Back-phasing the Mode-S thinning and adaptive weighting from AROME cy49 to cy46

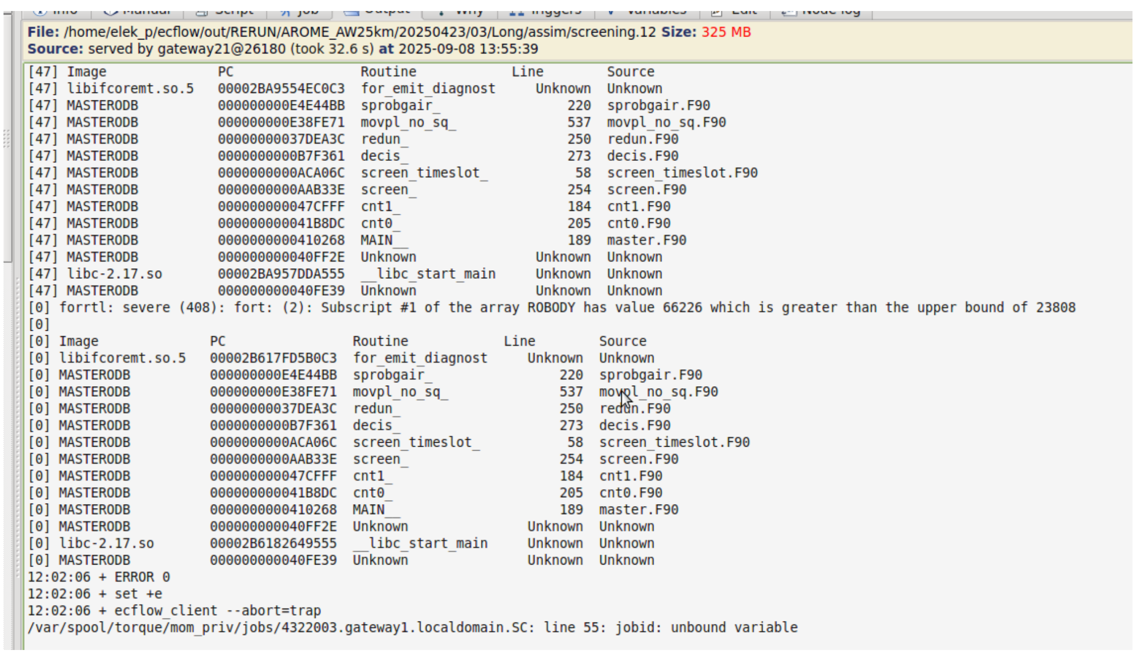
Purpose: Report from RC LACE stay

Author: Péter Elek (HungaroMet)

Host: Vivien Pourret (Meteo-France)

Place: Meteo-France, France, Toulouse

Date(s): 18 August - 30 August 2025



```
File: /home/elek_p/ecflow/out/RERUN/AROME_AW25km/20250423/03/Long/assim/screening.12 Size: 325 MB
Source: served by gateway21@26180 (took 32.6 s) at 2025-09-08 13:55:39
[47] Image              PC                Routine            Line        Source
[47] libifcoremt.so.5   00002BA9554EC0C3  for_emit_diagnost     Unknown  Unknown
[47] MASTERODB          000000000E4E44BB  sprobgair_                220  sprobgair.F90
[47] MASTERODB          000000000E38FE71  movpl_no_sq_              537  movpl_no_sq.F90
[47] MASTERODB          00000000037DEA3C  redun_                    250  redun.F90
[47] MASTERODB          0000000000B7F361  decis_                    273  decis.F90
[47] MASTERODB          0000000000ACA06C  screen_timeslot_           58  screen_timeslot.F90
[47] MASTERODB          0000000000AAB33E  screen_                   254  screen.F90
[47] MASTERODB          000000000047CFFF  cnt1_                     184  cnt1.F90
[47] MASTERODB          000000000041B8DC  cnt0_                     205  cnt0.F90
[47] MASTERODB          0000000000410268  MAIN__                    189  master.F90
[47] MASTERODB          000000000040FF2E  Unknown               Unknown  Unknown
[47] libc-2.17.so       00002BA957DDA555  __libc_start_main     Unknown  Unknown
[47] MASTERODB          000000000040FE39  Unknown               Unknown  Unknown
[0] forrtl: severe (408): fort: (2): Subscript #1 of the array ROBODY has value 66226 which is greater than the upper bound of 23808
[0]
[0] Image              PC                Routine            Line        Source
[0] libifcoremt.so.5   00002B617FD5B0C3  for_emit_diagnost     Unknown  Unknown
[0] MASTERODB          000000000E4E44BB  sprobgair_                220  sprobgair.F90
[0] MASTERODB          000000000E38FE71  movpl_no_sq_              537  movpl_no_sq.F90
[0] MASTERODB          00000000037DEA3C  redun_                    250  redun.F90
[0] MASTERODB          0000000000B7F361  decis_                    273  decis.F90
[0] MASTERODB          0000000000ACA06C  screen_timeslot_           58  screen_timeslot.F90
[0] MASTERODB          0000000000AAB33E  screen_                   254  screen.F90
[0] MASTERODB          000000000047CFFF  cnt1_                     184  cnt1.F90
[0] MASTERODB          000000000041B8DC  cnt0_                     205  cnt0.F90
[0] MASTERODB          0000000000410268  MAIN__                    189  master.F90
[0] MASTERODB          000000000040FF2E  Unknown               Unknown  Unknown
[0] libc-2.17.so       00002B6182649555  __libc_start_main     Unknown  Unknown
[0] MASTERODB          000000000040FE39  Unknown               Unknown  Unknown
12:02:06 + ERROR 0
12:02:06 + set +e
12:02:06 + ecflow_client --abort=trap
/var/spool/torque/mom_priv/jobs/4322003.gateway1.localdomain.SC: line 55: jobid: unbound variable
```

1 Introduction

Vivien Pourret at Meteo-France achieved forecast improvements by using adaptive weighting on Mode-S EHS in AROME cy49. On the other hand, AROME cy46 is operational in several meteorology services. Therefore, we decided to back-phase the adaptive weighting.

