
Listing of file subroutine qsadd_moist_stg

```
c
c Twall-driver
c
c THIS VERSION INCLUDES ADJUSTMENT FOR A VARIABLE TEMPERATURE WALL.
c CURRENTLY, THE WALL MUST BE A PART OF THE CELL.
c THE CODE WILL NOT ACCOUNT FOR A CELL THAT IS VF=1 BUT NEXT TO AN OBSTACLE.
c
c This subroutine contains the phase change algorithm created by
c S. Green to support work for analyzing the in-drift transport
c of water vapor for SwRI Center for Nuclear Waste Regulatory
c Anlysis, SwRI Project 20.06002.01.091, during the timeframe
c 2003-2004.
c
c This subroutine is called from the FLOW-3D subroutine QSADD.
c The algorithm is described in the CNWRA Sientific Notebook #536E
c maintained by Steve Green. The property routines RHOICAL and RHOECAL
c are also modified to handle composition-dependent density
c energy.
c
c
c
c     use mblock_module
c
c     use arrays_module
c
c     use arrayp_module
c
c     use meshcb_module
c
c     use voids_module
c
c
c -----
c Modified by STG, 3-05
c SwRI project 20.06002.01.091, Coldtrap effect in Yucca Mtn.
c Version removed RCOBS from the OBSD common block. Have to attach the OBSIJK
module
c to get this value
c     use obsijk_module
c -----
c
c
c #ifdef SINGLE
c     include '../comdeck/precis4.f'
c #else
c     include '../comdeck/precis.f'
c #endif
c     include '../comdeck/params.f'
c     include '../comdeck/cntrl.f'
c     include '../comdeck/const.f'
c     include '../comdeck/dumn.f'
c     include '../comdeck/phiou.f'
c     include '../comdeck/scala.f'
c     include '../comdeck/state.f'
c     include '../comdeck/pardat.f'
c -----
c Modified by STG, 9-03
c SwRI project 20.06002.01.091, Coldtrap effect in Yucca Mtn.
c Added items for evaporation/condensation model at walls
```

```

c
c      include '../comdeck/obsd.f'
c-----
c
c      Include the user data common block to activate/deactivate the code in this
c      subroutine for the
c      special case(s)
c          include '../comdeck/cbusr.f'
c
c
c      Skip over if no scalars exist and this subroutine is used for
c          scalar sources
c
c          if(nsc.eq.0) then
c              return
c          endif

cc
cc  \/  \/  \/  \/  \/  \/  \/  \/  \/  \/  \/  \/  \/  \/  \/  \/  \/  \/  \/  \/  \/
cc----- Simple Test case for scalar advection from a source -----
c          Used with a special input file for channel flow where
c          a mass source is at the bottom wall with flow
c          from left to right
cc          if (k.eq.2 .and. (8.le.i.and.i.le.12)) then
cc              sclr(ijk,1) = 1.
cc          endif
cc  \  \  \  \  \  \  \  \  \  \  \  \  \  \  \  \  \  \  \  \  \  \  \  \
/\
cc
c
c      Define the parameters in the water vapor saturation vapor pressure equation.
c
c          F0kkhm=-741.9242
c          F1kkhm=-29.721
c          F2kkhm=-11.55286
c          F3kkhm=-0.8685635
c          F4kkhm=0.1094098
c          F5kkhm=0.439993
c          F6kkhm=0.2520658
c          F7kkhm=0.05218684
c          vapmw = 18.01534
c          airmw = 28.9645
c          Platm = 101325.

c
c      Loop over all the obstacles to define local phase change conditions
c      This looping method is based on the example in drgcl.f
c
c          do 2000 nob=1,nobs
c
c          Do not execute unless the user indicates that this obstacle
c          is to be included in evap/condensation
c
c              if(imoist_stg(nob) .ne. -nob .and.
c                  1          ilqonly_stg(nob) .ne. -nob) go to 2000
c          !=====
c
c          mincel=kajk(nob,nbl)
c          maxcel=kajk(nob+1,nbl)-1
c          if(maxcel.lt.mincel) go to 2000
c          do 1950 m=mincel,maxcel
c              ijk=ijkobs(m)
c              call inijk(ijk,i,j,k)

```

```

      if(ijk.lt.1) go to 1950
      if(vf(ijk).lt.em6) go to 1950
      include '../comdeck/mijk.f'
      include '../comdeck/pijk.f'
c
c Retrieve the cell surface area and fluid volume
c
      vcell=vf(ijk)*delx(i)*dely(j)/rri(i)*delz(k)
      sa=waobs(m)
c
c
c Compute the concentration at saturation as follows:
c
c----- CODE MUST BE RUN WITH SI UNITS FOR THIS MODEL TO WORK!!!!!!!!!!!!
c
c   a. Guess the final temperature
c   b. Compute the saturation vapor pressure
c       Use the Keenan, Keyes, Hill, and Moore equation
c   c. Compute the saturation molar concentration (moles of vapor per total moles)
c       as the ratio of saturation vapor pressure to total pressure
c   d. Compute the mass concentration of vapor using the respective molecular
weights
c   e. Compute the final temperature resulting from the phase change
c   f. Repeat steps b-e as needed.
c
c
c Get the current fluid temperature in the cell
c Save the initial temperature
c
      tfinal= teval(ijk)
      tinit= tfinal
      twalli = tw(ijk)
c
c If this part of the code gets activated but there is not a wall temperature then
there
c is a wall on an adjacent face. Find that wall and use its temperature.
c
c
c
c
c
      if (ijk .eq. 243) then
c   do 900 il2345=1,21168
c       write(77,*) '-----'
c       write(77,*) t
c       write(77,*) iii,m,ijk,i,j,k
c       write(77,*) vf(ijk), sa
c       write(77,*) tn(ijk), tw(ijk)
c       write(77,*) tns(il2345), tws(il2345)
c 900   continue
c       jxxx=2
c       kxxx=4
c       write(77,*) '-----'
c       do 900 ixxx=1,imax
c           ijkxxx=ii2*(kxxx-1)+imax*(jxxx-1)+ixxx+ii5
c           write(77,*) ixxx,jxxx,kxxx, nf(ijkxxx)
c           write(77,*) vf(ijkxxx)
c           write(77,*) tn(ijkxxx),tw(ijkxxx)
c           write(77,*) tns(ijkxxx),tws(ijkxxx)
c 900   continue
c
c       endif

```

```

c
c Find the maximum neighbor wall temperature
  if (twalli .eq. 0) then
    include '../comdeck/mijk.f'
    include '../comdeck/pijk.f'
    twmax = 0.
    if (tw(imjk) .gt. twmax) then
      twmax = tw(imjk)
      ijktw = imjk
    endif
    if (tw(ijmk) .gt. twmax) then
      twmax = tw(ijmk)
      ijktw = ijmk
    endif
    if (tw(ijkm) .gt. twmax) then
      twmax = tw(ijkm)
      ijktw = ijkm
    endif
    if (tw(ipjk) .gt. twmax) then
      twmax = tw(ipjk)
      ijktw = ipjk
    endif
    if (tw(ijpk) .gt. twmax) then
      twmax = tw(ijpk)
      ijktw = ijpk
    endif
    if (tw(ijkp) .gt. twmax) then
      twmax = tw(ijkp)
      ijktw = ijkp
    endif
    twalli = twmax
  endif
  if (twalli .le. 0) go to 1950

    twallf = twalli
    tnk = tfinal
    tnc = tnk-273.15
    arg = 0.65-0.01*tnc
c
c Compute the initial concentration before phase change
c Define 1 atmosphere as 101300 Pascal
c
    term1 = F0kkhm + arg*(F1kkhm+ arg*(F2kkhm + arg*(F3kkhm +
1           arg*(F4kkhm+ arg*(F5kkhm + arg*(F6kkhm +
2           arg*(F7kkhm))))))
    Pvsati=217.99*exp(0.01/tnk*(374.136-(tnc))*term1)
    Pvsati = Pvsati * Platm
c
c Saturation Molar concentration ..... moles of vapor per total moles
c If Pvsati > p, the saturation values will be invalid, but invalid values are
trapped below
    Xvsati = Pvsati/p(ijk)
c
c Saturation Mass Concentration .....mass of vapor to total mass
    Yvsati = Xvsati/((1.-Xvsati)*airmw/vapmw+Xvsati)
c
c Current vapor mass concentration
    Yvacti = sclr(ijk, isvap_stg)
c
c Begin the loop to balance the energy and evaporated/condensed mass
c
    nitr_moist = 0
100    continue

```

```

c
c Compute the saturation pressure at the final fluid film temperature.
c Assume this is the wall temperature.
c
c PATCH - If there is not a wall temperature, this means the volume fraction is 1 (no
wall)
c but the cell is next to an obstacle. Need to fix this later so that the
obstacle
c and mesh lines can coincide an the code will recognize that an adjacent
obstacle needs
c to be included in the energy balance.
c
      if (twalli .le. 0.) go to 1950
      tnk = twallf
      tnc = tnk-273.15
      arg = 0.65-0.01*tnc
      term1 = F0kkhm + arg*(F1kkhm+ arg*(F2kkhm + arg*(F3kkhm +
1          arg*(F4kkhm+ arg*(F5kkhm + arg*(F6kkhm +
2          arg*(F7kkhm))))))
      Pvsat=217.99*exp(0.01/tnk*(374.136-(tnc))*term1)
      Pvsat = Pvsat * Platm
c
c Saturation Molar concentration ..... moles of vapor per total moles
c Limit the value to its maximum of Xvsat=1 if Pvsat > P(ijk)
      Xvsat = Pvsat/p(ijk)
      if (Xvsat .gt. 1.) Xvsat = 1.
c Saturation Mass Concentration .....mass of vapor to total mass
      Yvsat = Xvsat/((1.-Xvsat)*airmw/vapmw+Xvsat)
c
c Total density of cell mixture at saturation conditions at final temperature
c Same calculation as in RHOCAL STG
      rhosat = p(ijk)/tfinal*
1          (xvsat/rvap_stg+(1.-xvsat)/rgas_stg)
      rhosat = p(ijk)/twallf*
1          (xvsat/rvap_stg+(1.-xvsat)/rgas_stg)
c
c Evaporated mass if entire cell goes to saturation condition
      delmmax = (rhosat*Yvsat - rho(ijk)*Yvacti)*vcell
c
c Evaporated mass based on diffusion rate from the surface across the cell
c diffusion mass flux = rho * diff.coef * (Yvsat@Tfinal - Yvacti)/(distance normal
to cell)
c distance normal =~ open volume of cell divided by wall surface area
      delmdif = rho(ijk)*cmisc(isvap_stg)*delt*
1          (Yvsat-Yvacti)*sa*sa/vcell/0.5
c
c Mass flow into cell is the minimum of the two delta-mass estimates
      delm = delmmax
      if (abs(delmdif) .lt. abs(delmmax)) delm=delmdif
c
c New estimate of final temperature
      tfsave = tfinal
c
c Solid wall energy change based on energy balance with the mass undergoing ophase
change
      twfsave = twallf
      vwall = vcell/vf(ijk)*(1.-vf(ijk))
      if (rcobs(nob) .gt. 0.) then
          twallf = twalli-delm*hvvap_stg/rcobs(nob)/vwall
      endif
      twallf = twfsave + vaprlx_stg*(twallf-twfsave)
      nitr_moist = nitr_moist+1

```

```

c
c
c For walls that have limited water available, check to be sure there is water.
c Bypass remaining calcs if there is no water to evaporate.
      if (ilqonly_stg(nob) .eq. -nob .and.
1         sclr(ijk,istlq_stg) .le. 0. .and.
2         delm .gt. 0.) then
          delm = 0.
          sclr(ijk,istlq_stg) = 0.
          twallf = twalli
          twfsave=twallf
        endif
c
c
c Check for convergence
      if (abs(twallf-twfsave) .gt. .001 .and.
1         nitr_moist .lt. 25) go to 100
c
c New value of concentration
      delmrat = delm/rho(ijk)/vcell
      sclr(ijk,isvap_stg) = (Yvacti + delmrat)/(1.+delmrat)
c
c Liquid mass flux at wall
c positive for condensation, negative for evaporation
      sclr(ijk,isliq_stg) = -delm/delt/sa
c
c Total net liquid mass exchanged since beginning of simulation.
      sclr(ijk,istlq_stg) = sclr(ijk,istlq_stg) - delm

c
c      if (ijk .eq. 243) then
c          write(77,*) '-----'
c          write(77,*) 'time',t
c          write(77,*) 'i,j,k,ijk', i,j,k,ijk
c          write(77,*) 'tn,twall,rho', tn(ijk), twallf, rho(ijk)
c          write(77,*) 'delmrat', delmrat
c          write(77,*) 'cmisc, delt', cmisc(isvap_stg), delt
c          write(77,*) 'yvsat,yvacti,sa,vcell', yvsat, yvacti, sa, vcell
c          write(77,*) 'delmmax, delmdif', delmmax, delmdif
c          write(77,*) 'sclr', sclr(ijk,isvap_stg)
c      endif

cc
cc Update the energy to account only for the vapor phase entering or leaving the cell
c PATCH - Do not execute the following three lines until the FLOW-3D energy/temp
calc's are resolved
cc
      tvap = tfinal
      if (delm .gt. 0) tvap=tw(ijk)
c          rhoe(ijk) = rhoe(ijk) + delm/vcell*cvvap_stg*tvap
c Update the energy in the cell for the new concentration. Use TN for the
temperature in RHOCAL
c          rhoe(ijk) = rhoecl(ijk)

1          rhoe(ijk) = rhoe(ijk) + delm*hvvap_stg/vcell +
              cvvap_stg*delm/vcell*(tfinal-twallf)

```

```

c
c
c  Change the wall temperature according to the energy balance above.
      if (rcobs(nob) .gt. 0) then
          tw(ijk) = twallf
          endif
1950      continue
2000 continue
c
c  Loop over all real cells to compute the relative humidity
c
      do 3000 k=2,km1
          do 3000 j=2,jm1
              do 3000 i=2,im1
c ----- calculate current cell index
                  include '../comdeck/ijk.f'
c ----- skip calculation for completely blocked cells
                  if(vf(ijk).lt.em6) goto 3000

                      tnk = teval(ijk)
                      tnc = tnk-273.15
                      arg = 0.65-0.01*tnc
                      term1 = F0kkhm + arg*(F1kkhm+ arg*(F2kkhm + arg*(F3kkhm +
1                          arg*(F4kkhm+ arg*(F5kkhm + arg*(F6kkhm +
2                          arg*(F7kkhm))))))
                      Pvsat=217.99*exp(0.01/tnk*(374.136-(tnc))*term1)
                      Pvsat = Pvsat * Platm
c
c  Relative Humidity ..... moles of vapor per moles of vapor at saturation
                  Yvact = sclr(ijk, isvap_stg)
                  Xvact = Yvact/(Yvact+(1.-Yvact)*vapmw/airmw)
                  sclr(ijk, isrh_stg) = p(ijk)*Xvact/Pvsat

                  sclr(ijk, isyvw_stg) = sclr(ijk, isvap_stg)
                  sclr(ijk, isywl_stg) = 0.

3000 continue

5000 continue
      return
      end

```

End of listing of file subroutine qsadd_moist_stg

Listing of file subroutine teval_stg.F

```

      function teval(ijk)
c
c *****
c ** notice:  this routine contains flow science, inc. proprietary **
c **          trade secret and confidential information.           **
c **                                                                 **
c **          unauthorized use prohibited                          **
c *****
c
c      use mblock_module
c
c      use arrays_module
c
c      use arrayp_module
c
c      use meshcb_module
c
c      use voids_module
c
c #ifdef SINGLE
c      include '../comdeck/precis4.f'
c #else
c      include '../comdeck/precis.f'
c #endif
c      include '../comdeck/params.f'
c      include '../comdeck/phiou.f'
c      include '../comdeck/cntrl.f'
c      include '../comdeck/const.f'
c      include '../comdeck/state.f'
c
c      include '../comdeck/cbusr.f'
c
c      evaluate fluid temperature from total mixture internal energy
c              and total mixture density
c              AND VAPOR CONCENTRATION
c
c      if (isvap_stg .le. 0) then
c          iosval=-999
c          if(iosval.ne.0) call fsystem(iosval,
1          'THIS TEVAL ROUTINE IS ONLY FOR SPECIAL MOISTURE CALCs')
c          stop
c      endif
c
c1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
c Original code by STG
c      yv = sclr(ijk, isvap_stg)
c      wtmolv = 18.015
c      wtmola = 28.97

```



```

c      rmolav = wtmola/wtmolv
c      xmolv = yv*rmolav/(1-yv*(1-rmolav))
c      xmola = 1-xmolv
c
c
c      teval = rhoe(ijk)/rhof - yv*hvvap_stg
cc
cc Don't include the latent heat
c      teval = rhoe(ijk)/rhof
c      teval = teval/((1-yv)*cv1+yv*cvvap_stg)
c1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
c2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
c 5-20-05 code by STG Allows for liquid fraction in excess of saturation
vapor fraction
      vapmw = 18.01534
      airmw = 28.9645
      rmolav = airmw/vapmw

      ywv = sclr(ijk,isywv_stg)
      ywl = sclr(ijk,isywl_stg)
      ya = 1.-ywl-ywv
      xwv = ywv*rmolav/(1-ywv*(1-rmolav))
      pvact = xwv*p(ijk)
      tvsat = vapidp_stg(pvact)
      if (ywl .gt. 0) tvsat = tn(ijk)
      rhomix = rhocal(ijk)

      write(77,*) ' '
      write(77,*) '-----'
      write(77,*) 'TEVAL'
      write(77,*) 'ijk,rhoecl,rhomix'
      write(77,*) ijk,rhoe(ijk), rhomix
      write(77,*) 'ywv,cvliq_stg, cvvap_stg,tvsat,hvvap_stg'
      write(77,*) ywv,cvliq_stg, cvvap_stg,tvsat,hvvap_stg
      write(77,*) 'ya,cv1,ywl'
      write(77,*) ya,cv1,ywl

      teval = rhoe(ijk)/rhomix -
1      ywv*((cvliq_stg-cvvap_stg)*tvsat+hvvap_stg)
      teval = teval/(ya*cv1+ywl*cvliq_stg+ywv*cvvap_stg)

c      rhoecl = rhomix*(ya*cv1*tn(ijk) + ywl*cvliq_stg*tn(ijk) +
c 1      ywv*(cvliq_stg*tvsat + hvvap_stg + cvvap_stg*(tn(ijk)-tvsat)))

c2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
c-----

      return
c
c      end

```

End of listing of file subroutine teval_stg.F

Listing of file subroutine rusrd_stg.F

```

subroutine rusrd
c
c * *   read special user defined namelist data
c
c *****
c **               notice **
c ** this subprogram contains flow science, inc. proprietary **
c **   trade secret and confidential information. **
c **               **
c **               unauthorized use prohibited **
c **               copyright 1985-2003 flow science, inc. **
c *****
c
#ifdef SINGLE
  include '../comdeck/precis4.f'
#else
  include '../comdeck/precis.f'
#endif
  include '../comdeck/params.f'
  include '../comdeck/cntrl.f'
  include '../comdeck/dumn.f'
  include '../comdeck/phiou.f'
  include '../comdeck/cbusr.f'
c-----
c   Modified by STG, 9-03
c   SwRI project 20.06002.01.091, Coldtrap effect in Yucca Mtn.
c   Added items for evaporation/condensation model at walls
  include '../comdeck/obsd.f'
c-----
c
  include '../comdeck/usrdat.f'
c
  iosval=0
c
  if(iusrd.lt.1) return
c-----
c   Modified by STG, 9-03
c   SwRI project 20.06002.01.091, Coldtrap effect in Yucca Mtn.
c   Added items for evaporation/condensation model at walls
c
c   Modified by STG, 4-04
c   Added variable ilqonly_stg to flag surfaces that can evaporate water
c   up to the amount that has been already condensed previously.
c
c Initialize the wall evap/cond. paramters
c
  isvap_stg=0
  isliq_stg=0
  isrh_stg=0
  istlq_stg=0
  isywl_stg=0
  isyvw_stg=0
  istwtf_stg='xx'

```

```

do 10 nob=1,nobx
  imoist_stg(nob) = 0
  ilgonly_stg(nob) = 0
  idrftw_stg(nob) = 0
  vaprlx_stg = .3
c-----
c Modified by STG, 2-04
c Initialize rocktemp function coefficients
c
  rocktemp_a(nob)=0.
  rocktemp_b(nob)=0.
  rocktemp_c(nob)=0.
  rocktemp_d(nob)=0.
  rocktemp_e(nob)=0.
  rocktemp_f(nob)=0.
c
  10 continue
c-----
c-----
c Modified by STG, 12-03
c Added sinusoidal z-motion for obstacles - hooks for VELMOV
c
c
c Initialize the amplitude, frequency, and phase
c
  velobj_amp=0.
  velobj_fhz=0.
  velobj_prd=0.
c-----
c
c-----
c
c * * read input data from namelist / usrdat /
c
c*****read(ihd3in,usrdat)
  include '../comdeck/husrdt.f'
c
  if(iosval.ne.0) call fsystem(iosval,
1    'input error while reading namelist usrdat')
  call rnlend('usrdat')
c
c=====
c Trap out a problem if Vapor Model flag is set but no choice for
c energy source is given
  if (isvap_stg .gt. 0 .and.
1    istwtf_stg .ne. 'tf' .and.
2    istwtf_stg .ne. 'tw') then
    iosval=-999
    if(iosval.ne.0) call fsystem(iosval,
1    'BAD VALUE FOR ISTWTF_STG IN USRDAT')
    stop
  endif
c=====
c
c
c
  return
end

```

End of listing of file subroutine rusrd_stg.F

Listing of file subroutine prusrd_stg.F

```

subroutine prusrd(icud)
c
c * * read special user defined input data
c
c *****
c **                               notice                               **
c ** this subprogram contains flow science, inc. proprietary **
c ** trade secret and confidential information. **
c **                               **
c **                               unauthorized use prohibited **
c **                               copyright 1985-2003 flow science, inc. **
c *****
c
c arg disp description
c icud i flag to indicate location of
c         this call to prusrd
c
c         icud=2, read usrdat according to
c                 iusrd=2, i.e. before mesh
c
c         icud=1, read usrdat according to
c                 iusrd=1, i.e. after parts
c         and
c                 write out usrdat data to
c                 both prpout and hd3in
c
#ifdef SINGLE
include '../comdeck/precis4.f'
#else
include '../comdeck/precis.f'
#endif
c
include '../comdeck/params.f'
include '../comdeck/cntrl.f'
include '../comdeck/dumn.f'
include '../comdeck/phiou.f'
include '../comdeck/cbusr.f'
c
include '../comdeck/usrdat.f'
c
iosval=0
if(iusrd.lt.1) return
c-----
c Modified by STG, 9-03
c SwRI project 20.06002.01.091, Coldtrap effect in Yucca Mtn.
c Added items for evaporation/condensation model at walls
c
c
c Modified by STG, 4-03

```

```

c      Added initialization for variable ilqonly_stg
c
c Initialize the wall evap/cond. paramters
c
      isvap_stg=0
      isliq_stg=0
      isrh_stg=0
      istlq_stg=0
      istwtf_stg='xx'
      do 10 nob=1,nobx
         imoist_stg(nob) = 0
         ilqonly_stg(nob) = 0
         idrftw_stg(nob) = 0
         vaprlx_stg = .3
c-----
c      Modified by STG, 2-04
c      Initialize rocktemp function coefficients
c
      rocktemp_a(nob)=0.
      rocktemp_b(nob)=0.
      rocktemp_c(nob)=0.
      rocktemp_d(nob)=0.
      rocktemp_e(nob)=0.
      rocktemp_f(nob)=0.
c
c-----
      10 continue
c-----
c-----
c      Modified by STG, 12-03
c      Added sinusoidal z-motion for obstacles - hooks for VELMOV
c
c Initialize the amplitude, frequency, and phase
c
      velobj_amp=0.
      velobj_fhz=0.
      velobj_prd=0.
c-----

c
c      read namelist /usrdat/ from input file 'prepin'
c
      if(icud.eq.iusrd) then
c          *****
c*****read(iprpin,usrdat)
          call fndnml(iprpin,'usrdat',ifind)
          if(ifind.eq.0) then
              iosval=-1
          else
              include '../comdeck/pusrdt.f'
          endif
c          *****
          if(iosval.ne.0) call error(iosval,'prusrd',
1              'input error while reading namelist usrdat')
      endif

```

```

c
c      only write output if icud=1
c
c      if(icud.ne.1) return
c
c      write namelist /usrdat/ to input file 'hd3in'
c
c      *****
c      remark=' '
c      commnt=' '
c      write(ihydin,usrdat)
c      call wnlesp('usrdat')
c      *****
c
c      write namelist /usrdat/ to output file 'prpout'
c
c      write(iout,1000) iusrd
1000 format(//' special user defined namelist data ',
1       ' option= ',i5,/)
c      write(iout,usrdat)
c      write(iout,2000)
2000 format(//' end of user defined namelist data ',///)
c
c      return
c      end

```

End of listing of file subroutine prusrd_stg.F

Listing of file subroutine rhoecl_stg.F

```

      function rhoecl(ijk)
c
c      evaluate cell macroscopic energy
c
c      *****
c      **              notice              **
c      ** this subprogram contains flow science, inc. proprietary **
c      ** trade secret and confidential information.             **
c      **              **
c      **              unauthorized use prohibited              **
c      **              copyright 1985-2003 flow science, inc.   **
c      *****
c
c      use arrays_module
c
c      #ifdef SINGLE
c      include '../comdeck/precis4.f'
c      #else
c      include '../comdeck/precis.f'
c      #endif
c      include '../comdeck/params.f'
c      include '../comdeck/scala.f'
c      include '../comdeck/cntrl.f'
c      include '../comdeck/state.f'

```



```
c 5-20-05 code by STG Allows for liquid fraction in excess of saturation
vapor fraction
```

```
    vapmw = 18.01534
    airmw = 28.9645
    tnk = tn(ijk)
```

```
    write(77,*) ' RHOECL 1 before pvsat'
    write(77,*) 'ijk, tnk', ijk,tnk
```

```
        Pvsat = Pvsat_stg(tnk)
```

```
c
c
```

```
    write(77,*) ' RHOECL 1 after pvsat'
```

```
c
```

```
c
```

```
c Saturation Molar concentration ..... moles of vapor per total moles
c If Pvasati > p, the seaturation values will be invalid, but invalid values
are trapped below
```

```
    Xvsat = Pvsat/p(ijk)
```

```
c
```

```
c Saturation Mass Concentration .....mass of vapor to total mass
```

```
    Yvsat = Xvsat/((1.-Xvsat)*airmw/vapmw+Xvsat)
```

```
c
```

```
c Vapor concentration
```

```
    yw = sclr(ijk, isvap_stg)
```

```
    xw = yw*airmw/vapmw/(1.-yw*(1.-airmw/vapmw))
```

```
    pvact = xw*p(ijk)
```

```
c
```

```
c
```

```
c
```

```
c
```

```
    write(77,*) ' RHOECL 1 before vapidp'
```

```
    write(77,*) 'tnk', tnk
```

```
c
```

```
c
```

```
c
```

```
        tvsat = vapidp_stg(pvact)
```

```
c
```

```
c
```

```
    write(77,*) ' RHOECL 1 after vapidp'
```

```
c
```

```
    ywv = yw
```

```
    if (yw .gt. yvsat) then
```

```
        ywv = yvsat
```

```
        tvsat = tn(ijk)
```

```
    endif
```

```
c
```

```
c Liquid concentration
```

```
    ywl = yw - ywv
```

```
c
```



```

c Air concentration
  ya = 1.-yw
c
c Air and vapor mole concentrations in the gas portion
  ywvg = ywv/(ywv+ya)
  yag = 1.-ywvg
  xwvg = ywvg*airmw/vapmw/(1.-ywvg*(1.-airmw/vapmw))
  xag = 1.-xwvg
c
c Bulk Density
  rhomix = rhocal(ijk)
c

  write(77,*) ' '
  write(77,*) '-----'
  write(77,*) 'RHOECL'
  write(77,*) 'ijk,rhoec1,rhomix'
  write(77,*) ijk,rhoe(ijk), rhomix
  write(77,*) 'ywv,cvliq_stg,cvvap_stg,tvsat,hvvap_stg'
  write(77,*) ywv,cvliq_stg,cvvap_stg,tvsat,hvvap_stg
  write(77,*) 'ya,cv1,ywl'
  write(77,*) ya,cv1,ywl

c
  rhoec1 = rhomix*(ya*cv1*tn(ijk) + ywl*cvliq_stg*tn(ijk) +
1  ywv*(cvliq_stg*tvsat + hvvap_stg + cvvap_stg*(tn(ijk)-tvsat)))

  return
endif
c2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
c-----
c
  rhoec1=f(ijk)*rho1*e1+(one-f(ijk))*rho2*e2
c
  return
end

function pvsat_stg(tnk)
c
c Saturation pressure of water vapor.,
c Uses Keenan Keyes Hill and Moore relation for 0 deg.C < tnk < 350 deg.C
c Uses Goff formula for tnk < 0 deg C
c
c
c Input:
c   tnk = temperature in Kelvin
c
c Output:
c   pvsat = saturation pressure in Pascals (absolute)
c
c
#ifdef SINGLE
  include '../comdeck/precis4.f'
#else

```

```

        include '../comdeck/precis.f'
#endif
c
c Define 1 atm pressure
    Platm = 101325.
c
c KKHM equation coefficients
    F0kkhm=-741.9242
    F1kkhm=-29.721
    F2kkhm=-11.55286
    F3kkhm=-0.8685635
    F4kkhm=0.1094098
    F5kkhm=0.439993
    F6kkhm=0.2520658
    F7kkhm=0.05218684
c
c Goff equation coefficients
    f0goff = -9.096936
    f1goff = -3.56654
    f2goff = 0.867817
    f3goff = -2.2195983
c
c Use correct equation
    if (tnk .ge. 273.15) then
c
c Function arguments
        tnc = tnk-273.15
        arg = 0.65-0.01*tnc
c
c Compute the initial concentration before phase change
c Define 1 atmosphere as 101300 Pascal
c
    term1 = F0kkhm + arg*(F1kkhm+ arg*(F2kkhm + arg*(F3kkhm +
1          arg*(F4kkhm+ arg*(F5kkhm + arg*(F6kkhm +
2          arg*F7kkhm))))))
    pvsat_stg = platm * 217.99*exp(0.01/tnk*(374.136-(tnc))*term1)
    else
        theta = 273.16/tnk
        term1 = f0goff*(theta-1.) + f1goff*log10(theta) +
1          f2goff*(1.-1./theta) + f3goff
    pvsat_stg = platm * 10.**term1
    endif
    return
    end

        function vapdp_stg(pvsat)
c
c Dewpoint of water.
c Uses Keenan, Keyes, Hill, and Moore formula for saturation pressure of
water over liquid
c and Goff formula for water vapor over ice.
c
c Iteratively solves saturation pressure equation to get temperature
c
c Input:

```

```
c      pvsat = saturation pressure in Pascals
c
c      Output:
c      vapdp_stg = saturation temperature Kelvins
c
c
c      #ifdef SINGLE
c      include '../comdeck/precis4.f'
c      #else
c      include '../comdeck/precis.f'
c      #endif
c
c      Get a reasonable estimates of temperature before getting to accurate
c      expressions
c
c
c      write(77,*) 'In Vapdp, @tguess pvsat=',pvsat
c
c      if (pvsat .le. 0) then
c      vapdp_stg = 0.
c      return
c      endif
c      tguess = (log10(pvsat)-2.8)/4.5*370 + 273.15
c      if (pvsat .lt. 610.8) tguess = (log10(pvsat)-1.1)/1.68*40+233.15
c
c      Initialize the counter to limit number of iterations
c      icnt = 0
c
c      100 continue
c
c      icnt = icnt + 1
c      pguess = pvsat_stg(tguess)
c
c      err = (pvsat-pguess)/pvsat
c      dpdt = (pvsat_stg(tguess+1)-pvsat_stg(tguess-1))/2.
c      derrdt = -pguess/pvsat/pvsat * dpdt
c      tguess = tguess - err/derrdt
c      if (err .ge. 0.0001 .and. icnt .lt.200) go to 100
c
c
c      vapdp_stg = tguess
c
c      write(77,*) 'eND OF vapdp',pvsat
c
c      return
c      end
```

End of listing of file subroutine rhoecl_stg.F

Listing of file subroutine rhocal_stg.F

```
function rhocal(ijk)
c
c   evaluate cell density from microscopic values
c
c   *****
c   **                               **
c   ** this subprogram contains flow science, inc. proprietary **
c   **   trade secret and confidential information.           **
c   **                               **
c   **                               **
c   **   unauthorized use prohibited                          **
c   **   copyright 1985-2003 flow science, inc.               **
c   *****
c
c   use arrays_module
c
c   #ifdef SINGLE
c     include '../comdeck/precis4.f'
c   #else
c     include '../comdeck/precis.f'
c   #endif
c     include '../comdeck/params.f'
c     include '../comdeck/cntrl.f'
c     include '../comdeck/const.f'
c     include '../comdeck/scala.f'
c     include '../comdeck/state.f'
c-----
c   Modified by STG, 9-03
c
c     include '../comdeck/cbusr.f'
c-----
c-----
c   Modified by STG, 9-03
c   SwRI project 20.06002.01.091, Coldtrap effect in Yucca Mtn.
c   Added items for evaporation/condensation model at walls
c
c   Define mixture density in terms of air and water vapor for wall
c   evap/cond. paramters
c   Compute the bulk densities using ideal gas law. The scalar is defined as
c   mass concentration
c   of water vapor
c
c     if (isvap_stg .gt. 0) then
c       vapmw = 18.01534
c       airmw = 28.9645
cc
c1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
c   Original code by STG
cc   Compute the mole fraction of the water vapor
cc   using hard-wired molecular weight
c     wtmolv = 18.015
c     wtmola = 28.97
```

```

c      rmolav = wtmola/wtmolv
c      yv = sclr(ijk, isvap_stg)
c      xmolv = yv*rmolav/(1-yv*(1-rmolav))
c      xmola = 1-xmolv
c      rhov = p(ijk)*xmolv/rvap_stg/tn(ijk)
c      rhoa = p(ijk)*xmola/rgas_stg/tn(ijk)
c      rhocal = rhov+rhoa
c      return
c  endif
c1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
c2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
c 5-20-05 code by STG Allows for liquid fraction in excess of saturation
vapor fraction
c
c      tnk = tn(ijk)
c      Pvsat = Pvsat_stg(tnk)
c
c Saturation Molar concentration ..... moles of vapor per total moles
c If Pvasati > p, the saturation values will be invalid, but invalid values
are trapped below
c      Xvsat = Pvsat/p(ijk)
c
c Saturation Mass Concentration .....mass of vapor to total mass
c      Yvsat = Xvsat/((1.-Xvsat)*airmw/vapmw+Xvsat)
c
c Vapor concentration
c      yw = sclr(ijk, isvap_stg)
c      ywv = yw
c      if (yw .gt. yvsat) ywv = yvsat
c
c Liquid concentration
c      ywl = yw - ywv
c
c Air concentration
c      ya = 1.-yw
c
c Air and vapor mole concentrations in the gas portion
c      ywvg = ywv/(ywv+ya)
c      yag = 1.-ywvg
c      xwvg = ywvg*airmw/vapmw/(1.-ywvg*(1.-airmw/vapmw))
c      xag = 1.-xwvg
c
c Bulk Density
c      rhocal = p(ijk)/tn(ijk)/(1-ywl)*(xag/rgas_stg+xwvg/rvap_stg)
c      return
c  endif
c2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
c
c      if(ifenrg.gt.0) then
c          temp=tn(ijk)
c      else
c          temp=zero
c      endif
c      rhocal=rholcl(p(ijk),temp)
c
c * * test for solidification
c

```

Listing of file subroutine usrdat.f

```
c          *****
c          **   usrdat   **
c          *****
c
c      this namelist is specifically supplied to
c      permit the user to add additional input data.
c
c      the standard version will always contain only
c      the "remark" variable and the dummy variables
c      "udumvr", "cdumvr" and "iudumv".
c
c      this namelist will be read, printed, and transferred
c      from prep3d to hydr3d only if the iusrd variable is
c      set to "1" in the xput namelist.
c
c      namelist / usrdat / udumvr,cdumvr,iudumv,remark,commnt
c      namelist / usrdat / udumvr,cdumvr,iudumv,remark,commnt,
1      stg_force, ibelt_stg, imoist_stg, ilgonly_stg, isvap_stg,
2      isliq_stg, isrh_stg, istlq_stg, hvvap_stg, rvap_stg,
3      rgas_stg, cvvap_stg, vaprlx_stg, cvliq_stg,
4      istwtf_stg, idrftw_stg, isywl_stg, isywv_stg,
4      velobj_amp, velobj_fhz, velobj_prd,
5      rocktemp_a,rocktemp_b,rocktemp_c,rocktemp_d,
6      rocktemp_e,rocktemp_f
```

End of listing of file subroutine usrdat.f

Listing of file subroutine cbrusr.f

```
c          *****
c          **   cbrusr   **
c          *****
c
c      this common block is specifically supplied to
c      permit the user to transfer data between standard
c      FLOW-3D routines and his own special purpose routines
c
c      the standard version will always contain only
c      dummy variables "udumvr", "cdumvr" and "iudumv" as samples.
c
c      common / cbrusr /   udumvr
c      save /cbrusr/
c
```

```

      if(f(ijk).lt.emf .or. (ifslld.eq.0 .and. iscour1.eq.0)) go to 100
c
      if(iscour1.gt.0) then
        rhocal=rhocal+(one-fliq(ijk))*(rhofs-rhof)
      elseif(ifslld.eq.2) then
c account for solutal buoyancy
        sclrq=sclr(ijk,iseq)
        if(fliq(ijk).gt.zero) then
          if(tn(ijk).gt.teut) then
            denom=fliq(ijk)+(one-fliq(ijk))*pcoef
            sclrq=sclr(ijk,iseq)/denom
          else
c ceut is eutectic liquid concentration
            sclrq=ceut
          endif
        else
          rhocal=rhofs
        endif
        rhocal=rhocal-rhcexf*(sclrq-cstar)
        if(rhocal.lt.ztest) rhocal=rhof
      elseif(ifslld.eq.1) then
        rhocal=rhofs+(rhocal-rhofs)*fliq(ijk)
      endif
c
      100 continue
c
      if(rhocal.lt.ztest) rhocal=rhof
c
c add mass associated with scalars assuming they do not
c displace fluid #1
c
      do ns=1,nsc
        if(irhosc(ns).eq.0) cycle
        if(isclr(ns).eq.2 .or. isclr(ns).eq.3)
1          rhocal=rhocal+max(zero,sclr(ijk,ns))
      enddo
c
      if(nmat.eq.1) return
c
      rho2=rho2cl(p(ijk),temp)
c
c add mass associated with scalars assuming they do not
c displace fluid #2
c
      do ns=1,nsc
        if(irhosc(ns).eq.0) cycle
        if(isclr(ns).gt.3) rho2=rho2+max(zero,sclr(ijk,ns))
      enddo
c
      rhocal=f(ijk)*rhocal+(one-f(ijk))*rho2
c
      return
      end

```

End of listing of file subroutine rhocal_stg.F

```
common / cbusrc / cdumvr
save /cbusrc/
character*10 cdumvr
c
common / cbusri / iudumv
save /cbusri/

common /force_stg/ stg_force
save /force_stg/

common /belt_stg/ ibelt_stg
save /belt_stg/

dimension imoist_stg(nobx), ilqonly_stg(nobx)
common /moist_stgi/ isvap_stg, isliq_stg, isrh_stg, istlq_stg,
1 imoist_stg, ilqonly_stg, isywl_stg, isyvw_stg
save /moist_stgi/

common /moist_stgc/ istwtf_stg
save /moist_stgc/
character*2 istwtf_stg

common /moist_stg/ hvvap_stg, rvap_stg, rgas_stg, cvvap_stg,
1 vaprlx_stg, cvliq_stg
save /moist_stg/

dimension idrftw_stg(nobx)
common /drftw_stg/ idrftw_stg
save /drftw_stg/

common /velobj_dw/ velobj_amp, velobj_fhz, velobj_prd
save /velobj_dw/

common /rocktemp_stg/ rocktemp_a(nobx), rocktemp_b(nobx),
1 rocktemp_c(nobx), rocktemp_d(nobx),
2 rocktemp_e(nobx), rocktemp_f(nobx)
save /rocktemp_stg/
```

End of listing of file subroutine cbusr.f

END OF ENTRY FOR 6/23/05

STG

7/31/05

STG

4/17/08
SP7
6A3

The new scoping version of the vapor transport model software was used to model the condensation cell test setup. This test setup is described in Scientific Notebook _____ maintained by Steve Svedeman for the lab testing of the condensation cell.

The test runs included operation at several combinations of water pan temperature and cold wall temperature. The FLOW-3D input file

prepin.Tw_51-0_19-3_actual-geom

describes the FLOW-3D specifications corresponding to a pan temperature of 51.0°C and a cold wall temperature of 19.3°C. This file was modified accordingly for each of the test run configurations.

