A new formulation for using IFS-HTessel Soil Wetness in Aladin-ISBA

João Pestana Ferreira [1]

Advised by:

François Bouyssel [2]
Pedro Viterbo [3]

Report of the work done in Meteo-France Headquarters in Toulouse
2 to 27 November 2008

[1] Instituto de Meteorologia, Portugal
[2] Meteo-France
[3] Instituto de Meteorologia, Portugal

A new formulation for using IFS-HTessel Soil Wetness in Aladin-ISBA

1. Introduction

Having the possibility to initialize ARPEGE or ALADIN models from IFS fields is of great importance for research or operational purposes. This is currently possible with "ARPEGE Configuration 901" which uses the IFS upper air and surface fields to create an ARPEGE (global) file which is then interpolated to run ALADIN model.

However, it has been detected that running ARPEGE or ALADIN with the current "901 configuration" generates poor quality two-metre temperature relative humidity fields, with a very cold and wet bias over large areas of the domain, especially during summer because of inappropriate soil moisture initialization. The temporary solution was to use a "blended" approach in which the IFS provide the upper air and operational ARPEGE model provides the surface fields. This solution is implemented and used by several groups for over a year, but of course it is not a satisfactory solution, for instance the ARPEGE analyses are not available on old dates such as covered by ERA40.

From a general point of view the interpolation of surface fields (soil temperature, soil moisture (liquid and solid), snow, …) from a NWP model to another is not straightforward because there is a large variety of surface parameterizations (number of layers, underlying assumptions, etc) and physiographic databases representing soil and vegetation characteristics. A physical interpolation should be done preserving as much as possible the surface fluxes (sensible, latent heat fluxes and momentum fluxes).

2. Objective

In "ARPEGE Configuration 901", the routine responsible for interpolating soil wetness from IFS-Tessel to Aladin-ISBA, is *cprep1.F90.* The objective of this work is to change this routine in order to use a more physically based assumption than the one currently used.

3. Methodology

The IFS soil scheme (H-TESSEL) is much different from the one used in ARPEGE/ALADIN (ISBA). IFS-Tessel uses 4 layers with fixed thicknesses, from top to bottom, 7 cm, 21 cm, 72 cm and 189 cm; each layer has its own water content. ISBA uses 2 layers in which the first layer has a fixed size of 1 cm and the second layer overlaps the first one and has a variable size defined has the depth (d) at which the heat flux in the soil vanishes after about 1 week of integration.

Presently, to calculate the deep soil water reservoir in ISBA, the following method is used:

Step 1) Calculate the weighted mean of IFS soil water using the 4 layers according to the formula:

$$SWL = \frac{\sum_{i=1}^{4} SWL_i * d_i}{\sum_{i=1}^{4} d_i} \qquad (3.1)$$

Step 2old) Calculate the IFS-Tessel saturation fraction according to the formula:

$$Fs = \frac{SWL}{SWL_{sat}} \qquad (3.2)$$

where SWL_{sat} is the saturation value wish only depends of the soil type according to the following table:

Table 1

Soil number	**Soil type**	SWL_{sat} **[m³/m³]**	**PWP [m³/m³]**	**FC [m³/m³]**
1	Coarse	0.403	0.059	0.242
2	Medium	0.439	0.151	0.346
3	Medium-fine	0.430	0.133	0.382
4	Fine	0.520	0.279	0.448
5	Very fine	0.614	0.335	0.541
6	Organic	0.766	0.267	0.662
7	Reserved	0.439	0.151	0.346

Step 3old) Assume that the IFS-Tessel saturation fraction is equal to the ISBA saturation fraction, which is:

$$\frac{SWL}{SWL_{sat}} = \frac{W_p}{\omega_{sat} * d * 10^3}$$

with

$$\omega_{sat} = G1WSAT * sab + G2WSAT$$
$$G1WSAT = -1.08 * 10^{-3}$$
$$G2WSAT = 494.314 * 10^{-3}$$
$$sab = percentage\, of\ sand$$

giving

$$W_p = \frac{10^3 * d * \omega_{sat} * SWL}{SWL_{sat}} \quad \left[kg / m^2\right] \qquad (3.3)$$

As already referred, this approach (called OLD in this document) is far from perfect, resulting in a soil too wet. In the new approach presented in this document, both models should have the same Soil Wetness Index (SWI), rather than the same Saturation Fraction.

The so called NEW approach replaces Step 2old) by the SWI computation

Step 2new) IFS-Tessel SWI computation

$$SWI_{ec} = \frac{SWL - PWP}{CAP - PWP}$$

with CAP and PWP depending of soil type according to Table 1.
Step 3new) Assume that IFS-Tessel SWI is equal to the ISBA SWI, that is:

$$\frac{SWL_{ec} - PWP}{CAP - PWP} = \frac{\omega_p - \omega_{wilt}}{\omega_{fc} - \omega_{wilt}}$$

with

$$\omega_{wilt} = GWWILT * \arg^{EWWILT} \quad ; GWWILT = 37.1342 * 10^{-3} \quad and \quad EWWILT = 0.5$$
$$\omega_{fc} = GWFC * \arg^{EWFC} \quad ; GWFC = 89.0467 * 10^{-3} \quad and \quad EWFC = 0.35$$
$$\arg = Percentage\ of\ \ clay$$

giving

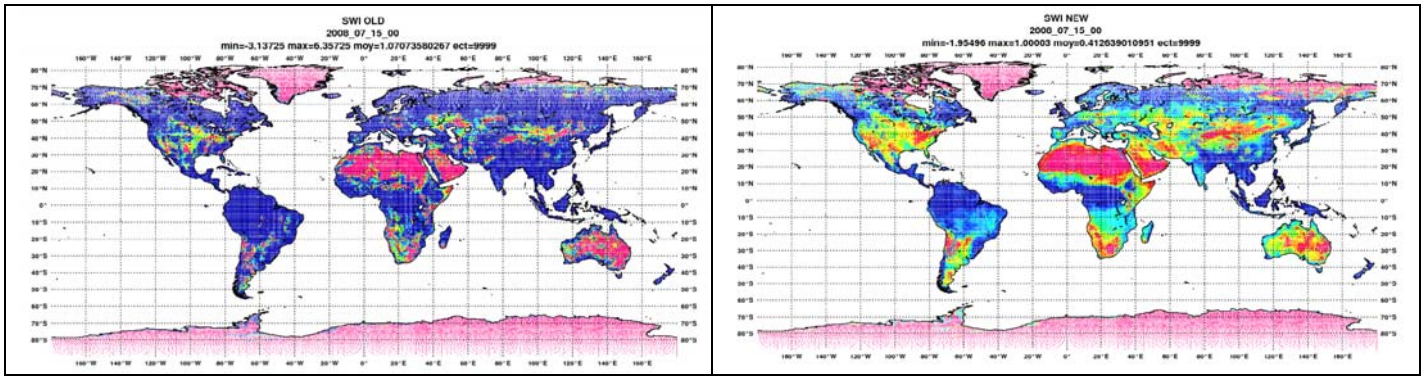
$$\omega_p = \frac{SWL - PWP}{CAP - PWP} * (\omega_{fc} - \omega_{wilt}) + \omega_{wilt} \quad \left[m^3 / m^3\right]$$

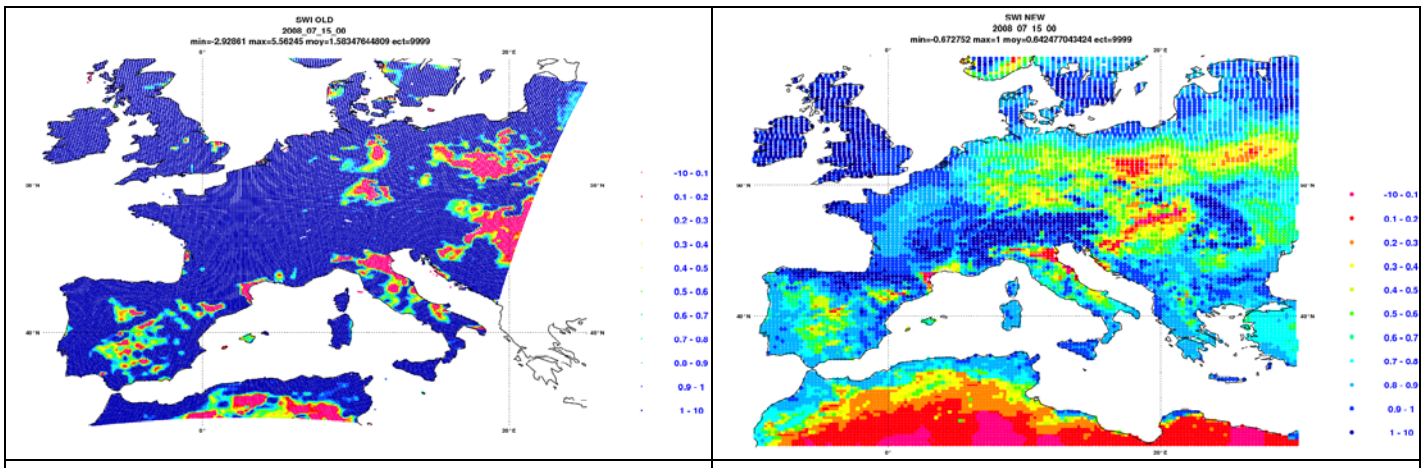
$$W_p = \omega_p * 10^3 * d \quad \left[Kg / m^2\right] \quad (3.4)$$

In practice, the NEW scheme was implemented in the cprep1.F90 routine of configuration 901 (listed in Annex 1), producing a drier soil than the OLD scheme as shown in the next section, with important repercussions in atmospheric parameters such as 2 metre temperature, 2 metre relative humidity and Evaporation.

4. Analysis of results

To test the NEW implementation, a summer day over Europe was chosen, namely the 15th of July 2008 at 00h UTC. Fig.1 shows the SWI over the world and a zoom covering ALADIN-FRANCE (Europe)





OLD scheme	NEW scheme

Fig.1 – Soil Wetness Index (SWI) for the 15th of July 2008 at 00UTC (OLD and NEW scheme)

There are remarkable differences between the two interpolations, with the NEW scheme much drier. However, when compared with Fig. 2, which can be taken as a good proxy of reality since it is the SWI of the SIM (offline ISBA model forced by analysed precipitation, radiative fluxes and PBL parameters), the NEW SWI is still too wet over large areas of France.

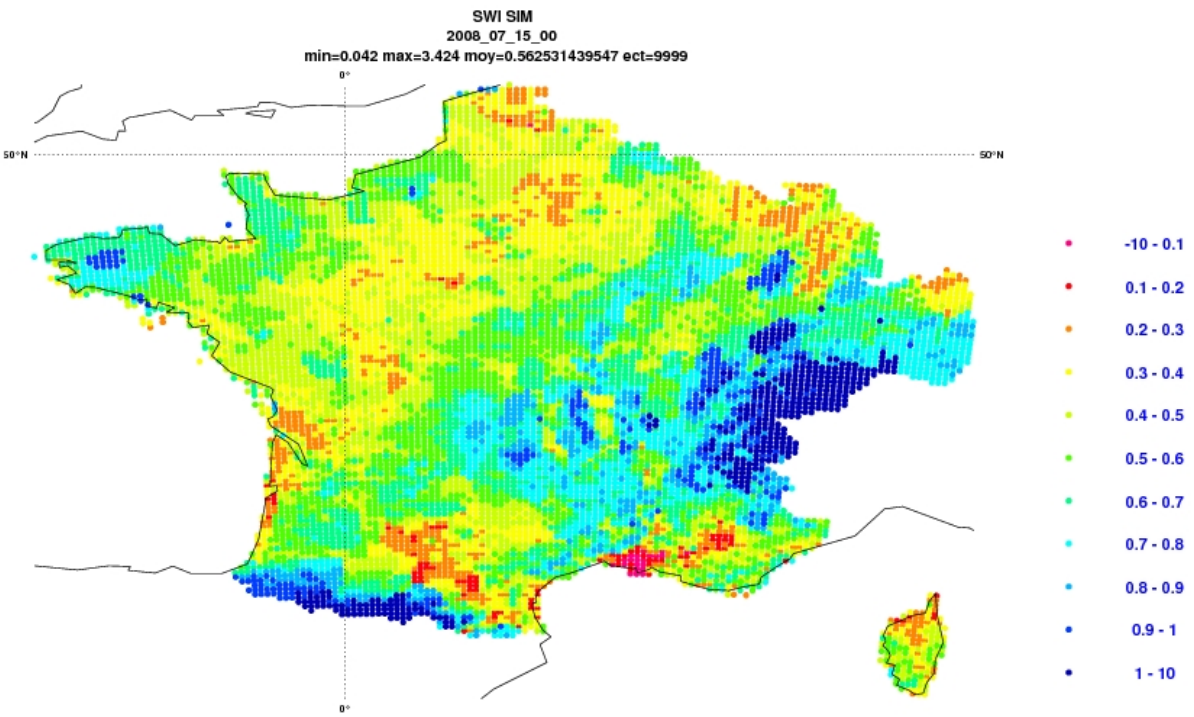


Fig.2 – SIM SWI for the 15th of July 2008 at 00UTC

Fig.3 shows the ALADIN OPER SWI obtained only with ISBA fields (both surface and upper-air). The OPER SWI seems too much dry compared with SIM SWI, but the truth is that it produces good 2 metre temperature and relative humidity H+12 fields as shown in the next page, reinforcing the idea that the NEW SWI is not dry enough. In fact, as also shown, the 2 metre temperature and relative humidity H+12 fields produced by the NEW scheme are too cold and too wet, despite being better than the OLD scheme mainly in the relative humidity.