1.1 Settings

	HungaroMet	Meteo-France
Model version	AROME cy46t1_bf07	AROME cy46t1_op1.11
DA cycle	3 hourly 3D-Var + SEKF (AMDAR, Czech Mode-S MRAR)	1 hourly 3D-Var (AMDAR)
Horizontal and vertical resolution	2.5kmL60	1.3kmL90

The Mode-S EHS data available from EMADDC in 3 different files obsoul_2_ehs_eu*, bufr_2_ehs_eu*, bufr_2_ehs_fa*, we used the bufr_2_ehs_fa* files because they are available in every 5 minutes (the bufr_2_ehs_eu* files are available in every 15 minutes). At HungaroMet we are experimented with Siebren de Haan's box thinning (with the following settings –box_heights 300, 300, 600, 1000 –box_width 40) but its usage didn't yield to any improvement on the forecasts. On the other hand, the colleagues from Meteo-France achieved a forecast improvement with the usage of thinning and adaptive weighting in AROME cy49. This thinning is done in 2 steps:
step 1: Thiair.F90 performs a half resolution (e.g. 12.5km) thinning with a classical box thinning. This step is only used to reduce the data number to be managed by the next step.
step 2: Thiair2.F90 performs a true (e.g. 25km) thinning by computing inter-distances matrix by aircraft and by model level with the remaining output locations of the step 1. Then we select locations spaced by more than the thinning distance (25km) on the aircraft track by model level.
After the thinning steps, the adaptive weighting is performed.
Based on Meteo-France's results, we decided to back-phase these modifications to cy46t1_bf07.01 and test their impact.

1.2 Adaptive weighting

The main idea behind adaptive weighting is the following: after the thinning which is only used aircraft by aircraft to avoid horizontal obs error correlations. Mode-S EHS data still remains dense with numerous aircraft providing data, especially near airports ($\sim$ 10000 aircraft flying across AROME France Domain, $\sim$ 6000 aircraft flying across AROME Hungary Domain), but we don't want to overfit the analysis for that data type. Therefore, to avoid overfitting, we define dense areas on a grid with a resolution of about the representativeness of our model grid. By modifying the observation error of Mode-S data on dense boxes, the goal is to keep all observations and reducing their weights in the analysis process. If we choose the weights as the square root of the number of Mode-S EHS data in the dense box, then everything happens as if we have a homogeneous weighted grid of Mode-S data.

$$J(x_a) := \sum_{k,l \in \text{other data}} (H(x_a)_k - y_k) R^{-1}_{kl} (H(x_a)_l - y_l) + \sum_{i \in \text{modes}} \frac{(H(x_a)_i - y_i)^2}{(w_i \sigma_i)^2} + \\ + \sum_{\substack{p,q \in \text{modes} \\ p \neq q}} (H(x_a)_p - y_p) R^{-1}_{pq} (H(x_a)_q - y_q) + \text{background terms}$$

Where R is the observation error covariance matrix, $\sigma_i^2 := R_{ii}$, H is the observation operator, w is the adaptive weight, y is the observation value and x_a is the model state vector in the analysis time (during the minimization we calculate this: $\arg\min J(x)$).
Let's decompose the sum that includes the adaptive weighting. We could define $n_a(i_g)$ which is the index

set of Mode-S data in the AW grid with index i_g. In addition, n_g denotes the number of AW grid points. By using the fact that these weights only depend on the AW grid $w(i_g) := w_i = w_j$ if, $i, j \in i_g$.

$$\sum_{i\in \text{modes}} \frac{(H(x_a)_i - y_i)^2}{(w_i\sigma_i)^2} = \sum_{i_g=0}^{n_g} \sum_{i\in n_a(i_g)} \frac{(H(x_a)_i - y_i)^2}{(w_i\sigma_i)^2} = \sum_{i_g=0}^{n_g} \frac{1}{w(i_g)^2} \sum_{i\in n_a(i_g)} \frac{(H(x_a)_i - y_i)^2}{(\sigma_i)^2} =$$

$$= \sum_{i_g=0}^{n_g} \frac{1}{|n_a(i_g)|} \sum_{i\in n_a(i_g)} \frac{(H(x_a)_i - y_i)^2}{(\sigma_i)^2}$$

As we can see in the last term, if we choose $w(i_g) = \sqrt{|n_a(i_g)|}$ then we use the average in the AW grid during the minimization. In practice in a simple example we have 2 data types: one of them is Mode-S EHS. In that case, when we do the minimization without adaptive weighting, we minimize this (in the simplest case where R is diagonal with values σ_o^2 and σ_i^2 where the indices $i \in$ modes are the values related to Mode-S EHS data. Also I don't write the background terms because it's not changing when we use adaptive weighting).

$$\frac{(H(x_a)_o - y_o)^2}{\sigma_o^2} + \sum_{i\in \text{modes}} \frac{(H(x_a)_i - y_i)^2}{\sigma_i^2} + \text{background terms}$$

Where x_a is the analysis, y_o is the non Mode-S EHS observation data and y_i is the Mode-S EHS observation data when $i \in$ modes. Let's assume that the total number of Mode-S EHS data is $n = |\text{modes}|$. Then with adaptive weighting we minimize this:

$$\frac{(H(x_a)_o - y_o)^2}{\sigma_o^2} + \frac{1}{n} \sum_{i\in \text{modes}} \frac{(H(x_a)_i - y_i)^2}{\sigma_i^2} + \text{background terms}$$

In the special case when we have identical Mode-S data (identical location and data, then):
$H(\cdot)_i = H(\cdot)_j \;\wedge\; y_i = y_j \;\wedge\; \sigma_i = \sigma_j \;\forall i, j \in \textit{modes}$ then define $y = y_i$ and $\sigma = \sigma_i$ where i is a fixed element of the set *modes*.

$$\frac{(H(x_a)_o - y_o)^2}{\sigma_o^2} + \frac{(H(x_a)_i - y)^2}{\sigma^2} + \text{background terms}$$

As we can see (in this very special case), this does not depends on the Mode-S EHS observation number anymore, therefore we don't overfit on Mode-S EHS data due to their very high density compared to the other observations.
At the codes level, the RFIND_CLSTR namelist parameter sets the box size for adaptive weighting, which means that all observation errors in this box are multiplied by the square root of the number of aircraft in the box.

