DecoChem v1.0 source code

```
program decochem
  implicit none
  character(len=80) :: inFile,climFile,paramFile,outFolder
  integer :: steps
  real :: airtemp,Rdecomp,Rlitter,Rsoil,&
       rate,q10,f_lignin,kslow,kfol,kroot,&
       kwood,kfast,efficiency_decomposition,&
       Cslow,Cfast,gluc,CLr,&
       CLf,CLw,Lf,Lr,Lw
  real :: ts_length,dCroot_dt,dCfoliage_dt,&
       dCwood_dt,dCslow_dt,dCfast_dt,dCO2_dt,totalC,numsecs
  integer :: i,j,k
  real :: flux1,flux2,flux3,flux4,flux5,flux6,&
       flux7,flux8,flux9,flux10,flux11,flux12,&
       flux13,flux14,flux15,flux16
  real :: sumLf,sumLr,sumLw,sumRdecomp,sumRlitter,&
       sumRsoil,sumFlux1,sumFlux2,sumFlux3,sumFlux4,sumFlux5,&
       sumFlux6,sumFlux7,sumFlux8,sumFlux9,sumFlux10,&
       sumFlux11,sumFlux12,sumFlux13,sumFlux14,sumFlux15,sumFlux16,&
       sumdCfoliage_dt,sumdCroot_dt,sumdCwood_dt,&
       sumdCfast_dt,sumdCslow_dt,sumGluc
  real :: an_sumFlux1,an_sumFlux2,an_sumFlux3,an_sumFlux4,&
       an_sumFlux5,an_sumFlux6,an_sumFlux7,an_sumFlux8,&
       an_sumFlux9,an_sumFlux10,an_sumFlux11,an_sumFlux12,&
       an_sumFlux13,an_sumFlux14,an_sumFlux15,an_sumFlux16,&
       an_sumdCfoliage_dt,an_sumdCroot_dt,an_sumdCwood_dt,&
       an_sumdCfast_dt,an_sumdCslow_dt,&
       an_sumLf,an_sumLr,an_sumLw,an_sumGluc,&
       an_sumRlitter,an_sumRdecomp,an_sumRsoil
  real :: avgFlux1,avgFlux2,avgFlux3,avgFlux4,avgFlux5,&
       avgFlux6,avgFlux7,avgFlux8,&
       avgFlux9,avgFlux10,avgFlux11,avgFlux12,&
       avgFlux13,avgFlux14,avgFlux15,avgFlux16,&
       avgdCfoliage_dt,avgdCroot_dt,avgdCwood_dt,&
       avgdCfast_dt,avgdCslow_dt,&
       avgLf,avgLr,avgLw,avgRlitter,avgRdecomp,avgRsoil,avgGluc
  real(kind=8) :: preC,curC,&
       carbongain,carbonloss,&
       sumflux,delC,checkdiffC
  real,allocatable :: Lft(:),Lrt(:),Lwt(:),Rdecompt(:),&
       Rlittert(:),Rsoilt(:),climate(:,:),flux1t(:),flux2t(:),&
       flux3t(:),flux4t(:),flux5t(:),flux6t(:),flux7t(:),&
       flux8t(:),flux9t(:),flux10t(:),flux11t(:),flux12t(:),&
       flux13t(:),flux14t(:),flux15t(:),flux16t(:),&
       dCfoliage_dtt(:),dCroot_dtt(:),dCwood_dtt(:),&
       dCfast_dtt(:),dCslow_dtt(:),gluct(:)
  integer :: daysYear,years
  integer :: stat,sta
  character(len=20) :: dummy
  character(len=3) :: soilExtra
  character(len=10) :: outFreq
  character(len=20) :: dum
  real,parameter :: seconds_per_step=3600

  ! ---- START  ---- !

  stat=0
  call get_command_argument( 1,inFile )

  ! Check if a file argument was given
  open(100,file=inFile,status='old',iostat=stat)
  if(stat/=0) then
     write(*,*) 'No input file entered.'
     stop
  endif

  ! Read the input file
```

```
read(100,*) dum,climFile
read(100,*) dum,paramFile
read(100,*) dum,outFolder
read(100,*) dum,daysYear
read(100,*) dum,steps
read(100,*) dum,years
read(100,*) dum,outFreq
read(100,*) dum,soilExtra

close(100)

allocate(Lft(steps),Lrt(steps),Lwt(steps),&
     Rdecompt(steps),Rlittert(steps),Rsoilt(steps),&
     flux1t(steps),flux2t(steps),flux3t(steps),&
     flux4t(steps),flux5t(steps),&
     flux6t(steps),flux7t(steps),flux8t(steps),&
     flux9t(steps),flux10t(steps),&
     flux11t(steps),flux12t(steps),&
     flux13t(steps),flux14t(steps),&
     flux15t(steps),flux16t(steps),dCfoliage_dtt(steps),&
     dCroot_dtt(steps),dCwood_dtt(steps),&
     dCfast_dtt(steps),dCslow_dtt(steps),gluct(steps),&
     stat=sta)
if(sta/=0) then
   write(*,*) 'Problem with allocating respiration arrays. Program terminated'
   stop
endif

allocate(climate(((daysYear+1)*steps),5),stat=sta)
if(sta/=0) then
   write(*,*) 'Problem with allocating respiration arrays. Program terminated'
   stop
endif

! Read the climate file
open(200,file=climFile,status='old',iostat=sta)
if(sta/=0) then
   write(*,*) 'Problem reading the climate file. Program terminate'
   stop
endif

read(200,*) dummy
do i=1,(daysYear+1)*steps
   read(200,*) (climate(i,j),j=1,5)
enddo

! Read parameters file
open(300,file=paramFile,status='old',iostat=sta)
if(sta/=0) then
   write(*,*) 'Problem reading the parameter file. Program terminate'
   stop
endif

read(300,*) dum,Lf
read(300,*) dum,Lw
read(300,*) dum,Lr
read(300,*) dum,gluc
read(300,*) dum,CLf
read(300,*) dum,CLr
read(300,*) dum,CLw
read(300,*) dum,Cfast
read(300,*) dum,Cslow
read(300,*) dum,f_lignin
read(300,*) dum,kslow
read(300,*) dum,kfol
read(300,*) dum,kroot
read(300,*) dum,kwood
read(300,*) dum,kfast
read(300,*) dum,efficiency_decomposition
read(300,*) dum,q10
```

```
! Open output files
if(ans(soilExtra)) then

   open(2,file=trim(outFolder)//'soil_fluxes.csv',status='unknown')
   write(2,'(20(A7,","),A7)') 'time','Lf','Lr','Lw','gluc',&
        'flux1','flux2','flux3','flux4','flux5',&
        'flux6','flux7','flux8','flux9','flux10','flux11',&
        'flux12','flux13','flux14','flux15','flux16'

   open(3,file=trim(outFolder)//'differential.csv',status='unknown')
   write(3,*) 'time,dCfoliage_dt,dCroot_dt,dCwood_dt,dCfast_dt,dCslow_dt'

   open(4,file=trim(outFolder)//'massC.csv',status='unknown')
   write(4,*) 'time,preC,curC,delC,sumflux,checkdiff'

   open(7,file=trim(outFolder)//'rate.csv',status='unknown')
   write(7,*) 'time,rate,airtemp'
endif

open(5,file=trim(outFolder)//'stocks.csv',status='unknown')
write(5,*) 'time,CLf,CLr,CLw,Cfast,Cslow'
write(5,'(5(f0.5,","),f0.5)') 0.,CLf,CLr,CLw,&
     Cfast,Cslow

open(6,file=trim(outFolder)//'fluxes.csv',status='unknown')
write(6,*) 'time,Rs,Rl,Rd'

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

! Start decomposition

do k=1,years

   an_sumLf = 0.; an_sumLr = 0; an_sumLw = 0.; an_sumGluc = 0; an_sumFlux1 = 0.;
   an_sumFlux2 = 0.; an_sumFlux3 = 0.; an_sumFlux4 = 0.; an_sumFlux5 = 0.;
   an_sumFlux6 = 0.; an_sumFlux7 = 0.; an_sumFlux8 = 0.; an_sumFlux9 = 0.;
   an_sumFlux10 = 0.; an_sumFlux11 = 0.; an_sumFlux12 = 0.; an_sumFlux13 = 0.;
   an_sumFlux14 = 0.; an_sumFlux15 = 0.; an_sumFlux15 = 0.; an_sumFlux16 = 0.;
   an_sumdCfoliage_dt = 0.; an_sumdCroot_dt = 0.;
   an_sumdCwood_dt = 0.; an_sumdCfast_dt = 0.; an_sumdCslow_dt = 0.;
   an_sumRsoil = 0.; an_sumRlitter = 0.; an_sumRdecomp = 0.