Listing of file prepin.Tw_51-0_19-3_actual-geom

```
Condense Cell w/condensation, Test#11, Baro=98544 Pa, 51.0C/19.3C, TW ,
90x23, 2-D, Uniform, Laminar, NO RHOE Update
6" high, 23" long
2" insulation all around
```

```
$xput
  remark='units are SI',
  itb=0,    ifvis=0,    ifenrg=2, ifrho=1,  ihtc=2,  ipdis=1,
  iwsh=1,
  gz=-9.8,
  iadiz=1,
  iusrd=1,
  deltd=1.e-4, pltdt=50., sprtdt=1.,
  twfin=1000.,
  isolid=0,
  hpltdt=5.,
  rmrhoe=1.,
  rmrho=1.,
$end

$limits
$end
```

```

$props
  rhof=1.095,
  mul=2.e-05, units='si',
  cv1=717., thc1=0.026,
  thexf1=0.003235, tstar = 308.3,
$send

$scalar
  nsc=8,
  isclr(1)=3, cmisc(1)=0.26e-04, scltit(1)='Tot.Water', rmisc=1.,
  isclr(2)=0, cmisc(2)=0., scltit(2)='Liq.Flux',
  isclr(3)=0, cmisc(3)=0., scltit(3)='Rel.Hum',
  isclr(4)=0, cmisc(4)=0., scltit(4)='Net.Liq',
  isclr(5)=0, cmisc(5)=0., scltit(5)='Vap.Wat',
  isclr(6)=0, cmisc(6)=0., scltit(6)='Liq.Wat',
  isclr(7)=0, cmisc(7)=0., scltit(7)='Itr.Wall',
  isclr(8)=0, cmisc(8)=0., scltit(8)='Itr.Mesh',
$send

$bcdata
  wl=2, wr=2, tbc(1)=300., tbc(2)=300.,
  wf=1, wbk=1,
  wb=2, wt=2, tbc(5)=300., tbc(6)=300.,
$send

$mesh
  px(1)=-0.0931,          pz(1)=-0.0931,
  px(2)=-0.0762,          py(1)=0.0,      pz(2)=-0.0762,
  px(3)=-0.0254,          py(2)=0.3048,   pz(3)=-0.0254,
  px(4)=0.00,             pz(4)= 0.0,
  px(5)=0.1016,          pz(5)= 0.1524,
  px(6)=0.5842,          pz(6)= 0.1778,
  px(7)=0.6096,          pz(7)= 0.2286,
  px(8)=0.6604,          pz(8)= 0.2455,
  px(9)=0.6773,          nzcell(1)=2,
  nxcell(1)=2,           nzcell(2)=6,
  nxcell(2)=6,           nzcell(3)=3,
  nxcell(3)=3,           nzcell(4)=23,
  nxcell(4)=15,          nzcell(5)=3,
  nxcell(5)=75,          nzcell(6)=5,
  nxcell(6)=3,           nzcell(7)=2,
  nxcell(7)=6,
  nxcell(8)=2,
  nxcelt=112,           nycelt=1,        nzcelt=44,
$send

$obs
  avrck=-3.1,
  nobs = 13,
  tobs(1)=0., tobs(2)=1000.,
  remark='Obstacle 1. Hot water surface and Aluminum Tray',
  zl(1)=-0.0254, zh(1)=0.,
  xl(1)= -0.00, xh(1)=0.1016,
  twobs(1,1)=324.05, twobs(2,1)=324.05,
  remark='Obstacle 2. Cold endwall surface',
  zl(2)=-0.0254, zh(2)=0.1778,
  xl(2)=0.5842, xh(2)=0.6096,

```

```
twobs(1,2)=292.45, twobs(2,2)=292.45,
remark='Obstacle 3. Lexan bottom wall',
zl(3)=-0.0254, zh(3)=0.0,
xl(3)=0.1016, xh(3)=0.5842,
kobs(3)=0.19, rcobs(3)=10000.,
twobs(1,3)=300.,
remark='Obstacle 4. Styrofoam bottom wall',
xl(4)=-0.0762, xh(4)=0.6604,
zl(4)=-0.0762, zh(4)=-0.0254,
kobs(4)=0.03, rcobs(4)=10000.,
twobs(1,4)=300.,
remark='Obstacle 5. Lexan end wall',
zl(5)=-0.0254, zh(5)=0.1778,
xl(5)=-0.0254, xh(5)=0.0,
kobs(5)=0.19, rcobs(5)=10000.,
twobs(1,5)=300.,
remark='Obstacle 6. Styrofoam end wall',
xl(6)=-0.0762, xh(6)=-0.0254,
zl(6)=-0.0254, zh(6)=0.2286,
kobs(6)=0.03, rcobs(6)=10000.,
twobs(1,6)=300.,
remark='Obstacle 7. Lexan top wall',
zl(7)=0.1524, zh(7)=0.1778,
xl(7)=0., xh(7)=0.5842,
kobs(7)=0.19, rcobs(7)=10000.,
twobs(1,7)=300.,
remark='Obstacle 8. Styrofoam top wall',
zl(8)=0.1778, zh(8)=0.2286,
xl(8)=-0.0762, xh(8)=0.6604,
kobs(8)=0.03, rcobs(8)=10000.,
twobs(1,8)=300.,
remark='Obstacle 6. Styrofoam end wall',
xl(9)=0.6096, xh(9)=0.6604,
zl(9)=-0.0254, zh(9)=0.2286,
kobs(9)=0.03, rcobs(9)=10000.,
twobs(1,9)=300.,
remark='Obstacle 10. Bottom Boundary Condition, Const Temp',
zl(10)=-0.0931, zh(10)=-0.0762,
twobs(1,10)=300.,
remark='Obstacle 11. Top Boundary Condition, Const Temp',
zl(11)=0.2286, zh(11)=0.2455,
twobs(1,11)=300.,
remark='Obstacle 12. Left Boundary Condition, Const Temp',
xl(12)=-0.0931, xh(12)=-0.0762,
twobs(1,12)=300.,
remark='Obstacle 13. Right Boundary Condition, Const Temp',
xl(13)=0.6604, xh(13)=0.6773,
twobs(1,13)=300.,
htcob(1,5)=78.7, remark='1/8 Buna gasket between aluminum and Lexan
endwall',
htcob(5,7)=78.7, remark='1/8 Buna gasket between Lexan top and Lexan
endwall',
htcob(2,3)=78.7, remark='1/8 Buna gasket between Lexan bottom and cold
wall',
htcob(2,7)=78.7, remark='1/8 Buna gasket between Lexan top and cold
wall',
$end
```

H.T. coefficients for Buna are computed from
htcob = k/(thickness
k = thermal conductivity = 0.25 W/m/K
from Handbook of Tables for Engineering Sciences, 2nd ed., CRC Press,
1973. p. 156
thickness = 1/8"

```
$fl
  remark=' 100RH at 308., 99059 kPa',
  sclri(1)=0.036,
  sclri(5)=0.036,
  sclri(6)=0.,
  presi=99059.,
$end
```

```
$bf
$end
```

```
$temp
  ntmp=1,
  tempi=308.3,
$end
```

```
$motn
$end
```

```
$grafic
$end
```

```
$parts
$end
```

```
$usrdat
  istwtf_stg='tw',
  imoist_stg(1) = -1,
  imoist_stg(2) = -2,
  ilqonly_stg(3) = -3,
  ilqonly_stg(5) = -5,
  ilqonly_stg(7) = -7,
  isvap_stg = 1,
  isliq_stg = 2,
  isrh_stg = 3,
  istlq_stg = 4,
  isywv_stg = 5,
  isywl_stg = 6,
  hvvap_stg=2300000.,
  cvvap_stg=1411.,
  cvliq_stg=4186.,
  rvap_stg=416.,
  rgas_stg=289.,
  vaprlx_stg=0.8,
$end
```

End of listing of file prepin.Tw_51-0_19-3_actual-geom

The model predictions for all the simulations and test conditions chosen for the model checkout were compiled into an Excel spreadsheet,

[Data_Prediction_Summary.xls](#)

for convenient plotting and comparison.

The FLOW-3D predictions for the temperature at select locations are compared to the test measurements in Figure 7-51-05-A.

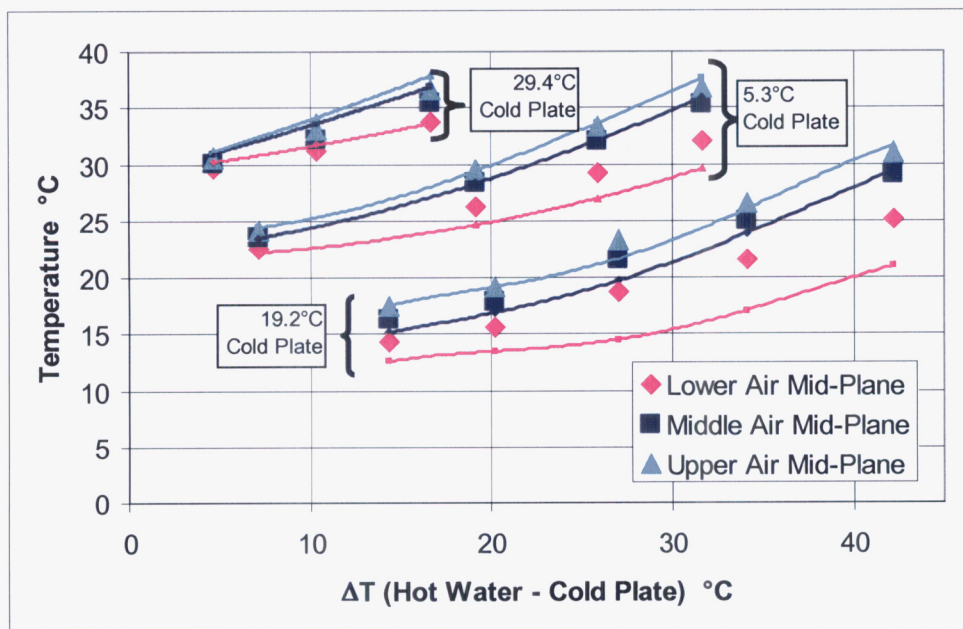


Figure 7-51-05-A. Measured and Predicted Temperature in Condensation Test Cell.

In general there is good agreement between the measured and predicted values with respect to the data trends. The predicted condensation rate is compared to the measured condensation rate in Figure 7-51-05-B.

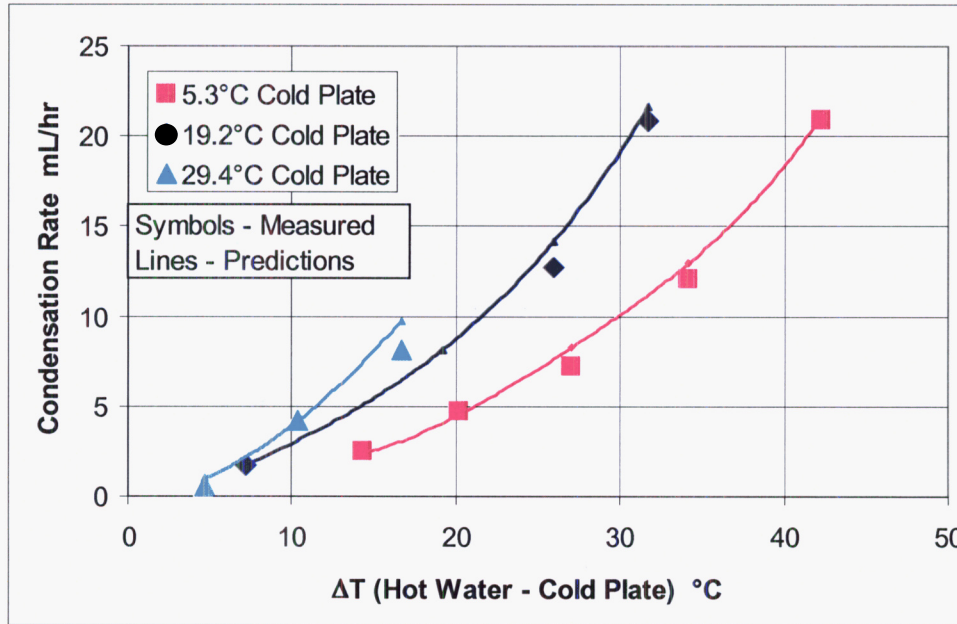


Figure 7-51-05-B. Measured and Predicted Condensation Rates in Condensation Test Cell.

There is excellent agreement in these results.

END OF ENTRY FOR 7/31/05 *STG*

=====
Entries made into Scientific Notebook #536E for the period April 10, 2005 to November 30, 2005, have been made by Steven Green (December 1, 2005).

No original text or figures entered into this Scientific Notebook has been removed

STG 12/1/2005

1/5/06 STG

During executions of the vapor transport model for the 1/5 scale test setup, it was discovered that FLOW3D cell references were being made outside the valid range of cells in subroutine E1CAL_STG. For example, $ijk < 1$ was an input from the calling routine. This is not possible. The call was being made outside of any subroutine that had been modified by SwRI for the vapor transport mode. That is, it appears that the call was being made from the stock portions of the code. This problem was manifested in a failure due to overflow in the EXP functions in which the pressure value for the dewpoint calculations was NaN.

It was discovered that when $ijk < 0$ in the argument list, the value of ijk refers to a boundary condition that is to be applied. This error was rectified by inserting the following modified code segment.

```
if (ijk .eq. 0) return
if (ijk .lt. 0) then
  yw = sclbc(-ijk, isvap_stg)
  ywv = sclbc(-ijk, isywv_stg)
  ywl = sclbc(-ijk, isywl_stg)
  if (ijk .lt. -6) return
else
  yw = sclr(ijk, isvap_stg)
  ywv = sclr(ijk, isywv_stg)
  ywl = sclr(ijk, isywl_stg)
endif
```

END OF ENTRY FOR 1/5/06 STG

3/3/06 STG

This entry records the work performed to conclude the development of a moisture transport module and a surface radiation module into FLOW-3D for use in the analysis of post-closure (and possibly pre-closure?) waste repository heat transfer and fluid flow. This entry is intended to serve as a draft of a software documentation document describing these modules

The thermal radiation heat transfer module was written in November 2005 to January 2006. This module was incorporated into the FLOW-3D code along with the moisture transport module. In doing so, the entire process of computing the moisture transport, phase change, and the radiation heat transfer were coordinated and streamlined.

This entry is meant to fully document the module development activity to serve as the starting point for a software description document for these two modules. This entry covers the mathematical foundation for the two modules, describes the major variables in the software, presents the pseudocode for demonstrating the logic of the routines, and provides a set of instructions for using the modules.

References

ASHRAE Handbook, 1977 Fundamentals, New York City, New York: American Society of Heating Refrigeration and Air-Conditioning Engineers, Inc. 1977.

Bird, R.B., Stewart, W.E., and Lightfoot, E. N., **Transport Phenomena**. New York City, New York, John Wiley and Sons, Inc. 1960.

FLOW-3D User's Manual, Version 9.0, Flow Science Inc., Santa Fe, New Mexico, 2005.

Goff, J. A., "Saturation Pressure of Water on the new Kelvin Scale," **Humidity and Moisture Measurement and Control in Science and Industry**, Wexler, A., and Wildhack, W. H., eds., Reinhold Publishing Corp., New York, 1965, p. 289.

Keenan, J. H., Keyes, F. G., Hill, P. G., Moore, J. G., **Steam Tables: Thermodynamic Properties of Water, Including Vapor, Liquid, and Solid Phases**, John Wiley and Sons, Inc, 1969.

Siegel, R., and Howell, J., **Thermal Radiation Heat Transfer**, Third Edition, Hemisphere Publishing Corp., Washington, D. C., 1992.

Basic Equations

The basic equations describing the water moisture transport and thermal radiation heat transfer processes specific to the conditions expected in nuclear waste repository drifts are described in this section. These equations are translated into software for including into FLOW-3D as described in the "Software Description" section below

Water Transport Module

The simulation of moisture transport in FLOW-3D requires that evaporation and condensation processes be modeled under high-humidity conditions. Laboratory tests and engineering analyses indicate that the software should also be capable of allowing for condensed water to be present in the bulk of the flow domain – not only at the walls of the enclosure. The key assumptions for the moisture transport model are listed below. The equations for estimating the fluid density and thermal energy that are required for the model are described next. Finally, the defining equations for the mass transfer and heat transfer rates associated with water moisture transport are presented.

Assumptions

Several assumptions are made here to incorporate high-humidity conditions with a robust moisture transport model into FLOW-3D.

1. Air and water vapor act as ideal gases with temperature dependent density. This allows the use of the ideal gas equation of state to be used to compute the mixture density as a function of temperature and composition for simulating the buoyancy effects on fluid momentum.
2. A Boussinesq-like assumption is used in that the density is assumed to be constant in the energy equation.
3. Water can enter and exit the flow domain only at walls specified as sources or sinks of water.
4. Walls not specified as sources/sinks of water can have water condense on them. This water is available for re-evaporation from these walls.
5. The energy of phase change is taken from or given to the walls for evaporation and condensation, respectively.
6. Condensed 'fog' acts as a mist that diffuses and advects like water vapor. When the relative humidity is limited to 100%, water can condense in the bulk of the flow domain. This water is not allowed to coalesce and 'rain' out. That phenomenon is outside the scope of this model.

Density Evaluation

The bulk density of the air/vapor mixture is obtained from engineering psychrometrics (for example, ASHRAE, 1977) with the modification that allows for some water to be condensed as a mist in the mixture. Recall that the species diffusion equation solved here is in terms of the mass fraction of the air and water; so, the density property in FLOW-3D must be defined in terms of the mass fractions of these constituents.

The bulk density is defined as

$$\rho = \frac{m_{tot}}{V} = \frac{m_a + m_{wv} + m_{wl}}{V} \quad (1)$$

where m_{tot} = total mass of all species

m_a = mass of air

m_{wv} = mass of water vapor

m_{wl} = mass of water mist

V = volume of control volume

The mass of air is closely approximated from the ideal gas equation of state,

$$m_a = \frac{P_a V_g}{R_a T} \approx \frac{P_a V}{R_a T} = \frac{P V}{R_a T} x_{ag} \quad (2)$$

where P_a = partial pressure of air

P = total static pressure

$R_a = 287 \text{ J}/(\text{kg}\cdot\text{K})$ = ideal gas constant for air

x_{ag} = air molar concentration in the gas/vapor portion of the volume

V_g = volume of gas only in control volume

T = temperature

Likewise, the mass of water vapor is

$$m_{wv} = \frac{P_{wv}V_g}{R_vT} \approx \frac{P_{wv}V}{R_vT} = \frac{PV}{R_vT} x_{wv} \quad (3)$$

where P_{wv} = partial pressure of water vapor

$R_v = 462 \text{ J}/(\text{kg}\cdot\text{K})$ = ideal gas constant for water vapor

x_{wv} = air molar concentration in the gas/vapor portion of the volume

In the above expressions, we have assumed that the gas volume is approximately equal to the total volume $V_g \approx V$. This is substantiated by the following numerical example. Consider the scenario in which

$$T = 50^\circ\text{C} = 323 \text{ K}$$

$$P = 1 \text{ atm} = 101325 \text{ Pa}$$

Effective RH=200% (mass of water mist is equal to mass of water vapor)

From steam tables for these conditions, the saturation pressure of water is $P_{v,sat} = 0.122 \text{ atm}$. Thus, the partial pressure of water vapor is $P_{wv} = P_{v,sat}$. Using $V_g = 1 \text{ m}^3$ as a basis volume one finds that

$$m_a = 0.9597 \text{ kg}$$

$$m_{wv} = 0.0827 \text{ kg}$$

$$m_{wl} = 0.0827 \text{ kg}$$

The volume of the liquid water is thus $8.27 \times 10^{-5} \text{ m}^3 \approx 10^{-4} V_g$. So, the approximation the $V_g \approx V$ is valid even for this extreme example.

The mass fractions of the vapor water and liquid water are computed as part of the humidity calculations and are known inputs to the density evaluation algorithm. The mass fractions of water vapor and air in only the gas space can now be defined,

$$c_{wv} = \frac{c_{wv}}{c_{wv} + c_a} = \frac{c_{wv}}{c_{wv} + (1 - c_w)} \quad c_{ag} = 1 - c_{wv} \quad (4)$$

The respective mole fractions are

$$x_{wv} = \frac{c_{wv} \frac{M_a}{M_w}}{1 - c_{wv} \left(1 - \frac{M_a}{M_w}\right)} \quad x_{ag} = 1 - x_{wv} \quad (5)$$

Substituting all this back into Eq. (1)

$$\begin{aligned}\rho &= \frac{1}{V} \left[\frac{PV}{R_a T} x_{ag} + \frac{PV}{R_w T} x_{wvg} + c_{wl} m_{tot} \right] \\ &= \frac{P}{T} \left(\frac{x_{ag}}{R_a} + \frac{x_{wvg}}{R_w} \right) + c_{wl} \rho\end{aligned}\quad (6)$$

which can be reduced to

$$\rho = \frac{P}{T(1 - c_{wl})} \left(\frac{x_{ag}}{R_a} + \frac{x_{wvg}}{R_w} \right) \quad (7)$$

Equation (7), with Eq. (5), serves as the basis for the evaluating the bulk density for the purposes of this moisture transport model. The inputs to this function are the local pressure, temperature, water vapor mass fraction, and liquid water (as mist) mass fraction.

Air/Water Vapor Energy

The energy of an air/water mixture is evaluated following the basic procedures in the standard installation of the FLOW-3D code. In this case, the mass-specific energy (e.g., joule/kg) is a function of the local temperature and constituent concentrations,

$$e = c_a C_{va} T + c_{wv} C_{vv} T - c_{wl} h_{vap} \quad (8)$$

where c_a = mass fraction of air

c_{wv} = mass fraction of water vapor

c_{wl} = mass fraction of liquid water (as mist)

T = temperature

C_{va} = constant volume specific heat of air

C_{vv} = constant volume specific heat of water vapor

h_{vap} = energy of vaporization of water

Temperature Evaluation

The conservation of energy equation used in FLOW-3D is one based on the volume specific energy defined as the product of the nominal fluid density and the mass specific energy,

$$(\rho I) = \rho_{nom} e \quad (9)$$

where (ρI) = volume specific energy (this is the FLOW-3D nomenclature)

ρ_{nom} = constant nominal density provided by the user in the problem setup

e = mass-specific energy for the air/water mixture

The (ρI) product is the conserved quantity and the temperature is back-calculated for use in various property functions and output files. Consequently, the temperature evaluation function in FLOW-3D must be modified when using the moisture transport model in which the mass-specific energy is different from the standard FLOW-3D installation. The mass-specific energy equation, Eq. (18), can be rearranged as

$$T = \frac{(\rho I) + c_{wl}h_{vap}}{c_a C_{va} + c_{wv} C_{vv}} \quad (10)$$

where c_a = mass fraction of air

c_{wv} = mass fraction of water vapor

c_{wl} = mass fraction of liquid water (as mist)

C_{va} = constant volume specific heat of air

C_{vv} = constant volume specific heat of water vapor

h_{vap} = energy of vaporization of water

Moisture Transport Rates

The module for the transport of water vapor will be based on the well established approach of the conservation of mass of multiple chemical species in non-reacting mixtures (see Bird, Stewart and Lightfoot, 1960). A conservation equation for the transport of water vapor can be written as

$$\rho \left(\frac{\partial c_w}{\partial t} + \frac{\partial c_w}{\partial x} + \frac{\partial c_w}{\partial y} + \frac{\partial c_w}{\partial z} \right) = \nabla \cdot (\rho D_{va} c_w) + \dot{m}_{source}^m \quad (11)$$

where ρ = bulk density of the air/water mixture (kg/m^3)

c_w = water mass concentration mass; i.e., ratio of water mass to total mass (kg/kg)

\dot{m}_{source}^m = volume specific rate of vapor generation in the domain ($\text{kg}/(\text{sec} \cdot \text{m}^3)$)

u, v, w = vector components of bulk fluid flow (m/s)

x, y, z = coordinate directions (m)

D_{va} = diffusion coefficient for water vapor/air (m^2/sec)

The FLOW-3D software solves this equation in conjunction with the equations for the conservation of mass and momentum of the air-vapor mixture. The moisture transport module does not directly specify the source term in Eq. (11) for the mass transfer rate at walls. Rather, the module essentially provides the boundary conditions for Eq. (11) at wall surfaces. The methodology is described below.

Phase Change at Walls

The scenario for mass transfer at a wall is shown in Figure 3/3/06-1.

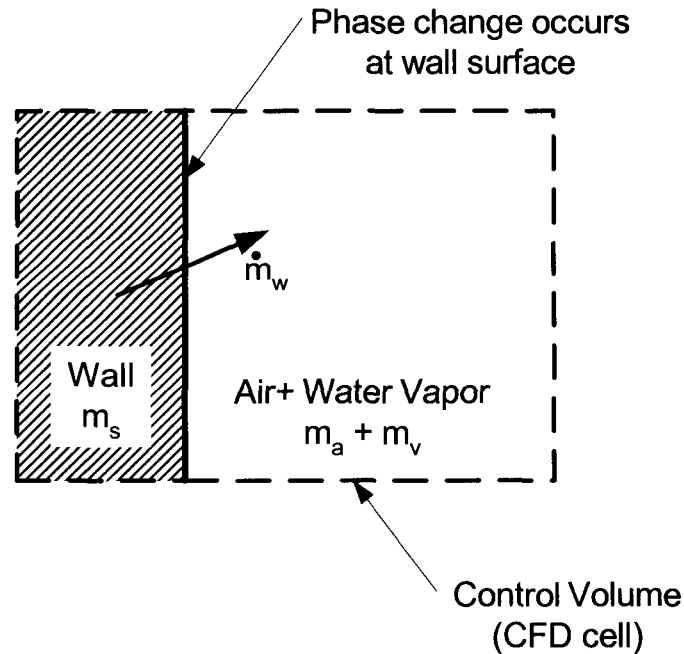


Figure 3/3/06-1. Phase Change Process at Walls

The conservation of energy is applied to the solid wall and the liquid film on the wall over a single time step in the calculation. Evaporation or condensation will take place as necessary to satisfy the energy change of the solid material. Any heat transfer to/from the air/vapor mixture takes place via conduction/convection at the wall; the air/vapor mixture next to the wall does not exchange energy directly with the water that is undergoing the phase change.

Armed with these assumptions, the conservation of energy is expressed as

$$m_s \frac{e_{s,2} - e_{s,1}}{\Delta t} = \dot{m}_w h_{vap}$$

$$\rho_s C_s V_s \frac{T_{s,2} - T_{s,1}}{\Delta t} = \dot{m}_w h_{vap} \quad (12)$$

where m_s = mass of solid wall inside the control volume (kg)

\dot{m}_w = mass flow rate of water evaporating/condensing (kg/sec)

$e_{s,1}$ = mass-specific energy of solid at the beginning of the time step (J/kg)

$e_{s,2}$ = mass-specific energy of solid at the end of the time step (J/kg)

ρ_s = solid material density (kg/m³)

C_s = solid material specific heat (J/(kg K))

V_s = solid material volume (m³)

h_{vap} = heat of vaporization (J/kg)

Δt = time step (sec)

$T_{s,1}$ = wall temperature at beginning of time step

$T_{s,2}$ = wall temperature at end of time step

There are two unknowns in Eq. (12); namely, $T_{s,2}$ and \dot{m}_w . The mass flow of water is obtained either from the diffusion flux of water vapor from the wall into the control volume adjacent to the wall or the mass flow of water that will provide for a saturated air/vapor mixture in the control volume at the fluid temperature.

One estimate of the mass transfer rate is based on the diffusion rate of vapor across the computational cell. The diffusion flow of vapor is estimated from an approximation to the Fickian diffusion law (Bird, Stewart and Lightfoot, 1960),

$$\dot{m}_{w,d} = \rho D_{va} A_w \frac{dc}{dx} \approx \rho D_{va} (c_{w,sat} - c_w) \frac{A_w^2}{V_f} \quad (13)$$

where $c_{w,sat}$ = mass concentration of water vapor at saturation

c_w = current mass concentration of water vapor

A_w = wall surface area inside the cell

V_f = volume of fluid in the cell

D_{va} = diffusion coefficient for air/vapor

ρ = bulk density of mixture

The other estimate of the mass transfer rate is the change in water content required to bring the vapor concentration to the saturation value. Based on the law of conservation of mass, the flow rate required to saturate the air/vapor mixture adjacent to the wall in a single computational time step is estimated as

$$\dot{m}_{w,s} = (\rho_{sat} c_{w,sat} - \rho c_w) \frac{V_f}{\Delta t} \quad (14)$$

where ρ_{sat} = bulk density at saturation condition

$c_{w,sat}$ = concentration at saturation condition

ρ = current bulk density of mixture

c_w = current mass concentration of water vapor

V_f = fluid volume in the FLOW-3D grid cell

Δt = time step

The phase change rates from Eq. (13) and (14) are compared. The equation providing the minimum value is used in conjunction with Eq. (12) to solve for $T_{s,2}$ and \dot{m}_w .

This process allows for the source terms of water mass into and out of the flow domain to be computed for use in the species diffusion equation. This process also provides for the source/sink of energy at the surfaces of solid obstacles through the adjustment of the obstacle temperature in conjunction with the water phase change at the surface.

Phase change away from walls

During a simulation, the transport of energy, mass and the diffusion of the water may create a situation in which the relative humidity becomes greater than 100%. If so, the local temperature and relative portions of water vapor and water mist must be altered to satisfy this constraint.

To accomplish the vapor-liquid mass adjustment, assume that the total energy of a control volume (i.e., a FLOW-3D grid cell) is constant,

$$(\rho I)_2 = (\rho I)_1 = \rho_{nom} (c_{a,1} C_{va} T_1 + c_{wv,1} C_{vv} T_1 - c_{wl,1} h_{vap}) \quad (15)$$

where (ρI) = volume specific energy (this is the FLOW-3D nomenclature)

ρ_{nom} = constant nominal density provided by the user in the problem setup

c_a = mass fraction of air

c_{wv} = mass fraction of water vapor

c_{wl} = mass fraction of liquid water (as mist)

T = temperature

C_{va} = constant volume specific heat of air

C_{vv} = constant volume specific heat of water vapor

h_{vap} = energy of vaporization of water

The subscript '1' refers to the conditions before the adjustment of the water vapor and water liquid mass fraction adjustment and '2' refers to the conditions after the adjustment. State '2' is one in which the vapor is at the saturation pressure at T . The state '1' is assumed to be when the vapor mass fraction is greater than allowed by the water vapor saturation pressure at temperature T . If there is no liquid water and the temperature is greater than the dewpoint, no adjustment is necessary

The temperature and concentrations in state 2 must be solved iteratively. One approach is to use a back-substitution method in which the new value of T_2 is guessed and new values of $c_{a,2}$, $c_{wv,2}$, and $c_{wl,2}$ are computed. These are then used to compute a new value of T_2 and iterations continue until a converged solution is reached. This method, in fact, does not converge well, and in some cases not at all, over the expected temperature and pressure ranges.

A modified Newton-Raphson technique was found to be more robust. Define an error function,

$$E = \frac{(\rho I)}{\rho_{nom}} - e_2 = \frac{(\rho I)}{\rho_{nom}} - [c_{a,2} C_{va} T + c_{wv,2} C_{vv} T - c_{wl,2} h_{vap}] \quad (16)$$

where (ρI) = current value of volume specific energy

ρ_{nom} = nominal fluid density for energy equation

The objective is to drive the error function to $E=0$ according to

$$T'_2 = T_1 - \frac{E}{\left(\frac{dE}{dT}\right)} = T_1 + \frac{E}{\left(\frac{de_2}{dT}\right)} \quad (17)$$

where T'_2 = new estimate for the final temperature

When there is liquid water present, the relative mass fractions of the vapor and liquid will change in opposite ways to meet the constant energy requirement. The derivative in Eq. 17 is given by

$$\begin{aligned} \frac{de_2}{dT} &= c_{a,2}C_{va} + c_{wv,2}C_{vv} + C_{vv}T_2 \frac{dc_{wv,2}}{dT} - h_{vap} \frac{dc_{wl,2}}{dT} \\ &= c_{a,2}C_{va} + c_{wv,2}C_{vv} + (C_{vv}T_2 + h_{vap}) \frac{dc_{wv,2}}{dT} \end{aligned} \quad (18)$$

The mass fractions of the air and total water are fixed and the mass fractions of the water vapor and liquid water are given by

$$c_{wv} = \begin{cases} c_{v,sat} & \text{for } c_w > c_{v,sat} \\ c_w & \text{for } c_w \leq c_{v,sat} \end{cases} \quad (19)$$

$$c_{wl} = c_w - c_{wv} \quad (20)$$

The mass fraction of water vapor under saturation conditions is

$$c_{v,sat} = \frac{x_{v,sat}}{(1 - x_{v,sat}) \frac{M_a}{M_w} + x_{v,sat}} \quad (21)$$

$$\text{where } x_{v,sat} = \frac{P_{v,sat}}{P} = \text{mole fraction of water vapor under saturation conditions} \quad (22)$$

$M_w = 18.01534$ gm/mol = molecular weight of water

$M_a = 28.9645$ gm/mol = molecular weight of air

The vapor pressure is obtained from well-established curve fit equations. For $273.15 \text{ K} < T < 647 \text{ K}$, the Keenan, Keys, Hill and Moore (1969) correlation is used,

$$P_{v,sat}(T_C) = 217.99 * \exp \left[\frac{0.01}{273.15 + T_C} (374.136 - T_C) \sum_{k=0}^7 F_k (0.65 - 0.01T_C)^k \right] \quad (23)$$

where $P_{v,sat}$ = water vapor saturation pressure in atm

T_C = temperature in °C

$F_0 = -741.9242$ $F_1 = -29.721$

$F_2 = -11.55286$ $F_3 = -0.8685635$

$F_4 = 0.1094098$ $F_5 = 0.439993$

$F_6 = 0.2520658$ $F_7 = 0.05218684$

For $223.15 \text{ K} < T < 273.15$, the Goff (1965) equation is used,

$$\log_{10}(P_{v,sat}(\theta)) = B_0 * (\theta - 1) + B_1 \log_{10}(\theta) + B_2 \left(1 - \frac{1}{\theta}\right) + B_3 \quad (24)$$

where $P_{v,sat}$ = water vapor saturation pressure in atm

$$\theta = \frac{273.16}{T}$$

T = temperature in K

$$B_0 = -9.096936 \quad B_1 = -3.56654$$

$$B_2 = 0.867817 \quad B_3 = -2.2195983$$

When there is liquid water present in the bulk flow, the relative amounts of the vapor and liquid will change in opposite directions but equal magnitude as the temperature is adjusted to meet the constant energy constraint. Consequently, the derivative of the vapor mass fraction is

$$\frac{dc_{wv,2}}{dT} = \frac{d}{dP_{v,sat}} \left(\frac{x_{v,sat}}{\left(1 - x_{v,sat}\right) \frac{M_a}{M_w} + x_{v,sat}} \right) \frac{dP_{v,sat}}{dT} = \frac{d}{dP_{v,sat}} \left(\frac{P_{v,sat} \frac{M_w}{M_a}}{P + P_{v,sat} \left(\frac{M_w}{M_a} - 1 \right)} \right) \frac{dP_{v,sat}}{dT}$$

$$\frac{dc_{wv,2}}{dT} = \frac{1}{P} \left(\frac{\frac{M_w}{M_a}}{\left(1 + x_{wv,sat} \left(\frac{M_w}{M_a} - 1 \right)\right)^2} \right) \frac{dP_{v,sat}}{dT} \quad (25)$$

The derivative of the saturation pressure is evaluated numerically based on the functions described in Eq. (23) and (24). Equation (25) is substituted into Eq. (18) which is in turn substituted into Eq. (17) to arrive at the new estimate of the cell temperature that satisfies Eq. (15).

It is found that a relaxation factor stabilizes the algorithm, so

$$T'_2 = T_1 + F_{relax} \frac{E}{\left(\frac{de_2}{dT}\right)} \quad (26)$$

where $0.5 < F_{relax} < 0.8$ is a suggested choice. The value of T'_2 is updated until the sequence converges to within an acceptable value, say 0.001 K. It has been found that 3-4 iterations is typical.

Radiation Module

The thermal radiation processes in nuclear waste repository drifts is that of radiative exchange between surfaces in an enclosure at relatively low temperature. This will be in the infrared range of the electromagnetic spectrum and is handled well by established engineering analysis of thermal radiation.

Assumptions

Several assumptions are made here to simplify the radiation model for the purposes of the intended analyses.

1. All surfaces are diffuse and gray.
The spectral characteristics of the surfaces are ignored by assuming that the surface emissivity is a uniform and constant value. Each surface can have a different value.
2. The gas is transparent to radiation.
The gas in the drifts will be a mixture of air and water vapor. Water vapor will absorb thermal radiation in several bands of wavelengths in the infrared part of the electromagnetic spectrum. These effects are neglected here as being too detailed considering the overall fidelity of the analysis.

Radiation Exchange Equation

Siegel and Howell (1992) show that the heat transfer by radiation between gray, diffuse surfaces in an enclosure is given by

$$\sum_{m=1}^{N_r} \left(\frac{\delta_{nm}}{\varepsilon_m} - F_{n-m} \frac{1 - \varepsilon_m}{\varepsilon_m} \right) \frac{Q_m}{A_m} = \sum_{m=1}^{N_r} \sigma F_{n-m} (T_n^4 - T_m^4) \quad (27)$$

where $\sigma = 5.67 \times 10^{-8}$ watt/(m²K⁴) = Stefan-Boltzmann constant

δ_{nm} = Kronecker delta

ε_m = surface emissivity

F_{n-m} = configuration factor for surface 'n' viewing surface 'm'

Q_m = heat transfer rate from surface via radiation (W)

A_m = surface area (m²)

T_m = temperature of surface (K)

N_r = number of surfaces active for radiation

In Eq. (27), it is seen that the computation of radiation heat transfer involves the use of a completely full matrix of coefficients on the left and right sides of Eq. (27) for the heat fluxes and temperatures, respectively. That is, the heat flux at an individual surface is directly dependent on the temperatures of all the other surfaces and its orientation to all other surfaces. This is unlike some other field problems where the variable of interest directly depends only on the conditions of neighboring points. For example, in conduction heat transfer analysis, the coefficient matrices in the discretized form of the conduction equation are banded, not full. Like

wise for the discretized forms of the fluid momentum equation and the stress-strain equation in CFD and FEA calculations.

FLOW-3D (or any CFD code for that matter) discretizes surfaces into many element or cells. If one defines each of these discrete elements as a single surface in Eq. (27), it can be seen that this will lead to huge full matrices that must be manipulated. Consequently, radiation surface elements are defined as groups of the individual surfaces that result from the discretization process. It is left to the user to decide how finely to resolve the surfaces for radiation heat transfer purposes.

With this in mind, the following nomenclature will be adopted. A cell is a finite volume of space defined by the FLOW-3D grid discretization scheme. FLOW-3D uses the term 'obstacle' to refer to solid objects that are in the flow domain. A radiation surface is a group of cells comprising an obstacle surface or a part of an obstacle surface defined by the user. The temperatures, T_m , are taken to be the average temperature over all the cells comprising the radiation surface m .

For incorporating radiation effects into FLOW-3D, we will assume that the surface temperatures are known from previous calculations at the current time step. Eq. (27) can be rearranged into the matrix equation representing a system of N equations with N unknowns,

$$\mathbf{GQ} = \mathbf{HT}_r \quad (28)$$

$$\mathbf{q}'' = \mathbf{G}^{-1}\mathbf{HT}_r$$

where \mathbf{q}'' = vector of N unknown heat fluxes

T_r = vector of N known surface temperatures (raised to the fourth power)

\mathbf{G} = N × N matrix of heat flux coefficients

\mathbf{H} = N × N matrix of temperature coefficients

The elements of the heat flux vector are

$$q_m'' = \frac{Q_m}{A_m} \quad (29)$$

The elements of the coefficient matrices are

$$G_{n,m} = \left(\frac{\delta_{nm}}{\varepsilon_m} - F_{n-m} \frac{1 - \varepsilon_m}{\varepsilon_m} \right) \quad (30)$$

$$H_{n,m} = (\delta_{nm} - F_{n-m})\sigma$$

The elements of the T_r vector are simply,

$$T_{r,m} = (T_m)^4 \quad (31)$$

Once the surface heat fluxes are known, they are incorporated into the overall solution procedure to simulate the effects of radiation heat transfer at a surface. There is not a method of directly incorporating the radiation heat transfer rates into the overall energy equations that are being solved for the solid and fluid materials. Instead, the surface element temperatures are adjusted prior to the next time step of the calculations. The temperature adjustment is defined as follows.

$$\Delta T_m = T_m + q_m'' \frac{\Delta t A_m}{\rho_m C_m V_m} \quad (32)$$

where Δt = time step

A_m = area of surface m

ρ_m = density of material in surface m

C_m = specific heat of material m

V_m = effective volume of the surface m

The volume, V_m , corresponds to a thin layer of material at the surface represented by a portion or a single grid cell at each discretized node on the surface.

The temperature adjustment, ΔT_m , is applied to all of the individual cells making up the radiation surface.

The remaining terms to be defined are as follows

ε_m = surface emissivity is provided by the user in the specifications for the radiation surfaces

ρ_m = obstacle density is provided by the user in the specifications for the obstacles

C_m = obstacle specific heat is provided by the user in the specifications for the obstacles

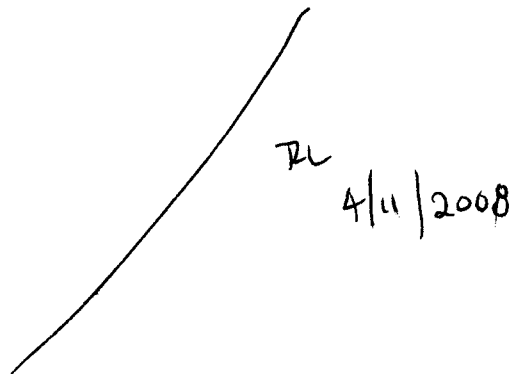
A_m = radiation surface area is computed from the FLOW-3D obstacle geometry as described below in the flowcharts

V_m = radiation surface effective volume is the solid volume portion of the FLOW-3D cells making up the radiation surface as described below in the flowcharts

F_{n-m} = radiation configuration factor is specified by the user in the problem input file or can be computed by the software module as described below.

