D. Liermann, November 2019

# TSA – Meteorological forcing: Some information

Meteorological forcing data are read in every hour (oldbc, newbc) and the fields are interpolated in time to get values for the current time step nt (oldbc < "nt" < newbc):

- Temperature, wind, moisture, surface pressure are interpolated linearly (see chapter 1)
- Precipitation is required as rate for TSA; the average rate of the (1hour) period is taken for every time step of this interval (for details see chapter 2)
- Solar and thermal radiation at the ground (sobs, thbs, net fluxes) are needed for TSA, time average fluxes are read in (ASOB_S/ATHB_S) for both times oldbc and newbc and the average is done for this time interval (chapter 3)
- *Some remarks for ICON input are in blue.*

## 1. "Atmospheric" forcing

The NL(METFORCING)  parameter ntype_atminput controls the atmospheric forcing, .e. g. which fields are read in (**read_lmgrib**) for temperature (t_in), wind (u_in, v_in), surface humidity (qv_in) and surface pressure (ps_in); see also TSA TABLE, → means conversion:

| ntype_atminput = 1 | ntype_atminput = 2 | ntype_atminput = 3 | Input name | TSA name |
|---|---|---|---|---|
| U,V (ke:ke+1) → SP (ke:ke+1) | SP_10M - nudging | SP_10M - analysis | u_in | u |
| T(ke:ke+1) | T_2M - nudging | T_2M - analysis | t_in | t |
| QV(ke:ke+1) | TD_2M – nudging → QV_2M | RELHUM_2M – analysis → QV_2M | qv_in | qv |
| PS | PS | PS | ps_in | ps |

*ntype_atminput=2 not possible for ICON (atmospheric fields should be analysis of forecast fields).*

Input varibales are stored in …_bd arrays:

```
IF (ntype_atminput<=3) THEN
  IF (lcrop) THEN
    u_bd(:,:,ke,nx) =u_in (i0_crop:i0_crop+ie-1,j0_crop:j0_crop+je-1)
    t_bd(:,:,ke,nx) =t_in (i0_crop:i0_crop+ie-1,j0_crop:j0_crop+je-1)
    qv_bd(:,:,ke,nx)=qv_in(i0_crop:i0_crop+ie-1,j0_crop:j0_crop+je-1)
!     qv_bd(:,:,ke)=qv_in(i0_crop:i0_crop+ie-1,j0_crop:j0_crop+je-1)    !XYZ> in TSA, qv is 4 dimensional
    ps_bd(:,:,nx)   =ps_in(i0_crop:i0_crop+ie-1,j0_crop:j0_crop+je-1)
  ELSE
..... (spatial interpolation)
```

In **read_metforc** (after read_lmgrib/*read_icongrib*): linear interpolation to current time step

```
 !=======================================================
 ! Section 5: Temporal Interpolation
 !=========================================================
 fac = (date_delta(actual_date*100,oldbc_date*100)    &
      + (acthour-INT(acthour))*60.0_wp) / REAL(delta_bc)
 fac1=1.0-fac

 ! Atmospheric variables at the reference level
 IF (ntype_atminput<=4) THEN
   u(:,:,ke,nnew) = fac1*u_bd(:,:,ke,n1) + fac*u_bd(:,:,ke,n2)
   t(:,:,ke,nnew) = fac1*t_bd(:,:,ke,n1) + fac*t_bd(:,:,ke,n2)
   qv(:,:,ke,nnew)= fac1*qv_bd(:,:,ke,n1) + fac*qv_bd(:,:,ke,n2)
!   qv(:,:,ke)= fac1*qv_bd(:,:,ke) + fac*qv_bd(:,:,ke)             !XYZ> uncommented qv non-tracer is 4 dimentional
   ps(:,:,nnew)  = fac1*ps_bd(:,:,n1) + fac*ps_bd(:,:,n2)
 ELSE
   WRITE(6,*) "Invalid ntype_atminput! (Time interpolation)"
   STOP
  ENDIF
....

IF (lcalc) THEN
   !=========================================================
   ! Section 6: Conversions
   !=========================================================
   ! Copy time levels
   u(:,:,ke,nnow)  = u(:,:,ke,nnew)
   t(:,:,ke,nnow)  = t(:,:,ke,nnew)
   qv(:,:,ke,nnow) = qv(:,:,ke,nnew)
   ps(:,:,nnow)    = ps(:,:,nnew)
   p0(:,:,ke)      = ps(:,:,nnew) * EXP( -g*dz/(r_d*t(:,:,ke,nnew)) )

...

! diagnosis of T_2m and u_10m
   IF ( ABS(dz_u-10.0_wp) < 1.0E-6_wp ) THEN
     u_10m(:,:) = u(:,:,ke,nnew)
     lu10m=.FALSE.
   ELSE IF ( dz_u>10.0_wp) THEN
```

```
    lu10m=.TRUE.
  ELSE
    WRITE(6,*) "dz_u lower than 10m is not allowed!"
  ENDIF

  IF ( ABS(dz-2.0_wp) < 1.0E-6_wp ) THEN
    t_2m(:,:)  = t(:,:,ke,nnew)
    lt2m=.FALSE.
  ELSE IF ( dz > 2.0_wp ) THEN
    lt2m=.TRUE.
  ELSE
    WRITE(6,*) "dz lower than 2m is not allowed!"
  ENDIF

  IF ( lt2m .OR. lu10m) THEN
   CALL near_surface(nnew,lt2m,lu10m)
  ENDIF
…
```