   do j=0,daysYear

      do i=1,steps

         airtemp = climate(steps*j+i,2)

         ! Calculate some variables
         numsecs = 3600. * 24. / real(steps) ! Number of seconds per timestep
         ts_length = 24. / real(steps)       ! Variable used if simulation
                                             ! time step is other than an hour

         ! Rate from Q10 values for exponential response of decomposition to temperature
         rate = exp((log(q10)/10)*airtemp)

         ! Calculate the differential equations of the litter pools (flux gC m-2 timestep-1)
         dCfoliage_dt = Lf - ( rate * kfol * CLf * ts_length )
         dCroot_dt = Lr - ( rate * kroot * CLr * ts_length )
         dCwood_dt = Lw - ( rate * kwood * CLw * ts_length )

         dCfoliage_dtt(i) = dCfoliage_dt
         dCroot_dtt(i) = dCroot_dt
         dCwood_dtt(i) = dCwood_dt

         Lft(i) = Lf
         Lrt(i) = Lr
         Lwt(i) = Lw
```

```
! Previous pool status
preC = real(CLf,kind=8) + real(CLr,kind=8) + real(CLw,kind=8) + real(Cfast,kind=8) &
     + real(Cslow,kind=8)
! preC = real(Cfast,kind=8) + real(Cslow,kind=8)

! Calculate the differential equations for the carbon pools (flux gC m-2 timestep-1)
dCfast_dt = &
     (1-f_lignin) * efficiency_decomposition * rate * kfol * CLf * ts_length &
     + (1-f_lignin) * efficiency_decomposition * rate * kroot * CLr * ts_length &
     + (1-f_lignin) * efficiency_decomposition * rate * kwood * CLw * ts_length &
     - rate * kfast * Cfast * ts_length &
     + gluc
dCfast_dtt(i) = dCfast_dt

dCslow_dt = &
     f_lignin * efficiency_decomposition * rate * kfol * CLf * ts_length &
     + f_lignin * efficiency_decomposition * rate * kroot * CLr * ts_length &
     + f_lignin * efficiency_decomposition * rate * kwood * CLw * ts_length &
     + efficiency_decomposition * rate * kfast * Cfast * ts_length &
     - rate * kslow * Cslow * ts_length
dCslow_dtt(i) = dCslow_dt

dCO2_dt = &
     (1-efficiency_decomposition) * rate * kfol * CLf * ts_length &
     + (1-efficiency_decomposition) * rate * kroot * CLr * ts_length &
     + (1-efficiency_decomposition) * rate * kwood * CLw * ts_length &
     + rate * kslow * Cslow * ts_length &
     + (1-efficiency_decomposition) * rate * kfast * Cfast * ts_length

! Update carbon pools
Cfast = Cfast + dCfast_dt
Cslow = Cslow + dCslow_dt

! Update litter pools
CLf = CLf + dCfoliage_dt
CLw = CLw + dCwood_dt
CLr = CLr + dCroot_dt

! Current pool status
curC = real(CLf,kind=8) + real(CLr,kind=8) + real(CLw,kind=8) &
     + real(Cfast,kind=8) + real(Cslow,kind=8)
! curC = real(Cfast,kind=8) + real(Cslow,kind=8)

carbongain = real(Lf,kind=8) + real(Lr,kind=8) + real(Lw,kind=8) + real(gluc,kind=8)

carbonloss = real((1-efficiency_decomposition) * rate * kfol * CLf * ts_length,kind=8)&
     + real((1-efficiency_decomposition) * rate * kroot * CLr * ts_length,kind=8)&
     + real((1-efficiency_decomposition) * rate * kwood * CLw * ts_length,kind=8)&
     + real((1-efficiency_decomposition) * rate * kfast * Cfast * ts_length,kind=8)&
     + real(rate * kslow * Cslow * ts_length,kind=8)

sumflux = carbongain - carbonloss

delC = curC - preC
checkdiffC = delC - sumflux

! Respiration from decomposition of foliage, root and wood (gC m-2 timestep-1)
Rlitter = (1-efficiency_decomposition) * rate * kfol * CLf * ts_length &
     + (1-efficiency_decomposition) * rate * kroot * CLr * ts_length &
     + (1-efficiency_decomposition) * rate * kwood * CLw * ts_length

Rlittert(i) = Rlitter

! Respiration from decomposition (flux gC m-2 timestep-1)
Rdecomp =  rate * kslow * Cslow * ts_length &
     + (1-efficiency_decomposition) * rate * kfast * Cfast * ts_length

Rdecompt(i) = Rdecomp

! Soil respiration (gC m-2 s-1)
```

```
Rsoil = Rlitter + Rdecomp
Rsoilt(i) = Rlittert(i) + Rdecompt(i)

! Calculate some output variables
totalC = Cfast + Cslow

! Individual fluxes

! foliage output
flux1 = rate * kfol * CLf * ts_length
flux1t(i) = flux1
! root output
flux2 = rate * kroot * CLr * ts_length
flux2t(i) = flux2
! wood output
flux3 = rate * kwood * CLw * ts_length
flux3t(i) = flux3
! input foliage to fast
flux4 = (1-f_lignin) * efficiency_decomposition * rate * kfol * CLf * ts_length
flux4t(i) = flux4
! input root to fast
flux5 = (1-f_lignin) * efficiency_decomposition * rate * kroot * CLr * ts_length
flux5t(i) = flux5
! input wood to fast
flux6 = (1-f_lignin) * efficiency_decomposition * rate * kwood * CLw * ts_length
flux6t(i) = flux6
! input foliage to slow
flux7 = f_lignin * efficiency_decomposition * rate * kfol * CLf * ts_length
flux7t(i) = flux7
! input root to slow
flux8 = f_lignin * efficiency_decomposition * rate * kroot * CLr * ts_length
flux8t(i) = flux8
! input wood to slow
flux9 = f_lignin * efficiency_decomposition * rate * kwood * CLw * ts_length
flux9t(i) = flux9
! foliage respiration
flux10 = (1-efficiency_decomposition) * rate * kfol * CLf * ts_length
flux10t(i) = flux10
! root respiration
flux11 = (1-efficiency_decomposition) * rate * kroot * CLr * ts_length
flux11t(i) = flux11
! wood respiration
flux12 = (1-efficiency_decomposition) * rate * kwood * CLw * ts_length
flux12t(i) = flux12
! input fast to slow
flux13 = efficiency_decomposition * rate * kfast * Cfast* ts_length
flux13t(i) = flux13
! respiration fast
flux14 = (1-efficiency_decomposition) * rate * kfast * Cfast * ts_length
flux14t(i) = flux14
! respiration slow
flux15 = rate * kslow * Cslow * ts_length
flux15t(i) = flux15
! output fast
flux16 = rate * kfast * Cfast * ts_length
flux16t(i) = flux16

! $$$ Write subroutine outputs $$$ !