Configuration Factors

Radiation configuration factors are defined by the geometric shapes of the surfaces and their respective orientation. Consider the differential areas, dA_m and dA_n , on two differential surfaces in three-dimensional space in Figure 3/3/06-2



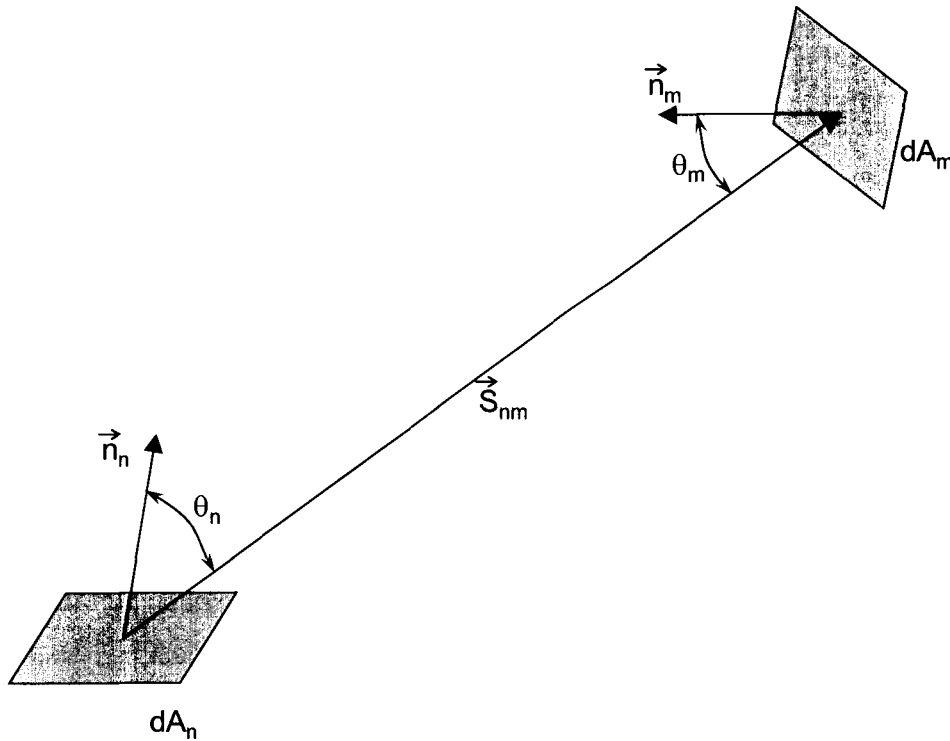


Figure 3/3/06-2. Configuration Factor Definition for 3-D Surfaces

The basis of the radiation configuration factor calculations is the defining equation for configuration factors [Siegel and Howell (1992)],

$$F_{n-m} = \frac{1}{A_n} \int_{A_n} \int_{A_m} \frac{\cos \theta_n \cos \theta_m}{\pi S^2} dA_m dA_n \tag{33}$$

$$\cos \theta_n = \frac{\vec{n}_n \cdot \vec{S}_{nm}}{|\vec{n}_n| |\vec{S}_{nm}|} \tag{34}$$

$$\cos \theta_m = \frac{\vec{n}_m \cdot \vec{S}_{mn}}{|\vec{n}_m| |\vec{S}_{mn}|} \tag{35}$$

$S = |\vec{S}_{nm}|$ = distance between differential elements on the surfaces

\vec{n}_n = unit normal vector for element dA_n

\vec{n}_m = unit normal vector for element dA_m

\vec{S}_{nm} = vector from element dA_n to dA_m

$\vec{S}_{mn} = -\vec{S}_{nm}$ = vector from element dA_m to dA_n

The elemental areas, dA_m and dA_n , are equated to the individual FLOW-3D cell on each surface and the integration is carried out by summing over all of the cells comprising radiation surfaces. To do this, the FLOW-3D obstacle is used to compute all the geometry and vector parameters as follows.

FLOW-3D uses Cartesian geometry for its obstacle definitions. In this framework, the surface normal vector components are

$$\vec{n} = \frac{A_{r,i,j,k} - A_{r-1,j,k}}{A_{i,j,k}} \vec{e}_x + \frac{A_{bk,i,j-1,k} - A_{bk,i,j,k}}{A_{i,j,k}} \vec{e}_y + \frac{A_{t,i,j,k} - A_{t,i,j,k-1}}{A_{i,j,k}} \vec{e}_z \quad (36)$$

where $A_{r,i,j,k}$ = projection of obstacle area in cell (i,j,k) in a plane normal to the x-axis

$A_{bk,i,j,k}$ = projection of obstacle area in cell (i,j,k) in a plane normal to the y-axis

$A_{t,i,j,k}$ = projection of obstacle area in cell (i,j,k) in a plane normal to the z-axis

$A_{i,j,k}$ = obstacle area in cell (i,j,k)

$\vec{e}_x, \vec{e}_y, \vec{e}_z$ = unit vectors in the three coordinate directions

Note that FLOW-3D uses the nomenclature of 'right' or 'r' to mean in the positive x-direction, 'back' or 'bk' to mean in the positive y-direction, and 'top' or 't' to mean in the positive z-direction.

The element-to-element vector, \vec{S}_{nm} , is computed as

$$\vec{S}_{nm} = \left[(x_{i,j,k})_m - (x_{i,j,k})_n \right] \cdot \vec{e}_x + \left[(y_{i,j,k})_m - (y_{i,j,k})_n \right] \cdot \vec{e}_y + \left[(z_{i,j,k})_m - (z_{i,j,k})_n \right] \cdot \vec{e}_z$$

The appropriate vector component and magnitude values are substitutes into Eq. (33), (34), and (35) to compute the kernel of the double integral. The calculation procedure for the integral kernel includes checks to ensure that the surface elements can actually 'see' each other.

First, if either of the two cosine terms is less than zero, then the surfaces cannot see each other and the kernel is set to zero for that particular pair of surface cells. Second, if there is an obstacle blocking the view, then the kernel is also zeroed out for that particular pair of cells.

This is determined by tracing a ray along the path of \vec{S}_{nm} . If any cell along this path has more than half its volume occupied by an obstacle, the kernel is zeroed out.

The kernel computations are repeated and summed over the lists of cells in each surface 'n' and 'm' to compute the integral.

For two dimensional problems the configuration integral is different than is shown in Eq. (33). Consider the configuration of two infinite differential strips in Figure 3/3/06-3.

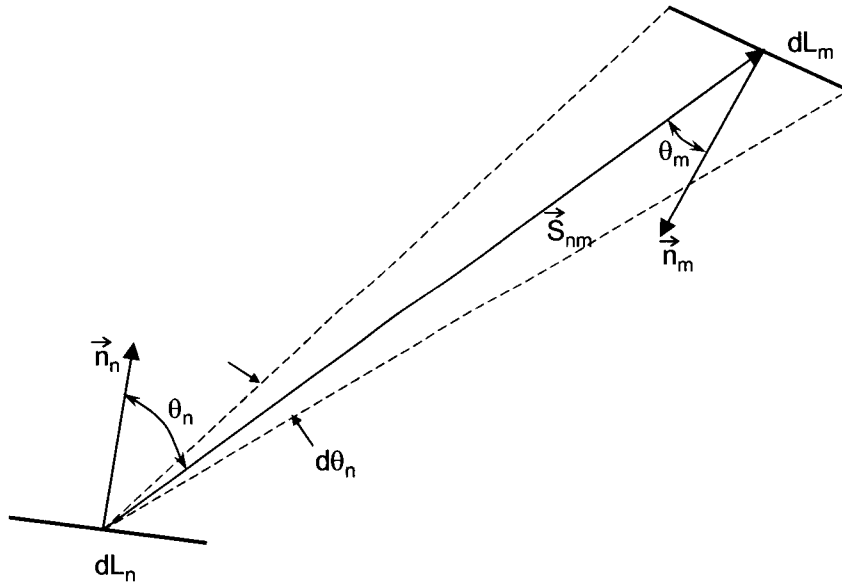


Figure 3/3/06-3. Configuration Factor Definition for 2-D Surfaces

The configuration factor between these two differential strips is given by

$$dF_{dn-dm} = \frac{1}{2} \cos \theta_n d\theta_n = \frac{1}{2} \frac{\cos \theta_n \cos \theta_m}{S} dL_m \tag{38}$$

where S = defined as above

Eq. (38) can be integrated to yield the required two-dimensional configuration factor,

$$F_{n-m,2D} = \frac{1}{L_n} \int_{L_n} \int_{L_m} \frac{\cos \theta_n \cos \theta_m}{2S} dL_m dL_n \tag{39}$$

In FLOW-3D, the out-of-plane dimension in 2-D problems is given a finite value. Making use of this, a single expression can be used to compute both the integrals in Eq. (33) and (39),

$$F_{n-m} = \frac{C_{F2D}}{A_n} \int_{A_n} \int_{A_m} \frac{\cos \theta_n \cos \theta_m}{D} dA_m dA_n \tag{40}$$

where

$$C_{F2D} = \begin{cases} 1 & \text{for 3D} \\ \Delta x^{-1} & \text{for 2D in } y - z \text{ plane} \\ \Delta y^{-1} & \text{for 2D in } x - z \text{ plane} \\ \Delta z^{-1} & \text{for 2D in } x - y \text{ plane} \end{cases} \tag{41}$$

$\Delta x, \Delta y,$ or Δz = out of plane thickness for the respective 2-D problem setup

$$D = \begin{cases} \pi S^2 & \text{for 3D} \\ 2S & \text{for 2D} \end{cases} \quad (42)$$

The reciprocity relation is used to eliminate the the need for integrating over all the surfaces,

$$F_{n-m} = \frac{A_m}{A_n} F_{m-n} \quad (43)$$

Equation (40) is the operational equation that is programmed into the software for radiation configuration factors.

Software Description

New software was written for implementing the moisture transport and thermal radiation models described above in the FLOW-3D computer program. To fully implement the moisture transport model, some existing subroutines in FLOW-3D were modified. The new and modified software are described in this section.

Code Development Environment

All of the software described here was developed in accordance with the guidelines provided by FLOW-3D for modifying existing code modifications and adding new subroutines. The description of these guidelines is not within the scope of this document but the pertinent aspects are included in the description of each subroutine. All of the software was written for use on a Windows XP platform using Compaq Visual FORTRAN Version 6.6c in accordance with the FLOW-3D guidelines.

It is expected that all of this software is compatible with other platforms and compilers with which FLOW-3D can be used with two exceptions. First, there is one new array added here (see subroutine NBR_INIT) that uses dynamic memory allocation. The declarations for dynamic memory allocation are compiler-specific. Second, some features of the OPEN statements used to handle the radiation heat transfer output file (see subroutine RAD_CALC) are compiler-specific. These lines of code will require alteration for use on other platforms or compilers. There could be other alterations necessary if the software is implemented on other platforms or with other compilers.

Software Implementation Overview

The standard FLOW-3D code allows the user to change some of the existing subroutines and add entire new subroutines for special models or simulation algorithms. The software described here falls in both categories. The modified subroutines are those involving property evaluations for high-humidity air/water mixtures and the preprocessor subroutine for communicating user-provided input specifications to the customized software. The subroutines that fall into this category are listed in Table 3/3/06-1.

Table 3/3/06-1. Modified FLOW-3D Subroutines and Specification Files for Implementing the Moisture Module and Radiation Module

Subroutine	Original File Name	New File Name	Description
e1cal	e1cal.f	e1cal_stg.f	Fluid mass-specific energy. This subroutine required modification to make the fluid energy a function of temperature and composition
rhocal	rhocal.f	rhocal_stg.f	Fluid density. This subroutine required modification to make the fluid density a function of temperature and composition.
prusrd	prusrd.f	prusrd_stg.f	Preprocessor input file reader. This subroutine was modified to define and initialize the appropriate variable for the moisture transport and radiation modules.
rusrd	rusrd.f	rusrd_stg.f	Simulation code file reader. This subroutine was modified to define and initialize the appropriate variable for the moisture transport and radiation modules.
n/a	cbusr.f	cbusr.f	COMMON block definitions for customized user inputs. The name of this file cannot be changed because it is part of INCLUDE statements throughout the software.
n/a	usrdat.f	usrdat.f	NAMelist definition for customized user inputs. The name of this file cannot be changed because it is part of INCLUDE statements throughout the software.

The modified sections of these files are straightforward and are not described in detail here. Instead, refer to the files themselves for a description of the changes.

New subroutines were created to implement the model equations described in the “Basic Equations” section. These new subroutines and modules are listed in Table 3/3/06-2. Note that the subroutine teval replaces the object module of the same name that is supplied with the basic FLOW-3D installation. The computations in the teval subroutine must be a mirror image of those in e1cal, but the source code for teval is not available. The version of teval used here is to be used only with the moisture transport module software. This subroutine has error trapping so that if the moisture module is not specified, then code execution is halted. This prevents the customized software being used in cases for which it is not designed.

Table 3/3/06-2. New FLOW-3D Subroutines and Specification Files

Subroutine	File Name	Description
qsadd	qsadd_pcg_rad.f	Links user-custom subroutines to the rest of FLOW-3D. This subroutine was patterned after the sample QSADD subroutine supplied in the standard code installation.
nbr_init	qsadd_pcg_rad.f	Initializes the cell type identifier array for use in PCG_INIT, PCG_CALC, RAD_INIT, and RAD_CALC.
pcg_rad_module	pcg_rad_module.f	Defines the array nbrobs array for dynamic memory allocation.
pcg_init	pcg_init_calc.f	Provides proper initialization and identification of obstacle surface cells for moisture transport calculations.
pcg_calc	pcg_init_calc.f	Performs calculations for water evaporation/condensation and moisture transport heat transfer at walls and the adjustments to the temperature and vapor concentrations to meet humidity constraints.
teval	teval_stg.f	Calculates the fluid temperature from the fluid energy and composition. The object module for this subroutine replaces the one provided in the standard FLOW-3D installation because the source code for the standard TEVAL routine is not made available to users.
rad_init	rad_init_calc.f	Provides proper initialization and identification of obstacle surface cells for radiation calculations and performs configuration factor calculations.
matinv	matinv_stg.f	Inverts the heat flux coefficient matrix in PCG_INIT for use in PCG_CALC. This matrix inversion is performed once only at the beginning of a simulation.
rad_calc	rad_init_calc.f	Computes the radiation heat transfer rates between surfaces and computes the wall temperature adjustments to meet the heat flux requirements

The overall program flow is very roughly described as follows. The steps performed by the main parts of FLOW-3D are shown *italics*.

1. *Read and process the problem setup specifications in the FLOW-3D prepin file.*
2. *Perform basic FLOW-3D setup and initialization.*
3. *Compute the fluid velocity field in accordance with the conservation of mass and momentum.*

4. *Compute the fluid energy, chemical species, and solid energy fields in accordance with the conservation of fluid thermal energy, chemical species, and solid object energy equations.*
5. *Call QSADD for customized routines*
6. In QSADD, for the first time step only.
Call PCG_INIT to set up the moisture transport module variables and data arrays that will remain constant. Call RAD_INIT to set up the radiation module variables and data arrays that will remain constant.
7. In QSADD, call PCG_CALC to compute the wall evaporation and condensation rates and the wall temperature changes in accordance with the conservation of mass and energy for the wall/fluid interface.
8. In QSADD, call RAD_CALC to compute the wall-to-wall radiation heat transfer rates in accordance with the radiation exchange equations. Adjust the wall surface temperature accordingly.
9. Return to main part of FLOW-3D
10. *Perform remaining end-of-time-step operations (e.g., updates, file operations)*
11. *Advance to new time step.*
12. *Repeat steps 3-12 until end of simulation is reached.*

The logic and coding for the subroutines listed in Table 3/3/06-2 are described in the following portions of this section to show how the model equations are implemented in the software.

Software Descriptions

The new software developed in this effort is described here through the use of pseudo-code. The portions of the FLOW-3D source code that are simply modified are not described here because those modifications are straightforward and that coding is sufficiently described within the affected subroutines.

Major Variables

The major variables used in this software modification are described in Table 3/3/06-3

Table 3/3/06-3. Major Variables in Moisture Transport and Radiation Modules

Variable	Subroutines	Standard/New	Description
ijk	all	Standard	Grid cell index and subscript
kajk	All	Standard	List of cells with a fluid/solid interface

kvjk	All	Standard	List of cells with a partial or full solid volume
vf	all	Standard	Fluid volume fraction of each cell
nbrobs	pcg_init, pcg_calc rad_init, rad_calc	New	Defines the direction in which a cell containing a solid material is with respect to the cell indicated by the subscript.
sclr	all	Standard	Holds fluid composition and auxiliary quantities
rhof	all	Standard	Nominal fluid density for energy equation
rhoe	all	Standard	Fluid volume specific energy
cv1	e1cal, teval,	Standard	Air specific heat
hvvap_stg	e1cal, teval, pcg_calc	New	Energy of vaporization of water
cvvap_stg	e1cal, teval,pcg_calc	New	Water vapor specific heat
npsrf_stg	pcg_init, pcg_calc	New	Number of surfaces active for phase change
imoist_stg	pcg_init, pcg_calc	New	Identifier for obstacles that are infinite sources/sinks of water
ilqonly_stg	pcg_init, pcg_calc	New	Identifier for obstacles that can condense water and re-evaporate only what has condensed
kajk_pcg	pcg_init, pcg_calc	New	List of subscripts in the moisture surface area array, waobs_pcg, and moisture surface cell index array, ijk_pcg
waobs_pcg	pcg_init, pcg_calc	New	Surface area of in cells that have moisture surfaces
ijk_pcg	pcg_init, pcg_calc	New	FLOW-3D ijk index for moisture surface cells
wvobs_pcg	pcg_init, pcg_calc	New	Solid volume of moisture surface cells
itw_pcg	pcg_init, pcg_calc	New	FLOW-3D index for the cell with the solid associated with this moisture surface cell
sclr	All	Standard	Stores the values of the species concentrations and stores bookkeeping values also
isvap_stg	pcg_init, pcg_calc	New	Total water concentration Computed by species diffusion, global value

isliq_stg	pcg_init, pcg_calc	New	Moisture transport mass flux at moisture surface cells Bookkeeping only, local value only
isrh_stg	pcg_init, pcg_calc	New	Relative humidity Bookkeeping only, local value only
istlq_stg	pcg_init, pcg_calc	New	Total mass transfer from start of simulation Global value
iswv_stg	pcg_init, pcg_calc	New	Water vapor concentration Computed from total water and saturation value
iswl_stg	pcg_init, pcg_calc	New	Liquid water as mist; computed from total water and vapor, Global value
tn	All	Standard	Fluid temperature
tw	All	Standard	Solid temperature
nrsrf_stg	rad_init, rad_calc	New	Number of radiation surfaces
radl_rad	rad_init, rad_calc	New	Radiation heat flux coefficient matrix and its inverse
rad_stg	rad_init, rad_calc	New	Information array for identifying radiation surfaces from obstacle cells surfaces
rcobs_rad	rad_init, rad_calc	New	Density-specific heat product for radiation surfaces
kajk_rad	rad_init, rad_calc	New	List of subscripts in the radiation area array, waobs_rad, and radiation surface cell index array, ijk_rad
waobs_rad	rad_init, rad_calc	New	Surface area of in cells that are part of radiation surfaces.
ijk_rad	rad_init, rad_calc	New	FLOW-3D ijk index for radiation surface cells
sa_rad	rad_init, rad_calc	New	Area of radiation surface
sv_rad	rad_init, rad_calc	New	Solid volume of radiation surface
cf_stg	rad_init, rad_calc	New	Radiation configuration factors
eps_stg	rad_init, rad_calc	New	Radiation surface emissivity

qflx_rad	rad_init, rad_calc	New	Radiation heat flux
radr_rad	rad_init, rad_calc	New	Right side of radiation heat flux equation

Module *pcg_rad_module*

This module contains only a specification statement for defining the dynamic memory allocation of the variable *nbrobs*. This is a two-dimensional array that is sized at $6 \times N_{cell}$, where N_{cell} is the total number of grid cells in the simulation. Dynamic memory allocation allows the dimensioned size of this array to vary depending in the size of the problem and precludes the need to recompile the code if the simulation requires an array larger than was anticipated for static allocation.

Subroutine *nbr_init*

This subroutine initializes and defines the array *nbrobs*. This subroutine is contained in the file *qsadd_pcg_rad.f*.

<u>nbr_init pseudocode</u>	
subroutine <i>nbr_init</i>	! No arguments required
<i>nbr_siz</i> =size(vf)	! Get the total number of grid cells from the vf array
Allocate the size of <i>nbrobs</i> to (6, <i>nbr_siz</i>)	! Set the size of <i>nbrobs</i> to the problem size
Initialize <i>nbrobs</i> (<i>m</i> , <i>ijk</i>) = 0	
For <i>nob</i> = 1 to <i>nobs</i>	
<i>mincel</i> = <i>kvjk</i> (<i>nob</i>)	! Get the range of cells with solid volume in them
<i>maxcel</i>	! for this obstacle
<i>ijk</i> = <i>ijkvob</i> (<i>m</i>)	
for <i>m</i> = <i>mincel</i> to <i>maxcel</i>	
<i>ijk</i> = <i>ijkvob</i> (<i>ijk</i>)	! Get the cell index for this obstacle cell location
Neighbor cell indexes from <i>mijk</i> , <i>pijk</i>	
Set <i>nbrobs</i> (1, <i>ipjk</i>), <i>nbrobs</i> (2, <i>imjk</i>).....	! Assign <i>nbrobs</i> for the neighbor cells
<i>nbrobs</i> (2, <i>ijpk</i>), <i>nbrobs</i> (3, <i>ijmk</i>),	! Left direction of the cell to the right, etc.
<i>nbrobs</i> (5, <i>ijkp</i>), <i>nbrobs</i> (6, <i>ijkm</i>) = <i>nob</i>	
End loop for <i>m</i> = <i>mincel</i> to <i>maxcel</i>	
End loop for <i>nob</i> = 1 to <i>nobs</i>	
return	

Subroutine *teval*

This subroutine computes the temperature corresponding to fluid volume specific energy and fluid composition. The code used here must be the reverse of the code used to compute the fluid energy in subroutine *e1cal* so that there is a one-to-one correspondence between energy and temperature for a specific combination of air concentration, water vapor concentration, and liquid water concentration. The *teval* object module contained in the standard FLOW-3D environment must be removed so that the object module created by compiling the code listed here replaces the standard version. This subroutine is in the file

named `teval_stg.f` to distinguish it from the standard version and as a reminder that a substitute version is being used.

teval pseudocode

```
function teval(ijk)                ! Cell index ijk is the required argument
if (isvap .le. 0) stop             ! If moisture model is not in use stop execution
ywv = sclr(ijk,isywv_stg)         ! Get the vapor and liquid concentration from
yw1 = sclr(ijk,isyw1_stg)         !      from the scalar array
ya = 1. - ywv - yw1               ! Compute the air concentration
rhomix = rhof                     ! Use the nominal density defined for energy calc's
teval = rhoe(ijk)/rhomix + yw1*hvvap_stg ! Temperature corresponding to this energy and
teval = teval/(ya*cv1+ywv*cvvap_stg) ! composition
return
```

Subroutine qsadd

This subroutine is the interface between the base FLOW-3D code and the software for the moisture transport and radiation modules and is based on the sample code provided in the standard FLOW-3D installation. This subroutine provides for the initialization requirements of these modules and calls the modules during FLOW-3D execution. This subroutine is in the file named `qsadd_pcg_rad.f` to distinguish it from the standard version.

qsadd pseudocode

```
subroutine qsadd                    ! No arguments are required
if (nrsrf_stg and isvap_stg <= 0) return ! Return if phase change and radiation are not active
if (cycle <=0) then                ! Call initialization routines for first cycle
  call nbr_init                    ! Initialize the nbrobs array
  if (nrsrf_stg .gt. 0) call rad_init ! Initialization for radiation calculations
  if (isvap_stg .gt. 0) call pcg_init ! Initialization for moisture transport calculations
endif
if (nrsrf_stg .gt. 0) call rad_calc ! Perform radiation calculations
if (isvap_stg .gt. 0) call pcg_calc ! Perform moisture transport calculations
return
```

Subroutine pcg_init

There are several information items that are extracted from the FLOW-3D data arrays and stored. These procedures need to be performed only once at the start of the code execution and are separated from the main part of the moisture transport module to make the main part of the module more efficient. This subroutine provides for the initialization requirements of the moisture transport module and is in the file named `pcg_init_calc.f`.

pcg_init pseudocode

```
subroutine pcg_init                ! No arguments are required
mpcg=0                            ! Initialize the surface cell index
npsrf_stg=0                        ! and surface counter
Do 2000 nob =1 to nob              ! Loop over all the obstacles
  if (imoist_stg(nob),ilqonly(nob) .ne. -nob) goto 2000 ! Skip if this obstacle is not moisture-active
  npsrf_stg=npsrf_stg+1           ! Increment the surface counter
```

```

rcobs_pcg(npsrf_stg) = rcobs(nob)           ! Set the rho-c product for this surface
implpcg(stg(npsrf_stg) = 'm' or 'l')       ! Define surface as infinite source/sink
                                             ! or limited-liquid
if (ilqonly(nob) .eq. -nob) implpcg_stg='l'
mincel=kajk(nob,nbl)                       ! Get the range of cells on the cell surface
maxcel=kajk(nob+1,nbl)-1                   ! surface ID array for this obstacle
mmm=0                                       ! Initialize counter for moisture surface cells
for m=mincel to maxcel                     ! Loop over all cells for this obstacle surface
  ijk-ijkobs(m)                            ! Get the ijk index for this obstacle surf. cell
  call inijk                                ! Get the individual i,j,k
  get the neighbor indexes, imjk, ipjk, etc....
  if (vf(ijk) .le. 1e-6) skip to end of do m loop ! Skip if this cell is fully blocked by obstacle
  compute vcell, vwall                     ! Total cell volume and solid-only volume
  mmm = mmm + 1                             ! Increment the moisture surface cell index
  mpcg = mpcg + 1                           ! Increment the moisture cell index limit
  if (mmm=1) kajk_pcg(npsrf_stg) = mpcg    ! Assign limits for the moisture cell index
  kajk_pcg(npsrf_stg+1) =- mpcg+1         ! ID array. kajk_pcg is analogous to kajk
  waobs_pcg(mpcg) = waobs(m)              ! Obstacle surface area for moisture surface cells
  ijk_pcg(mpcg) = ijk                     ! FLOW-3D cell index for this moisture surface cell
  ijktw = ijk                              ! Assume this cell actually has the surface inside it
  If this cell is fully open find fully closed neighbor ! Find the cell that has the wall obstacle and get
    ijktw = index of neighbor that is fully closed ! its index. This is used to get right wall temp. later
    vwall = solid volume of correct neighbor
  wvobs_pcg(mpcg)=vwall                    ! Store the solid volume associated with this surface
  itw_pcg(mpcg) = ijktw                   ! FLOW-3D index for solid for with this surface
End loop for m=mincel to maxcel
End Do 2000
return

```

Subroutine *pcg_calc*

This subroutine calculates the mass transfer at the wall and adjusts the wall surface cell temperatures to account for the heat of vaporization effects on the heat transfer in the walls. The mass transfer at walls is effectively the source term in the species diffusion equation. The adjustment of wall surface temperature is done here because there is no way for the user to directly assign the heat source terms in the solid energy equation in FLOW-3D. The *pcg_calc* subroutine is in the file named *pcg_init_calc.f*.

```

                                     pcg_calc pseudocode
subroutine pcg_calc                   ! No arguments are required
sclr(ijk,iii)=0   for iii=isliq_stg, isrh_stg, uslq_stg, 7, 8 ! Initialize the bookkeeping scalars to zero
for ipsrf = 1 to npsrf_stg           ! Loop over all moisture surfaces
  Get mincel,maxcel from kajk_pcg    ! Moisture surface cell ID limits for this surface
  for m = mincel to maxcel           ! Loop over all cells for this moisture surface
    ijk = ijk_pcg(m), ijktw-itw_pcg(m) ! FLOW-3D indexes for this fluid and wall cell
    call inijk                        ! Decode the i,j,k from index ijk
    compute imjk, ipjk, etc.....     ! Get the neighbor cell indexes
    skip the rest of m loop if cell is fully blocked ! Trap possible error condition
    vcell = fluid volume in cell
    vwall = wvobs_pcg(m)              ! Get the solid volume and surface area for this
    sa = waobs_pcg(m)                 ! cell on this moisture surface
    if (tw(ijktw).le. 0) stop with error message ! If wall temp is non-existent, this is error
    tfinal=teval(ijk), tinit=tfinal  ! Initialize cell temp according to current energy

```



```

twalli = tw(ijktw), twalf=twalli      ! Initialize the wall temp to current value
yvacti=sclr(ijk,isvap_stg)           ! Get current water mass fraction from previous sol'n
nitr_moist=0                          ! Initialize counter for wall calculations
Iterate until temp is converged or count=25 ! Loop for finding wall temp and water mass flux
  tnk = twalf or tfinal                ! Set the temp for humidity properties according
  compute sat. mass fraction at fluid or wall temp
  compute rhosat from mass fractions and temps ! Bulk density at saturation
  compute delmmax                       ! Mass required to bring entire cell to saturation
  compute delmdif                        ! Mass entering cell if diffusion from wall is limiting
  delm=min(delmmax,delmdif)             ! Choose lower mass flow
  compute new wall temp or fluid temp
  check if we're out of condensed water

  If not converged on new temp, loop again for max of 25 iterations
  delmrat=delm/rho(ijk)/vcell           ! New water mass as fraction
  sclr(ijk,isvap_stg)=(yvacti+delmrat)/(1+delmrat) ! New value of total water fraction
  sclr(ijk,isliq_stg)=sclr(ijk,isliq_stg)-delm/delt/sa ! Sum mass flux to get values from all wall neighbors
  sclr(ijk,istlq_stg)=sclr(ijk,istlq_stg)-delm ! Total liquid accumulation
  Update the FLOW-3D arrays for wall temp or fluid temp
End loop for m=mincel to maxcel
End loop for ipsrf = 1 to npsrf_stg
Loop over all real cells in flow domain that are not fully blocked
  yw=sclr(ijk,isvap_stg)
  nitr_moist2 = 0
  if (yw .le. yvsat .or. rhlim_stg .eq. 'n') skip calculations for RH shift
  Iterate until converged on new saturation temp for max of 25 iterations
    nitr_moist2 = nitr_moist2 + 1
    ed0 = rhoe(ijk)/rhof                ! Get the current value of mass-specific energy
    Get yvsat, saturation mass fraction
    ed1 = current value of energy        ! The energy derivative depends on whether
    d1edt = derivative of energy w.r.t. temp ! there is liquid present
    tfinal=tfsave+vapr1x*(ed0-ed1)/de1dt ! Newton-Raphson equation
  If not converged on new temp, loop again for max of 25 iterations
  Update vapor, liquid fractions, cell temp ! Get new conditions that satisfy constant energy
End loop over all real cells in flow domain that are not fully blocked
return

```

Subroutine rad_init

There are several information items that are extracted from the FLOW-3D data arrays and stored for radiation surface areas and volumes. These procedures need to be performed only once at the start of the code execution and are separated from the main part of the moisture transport module to make the main part of the module more efficient. This subroutine also computes the radiation configuration factors, forms and then inverts the heat flux coefficient matrix for use in computing the radiation heat fluxes. This subroutine is in the file named rad_init_calc.f.

rad_init pseudocode

```

subroutine rad_init
Loop to set radl_rad(k_r,j_r) = 0      ! Initialize the heat flux coefficients

mrad = 0                               ! Initialize the radiation cell index
For irsrf = 1 to nrsrf_stg             ! Transfer geometry limits to local variables
  xl_srf = rad_stg(2,irsrf)           ! for this loop
  xh_srf = rad_stg(3,irsrf)

```

```

etc. . . . For all geometry limits
a_rad = 0
v_rad = 0
For nob = 1 to nob
  if (rad_stg(1,irsrf) .eq. nob) then
    rcobs_rad(irsrf)=rcobs(nob)
    mincel=kajk(nob,nbl)
    maxcel=kajk(nob+1,nbl)
    mmm=0
    For m=mincel to maxcel
      if (cell within geometry limits) then
        vwall= compute obstacle volume
        if cell is empty, search the neighbor
          cells for the obstacle volume
          mmm = mmm+1
          mrad = mrad + 1
          if (mmm=1) kajk_rad(irsrf)=mrad
          kajk_rad(irsrf+1)=mrad+1
          waobs_rad(mrad) = waobs(m)
          a_rad = a_rad+waobs(m)
          v_rad = v_rad + vwall

          ijk_rad(mrad)=ijkw
        endif
      End loop for m=mincel to maxcel
    End loop for nob=1 to nob
  sa_rad(irsrf) = a_rad
  sv_rad(irsrf) = v_rad
End loop for irsrf=1 to nrsrf

```

! Initialize radiation surface area
! and volume for summing
! Loop over all obstacles to find the one
! for this radiation surface
! Get the $\square c$ product for this radiation surface
! Get the range of cells for all the surface
! cells for this obstacle. kajk is a FLOW3D
! Initialize radiation surface cell index
! Loop over all FLOW3D cells
! that make up the surface of this obstacle

! Increment the radiation cell counters

! Set the minimum and maximum cell
! indexes for this radiation surface

! Get the surface area for this cell
! Sum to get the total area for this surface
! Sum to get the total solid volume
! of surface cells
! Save the index for this rad surface cell

! Save the total area and volume for this
! radiation surface

Compute the configuration factors

Find the minimum grid cell spacing for defining the ray tracing step

```

For n=1 to nrsrf
  cf_stg(n,n)
  min_ns = kajk_rad(n)
  max_ns = kajk_rad(n+1)-1
  For m= n+1 to nrsrf
    min_ms = kajk_rad(m)
    max_ms = kajk_rad(m+1)-1
    sum = 0
    For nn= min_ns to max_ns
      ijk_m=ijk_rad(nn)
      compute all local FLOW-3D indexes
      compute the dAn vector and magnitude
      For mm=man_ms to max_ms
        ijk_n=ijk_rad(nn)
        compute all local FLOW-3D indexes
        compute the dAm vector and magnitude
        compute the vector Smm
        compute ns_ray, steps for ray tracing
        rqay = 1
        For nss=1 to ns_ray
          if cell volume fraction <10-6, ray=0
        End loop for nss = 1 to ns_ray

```

! Loop through all the surfaces
! Assume surface cannot see itself
! Get max and min indexes for
! cells in radiation surface 'n'
! Loop over surfaces m>n
! Get max and min indexes for
! cells in radiation surface 'm'
! Initialize the CF integral
! Loop over all cells in surface 'n'

! Get the cell surface area vector compinents
! and magnitude for element in 'n'
! Loop over all cells in surface 'm'

! Get the cell surface area vector compinents
! and magnitude for element in 'n'

! Trace along Smm to check for blockage
! Set factor to zero if blocked

```

costnn = from dot product of dAn and Smn
costmm = from dot product of dAm and Snm
denom=pi*Smn for 3-D ! Find the correct denominator for 2D or 3D
denom = 2*Smn for 2-D problem
sum=sum+ ray*costnn*costmm/denom ! Sum this contribution into the integral
End of loop for mm=min_ms to max_ms
End of loop for nn=min_ns to max_ns
cf2d = 1 for 3-D ! Find the correct factor for 2-D or 3-D
cf2d = 1.delx or 1/dely or 1/delz for 2-D
cf_stg(n,m) = cf2d*sum/sa_rad(n) !compute the CF for n-m
cf_stg(m,n) = cf_stg(n,m)*sa_rad(n)/sa_rad(m) ! Use reciprocity for CF of m-n
End of loop for m=n+1 to nrsrf
End of loop for n=1 to nrsrf

For n=1 to nrsrf
For m=1 to nrsrf !Compute all the heat flux coefficients
kdelta=0
if (m=n) kdelta = 1
radl_rad(m,n) = kdelta/eps_stg(m)-cf_stg(n,m)*(1-eps_syg(m))/eps_stg(m)
End of loop for m=1 to nrsrf
End of loop for n=1 to nrsrf

call matinv(radl_rad) ! Invert the matrix for calculations later

record the emissivities and configuration factors for review

end of subroutine rad_init

```

Subroutine *matinv*

This subroutine performs a matrix inversion by modified Gaussian elimination. This is a standard numerical technique and will not be described by pseudo-code here. This subroutine is called only by *rad_init*.

Subroutine *rad_calc*

This subroutine computes the radiation heat transfer rate from each radiation-active surface. The routine multiplies the inverted heat flux coefficient by the T^4 array to compute each radiation heat flux. The heat fluxes are used to adjust the surface cell temperatures to simulate the effects of the radiation heat flux as if it is a source or sink in the conduction energy equation for the affected cells. This subroutine is called only by *qsadd*.

rad_calc pseudocode

```

subroutine rad_calc
For irsrf = 1 to nrsrf ! Find the average temp radiation surfaces
ta_rad=0
mincel=kajk_rad(irsrf) !Min and max cell indexes for this surface
maxcel=kajk_rad(irsrf+1)-1
for m=mincel to maxcel
ijk=ijk_rad(m) !FLOW-3D ijk index for this cell

```

```

        compute the local neighbor index
        ta_rad=ta_rad+tw(ijk)*waobs_rad(m)           !Use area-weighted average
    End loop for m=mincel to maxcel
    tr_rad(irsrf)=ta_rad/sa_rad(irsrf)
end loop for irsrf=1 to nrsrf

For n=1 to nrsrf                                   !Compute the T^4 terms in the equations
    radr_rad(n)=0                                  !This is the RHS of model equation
    for m=1 to nrsrf                               ! The H*T terms
        kdelta=0
        if (m=n) kdelta = 1
        radr_rad(n) = radr(n)+(kdelta-cf_stg(n,m))*tr_rad(m)**4
    end loop for m=1 to nrsrf
end loop for n=1 to nrsrf

for n=1 to nrsrf                                   !Compute the radiation heat fluxes
    qflx_rad(n)=0
    for m=1 to nrsrf
        qflx_rad(n)=qflx(n)-radl_rad(n,m)*radr(m)   !Heat flux is positive away from surface
    end loop for m=1 to nrsrf
end loop for n=1 to nrsrf

prt_rad = 0                                       ! Record the heat fluxes for
if (t .ge. tnext) then                             ! processing later
    prt_rad=1
    tnext = tnext + pltdt                          ! Save on the FLOW-3D pltdt interval
    write t, qflx_rad
for n=1 to nrsrf                                   !Compute the delta-T for surfaces
    dtmp_rad=0                                     !that are not fixed temp
    if (rcobs_rad(n) .gt. 0) then
        dtmp_rad = qflx_rad(n)*sa_rad(n)*delt/
                    rcobs_rad(n)/sv_rad(n)

    mincel=kajk_rad(n)
    maxcel=kajk_rad(n+1)-1                         ! Apply the delta-T to all obstacle cells
    for m=mincel to maxcel                          ! in this surface
        ijktw=ijk_rad(m)
        tw(ijktw) = tw(ijktw) + dtmp_rad
    end loop for m=mincel to maxcel
end loop for n=1 to nrsrf

end subroutine rad-calc

```

Input/Output Instructions

This section describes the entries in the FLOW-3D input file, prepin, for using the moisture module and the radiation module. The modules can be activated independently or can be used together. The instructions provided here have been prepared under the assumption that the reader is an experienced FLOW-3D user and understands the material in the preceding sections of this entry. This section describes the input specifications required to use the modules in a simulation and the simulation results that are available for review and analysis.

Input Specifications

FLOW-3D uses a graphical user interface (GUI) for communicating the simulation specification to the compute program and for viewing the simulation results, but the user can also directly manipulate the text file that the FLOW-3D preprocessor reads. The GUI is not available for the user to modify; so, all parameters specific to the moisture module and radiation modules must be defined by editing the preprocessor input file. The preprocessor input file is organized into namelist blocks; so, the instructions for implementing these custom modules is organized according to the namelist blocks.

It is important to remember that the moisture transport module requires the use of SI units for all variables.

Moisture Module

The moisture module requires that the user provide values for 6 of the FLOW-3D namelist blocks.

\$xput

The user must specify

iusrd = 1
rmrho = 1 for pure diffusion problems
= 0 otherwise
rmrhoe = 1 for pure diffusion problems
= 0 otherwise

Setting iusrd=1 indicates that the USRDAT namelist block is to be read. The rmrho and rmrhoe settings dictate how mass diffusion affects the energy conservation equation.

\$props

The user must specify the fluid nominal properties for the simulation:

rhof = nominal mixture density for energy equation
mu1 = nominal mixture dynamic viscosity
cv1 = constant volume specific heat for dry air
thc1 = mixture thermal conductivity for energy equation

Note that only the value for the constant volume specific heat of dry air, cv1, is specific to air in this namelist block. The other values are nominal values. If the simulation shows that the mixture temperature is not close to that assumed for specifying the density, viscosity, and thermal conductivity, it may be necessary to begin the simulation with better estimates for these nominal values.