QUESTIONS:

u_10m? Seems to be SP_10M?!

In SR init_variables: v=0,pp=0. Where is pp computed? Only p0 needed?

2. **Precipitation**

Input for “Terra” or TSA are precipitation rates, e. g. the average rainfall and snowfall rate of the time interval oldbc → newbc is taken for every time step. It has to be checked, if there are precipitation amounts over one hour (lhourly_data = .true. is set in the NL METFORCING, but not checked in the code) or accumulated values since start of nudging (forecast) run.

- Here are the corresponding lines of the code:

**read_metforc**

|

>>>>>>> call **read_lmgib / call read_icongrib**

* read in RAIN_GSP/CON and SNOW_GSP/CON and store in *rain_gsp/con_in* and *snow_gsp/con_in* *(alternative: TOT_PREC → tot_prec_in)*

* NEW: when missing (all values == rundef), rates = 0.0

```
* rain_gsp_in=rain_gsp_in/3600.0
  rain_con_in=rain_con_in/3600.0
```

```
snow_gsp_in=snow_gsp_in/3600.0

snow_con_in=snow_con_in/3600.0

tot_prec_in=tot_prec_in/3600.0_wp
```

* The sum of (rain_gsp_in + rain_con_in) rsp. (snow_gsp_in + snow_con_in) is stored in *prr_gsp_bd* rsp. *prs_gsp_bd* *(or tot_prec_in in prr_gsp_bd with prs_gsp_bd=0.0)*

```
IF (ltot_prec) THEN

   prr_gsp_bd(:,:,nx) = tot_prec_in

   prs_gsp_bd(:,:,nx) = 0.0_wp

ELSE

    prr_gsp_bd(:,:,nx) = (rain_gsp_in + rain_con_in)

    prs_gsp_bd(:,:,nx) = (snow_gsp_in + snow_con_in)

 ENDIF


IF (lcrop) THEN

            prr_gsp_bd(:,:,nx) =(rain_gsp_in(i0_crop:i0_crop+ie-1,j0_crop:j0_crop+je-1)+ &

                         rain_con_in(i0_crop:i0_crop+ie-1,j0_crop:j0_crop+je-1))

            prs_gsp_bd(:,:,nx) =(snow_gsp_in(i0_crop:i0_crop+ie-1,j0_crop:j0_crop+je-1)+ &

                         snow_con_in(i0_crop:i0_crop+ie-1,j0_crop:j0_crop+je-1))

ELSE

…
```

```
| !===========================================================

 ! Section 5: Temporal Interpolation

 !===========================================================

! Rain data

 IF (ntype_raininput==1) THEN

   IF (lnew_lmana) THEN

     IF (lhourly_data) THEN

       prr_gsp(:,:)= prr_gsp_bd(:,:,n2)

       prs_gsp(:,:)= prs_gsp_bd(:,:,n2)

     ELSE IF (nave_new>nave_old) THEN

       prr_gsp(:,:)= (prr_gsp_bd(:,:,n2)-prr_gsp_bd(:,:,n1))   / FLOAT(nave_new-nave_old)

       prs_gsp(:,:)= (prs_gsp_bd(:,:,n2)-prs_gsp_bd(:,:,n1))   / FLOAT(nave_new-nave_old)

     ELSE

       prr_gsp(:,:)= prr_gsp_bd(:,:,n2)/FLOAT(nave_new)
```

```
        prs_gsp(:,:)= prs_gsp_bd(:,:,n2)/FLOAT(nave_new)
      ENDIF
    ENDIF
  ELSEIF ((ntype_raininput==2).OR.(ntype_raininput==4)) THEN
    IF (lnew_rado) THEN
      WHERE (t(:,:,ke,nnew)>273.15)
        prr_gsp(:,:)=prr_gsp_bd(:,:,nrain2)
        prs_gsp(:,:)=0.0_wp
      ELSEWHERE
        prr_gsp(:,:)=0.0_wp
        prs_gsp(:,:)=prr_gsp_bd(:,:,nrain2)
      END WHERE
    ENDIF
  ELSE
    WRITE(6,*) "Invalid ntype_raininput! (Time interpolation)"
  ENDIF
```