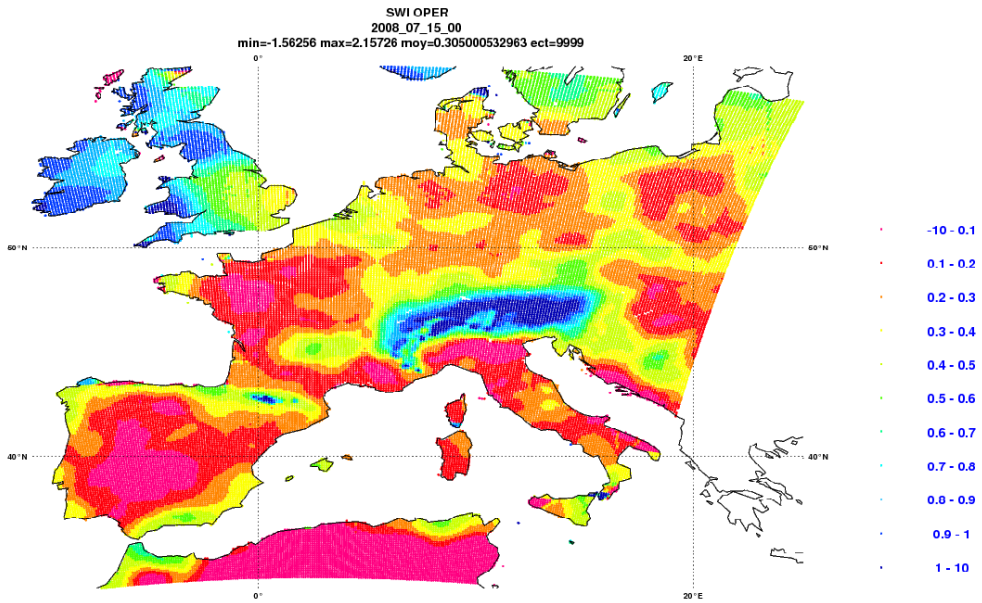
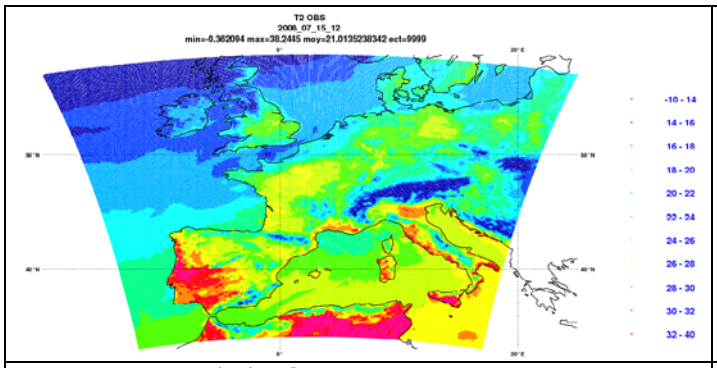
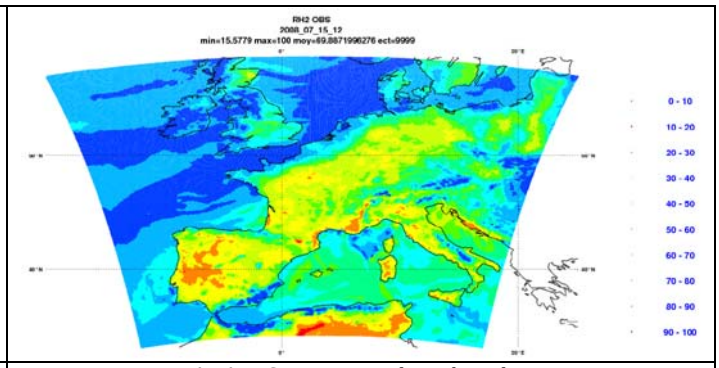
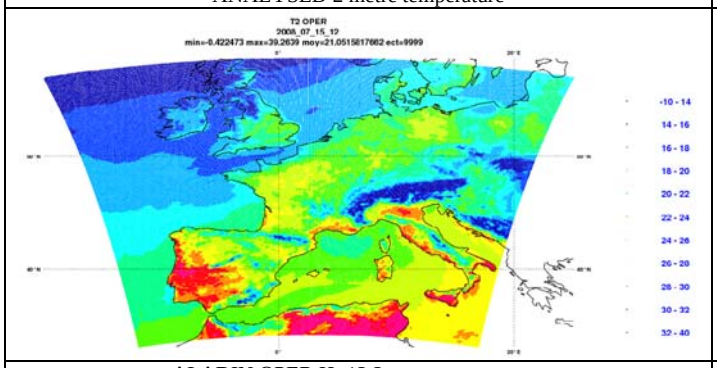
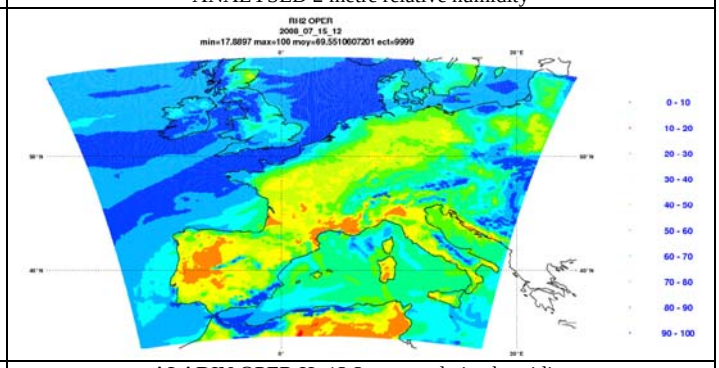
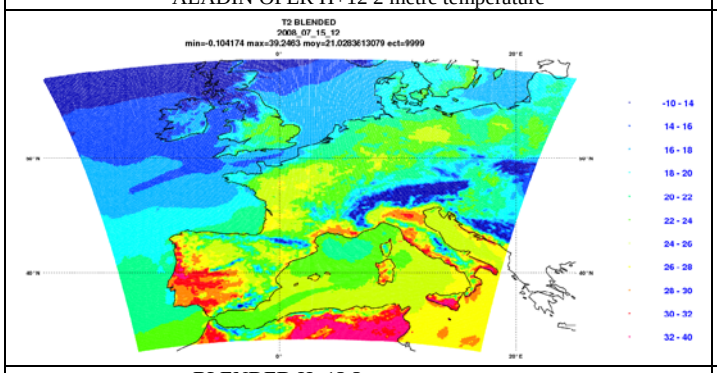
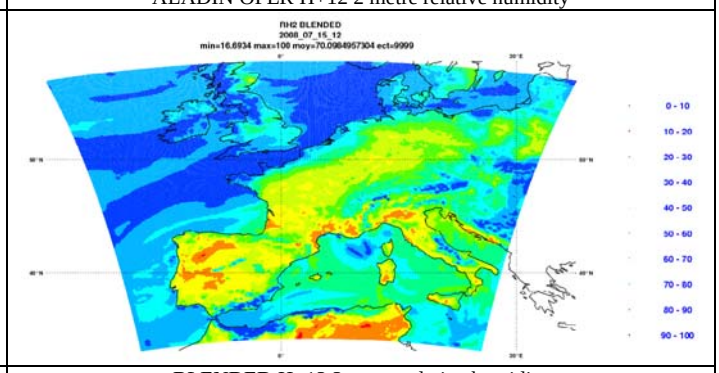
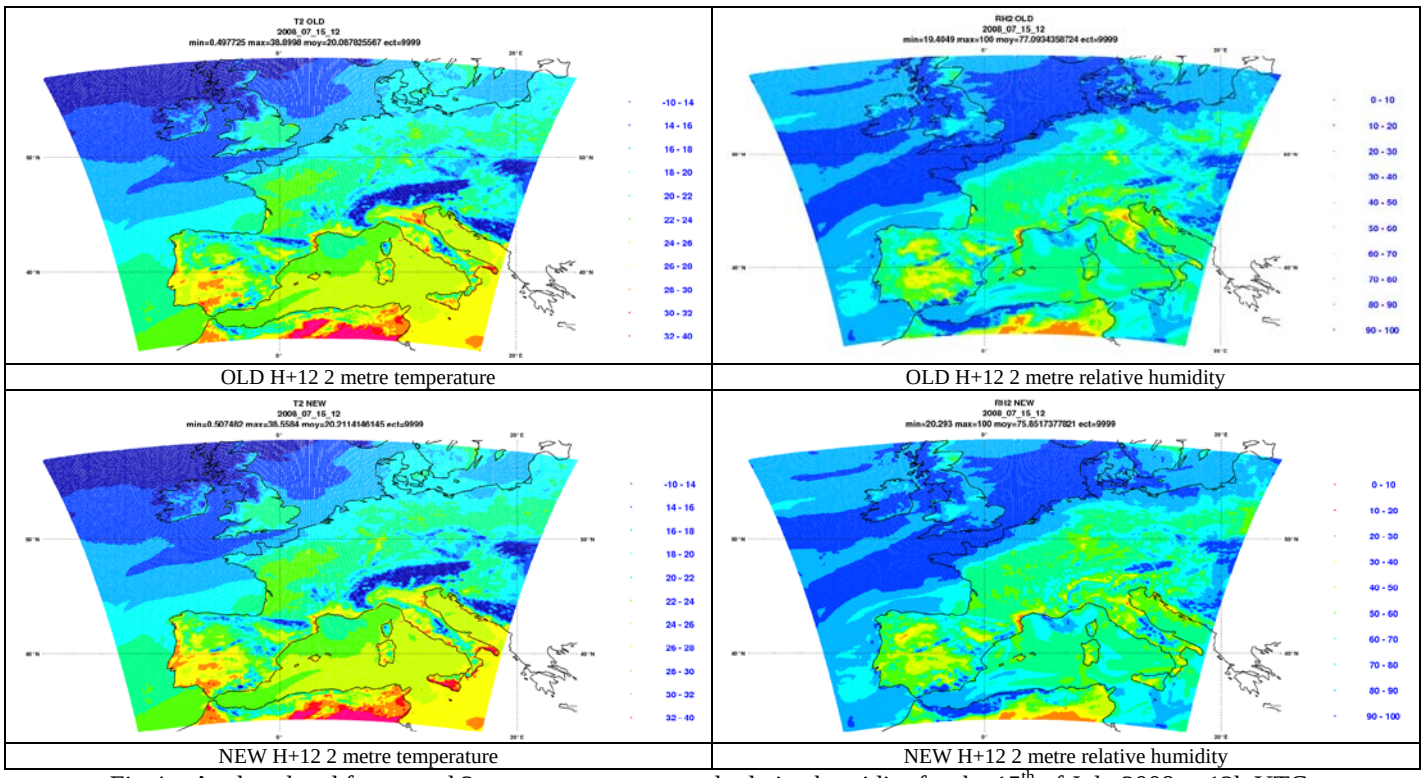


Fig.3 – ALADIN OPER SWI for the 15th of July 2008 at 00UTC

	
ANALYSED 2 metre temperature	ANALYSED 2 metre relative humidity
	
ALADIN OPER H+12 2 metre temperature	ALADIN OPER H+12 2 metre relative humidity
	
BLENDED H+12 2 metre temperature	BLENDED H+12 2 metre relative humidity

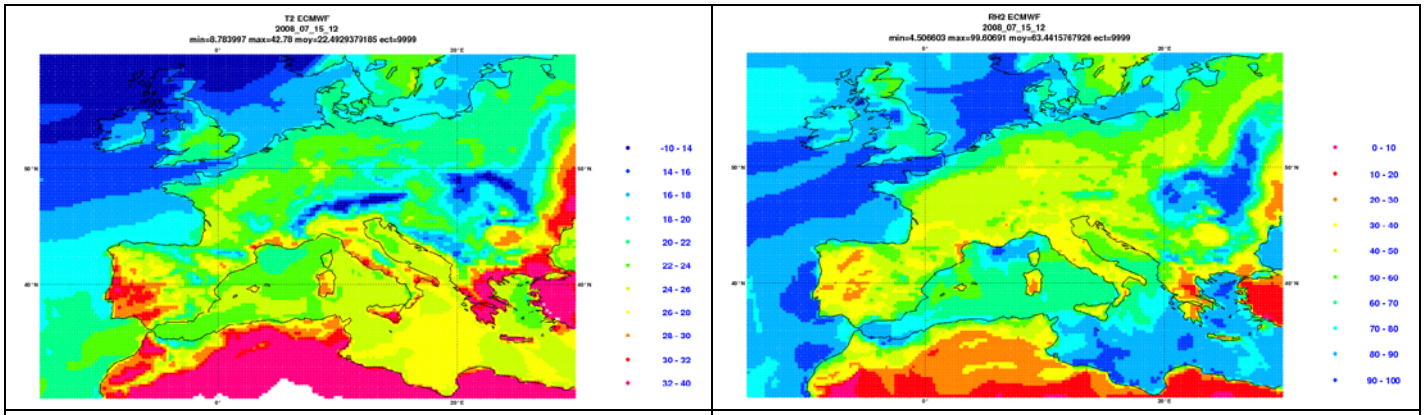


OLD H+12 2 metre temperature | OLD H+12 2 metre relative humidity

NEW H+12 2 metre temperature | NEW H+12 2 metre relative humidity

Fig.4 – Analysed and forecasted 2 metre temperature and relative humidity for the 15th of July 2008 at 12h UTC.

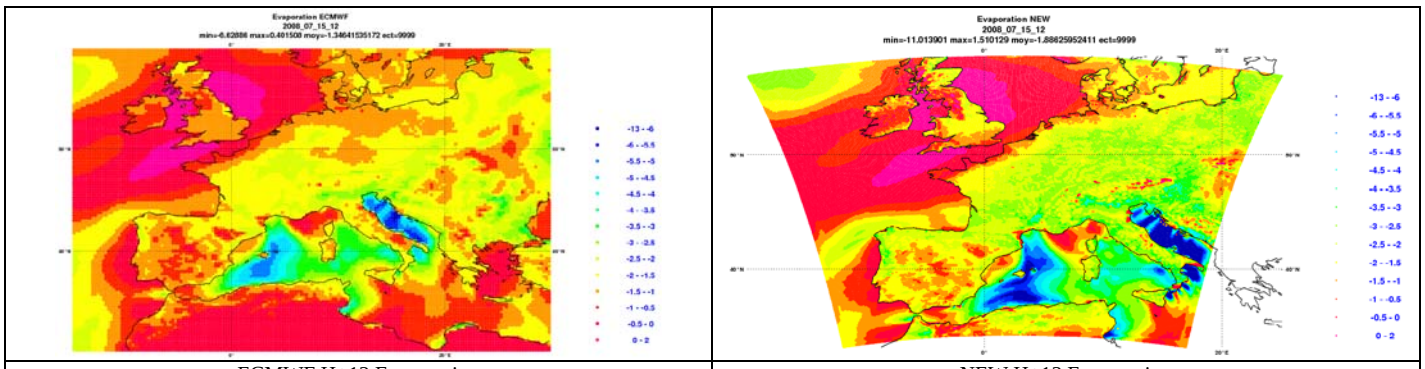
Both BLENDED H+12 (Obtained with Surface ISBA and upper-air IFS initial conditions) and ALADIN OPER H+12 with the ANALYSED fields, reproduce well the 2 metre temperature and relative humidity. The 2 metre temperature and relative humidity OLD H+12 and NEW H+12 are too cold and too wet. The ECMWF by itself also gives a good H+12 T2m and RH2m representation as shown in Fig.5.



ECMWF H+12 2 metre temperature | ECMWF H+12 2 metre relative humidity

Fig.5 – ECMWF H+12 2 metre temperature and relative humidity forecast for the 15th of July 2008 at 12h UTC

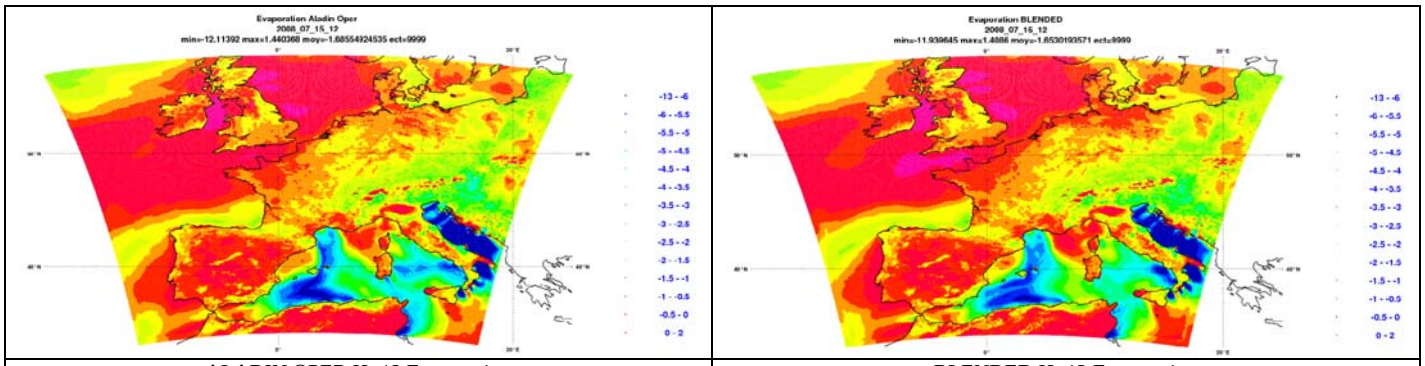
So, the ISBA model requires a much drier soil than ECMWF to produce similar near-surface weather parameters. The most reasonable cause is that ISBA creates much more evaporation than ECMWF with the same soil wetness. To prove this, Fig. 6 shows the H+12 accumulated Evaporation obtained with the NEW and with ECMWF.



ECMWF H+12 Evaporation	NEW H+12 Evaporation

Fig.6 – ECMWF H+12 and NEW H+12 Accumulated Evaporation forecast for the 15th of July 2008 at 12h UTC

In fact, the evaporation is considerably larger in NEW than in ECMWF, especially over the regions where the 2 metre temperature bias is larger.



ALADIN OPER H+12 Evaporation	BLENDED H+12 Evaporation

Fig.7 – ALADIN OPER H+12 and BLENDED H+12 Accumulated Evaporation Forecast for the 15th of July 2008 at 12h UTC

As expected, ALADIN (OPER and Blended) forced by surface ISBA initial conditions (much drier SWI) have much less evaporation than NEW and even less than ECMWF. Different evaporations with the same SWI, as seen in Figure 6, suggest that the interpolation used is not an effective evaporation-matching algorithm. The reasons for this and the possible solutions will be presented in the next sections.

5. The Evaporation problem

In the previous section, it was seen that using the same high SWI in IFS and ISBA, will produce too much accumulated evaporation inside ISBA code after 12h integration and that should be the reason for the high negative bias of 2 metre temperature.

What should be similar in both models is the Evaporation and not the soil wetness. For fully vegetated surfaces, assuming similar aero-dynamical resistances (a non-linear combination of wind speed and roughness length), the procedure should take into account that ISBA and IFS models are using different databases for minimum canopy resistance (Rsmin/LAI). If the minimum canopy resistance in ISBA is much lower than in IFS, than, for the same soil wetness, ISBA will have much more Evaporation than IFS.

