```
c*****************************************************************
c       $RCSfile: h2olos.F,v $
c       $Revision: 1.1 $
c       $Date: 2007/01/04 22:26:56 $
c       $Locker:  $
c
c$Log: h2olos.F,v $
cRevision 1.1  2007/01/04 22:26:56  wellsj
cInitial revision
c
cRevision 1.1  2006/10/13 18:14:20  wellsj
cInitial revision
c
cRevision 1.2  2004/09/24 20:44:29  bachelet
cheader
c
c*****************************************************************
c           Copyright 1993 Colorado State University
c                All Rights Reserved


      subroutine h2olos(month, aliv, alit, adead, co2val)

      include 'const.inc'
      include 'param.inc'
      include 'parfx.inc'
      include 'plot1.inc'
      include 'plot4.inc'

      integer month
      real aliv, alit, adead, co2val


c...Water Submodel for Century - written by Bill Parton
c     Updated from Fortran 4 - rm 2/92
c     Rewritten by Bill Pulliam - 9/94

c...Local Variables
      integer   j
      real      abs, add, afl, aint, amelt, asimx, avhsm,
     $            avinj, avw, awwt(MAXLYR), base, evl,
     $            evlos, evmt, evsnow, fwlos, inputs,
     $            petrem, pevp, rwc1, sd, snow1, strm,
     $            tot, tot2, trap, trl, winputs

c...to be changed later
      real runoff

c.... removed by jrw ...9/22/06,annet
```

```
c...Description of variables
c
c  adead      the average monthly standing dead biomass(gm/m..2)
c  adep       depth of the ith soil layer(cm)
c  afiel      the field capacity of the ith soil layer(fraction)
c  alit       the average monthly litter biomass(gm/m..2)
c  aliv       the average monthly live plant biomass(gm/m..2)
c  amov       the index for water movement(0-no flow,1-satruated flow)
c  annoff     annual accumulator runoff (cm water per year)
c  asmos      the soil water content of the ith soil layer(cm h2o)
c  asnow      the snow pack water contint(cm-h2o)
c  avh2o (1)  water available to plants for growth
c  avh2o (2)  water available to plants for survival
c            (available water in the whole soil profile)
c  avw                available water in current soil layer
c  awilt      the wilting point of the  ith soil layer(fraction)
c  awtl       the weight factor for transpiration water loss from the ith
c            soil layer(nod)
c  evap       the water evaporated from the  soil and vegetation(cm/mon)
c  evsnow     snow evaporated
c  inputs     rain + irrigation
c  winputs    inputs which are water (not converted to snow)
c  irract     ammount of irrigation (cm)
c  nlayer     number of soil layers with water available for plant survival
c  nlaypg     number of soil layers with water available for plant growth
c  petrem     remaining pet, updated after each incremental h2o loss
c  pevp       the potential evaporation rate from the top  soil layer (cm/day)
c  rain       the total monthly rainfall (cm/month)
c  runoff     the monthly runoff (cm/month)
c  rwcf       the relative water content of the ith soil layer(0-1)
c  snlq       the liquid water in the snow pack
c  tav        average monthly air temperature (2m-c)
c  tran       transpiration water loss(cm/mon)
c  trl        transpiration water loss

c...Initialize Variables
      add = 0.0
      amelt = 0.0
      asimx = 0.0
      avh2o(1) = 0.0
      avh2o(2) = 0.0
      avh2o(3) = 0.0
      evap = 0.0
      pevp = 0.0
      pttr = 0.0
      rwc1 = 0.0
      tran = 0.0
      trap = 0.01
           abs = 0.0
```

```
          evsnow = 0.0
      if(month .eq. 1) annoff = 0.0

c...Calculate total inputs
        inputs = rain + irract



c...Compute runoff -rm 12/96
c   Probert, M.E., B.A. Keating, J.P. Thompson, and W.J. Parton. 1995.
