Use of the 2D structure of GOM-arrays for the assimilation of radar reflectivities in the AROME model at Météo-France -

Eric Wattrelot (CNRM/GMAP)

11 avril 2012

1 A 1D method to get relative humidity retrievals from radar reflectivities : why and how ?

Radar reflectivities are not directly assimilated in the 3DVar AROME. For several reasons. Firstly, the coding of the tangent linear of Arome's microphysics is difficult, and would require the linearization of discontinuous functions (processes with thresholds). Secondly, the observable quantity, reflectivity is calculated starting from the specific contents of hydrometeors ($q_{\mathbf{r}}$, $q_{\mathbf{s}}$, $q_{\mathbf{g}}$). Thus, one should correct precipitations in the analysis. These precipitations would fall quickly and would not be adjusted with other quantities such as humidity, temperature etc., if those are not corrected at the same time. The idea indeed was to retrieve columns of observed reflectivities as columns of relative humidity, then to assimilate these columns in the 3DVar. Instead of using a 1DVar, a 1D Bayesian inversion has been implemented. The chosen solution consists in considering that the model is able to create profiles of coherent moisture and reflectivity and to exploit this capability for deriving relative humidity profiles. The columns of observed reflectivities are compared with columns of simulated reflectivities. The closer a simulated column with an observed one, more likely the column of associated moisture will be similar to the column of observed moisture. It appears interesting to use columns of the model close to the observed one : this is more likely to help moving a poorly located precipitating system closer to observation.

Mathematically, for each column of reflectivities observed $\boldsymbol{y_Z}$, a column of relative humidity pseudo-observation $\boldsymbol{y}_{\mathbf{po}}$ is calculated in the following way :

$$\boldsymbol{y}_{\mathbf{po}} = \sum_i \boldsymbol{x_i} \frac{\exp\left[-\frac{1}{2} J_i(\boldsymbol{x_b})\right]}{\sum_j \exp\left[-\frac{1}{2} J_j(\boldsymbol{x_b})\right]} \, , \tag{1}$$

where

$$J_i(\boldsymbol{x_b}) \doteq {}^{\mathrm{t}}\left[\boldsymbol{y_Z} - \boldsymbol{H}_{Z,i}(\boldsymbol{x_b})\right] \mathbf{R}_{\mathbf{Z}}^{-1} \left[\boldsymbol{y_Z} - \boldsymbol{H}_{Z,i}(\boldsymbol{x_b})\right] \, , \tag{2}$$

where $\boldsymbol{x_b}$ is the model state vector, the $\boldsymbol{x_i}$ are the columns of relative humidity from the model in the vicinity of the observed column of reflectivities, $\boldsymbol{H}_{Z,i}$ is the observation operator of reflectivity co-located with the model column $\boldsymbol{x_i}$ (with the characteristics of the radar beam at the observation point $\boldsymbol{y_Z}$). $\mathbf{R}_{\mathbf{Z}}$ is the covariance matrix of observation error and observation operator error.

In pratical terms, from Eq. (1), the following fields are needed, in order to compute each column of pseudo-observation of relative humidity :

1. vertical column of observed reflectivity ($\boldsymbol{y_Z}$),
2. vertical columns of the model relative humidity in the vicinity of the observed column ($\boldsymbol{x_i}$),
3. vertical columns of the simulated reflectivities corresponding to the columns of relative humidity ($\boldsymbol{H}_{Z,i}(\boldsymbol{x_b})$). In order to compute these last fields, columns of model hydrometeors are needed. The choice has been made to store these fields in the 2D-GOM arrays associated to each reflectivity observation.

2 Dataflow of the 2D-GOM and specific points linked to radar assimilation

At the beginning of the development of the assimilation of radar reflectivities, the choice has been made to use the 2D structure of the GOM-arrays of model fields which are interpolated on each observation point. These fields are thus co-located at the observation points. This new structure allows indeed, at each point of observation, to store a vicinity of model profiles interpolated horizontally on an arbitrarily selected geographical vicinity. Thus, the knowledge of model fields in the vicinity of an observation can be used when computing the simulated reflectivities with the observation operator. This approach allows to compute the 1D-Bayesian retrievals on the vertical part of the observation operator, and retrieve the pseudo-profile of relative humidity in one go.

The main disadvantage of this method is to be expensive in memory, since it requires to store bulky 2D-GOM arrays.

The algorithm can be described as follows :

1. setup of observations and of profiles : initialization of the tables and allocation for the use of observations and of model counterparts interpolated at observation locations.
   (a) from SU0YOMA/SUDIM1/SUDIMO/OBADAT : call to RD_OBS_BOXES which reads the value of the number of profiles NOBSPROFS per observation type (at the same time as it initializes NOBTOT, the number of observations per timeslot, and the total number of profiles NOBPROF) in namelist NAMNPROF ; then call to SUOBSADDR which sets the number of arrays of model variables at observation locations, both for 1D-GOM and 2D-GOM.

   => no modification, except define NOBSPROFS for the radar observation type in NAMNPROF.

   (b) from SU0YOMA/SUALLO/SUALOBS : call to SUGOMS.F90 which computes the sizes of the arrays of model variables at observation points. SUGOMS uses GOMS_MIX.F90. Some updates have been done in order to introduce new arrays for hydrometeor variables : rain, snow and graupel).

   => Introduction of new GOM-arrays for hydrometeor variables : rain, snow and graupel in SUGOMS and GOMS_MIX. Large development of GOM2D in order to handle all the variables required for the radar observation operator, similar to was for GOM1D.

   It is worth to mention two problems that were encountered here :

      i. first, it was not possible with Meteo-France's former Fujitsu VPP5000 compiler to pass unallocated arrays as arguments. The problem was first found in the call to GOMCONSTRB in sugoms.F90 with the argument GOMDATUA.

