to hold the model columns, together with routines to select and distribute the columns.

24.3 PSAS

24.3.1 Block decomposition of the matrix B + R

Figure 24.1: Distribution of the matrix blocks over processors for the test case described in Section 27. The processor number for each row is given in the column at the left, the representation of each matrix block is shown in the center panel, and the percentage of non-zero elements is shown at the right.

Solving equation (**??**) involves evaluation of matrix-vector products with the matrix $\mathbf{B}+\mathbf{R}$. The matrix is subdivided into blocks of approximate size 500×500, so that the block-

diagonals may be explicitely inverted for pre-conditioning. The blocks are distributed over the processors in a parallel environment. In order to facilitate the evaluation of the matrix-vector product required for the solution of (**??**), the rows of the block-matrix are distributed over the processor as indicated in Figure 24.1. Vectors and matrices are represented in variables of type `t_dec_vector` and `t_dec_matrix` defined in module `mo_dec_matrix` together with the algebraic operators (*,+,...), which take care about the communication between processors. Matrix blocks are stored in compressed storage format if they are sufficiently sparse. The representation type and percentage of nonzero elements for each matrix block is given in Figure 24.1 as well.

24.3.2 Post multiplication

The post multiplication step consists of evaluating the product of the Matrix $\mathbf{P}_b\mathbf{H}^T$ with a vector. Because the matrix is used only once, its elements are not stored but calculated on the fly. The rows of the matrix (associated with the analysis increment on the model grid) are distributed over the processors.

Chapter 25

Sequence of operations

Rework this Chapter to make clear that it is specific to the 3DVAR computations!

Missing: Chapter for the LETKF-Implementation!

Initialization

program var3d:

At first, the Message Passing Interface (MPI) is initialized by calling subroutine `open_nml` from module `mo_mpi`. All the communication routines are encapsulated within this module. The communication routines may be called in a parallel run as well as in a single processor run. In the latter case, they return without having any effect.

Job control input is read from a file `'namelist'` (cf. Section 16) in the working directory. On the processor dedicated to I/O processing `(p_pe==p_io)`, the file is opened by calling subroutine `open_nml` defined in module `mo_namelist`. The main purpose of this module is to provide routines to position the input file at the beginning of a specified namelist-group, so that namelist-groups may appear in arbitrary order. It can be determined if namelist groups are missing or occurring more than once.

Subroutine `nml_run_flags` defined in module `mo_run_params` is called to read the namelist group `/RUN/`. This group specifies analysis time, file names, and the number of processors to be used.

Reading the background state

The routine `stop_time` from module `mo_cpu_time` is called here and wherever the cpu-time and wall-time required for a part of the program shall be determined. The times are written to the standard output file.

The gridded model input and output is encoded in `GRIB` format. Some information to be encoded in the output files (identification number of the weather center, process identifier) is set by calling subroutine `set_defaults` defined in module `mo_grib_handling`.

Finally the background state is read by calling subroutine `read_background`.

Gridded model data (3-dimensional atmospheric fields as well as 2-dimensional surface fields and meta-data) is stored in variable `bg` of derived data type `t_atm`, defined in module

mo_atm_state. In a parallel environment, the model state is stored distributed over the processor elements.

Reading the observations

Reading the observations is handled by subroutine prepare_obs (currently defined in module mo_3dvar called next.

subroutine prepare_obs (module mo_3dvar):

First the blacklist is read by calling subroutine read_blacklists from module mo_blacklist. The blacklist identifies the observations which are known to be unreliable. These observations are rejected so that they need not enter the assimilation system.

Now the observations are read by calling process_obs(TSK_INIT+TSK_READ,obs,bg) defined in module mo_obs. Observations are currently read only on one processor. The observations are stored in the variable obs of data type t_obs (defined in module mo_t_obs). Most operations on this data type are handled by this subroutine. As the first parameter a flag is passed, indicating the task to be performed. In this case, TSK_INIT triggers the initialization of the modules concerned with observation handling. The flag TSK_READ triggers reading of the observations.

back in program var3d:

Spreading the observations over processors and boxes

The raw observations have been read on one processor only. They have to be distributed over analysis boxes (used for pre-conditioning) and the analysis boxes themselves are distributed over the processors. By calling process_obs (TSK_SETUP_DIMS,...) meta-information required for distributing the observations is gathered, for instance the number and type of the observations to be processed.

Now execution branches into the Physical Space Assimilation System (PSAS) specific routine setup_psas.

subroutine setup_psas (module mo_psas):

At the start of this subroutine the namelists /PSAS/ and /VARQC/ are read, specifying PSAS specific parameters (e.g. convergence criteria) and parameters for the variational quality control procedure.

On the processor which has read the observations, the association of the observations with analysis boxes and processors is established. First the horizontal distance between each pair of observation points is determined. Based on this information, subroutine set_boxes decides on the association with boxes and processors: Starting from an arbitrary observation, the closest neighbors are assembled into a box, until 500 observations are reached. This procedure is repeated with a new starting point at the boundary of the box. (The distribution of observations, boxes, and processors is written to a GRADS-file spots.dat.) Finally the observations are broadcasted. So the basic information on the observations is present on all processor elements. Merely the more expensive operations – for instance the evaluation of nonlinear observation operators – are performed on dedicated processors. Observations are stored in variable obs, which is an array of a derived data type t_obs. Each array element corresponds to one observation box, containing all

observations associated with it.

Scatter model columns

In order to interpolate the atmospheric state to the location of the observations (or to the location required by the observation operators) the model columns involved must be assembled on the processor holding the respective observations. They are determined by calling `process_obs(TSK_SETUP_COLS,obs(i),atmf)` for each observation box `obs(i)` associated with the actual processor element. The model columns are send to the respective boxes and processor elements. They are stored in an array `cbgb` of data type `t_cols`. Again, each element of `cbgb` corresponds to an observation box.

Interpolation to observation points

The call to `process_obs(TSK_SETUP_FULL,obs,atmf)` allocates space and determines the levels to interpolate to. The interpolated model values are stored in variable `fi` of type type `t_vector`. This data type holds vectors those elements are distributed over 'segments', and segments distributed over processor elements. This data type is intended to be used in algebraic operations involving large matrices and vectors. The 'segments' of the vector here correspond to the observation boxes.

Set background error covariances

A number of variables holding matrices and vectors are allocated. Error covariance and pre-conditioning matrices are stored in variables of type `t_matrix`. Internally these matrices are partitioned in blocks; The blocks are distributed over processor element. In dependence of the number of nonzero elements, each block is stored in full or in sparse representation. Algebraic operations $(+.-, *, /, \dots)$ are defined on variables of type `t_matrix` and type `t_vector`, hiding implementation details and communication issues.

The background error covariance matrix $\mathbf{H}\mathbf{P}_b\mathbf{H}^T$ in the space of observed quantities can be written as $\mathbf{H}_O(\mathbf{H}_I\mathbf{P}_b\mathbf{H}_I^T)\mathbf{H}_O^T$, with the operator $\mathbf{H}$, consisting of the interpolation operator to the observation points $\mathbf{H}_I$ and the observation operator $\mathbf{H}_O$.

The observation operator Jacobi covariance matrices $\mathbf{H}_O$ are calculated by calling `process_obs(TSK_K,obs,atmf,xi=fi)`. The elements of the matrix $(\mathbf{H}_I\mathbf{P}_b\mathbf{H}_I^T)$ are calculated explicitely as a function of the location and type of the observations involved by subroutine `set_Pb(P_b,e_f,e_o,obs,cbgb,er,atmf%time)`. The observation error covariance matrix $\mathbf{R}$ is calculated by calling `set_R(R,e_o,obs)`. In addition to the error covariance matrices `P_b` and `R`, the errors (square root of the diagonal elements) are stored in the vectors `e_f` and `e_o`. The pre-conditioning matrix `P_R = P_b+R,e` (diagonal elements only) as well as the inverse `R` and the approximate inverse of `P_R` (Cholewsky decomposition of the block diagonals) are calculated as well. Optionally the matrices `P_b`, `R`, and `P_R` are written to a file in sparse format.

The observations are stored in a variable of type `t_vector` by subroutine `get_obs`. The corresponding model equivalents `f` are calculated from the model variables `fi` interpolated to the locations of observations by subroutine `apply_H`.

Solve the system of equations in observation space

The details of the solution algorithm are given in Section 26. Equation **??** is solved iteratively by a Newton algorithm. At the beginning of each iteration, the values of some flags (their value may be set individually set for each iteration by namelist /PSAS/) are printed.

In the first iteration, the current guess is either set to the background value, or to a value close to the observations, depending on the flags `fg_b` and `fg_o` respectively.

Line search (subroutine `lnsrch`)

The Newton algorithm does not ensure that the value of the cost function decreases in each iteration, because the step size and direction is based on the approximation of the cost function in the vicinity of the current guess. Thus a line search is applied (call to subroutine `lnsrch`) in order to guarantee that the cost function decreases.

Because evaluation of the nonlinear terms is expensive, the line search is designed to evaluate them only a few times, and thus is not very exact. In general, the nonlinear terms are calculated only once or twice. The nonlinear quantities are the cost function, its gradient, as well as the observation error covariance matrix (which depends on the current guess if variational quality control is enabled) and the background error covariance matrix (if nonlinear observation operators are present). They are evaluated in function `Rfunc`. For the case that no line search is performed (i.e. in the first iteration), instead of `lnsrch` the routine `Rfunc` is called directly.

Finally the the pre-conditioning matrix (inverse of the block-diagonals of $\mathbf{B}+\mathbf{R}$) and the inverse of the observation error covariance matrix (required for diagnostics only) are recalculated before calculating the search direction and step size is determined by the Conjugent Gradient algorithm (subroutine `nlpcg`). For both, the outer Newton algorithm and the inner CG algorithm, parameters (for instance convergence criteria) can be set in namelist `/PSAS/` (Section 16.16).

Inner loop (CG algorithm)

The preconditioned Conjugate Gradient algorithm is fairly standard and taken from a textbook ([20], p. 118). A number of quantities is monitored in the standard output and in file `nlpcg.info` (Section 20.1).

Estimate the analysis error

The analysis error is estimated by subroutine `ana_err`. Currently the errors are estimated on 7 pressure levels (1000, 500, 300, 200, 100, 50, 10) on a $6^o \times 6^o$ grid for the geopotential and the two wind components. This estimate is written to a GRIB file and used in the subsequent analysis cycle to estimate the background error. The expense of the algorithm as implemented here does not depend on the number of observations or the size of the analysis grid. It can be switched off by the flag `anaerr` in namelist `/PSAS/`.

Post multiplication

The effort of the post-multiplication step grows linearly with the number of observations times the number of model prognostic variables. This step is quite expensive and cpu-time may be saved if the analysis increment is calculated on a coarser resolution, then interpolated on the finer model grid, and finally added to the first guess. In this case a coarser model reference state (variable `atma`) is derived from the full resolution state.

The post multiplication basically consists of a matrix-vector multiplication, involving the large matrix $\mathbf{P}_b\mathbf{H}^T$. This operation is parallelized by assigning the rows of the matrix to certain processors. The right hand side (quantity $\mathbf{z}$ in observation space) is hold on each processor whereas the left hand side (analysis increment on the model grid) is distributed over the processor round robin. The relevant information required for updating the left hand side (mainly surface pressure of the background state) is distributed over the processor elements by calling subroutine `setup_post(cbg,atma)`, and stored in variable `cbg` of type `t_cols`.

The post multiplication is performed by subroutine `post_mult`. Basically the matrix is only used once. Thus it is not stored, but the values of its elements are calculated on the fly when the multiplication is performed. Because the analysis grid depends on the surface pressure of the analyzed state itself, the post multiplication is performed in two steps. First analysis increments for the surface pressure are derived and added to the background value to obtain the new surface pressure. Then the pressure levels of the analysis grid are calculated, and the analysis increments for the remaining variables are calculated on this grid.

The result of the post multiplication (the analysis increments were obtained column by column in variable `ani` of type `t_cols`. They are send back to the processor holding the respective part of the grid by calling subroutine `finish_post(ana,ani,atma)` and finally stored in variable `ana` of type `t_atm`. If the analysis increments were derived on a coarser grid, they are now interpolated to the full resolution grid by subroutine `interpolate_atm`.

Add analysis increment to forecast

The surface pressure of the analyzed state differs from that of the background. Consequently, the vertical grid of the analysis differs from that of the background as well, as the pressure levels depend on the surface pressure as well. In order to account for the change of vertical coordinates, the analysis is performed as follows:

1. The atmospheric state `atma` is represented in terms of virtual temperature and relative humidity. For these quantities the analysis increments were derived.

2. A preliminary analysis is performed by adding the analysis increment (evaluated at the levels of the background grid) to the background state. The pressure increment is added to the background surface pressure without changing the height of the atmospheric levels. (The resulting state is stored in variable `ana`.

3. The pressure levels of the analysis grid are derived, taking into account the analyzed surface pressure.

4. The preliminary analysis is vertically interpolated to the analysis grid, resulting in the state `atmtmp`. Interpolation is performed by subroutine `vert_intp`, consistently with the interpolation algorithm used to interpolate to the locations of the observations.

5. Finally temperature and specific humidity are derived from virtual temperature and relative humidity on the analysis grid.

As an option, the procedure performed in the preceding Optimum Interpolation code is present: deriving the analysis increment at the levels of the analysis grid and adding them to the background values (taken from the background grid) without further interpolation.

For diagnostics, both the background `atma` and the analysis `atmtmp` may be vertically interpolated to standard pressure levels (`fg_p` and `ani_p`) and the difference written to a GRIB file. The difference of the analysis and the background (defined on the respective pressure levels) may be written as well.

Interpolate analysis to locations of observations

In order to check for the consistency of the interpolation (from background to observation points) and the projection of the increments from observation space to the model grid (by the post-multiplication step) the final analysis is interpolated to the locations of the observations by the same algorithm which has been used for the background. The resulting analysis of the observed quantities may be compared to their values estimated in observation space ($\mathbf{Bz} + H(\mathbf{x}_b)$).

Output

The analysis (and optionally the analysis increment on pressure and model levels) is written to the respective `GRIB` files. The analysis at the location of the observation is written to an ASCII file (`psas.info`).

End of program

back in program var3d

Finally all allocated memory is explicitely deallocated. This is mainly done for debug purposes so that memory leaks can be easily detected. The program is stopped in a controlled way on all processors by calling subroutine `p_stop` from module `mo_mpi`.

subroutine setup_psas (module mo_psas)

Chapter 26

Solution of the set of equations in physical space

The set of equations (**??**) is solved in subroutines `setup_psas` (outer loop, module `mo_psas`), `nlpcg` and `lnsrch` (linear CG algorithm and line search, module `mo_nlpcg`).

We distinguish the following vector spaces:

X, the space of model control variables (temperature, humidity, wind components), interpolated to the location of the observation. Within this chapter we do not deal with the original control variables located on the model grid.

Y, the space of observed quantities. These quantities are related to the control variables in X by the nonlinear observation operator: $\mathbf{y} = H(\mathbf{x})$.

U_i, the space of observed quantities, related to the control variables in X by the observation operator linearized at the guess within the ith iteration: $\mathbf{w} - \mathbf{w}_b = (H_i\mathbf{x} - \mathbf{x}_b)$.

Z, the dual space, related to U_i by $\mathbf{w} - \mathbf{w}_b = \mathbf{H}_i\mathbf{P}_b\mathbf{H}_i^T\mathbf{z}$ and to X by $\mathbf{x} - \mathbf{x}_b = \mathbf{P}_b\mathbf{H}_i^T\mathbf{z}$.

Within subroutine `setup_psas` the following variables are defined:

Symbol	Fortran -name	Description
		Matrices
$\mathbf{H}_i$	`Hi`	Jacobi matrix of the observation operator H linearized at at $\mathbf{x}_i$, with $\mathbf{y}_i = H(\mathbf{x}_i)$. (*Currently* $\mathbf{H}_i$ *is stored within the derived data type* `obs`.)
$\mathbf{R}_j$	`Rj`	Observational error covariance matrix in Y. If variational quality control is enabled, `R` is the inverse of the Hessian at the location $\mathbf{y}_j$. In general $\mathbf{y}_j = \mathbf{y}_i$, i.e. the observational cost function and its derivatives are evaluated at the same location there the observation operator was linearized.
$\mathbf{H}_i\mathbf{P}_b\mathbf{H}_i^T$	`HBHi`	Background (forecast) error covariance matrix in U_i.

$\widehat{\mathbf{H}_i\mathbf{P}_b\mathbf{H}_i^T}+\mathbf{R}_j$	HBH_R	Preconditioning matrix in U_i. $\widehat{\mathbf{H}_i\mathbf{P}_b\mathbf{H}_i^T}$ denotes the block-diagonal of $\mathbf{H}_i\mathbf{P}_b\mathbf{H}_i^T$.
$(\widehat{\mathbf{H}_i\mathbf{P}_b\mathbf{H}_i^T}+\mathbf{R}_j)^{-1}$	HBH_R_inv	Inverse preconditioning matrix in U_i.
$\mathbf{R}_j^{-1}$	R_inv	Inverse observational error covariance matrix $\mathbf{R}_j$.
$\mathbf{S}^{-1}$	S_inv	Inverse scaling matrix, i.e. inverse of the diagonal of the preconditioning matrix. The inverse preconditioning matrix is scaled by $\mathbf{S}^{-1}$.
		Vectors in observation space Y
$\mathbf{o}$	o	Observations
$\mathbf{y}_b$	yb	Background value $\mathbf{y}_b = H(\mathbf{x}_b)$.
$\mathbf{o}-\mathbf{y}_b$	o_yb	Observation minus background (forecast).
$\mathbf{y}-\mathbf{o}$	y_o	Guess minus observation.
$\mathbf{y}_i$	yi	Model aequivalents of observations $\mathbf{y}_i = H(\mathbf{x}_i)$, evaluated at the location $\mathbf{x}_i$ there the observation operator was linearized.
$\partial J_o/\partial\mathbf{y}$	dJody	Gradient of the observational cost function with respect to $\mathbf{y}$. This quantity is calculated by the variational quality control routine.
$\mathbf{w}_{qc}$	w_qc	Weight assigned to the observations by the variational quality control routine.
$\mathbf{y}_a-\mathbf{y}_b$	ya_yb	Final analysis minus forecast.
$\mathbf{e}_f$	e_f	Forecast error (standard deviation) for the observed quantities.
$\mathbf{e}_o$	e_o	Observational error (standard deviation).
$\partial J_{j-1}/\partial\mathbf{z}$	dJold	Gradient of the cost function in the previous iteration.
		Vectors in dual space Z
$\mathbf{z}_j$	z	Solution of the set of linear equations (**??**).
$\mathbf{z}_{j-1}$	zold	Previous solution of the set of linear equations (**??**).
$\partial J_j/\partial\mathbf{z}$	dJdz	Gradient of the cost function with respect to $\mathbf{z}$. This quantity is calculated by the variational quality control routine.
		Vectors in linearized observation space U_i
$\mathbf{u}_i-\mathbf{u}_b$	ui_ub	Model aequivalent of observation minus background $\mathbf{u}_i-\mathbf{u}_b = \mathbf{H}_i(\mathbf{x}_i-\mathbf{x}_b)$, evaluated at the location there H was linearized.
$\mathbf{u}-\mathbf{u}_b$	u_ub	Current guess (analysis) minus background.
$\mathbf{u}_r$	ur	Residuum, right hand side of equation (**??**).