- *Take care for time units in COSMO and ICON, normally ICON output is in minutes and COSMO in hours; in the code it is assumed, that the time stamps are in hours*
- *prr_gsp/prs_gsp* are then used as average rates in the terra MODULE *src_soil_multlay* (new: sfc_terra) with SUBROUTINE *terra_multlay (new: terra)* for every time step in the time interval [oldbc, new_bc].

- SUBROUTINE init_variables (terra_lmenv.f90): Presettings prr_con, prs_con, prs_gsp mit 0.0. Why not prr_gsp?
- Attention: There could be a division by zero, if nave_new=0 !!
- QUESTIONS: What about graupel, hail if available? Why no splitting convective/grid scale prep.?

## 3. Radiation

Net radiation fluxes are required by TSA (?), although Julian Tödter (GUF) states that the radiation forcing data are longwave and shortwave downward radiation fluxes at the surface (so_down_bd, th_down_bd).

- Here are the corresponding lines of the code:

**read_metforc**

|

>>>>>>>                    call **read_lmgib**

                    * read in ASOB_S/ATHB_S and store *sobs2_in/ thbs2_in*

* NEW: in case of missing the fields are set to zero!! Or should there be an abort??

* NEW: if only the instant values SOBS_RAD/THBS_RAD are available they will be taken

** ! convert net radiation to downwelling radiation*

```
 sobs2_in(:,:) = sobs2_in(:,:) /(1.0_wp-albrad_in(:,:)/100.0_wp)

thbs2_in(:,:) = (thbs2_in(:,:)+(1.0_wp-ctalb)*sigma*t_g_in(:,:)**4)/(1.0_wp-ctalb)
```

* Downwelling radiation is stored in xx_down_bd (:,:,nx) for two time levels

*! check domains and if simple croping is possible*

```
IF (lcrop) THEN

    so_down_bd(:,:,nx) =sobs2_in (i0_crop:i0_crop+ie-1,j0_crop:j0_crop+je-1)

    th_down_bd(:,:,nx) =thbs2_in (i0_crop:i0_crop+ie-1,j0_crop:j0_crop+je-1)

    pabs_bd   (:,:,nx) = pabs_in (i0_crop:i0_crop+ie-1,j0_crop:j0_crop+je-1)
```

…..