2 Technical details of the code changes

The back-phasing was done in 2 phases. First, we back-phased the relevant code to the latest operational AROME cy46t1_op1.11 at MF. That gave us the benefit of using belenos (one of the supercomputers at MF) and other internal tools that make the experiment creation and debugging faster. Then we ported the modifications to the current operational cy46t1_bf07 at HungaroMet. In the second phase, we mainly fixed the sql files that describes the content of ROBODY and ROBHDR variables and copied all the other Mode-S EHS related code to the codebase on diana (supercomputer of HungaroMet under my user */home-/elek_p/pack/cy46_ehs_aw2*). This later step was much easier because the two codebases are in the same cycle. Therefore, fewer code differences were involved, which means we had to investigate fewer lines of code.
The work directory on belenos: /home/gmap/mrpa/elekp
The work directory on diana: /home/elek_p
At HungaroMet, we had to make changes in the following files:

arpifs/obs_preproc/thiair.F90
arpifs/obs_preproc/thiair2.F90
arpifs/obs_preproc/sprobgair.F90
arpifs/obs_preproc/checkairpos2.F90
arpifs/obs_preproc/defrun.F90
arpifs/obs_preproc/movpl_no_sq.F90
arpifs/obs_preproc/redun.F90
arpifs/obs_preproc/flspeedbad2.F90

arpifs/cma2odb/getdb.F90
arpifs/setup/cmoctmap.F90
arpifs/setup/cmoctmap_inv.F90
arpifs/module/yomscc.F90
arpifs/module/yomcoctp.F90
arpifs/module/parersca.F90
arpifs/namelist/namscc.nam.h
odb/ddl.CCMA/redun_robhdr_4.sql
odb/ddl.CCMA/redun_robody_4.sql
odb/ddl.CCMA/CCMA_Sstatic.c (generated with ics_script by gmkpack)
odb/ddl/redun_robhdr_4.sql
odb/ddl/redun_robody_4.sql
odb/pandor/module/bator_decodbufr_mod.F90
odb/pandor/module/bator_init_mod.F90
odb/pandor/module/bator_saisies_mod.F90
odb/pandor/module/bator_lectures_mod.F90
odb/pandor/module/bator_util_mod.F90
odb/pandor/module/bator_ecritures_mod.F90
odb/pandor/module/bator_module.F90
odb/pandor/namelist/bator_namelist.nam.h
odb/tools/Bator.F90
odb/ddl.ECMA/redun_robhdr_4.sql
odb/ddl.ECMA/redun_robody_4.sql

To implement the adaptive weighting in AROME cy46 I used the following FORTRAN compiler flags: -g -O0 -check bounds.
For the first step modified the odb/ddl/redun_robhdr_4.sql, and odb/ddl/redun_robody_4.sql files and ensured that the ddl.ECMA and ddl.CCMA folders contain the links to the redun_robody_4.sql, redun_robhdr_4.sql (e.g. ln -sf ../odb/ddl/redun_robhdr_4.sql redun_robhdr_4.sql ran from the ddl.ECMA and ddl.CCMA folder (do the same with robody)).
In the robhdr the thinningkey_1 UPDATED, thinningkey_2 UPDATED,thinningkey_3 UPDATED are newly introduced in the query. These variables don't have physical meaning more like bookkeeping variables:

1. thinningkey_1 modified by movpl_no_sq.F90 → the number of the level closest to the data location
2. thinningkey_2 modified by thiair.F90 and then thiair2.F90 → (number of the thinning boxes from thiair) + (0.01 · the location number on the track closer than the thinning distance from thiair2)
3. thinningkey_3 modified by sprobgair.F90 → (number of the AW box) + (0.01 · aircraft number in the AW box for wind) +(0.0001 · aircraft number in the AW box for temperature)

In the robody the obs_error@errstat UPDATED, final_obs_error@errstat UPDATED are newly introduced in the query because during the adaptive weighting we use and update the obs_error based on the whitelist (see it later) and final_obs_error which is the final observation error that used in the minimization.
If your local code base can use Mode-S EHS data from bufr file format you might skip most of the modification of odb/pandor/module/bator_decodbufr_mod.F90 but I recommend verifing that the Modes subroutine matches especially the white_list related code parts (search for whiteModes variable (Also "include" it from BATOR_MODULE)). Based on Meteo-France experience the usage of white list is the right choice. By default I used the Meteo-France's white list because for our domain it is also suitable for testing purposes (lots of the aircrafts share the same id (aka same aircraft) on the domains). If it is not suitable for your domain (so there is no aircraft in the file that is also has data in the domain) you could easily create one from the bufr files for testing and then fine tune it (Siebren's thinning script contains how to read it in to python pandas). The file looks like this:
011007
'M000019 ' 1.372
'M000058 ' 1.514
'M000649 ' 1.351
'M00068b ' 1.203
'M000ebe ' 1.652
'M0010aa ' 1.064
'M0020b0 ' 1.446
'M002a66 ' 1.610
'M0031f2 ' 1.318 ...
The first line is the number of aircraft in the file and the program expects it as a file with the name "list_modes". The following lines contain the modesId (aircraftRegistrationNumberOrOtherIdentification in bufr) and the MaxError. At this stage of the development of Mode-S EHS data assimilation, the MaxError values are not used, but the plan is to use aircraft specific obs error profile rather than a common obs error profile to all aircraft.
Also, ensure that there is a case in the ExpandBufrFile subroutine to handle Mode-S EHS correctly:

```
CASE ('amdaromm')
```

```
IF ( ksec1(6) == 4 .AND. ksec1(7) == NMDEHS) THEN
CALL modes(kobs,kw,kel,kfic,tconfig(NoConfig),TabSlots,iterr(:))
       ELSE
CALL amdaromm(kobs,kw,kel,kfic,tconfig(NoConfig),TabSlots,iterr(:))
       ENDIF
CurrentThinning = ObsThinning%aero
```