$\mathbf{u}_j - \mathbf{u}_b$	`uj_ub`	Model aequivalent of observation minus background, evaluated at the location there the observational cost function J_o was evaluated and linearized ($\mathbf{R}_j, \partial J/\partial \mathbf{y}$).
		Vectors in interpolated model space X
$\mathbf{x}_b$	`xb`	Forecast (background) interpolated to the locations of observations.
$\mathbf{x}_i - \mathbf{x}_b$	`xi_xb`	Location of linearisation of H minus background.
$\mathbf{x} - \mathbf{x}_b$	`x_xb`	Current guess (analysis) minus background.
		Scalars

The following Table traces the locations in the program flow, there these variables are used (i), used and modified (io), or overwritten (o). This corresponds to the intend-attributes (in,out,inout) of the respective Fortran 90-routines[1].

[1] arguments of derived data type may require the `intend(inout)` attribute even if the values of the variable are merely overwritten if the routine requires access to the meta-data (dimensions, etc.) of the data type.

constants			observation operator					observational cost function							preconditioner				current guess					saved	
yb	o	o_yb	xi_xb	ui_ub	yi	Hi	HBHi	Rj	J_qc	dJdz	dJody	uj_ub	w_qc	HBH_R	R_inv	S_inv	HBH_R_inv	ur	z	u_ub	x_xb	y_o	zold	dJold
Initial set up																								
o	o	o	o	o	o	o	o	o	o	.	o	o	.	o	o	o	o	o	o	o	o	.	.	
Iterative Newton search																								
Line search																								
.	.	.	io?	io?	io?	.	.	o	o	o	o	.	o	.	.	.	.	.	io	o	.	.	i	
Recalculate **x**																								
.	.	.	i	i	.	.	i	.	.	.	.	.	.	.	.	.	.	.	i	o	o	o	.	
Rfunc																								
.	.	.	.	.	.	.	.	o	o	o	o	.	o	.	.	.	.	.	i	i	.	i	.	
Recalculate $\mathbf{H}_i$																								
.	.	.	o	o	o	o	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	i	.	
Recalculate $\mathbf{H}_i\mathbf{P}_b\mathbf{H}_i^T$																								
.	.	.	.	.	.	i	o	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	
.	.	.	.	.	.	.	i	.	.	.	.	.	.	.	.	.	.	.	i	o	.	.	.	
Recalculate $\mathbf{R}_j$ (Rfunc)																								
.	i	.	.	i	i	.	.	o	.	o	o	.	o	.	.	.	.	.	i	i	o	.	.	
Save old values																								
.	.	.	.	.	.	.	.	.	.	i	.	.	.	.	.	.	.	.	i	.	.	.	o	
ζ^{-1}																								
.	.	.	.	.	.	.	.	i	.	.	.	.	.	.	o	.	.	.	.	.	.	.	.	
hs																								
.	.	.	.	.	.	.	.	i	.	.	i	o	.	.	.	.	.	o	.	i	.	.	.	
without VQC																								
.	o	.	.	.	.	.	.	.	o	.	o	o	.	.	.	.	.	o	.	i	i	.	.	
Preconditioner																								
.	.	.	.	.	.	.	i	i	.	.	.	.	.	o	.	o	o	.	.	.	.	.	.	
Linear (CG) solver																								
.	.	.	.	.	.	.	i	i	(i)	.	(i)	(i)	.	.	(i)	.	i	i	io	o	.	.	.	

Initial set up

Before entering the main loop the variables are set to the following initial values:

`xb, yb, o, o_yb` (constants):
The forecasted and observed quantites (in interpolation space U and observation space Y) are set to their respective values which remain unchanged furtheron.

`xi_xb, ui_ub, yi, Hi, HBHi` (observation operator):
The observation operator H is evaluated and linearised at the background state(`xi`=`xb`). The reference values (`xi_xb, ui_ub, yi`) in interpolation and observation space, the Jakoby-matrix (`Hi`), as well as the forecast error covariance matrix in observation space (`HBHi`) are set accordingly.

`Rj, J_qc, dJody, uj_ub` (observational cost function):
Initally, the observational cost function is evaluated with variational quality control

switched off. This means that `Ri`, the inverse Hessian of the observational cost function, is equal to the nominal observational error covariance matrix $\mathbf{R}$. The observational cost function is quadratic and equal to its quadratic approximation everywhere. The reference point for this approximation is chosen arbitrarily at the value of the observations, leading to the values of `J_qc`$= 0$ (cost function), `dJody`$= 0$ (gradient). The reference value `uj_ub` is set to $o - u_b$. (*For nonlinear observation operators this is a bad approximation. However, up to now, this quantity is used for diagnostics only within the CG routine.*

`HBH_R, R_inv, S_inv, HBH_R_inv` (preconditioner):
The preconditioning matrix `HBH_R` is calculated as the block diagonal of `HBHi`+`Rj`. Its inverse `HBH_R_inv` is scaled by the diagonal of `HBHi`+Rj, whose inverse is kept in `S_inv`.

`z, u_ub, x_xb, ur` (current guess):
The first guess is set to the background value. Consequently `z,u_ub,x_xb`$= 0$ and `ur`=`o_yb`.

Optionally, some of the quantities set so far (at the background state) may be written for diagnostic purposes (for instance the matrices `HBHi` and `Rj` in a sparse representation).

Iterative Newton search

The set of equations is solved iteratively. Within each iterations the following steps are performed:

1. Perform a line search.
   To ensure covergence, the increment is determined by a line search along the direction of the increment given by the estimate of the CG routine from the (linearized) set of wquations in the previous iterate.

2. Update quantities.
   For the new estimate, a number of quantities is updated (e.g. linearisation of H and J_o.)

3. Derive new estimate of the linearized equations.
   A new estimate is sought by the conjugate gradient algorithm.

These steps are described in more detail in the subsequent sections:

Line Search (`subroutine lnsrch`)

In each iteration i the conjugate gradient algorithm provides an estimate $\mathbf{z}_{\mathtt{lin}}$ of the location of the cost function $J(\mathbf{z})$, assuming a linear relationships $\mathbf{y} - \mathbf{y}_i = \mathbf{H}(\mathbf{x} - \mathbf{x}_i)$ and a quadratic observational cost function $J_o(\mathbf{y})$. (The relationship $\mathbf{y} - \mathbf{y}_b = \mathbf{H}_i \mathbf{P}_b \mathbf{H}_i^T \mathbf{z}$ is linear by definition.) These assumptions do not hold and in order to account for the nonlinearities the linear estimate is iterated in an outer loop. In order to ensure

convergence a line seach is performed along a search direction $\mathbf{p} = \mathbf{z}_{\texttt{lin}} - \mathbf{z}_{\texttt{old}}$ there $\mathbf{z}_{\texttt{old}}$ is the estimate in the previous iteration.

The line search algorithm updates its argument `z`, i.e. the estimate of the location of the minimum of the cost function. The (linearily) related quantities `u_ub` and `x_xb` are updated as well.

On entry, the high dimensional problem is reduced to a 1-dimensional problem by calculating $\mathbf{p}$ and the related search direction $\mathbf{p}_u = \mathbf{H}_i \mathbf{P}_b \mathbf{H}_i^T \mathbf{p}$ in U. The corresponding direction $\mathbf{p}_x$ in X is derived as well as described below.

1-dimensional search

With the definition $\mathbf{z} = \mathbf{z}_{\texttt{old}} + \lambda \mathbf{p}$ the approximate minimum of $J(\lambda)$ is sought. The first estimate is chosen as $\lambda_1 = 1.$ (i.e. the step size proposed by the CG algorithm, which may be changed by the subroutine argument `lambda1`). The second estimate is chosen by assuming a quadratic approximation of $J(\lambda)$, given the 3 pieces of information: $J(\lambda_0)$, $\partial J / \partial \lambda|_{\lambda_0}$ and $J(\lambda_1)$. Subsequent estimates assume a quadratic approximation as well, derived from the 3 modt recent evaluations of the cost function J_k, J_{k-1}, J_{k-2}.

Because the evaluation of the cost function J in general will be expensive, only a limited number of trials is performed until the condition $J(\lambda_k) < J(\lambda_0 = 0)$ is fullfilled. If the argument `exac` is set to `.true.` at least 2 estimates are tried.

Evaluation of the cost function

In order to derive the cost function for different values of Vz, (i) the respective values of $\mathbf{u}$, $\mathbf{x}$ and $\mathbf{y}$, and (ii) the cost function and its gradient have to be calculated:

Update of the guess x in interpolation space

For a given linearization (`xi_bg, ui_bg, Hi`) of the observation operator H for each guess of `z` the corresponding values of `u_ub`, `x_xb` and `y` have to be calculated as a prerequisite for calculating the cost function. The following steps are performed by `subroutine set_new_x`:

1. `u_ub`
   is calculated by `u_ub = HBHi z`.

2. `x_xb`
   Calculating of `x_xb` in general is expensive. For that reason a strategy for deriving an approximate value is followed if the flag `new_x` is set to 1. Calculation of `x_xb` without approximation is enabled by setting `x_xb` to 2.

   Approximate solution `new_x = 1`:
   `x_xb` is calculated by $\mathbf{x} - \mathbf{x}_b = \mathbf{H}^{-1}(\mathbf{u} - \mathbf{u}_b)$. In general $\mathbf{H}$ may be not invertable. In this case the generalised inverse is used. This method runs the risk that the inversion of the matrix may be ill posed and the result may become inaccurete. To improve accuracy the linearization values `xi_xb`,`ui_ub` may be used to apply the inversion only to the deviations from these coordinates. Another problem may occur because $\mathbf{H}$ in general is not square and there may be unobserved components of `x_xb` which will be unaltered by this procedure.

Exact solution `new_x = 2`:

This option is currently not implemented.

The exact solution is obtained by $\mathbf{x}-\mathbf{x}_b = \mathbf{P}_b\mathbf{H}_i^T\mathbf{z}$. The matrix $\mathbf{P}_b\mathbf{H}_i^T$ currently is not kept and must be recalculated for this purpose.

3. `y`
   Filally `y` is calculated by $\mathbf{y} = H(\mathbf{x})$.

Evaluation of the cost function and its gradient

The cost function and its derivatives are calculated by subroutine `Rfunc`. Given $\mathbf{z}$ and $u - u_b$, the background cost function is easily calculated as $J_b = (u - u_b)(\mathbf{H}_i\mathbf{P}_b\mathbf{H}_i^T)\mathbf{z}$. The observational cost function is calculated for given $\mathbf{y} - \mathbf{o}$ by the variational quality control routine `new_R`. This routine also returns information on the first (`dJody`) and second (`R`) derivative of the cost function. *These quantities are given in Y and as long as $\mathbf{y}_i \neq \mathbf{y}_j$ it is inexact to apply them in U:)* The final gradient of $J_b + J_o$ is given as `dJody + z`.

Update quantities

Recalculate $\mathbf{H}_i$

Recalculate $\mathbf{H}_i\mathbf{P}_b\mathbf{H}_i^T$

Recalculate R_j

Recalculate R

Update rhs

Recalculate the Preconditioning matrix

Linear (CG) solver

Chapter 27

Performance

This test case is based on approximately 20 000 observations (Table 27.1, consisting of the SYNOP and TEMP observations present at 0 UTC for an arbitrary day. The TEMP observations were used on the main pressure levels only. The times required for different parts of the program is shown in Table 27.2. The analysis was performed on one node (16 processors) of the IBM SP2 (cos5).

The operational constraint of 10 minutes time is approximately met by this test case. 70 000 observations (the approximate number of the operational OI scheme) can be processed by 8 nodes within the time constraint.

CPU time is spent to approximately equal parts for performing I/O, for solving the set of equations in observation space, and for the post-multiplication step. Note that performing I/O is not parallelized yet, and that a certain amount of the diagnostics written is not required for an operational analysis. For the first step of the iterative solver, CPU times required for parts of that task are explicitely given. Approximately the same amount of time is spent for the Conjugent Gradient solver and for the evaluation of the nonlinear terms (here variational quality control). In total, 20 iterations were performed.

observation type	accepted	rejected
SYNOP surface pressure	3230	233
SYNOP wind component u	67	1
SYNOP wind component v	67	1
TEMP geopotential	4355	1193
TEMP relative humidity	3330	955
TEMP wind component u	4854	317
TEMP wind component v	4854	317
total	20757	3017

Table 27.1: Number of observations processed in this test case. The columns denoted 'accepted' and 'rejected' give the number of observations with a weight of $w_{vqc} > 0$ and $w_{vqc} < 0$, respectively, assigned by the variational quality control routine.

cpu(min)	(max)	(mean)	wall	percent	task
5.88	6.71	6.64	6.69	99.29	read inventory
0.68	0.74	0.71	0.77	92.86	read grid
17.44	98.18	23.39	100.78	23.21	read background and observations
11.41	12.41	12.32	12.41	99.24	set up PSAS boxes
1.05	2.87	2.01	2.87	69.93	broadcast observations to PEs
0.77	0.84	0.81	0.82	98.55	scatter model columns
0.31	0.34	0.33	0.35	94.11	interpolate
5.57	5.66	5.58	5.65	98.81	run observation operators (for H)
2.66	9.57	5.09	9.71	52.41	set background error correlations
0.03	0.10	0.08	0.16	48.83	calculate pre-conditioning matrix
0.24	0.37	0.31	0.36	86.11	calculate inverse pre-conditioning matrix
0.01	0.02	0.02	0.03	60.42	interpolate first guess
					1st iteration:
3.13	3.20	3.17	3.19	99.24	variational quality control
0.06	0.06	0.06	0.07	85.71	recalculate R^{-1}
0.20	0.32	0.27	0.32	84.37	recalculate pre-conditioner
3.44	3.60	3.59	3.59	99.88	CG solver
225.38	226.35	225.97	226.44	99.79	iterations > 1
10.13	10.29	10.25	10.29	99.57	estimate analysis error
0.12	0.25	0.19	0.24	77.87	distribute model columns for post mult.
195.31	199.83	196.81	200.14	98.33	post multiplication
0.87	1.63	1.29	1.65	78.07	gather results on model grid
10.82	10.90	10.88	10.91	99.74	add (interpolate) increment
50.85	51.91	51.76	51.89	99.75	write GRIB-file
0.03	1.16	0.10	1.32	7.62	write psas.info file
644.67	649.17	648.15	650.87	99.58	Total

Table 27.2: CPU- and wall time used for different parts of the program (20 000 observations, run on 16 processors). The first 3 columns show the minimum, maximum, and mean CPU-time (in seconds) spend on any processor. The forth and fifth column shows wall-time (seconds) and relation of mean CPU-time to wall-time (%), respectively. Values larger than 100% of the latter are due to the limited precision of the timing functions. The times are stopped by the Fortran 90 intrinsic routines `cpu_time` and `system_clock`, respectively.

Chapter 28

Source Code Reference

28.1 Observations

28.1.1 Observation data type – Module mo_t_obs

28.1.2 Observation operator – Module mo_obs

28.1.3 SYNOP observation operator – Module mo_synop

28.1.4 TEMP observation operator – Module mo_temp

28.1.5 TOVS observation operator – Module mo_tovs

28.1.6 GPS Radio Occultation operator – Module mo_occ

This module contains the routines specific to the GPS radio occultation operator. The routines make use of the data type definitions of module `mo_t_obs`. In subroutine `process_occ` the following actions are performed, triggered by the values of the variable `tsk`:

Module initialisation (TSK_INIT)

At first, the namelist group `GPS_RO` is read (subroutine `read_occ_nml`). The geoid anomalies are read (subroutine `set_geoid`) and stored in the grid data structure.

Input and Data selection (TSK_READ)

Currently input files (vda-format) provided by the CT algorithm written by Michael Gorbunov are read, using subroutine `read_occ_infvda` . The path and filenames of the data sources are explicitely specified by namelist `RO_VDA`. Bending angles are provided in high vertical resolution and thus rays at a reasonable spacing are selected according to the parameters provided by namelist `GPS_RO`.

The content of the data file is stored in variables of type `t_occ` and `t_rays` (selected rays only). These variables are kept in the observational data structure (component `par` of variable `obs`, type `t_obs`). Their components very closely reflect the content of the data file:

```
type t_ray                              ! obs for one bending angle
  real(wp)        :: p    = invalid !   impact parameter       [km]
```

```
  real(wp)        :: eps  = invalid !   bending angle          [rad]
  real(wp)        :: var  = invalid !   bending angle variance [rad^2]
  real(wp)        :: var2 = invalid !   variance interpolated-uninterpolated
  type(cartesian) :: rleo           !   LEO position
  type(cartesian) :: vleo           !   LEO speed
  type(cartesian) :: rgps           !   GPS position
  type(cartesian) :: vgps           !   GPS speed
end type t_ray


type t_occ                        ! GNNS occultation data type (ray tracer)
  character(len=64)     :: file   !  Input file name
  integer               :: yyyy   !  Occultation date: year
  integer               :: mo     !  Occultation date: month
  integer               :: dd     !  Occultation date: day
  integer               :: hh     !  Occultation UTC : hour
  integer               :: mi     !  Occultation UTC : minute
  real(wp)              :: sec    !  Occultation UTC : second
  integer               :: occid  !  Occultation ID
  type (geodetic)       :: gp     !  Geodetic latitude ,longitude [deg]
  type (cartesian)      :: xlc    !  Local curvature center (cartesian)
  real(wp)              :: rlc    !  Local curvature radius [km]
  real(wp)              :: xb     !  Back propagation plane position [km]
  integer               :: wf     !  GPS data smoothing window [points]
  integer               :: w      !  BP data smoothing window [points]
  integer               :: wi     !  Ionospheric smoothing window [points
  integer               :: nray    =  0      ! number of rays used
  logical               :: checked = .false. ! rays are all valid
end type t_occ
```

Identify the model columns required (TSK_SETUP_COLS)

The model columns required by the ray-tracing operator are identified by calling subroutine `ECHAM_rays_idx_init` . The indices of the model columns are stored in the components `mcol` and `spot% imcol` of the observation variable `obs` (type `t_obs`).

Set up interpolation space (TSK_SETUP_FULL)

The variables used in the interpolated space (temperature and humidity at model levels at the model columns identified by `obs% spot% imcol`) are marked in `obs% t_int`.