\$scalar

Currently, there are eight scalar variables used in the moisture transport module. These are described in Table 3/3/06-4

Table 3/3/06-4. FLOW-3D Scalar Variable Definitions for the Moisture Transport Module

FLOW-3D Scalar	Definition
1	<p>Total water mass fraction This is the only scalar that diffuses and advects. It is assumed that any liquid water in the bulk flow exists as a fog and moves as if it was vapor. Also, liquid water in the bulk flow does not coalesce or fall out of the flow domain.</p>
2	<p>Mass flux at wall The scalar is for bookkeeping only. This variable holds the current water mass flux at each wall surface cell for recording to the output data file so that the user can view/process the information. For example, total evaporation rate or condensation rate at a wall can be computed from data extracted from FLOW-3D and transferred to Excel. The scalar is undefined except for cells that contain a wall surface.</p>
3	<p>Relative humidity The scalar is for bookkeeping only. This variable holds the current value of relative humidity at all flow domain cells. This value is intended for visualization of the humidity during a simulation.</p>
4	<p>Net water mass transferred at wall The scalar is for bookkeeping only. This variable holds the total amount of water that has changed phase at a wall. The start-up transients are included; so, if a user wants to make use of this variable, it is suggested that a restart be accomplished once the start-up transients are completed.</p>
5	<p>Water vapor mass fraction The scalar is for bookkeeping only. This variable holds the mass fraction of the liquid water in each flow domain cell.</p>
6	<p>Liquid water mass fraction The scalar is for bookkeeping only. This variable holds the mass fraction of the water vapor in each flow domain cell.</p>
7	<p>Wall phase change calculation iterations The scalar is for bookkeeping only. This variable holds the current number of iterations required to converge on the vapor concentration and wall temperature at each wall surface cell. This is defined only for wall surface cells. This variable is useful for troubleshooting and can be used to aid in setting the relaxation factor vaprx.</p>
8	<p>Bulk flow humidity adjustment iterations The scalar is for bookkeeping only. This variable holds the current number of iterations required to converge on the vapor concentration and fluid temperature at each flow domain cell to satisfy the energy-humidity constraint. This variable is useful for troubleshooting and can be used to aid in setting the relaxation factor vaprx.</p>

The following lines are required for defining the moisture module parameters. Note that the variable `cmisc(1)` is the diffusion coefficient for the water vapor – air system. Also, the variable `rmisc` should be set to `rmisc=0`. This variable is for defining a Schmidt Number for diffusion in turbulent flows. All the flows likely to be encountered in waste repository drift flows will be driven by natural convection and will be mostly laminar.

```
nsc=8,
isclr(1)=3, cmisc(1)=0.26e-04, scltit(1)='Tot.Water', rmisc=0.,
isclr(2)=0, cmisc(2)=0., scltit(2)='Liq.Flux',
isclr(3)=0, cmisc(3)=0., scltit(3)='Rel.Hum',
isclr(4)=0, cmisc(4)=0., scltit(4)='Net.Liq',
isclr(5)=0, cmisc(5)=0., scltit(5)='Vap.Wat',
isclr(6)=0, cmisc(6)=0., scltit(6)='Liq.Wat',
isclr(7)=0, cmisc(7)=0., scltit(7)='Itr.Wall',
isclr(8)=0, cmisc(8)=0., scltit(8)='Itr.Mesh',
```

\$fl

The initial mass fractions for total water, water vapor, and liquid water (scalar #1, #5, and #6 defined in the `$scalar` namelist) are specified following the rules for these values in the FLOW-3D input file setup. Also, it is required that the user specify the initial uniform pressure in the flow domain.

The following is a sample input specification for specifying a uniform 100% RH at a pressure of 99.379 kPa and a temperature of 299.7 K

```
sclri(1)=0.022,
sclri(5)=0.022,
sclri(6)=0.,
presi=99379.,
```

Spatial variations of these values are also available. Consult the FLOW-3D User's Manual for specifying fluid regions.

\$temp

This namelist block is used to set the initial temperature distribution in the flow field. The values specified here must be coordinated with those in the `$fl` block. The following is a sample input specification to define a uniform temperature of 299.7 K in the entire flow domain.

```
ntmp=1,

tempi=299.7,
```

Spatial variations of the initial temperature values are also available. Consult the FLOW-3D User's Manual for specifying fluid temperature regions.

\$usrdat

This namelist block is where the user defines the main details of the moisture module parameters and the walls where moisture can evaporate/condense. All of the required variables in this block pertinent to the moisture module are defined in Table 3/3/06-5.

Table 3/3/06-5. FLOW-3D Scalar Variable Definitions for the Moisture Transport Module

Variable	Value	Units	Definition
istwtf_stg	'tw' (recommended)		Energy for phase change comes from wall
	'tf'	n/a	Energy for phase change comes from fluid
imoist_stg	depends on \$OBS definitions negative of the obstacle with this type of surface specification	n/a	Define obstacles that can condense and evaporate without limit. These are essentially saturated porous surfaces
	depends on \$OBS definitions negative of the obstacle with this type of surface specification	n/a	Define surfaces that can condense without limit. These surfaces can also re- evaporate the accumulated water only as long as it lasts.
ilqonly_stg			
isvap_stg	1*	n/a	Define scalar index for water concentration
isliq_stg	2*	n/a	Define scalar index for liquid flux
isrh_stg	3*	n/a	Define scalar index for relative humidity
istlq_stg	4*	n/a	Define scalar index for total liquid accumulation
isywv_stg	5*	n/a	Define scalar index for vapor water
isywl_stg	6*	n/a	Define scalar index for liquid water (mist)
hvvap_stg	2304900. **	joule/kg	Energy of vaporization of water
cvvap_stg	1370. **	joule/(kg*K)	Water vapor constant volume specific heat.
cvliq_stg	4186. **	joule/(kg*K)	Water liquid constant volume specific heat
rvap_stg	416. **	joule/(kg*K)	Gas constant for water vapor
rgas_stg	289. **	joule/(kg*K)	Gas constant for air
vaprlx_stg	0.8	n/a	Relaxation factor for phase change iterations
rhlm_stg	'y'		Limit relative humidity to 100 percent
	'n'	n/a	No limit to relative humidity

*Fixed value – DO NOT CHANGE. Required values for correct operation of software

**Values are specified according to the temperature range of the simulation.

Consider a simulation in which obstacles 1 and 2 have surfaces at which water can condense and can also serve as sources of unlimited evaporation. Obstacles 3, 5, and 7 have surfaces at which water can condense, but only the water that has condensed during the simulation is available for evaporation. For example, suppose that a cell on obstacle 3 has a temperature such that 1 mg of water condenses up to a time of 100 sec. At that time the temperature conditions change such that evaporation from that cell can take place. Evaporation will occur up to the time that the 1 mg of water is gone; then, evaporation will stop. The following are the values for imoist_stg and ilqonly_stg that are used to specify these surfaces for this example:

```
imoist_stg(1) = -1, remark='Define obstacles that can condense and evaporate',
imoist_stg(2) = -2, remark='  without limit',
ilqonly_stg(3) = -3, remark='Define surfaces that can condense without limit',
ilqonly_stg(5) = -5, remark='  but evaporate what has condensed since time=0',
```


ilqonly_stg(7) = -7,

Radiation Module

The radiation module requires that the user provide values for two of the FLOW-3D namelist blocks.

\$xput

The user must specify

iusrd = 1

in the XPUT namelist block. This value indicates that the USRDAT namelist block is to be read.

\$usrdat

This namelist block is used in conjunction with the obstacle definitions to specify active radiation surfaces. Table 3/3/06-6 lists the variables required to use the radiation module.

Table 3/3/06-6. FLOW-3D Scalar Variable Definitions for the Moisture Transport Module

Parameter	Definition
nrsrf_stg	Total number of radiation surfaces
eps_stg(n)	Emissivity of surface 'n'
rad_stg(1,n)	Obstacle number of that has radiation surface 'n'
rad_stg(2,n)	Lower x-limit of cells on radiation surface 'n'
rad_stg(3,n)	Upper x-limit of cells on radiation surface 'n'
rad_stg(4,n)	Lower y-limit of cells on radiation surface 'n'
rad_stg(5,n)	Upper y-limit of cells on radiation surface 'n'
rad_stg(6,n)	Lower z-limit of cells on radiation surface 'n'
rad_stg(7,n)	Upper z-limit of cells on radiation surface 'n'
iusrcf_stg	Source of radiation configuration factors =0 for configuration factors are computed by the code =1 for configuration factors are specified by the user
cf_stg(n,m)	User-specified value of radiation configuration factor, F_{n-m}

As an example of the inputs for this namelist block, consider the case of two obstacles. Obstacle #1 has a surface in the plane defined by $x=0.0$ that is to be active for radiation heat transfer. Obstacle #2 has a surface in the plane defined by $x=0.1$ that is to be active for radiation heat transfer. The overall grid is 2-D in the x-z plane and the flow domain is defined in the region $-0.12 < x < +0.12$, $-0.12 < z < +0.12$. Each surface has an emissivity of $\epsilon=0.9$. The following input values are required to define this scenario.

```
nrsrf_stg=2,
eps_stg(1) = .9,
rad_stg(1,1) = 1.,
rad_stg(2,1) = -0.004, rad_stg(3,1) = 0.004,
rad_stg(4,1) = -1., rad_stg(5,1) = 2.,
```

```
rad_stg(6,1) = 0.0, rad_stg(7,1) = 0.2,  
eps_stg(2) = .9,  
rad_stg(1,2) = 2.,  
rad_stg(2,2) = 0.096, rad_stg(3,2) = 0.104,  
rad_stg(4,2) = -1., rad_stg(5,2) = 2.,  
rad_stg(6,2) = 0.0, rad_stg(7,2) = 0.2,
```

Note that the box defined by the `rad_stg` array can be any size box that contains the cells in question. This box can contain other obstacle's cells, because only the cells from the desired obstacle are chosen.

Output Specifications

The FLOW-3D graphical user interface (GUI) is used to graphically display the many variables that are available from the simulation. Values such as temperature, pressure, water vapor concentration, and relative humidity can be presented in contour plots.

There is an important limitation of the GUI with respect to the moisture transport and radiation modules. The GUI cannot compute the heat transfer due to the phase change or due to radiation and the GUI is not available for modification to remove this limitation. The GUI will only compute the heat transfer due to conduction and convection from a surface to the fluid. It is left to the user to extract the variables that are needed to compute the energy transfer associated with mass transfer at walls in the moisture transport module.

In the case of the radiation module, a special file is created to record the radiation-specific input parameters and to record the values of the radiation heat transfer rates from each radiation surface. These heat transfer rates must be added to those computed by the FLOW_3D GUI and those computed by the user for the phase changes effects to arrive at the total heat transfer rate from a surface.

Moisture Module

All of the scalar variables defined above for the `$scalar` namelist block can be viewed as contour plots in the FLOW-3D graphics package. These values can be extracted into text files as well for post-processing by the user.

To compute the energy transfer via mass transfer, the user must extract the following values for the surfaces in question,

- Fluid temperature
- Wall temperature
- Heat transfer coefficient
- Liquid flux (special moisture transport module variable)

The heat flux due to conduction/convection at each cell is computed by the FLOW-3D post-processor. The energy transfer via phase change has to be computed outside of FLOW-3D such as by Excel.

Radiation Module

A special code section written into the subroutines rad_init and rad_calc record the pertinent output information from the radiation module. This file has the name check_rad.* where the '*' is the name of the problem being executed as is the case with the other problem-specific FLOW3D files. The information recorded in this file is the configuration factor matrix terms and the time history of the radiation heat transfer rates of the surfaces that are activated for radiation.

END OF ENTRY FOR 3/3/06

STG

3/14/06

STG

This entry records the results of a validation test of the modified FLOW-3D software described in the entry for 3/3/06. This exercise is a repeated/corrected analysis of the conduction/diffusion problem described the 5/16/05 entry. The analysis preseted in the entry of 5/16/05 did not account for the presenceof water mist tha occurs if the relative humidity exceeds 100%.

Consider the gap between two flat surfaces. One surface is held at a temperature of 320 K and the other is at 280 K. The gap contains a mixture of air and water vapor at a total pressure of 1 atm. Each surface is saturated with water and the plates are large enough that one-dimensional heat conduction and molecular diffusion can be assumed. The plates are oriented so that there is no macroscopic motion of the air/vapor mixture. The temperature profile and water concentration in the gap are to be determined.

Two cases are considered here. In the first case, it is assumed that there is no limit to the concentration of water vapor in the air/vapor mixture. In the second case, the mixture is limited to a humidity of 100% in accordance with the thermodynamics of air/water mixtures. The theoretical analyses is described in the Mathcad Version 11.2a file

1-D_Heat_Mass_Phase-Change.mcd

A listing of this file is in the following 13 pages.

One dimensional Conduction and Mass Diffusion

Introduction

This document describes an analytical solution to the combined heat transfer and mass diffusion in a volume of two chemical species. The opposing surfaces are maintained at different temperatures and the surfaces are liquids of one species. The other species is not condensable. The heat transfer is by conduction only through the mixture.

The general equations of the two diffusion processes are presented and a specific example is investigated. The specific example is that of water vapor diffusing through air between one liquid surface held at 320 K and the other at 280 K.

This specific example serves as a validation test case for the software written to incorporate a moisture transport model into the computer program FLOW-3D.

References

The equations describing this process are given in

Bird, R. B., Stewart, W. E., Lightfoot, E. N., *Transport Phenomena*, John Wiley and Sons, New York, 1960. Specifically, Sections 17 and 18 of this text provide the general equations and examples.

General Case

Consider the scenario described in Figure 1.

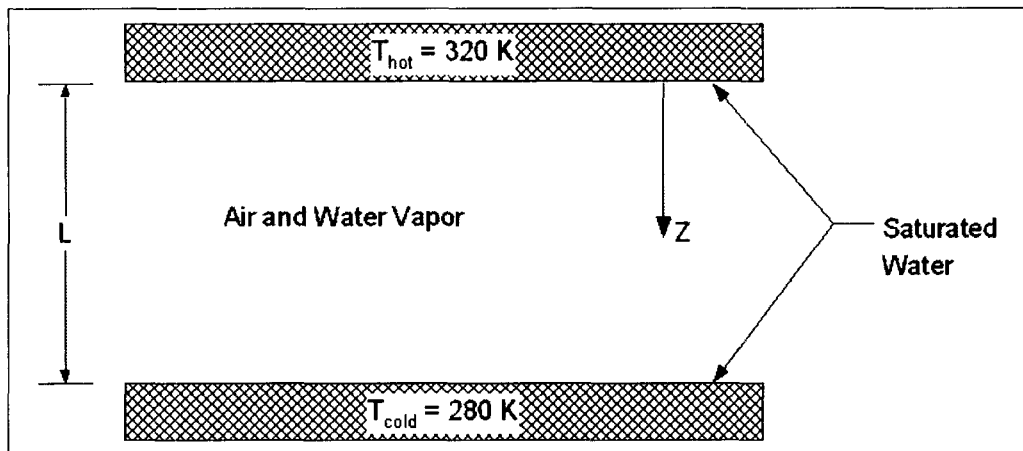


Figure 1. 1-D Conduction and Diffusion in a Mixture

A mixture of air and water vapor is maintained between two surfaces which are at two different temperatures. Each surface is able to maintain a liquid water interface that is in equilibrium with the water vapor adjacent to the wall. In this scenario we will assume that there is no advection of the bulk mixture and we will neglect the Soret and Dufour effects.

The water vapor evaporates from the hot surface, diffuses through the mixture and condenses on the cold surface. The air is incondensable and is stationary. We will also assume that the bulk properties of the mixture are uniform and constant.

Bird, Stewart and Lightfoot show that the energy flux at any point in the z-direction is given by

$$e(z) = -k_{\text{mix}} \left(\frac{d}{dz} T \right) + (h_a \cdot N_a + h_w \cdot N_w) \quad \text{Eq. 1}$$

where	e	energy flux, watt/m²
	k_{mix}	mixture thermal conductivity, watt/(m²*K)
	T	local mixture temperature, K
	h_a	mole-specific enthalpy of air, joule/mol
	N_a	mole flux of air, mol/(sec*m²)
	h_w	mole specific enthalpy of water vapor, joule/mol
	N_w	mole flux of water, mol/(sec*m²)

At steady conditions the energy flux is constant,

$$\frac{d}{dz} e(z) = 0 \quad \text{Eq. 2}$$

Bird, Stewart and Lightfoot also show that the mole flux of vapor in this 1-D model is described by

$$N_w = -\rho_{\text{mix}} D_{\text{va}} \left(\frac{d}{dz} x_w \right) + x_w \cdot N_{\text{mix}}$$

where	N_w	mole flux of water vapor, mol/(sec*m²)
	ρ_{mix}	mole density of the mixture, mol/m³
	D_{wa}	diffusion coefficient for water vapor and air, m²/sec
	x_w	mole fraction of water vapor, mol/(total moles)
	N_{mix}	mole flux of mixture, mol/(sec*m²)

The mole flux of the mixture is

$$N_{\text{mix}} = N_w + N_a \quad \text{Eq. 3}$$

but the air is stationary; so, $N_{\text{mix}} = N_w$ Eq. 4 becomes

$$N_w = -\frac{\rho_{\text{mix}} D_{\text{wa}}}{1 - x_w} \frac{d}{dz} x_w \quad \text{Eq. 4}$$

Again, under steady conditions, the mole flux of vapor is constant,

$$\frac{d}{dz} N_w = 0 \quad \text{Eq. 5}$$

The concentration profile $x_w(z)$ prescribed by Eq. 4 and Eq. 5 can be solved analytically independent of the energy equation. The resulting water concentration profile and mole flux can then be used to find the temperature profile from Eq. 1 and Eq. 2

Start by substituting Eq. 4 into Eq. 5 and rearranging

$$\frac{d}{dz} \left(-\frac{\rho_{\text{mix}} D_{\text{wa}}}{1 - x_{\text{w}}} \frac{d}{dz} x_{\text{w}} \right) = 0$$

$$\frac{d}{dz} \left[\frac{1}{(1 - x_{\text{w}})} \frac{d}{dz} x_{\text{w}} \right] = 0$$

with the boundary conditions

$$x_{\text{w}} = x_{\text{w},L} \quad \text{at} \quad z=L$$

$$x_{\text{w}} = x_{\text{w},0} \quad \text{at} \quad z=0$$

Integrate twice to yield

$$-\ln(1 - x_{\text{w}}) = C_1 \cdot z + C_2$$

Applying the boundary conditions provides the required solution

$$\frac{1 - x_{\text{w}}}{1 - x_{\text{w},0}} = \left(\frac{1 - x_{\text{w},L}}{1 - x_{\text{w},0}} \right)^{\frac{z}{L}}$$

or

$$x_{\text{w}} = 1 - (1 - x_{\text{w},0}) \cdot \left(\frac{1 - x_{\text{w},L}}{1 - x_{\text{w},0}} \right)^{\frac{z}{L}}$$

Eq. 6

The vapor mole flux can be obtained by substituting Eq. 6 back into Eq. 4,

$$N_{\text{w}} = \frac{\rho_{\text{mix}} D_{\text{wa}}}{L} \cdot \ln \left(\frac{1 - x_{\text{w},L}}{1 - x_{\text{w},0}} \right)$$

Eq. 7

The water vapor mole flux is defined relative to the coordinate system. The value is negative in Eq. 7 if the high concentration surface (i.e. the hot surface) is at $z=L$ (instead of as shown) because the mole flux would be in the negative z -direction in that case.

Also, note that the concentration profile is nonlinear. This is a result of the overall flow of vapor through the domain.

Now we can return to the energy equation to solve for the temperature profile. Accounting for the fact that there is no flow of air, $N_g = 0$, Eq. 1 can be rewritten as

$$e(z) = -k_{\text{mix}} \left(\frac{d}{dz} T \right) + h_w \cdot N_w \quad \text{Eq. 8}$$

Substitute Eq. 8 into Eq. 2 and assume that the thermal conductivity is constant,

$$\frac{d}{dz} \left[-k_{\text{mix}} \left(\frac{d}{dz} T \right) \right] + \frac{d}{dz} (h_w \cdot N_w) = -\frac{d}{dz} \left(\frac{d}{dz} T \right) + \frac{N_w}{k_{\text{mix}}} \frac{d}{dz} (h_w) = 0 \quad \text{Eq. 9}$$

There are two scenarios to consider:

1. There is no limit to the water vapor concentration. This would be similar to the case of two gases such as nitrogen and oxygen that are completely miscible at typical room temperature conditions. Conversely, we could assume that the water can exist in a supersaturated state at temperatures below the dewpoint. In this case, the relative humidity will be more than 100%.
2. The water vapor will condense if the temperature is less than the dewpoint. In this case, we will make the additional assumption that the condensed water is a 'fog' that has the same diffusion coefficient as the vapor so that the diffusion solution above for the total water content is still valid.

Solution 1. No limit to the vapor concentration (no condensate can form)

This is the scenario treated by Bird, Stewart, and Lightfoot as Example 18.5-1. In this case, the ideal gas assumption is taken for the entire water content and the enthalpy is given by

$$h_w = C_{pw}(T - T_0) \quad \text{Eq. 10}$$

Here, we have taken the temperature at $z=0$ as the reference point for the enthalpy and the specific heat of the water vapor. C_{pw} is constant. Furthermore, if we make the substitution,

$$T' = \frac{T - T_0}{T_L - T_0} \quad \text{Eq. 11}$$

Eq. 9 simplifies to

$$-(T_L - T_0) \frac{d}{dz} \left(\frac{d}{dz} T' \right) + \frac{N_w \cdot C_{pw}}{k_{\text{mix}}} \cdot (T_L - T_0) \frac{d}{dz} (T') = -\frac{d}{dz} \left(\frac{d}{dz} T' \right) + \frac{N_w \cdot C_{pw}}{k_{\text{mix}}} \frac{d}{dz} T' = 0 \quad \text{Eq. 12}$$

The boundary conditions are

$$\begin{aligned} T' &= 0 & \text{at} & \quad z = 0 \\ T' &= 1 & \text{at} & \quad z = L \end{aligned}$$

Eq. 12 is rearranged to show the first integration step

$$\int \frac{1}{\left(\frac{d}{dz} T' \right)} d \left(\frac{d}{dz} T' \right) = \int \frac{N_w \cdot C_{pw}}{k_{\text{mix}}} dz \quad \text{Eq. 13}$$

$$\ln \left(\frac{d}{dz} T' \right) = \frac{N_w \cdot C_{pw}}{k_{\text{mix}}} \cdot z + C_3$$

$$\ln\left(\frac{d}{dz} \Gamma\right) = \frac{N_w \cdot C_{pw}}{k_{mix}} \cdot z + C_3$$

$$\frac{d}{dz} \Gamma = C_3 \cdot \exp\left(\frac{N_w \cdot C_{pw}}{k_{mix}} \cdot z\right)$$

Eq. 14

Now integrate again,

$$\Gamma = C_3 \cdot \frac{k_{mix}}{N_w \cdot C_{pw}} \cdot \exp\left(\frac{N_w \cdot C_{pw}}{k_{mix}} \cdot z\right) + C_4$$

Eq. 15

Applying the boundary condition at $z=0$ yields

$$C_4 = -C_3 \cdot \frac{k_{mix}}{N_w \cdot C_{pw}}$$

Simplify Eq. 15,

$$\Gamma = C_3 \cdot \frac{k_{mix}}{N_w \cdot C_{pw}} \cdot \left(1 - \exp\left(\frac{N_w \cdot C_{pw}}{k_{mix}} \cdot z\right)\right)$$

Eq. 16

Applying the boundary condition at $z=L$ yields

$$C_3 = \frac{1}{\frac{k_{mix}}{N_w \cdot C_{pw}} \cdot \left(1 - \exp\left(\frac{N_w \cdot C_{pw}}{k_{mix}} \cdot L\right)\right)}$$

Simplify Eq. 16,

$$\Gamma = \frac{T - T_0}{T_L - T_0} = \frac{1 - \exp\left(\frac{N_w \cdot C_{pw}}{k_{mix}} \cdot z\right)}{1 - \exp\left(\frac{N_w \cdot C_{pw}}{k_{mix}} \cdot L\right)}$$

Eq. 17

or,

$$T(z) = T_0 + (T_L - T_0) \cdot \frac{1 - \exp\left(\frac{N_w \cdot C_{pw}}{k_{mix}} \cdot z\right)}{1 - \exp\left(\frac{N_w \cdot C_{pw}}{k_{mix}} \cdot L\right)}$$

Eq. 18

As with the concentration profile, the temperature distribution is nonlinear. The extent of the deviation from a linear profile is governed by the fluid properties and the mole flux.

Solution 2. Condensate forms

This scenario is derived originally here. In this case, the expression for the enthalpy of the water must account for condensation. The total enthalpy of the water is

$$N_w \cdot h_w = (N_v + N_l) \cdot h_w = (N_v \cdot h_v + N_l \cdot h_l) \quad \text{Eq. 19}$$

where

N_v mole flux of vapor water

N_l mole flux of liquid water

$h_v = C_{pw} \cdot (T - T_0)$ specific enthalpy of vapor using temperature at $z=0$ as the reference

$h_l = h_v - h_{vap}$ specific enthalpy of liquid

h_{vap} heat of vaporization

Substitute these into Eq. 19 to derive an expression for the specific enthalpy of the water

$$\begin{aligned} N_w \cdot h_w &= N_v \cdot h_v + N_l \cdot h_v - N_l \cdot h_{vap} = N_w \cdot h_v - (1 - N_v) \cdot h_{vap} \\ h_w &= h_v - \left(1 - \frac{N_v}{N_w}\right) \cdot h_{vap} = C_{pw} \cdot (T - T_0) - \left(1 - \frac{x_v}{x_w}\right) \cdot h_{vap} \end{aligned} \quad \text{Eq. 20}$$

x_v mole fraction of the water vapor with respect to the total number of moles

x_w mole fraction of all the water, solved above in Eq. 6

Substitute Eq. 1 and Eq. 20 into Eq. 2,

$$\frac{d}{dz} \left(\frac{d}{dz} T \right) - \frac{N_w}{k_{mix}} \cdot \left[C_{pv} \cdot \left(\frac{d}{dz} T \right) + h_{vap} \cdot \left[\frac{d}{dz} \left(\frac{x_v}{x_w} \right) \right] \right] = 0 \quad \text{Eq. 21}$$

Consider the derivative of the vapor concentration term

$$\frac{d}{dz} \left(\frac{x_v}{x_w} \right) = -\frac{x_v}{x_w^2} \cdot \left(\frac{d}{dz} x_w \right) + \frac{1}{x_w} \cdot \left(\frac{d}{dz} x_v \right)$$

Using Eq. 6 above for the total water concentration in the first term and the chain rule on the second term,

$$\frac{d}{dz} \left(\frac{x_v}{x_w} \right) = \frac{P_{v.sat}}{P_{tot}} \cdot \frac{1}{x_w^2} \cdot \frac{1 - x_w}{L} \cdot \ln \left(\frac{1 - x_{w,0}}{1 - x_{w,L}} \right) + \frac{1}{x_w \cdot P_{tot}} \cdot \left(\frac{d}{dT} P_{v.sat} \right) \cdot \left(\frac{d}{dz} T \right) \quad \text{Eq. 22}$$

Substitute this back into Eq. 21, we arrive at the equation,

$$\frac{d}{dz} \left(\frac{d}{dz} T \right) + g_1(T, z) \cdot \left(\frac{d}{dz} T \right) + g_2(T, z) = 0 \quad \begin{array}{ll} T = T_C & \text{at } z=0 \\ T = T_H & \text{at } z=L \end{array} \quad \text{Eq. 23}$$

where T temperature

$$g_1(T, z) = -\frac{N_w}{k_{mix}} \left[C_{pv} + \frac{h_{vap}}{x_w(z) \cdot P_{mix}} \left(\frac{d}{dT} P_{v,sat} \right) \right] \tag{Eq. 24}$$

$$g_2(T, z) = -\frac{N_w}{k_{mix}} \cdot h_{vap} \cdot \frac{P_{v,sat}(T)}{P_{tot}} \cdot \frac{1}{(x_w(z))^2} \cdot \frac{1 - x_w(z)}{z_H} \cdot \ln \left(\frac{1 - x_{w,0}}{1 - x_{w,L}} \right) \tag{Eq. 25}$$

Note that if $h_{vap} = 0$, we can recover Eq. 12 where condensation was ignored.

Note especially that Eq. 23 is a nonlinear differential equation for which it may be difficult to find a general solution, given the fact that the functions $g_1(T, z)$ and $g_2(T, z)$ are nonlinear expressions themselves. So, propose a numerical solution as follows.

The first derivative of temperature at the solution points is discretized as

$$\left(\frac{dT}{dz} \right)_k = \frac{0.5 \cdot (T_{k+1} + T_k) - 0.5 \cdot (T_k + T_{k-1})}{0.5 \cdot (z_{k+1} + z_k) - 0.5 \cdot (z_k + z_{k-1})} = \frac{T_{k+1} - T_{k-1}}{z_{k+1} - z_{k-1}} = \frac{T_{k+1} - T_{k-1}}{2 \cdot \Delta z} \tag{k = index of solution vector}$$

The second derivative at the solution points is now given as

$$\left[\frac{d}{dz} \left(\frac{dT}{dz} \right) \right]_k = \frac{\frac{T_{k+1} - T_k}{z_{k+1} - z_k} - \frac{T_k - T_{k-1}}{z_k - z_{k-1}}}{0.5 \cdot (z_{k+1} + z_k) - 0.5 \cdot (z_k + z_{k-1})} = \frac{T_{k+1} - (2 \cdot T)_k + T_{k-1}}{\Delta z^2}$$

So, at the solution point, k , the algebraic approximation to the D.E. is

$$\frac{T_{k+1} - (2 \cdot T)_k + T_{k-1}}{\Delta z^2} + g_1(T_k, z_k) \cdot \frac{T_{k+1} - T_{k-1}}{2 \cdot \Delta z} + g_2(T_k, z_k) = 0$$

Gather terms

$$\left(1 + g_1(T_k, z_k) \cdot \frac{\Delta z}{2} \right) \cdot T_{k+1} - 2 \cdot T_k + \left(1 - g_1(T_k, z_k) \cdot \frac{\Delta z}{2} \right) \cdot T_{k-1} = -g_2(T_k, z_k) \cdot \Delta z^2$$

$$T_{new, k} = \frac{\left(1 + g_1(T_k, z_k) \cdot \frac{\Delta z}{2} \right) \cdot T_{k+1} + \left(1 - g_1(T_k, z_k) \cdot \frac{\Delta z}{2} \right) \cdot T_{k-1} + g_2(T_k, z_k) \cdot \Delta z^2}{2}$$

Solve this by successive relaxation:

1. Assume a temperature distribution; e.g., linear or provided by Eq. 18 above for $h=0$
2. Compute the functions $g_1(T_k)$, $g_2(T_k)$ using the concentration profile and the known saturation pressure functions
3. Compute a new T_k for $2 \leq k \leq N-1$. Check for convergence.
4. Back-substitute the 'new' temperature values into the 'old' array
5. Repeat as necessary

Example Calculation

The example calculations are for the case described in Figure 1 where

$$\begin{aligned} T_L &:= 280\text{-K} & T_0 &:= 320\text{-K} \\ L &:= 0.2\text{-m} & & \text{Distance between surfaces} \\ P_{\text{mix}} &:= 1\text{-atm} & & \text{Total pressure of mixture, assumed uniform, constant} \end{aligned}$$

To define the saturation vapor pressure of water vapor, use the correlation of Keenan, Keyes, Hill, and Moore,

$$\begin{aligned} F_0 &:= -741.9242 & F_2 &:= -11.55286 & F_4 &:= 0.1094098 & F_6 &:= 0.2520658 \\ F_1 &:= -29.721 & F_3 &:= -0.8685635 & F_5 &:= 0.439993 & F_7 &:= 0.05218684 \end{aligned}$$

$$FP_{\text{vsat}}(T) := 217.99 \cdot \exp\left[\frac{0.01}{T} \cdot [374.136 - (T - 273.15)] \cdot \sum_{k=0}^7 [F_k \cdot [0.65 - 0.01 \cdot (T - 273.15)]^k]\right] \text{atm}$$

The material properties of water vapor and air are taken from handbook values,

$$\begin{aligned} R_{\text{gas}} &:= 8.3143 \frac{\text{joule}}{\text{mol}\cdot\text{K}} & & \text{Universal gas constant} \\ MW_{\text{w}} &:= 18.01534 \frac{\text{gm}}{\text{mol}} & MW_{\text{a}} &:= 28.9645 \frac{\text{gm}}{\text{mol}} & & \text{Molecular weights} \\ C_{\text{pw}} &:= 1820 \frac{\text{joule}}{\text{kg}\cdot\text{K}} \cdot MW_{\text{w}} & & \text{Mole-specific heat of water vapor} \\ h_{\text{vap}} &:= 2442300 \frac{\text{joule}}{\text{kg}} \cdot MW_{\text{w}} & & \text{Heat of vaporization in the temperature range of interest} \\ D_{\text{wa}} &:= 2.6 \cdot 10^{-5} \frac{\text{m}^2}{\text{sec}} & & \text{Species diffusion coefficient for air - water vapor} \\ k_{\text{mix}} &:= 0.026 \frac{\text{watt}}{\text{m}\cdot\text{K}} & & \text{Mixture thermal conductivity, assumed constant} \end{aligned}$$

The vapor conditions adjacent to each surface are now computed

$$\begin{aligned} P_{\text{v.sat.0}} &:= FP_{\text{vsat}}\left(\frac{T_0}{\text{K}}\right) & P_{\text{v.sat.0}} &= 1.054 \times 10^4 \text{ Pa} & P_{\text{v.sat.L}} &:= FP_{\text{vsat}}\left(\frac{T_L}{\text{K}}\right) & P_{\text{v.sat.L}} &= 991.24 \text{ Pa} \\ x_{\text{w.sat.0}} &:= \frac{P_{\text{v.sat.0}}}{P_{\text{mix}}} & x_{\text{w.sat.0}} &= 0.104 & x_{\text{w.sat.L}} &:= \frac{P_{\text{v.sat.L}}}{P_{\text{mix}}} & P_{\text{mix}} &= 1.01325 \times 10^5 \text{ Pa} \end{aligned}$$

Assume a linear temperature profile to estimate the average temperature for computing the bulk molar density

$$T_{\text{avg}} := 0.5 \cdot (T_0 + T_L) \quad \rho_{\text{mix}} := \frac{P_{\text{mix}}}{R_{\text{gas}} \cdot T_{\text{avg}}} \quad \text{Bulk molar density}$$

$$x_w(z) := 1 - (1 - x_{w,\text{sat},0}) \left[\frac{(1 - x_{w,\text{sat},L})}{1 - x_{w,\text{sat},0}} \right]^{\frac{z}{L}}$$

Vapor concentration profile

$$N_w := \frac{\rho n_{\text{mix}} D_{wa}}{L} \ln \left(\frac{1 - x_{w,\text{sat},L}}{1 - x_{w,\text{sat},0}} \right)$$

Mol flux of vapor

$$N_w = 5.283 \times 10^{-4} \frac{\text{mol}}{\text{m}^2 \text{ s}}$$

$$N_w \cdot MW_w = 9.518 \times 10^{-6} \frac{\text{kg}}{\text{m}^2 \text{ s}}$$

Mass flux of vapor

Consider scenario 1 above where the vapor is allowed to be supersaturated and there is no condensate in the air/vapor mixture. The temperature profile is given by Eq. 18.

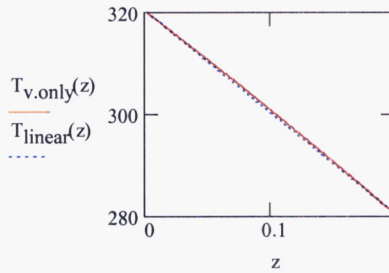
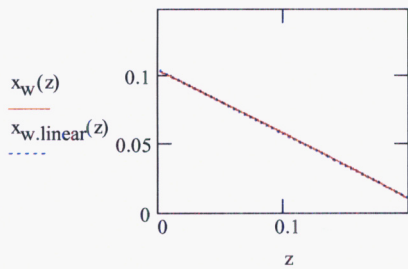
$$T_{v,\text{only}}(z) := T_0 + (T_L - T_0) \frac{\left(1 - \exp \left(\frac{N_w \cdot C_{pw}}{k_{\text{mix}}} \cdot z \right) \right)}{\left(1 - \exp \left(\frac{N_w \cdot C_{pw}}{k_{\text{mix}}} \cdot L \right) \right)}$$

Temperature profile where vapor is not allowed to condense

$$x_{w,\text{linear}}(z) := x_{w,\text{sat},0} + \frac{z}{L} \cdot (x_{w,\text{sat},L} - x_{w,\text{sat},0})$$

$$T_{\text{linear}}(z) := T_0 + \frac{z}{L} \cdot (T_L - T_0)$$

Linear profiles for reference

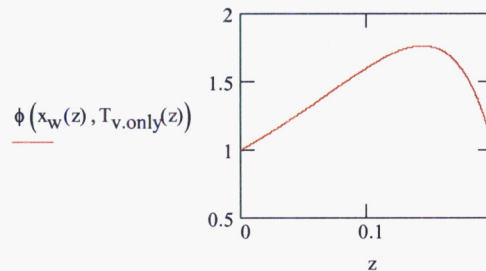


The concentration and temperature distributions are very nearly linear for this example case.

$$\phi(x, T) := \frac{x \cdot P_{\text{mix}}}{FP_{\text{vsat}} \left(\frac{T}{K} \right)}$$

Relative humidity

The relative humidity is greater than 100% except at the two boundaries. The model equations are based on the ideal gas assumption and do not allow for the possibility that the partial pressure of one component can exceed its saturation vapor pressure as a result of the temperature distribution.



Now consider the case in which condensation in the air/vapor mixture is allowed. The procedure outlined above for numerically solving the nonlinear differential equation for the temperature distribution will be followed.

First, define the points to form the difference equations.

$$N_{\text{intervals}} := 80 \quad k := 0..N_{\text{intervals}} \quad z_k := \frac{k}{N_{\text{intervals}}} \cdot L \quad \Delta z := z_1 - z_0$$

We will need to define the derivative of pressure with respect to the temperature. This can be obtained by differentiating the Keenan, Keyes, Hill, and Moore equation above.

$$dPdT(T) := \frac{FPvsat(T)}{K} \left[\begin{aligned} & \left[\frac{-0.01}{T^2} \cdot [374.136 - (T - 273.15)] \cdot \sum_{k=0}^7 \left[F_k \cdot [0.65 - 0.01 \cdot (T - 273.15)]^k \right] \right] \dots \\ & + \left[\frac{0.01}{T} \cdot (-1) \cdot \sum_{k=0}^7 \left[F_k \cdot [0.65 - 0.01 \cdot (T - 273.15)]^k \right] \right] \dots \\ & + \frac{0.01}{T} \cdot [374.136 - (T - 273.15)] \cdot \sum_{k=1}^7 \left[F_k \cdot k \cdot [0.65 - 0.01 \cdot (T - 273.15)]^{k-1} \cdot (-0.01) \right] \end{aligned} \right]$$

The functions, g_1 and g_2 in Eq. 23 can now be expressed as Mathcad functions

$$g_1(T, z) := -\frac{N_w}{k_{\text{mix}}} \left(C_{pw} + \frac{h_{\text{vap}}}{x_w(z) \cdot P_{\text{mix}}} \cdot dPdT\left(\frac{T}{K}\right) \right)$$

$$g_2(T, z) := \frac{N_w}{k_{\text{mix}}} \cdot h_{\text{vap}} \cdot \frac{FPvsat\left(\frac{T}{K}\right)}{P_{\text{mix}}} \cdot \frac{1}{(x_w(z))^2} \cdot \frac{1 - x_w(z)}{L} \cdot \ln\left(\frac{1 - x_{w,\text{sat}.0}}{1 - x_{w,\text{sat}.L}}\right)$$

The successive relaxation solution described above is programmed as an iterative Mathcad procedure. The inputs to the procedure are an initial guess at the temperature profile at all points in the solution interval and the maximum number of iterations allowed. The output is the solution for the temperature profile for all the points $1 \leq k \leq N_{\text{intervals}}$. The output solution vector at index '0' is the number of iterations required to converge to the tolerance specified inside the procedure (a convergence tolerance of 10^{-5} K is suggested). If the maximum allowed number of iterations is achieved, the procedure is terminated and this maximum value is inserted at $k=0$ in the solution array. It is assumed that the temperature at $k=0$ is a specified boundary condition; so, the user can replace the value after viewing the number of iterations executed in the procedure. If the maximum is reached, then the maximum value should be increased until a converged solution can be obtained.

```

T_iter_new(T_old, n_iter) :=
  nlast ← rows(T_old) - 2
  nn ← 0
  while nn ≤ nlast + 1
    T_temp_nn ← T_old_nn
    T_new_nn ← T_old_nn
    nn ← nn + 1
  n ← 0
  while n < n_iter
    nn ← 1
    maxerr ← -1000 · K
    while nn ≤ nlast
      g1p ← 1 + g1(T_new_nn, z_nn) ·  $\frac{\Delta z}{2}$ 
      g1m ← 1 - g1(T_new_nn, z_nn) ·  $\frac{\Delta z}{2}$ 
      g2 ← g2(T_new_nn, z_nn) ·  $\Delta z^2$ 
      T_new_nn ← T_temp_nn · (1 - F_relax) + F_relax ·  $\frac{g1p \cdot T_{new_{nn+1}} + g1m \cdot T_{new_{nn-1}} + g2}{2}$ 
      nn ← nn + 1
    nn ← 1
    while nn ≤ nlast
      err ← |T_new_nn - T_temp_nn|
      maxerr ← if(err > maxerr, err, maxerr)
      T_temp_nn ← T_new_nn
      nn ← nn + 1
    if maxerr < 10-5 · K
      T_new_0 ← n · K
      return T_new
    n ← n + 1
  T_new_0 ← n · K
  return T_new

```

Start the solution process with the temperature profile for the non-condensation scenario above

$$T_{old_k} := T_{v,only}(z_k)$$

Set the maximum number of iterations.

$$maxit := 2000$$

A relaxation factor is hardwired into the procedure as an aid in speeding up the convergence. $F > 1$ speeds up the convergence most of the time. $F < 1$ makes the solution more stable.

$$F_{relax} = 1.7$$

Now obtain the approximate solution to the differential equation.

$$T_{vwfog_j} := T_{iter_new}(T_{old, maxit-1})$$

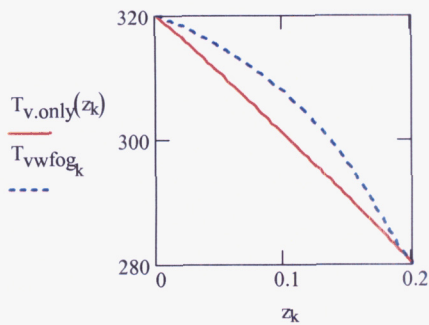
$$N_{iter2} := \frac{T_{vwfog_j}}{K} \quad N_{iter2} = 383$$

Iterations required for the solution

$$T_{vwfog_j} := T_{old_0}$$

Recover the temperature at the boundary

Compare the solutions to the two scenarios.



When condensation is allowed the temperature is greater than in the case where condensation is not allowed. This is due to the condensation releasing energy which will result in an increase in the temperature of the mixture to satisfy the conservation of energy requirement.

