

Code for the implementation of the forcings

To use the transient forcings several changes need to be done in the source code of CCLM. The changes are explained in the following:

- In *src_radiation.f90* between line 1384 and 1396: add integer *mm*
- In *src_radiation.f90* between line 206 and 675: add variables for forcings as shown in Figure 1
- 5 – In *src_radiation.f90* between line 1539 and 1546: add reals *aod* (28812), *solc_tbl* (28224) and *solc*
- Create a new file in *src* which contains the values for the greenhouse gas, the orbital, the solar and the volcanic forcing. It is named *data_forcing.f90* and has the structure shown in Figure 2 and Figure 2 where the values according to the used dataset needs to be filled
- In *ObjDependencies* add the file *data_forcing.f90* to the section *Dependencies of the Data Modules*
- 10 – In *ObjFiles* add the file *data_forcing.o* to the section *DATAOBJ*

1 Greenhouse Gases

The greenhouse gas forcing is added as a namelist option. Originally one can choose for the namelist parameter *ico2_rad* different CO₂ or effective CO₂ scenarios with the numbers 1 to 10. Here we add option 11. To use the transient greenhouse gas forcing the following changes need to be done in the namelist and the source code:

- 15 – Set CCLM namelist parameter *ico2_rad* to 11

```
1  USE data_forcing, ONLY: &
2  aod_1,           & ! stratospheric aerosol optical depth
3  aod_2,           & ! stratospheric aerosol optical depth
4  aod_3,           & ! stratospheric aerosol optical depth
5  solc_1,          & ! solar constant
6  solc_2,          & ! solar constant
7  solc_3,          & ! solar constant
8  recc,            & ! eccentricity
9  robld,           & ! obliquity
10 rlonp,           & ! perihelion
11 tco2,            & ! CO2
12 tch4,            & ! CH4
13 tn o             ! N2O
```

Figure S1. Code to be implemented in *src_radiation.f90* for the transient greenhouse gas forcing.

```

1  !+ Data module for all parametric data for the climate forcings
2  !-----
3
4  MODULE data_forcing
5
6  !-----
7  !
8  ! Description:
9  ! This module declares and initializes all parametric scalar and array
10 ! data which are used for the climate forcings
11 !
12 ! Current Code Owner: Eva Hartmann
13 ! email: Eva.Hartmann@geogr.uni-giessen.de
14 !!
15 ! Code Description:
16 ! Language: Fortran 90.
17 ! Software Standards: "European Standards for Writing and
18 ! Documenting Exchangeable Fortran 90 Code".
19 !=====
20 !
21 ! Modules used:
22 USE data_parameters, ONLY : &
23     ireals, & ! KIND-type parameter for real variables
24     integers ! KIND-type parameter for standard integer variables
25
26 !=====
27
28 IMPLICIT NONE
29
30 !=====
31
32 ! Global (i.e. public) Declarations:
33
34 ! 1. Data arrays for properties of different forcings
35 ! -----
36
37 REAL (KIND=ireals) :: &
38 ! a) parameters for the volcanic forcing
39 aod_1 (6000), & ! stratospheric aerosol optical depth
40 aod_2 (12000), & ! stratospheric aerosol optical depth
41 aod_3 (10812), & ! stratospheric aerosol optical depth
42
43 ! b) parameters for the orbital forcing
44 recc (3451), & ! eccentricity
45 robld (3451), & ! obliquity
46 rlonp (3451), & ! longitude of perihelion
47
48 ! c) parameters for the solar forcing
49 solc_1 (6000), & ! solar constant
50 solc_2 (12000), & ! solar constant
51 solc_3 (10224), & ! solar constant
52
53 ! d) parameters for the greenhouse gases
54 tco2 (2351), & ! CO2 concentration
55 tch (2351), & ! CH4 concentration
56 tn o (2351) ! N2O concentration
57
58 ! Initialization of forcing values
59
60 ! Volcanic Forcing data on monthly resolution for chosen Latitudes
61 DATA aod_1 / 0.0049, 0.0047, 0.0045, ... / ! monthly_data from 500 to 1 BCE
62
63 DATA aod_2 / ... / ! monthly_data from 0 to 999 CE
64
65 DATA aod_3 / ... / ! monthly_data from 1000 to 1900 CE
66
67 ! Solar Forcing data on monthly resolution
68 DATA solc_1 / 1360.7570, 1360.7861, ... / ! monthly_data from 500 to 1 BCE
69
70 DATA solc_2 / ... / ! monthly_data from 0 to 999 CE
71
72 DATA solc_3 / ... / ! monthly_data from 1000 to 1851 CE
73