Set up the Jacobi-matrix (TSK_K)

Run the tangent linear operator (TSK_H)

Setup of the radio occultation data structures

derivation of refractivity

Ray tracing

Adjoint calculations

Background error covariances

28.1.7 GNSS slant delay operator – Module STD_operator

Empty: GNSS Slant Total Delay operator: Theory

Signal Delays

The optical path length L is defined by the integral

$$L = \int_S n(s)\, ds \tag{28.1}$$

S beeing the true bended signal path in the atmosphere and $n(s)$ beingt the refraction index along this path. The optical path difference ΔL as defined by

$$\Delta L = \int_S n(s)\, ds - \int_G ds \tag{28.2}$$

is the difference between L and the undisturbed signal propagation in vacuum ($n = 1$). The signal path in vacuum is a straight line G and $\int_G\ ds$ is the geometric distance between transmitter and receiver. After some algebra two contributions to ΔL can be identified:

$$\begin{aligned} \Delta L &= \int_S n(s)\, ds + \int_S ds - \int_S ds - \int_G ds && (28.3)\\ &= \int_S (n(s) - 1)\, ds + \int_S ds - \int_G ds && (28.4)\\ &= \int_S (n(s) - 1)\ ds + (S - G) && (28.5) \end{aligned}$$

The first term $\int_S (n(s) - 1)\ ds$ describes the delay due to the delayd signal propagation in matter ($v < c$) and the second term describes the extra travel time due to the increased path length of the bended signal path (S-G). The latter is referred to as the *geometric delay*. The speed of light enters the equations implicitly as the refraction index $n = \frac{c}{v}$, v beeing the speed of light in matter, c beeing the vacuum speed of light. The term $n - 1$ is usually rather small, especially in gases. Therefore, the refractivity N is defined:

$$N = 10^6 \cdot (n - 1) \tag{28.6}$$

leading to

$$\Delta L = STD = 10^{-6} \cdot \int_S N\, ds + (S - G) \tag{28.7}$$

ΔL is referred to as the *slant total delay* (STD). The term $S - G$ is much smaller than the first term and is often neglected, leading to the well known approximation

$$STD \approx 10^{-6} \cdot \int_S N\, ds \tag{28.8}$$

This approximation is not used by the STD observation operator. At low elevations the STD is about 20 m – 25 m and the geometric delay is about 15 cm – 25 cm, i.e. approx.

1 % of the total delay. In zenith direction the STD (= ZTD) is about 2.4 m and the geometric delay is almost zero.

The STD provided by GNSS data processing is a quantity related to the neutral atmosphere [1], i. e. ionospheric effects were separated before the STD was estimated. As the STD observation operator must provide the same physical quantity the integral 1.7 has to deal only with the subpath S inside the neutral atmosphere. This is not necessarily true for the raytracing. The free charges in the ionosphere have a strong impact on the signal path. i. e. on the point where the signal enters the neutral atmosphere. However, this effect is small and is neglected by the STD operator.

Raytracing

According to Fermat's principle each electromagnetic wave propagates along a path S which minimizes the optical path length:

$$\int_S n(s)\, ds \quad \Rightarrow \quad \text{min.} \tag{28.9}$$

Assuming that the field of the refracion index n is known, this equation can be used to estimate the bended signal path in the atmosphere. The variational problem is best solved in a Cartesian system with the x axis pointing from the receiver to the satellite. The y and z coordinates of the signal path become functions of x which describe the deviations from a straight line:

$$\int_a^b n(x,y,z)\sqrt{1+y'(x)^2+z'(x)^2}\, dx = \int_a^b F(x,y(x),y'(x),z(x),z'(x))\, dx \quad \Rightarrow \quad \text{min.}$$

The variational problem can be translated to a system of partial differential equations using the Euler-Lagrange equations:

$$\begin{aligned} \frac{\partial F}{\partial y} - \frac{d}{dx}\frac{\partial F}{\partial y'} &= 0 \quad \Rightarrow \quad \frac{\partial F}{\partial y} - \frac{\partial^2 F}{\partial x \partial y'} - \frac{\partial^2 F}{\partial y \partial y'} - \frac{\partial^2 F}{\partial y'^2} = 0 \\ \frac{\partial F}{\partial z} - \frac{d}{dx}\frac{\partial F}{\partial z'} &= 0 \quad \Rightarrow \quad \frac{\partial F}{\partial z} - \frac{\partial^2 F}{\partial x \partial z'} - \frac{\partial^2 F}{\partial z \partial z'} - \frac{\partial^2 F}{\partial z'^2} = 0 \end{aligned} \tag{28.10}$$

This leads to two coupled ordinary differential equations for the functions $y(x)$ and $z(x)$

$$\begin{aligned} y'' &= \left(\frac{n_y}{n} - \frac{n_x}{n}y'\right)(1+y'^2+z'^2) \\ z'' &= \left(\frac{n_z}{n} - \frac{n_x}{n}z'\right)(1+y'^2+z'^2) \end{aligned} \tag{28.11}$$

with

$$n_x = \frac{\partial\, n(x,y,z)}{\partial x} \quad , \quad n_y = \frac{\partial\, n(x,y,z)}{\partial y} \quad , \quad n_z = \frac{\partial\, n(x,y,z)}{\partial z} \tag{28.12}$$

[1] In the GNSS literature the term "troposphere" is very often used as a sysnonym for the neutral atmosphere which reaches up to ~ 100 km. Regarding the propagation of radio waves the atmosphere is divided into the neutral atmosphere and the ionosphere and everything below the ionosphere is regarded as "troposphere".

The equations 28.11 have to be solved with the boundary conditions

$$y(a) = z(a) = y(b) = z(b) = 0 \tag{28.13}$$

which guarantee that the signal propagates from the transmitter at $x = b$ to the receiver at $x = a$.

For the numerical solution a set of supporting points on the x axis needs to be defined:

$$x_i \ , \quad i = 1, \ldots, N \ , \quad \text{with } x_1 = a \ , \quad x_N = b$$

Using these x_i the differential equations could be approximated by finite differences but a more general solution can be found by connecting neigbored points by polynomials and differentiating the polynomials. The Lagrange polynomials $L(x)$ were chosen to connect sets of three neighbored points $(x_{i-1},\ x_i,\ x_{i+1})$:

$$L(x) = \sum_{k=i-1}^{k=i+1} f_k \cdot l_k(x) \quad , \qquad l_k(x) = \prod_{\substack{m=i-1 \\ m \neq k}}^{i+1} \frac{(x - x_m)}{(x_k - x_m)}$$

Here f_k is the value of the function at the node x_k, i. e. $f_k = y_k$ or $f_k = z_k$ and the l_k are the Lagrange basis polynomials. After replacing the functions $y(x)$ and $z(x)$ with the corresponding Lagrange polynomials and their derivatives the following system of algebraic equations is obtained:

$$\begin{aligned}
\sum_{k=i-1}^{k=i+1} y_k \, l_k''(x_i) \quad &= \quad \left(\frac{n_y}{n} - \frac{n_x}{n} \sum_{k=i-1}^{k=i+1} y_k \, l_k'(x_i) \right) \cdot \\
&\quad \left[1 + \left(\sum_{k=i-1}^{k=i+1} y_k \, l_k'(x_i) \right)^2 + \left(\sum_{k=i-1}^{k=i+1} z_k \, l_k'(x_i) \right)^2 \right] \qquad (28.14) \\
\sum_{k=i-1}^{k=i+1} z_k \, l_k''(x_i) \quad &= \quad \left(\frac{n_z}{n} - \frac{n_x}{n} \sum_{k=i-1}^{k=i+1} z_k \, l_k'(x_i) \right) \cdot \\
&\quad \left[1 + \left(\sum_{k=i-1}^{k=i+1} y_k \, l_k'(x_i) \right)^2 + \left(\sum_{k=i-1}^{k=i+1} z_k \, l_k'(x_i) \right)^2 \right] \qquad (28.15)
\end{aligned}$$

The $2(N-2)$ non-linear equations with $2(N-2)$ unknowns can now be solved in order to obtain the y_i and z_i which define the functions $y(x)$ and $z(x)$ at the supporting points x_i, $i = 2, \ldots, N-1$. The values $y_1 = y_N = 0$ and $z_1 = z_N = 0$ are defined by the boundary conditions and need not to be estimated. The derivatives of the Lagrange basis polynomials l_k are required only at the supporting points x_i and are simple combinations of the point distances Δx_i.

The equations XX can be rearenged and written in vector notation as $\boldsymbol{G}(\boldsymbol{x}) = 0$, where the vector function

$$\boldsymbol{G} = (g_{1,2}, g_{2,2}, \ldots, g_{1,i}, g_{2,i}, \ldots, g_{1,N-1}, g_{2,N-1})$$

consists of the functions $g_1(x_i) = g_{1,i}$ defined by equ. 28.14 and the functions $g_2(x_i) = g_{2,i}$ defined by equ. 28.15 evaluated at each supporting point x_i, $i = 2, \ldots, N-1$. The vector $\boldsymbol{x}$ of unknown is given by

$$\boldsymbol{x} = (y_2, z_2, \ldots, y_i, z_i, \ldots, y_{N-1}, z_{N-1})$$

A system of non-linear equatins $\boldsymbol{G}(\boldsymbol{x}) = 0$ can be solved with the Newton algorithm which finds the roots of the equations iteratively:

$$\boldsymbol{x}^{k+1} = \boldsymbol{x}^k - \left(\nabla \boldsymbol{G}(\boldsymbol{x}^k)\right)^{-1} \cdot \boldsymbol{G}(\boldsymbol{x}^k)$$

In this form the inverse of the Jacobian $J = \nabla \boldsymbol{G}$ is required. In most cases it is more efficient to solve a system of linear equations which provides the corrections $\Delta \boldsymbol{x}^k$ to the latest estimate of the root $\boldsymbol{x}^k$:

$$\Delta \boldsymbol{x}^k \cdot \nabla \boldsymbol{G}(\boldsymbol{x}^k) + \boldsymbol{G}(\boldsymbol{x}^k) = 0 \qquad \Rightarrow \qquad \boldsymbol{x}^{k+1} = \boldsymbol{x}^k + \Delta \boldsymbol{x}^k \tag{28.16}$$

The iteration continues until the norm $\left|\Delta \boldsymbol{x}^k\right|$ is sufficiently small and $\boldsymbol{x}^k$ is a good estimate of the root.

The Newton algorithm requires the Jacobian of $\boldsymbol{G}$, i. e. the partial derivatives of $\boldsymbol{G}$ with respect to the $2(N-2)$ unknowns y_i and z_i. In equ. 28.14/28.15 it was decided to connect 3 neigbored supporting points by the interpolating Lagrange polynomial and consequently each single function g_i depond on only 6 variables:

$$g_i(\boldsymbol{x}) = g_i\left(y_{i-1}, z_{i-1}, y_i, z_i, y_{i+1}, z_{i+1}\right)$$

This leads to a Jacobian with only 6 non-zero entries per row. The consecutive functions $g_{1,i}$ and $g_{2,i}$ always depend on the same set of unknowns and the Jacobian is a band matrix of band width 7.

The band width depends on the number of supporting points which are connected by the interpolating Polynom. If r points are connected the band width is $2r + 1$. In case of the Lagrange polynomials $r = 3$ is the minimum number of points and 7 is the minimum band width. Chosing a larger r should lead to more accurate and stable results but increases the number of partial derivatives which need to be computed and also the computing time to solve a system of linear equation with a larger band width. It seems that $r = 3$ is entirely sufficient.

The system of linear equation defined by equ. 28.16 can be solved by any standard technique but there are especially efficient algorithms for solving band matrices.

The Newton algorithm does in general not provide stable solutions and its results can depend to a large degree on the start vector $\boldsymbol{x}_0$ of the iteration. Fortunately, the deviations from a linear signal propagation are rather small, i. e. several 100 m over distances of several 100 km, and a straight line ($\boldsymbol{x}_0 = 0$) is a good starting point. It turned out that even a rather simple implementation of a Newton algorithm leads to stable and fast results. Usually, 1 – 3 iterations are sufficient to obtain accurate results.

The vector $\boldsymbol{x}$ provided by the Newton algorithm is the final solution for the bended signal path in the atmosphere. The supporting points x_i and the y_i and z_i from $\boldsymbol{x}$ define the bended signal path (x_i, y_i, z_i) which can in the next step be used to estimate the STD.

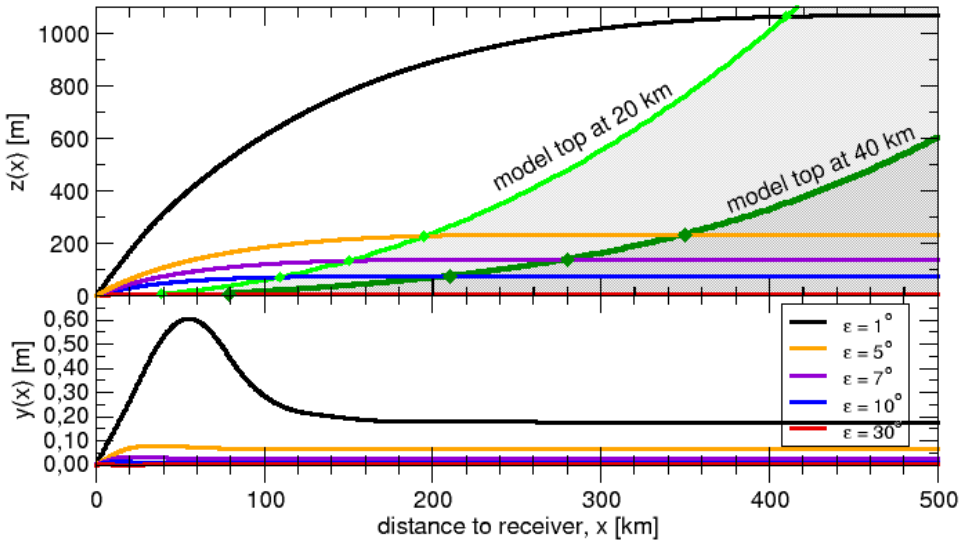


Figure 28.1: Raybendeing

The signal delay

After the raytracer provided an estimate of the signal path S in the atmosphere the signal delay can be computed using equ. 28.2. The second part $\int_G ds$ is the geometric distance between satellite and receiver which is known from the corresponding coordinates and the first part $\int_S n(s)\, ds$ was already computed within the last step of the raytracer:

$$STD = \int_a^b n(x,y,z)\sqrt{1 + y'(x)^2 + z'(x)^2}\, dx - \int_G ds$$

The derivatives of the discrete functions $y(x)$ and $z(x)$ are obtained by computing the derivatives of the corresponding Lagrange polynomials.

Reference systems, coordinates

geographische Koordinaten, ECEF, lokales Horizontsystem, Slant-System, ...

Transformations

Transformations between ellipsoidal and cartesian coordinates The ellipsoidal coordinates longitude λ, latitude β and height h can be transformed into ECEF kartesian coordinates X, Y and Z and vice versa. It is assumed that the origin of the ECEF cartesian system, i. e. the center of Earth, is also the center of the ellipsoid.

1) Transformations from ellipsoidal to cartesian coordinates Querkrümmungsradius des Ellipsoids:

$$N = \frac{a}{\sqrt{1 - e^2 \cdot \sin^2\beta}} = \frac{a}{\sqrt{1 - f(2-f)\sin^2\beta}} = \frac{a^2}{\sqrt{a^2 + \cos^2\beta + b^2\sin^2\beta}} \qquad (28.17)$$

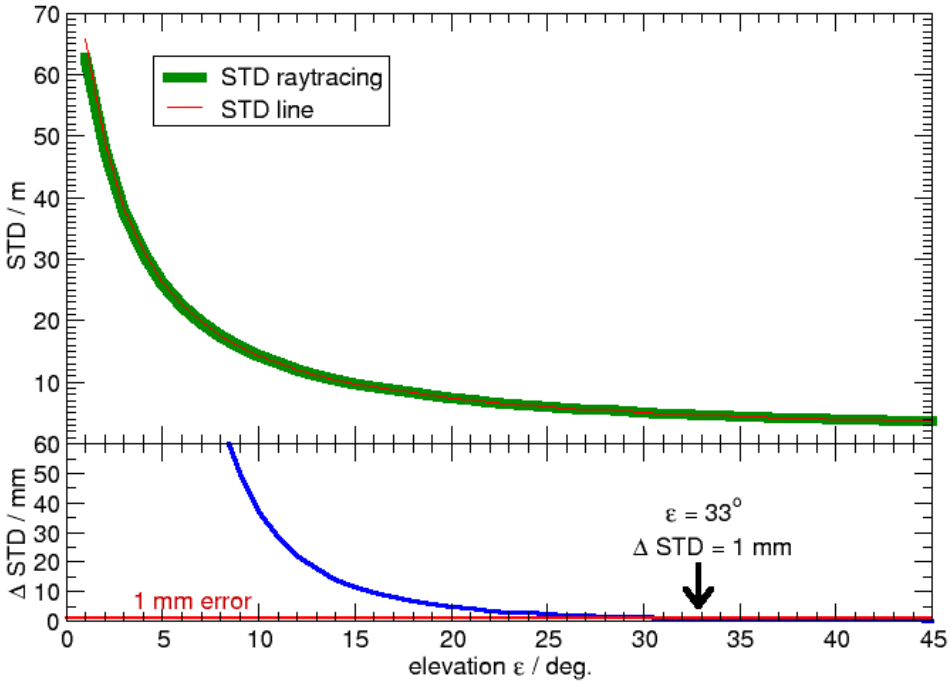


Figure 28.2: Raybendeing

The transformation ellipsoidal → cartesian coordinates is given by:

$$X = (N+h)\cdot\cos\beta\cdot\cos\lambda \tag{28.18}$$
$$Y = (N+h)\cdot\cos\beta\cdot\sin\lambda \tag{28.19}$$
$$Z = \left((1-e^2)N+h\right)\cdot\sin\beta \tag{28.20}$$
$$= \left(\frac{N}{1+e'^2}+h\right)\cdot\sin\beta \tag{28.21}$$
$$= \left(\frac{b^2}{a^2}N+h\right)\sin\beta \tag{28.22}$$

These equations show no singularities and can be translated to computer code without any modifications.

The specific form of the transformation depends on the choice of axes and angles as given in ??. Here it is assumed that

$$0 \le \lambda \le 2\,\pi \tag{28.23}$$
$$-\frac{\pi}{2} \le \beta \le -\frac{\pi}{2} \tag{28.24}$$

The results are identical if the longitude is between $-\pi$ and $+\pi$, positive values indicating eastern directions and negative values indicating western directions. However, the transformation from cartesian to ellipsoidal coordinates always provides longitudes between 0 and $2\,\pi$.

2) Transformations from cartesian to ellipsoidal coordinates The transformation from cartesian to ellipsoidal coordinates is more complicated and several special conditions need to be regarded (routine `Cart2Ellips`):

$$\gamma = \arctan\left\{\frac{Z + e'^2\, b \sin^3\theta}{\rho - e^2\, a \cos^3\theta}\right\} \tag{28.25}$$

$$\varphi = 2 \cdot \arctan\left(\frac{Y}{|X| + \rho}\right) \tag{28.26}$$

$$H = \frac{\rho}{\cos\beta} - N \tag{28.27}$$

with

$$\rho = \sqrt{X^2 + Y^2} \tag{28.28}$$

$$\theta = \arctan\frac{Z\, a}{\rho\, b}, \tag{28.29}$$

$$N = \frac{a}{\sqrt{1 - e^2 \cdot \sin^2\beta}} \tag{28.30}$$

(β needs to be computed before N! Here, the latitude β is required, not φ!)