c     Modelling water, nitrogen, and crop yield for a long-term
c     fallow management experiment. Australian Journal of Experimental
c     Agriculture 35:941-950.
c   Using equation for plots with surface residue.
c     runoff = MAX(0.0, 0.41 * (inputs - 7.0))
        runoff = MAX(0.0, 0.55 * (inputs - 5.0))
c   Do not allow runoff to be > 50% of inputs
c     runoff = MIN(runoff, 0.5 * inputs)

c...Create new local variable for water inputs.
c   Set to zero later if it snows.  -mdh 4/95
         winputs = inputs - runoff
c      winputs = inputs

c...Throughout, uses petrem as remaining energy for pet after
c     each melting and evaporation step.  Initially calculated
c     pet is not modified.  Pulliam 9/94
        petrem = pet

c...Determine the snow pack, melt snow, and evaporate from the snow pack

c...When mean monthly air temperature is below freezing,
c     precipitation is in the form of snow.
        if (tave .le. 0.0) then
           snow = snow + inputs
           winputs = 0.0
        endif

c...Melt snow if air temperature is above minimum (tmelt(1))
        if (tave .ge. tmelt(1)) then
c...Calculate the amount of snow to melt:
          amelt = tmelt(2) * (tave - tmelt(1))
          if (amelt .gt. snow) amelt = snow
          snow = snow - amelt

c...Melted snow goes to snow pack and drains excess
c...  add rain-on-snow  and melted snow to snowpack liquid (snlq):
          if (tave .gt. 0 .and. snow .gt. 0) snlq = snlq + inputs
          snlq = snlq + amelt
c...Drain snowpack to 5% liquid content (weight/weight), excess to soil:
          if (snlq .gt. (0.05 * snow)) then
```

```
            add = snlq - 0.05 * snow
            snlq = snlq - add
          endif
        endif

c...Evaporate water from the snow pack (rewritten Pulliam 9/94 to
c     evaporate from both snow aqnd snlq in proportion)
c...Coefficient 0.87 relates to heat of fusion for ice vs. liquid water
c     wasnt modified as snow pack is at least 95% ice.
        if (snow .gt. 0) then

c...Calculate cm of snow that remaining pet energy can evaporate:
          evsnow = petrem * 0.87

c...Calculate total snowpack water, ice + liquid:
          snow1 = snow + snlq

c...Dont evaporate more snow than actually exists:
          if (evsnow .gt. snow1) evsnow = snow1

c...Take evsnow from snow and snlq in proportion:
          snow = snow - evsnow * (snow/snow1)
          snlq = snlq - evsnow * (snlq/snow1)

c...Add evaporated snow to evaporation accumulator (evap):
          evap = evap + evsnow

c...Decrement remaining pet by energy used to evaporate snow:
          petrem = petrem - evsnow / 0.87
          if (petrem .lt. 0.0)  petrem = 0.0
        endif

c...Calculate bare soil water loss and interception
c     when air temperature is above freezing and no snow cover.
c...Mofified 9/94 to allow interception when t < 0 but no snow
c     cover, Pulliam
        if (snow .eq. 0.0) then

c...Calculate total canopy cover and litter, put cap on effects:
            sd = aliv + adead
            if (sd .gt. 800.0) sd = 800.0
            if (alit .gt. 400.0) alit = 400.0

c...canopy interception, fraction of  precip (aint):
            aint = (0.0003 * alit + 0.0006 * sd) * fwloss(1)

c...Bare soil evaporation, fraction of precip (abs):
            abs = 0.5 * exp((-0.002 * alit) - (0.004 * sd)) * fwloss(2)

c...Calculate total surface evaporation losses, maximum
```

```
c    allowable is 0.4 * pet. -rm 6/94
           evl = MIN(((abs + aint) * inputs), (0.4 * petrem))
           evap = evap + evl

c...Calculate remaining water to add to soil and potential
c    transpiration as remaining pet:
c      add = add + inputs - evl
           add = add + winputs - evl
           trap = petrem - evl
         endif


c...Determine potential transpiration water loss (trap, cm/mon) as a
c    function of precipitation and live biomass.