!-------------------------
! HOURLY OUTPUT
!-------------------------

! Individual fluxes
if( out(outFreq)== 3 ) then
   if( ans(soilExtra) ) write(2,'(20(f0.5,","),f0.5)') &
        (365.0*(k-1)+(j+(real(i)/real(steps)))),&
        Lf,Lr,Lw,gluc,flux1,flux2,flux3,flux4,flux5,&
        flux6,flux7,flux8,flux9,flux10,&
        flux11,flux12,flux13,flux14,flux15,flux16
```

```
      ! Differential equations
      if( ans(soilExtra) ) write(3,'(5(f0.5,","),f0.5)') &
           (365.0*(k-1)+(j+(real(i)/real(steps)))),&
           dCfoliage_dt,dCroot_dt,&
           dCwood_dt,dCfast_dt,dCslow_dt

      ! Mass balance
      if( ans(soilExtra) ) write(4,'(5(f0.5,","),f0.5)') &
           (365.0*(k-1)+(j+(real(i)/real(steps)))),&
           real(preC,kind=4),&
           real(curC,kind=4),real(delC,kind=4),real(sumflux,kind=4),real(checkdiffC,kind=4)

      ! Rate
      if( ans(soilExtra) ) write(7,'(2(f0.5,","),f0.5)') &
           (365.0*(k-1)+(j+(real(i)/real(steps)))),&
           rate,airtemp

      ! Stocks
      write(5,'(5(f0.5,","),f0.5)') &
           (365.0*(k-1)+(j+(real(i)/real(steps)))),&
           CLf,CLr,CLw,Cfast,Cslow

      ! Fluxes
      write(6,'(3(f0.5,","),f0.5)')&
           (365.0*(k-1)+(j+(real(i)/real(steps)))),Rsoil,Rlitter,Rdecomp
   endif
enddo
!-------------------------
! DAILY OUTPUT
!-------------------------

! Calculate daily sums
sumLf = sum(Lft);sumLr = sum(Lrt);sumLw = sum(Lwt)
sumFlux1 = sum(flux1t);sumFlux2 = sum(flux2t);sumFlux3 = sum(flux3t);
sumFlux4 = sum(flux4t);sumFlux5 = sum(flux5t);sumFlux6 = sum(flux6t);
sumFlux7 = sum(flux7t);sumFlux8 = sum(flux8t);sumFlux9 = sum(flux9t);
sumFlux10 = sum(flux10t);sumFlux11 = sum(flux11t);sumFlux12 = sum(flux12t);
sumFlux13 = sum(flux13t);sumFlux14 = sum(flux14t);sumFlux15 = sum(flux15t);
sumFlux15 = sum(flux15t);sumFlux16 = sum(flux16t)

sumdCfoliage_dt = sum(dCfoliage_dtt);sumdCroot_dt = sum(dCroot_dtt);
sumdCwood_dt = sum(dCwood_dtt);sumdCfast_dt = sum(dCfast_dtt);
sumdCslow_dt = sum(dCslow_dtt)

sumRsoil = sum(Rsoilt);sumRlitter = sum(Rlittert)
sumRdecomp = sum(Rdecompt)
an_sumRsoil = an_sumRsoil + sumRsoil
an_sumRlitter = an_sumRlitter + sumRlitter
an_sumRdecomp = an_sumRdecomp + sumRdecomp



sumLf = sum(Lft);sumLr = sum(Lrt);
sumLw = sum(Lwt);sumGluc = sum(gluct);
an_sumLf = an_sumLf + sumLf;
an_sumLr = an_sumLr + sumLr;an_sumLw = an_sumLw + sumLw
an_sumGluc = an_sumGluc + sumGluc

sumFlux1 = sum(flux1t);sumFlux2 = sum(flux2t);
sumFlux3 = sum(flux3t);sumFlux4 = sum(flux4t)
sumFlux5 = sum(flux5t);sumFlux6 = sum(flux6t);
sumFlux7 = sum(flux7t);sumFlux8 = sum(flux8t)
sumFlux9 = sum(flux9t);sumFlux10 = sum(flux10t)
sumFlux11 = sum(flux11t);sumFlux12 = sum(flux12t);
sumFlux13 = sum(flux13t);sumFlux14 = sum(flux14t)
sumFlux15 = sum(flux15t);sumFlux16 = sum(flux16t)

an_sumFlux1 = an_sumFlux1 + sumFlux1;an_sumFlux2 = an_sumFlux2 + sumFlux2
an_sumFlux3 = an_sumFlux3 + sumFlux3;an_sumFlux4 = an_sumFlux4 + sumFlux4
```

```
   an_sumFlux5 = an_sumFlux5 + sumFlux5;an_sumFlux6 = an_sumFlux6 + sumFlux6
   an_sumFlux7 = an_sumFlux7 + sumFlux7;an_sumFlux8 = an_sumFlux8 + sumFlux8
   an_sumFlux9 = an_sumFlux9 + sumFlux9;an_sumFlux10 = an_sumFlux10 + sumFlux10
   an_sumFlux11 = an_sumFlux11 + sumFlux11;an_sumFlux12 = an_sumFlux12 + sumFlux12
   an_sumFlux13 = an_sumFlux13 + sumFlux13;an_sumFlux14 = an_sumFlux14 + sumFlux14
   an_sumFlux15 = an_sumFlux15 + sumFlux15;an_sumFlux16 = an_sumFlux16 + sumFlux16

   sumdCfoliage_dt = sum(dCfoliage_dtt);sumdCroot_dt = sum(dCroot_dtt);
   sumdCwood_dt = sum(dCwood_dtt);sumdCfast_dt = sum(dCfast_dtt);
   sumdCslow_dt = sum(dCslow_dtt)
   an_sumdCfoliage_dt = an_sumdCfoliage_dt + sumDCfoliage_dt
   an_sumdCroot_dt = an_sumdCroot_dt + sumDCroot_dt
   an_sumdCwood_dt = an_sumdCwood_dt + sumDCwood_dt
   an_sumdCwood_dt = an_sumdCwood_dt + sumDCwood_dt
   an_sumdCfast_dt = an_sumdCfast_dt + sumDCfast_dt
   an_sumdCslow_dt = an_sumdCslow_dt + sumDCslow_dt

   sumRsoil = sum(Rsoilt)
   sumRlitter = sum(Rlittert)
   sumRdecomp = sum(Rdecompt)
   an_sumRsoil = an_sumRsoil + sumRsoil
   an_sumRlitter = an_sumRlitter + sumRlitter
   an_sumRdecomp = an_sumRdecomp + sumRdecomp


   ! Write output
   if( out(outFreq)==2 ) then
      ! Individual fluxes
      if( ans(soilExtra) ) write(2,'(20(f0.5,","),f0.5)') &
           (365.0*(k-1)+(j+(real(steps-1)/real(steps)))),sumLf,sumLr,sumLw,&
           sumGluc,sumFlux1,sumFlux2,sumFlux3,sumFlux4,sumFlux5,&
           sumFlux6,sumFlux7,sumFlux8,sumFlux9,sumFlux10,&
           sumFlux11,sumFlux12,sumFlux13,sumFlux14,sumFlux15,sumFlux16

      ! Differential equations
      if( ans(soilExtra) ) write(3,'(5(f0.5,","),f0.5)') &
           (365.0*(k-1)+(j+(real(steps-1)/real(steps)))),&
           sumdCfoliage_dt,sumdCroot_dt,sumdCwood_dt,sumdCfast_dt,sumdCslow_dt

      ! Mass balance
      if( ans(soilExtra) )  write(4,'(5(f0.5,","),f0.5)')&
           (365.0*(k-1)+(j+(real(steps-1)/real(steps)))),real(preC,kind=4),&
           real(curC,kind=4),real(delC,kind=4),real(sumflux,kind=4),real(checkdiffC,kind=4)

      ! Rate
      if( ans(soilExtra) ) write(7,'(2(f0.5,","),f0.5)')&
           (365.0*(k-1)+(j+(real(steps-1)/real(steps)))),rate,airtemp

      ! Stocks
      write(5,'(5(f0.5,","),f0.5)')&
           (365.0*(k-1)+(j+(real(steps-1)/real(steps)))),CLf,CLr,CLw,Cfast,Cslow

      ! Fluxes
      write(6,'(3(f0.5,","),f0.5)') &
           (365.0*(k-1)+(j+(real(steps-1)/real(steps)))),sumRsoil,sumRlitter,sumRdecomp
   endif
   !-------------------------
enddo

!-------------------------
! ANNUAL OUTPUT
!-------------------------
! Calculate annual averages
avgLf = an_sumLf / daysYear
avgLr = an_sumLr / daysYear
avgLw = an_sumLw / daysYear
avgGluc = an_sumGluc / daysYear

avgFlux1 = an_sumFlux1 / daysYear;avgFlux2 = an_sumFlux2 / daysYear;
avgFlux3 = an_sumFlux3 / daysYear;avgFlux4 = an_sumFlux4 / daysYear;
```

```
      avgFlux5 = an_sumFlux5 / daysYear;avgFlux6 = an_sumFlux6 / daysYear;
      avgFlux7 = an_sumFlux7 / daysYear;avgFlux8 = an_sumFlux8 / daysYear;
      avgFlux9 = an_sumFlux9 / daysYear;avgFlux10 = an_sumFlux10 / daysYear;
      avgFlux11 = an_sumFlux11 / daysYear;avgFlux12 = an_sumFlux12 / daysYear;