The ellipsoidal coordinates (λ, β, h) are:

$$\lambda = \begin{cases} 0^\circ & \text{if } X = 0,\ Y = 0 \\ \varphi & \text{if } X \geq 0,\ Y \geq 0 \\ 360^\circ + \varphi & \text{if } X \geq 0,\ Y < 0 \\ 180^\circ - \varphi & \text{if } X < 0 \end{cases} \tag{28.31}$$

$$\beta = \begin{cases} +90^\circ & \text{if } \rho = 0,\ Z > 0 \\ 0^\circ & \text{if } \rho = 0,\ Z = 0 \\ -90^\circ & \text{if } \rho = 0,\ Z < 0 \\ \gamma & \text{if } \rho \neq 0 \end{cases} \tag{28.32}$$

$$h = \begin{cases} -b & \text{if } \rho = 0,\ Z = 0 \\ |Z| - b & \text{if } \rho = 0,\ Z \neq 0 \\ H & \text{if } \rho \neq 0 \end{cases} \tag{28.33}$$

The longitude λ and the latitude β are within the range

$$0 \leq \lambda \leq 2\,\pi \tag{28.34}$$

$$-\frac{\pi}{2} \leq \beta \leq -\frac{\pi}{2} \tag{28.35}$$

the height h is givem in meters.

Transformations between the local horizon system and the "slant system" The STD observations are given with respect to the local horizon system, i. e. the azimuth α and elevation ε of the GNSS satellite are given for a specific GNSS station. Azimuth and

elevation define a vector pointing from the GNSS station to the GNSS satellite and are independent from the bended signal path in the atmosphere.

The signal path can most easily be computed in a reference system which is aligned to the satellite-receiver-axis by solving Fermats's principle. Therefore, it is necessary to transform coordinates from the local horizon system to the "slant system". The "slant system" is defined as follows:

1. The origin of both reference systems is the GNSS station.

2. The x axis of the slant system points from the GNSS station to the GNSS satellite, i. e. defines the satellite-receiver-axis.

3. The z axis is restricted to a plane defined by the the center of Earth, the GNSS station and the GNSS satellite, i. e. for $\varepsilon = 0°$ the z axis is the vertical axis and for $\varepsilon = 90°$ the z axis is a horizontal axis.

4. The y axis is chosen in order to obtain a right handed Cartesian system.

It is assumed that azimuth and elevation are transformed to Cartesian coordinates defining a left handed local horizon system (x, y, z) (see XXX). The transformation from the local horizon system (x, y, z) to the slant system (x', y', z') is described by two basic rotations and one reflection at the y plane. The corresponding matrix R is given by

$$R = R_y(\varepsilon) \cdot R_z(\alpha) \cdot S_{xz} \tag{28.36}$$

with

$$R_y(\varepsilon) = \begin{pmatrix} \cos\varepsilon & 0 & \sin\varepsilon \\ 0 & 1 & 0 \\ -\sin\varepsilon & 0 & \cos\varepsilon \end{pmatrix}, \quad R_z(\alpha) = \begin{pmatrix} \cos\alpha & \sin\alpha & 0 \\ -\sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{pmatrix}, \quad S_{xz} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

The combined matrix

$$R(\varepsilon, \alpha) = \begin{pmatrix} \cos\varepsilon\cos\alpha & -\cos\varepsilon\sin\alpha & \sin\varepsilon \\ -\sin\alpha & -\cos\alpha & 0 \\ -\sin\varepsilon\cos\alpha & \sin\varepsilon\sin\alpha & \cos\varepsilon \end{pmatrix} \tag{28.37}$$

leads to a system of equations

$$\begin{aligned} x' &= x \cdot \cos\varepsilon\cos\alpha - y \cdot \cos\varepsilon\sin\alpha + z \cdot \sin\varepsilon \\ y' &= -x \cdot \sin\alpha - y \cdot \cos\alpha \\ z' &= -x \cdot \sin\varepsilon\cos\alpha + y \cdot \sin\varepsilon\sin\alpha + z \cdot \cos\varepsilon \end{aligned}$$

which is implemented in the subroutine `LocalHorz2Slant`, `STD_coord.f90`.

The inverse transformation from the slant system to the Cartesian local horizon system is given by

$$R^{-1} = R^{\mathrm{T}} = S_{xz} \cdot R_z(\alpha) \cdot R_y(\varepsilon) \tag{28.38}$$

with

$$R^{-1}(\varepsilon,\alpha) = \begin{pmatrix} \cos\varepsilon\cos\alpha & -\sin\alpha & -\sin\varepsilon\cos\alpha \\ -\cos\varepsilon\sin\alpha & -\cos\alpha & \sin\varepsilon\sin\alpha \\ \sin\varepsilon & 0 & \cos\varepsilon \end{pmatrix} \tag{28.39}$$

The corresponding system of equations

$$\begin{aligned} x &= x' \cdot \cos\varepsilon\cos\alpha - y' \cdot \sin\alpha - z' \cdot \sin\varepsilon\cos\alpha \\ y &= -x' \cdot \cos\varepsilon\sin\alpha - y' \cdot \cos\alpha + z' \cdot \sin\varepsilon\sin\alpha \\ z &= x' \cdot \sin\varepsilon + z' \cdot \cos\varepsilon \end{aligned}$$

is implemented in the subroutine `Slant2LocalHorz`, `STD_coord.f90`.

Empty: GNSS Slant Total Delay operator: Implementation

Subroutine structure of the STD operator std_delay_1

- SignalPath
  - Newton03Ell
    - FermatDglJacobi
      - Slant2LocalHorzMat
      - LocalHorz2CartMat
      - Slant2EllipsMat
      - RefracModelGrad2
      - ExpInt1Dgrad2
      - Slant2EllipsMatGrad2
    - BandSolv
- PathRefrac
  - Slant2Ellips
  - RefracModel
    - LayerSearch
    - NWein
- PathDelay
  - IntegPolyCube
- LineRefrac (optional)
  - RefracModel
  - IntegPolyCube

COSMO Interface to the STD Operator

The main part of the COSMO interface to the STD observation oprator is the routine `organize_gnss_std` in file `src_obs_use_org.f90`.

The purpose of the COSMO interface is to

1. read and distribute the station data, e. g. coordinates, names,

2. read and distribute the STD observations and

3. collect and distribute the model fields (p, T, qv) required to compute the delays.

The station information needs to be read only once and is distributed to all nodes as the observations will be distributed in order to optimize the load balance and all nodes must be able to process observations from all stations. The STD observations are read as soon as new data are available. Currently, the STDs come in hourly batches which are read and distributed in one pass. Within each time step of the COSMO model it is checked if/how many STDs with the corresponding observation time are available. The required module columns along the signal paths are requested and distributed between the nodes and the appropriate columns are transferred to the STD operator.

28.2 Model data structures

28.3 Vectors and Matrices – Module mo_dec_matrix

This module holds the declaration of data types `t_dec_matrix` for matrices and `t_dec_vector` for vectors in block decomposed form to be stored on different processors in an distributed memory environment. Arithmetic operations +, -, *, /, ... are defined on these data types. Using this data structures and operations, algebraic operations using vectors and matrices can be furmulated on a high level of abstraction hiding implementation details as sparse matrix representations or parallelization issues. Based on these data types and operators preconditioned conjugate gradient algorithms are implemented in module `mo_solver`.

Contents of module `mo_dec_matrix`

Data Type definitions:
Data types `t_vector`, `t_ivector`, and `t_matrix` hold matrices vectors in block decomposed form to be stored on different processors in an distributed memory environment. General information on the distribution is stored in the data type `t_dec_info`.

`t_vector`:	decomposed vector.
`t_ivector`:	decomposed integer-vector.
`t_matrix`:	block decomposed matrix.
`t_dec_info`:	decomposition meta data.

components of these data types (in general not referenced directly):

`t_matrix_block`:	matrix blocks.
`t_vector_segm` :	vector segments.
`t_ivector_segm`:	vector segments.

`t_info_block` : decomposition meta data.

The characteristics of the matrices (component `%qual` of type `t_matrix`) is indicated by one of the following predifined values:

`GENERAL`:	general representation, no restrictions.
`BDIAG`:	the matrix only consists of diagonal blocks.

Matrix blocks may be stored in different representations (component `%repr` of type `t_matrix_block`):

`ZERO`:	all elements are zero, no storage required.
`IDENT`:	identity matrix, no storage required.
`DIAGONAL`:	diagonal matrix.
`FULL`:	full n x m representation.
`PACKED`:	only upper triangle is stored.
`CHOLES`:	Cholesky decomposed (inverse matrices).
`CSR`:	compressed sparse row representation.
`CSC`:	compressed sparse column representation.
`MIRROR`:	this i,j block is not stored on this pe, use j,i instead.

For initialization and deinitialization the routines `construct` and `destruct` are defined on variables of data types `t_matrix`, `t_vector` and `t_dec_info`. The effect of `delete_storage` is similar to `destruct`, but dedicated to the deallocation of pointer components of return arguments from functions. Subroutine `dec_matrix_mem` diagnoses the memory used.

`construct`:	Initialization.
`destruct`:	Deinitialization, release memory of pointer components.
`delete_storage`:	Release memory of temporary variables (function return arguments).
`dec_matrix_mem`:	Diagnostic printout of memory used.

Arithmetic operations are defined on variables of data types `t_matrix`, `t_vector`:

`operator (+)`:	vector + vector.
`operator (-)`:	vector − vector.
`operator (*)`:	matrix ∗ vector; scalar ∗ vector.
`operator (/)`:	'elemental' vector / vector.
`operator (**)`:	vector ∗∗ integer.
`operator (==)`:	comparison: vector == vector.
`assignment (=)`:	assignment: vector = vector; vector = scalar; matrix = matrix.

`assign`:	matrix = matrix with optional parameters.
`reorder`:	reorder the elements of a matrix or vector.
`add_to`:	matrix = matrix + matrix.
`diag`:	diagonal of a matrix matrix.
`dot_product`:	dot product.
`norm2`:	2-norm of a vector.
`abs`:	'elemental' absolute value.
`max`:	'elemental' maximum value.
`min`:	'elemental' minimum value.
`amax`:	maximum of absolut value.
`sum`:	vector element sum.
`maxval`:	maximum value of elements.
`minval`:	minimum value of elements.
`random_number`:	set vector elements with random numbers (equaly distributed between 0 and 1).
`random_gauss`:	set vector elements with normal distributed random numbers.
`print`:	print vector.
`size`:	size of vector.
`get_row`:	return one row of the matrix.
`crep`:	character used for visualisation.

Operations on matrix blocks:

`allocate_block`:	allocate components of a matrix block.
`pack_matrix`:	store matrix in packed format.
`csr_matrix`:	store matrix in compressed sparse row format.
`csc_matrix`:	store matrix in compressed sparse column format.
`cholesky`:	calculate Cholesky factorisation.

The following routines gather the vector elements on one or all PE's.

`global`:	Store the vector segments redundantly on all processor elements.
`gather`:	Gather the vector segments temporarily on a specific processor element.
`release_mem`:	Release the temporarily stored vector segments.

The following routines send/receive/broadcast matrix blocks.

`p_send`:	overloaded `MPI_SEND` routine.
`p_recv`:	overloaded `MPI_RECV` routine.
`p_bcast`:	overloaded `MPI_BCAST` routine.

Examples

Interfaces

In detail the definitions of the public entities in this module are defined as follows:

Type **t_dec_info**

Decomposition meta data.

For a given vector or matrix detailed information on thet_matrix distribution over processors is stored in the data type t_dec_info. This information is required as a parameter for the specific construct routines for the data types t_vector, t_ivector and t_matrix.

A variable of type t_dec_info holds the total number of elements n, the number of matrix blocks for a row or column or the number of vector segments n_b as well as the array b of type t_dec_info:

```
type t_dec_info
  integer                       :: n     =  0      ! number of elements
  integer                       :: n_b   =  0      ! number of blocks
  type (t_info_block) ,pointer :: b (:) => NULL() ! info blocks
end type t_dec_info
```

For each block, b specifies the number of vector components n and the processor element pe the vector elements and matrix diagonal blocks or vector segments are stored on.

```
type t_info_block
  integer           :: n          =  0      ! number of elements
  integer           :: pe         = -1      ! processor hosting this block
end type t_info_block
```

Type **t_vector**

Vector data-type definition.

Vectors are decomposed into segments (corresponding to the respective blocks of decomposed matrices). The n coefficients of a segment are stored on processor element PE within the component x(:). More specific information on the decompostition may be retrieved by the reference info to data type t_dec_info.

```
type t_vector_segm
  integer           :: pe   = -1      ! processor hosting this block
  integer           :: n    =  0      ! number of elements
  real(wp) ,pointer :: x(:) => NULL() ! coefficients
end type t_vector_segm
```

The segments of the decomposed vector are kept in the component `s(:)` of the data type `t_vector`. `n` and specifies the total number of elements and `n_s` the number of segments. `alloc_l` indicates if the vector is a local variable within a subroutine or function. This information is required for the deallocation of pointer componebts in function return arguments. Arrays segments may be either distributed over processors ore stored rdundantly on each processor. The latter case is indicated by `global=.false.`.

```
type t_vector
  type (t_dec_info)    ,pointer :: info    => NULL()  !
  decomposition info
  character(len=8)              :: name    = '???? '  ! name
  integer                       :: n       =  0       ! number of elements
  integer                       :: n_s     =  0       ! number of segments
  integer                       :: alloc_l =  0       ! allocation level
  logical                       :: global  = .false.  ! global allocation
  type (t_vector_segm) ,pointer :: s (:)   => NULL()  ! vector blocks
  type (t_vector)      ,pointer :: next    => NULL()  ! linked list for
  type (t_vector)      ,pointer :: prev    => NULL()  !   memory usage diagnost
end type t_vector
```

Type **t_ivector**

Integer-vector data-type definition.

The definition of type `t_ivector` is similar to that of type `t_ivector`; the main difference is that the component `s%(:)% x(:)` is of type `integer` instead of `real`:

```
type t_ivector
  ...
  type (t_ivector)      ,pointer :: next    => NULL()  ! linked list for
  type (t_ivector)      ,pointer :: prev    => NULL()  !   memory usage diagnos
end type t_ivector

type t_ivector_segm
  ..
  integer ,pointer :: x(:) => NULL() ! coefficients
end type t_ivector_segm
```

Type **t_matrix**

Matrix data-type definition.

The data type `t_dec_matrix` holds a symmetric `nxn` matrix decomposed in `n_b` times `n_b` blocks `b(:,:)` of data type `t_dec_matrix`. More specific information on

the decompostition may be retrieved by the reference `info`. Special characteristics of the matrix (e.g. blog diagonal matrices) are indicated by the component `qual` with allowed values given by the predefined constants `GENERAL` or `BDIAG`. The latter value indicates that only diagonal blocks `b`(i,i) are present.

Each matrix block is stored in a variable of type `t_matrix_block` defining the number of rows `m` and columns `n` within this block as well as the number of nonzero elements `nonzero`. (For nonzero==0 no further storage is required). Matrix blocks are distributed over processor elements in a parallel environment. The index of the processor element hosting this block is specified in `pe`. Currently the parallisation strategy is to hold all elements of a row on the same processor element. Depending on the value of the representation flag `repr` the matrix elements are stored in different representations:

`ZERO`:
: All elements of the block are zero, no storage is required.

`IDENT`:
: The matrix block is the identity matrix, no storage is required.

`DIAGONAL`:
: The matrix block is a diagonal matrix. The diagonal is stored in component `packed(1:n)`.

`FULL`:
: The block is stored in full representation in component `full(1:n,1:m)`.

`PACKED`:
: The block is symmetric, only the upper triangle is stored in component `packed`

`CHOLES`:
: The block represents the inverse of a matrix block, stored in Cholesky decomposed form in component `packed(:)`

`CSR, CSC`:
: If the number of nonzero elements is small, the matrix may be stored in a compressed sparse row or column representation using `packed(:)`, `ia(:)` and `ja(:)`.

`MIRROR`:
: Because of symmetry, this block `b(i,j)` is not stored, the transpose of `b(j,i)` shall be used instead.

The matrix may have the following characteristics (

```
integer ,parameter :: GENERAL = 0 ! general representation, no restrictions
integer ,parameter :: BDIAG   = 1 ! block diagonal representation
```

Each block may be represented in one of the following forms (% repr)

```
integer  ,parameter :: ZERO     = 0 ! all elements are zero, not stored
integer  ,parameter :: IDENT    = 1 ! identity matrix, no storage required
integer  ,parameter :: DIAGONAL = 2 ! diagonal matrix
integer  ,parameter :: FULL     = 3 ! full n x m representation
integer  ,parameter :: PACKED   = 4 ! only upper or lower triangle is stored
integer  ,parameter :: CHOLES   = 5 ! Cholesky decomposed (inverse matrices)
integer  ,parameter :: CSR      = 6 ! compressed sparse row representation
integer  ,parameter :: CSC      = 7 ! compressed sparse column representation
integer  ,parameter :: MIRROR   = 9 ! this i,j block is not stored, use j,i


type t_matrix_block
  integer           :: m          =  0       ! number of rows
  integer           :: n          =  0       ! number of columns
  integer           :: nonzero    = -1       ! number of nonzero elements
  integer           :: pe         = -1       ! processor hosting this block
  integer           :: repr       = -1       ! representation (FULL,..)
  character         :: tri        = ''       ! triangle ('U' or 'L') stored
  real(wp) ,pointer :: full (:,:) => NULL()  ! coefficients   FULL   repres.
  real(wp) ,pointer :: packed (:) => NULL()  ! coefficients   PACKED repres.
  integer  ,pointer :: ia     (:) => NULL()  ! row indices    CSR    repres.
  integer  ,pointer :: ja     (:) => NULL()  ! column indices CSR    repres.
end type t_matrix_block


type t_matrix
  type(t_dec_info)     ,pointer :: info    => NULL()  !
  decomposition info
  character(len=8)              :: name    = '???? '  ! name
  integer                       :: n       =  0       ! number of rows/cols
  integer                       :: n_b     =  0       ! number of blocks/col
  integer                       :: qual    =  GENERAL ! qualities
  type(t_matrix_block) ,pointer :: b (:,:) => NULL()  ! matrix blocks
  type(t_matrix)       ,pointer :: next    => NULL()  ! linked list for
  type(t_matrix)       ,pointer :: prev    => NULL()  !   memory
  usage diagn.
end type t_matrix
```

Interface **construct**

Construct is a generic routine name acting on variables of type t_vector, t_ivector, t_matrix, or t_dec_info. It must be called for any variable of these types before its first use to set up information on the data distribution and to allocate pointer components. Furthermore the variables are inserted into a linked list in order to monitor the memory consumption. The specific interfaces are as follows:

subroutine **construct_dec_info** `(z, n, n_b)`

This routine is called to set the number of elements of the vectors or the number of rows and columns of the matrices used, as well as the number of blocks:

The parameters denote:

`z`	*type:* `type (t_dec_info) ,intent(out)` Variable to hold information on the data distribution.
`n`	*type:* `integer ,intent(in)` Number of elements of vectors or number of rows and columns of the matrices used.
`n_b`	*type:* `integer ,intent(in)` Number of vector segments or matrix blocks per row or column.

subroutine **construct_info_block** `(z, n, pe)`

This routine must be called for any element of component `b(:)` of a variable of type `t_dec_info` in order to specify the number of elements in a vector segment or matrix block and to specify the processor index the segment or matrix is stored on.