In ISBA:

Rsmin is 'SURFRESI.STO.MIN'
LAI is 'SURFIND.FOLIAIRE'
It is enough to ask for these fields and just divide them.
In IFS:

$$LAI = LAI_l * cvl * c_{veg_l} + LAI_h * cvh * c_{veg_h}$$ [4]

$$Rs\min = Rs\min_l * cvl * c_{veg_l} + Rs\min_h * cvh * c_{veg_h}$$

Cvl is low vegetation cover (gribcode 27)
Cvh is high vegetation cover (gribcode 28)

with LAI_l, LAI_h, $Rs\min_l$ and $Rs\min_h$ taken from Table 2

Table 2 – Rsmin and Lai for each type of vegetation

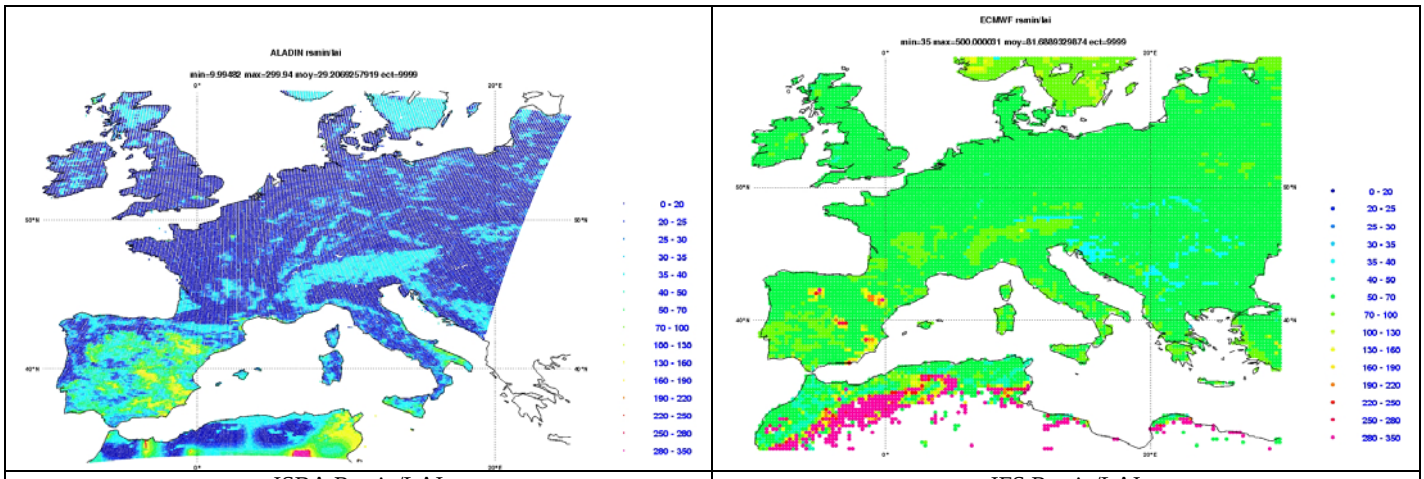
Index	Vegetation type	H/L	$r_{s,\min}$ (sm^{-1})	LAI (m^2m^{-2})	c_{veg}	g_D (hPa^{-1})	a_r	b_r
1	Crops, mixed farming	L	180	3	0.90	0	5.558	2.614
2	Short grass	L	110	2	0.85	0	10.739	2.608
3	Evergreen needleleaf trees	H	500	5	0.90	0.03	6.706	2.175
4	Deciduous needleleaf trees	H	500	5	0.90	0.03	7.066	1.953
5	Deciduous broadleaf trees	H	175	5	0.90	0.03	5.990	1.955
6	Evergreen broadleaf trees	H	240	6	0.99	0.03	7.344	1.303
7	Tall grass	L	100	2	0.70	0	8.235	1.627
8	Desert	–	250	0.5	0	0	4.372	0.978
9	Tundra	L	80	1	0.50	0	8.992	8.992
10	Irrigated crops	L	180	3	0.90	0	5.558	2.614
11	Semidesert	L	150	0.5	0.10	0	4.372	0.978
12	Ice caps and glaciers	–	–	–	–	–	–	–
13	Bogs and marshes	L	240	4	0.60	0	7.344	1.303
14	Inland water	–	–	–	–	–	–	–
15	Ocean	–	–	–	–	–	–	–
16	Evergreen shrubs	L	225	3	0.50	0	6.326	1.567
17	Deciduous shrubs	L	225	1.5	0.50	0	6.326	1.567
18	Mixed forest/woodland	H	250	5	0.90	0.03	4.453	1.631
19	Interrupted forest	H	175	2.5	0.90	0.03	4.453	1.631
20	Water and land mixtures	L	150	4	0.60	0	–	–

The vegetation type is subdivided in two categories :

Tvl is low vegetation type (gribcode 29)
Tvh is high vegetation type (gribcode 30)

Tvl and Tvh enter as Index in the previous table to access LAI_l, LAI_h, $Rs\min_l$ $Rs\min_h$, c_{veg_l} and c_{veg_h}.

This procedure generates Fig. 8 comparing (Rsmin/LAI)isba with (RSmin/LAI)ifs



ISBA Rsmin/LAI | IFS Rsmin/LAI

Fig.8 – ISBA and IFS Rsmin/LAI for the 15th of July 2008 at 12h UTC

[4] For the sake of simplification, the following figures show results obtained with cveg=1. Results presented here should not be qualitatively affected. The solution adopted in the final code should include the ECMWF definition of cveg.

The ISBA minimum canopy resistance is in fact much lower than the IFS minimum canopy resistance over large areas of the domain and this can explain the difference in evaporation.

If the aero-dynamical drag is small, the SWI scaling using the $\frac{\left[\frac{Rs\min}{LAI}\right]_{ISBA}}{\left[\frac{Rs\min}{LAI}\right]_{IFS}}$ factor has physical meaning and will produce a drier soil as required by ISBA code, producing approximately the same evaporation as ECMWF and ALADIN OPER, leading to better 2 metre temperature and relative humidity fields.

This LAI scaling was introduced in the cprep1 routine, as the last test (called LAIscal) before finishing this report and as a proof of concept for future work. The LAIscal SWI is shown in Fig. 9.

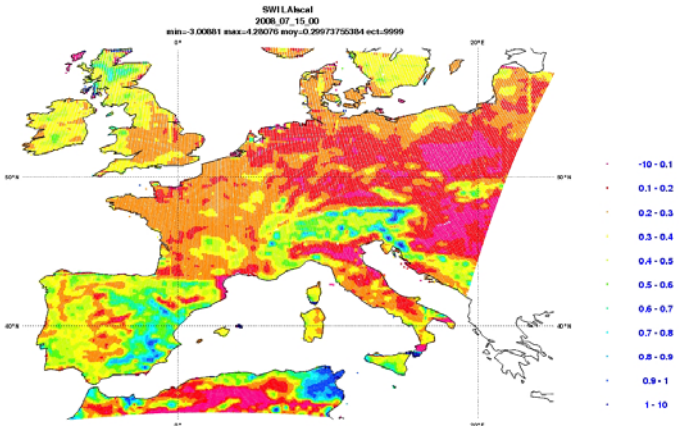


Fig.9 – LAIscal SWI for the 15th of July 2008 at 00UTC

There are some regions where this procedure seems to have dried up the soil too much, namely in the British Isles and Eastern Europe. Over France the SWI is now intermediate between ALADIN OPER and SIM SWI. Over the Iberian Peninsula maybe the soil is not dry enough over some areas and too dry over other areas; but all these comments deserve a deeper analysis.

Fig. 10 shows the 12h accumulated evaporation forecast.

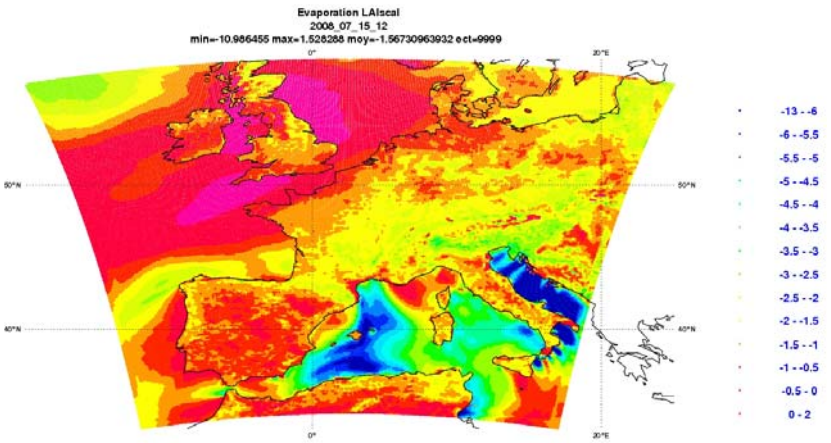
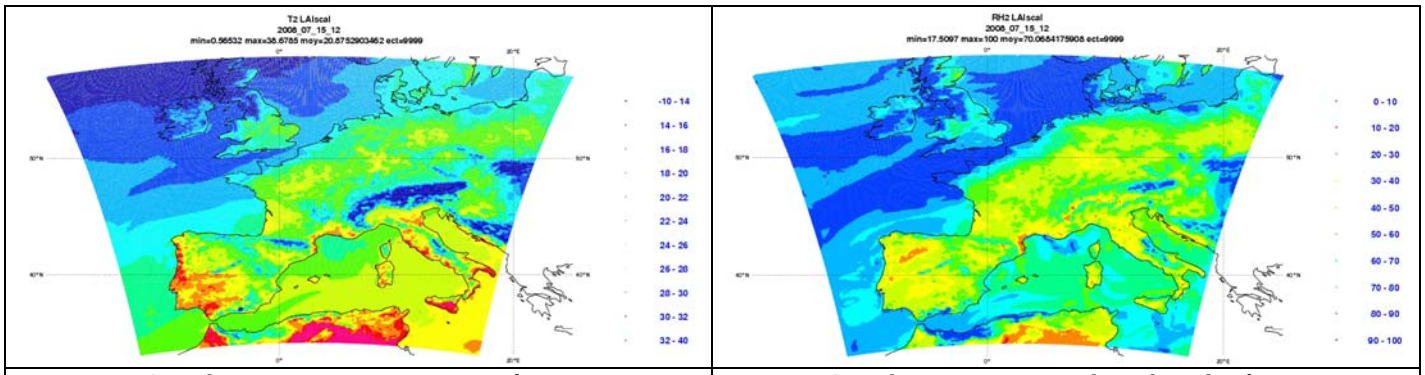


Fig.10 – LAIscal H+12 Accumulated Evaporation forecast for the 15th of July 2008 at 12UTC

This is similar to the evaporation of ALADIN OPER presented at Fig. 7.

Fig. 11 shows the 2 metre temperature and relative humidity forecast, obtained with the Rsmin/LAI scaling.



LAI scaling H+12 2 metre temperature forecast	LAI scaling H+12 2 metre relative humidity forecast

Fig. 11 – Forecast 2 metre temperature and relative humidity for the 15th of July 2008 at 12h UTC

This is considerably warmer and drier than the NEW scheme and much closer to the ANALYSED field of Fig. 4.

6. Conclusions and perspectives

A new interpolation method preserving “Soil Wetness Index” rather than “Moisture Ratio” has been coded in “901 configuration” which improves soil moisture initialization of ISBA from H-TESSEL. However it is still overestimating soil moisture. Some additional information should be taken into account such as differences in surface parameterizations or physiographic parameters. For instance the Rsmin/LAI value has been found very different in H-TESSEL and ISBA. A first trial has been coded to take into account this difference but it is too crude to be satisfactory. The work should be continued to better understand the ISBA and H-TESSEL characteristics including physiographic databases leading to differences on surface evapo-transpiration. The interpolation of soil temperatures, frozen soil moisture and snow from H-TESSEL to ISBA should be also studied carefully.

7. References

Descamp, L. (2007): Passage IFS-ARPEGE et schémas de surface. French Internal Note.