      avgFlux13 = an_sumFlux13 / daysYear;avgFlux14 = an_sumFlux14 / daysYear;
      avgFlux15 = an_sumFlux15 / daysYear;avgFlux16 = an_sumFlux16 / daysYear;

      avgdCfoliage_dt = an_sumdCfoliage_dt / daysYear
      avgdCroot_dt = an_sumdCroot_dt / daysYear
      avgdCwood_dt = an_sumdCwood_dt / daysYear
      avgdCwood_dt = an_sumdCwood_dt  / daysYear
      avgdCfast_dt = an_sumdCfast_dt / daysYear
      avgdCslow_dt = an_sumdCslow_dt  / daysYear

      avgRsoil = an_sumRsoil / daysYear
      avgRlitter = an_sumRlitter / daysYear
      avgRdecomp = an_sumRdecomp / daysYear

      ! Now write annual output
      if( out(outFreq)==1 ) then
         ! Individual fluxes
         if( ans(soilExtra) ) write(2,'(20(f0.5,","),f0.5)') &
              (365.0*(k-1)+(j+(real(steps-1)/real(steps)))),avgLf,avgLr,avgLw,&
              avgGluc,avgFlux1,avgFlux2,avgFlux3,avgFlux4,avgFlux5,&
              avgFlux6,avgFlux7,avgFlux8,avgFlux9,avgFlux10,&
              avgFlux11,avgFlux12,avgFlux13,avgFlux14,avgFlux15,avgFlux16

         ! Differential equations
         if( ans(soilExtra) ) write(3,'(5(f0.5,","),f0.5)') &
              (365.0*(k-1)+(j+(real(steps-1)/real(steps)))),&
              avgdCfoliage_dt,avgdCroot_dt,avgdCwood_dt,avgdCfast_dt,avgdCslow_dt

         ! Mass balance
         if( ans(soilExtra) )  write(4,'(5(f0.5,","),f0.5)') &
              (365.0*(k-1)+(j+(real(steps-1)/real(steps)))),real(preC,kind=4),&
              real(curC,kind=4),real(delC,kind=4),real(sumflux,kind=4),real(checkdiffC,kind=4)

         ! Rate
         if( ans(soilExtra) ) write(7,'(2(f0.5,","),f0.5)') &
              (365.0*(k-1)+(j+(real(steps-1)/real(steps)))),rate,airtemp

         ! Stocks
         write(5,'(5(f0.5,","),f0.5)') &
              (365.0*(k-1)+(j+(real(steps-1)/real(steps)))),CLf,CLr,CLw,Cfast,Cslow

         ! Fluxes
         write(6,'(3(f0.5,","),f0.5)') &
              (365.0*(k-1)+(j+(real(steps-1)/real(steps)))),avgRsoil,avgRlitter,avgRdecomp

      endif
      !-------------------------
   enddo

   ! Close all output files
   if(ans(soilExtra)) then
      close(2)
      close(3)
      close(4)
      close(7)
   endif
   close(5)
   close(6)

   !--------------------------------------
   ! Containing subroutines and functions
   !--------------------------------------


contains
   !--------------------------------------
```

```
  logical function ans(in)
    implicit none
    character(len=3),intent(in):: in
    if(in=='Yes'.or.in=='YES'.or.in=='yes') then
       ans = .true.
    else if(in=='No'.or.in=='NO'.or.in=='no') then
       ans = .false.
    endif
  end function ans
  !-------------------------------------
  integer function out(in)
    implicit none
    character(len=*),intent(in):: in
    if(in=='Annualy'.or.in=='ANNUALY'.or.in=='annualy') then
       out = 1
    else if(in=='Daily'.or.in=='DAILY'.or.in=='daily') then
       out = 2
    else if(in=='Hourly'.or.in=='HOURLY'.or.in=='hourly') then
       out = 3
    endif
  end function out
  !-------------------------------------

end program decochem
```

DecoBio v1.0 source code

```
program decobio

  implicit none

  character(len=80) :: inFile,climFile,paramFile,outFolder

  integer :: steps

  real :: airtemp,Rlitter,Rdecomp,Rsoil,&
       kslow,kmc,initial_mact,&
       microbial_activity,inhibition_constant,&
       efficiency_substrate_uptake,maintenance_coefficient,&
       efficiency_decomposition_first,&
       efficiency_decomposition_mic,&
       f_lignin,microbial_death,&
       md_max,md_i,dmact,dCmicrobes_dt,&
       rate,q10,bulkDen,depth,&
       conversion,kfol,kroot,kwood,&
       CLf,CLr,CLw,Lf,Lr,Lw,&
       gluc,Cslow,Cfast,Cmicrobes
  real :: ts_length,dCslow_dt,dCfast_dt,&
       dCO2_dt,dCfoliage_dt,dCroot_dt,&
       dCwood_dt,totalC
  integer :: i,j,k
  real :: flux1,flux2,flux3,flux4,flux5,flux6,flux7,flux8,flux9,flux10,flux11,&
       flux12,flux13,flux14,flux15,flux16,flux17,flux18,flux19,flux20,flux21,&
       flux22
  real :: sumFlux1,sumFlux2,sumFlux3,sumFlux4,&
       sumFlux5,sumFlux6,sumFlux7,sumFlux8,&
       sumFlux9,sumFlux10,sumFlux11,sumFlux12,&
       sumFlux13,sumFlux14,sumFlux15,sumFlux16,&
       sumFlux17,sumFlux18,sumFlux19,sumFlux20,&
       sumFlux21,sumFlux22,sumdCfoliage_dt,&
       sumdCroot_dt,sumdCwood_dt,sumdCfast_dt,sumdCslow_dt,&
       sumdCmicrobes_dt,sumdmact,sumLf,sumLr,sumLw,sumGluc,&
       sumRlitter,sumRdecomp,sumRsoil
  real :: an_sumFlux1,an_sumFlux2,an_sumFlux3,an_sumFlux4,&
       an_sumFlux5,an_sumFlux6,an_sumFlux7,an_sumFlux8,&
```

```
     an_sumFlux9,an_sumFlux10,an_sumFlux11,an_sumFlux12,&
     an_sumFlux13,an_sumFlux14,an_sumFlux15,an_sumFlux16,&
     an_sumFlux17,an_sumFlux18,an_sumFlux19,an_sumFlux20,&
     an_sumFlux21,an_sumFlux22,an_sumdCfoliage_dt,&
     an_sumdCroot_dt,an_sumdCwood_dt,an_sumdCfast_dt,an_sumdCslow_dt,&
     an_sumdCmicrobes_dt,an_sumdmact,an_sumLf,an_sumLr,an_sumLw,an_sumGluc,&
     an_sumRlitter,an_sumRdecomp,an_sumRsoil
real :: avgFlux1,avgFlux2,avgFlux3,avgFlux4,&
     avgFlux5,avgFlux6,avgFlux7,avgFlux8,&
     avgFlux9,avgFlux10,avgFlux11,avgFlux12,&
     avgFlux13,avgFlux14,avgFlux15,avgFlux16,&
     avgFlux17,avgFlux18,avgFlux19,avgFlux20,&
     avgFlux21,avgFlux22,avgdCfoliage_dt,&
     avgdCroot_dt,avgdCwood_dt,avgdCfast_dt,avgdCslow_dt,&
     avgdCmicrobes_dt,avgdmact,avgLf,avgLr,&
     avgLw,avgGluc,avgRlitter,avgRdecomp,avgRsoil
real(kind=8) :: preC,curC,carbongain,carbonloss,sumflux,delC,checkdiffC
real,allocatable :: Rsoilt(:),Rlittert(:),Rdecompt(:),climate(:,:),&
     Lft(:),Lrt(:),Lwt(:),gluct(:),&
     flux1t(:),flux2t(:),flux3t(:),flux4t(:),&
     flux5t(:),flux6t(:),flux7t(:),flux8t(:),&
     flux9t(:),flux10t(:),flux11t(:),flux12t(:),&
     flux13t(:),flux14t(:),flux15t(:),&
     flux16t(:),flux17t(:),flux18t(:),&
     flux19t(:),flux20t(:),flux21t(:),&
     flux22t(:),dCfoliage_dtt(:),&
     dCroot_dtt(:),dCwood_dtt(:),&
     dCfast_dtt(:),dCslow_dtt(:),&
     dCmicrobes_dtt(:),dmactt(:)
integer :: daysYear,years
integer :: stat,sta
character(len=3) :: soilExtra
character(len=10) :: outFreq
character(len=20) :: dum,dummy
real,parameter :: seconds_per_step=3600

! ---- START  ---- !