The parameters denote:

`z`	*type:* `type (t_info_block) ,intent(out)` Type of component `b` of type `t_dec_info`.
`n`	*type:* `integer ,intent(in)` Number of elements in this segment or block.
`pe`	*type:* `integer ,intent(in)` Processor index the segment or block is stored on.

subroutine **construct_vector** `(x ,info [,name] [,global])`

The parameters denote:

`x`	*type:* `type (t_vector) ,intent(out)` Variable to initialise.
`info`	*type:* `type (t_dec_info) ,pointer` Information on memory decomposition used to set up the variable `x`. A reference to this structure is assigned to a component of `x`. It may be used in subsequent operations.
`name`	*type:* `character(len=*) ,intent(in) ,optional` If present, this string is assigned to component `x% name`. It may used in subsequent routines (printout, diagnostics of memory usage) to identify this vector.
`global`	*type:* `logical ,intent(in) ,optional` If present and `.true.`, The content of the vector is always stored redundantly on all processor elements.

subroutine **construct_ivector** (x ,info [,name] [,global])

The parameters denote:

x	*type:* type (t_ivector) ,intent(out) Variable to initialise.
...	The remaining variables are the same as those in subroutine construct_vector.

subroutine **construct_matrix** (a ,info [,name] [,qual])

The parameters denote:

a	*type:* type (t_matrix) ,intent(out) Variable to initialise.
info	*type:* t_dec_info ,pointer Information on memory decomposition used to set up the variable a. A reference to this structure is assigned to a component of a. It may be used in subsequent operations.
name	*type:* character(len=*) ,intent(in) ,optional If present, this string is assigned to component a% name. It may used in subsequent routines (printout, diagnostics of memory usage) to identify this matrix.
qual	*type:* integer ,intent(in) ,optional If present, this parameter should have one of the predefined values GENERAL (default) or BDIAG. If BDIAG is passed, the matrix is a block-diagonal matrix. In this case the offdiagonal elements are assigned to zero.

Subroutine **destruct** (x)

Destruct is a generic routine name acting on variables x of type: t_vector, t_ivector, t_matrix, t_matrix_block, or t_dec_info. Destruct must be called for any variable of these types before they go out of scope. Pointer components of the variables are deallocated. The variables are removed from the linked list maintained for monitoring of allocated memory.

Subroutine **delete_storage** (x)

The parameters denote:

x	*type:* type (t_vector) ,intent(in) Variable to clean up.

This routine deallocates pointer components of temporary variables (function return arguments) passed to other routines. The argument is intent(in) because the arguments of the subroutines or function may require this. This routine shall be used only inside routines with actual arguments of type t_vector. Details of usage are given in Section 28.3.

Implementation details: Memory allocation diagnostics

Implementation details: Deallocation of temporary data

Implementation details: Matrix block representations

28.4 Background error covariance model mo_fg_cov

Module `mo_fg_cov` provides the routines to set up the background error covariance matrix. The routines calculate covariances for a pair of quantities (out of geopotential height, virtual temperature, wind component u, or v) at two locations (specified by latitude, longitude and log(pressure)). In order to make use of simlifications in case of separability of vertical and horizontal correlations a two-step aooroach is followed:

1. Observation points with the same horizontal coordinates (in general all observed quantities of a report) are gathered in a set. A routine is called to precompile quantities applicable to this set (variances, vertical correlations, etc.). These quantities are kept in a derived data type `t_rowcol`.

2. For any pair of observation sets the corresponding covariance matrix is calculated by subroutine making use of the precompiled quantities in data type `t_rowcol`.

As an alternative to step 2, instead of the covariance matrix between the two sets of observations, the product of this matrix with a vector may be compiled on the fly. This mode is used in the postmultiplication step.

28.5 Utilities

28.5.1 Write or Read GRADS Plot-Files – Module mo_grads

GRADS is a freeware plotting software (Home-page[2], Users Guide[3]) able to handle (regular, not icosahedral) gridded model data and observational data as well. Data is stored as IEEE unformatted data without Fortran record marks. (GRADS also has capabilities to read GRIB encoded files not used here). Information on the file content is stored in a descriptor file (extension `.ctl`). This module provides routines to read and write data and description files.

Contents of module `mo_grads`

Data Type definitions:

`t_ctl`: Data type holding the content of the GRADS descriptor file.

Routines to set up the content of the GRADS descriptor file:

[2] http://www.iges.org/grads/

[3] http://www.iges.org/grads/gadoc/

init_ctl: Routine to set the initial content of a variable of type t_ctl.

read_ctl: Routine to read the content of a GRADS descriptor file into a variable of type t_ctl.

add_var: Add an entry for a field to the descriptor file (variable of type t_ctl) of a gridded data set.

write_ctl: Routine to write the content of a variable of type t_ctl to a GRADS descriptor file.

destruct: Deallocate memory used by components of a variable of type t_ctl.

The following routines assist by converting components of t_ctl:

c_month: Character array to convert an integer (1..12) to a string ('jan'..'dec') encoding the month in the GRADS descriptor file.

mmm2mm: Function to convert a string ('jan'..'dec' or 'JAN'..'DEC') to an integer (1..12). 0 is returned in case of an error.

The following routines read or write from/to the GRADS data file. While writing the content of the content of the descriptor file stored in some variable of type t_ctl is updated.

write_var: Write a gridded field to the GRADS file.

read_var: Read a gridded field from the GRADS file.

write_stat: Write station data to the GRADS file.

Examples

To write a gridded GRADS data set the following sequence of operations may be performed:

```
type (t_ctl) ::  ctl
```

Declare a variable ctl to hold the descriptor of the GRADS file.

```
call init_ctl (ctl, file='filename', nx=128, ny=64, ke=31)
```

Initialize the descriptor file content for a gridded data set holding fields of size 128x64x31.

```
call add_var (ctl, 't', 31, 'temperature (K)')
call add_var (ctl, 'ps', 'surface pressure (hPa)')
```

Add two variables to the list in the file descriptor. In most cases these calls are not required because the list is extended automatically as long as variables are added within the first time step.

```
call write_var (ctl, t(:,:,:), 't')
call write_var (ctl, q(:,:,:), 'q')
call write_var (ctl, ps(:,:), 'ps',t=2)
```

Write the fields `t` and `q` at time step 1 and `ps` at time step 2 to the GRADS file at disk.

```
call write_ctl (ctl)
call destruct (ctl)
```

Finally write the file descriptor to the disk. Deallocate pointer components of the descriptor in not needed any more.

To read back the surface pressure the following sequence may be used:

```
call read_ctl (ctl, 'filename')
call read_var (ctl, ps(:,:), ps, t=2)
call destruct (ctl)
```

In order to facilitate to write gridded atmospheric fields to a grads file a variant of the generic routine **init_ctl** is defined in module `mo_atm_grid` to generate the content of the GRADS descriptor file from the data structure `t_grid` which describes the model grid. Grid data (orography, etc.), atmospheric fields or a complete atmospheric field are written to a GRADS data set by calling the generic subroutine **to_grads** defined in modules `mo_atm_grid`, `mo_memory`, and `mo_atm_state`, respectively.
The following sequence writes the atmospheric state stored in variable `atma` as well as the fields of its grid `atma% grid` to a gridded GRADS file. The atmospheric state stored in variable `ana` is written to time step 2. Note that the icosahedral grid is not supported by grads.

```
call init_ctl (ctl, 'filename', atma% grid)
call to_grads (ctl, atma% grid)
call to_grads (ctl, atma)
call to_grads (ctl, ana, t=2)
call write_ctl (ctl)
call destruct (ctl)
```

In order to write a gridded GRADS data set using these routines the following sequence of operations may be performed:

```
call init_ctl (ctl,'synop.dat',dtype='station',surdat= (/'ps
','t2m','q2m'/))
call write_stat (ctl, '10056', dlat, dlon, surf=(/1023.,231.,0.001/))
...
call write_ctl(ctl)
call destruct (ctl)
```

Interfaces

In detail the definitions of the public entities in this module are defined as follows:

Type **t_ctl**

The names of the components in correspond to whose of the entries in a GRADS descriptor file. In general direct access to the components is not required becaue most entries will be set by passing appropriate parameters to subroutine `init_ctl`.

Subroutine **init_ctl** `(ctl, [file,] [dtype,] [title,] [nx,] [ny,] [ngl,] [ke,] [di,] [dj,] [lo1,] [la1,] [tdefn,] [tdefi,] [tdefd,] [undef,] [yrev,] [surdat,] [levdat,] [surcom,] [levcom])`

Set the initial content of the Grads data descriptor.

The parameters denote:

`ctl` — *type:* `type (t_ctl) ,intent(inout)`
Data descriptor information to be set.

`file` — *type:* `character(len=*) ,intent(in) ,optional`
Name of the corresponding data file. This parameter shall provide the full path name of the file to write. The name of the descriptor file is derived by removing the extension `.dat` (if present) and appending the extension `.ctl`. The name of the data file and description file will be used to write (or read) these files later. The name of the data file is written to the `dset` entry of the descriptor file as well. A în front of the basename indicates that it is sought always in the directory of the descriptor file. In this case the leading pathname is removed from the string written to the descriptor file. In case of a station data set the name of the station map file is constructed by appending the extension `.map` .

`dtype` — *type:* `character(len=*) ,intent(in) ,optional`
This entry specifies the type of data being described. Pass `'station'` for station data, otherwise gridded binary data is assumed. `GRIB` data is not handled by this module. The default is gridded binary but if one of the parameters `surdat` or `levdat` is provided, 'station' is assumed.

`title` — *type:* `character(len=*) ,intent(in) ,optional`
This string is written to the respective entry in the descriptor file and should give a brief description of the contents of the data set.

`nx, ny, ngl, ke` — *type:* `integer ,intent(in) ,optional`

These parameters denote the horizontal (`nx, ny`) and vertical (`ke`) size of the gridded data. They are used to set the respective entries `xdef`, `ydef` and `zdef` in the description file. If the parameter `ngl` is provided, it denotes the number of Gaussian latitudes and `ydef` is set appropriately.

`di, dj, lo1, la1` *type:* `real(wp) ,intent(in) ,optional`
These parameters specify the horizontal spacing (increments for longitudes and latitudes as well as starting values). The respective entries `xdef`, `ydef` in the description file are set appropriately.

`tdefn` *type:* `integer ,intent(in) ,optional`
Number of time slots.

`tdefi` *type:* `character(len=*) ,intent(in) ,optional`
Initial time given in the format *hh:mmZddmmmyyyy*. If not specified, *hh* defaults to 00, *mm* defaults to 00, and *dd* defaults to 1. The character string *mmm* specifying the month may be derived by means of the constant array `c_month` defined in this module.

`tdefd` *type:* `character(len=*) ,intent(in) ,optional`
Time increment given in the format `vvkk` where `kk` is a one or two integer digit and `kk` is one of `mn` (minute), `hr` (hour), `dy` (day), `mo` (month), or `yr` (year).

`undef` *type:* `real(wp) ,intent(in) ,optional`
Specifies the undefined, or missing, data value. `-huge(1)` (in single precision) is used as the default.

`yrev` *type:* `logical ,intent(in) ,optional`
Indicates that the Y dimension (latitude) in the data file has been written in the reverse order from what GRADS assumes. An important thing to remember is that GRADS still presents the view that the data goes from south to north. The YDEF statement does not change; it still describes the transformation from a grid space going from south to north. The reversal of the Y axis is done as the data is read from the data file.

`surdat, levdat` *type:* `character(len=*) ,intent(in) ,optional`
In case of a surface data file the names of the surface variables and the names of the level-dependent variables are listed in `surdat` and `levdat`, respectively.

`surcom, levcom` *type:* `character(len=*) ,intent(in) ,optional`
In case of a surface data file a text description of the variable (max 40 characters).

Subroutine **read_ctl** `(ctl, file, [iostat])`

Read the content of a GRADS descriptor file

The parameters denote:

`ctl`	*type:* `type (t_ctl) ,intent(inout)` GRADS data descriptor to read.
`file`	*type:* `character(len=*) ,intent(in)` Name of the (.ctl-)file to read.
`iostat`	*type:* `integer ,intent(out) ,optional` I/O status variable, returns zero if no error occurred. If an error occurs while the parameter is not present the routine will abort.

Subroutine **add_var** `(ctl, name, [levels], [comment])`

This routine adds a variable entry to the GRADS descriptor. In most cases calling this routine is not necessary because variables are added as long they are written by subroutine `write_var` into the first time slot.

The parameters denote:

`ctl`	*type:* `type (t_ctl) ,intent(inout)` Descriptor data to be modified.
`name`	*type:* `character(len=*) ,intent(in)` Name of the variable.
`levels`	*type:* `integer ,intent(in) ,optional` Number of levels for the variable. If levels is 0, the variable does not correspond to any vertical level (e.g. surface variables).
`comment`	*type:* `character(len=*) ,intent(in) ,optional` Description of the variable (max 40 characters).

Subroutine **write_ctl** `(ctl, [unit])`

This routine writes the GRADS descriptor file. It must be called when all fields have been written.

The parameters denote:

`ctl`	*type:* `type (t_ctl) ,intent(in)` Variable holding the content of the GRADS descriptor.
`unit`	*type:* `integer ,intent(in) ,optional` Unit number of the file. If missing, a file unit is opened by the routine, deriving the file name from the name of the GRADS data file.

Subroutine **destruct** `(ctl)`

Deallocate memory used by components of a variable of type `t_ctl`.

The parameters denote:

`ctl`	*type:* `type (t_ctl) ,intent(inout)`

Parameter **c_month** `(imonth)`

Character array to convert an integer (1..12) to a string ('jan'..'dec') encoding the month in the GRADS descriptor file.

The parameter is defined as:

```
character(len=3) ,parameter ::  c_month (12) =
(/'jan','feb','mar','apr','may','jun','jul','aug','sep','oct','nov','dec'/)
```

Function **mmm2mm** `(c)`

Function to convert a string ('jan'..'dec' or 'JAN'..'DEC') to an integer (1..12). 0 is returned in case of an error.

The parameters denote:

`mmm2mm`	*type:* `integer`
`c`	*type:* `character (len=3) ,intent(in)`

Subroutine **write_var** `(ctl, x, name, [t], [comment], [yrev], [iostat])`

Write a gridded field to the GRADS data file. The file name as well as the position within the file (depending on the time step and name) is determined from the descriptor. If the variable is not yet listed in the file descriptor and the first time step is written, the variable is added to the list.

The parameters denote:

`ctl`	*type:* `type (t_ctl) ,intent(inout)` Variable holding the GRADS file descriptor information.
`x (:,:[,:])`	*type:* `real(wp) ,intent(in)` 2- or 3-dimensional field to write.
`name`	*type:* `character(len=*) ,intent(in)` Name of the field.
`t`	*type:* `integer ,intent(in) ,optional` Time step to be written. Default is 1.
`comment`	*type:* `character(len=*) ,intent(in) ,optional` Description of the variable.
`yrev`	*type:* `logical ,intent(in) ,optional` Indicates that the Y dimension (latitude) in the data file is passed in the reverse order from what GRADS assumes.
`iostat`	*type:* `integer ,intent(out),optional` Return status, 0 if no error occurred. If an error occurs while the parameter is not present the routine will abort.

Subroutine **read_var** `(ctl, x, name, [t], [yrev], [iostat])`

A gridded field is read from the GRADS data file.

The parameters denote:

`ctl`	*type:* `type (t_ctl) ,intent(inout)` Variable holding the GRADS file descriptor information.
`x (:,:)`	*type:* real(wp) ,intent(out) Field to be read. currently only 2-d arrays are supported.
`name`	*type:* `character(len=*) ,intent(in)` Name of the field.
`t`	*type:* `integer ,intent(in) ,optional` Time step to be read. Default is 1.
`yrev`	*type:* `logical ,intent(in) ,optional` Indicates that the Y dimension (latitude) in the data file shall be passed in the reverse order from what GRADS assumes.
`iostat`	*type:* `integer ,intent(out),optional` Return status, 0 if no error occurred. If an error occurs while the parameter is not present the routine will abort.

Subroutine **write_stat** `(ctl, id, lat, lon, [t], [z], [surf], [lev], [new])`

The parameters denote:

`ctl`	*type:* `type (t_ctl) ,intent(inout)` Variable holding the GRADS file descriptor information.
`id`	*type:* `character(len=*) ,intent(in)` The station ID uniquely identifies the station. It can be 1 to 7 characters long.
`lat, lon`	*type:* `real(wp) ,intent(in)` Coordinates of the station (degree)
`t`	*type:* `real(wp) ,optional ,intent(in)` Time of the observation. *currently disabled!*
`z`	*type:* `real(wp) ,optional ,intent(in)` Height of the observation in case of level-dependent data.
`surf(:)`	*type:* `real(wp) ,optional ,intent(in)` Values of surface observations.
`lev (:)`	*type:* `real(wp) ,optional ,intent(in)` Values of level-dependent observations.
`new`	*type:* `logical ,optional ,intent(in)` Setting this flag to enforces a new station record to be written or not. In general a new record is written whenever the station ID `id` changes.

28.5.2 Adaptive estimation of error statistcs – Module mo_err_tune

Detlef Pingel

The adaptive method:

Tuning ansatz The analysis vector obtained by the 3dVar assimilation system can be regarded as a linear combination of background and observation data. Ideally, this combination minimises the variance of the analysis with respect to the (unknown) true state of the atmosphere. However, this requires an optimal choice for the covariance matrices of the observation and back ground errors used. As these covariances are not known exactly, approximations are used in the actual data assimilation schemes. Therefore, a good estimate of the error covariances used is crucial for an efficient operation of the data assimilation. In the assimilation scheme of the 3dVar, the correlation patterns themselves are given by NMC approximations using forecast statistics. As correlations are normalised by definition, the absolute size of the error covariances has to be determined separately.