Annex I

Cprep1.F90 routine

```
SUBROUTINE CPREP1

USE PARKIND1  ,ONLY : JPIM     ,JPRB
USE YOMHOOK   ,ONLY : LHOOK,   DR_HOOK

USE YOMDIM   , ONLY : NDGLG    ,NDGNH    ,NDGSAG   ,NDGENG   ,&
 & NDLON    ,NFLEVG   ,NSMAX    ,NSEFRE   ,&
 & NSPEC    ,NSPEC2   ,NS3D     ,NS2D
USE YOMCT0   , ONLY : CNMEXP   ,NCONF
USE YOMCST   , ONLY : RG       ,R
USE YOMLUN   , ONLY : NULOUT   ,NULNAM   ,NPOSSH   ,NINISH
USE YOMGEM   , ONLY : NLOENG   ,NHTYP    ,NGPTOTG
USE YOMLAP   , ONLY : NVALUE   ,RLAPIN
USE SURFACE_FIELDS, ONLY : &
 & YSP_SB, YSP_SBD, YSP_SG, YSP_SGD, YSP_RR, YSP_RRD, &
 & YSD_WS, YSD_WSD, YSP_EP, YSP_EPD, YSP_X2, YSP_X2D,&
 & YSD_VF, YSD_VFD, YSD_VP, YSD_VPD

USE YOMGRB   , ONLY : NGRBSWL1 ,NGRBSD   ,NGRBSDOR ,NGRBSWL2 ,&
 & NGRBSR   ,NGRBSWL3 ,NGRBSRC  ,NGRBSWL4, NGRBCVL ,&
 & NGRBCVH  , NGRBTVL, NGRBTVH
USE YOMOP    , ONLY : CNMCA
USE YOMMSC   , ONLY : NINTLEN
USE YOMPHY1  , ONLY :      RSMAX    ,NTVMER,  RD1,&
 & RD2MER, GCONV
USE YOMCLI   , ONLY : SARGN    ,SSABN, SDEPX
USE YOEVDF   , ONLY : NVTYPES

#ifdef DOC

! **** *CPREP1*  - Transform GRIB file to FA file (NCONF=901)

! Purpose.
! --------
! 901 : GRIB file format

! **   Interface.
! ----------
! *CALL* *CPREP1

! Explicit arguments :
! --------------------

! Implicit arguments :
! --------------------
! None

! Method.
```

```
! -------
! See documentation

! Externals.
! ----------

! Reference.
! ----------
! ECMWF Research Department documentation of the IFS
! Note de travail ARPEGE NR 17 et  xx

! Author.
! -------
! Mats Hamrud and Philippe Courtier  *ECMWF*

! Modifications.
! --------------
! Original : 87-10-15
! Modification : 90-07-20 (Yves Bouteloup *EERM/CRMD*)
! ALL ACTIVES LINES
! Modification : 91-10-08 (Erik Andersson *ECMWF*)
! Modification : 91-10-27 (Mats Hamrud *ECMWF*)
! Modification : 91-10-27 (J.-J. Morcrette *ECMWF*)
! Modification : 92-04-15 (Y.Bouteloup *DMN*) Addition of
! poles in FA file
! Modification : 92-04-29 (J.-J. Morcrette *ECMWF*)
! Modification : 92-08-01 (Mats Hamrud *ECMWF*)
! Surface fields IO-scheme
! Modification : 94-02-02 (J.M. Piriou *CNRM/GMAP*)
! Initial date of FA output file read on GRIB input file.
! Orography anisotropy and direction
! read from climatological file.
! Modification : 95-06-21 (J.M. Piriou *CNRM/GMAP*)
! Grib file directed reading
! instead of namelist directed reading.
! Modification : 95-12-20 (J.M. Piriou *CNRM/GMAP*)
! Use PBOPEN instaed of BUFFER IN.
! Modified 98-08-10 by K. YESSAD:
!  - removal of LRPOLE option, no pole written on ARPEGE files, argument
!    .FALSE removed from 'CALL SPEREE'.
!  - use only the DM-global variables NDGLG, NDGSAG, NDGENG, NLOENG.
!  - some optimisations using NGPTOTG rather than (NDLON,NDGLG).
!  - ZFAGG and ZFAGGA have been redimensioned with NGPTOTG.
! Modified 98-11-20 by P. Saez and J.M. Piriou:
!     debug an error in soil wetness computations.
! Modified 98-12-02 by P. Saez and J.M. Piriou:
!     introduce 134 GRIB parameter (SURFACE PRESSURE).
! Modified 99-03-01 (Patrick SAEZ *CNRM/GMAP*)
!  - Cleaning on Conf. 902
!  - Corrections date for year 2000
!  - debug an error in Surface emissivity field
!  - Remove YOMMARS
```

```
! Modified 99-04-20 (Patrick SAEZ *CNRM/GMAP*)
!  - Add ISBA fields
!     Climatological values can be used by LLCLIM switch.
!    R. El Khatib : 99-12-28 remove poles rows
! Modified 00-06-30 (Patrick SAEZ *CNRM/GMAP*)
!  Computations of SURFRESERV.EAU and PROFRESERV.EAU whith SWVLi instead of
SWLi
! Modified 00-10-25 (Patrick SAEZ *CNRM/GMAP*)
!  Computations of Percentage of Vegetation whith new mars fields(27/06/00-> )
! Modified 00-12-12 (Patrick SAEZ *CNRM/GMAP*)
!  portage en distribue 1 PROC.
!       R. El Khatib : 01-08-07 Pruning options
! Modified 02-02-04 (Patrick SAEZ *CNRM/GMAP*)
!  Corrections on conversion GRIB DATE to ARPEGE DATE
! Modified 03-02-25 (Patrick SAEZ *CNRM/GMAP*)
!  Add LLOLDSWL,LLHUPDG,LLCONTROL options
! Modified 03-04-10 (Patrick SAEZ *CNRM/GMAP*)
!  Add computations of 'SURFALBEDO
NEIGE','SURFDENSIT.NEIGE','SURFALBEDO.SOLNU',
! 'SURFALBEDO.VEG' and LLALBEDO2 switch.
! Modified 03-04-17 (Patrick SAEZ *CNRM/GMAP*)
!  compute estimated values of SURFRESERV.GLACE and PROFRESERV.GLACE
!    R. El Khatib : 03-08-18 Roughness lengths not packed
!       M.Hamrud      01-Oct-2003 CY28 Cleaning
!      D. Paradis & R. El Khatib : 04-07-22 GRIBEX
!      R. Brozkova : 05-10-31 Set forecast range in the file date.
!      R. Brozkova: 28-07-06 allow the treatment of the upper-air only by
!                          relaxing the LSM test via LLCONTROL switch
! Modified 07-11-21 (Patrick SAEZ *CNRM/GMAP*)
!   Add aerosols and ozone climatological fields
!   Compute RESERV_EAU with new SLT MARS field (Soil Moisture Index)
! ------------------------------------------------------------------
#endif

IMPLICIT NONE

INTEGER(KIND=JPIM),PARAMETER :: JPFILN=10 ! max number of input GRIB files to
be read
INTEGER(KIND=JPIM),PARAMETER :: JPSTOIA=4 ! maximum number of intermediate
grid-point arrays to be stored.
INTEGER(KIND=JPIM),PARAMETER :: JPNTVMER=1 !valeur sur mer dans PIVEG
(type de surface)
INTEGER(KIND=JPIM),PARAMETER :: JPNTVTER=3 !valeur sur terre dans PIVEG
(type de surface)
INTEGER(KIND=JPIM),PARAMETER :: JPOLDSWL1=140 !value of NGRBSWL1
before june 2000
INTEGER(KIND=JPIM),PARAMETER :: JPOLDSWL2=171 !value of NGRBSWL2
before june 2000
INTEGER(KIND=JPIM),PARAMETER :: JPOLDSWL3=184 !value of NGRBSWL3
before june 2000
INTEGER(KIND=JPIM),PARAMETER :: JPOLDSWL4=237 !value of NGRBSWL4
before june 2000
```

```
        REAL(KIND=JPRB), PARAMETER   :: PPWSAT_CEP=0.472_JPRB ! Teneur en eau a la
saturation au CEP
        REAL(KIND=JPRB), PARAMETER   :: PPWWILT_CEP=0.171_JPRB ! Point de
fletrissement au CEP
        REAL(KIND=JPRB), PARAMETER   :: PPWCAP_CEP=0.323_JPRB ! Capacite au champ
au CEP
        REAL(KIND=JPRB), PARAMETER   :: PPPSWL1=0.07_JPRB ! Reservoir superficiel du
CEP
        REAL(KIND=JPRB), PARAMETER   :: PPPSOL=1.5_JPRB ! Prodondeur pour SWL2 si
LLCLIM=.false.
        ! epaisseur des couches au CEP
        REAL(KIND=JPRB), PARAMETER   :: PPZD1=0.07_JPRB
        REAL(KIND=JPRB), PARAMETER   :: PPZD2=0.21_JPRB
        REAL(KIND=JPRB), PARAMETER   :: PPZD3=0.72_JPRB
        REAL(KIND=JPRB), PARAMETER   :: PPZD4=1.89_JPRB

        INTEGER(KIND=JPIM), ALLOCATABLE :: IGRIB(:)
        REAL(KIND=JPRB),    ALLOCATABLE :: ZGRBDAT(:)

        INTEGER(KIND=JPIM) :: IDATEF(11)
        INTEGER(KIND=JPIM) :: ISEC0(2),ISEC1(60),ISEC2(22+NDGLG)
        INTEGER(KIND=JPIM) :: ISEC3(2),ISEC4(42)
        INTEGER(KIND=JPIM) :: IDATEC(11) ! Date of climatological file if LLCLIM
        INTEGER(KIND=JPIM) :: IGRBS(JPSTOIA) ! grib codes of the stored fields.
        INTEGER(KIND=JPIM) :: IUCLIM,IDEB2,IFIN,IINDEX
        INTEGER(KIND=JPIM) :: ISWL1,ISWL2,ISWL3,ISWL4,INBCLIB,INBCGLA

        REAL(KIND=JPRB) :: ZSEC2(10+2*(NFLEVG+1)),ZSEC3(2)
        REAL(KIND=JPRB) :: ZFASP(NSPEC2)    ! Spectral data to write on FA file.
        REAL(KIND=JPRB) :: ZFASPS(NSPEC2)   ! Spectral data to send/receive to
reespe/speree.
        REAL(KIND=JPRB) :: ZFPDAT(NSEFRE)   ! Spectral working array.
        REAL(KIND=JPRB) :: ZFAGG(NGPTOTG)   ! Grid-point data to write on FA file.
        REAL(KIND=JPRB) :: ZFAGGA(NGPTOTG,JPSTOIA) ! Grid-point data to store.
        REAL(KIND=JPRB) :: ZFACDST(NGPTOTG) ! store CDST to compute
PROFTEMPERATURE
        REAL(KIND=JPRB) :: ZFASTL4(NGPTOTG) ! store STL4 to compute
PROFTEMPERATURE
        REAL(KIND=JPRB) :: ZFALSM(NGPTOTG)  ! store Land/Sea Mask
        REAL(KIND=JPRB) :: ZFZ0G(NGPTOTG)   ! store SURFZ0.FOIS.G to compute
SURFGZ0.THERM
        REAL(KIND=JPRB) :: ZFCVL(NGPTOTG)   ! store CVL (Low Vegetation Cover)
        REAL(KIND=JPRB) :: ZFCVH(NGPTOTG)   ! store CVH (High Vegetation Cover)
        REAL(KIND=JPRB) :: ZFTVL(NGPTOTG)   ! store Type of Low Vegetation
        REAL(KIND=JPRB) :: ZFTVH(NGPTOTG)   ! store Type of High Vegetation

        ! Array used to compute SURFRESERV.EAU and PROFRESERV.EAU in ISBA scheme
        REAL(KIND=JPRB) :: ZFSWL1(NGPTOTG)  ! store SWL1
        REAL(KIND=JPRB) :: ZFSWL2(NGPTOTG)  ! store SWL2
        REAL(KIND=JPRB) :: ZFSWL3(NGPTOTG)  ! store SWL3
        REAL(KIND=JPRB) :: ZFSWL4(NGPTOTG)  ! store SWL4
```

```
! Array used in argument to ACSOLW to find Wsat_isba (ZFALSM is also used)
REAL(KIND=JPRB) :: ZFPARG(NGPTOTG)  ! store ARGILE
REAL(KIND=JPRB) :: ZFPSAB(NGPTOTG)  ! store SABLE
REAL(KIND=JPRB) :: ZFPIVEG(NGPTOTG) ! store TYPE DE SURFACE
REAL(KIND=JPRB) :: ZFPD2(NGPTOTG)   ! store PD2 (SURFEPAIS.SOL)

REAL(KIND=JPRB) :: ZFPLAI(NGPTOTG)  ! store LAI (SURFIND.FOLIAIRE)
REAL(KIND=JPRB) :: ZFPRSMIN(NGPTOTG)  ! store RSMIN (SURFRESI.STO.MIN)
REAL(KIND=JPRB) :: ZFLAI(NGPTOTG)  ! store LAI (ECMWF)
REAL(KIND=JPRB) :: ZFRSMIN(NGPTOTG)  ! store RSMIN (ECMWF)

! Work Array for ACSOLW()
REAL(KIND=JPRB) :: ZFPWFC(NGPTOTG)
REAL(KIND=JPRB) :: ZFPWPMX(NGPTOTG)
REAL(KIND=JPRB) :: ZFPWSAT(NGPTOTG)
REAL(KIND=JPRB) :: ZFPWSMX(NGPTOTG)
REAL(KIND=JPRB) :: ZFPWWILT(NGPTOTG)

! Array used to store SURFTEMPERATURE and PROFTEMPERATURE if
LLGLACE=.T.
REAL(KIND=JPRB) :: ZFSURFT(NGPTOTG)
REAL(KIND=JPRB) :: ZFPROFT(NGPTOTG)
REAL(KIND=JPRB) :: ZFGLACE(NGPTOTG)

! NAMELIST variables for SURFRESERV.GLACE and PROFRESERV.GLACE
REAL(KIND=JPRB) :: TSRESERV1,TSRESERV2,TDELTA1,TDELTA2

REAL(KIND=JPRB) :: ZRVCOV(0:NVTYPES)
REAL(KIND=JPRB) :: ZRVLAI(0:NVTYPES)
REAL(KIND=JPRB) :: ZRVRSMIN(0:NVTYPES)

INTEGER(KIND=JPIM) :: IDATRT, IDATRTG, IDATRTS, IDONR, IDONRN,&
 & IDONRX, IERR, IGRBSN, ILENBYT, ILENWOR, ILEVEL, &
 & ILEVTY, ILONS, INBARI, INBARP, &
 & INDEX, INIVEAU, INTIMES, IOK, IPARAM, IREP, &
 & IREPRM, IRET, IWORD, JB, JC, JFILN, &
 & JGL, JGPTOTG, JINDEX, JM, JN, JNM, &
 & JSTOIA, JV, IIND, NBUF901

INTEGER(KIND=JPIM) ::
INGRIG,INGRIB,INBPDG,INBCSP,ISTRON,IPUILA,IDMOPL

LOGICAL :: LL3DMOD, LLFIRAR, LLGRB190,&
 & LLGRB191, LLGRB192, LLGRB193, &
 & LLINPFS, LLOUTFS, LLSCAL, LLWRIF, &
 & LLGLACE, LLCLIM, LLLSM, LLCDST, LLSTL4, LLZ0, &
 & LLSWL1, LLSWL2, LLSWL3, LLSWL4, LLSLT, LLCVL, LLCVH, LLTVL, LLTVH,
&
 & LLOLDSWL, LLHUPDG, LLCONTROL, LLALBEDO2, LLSURFT, LLAEROSOL
LOGICAL :: LLFILE_TO_MODEL

CHARACTER (LEN = 80) :: CLFAFILN
```

```
CHARACTER (LEN = 80) :: CLLIBELLE
CHARACTER :: CLPREF*4,CLSUFF*16
CHARACTER (LEN = 200) ::  CLFILN(JPFILN)

REAL(KIND=JPRB) :: ZGOL, ZMAX, ZMEA, ZMIN, ZSCALE, ZVAL, ZPSOL,
ZPGLACE, ZDELTA, ZWSATSLT
REAL(KIND=JPRB) :: LAIscal
REAL(KIND=JPRB) :: ZGOL1, ZGOL2, ZGOL3, ZGOL4
REAL(KIND=JPRB) ::
ZPWWILT_CEP,ZPWCAP_CEP,SWI_CEP,SWI_CEP1,SWI_CEP2,SWI_CEP3,SWI_CEP4
REAL(KIND=JPRB) :: ZFSWL1_aux,ZFSWL2_aux,ZFSWL3_aux,ZFSWL4_aux

! WP_SAT, WP_WILT, WP_CAP and Soil Moisture Index
REAL(KIND=JPRB), DIMENSION(0:7) :: ZWP_SAT
REAL(KIND=JPRB), DIMENSION(0:7) :: ZWP_WILT
REAL(KIND=JPRB), DIMENSION(0:7) :: ZWP_CAP
REAL(KIND=JPRB) :: ZFSLT(NGPTOTG)  ! store SLT

! ** Array of fields to be simply done if LLCLIM
CHARACTER(len=20), DIMENSION(20) :: CLLIB =&
 & (/'SURFALBEDO          ',&
 & 'SURFEMISSIVITE      ','SURFEPAIS.SOL       ','SURFZ0.FOIS.G       ',&
 & 'SURFGZ0.THERM       ','SURFIND.FOLIAIRE    ','SURFIND.VEG.DOMI    ',&
 & 'SURFPROP.ARGILE     ','SURFPROP.SABLE      ','SURFPROP.VEGETAT    ',&
 & 'SURFRESI.STO.MIN    ','SURFALBEDO.SOLNU    ','SURFALBEDO.VEG      ',&
 & 'SURFA.OF.OZONE      ','SURFB.OF.OZONE      ','SURFC.OF.OZONE      ',&
 & 'SURFAEROS.SEA       ','SURFAEROS.LAND      ','SURFAEROS.SOOT      ',&
 & 'SURFAEROS.DESERT    '/)

! ** Structure pour les champs (constants) qui n'existent pas au CEP
! (depend parfois du Land/Sea Mask)

TYPE CHAMPSCST
CHARACTER(len=20) :: CLHAMP    ! Nom du champ
REAL(KIND=JPRB)           :: ZVALTERRE ! Valeur sur terre (ou globale)
REAL(KIND=JPRB)           :: ZVALMER   ! Valeur sur mer
LOGICAL           :: LLISLSM   ! .TRUE. si depend du LS Mask
END TYPE CHAMPSCST

TYPE(CHAMPSCST),DIMENSION(11) :: YL_FIELDCST
REAL(KIND=JPRB) :: ZHOOK_HANDLE


INTERFACE
#include "surf_inq.h"
END INTERFACE
#include "abor1.intfb.h"
#include "acsolw.intfb.h"
#include "openfa.intfb.h"
#include "posnam.intfb.h"
#include "reespe.intfb.h"
#include "speree.intfb.h"
```

```
#include "val923.intfb.h"
#include "spreord.intfb.h"

#include "nammars.h"

! ** Initialization.
IF (LHOOK) CALL DR_HOOK('CPREP1',0,ZHOOK_HANDLE)
YL_FIELDCST(1)=CHAMPSCST('SURFRESERV.GLACE    ',0.,0.,.FALSE.)
YL_FIELDCST(2)=CHAMPSCST('PROFRESERV.GLACE    ',0.,0.,.FALSE.)
YL_FIELDCST(3)=CHAMPSCST('SURFRES.EVAPOTRA    ',1.,1.,.FALSE.)
YL_FIELDCST(4)=CHAMPSCST('SURFRESERV.INTER    ',0.,0.,.FALSE.)
YL_FIELDCST(5)=CHAMPSCST('SURFEMISSIVITE      ',0.96,0.,.FALSE.)
YL_FIELDCST(6)=CHAMPSCST('SURFEPAIS.SOL
',PPPSOL,REAL(RD2MER),.TRUE.)
YL_FIELDCST(7)=CHAMPSCST('SURFIND.FOLIAIRE    ',2.,0.,.TRUE.)
YL_FIELDCST(8)=CHAMPSCST('SURFIND.VEG.DOMI
',3.,REAL(NTVMER),.TRUE.)
YL_FIELDCST(9)=CHAMPSCST('SURFPROP.ARGILE     ',30.,REAL(SARGN),.TRUE.)
YL_FIELDCST(10)=CHAMPSCST('SURFPROP.SABLE
',45.,REAL(SSABN),.TRUE.)
YL_FIELDCST(11)=CHAMPSCST('SURFRESI.STO.MIN
',150.,REAL(RSMAX),.TRUE.)

! values associated with SLT coeff.
ZWP_SAT = (/PPWSAT_CEP, 0.403_JPRB, 0.439_JPRB, 0.430_JPRB, 0.520_JPRB,
0.614_JPRB, &
        & 0.766_JPRB, 0.439_JPRB/)
ZWP_WILT = (/PPWWILT_CEP, 0.059_JPRB, 0.151_JPRB, 0.133_JPRB, 0.279_JPRB, &
        & 0.335_JPRB, 0.267_JPRB, 0.151_JPRB/)
ZWP_CAP = (/PPWCAP_CEP, 0.242_JPRB, 0.346_JPRB, 0.382_JPRB, 0.448_JPRB, &
        & 0.541_JPRB, 0.662_JPRB, 0.346_JPRB/)

LLFIRAR=.TRUE. ! .true. if first grib article read.
LLGRB190=.FALSE. ! .true. if this GRIB code filed has been read from file.
LLGRB191=.FALSE.
LLGRB192=.FALSE.
LLGRB193=.FALSE.
IGRBSN=0 ! number of intermediate grid-point arrays stored.
LLCLIM=.FALSE. ! .true. if climatological file is used
LLLSM=.FALSE.  ! .true. id Land/sea Mask is read
LLCDST=.FALSE. ! to compute PROFTEMPERATURE whith CDST
LLGLACE=.FALSE.  ! .true. to compute ice reservoir
LLSTL4=.FALSE. ! to compute PROFTEMPERATURE whith STL4
LLZ0=.FALSE.   ! to compute SURFGZ0.THERM whith SURFZ0.FOIS.G if read
LLSWL1=.FALSE.
LLSWL2=.FALSE.
LLSWL3=.FALSE.
LLSWL4=.FALSE.
LLSLT=.FALSE.
LLCVL=.FALSE.
LLCVH=.FALSE.
LLTVL=.FALSE.