stat=0
call get_command_argument( 1, inFile )

! Check if a file argument was given
open(100,file=inFile,status='old',iostat=stat)
if(stat/=0) then
   write(*,*) 'No input file entered.'
   stop
endif

! Read the input file
read(100,*) dum,climFile
read(100,*) dum,paramFile
read(100,*) dum,outFolder
read(100,*) dum,daysYear
read(100,*) dum,steps
read(100,*) dum,years
read(100,*) dum,outFreq
read(100,*) dum,soilExtra
close(100)

allocate(Rsoilt(steps),Rlittert(steps),Rdecompt(steps),&
     flux1t(steps),flux2t(steps),flux3t(steps),flux4t(steps),&
     flux5t(steps),flux6t(steps),flux7t(steps),&
     flux8t(steps),flux9t(steps),flux10t(steps),&
     flux11t(steps),flux12t(steps),flux13t(steps),&
     flux14t(steps),flux15t(steps),flux16t(steps),&
     flux17t(steps),flux18t(steps),flux19t(steps),&
     flux20t(steps),flux21t(steps), &
     flux22t(steps),dCfoliage_dtt(steps),&
     dCroot_dtt(steps),dCwood_dtt(steps),&
     dCfast_dtt(steps),dCslow_dtt(steps),&
```

```
     dCmicrobes_dtt(steps),dmactt(steps),&
     Lft(steps),Lrt(steps),Lwt(steps),gluct(steps),&
     stat=sta)
if(sta/=0) then
   write(*,*) 'Problem with allocating respiration arrays. Program terminated'
   stop
endif

allocate(climate(((daysYear+1)*steps),5),stat=sta)
if(sta/=0) then
   write(*,*) 'Problem with allocating respiration arrays. Program terminated'
   stop
endif

! Read the climate file
open(200,file=climFile,status='old',iostat=sta)
if(sta/=0) then
   write(*,*) 'Problem reading the climate file. Program terminate'
   stop
endif

read(200,*) dummy
do i=1,(daysYear+1)*steps
   read(200,*) (climate(i,j),j=1,5)
enddo

! Read parameters file
open(300,file=paramFile,status='old',iostat=sta)
if(sta/=0) then
   write(*,*) 'Problem reading the parameter file. Program terminate'
   stop
endif

read(300,*) dum,Lf
read(300,*) dum,Lw
read(300,*) dum,Lr
read(300,*) dum,gluc
read(300,*) dum,CLf
read(300,*) dum,CLr
read(300,*) dum,CLw
read(300,*) dum,Cfast
read(300,*) dum,Cslow
read(300,*) dum,Cmicrobes
read(300,*) dum,f_lignin
read(300,*) dum,kfol
read(300,*) dum,kroot
read(300,*) dum,kwood
read(300,*) dum,kslow
read(300,*) dum,kmc
read(300,*) dum,initial_mact
read(300,*) dum,inhibition_constant
read(300,*) dum,efficiency_decomposition_first
read(300,*) dum,efficiency_decomposition_mic
read(300,*) dum,efficiency_substrate_uptake
read(300,*) dum,maintenance_coefficient
read(300,*) dum,md_max
read(300,*) dum,md_i
read(300,*) dum,q10
read(300,*) dum,bulkDen
read(300,*) dum,depth

close(300)

! Calculate the conversion factor
conversion = 0.001*bulkDen*depth

! Convert from gC m-2 to mgC g soil
Lf = Lf / conversion
Lr = Lr / conversion
Lw = Lw / conversion
```

```
gluc = gluc / conversion
CLf = CLf / conversion
CLr = CLr / conversion
CLw = CLw / conversion
Cfast = Cfast / conversion
Cslow = Cslow / conversion
Cmicrobes = Cmicrobes / conversion
kmc = kmc * conversion ! This is g mgC-1 day-1
inhibition_constant = inhibition_constant / conversion
md_i = md_i * conversion ! This is g mgC-1

! Open output files
if(ans(soilExtra)) then
   open(2,file=trim(outFolder)//'soil_fluxes.csv',status='unknown')
   write(2,'(26(A10,","),A10)') 'time','Lf','Lr','Lw','gluc',&
        'flux1','flux2','flux3','flux4','flux5',&
        'flux6','flux7','flux8','flux9','flux10','flux11','flux12','flux13',&
        'flux14','flux15','flux16','flux17','flux18','flux19','flux20','flux21','flux22'

   open(3,file=trim(outFolder)//'differential.csv',status='unknown')
   write(3,*) 'time,dCfoliage_dt,dCroot_dt,dCwood_dt,dCfast_dt,dCslow_dt,dCmicrobes_dt'

   open(4,file=trim(outFolder)//'massC.csv',status='unknown')
   write(4,*) 'time,preC,curC,delC,sumflux,checkdiff'

   open(7,file=trim(outFolder)//'rate.csv',status='unknown')
   write(7,*) 'time,rate,airtemp'

endif

open(1,file=trim(outFolder)//'mact.csv',status='unknown')
write(1,*) 'time,dmact_dt,mact,Cmicrobes,md'

open(5,file=trim(outFolder)//'stocks.csv',status='unknown')
write(5,*) 'time,CLf,CLr,CLw,Cfast,Cslow,Cmicrobes'
write(5,'(6(f0.8,","),f0.8)') 0.,CLf*conversion,CLr*conversion,CLw*conversion,&
     Cfast*conversion,Cslow*conversion,Cmicrobes*conversion

open(6,file=trim(outFolder)//'fluxes.csv',status='unknown')
write(6,*) 'time,Rs,Rl,Rd'

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

microbial_activity = initial_mact

! Start decomposition

do k=1,years

   an_sumLf=0.;an_sumLr=0;an_sumLw=0.;an_sumGluc=0;an_sumFlux1=0.;
   an_sumFlux2=0.;an_sumFlux3=0.;an_sumFlux4=0.;an_sumFlux5=0.;
   an_sumFlux6=0.;an_sumFlux7=0.;an_sumFlux8=0.;an_sumFlux9=0.;
   an_sumFlux10=0.;an_sumFlux11=0.;an_sumFlux12=0.;an_sumFlux13=0.;
   an_sumFlux14=0.;an_sumFlux15=0.;an_sumFlux15=0.;an_sumFlux16=0.;
   an_sumFlux17=0.;an_sumFlux18=0.;an_sumFlux19=0.;an_sumFlux20=0.;
   an_sumFlux21=0.;an_sumFlux22=0.;an_sumdCfoliage_dt=0.;an_sumdCroot_dt=0.;
   an_sumdCwood_dt=0.;an_sumdCfast_dt=0.;an_sumdCslow_dt=0.;
   an_sumdCmicrobes_dt=0.;an_sumdmact=0.;an_sumRsoil=0.;an_sumRlitter=0.;an_sumRdecomp=0.