Also, if you want to turn off Mode-S EHS data with a convenient flag (it's like a light switch .TRUE. value means Mode-S EHS data are included), include this (LMDEHS is located in BATOR_MODULE):

```
IF (.not. LMDEHS) THEN
    CALL Dr_Hook('MODES',1,zhook_handle)
    return
    ENDIF
```

Also fix the "includes" MXYTHINWEIGHT (it's a function) from BATOR_UTIL_MOD File odb/pandor/module/bator_init_mod.F90. In this file, you should add the CASE:

```
CASE ('modes')
          knbw = 2
          kilsup = 1
          kilw = 6
```

kilw=6 is needed because the zwagon(kw,6) is loaded with the MaxError from the white list. (Note: The HungaroMet code already contains SIGMAO_AIREP_T, SIGMAO_AIREP_U which are unrelated to adaptive weighting but it's another way to inflate obs error, if you set the values to 1.0_JPRD it has no effect) Also, set the following:

```
! MODE-S
! modes temperature
ECTERO(NAIREP,2,2,1:19)=(/ 2.10_JPRD, 1.95_JPRD, 1.80_JPRD, 1.65_JPRD, &
& 1.50_JPRD, 1.50_JPRD, 1.50_JPRD, 1.50_JPRD, 1.50_JPRD, 1.65_JPRD,  &
& 1.65_JPRD, 2.70_JPRD, 1.95_JPRD, 3.15_JPRD, 2.25_JPRD, 2.40_JPRD, &
& 2.55_JPRD, 2.70_JPRD, 3.15_JPRD /)
! EMADDC modes wind
ECTERO(NAIREP,2,3,1:19)=(/1.19_JPRD, 1.18_JPRD, 1.17_JPRD, 1.14_JPRD, &
& 1.16_JPRD, 1.20_JPRD, 1.23_JPRD, 1.26_JPRD, 1.30_JPRD, 1.34_JPRD, &
& 1.46_JPRD, 1.65_JPRD, 2.01_JPRD, 2.63_JPRD, 3.50_JPRD, 3.50_JPRD,&
& 3.50_JPRD, 3.50_JPRD, 3.50_JPRD /)
ECTERO(NAIREP,2,4,1:19) = ECTERO(NAIREP,2,3,1:19)
```

File odb/pandor/module/bator_saisies_mod.F90 only needs these 3 lines in addition:

```
if (icodmes == NMDEHS) then
  ZDATALOC(jobs,5) = idivers
endif
```

In the file odb/pandor/module/bator_lectures_mod.F90 you just add the subroutine ReadModesWhiteList
File odb/pandor/module/bator_util_mod.F90 you have to add function MXYTHINWEIGHT.
File odb/pandor/module/bator_ecritures_mod.F90 in this file there are a lot of small changes. I used vimdiff and /\cmodes (or /\cmode-s or /\cairep or /\cnmdehs) and replace the related code parts. (Also this file contains the obs inflation with SIGMAO_AIREP_T, SIGMAO_AIREP_U.) Also don't forget to fix the "includes" from PARERSCA

```
(JPHHY2C,&
& JPCELL, JPACELL, JPACELL_HR, JPACELL_XHR, JPICELL, JPICELL_HR, &
& JPRCELL, JPHCELL, JPHCELL_HR, JPSCELL, JPSCELL_HR, &
& JPFCELL, JPFCELL_HR, JPWCELL) ! you might not need all.
```

In the case some of them are needed, you might have to define it in arpifs/module/parersca.F90. Also, generally the following pattern holds: something_HR=2·something (e.g. JPFCELL_HR=76, JPFCELL=38) but I'm not sure about the rationale behind this. (Similar with XHR and HR.)

Optional but I recommend checking arpifs/cma2odb/getdb.F90 because one bound checking is wrong and another-one bound checking is missing (around GETACTIVE and GETACTIVE_SAT)
File odb/pandor/module/bator_module.F90 includes these:

```
type T_whiteModes
    character(len=8) :: modesId
    real(kind=jprd)  :: MaxError
  end type T_whiteModes
  type(T_whiteModes), dimension(:), allocatable :: whiteModes
LOGICAL :: LMDEHS   = .FALSE.
LOGICAL :: LMWTS2_MANDATORY_AVG
```

File odb/pandor/namelist/bator_namelist.nam.h includes these 3 namelist parameters:
LMDEHS, SIGMAO_AIREP_U, SIGMAO_AIREP_T
File arpifs/namelist/namscc.nam.h: You have to include RFIND_CLSTR, LSPROBG (this is the namelist variable that turns on adaptive weighting if the value is .TRUE.)
File arpifs/module/yomscc.F90: That is the place where you have to define the types of RFIND_CLSTR, LSPROBG, FL_MINSPEED (the newly introduced counter part of the already existing FL_MAXSPEED)

```
LOGICAL :: LSPROBG
REAL(KIND=JPRB) :: RFIND_CLSTR
REAL(KIND=JPRB) :: FL_MINSPEED
```

File arpifs/module/yomcoctp.F90: Increase NAIRCT to 10 from 9 because the array size is 10.

```
INTEGER(KIND=JPIM), PARAMETER :: NAIRCT    =  10
```

File arpifs/setup/cmoctmap.F90: include NMDEHS, NMDEHSSQ and add these lines (to include Mode-S EHS related conversion from NMDEHS to NMDEHSSQ):

```
!*           1.2.10  MODES AIRCRAFT REPORT
ELSEIF(KCDTYP == NMDEHS)       THEN
                KCDSQNO = NMDEHSSQ
```

File arpifs/setup/cmoctmap_inv.F90: Similarly to cmoctmap.F90 (but for inverse conversion):

```
!*           3.2.10  MODE-S EHS AIRCRAFT REPORT
ELSEIF(KCDSQNO == NMDEHSSQ)       THEN
  KCDTYP   = NMDEHS
```

File arpifs/obs_preproc/movpl_no_sq.F90: Define these:

```
REAL(KIND=JPRD), POINTER :: ZTHINNINGKEY(:)
INTEGER(KIND=JPIM) :: IEND
```

Include these:

```
#include "checkairpos2.intfb.h"
#include "thiair2.intfb.h"
#include "sprobgair.intfb.h"
```