```

```
LLTVH=.FALSE.
LLOLDSWL=.FALSE. ! used to compute SWLi before june 2000 if true.
LLHUPDG=.FALSE.  ! used to not write U% in a spectral way if true.
LLCONTROL=.TRUE. ! to not abort (if false) and force to write fields.
LLALBEDO2=.TRUE.  ! used to compute 'SURFALBEDO.SOLNU' and
'SURFALBEDO.VEG'
LLAEROSOL=.TRUE. ! use to add aerosols and ozone climatological fields
LLSURFT=.FALSE.
INBCLIB=20       ! default: LLAEROSOL=.TRUE.
INBCGLA=1
TSRESERV1=273.15_JPRB
TSRESERV2=273.15_JPRB
TDELTA1=3.0_JPRB
TDELTA2=7.0_JPRB
NBUF901=1500000_JPIM

! ** Read namelist.

DO JFILN=1,JPFILN
  CLFILN(JFILN)=' '
ENDDO
CALL POSNAM(NULNAM,'NAMMARS')
READ(NULNAM,NAMMARS)

IF (LLOLDSWL) THEN
  ISWL1=JPOLDSWL1
  ISWL2=JPOLDSWL2
  ISWL3=JPOLDSWL3
  ISWL4=JPOLDSWL4
ELSE
  ISWL1=NGRBSWL1
  ISWL2=NGRBSWL2
  ISWL3=NGRBSWL3
  ISWL4=NGRBSWL4
ENDIF

IF (.NOT.LLAEROSOL) INBCLIB=13
IF (.NOT.LLALBEDO2) INBCLIB=11

!Allocate buffers to read and stock grib
ALLOCATE(IGRIB(NBUF901), STAT=IRET)
IF (IRET /= 0) THEN
  WRITE(NULOUT,*) '* NBUF901=',NBUF901,' IRET=',IRET
  CALL ABOR1('CPREP1: PROBLEM IN ALLOCATE IGRIB buffer')
ENDIF

ALLOCATE(ZGRBDAT(NBUF901), STAT=IRET)
IF (IRET /= 0) THEN
  WRITE(NULOUT,*) '* NBUF901=',NBUF901,' IRET=',IRET
  CALL ABOR1('CPREP1: PROBLEM IN ALLOCATE ZGRBDAT buffer')
ENDIF
```

```
DO JFILN=1,JPFILN
  IF(CLFILN(JFILN) /= ' ') THEN

! ** Open grib file.

    WRITE(NULOUT,FMT='(2a)')&
     & 'INITIAL GRIB DATA TO BE READ FROM FILE ',CLFILN(JFILN)
    CALL PBOPEN(NINISH,CLFILN(JFILN),'r',IRET)
    IF(IRET /= 0) THEN
      WRITE(NULOUT,'(A,I2)')'CPREP1: PROBLEM IN PBOPEN, IRET=',IRET
      CALL ABOR1('CPREP1: PROBLEM IN PBOPEN')
    ENDIF
    ILENWOR=NBUF901 ! array dimension in words.
    ILENBYT=ILENWOR*NINTLEN ! array dimension in bytes.

! Control dimensioning.

    WRITE(NULOUT,*) '* CPREP1 outputs:'
    WRITE(NULOUT,*) 'ILENWOR=',ILENWOR
    WRITE(NULOUT,*) 'ILENBYT=',ILENBYT
    WRITE(NULOUT,*) 'NSPEC=',NSPEC
    WRITE(NULOUT,*) 'NSPEC2=',NSPEC2
    WRITE(NULOUT,*) 'NSEFRE=',NSEFRE
    WRITE(NULOUT,*) 'NDLON=',NDLON
    WRITE(NULOUT,*) 'NDGLG=',NDGLG
    WRITE(NULOUT,*) 'NFLEVG=',NFLEVG
    WRITE(NULOUT,*) 'NLOENG='
    WRITE(NULOUT,'(25(1X,I4))')(NLOENG(JGL),JGL=NDGSAG,NDGENG)
    WRITE(NULOUT,*) 'NGPTOTG=',NGPTOTG
    WRITE(NULOUT,*) 'LLCLIM=',LLCLIM
    WRITE(NULOUT,*) 'LLOLDSWL=',LLOLDSWL
    WRITE(NULOUT,*) 'LLHUPDG=',LLHUPDG
    WRITE(NULOUT,*) 'LLCONTROL=',LLCONTROL
    WRITE(NULOUT,*) 'LLALBEDO2=',LLALBEDO2
    WRITE(NULOUT,*) 'LLAEROSOL=',LLAEROSOL
    WRITE(NULOUT,*) 'LLGLACE=',LLGLACE
    WRITE(NULOUT,*) 'TSRESERV1=',TSRESERV1
    WRITE(NULOUT,*) 'TSRESERV2=',TSRESERV2
    WRITE(NULOUT,*) 'TDELTA1=',TDELTA1
    WRITE(NULOUT,*) 'TDELTA2=',TDELTA2
    WRITE(NULOUT,*) 'NBUF901=',NBUF901

    100 CONTINUE

! ** Read grib article.

    WRITE(NULOUT,*)'---------------------------------------------'
    IF(NCONF == 901) THEN
      IRET=0
      CALL PBGRIB(NINISH,IGRIB,ILENBYT,ILONS,IRET)
      IF(IRET == 0) THEN
      ELSEIF(IRET == -1) THEN
```

```
! End of file.

      GOTO 200
    ELSE
      WRITE(NULOUT,'(A,I2)')'CPREP1: PROBLEM IN PBGRIB, IRET=',IRET
      CALL ABOR1('CPREP1: PROBLEM IN PBGRIB')
    ENDIF
  ELSE
    CALL ABOR1('ERROR CPREP1 NCONF!')
  ENDIF

! ** Decode grib article.

  ZGOL=8.888888E20_JPRB
  DO JB=1,ILENWOR
    ZGRBDAT(JB)=ZGOL
  ENDDO
  CALL GSTATS(1703,0)
  CALL GRIBEX(ISEC0,ISEC1,ISEC2,ZSEC2,ISEC3,ZSEC3,ISEC4,&
   & ZGRBDAT,ILENWOR,IGRIB,ILENWOR,IWORD,'D',IERR)
  CALL GSTATS(1703,1)
  IF(IERR /= 0) THEN
    WRITE(NULOUT,*) 'CPREP1: PROBLEM IN GRIBEX, IERR=',IERR
    CALL ABOR1('ERROR CPREP1 GRIBEX!')
  ENDIF

! Check extrema of decoded field.

  ZMIN=ZGRBDAT(1)
  ZMAX=ZGRBDAT(1)
  ZMEA=0.0_JPRB
  IDONR=0
  IDONRN=ILENWOR
  IDONRX=1
  DO JB=1,ILENWOR
    IF(ZGRBDAT(JB) /= ZGOL) THEN
      IDONR=IDONR+1
      IDONRN=MIN(IDONRN,JB)
      IDONRX=MAX(IDONRX,JB)
      ZMIN=MIN(ZMIN,ZGRBDAT(JB))
      ZMAX=MAX(ZMAX,ZGRBDAT(JB))
      ZMEA=ZMEA+ZGRBDAT(JB)
    ENDIF
  ENDDO
  ZMEA=ZMEA/IDONR
  WRITE(NULOUT,'(a,i4,a,i8,3(a,e16.7))')&
   & ' Parameter ',ISEC1(6),' read: size '&
   & ,IDONR,' min ',ZMIN,' max ',ZMAX,' mea ',ZMEA

  IPARAM=ISEC1(6) ! grib code of the field.
  ILEVTY=ISEC1(7) ! type of level.
```

```
          ILEVEL=ISEC1(8) ! level.
          IDATRT=ISEC2(1) ! data representation type.
          IREPRM=ISEC2(6) ! resolution flag.

          IF(IDONR > NGPTOTG) THEN
            WRITE(NULOUT,*) 'WARNING: skip parameter with incorrect resolution
IPARAM=',IPARAM,&
             & ' ILEVTY=',ILEVTY,' ILEVEL=',ILEVEL,' IDATRT=',IDATRT,'
IREPRM=',IREPRM
            GOTO 100
          ENDIF

        ! ** Open FA file.

          IF(LLFIRAR) THEN

        ! First grib article read.
        ! The FA file has not yet been opened.
        ! It is necessary to open FA file here
        ! in order to get date and time from grib file.

            LLFIRAR=.FALSE.

        ! Initialize date.

            IDATEF(1)=100*(ISEC1(21)-1)+ISEC1(10)
            IDATEF(2)=ISEC1(11)
            IDATEF(3)=ISEC1(12)
            IDATEF(4)=ISEC1(13)
            IDATEF(5)=ISEC1(14)
            IDATEF(6)=ISEC1(15)
            IDATEF(7)=ISEC1(16)
            IDATEF(8)=ISEC1(17)
            IDATEF(9)=ISEC1(18)
            IDATEF(10)=ISEC1(19)
            IDATEF(11)=ISEC1(20)

        ! Write out date.

            WRITE(NULOUT,*) 'Grib file date =',IDATEF

        ! Initialize FA parameters.

            INTIMES=1
            INBARP=NS3D*NFLEVG + NS2D

        ! Open file.

            CLFAFILN='CN90x'//CNMEXP(1:4)//'INIT' ! ARPEGE file name.
            WRITE(NULOUT,*) 'OPEN ARPEGE FILE ',CLFAFILN
            CALL FAITOU(IREP,NPOSSH,.TRUE.,CLFAFILN,'UNKNOWN',&
             & .TRUE.,.TRUE.,INTIMES,INBARP,INBARI,CNMCA)
```

```
      CALL LFIMST(IREP,NPOSSH,.FALSE.)
      CALL FANDAR(IREP,NPOSSH,IDATEF)

! ** Read climatologies and check month

      IF (LLCLIM) THEN
        WRITE(UNIT=NULOUT,FMT='(A)')&
         & ' GRIDPOINT CLIMATOLOGIC INPUT DATA TO BE READ FROM FILE  '
! Les controles de coherence entre fichier clim et ARP sont faits par OPENFA()
        CALL OPENFA(10,IUCLIM,IDEB2,IFIN,IINDEX,IDATEC)

! Initialize Constantes from YOMCLI
        CALL VAL923(.TRUE.)
      ENDIF

    ELSE

! This grib article is not the first read.
! We check now if its date is compatible
! with previous ones.

      IF((100*(ISEC1(21)-1)+ISEC1(10)) /= IDATEF(1)&
         & .OR.ISEC1(11) /= IDATEF(2)&
         & .OR.ISEC1(12) /= IDATEF(3)&
         & .OR.ISEC1(13) /= IDATEF(4)&
         & .OR.ISEC1(14) /= IDATEF(5)) THEN
        WRITE(NULOUT,*) 'CPREP1: WARNING DATE INCONSISTENCY!'
        WRITE(NULOUT,*) 'isec1=',ISEC1
        WRITE(NULOUT,*) 'idatef=',IDATEF
      ENDIF
    ENDIF

! Grib coding conventions values.

    IDATRTG=4 ! data representation type: grid-point fields.
    IDATRTS=50 ! data representation type: spectral fields.

    IF(IDATRT == IDATRTS) THEN

! The field is spectral type.

      WRITE(NULOUT,*) 'Spectral field read.'
      LLINPFS=.TRUE. ! .true. if the input field is spectral.

! Initialize to zero.

      DO JINDEX=1,NSPEC2
        ZFASP(JINDEX)=0.0_JPRB
      ENDDO

! Initialize from grib file.
```

```
          WRITE(NULOUT,*) 'ireprm=',IREPRM
          INDEX=0
          DO JM=0,NSMAX
            DO JN=JM,NSMAX
              INDEX=INDEX+1
              ZFASP(INDEX)=ZGRBDAT(INDEX)
              INDEX=INDEX+1
              IF(JM /= 0) THEN
                ZFASP(INDEX)=ZGRBDAT(INDEX)
              ELSE
                ZFASP(INDEX)=0.0_JPRB
              ENDIF
            ENDDO
          ENDDO

        ELSEIF(IDATRT == IDATRTG) THEN

      ! The field is grid-point type.

          WRITE(NULOUT,*) 'Grid-point field read.'
          LLINPFS=.FALSE.

      ! Initialize from grib file.

          ZFAGG(1:NGPTOTG)=ZGRBDAT(1:NGPTOTG)

          IF(NGPTOTG /= IDONR) THEN
            WRITE(NULOUT,*) 'WARNING: skip Grid-point parameter with incorrect resolution
IPARAM=',IPARAM
            GOTO 100
          ENDIF

        ELSE

          WRITE(NULOUT,*)'CPREP1 parameter',IPARAM,': idatrt wrong 1!...'
          WRITE(NULOUT,*) 'ISEC2=',ISEC2
          CALL ABOR1('CPREP1: ABOR1 CALLED')

        ENDIF

      ! Check consistency between grib file grid and ARPEGE grid.

        IF(ILEVTY /= 1.AND.ILEVTY /= 109.AND.ILEVTY /= 112) THEN
          WRITE(NULOUT,*) 'LEVEL TYPE ' ,ILEVTY,'!'
          CALL ABOR1('ERROR CPREP1 LEVEL!')
        ENDIF
        IF(.NOT.LLINPFS.AND.ISEC2(10) /= NDGNH) THEN
          WRITE(NULOUT,*) 'RESOLUTION OF MODEL ',NDGNH &
           & ,' ,OF INITIAL DATA ',ISEC2(10),'!'
          CALL ABOR1('ERROR CPREP1 NDGNH')
        ENDIF
        IF(.NOT.LLINPFS.AND.NHTYP /= 0) THEN
```

```
      DO JGL=1,NDGLG
        IF(NLOENG(JGL) /= ISEC2(22+JGL)) THEN
          WRITE(NULOUT,*) 'INCONSISTENT REDUCED GRID'
          WRITE(NULOUT,*) 'IN MODEL ',(NLOENG(JB),JB=1,NDGLG)
          WRITE(NULOUT,*) 'IN FILE  ',(ISEC2(22+JB),JB=1,NDGLG),'!'
          CALL ABOR1('ERROR CPREP1 NLOENG!')
        ENDIF
      ENDDO
    ENDIF

! ** Determine what to do on current parameter.

! Actions to do:

    LLSCAL=.FALSE. ! .true. if the current parameter has to be scaled.
    LLOUTFS=.TRUE. ! .true. if the output field has to be spectral.
    LL3DMOD=.FALSE. ! .true. if the spectral field has to be scaled by rlapin.
    LLWRIF=.TRUE. ! .true. if the field has simply to be rewritten on the FA file.

    IF(IPARAM == 138) THEN

! Vorticity.

      CLPREF='S'
      INIVEAU=ILEVEL
      CLSUFF='FONC.COURANT'
      LL3DMOD=.TRUE.
    ELSEIF(IPARAM == 155) THEN

! Divergence.

      CLPREF='S'
      INIVEAU=ILEVEL
      CLSUFF='POT.VITESSE '
      LL3DMOD=.TRUE.
    ELSEIF(IPARAM == 130) THEN

! Temperature.

      CLPREF='S'
      INIVEAU=ILEVEL
      CLSUFF='TEMPERATURE '
    ELSEIF(IPARAM == 133) THEN

! Specific humidity.
      IF (LLHUPDG) LLOUTFS=.FALSE.
      CLPREF='S'
      INIVEAU=ILEVEL
      CLSUFF='HUMI.SPECIFI'
    ELSEIF(IPARAM == 139) THEN

! surface temperature.
```

```
      LLSURFT=.TRUE.
      IF (LLGLACE) ZFSURFT(1:NGPTOTG)=ZFAGG(1:NGPTOTG)
      LLOUTFS=.FALSE.
      CLPREF='SURF'
      INIVEAU=0
      CLSUFF='TEMPERATURE '
    ELSEIF(IPARAM == 183) THEN

! CDST.

      LLOUTFS=.FALSE.
      CLPREF='SURF'
      INIVEAU=0
      CLSUFF='CDST       '
! Store DST to compute some fields at the end
      LLCDST=.TRUE.
      ZFACDST(1:NGPTOTG)=ZFAGG(1:NGPTOTG)

    ELSEIF(IPARAM == 236) THEN

! STL4.

      LLOUTFS=.FALSE.
      CLPREF='SURF'
      INIVEAU=0
      CLSUFF='STL4       '
! Store STL4 to compute some fields at the end
      LLSTL4=.TRUE.
      ZFASTL4(1:NGPTOTG)=ZFAGG(1:NGPTOTG)

    ELSEIF(IPARAM == 172) THEN

! Land/Sea Mask.
      LLOUTFS=.FALSE.
      CLPREF='SURF'
      INIVEAU=0
      CLSUFF='IND.TERREMER'
! Store LSM to compute some fields at the end
      LLLSM=.TRUE.
      ZFALSM(1:NGPTOTG)=ZFAGG(1:NGPTOTG)

    ELSEIF(IPARAM == 174) THEN

! surface albedo.
      IF (LLCLIM) THEN
        LLWRIF=.FALSE.
      ENDIF
      LLOUTFS=.FALSE.
      CLPREF='SURF'
      INIVEAU=0
      CLSUFF='ALBEDO     '
```

```
   ELSEIF(IPARAM == 32) THEN

! snow albedo.
     LLOUTFS=.FALSE.
     CLPREF='SURF'
     INIVEAU=0
     CLSUFF='ALBEDO NEIGE'
   ELSEIF(IPARAM == 33) THEN

! snow density.
     LLSCAL=.TRUE.
     ZSCALE=0.001_JPRB
     LLOUTFS=.FALSE.
     CLPREF='SURF'
     INIVEAU=0
     CLSUFF='DENSIT.NEIGE'
   ELSEIF(IPARAM == 199) THEN

! surface prop.vegetat.
     IF (LLCLIM) THEN
       LLWRIF=.FALSE.
     ENDIF
     LLOUTFS=.FALSE.
     CLPREF='SURF'
     INIVEAU=0
     CLSUFF='PROP.VEGETAT'
   ELSEIF(IPARAM == 173) THEN

! surface Z0.FOIS.G.
     IF (LLCLIM) THEN
       LLWRIF=.FALSE.
     ENDIF
     LLOUTFS=.FALSE.
     CLPREF='SURF'
     INIVEAU=0
     CLSUFF='Z0.FOIS.G'

   ELSEIF(IPARAM == 152.OR.IPARAM == 134) THEN

! Log(surface pressure) OR surface pressure.

     CLPREF='SURF'
     INIVEAU=0
     CLSUFF='PRESSION    '
   ELSEIF(IPARAM == 129) THEN

! Orography.

     CLPREF='SPEC'
     INIVEAU=0
     CLSUFF='SURFGEOPOTENTI'
   ELSEIF(IPARAM == 190) THEN
```

```
! Orography: EW component of sub-grid scale orographic variance.

      LLOUTFS=.FALSE.
      LLWRIF=.FALSE.
      LLGRB190=.TRUE.
      CLPREF=' '
      INIVEAU=0
      CLSUFF=' '
    ELSEIF(IPARAM == 191) THEN

! Orography: NS component of sub-grid scale orographic variance.

      LLOUTFS=.FALSE.
      LLWRIF=.FALSE.
      LLGRB191=.TRUE.
      CLPREF=' '
      INIVEAU=0
      CLSUFF=' '
    ELSEIF(IPARAM == 192) THEN

! Orography: NWSE component of sub-grid scale orographic variance.

      LLOUTFS=.FALSE.
      LLWRIF=.FALSE.
      LLGRB192=.TRUE.
      CLPREF=' '
      INIVEAU=0
      CLSUFF=' '
    ELSEIF(IPARAM == 193) THEN

! Orography: NESW component of sub-grid scale orographic variance.

      LLOUTFS=.FALSE.
      LLWRIF=.FALSE.
      LLGRB193=.TRUE.
      CLPREF=' '
      INIVEAU=0
      CLSUFF=' '
    ELSEIF(IPARAM == 160) THEN

! Orography: sigma.

      LLOUTFS=.FALSE.
      CLPREF='SURF'
      INIVEAU=0
      CLSUFF='ET.GEOPOTENT'
    ELSEIF(IPARAM == 161) THEN

! Orography: anisotropy.

      LLOUTFS=.FALSE.
```

```
      CLPREF='SURF'
      INIVEAU=0
      CLSUFF='VAR.GEOP.ANI'
    ELSEIF(IPARAM == 162) THEN

! Orography: angle of principal axis.

      LLOUTFS=.FALSE.
      CLPREF='SURF'
      INIVEAU=0
      CLSUFF='VAR.GEOP.DIR'
    ELSEIF(IPARAM == 200) THEN

! Orography: variance (old).

      DO JGPTOTG=1,NGPTOTG
        ZFAGG(JGPTOTG)=RG*SQRT(ZFAGG(JGPTOTG))
      ENDDO
      LLOUTFS=.FALSE.
      CLPREF='SURF'
      INIVEAU=0
      CLSUFF='ET.GEOPOTENT'
    ELSEIF (IPARAM==ISWL1) THEN
      LLWRIF=.FALSE.
    ELSEIF (IPARAM==ISWL2) THEN
      LLWRIF=.FALSE.
    ELSEIF (IPARAM==ISWL3) THEN
      LLWRIF=.FALSE.
    ELSEIF (IPARAM==ISWL4) THEN
      LLWRIF=.FALSE.
    ELSEIF (IPARAM==43) THEN
      LLWRIF=.FALSE.
    ELSEIF (IPARAM==NGRBCVL) THEN
      LLWRIF=.FALSE.
    ELSEIF (IPARAM==NGRBCVH) THEN
      LLWRIF=.FALSE.
    ELSEIF (IPARAM==NGRBTVL) THEN
      LLWRIF=.FALSE.
    ELSEIF (IPARAM==NGRBTVH) THEN
      LLWRIF=.FALSE.

    ELSE