c...If temperature is less than 2C turn off transpiration. -rm 6/94
         if (tave .lt. 2.0) then
            pttr = 0.0
c-----C. Daly 10-2-96.  Catch domain errors on exp.
         elseif (-0.020 * aliv .lt. -10.) then
           pttr = petrem * 0.65 * co2val
         else
            pttr = petrem * 0.65 * (1.0 - exp(-0.020 * aliv)) * co2val
         endif
         if (pttr .le. trap) trap = pttr
         if (trap .le. 0.0) trap = 0.01

c...Maintain pttr on a monthly basis for harvest
         hpttr(month) = pttr

c...Calculate the potential evaporation rate from the top soil layer
c    (pevp-cm/day).  This is not actually taken out until after
c    transpiration losses
         pevp = petrem - trap - evl
         if (pevp .lt. 0.0) pevp = 0.0


c...Transpire water from added water first, before passing water
c    on to soil.  This is necessary for a monthly time step to
c    give plants in wet climates adequate access to water for
c    transpiration. -rm 6/94, Pulliam 9/94
         tran = MIN((trap - .01), add)*fwloss(3)
         trap = trap - tran
         add = add - tran

c...Add water to the soil
c...Changed to add base flow and storm flow.  -rm 2/92

         strm = 0.0
         base = 0.0
         stream(1) = 0.0
```

```
      do 10 j=1,nlayer
c...Add water to layer j:
         asmos(j) = asmos(j) + add

c...Calculate field capacity of soil, drain soil, pass excess
c    on to amov:
         afl = adep(j) * afiel(j)
         if (asmos(j) .gt. afl) then
            amov(j) = asmos(j) - afl
            asmos(j) = afl

c...If you are at the bottom layer, compute storm flow.
            if (j .eq. nlayer) strm = amov(j) * stormf
         else
            amov(j) = 0.0
         endif

c...Copy amov to add, continue with next layer:
         add = amov(j)
10    continue


c...Compute base flow and stream flow for H2O.
c...Put water draining out bottom that doesnt go to stormflow
c    into nlayer+1 holding tank:
      asmos(nlayer+1) = asmos(nlayer+1) + add - strm

c...Drain baseflow fraction from holding tank:
c... Make sure there is some water in the last layer DB 10-1-98
      if(asmos(nlayer+1) .gt. .01) then
        base = asmos(nlayer+1) * basef
      else
     base= 0.0
      endif
      asmos(nlayer+1) = asmos(nlayer+1) - base

c...Streamflow = stormflow + baseflow + runoff:
c  Added runoff into stream(1) -rm 12/96

      stream(1) = strm + base + runoff
      annoff = annoff + stream(1)

c...Save asmos(1) before transpiration for future use:
      asimx=asmos(1)

c...Calculate transpiration water loss from each layer
c...This section was completely rewritten by Pulliam, though it
c    should still do the same thing.  9/94
      rwc1 = 0.0
```

```
      tot = 0.0
      tot2 = 0.0
      do 20 j = 1, nlayer

c...Calculate available water in layer, asmos minus wilting point:
         avw = asmos(j) - awilt(j) * adep(j)
         if (avw. lt. 0.0) avw = 0.0
c...Calculate available water weighted by transpiration depth
c    distribution factors:
         awwt(j) = avw * awtl(j)
c...Sum up available water:
         tot = tot + avw
c...Sum up weighted available water:
         tot2 = tot2 + awwt(j)
20     continue

c...Calculate the actual transpiration water loss(tran-cm/mon)
c...Also rewritten by Pulliam 9/94, should do the same thing
c...Update potential transpiration to be no greater than available water:
      trap = MIN(tot,trap)

c...Transpire water from each layer:
      if (tot2 .gt. 0.) then
c...Calculate available water in layer j:
         do 30  j = 1, nlayer
            avinj = asmos(j) - awilt(j) * adep(j)
            if (avinj .ne. avinj) then
              print *, 'avinj is a NaN. j = ', j
              print *, 'asmos(j), awilt(j), adep(j) = ',
     $             asmos(j), awilt(j), adep(j)
              stop
            endif
            if (avinj .lt. 0.0) avinj = 0.0

c...Calculate transpiration loss from layer j, using weighted
c    water availabilities:
            trl = ((trap * awwt(j))/tot2)*fwloss(3)
            if (trl .gt. avinj) then
               trl = avinj*fwloss(3)
            endif
            asmos(j) = asmos(j) -trl
            avinj = avinj - trl
            if (avinj .ne. avinj) then
              print *, 'avinj is a NaN. j = ', j
              print *, 'trl = ',
     $             trl
              stop
            endif
            tran = tran + trl
```

```
c...  getting a NaN value in rcwf when adep=0
c...  added some code to prevent problem
c...  but rcwf should be infinity not zero
c...  D. Bachelet 10-8-97

          if (adep(j) .eq. 0.0) then
          rwcf(j) = 0.0
          else
                  rwcf(j) = (asmos(j)/adep(j)-awilt(j)) / (afiel(j)-awilt(j))
              endif

c...Sum up water available to plants for growth:
              if (j .le. nlaypg) avh2o(1) = avh2o(1) + avinj

c...Sum up water available to plants for survival:
              avh2o(2) = avh2o(2) + avinj

c...Calculate parameter of H2O accumulation in top 2 soil layers:
              if (j .le. 2) avh2o(3) = avh2o(3) + avinj
30          continue
        endif


c...Set htran for use in harvst.f
           htran(month) = tran


c...Evaporate water from the top layer
c...Rewritten by Pulliam, should still do the same thing 9/94

c...Minimum relative water content for top layer to evaporate:
        fwlos = 0.25

c...Fraction of water content between fwlos and field capacity:
        evmt = (rwcf(1)-fwlos)/(1.-fwlos)
        if (evmt .le. 0.01) evmt = 0.01

c...Evaporation loss from layer 1:
        evlos = evmt * pevp * abs * 0.10
        avinj = asmos(1) - awilt(1) * adep(1)
              if (avinj .ne. avinj) then
                print *, 'avinj is a NaN. '
                print *, 'asmos(1), awilt(1), adep(1) = ',
     $               asmos(1), awilt(1), adep(1)
                stop
              endif
        if (avinj .lt. 0.0) avinj = 0.0
        if (evlos .gt. avinj) evlos = avinj
        asmos(1) = asmos(1) - evlos
        evap = evap + evlos
```

```
c...Recalculate rwcf(1) to estimate mid-month water content
      avhsm = (asmos(1) + rwc1 * asimx)/(1. + rwc1)

c...  getting a NaN value in rcwf when adep=0
c...  added some code to prevent problem
c...  but rcwf should be infinity not zero
c...  D. Bachelet 10-8-97

     if (adep(1) .eq. 0.0) then
         rwcf(1) = 0.0
     else
             rwcf(1)=(avhsm/adep(1)-awilt(1))/(afiel(1)-awilt(1))
     endif

c...Update available water pools minus evaporation from top layer
      avh2o(1) = avh2o(1) - evlos
      avh2o(2) = avh2o(2) - evlos
      avh2o(3) = avh2o(3) - evlos


c...Compute annual actual evapotranspiration
     annet = annet + evap + tran

      if (avh2o(3) .ne. avh2o(3)) then
        print *, '*** h2olos.F: avh2o(3) is a NaN.'
        print *, 'evlos, avinj = ', evlos, avinj
        stop
      endif

      return
      end
```