For the 3dVar assimilation scheme, an optimisation method for the error covariances has been implemented, following the studies by G.Desrozier + al. (G.Desrozier, S.Ivanov, Q.J.R. Meteorol. Soc. 2001, 127, p1433 ; B.Chapnik+al, Q.J.R. Meteorol. Soc. 2004, 130, p2253). It tunes the covariance matrices of the observation and background error by appying a scaling approach as follows: Let $\mathbf{B}$ and $\mathbf{R}$ the (unknown) optimal background and observation error covariance matrices, respectively, with $\tilde{\mathbf{B}}$ and $\tilde{\mathbf{R}}$ the corresponding matrices as specified in the actual 3dVar assimilation system. Let the observation and background errors be well represented by $\mathbf{R}$ and $\mathbf{B}$, i.e.

$$\mathbf{B} = E\{(\mathbf{x}^b - \mathbf{x}^t)(\mathbf{x}^b - \mathbf{x}^t)^T\} \tag{28.40}$$
$$\mathbf{R} = E\{(\mathbf{y} - H(\mathbf{x}^t))(\mathbf{y} - H(\mathbf{x}^t))^T\} \tag{28.41}$$

with the unknown true state $\mathbf{x}^t$ of the atmosphere, the bachground state $\mathbf{x}^b$, the observation vector $\mathbf{y}$ and the forward operator H.

The tuning approach relies on the assumption that $\tilde{\mathbf{B}}$ and $\tilde{\mathbf{R}}$ can be brought closer to the optimal matrices $\mathbf{B}$ and $\mathbf{R}$ by a scaling relation

$$\mathbf{B} = \mathbf{B}(s^b) = s^b\tilde{\mathbf{B}} \tag{28.42}$$
$$\mathbf{R} = \mathbf{R}(s^o) = s^o\tilde{\mathbf{R}} \tag{28.43}$$

using the tuning coefficients s^b and s^o. Multiplication of the covariance matrices in this way to approximate the true covariance matrices basically represents the tuning operation.

Expectation value of the cost function In 3dVar, the analysis vector

$$\mathbf{x}^a = \mathbf{x}^b + \delta\mathbf{x}$$

minimises the cost function

$$J(\delta\mathbf{x}) = J^b(\delta\mathbf{x}) + J^o(\delta\mathbf{x}) \tag{28.44}$$
$$= \frac{1}{2}\delta\mathbf{x}^T\mathbf{B}^{-1}\delta\mathbf{x} + \frac{1}{2}(\mathbf{d} - \mathbf{H}\delta\mathbf{x})^T\mathbf{R}^{-1}(\mathbf{d} - \mathbf{H}\delta\mathbf{x})$$

with the innovation vector

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

and the linearised forward operator $\mathbf{H}$. The analysis increment $\delta\mathbf{x}$ is given as

$$\delta\mathbf{x} = \mathbf{B}\mathbf{H}^T(\mathbf{H}\mathbf{B}\mathbf{H}^T + \mathbf{R})^{-1}\mathbf{d} = \mathbf{K}\mathbf{d}$$

with the Kalman gain matrix

$$\mathbf{K} = \mathbf{B}\mathbf{H}^T(\mathbf{H}\mathbf{B}\mathbf{H}^T + \mathbf{R})^{-1}$$

For the evaluation of the covariance errors the value of the cost function in its minimum is of relevance. Generally (as in the case of the 3dVar) the cost function J is composed of several terms J^j,

$$J = \sum_j J^j$$

each of which is the sum of m_j elements. Each term J^j can be associated with an error covariance matrix $\mathbf{S}_j$ and a corresponding linearised interpolation operator $\mathbf{\Gamma}_j$. The expectation value of each term J^j can then be specified under certain assumptions (O.Talagrand, Proceedings ECMWF, 1999). Then the minimum expectation values of the components J^j of the cost function are given by

$$E\{J^j\} = \frac{1}{2}\{m_j - Tr(\Gamma_j^T\mathbf{S}_j^{-1}\Gamma_j\mathbf{P}^a)\}$$

In this relation $\mathbf{P}^a$ indicates the covariance matrix of the estimated analysis error of the analysis including the complete set of observations. For a proof of this relation, see above reference and G.Desroziers and S.Ivanov, Q.J.R. Meteorol. Soc. (2001), **127**, pp. 1433-1452.

For the tuning of error statistics within the 3dVar, the relation (28.5.2) can be employed in the following way: For tuning of the background or observation error statistics, the matrices Γ_j and $\mathbf{S}_j$ are either ($j = b$ and $j = o$, resp.)

$$\begin{array}{ll} \text{background cost function} J^b & \Gamma_j = \Gamma_b = \mathbf{H}, \quad \mathbf{S}_j = \mathbf{S}_b = \mathbf{B} \\ \text{observation cost function} J^o & \Gamma_j = \Gamma_o = \mathbf{I}, \quad \mathbf{S}_j = \mathbf{S}_o = \mathbf{R} \end{array}$$

with the identity matrix $\mathbf{I}$. For this application, the analysis error covariance matrix is $\mathbf{P}^a = \mathbf{B} - \mathbf{K}H\mathbf{B}$. An expectation value of the background cost function $J^j = J^b$ can, using $m_j = m_b = n$ background data, therefore be obtained as

$$\begin{aligned} E\{J^b\} &= \frac{1}{2}\{n - Tr(\mathbf{B}^{-1}(\mathbf{B} - \mathbf{K}\mathbf{H}\mathbf{B}))\} \\ &= \frac{1}{2}\{n - Tr(\mathbf{I}_n - \mathbf{B}^{-1}\mathbf{K}\mathbf{H}\mathbf{B})\} \\ &= \frac{1}{2}Tr(\mathbf{K}\mathbf{H}) \\ &= \frac{1}{2}Tr(\mathbf{H}\mathbf{K}) \end{aligned} \tag{28.45}$$

with the Kalman gain matrix $\mathbf{K} = \mathbf{P}^a\mathbf{H}^T\mathbf{R}^{-1}$ and $\mathbf{I}_n$ indcating the identity matrix of size n. In a similair way, using $m_j = m_o = p$ observations, the value of the observation cost function $J^j = J^o$ can be determined by

$$\begin{aligned} E\{J^o\} &= \frac{1}{2}\{p - Tr(\mathbf{H}^T\mathbf{R}^{-1}\mathbf{H}\mathbf{P}^a)\} \\ &= \frac{1}{2}\{p - Tr(\mathbf{H}^T\mathbf{K}^T)\} \\ &= \frac{1}{2}Tr(\mathbf{I}_p - \mathbf{H}\mathbf{K}) \end{aligned} \tag{28.46}$$

Combination of the two above results yields the value of the complete cost function

$$E\{J\} = E\{J^o + J^b\} = E\{J^o\} + E\{J^b\} = Tr(\mathbf{I}_p)/2 = p/2$$

The expectation value of the cost function is therefore proportional to the number of observations, if $\mathbf{B}$ and $\mathbf{R}$ are specified consistently. However, a deviation of the value of $E\{J\}$ from $p/2$ indicates ill-specified observation- and/or background errors.

The equations (28.45) and (28.46) relate expectation values of the cost function to expectation values of error covariance matrices (via assumptions (28.40) and (28.41)), to be valid when considering large ensembles of obsevation and background data. For the actual application of the tuning algorithm, it is assumed that these relations hold also for a single realization of these data (i.e. for data in a single assimilation time window). Therefore, in the following the expectation value of the cost function $E\{J\}$ is approximated by its value of a single realization J. The covariance matrices $\mathbf{B}$ and $\mathbf{R}$ are supposed to reflect the error statistics of the given set of data according to equations (28.40) and (28.41) except for an overall scaling coefficient (which is to be optimised in the tuning algorithm).

Application to tuned covariance matrices: When considering the scaling relation of the tuned covariance matrices, equations (28.42) and (28.43), the left sides of above equations (28.45) and (28.46) read

$$\begin{aligned} J^b(\delta\mathbf{x}) &= = \frac{1}{2}\delta\mathbf{x}^T[\mathbf{B}(s^b)]^{-1}\delta\mathbf{x} = \frac{1}{2}\delta\mathbf{x}^T\frac{\tilde{\mathbf{B}}^{-1}}{s^b}\delta\mathbf{x} \\ &= \frac{\tilde{J}^b(\delta\mathbf{x})}{s^b} \\ J^o(\delta\mathbf{x}) &= = \frac{1}{2}(\mathbf{d} - \mathbf{H}\delta\mathbf{x})^T[\mathbf{B}(s^o)]^{-1}(\mathbf{d} - \mathbf{H}\delta\mathbf{x}) = \frac{1}{2}(\mathbf{d} - \mathbf{H}\delta\mathbf{x})^T\frac{\tilde{\mathbf{B}}^{-1}}{s^b}(\mathbf{d} - \mathbf{H}\delta\mathbf{x}) \\ &= \frac{\tilde{J}^b(\delta\mathbf{x})}{s^b} \end{aligned}$$

Note that, although the cost function terms $\tilde{J}^b$ and $\tilde{J}^o$ in equations (28.47) and (28.47) are calculated using the untuned covariance matrices $\tilde{\mathbf{B}}$ and $\tilde{\mathbf{R}}$, the analysis increment $\delta\mathbf{x}$ is derived using the tuned covariance matrices $\mathbf{B}$ and $\mathbf{R}$:

$$\delta\mathbf{x} = \mathbf{B}(s^b)\mathbf{H}^T[\mathbf{H}\mathbf{B}(s^b)\mathbf{H}^T + \mathbf{R}(s^o)]^{-1}\mathbf{d} = \mathbf{K}(s^b, s^o)\mathbf{d} \tag{28.47}$$

The right sides of equations (28.45) and (28.46) depend on the traces of the Kalman gain matrix obtained from the tuned matrices as in equation (28.47).

The equations (28.45) and (28.46) relating the expectation value of the cost function to the trace of the Kalman gain matrix therefore read for the tuned matrices:

$$J^b(\delta\mathbf{x}) = \frac{s^b}{2} Tr(\mathbf{HK}(s^b, s^o)) \tag{28.48}$$

$$J^o(\delta\mathbf{x}) = \frac{s^o}{2} Tr(\mathbf{I}_p - \mathbf{HK}(s^b, s^o)) \tag{28.49}$$

Uncorrelated subparts: For more specific results, independent scaling coefficients can be assigned to subparts of $\tilde{\mathbf{B}}$ and $\tilde{\mathbf{R}}$ with no or only negligible correlation. This is an assumption which is reasonable at least for the observation errors. Let these subparts be

$$\begin{aligned} \tilde{\mathbf{B}}_k &= \mathbf{\Pi}_k^b \tilde{\mathbf{B}} \mathbf{\Pi}_k^{bT} \quad , \quad 1 \le k \le \nu_b \\ \tilde{\mathbf{R}}_l &= \mathbf{\Pi}_l^o \tilde{\mathbf{R}} \mathbf{\Pi}_l^{oT} \quad , \quad 1 \le l \le \nu_o \end{aligned}$$

with the projection operators $\mathbf{\Pi}_k^b$ and $\mathbf{\Pi}_l^o$:

$$\begin{aligned} \tilde{\mathbf{B}} &= \sum_{k=1}^{\nu_b} \mathbf{\Pi}_k^T \tilde{\mathbf{B}}_l \mathbf{\Pi}_k \\ \tilde{\mathbf{R}} &= \sum_{l=1}^{\nu_o} \mathbf{\Pi}_l^T \tilde{\mathbf{R}}_l \mathbf{\Pi}_l \end{aligned}$$

The corresponding subset of background and observation data is then given by

$$\begin{aligned} \mathbf{x}_k &= \mathbf{\Pi}_k^b \mathbf{x} \quad , \quad 1 \le k \le \nu_b \\ \mathbf{y}_l &= \mathbf{\Pi}_l^o \mathbf{y} \quad , \quad 1 \le l \le \nu_o \end{aligned}$$

The tuning relations for these subparts now read

$$\mathbf{B}_k = \mathbf{B}_k(\mathbf{s}_k^b) = s_k^b \tilde{\mathbf{B}}_k \quad , \quad 1 \le k \le \nu_b \tag{28.50}$$

$$\mathbf{R}_l = \mathbf{R}_l(\mathbf{s}_l^o) = s_l^o \tilde{\mathbf{R}}_l \quad , \quad 1 \le l \le \nu_o \tag{28.51}$$

with the sets of tuning coefficients

$$\begin{aligned} \mathbf{s}^b &= (s_1^b, ..., s_{\nu_b}^b) \\ \mathbf{s}^o &= (s_1^o, ..., s_{\nu_o}^o) \end{aligned}$$

Extended to uncorrelated subparts of the untuned covariance matrices $\tilde{\mathbf{B}}$ and $\tilde{\mathbf{R}}$ with ν_b background data and ν_o observations, the cost function for the individual subparts $\tilde{J}_k^b$ and $\tilde{J}_l^o$ of the untuned cost function $\tilde{J}$ read according to equation (28.44)

$$\begin{aligned} \tilde{J}^b(\delta\tilde{\mathbf{x}}) &= \sum_{k=1}^{\nu_b} \tilde{J}_k^b(\delta\tilde{\mathbf{x}}) = \frac{1}{2} \sum_{k=1}^{\nu_b} \delta\tilde{\mathbf{x}}_k^T \tilde{\mathbf{B}}_k^{-1} \delta\tilde{\mathbf{x}}_k \\ \tilde{J}^o(\delta\tilde{\mathbf{x}}) &= \sum_{l=1}^{\nu_o} \tilde{J}_l^o(\delta\tilde{\mathbf{x}}) = \frac{1}{2} \sum_{l=1}^{\nu_o} (\mathbf{d} - H\delta\tilde{\mathbf{x}})_l^T \tilde{\mathbf{R}}_\mathbf{l}^{-1} (\mathbf{d} - H\delta\tilde{\mathbf{x}})_l \end{aligned}$$

with the subparts of background and observation vectors obtained as

$$\begin{aligned} \delta\tilde{\mathbf{x}}_k &= \mathbf{\Pi}_k^b \delta\tilde{\mathbf{x}} \quad , \quad 1 \le k \le \nu_b \\ (\mathbf{d} - H\delta\tilde{\mathbf{x}})_l &= \mathbf{\Pi}_l^o (\mathbf{d} - H\delta\tilde{\mathbf{x}})_l \quad , \quad 1 \le l \le \nu_o \end{aligned}$$

The cost function J of the tuned assimilation system now reads

$$\begin{aligned} J^b(\delta\mathbf{x}(\mathbf{s})) &= \sum_{k=1}^{\nu_b} J_k^b(\delta\mathbf{x}) = \frac{1}{2}\sum_{k=1}^{\nu_b} \delta\mathbf{x}_k(\mathbf{s})^T \frac{\tilde{\mathbf{B}}_\mathbf{k}^{-1}}{s_k^b} \delta\mathbf{x}_k(\mathbf{s}) = \sum_{k=1}^{\nu_b} \frac{\tilde{J}_k^b(\delta\mathbf{x}(\mathbf{s}))}{s_l^b} \\ J^o(\delta\mathbf{x}(\mathbf{s})) &= \sum_{l=1}^{\nu_o} J_l^o(\delta\mathbf{x}(\mathbf{s})) = \frac{1}{2}\sum_{k=1}^{\nu_b} \delta\mathbf{x}_k(\mathbf{s})^T \frac{\tilde{\mathbf{B}}_\mathbf{k}^{-1}}{s_k^b} \delta\mathbf{x}_k(\mathbf{s}) + \frac{1}{2}\sum_{l=1}^{\nu_o} (\mathbf{d} - \mathbf{H}\delta\mathbf{x}(\mathbf{s}))_l^T \frac{\tilde{\mathbf{R}}_\mathbf{l}^{-1}}{s_l^o} (\mathbf{d} - \mathbf{H}\delta\mathbf{x}(\mathbf{s}))_l \\ &= \sum_{l=1}^{\nu_o} \frac{\tilde{J}_l^o(\delta\mathbf{x}(\mathbf{s}))}{s_l^o} \end{aligned}$$

Equations (28.48) and (28.49) now read for each statistically independent component of the cost function of the tuned system

$$J_k^b(\delta\mathbf{x}(\mathbf{s})) = \frac{s_k^b}{2} Tr(\mathbf{\Pi}_k^b \mathbf{H}\mathbf{K}(\mathbf{s})\mathbf{\Pi}_k^{bT}) \quad , \quad 1 \le k \le \nu_b \tag{28.52}$$

$$J_l^o(\delta\mathbf{x}(\mathbf{s})) = \frac{s_l^o}{2} Tr(\mathbf{I}_p - \mathbf{\Pi}_l^o \mathbf{H}\mathbf{K}(\mathbf{s})\mathbf{\Pi}_l^{oT}) \quad , \quad 1 \le l \le \nu_o \tag{28.53}$$

Fixed point algorithm: According to equations (28.52) and (28.53), the tuning coefficients obey the implicit equations

$$\begin{aligned} s_k^b &= \frac{2J_k^b\{\delta\mathbf{x}(\mathbf{s})\}}{Tr(\mathbf{\Pi}_k^b \mathbf{H}\mathbf{K}(\mathbf{s})\mathbf{\Pi}_l^{bT})} \quad , \quad 1 \le k \le \nu_b \\ s_l^o &= \frac{2J_l^o\{\delta\mathbf{x}(\mathbf{s})\}}{Tr(\mathbf{\Pi}_l^o(\mathbf{I}_p - \mathbf{H}\mathbf{K}(\mathbf{s})\mathbf{\Pi}_l^{oT}))} \quad , \quad 1 \le l \le \nu_o \end{aligned}$$

As $\mathbf{K}(\mathbf{s})$ and $\delta\mathbf{x}(\mathbf{s})$ both depend on the tuning coefficients $\mathbf{s}$, these relations are nonlinear. An approximate solution can be obtained by performing successive steps with the following iteration rule:

$$s_{k\,i+1}^b = \frac{2J_k^b\{\delta\mathbf{x}(\mathbf{s}_i)\}}{Tr(\mathbf{\Pi}_k^b \mathbf{H}\mathbf{K}(\mathbf{s}_i)\mathbf{\Pi}_k^{bT})} \tag{28.54}$$

$$s_{l\,i+1}^o = \frac{2J_l^o\{\delta\mathbf{x}(\mathbf{s}_i)\}}{Tr(\mathbf{\Pi}_l^o(\mathbf{I}_p - \mathbf{H}\mathbf{K}(\mathbf{s}_i)\mathbf{\Pi}_l^{oT}))} \tag{28.55}$$

Starting with a given value $\mathbf{s} = 1.0$ of the scaling coefficient, the right side of equations (28.54) and (28.55) is calculated, resulting in an updated value of the scaling coefficients. The cost function as well as the trace are determined as sums over the corresponding submatrices of $\mathbf{B}(\mathbf{s})$, $\tilde{\mathbf{B}}$ and $\mathbf{R}(\mathbf{s})$, $\tilde{\mathbf{R}}$. It is of importance to calculate the cost function with the analysis vector of the tuned system, but with the untuned matrices $\tilde{\mathbf{B}}$ and $\tilde{\mathbf{R}}$. The Kalman gain matrix, however, is determined with the tuned matrices $\mathbf{B}$ und $\mathbf{R}$

$$\mathbf{K}(\mathbf{s}_i) = \mathbf{B}(\mathbf{s}_i)[\mathbf{B}(\mathbf{s}_i) + \mathbf{R}(\mathbf{s}_i)]^{-1}$$

The equations (28.52) and (28.53) have trivial solution s$s_k^o = 0$ and $s_l^b = 0$. However, calculations performed with actual data sets of the 3dVar assimilations scheme never converged in these solutions.