   do j=0,daysYear

      do i=1,steps

         airtemp = climate(steps*j+i,2)

         ! Rate from Q10 values for exponential response of decomposition to temperature
         rate = exp((log(q10)/10)*airtemp)

         ! Calculate some variables
         ts_length = 24. / real(steps)   ! Variable used if simulation
```

```
                                !time step is other than an hour

! Calculate the dynamic parameter of microbial activity
! in relation to soluble, cellulose and ligning
dmact = rate * kmc * Cfast * &
      ( ( Cfast  / ( Cfast + inhibition_constant ) ) - microbial_activity )
microbial_activity = microbial_activity + dmact
dmactt(i) = dmact

flux12 = min(Cslow, microbial_activity * kslow * Cmicrobes * ts_length)

! Calculate the microbial death rate
microbial_death = md_max / (1 + md_i * Cfast)

dCfoliage_dt = Lf - ( kfol * rate * CLf * ts_length )
dCroot_dt = Lr - ( kroot * rate * CLr * ts_length )
dCwood_dt = Lw - ( kwood * rate * CLw * ts_length )

dCfoliage_dtt(i) = dCfoliage_dt
dCroot_dtt(i) = dCroot_dt
dCwood_dtt(i) = dCwood_dt

Lft(i) = Lf
Lrt(i) = Lr
Lwt(i) = Lw

! Previous pool status
preC = real(CLf,kind=8) + real(CLr,kind=8) + real(CLw,kind=8) + real(Cfast,kind=8) &
     + real(Cslow,kind=8) + real(Cmicrobes,kind=8)

! Calculate the differential equations for the carbon pools (flux gC m-2 timestep-1)
dCfast_dt = &
     (1-f_lignin) * efficiency_decomposition_first * rate * kfol * CLf * ts_length &
     + (1-f_lignin) * efficiency_decomposition_first * rate * kroot * CLr * ts_length &
     + (1-f_lignin) * efficiency_decomposition_first * rate * kwood * CLw * ts_length &
     + efficiency_decomposition_mic * flux12 &
     - kmc * Cfast * Cmicrobes * microbial_activity * ts_length &
     + gluc
dCfast_dtt(i) = dCfast_dt
gluct(i) = gluc


dCslow_dt = &
     f_lignin * efficiency_decomposition_first * rate * kfol * CLf * ts_length &
     + f_lignin * efficiency_decomposition_first * rate * kroot * CLr * ts_length &
     + f_lignin * efficiency_decomposition_first * rate * kwood * CLw * ts_length &
     - flux12 &
     + microbial_death * microbial_activity * Cmicrobes * ts_length
dCslow_dtt(i) = dCslow_dt

dCmicrobes_dt = &
     efficiency_substrate_uptake * kmc * Cfast &
     * Cmicrobes * microbial_activity * ts_length &
     - maintenance_coefficient * microbial_activity * Cmicrobes * ts_length &
     - microbial_death * microbial_activity * Cmicrobes *  ts_length
dCmicrobes_dtt(i) = dCmicrobes_dt

dCO2_dt = &
     (1-efficiency_decomposition_first) * rate * kfol * CLf * ts_length &
     + (1-efficiency_decomposition_first) * rate * kroot * CLr * ts_length &
     + (1-efficiency_decomposition_first) * rate * kwood * CLw * ts_length &
     + (1-efficiency_decomposition_mic) * flux12 &
     + (1-efficiency_substrate_uptake) * kmc * Cfast * Cmicrobes &
     * microbial_activity * ts_length &
     + maintenance_coefficient * microbial_activity * Cmicrobes * ts_length

! Update carbon pools
Cfast = Cfast + dCfast_dt
Cslow = Cslow + dCslow_dt
Cmicrobes = Cmicrobes + dCmicrobes_dt
```

```
! Update litter pools
CLf = CLf + dCfoliage_dt
CLw = CLw + dCwood_dt
CLr = CLr + dCroot_dt

! Current pool status
curC = real(CLf,kind=8) + real(CLr,kind=8) + real(CLw,kind=8) &
     + real(Cfast,kind=8) + real(Cslow,kind=8) + real(Cmicrobes,kind=8)

carbongain = real(Lf,kind=8) + real(Lr,kind=8) + real(Lw,kind=8) + real(gluc,kind=8)

carbonloss = &
     real(((1-efficiency_decomposition_first) * rate * kfol * CLf * ts_length), kind=8) &
     + real(((1-efficiency_decomposition_first) * rate * kroot * CLr * ts_length),kind=8) &
     + real(((1-efficiency_decomposition_first) * rate * kwood * CLw * ts_length),kind=8) &
     + real((1-efficiency_decomposition_mic) * flux12,kind=8) &
     + real((1-efficiency_substrate_uptake) * kmc * Cfast &
     * Cmicrobes * microbial_activity * ts_length,kind=8) &
     + real(maintenance_coefficient * microbial_activity * Cmicrobes * ts_length,kind=8)

sumflux = carbongain - carbonloss

delC = curC - preC
checkdiffC = delC - sumflux

! Sum the respiration decomposition of foliage, root and wood (gC m-2 timestep-1)
Rlitter = (1-efficiency_decomposition_first) * rate * kfol * CLf * ts_length &
        + (1-efficiency_decomposition_first) * rate * kroot * CLr * ts_length &
        + (1-efficiency_decomposition_first) * rate * kwood * CLw * ts_length
Rlittert(i) = Rlitter

! Heterotrophic respiration (flux gC m-2 timestep-1)
Rdecomp = (1-efficiency_decomposition_mic) * flux12 &
     + (1-efficiency_substrate_uptake) * kmc * Cfast &
     * Cmicrobes * microbial_activity * ts_length &
     + maintenance_coefficient * microbial_activity * Cmicrobes * ts_length
Rdecompt(i) = Rdecomp

Rsoil = Rdecomp + Rlitter
Rsoilt(i) = Rdecompt(i) + Rlittert(i)

! Calculate some output variables
totalC = Cfast + Cslow

! Calcualte individual fluxes
! foliage output
flux1 = rate * kfol * CLf * ts_length
flux1t(i) = flux1
! root output
flux2 = rate * kroot * CLr * ts_length
flux2t(i) = flux2
! wood output
flux3 = rate * kwood * CLw * ts_length
flux3t(i) = flux3
! input foliage to fast
flux4 = (1-f_lignin) * efficiency_decomposition_first * rate * kfol * CLf * ts_length
flux4t(i) = flux4
! input root to fast
flux5 = (1-f_lignin) * efficiency_decomposition_first * rate * kroot * CLr * ts_length
flux5t(i) = flux5
! input wood to fast
flux6 = (1-f_lignin) * efficiency_decomposition_first * rate * kwood * CLw * ts_length
flux6t(i) = flux6
! input slow to fast
flux7 = efficiency_decomposition_mic * flux12
flux7t(i) = flux7
! output fast
flux8 = kmc * Cfast * microbial_activity * Cmicrobes * ts_length
flux8t(i) = flux8
```

```
! input foliage to slow
flux9 = f_lignin * efficiency_decomposition_first * rate * kfol * CLf * ts_length
flux9t(i) = flux9
! input root to slow
flux10 = f_lignin * efficiency_decomposition_first * rate * kroot * CLr * ts_length
flux10t(i) = flux10
! input wood to slow
flux11 = f_lignin * efficiency_decomposition_first * rate * kwood * CLw * ts_length
flux11t(i) = flux11
! output slow
flux12 = min(Cslow,microbial_activity * kslow * Cmicrobes * ts_length)
flux12t(i) = flux12
! input microbes to slow (microbial death)
flux13 = microbial_death * microbial_activity * Cmicrobes * ts_length
flux13t(i) = flux13
! input fast to microbes
flux14 = efficiency_substrate_uptake * kmc * Cfast &
       * microbial_activity * Cmicrobes * ts_length
flux14t(i) = flux14
! output microbes (maintentance respiration)
flux15 = maintenance_coefficient * microbial_activity * Cmicrobes * ts_length
flux15t(i) = flux15
! output microbes to slow (microbial death)
flux16 = microbial_death * microbial_activity * Cmicrobes * ts_length
flux16t(i) = flux16
! foliage respiration
flux17 = (1-efficiency_decomposition_first) * rate * kfol * CLf * ts_length
flux17t(i) = flux17
! root respiration
flux18 = (1-efficiency_decomposition_first) * rate * kroot * CLr * ts_length
flux18t(i) = flux18
! wood respiration
flux19 = (1-efficiency_decomposition_first) * rate * kwood * CLw * ts_length
flux19t(i) = flux19
! respiration slow
flux20 = (1-efficiency_decomposition_mic) * flux12
flux20t(i) = flux20
! growth respiration
flux21 = (1-efficiency_substrate_uptake) * kmc * Cfast &
       * microbial_activity * Cmicrobes * ts_length
flux21t(i) = flux21
! maintenance respiration
flux22 = maintenance_coefficient * microbial_activity * Cmicrobes * ts_length
flux22t(i) = flux22

! $$$ Write subroutine outputs $$$ !