Initialize ZTHINNINGKEY with thinningkey_1:

```
ZTHINNINGKEY => ROBHDR(:,MDB_THINNINGKEY_AT_HDR(1))
```

These lines manage to turn off and on the adaptive weighting and the usage of Mode-S EHS data.

```
  ! TYPE=1 FOR REJECTION OF CRAZY POSITIONS
  !        ALL OBSERVATIONS FROM SAME AIRCRAFT HANDLED SIMULTANEOUSLY
  !        THIS SCAN MAY REJECT REPORTS SO SECOND SCAN MAY BE SHORTER
  ! TYPE=2 FOR REJECTION OF TOO CLOSE REPORTS FROM SAME AIRCRAFT WITH
  !        SAME CODE TYPE, INSTRUMENT TYPE AND VERTICAL LEVEL
  ! TYPE=3 REFINE TYPE 2 FOR MODES
  ! TYPE=4 SUPEROBBING: SIGMAO-> SIGMAO x SQRT(N), WITH N THE ALL
  !        AIRCRAFT DATA NUMBER IN A THINNING GRID (MODEL REPRESENTATIVNESS)
```

```
IF (LECMWF) THEN
    ISTART=1
    IEND=2
  ELSE
    ISTART=1
    IF (LSPROBG) THEN
      IEND=4 ! strict thinning + super obbing for MODE-S
    ELSE
      IEND=3 ! strict thinning for MODE-S
    ENDIF
  ENDIF

REDUNDANCY_TYPE_LOOP: DO JTYPE=ISTART,IEND

!            1.1   LOOP OVER THE OBSERVATIONS

  IGLAIR = 0
  LLGOOD(:) = .FALSE.
  OBS_LOOP1: DO JOBS = 1, ILEN
    IOBS   = ZKEY(JOBS,1)
   IOBTYP = ROBHDR(IOBS,MDBOTP)
    IOBCTP = ROBHDR(IOBS,MDB_CODETYPE_AT_HDR)

    IF ( JTYPE==1 .AND. .NOT. LECMWF ) THEN
     IF (IOBCTP /= NMDEHS) CYCLE OBS_LOOP1
!*           1.2   ACCEPT ACTIVE AND PASSIVE REPORTS ONLY for MODES VARBC

     CALL NGERSTA(INT(ROBHDR(IOBS,MDBRST)),ICOUNT,IRSTA)
     IF( IRSTA(3) == 1.OR. IRSTA(4) == 1) CYCLE OBS_LOOP1
     IGLAIR = IGLAIR + 1
     LLGOOD(IOBS) = .TRUE.

    ELSE

    IF(IOBTYP /= NAIREP) CYCLE OBS_LOOP1
    IF ( JTYPE==1 .AND. .NOT. LECMWF .AND. IOBCTP /= NMDEHS ) CYCLE OBS_LOOP1
```

Modify this IF statement:

```
!  IF(IGLAIR == 0)  GO TO 10                  !from this
IF(IGLAIR == 0) CYCLE REDUNDANCY_TYPE_LOOP    !to this
```

In section 5.4 EXTRACT CODE TYPE/INST.TYPE/STAT.ID OR JUST STAT.

```
!IF (JTYPE == 2) THEN                    !from this
IF (JTYPE == 2 .OR. JTYPE == 3) THEN     !to this
```

In section 5.5 FIND INDEX OF CLOSEST QUASI MODEL LEVEL set ZTHINNINGKEY

```
ZTHINNINGKEY(IOBS) = IOBIND
```

Replace IOBIDE = IOBIND with this in section 5.6 COMPOSE AN IDENTIFIER:

```
IF (JTYPE == 1) THEN
   IOBIDE = 0
 ELSE
   IOBIDE = IOBIND
ENDIF
IF (JTYPE == 4) CLOBID="SUPEROBG"
```

These are the lines where the thinning and adaptive weighting are called (THIAIR2,SPROBGAIR)

```
!*          8.4   CALL THINNING ROUTINE (SIMILAR TO TOVS THINNING)
      IOBS = IDPAI(JJPATH,1)
      IOBCTP =  ROBHDR(IOBS,MDB_CODETYPE_AT_HDR) !ROBHDR%HDR%CODETYPE(IOBS)
      SELECT CASE(JTYPE)
      CASE (1)
          IF (LECMWF) THEN
             CALL CHECKAIRPOS(JJPATH,INOAIR,IIDAIR,IMAX,IDPAI)
          ELSE
             CALL CHECKAIRPOS2(JJPATH,INOAIR,IIDAIR,IMAX,IDPAI)
          ENDIF
      CASE (2)
          IF (IOBCTP==NMDEHS) THEN
           CALL THIAIR(JJPATH,INOAIR,IIDAIR,IMAX,IDPAI,RFIND_AIREP/2)
          ELSE
           CALL THIAIR(JJPATH,INOAIR,IIDAIR,IMAX,IDPAI,RFIND_AIREP)
          ENDIF
      CASE (3)
          IF (IOBCTP==NMDEHS) THEN
           CALL THIAIR2(JJPATH,INOAIR,IIDAIR,IMAX,IDPAI,RFIND_AIREP)
          ENDIF
      CASE (4)
          CALL SPROBGAIR(JJPATH,INOAIR,IIDAIR,IMAX,IDPAI,RFIND_CLSTR)
      END SELECT
```

creating and using CHECKAIRPOS2 (this includes the algorithm that removes reports due to unrealistic aircraft speed note that preselection is done when this called, also it's) only necessary if your codebase includes LECMWF. Otherwise, you can make the changes directly in CHECKAIRPOS without creating CHECKAIRPOS2. Let's continue with arpifs/obs_preproc/checkairpos2.F90:

```
REAL(KIND=JPRB) :: ZMINDIFF,  ZFLMINSPEED, ZOBRTM(KNOAIR)
#include "flspeedbad2.intfb.h"
```

Fix the DR_HOOK calls if you created the CHECKAIRPOS2 set

```
KNWDW=10
ZFLMINSPEED=FL_MINSPEED
```

In section 2.2.2 OBSERVATION TIME (IN MINUTES FROM REFERENCE DATE) set the newly introduced

```
ZOBRTM(JOBS)=-60.0_JPRB*IOBRTM(JOBS)+1.0_JPRB*MOD(IOBETM,100)
```