Validation of tuning: The values of s_k^o and s_l^b represent the statistical character of the data set and are statistical quantities themselves. An estimate of the variance and reliability of the values of the scaling coefficients can be achieved by tuning the error covariance matrices with a data set of observations that is in ideal consistency with the specified $\tilde{\mathbf{B}}$ and $\tilde{\mathbf{R}}$ (see equations (28.40), (28.41)). To create the set of simulated data, the first guess data as used in the 3dVar is considered as the "true" background $\mathbf{x}^t$. The background actual data $\mathbf{x}$ is then constructed by

$$\mathbf{x}^b = \mathbf{x}^t + \tilde{\mathbf{B}}^{1/2}\zeta^b$$

using a n-dimensional random vector ζ^b with Gaussian distribution and unit variance and the square root of the background error covariance matrix $\tilde{\mathbf{B}}^{1/2}$. The simulated observation data is then obtained as

$$\mathbf{y} = \mathbf{H}\mathbf{x}^t + \tilde{\mathbf{R}}^{1/2}\zeta^o$$

with the p-dimensional random vector ζ^o with Gaussian distribution and unit variance. Within this setup, the statistical properties that are preconditions for the equation (28.5.2) to be valid are fullfilled: The simulated observations are consistent with the background and observation error statistics und have a covariance matrix $\tilde{\mathbf{R}}$. Therefore, the tuning algorithm is expected to yield values $\mathbf{s} \approx 1.0$ for the scaling coefficients. Deviations of the actual values from this expected one measure the statistical reliability of the scaling coefficients that can be determined using this set of data. The number of observation and background data and their horzontal distribution seem to have the strongest influence on the broadening of the distribution of the scaling coefficients. see also G.Desroziers and S.Ivanov, Q.J.R. Meteorol. Soc. (2001), **127**, pp. 1433-1452; Chapnik+al, Q.J.R. Meteorol. Soc. 2004, 130, p2253 and Zhuo Liu, M.Buehner and P.Gauthier, proceedings of the 21st Conference on Weather Analysis and Forecasting/17th Conference on Mumerical Weather Prediction, 15B.8.

Relation to maximum-likelihood method: There is a relation of the solution of equations (28.54) and (28.54) to the "maximum-likelihood" method for parameter estimation, given a set of observatiuonal data. (D.Dee, A.da Silva, Mon.Weather Rev. 1998, 124, p1822). It can be shown (B.Chapnik+al, Q.J.R. Meteorol. Soc. 2004,130,p2253) that the non-trivial solutions s_k^o and s_l^b of (28.54) and (28.54) maximise the likelihood-function.

Aspects of the tuning method in 3dVar For the application within the 3dVar assimilation scheme, a few minor adjustments of the tuning method are to be done.

- The specified background error covariance matrix $\tilde{\mathbf{B}}$ is readily available in the observation space. Therefore, the linearised observation operator $\mathbf{H}$ in above equations can simply replaced by the identity matrix $\mathbf{I}$. Furthermore, $\tilde{\mathbf{B}}$ and $\tilde{\mathbf{R}}$ are of equal

size, as the number of background data in observation space equals the number of observation data, $n = p$.

- In the 3dVar, observation errors are supposed to be statistically independent (not correlated) as long as they refer to
  - different observation operators:
    SYNOP, TEMP, AIREP, PILOT, PAOB, DRIBU, AMV, TOVS ...
  - different observed quantities:
    geopotential, temperature, relative humidity, wind (u,v)
  - different pressure level of the observation (actually implemented: TEMP main pressure levels)
- Within the background error covariance matrix $\tilde{\mathbf{B}}$, the quantities geopotential/temperature/realtive humidity/wind are supposed to be generally correlated (default). However, the correlation of temperature and relative humidity might not be significant. Optionally, these two quantities can be treated as statistically independent variable within the tuning algorithm. Similarly, an uncorrelated treatment of geopotential/temperature and wind data can be reasonable for certain areas (e.g. the tropics). A decoupling of these blocks can be obtained optionally.

Entry data of the tuning method:

In the module `mo_psas.f90`, the tuning routine `mo_err_tune.f90` is called with the arguments

`Rj_0`: observation error covariance matrix for error tuning, untouched by VQC

`HBHi`: forecast error covariances

`obs`: vector of observations distributed over boxes, `obs% o(:)% obs`

`yb`: vector of forecast (background,first guess) data distributed over boxes, `yb% s(:)% x`

`w_qc`: vector of VQC probability for correct data

`name`: file basename for output file

namelist TUNE_ERRORS

Namelist parameters for the tuning algorithm

```
real(wp)            :: lev_top        ! top height boundary
real(wp)            :: lev_bot        ! bottom height boundary
real(wp)            :: lat_n          ! latitude boundary north
real(wp)            :: lat_s          ! latitude boundary south
real(wp)            :: lon_w          ! longitude boundary west
```

```
  real(wp)              :: lon_e           ! longitude boundary east
  character(len=12)     :: type            ! observation type operator
  integer               :: obstype         ! observation type quantity
  logical               :: sep_geop_wind   ! T: no correlation bg geop/wind
  logical               :: sep_temp_hum    ! T: no correlation bg temp/hum
  logical               :: sep_temp_hum    ! T: no correlation bg temp/hum
  logical               :: ck_consist      ! consistency check
  logical               :: no_hor_corr     ! no horizontal correlations in B
  integer               :: n_iter          ! no. of iterations
  real(wp)              :: w_qcc           ! critical value of vqc statistical weight
```

The namelist can be read repeatedly. The different sets of namelist parameters are stored in an array of type `t_tune_errors`. Therefore, the setup for the tuning algorithm can be altered e.g. to successively tune the error statistics of the different data types that are assimilated in the 3dVar assimilation scheme.

data structures

```
type t_te_level
   real(wp)              :: lev_top        = rud     ! top height boundary
   real(wp)              :: lev_mid        = rud     ! mid height of level
   real(wp)              :: lev_bot        = rud     ! bottom height boundary
   real(wp),allocatable :: s_b(:)                    ! scaling coefficient bg, n_par_
   real(wp),allocatable :: s_o(:)                    ! scaling coefficient obs, n_par
   real(wp),allocatable :: diff_s_b(:)               ! size of last iteration step, r
   real(wp),allocatable :: diff_s_o(:)               ! size of last iteration step, r
   integer, allocatable :: n_b(:)                    ! accepted no. of bg, n_par_b ty
   integer, allocatable :: n_o(:)                    ! accepted no. of obs, n_par_o t
   integer               :: n_init         = iud     ! initial no. of obs
   integer               :: n_obs          = iud     ! accepted no. of obs
end type t_te_level


type t_tune_errors
   integer               :: obstype        = iud     ! observation type (int)
   logical               :: sep_geop_wind  = .false. ! T: no correlation bg geop/wind
   logical               :: sep_temp_hum   = .false. ! T: no correlation bg temp/hum
   logical               :: ck_consist     = .false. ! consistency check
   logical               :: no_hor_corr    = .false. ! no horizontal correlations in
   integer               :: n_iter         = iud     ! no. of iterations
   integer               :: ils            = iud     ! SYNOP land/sea flag
   real(wp)              :: lat_n          = rud     ! latitude boundary north
   real(wp)              :: lat_s          = rud     ! latitude boundary south
   real(wp)              :: lon_w          = rud     ! longitude boundary west
   real(wp)              :: lon_e          = rud     ! longitude boundary east
   integer               :: n_par_b        = iud     ! no. of bg type
```

```
    integer              :: n_par_o        = iud     ! no. of obs type
    integer              :: n_boxes        = iud     ! no. of boxes
    integer              :: nlev           = iud     ! no. of levels
    real(wp)             :: w_qcc          = rud     ! critical value of vqc statisti
    type(t_te_level)     :: lev(40)                  ! data on levels
    type(t_te_level)     :: average                  ! data average
  end type t_tune_errors
```

Steps of the tuning algorithm:

To determine the scaling coefficients, the following steps are performed successively:

1. Selection of observation data:
   (a) Read input data:
      i. background error covariance matrix in observation space $\tilde{\mathbf{B}} = \mathbf{H}\mathbf{B}_{physical}\mathbf{H}^T$ (derived from the background error covariance matrix in physical space $\mathbf{B}_{physical}$)
      ii. observation error covariance matrix (diagonal elements only) $\tilde{\mathbf{R}}$
      iii. first guess vector in observation space $H(\mathbf{x}^b)$
      iv. observation vector $\mathbf{y}$
      v. vector with first guess statistical weights $\mathbf{w}_{qcc}$
      vi. vector with ovservation quantity type **t_int**
   (b) Identification and elimination of observation data that are apparently identical (identical station name, observation type, latitude, longitude, observation height)
   (c) optional: selection of SYNOP land/sea observation
   (d) selection of observation with a VQC weight less than the critical value w_{qcc} specified in the namelist
   (e) Construct the block matrices of $\tilde{\mathbf{B}}_k$, $\tilde{\mathbf{R}}_l$ and the corresponding vectors $\mathbf{x}_k$, $\mathbf{y}_l$, $\mathbf{w}_{qcc_l}$, $\mathbf{t\_int}_l$
   (f) Determination of the number of observations of the individual observation types
      i. geopotential geop
      ii. temperature t
      iii. relative humidity h
      iv. wind speed u,v
2. Begin of iteration loop to solve implicit equation:
   (a) Initialization of the tuning coefficients $s_k^b = 1$, $s_l^o = 1$

(b) Start of the iteration loop to iteratively determine the tuning coefficients as solutions of the fixed point equation criterion for determination: fixed number of iteration steps

i. Scaling of matrices $\tilde{\mathbf{R}}_l$ and $\tilde{\mathbf{B}}_k$ with the actual value of tuning coefficients $\mathbf{s}_i$.
Obtain the scaled matrices $\mathbf{R}(\mathbf{s}_i)$, $\mathbf{B}(\mathbf{s}_i)$

ii. determine $[\mathbf{B}(\mathbf{s}_i) + \mathbf{R}(\mathbf{s}_i)]^{-1}$ applying function inverse in mo_matrix (singular value decomposition)

iii. determine Kalman gain matrix $\mathbf{K}(\mathbf{s}_i) = \mathbf{B}(\mathbf{s}_i) * [\mathbf{B}(\mathbf{s}_i) + \mathbf{R}(\mathbf{s}_i)]^{-1}$ applying function matmul(,)

iv. determine trace of $\mathbf{K}(\mathbf{s}_i)$ for each observation type by summing up the diagonal elements

v. determine value of cost function J in successive steps:

A. determine inverse $\tilde{\mathbf{B}}^{-1}$, $\tilde{\mathbf{R}}^{-1}$ of the untuned matrices $\tilde{\mathbf{B}}$, $\mathbf{R}$ applying the function inverse in mo_matrix (singular value decomposition)

B. determine innovation vector $\mathbf{d} = \mathbf{y} - H(\mathbf{x}^b)$

C. determine the analysis vector $\mathbf{y}(\mathbf{s}_i) = \mathbf{K}(\mathbf{s}_i) * (obs - fg) + fg$ applying function matmul(,)

D. right hand side of cost function: Multiply $\mathbf{B^{-1}}$, $\mathbf{R^{-1}}$ with difference vectors $ana - fg$ and $ana - obs$, respectively
This is done by multiplication of the complete matrices without any distinction of the observation types

E. left hand side of cost function: Multiplication of the corresponding vector from step 4 with $ana - fg$ and $ana - obs$, rescectively. Scale with 0.5.
In this step, distinguish between different observation types. Results in a cost function for each observation type.

vi. Update the values of the tuning coefficients by equation (**??**), using cost functions of traces, for each observation type

End of iteration loop

3. Write into file tune_errors.info: tuning coefficients, number of observation/bachground data used, convergence properties (absolute value of last iteration step)

28.5.3 Tuning the error statistics in 3dVar: Results

Tuning og observation errors: Based of the 3dVar reference experiment 5293 with untuned error statistics, a first set of scaling coefficients is determined to optimise the specified observation errors. The tuning algorithm is applied to sets of observation and background data of a single analysis. The size of the data sets for each of the observation quantities geopotential, temperature, relative humidity and wind speed is of about 200

type	TEMP	TEMP	AIREP	AIREP	PILOT	SATOB	SATOB	SATOB
parameter	t	u,v	t	u,v	u,v	u,v TR	u,v NH, SH	u,v MOD
ass. system	IFS	OI	IFS	OI	OI	IFS	IFS	IFS
10 hPa	2.5 (1.0)	3.0 (1.0)	2.2 (1.0)	4.0 (1.0)	3.0 (1.0)	5.7 (1.0)	5.7 (1.0)	5.7 (1.0)
20 hPa	2.2 (1.0)	2.5 (1.0)	2.0 (1.0)	4.0 (1.0)	2.5 (1.0)	5.0 (1.0)	5.0 (1.0)	5.0 (1.0)
30 hPa	2.0 (1.0)	2.0 (1.0)	1.8 (1.0)	4.0 (1.0)	2.0 (1.0)	5.0 (1.0)	5.0 (1.0)	5.0 (1.0)
50 hPa	1.9 (1.0)	2.0 (1.0)	1.6 (1.0)	4.0 (1.0)	2.0 (1.0)	5.0 (1.0)	5.0 (1.0)	5.0 (1.0)
70 hPa	1.8 (1.0)	2.0 (1.0)	1.5 (1.0)	4.0 (1.0)	2.0 (1.0)	5.0 (1.0)	5.0 (1.0)	5.0 (1.0)
100 hPa	1.7 (1.0)	2.0 (1.0)	1.4 (1.0)	4.0 (1.0)	2.2 (1.0)	5.0 (1.0)	5.0 (1.0)	5.0 (1.0)
150 hPa	1.6 (1.0)	2.2 (1.0)	1.4 (1.0)	4.0 (1.0)	2.4 (1.0)	5.0 (1.0)	5.0 (1.0)	5.0 (1.0)
200 hPa	1.5 (1.0)	2.4 (1.0)	1.4 (1.0)	4.0 (1.0)	3.2 (1.0)	5.0 (1.0)	5.0 (1.0)	5.0 (1.0)
250 hPa	1.5 (1.0)	3.2 (1.0)	1.3 (1.0)	4.0 (1.0)	3.2 (1.0)	5.0 (1.0)	5.0 (1.0)	5.0 (1.0)
300 hPa	1.4 (1.0)	3.2 (1.0)	1.3 (1.0)	4.0 (1.0)	3.8 (1.0)	5.0 (1.0)	5.0 (1.0)	5.0 (1.0)
400 hPa	1.2 (1.0)	3.8 (1.0)	1.2 (1.0)	3.5 (1.0)	3.6 (1.0)	4.3 (1.0)	4.3 (1.0)	4.3 (1.0)
500 hPa	1.2 (1.0)	3.6 (1.0)	1.2 (1.0)	3.0 (1.0)	3.4 (1.0)	3.5 (1.0)	3.5 (1.0)	3.5 (1.0)
700 hPa	1.3 (1.0)	3.4 (1.0)	1.2 (1.0)	3.0 (1.0)	2.5 (1.0)	2.0 (1.0)	2.0 (1.0)	2.0 (1.0)
850 hPa	1.5 (1.0)	2.4 (1.0)	1.3 (1.0)	3.0 (1.0)	2.4 (1.0)	2.0 (1.0)	2.0 (1.0)	2.0 (1.0)
1000 hPa	1.7 (1.0)	2.0 (1.0)	1.4 (1.0)	3.0 (1.0)	2.0 (1.0)	2.0 (1.0)	2.0 (1.0)	2.0 (1.0)

Table 28.4: Untuned observation errors, experiments 5393, 5322, 5338/5405,5339, 5340

type	TEMP	TEMP	AIREP	AIREP	PILOT	SATOB	SATOB	SATOB
parameter	t	u,v	t	u,v	u,v	u,v TR	u,v NH, SH	u,v MOD
ass. system	IFS	OI	IFS	OI	OI	IFS	IFS	IFS
10 hPa	1.3 (0.5)	2.1 (0.7)	1.1 (0.5)	2.8 (0.7)	2.1 (0.7)	4.0 (0.7)	4.0 (0.7)	4.0 (0.7)
20 hPa	1.1 (0.5)	1.8 (0.7)	1.0 (0.5)	2.8 (0.7)	1.8 (0.7)	3.5 (0.7)	3.5 (0.7)	3.5 (0.7)
30 hPa	1.0 (0.5)	1.4 (0.7)	0.9 (0.5)	2.8 (0.7)	1.4 (0.7)	3.5 (0.7)	3.5 (0.7)	3.5 (0.7)
50 hPa	1.0 (0.5)	1.4 (0.7)	0.8 (0.5)	2.8 (0.7)	1.4 (0.7)	3.5 (0.7)	3.5 (0.7)	3.5 (0.7)
70 hPa	0.9 (0.5)	1.4 (0.7)	0.7 (0.5)	2.8 (0.7)	1.4 (0.7)	3.5 (0.7)	3.5 (0.7)	3.5 (0.7)
100 hPa	0.9 (0.5)	1.4 (0.7)	0.7 (0.5)	2.8 (0.7)	1.4 (0.7)	3.5 (0.7)	3.5 (0.7)	3.5 (0.7)
150 hPa	0.8 (0.5)	1.5 (0.7)	0.7 (0.5)	2.8 (0.7)	1.5 (0.7)	3.5 (0.7)	3.5 (0.7)	3.5 (0.7)
200 hPa	0.8 (0.5)	1.7 (0.7)	0.7 (0.5)	2.8 (0.7)	1.7 (0.7)	3.5 (0.7)	3.5 (0.7)	3.5 (0.7)
250 hPa	0.8 (0.5)	2.2 (0.7)	0.6 (0.5)	2.8 (0.7)	2.2 (0.7)	3.5 (0.7)	3.5 (0.7)	3.5 (0.7)
300 hPa	0.7 (0.5)	2.2 (0.7)	0.6 (0.5)	2.8 (0.7)	2.2 (0.7)	3.5 (0.7)	3.5 (0.7)	3.5 (0.7)
400 hPa	0.6 (0.5)	2.7 (0.7)	0.6 (0.5)	2.5 (0.7)	2.7 (0.7)	3.0 (0.7)	3.0 (0.7)	3.0 (0.7)
500 hPa	0.6 (0.5)	2.5 (0.7)	0.6 (0.5)	2.1 (0.7)	2.5 (0.7)	2.5 (0.7)	2.5 (0.7)	2.5 (0.7)
700 hPa	0.7 (0.5)	2.4 (0.7)	0.6 (0.5)	2.1 (0.7)	2.4 (0.7)	1.4 (0.7)	1.4 (0.7)	1.4 (0.7)
850 hPa	0.8 (0.5)	1.7 (0.7)	0.7 (0.5)	2.1 (0.7)	1.7 (0.7)	1.4 (0.7)	1.4 (0.7)	1.4 (0.7)
1000 hPa	0.9 (0.5)	1.4 (0.7)	0.7 (0.5)	2.1 (0.7)	1.4 (0.7)	1.4 (0.7)	1.4 (0.7)	1.4 (0.7)