DL
4/11/2008

Process the data for export to Excel via a text file.

kk := 1,3..79

$$a_{\frac{kk+1}{2},0} := \frac{z_{kk}}{m} \quad \text{z-position. m}$$

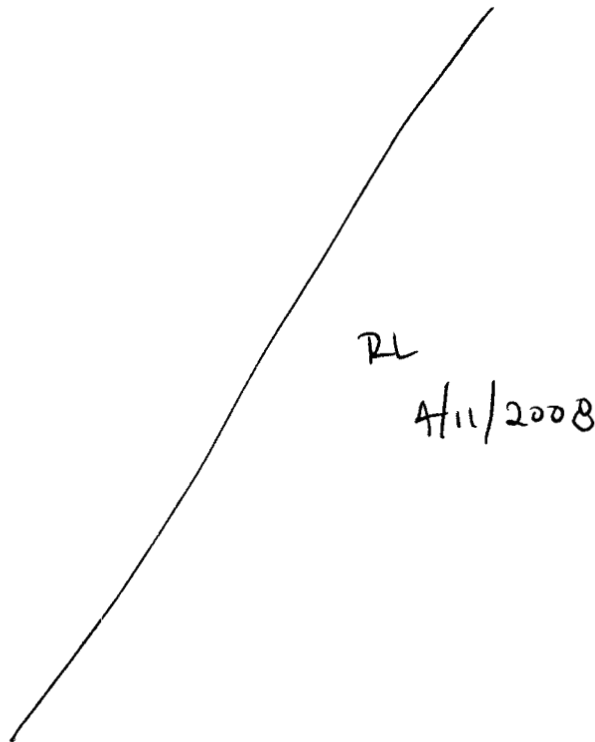
$$a_{\frac{kk+1}{2},1} := \frac{T_{v,only}(z_{kk})}{K} \quad \text{Temperature for no condensation, K}$$

$$a_{\frac{kk+1}{2},2} := x_w(z_{kk}) \quad \text{Total water mole fraction}$$

$$a_{\frac{kk+1}{2},3} := \frac{T_{vwfog_{kk}}}{K} \quad \text{Temperature when water condensation is allowed,}$$

$$a_{\frac{kk+1}{2},4} := \frac{FP_{vsat}\left(\frac{T_{vwfog_{kk}}}{K}\right)}{P_{mix}} \quad \text{Water vapor mole fraction}$$

WRITEPRN("1-D_Heat_Mass.txt") := a



The theoretical analysis is based on the assumption that the total water concentration distribution is the same regardless of the limitations of the relative humidity. This assumption is consistent with the assumption made for the moisture module added to the FLOW-3D software. There is however, a marked difference between the temperature contours depending on the presence of the mist or fog as a result of the limiting of the relative humidity to 100%. When water condenses to meet the 100% RH limit, the temperature of the mixture is greater than when the water is allowed to remain all vapor in violation of the 100% RH limit. Condensed water has a lower energy than in the vapor state; so, if some water condenses to maintain the vapor pressure at the saturation condition, the temperature of the entire mixture must increase so that the total energy is constant.

Two FLOW-3D simulations were prepared corresponding to the theoretical analyses described above. For the first scenario in which the relative humidity is allowed to exceed 100% the FLOW-3D input file is

```
prepin.p-r_noliq_1-D_FULL_40_diff_Twall
```

For the second scenario in which the relative humidity is limited to a maximum of allowed to exceed 100% the FLOW-3D input file is

```
prepin.p-r_1-D_FULL_40_diff_Twall
```

The listings of these files are on the following 4 pages.

Both of these files make use of the custom moisture module software described in this notebook's entry for 3/3/06.

The analytical predictions described above and the FLOW-3D simulation results were all copied into the Excel spreadsheet

```
1-D_Mass_Heat.xls
```

for graphing and comparison. The spreadsheet was used to convert the water mass fractions predicted by the FLOW-3D simulation into mole fractions for comparing the analytical results.

Listing of file prepin.p-r_noliq_1-D_FULL_40_diff_Twall

1-D Combined Heat Conduction and Vapor Diffusion, Right=320K, Left=280K DIFFUSION ONLY
5-12-05

Tests the transport of vapor and thermal energy by concentration gradients only.
This file allows RH to be greater than 100; i.e., does not allow
liquid to condense in the interior of the fluid as a fog.

\$xput

```

remark='units are SI',
itb=0,      remark='No sharp interface',
ifvis=0,    remark='Laminar flow ',
ifenrg=2,   remark='Solve energy equation, 1st order',
ifrho=1,    remark='Temperature dependent density',
ihtc=2,     remark='Evaluate heat transfer and solve solid conduction equation',
rmrhoe=1.,  remark='Density diffusion term coefficient',
rmrho=1.,   remark='Energy diffusion term coefficient',
            remark=' RMRHO, RMRHOE REQUIRED FOR VAPOR TRANSPORT MODEL ',
iusrd=1,    remark='Flag for reading the USRDAT section',
delt=1.e-4, remark='Initial time step',
pltdt=20.,  remark='Interval to save field variables in FLSGRF',
sprtdt=1.,  remark='Interval for status prints to screen',
hpltdt=5.,  remark='Interval to save history variables in FLSGRF',
twfin=1000., remark='Simulation end time',

```

\$end

\$limits

\$end

\$props

```

units='si',
rho=1.097,  remark='Bulk nominal density for energy equation',
mul=2.e-05, remark='Bulk nominal dynamic viscosity density for momentum equation',
cv1=717.,   remark='Const. Vol. Specific heat of dry air',
thc1=0.026, remark='Bulk nominal thermal conductivity for energy equation',
thexf1=0.003235, remark='Approximate thermal expansion coefficient',
tstar = 295.5, remark='Reference temperature for thermal expansion coefficient',
            remark='THEXF1, TSTAR not needed for vapor transport model',
            remark='but are used in some parts of code to set up simulation',

```

\$end

\$scalar

```

remark='Scalar 1 is the diffusing/advecting water',
remark='Scalar 2 (non-diffusing) is for storing the surface phase change flux values',
remark='Scalar 3 (non-diffusing) is for storing the relative humidity values',
remark='Scalar 4 (non-diffusing) is for storing the net surface liquid accumulation',
remark='Scalar 5 (non-diffusing) is for storing the vapor water concentration',
remark='Scalar 6 (non-diffusing) is for storing the liquid water (mist) concentration',
remark='Scalar 7 (non-diffusing) is for storing the calculation iterations for the',
remark='  wall mass flux',
remark='Scalar 8 (non-diffusing) is for storing the calculation iterations for the',
remark='  mist pahse change in the fluid interior',
remark='Scalars 7 and 8 are helpful in tuning the value of vaprlx if necessary',
nsc=8,
isclr(1)=3, cmisc(1)=0.26e-04, scltit(1)='Tot.Water', rmisc=0.,
isclr(2)=0, cmisc(2)=0., scltit(2)='Liq.Flux',
isclr(3)=0, cmisc(3)=0., scltit(3)='Rel.Hum',
isclr(4)=0, cmisc(4)=0., scltit(4)='Net.Liq',
isclr(5)=0, cmisc(5)=0., scltit(5)='Vap.Wat',
isclr(6)=0, cmisc(6)=0., scltit(6)='Liq.Wat',
isclr(7)=0, cmisc(7)=0., scltit(7)='Itr.Wall',
isclr(8)=0, cmisc(8)=0., scltit(8)='Itr.Mesh',

```

\$end

\$bcdata

```

    wl=2, wr=2,    remark='Walls at left and right',
    wf=1, wbk=1,  remark='Symmetry at front, back, bottom, and top fopr 1-D',
    wb=1, wt=1,
    tbc(1)=320., tbc(2)=280., remark=' Set boundary temps equal to obstacles',
$end

$mesh
    px(1)=-0.005, py(1)=0., pz(1)=0.,
    px(2)=0.205, py(2)=1., pz(2)=1.,
    nxcelt=42, nycelt=1, nzcelt=1,
$end

$obs
    avrck=-3.1,
    nobk =2,
    remark='Obstacle 1. Left hot wall',
    xl(1)=-0.02, xh(1)=0.0,
    twobs(1,1)=320.,
    remark='Obstacle 2. Cold right wall',
    xl(2)= 0.2, xh(2)=0.22,
    twobs(1,2)=280.,
$end

$f1
    remark=' No vapor in fluid for initial condition',
    sclri(1)=0.0,
    presi=101325.,
$end

$bf
$end

$temp
    remark=' 300 K initial temperature',
    ntmp=1,
    tempi=300.,
$end

$motn
$end

$grafic
$end

$parts
$end

$usrdat
    istwtf_stg='tw',    remark='Energy for phase change comes from wall',
    imoist_stg(1) = -1, remark='Define obstacles that can condense and evaporate',
    imoist_stg(2) = -2, remark=' without limit',
    isvap_stg = 1,    remark='Define scalar index for water concentration DO NOT CHANGE',
    isliq_stg = 2,    remark='Define scalar index for liquid flux DO NOT CHANGE',
    isrh_stg = 3,    remark='Define scalar index for rel. hum. DO NOT CHANGE',
    istlq_stg = 4,    remark='Define scalar index for tot.liq. accum. DO NOT CHANGE',
    isyvw_stg = 5,    remark='Define scalar index for vapor water DO NOT CHANGE',
    isywl_stg = 6,    remark='Define scalar index for liquid water (mist) DO NOT CHANGE',
    hvvap_stg=2304900., remark='Heat of vaporization',
    cvvap_stg=1370.,  remark='Water vapor specific heat (const. vol.)',
    cvliq_stg=4186.,  remark='Water liquid specific heat (const. vol.)',
    rvap_stg=416.,    remark='Gas constant for water vapor',
    rgas_stg=289.,    remark='Gas constant for air',
    vaprlx_stg=0.8,   remark='Relaxation factor for phase change iterations',
    rhlim_stg='n',    remark='No limit for rel humidity',
$end

```

End of listing of file prepin.p-r_noliq_1-D_FULL_40_diff_Twall

Listing of file prepin.p-r_1-D_FULL_40_diff_Twall

1-D Combined Heat Conduction and Vapor Diffusion, Right=320K, Left=280K DIFFUSION ONLY
5-12-05

Tests the transport of vapor and thermal energy by concentration gradients only.
This file allows liquid to form in the interior of the fluid as a fog.

\$xput

```

remark='units are SI',
itb=0,      remark='No sharp interface',
ifvis=0,    remark='Laminar flow ',
ifeng=2,    remark='Solve energy equation, 1st order',
ifrho=1,    remark='Temperature dependent density',
ihtc=2,     remark='Evaluate heat transfer and solve solid conduction equation',
rmrhoe=1.,  remark='Density diffusion term coefficient',
rmrho=1.,   remark='Energy diffusion term coefficient',
            remark=' RMRHO, RMRHOE REQUIRED FOR VAPOR TRANSPORT MODEL ',
iusrd=1,    remark='Flag for reading the USRDAT section',
delt=1.e-4, remark='Initial time step',
pltdt=20.,  remark='Interval to save field variables in FLSSGRF',
sprtdt=1.,  remark='Interval for status prints to screen',
hpltdt=5.,  remark='Interval to save history variables in FLSSGRF',
twfin=1000., remark='Simulation end time',

```

\$end

\$limits

\$end

\$props

```

units='si',
rhof=1.097, remark='Bulk nominal density for energy equation',
mul=2.e-05, remark='Bulk nominal dynamic viscosity density for momentum equation',
cvl=717.,   remark='Const. Vol. Specific heat of dry air',
thcl=0.026, remark='Bulk nominal thermal conductivity for energy equation',
thexf1=0.003235, remark='Approximate thermal expansion coefficient',
tstar = 295.5, remark='Reference temperature for thermal expansion coefficient',
            remark='THEXF1, TSTAR not needed for vapor transport model',
            remark='but are used in some parts of code to set up simulation',

```

\$end

\$scalar

```

remark='Scalar 1 is the diffusing/advecting water',
remark='Scalar 2 (non-diffusing) is for storing the surface phase change flux values',
remark='Scalar 3 (non-diffusing) is for storing the relative humidity values',
remark='Scalar 4 (non-diffusing) is for storing the net surface liquid accumulation',
remark='Scalar 5 (non-diffusing) is for storing the vapor water concentration',
remark='Scalar 6 (non-diffusing) is for storing the liquid water (mist) concentration',
remark='Scalar 7 (non-diffusing) is for storing the calculation iterations for the',
remark='    wall mass flux',
remark='Scalar 8 (non-diffusing) is for storing the calculation iterations for the',
remark='    mist phase change in the fluid interior',
remark='Scalars 7 and 8 are helpful in tuning the value of vaprlx if necessary',
nsc=8,
isclr(1)=3, cmisc(1)=0.26e-04, scltit(1)='Tot.Water', rmisc=0.,
isclr(2)=0, cmisc(2)=0., scltit(2)='Liq.Flux',
isclr(3)=0, cmisc(3)=0., scltit(3)='Rel.Hum',
isclr(4)=0, cmisc(4)=0., scltit(4)='Net.Liq',
isclr(5)=0, cmisc(5)=0., scltit(5)='Vap.Wat',
isclr(6)=0, cmisc(6)=0., scltit(6)='Liq.Wat',
isclr(7)=0, cmisc(7)=0., scltit(7)='Itr.Wall',
isclr(8)=0, cmisc(8)=0., scltit(8)='Itr.Mesh',

```

\$end

\$bcdata

```

wl=2, wr=2,    remark='Walls at left and right',

```

```

wf=1, wbk=1,   remark='Symmetry at front, back, bottom, and top fopr 1-D',
wb=1, wt=1,
tbc(1)=320., tbc(2)=280.,   remark=' Set boundary temps equal to obstacles',
$end

$mesh
px(1)=-0.005,   py(1)=0.,   pz(1)=0.,
px(2)=0.205,   py(2)=1.,   pz(2)=1.,
nxcelt=42,      nycelt=1,   nzcelt=1,
$end

$obs
avrck=-3.1,
nobs =2,
remark='Obstacle 1. Left hot wall',
xl(1)=-0.02, xh(1)=0.0,
twobs(1,1)=320.,
remark='Obstacle 2. Cold right wall',
xl(2)= 0.2,   xh(2)=0.22,
twobs(1,2)=280.,
$end

$f1
remark=' No vapor in fluid for initial condition',
sclri(1)=0.0,
presi=101325.,
$end

$bf
$end

$temp
remark=' 300 K initial temperature',
ntmp=1,
tempi=300.,
$end

$motn
$end

$grafic
$end

$parts
$end

$usrdat
istwtf_stg='tw',   remark='Energy for phase change comes from wall',
imoist_stg(1) = -1, remark='Define obstacles that can condense and evaporate',
imoist_stg(2) = -2, remark='      without limit',
isvap_stg = 1,     remark='Define scalar index for water concentration DO NOT CHANGE',
isliq_stg = 2,     remark='Define scalar index for liquid flux           DO NOT CHANGE',
isrh_stg = 3,      remark='Define scalar index for rel. hum.             DO NOT CHANGE',
istlq_stg = 4,     remark='Define scalar index for tot.liq. accum.      DO NOT CHANGE',
isywv_stg = 5,     remark='Define scalar index for vapor water          DO NOT CHANGE',
isywl_stg = 6,     remark='Define scalar index for liquid water (mist) DO NOT CHANGE',
hvvap_stg=2304900., remark='Heat of vaporization',
cvvap_stg=1370.,   remark='Water vapor specific heat (const. vol.)',
cvliq_stg=4186.,   remark='Water liquid specific heat (const. vol.)',
rvap_stg=416.,    remark='Gas constant for water vapor',
rgas_stg=289.,    remark='Gas constant for air',
vaprlx_stg=0.8,   remark='Relaxation factor for phase change iterations',
rhlim_stg='y',    remark='Limit rel humidity to 100 percent',
$end

```

End of listing of file prepin.p-r_1-D_FULL_40_diff_Twall

RH > 100% Allowed

The FLOW-3D predictions are compared to the theoretical predictions for this case in Figure 3/14/06-1. The FLOW-3D predictions for both the temperature profile and the water concentration profile are close to those from the theoretical analysis. The deviation in temperature values is defined as the ratio of the local variance and the overall temperature difference between the surfaces. Similarly, the vapor concentration variance is defined as the ratio of the local difference in the predictions and the difference in concentration values at the two surfaces. Thus

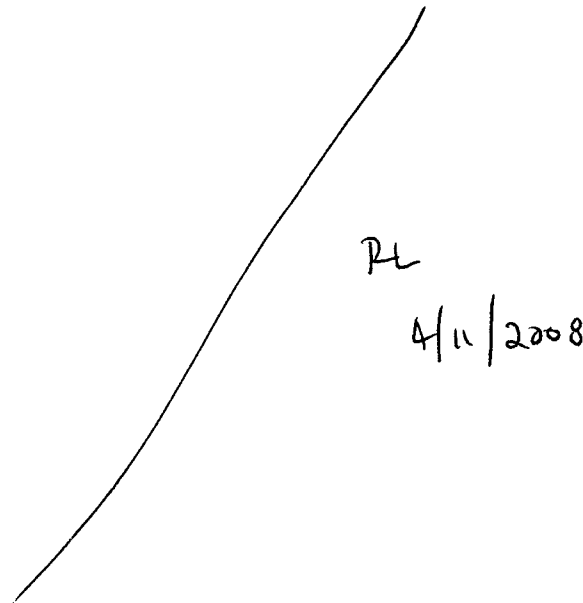
$$Deviation = \begin{cases} \frac{T_{FLOW3D}(z) - T_{theory}(z)}{T_{hot} - T_{cold}} & \text{temperature deviation} \\ \frac{y_{FLOW3D}(z) - y_{theory}(z)}{y_{hot} - y_{cold}} & \text{concentration deviation} \end{cases}$$

Note that the concentrations values are the water vapor mass fraction with respect to the total fluid mass. The Mathcad file listed above shows that the water vapor concentration is greater than 100% across the entire gap in this case.

The FLOW-3D predictions and the theoretical results agree to within 1.5%.

RH > 100% Allowed

The FLOW-3D predictions are compared to the theoretical predictions for this case in Figure 3/14/06-2. The FLOW-3D predictions for both the temperature profile and the water concentration profile are in close agreement with those of the theoretical analysis. Recall, however, that the theoretical results in this case were obtained by a numerical approximation of the governing differential equations. The FLOW-3D predictions and the theoretical results agree to within 5% over the entire domain.



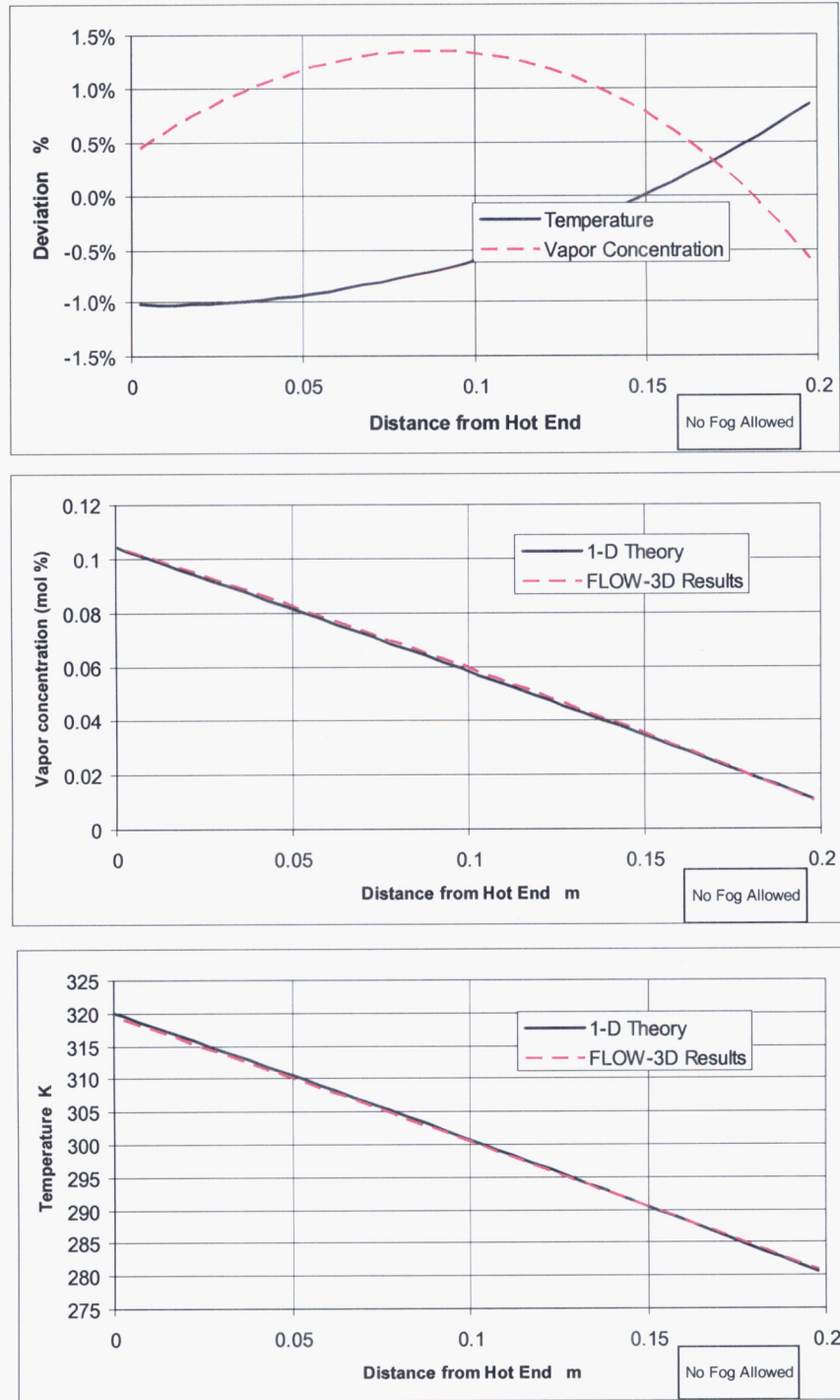


Figure 3/14/06-1. 1-D Conduction and Water Diffusion Results. RH >100% Allowed

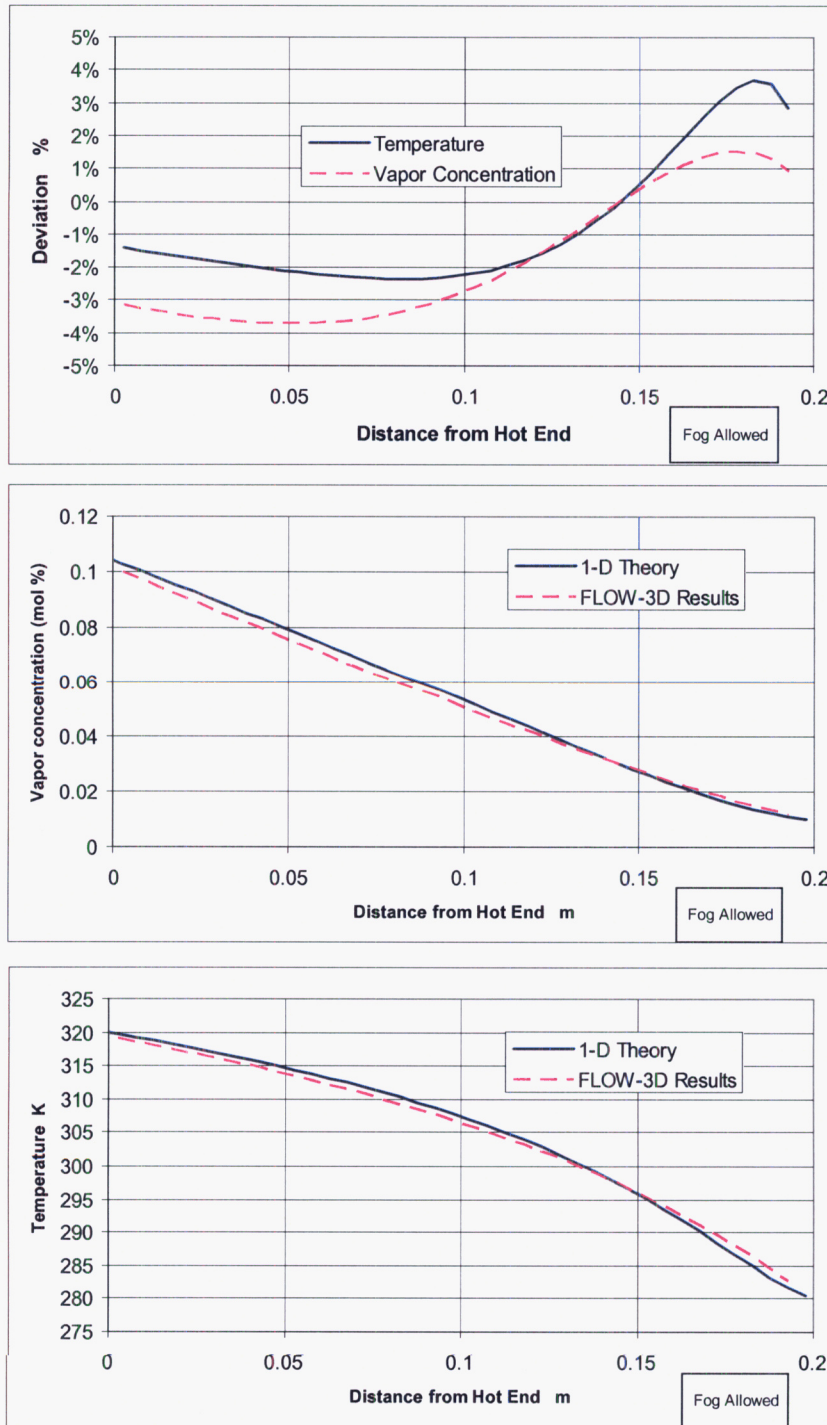


Figure 3/14/06-2. 1-D Conduction and Water Diffusion Results. RH =100% Maximum

END OF ENTRY FOR 3/14/06 *STG*

3/16/06

STG

This entry documents the work performed to partially validate the radiation module written for simulating the radiation heat transfer processes pertinent to waste repository drifts. This module is described in the entry for 3/3/06 in this notebook.

Problem Statement:

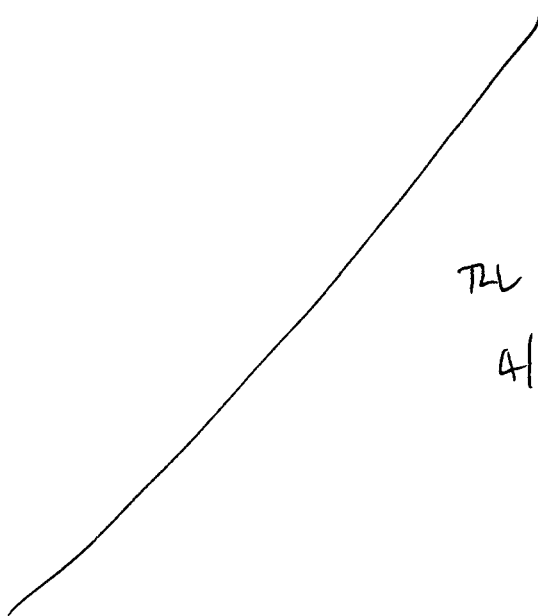
Two large flat plates are situated parallel to each other such that there is a horizontal gap of 0.1 m between them. Each plate is 0.02 m thick and the facing surfaces are gray and diffuse with respect to thermal radiation. The emissivity of each surface is 0.9. The top surface has an internal heat source such that the heat flux from the surface facing the gap is 255 W/m^2 . The outside face of the lower plate is maintained at 300 K. The gap is filled with dry air and the thermal conductivity of each plate is $1.0 \text{ W/(m}\cdot\text{K)}$.

We are to find the temperature distribution in the plates and the air gap under these conditions.

A Mathcad worksheet was developed to define and solve the governing equations for this scenario. This file is

Gap-radiation.mcd

and is presented in the following 5 pages.



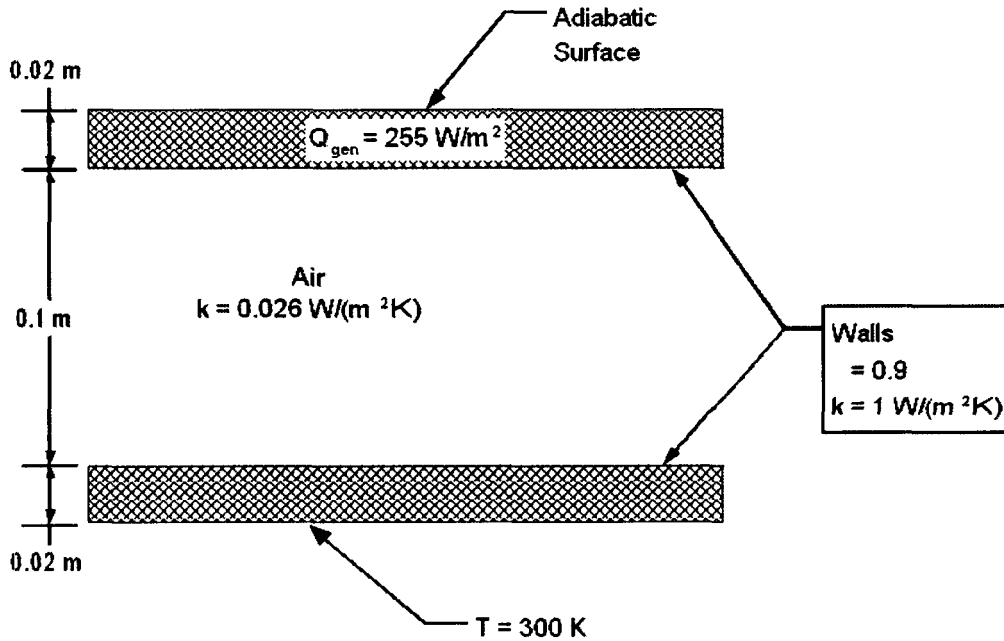
TL

4/11/2008

Radiation and Conduction Heat Transfer Across a Gap

Introduction

This document describes an analytical solution to the combined conduction heat transfer and radiation heat transfer between two large solid plates separated by an air gap, Figure 1. One plate generates heat internally such that the heat flux into the gap region uniform; the outer surface of the other plate is held at a fixed temperature. 'Outer' refers to the surface or a plate that faces away from the other.



This analysis serves as a validation test case for the software written to incorporate a thermal radiation heat transfer model into the computer program FLOW-3D.

References

The equations describing the radiation heat transfer process are given in

Siegel, R., Howell, W., *Radiation Heat Transfer*, Hemisphere Publishing, Washington, D. C., 1992

Assumptions

1. Heat conduction across the air gap and across the plate thickness is one-dimensional.
2. The plate surfaces are gray and diffuse with respect to thermal radiation.
3. Material properties are uniform and not temperature dependent.
4. The air in the gap does not affect the thermal radiation.

Conduction Heat Transfer

The conduction heat transfer across the air gap is given by

$$q_{\text{air.cond}} = k_{\text{air}} \frac{T_1 - T_2}{t_{\text{gap}}} \quad \text{Eq. 1}$$

where

$q_{\text{air.cond}}$	Conduction heat flux across air gap
T_1	Inner surface temperature of upper plate
T_2	Inner surface temperature of lower plate.
$t_{\text{gap}} := 0.1 \cdot \text{m}$	Air gap thickness
$k_{\text{air}} := 0.026 \cdot \frac{\text{W}}{\text{m}\cdot\text{K}}$	Air thermal conductivity

Similarly, the heat flux across the lower plate is

$$q_{\text{plate.cond}} = k_{\text{plate}} \frac{T_2 - T_C}{t_{\text{plate}}} \quad \text{Eq. 2}$$

where

$q_{\text{plate.cond}}$	Conduction heat flux across lower plate
T_2	Inner surface temperature of lower plate
$T_C := 300 \cdot \text{K}$	Outer surface temperature of lower plate.
$t_{\text{plate}} := 0.02 \cdot \text{m}$	Plate thickness
$k_{\text{plate}} := 1 \cdot \frac{\text{W}}{\text{m}\cdot\text{K}}$	Plate thermal conductivity

The thermal radiation between two opposing infinite flat gray, diffuse surfaces is given by

$$q_{\text{plate.rad}} = \frac{\sigma \cdot (T_1^4 - T_2^4)}{\frac{1}{\varepsilon_1} + \frac{1}{\varepsilon_2} - 1} \quad \text{Eq. 3}$$

where

$q_{\text{plate.rad}}$	Radiation heat flux between the two plates
T_1	Inner surface temperature of upper plate
T_2	Inner surface temperature of lower plate.
$\varepsilon_1 := 0.9$	Emissivity of upper plate surface
$\varepsilon_2 := 0.9$	Emissivity of lower plate surface
$\sigma := 5.67 \cdot 10^{-8} \cdot \frac{\text{W}}{\text{m}^2 \cdot \text{K}^4}$	Stefan-Boltzmann constant

The overall energy balance for the system is

$$q_{\text{upper}} = q_{\text{plate.rad}} + q_{\text{air.cond}} = q_{\text{plate.cond}}$$

Eq. 4

where $q_{\text{upper}} := 255 \frac{\text{W}}{\text{m}^2}$

Equations 1-4 can be formed into a system of two equations with two unknowns, T_1 and T_2 . These will be solved with the aid of a Mathcad Given/Find block. First provide a guess for the two unknowns as seed values for the solution algorithm

$$T_1 := 350 \cdot \text{K} \quad T_2 := T_C$$

The two solution equations are

Given

$$q_{\text{upper}} = \frac{\sigma \cdot (T_1^4 - T_2^4)}{\frac{1}{\epsilon_1} + \frac{1}{\epsilon_2} - 1} + k_{\text{air}} \frac{T_1 - T_2}{t_{\text{gap}}}$$

$$q_{\text{upper}} = k_{\text{plate}} \frac{T_2 - T_C}{t_{\text{plate}}}$$

$$\begin{pmatrix} T_1 \\ T_2 \end{pmatrix} := \text{Find}(T_1, T_2)$$

Solution for the two inner surface temperatures

$$T_1 = 343.645 \text{ K}$$

$$T_2 = 305.1 \text{ K}$$

$$q_{\text{air.cond}} := k_{\text{air}} \frac{T_1 - T_2}{t_{\text{gap}}}$$

$$q_{\text{air.cond}} = 10.022 \frac{\text{W}}{\text{m}^2}$$

$$q_{\text{plate.rad}} := \frac{\sigma \cdot (T_1^4 - T_2^4)}{\frac{1}{\epsilon_1} + \frac{1}{\epsilon_2} - 1}$$

$$q_{\text{plate.rad}} = 244.978 \frac{\text{W}}{\text{m}^2}$$

$$q_{\text{plate.cond}} := k_{\text{plate}} \frac{T_2 - T_C}{t_{\text{plate}}}$$

$$q_{\text{plate.cond}} = 255 \frac{\text{W}}{\text{m}^2}$$

Define some terms consistent with the FLOW-3D problem in prep.in.p-r_1D_norad_Q-T-wall:

$$q_{\text{slab}} := \frac{q_{\text{upper}}}{t_{\text{plate}}} \quad q_{\text{slab}} = 1.275 \times 10^4 \frac{\text{W}}{\text{m}^3} \quad \text{Volumetric heat generation in the upper plate}$$

$$Z_H := 0.07\text{-m} \quad \text{Location of adiabatic (outer) face of the upper plate}$$

$$Z_1 := 0.05\text{-m} \quad \text{Location of the inner face of upper plate in contact with the air gap.}$$

The FLOW-3D simulation simulates the upper plate described above as an infinite slab with internal heat generation, the wall at $z=Z_H$ is an adiabatic wall (or plane of symmetry), and the wall at $z=Z_1$ is in contact with the air. The temperature distribution in this material is given by

$$T_{\text{upper.platd}}(z, T_1) := T_1 + \frac{q_{\text{slab}}(Z_1 - Z_H)^2}{2 \cdot k_{\text{plate}}} \left[1 - \left(\frac{z - Z_H}{Z_1 - Z_H} \right)^2 \right] \quad \text{Eq. 5}$$

The air gap extends from $z=Z_1$ to $z=Z_2$. The temperature distribution is linear through the air,

$$Z_2 := -0.05\text{-m}$$

$$T_{\text{air}}(z, T_1, T_2) := T_1 - (T_1 - T_2) \cdot \frac{z - Z_1}{Z_2 - Z_1}$$

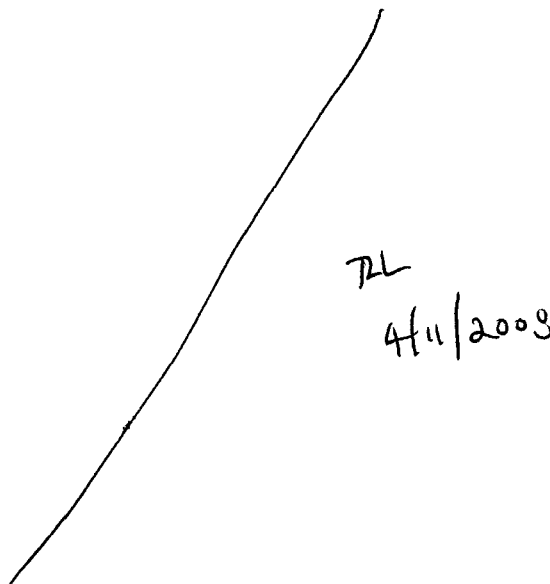
The lower plate is from $z=Z_2$ to $z=Z_C$. The temperature distribution also is linear through the lower plate,

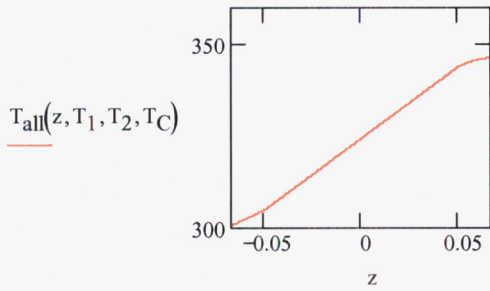
$$Z_C := -0.07\text{-m}$$

$$T_{\text{lower.platd}}(z, T_2, T_C) := T_2 - (T_2 - T_C) \cdot \frac{z - Z_2}{Z_C - Z_2}$$

We can combine the equations for all three regions into one function

$$T_{\text{all}}(z, T_1, T_2, T_C) := \text{if}(z > Z_H, 9999, \text{if}(z > Z_1, T_{\text{upper.platd}}(z, T_1), \text{if}(z > Z_2, T_{\text{air}}(z, T_1, T_2), \text{if}(z > Z_C, T_{\text{lower.platd}}(z, T_2, T_C), 9999))))))$$





i := 0..55

$$z_i := \left(-0.07 + \frac{0.0025}{2} + i \cdot 0.0025\right) \cdot m \quad a_{i,0} := \frac{z_i}{m} \quad a_{i,1} := \frac{T_{all}(z_i, T_1, T_2, T_C)}{K}$$

	0
0	-0.0688
1	-0.0663
2	-0.0638
3	-0.0613
4	-0.0588
5	-0.0563
6	-0.0538
z = 7	-0.0513
8	-0.0488
9	-0.0463
10	-0.0438
11	-0.0413
12	-0.0388
13	-0.0363
14	-0.0338
15	-0.0313

	0	1
0	$-6.875 \cdot 10^{-2}$	$3.0032 \cdot 10^2$
1	$-6.625 \cdot 10^{-2}$	$3.0096 \cdot 10^2$
2	$-6.375 \cdot 10^{-2}$	$3.0159 \cdot 10^2$
3	$-6.125 \cdot 10^{-2}$	$3.0223 \cdot 10^2$
4	$-5.875 \cdot 10^{-2}$	$3.0287 \cdot 10^2$
5	$-5.625 \cdot 10^{-2}$	$3.0351 \cdot 10^2$
6	$-5.375 \cdot 10^{-2}$	$3.0414 \cdot 10^2$
a = 7	$-5.125 \cdot 10^{-2}$	$3.0478 \cdot 10^2$
8	$-4.875 \cdot 10^{-2}$	$3.0558 \cdot 10^2$
9	$-4.625 \cdot 10^{-2}$	$3.0655 \cdot 10^2$
10	$-4.375 \cdot 10^{-2}$	$3.0751 \cdot 10^2$
11	$-4.125 \cdot 10^{-2}$	$3.0847 \cdot 10^2$
12	$-3.875 \cdot 10^{-2}$	$3.0944 \cdot 10^2$
13	$-3.625 \cdot 10^{-2}$	$3.104 \cdot 10^2$
14	$-3.375 \cdot 10^{-2}$	$3.1136 \cdot 10^2$
15	$-3.125 \cdot 10^{-2}$	$3.1233 \cdot 10^2$

These values can be copied and pasted into Excel

PL 4/11/2008

A FLOW-3D input file was prepared for a simulation of this heat transfer process. The file was prepared in accordance with the FLOW manual and the instructions for using the custom radiation heat transfer module. The FLOW-3D input file for this case is

prepin.p-r_1D_rad_Q-T-wall

This file is listed in its entirety as follows. This file was created to define 20 cells across the air gap and four cells in each of the plates. Another file was created to specify twice these numbers of cells in the respective regions to investigate the sensitivity of the results to grid resolution. This file is not listed here since it can be re-created directly from the listing below.