Replace section 2.4.1 FIND ALL SUSPECT HIGH SPEED PAIRS

```
!*            2.4.1   FIND ALL SUSPECT HIGH SPEED PAIRS

  LLSPEEDOK(:,:)=.TRUE.

  DO J1=1,KNOAIR-1
    KNTESTMX=MIN(J1+KNWDW, KNOAIR)
    DO J2=J1+1,KNTESTMX
      ZMINDIFF=ABS(ZOBRTM(J1)-ZOBRTM(J2))/60._JPRB
      CALL FLSPEEDBAD2(ZOBLAT(J1),ZOBLON(J1),ZOBLAT(J2),ZOBLON(J2),&
       & ZMINDIFF,ZFLMINSPEED,ZFLMAXSPEED,LLFLSPEEDBAD)
      IF (LLFLSPEEDBAD) THEN
        LLSPEEDOK(J1,J2)=.FALSE.
        LLSPEEDOK(J2,J1)=.FALSE.
        IF(LSCDPR) THEN
         WRITE(NULOUT,'(A,A8,A)') &
         & 'CHECKAIRPOS2;␣SUSPECT␣MODE-S␣PAIR␣',CLSTID,'␣WITH␣LAT/LON:␣'
         IPOS = KIDPAI(KPATH,J1)
```

```
          IOBDAT = ROBHDR(IPOS,MDBDAT)!ROBHDR%HDR%DATE(IPOS)
          IOBETM = ROBHDR(IPOS,MDBETM)!!ROBHDR%HDR%TIME(IPOS)
          WRITE(NULOUT,'(2I15,2F8.3,F20.3,2I15)') &
          & IOBDAT,IOBETM,&
          & ZOBLAT(J1)*RDEGREES,ZOBLON(J1)*RDEGREES,ZOBRTM(J1)
          IPOS = KIDPAI(KPATH,J2)
          IOBDAT = ROBHDR(IPOS,MDBDAT)!ROBHDR%HDR%DATE(IPOS)
          IOBETM = ROBHDR(IPOS,MDBETM)!!ROBHDR%HDR%TIME(IPOS)
          WRITE(NULOUT,'(2I15,2F8.3,F20.3,2I15)') &
          & IOBDAT,IOBETM,&
          & ZOBLAT(J2)*RDEGREES,ZOBLON(J2)*RDEGREES,ZOBRTM(J2)
        ENDIF
      ENDIF
    ENDDO
  ENDDO
```

fix the for loop;

```
!DO J2=1,KNOAIR !from this
KNTESTMN=MAX(J1-KNWDW, 1)
KNTESTMX=MIN(J1+KNWDW,KNOAIR)
DO J2=KNTESTMN,KNTESTMX
```

Set to 95% (from 80% this means 8.0 → 9.5) in section 2.4.3 (or don't change it but we could do that because we could be picky about the data with that density)

```
IF (KNOAIR-INUMREJ == 1 .OR. INUMREJ*10/MAX(KNOAIR,1) >= 9.5) THEN
```

File arpifs/obs_preproc/flspeedbad2.F90:
In the subroutine flspeedbad2 we changed the arguments compared to flspeedbad. Other than that, you could include it in the file without any modification. But I would highlight this code snippet because that is the main change:

```
  IF (KMINDIFF==0) THEN
     LDFLSPEEDBAD =.TRUE.
  ELSE
  ZFLSPEED=(ZAA+ZBB+ZCC)/(KMINDIFF)**2
  LDFLSPEEDBAD = (ZFLSPEED > PFLMAXSPEED) .OR. (ZFLSPEED < PFLMINSPEED)
  ENDIF
```

Files arpifs/obs_preproc/thiair.F90:
Add NMDEHS:

```
USE YOMCOCTP , ONLY : NAIREP   ,NMDEHS
```

define IOBCTP:

```
 INTEGER(KIND=JPIM) :: IOBCTP
```

also add the following:

```
IOBCTP = NINT(ROBHDR(IPOS,MDB_CODETYPE_AT_HDR))

!*          3.4   ALLOW ONE REPORT PER THINNING BOX ONLY

  IF(NINT(ZKEY(INDEX(JOBS))/100._JPRB) == IOLDKEY) THEN
    ICMONM = ROBHDR(IPOS,MDBONM)
    CALL REJMV(IPOS,ICMOTP,ICMONM)
    LLREJEC = .TRUE.
  ELSE
    IOLDKEY = NINT(ZKEY(INDEX(JOBS))/100._JPRB)
    LLREJEC = .FALSE.
  ENDIF
```

Files arpifs/obs_preproc/thiair2.F90 and arpifs/obs_preproc/sprobgair.F90:
These files are newly created by Vivien Pourret. Therefore, I recommend opening the files and taking a peek at the source code if you have questions. Also, I think it is very useful to take a look into arpifs/module/ifs_dbase_view_mod.F90 in cy49 and arpifs/cma2odb/initmdb.F90 in cy46 (especially if you have questions about how to access to a given field in ROBODY or ROBHDR. File arpifs/obs_preproc/defrun.F90:

```
!*          'MODES, Aircraft data            '
!*          'MODES, Aircraft data stratosphere
CALL CMOCTMAP('CODETOSQ',NULOUT,NAIREP,NMDEHS,IOBT,ICDT)
RHUBERLEFT (NVAR_U,ICDT,IOBT,1) = 1.4_JPRB
RHUBERRIGHT(NVAR_U,ICDT,IOBT,1) = 1.4_JPRB
RHUBERBGQC (NVAR_U,ICDT,IOBT,1,1:3) = (/9.0_JPRB,324.0_JPRB,484.0_JPRB/)
RHUBERLEFT (NVAR_T,ICDT,IOBT,1) = 1.4_JPRB
RHUBERRIGHT(NVAR_T,ICDT,IOBT,1) = 1.3_JPRB
RHUBERBGQC (NVAR_T,ICDT,IOBT,1,1:3) = (/9.0_JPRB,256.0_JPRB,400.0_JPRB/)
!*          'MODES, Aircraft data troposphere  '
RHUBERLEFT (NVAR_U,ICDT,IOBT,2) = 1.3_JPRB
RHUBERRIGHT(NVAR_U,ICDT,IOBT,2) = 1.3_JPRB
RHUBERBGQC (NVAR_U,ICDT,IOBT,2,1:3) = (/9.0_JPRB,484.0_JPRB,576.0_JPRB/)
RHUBERLEFT (NVAR_T,ICDT,IOBT,2) = 1.3_JPRB
RHUBERRIGHT(NVAR_T,ICDT,IOBT,2) = 1.3_JPRB
RHUBERBGQC (NVAR_T,ICDT,IOBT,2,1:3) = (/9.0_JPRB,324.0_JPRB,484.0_JPRB/)
!*          'MODES, Aircraft data boundary layer  '
RHUBERLEFT (NVAR_U,ICDT,IOBT,3) = 1.6_JPRB
RHUBERRIGHT(NVAR_U,ICDT,IOBT,3) = 1.5_JPRB
RHUBERBGQC (NVAR_U,ICDT,IOBT,3,1:3) = (/9.0_JPRB,324.0_JPRB,484.0_JPRB/)
RHUBERLEFT (NVAR_T,ICDT,IOBT,3) = 1.0_JPRB
RHUBERRIGHT(NVAR_T,ICDT,IOBT,3) = 1.2_JPRB
RHUBERBGQC (NVAR_T,ICDT,IOBT,3,1:3) = (/9.0_JPRB,256.0_JPRB,400.0_JPRB/)
```