! Look for other surface fields.

      IOK=0

! soilb.

      DO JC=1,YSP_SBD%NLEVS
        DO JV=1,YSP_SBD%NUMFLDS
```

```
                IF(YSP_SB%YSB(JV)%CNAME(JC) /= '
'.AND.YSP_SB%YSB(JV)%IGRBCODE(JC) == IPARAM) THEN
                  IOK=1
                  LLOUTFS=.FALSE.
                  CLPREF=YSP_SB%YSB(JV)%CNAME(JC)(1:4)
                  INIVEAU=0
                  CLSUFF=YSP_SB%YSB(JV)%CNAME(JC)(5:)
                ENDIF
              ENDDO
            ENDDO

        ! snowg.

            DO JV=1,YSP_SGD%NUMFLDS
              IF(YSP_SG%YSG(JV)%CNAME /= ' '.AND.YSP_SG%YSG(JV)%IGRBCODE ==
IPARAM) THEN
                IOK=1
                LLOUTFS=.FALSE.
                CLPREF=YSP_SG%YSG(JV)%CNAME(1:4)
                INIVEAU=0
                CLSUFF=YSP_SG%YSG(JV)%CNAME(5:)
              ENDIF
            ENDDO

        ! resvr.

            DO JV=1,YSP_RRD%NUMFLDS
              IF(YSP_RR%YRR(JV)%CNAME /= ' '.AND.YSP_RR%YRR(JV)%IGRBCODE ==
IPARAM) THEN
                IOK=1
                LLOUTFS=.FALSE.
                CLPREF=YSP_RR%YRR(JV)%CNAME(1:4)
                INIVEAU=0
                CLSUFF=YSP_RR%YRR(JV)%CNAME(5:)
              ENDIF
            ENDDO


        ! extrp.

            DO JC=1,YSP_EPD%NLEVS
              DO JV=1,YSP_EPD%NUMFLDS
                IF(YSP_EP%YEP(JV)%CNAME(JC) /= ' '.AND.
YSP_EP%YEP(JV)%IGRBCODE(JC) == IPARAM) THEN
                  IOK=1
                  LLOUTFS=.FALSE.
                  CLPREF=YSP_EP%YEP(JV)%CNAME(JC)(1:4)
                  INIVEAU=0
                  CLSUFF=YSP_EP%YEP(JV)%CNAME(JC)(5:)
                ENDIF
              ENDDO
            ENDDO
```

```
! xtrp2.

      DO JV=1,YSP_X2D%NUMFLDS
        IF(YSP_X2%YX2(JV)%CNAME /= ' ' .AND. YSP_X2%YX2(JV)%IGRBCODE ==
IPARAM) THEN
          IOK=1
          LLOUTFS=.FALSE.
          CLPREF=YSP_X2%YX2(JV)%CNAME(1:4)
          INIVEAU=0
          CLSUFF=YSP_X2%YX2(JV)%CNAME(5:)
        ENDIF
      ENDDO

! varsf.

      DO JV=1,YSD_VFD%NUMFLDS
        IF(YSD_VF%YVF(JV)%CNAME /= ' '.AND. YSD_VF%YVF(JV)%IGRBCODE ==
IPARAM) THEN
          IOK=1
          LLOUTFS=.FALSE.
          CLPREF=YSD_VF%YVF(JV)%CNAME(1:4)
          INIVEAU=0
          CLSUFF=YSD_VF%YVF(JV)%CNAME(5:)
        ENDIF
      ENDDO

! vclip.

      DO JV=1,YSD_VPD%NUMFLDS
        IF(YSD_VP%YVP(JV)%CNAME /= ' '.AND. YSD_VP%YVP(JV)%IGRBCODE ==
IPARAM) THEN
          IOK=1
          LLOUTFS=.FALSE.
          CLPREF=YSD_VP%YVP(JV)%CNAME(1:4)
          INIVEAU=0
          CLSUFF=YSD_VP%YVP(JV)%CNAME(5:)
        ENDIF
      ENDDO

! waves.

      DO JV=1,YSD_WSD%NUMFLDS
        IF(YSD_WS%YWS(JV)%CNAME /= ' '.AND. YSD_WS%YWS(JV)%IGRBCODE
== IPARAM) THEN
          IOK=1
          LLOUTFS=.FALSE.
          CLPREF=YSD_WS%YWS(JV)%CNAME(1:4)
          INIVEAU=0
          CLSUFF=YSD_WS%YWS(JV)%CNAME(5:)
        ENDIF
      ENDDO
```

```
      IF(IOK == 0) THEN

! Unattended parameter.

        WRITE(NULOUT,*)'Parameter ',IPARAM,' unattended!...'
        GOTO 100
      ENDIF
    ENDIF
    IF(LL3DMOD.AND.LLINPFS) THEN

! Modify 3D spectral fields spectrum.

      WRITE(NULOUT,*) 'Modify 3D spectral fields spectrum.'
      DO JNM=1,NSPEC2
        ZFASP(JNM)=ZFASP(JNM)*RLAPIN(NVALUE(JNM))
      ENDDO
    ENDIF

! ** Spectral direct/inverse transform.

    IF(LLINPFS.AND..NOT.LLOUTFS) THEN

! The input field is spectral and the output one
! has to be grid-point.

      WRITE(NULOUT,*) 'Call speree.'
      CALL SPEREE(1,1,ZFASP,ZFAGG)
    ELSEIF(.NOT.LLINPFS.AND.LLOUTFS) THEN

! The input field is grid-point and the output one
! has to be spectral.
      WRITE(NULOUT,*) 'Call reespe.'
      CALL REESPE(1,1,ZFASP,ZFAGG)
    ENDIF

! ** Odd scalings.

    IF(IPARAM == NGRBSD  ) THEN
      LLSCAL=.TRUE.
      ZSCALE=1000.0_JPRB
    ELSEIF(IPARAM == NGRBSR  ) THEN
      LLSCAL=.TRUE.
      ZSCALE=RG
    ELSEIF(IPARAM == NGRBSRC ) THEN
      LLSCAL=.TRUE.
      ZSCALE=1000.0_JPRB
    ELSEIF(IPARAM == NGRBSDOR) THEN
      LLSCAL=.TRUE.
      ZSCALE=RG
    ENDIF
    IF(LLSCAL.AND..NOT.LLOUTFS) THEN
```

```
! A scaling is required and the output field
! is grid-point type.

    WRITE(NULOUT,*) 'Scaling: ',ZSCALE
    ZFAGG(1:NGPTOTG)=ZFAGG(1:NGPTOTG)*ZSCALE
  ELSEIF(LLSCAL) THEN

! A scaling is required and the output field
! is spectral type.

    CALL ABOR1('CPREP1 scaling on spectral!...')
  ENDIF

  IF(LLWRIF) THEN

! ** Write field on FA file.

    IF(LLOUTFS) THEN

! The output field has to be spectral.
! One computes and prints
! its minima/maxima in grid-point space.

! One copies zfasp on zfasps because
! the input spectral data will be corrupted
! by speree.

      WRITE(NULOUT,*) 'Write spectral field'
      ZFASPS(1:NSPEC2)=ZFASP(1:NSPEC2)
      CALL SPEREE(1,1,ZFASPS,ZFAGG)
      CLLIBELLE='Extrema in grid-point space:'
      CALL CPREP1_DIAG(ZFAGG,CLLIBELLE)

! Write spectral field.

      CALL
FAVEUR(IREP,NPOSSH,INGRIB,INBPDG,INBCSP,ISTRON,IPUILA,IDMOPL)
!     data should be reordered only if packing type is not -1 and not 3
!     Notice : If INGRIB=-1 or 3 the field will be a bit larger than if
!     INGRIB=0, 1 or 2 because then there is no reordering : so the
!     column 0 is doubled.
      IF (INGRIB==-1 .OR. INGRIB==3) THEN
        INGRIG=-1
      ELSE
        INGRIG=0
      ENDIF
      IF (INGRIG==0) THEN
        LLFILE_TO_MODEL=.FALSE.
        CALL SPREORD(1,ZFPDAT,ZFASP,LLFILE_TO_MODEL)
      ENDIF
      IF (TRIM(CLPREF)//CLSUFF == 'SPECSURFGEOPOTENTI') THEN
```

```
!        Do not pack surface geopotential
          CALL
FAGOTE(IREP,NPOSSH,INGRIG,INBPDG,INBCSP,ISTRON,IPUILA,IDMOPL)
        ENDIF
        IF (INGRIG==0) THEN
          CALL FAIENC(IREP,NPOSSH,CLPREF,INIVEAU,CLSUFF,ZFPDAT,LLOUTFS)
        ELSE
          CALL FAIENC(IREP,NPOSSH,CLPREF,INIVEAU,CLSUFF,ZFASP,LLOUTFS)
        ENDIF
        IF (TRIM(CLPREF)//CLSUFF == 'SPECSURFGEOPOTENTI') THEN
!        Reset packing after the treatment of surface geopotential
          CALL
FAGOTE(IREP,NPOSSH,INGRIB,INBPDG,INBCSP,ISTRON,IPUILA,IDMOPL)
        ENDIF
        WRITE(NULOUT,*) 'Write spectral field param=',&
         & IPARAM,' ',CLPREF(1:4),INIVEAU,CLSUFF(1:16)
      ELSE

! Write grid-point field.

        WRITE(NULOUT,*) 'Write grid-point field param=',&
         & IPARAM,' ',CLPREF(1:4),INIVEAU,CLSUFF(1:16)
        IF (TRIM(CLPREF)//CLSUFF == 'SURFZ0.FOIS.G' .OR. &
           & TRIM(CLPREF)//CLSUFF == 'SURFZ0REL.FOIS.G' .OR. &
           & TRIM(CLPREF)//CLSUFF == 'SURFGZ0.THERM') THEN
!        Do not pack roughness lengths
          CALL
FAVEUR(IREP,NPOSSH,INGRIB,INBPDG,INBCSP,ISTRON,IPUILA,IDMOPL)
          INGRIG=0
          CALL
FAGOTE(IREP,NPOSSH,INGRIG,INBPDG,INBCSP,ISTRON,IPUILA,IDMOPL)
        ENDIF
        CALL FAIENC(IREP,NPOSSH,CLPREF,INIVEAU,CLSUFF,ZFAGG,LLOUTFS)
        IF (TRIM(CLPREF)//CLSUFF == 'SURFZ0.FOIS.G' .OR. &
           & TRIM(CLPREF)//CLSUFF == 'SURFZ0REL.FOIS.G' .OR. &
           & TRIM(CLPREF)//CLSUFF == 'SURFGZ0.THERM') THEN
!        Reset packing after the treatment of roughness lengths
          CALL
FAGOTE(IREP,NPOSSH,INGRIB,INBPDG,INBCSP,ISTRON,IPUILA,IDMOPL)
        ENDIF
        IF (IPARAM == 173.AND..NOT.LLCLIM) THEN
          ZFZ0G(1:NGPTOTG)=ZFAGG(1:NGPTOTG)
          LLZ0=.TRUE.
        ENDIF
      ENDIF

!The field has not to be written on FA file.
    ELSEIF (IPARAM==ISWL1) THEN
      ZFSWL1(1:NGPTOTG)=ZFAGG(1:NGPTOTG)
      LLSWL1=.TRUE.
    ELSEIF (IPARAM==ISWL2) THEN
      ZFSWL2(1:NGPTOTG)=ZFAGG(1:NGPTOTG)
```

```
      LLSWL2=.TRUE.
    ELSEIF (IPARAM==ISWL3) THEN
      ZFSWL3(1:NGPTOTG)=ZFAGG(1:NGPTOTG)
      LLSWL3=.TRUE.
    ELSEIF (IPARAM==ISWL4) THEN
      ZFSWL4(1:NGPTOTG)=ZFAGG(1:NGPTOTG)
      LLSWL4=.TRUE.
    ELSEIF (IPARAM==43) THEN
      ZFSLT(1:NGPTOTG)=ZFAGG(1:NGPTOTG)
      LLSLT=.TRUE.
    ELSEIF (IPARAM==NGRBCVL) THEN
      ZFCVL(1:NGPTOTG)=ZFAGG(1:NGPTOTG)
      LLCVL=.TRUE.
    ELSEIF (IPARAM==NGRBCVH) THEN
      ZFCVH(1:NGPTOTG)=ZFAGG(1:NGPTOTG)
      LLCVH=.TRUE.
    ELSEIF (IPARAM==NGRBTVL) THEN
      ZFTVL(1:NGPTOTG)=ZFAGG(1:NGPTOTG)
      LLTVL=.TRUE.
    ELSEIF (IPARAM==NGRBTVH) THEN
      ZFTVH(1:NGPTOTG)=ZFAGG(1:NGPTOTG)
      LLTVH=.TRUE.
    ELSEIF (IPARAM /= 173.AND.IPARAM /= 174.AND.IPARAM /= 199) THEN
! In this case,  surfz0.fois.g, surfalbedo and surfprop.vegetat are read
! from climatological file and not stored.

! The field has not to be written on FA file.
! It has to be stored in an intermediate array.

      IGRBSN=IGRBSN+1
      IGRBS(IGRBSN)=IPARAM
      INDEX=0
!      CLLIBELLE='Field '// IGRBSN //' stored in an intermediate array.'
      WRITE(CLLIBELLE,FMT=*) 'Field ',IGRBSN,' stored in an intermediate array.'
      CALL CPREP1_DIAG(ZFAGG,CLLIBELLE)
      ZFAGGA(1:NGPTOTG,IGRBSN)=ZFAGG(1:NGPTOTG)
    ENDIF

! Go to next article in the GRIB file.

    GOTO 100
    200 CONTINUE

! Close grib file.

    CALL PBCLOSE(NINISH,IRET)
    IF(IRET /= 0) THEN
      WRITE(NULOUT,'(A,I2)')'CPREP1: PROBLEM IN PBCLOSE, IRET=',IRET
      CALL ABOR1('CPREP1: PROBLEM IN PBCLOSE')
    ENDIF
  ENDIF
ENDDO   ! end of work on each GRIB file
```

```
      IF (.NOT.LLLSM) THEN
        IF (.NOT. LLCONTROL) THEN
          WRITE(NULOUT,*) 'WARNING: NO LAND/SEA MASK IN GRIB FILE!!!'
          WRITE(NULOUT,*) 'ALL POINTS ARE SEA POINTS!!!'
          ZFALSM(1:NGPTOTG)=0._JPRB
        ELSE
          CALL ABOR1('CPREP1: NO LAND/SEA MASK IN GRIB FILE')
        ENDIF
      ENDIF

      IF (LLCLIM) THEN ! ** Working simply whith climatological fields
        WRITE(NULOUT,*) 'COPY CLIMATOLOGICAL FIELDS FROM FILE'
        INIVEAU=0
        DO IIND=1,INBCLIB
          CLPREF=CLLIB(IIND)(1:4)
          CLSUFF=CLLIB(IIND)(5:20)
          CALL FACOCH(IREP,IUCLIM,NPOSSH,CLPREF,INIVEAU,CLSUFF)
          WRITE(NULOUT,*) 'Write grid-point field '&
           & ,CLPREF(1:4),INIVEAU,CLSUFF(1:16)
        ENDDO
      ELSE
        LLOUTFS=.FALSE.
        INIVEAU=0
        DO IIND=5,11
          CLPREF=YL_FIELDCST(IIND)%CLHAMP(1:4)
          CLSUFF=YL_FIELDCST(IIND)%CLHAMP(5:20)
          IF (YL_FIELDCST(IIND)%LLISLSM) THEN ! link whith Land/Sea Mask
            DO JGPTOTG=1,NGPTOTG
              IF (ZFALSM(JGPTOTG) > 0.5) THEN
                ZFAGG(JGPTOTG)=YL_FIELDCST(IIND)%ZVALTERRE
              ELSE
                ZFAGG(JGPTOTG)=YL_FIELDCST(IIND)%ZVALMER
              ENDIF
            ENDDO

ZMIN=MIN(YL_FIELDCST(IIND)%ZVALTERRE,YL_FIELDCST(IIND)%ZVALMER)

ZMAX=MAX(YL_FIELDCST(IIND)%ZVALTERRE,YL_FIELDCST(IIND)%ZVALMER)
          ELSE
            ZVAL=YL_FIELDCST(IIND)%ZVALTERRE
            ZFAGG(1:NGPTOTG)=ZVAL
            ZMIN=ZVAL
            ZMAX=ZVAL
          ENDIF
          WRITE(NULOUT,*)'---------------------------------------------'
          WRITE(NULOUT,*) YL_FIELDCST(IIND)%CLHAMP ,' field'
          WRITE(NULOUT,*) 'Min=',ZMIN,' Max=',ZMAX
          CALL FAIENC(IREP,NPOSSH,CLPREF,INIVEAU,CLSUFF,ZFAGG,LLOUTFS)
          WRITE(NULOUT,*) 'Write grid-point field '&
           & ,CLPREF(1:4),INIVEAU,CLSUFF(1:16)
```

```
        !store ARGILE, SABLE, SURFEPAIS.SOL for ACSOLW (need to compute
RESERV.EAU)
          IF (YL_FIELDCST(IIND)%CLHAMP=='SURFPROP.ARGILE     ') THEN
            ZFPARG(1:NGPTOTG)=ZFAGG(1:NGPTOTG)
          ELSEIF (YL_FIELDCST(IIND)%CLHAMP=='SURFPROP.SABLE      ') THEN
            ZFPSAB(1:NGPTOTG)=ZFAGG(1:NGPTOTG)
          ELSEIF (YL_FIELDCST(IIND)%CLHAMP=='SURFEPAIS.SOL       ') THEN
            ZFPD2(1:NGPTOTG)=ZFAGG(1:NGPTOTG)
          ELSEIF (YL_FIELDCST(IIND)%CLHAMP=='SURFRESI.STO.MIN    ') THEN
            ZFPRSMIN(1:NGPTOTG)=ZFAGG(1:NGPTOTG)
          ELSEIF (YL_FIELDCST(IIND)%CLHAMP=='SURFIND.FOLIAIRE    ') THEN
            ZFPLAI(1:NGPTOTG)=ZFAGG(1:NGPTOTG)
          ENDIF

        ENDDO
      ENDIF  ! OF (LLCLIM)

      ! Les champs SURFRESERV.GLACE,PROFRESERV.GLACE,SURFRES.EVAPOTRA et
      !SURFRESERV.INTER n'existent pas en CLIM et sont codes independamment de
LLCLIM.

      LLOUTFS=.FALSE.
      INIVEAU=0
      IF (LLGLACE) INBCGLA=3
      DO IIND=INBCGLA,4
        CLPREF=YL_FIELDCST(IIND)%CLHAMP(1:4)
        CLSUFF=YL_FIELDCST(IIND)%CLHAMP(5:20)
        ZVAL=YL_FIELDCST(IIND)%ZVALTERRE
        ZFAGG(1:NGPTOTG)=ZVAL
        ZMIN=ZVAL
        ZMAX=ZVAL
        WRITE(NULOUT,*)'---------------------------------------------'
        WRITE(NULOUT,*) YL_FIELDCST(IIND)%CLHAMP ,' field'
        WRITE(NULOUT,*) 'Min=',ZMIN,' Max=',ZMAX
        CALL FAIENC(IREP,NPOSSH,CLPREF,INIVEAU,CLSUFF,ZFAGG,LLOUTFS)
        WRITE(NULOUT,*) 'Write grid-point field '&
         & ,CLPREF(1:4),INIVEAU,CLSUFF(1:16)
      ENDDO

         IF (LLCVL.AND.LLCVH.AND.LLTVL.AND.LLTVH) THEN

          CALL SURF_INQ(PRVLAI=ZRVLAI)
          DO JGPTOTG=1,NGPTOTG
            ZFLAI(JGPTOTG)=(ZFCVH(JGPTOTG)*ZRVLAI(NINT(ZFTVH(JGPTOTG)))) + &
             & (ZFCVL(JGPTOTG)*ZRVLAI(NINT(ZFTVL(JGPTOTG))))
          ENDDO
      !    CALL SURF_INQ(PRVRSMIN=ZRVRSMIN)

          ZRVRSMIN(1)=180._JPRB    ! Crops, Mixed Farming
          ZRVRSMIN(2)=110._JPRB    ! Short Grass
          ZRVRSMIN(3)=500._JPRB    ! Evergreen Needleleaf Trees
          ZRVRSMIN(4)=500._JPRB    ! Deciduous Needleleaf Trees
```

```
          ZRVRSMIN(5)=175._JPRB    ! Deciduous Broadleaf Trees
          ZRVRSMIN(6)=240._JPRB    ! Evergreen Broadleaf Trees
          ZRVRSMIN(7)=100._JPRB    ! Tall Grass
          ZRVRSMIN(8)=250._JPRB    ! Desert
          ZRVRSMIN(9)=80._JPRB     ! Tundra
          ZRVRSMIN(10)=180._JPRB   ! Irrigated Crops
          ZRVRSMIN(11)=150._JPRB   ! Semidesert
          ZRVRSMIN(12)=0.0_JPRB    ! Ice Caps and Glaciers
          ZRVRSMIN(13)=240._JPRB   ! Bogs and Marshes
          ZRVRSMIN(14)=0.0_JPRB    ! Inland Water
          ZRVRSMIN(15)=0.0_JPRB    ! Ocean
          ZRVRSMIN(16)=225._JPRB   ! Evergreen Shrubs
          ZRVRSMIN(17)=225._JPRB   ! Deciduous Shrubs
          ZRVRSMIN(18)=250._JPRB   ! Mixed Forest/woodland
          ZRVRSMIN(19)=175._JPRB   ! Interrupted Forest
          ZRVRSMIN(20)=150._JPRB   ! Water and Land Mixtures
          ZRVRSMIN(0)=ZRVRSMIN(8)

          DO JGPTOTG=1,NGPTOTG

ZFRSMIN(JGPTOTG)=(ZFCVH(JGPTOTG)*ZRVRSMIN(NINT(ZFTVH(JGPTOTG)))) + &