Listing of file prepin.p-r_1D_rad_Q-T-wall

```
Heat transfer across air gap, 1-D grid, 1-D H.T. No flow, No Radiation,
0.1 m wide, 300 K at bottom, 255 W in top obstacle all goes into the gap, no buoyancy

$хput
  remark='units are SI',
  itb=0,      remark='No sharp interface',
  ifvis=0,    remark='Laminar flow ',
  ifenrg=2,   remark='Solve energy equation, 1st order',
  ifrho=0,    remark='Constant density, bbut will be overridden by vapor model',
  ihtc=2,     remark='Evaluate heat transfer and solve solid conduction equation',
  rmrhoe=1.,  remark='Density diffusion term coefficient',
  rmrho=1.,   remark='Energy diffusion term coefficient',
              remark=' RMRHO, RMRHOE REQUIRED FOR VAPOR TRANSPORT MODEL ',
  iusrd=1,    remark='Flag for reading the USRDAT section',
  delt=1.e-4, remark='Initial time step',
  twfin=500., remark='Simulation end time',
$send

$limits
$send

$props
  units='si',
  rhof=1.095, remark='Bulk nominal density for energy equation',
  mul=2.e-05, remark='Bulk nominal dynamic viscosity density for momentum equation',
  cvl=717.,   remark='Const. Vol. Specific heat of dry air',
  thc1=0.026, remark='Bulk nominal thermal conductivity for energy equation',
$send

$scalar
  remark=' MOISTURE IS NOT USED HERE, BUT USE THE MODEL FOR CONSISTENCY WITH CODE VERSION',
  remark='Scalar 1 is the diffusing/advecting water',
  remark='Scalar 2 (non-diffusing) is for storing the surface phase change flux values',
  remark='Scalar 3 (non-diffusing) is for storing the relative humidity values',
  remark='Scalar 4 (non-diffusing) is for storing the net surface liquid accumulation',
  remark='Scalar 5 (non-diffusing) is for storing the vapor water concentration',
  remark='Scalar 6 (non-diffusing) is for storing the liquid water (mist) concentration',
  remark='Scalar 7 (non-diffusing) is for storing the calculation iterations for the',
  remark=' wall mass flux',
```

```

remark='Scalar 8 (non-diffusing) is for storing the calculation iterations for the',
remark='      mist pahse change in the fluid interior',
remark='Scalars 7 and 8 are helpful in tuning the value of vaprlx if necessary',

```

```

nsc=8,
isclr(1)=3, cmisc(1)=0.26e-04, scltit(1)='Tot.Water', rmisc=0.,
isclr(2)=0, cmisc(2)=0., scltit(2)='Liq.Flux',
isclr(3)=0, cmisc(3)=0., scltit(3)='Rel.Hum',
isclr(4)=0, cmisc(4)=0., scltit(4)='Net.Liq',
isclr(5)=0, cmisc(5)=0., scltit(5)='Vap.Wat',
isclr(6)=0, cmisc(6)=0., scltit(6)='Liq.Wat',
isclr(7)=0, cmisc(7)=0., scltit(7)='Itr.Wall',
isclr(8)=0, cmisc(8)=0., scltit(8)='Itr.Mesh',
$end

```

```
$bcdata
```

```

wl=1, wr=1,          remark=' Symmetry at left, right, front, back for 1-D in Z',
wf=1, wbk=1,
wb=2, tbc(5)=300., remark=' Constant temp wall at bottom',
wt=1,              remark=' Use symmetry at top for adiabatic',
$end

```

```
$mesh
```

```

px(1) = 0.0,      py(1) =0.0,      pz(1) = -0.07,
px(2) = 1.0,      py(2) =1.,      pz(2) = -0.05,
                                   pz(3) =  0.05,
                                   pz(4) =  0.07,
                                   nzcell(1)=4,
                                   nzcell(2)=20,
                                   nzcell(3)=4,
nxcelt= 1,      nycelt=1,      nzcelt=28,
$end

```

```
$obs
```

```

avrck=-3.1,
nobs = 2,
tobs(1)=0., tobs(2)=1000.,
remark='Obstacle 1. Conductive wall at bottom',
zl(1)=-0.07, zh(1)=-0.05,
kobs(1)= 1., rcobs(1)=10000.,
twobs(1,1)=300.,
remark='Obstacle 2. Constant heat output from top surface',
zl(2)= 0.05, zh(2)=0.07,
kobs(2)= 1., rcobs(2)=10000.,
pobs(1,2)=255., pobs(2,2)=255., twobs(1,2)=350.,
$end

```

```
$fl
```

```

nfls=1,
remark=' No vapor in fluid',
sclri(1)=0.0,
sclri(5)=0.0,
sclri(6)=0.0,
presi=101300.,
$end

```

```
$bf
```

```
$end
```

```
$temp
```

```

ntmp=1,
tempi=300.,
$end

```

```
$motn
```

```
$end
```



```

$grafic
$end

$parts
$end

$susrdat

remark=' Define the locations of the radiation surfaces',

nrsrf_stg=2,

remark=' Rad. surface 1 is obstacle 1',
eps_stg(1) = .9,
rad_stg(1,1) = 1.,
rad_stg(2,1) = -1.,    rad_stg(3,1) = 2.,
rad_stg(4,1) = -1.,    rad_stg(5,1) = 2.,
rad_stg(6,1) = -0.055, rad_stg(7,1) = -0.045,

remark=' Rad. surface 2 is obstacle 2',
eps_stg(2) = .9,
rad_stg(1,2) = 2.,
rad_stg(2,2) = -1.,    rad_stg(3,2) = 2.,
rad_stg(4,2) = -1.,    rad_stg(5,2) = 2.,
rad_stg(6,2) = 0.045,  rad_stg(7,2) = 0.055,

remark=' User-defined configuration factors',
iusrcf_stg=1,
cf_stg(1,1)=0., cf_stg(1,2) = 1.,
cf_stg(2,1)=1., cf_stg(2,2) = 0.,

remark=' Specify the vapor model for defining the thermal properties only',
istwtf_stg='tw',    remark='Energy for phase change comes from wall',
isvap_stg = 1,      remark='Define scalar index for water concentration DO NOT CHANGE',
isliq_stg = 2,      remark='Define scalar index for liquid flux DO NOT CHANGE',
isrh_stg = 3,       remark='Define scalar index for rel. hum. DO NOT CHANGE',
istlq_stg = 4,      remark='Define scalar index for tot.liq. accum. DO NOT CHANGE',
isywv_stg = 5,      remark='Define scalar index for vapor water DO NOT CHANGE',
isywl_stg = 6,      remark='Define scalar index for liquid water (mist) DO NOT CHANGE',
hvvap_stg=2304900., remark='Heat of vaporization',
cvvap_stg=1370.,    remark='Water vapor specific heat (const. vol.)',
cvliq_stg=4186.,    remark='Water liquid specific heat (const. vol.)',
rvap_stg=416.,      remark='Gas constant for water vapor',
rgas_stg=289.,      remark='Gas constant for air',
vaprlx_stg=0.8,     remark='Relaxation factor for phase change iterations',
rhlim_stg='y',      remark='Limit rel humidity to 100 percent',
$end

```

End of listing of file prepin.p-r_1D_rad_Q-T-wall

The analytical solution and the FLOW-3D simulation results were copied into an Excel spreadsheet for comparison:

1-D_rad_Analysis-FLOW-3D.xls

The results are compared in the graphs shown in Figure 3/16/06-1. These graphs show the expected trends of linear temperature profiles in the lower plate and in the air gap. The upper plate shows a quadratic shape because of the presence of the heat source within the plate.

Correction made by STG 3/14/2008 - see entry for 3/14/08. The filename **1-D_rad_Analysis-FLOW-3D.xls** is incorrect. The correct file name is **Compare Analysis-FLOW-3D.xls**

There is good agreement between the analytical results and the FLOW-3D results. The deviation between the two sets of results is less than 5% (The basis of this percentage is the overall temperature difference in the system, approximately 45 K). The finer grid resolution does not substantially improve on the results obtained with the coarser grid.

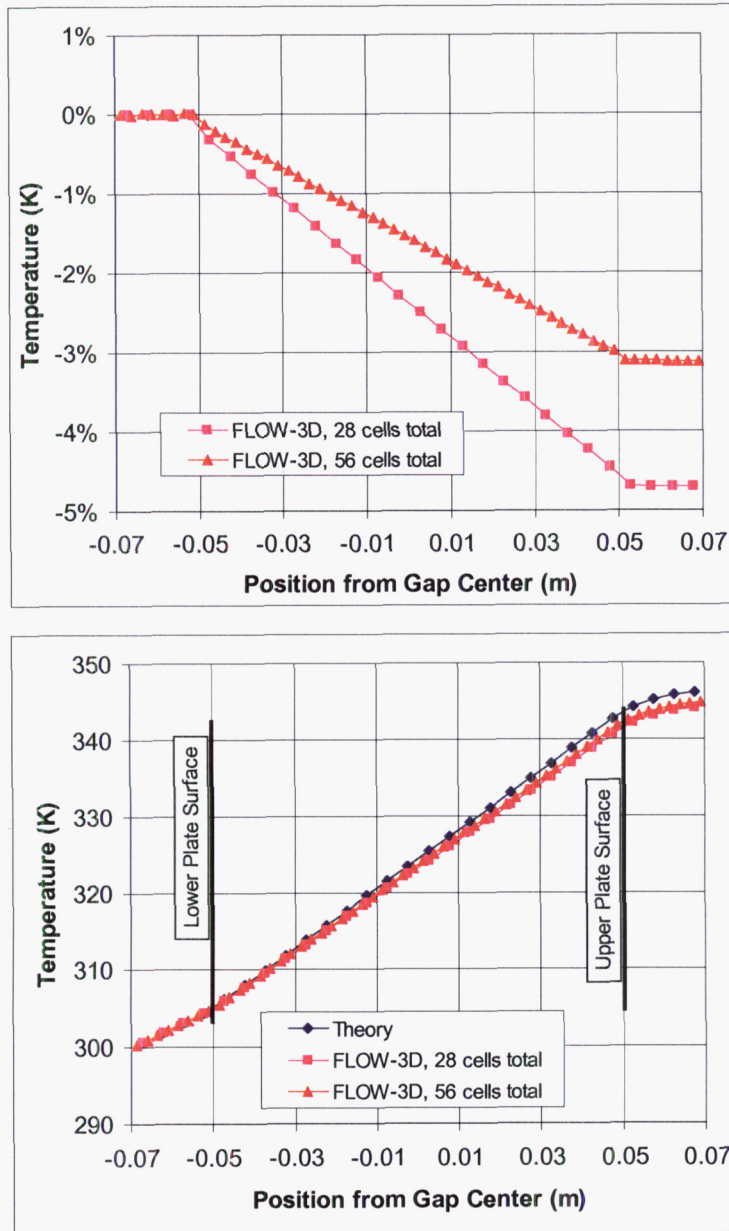


Figure 3/16/06-1. Conduction & Radiation Across Air Gap

END OF ENTRY FOR 3/16/06 *STG*

3/20/06 *STG*

This entry records the work performed to validate the configuration factor calculations in the custom radiation module developed for use in FLOW-3D. The specification of radiation configuration factors is an important part the overall radiation analysis. The analyst must develop an expression of the configuration factor or rely on software tools such as CAD packages to find configuration factors by approximate analysis.

The radiation module uses the FLOW-3D data arrays that described the surfaces of solid obstacles to compute these radiation factors. The mathematical basis of the calculations is found in the entry for 3/3/06 in this notebook. The ability of the code to compute these factors relieves the user of this burden.

Two cases are considered here. First is the radiation in the two-dimensional space between concentric infinite cylinders as shown in Figure 3/20/06-1. The outer cylinder has a diameter of 1 m and its inner surface is subdivided in to four equal segments. The inner cylinder has a diameter of 0.3 m and is likewise divided into four equal segments. There are a total of 64 configuration factors for this system of surfaces; although, many of them are null. The subdivision of these obstacles is entirely arbitrary for the purposes of this validation exercise. The number of specified radiation surfaces for ant particular problem is dictated by the required precision to which the spatial variations of radiation heat fluxes are to be computed.

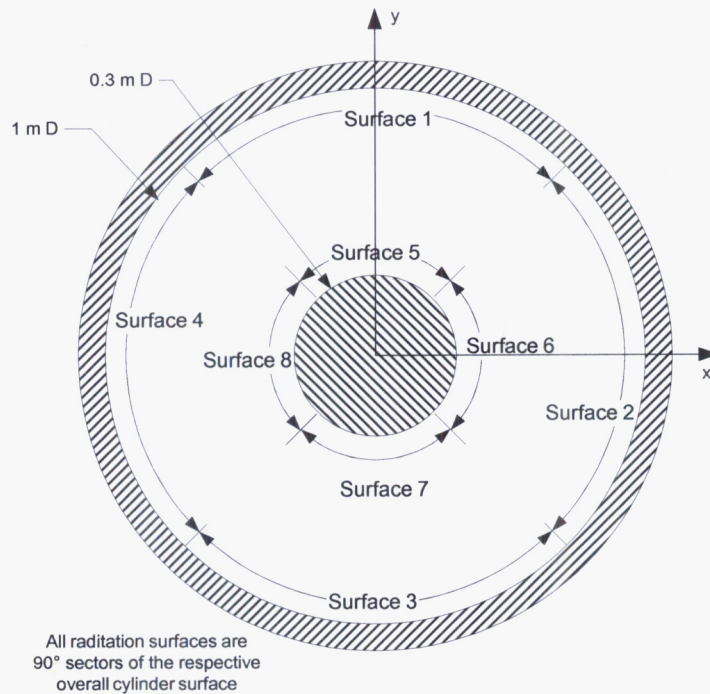
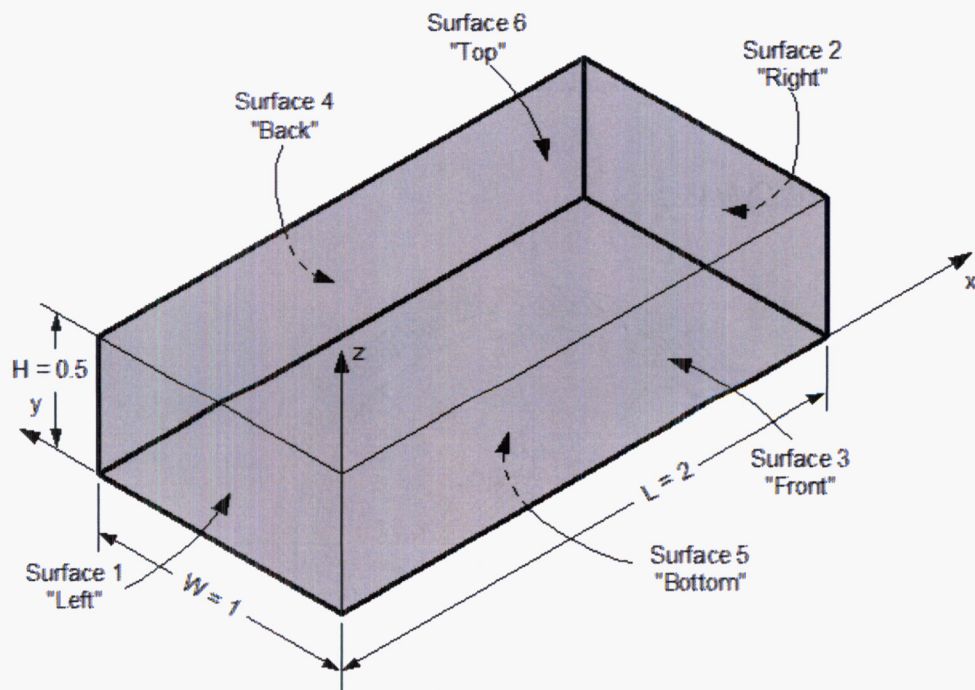


Figure 3/20/06-1. Radiation Between Concentric Cylinders.

The second case is the radiation between surfaces in a three-dimensional rectangular enclosure with dimensions of $2 \times 1 \times 0.5$. This enclosure is depicted in Figure 3/20/06-2. The units of these dimensions do not affect the computation of configuration factors, since the configuration factors are dependent on the relative shapes and positions of the surfaces.



The choice of the nomenclature 'top', 'left', etc. is consistent with the FLOW-3D designations of surfaces.

The analysis and computations for both of these cases are considered in the Mathcad file

[CFs_ver-test.mcd](#)

The analysis is completely described and documented in this Mathcad file. A listing of this file is included here in the following 10 pages.

Radiation Heat Transfer Configuration Factors

Introduction

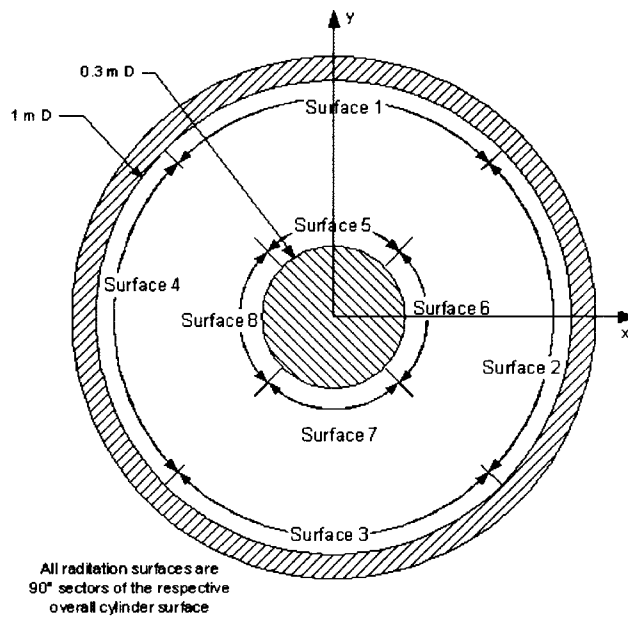
This document contains the theoretical analysis of thermal radiation configuration factors for two cases that serve to validate the configuration factor calculation algorithm in the SwRI-developed radiation heat transfer module written for FLOW-3D. The first case is that of infinite concentric cylinders with a diameter ratio of 0.3. The second case is that of the walls in a rectangular parallelepiped with dimensions of 2 x 1 x 0.5.

References

Siegel, R., Howell, J. R., Thermal Radiation Heat Transfer, Third Edition, Hemisphere Publishing, Washington, D. C., 1992.

Howell, J. R., A Catalog of Radiation Configuration Factors, McGraw-Hill Book Co., 1982 (With errata dated November 10, 1993)

Segmented Concentric Cylinders



$R_0 := 0.5\text{-m}$	$\theta_a := 135\text{-deg}$	$\theta_e := 135\text{-deg}$	$\theta_c := -45\text{-deg}$	$\theta_g := -45\text{-deg}$
$R_1 := 0.15\text{-m}$	$\theta_b := 45\text{-deg}$	$\theta_f := 45\text{-deg}$	$\theta_d := -135\text{-deg}$	$\theta_h := -135\text{-deg}$

All angles are relative to the +x-axis

The Hottel crossed string method as described by Siegel and Howell will be used for these 2-D configuration factors.

Hottel Crossed String Method

The geometry for this problem is two-dimensional. In these cases, the 'Hottel Crossed String Method' for computing radiation configuration factors works very well. This method, described by Siegel and Howell, uses the relation,

$$F_{12} = \frac{\left[\sum_i (\text{Crossed_String_Length}) - \sum_i (\text{Uncrossed_String_Length}) \right]}{2 \cdot L_1}$$

where L_1 Length of surface 1

The meaning of the terms 'crossed strings' and 'uncrossed strings' will be evident in the following example calculations.

Configuration Factor F_{15} and F_{51}

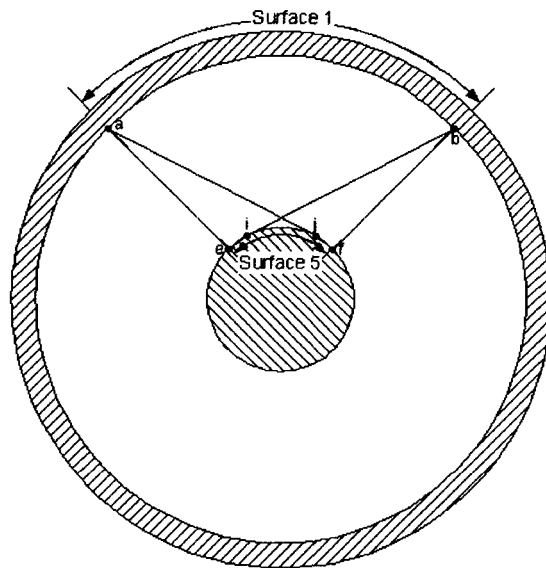
One of the crossed strings in this configuration is made up of the straight segment, L_{aj} and the arc L_{jf} .

$$L_{aj} := (R_o^2 - R_i^2)^{0.5} \quad L_{aj} = 0.477 \text{ m}$$

$$\theta_j := \theta_a - \left(90 \cdot \text{deg} - \arccos \left(\frac{L_{aj}}{R_o} \right) \right) \quad \theta_j = 62.458 \text{ deg}$$

$$L_{jf} := R_i \cdot (\theta_j - \theta_f) \quad L_{jf} = 0.046 \text{ m}$$

$$L_{ajf} := L_{aj} + L_{jf} \quad L_{ajf} = 0.523 \text{ m}$$



The two crossed strings in this configuration are equal.

One of the uncrossed strings for F_{15} is the straight segment L_{ae} . The other uncrossed string is $L_{bf} = L_{ae}$

$$L_{ae} := R_o - R_i \quad L_{ae} = 0.35 \text{ m}$$

The length of Surface 1 is the arc L_{ab}

$$L_1 := R_o \cdot (\theta_a - \theta_b) \quad L_1 = 0.785 \text{ m}$$

The length of surface 5 is the arc L_{eff}

$$L_5 := R_i \cdot (\theta_e - \theta_f) \quad L_5 = 0.236 \text{ m}$$

The configuration factor F_{15} is

$$F_{15} := \frac{2 \cdot L_{ajf} - 2 \cdot L_{ae}}{2 \cdot L_1} \quad F_{15} = 0.22$$

Using the reciprocity relation, the configuration factor F_{51} is

$$F_{51} := F_{15} \cdot \frac{L_1}{L_5} \quad F_{51} = 0.733$$

Configuration Factor F_{16} and F_{61}

One of the crossed strings in this configuration is made up of the straight segment, L_{aj} , and the arc L_{jg} .

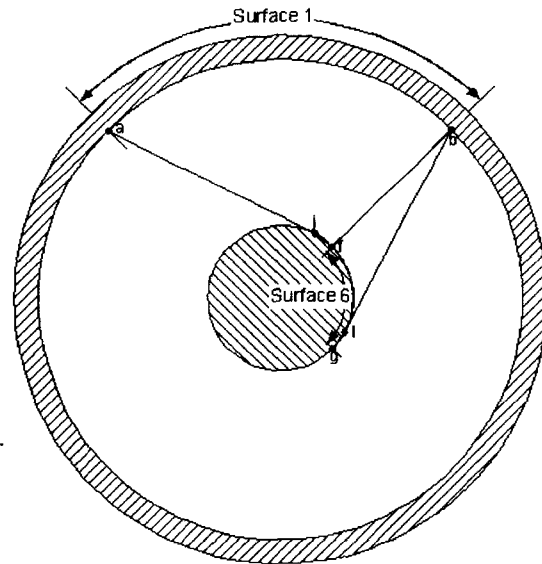
$$L_{jg} := R_i \cdot (\theta_j - \theta_g) \quad L_{jg} = 0.281 \text{ m}$$

$$L_{ajg} := L_{aj} + L_{jg} \quad L_{ajg} = 0.758 \text{ m}$$

The other crossed string is the straight segment L_{bf} .

$$L_{bf} := L_{ac} \quad L_{bf} = 0.35 \text{ m}$$

One of the uncrossed strings for F_{16} is the segment L_{ajf} defined above. The other uncrossed string in this configuration is made up of the straight segment, L_{bk} , and the arc L_{kg} . This has the same length as L_{ajf} .



The length of surface 6 is the same as surface 5

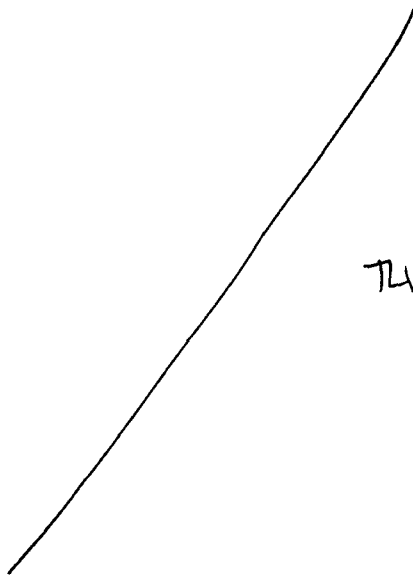
$$L_6 := L_5 \quad L_5 = 0.236 \text{ m}$$

The configuration factor F_{16} is

$$F_{16} := \frac{L_{ajg} + L_{bf} - 2 \cdot L_{ajf}}{2 \cdot L_1} \quad F_{16} = 0.04$$

Using the reciprocity relation, the configuration factor F_{51} is

$$F_{61} := F_{16} \cdot \frac{L_1}{L_6} \quad F_{61} = 0.134$$



TL

4/11/2008

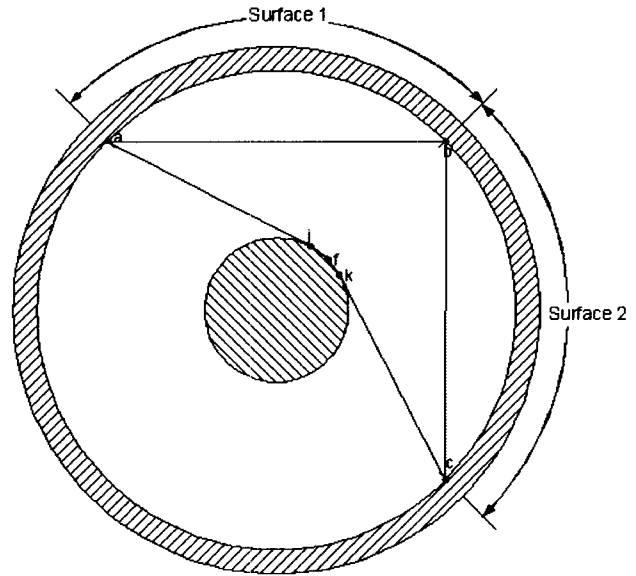
Configuration Factor F_{12} and F_{21}

The view of Surface 2 by Surface 1 is partially blocked around the right side of the inner cylinder. The view is completely blocked around the left side of the inner cylinder.

One of the crossed strings in this configuration is made up of the straight segment, L_{ab} and the other is the straight segment L_{bc} .

$$L_{ab} := \sqrt{2 \cdot R_o^2} \quad L_{ab} = 0.707 \text{ m}$$

$$L_{bc} := L_{ab}$$



One of the uncrossed strings in this configuration is made up of the straight segment, L_{aj} , and the arc L_{jk} , and the straight segment L_{kc} .

$$L_{kc} := L_{aj}$$

$$\theta_k := \theta_c + \left(90 \cdot \text{deg} - \arccos \left(\frac{L_{kc}}{R_o} \right) \right) \quad \theta_k = 27.542 \text{ deg} \quad \text{Angular position of point i.}$$

$$L_{jk} := R_i \cdot (\theta_j - \theta_k) \quad L_{jk} = 0.091 \text{ m} \quad \text{Arc length segment}$$

$$L_{ajkc} := L_{aj} + L_{jk} + L_{kc} \quad L_{ajkc} = 1.045 \text{ m}$$

The other uncrossed string is of zero length since the edges of Surface 1 and Surface 2 coincide at point b

The configuration factor F_{12} is

$$F_{12} := \frac{L_{ab} + L_{bc} - L_{ajkc}}{2 \cdot L_1} \quad F_{12} = 0.235$$

The length L_2 is the same as L_1 . Using the reciprocity relation, the configuration factor F_{21} is

$$L_2 := L_1$$

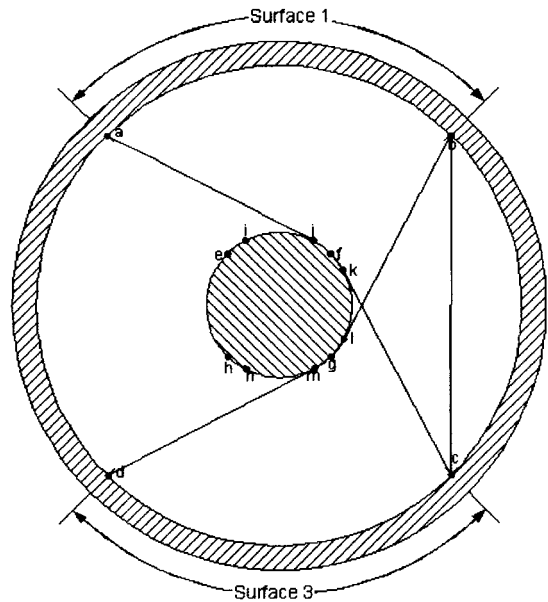
$$F_{21} := F_{12} \cdot \frac{L_1}{L_2} \quad F_{21} = 0.235$$

Configuration Factor F_{13} and F_{31}

The view of Surface 3 by Surface 1 is partially blocked around the right and left sides of the inner cylinder. The overall configuration factor must take this into account by summing the configuration factors for the two viewing paths. In this case the viewing paths are symmetric.

One of the crossed strings in this configuration is made up of the straight segment, L_{aj} , and the arc L_{jk} , and the straight segment L_{kc} . This length has already been computed above. The other crossed string is the straight segment, L_{bl} , and the arc L_{lm} , and the straight segment L_{md} . This is the same length as L_{ajkc} .

One of the uncrossed strings for F_{13} is the segment L_{ajmd} . The other crossed string in this configuration is made up of the straight segment, L_{bc} .



$$L_{md} := L_{aj}$$

$$\theta_m := \theta_d + 90 \cdot \text{deg} - \arccos\left(\frac{L_{md}}{R_o}\right) \quad \theta_m = -62.458 \text{ deg}$$

$$L_{jm} := R_i \cdot (\theta_j - \theta_m) \quad L_{jm} = 0.327 \text{ m}$$

$$L_{ajmd} := L_{aj} + L_{jm} + L_{md} \quad L_{ajmd} = 1.281 \text{ m}$$

The other uncrossed string is the straight segment L_{bc} .

$$L_{bc} := L_{ab} \quad L_{bc} = 0.707 \text{ m}$$

The length of surface 3 is the same as surface 1

$$L_3 := L_1 \quad L_3 = 0.785 \text{ m}$$

The configuration factor F_{13} is

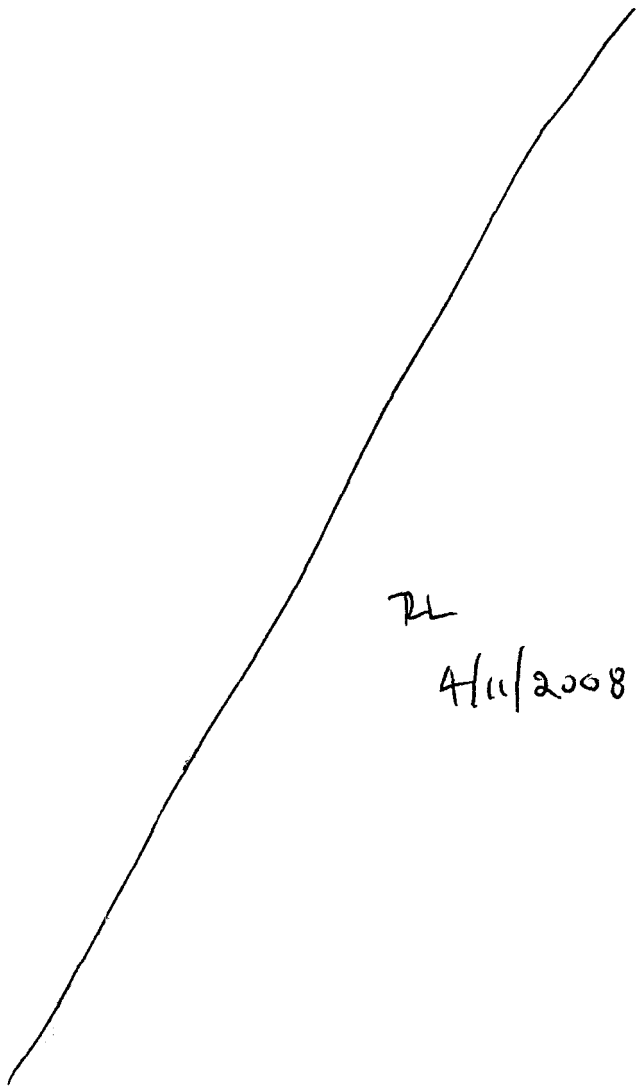
$$F_{13} := 2 \cdot \frac{2 \cdot L_{ajkc} - (L_{bc} + L_{ajmd})}{2 \cdot L_1} \quad F_{13} = 0.131$$

Using the reciprocity relation, the configuration factor F_{31} is

$$F_{31} := F_{13} \cdot \frac{L_1}{L_3} \quad F_{31} = 0.131$$

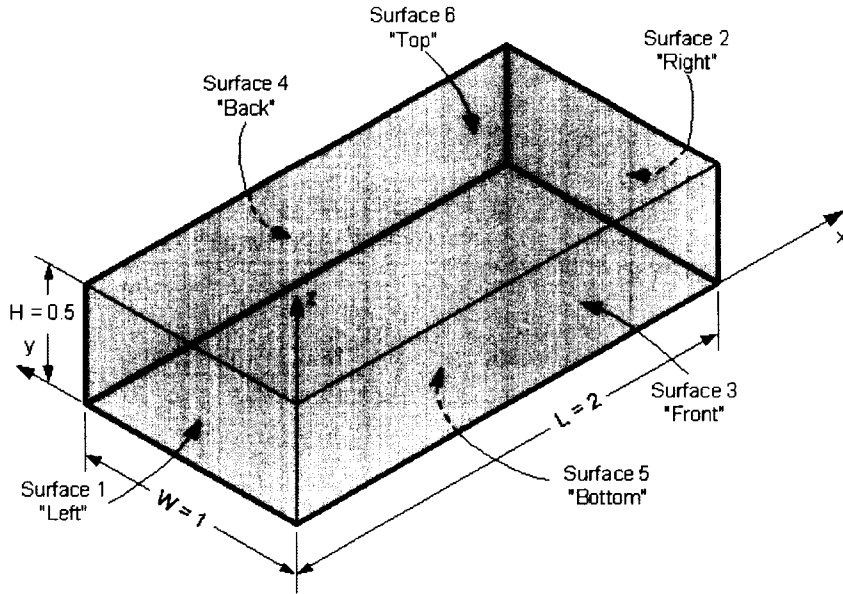
FLOW-3D Results

$F_{12} = 0.23483$	$F_{12.FLOW3D} := 0.22539$	$\frac{F_{12.FLOW3D} - F_{12}}{F_{12}} = -4.019\%$
	$F_{14.FLOW3D} := .23465$	$\frac{F_{14.FLOW3D} - F_{12}}{F_{12}} = -0.076\%$
$F_{13} = 0.13066$	$F_{13.FLOW3D} := 0.13583$	$\frac{F_{13.FLOW3D} - F_{13}}{F_{13}} = 3.956\%$
$F_{15} = 0.21985$	$F_{15.FLOW3D} := 0.21956$	$\frac{F_{15.FLOW3D} - F_{15}}{F_{15}} = -0.134\%$
$F_{16} = 0.04007$	$F_{16.FLOW3D} := 0.040401$	$\frac{F_{16.FLOW3D} - F_{16}}{F_{16}} = 0.819\%$



RL
4/11/2008

3-D Rectangular Enclosure



The configuration factors for finite plane surfaces as described by Howell will be used to compute all the configuration factors for this case.

Define the dimensions for this box

$$L_{\text{box}} := 2 \cdot \text{m}$$

$$W_{\text{box}} := 1 \cdot \text{m}$$

$$H_{\text{box}} := 0.5 \cdot \text{m}$$

Units will have no bearing on the computed results. They are used here as an aid in checking the equations for errors,

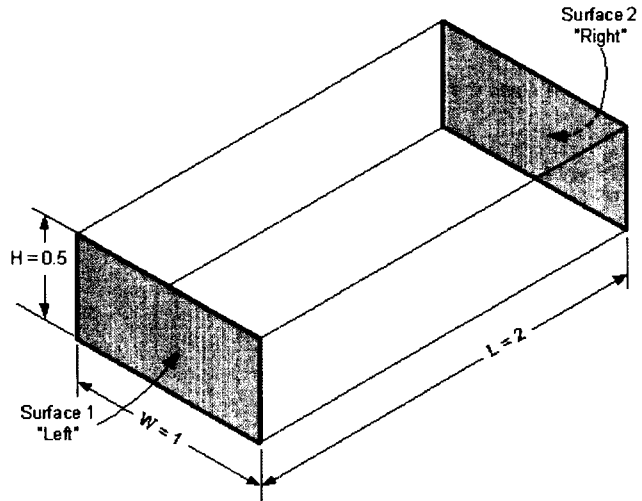
TZL
4/11/2008

Configuration Factor F_{12} and F_{21}

This is Case C-10 in Howell.

Case C-10. Identical, parallel, directly opposed rectangles

$$X = \frac{\text{width}}{\text{gap}} \quad Y = \frac{\text{length}}{\text{gap}}$$



$$F_{1,2,C10}(X, Y) := \frac{2}{\pi \cdot X \cdot Y} \left[\ln \left[\frac{(1 + X^2) \cdot (1 + Y^2)^{0.5}}{1 + X^2 + Y^2} \right] + X \cdot (1 + Y^2)^{0.5} \cdot \text{atan} \left[\frac{X}{(1 + Y^2)^{0.5}} \right] \right. \\ \left. + Y \cdot (1 + X^2)^{0.5} \cdot \text{atan} \left[\frac{Y}{(1 + X^2)^{0.5}} \right] - X \cdot \text{atan}(X) - Y \cdot \text{atan}(Y) \right]$$

Box with dimensions $L \times W \times H = 2 \times 1 \times 0.5$

Faces at ends of length dimension (Left-Right)

$$F_{12,\text{box}} := F_{1,2,C10} \left(\frac{W_{\text{box}}}{L_{\text{box}}}, \frac{H_{\text{box}}}{L_{\text{box}}} \right) \quad F_{12,\text{box}} = 0.036 \quad F_{21,\text{box}} := F_{12,\text{box}}$$

Configuration Factor F_{34} and F_{43}

Faces at ends of depth dimension (Front-Back)

$$F_{34,\text{box}} := F_{1,2,C10} \left(\frac{L_{\text{box}}}{W_{\text{box}}}, \frac{H_{\text{box}}}{W_{\text{box}}} \right) \quad F_{34,\text{box}} = 0.165 \quad F_{43,\text{box}} := F_{34,\text{box}}$$

Configuration Factor F_{56} and F_{65}

Faces at ends of height dimension (Top-Bottom)

$$F_{56,\text{box}} := F_{1,2,C10} \left(\frac{L_{\text{box}}}{H_{\text{box}}}, \frac{W_{\text{box}}}{H_{\text{box}}} \right) \quad F_{56,\text{box}} = 0.509 \quad F_{65,\text{box}} := F_{56,\text{box}}$$

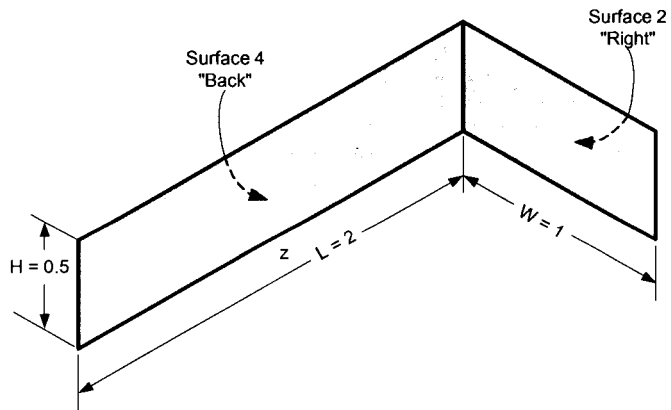
Configuration Factor F_{42} and F_{24}

This is Case C-13 if Howell.