Also set:

```
RFIND_CLSTR=RFIND_AIREP
FL_MINSPEED = 100.0_JPRB
```

These settings are unrelated to Mode-S and adaptive weighting, but we had to set these because of the compiler debug options (we are going to isolate this type of changings, but at the time of the visit was a feasible temporary solution to move on):

```
ASCAT_XYGRID=25000._JPRD !CP heritage grid, may be changed by namelist NAMSCC
OSCAT_XGRID=50000._JPRD
HSCAT_XGRID=50000._JPRD
SSCAT_XGRID=50000._JPRD
FSCAT_XGRID=50000._JPRD
```

Also, check the includes (like ASCAT_XYGRID,LMDEHS and so on). Moreover, the previously defined JPICELL_HR and other variables from PARERSCA, I recommend creating these code entries like this (made the values compatible):

```
!*          1.11.7.b High-Resolution
IF(JPICELL_HR /= 72 ) THEN
WRITE(NULOUT,'(A,I10,A)') "DEFRUN ERROR: ",JPICELL_HR," JPICELL_HR SHOULD BE 76"
CALL ABOR1('SUBROUTINE DEFRUN') ENDIF
```

And last but not least, let's modify odb/tools/Bator.F90. In this file, we only need to load the white list that we created previously:

```
CALL ReadGpssolWhiteList(INbAllowedGpssol,TS_Gpssol,GpssolMethod)
print *, "INFO -- BATOR LMDEHS"
print *, LMDEHS
if (LMDEHS) CALL ReadModesWhiteList(whiteModes)
```

3 How to create an experiment

After creating the new binaries: Ensure that the white list for Mode-S copied into the workdir when the BATOR binary called with the name: "list_modes". Also, you have to modify the bator and screening namelist

3.1 Adaptive weighting: ON

bator namelist:
&BUFR
LMDEHS=.TRUE.
screening namelist:
&NAMSCC
LSPROBG=.TRUE.,
RFIND_CLSTR=4500.

3.2 Adaptive weighting: OFF

bator namelist:
&BUFR
LMDEHS=.TRUE.
screening namelist:
&NAMSCC
LSPROBG=.FALSE.,

3.3 Mode-S EHS data: OFF

bator namelist:
&BUFR
LMDEHS=.FALSE.
screening namelist:
LSPROBG=.FALSE.,

4 Experiment results

The longer experiments are not finished during the stay, but with the internal tools at Meteo-France we calculated the following quantities to ensure that we got the result as we expected (e.g. Desroziers diagnostics, see below). At the time of writing this report, we noticed that the cy49 (which produces correct behavior) and cy46 produced different behavior with adaptive weighting, the reason for that is unknown for now. Investigating the odbs (ECMA,CCMA) produced by the version run on diana I got unexpected results (suspiciously high adaptive weights), near to airports at high vertical coordinates (∼ 10000m). Resolving these issues is an ongoing task.