!-------------------------
! HOURLY OUTPUT
!-------------------------
! Individual fluxes
if( out(outFreq)== 3 ) then
   if( ans(soilExtra) ) write(2,'(26(f0.8,","),f0.8)') &
        (365.0*(k-1)+(j+(real(i)/real(steps)))),&
        Lf*conversion,Lr*conversion,&
        Lf*conversion,gluc*conversion,&
        flux1*conversion,flux2*conversion,&
        flux3*conversion,flux4*conversion,flux5*conversion,&
        flux6*conversion,flux7*conversion,flux8*conversion,&
        flux9*conversion,flux10*conversion,&
        flux11*conversion,flux12*conversion,flux13*conversion,&
        flux14*conversion,flux15*conversion,&
        flux16*conversion,flux17*conversion,&
        flux18*conversion,flux19*conversion,&
        flux20*conversion,flux21*conversion,flux22*conversion !gC m-2 h-1

   ! Differential equations
   if( ans(soilExtra) ) write(3,'(6(f0.8,","),f0.8)') &
        (365.0*(k-1)+(j+(real(i)/real(steps)))),&
        dCfoliage_dt*conversion,dCroot_dt*conversion,dCwood_dt*conversion,&
```

```
            dCfast_dt*conversion,dCslow_dt*conversion,dCmicrobes_dt*conversion

      ! Mass balance
      if( ans(soilExtra) )  write(4,'(5(f0.8,","),f0.8)') &
            (365.0*(k-1)+(j+(real(i)/real(steps)))),&
            real(preC,kind=4),&
            real(curC,kind=4),real(delC,kind=4),real(sumflux,kind=4),real(checkdiffC,kind=4)

      ! Microbial activity
      write(1,'(4(f0.8,","),f0.8)') &
            (365.0*(k-1)+(j+(real(i)/real(steps)))),dmact,microbial_activity,&
            Cmicrobes*conversion,microbial_death

      ! Stocks
      write(5,'(6(f0.8,","),f0.8)') &
            (365.0*(k-1)+(j+(real(i)/real(steps)))),CLf*conversion, &
            CLr*conversion, CLw*conversion,Cfast*conversion,&
            Cslow*conversion,Cmicrobes*conversion

      ! Fluxes
      write(6,'(3(f0.8,","),f0.8)') &
            (365.0*(k-1)+(j+(real(i)/real(steps)))),Rsoil*conversion,&
            Rlitter*conversion, Rdecomp*conversion

      ! Rate
      if( ans(soilExtra) ) write(7,'(2(f0.8,","),f0.8)') &
            (365.0*(k-1)+(j+(real(i)/real(steps)))),rate,airtemp

   endif
enddo
!-------------------------
! DAILY OUTPUT
!-------------------------

! Calculate daily sums
sumLf=sum(Lft*conversion);sumLr=sum(Lrt*conversion)
sumLw=sum(Lwt*conversion);sumGluc=sum(gluct*conversion);
an_sumLf=an_sumLf+sumLf;an_sumLr=an_sumLr+sumLr
an_sumLw=an_sumLw+sumLw;an_sumGluc=an_sumGluc+sumGluc;

sumFlux1=sum(flux1t*conversion);sumFlux2=sum(flux2t*conversion)
sumFlux3=sum(flux3t*conversion);sumFlux4=sum(flux4t*conversion)
sumFlux5=sum(flux5t*conversion);sumFlux6=sum(flux6t*conversion)
sumFlux7=sum(flux7t*conversion);sumFlux8=sum(flux8t*conversion)
sumFlux9=sum(flux9t*conversion);sumFlux10=sum(flux10t*conversion)
sumFlux11=sum(flux11t*conversion);sumFlux12=sum(flux12t*conversion);
sumFlux13=sum(flux13t*conversion);sumFlux14=sum(flux14t*conversion);
sumFlux15=sum(flux15t*conversion);sumFlux16=sum(flux16t*conversion);
sumFlux17=sum(flux17t*conversion);sumFlux18=sum(flux18t*conversion);
sumFlux19=sum(flux19t*conversion);sumFlux20=sum(flux20t*conversion);
sumFlux21=sum(flux21t*conversion);sumFlux22=sum(flux22t*conversion)

an_sumFlux1=an_sumFlux1+sumFlux1;an_sumFlux2=an_sumFlux2+sumFlux2;
an_sumFlux3=an_sumFlux3+sumFlux3;an_sumFlux4=an_sumFlux4+sumFlux4;
an_sumFlux5=an_sumFlux5+sumFlux5;an_sumFlux6=an_sumFlux6+sumFlux6;
an_sumFlux7=an_sumFlux7+sumFlux7;an_sumFlux8=an_sumFlux8+sumFlux8;
an_sumFlux9=an_sumFlux9+sumFlux9;an_sumFlux10=an_sumFlux10+sumFlux10;
an_sumFlux11=an_sumFlux11+sumFlux11;an_sumFlux12=an_sumFlux12+sumFlux12;
an_sumFlux13=an_sumFlux13+sumFlux13;an_sumFlux14=an_sumFlux14+sumFlux14;
an_sumFlux15=an_sumFlux15+sumFlux15;an_sumFlux16=an_sumFlux16+sumFlux16;
an_sumFlux17=an_sumFlux17+sumFlux17;an_sumFlux18=an_sumFlux18+sumFlux18;
an_sumFlux19=an_sumFlux19+sumFlux19;an_sumFlux20=an_sumFlux20+sumFlux20;
an_sumFlux21=an_sumFlux21+sumFlux21;an_sumFlux22=an_sumFlux22+sumFlux22;

sumdCfoliage_dt=sum(dCfoliage_dtt*conversion)
sumdCroot_dt=sum(dCroot_dtt*conversion)
sumdCwood_dt=sum(dCwood_dtt*conversion)
sumdCfast_dt=sum(dCfast_dtt*conversion)
sumdCslow_dt=sum(dCslow_dtt*conversion)
sumdCmicrobes_dt=sum(dCmicrobes_dtt*conversion)
```

```
   an_sumdCfoliage_dt=an_sumdCfoliage_dt+sumDCfoliage_dt
   an_sumdCroot_dt=an_sumdCroot_dt+sumDCroot_dt
   an_sumdCwood_dt=an_sumdCwood_dt+sumDCwood_dt
   an_sumdCwood_dt=an_sumdCwood_dt+sumDCwood_dt
   an_sumdCfast_dt=an_sumdCfast_dt+sumDCfast_dt
   an_sumdCslow_dt=an_sumdCslow_dt+sumDCslow_dt

   sumdmact=sum(dmactt)
   an_sumdmact=an_sumdmact+sumdmact

   sumRsoil=sum(Rsoilt*conversion);sumRlitter=sum(Rlittert*conversion)
   sumRdecomp=sum(Rdecompt*conversion)
   an_sumRsoil=an_sumRsoil+sumRsoil;an_sumRlitter=an_sumRlitter+sumRlitter
   an_sumRdecomp=an_sumRdecomp+sumRdecomp

   ! Now write the outputs
   if( out(outFreq)==2 ) then

      ! Individual fluxes
      if( ans(soilExtra) ) write(2,'(26(f0.8,","),f0.8)')&
           (365.0*(k-1)+(j+(real(steps-1)/real(steps)))),&
           sumLf,sumLr,sumLw,sumGluc,&
           sumFlux1,sumFlux2,sumFlux3,&
           sumFlux4,sumFlux5,sumFlux6,sumFlux7,sumFlux8,&
           sumFlux9,sumFlux10,sumFlux11,sumFlux12,sumFlux13,&
           sumFlux14,sumFlux15,sumFlux16,sumFlux17,sumFlux18,&