Table 28.5: 1st set of tuned observation errors, experiments 5345, 5346, 5364, 5406/5477, 5417, 5419, 5420, 5429

type	TEMP	TEMP	AIREP	AIREP	PILOT	SATOB	SATOB	SATO
parameter	t	u,v	t	u,v	u,v	u,v TR	u,v NH, SH	u,v MOI
ass. system	IFS	OI	IFS	OI	OI	IFS	IFS	IFS
10 hPa	1.8 (0.7)	3.0 (1.0)	1.5 (0.7)	4.0 (1.0)	3.0 (1.0)	2.9 (0.5)	4.0 (0.7)	4.6 (0.8
20 hPa	1.5 (0.7)	2.5 (1.0)	1.4 (0.7)	4.0 (1.0)	2.5 (1.0)	2.5 (0.5)	3.5 (0.7)	4.0 (0.8
30 hPa	1.4 (0.7)	2.0 (1.0)	1.3 (0.7)	4.0 (1.0)	2.0 (1.0)	2.5 (0.5)	3.5 (0.7)	4.0 (0.8
50 hPa	1.3 (0.7)	2.0 (1.0)	1.1 (0.7)	4.0 (1.0)	2.0 (1.0)	2.5 (0.5)	3.5 (0.7)	4.0 (0.8
70 hPa	1.8 (0.7)	2.0 (1.0)	1.1 (0.7)	4.0 (1.0)	2.0 (1.0)	2.5 (0.5)	3.5 (0.7)	4.0 (0.8
100 hPa	1.7 (0.7)	2.0 (1.0)	1.0 (0.7)	4.0 (1.0)	2.2 (1.0)	2.5 (0.5)	3.5 (0.7)	4.0 (0.8
150 hPa	1.6 (0.7)	2.2 (1.0)	1.0 (0.7)	4.0 (1.0)	2.4 (1.0)	2.5 (0.5)	3.5 (0.7)	4.0 (0.8
200 hPa	1.5 (0.7)	2.4 (1.0)	1.0 (0.7)	4.0 (1.0)	3.2 (1.0)	2.5 (0.5)	3.5 (0.7)	4.0 (0.8
250 hPa	1.5 (0.7)	3.2 (1.0)	0.9 (0.7)	4.0 (1.0)	3.2 (1.0)	2.5 (0.5)	3.5 (0.7)	4.0 (0.8
300 hPa	1.4 (0.7)	3.2 (1.0)	0.9 (0.7)	4.0 (1.0)	3.8 (1.0)	2.5 (0.5)	3.5 (0.7)	4.0 (0.8
400 hPa	1.2 (0.7)	3.8 (1.0)	0.8 (0.7)	3.4 (0.96)	3.6 (1.0)	2.2 (0.5)	3.0 (0.7)	3.5 (0.8
500 hPa	1.2 (0.7)	3.6 (1.0)	0.8 (0.7)	2.7 (0.91)	3.4 (1.0)	1.8 (0.5)	2.5 (0.7)	2.8 (0.8
700 hPa	1.3 (0.7)	3.4 (1.0)	0.8 (0.7)	2.5 (0.83)	2.5 (1.0)	1.0 (0.5)	1.4 (0.7)	1.6 (0.8
850 hPa	1.5 (0.7)	2.4 (1.0)	0.9 (0.7)	2.3 (0.76)	2.4 (1.0)	1.0 (0.5)	1.4 (0.7)	1.6 (0.8
1000 hPa	1.7 (0.7)	2.0 (1.0)	1.0 (0.7)	2.1 (0.70)	2.0 (1.0)	1.0 (0.5)	1.4 (0.7)	1.6 (0.8

Table 28.6: 2nd set of tuned observation errors, experiments 5447 ff

observations. The tuning procedure results in a set of scaling coefficients given in table 28.5. For reference, table 28.4 shows the values of the untuned observation error statistics.

The analysis fields of the experiment 5345 with tuned error statistics generally deviate less from the corresponding IFS and GME fields compared to the untuned statistics, most prominetly in the tropics.

However, this impact seems to be induced by an implicitely stricter variational quality control: By reducing the observation error, the scaling factor used to transform the set innovations to a standardised distribution with unitary width within the variational quality control, increases. Since the cutoff parameters of the quality control are not increased in the same way, observations with large departures from the background are additionally rejected. This selection takes place even though the background and observation data are extracted before entering the variational quality check, as the first guess check (done before error tuning) is performed in the same way.

In the following experiment 5364 the VQC-bound are increased to avoid this effect. As a consequence, the positive impact of the tuning in experiment 5345 is more or less neutralised, in some regions it turned even slightly into the opposite. But as the background error is yet not optimised, only limited conclusions can be drawn from this results.

Tuning of background error: The background error is to be tuned along with the observation errors to guarantee the balance of the analysis between first guess and observations in the course of the variational assimilation process. A one-sided reduction of observation errors would result in a shift of the final analysis towards the observations, away from the first guess. Additionally, the application of the tuning methods to sets of background and observation data suggest a reduction of the background error. Different

to the observation errors which are specified in the 3dVar source code, the background error of the analysis of a particular date is derived from the analysis error of the assimilation of the preceeding date. The analysis error is relaxed towards a climatological mean error, depending on the time intervall between succeeding assimilations. The relaxation is triggered by a zonally dependent relaxation parameter (for documentation, see e.g. Research Manual 1, ECMWF Data Assimilation, Scientific Documentations, chapter 2.3).

Although a reduction of the observation errors has a reduced analysis error as a consequence, the effect of tuning the observation error statistics turned out to be not sufficient to yield an adequate reduction of the background error. Also, a straightforward scaling of the analysis error turned out to be no favourable way, since it results in a successive reduction of the background error to smaller and smaller values.

Contrary to this, an increase of the relaxation parameter yields a relatively smooth reduction of the error, that is temporally constant after a sufficiently long onset time (ca. 4 days). In a series of succesive assimilations without (forecasts) a scaling of the relaxation parameter yields a reduction of the background error by a factor of about 0.8. For the southern hemisphere, a scaling of the relaxation parameter by 2 seems sufficient. An updated set of tuned observation errors is determined, based on the reduced first guess error obtained with the increased relaxation parameters (see table 28.6).

Generally, the values of the scaling coefficients tend to be slightly larger than the values of the first tuning process of experiments 5345. This effect is probably caused by the tendency of the tuning method to shift the existing correlations of the data set according to the balance of the specified error covariances: The reduction of the background error causes an increase in the estimated observation errors. Therefore, an update of the tuned observation error statistics is inevitable. Additionally, the tuning of the SATOB wind error covariances is done in a more detailed way. The tuning algorithm suggests smaller observation errors of the SATOB wind observations from geostationary satellite in the tropics, in contrast to the error values for observations in the extratropical latitudes. Observations from polar circulating satellites are supposed to have a slightly larger value of the observation error, but still less than the value specified for the IFS.

A series of experiments with background and observation errors specified in this way are performed, mostly testing technical modifications.

Background errors with the NMC fitted vertical covariance model: Starting with experiment 5610, a NMC fitted covariance function is used for modelling the 3-dimensional background error covariances within the 3dVar. Compared to prior vertical covariance function of the 3dVar, the new model is characterised by broader correlations and generally higher values of the background errors themselves in the tropospheric height levels. In the stratospheric regime, the values of the NMC background error covariances are generally smaller than the OI-covariance model used so far in the 3dVar. As the introduction of the new covariance model changes the overall background error statistics, new values for the global scaling coefficients of the background error covariance matrix have to be determined.

Since the tuning algorithms proved to be less effective for the optimisation of the background error covariances (the results of the tuning method show quite diverse results for the estimation of the overall background error, depending on the observation operator

Experiment id.	relaxation parameter			climat. error
	NH	TR	SH	
5610	4*144	4*48	2*144	OI
5630	2*144	2*48	1*144	OI
5631	1*144	1*48	0.5*144	OI
5650	1*144	1*48	0.5*144	NMC
5652	2*144	2*48	2*144	NMC

Table 28.7: Tuning background error: Values of relaxation parameters

and the area), the fitting is done in a more ad hoc way, based upon the values applied in the experiments 5490. As in foregoing experiments, the background error covariances are tuned by applying scaling coefficients to the time parameters that triggers the relaxation of the background error covariances towards a climatological mean error. These parameters have the values 144 h in NH and SH and 48 in TR with a smooth transition between the zonal areas. An increase of the relaxation parameter results in a reduction of the background error in the particular region. These relaxation parameters have been scaled by various sets of factors as given in table 28.7.

Since the tropospheric background errors of the NMC method are generally large compared to the backgroud error values employes so far, it is expected that a reduction of the error, i.e. an increase of the relaxation parameter, is necessary in order to describe the data statistics properly. As an indicator to judge the quality of the error tuning, the mean deviation of the 3dVar analysis to the IFS analysis in monitored (average over 24 days). The fields considered are geopotential at 500 hPa, z1000, relative humidity at 700 hPa as well as temperature, wind u and v at 850 250 and 50 hPa. The monitoring included the bias, standard deviation and rms of the deviances. Comparing the results of the three experiments 5610, 6530 and 6531, a general decrease of standard deviation and rms with increasing background error (i.e. decreasing relaxation parameter) is noticable.

However, the climatological error used to model the increase of the analysis error with time depends on the background error covariance model, and has to be adapted additionally to the tuning of the background error. The values of the climatological error as used in the experiments 6510, 6530 and 6531 is generally too small for the NMC fitted errors of the new vertical covariance model. Therefore, the (old) climatological error is not significantly larger than the analysis error and the value of the relaxation parameter has nearly no influence on the increase of the analysis error with time. Therefore, a new climatological error is determined, based on the NMC fitted background error vertical covariances.

Two experiments (5650 and 5652) are performed using the updated climatological error. A monitoring of time averages of the same fields and parameters of deviance to the IFS analysis as for the experiments 6510, 5630 and 5631 is performed. It shows that generally the 3dVar analysis is closer to the IFS analysis in experiment 5650 in the northern hemisphere and in experiment 5652 in the southern hemisphere. In the tropics neither experiment showed a definitely better or worse performance.

Principally, a combination of the background error setup of experiment 5650 in NH and of experiment 5652 in SH is possible. However, this solution would lack a physically con-

sistent foundation, as northern and southern hemisphere should be physically described in the same way. Different relaxation parameters for both hemispheres would result in first guess error, which are of about the same value in NH and SH, which is contrary to the much higher density of observations data in the northern hemisphere. Therefore, the background error setup of the experiment 5652 is chosen as a basis for the following experiments. Although the NH performs better in 5650 than in 5652, the overall deviance to IFS is in 5652 generally still better than of the OI reference experiment.

Application of the tuning method to RTTOV radiances: The method to optimise the error statistics is applicable not only to conventional data, but also to satellite data, such as RTTOV radiances. A scaling coefficient is determined for the background error covariance matrix, with an independent treatment of geopotential/temperature/wind and relative humidity. As for the observation errors, an independent scaling coefficient is assigned to each assimilated RTTOV channel. The sensitive regions of different RTTOV channels are located in different height levels (at least as the dependence on the temperature is concerned). The profile of tuning coefficients of the different channels, suitably ordered, therefore corresponds to the vertical profile of tuning coefficients of the conventional observation operators like TEMP, AIREP, PILOT etc. Storage availability limits the data used to one satellite and an single device (AMSU-A). The first guess used for the tuning algorithm is taken from experiment 5668, date 2006051400.

Tuning of suitably modified data sets indicate that, contrary to the conventional observation operators, the statistics of the radiances seems to be affected by a general bias as well as by correlations of the observation errors. A first modification consists of the elimination of the horizontal correlations between different RTTOV observation spots. By doing this, the 3dVar performs essentially a 1D-Var assimilation for each observation spot. Application of the tuning method to the modified data set resulted in an increase in the estimated observation errors. This result is independent from the thinning used (ni=32, 64). The error tuning method has the property to react with a decrease of the estimated error values if correlation in the observation errors are introduced (B.Chapnik+al, Q.J.R. Meteorol. Soc. 2004, 130, p2253). Therefore, for the RTTOV observation data, a correlation of observation errors seems probable.

The background fields of the 3dVar differ from the 1D-Var background fields as used to specify the bias correction. The application of the 1D-Var bias correction to the 3dVar therefore results in an additional bias, which is located in the stratospheric levels. This can be avoided by a modification of the RTTOV observation data in such a way that the resulting innovations (obs-fg) correpond to a correctly bias corrected background field. The effect of this modified bias correction can be read off the results of the applied tuning algorithm: As a deficient bias correction appears like a long-distance horizontal correlation of the RTTOV observation data, generally resulting in an underestimation of the observation errors, a more adequate bias correction should yield in higher values for the tuning coefficients. This is in fact the case for the RTTOV channels with sensitivity maxima close to the stratospheric levels (channels 9,10, 13 of AMSU-A). Channel 312 responded in an indefinite way to the tuning, which may be related to general problems with this channel's observation data.

Part V

Test Cases

Part **Test Cases** is completely empty! As soon as testcases are available, they have to be described here **AND** in the User Guide.

Part VI

Appendix

Chapter 29

Writing Guideline

The LaTeX-sources of this documentation are available in the repository

```
git@git.mpimet.mpg.de:dace_doc.git
```

and can be checked out, edited and updated analogously to the `dace`-code (Chapter 12).

Structure

The main document is `dace.tex`. It sets up the document and the packages to be used. It includes `.tex`-files for the five parts of the docu, which themselves include further `.tex`-files for the chapters.

Compiling

The following commands compile the document with all links and references:

```
pdflatex dace
makeglossaries dace
pdflatex dace
pdflatex dace
```

Figure

Figures are stored in the directory `/figs` as `eps`- and `png` files. `pdflatex` uses by default the `png` files.

It is possible to use also the `eps`-figures for `pdflatex` using `eps-to-pdf` first when compiling the package.

LaTeX packages

The following packages need to be present in the Latex-distribution (e.g. `texlive`) to compile:

```
\usepackage{hyperref}
\usepackage{graphicx}
\usepackage{amsmath}
\usepackage{amsfonts}
\usepackage{amssymb}
\usepackage{units}
\usepackage{longtable}
\usepackage{html}
\hypersetup{pdfpagelabels}
\usepackage{placeins}
\usepackage{glossaries}
\usepackage{todonotes}
```

todo-notes

The `todonotes`-package enables to include remarks and notes where the docu is incomplete or something has to be done otherwise. The usage of the package (and how to enable/disable the todo-notes) is documented in the main file `dace.tex` or at `http://www.tex.ac.uk/ctan/macros/latex/contrib/todonotes/todonotes.pdf`.

glossaries

Some acronyms in the docu use the `glossaries`-package which defines the acronyms centrally in one file `dace_glossaries` and then just pastes them at the places where they are used. A good documentation for this package is available at `http://en.wikibooks.org/wiki/LaTeX/Glossary`.

Bibliography

[1] J. L. Anderson and S. L. Anderson. A monte carlo implementation of the nonlinear filtering problem to produce ensemble assimilations and forecasts. *Mon. Wea. Rev.*, 127(12):2741–2758, Dec. 1999.

[2] E. Andersson and H. Jarvinen. Variational quality control. *Q. J. R. Meteorol. Soc.*, 125(554):697–722, 1999.

[3] C. H. Bishop and Z. Toth. Ensemble Transformation and Adaptive Observations. *Journal of the Atmospheric Sciencies*, 56:1748–1765, 1999.

[4] M. Buehner and M. Charron. Spectral and spatial localization of background-error correlations for data assimilation. *Q.J,R.Meteorol.Soc*, 133:615–630, 2007.

[5] R. Daley and E. Barker. Navdas source book 2000. Technical report, Naval Research Laboratory, Marine Meteorology Division, Montery, CA 93943-5502, 2000.

[6] G. Desroziers, L. Berre, B. Chapnik, and P. Poli. Diagnosis of observation, background and analysis-error statistics in observation space. *Quarterly Journal of the Royal Meteorological Society*, 131:3385–3396, 2005.

[7] I. Dharssi, A. Lorenc, and N. Ingleby. Treatment of gross errors using maximum probability-theory. *Q. J. R. Meteorol. Soc.*, 118(507):1017–1036, 1992.

[8] ECMWF Research Department. *Research Manual 1 - ECMWF Data Assimilation Scientific Documentation*. ECMWF, 3rd edition, 1992.

[9] H. W. Engl, A. Neubauer, and M. Hanke. *Regularization of Inverse Problems*. Springer, 2003.

[10] G. Evensen. Sequential data assimilation with a nonlinear quasi-geostrophic model using Monte Carlo methods to forecast error statistics. *Journal of Geophysical Research*, 99:10143–10162, 1994.

[11] G. Gaspari and S. E. Cohn. Construction of correlation functions in two and three dimensions. *qjrms*, 125:723–757, 1999.

[12] S. J. Greybush, E. Kalnay, T. Miyoshi, K. Ide, and B. R. Hunt. Balance and Ensemble Kalman Filter Localization Techniques. *Monthly Weather Review*, 139:511–522, 2011.

[13] T. M. Hamill, J. S. Whitaker, and C. Snyder. Distance-Dependent Filtering of Background Error Covariance Estimates in an Ensemble Kalman Filter. *Monthly Weather Review*, 129:2776–2790, 2001.

[14] P. L. Houtekamer and H. L. Mitchell. Data assimilation using an ensemble kalman filter technique. *Mon. Wea. Rev.*, 126(3):796–811, Mar. 1998.

[15] P. L. Houtekamer and H. L. Mitchell. A Sequential Ensemble Kalman Filter for Atmospheric Data Assimilation. *Mon. Wea. Rev.*, 129:123–137, 2001.

[16] B. R. Hunt, E. J. Kostelich, and I. Szunyogh. Efficient data assimilation for spatiotemporal chaos: A local ensemble transform Kalman filter. *Physica D*, 203:112–126, 2007.

[17] J. Kaipio and E. Somersalo. *Statistical and computational inverse problems*, volume 160 of *Applied Mathematical Sciences*. Springer, 2005.

[18] E. Kalnay. *Atmospheric Modeling, Data Assimilation and Predictability*. Cambridge University Press, 2003.

[19] H. Li, E. Kalnay, and T. Miyoshi. Simultaneous estimation of covariance inflation and observation errors within an ensemble kalman filter. *Quarterly Journal of the Royal Meteorological Society*, 135(639):523–533, 2009.

[20] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer Series in Operations Research. Springer, 1999.

[21] D. Patil, B. R. Hunt, E. Kalnay, J. A. Yorke, and E. Ott. Local Low Dimensionality of Atmospheric Dynamics. *Physical Review Letters*, 86:5878–5881, 2001.

[22] C. Schraff and R. Hess. A description of the nonhydrostatic regional cosmo-model part iii: Data assimilation, Feb. 2012.

[23] S.-C. Yang, E. Kalnay, B. Hunt, and N. E. Bowler. Weight interpolation for efficient data assimilation with the local ensemble transform kalman filter. *Q.J.R. Meteorol. Soc.*, 135(638):251–262, Jan. 2009.