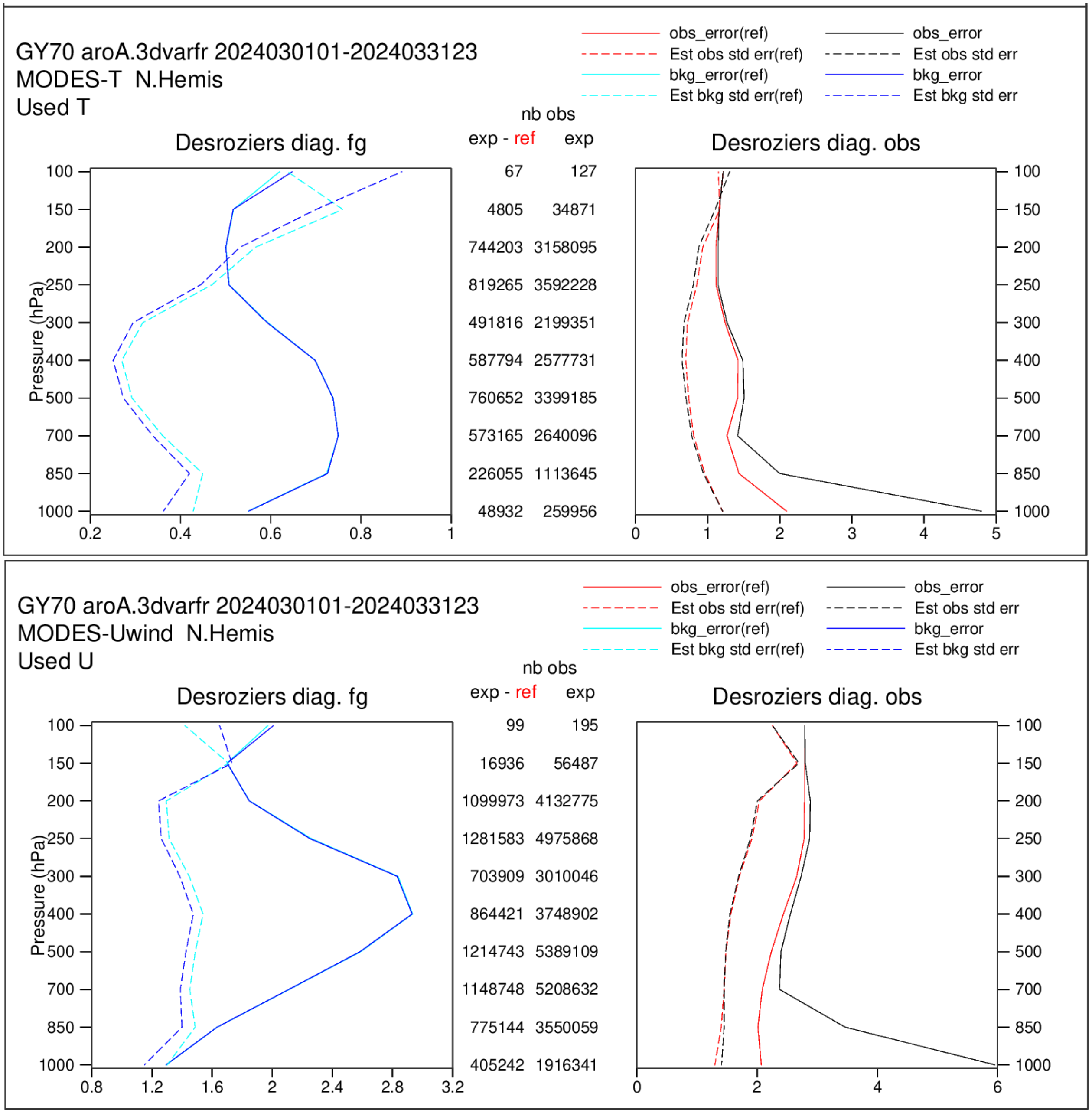


Figure 1: cy49 results
reference: with Mode-S EHS without adaptive weighting
experiment: with adaptive weighting

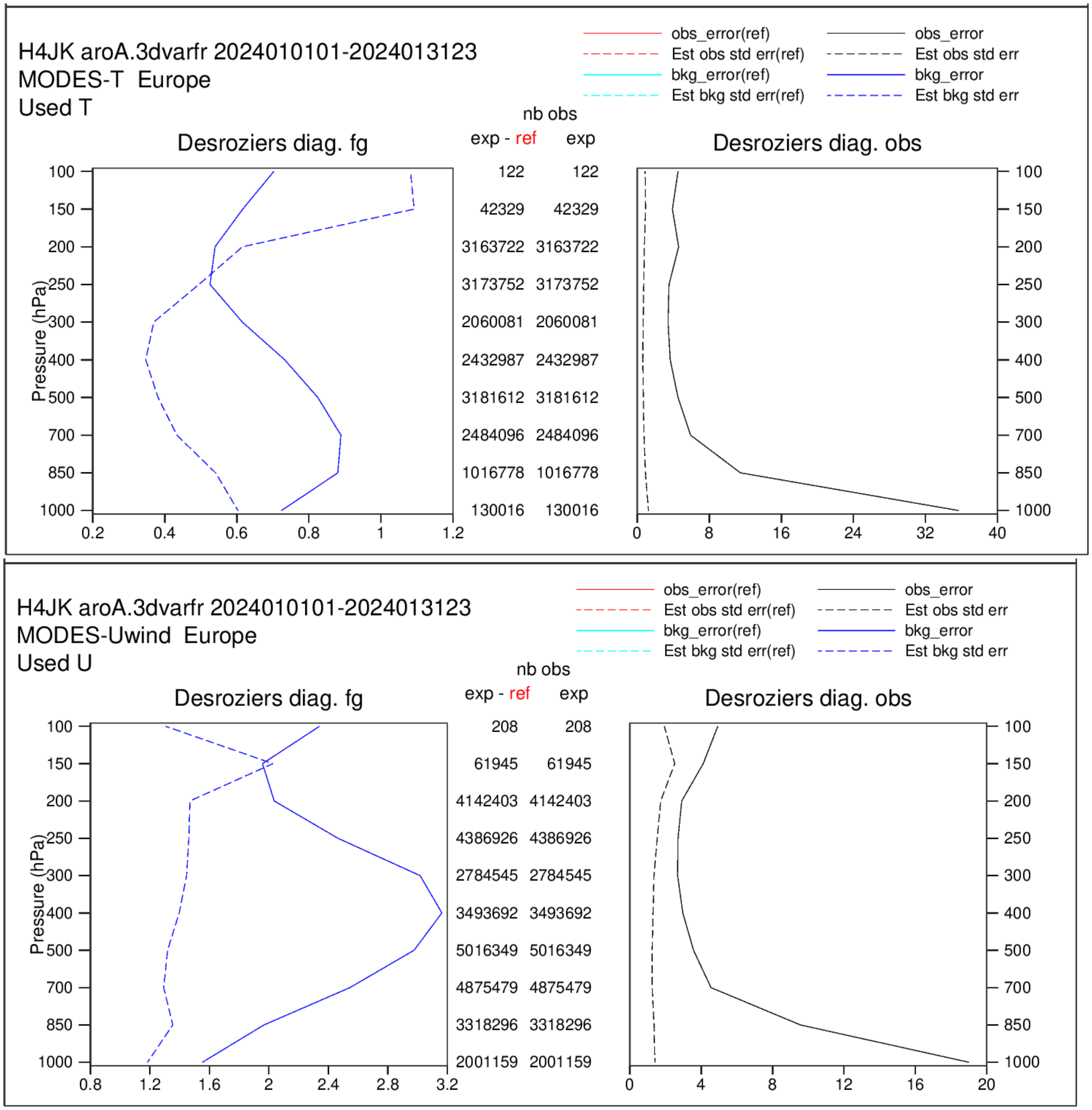


Figure 2: cy46 results
no reference on the graph
experiment: with adaptive weighting