Case C-13. Finite rectangles of same length, having one common edge, 90° to each other

$$H' = \frac{\text{height}}{\text{length}} \quad W' = \frac{\text{width}}{\text{length}}$$

Length is the common edge
 A_1 is the face with the 'width'
 A_2 is the face with the 'height'



$$F_{1.2.C13}(H', W') := \frac{1}{\pi \cdot W'} \left[W' \cdot \text{atan}\left(\frac{1}{W'}\right) + H' \cdot \text{atan}\left(\frac{1}{H'}\right) - (H'^2 + W'^2)^{0.5} \cdot \text{atan}\left[\frac{1}{(H'^2 + W'^2)^{0.5}}\right] \dots \right. \\ \left. + \frac{1}{4} \cdot \ln \left[\frac{(1 + W'^2) \cdot (1 + H'^2)}{1 + W'^2 + H'^2} \cdot \left[\frac{W'^2 \cdot (1 + W'^2 + H'^2)}{(1 + W'^2) \cdot (W'^2 + H'^2)} \right]^{W'^2} \cdot \left[\frac{H'^2 \cdot (1 + H'^2 + W'^2)}{(1 + H'^2) \cdot (H'^2 + W'^2)} \right]^{H'^2} \right] \right]$$

Faces with the box height common (Bottom-Back, Bottom-Front, Top-Back, Top-Front)

$$F_{24.\text{box}} := F_{1.2.C13}\left(\frac{L_{\text{box}}}{H_{\text{box}}}, \frac{W_{\text{box}}}{H_{\text{box}}}\right) \quad F_{24.\text{box}} = 0.167 \quad F_{42.\text{box}} := F_{1.2.C13}\left(\frac{W_{\text{box}}}{H_{\text{box}}}, \frac{L_{\text{box}}}{H_{\text{box}}}\right) \quad F_{42.\text{box}} = 0.084$$

$$F_{14.\text{box}} := F_{24.\text{box}} \quad F_{13.\text{box}} := F_{24.\text{box}} \quad F_{23.\text{box}} := F_{24.\text{box}} \quad F_{41.\text{box}} := F_{42.\text{box}} \quad F_{31.\text{box}} := F_{42.\text{box}} \quad F_{32.\text{box}} := F_{42.\text{box}}$$

Configuration Factor F_{52} and F_{25}

Faces with the box width common (Bottom-Left, Bottom-Right, Top-Left, Top-Right)

$$F_{25.\text{box}} := F_{1.2.C13}\left(\frac{L_{\text{box}}}{W_{\text{box}}}, \frac{H_{\text{box}}}{W_{\text{box}}}\right) \quad F_{25.\text{box}} = 0.315 \quad F_{52.\text{box}} := F_{1.2.C13}\left(\frac{H_{\text{box}}}{W_{\text{box}}}, \frac{L_{\text{box}}}{W_{\text{box}}}\right) \quad F_{52.\text{box}} = 0.079$$

$$F_{26.\text{box}} := F_{25.\text{box}} \quad F_{15.\text{box}} := F_{25.\text{box}} \quad F_{16.\text{box}} := F_{25.\text{box}} \quad F_{62.\text{box}} := F_{52.\text{box}} \quad F_{51.\text{box}} := F_{52.\text{box}} \quad F_{61.\text{box}} := F_{52.\text{box}}$$

Configuration Factor F_{54} and F_{45}

Faces with the box length common (Left-Back, Left-Front, Right-Back, Right-Front)

$$F_{45.\text{box}} := F_{1.2.C13}\left(\frac{W_{\text{box}}}{L_{\text{box}}}, \frac{H_{\text{box}}}{L_{\text{box}}}\right) \quad F_{45.\text{box}} = 0.334 \quad F_{54.\text{box}} := F_{1.2.C13}\left(\frac{H_{\text{box}}}{L_{\text{box}}}, \frac{W_{\text{box}}}{L_{\text{box}}}\right) \quad F_{54.\text{box}} = 0.167$$

$$F_{46.\text{box}} := F_{45.\text{box}} \quad F_{35.\text{box}} := F_{45.\text{box}} \quad F_{36.\text{box}} := F_{45.\text{box}} \quad F_{64.\text{box}} := F_{54.\text{box}} \quad F_{53.\text{box}} := F_{54.\text{box}} \quad F_{63.\text{box}} := F_{54.\text{box}}$$

$F_{12.\text{box}} = 0.036$	$F_{12.\text{box.FLOW3D}} := 3.62 \cdot 10^{-2}$	$\frac{F_{12.\text{box.FLOW3D}} - F_{12.\text{box}}}{F_{12.\text{box}}} = 0.057\%$
$F_{34.\text{box}} = 0.165$	$F_{34.\text{box.FLOW3D}} := 1.65 \cdot 10^{-1}$	$\frac{F_{34.\text{box.FLOW3D}} - F_{34.\text{box}}}{F_{34.\text{box}}} = -0.163\%$
$F_{56.\text{box}} = 0.509$	$F_{56.\text{box.FLOW3D}} := 5.09 \cdot 10^{-1}$	$\frac{F_{56.\text{box.FLOW3D}} - F_{56.\text{box}}}{F_{56.\text{box}}} = 2.226 \times 10^{-3}\%$
$F_{25.\text{box}} = 0.315$	$F_{25.\text{box.FLOW3D}} := 3.21 \cdot 10^{-1}$	$\frac{F_{25.\text{box.FLOW3D}} - F_{25.\text{box}}}{F_{25.\text{box}}} = 2.034\%$
$F_{52.\text{box}} = 0.079$	$F_{52.\text{box.FLOW3D}} := 8.03 \cdot 10^{-2}$	$\frac{F_{52.\text{box.FLOW3D}} - F_{52.\text{box}}}{F_{52.\text{box}}} = 2.098\%$
$F_{54.\text{box}} = 0.167$	$F_{54.\text{box.FLOW3D}} := 1.70 \cdot 10^{-1}$	$\frac{F_{54.\text{box.FLOW3D}} - F_{54.\text{box}}}{F_{54.\text{box}}} = 1.885\%$
$F_{45.\text{box}} = 0.334$	$F_{45.\text{box.FLOW3D}} := 3.40 \cdot 10^{-1}$	$\frac{F_{45.\text{box.FLOW3D}} - F_{45.\text{box}}}{F_{45.\text{box}}} = 1.885\%$
$F_{42.\text{box}} = 0.084$	$F_{42.\text{box.FLOW3D}} := 8.53 \cdot 10^{-2}$	$\frac{F_{42.\text{box.FLOW3D}} - F_{42.\text{box}}}{F_{42.\text{box}}} = 1.967\%$
$F_{24.\text{box}} = 0.167$	$F_{24.\text{box.FLOW3D}} := 1.71 \cdot 10^{-1}$	$\frac{F_{24.\text{box.FLOW3D}} - F_{24.\text{box}}}{F_{24.\text{box}}} = 2.206\%$

TL

4/11/2008

The following FLOW-3D input file was created for the case of the concentric cylinders,

prepin.2-D_conc-cyls_qtr

This file specifies a uniform grid of 74x74 cells over the cylinders. A listing of this file is provided below.

Listing of file prepin.2-D_conc-cyls_qtr

2-D Concentric Cylinders, Inner=0.3 m, Outer=1 m, 1/4 cylinders for Testing CF Calculations

```

$xput
  remark='units are SI',
  remark=' Set up the problem for natural convection, but not meant for complete
execution',
  itb=0,      remark='No sharp interface',
  ifvis=0,    remark='Laminar flow ',
  ifenrg=2,   remark='Solve energy equation, 1st order',
  ifrho=0,    remark='Constant density, bbut will be overridden by vapor model',
  ihtc=2,     remark='Evaluate heat transfer and solve solid conduction equation',
  gz=-9.8,   remark='Gravity in -z direction',
  ipdis=1,    remark='Hydrostatic pressure distribution',
  rmrhoe=1.,  remark='Density diffusion term coefficient',
  rmrho=1.,   remark='Energy diffusion term coefficient',
              remark=' RMRHO, RMRHOE REQUIRED FOR VAPOR TRANSPORT MODEL ',
  iusrd=1,    remark='Flag for reading the USRDAT section',
  delt=1.e-4, remark='Initial time step',
  twfin=1.,   remark='Simulation end time',
  hpltdt=5.,  remark='History plot interval',
$end

$limits
$end

$props
  units='si',
  rhof=1.197, remark='Bulk nominal density for energy equation',
  mul=2.e-05, remark='Bulk nominal dynamic viscosity density for momentum equation',
  cvl=717.,   remark='Const. Vol. Specific heat of dry air',
  thcl=0.026, remark='Bulk nominal thermal conductivity for energy equation',
  thexf1=0.003235, remark='Approximate thermal expansion coefficient',
  tstar = 285.7, remark='Reference temperature for thermal expansion coefficient',
              remark='THEXF1, TSTAR not needed for vapor transport model',
              remark='but are used in some parts of code to set up simulation',
$end

$scalar
  remark=' MOISTURE IS NOT USED HERE, BUT USE THE MODEL FOR CONSISTENCY WITH CODE VERSION',
  remark='Scalar 1 is the diffusing/advection water',
  remark='Scalar 2 (non-diffusing) is for storing the surface phase change flux values',
  remark='Scalar 3 (non-diffusing) is for storing the relative humidity values',
  remark='Scalar 4 (non-diffusing) is for storing the net surface liquid accumulation',
  remark='Scalar 5 (non-diffusing) is for storing the vapor water concentration',
  remark='Scalar 6 (non-diffusing) is for storing the liquid water (mist) concentration',
  remark='Scalar 7 (non-diffusing) is for storing the calculation iterations for the',
  remark=' wall mass flux',

```

```

remark='Scalar 8 (non-diffusing) is for storing the calculation iterations for the',
remark='      mist pahse change in the fluid interior',
remark='Scalars 7 and 8 are helpful in tuning the value of vaprlx if necessary',

```

```

nsc=8,
  isclr(1)=3, cmisc(1)=0.26e-04, scltit(1)='Tot.Water', rmisc=0.,
  isclr(2)=0, cmisc(2)=0., scltit(2)='Liq.Flux',
  isclr(3)=0, cmisc(3)=0., scltit(3)='Rel.Hum',
  isclr(4)=0, cmisc(4)=0., scltit(4)='Net.Liq',
  isclr(5)=0, cmisc(5)=0., scltit(5)='Vap.Wat',
  isclr(6)=0, cmisc(6)=0., scltit(6)='Liq.Wat',
  isclr(7)=0, cmisc(7)=0., scltit(7)='Itr.Wall',
  isclr(8)=0, cmisc(8)=0., scltit(8)='Itr.Mesh',
$end

```

```
$bcdata
```

```

  wl=1, wr=1,      remark=' Symmetry everywhere for 2-D and/or zero heat transfer',
  wf=1, wbk=1,
  wb=1, wt=1,
$end

```

```
$mesh
```

```

remark=' Put fixed points where the corners of the radiation surfaces are',
  px(1)=-0.51,      py(1)=0.,      pz(1)=-0.51,
  px(2)=-0.50,      py(2)=1.,      pz(2)=-0.50,
  px(3)=-0.35355,   pz(3)=-0.35355,
  px(4)=-0.15,      pz(4)=-0.15,
  px(5)=-0.10607,   pz(5)=-0.10607,
  px(6)=0.,         pz(6)= 0.0,
  px(7)= 0.10607,   pz(7)= 0.10607,
  px(8)= 0.15,      pz(8)= 0.15,
  px(9)= 0.35355,   pz(9)= 0.35355,
  px(10)= 0.50,     pz(10)= 0.50,
  px(11)= 0.51,     pz(11)= 0.51,
  nxcell(1)=1,      nzcell(1)=1,
  nxcell(2)=11,     nzcell(2)=11,
  nxcell(3)=14,     nzcell(3)=14,
  nxcell(4)=3,      nzcell(4)=3,
  nxcell(5)=8,      nzcell(5)=8,
  nxcell(6)=8,      nzcell(6)=8,
  nxcell(7)=3,      nzcell(7)=3,
  nxcell(8)=14,     nzcell(8)=14,
  nxcell(9)=11,     nzcell(9)=11,
  nxcell(10)=1,     nzcell(10)=1,
  nxcelt=74,        nycelt=1,      nzcelt=74,
$end

```

```
$obs
```

```

  avrck=-3.,
  nob = 2,
  tobs(1)=0., tobs(2)=1000.,
remark='Outer cylinder',
  ral(1)=0.50,  rotx(1)=90.,
  twobs(1,1)=300., twobs(2,1)=300.,
remark='Inner cylinder top',
  rah(2)=0.15,  rotx(2)=90.,
  twobs(1,2)=336.55,
  pobs(1,2)=50., pobs(2,2)=50.,
  rcobs(2)=1000., kobs(2)=1.2,
$end

```

```
$fl
```

```

  remark='No water in the air',
  sclri(1)=0.0,
  sclri(5)=0.0,
  sclri(6)=0.,
  presi=101325.,
$end

```



```
$bf
$end

$temp
  ntmp=1,
  tempi=300.,
$end

$motn
$end

$grafic
$end

$parts
$end

$usrdat

remark=' Define the radiation surfaces as portions of the two cylinders',
  nrsrf_stg=8,
remark=' Radiation surface at outer cylinder top',
  eps_stg(1) = .9,
  rad_stg(1,1) = 1.,
  rad_stg(2,1) = -0.35355,   rad_stg(3,1) = 0.35355,
  rad_stg(4,1) = -1.,       rad_stg(5,1) = 2.,
  rad_stg(6,1) = 0.,        rad_stg(7,1) = 0.6,

remark=' Radiation surface at outer cylinder right',
  eps_stg(2) = .9,
  rad_stg(1,2) = 1.,
  rad_stg(2,2) = 0.35355,   rad_stg(3,2) = 0.6,
  rad_stg(4,2) = -1.,       rad_stg(5,2) = 2.,
  rad_stg(6,2) = -0.35355, rad_stg(7,2) = 0.35355,

remark=' Radiation surface at outer cylinder bottom',
  eps_stg(3) = .9,
  rad_stg(1,3) = 1.,
  rad_stg(2,3) = -0.35355,   rad_stg(3,3) = 0.35355,
  rad_stg(4,3) = -1.,       rad_stg(5,3) = 2.,
  rad_stg(6,3) = -0.6,      rad_stg(7,3) = 0.,

remark=' Radiation surface at outer cylinder left',
  eps_stg(4) = .9,
  rad_stg(1,4) = 1.,
  rad_stg(2,4) = -0.6,       rad_stg(3,4) = -0.35355,
  rad_stg(4,4) = -1.,       rad_stg(5,4) = 2.,
  rad_stg(6,4) = -0.35355, rad_stg(7,4) = 0.35355,

remark=' Radiation surface at inner cylinder top',
  eps_stg(5) = .9,
  rad_stg(1,5) = 2.,
  rad_stg(2,5) = -0.2,       rad_stg(3,5) = 0.2,
  rad_stg(4,5) = -1.,       rad_stg(5,5) = 2.,
  rad_stg(6,5) = 0.10607,   rad_stg(7,5) = 0.2,

remark=' Radiation surface at inner cylinder right',
  eps_stg(6) = .9,
  rad_stg(1,6) = 2.,
  rad_stg(2,6) = 0.,         rad_stg(3,6) = 0.2,
  rad_stg(4,6) = -1.,       rad_stg(5,6) = 2.,
  rad_stg(6,6) = -0.10607,  rad_stg(7,6) = 0.10607,

remark=' Radiation surface at inner cylinder bottom',
  eps_stg(7) = .9,
  rad_stg(1,7) = 2.,
```

```

rad_stg(2,7) =-0.2,      rad_stg(3,7) = 0.2,
rad_stg(4,7) =-1.,      rad_stg(5,7) = 2.,
rad_stg(6,7) =-0.2,      rad_stg(7,7) =-0.10607,

remark=' Radiation surface at inner cylinder left',
eps_stg(8) = .9,
rad_stg(1,8) = 2.,
rad_stg(2,8) =-0.2,      rad_stg(3,8) = 0.,
rad_stg(4,8) =-1.,      rad_stg(5,8) = 2.,
rad_stg(6,8) =-0.10607, rad_stg(7,8) = 0.10607,

remark=' Let the compute compute the configuration factors',
iusrcf_stg = 0,

remark=' Specify vapor model paramters for consistent density with cases for phase
change',
istwtf_stg='tw',
isvap_stg = 1,
isliq_stg = 2,
isrh_stg = 3,
istlq_stg = 4,
isywv_stg = 5,
isywl_stg = 6,
hvvap_stg=2304900.,
cvvap_stg=1370.,
cvliq_stg=4186.,
rvap_stg=416.,
rgas_stg=289.,
vaprlx_stg=0.8,

rhlim_stg='y',

$end

```

End of listing of file prepin.2-D_conc-cyls_qtr

The analytical and FLOW-3D results for the concentric cylinder configuration factors are summarized in Table 3/20/06-1 and Table 3/20/06-2, respectively. The errors in the FLOW-3D computations are summarized in Table 3/20/06-3.

Table 3/20/06-1. Configuration Factors, F_{a-b} , 2-D Cylinders, Exact Solution

		Surface 'b'							
		1	2	3	4	5	6	7	8
Surface 'a'	1		0.2254	0.1358	0.2347	0.2196	0.0404		0.0404
	2	0.2254	0.0000	0.2347	0.1358	0.0404	0.2195	0.0404	0.0000
	3	0.1358	0.2347	0.0000	0.2347	0.0000	0.0404	0.2195	0.0404
	4	0.2347	0.1358	0.2347	0.0000	0.0404	0.0000	0.0404	0.2195
	5	0.7274	0.1339	0.0000	0.1339	0.0000	0.0000	0.0000	0.0000
	6	0.1339	0.7274	0.1339	0.0000	0.0000	0.0000	0.0000	0.0000
	7	0.0000	0.1339	0.7274	0.1339	0.0000	0.0000	0.0000	0.0000
	8	0.1339	0.0000	0.1339	0.7274	0.0000	0.0000	0.0000	0.0000

Table 3/20/06-2. Configuration Factors, F_{a-b} , 2-D Cylinders, FLOW-3D Results

		Surface 'b'							
		1	2	3	4	5	6	7	8
Surface 'a'	1	0.0000	0.2348	0.1307	0.2348	0.2199	0.0401	0.0000	0.0401
	2	0.2348	0.0000	0.2348	0.1307	0.0401	0.2199	0.0401	0.0000
	3	0.1307	0.2348	0.0000	0.2348		0.0401	0.2199	0.0401
	4	0.2348	0.1307	0.2348	0.0000	0.0401		0.0401	0.2199
	5	0.7330	0.1340	0.0000	0.1340	0.0000	0.0000	0.0000	0.0000
	6	0.1340	0.7330	0.1340	0.0000	0.0000	0.0000	0.0000	0.0000
	7	0.0000	0.1340	0.7330	0.1340	0.0000	0.0000	0.0000	0.0000
	8	0.1340	0.0000	0.1340	0.7330	0.0000	0.0000	0.0000	0.0000

Table 3/20/06-2. FLOW-3D Configuration Factors Errors, 2-D Cylinders

		Surface 'b'							
		1	2	3	4	5	6	7	8
Surface 'a'	1		4.19%	-3.81%	0.08%	0.13%	-0.82%		-0.82%
	2	4.19%		0.08%	-3.81%	-0.82%	0.14%	-0.82%	
	3	-3.81%	0.08%		0.08%		-0.82%	0.15%	-0.82%
	4	0.08%	-3.81%	0.08%		-0.82%		-0.82%	0.14%
	5	0.77%	0.11%		0.11%				
	6	0.10%	0.77%	0.10%					
	7		0.10%	0.77%	0.10%				
	8	0.10%		0.10%	0.77%				

It is seen that the errors are all less than 5%. Also, note that the configuration factor F_{1-2} should be the same as F_{2-3} . The FLOW-3D computations do not show these as being identical. To date, there has not been a reason determined for this discrepancy. It is perhaps related to a slight error in the geometry specifications relative to grid line locations or round-off in the computations.

Nonetheless, this is considered to be good agreement in light of the fact that most of these factors are affected by partial or full blockage by the inner cylinder

Two different FLOW-3D input file were created for the case of the rectangular enclosure. The first file,

prepin.3-D_rect-box

uses a grid of 40 x 20 x 10 cells to cover the 2 x 1 x 0.5 inside dimensions of the box. A single layer of cells over all six faces is used to define the walls of the box. The second input file uses a grid of double this resolution,

prepin.3-D_rect-box_x2

The fine-resolution file was used to check the grid independence of the coarse-grid results. Only the coarse-grid file is listed here. The changes to make the grid finer are straightforward.

Listing of file prepin.3-D_rect-box

3-D Rectangular box for CF calculation checking.

```
$xput
  remark='units are SI',
  itb=0,   ifvis=-1,   ifenrg=2, ifrho=1,  ihtc=2,  ipdis=1,
  iwsh=1,
  gz=-9.8,
  iusrd=1,
  delt=1.e-4, pltdt=50., sprtdt=1.,
  twfin=1000.,
  isolid=0,
  hpltdt=5.,
  rmrhoe=1.,
  rmrho=1.,
$end

$limits
$end

$props
  rhof=1.197,
  mul=2.e-05, units='si',
  cvl=717., thcl=0.026,
  tstar = 288.4,
  thexf1=0.00333,
$end

$scalar
  nsc=8,
  isclr(1)=3, cmisc(1)=0.26e-04, scltit(1)='Tot.Water', rmisc=0.,
  isclr(2)=0, cmisc(2)=0., scltit(2)='Liq.Flux',
  isclr(3)=0, cmisc(3)=0., scltit(3)='Rel.Hum',
  isclr(4)=0, cmisc(4)=0., scltit(4)='Net.Liq',
  isclr(5)=0, cmisc(5)=0., scltit(5)='Vap.Wat',
  isclr(6)=0, cmisc(6)=0., scltit(6)='Liq.Wat',
  isclr(7)=0, cmisc(7)=0., scltit(7)='Itr.Wall',
  isclr(8)=0, cmisc(8)=0., scltit(8)='Itr.Mesh',
$end

$bcdata
  wf=2, wbk=2, ipbctp(4)=1, ipbctp(3)=1,
  tbc(4)=300., tbc(3)=300.,
$end

$mesh
  px(1)=-1.05,   py(1)=-0.55,   pz(1)=-0.30,
  px(2)=-1.0,   py(2)=-0.50,   pz(2)=-0.25,
  px(3)= 1.0,   py(3)= 0.50,   pz(3)= 0.25,
  px(4)= 1.05,   py(4)= 0.55,   pz(4)= 0.30,
  nxcell(1)=1,   nycell(1)=1,   nzcell(1)=1,
  nxcell(2)=40,  nycell(2)=20,  nzcell(2)=10,
  nxcell(3)=1,   nycell(3)=1,   nzcell(3)=1,
  nxcelt=42,     nycelt=22,     nzcelt=12,
$end
```

```
$obs
  avrck=-3.,
  nobns = 6,
  tobs(1)=0., tobs(2)=1000.,
  remark='Left end. Heated',
  xh(1) = -1.0,
  twobs(1,1)=300.,
  pobs(1,1)=100., pobs(2,1)=100.,
  kobs(1)=1., rcobs(1)=1000.,
  remark='Right end. Cooled',
  xl(2)=1.,
  kobs(2)=1., rcobs(2)=1000.,
  twobs(1,2)=300.,
  pobs(1,2)=-100., pobs(2,2)=-100.,
  remark='Front end. Const Temp.',
  yh(3)=-0.5,
  twobs(1,3)=300., twobs(2,3)=300.,
  remark='Back end. Const Temp.',
  yl(4)= 0.5,
  twobs(1,4)=300., twobs(2,4)=300.,
  remark='Bottom end. Variable Temp., insulated from boundary',
  zh(5)=-0.25,
  kobs(5)=1., rcobs(5)=1000.,
  twobs(1,5)=300.,
  remark='Top end. Variable Temp., insulated from boundary',
  zl(6)= 0.25,
  kobs(6)=1., rcobs(6)=1000.,
  twobs(1,6)=300.,
  htcob(1,3)=0., htcob(1,4)=0., htcob(1,5)=0., htcob(1,6)=0.,
  htcob(2,3)=0., htcob(2,4)=0., htcob(2,5)=0., htcob(2,6)=0.,
  htcob(3,5)=0., htcob(3,6)=0.,
  htcob(4,5)=0., htcob(4,6)=0.,
$end

$fl
  remark=' 50RH at 300, 1 atm',
  sclri(1)=0.0,   remark='Total water concentration',
  sclri(5)=0.0,   remark='Water vapor concentration',
  sclri(6)=0.,     remark='Water liquid concentration',
  presi=101325.,
$end

$bf
$end

$temp
  ntmp=1,
  tempi=300.,
$end

$motn
$end

$grafic
$end

$parts
$end

$surdat

  nrsrf_stg=6,
  remark=' Left surface',
  eps_stg(2) = .9,
  rad_stg(1,2) = 1.,
  rad_stg(2,2) =-1.1, rad_stg(3,2) =-0.9,
  rad_stg(4,2) =-1., rad_stg(5,2) = 1.,
```

```
rad_stg(6,2) = -1.,   rad_stg(7,2) = 1.,

remark=' Right surface',
eps_stg(1) = .9,
rad_stg(1,1) = 2. ,
rad_stg(2,1) = 0.9,   rad_stg(3,1) = 1.1,
rad_stg(4,1) = -1.,   rad_stg(5,1) = 1.,
rad_stg(6,1) = -1.,   rad_stg(7,1) = 1.,

remark=' Front surface',
eps_stg(3) = .9,
rad_stg(1,3) = 3. ,
rad_stg(2,3) = -1.1,   rad_stg(3,3) = 1.1,
rad_stg(4,3) = -0.6,   rad_stg(5,3) = -0.4,
rad_stg(6,3) = -1.,   rad_stg(7,3) = 1.,

remark=' Back surface',
eps_stg(4) = .9,
rad_stg(1,4) = 4. ,
rad_stg(2,4) = -1.1,   rad_stg(3,4) = 1.1,
rad_stg(4,4) = 0.4,   rad_stg(5,4) = 0.6,
rad_stg(6,4) = -1.,   rad_stg(7,4) = 1.,

remark=' Bottom surface',
eps_stg(5) = .9,
rad_stg(1,5) = 5. ,
rad_stg(2,5) = -1.1,   rad_stg(3,5) = 1.1,
rad_stg(4,5) = -1.,   rad_stg(5,5) = 1.,
rad_stg(6,5) = -0.4,   rad_stg(7,5) = -0.2,

remark=' Top surface',
eps_stg(6) = .9,
rad_stg(1,6) = 6. ,
rad_stg(2,6) = -1.1,   rad_stg(3,6) = 1.1,
rad_stg(4,6) = -1.,   rad_stg(5,6) = 1.,
rad_stg(6,6) = 0.2,   rad_stg(7,6) = 0.4,

iusrcf_stg = 0,

istwtf_stg='tw',
isvap_stg = 1,
isliq_stg = 2,
isrh_stg = 3,
istlq_stg = 4,
isywv_stg = 5,
isywl_stg = 6,
hvvap_stg=2304900.,
cvvap_stg=1370.,
rvap_stg=416.,
rgas_stg=289.,
vaprlx_stg=0.8,

rhlim_stg='y',
$end
```

End of listing of file prepin.3-D_rect-box

The analytical results for the rectangular enclosure are summarized in Table 3/20/06-4. The FLOW-3D results for the coarse grid resolution are summarized in Table 3/20/06-5 and 3/20-06-6.

Table 3/20/06-4. Configuration Factors, F_{a-b} , 3-D Box, Exact Solution

		Surface 'b'					
		1	2	3	4	5	6
Surface 'a'	1		0.0362	0.1673	0.1673	0.3146	0.3146
	2	0.0362		0.1673	0.1673	0.3146	0.3146
	3	0.0837	0.0837		0.1653	0.3337	0.3337
	4	0.0837	0.0837	0.1653		0.3337	0.3337
	5	0.0787	0.0787	0.1669	0.1669		0.5090
	6	0.0787	0.0787	0.1669	0.1669	0.5090	

Table 3/20/06-5. Configuration Factors, F_{a-b} , 3-D Box, FLOW-3D Coarse Results

		Surface 'b'					
		1	2	3	4	5	6
Surface 'a'	1	0.00000	0.03619	0.17407	0.17407	0.32800	0.32800
	2	0.03619	0.00000	0.17407	0.17407	0.32800	0.32800
	3	0.08703	0.08703	0.00000	0.16539	0.34707	0.34707
	4	0.08703	0.08703	0.16539	0.00000	0.34707	0.34707
	5	0.08200	0.08200	0.17353	0.17353	0.00000	0.50946
	6	0.08200	0.08200	0.17353	0.17353	0.50946	0.00000

Table 3/20/06-6. FLOW-3D (Coarse) Configuration Factor Errors, 3-D Box

		Surface 'b'					
		1	2	3	4	5	6
Surface 'a'	1		0.04%	4.04%	4.04%	4.26%	4.26%
	2	0.04%		4.04%	4.04%	4.26%	4.26%
	3	4.04%	4.04%		0.07%	4.00%	4.00%
	4	4.04%	4.04%	0.07%		4.00%	4.00%
	5	4.26%	4.26%	4.00%	4.00%		0.09%
	6	4.26%	4.26%	4.00%	4.00%	0.09%	

It is seen that the errors are all less than 5%. Also, note that there is reasonable symmetry to the factors.

The FLOW-3D results for the fine grid resolution are summarized in Table 3/20/06-7 and 3/20-06-8.

Table 3/20/06-7. Configuration Factors, F_{a-b} , 3-D Box, FLOW-3D Fine Results

		Surface 'b'					
		1	2	3	4	5	6
Surface 'a'	1	0.00000	0.03618	0.17069	0.17069	0.32133	0.32133
	2	0.03618	0.00000	0.17069	0.17069	0.32133	0.32133
	3	0.08535	0.08535	0.00000	0.16530	0.34043	0.34043
	4	0.08535	0.08535	0.16530	0.00000	0.34043	0.34043
	5	0.08033	0.08033	0.17022	0.17022	0.00000	0.50911
	6	0.08033	0.08033	0.17022	0.17022	0.50911	0.00000

Table 3/20/06-8. FLOW-3D (Fine) Configuration Factor Errors, 3-D Box

		Surface 'b'					
		1	2	3	4	5	6
Surface 'a'	1		0.01%	2.02%	2.02%	2.14%	2.14%
	2	0.01%		2.02%	2.02%	2.14%	2.14%
	3	2.02%	2.02%		0.02%	2.01%	2.01%
	4	2.02%	2.02%	0.02%		2.01%	2.01%
	5	2.14%	2.14%	2.02%	2.02%		0.02%
	6	2.14%	2.14%	2.02%	2.02%	0.02%	

The fine-grid results are an improvement over the coarse grid results in that the errors are approximately halved. It was found that the configuration factor computations for the fine-grid solution took considerably longer than in the coarse-grid. This is expected and leads to a possible suggested approach for future problems. Once the configuration factors have been computed satisfactorily, the user should consider specifying the factors in subsequent runs and restarts so that the factors do not have to be re-computed each time.

END OF ENTRY FOR 3/20/06

STG

3/31/06

STG

This entry documents the work performed on the last of the planned validation tests of the custom moisture transport and radiation modules for FLOW-3D. The objective of this particular test case is to validate that the two modules will work simultaneously in a CFD simulation that has a natural convection flow.

This test case is for the heat and mass transfer in a 2-D square enclosure. The inside dimensions of the enclosure are 10 cm x 10 cm. The vertical sidewalls are each 2.5 cm thick and have thermal conductivity of 1 W/(m*K). The left wall has an internal heat generation of 8000 W/m³ so that 20 W enters the enclosure per meter of depth. The outside surface of the right wall is held constant at 300 K. The acceleration due to gravity is only 0.001 g. This condition ensures that the convection flow is laminar.

Four different scenarios are investigated in this test case, depending on the radiation and moisture conditions of the surfaces:

- 1) The vertical walls are dry and radiation is neglected so that heat is transferred only by conduction and convection. This type of scenario has already been used a validation case, but this particular scenario is included as a basis for comparing the individual and combined effects of moisture transport and radiation.
- 2) The vertical walls are dry and heat transfer via radiation is allowed in addition to conduction and convection in the dry air. In this scenario, the radiation contributions of the horizontal upper and lower walls are neglected. This can be envisioned as these walls being non-reflecting and non-absorbing
- 3) The vertical walls are wet and heat transfer via radiation is ignored. In this scenario, the horizontal upper and lower walls do not allow for moisture to either evaporate from or condense on their respective surfaces.
- 4) The vertical walls exchange heat via radiation as in 2) and moisture is transported as in 3). This scenario tests the all the heat and mass transfer modes acting together.

The FLOW-3D simulations are to be compared to analysis results for the respective scenarios using appropriate empirical heat transfer correlations for laminar natural convection.

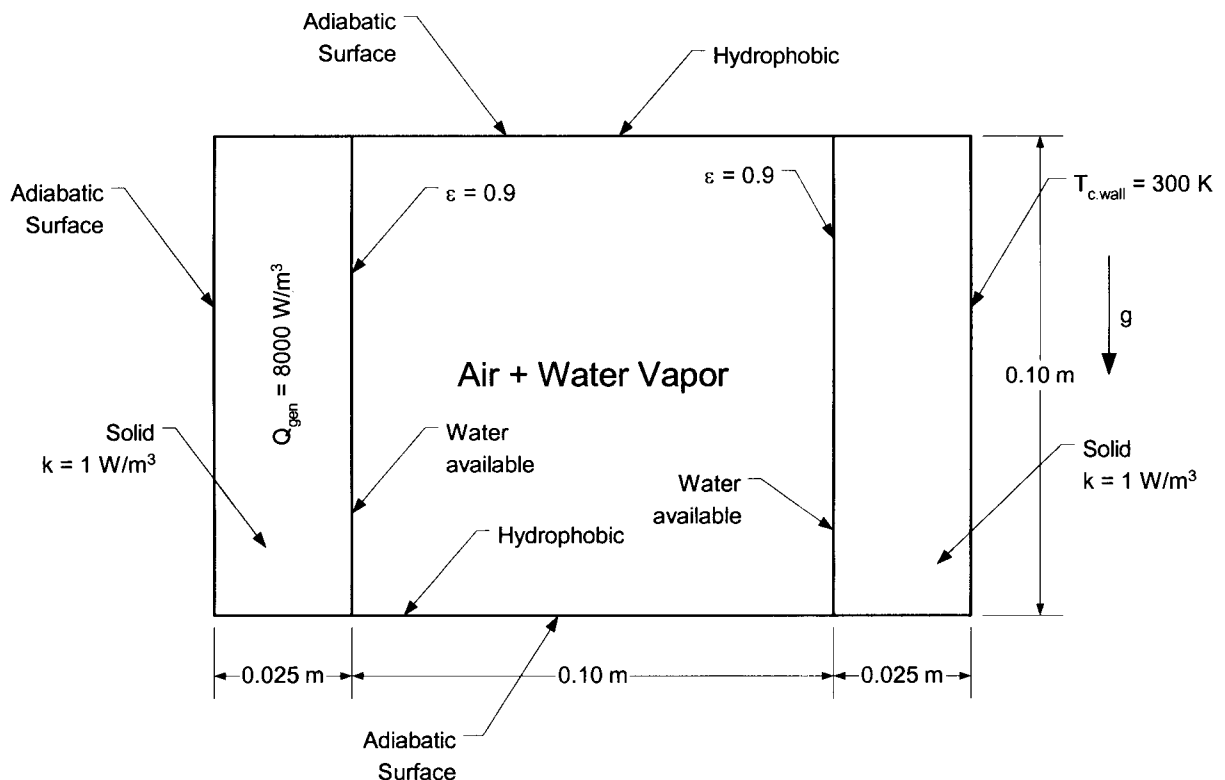


Figure 3/30/06-1. Heat and Mass Transfer in a Square Enclosure

The analysis of these four scenarios is described in the Mathcad file,

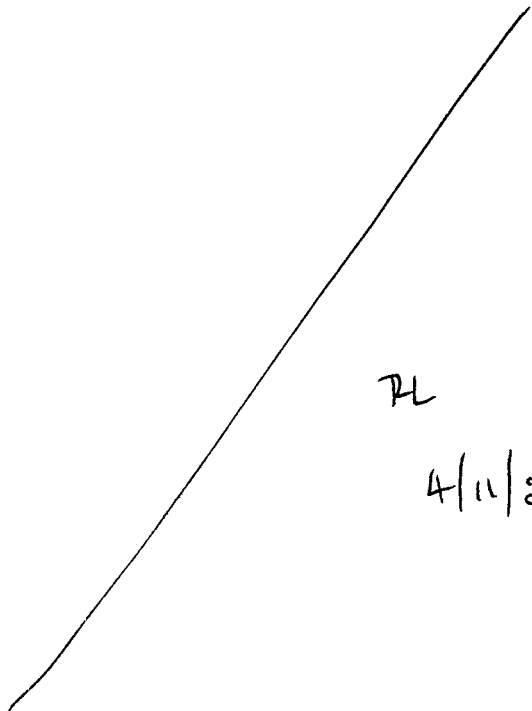
Sq-Box_Conv_Rad_PCG.mcd

The analysis equations and references are listed there along with a comparison of the predictions and the corresponding FLOW-3D simulation results. A listing of this Mathcad file is on the following 12 pages.

The FLOW-3D input files corresponding to these four scenarios are

1. prepin.2-D_box_norad_nopcg_001g
2. prepin.2-D_box_rad_nopcg_001g
3. prepin.2-D_box_norad_pcg_001g
4. prepin.2-D_box_rad_pcg_001g

The first of these files, prepin.2-D_box_norad_nopcg_001g is listed after the Mathcad file for reference. Note that all the lines necessary for specifying the other three scenarios are at the end of this listing in a comments section of the file. The other files are created by simply moving the appropriate lines to the \$usrdat namelist block.



PL
4/11/2008

Heat and Mass Transfer in a 2-D Square Enclosure

Purpose

The analysis presented here serves as a validation case for the radiation and moisture modules developed for FLOW-3D Version 9.0 for the CNWRA. The modules are described in CNWRA Notebook #536E and in the Software Requirements Document for these modules.

References

1. Berkovsky, B. M., and Polevikov, V. K., "Numerical Study of Problems on High-Intensive Free Convection," Heat Transfer and Turbulent Convection, Spalding, D. B., and Afgan, H., eds., Volumes I and II, Hemisphere Publishing, pp., 443-455.
2. Green, S., CNWRA Scientific Notebook #536E.
3. Green, S., Software Requirements Description for the Modification of OF FLOW-3D to Include High-Humidity Moisture Transport Model and Thermal Radiation Effects Specific to Repository Drifts,
4. Incropera, F., DeWitt, Fundamentals of Heat and Mass Transfer, Second Edition, John Wiley and Sons, Inc., 1985.
5. Siegel, R., Howell, W., Radiation Heat Transfer, Hemisphere Publishing, Washington, D. C. 1992.

Introduction

The problem is described schematically in Figure 1 below. A 2-D square enclosure has sides measuring 0.1 m at the internal surface. The box is bounded on the top and bottom by adiabatic wall. The left wall generates 20 W per meter of length uniformly within its 0.025 m thickness and 0.1 m height. The right wall is also 0.025 m thick and its outer surface is held at a constant temperature of 300 K. The left wall and right walls can allow for the evaporation and condensation of water in response to the temperature and flow conditions as required. Radiation between the left and right walls is to be considered.

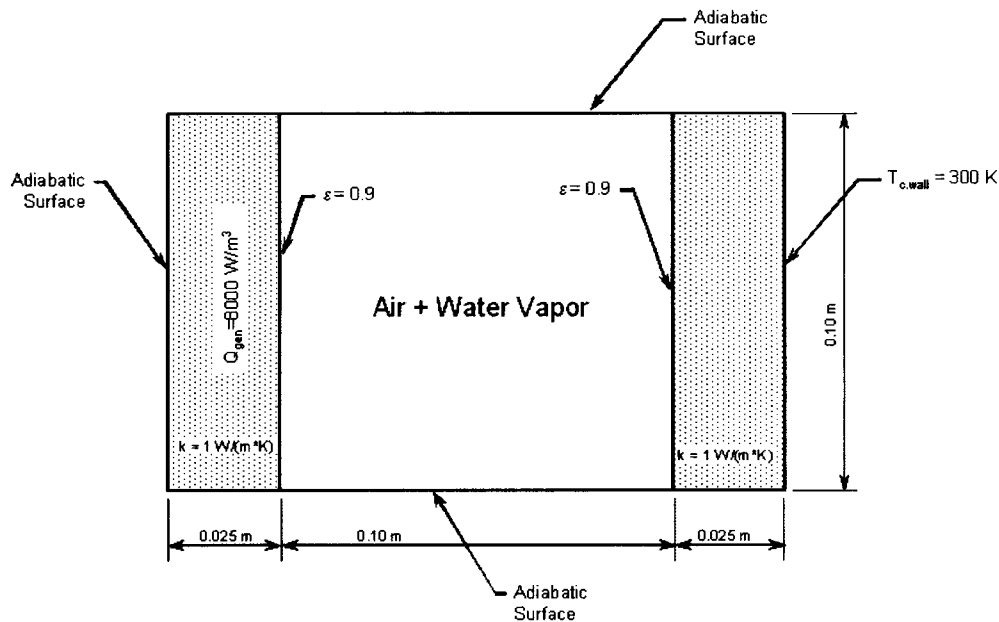


Figure 1. 2-D Square Enclosure

Assumptions

1. Air and water vapor act as ideal gases in accordance with the special CNWRA FLOW-3D moisture model.
2. The upper and lower enclosure surfaces do not interact with any surface with respect to thermal radiation .
3. Laminar flow is to be specified/analyzed to ensure that correlations are as accurate as possible.
4. Wall surfaces are gray and diffuse with respect to radiation heat transfer.

Approach

The analysis will be carried out in several stages. First, the heat transfer by convection alone will be considered. Next the radiation effects will be included with the convection processes while neglecting the phase change effect. Third, the phase change effects will be considered with the natural convection while neglecting the radiate exchange. Finally, all the heat transfer and mass transfer modes will be considered acting in concert.

Properties**Thermal properties**

$$P_{\text{tot}} := 1 \cdot \text{atm}$$

Total Pressure in system

$$R_a := 289 \cdot \frac{\text{joule}}{\text{kg} \cdot \text{K}}$$

Ideal gas constant for air

$$C_{v,a} := 717 \cdot \frac{\text{joule}}{\text{kg} \cdot \text{K}}$$

$$C_{p,da} := R_a + C_{v,a}$$

Specific heat for dry air

$$\rho_{da}(T) := \frac{P_{\text{tot}}}{R_a \cdot T}$$

Dry air density

$$\mu_a := 2 \cdot 10^{-5} \cdot \text{Pa} \cdot \text{sec}$$

Air viscosity

$$k_a := 0.026 \cdot \frac{\text{watt}}{\text{m} \cdot \text{K}}$$

Air thermal conductivity

$$Pr_{da} := \frac{\mu_a C_{p,da}}{k_a}$$

Air Prandtl Number

$$k_w := 1 \cdot \frac{\text{watt}}{\text{m} \cdot \text{K}}$$

Wall thermal conductivity

Geometry, etc.

$$L_e := 0.1 \cdot \text{m}$$

Enclosure sides

$$t_w := 0.025 \cdot \text{m}$$

Left and right wall thickness

$$a_{\text{grav}} := 0.001 \cdot \text{g}$$

Adjust gravity to set the Rayleigh number to get laminar flow.

$$A_w := L_e \cdot 1 \cdot \text{m}$$

Wall area per unit depth. Assume 1 m depth to box to make units checking easier.

$$V_w := A_w \cdot t_w$$

Volume of the walls.

For use in the equation for the 1-D temperature distribution in the heated wall

$$\varepsilon_h := 0.9 \quad \varepsilon_c := 0.9$$

Emissivity of the wall interior surfaces.