```

Figure S2. Code of the new created file *data_forcing.f90* for the transient forcing values - part I.

```

74 ! Orbital Forcing data on yearly resolution
75 ! (Eccentricity, Obliquity, Longitude of Perihelion)
76 DATA recc / 0.017640, 0.017640, ... / ! yearly data from 500 BCE to 2950 CE
77
78 DATA robld / 23.754, 23.754, ... / ! yearly data from 500 BCE to 2950 CE
79
80 DATA rlonp / 240.46, 240.48, ... / ! yearly data from 500 BCE to 2950 CE
81
82 ! Greenhouse Gas Concentration on yearly resolution
83 ! (CO2, CH4, N2O)
84 DATA tco2 / 2.76197e-04, 2.76199e-04, ... / ! yearly data from 500 BCE to 1850 CE
85
86 DATA tch4 / 6.1082e-07, 6.1085e-07, ... / ! yearly data from 500 BCE to 1850 CE
87
88 DATA tn2o / 2.6911e-07, 2.6910e-07, ... / ! yearly data from 500 BCE to 1850 CE
89
90 !======
91
92 END MODULE data_forcing
93

```

Figure S3. Code of the new created file *data_forcing.f90* for the transient forcing values - part II.

```

1      CASE (11)
2          ! specific transient effective CO2
3          ! 25 * ch4 and 298 * n2o
4          zqco2 = ( tco2(jj-6999+500) &
5                  + tch4(jj-6999+500)*25 &
6                  + tn2o(jj-6999+500)*298 ) * 1.519_ireals

```

Figure S4. Code to be implemented in *src_radiation.f90* for the transient greenhouse gas forcing.

- In *organize_physics.f90* line 2673: change 10 to 11
- In *src_radiation.f90* after line 1870: add case 11 as shown in Figure 4

2 Orbital Forcing

The orbital forcing is added as three values on yearly resolution. They are the eccentricity, the obliquity and the longitude of perihelion of the earth's orbit. The original code is owned by Patrick Ludwig and was modified for the transient simulation. To use the transient orbital forcing the following changes need to be done in the source code:

- In *src_radiation.f90* from line 2042 to 2058: comment out and replace by:
- In *src_radiation.f90* after line 2041: add orbital forcing as shown in Figure 5 and Figure 6

3 Solar Forcing

The solar forcing is added as monthly value for the solar constant. To use the transient solar forcing the following changes need to be done in the source code:

```

1  if (itype_calendar==0) then
2      year1 = 365 + IABS( MOD(jj,4) - 4) / 4
3  else
4      year1 = 360
5  end if
6
7      api = 2.*ASIN(1.) ! Pi
8      zve 0 = 80.5 ! Day of the vernal equinox in 1900
9      zvebas = year1-zve00 ! Remainder of the year
10     zyearve = float(itaja) - 1. + zvebas ! itaja: actual day of the year
11     ! zyearve: time of the year from
12     ! last years vernal equinox in degrees
13     ! zvetim = MOD(zyearve/year1,1.)*2.*api
14     ! Time of the year from the vernal equinox
15     ! in radians
16
17     zyeardif=zyearve/year1
18     zone=1.
19     zvetim = MOD(zyeardif,zone)*2.*api
20
21     zecc=recc(jj-6999+500)
22     zobld=robl(jj-6999+500)
23     zlonp=rlopp(jj-6999+500)
24
25     zoblr=zobld*api/180.
26     zlonpr=zlonp*api/180.
27     zsqecc=SQRT((1+zecc)/(1-zecc))
28
29     zeps=1.E-6
30
31     ! CALCULATION OF ECCENTRIC ANOMALY OF VERNAL EQUINOX!
32     zeve=2.*ATAN(TAN(0.5*zlonpr)/zsqecc)
33
34     ! CALCULATION OF TIME ANGLE IN RADIAN OF LONGITUDE OF PERIHELION
35     ! FROM VERNAL EQUINOX
36     ztlonpr= zeve - zecc*SIN(zeve)
37
38     ! CALCULATE ECCENTRIC ANOMALY:
39     ! USE FIRST DERIVATIVE (NEWTON) TO CALCULATE SOLUTION FOR
40     ! EQUATION OF ECCENTRIC ANOMALY *ZENEW*
41     zotime=zvetim-ztлонpr
42
43     zeold=zotime/(1.-zecc)
44     zenew=zotime
45     iter=0
46
47     DO
48         zzeps=zeold-zenew
49         IF (iter.GE.10) THEN
50             yzerrmsg = 'PALEO ORBIT: - eccentric anomaly not found!'
51             CALL model_abort (my_world_id, 2091, yzerrmsg, 'radiation')
52         END IF
53         IF (ABS(zzeps).LT.zeps) EXIT
54         iter=iter+1
55         zeold=zenew
56         zcose=COS(zenew)
57         zenew=(zotime+zecc*(SIN(zenew)-zenew*zcose))/(1.-zecc*zcose)
58     END DO
59
60     zsocof=(1./(1.-zecc*COS(zenew)))**2
61
62     ! CALCULATION OF THE DECLINATION.
63     ztgean=TAN(zenew*0.5)
64
65     ! *znu*: TRUE ANOMALY
66     ! (ACTUAL ANGLE OF EARTH'S POSITION FROM PERIHELION)
67     ! *zlambda*: TRUE LONGITUDE OF THE EARTH
68     ! (ACTUAL ANGLE FROM VERNAL EQUINOX)
69     znu=2.*ATAN(zsqecc*ztgean)
70     zlambda=znu+zlonpr
71     zsinde=SIN(zoblr)*SIN(zlambda)
72     zdek=ASIN(zsinde)
73
74     zdeksin_save = SIN (zdek)