           sumFlux19,sumFlux20,sumFlux21,sumFlux22

      ! Differential equations
      if( ans(soilExtra) ) write(3,'(6(f0.8,","),f0.8)') &
           (365.0*(k-1)+(j+(real(steps-1)/real(steps)))),&
           sumdCfoliage_dt,sumdCroot_dt,sumdCwood_dt,&
           sumdCfast_dt,sumdCslow_dt,sumdCmicrobes_dt

      ! Mass balance
      if( ans(soilExtra) ) write(4,'(5(f0.8,","),f0.8)') &
           (365.0*(k-1)+(j+(real(steps-1)/real(steps)))),real(preC,kind=4),&
           real(curC,kind=4),real(delC,kind=4),real(sumflux,kind=4),real(checkdiffC,kind=4)

      ! Microbial activity
      write(1,'(4(f0.8,","),f0.8)') &
           (365.0*(k-1)+(j+(real(steps-1)/real(steps)))),sumdmact,microbial_activity,&
           Cmicrobes*conversion,microbial_death

      ! Stocks
      write(5,'(6(f0.8,","),f0.8)') &
           (365.0*(k-1)+(j+(real(steps-1)/real(steps)))),&
           CLf*conversion,CLr*conversion,CLw*conversion,&
           Cfast*conversion,Cslow*conversion,Cmicrobes*conversion

      ! Fluxes
      write(6,'(3(f0.8,","),f0.8)') &
           (365.0*(k-1)+(j+(real(steps-1)/real(steps)))),sumRsoil,sumRlitter,sumRdecomp

   endif
   !-------------------------
enddo
!-------------------------
! ANNUAL OUTPUT
!-------------------------

!Calculate annua laverages

avgLf=an_sumLf/daysYear
avgLr=an_sumLr/daysYear
avgLw=an_sumLw/daysYear
avgGluc=an_sumGluc/daysYear

avgFlux1=an_sumFlux1/daysYear;avgFlux2=an_sumFlux2/daysYear;
```

```
   avgFlux3=an_sumFlux3/daysYear;avgFlux4=an_sumFlux4/daysYear;
   avgFlux5=an_sumFlux5/daysYear;avgFlux6=an_sumFlux6/daysYear;
   avgFlux7=an_sumFlux7/daysYear;avgFlux8=an_sumFlux8/daysYear;
   avgFlux9=an_sumFlux9/daysYear;avgFlux10=an_sumFlux10/daysYear;
   avgFlux11=an_sumFlux11/daysYear;avgFlux12=an_sumFlux12/daysYear;
   avgFlux13=an_sumFlux13/daysYear;avgFlux14=an_sumFlux14/daysYear;
   avgFlux15=an_sumFlux15/daysYear;avgFlux16=an_sumFlux16/daysYear;
   avgFlux17=an_sumFlux17/daysYear;avgFlux18=an_sumFlux18/daysYear;
   avgFlux19=an_sumFlux19/daysYear;avgFlux20=an_sumFlux20/daysYear;
   avgFlux21=an_sumFlux21/daysYear;avgFlux22=an_sumFlux22/daysYear;

   avgdCfoliage_dt=an_sumdCfoliage_dt/daysYear;avgdCroot_dt=an_sumdCroot_dt/daysYear
   avgdCwood_dt=an_sumdCwood_dt/daysYear;avgdCwood_dt=an_sumdCwood_dt/daysYear
   avgdCfast_dt=an_sumdCfast_dt/daysYear;avgdCslow_dt=an_sumdCslow_dt/daysYear

   avgdmact=an_sumdmact/daysYear
   avgRsoil=an_sumRsoil/daysYear
   avgRlitter=an_sumRlitter/daysYear
   avgRdecomp=an_sumRdecomp/daysYear

   ! Now write annual output
   if( out(outFreq)==1 ) then
      ! Individual fluxes
      if( ans(soilExtra) ) write(2,'(26(f0.8,","),f0.8)') &
           (365.0*(k-1)+(j+(real(steps-1)/real(steps)))),&
           avgLf,avgLr,avgLw,avgGluc,&
           avgFlux1,avgFlux2,avgFlux3,&
           avgFlux4,avgFlux5,avgFlux6,avgFlux7,avgFlux8,&
           avgFlux9,avgFlux10,avgFlux11,avgFlux12,avgFlux13,&
           avgFlux14,avgFlux15,avgFlux16,avgFlux17,avgFlux18,&
           avgFlux19,avgFlux20,avgFlux21,avgFlux22

      ! Differential equations
      if( ans(soilExtra) ) write(3,'(6(f0.8,","),f0.8)') &
           (365.0*(k-1)+(j+(real(steps-1)/real(steps)))),&
           avgdCfoliage_dt,avgdCroot_dt,avgdCwood_dt,&
           avgdCfast_dt,avgdCslow_dt,avgdCmicrobes_dt

      ! Mass balance
      if( ans(soilExtra) ) write(4,'(5(f0.8,","),f0.8)') &
           (365.0*(k-1)+(j+(real(steps-1)/real(steps)))),real(preC,kind=4),&
           real(curC,kind=4),real(delC,kind=4),real(sumflux,kind=4),real(checkdiffC,kind=4)

      ! Microbial activity
      write(1,'(4(f0.8,","),f0.8)') &
           (365.0*(k-1)+(j+(real(steps-1)/real(steps)))),avgdmact,microbial_activity,&
           Cmicrobes*conversion,microbial_death

      ! Stocks
      write(5,'(6(f0.8,","),f0.8)') &
           (365.0*(k-1)+(j+(real(steps-1)/real(steps)))),&
           CLf*conversion,CLr*conversion,CLw*conversion,&
           Cfast*conversion,Cslow*conversion,Cmicrobes*conversion

      ! Fluxes
      write(6,'(3(f0.8,","),f0.8)') &
           (365.0*(k-1)+(j+(real(steps-1)/real(steps)))),avgRsoil,avgRlitter,avgRdecomp

   endif
   !-------------------------

enddo

! Close all output files
if(ans(soilExtra)) then
   close(2)
   close(3)
   close(4)
   close(7)
endif
```

```
  close(1)
  close(5)
  close(6)

  !-------------------------------------
  ! Containing subroutines and functions
  !-------------------------------------

contains
  !-------------------------------------

  logical function ans(in)
    implicit none
    character(len=3),intent(in):: in
    if(in=='Yes'.or.in=='YES'.or.in=='yes') then
       ans = .true.
    else if(in=='No'.or.in=='NO'.or.in=='no') then
       ans = .false.
    endif
  end function ans
  !-------------------------------------
  integer function out(in)
    implicit none
    character(len=*),intent(in):: in
    if(in=='Annualy'.or.in=='ANNUALY'.or.in=='annualy') then
       out = 1
    else if(in=='Daily'.or.in=='DAILY'.or.in=='daily') then
       out = 2
    else if(in=='Hourly'.or.in=='HOURLY'.or.in=='hourly') then
       out = 3
    endif
  end function out
  !-------------------------------------


end program decobio
```