            & (ZFCVL(JGPTOTG)*ZRVRSMIN(NINT(ZFTVL(JGPTOTG))))
          ENDDO

         ENDIF

        ! ** Case of SURFGZ0.THERM and SURFPROP.VEGETAT
        IF (.NOT.LLCLIM) THEN
         LLOUTFS=.FALSE.
         INIVEAU=0
         CLPREF='SURF'
         IF (LLZ0) THEN
          CLSUFF='GZ0.THERM    '
          DO JGPTOTG=1,NGPTOTG
           IF (ZFALSM(JGPTOTG) > 0.5) THEN
            ZFAGG(JGPTOTG)=ZFZ0G(JGPTOTG)/10.0_JPRB
           ELSE
            ZFAGG(JGPTOTG)=ZFZ0G(JGPTOTG)
           ENDIF
          ENDDO
          WRITE(NULOUT,*)'---------------------------------------------'
          CLLIBELLE='SURFGZ0.THERM field'
          CALL CPREP1_DIAG(ZFAGG,CLLIBELLE)
        !   Do not pack roughness lengths
          CALL
FAVEUR(IREP,NPOSSH,INGRIB,INBPDG,INBCSP,ISTRON,IPUILA,IDMOPL)
          INGRIG=0
          CALL
FAGOTE(IREP,NPOSSH,INGRIG,INBPDG,INBCSP,ISTRON,IPUILA,IDMOPL)
          CALL FAIENC(IREP,NPOSSH,CLPREF,INIVEAU,CLSUFF,ZFAGG,LLOUTFS)
        !   Reset packing after the treatment of roughness lengths
```

```
            CALL
FAGOTE(IREP,NPOSSH,INGRIB,INBPDG,INBCSP,ISTRON,IPUILA,IDMOPL)
            WRITE(NULOUT,*) 'Write grid-point field '&
             & ,CLPREF(1:4),INIVEAU,CLSUFF(1:16)
          ELSE
            WRITE(NULOUT,*)&
             & 'SURFGZ0.THERM CANNOT BE COMPUTED BECAUSE SURFZ0.FOIS.G IS
NOT'
            IF (LLCONTROL) CALL ABOR1('CPREP1 : ABOR1 CALLED')
          ENDIF

          IF (LLCVL.AND.LLCVH.AND.LLTVL.AND.LLTVH) THEN
            CALL SURF_INQ(PRVCOV=ZRVCOV)
            CLSUFF='PROP.VEGETAT '
            DO JGPTOTG=1,NGPTOTG
              ZFAGG(JGPTOTG)=(ZFCVH(JGPTOTG)*ZRVCOV(NINT(ZFTVH(JGPTOTG)))) +
&
               & (ZFCVL(JGPTOTG)*ZRVCOV(NINT(ZFTVL(JGPTOTG))))
            ENDDO

            WRITE(NULOUT,*)'---------------------------------------------'
            CLLIBELLE='SURFPROP.VEGETAT field'
            CALL CPREP1_DIAG(ZFAGG,CLLIBELLE)
            CALL FAIENC(IREP,NPOSSH,CLPREF,INIVEAU,CLSUFF,ZFAGG,LLOUTFS)
            WRITE(NULOUT,*) 'Write grid-point field '&
             & ,CLPREF(1:4),INIVEAU,CLSUFF(1:16)
          ELSEIF (.NOT.LLOLDSWL) THEN
            WRITE(NULOUT,*) 'CPREP1: WARNING! ',&
             & 'SURFPROP.VEGETAT NOT COMPUTED WHITH NEW METHOD (MISSING
PARAMETERS)'
          ENDIF

        ENDIF

        ! ** Set ProftempeRATURE
        IF (LLCDST.AND.LLSTL4) THEN ! PROFTEMPERATURE can be compute
          ZFAGG=(ZFACDST + ZFASTL4) * 0.5
          IF (LLGLACE) ZFPROFT(1:NGPTOTG)=ZFAGG(1:NGPTOTG)
          LLOUTFS=.FALSE.
          CLPREF='PROF'
          INIVEAU=0
          CLSUFF='TEMPERATURE '
          CALL FAIENC(IREP,NPOSSH,CLPREF,INIVEAU,CLSUFF,ZFAGG,LLOUTFS)
          WRITE(NULOUT,*)'---------------------------------------------'
          CLLIBELLE='PROFTEMPERATURE field'
          CALL CPREP1_DIAG(ZFAGG,CLLIBELLE)

          WRITE(NULOUT,*) 'Write grid-point field '&
           & ,CLPREF(1:4),INIVEAU,CLSUFF(1:16)
        ELSE
          WRITE(NULOUT,*)&
```

```
          & 'PROFTEMPERATURE CANNOT BE COMPUTED BECAUSE CDST OR STL4 IS
NOT'
          IF (LLCONTROL) CALL ABOR1('CPREP1 : ABOR1 CALLED')
        ENDIF

        ! Set ZFPIVEG
        DO JGPTOTG=1,NGPTOTG
          IF (ZFALSM(JGPTOTG) > 0.5) THEN
            ZFPIVEG(JGPTOTG)=JPNTVTER
          ELSE
            ZFPIVEG(JGPTOTG)=JPNTVMER
          ENDIF
        ENDDO

        ! ** Set SURFRESERV.EAU and PROFRESERV.EAU
        IF (.NOT. LLOLDSWL) THEN
          ZPSOL=1.0_JPRB
        ELSE
          ZPSOL=PPPSWL1
        ENDIF

        !formule utilisee pour estimer les reservoirs de glace a partir des reservoirs d'eau:
        ! T0=TSRESERV1 ou TSRESERV2 : temperature de base proche de 273°
        ! delta=TDELTA1 ou TDELTA2  : intervalle autour de la temperature de base
        ! p= proportion de glace prise dans les reservoirs d'eau
        ! Tlue= temperature dans le sol profond ou superficiel
        ! p= 3 * (Teta*Teta) + 2 * (Teta*Teta*Teta)
        ! avec Teta= ((Tlue - T0)/(2 * delta)) - 0.5

        !Compute SURFRESERV.EAU
        IF (LLSWL1) THEN
          IF (LLCLIM) THEN
        ! Read ARGILE, SABLE, SURFEPAIS.SOL from climatological FILE if LLCLIM
            CALL FACILE(IREP,IUCLIM,'SURF',0,'PROP.ARGILE',ZFPARG,.FALSE.)
            CALL FACILE(IREP,IUCLIM,'SURF',0,'PROP.SABLE',ZFPSAB,.FALSE.)
            CALL FACILE(IREP,IUCLIM,'SURF',0,'EPAIS.SOL',ZFPD2,.FALSE.)
            CALL FACILE(IREP,IUCLIM,'SURF',0,'IND.FOLIAIRE',ZFPLAI,.FALSE.)
            CALL FACILE(IREP,IUCLIM,'SURF',0,'RESI.STO.MIN',ZFPRSMIN,.FALSE.)
          ENDIF

          CALL
ACSOLW(1,NGPTOTG,NGPTOTG,ZFPARG,ZFPD2,ZFALSM,ZFPIVEG,ZFPSAB,&
           & .FALSE.,ZFPWFC,ZFPWPMX,ZFPWSAT,ZFPWSMX,ZFPWWILT,'ACSOLW ')

          ZSCALE=1000.0_JPRB
          IF (.NOT.LLSLT) THEN
            WRITE(NULOUT,*) 'CPREP1 WARNING: SURFRESERV.EAU computed without
LST !!'
          ENDIF
          DO JGPTOTG=1,NGPTOTG
            IF (ZFALSM(JGPTOTG) > 0.5) THEN
              IF (LLSLT) THEN
```

```
            ZWSATSLT =ZWP_SAT(NINT(ZFSLT(JGPTOTG)))
            ZPWWILT_CEP=ZWP_WILT(NINT(ZFSLT(JGPTOTG)))
            ZPWCAP_CEP=ZWP_CAP(NINT(ZFSLT(JGPTOTG)))
          ELSE
            ZWSATSLT=PPWSAT_CEP
            ZPWWILT_CEP=PPWWILT_CEP
            ZPWCAP_CEP=PPWCAP_CEP
          ENDIF

          !CALCUL DU SWI CEP
          SWI_CEP= ((ZFSWL1(JGPTOTG)/ZPSOL)-ZPWWILT_CEP)/(ZPWCAP_CEP-
ZPWWILT_CEP)

          !CALCUL Wp ISBA
          IF (SWI_CEP > 1.0) THEN
           ZGOL = ZFPWFC(JGPTOTG)
          ELSEIF (SWI_CEP > 0.0) THEN
           ZGOL=ZFPWFC(JGPTOTG)*ACOS(1.0-2.0*SWI_CEP)/ACOS(-1.0)
          ELSE
           ZGOL = 0.0_JPRB
          ENDIF

          !Calcul Wp ISBA en m
          ZGOL=ZGOL*(ZSCALE*RD1)

          IF (LLGLACE) THEN
            IF (.NOT.LLSURFT) THEN
              WRITE(NULOUT,*)&
               & 'SURFRESERV.GLACE CANNOT BE COMPUTED BECAUSE
SURFTEMPERATURE IS NOT READ'
              IF (LLCONTROL) CALL ABOR1('CPREP1 : ABOR1 CALLED')
            ENDIF

            IF (ZFSURFT(JGPTOTG) > (TSRESERV1+TDELTA1)) THEN
              ZPGLACE=0.0
            ELSEIF (ZFSURFT(JGPTOTG) < (TSRESERV1-TDELTA1)) THEN
              ZPGLACE=1.0
            ELSE
              ZDELTA=((ZFSURFT(JGPTOTG)-TSRESERV1)/(2.0*TDELTA1)) - 0.5
              ZPGLACE=MIN(((3.0_JPRB*ZDELTA*ZDELTA)  &
               & + (2.0_JPRB*ZDELTA*ZDELTA*ZDELTA)),1.0_JPRB)
            ENDIF
            ZFGLACE(JGPTOTG)=ZGOL*ZPGLACE
            ZFAGG(JGPTOTG)=ZGOL-ZFGLACE(JGPTOTG)
          ELSE
            ZFAGG(JGPTOTG)=ZGOL
          ENDIF

        ELSE
          ZFAGG(JGPTOTG)=RD1*GCONV
          IF (LLGLACE) ZFGLACE(JGPTOTG)=0.0_JPRB
        ENDIF
```

```
          ENDDO

          CALL FAIENC(IREP,NPOSSH,'SURF',0,'RESERV.EAU  ',ZFAGG,.FALSE.)
          WRITE(NULOUT,*)'--------------------------------------------'
          CLLIBELLE='SURFRESERV.EAU field'
          CALL CPREP1_DIAG(ZFAGG,CLLIBELLE)
          WRITE(NULOUT,*) 'Write grid-point field '
          IF (LLGLACE.AND.LLSURFT) THEN
            CALL FAIENC(IREP,NPOSSH,'SURF',0,'RESERV.GLACE',ZFGLACE,.FALSE.)
            WRITE(NULOUT,*)'--------------------------------------------'
            CLLIBELLE='SURFRESERV.GLACE field'
            CALL CPREP1_DIAG(ZFGLACE,CLLIBELLE)
            WRITE(NULOUT,*) 'Write grid-point field '
          ENDIF
        ELSE
          WRITE(NULOUT,*)&
           & 'SURFRESERV.EAU NOT COMPUTED BECAUSE MISSING PARAMETERS IN
GRIB FILE'
          IF (LLCONTROL) CALL ABOR1('CPREP1 : ABOR1 CALLED')
        ENDIF

        !Compute PROFRESERV.EAU
        IF (LLSWL1.AND.LLSWL2.AND.LLSWL3.AND.LLSWL4) THEN
          IF (.NOT.LLSLT) THEN
            WRITE(NULOUT,*) 'CPREP1 WARNING: PROFRESERV.EAU computed without
LST !!'
          ENDIF
          DO JGPTOTG=1,NGPTOTG
            IF (ZFALSM(JGPTOTG) > 0.5) THEN
              IF (LLSLT) THEN
                ZWSATSLT =ZWP_SAT(NINT(ZFSLT(JGPTOTG)))
                ZPWWILT_CEP=ZWP_WILT(NINT(ZFSLT(JGPTOTG)))
                ZPWCAP_CEP=ZWP_CAP(NINT(ZFSLT(JGPTOTG)))
              ELSE
                ZWSATSLT=PPWSAT_CEP
                ZPWWILT_CEP=PPWWILT_CEP
                ZPWCAP_CEP=PPWCAP_CEP
              ENDIF

              ZFSWL1_aux=ZFSWL1(JGPTOTG)/ZPSOL
              ZFSWL2_aux=ZFSWL2(JGPTOTG)/ZPSOL
              ZFSWL3_aux=ZFSWL3(JGPTOTG)/ZPSOL
              ZFSWL4_aux=ZFSWL4(JGPTOTG)/ZPSOL

              !CALCUL DU SWI CEP
              SWI_CEP1= (ZFSWL1_aux-ZPWWILT_CEP)/(ZPWCAP_CEP-ZPWWILT_CEP)
              SWI_CEP2= (ZFSWL2_aux-ZPWWILT_CEP)/(ZPWCAP_CEP-ZPWWILT_CEP)
              SWI_CEP3= (ZFSWL3_aux-ZPWWILT_CEP)/(ZPWCAP_CEP-ZPWWILT_CEP)
              SWI_CEP4= (ZFSWL4_aux-ZPWWILT_CEP)/(ZPWCAP_CEP-ZPWWILT_CEP)

        !     SCALES SWI by (rsmin/lai)isba / (rsmin/lai)ifs
```

```
            IF ((ZFPLAI(JGPTOTG) /= 0.0).AND.(ZFRSMIN(JGPTOTG) /= 0.0)) THEN

             LAIscal=(ZFPRSMIN(JGPTOTG)*ZFLAI(JGPTOTG))/   &
          &          (ZFPLAI(JGPTOTG)*ZFRSMIN(JGPTOTG))

             SWI_CEP1=SWI_CEP1*LAIscal
             SWI_CEP2=SWI_CEP2*LAIscal
             SWI_CEP3=SWI_CEP3*LAIscal
             SWI_CEP4=SWI_CEP4*LAIscal

            ENDIF

            !CALCUL w ISBA
            ZGOL1= ZFPWWILT(JGPTOTG)+ ((ZFPWFC(JGPTOTG)-
ZFPWWILT(JGPTOTG))*SWI_CEP1)
            ZGOL2= ZFPWWILT(JGPTOTG)+ ((ZFPWFC(JGPTOTG)-
ZFPWWILT(JGPTOTG))*SWI_CEP2)
            ZGOL3= ZFPWWILT(JGPTOTG)+ ((ZFPWFC(JGPTOTG)-
ZFPWWILT(JGPTOTG))*SWI_CEP3)
            ZGOL4= ZFPWWILT(JGPTOTG)+ ((ZFPWFC(JGPTOTG)-
ZFPWWILT(JGPTOTG))*SWI_CEP4)

            IF (ZGOL1 > ZFPWFC(JGPTOTG)) ZGOL1 = ZFPWFC(JGPTOTG)
            IF (ZGOL2 > ZFPWFC(JGPTOTG)) ZGOL2 = ZFPWFC(JGPTOTG)
            IF (ZGOL3 > ZFPWFC(JGPTOTG)) ZGOL3 = ZFPWFC(JGPTOTG)
            IF (ZGOL4 > ZFPWFC(JGPTOTG)) ZGOL4 = ZFPWFC(JGPTOTG)

            !Calcul w ISBA en kg/m3
            ZGOL1=ZGOL1*ZSCALE
            ZGOL2=ZGOL2*ZSCALE
            ZGOL3=ZGOL3*ZSCALE
            ZGOL4=ZGOL4*ZSCALE

            IF (ZFPD2(JGPTOTG) > (PPZD1+PPZD2+PPZD3)) THEN
             ZGOL=ZGOL1*PPZD1+ZGOL2*PPZD2+ZGOL3*PPZD3+ &
          &    ZGOL4*(ZFPD2(JGPTOTG)-(PPZD1+PPZD2+PPZD3))
            ELSEIF (ZFPD2(JGPTOTG) > (PPZD1+PPZD2)) THEN
             ZGOL=ZGOL1*PPZD1+ZGOL2*PPZD2+ZGOL3*(ZFPD2(JGPTOTG)-
(PPZD1+PPZD2))
            ELSEIF (ZFPD2(JGPTOTG) > PPZD1) THEN
             ZGOL=ZGOL1*PPZD1+ZGOL2*(ZFPD2(JGPTOTG)-PPZD1)
            ELSE
             ZGOL=ZGOL1*ZFPD2(JGPTOTG)
            ENDIF

            IF (LLGLACE) THEN
              IF (.NOT.LLCDST.OR..NOT.LLSTL4) THEN
                WRITE(NULOUT,*)&
                 & 'PROFRESERV.GLACE CANNOT BE COMPUTED BECAUSE CDST OR
STL4 IS NOT READ'
                IF (LLCONTROL) CALL ABOR1('CPREP1 : ABOR1 CALLED')
              ENDIF
```

```
          IF (ZFPROFT(JGPTOTG) > (TSRESERV2+TDELTA2)) THEN
            ZPGLACE=0.0
          ELSEIF (ZFPROFT(JGPTOTG) < (TSRESERV2-TDELTA2)) THEN
            ZPGLACE=1.0
          ELSE
            ZDELTA=((ZFPROFT(JGPTOTG)-TSRESERV2)/(2.0*TDELTA2)) - 0.5
            ZPGLACE=MIN(((3.0_JPRB*ZDELTA*ZDELTA)  &
             & + (2.0_JPRB*ZDELTA*ZDELTA*ZDELTA)),1.0_JPRB)
          ENDIF
          ZFGLACE(JGPTOTG)=MIN(ZGOL*ZPGLACE,150.0_JPRB)
          ZFAGG(JGPTOTG)=ZGOL-ZFGLACE(JGPTOTG)
        ELSE
          ZFAGG(JGPTOTG)=ZGOL
        ENDIF
      ELSE
        ZFAGG(JGPTOTG)=SDEPX*GCONV
        IF (LLGLACE) ZFGLACE(JGPTOTG)=0.0_JPRB
      ENDIF
    ENDDO

    CALL FAIENC(IREP,NPOSSH,'PROF',0,'RESERV.EAU  ',ZFAGG,.FALSE.)
    WRITE(NULOUT,*)'--------------------------------------------'
    CLLIBELLE='PROFRESERV.EAU field'
    CALL CPREP1_DIAG(ZFAGG,CLLIBELLE)
    WRITE(NULOUT,*) 'Write grid-point field '
    IF (LLGLACE.AND.LLCDST.AND.LLSTL4) THEN
      CALL FAIENC(IREP,NPOSSH,'PROF',0,'RESERV.GLACE',ZFGLACE,.FALSE.)
      WRITE(NULOUT,*)'--------------------------------------------'
      CLLIBELLE='PROFRESERV.GLACE field'
      CALL CPREP1_DIAG(ZFGLACE,CLLIBELLE)
      WRITE(NULOUT,*) 'Write grid-point field '
    ENDIF

  ELSE
    WRITE(NULOUT,*)&
     & 'PROFRESERV.EAU NOT COMPUTED BECAUSE MISSING PARAMETERS IN
GRIB FILE'
    IF (LLCONTROL) CALL ABOR1('CPREP1 : ABOR1 CALLED')
  ENDIF

  IF(LLGRB190.AND.LLGRB191.AND.LLGRB192.AND.LLGRB193) THEN

  ! ** Set orography variance.

  ! One first sets zfagg to zero.
  ! Then one cumulates in zfagg the 190, 191, 192, 193
  ! grib code fields.
  ! The result is put on FA file.

    ZFAGG(1:NGPTOTG)=0.0_JPRB
    DO JSTOIA=1,IGRBSN
```

```
    IF(IGRBS(JSTOIA) == 190 &
       & .OR.IGRBS(JSTOIA) == 191 &
       & .OR.IGRBS(JSTOIA) == 192 &
       & .OR.IGRBS(JSTOIA) == 193) THEN

! The stored field has to be cumulated in the total
! variance.

      ZFAGG(1:NGPTOTG)=ZFAGG(1:NGPTOTG)+ZFAGGA(1:NGPTOTG,JSTOIA)
    ENDIF
  ENDDO
  DO JGPTOTG=1,NGPTOTG
    ZFAGG(JGPTOTG)=RG*SQRT(ZFAGG(JGPTOTG))
  ENDDO
  CLLIBELLE='Orography variance'
  CALL CPREP1_DIAG(ZFAGG,CLLIBELLE)

  LLOUTFS=.FALSE.
  CLPREF='SURF'
  INIVEAU=0
  CLSUFF='ET.GEOPOTENT'
  CALL FAIENC(IREP,NPOSSH,CLPREF,INIVEAU,CLSUFF,ZFAGG,LLOUTFS)
  WRITE(NULOUT,*) 'Write grid-point field ',CLPREF(1:4),INIVEAU,CLSUFF(1:16)
ENDIF

!Deallocate

IF (ALLOCATED(IGRIB)) DEALLOCATE(IGRIB)
IF (ALLOCATED(ZGRBDAT)) DEALLOCATE(ZGRBDAT)

! ** Close FA file.

CALL LFILAF(IREP,NPOSSH,.FALSE.)
CALL FAIRME(IREP,NPOSSH,'UNKNOWN')
IF (LLCLIM) THEN
  CALL FAIRME(IREP,IUCLIM,'UNKNOWN')
ENDIF

IF (LHOOK) CALL DR_HOOK('CPREP1',1,ZHOOK_HANDLE)

! ---------------------------------------------------------------

CONTAINS

SUBROUTINE CPREP1_DIAG(P_ZZFAGG,CDFIELD)
IMPLICIT NONE

REAL(KIND=JPRB), INTENT(IN), DIMENSION(NGPTOTG) :: P_ZZFAGG
CHARACTER(LEN=*), INTENT(IN) :: CDFIELD

REAL(KIND=JPRB) :: ZZMAX, ZZMIN
REAL(KIND=JPRB) :: ZHOOK_HANDLE
```

```
IF (LHOOK) CALL DR_HOOK('CPREP1:CPREP1_DIAG',0,ZHOOK_HANDLE)
ZZMIN=MINVAL(P_ZZFAGG(1:NGPTOTG))
ZZMAX=MAXVAL(P_ZZFAGG(1:NGPTOTG))
!WRITE(NULOUT,*)'---------------------------------------------'
WRITE(NULOUT,*) TRIM(CDFIELD)
WRITE(NULOUT,*) 'Min=',ZZMIN,' Max=',ZZMAX
IF (LHOOK) CALL DR_HOOK('CPREP1:CPREP1_DIAG',1,ZHOOK_HANDLE)

END SUBROUTINE CPREP1_DIAG

END SUBROUTINE CPREP1
```