```

Figure S5. Code to be implemented in *src_radiation.f90* for the transient orbital forcing - part I.

```

74  zdekcos_save = COS (zdek)
75
76  zsct_save = zsocof*solc
77
78  zsct_h = zsct_h + zsct_save
79  nz_zsct = nz_zsct + 1
80
81  ztwo = 0.681 + 0.2422*(jj-1949)-(jj-1949)/4
82  ztho = 2.*pi*( REAL(itaja, ireals) -1.0 + ztwo )/365.2422
83  zdtzgl_save = 0.000075 + 0.001868*COS( ztho) - 0.032077*SIN( ztho) &
84  - 0.014615*COS(2.*ztho) - 0.040849*SIN(2.*ztho)

```

Figure S6. Code to be implemented in *src_radiation.f90* for the transient orbital forcing - part II.

```

1      solc_tbl(1:6000) = solc_1
2      solc_tbl(6001:18000) = solc_2
3      solc_tbl(18001:28224) = solc_3

```

Figure S7. Code to be implemented in *src_radiation.f90* for the transient solar forcing - part I.

- In *src_radiation.f90* line 289: comment out old solar constant
- In *src_radiation.f90* after line 2033 and after line 3623: add solar forcing values as tables shown in Figure 7
- In *src_radiation.f90* after line 2036 and after line 3628: add how to get the solar forcing values for the actual month as shown in Figure 8

4 Volcanic Forcing

The volcanic forcing is added as monthly value for the stratospheric aerosol optical depth. The raw data is given in latitudinal bands. As preparation, the latitudes covering the domain of the RCM simulations are selected and we continue with an average of those latitudes. To use the transient volcanic forcing the following changes need to be done in the source code:

- In *src_radiation.f90* after line 1887: add volcanic forcing values as tables shown in Figure 9
- In *src_radiation.f90* line 1890 and line 7991: add variables *aod*, *ntstep* and *ydate_ini* to the function *aerdis*
- In *src_radiation.f90* after line 7884: add additional local variables for the calculation of the volcanic forcing as shown in Figure 10

```

1      READ (yradl(5:6), '(I2)') mm
2      solc = solc_tbl((jj-7000+500)*12 + mm)

```

Figure S8. Code to be implemented in *src_radiation.f90* for the transient solar forcing - part II.

```

1      aod(1:6000) = aod_1
2      aod(6001:18000) = aod_2
3      aod(18001:28812) = aod_3

```

Figure S9. Code to be implemented in *src_radiation.f90* for the transient volcanic forcing - part I.

```

1      REAL (KIND=ireals), DIMENSION(28812), INTENT(IN) :: &
2      aod ! aerosol optical depth of the volcanic forcing file
3      INTEGER (KIND=iintegers), INTENT(IN) :: &
4      ntstep
5      INTEGER (KIND=iintegers) :: &
6      itaja, & !
7      ist, & ! number of aod values
8      mm, & ! output from routine
9      jj ! output from routine
10     REAL (KIND=ireals ) :: &
11     zstunde ! output from routine get_utc_dat
12     CHARACTER (LEN=14) yrad1 ! output from routine get_utc_dat
13     CHARACTER (LEN=28) yrad2 ! output from routine get_utc_dat
14
15     CHARACTER (LEN=14), INTENT(IN) :: ydate_ini ! start of the forecast yyyyymmddhh

```

Figure S10. Code to be implemented in *src_radiation.f90* for the transient volcanic forcing - part II.

- In *src_radiation.f90* line 7912: comment out the calculation of *pstbga* and replace by:
- 40 – In *src_radiation.f90* after line 7912: add additional calculation of the volcanic forcing as shown in Figure 11

5 Land-Use Change

To make the land-use change transient in the CCLM simulation one has to convert the ESM output to the needs of CCLM with an appropriate converter for external data. It could be created a new external data file for each month, each year or whatever is needed. This can be read in by the INT2LM routine during the simulation run.

```

1      CALL get_utc_date ( ntstep, ydate_ini, dt, itype_calendar, yrad1, yrad2, &
2      itaja, zstunde )
3
4      READ (yrad1(1:4), '(I4)') jj
5      READ (yrad1(5:6), '(I2)') mm
6
7      ist = ((jj-7000 + 500) * 12) + mm
8
9      pstbga = aod(ist) / 19330.0_ireals

```

Figure S11. Code to be implemented in *src_radiation.f90* for the transient volcanic forcing - part III.