| (read_metfor)

```
!========================================================

! Section 5: Temporal Interpolation

!========================================================

! Radiation

IF (ntype_radinput<=2) THEN

  IF (lnew_lmana) THEN

    IF (lhourly_data) THEN

      so_down_bd(:,:,n1) = so_down_bd(:,:,n2)

      th_down_bd(:,:,n1) = th_down_bd(:,:,n2)

      pabs = pabs_bd(:,:,n2)

    ELSE IF (nave_new>nave_old) THEN

      so_down_bd(:,:,n1) = (nave_new*so_down_bd(:,:,n2) - nave_old*so_down_bd(:,:,n1)) / FLOAT(nave_new-nave_old)

      th_down_bd(:,:,n1) = (nave_new*th_down_bd(:,:,n2) - nave_old*th_down_bd(:,:,n1)) / FLOAT(nave_new-nave_old)

      pabs = (nave_new*pabs_bd(:,:,n2) - nave_old*pabs_bd(:,:,n1)) / FLOAT(nave_new-nave_old)

    ELSE

      so_down_bd(:,:,n1) = so_down_bd(:,:,n2)

      th_down_bd(:,:,n1) = th_down_bd(:,:,n2)

      pabs = pabs_bd(:,:,n2)

    ENDIF

  ENDIF

ELSEIF (ntype_radinput==4) THEN
```

```
    IF (lnew_rad) THEN
      so_down_bd(:,:,n1)=so_down_bd(:,:,n2)
      th_down_bd(:,:,n1)=th_down_bd(:,:,n2)
    ENDIF
  ELSE
    WRITE(6,*) "Invalid ntype_radinput! (Time interpolation)"
    STOP
  ENDIF
.....
IF (lcalc) THEN          !!!! lcalc = .true.  (default in NL RUN_TERRA)   set in read_namelist

    !=======================================================
    ! Section 6: Conversions
    !=======================================================
    ! diagnosis of net radiation from down-welling fluxes
    DO i=1,ie
      DO j=1,je
        IF (llandmask(i,j)) THEN
          CALL calc_albedo(i,j,nnow,also,alth)
          sobs(i,j)= so_down_bd(i,j,n1)*(1.0_wp-also)
          thbs(i,j)= th_down_bd(i,j,n1)-(1.0_wp-alth)*sigma*t_g(i,j,nnow)**4
        ENDIF
      ENDDO
    ENDDO

    IF (ntype_radinput==4) THEN
      pabs=0.5_wp*sobs
    ENDIF
....

ENDIF
```

- *sobs and thbs* are used in MODULE *src_soil_multlay* (new; sfc_terra) with SUBROUTINE *terra_multlay (new: terra)* for every time step in the time interval [oldbc, new_bc].

Questions:

- Why sob2_in → so_down_bd → sobs   and   thbs2_in → th_down_bd → thbs?
  Why are input ASOB_S/ATHB_S not used directly??

- albrad_in is not used for second conversion, but SUBROUTINE calc_albedo is used (→ also, alth; includes bare soil, snow, vegetation). May be it is more precise to compute it again??

- If ALB_RAD is missing, what should be taken as default? Use calc_albedo?? Set to csalb_p=0.15?

- Remarks to SUBROUTINE calc_albedo: csalb_p is replaced by vegalb(im,jm). NL parameter of RUN_TERRA lconstvegalb (default: .true.) controls setting of vegalb in read_const_fields:

    - lconstvegalb=.true. : vegalb=csalb_p (for lhomosoil=.true. and .false.)

    - lconstvegalb=.false.: vegalb=vegalb_const (= NL input in EXTPARA for lhomosoil=.true.)

    - lconstvegalb=.false.: vegalb=vegalb_in (lhomosoil=.false.)

  ➔ **lconstvegalb=.true. AND lhomosoil =.false. (default is .true.!!!! should be changed!!!)**

  **“vegalb” will not be read in as lconstvegalb=.true.; so vegalb=csalb_p will be uses in calc_albedo**

- Julian Tödter has found an error concerning longwave net radiation, which seems to be not corrected!? Old: lwnet = lwdown – (1-a) * sigma *T**4 (a=0.004=ctalb(?))
  New: : lwnet=lwdown–lwup=lwdown-(a*lwdown+(1+a)*sigma*T**4)=(1-a)*(lwdown-sigma*T**4.
  See here parts of the code:

  1. read_lm/icongrib:

     (1) thbs2_in is read in

     (2) thbs2_in(:,:) = (thbs2_in(:,:)+(1.0_wp-ctalb)*sigma*t_g_in(:,:)**4)/(1.0_wp-ctalb)

     (3) th_down_bd(:,:,nx) = thbs2_in

  2. read_metforc

     (1) time average

     (2) CALL calc_albedo(i,j,nnow,also,alth)

     (3) thbs(i,j)= th_down_bd(i,j,n1)-(1.0_wp-alth)*sigma*t_g(i,j,nnow)**4