Annex II

Technical Memorandum

The modifications in the cprep1.F90 routine were done in the meteo france machine “tori” under the user mrpa672. The initial base routine was written by Laurent Descamp, Descamp, L. (2007): and copied to:

tori://~/pack/cprep1/src/local/arp/control

the “pack” directory was built with gmkpack version 6.3.1, with the command:

gmkpack –r cy33t0 –b op1B –v 12 –u cprep1 –l SX8RV20 –o x –p aladin

After the modifications, the compilation is made with:

qsub ~/pack/cprep1/ics_aladin

that will produce the executable:

tori://~/pack/cprep1/bin/ALADIN_work

The GRIB files for a particular day are obtained from ECMWF, with the MARS request “requete_mars.txt” in

ecgate://home/ms/pt/pt7

The 3 GRIB files for each day are downloaded to:

tori://~/work/mrpa672

After the GRIB files are in place, one should run the five existing scripts in the directories:

tori://~/cprep1_scripts

and

tori://~/cprep1_scripts/common

in the following order:

qsub step1_C901_restart.F creates the orography file as:
tori://~/work/mrpa672/CN90xa001INIT.F

qsub step2_C923_restart creates the climatology files as:
tori://~/work/Const.Clim.mXX where is XX is the month

(important note: these 2 steps are required to run only once for each geometry)

qsub step3_new runs configuration 901 with the new changes in cprep1.F90 and creates:
tori://~/work/CN90xa001INIT.T

qsub step4_C927_restart_ald runs configuration 927 and creates:

tori://~/work/PFFPOS000+0000

./myqsub step5_e001 runs configuration 001 and creates:

tori://utmp/ftdir/mrpa672/mtool/spool/spool_XXXXXX/ICMSHALAD+HHHH, with XXXXXX always increasing on each myqsub submission and ICMSHALAD+HHHH the historic files with HHHH being the step in hours.