      A simple arbitrary allocation before the call to GOMCONSTRB has been implemented :

      ```
      !* allocate gomdatua before using this array in gomconstr
      ALLOCATE(GOMDATUA(0))
      ```

      (The first call allocates precisely GOMDATUA array with the number of 1D-GOM) This problem should no longer exist on NEC computer.

      ii. Some fields, including GOMGFL fields (linked to upper air arrays within LGOMUA) were not well constructed after GOMCONSTRB. For example, YGOMUA_2D(IMAPOMM)%LS was badly filled for the radar obstype after GOMCONSTR_2D, despite the fact that activating the corresponding GFL should have activated this key in GOMCONSTRUC_2D through the lines below (in goms_mix.F90) :

      ```
      IF (YS%LACTIVE) LGOMUA(NRADAR, YGOMV%MS) =.TRUE.
      ```

and

```
IF(ANY(LGOMUA( :,YGOMV%MS))) THEN
NGOMGFL = NGOMGFL+1
...
```

and

```
MGOMGFL(YGOMV%MS) = NGOMGFL
LGOMGFL( :,NGOMGFL) = LGOMUA( :,YGOMV%MS)
...
```

and finally in sugoms.F90 :

```
IF(MGOMGFL(YGOMV%MS) == JGFL) THEN
YDGOMUA%LS = .TRUE.
IS = YDGOMUA%IGOMGFL
```

However, all the GOM data were well filled (if specific keys are correct), so there was no problem with GOMGFL pointers :

```
IF(YDGOMUA%LS) THEN
IOFF = (IS-1)*KLEV
YDGOMUA%S => YDGOMUA%GFL(IOFF+1 :IOFF+KLEV, :)
```

Thus, I chose to directly force in sugoms.F90 the values of the badly initialized keys by assigning the value of the corresponding field in the upper-air GOM :

```
YGOMUA_2D(IMAPOMM)%LS = LGOMUA(IOTP, YGOMV%MS)
```

etc.

(c) After the call to CNT1/SUOBS/OBATABS which scans and writes the positions (lat, lon) of observations, call to mkglobstab.F90 which creates the tables needed for the message-passing of model counterparts. mkglobstab.F90 reads the total number of profiles NOBPROF.
=> a new subroutine radar_profs.F90 has been created to define positions of the vicinity of observations. The 2D GOM structure has initially been developed for a plane but it is actually a simple set of locations. The only limitation that remains is that the profiles themselves are always vertical, slanted profiles are not possible. radar_profs.F90 is called from MKGLOBSTABS, which defines global tables afterwards (identical for 2D-GOM). Global tables are used in COBSLAG, SLINT and MPOBSEQ : for example variable MAPSENDOBS = (NOSBSPROFS * NOBTOT) will be used in COBSLAG under the name IPROMB and KPROMB in SLINT to define the total number of observation positions to which the model must be interpolated.
=> new large developments in mkglobstab.F90 were needed to transform (lat,lon location) of the vicinity of an observation position in X,Y coordinates on the working grid of the limited area model AROME (CALL to elalo2xy.F90).

(d) mpobseq.F90 (and mpobseq_pack.F90 ...) message-passes the interpolated fields and stores these data in the 2D-GOM arrays by filling the global datafield (array of structure) YGOMUA defined and allocated in sugoms.F90 for each kind of observation type.
=> Thus, developments were needed to introduce new 1D-GOMGFL and 2D-GOMGFL fields for hydrometeor variables. One specific difficulty with the present code is that the introducing of a new 1D-GOM requires to systematically also create the same 2D-structure because of the size of ZBUFR used in obshor.F90. GOMGFL arrays which are filled in mpobseq.F90 for 2D-GOM arrays are constrained by the necessary consistency with the organisation, the layout and total size of semi-lagrangian bufrs defined in obshor.F90. *If the list of fields to be interpolated would depend on the obstype, then this problem would not occur. The idea to control in mpobseq.F90 the fields which must be sent or received by a 2D-control array as it is made with LGOMUA is in principle attractive. Nevertheless, the problems mentioned earlier with the forcing of logical keys*

like LGOMUA show that such a very flexible algorithm might be very difficult to implement. Therefore we leave it as an open question whether it is worth to go that far. Anyway, the merge of 1D and 2D GOM structures (by simply using an extra dimension of GOM structures) should already go in the direction of tidying up the code.

   (e) developments in preint_2d.F90 etc. in the vertical observation part for these new 2D-GOM arrays of hydrometeor variables.

2. Bayesian inversion and 3D-screening :
   (a) CALL from hretr.F90 the routine reflsim_2D.F90 to compute $\boldsymbol{H_{Z,i}}(\boldsymbol{x_b})$
   (b) CALL from hretr.F90 the routine inv_refl1dstat.F90 to compute retrievals.
   (c) A very specific aspect of radar assimilation is that the 1D-inversion of radar reflectivities is made within the screening configuration in order to compute in the same task both the retrievals and the observation minus guess departures of relative humidity. Thus, quality control of reflectivity can be performed by using the results of the Bayesian inversion.

       CALL to hop.F90 in order to compute radial winds and relative humidity model counterparts (same obstype “radar”). As concerns model fields needed in hop.F90 for the observation operators of relative humidity and Doppler radial wind, it is necessary to fill the 1D-GOM with the content of the 2D-GOM (with the first profile being the observation itself) : ZUF5(:, :) = ZUF5_2D(:, :, 1) etc..

       This last point is a potential difficulty because radial wind and reflectivity are observations which come from the same obstype, but need different sizes of GOM-arrays.