Boundary Conditions

$$T_{c.wall} := 300 \cdot K$$

$$Q_{h.wall} := 20 \cdot \text{watt}$$

$$Q_{h.wall.vol} := \frac{Q_{h.wall}}{V_w}$$

Natural Convection Only

For a square enclosure with vertical heated walls, Berkovsky and Polevikov show that the Nusselt Number is given by

$$Nu_{sq.box} = 0.18 \cdot \left(\frac{Pr_a}{0.2 + Pr_a} \cdot Ra \right)^{0.29} \quad \text{For the ranges} \quad \begin{array}{l} 10^3 < \frac{Ra \cdot Pr}{(0.2 + Pr)} \\ 10^{-3} < Pr < 10^5 \end{array} \quad \text{Eq. 1}$$

where

$$Ra = \frac{\rho_a^2 \cdot a_{grav} \beta}{\mu_a^2} (T_{h.surf} - T_{c.surf}) \cdot L_e^3 \cdot Pr_a \quad \text{Rayleigh Number} \quad \text{Eq. 2}$$

$$\beta = \frac{1}{T_{mean}} \quad (\text{approximately}) \quad \text{Thermal expansion coefficient for ideal gas}$$

$$T_{mean} = 0.5 \cdot (T_{h.surf} + T_{c.surf}) \quad \text{Use the mean temperature to specify the thermal expansion coefficient.}$$

$$T_{h.surf} \quad T_{c.surf} \quad \text{Hot and cold wall surface temperature, respectively}$$

The other variables are defined in the Properties section above. The heat transfer coefficient is obtained using the definition of the Nusselt Number,

$$h_{nc} = \frac{Nu_{sq.box} \cdot k_a}{L_e} \quad \text{Eq. 3}$$

The steady state heat transfer can now be defined

$$Q_{h.wall} = Q_{h.wall.vol} V_w = h_{nc} \cdot A_w (T_{h.surf} - T_{c.surf}) \quad \text{Eq. 4}$$

The heat transfer at the wall is specified, but the cold wall surface is not known. The cold wall surface temperature will be greater than the exterior wall surface temperature, $T_{c.wall}$, because of the conduction through the wall. Under steady conditions, the cold wall inner surface temperature is computed from the overall energy balance,

$$Q_{c.wall} = Q_{h.wall} = k_w \cdot A_w \cdot \frac{(T_{c.surf} - T_{c.wall})}{t_w} \quad \text{Eq. 5}$$

Equations 1 through 5 can be solved for the wall surface temperatures. The solution is iterative because of the dependence of the Rayleigh Number on the wall temperature. The number of solution equations can be reduced by substituting Eqs. 3, 2, and 1 into Eq. 4

$$Q_{h.wall} = 0.18 \cdot \left[\frac{Pr_a \cdot \rho_a^2 \cdot a_{grav} \cdot (T_{h.surf} - T_{c.surf}) \cdot L_e^3 \cdot Pr_a}{0.2 + Pr_a \cdot \mu_a^2 \cdot 0.5 \cdot (T_{h.surf} + T_{c.surf})} \right]^{0.29} \cdot \frac{k_a}{L_e} \cdot A_{wall} (T_{h.surf} - T_{c.surf}) \quad \text{Eq. 6}$$

Equations 5 and 6 form a set of two equations with two unknowns $T_{h.surf}$ and $T_{c.surf}$

$$Ra_1(T_h, T_c) := \frac{\rho_{da} [0.5 \cdot (T_h + T_c)]^2 \cdot a_{grav} \cdot (T_h - T_c)}{\mu_a^2 \cdot 0.5 \cdot (T_h + T_c)} \cdot L_e^3 \cdot Pr_{da}$$

$$Nu_{1.sq.box}(T_h, T_c) := 0.18 \cdot \left(\frac{Pr_{da}}{0.2 + Pr_{da}} \cdot Ra_1(T_h, T_c) \right)^{0.29}$$

$$h_{1.nc}(T_h, T_c) := \frac{Nu_{1.sq.box}(T_h, T_c) \cdot k_a}{L_e}$$

Use the Mathcad Given/Find method to solve these equations. First, start the solution with a guess at the values:

$$T_{c.surf} := T_{c.wall} + 1 \cdot K \quad T_{h.surf} := T_{c.surf} + 100 \cdot K$$

Define the solution equations.

Given

$$Q_{h.wall} = k_w \cdot A_w \cdot \frac{(T_{c.surf} - T_{c.wall})}{t_w}$$

$$Q_{h.wall} = h_{1.nc}(T_{h.surf}, T_{c.surf}) \cdot A_w \cdot (T_{h.surf} - T_{c.surf})$$

Solve the equations,

$$\begin{pmatrix} T_{h.surf} \\ T_{c.surf} \end{pmatrix} := \text{Find}(T_{h.surf}, T_{c.surf})$$

$$T_{h.surf} = 650.007 K \quad \text{Temperature of the hot wall surface}$$

$$T_{c.surf} = 305.000 K \quad \text{Temperature of the cold wall surface.}$$

These are the surface temperatures for the case of no thermal radiation and no phase change at the walls.

Check the value of the Rayleigh Number to make sure the correlation is valid for this condition.

$$\frac{Ra_1(T_{h,\text{surf}}, T_{c,\text{surf}}) \cdot Pr_{da}}{0.2 + Pr_{da}} = 5.872 \times 10^3$$

$$Nu_{1,\text{sq.box}}(T_{h,\text{surf}}, T_{c,\text{surf}}) = 2.23$$

The FLOW-3D results for this scenario are from **flsgrf.2-D_box_norad_nopcg_001g**

$$Nu_{\text{FLOW3D}} := 2.25 \quad T_{h,\text{avg.FLOW3D}} := 646.2 \cdot \text{K} \quad T_{c,\text{avg.FLOW3D}} := 304.5 \cdot \text{K}$$

$$\text{Var}_{Nu} := \frac{Nu_{\text{FLOW3D}} - Nu_{1,\text{sq.box}}(T_{h,\text{surf}}, T_{c,\text{surf}})}{Nu_{\text{FLOW3D}}} \quad \text{Var}_{Nu} = 0.906\%$$

$$\text{Var}_{T,h} := \frac{T_{h,\text{avg.FLOW3D}} - T_{h,\text{surf}}}{T_{h,\text{avg.FLOW3D}} - T_{c,\text{avg.FLOW3D}}} \quad \text{Var}_{T,h} = -1.114\%$$

$$\text{Var}_{T,c} := \frac{T_{c,\text{avg.FLOW3D}} - T_{c,\text{surf}}}{T_{h,\text{avg.FLOW3D}} - T_{c,\text{avg.FLOW3D}}} \quad \text{Var}_{T,c} = -0.146\%$$

TL

4/11/2008

Natural Convection with Thermal Radiation Heat Transfer

Now consider the radiation heat transfer between the two active walls. Using the methods of Siegel and Howell, the radiation heat transfer can be expressed as

$$Q_{h.c.r} = \frac{\sigma \cdot F_{h.c} \cdot A_w}{\left[\frac{1}{\epsilon_h} + F_{h.c} \cdot \frac{(1 - \epsilon_c)}{\epsilon_c} \right]} \cdot (T_{h.surf}^4 - T_{c.surf}^4)$$

where

$Q_{h.c.r}$ Heat transfer rate from the hot wall to the cold wall by radiation

ϵ_h, ϵ_c Emissivity of the hot and cold surfaces, respectively

$F_{h.c}$ Radiation configuration factor of the hot wall to the cold wall

$\sigma := 5.67 \cdot 10^{-8} \frac{\text{watt}}{\text{m}^2 \cdot \text{K}^4}$ Stefan-Boltzmann constant

The other variables are described in the Properties section above. With radiation as a heat transfer mode, the energy balance is now as follows.

$$Q_{h.wall} = Q_{c.wall} = Q_{h.c.conv} + Q_{h.c.rad}$$

Heat is transferred from the hot wall to the cold wall by convection and radiation. The sum of the convection and radiation heat transfers are equal to the heat generation rate in the heated wall and the heat transfer through the cooled wall via conduction.

The radiation configuration is computed by the Hottel method described by Siegel and Howell,

$$F_{h.c} := \frac{2 \cdot L_e \cdot \sqrt{2} - 2 \cdot L_e}{2 \cdot L_e} \quad F_{h.c} = 0.41421$$

We can now turn to the solution of the two equations

Given

$$Q_{h.wall} = k_w \cdot A_w \cdot \frac{(T_{c.surf} - T_{c.wall})}{t_w}$$

$$Q_{h.wall} = h_{l.nc} (T_{h.surf} - T_{c.surf}) \cdot A_w \cdot (T_{h.surf} - T_{c.surf}) \dots \\ + \frac{\sigma \cdot F_{h.c} \cdot A_w}{\left[\frac{1}{\epsilon_h} + F_{h.c} \cdot \frac{(1 - \epsilon_c)}{\epsilon_c} \right]} \cdot (T_{h.surf}^4 - T_{c.surf}^4)$$

$$\begin{pmatrix} T_{h.surf} \\ T_{c.surf} \end{pmatrix} := \text{Find}(T_{h.surf}, T_{c.surf})$$

$$T_{h.surf} = 362.069 \text{ K} \quad \text{Temperature of the hot wall surface}$$

$$T_{c.surf} = 305 \text{ K} \quad \text{Temperature of the cold wall surface.}$$

These are the surface temperatures for the case of convection plus radiation .

Check the value of the Rayleigh Number to make sure the correlation is valid for this condition.

$$\frac{\text{Ra}_l(T_{h.surf}, T_{c.surf}) \cdot \text{Pr}_{da}}{0.2 + \text{Pr}_{da}} = 2.85 \times 10^3$$

$$\text{Nu}_{1.sq.box}(T_{h.surf}, T_{c.surf}) = 1.808$$

*The FLOW-3D results for this scenario are from **fsgrf.2-D_box_rad_nopcg_001g***

$$\text{Nu}_{\text{FLOW3D}} := 1.52 \quad T_{h.avg.\text{FLOW3D}} := 362.7 \cdot \text{K} \quad T_{c.avg.\text{FLOW3D}} := 304.5 \cdot \text{K} \quad F_{h.c.\text{FLOW3D}} := 0.41427$$

$$\text{Var}_{\text{Nu}} := \frac{\text{Nu}_{\text{FLOW3D}} - \text{Nu}_{1.sq.box}(T_{h.surf}, T_{c.surf})}{\text{Nu}_{\text{FLOW3D}}} \quad \text{Var}_{\text{Nu}} = -18.946 \%$$

$$\text{Var}_{T,h} := \frac{T_{h.avg.\text{FLOW3D}} - T_{h.surf}}{T_{h.avg.\text{FLOW3D}} - T_{c.avg.\text{FLOW3D}}} \quad \text{Var}_{T,h} = 1.084 \%$$

$$\text{Var}_{T,c} := \frac{T_{c.avg.\text{FLOW3D}} - T_{c.surf}}{T_{h.avg.\text{FLOW3D}} - T_{c.avg.\text{FLOW3D}}} \quad \text{Var}_{T,c} = -0.859 \%$$

$$\text{Var}_{F,h,c} := F_{h.c.\text{FLOW3D}} - F_{h,c} \quad \text{Var}_{F,h,c} = 5.644 \times 10^{-3} \%$$

$$Q_{h.c.conv} := h_{l.nc}(T_{h.surf}, T_{c.surf}) \cdot A_w \cdot (T_{h.surf} - T_{c.surf}) \quad Q_{h.c.conv} = 2.683 \text{ W} \quad Q_{conv.\text{FLOW3D}} := 2.3 \cdot \text{watt}$$

$$Q_{h.c.rad} := \frac{\sigma \cdot F_{h.c} \cdot A_w}{\left[\frac{1}{\epsilon_h} + F_{h.c} \cdot \frac{(1 - \epsilon_c)}{\epsilon_c} \right]} \cdot (T_{h.surf}^4 - T_{c.surf}^4) \quad Q_{h.c.rad} = 17.317 \text{ W} \quad Q_{rad.\text{FLOW3D}} := 17.7 \cdot \text{watt}$$

Natural Convection with Moisture Transport

Now consider the transport of water through the enclosure where evaporation takes place at the left wall and condensation occurs at the right wall

Water Properties

Properties of water vapor and liquid water must be defined for the humidity analysis and diffusion rates

$$R_v := 416 \frac{\text{joule}}{\text{kg}\cdot\text{K}} \quad \text{Ideal gas constant for water vapor}$$

$$C_{vv} := 1370 \frac{\text{joule}}{\text{kg}\cdot\text{K}} \quad C_{pv} := C_{vv} + R_v \quad \text{Specific heat of water vapor}$$

$$h_{fg} := 2304900 \frac{\text{joule}}{\text{kg}} \quad \text{Heat of vaporization}$$

$$D_{v,a} := 2.6 \cdot 10^{-5} \frac{\text{m}^2}{\text{sec}} \quad \text{Diffusion coefficient for air and water vapor}$$

To define the saturation vapor pressure of water vapor, use the correlation of Keenan, Keyes, Hill, and Moore,

$$F_0 := -741.9242 \quad F_2 := -11.55286 \quad F_4 := 0.1094098 \quad F_6 := 0.2520658$$

$$F_1 := -29.721 \quad F_3 := -0.8685635 \quad F_5 := 0.439993 \quad F_7 := 0.05218684$$

$$FP_{\text{vsat}}(T) := 217.99 \cdot \exp \left[\frac{0.01}{T} \cdot [374.136 - (T - 273.15)] \cdot \sum_{k=0}^7 [F_k \cdot [0.65 - 0.01 \cdot (T - 273.15)]^k] \right] \text{atm}$$

$$\rho_v(T) := \frac{FP_{\text{vsat}}\left(\frac{T}{K}\right)}{R_v \cdot T} \quad \text{Water vapor density}$$

$$\rho_{\text{tot}}(T) := \rho_{\text{da}}(T) + \rho_v(T) \quad \text{Density of moist air at saturation conditions}$$

$$C_{p,\text{net}}(T) := \frac{C_{p,\text{da}} \cdot \rho_{\text{da}}(T) + C_{p,v} \cdot \rho_v(T)}{\rho_{\text{tot}}(T)} \quad \text{Effective specific heat of moist air}$$

$$Pr_{\text{mix}}(T) := \frac{\mu_a \cdot C_{p,\text{net}}(T)}{k_a} \quad \text{Effective Prandtl No. of moist air}$$

$$Le(T) := \frac{D_{v,a} \cdot \rho_{\text{tot}}(T) \cdot C_{p,\text{net}}(T)}{k_a} \quad \text{Lewis No. (ratio of species diffusion to thermal diffusion)}$$

$$Sc_{\text{mix}}(T) := \frac{\mu_a}{\rho_{\text{tot}}(T) D_{v,a}} \quad \text{Schmidt No. (ratio of momentum diffusion to species diffusion)}$$

$$Ra_2(T_h, T_c) := \frac{\rho_{tot} [0.5 \cdot (T_h + T_c)]^2 \cdot a_{grav} \cdot (T_h - T_c)}{\mu_a^2} \cdot L_e^3 \cdot Pr_{mix} [0.5 \cdot (T_h + T_c)] \quad \text{Mixture Rayleigh No.}$$

$$Nu_{2.sq.box}(T_h, T_c) := 0.18 \cdot \left[\frac{Pr_{mix} [0.5 \cdot (T_h + T_c)]}{0.2 + Pr_{mix} [0.5 \cdot (T_h + T_c)]} \cdot Ra_2(T_h, T_c) \right]^{0.29} \quad \text{Nusselt for Nat. Convection}$$

The Sherwood Number for mass transfer is obtained by following the established methodology of substituting the Schmidt No. for the Prandtl No. in the Nusselt Number equation. See Incropera and Dewitt.

$$Sh_{2.sq.box}(T_h, T_c) := 0.18 \cdot \left[\frac{Sc_{mix} [0.5 \cdot (T_h + T_c)]}{0.2 + Sc_{mix} [0.5 \cdot (T_h + T_c)]} \cdot Ra_2(T_h, T_c) \right]^{0.29} \quad \text{Sherwood No. for Mass Transfer}$$

$$h_{2.nc}(T_h, T_c) := \frac{Nu_{2.sq.box}(T_h, T_c) \cdot k_a}{L_e} \quad \text{Heat Transfer coefficient}$$

$$h_{2.m}(T_h, T_c) := \frac{Sh_{2.sq.box}(T_h, T_c) \cdot D_{v,a}}{L_e} \quad \text{Mass Transfer coefficient}$$

$$m\dot{f}_{2.vap}(T_h, T_c) := h_{2.m}(T_h, T_c) \cdot A_w \cdot (\rho_v(T_h) - \rho_v(T_c)) \quad \text{Mass transfer rate}$$

The total heat transfer rate across the box is due to natural convection heat transfer and the energy transfer via the phase change. Use the Mathcad Given/Find method to solve the system of equations

Given

$$Q_{h.wall} = k_w \cdot A_w \cdot \frac{(T_{c.surf} - T_{c.wall})}{t_w}$$

$$Q_{h.wall} = h_{2.nc}(T_{h.surf}, T_{c.surf}) \cdot A_w \cdot (T_{h.surf} - T_{c.surf}) + m\dot{f}_{2.vap}(T_{h.surf}, T_{c.surf}) \cdot h_{fg}$$

$$\begin{pmatrix} T_{h.surf} \\ T_{c.surf} \end{pmatrix} := \text{Find}(T_{h.surf}, T_{c.surf})$$

$T_{h.surf} = 342.512 \text{ K}$ *Temperature of the hot wall surface*

$T_{c.surf} = 305 \text{ K}$ *Temperature of the cold wall surface.*

These are the surface temperatures for the case of convection plus radiation .

Check the value of the Rayleigh Number to make sure the correlation is valid for this condition.

$$\frac{Ra_2(T_{h,\text{surf}}, T_{c,\text{surf}}) \cdot Pr_{\text{mix}}^{0.5} (T_{h,\text{surf}} + T_{c,\text{surf}})}{0.2 + Pr_{\text{mix}}^{0.5} (T_{h,\text{surf}} + T_{c,\text{surf}})} = 2.604 \times 10^3$$

$$Nu_{2,\text{sq},\text{box}}(T_{h,\text{surf}}, T_{c,\text{surf}}) = 1.761$$

The FLOW-3D results for this scenario are from **flsgrf.2-D_box_norad_pcg_001g**

$$Nu_{\text{FLOW3D}} := 1.85 \quad T_{h,\text{avg},\text{FLOW3D}} := 340.9 \cdot \text{K} \quad T_{c,\text{avg},\text{FLOW3D}} := 304.5 \cdot \text{K}$$

$$\text{Var}_{Nu} := \frac{Nu_{\text{FLOW3D}} - Nu_{2,\text{sq},\text{box}}(T_{h,\text{surf}}, T_{c,\text{surf}})}{Nu_{\text{FLOW3D}}} \quad \text{Var}_{Nu} = 4.805\%$$

$$\text{Var}_{T,h} := \frac{T_{h,\text{avg},\text{FLOW3D}} - T_{h,\text{surf}}}{T_{h,\text{avg},\text{FLOW3D}} - T_{c,\text{avg},\text{FLOW3D}}} \quad \text{Var}_{T,h} = -4.429\%$$

$$\text{Var}_{T,c} := \frac{T_{c,\text{avg},\text{FLOW3D}} - T_{c,\text{surf}}}{T_{h,\text{avg},\text{FLOW3D}} - T_{c,\text{avg},\text{FLOW3D}}} \quad \text{Var}_{T,c} = -1.374\%$$

$$Q_{h,c,\text{conv}} := h_{2,\text{nc}}(T_{h,\text{surf}}, T_{c,\text{surf}}) \cdot A_w \cdot (T_{h,\text{surf}} - T_{c,\text{surf}})$$

$$Q_{h,c,\text{vap}} := mf_{2,\text{vap}}(T_{h,\text{surf}}, T_{c,\text{surf}}) \cdot h_{fg}$$

$$Q_{h,c,\text{conv}} = 1.718 \text{ W} \quad Q_{h,c,\text{vap}} = 18.282 \text{ W}$$

$$Q_{\text{conv},\text{FLOW3D}} := 1.75 \cdot \text{watt} \quad Q_{\text{vap},\text{FLOW3D}} := 18.2 \cdot \text{watt}$$

$$mf_{2,\text{vap}}(T_{h,\text{surf}}, T_{c,\text{surf}}) = 7.932 \times 10^{-6} \frac{\text{kg}}{\text{s}}$$

$$mf_{\text{vap},\text{FLOW3D}} := 7.91 \cdot 10^{-6} \frac{\text{kg}}{\text{sec}}$$

$$\text{Var}_{mf} := \frac{mf_{\text{vap},\text{FLOW3D}} - mf_{2,\text{vap}}(T_{h,\text{surf}}, T_{c,\text{surf}})}{mf_{\text{vap},\text{FLOW3D}}} \quad \text{Var}_{mf} = -0.278\%$$

Natural Convection with Moisture Transport and Radiation Heat Transfer

Now consider the combination of all three modes of energy transfer. The conservation of energy states that the heat transfer rate through right wall is the sum of the convection heat transfer, radiation heat transfer and the energy transferred via the mass transfer of water vapor.

Given

$$Q_{h.wall} = k_w \cdot A_w \cdot \frac{(T_{c.surf} - T_{c.wall})}{t_w} \quad \text{Conduction heat transfer through the wall}$$

$$Q_{h.wall} = h_{2.nc}(T_{h.surf}, T_{c.surf}) \cdot A_w (T_{h.surf} - T_{c.surf}) + m \dot{f}_{2.vap}(T_{h.surf}, T_{c.surf}) \cdot h_{fg} \dots \quad \text{Convection heat transfer across box}$$

$$+ \frac{\sigma \cdot F_{h.c} \cdot A_w}{\left[\frac{1}{\epsilon_h} + F_{h.c} \frac{(1 - \epsilon_c)}{\epsilon_c} \right]} (T_{h.surf}^4 - T_{c.surf}^4) \quad \text{Radiation heat transfer from hot wall to cold wall}$$

$$\begin{pmatrix} T_{h.surf} \\ T_{c.surf} \end{pmatrix} := \text{Find}(T_{h.surf}, T_{c.surf})$$

$$T_{h.surf} = 334.238 \text{ K} \quad \text{Temperature of the hot wall surface}$$

$$T_{c.surf} = 305 \text{ K} \quad \text{Temperature of the cold wall surface.}$$

$$\frac{Ra_2(T_{h.surf}, T_{c.surf}) \cdot Pr_{mix} \left[0.5 \cdot (T_{h.surf} + T_{c.surf}) \right]}{0.2 + Pr_{mix} \left[0.5 \cdot (T_{h.surf} + T_{c.surf}) \right]} = 2.021 \times 10^3 \quad \text{Check the Rayleigh Number}$$

$$Nu_{2.sq.box}(T_{h.surf}, T_{c.surf}) = 1.636$$

The FLOW-3D results for this scenario are from **flsgrf.2-D_box_rad_pcg_001g**

$$Nu_{FLOW3D} := 1.67 \quad T_{h.avg.FLOW3D} := 333.8 \text{ K} \quad T_{c.avg.FLOW3D} := 304.5 \text{ K} \quad \text{FLOW-3D values for the } Nu, T_h, \text{ and } T_c$$

$$Var_{Nu} := \frac{Nu_{FLOW3D} - Nu_{2.sq.box}(T_{h.surf}, T_{c.surf})}{Nu_{FLOW3D}} \quad Var_{Nu} = 2.012\%$$

$$Var_{T.h} := \frac{T_{h.avg.FLOW3D} - T_{h.surf}}{T_{h.avg.FLOW3D} - T_{c.avg.FLOW3D}} \quad Var_{T.h} = -1.493\%$$

$$Var_{T.c} := \frac{T_{c.avg.FLOW3D} - T_{c.surf}}{T_{h.avg.FLOW3D} - T_{c.avg.FLOW3D}} \quad Var_{T.c} = -1.706\%$$

$$Q_{h.c.conv} := h_{2.nc}(T_{h.surf}, T_{c.surf}) \cdot A_w \cdot (T_{h.surf} - T_{c.surf})$$

$$Q_{h.c.vap} := mf_{2.vap}(T_{h.surf}, T_{c.surf}) \cdot h_{fg}$$

$$Q_{h.c.rad} := \frac{\sigma \cdot F_{h.c} \cdot A_w}{\left[\frac{1}{\epsilon_h} + F_{h.c} \cdot \frac{(1 - \epsilon_c)}{\epsilon_c} \right]} \cdot (T_{h.surf}^4 - T_{c.surf}^4)$$

$$Q_{h.c.conv} = 1.244 \text{ W}$$

$$Q_{h.c.rad} = 7.767 \text{ W}$$

$$Q_{h.c.vap} = 10.989 \text{ W}$$

$$Q_{h.c.conv} + Q_{h.c.rad} + Q_{h.c.vap} = 20 \text{ W}$$

$$Q_{conv.FLOW3D} := 1.29 \cdot \text{watt} \quad Q_{rad.FLOW3D} := 7.89 \cdot \text{watt}$$

$$Q_{vap.FLOW3D} := 11.0 \cdot \text{watt}$$

$$Q_{conv.FLOW3D} + Q_{rad.FLOW3D} + Q_{vap.FLOW3D} = 20.18 \text{ W}$$

$$mf_{2.vap}(T_{h.surf}, T_{c.surf}) = 4.768 \times 10^{-6} \frac{\text{kg}}{\text{s}}$$

$$mf_{vap.FLOW3D} := 4.76 \cdot 10^{-6} \frac{\text{kg}}{\text{sec}}$$

$$Var_{mf} := \frac{mf_{vap.FLOW3D} - mf_{2.vap}(T_{h.surf}, T_{c.surf})}{mf_{vap.FLOW3D}} \quad Var_{mf} = -0.165 \%$$

TL
4/11/2008

Listing of file prepin.2-D_box_norad_nopcg_001g

```
Heat transfer in 2-D Sq. Box w/ natural convection, NO radiation, NO moisture
Ra ~ 15000 for natural convection only
Box is 0.1 m x 0.1 m long,
Adiabatic at top and bottom,
20 W generated in left wall,
300 K at outer face of right wall.
Both walls have k=1 w/(m*K), rcobs=1000*joule/(m^3*K)
Moisture at left and right walls
```

```
$xput
```

```
  remark='units are SI',
  itb=0,      remark='Sharp interface not required',
  ifvis=0,    remark='Laminar',
  ifenrg=2,   remark='Solve energy eq., first order',
  ifrho=1,    remark='Temp. dependent density',
  gz=-9.8e-03, remark='Gravity',
  ipdis=1,    remark='Hydrostatic pressure distribution in z-direction',
  ihtc=2,     remark='Evaluate heat transfer with solid conduction',
  iwsh=1,     remark='Wall shear active',
  delt=1.e-4, remark='',
  twfin=4000., remark='',
  rmrhoe=0.,  remark='Density diffusion term coefficient',
  rmrho=0.,   remark='Energy diffusion term coefficient',
              remark=' RMRHO, RMRHOE = 1 REQUIRED FOR VAPOR TRANSPORT MODEL ',
  iusrd=1,
```

```
$end
```

```
$limits
```

```
$end
```

```
$props
```

```
  units='si',
  rhof=1.1687, remark='Bulk nominal density at 310 K for energy equation',
  mu1=2.e-05,  remark='Bulk nominal dynamic viscosity density for momentum equation',
  cv1=717.,    remark='Const. Vol. Specific heat of dry air',
  thc1=0.026,  remark='Bulk nominal thermal conductivity for energy equation',
  thexf1=0.003226, remark='Approximate thermal expansion coefficient',
  tstar = 300., remark='Reference temperature for thermal expansion coefficient',
              remark='THEXF1, TSTAR not needed for vapor transport model',
              remark='but are used in some parts of code to set up simulation',
```

```
$end
```

```
$scalar
```

```
  remark='Water vapor scalars not used for this problem but need to be defined to use the',
  ,
  remark=' radiation and water transport modules together',
  remark='Scalar 1 is the diffusing/advection water',
  remark='Scalar 2 (non-diffusing) is for storing the surface phase change flux values',
  remark='Scalar 3 (non-diffusing) is for storing the relative humidity values',
  remark='Scalar 4 (non-diffusing) is for storing the net surface liquid accumulation',
  remark='Scalar 5 (non-diffusing) is for storing the vapor water concentration',
  remark='Scalar 6 (non-diffusing) is for storing the liquid water (mist) concentration',
  remark='Scalar 7 (non-diffusing) is for storing the calculation iterations for the',
  remark='  wall mass flux',
  remark='Scalar 8 (non-diffusing) is for storing the calculation iterations for the',
  remark='  mist phase change in the fluid interior',
  remark='Scalars 7 and 8 are helpful in tuning the value of vaprlx if necessary',
  nsc=8,
  isclr(1)=3, cmisc(1)=0.26e-04, scltit(1)='Tot.Water', rMSC=0.,
```

```
isclr(2)=0, cmisc(2)=0., scltit(2)='Liq.Flux',
isclr(3)=0, cmisc(3)=0., scltit(3)='Rel.Hum',
isclr(4)=0, cmisc(4)=0., scltit(4)='Net.Liq',
isclr(5)=0, cmisc(5)=0., scltit(5)='Vap.Wat',
isclr(6)=0, cmisc(6)=0., scltit(6)='Liq.Wat',
isclr(7)=0, cmisc(7)=0., scltit(7)='Itr.Wall',
isclr(8)=0, cmisc(8)=0., scltit(8)='Itr.Mesh',
$end

$bcdata
wl=1,          remark='Symmetry at left to make all the heat go into the inner face',
wr=2,          remark='Constant temperature wall at right',
            tbc(2)=300.,
wf=1, wbk=1,   remark=' Symmetry at front and back',
wb=2,          remark=' Insulated wall',
wt=2,          remark=' Insulated wall',
$end

$mesh
px(1) =-0.025,      py(1) =0.0,      pz(1) = 0.0,
px(2) = 0.0,        py(2) =1.,      pz(2) = 0.1,
px(3) = 0.1,
px(4) = 0.125,
nxcell(1)=5,
nxcell(2)=20,
nxcell(3)=5,
nxcelt= 30,          nycelt=1,          nzcelt=30,
$end

$obs
avrck=-3.1,
nobs = 2,
tobs(1)=0., tobs(2)=1000.,
remark='Obstacle 1. Left channel wall. Total heat output 50W',
xl(1)=-0.025, xh(1)=0.,
kobs(1)= 1., rcobs(1)=10000.,
twobs(1,1)=300.,
pobs(1,1)=20., pobs(2,1)=20.,
remark='Obstacle 2. Right channel wall. Conduction to constant temp',
xl(2)=0.1, xh(2)=0.125,
kobs(2)= 1., rcobs(2)=10000.,
twobs(1,2)=300.,
$end

$f1
remark=' No vapor in fluid',
sclri(1)=0.0,
sclri(5)=0.0,
sclri(6)=0.0,
presi=101325.,
$end

$bf
$end

$temp
ntmp=1,
tempi=300.,
$end

$motn
$end

$grafic
```


\$end

\$parts
\$end

\$usrdat

```
istwtf_stg='tw',      remark='Energy for phase change comes from wall',

isvap_stg = 1,        remark='Define scalar index for water concentration DO NOT CHANGE',
isliq_stg = 2,        remark='Define scalar index for liquid flux DO NOT CHANGE',
isrh_stg = 3,         remark='Define scalar index for rel. hum. DO NOT CHANGE',
istlg_stg = 4,        remark='Define scalar index for tot.liq. accum. DO NOT CHANGE',
isywv_stg = 5,        remark='Define scalar index for vapor water DO NOT CHANGE',
isywl_stg = 6,        remark='Define scalar index for liquid water (mist) DO NOT CHANGE',
hvvap_stg=2304900.,   remark='Heat of vaporization',
cvvap_stg=1370.,      remark='Water vapor specific heat (const. vol.)',
cvliq_stg=4186.,      remark='Water liquid specific heat (const. vol.)',
rvap_stg=416.,        remark='Gas constant for water vapor',
rgas_stg=289.,        remark='Gas constant for air',
vaprlx_stg=0.8,       remark='Relaxation factor for phase change iterations',
rhlim_stg='y',        remark='Limit rel humidity to 100 percent',
```

\$end

```
nrsrf_stg=2,

eps_stg(1) = .9,
rad_stg(1,1) = 1.,
rad_stg(2,1) = -0.004, rad_stg(3,1) = 0.004,
rad_stg(4,1) = -1., rad_stg(5,1) = 2.,
rad_stg(6,1) = 0.0, rad_stg(7,1) = 0.2,

eps_stg(2) = .9,
rad_stg(1,2) = 2.,
rad_stg(2,2) = 0.096, rad_stg(3,2) = 0.104,
rad_stg(4,2) = -1., rad_stg(5,2) = 2.,
rad_stg(6,2) = 0.0, rad_stg(7,2) = 0.2,

iusrcf_stg=0,

cf_stg(1,1)=0., cf_stg(1,2) = 1.,
cf_stg(2,1)=1., cf_stg(2,2) = 0.,

imoist_stg(1)=-1,
imoist_stg(2)=-2,
```

End of listing of file prepin.2-D_box_norad_nopcg_001g

The analytical results in the Mathcad file are summarized in Table 3/30/06-1. This shows the predictions for the average surface temperatures, the breakdown of the heat transfer rates for each of the heat transfer modes and the mass transfer rates.

Table 3/30/06-1. Analysis Results for 2-D Enclosure Heat and Mass Transfer

Scenario	$T_{h, average}$ K	$T_{c, average}$ K	Heat Transfer Rate			Total W/m	Mass Transfer kg/sec/m
			Convection W/m	Radiation W/m	Phase Change W/m		
Convection Only	650	305.0	20	n/a	n/a	20	n/a
Convection + Radiation	362.1	305.0	2.7	17.3	n/a	20	n/a
Convection + Moisture Transport	342.5	305.0	1.7	n/a	18.3	20	7.9×10^{-6}
Convection + Radiation + Moisture Transport	334.2	305.0	1.2	7.8	11.0	20	4.8×10^{-6}

The FLOW-3D predictions for the four scenarios are summarized in Table 3/30/06-2. The values in parentheses are the relative variance from the corresponding value from the analytical results. The basis for the heat transfer variance is the total heat transfer rate of 20 W. The basis for the mass transfer variance is the value from the empirical analysis. The basis for the temperature variance is the overall temperature difference between the hot and cold surfaces.

Note that the radiation and the moisture transport have a significant effect on the temperature of the heated surface. Each of these energy transfer modes accounts for several times more energy flow than the convection heat transfer.

The FLOW-3D results are in good agreement with the analytical results for this case.

Table 3/30/-6-2. FLOW-3D Results for 2-D Enclosure Heat and Mass Transfer

Scenario	T _{h, average} * K	T _{c, average} * K	Heat Transfer Rate			Total W/m	Mass Transfer kg/sec/m
			Convection W/m	Radiation W/m	Phase Change W/m		
Convection Only	646.2 (-1.1%)	304.5 (0.1%)	20	n/a	n/a	20	n/a
Convection + Radiation	362.7 (1.1%)	304.5 (-1.0%)	2.3 (-2.0%)	17.7 (2.0%)	n/a	20	n/a
Convection + Moisture Transport	340.9 (-4.4%)	304.5 (-1.4%)	1.8 (0.2%)	n/a	18.2 (-0.4%)	20	7.9x10 ⁻⁶ (0%)
Convection + Radiation + Moisture Transport	333.8 (-1.5%)	304.5 (-1.7%)	1.3 (0.3%)	7.9 (0.6%)	11.0 (0%)	20.2 (1.0%)	4.8x10 ⁻⁶ (0%)

*These values are the average of the all the surface cells where the temperature values correspond to the FLOW-3D cell center located half the cell width from the obstacle surface.

END OF ENTRY FOR 3/30/06 *STG*

=====
 Entries made into Scientific Notebook #536E for the period December 1, 2005 to March 31, 2006, have been made by Steven Green (April 1, 2006).

No original text or figures entered into this Scientific Notebook has been removed

STG 4/1/2006

5/10/06

STG

The mountain-scale heat transfer analysis shows that the temperature of the rock surrounding the drifts will have both a temporal and spatial variation. The spatial variation at selected times after closure was provided by R. Fedors as explained in the entry of 3/12/04 in this notebook. As described in that entry, the rock temperature as a function of the distance along the drift axis is assumed to be uniform around the circumference of the drift at each axial location. The numerical values at selected times after closure (predicted by the TPA code) provided by Fedors were fit with an equation for use in the FLOW-3D code. For the simulations reported in the 3/12/04 entry and those immediately following, the equations were coded directly into FLOW-3D. It was required that the custom user defined functions in FLOW-3D had to be compiled specifically for each particular function for the rock temperatures.

Since that time, this customized feature of FLOW-3D has been made more user-friendly by allowing the use to prescribe the curve fit equation coefficients in the FLOW-3D input file. This feature has not been sufficiently documented, although, the document "Description of the Moisture Transport Module and Radiation Module for FLOW-3D® Simulations of Repository Drifts," has some information about its use.

The intent of this notebook entry is to document the curve fit equation for rock wall temperatures and its use in FLOW-3D simulations.

For simulations of the full-scale drifts in Yucca mountain, it has been the practice, so far, to include the conduction heat transfer processes in a 1-m thick layer of rock surrounding the drift. The temperature of the rock at the outer surface of this 1-m thick cylinder thereby becomes a required boundary condition for the simulations. All of the rock wall temperature provide to date have been found to be fit reasonably well with the following functional form:

$$T_{1-m}(x) = \frac{a + cx + ex^2 + gx^3}{1 + bx + dx^2 + fx^3} + T_{shift}$$

where x = distance from the reference point for the simulations (m)

T_{shift} = a temperature shift to convert °C to K

a,b,c,d,e,f = curve fit coefficients

Because of the details of the moisture transport module and the radiation module, it is required that the units of temperature be Kelvin. The TPA code uses °C as its temperature unit. The function coefficients can be used to fit the equation to the table of values in units of °C. In that case, the value of $T_{shift}=273.15$. Otherwise, if the curve fit function is already provided in Kelvin, $T_{shift}=0$.

The rock wall temperature capability uses the following arrays in the customized version of FLOW-3D. These are the values specified by the user in the USRDAT section of the FLOW-3D input files.

Parameter	Units	Definition
idrftw_stg(n)	n/a	Set this value equal to the NEGATIVE of the obstacle for which the function is to be applied. That is: $idrftw_stg(n)=-n$
rocktemp_a(n)	K or °C	Coefficient a in the curve fit function
rocktemp_b(n)	m^{-1}	Coefficient b in the curve fit function
rocktemp_c(n)	K/m or °C/m	Coefficient c in the curve fit function
rocktemp_d(n)	m^{-2}	Coefficient d in the curve fit function
rocktemp_e(n)	K/m^2 or $°C/m^2$	Coefficient e in the curve fit function
rocktemp_f(n)	m^{-3}	Coefficient f in the curve fit function
rocktemp_g(n)	K/m^3 or $°C/m^3$	Coefficient g in the curve fit function
rocktemp_shift(n)	K	Coefficient T_{shift} in the curve fit function

This feature of the customized FLOW-3D code is activated by specifying a nonzero value of the $idrftw(n)$ and the associated coefficient values to be applied to the obstacle 'n'. This assumes that obstacle 'n' is properly defined in the OBS section of the input file.

There is no error checking for bad or inconsistent input parameters. It is left to the user to check the input values

The obstacle for which this temperature specification is being made is not meant to represent a real object in the physical space surrounding the 1-m thick layer of rock around the drift. This obstacle serves only to specify the temperature boundary condition for the 1-m thick rock cylinder. As such, there are specific rules regarding how the properties of this fictitious boundary obstacle are specified:

1. Initial temperature (uniform is OK) must be specified in the input file.
2. For the uniform temperature obstacle as specified in accordance with the standard FLOW-3D installation, the solid properties must NOT BE SPECIFIED.

3. For the obstacles specified with the modified variable obstacle temperature method, both kobs and rcobs MUST BE SPECIFIED for the heat transfer to be computed properly

An unfortunate feature of the FLOW-3D approach appears to be that the specified solid boundary temperature appears to be applied not at the surface of the active obstacle but at the first node into the fixed-temperature obstacle. This feature must be taken into account when specifying the phantom obstacle and its properties. That is:

1. Set the phantom obstacle properties (kobs and rcobs) to be equal to the active obstacle.
2. Place the CENTER of the first node away from the physical interface at the location for which the desired temperature is to be specified.

END OF ENTRY FOR 5/10/06

STG

6/5/06

STG

The radiation module that was written for simulating radiation heat transfer within drifts was written under the implicit assumption that the geometries to be simulated were full 2-D or 3-D representations. Symmetry planes were not allowed. The inclusion of plane of symmetry is very convenient for many problems; so, there needs to be a way to allow this.

This entry describes a method of specifying the radiation parameters for a problem with a plane of symmetry such that the radiation heat transfer between obstacles across the plane of symmetry are included in the simulation.

Consider the equation on which the radiation module is based (see entry for 3/3/06):

$$\sum_{m=1}^{N_r} \left(\frac{\delta_{nm}}{\varepsilon_m} - F_{n-m} \frac{1 - \varepsilon_m}{\varepsilon_m} \right) \frac{Q_m}{A_m} = \sum_{m=1}^{N_r} \sigma F_{n-m} (T_n^4 - T_m^4) \quad (6/6/06-1)$$

where $\sigma = 5.67 \times 10^{-8}$ watt/(m²K⁴) = Stefan-Boltzmann constant

δ_{nm} = Kronecker delta

ε_m = surface emissivity

F_{n-m} = configuration factor for surface 'n' viewing surface 'm'

Q_m = heat transfer rate from surface via radiation (W)

A_m = surface area (m²)

T_m = temperature of surface (K)
 N_r = number of surfaces active for radiation

Equation 6/6/01-1 is repeated for all N_r surfaces so that there is a set of N_r equations with N_r unknowns. As described in the 3/3/06 entry, the current values of the wall surface temperatures, T_n , are used to compute the heat fluxes Q_n/A_n . The heat fluxes are used to compute new values of the wall surface temperatures for the next FLOW-3D as a means of simulating the heat flux due to radiation.

In cases where it is desired to simulate only half of a flow and heat transfer on one side of a plane of symmetry, Equation 6/6/06-1 cannot be directly applied. Let N_{sym} be the number of surfaces in the symmetric problem and arrange the surface such that surface $n+N_{sym}$ is the mirror image of surface n across the plane of symmetry. Equation 6/6/06-1 can now be written as

$$\sum_{m=1}^{N_{sym}} \left(\frac{\delta_{nm}}{\epsilon_m} - F_{n-m} \frac{1-\epsilon_m}{\epsilon_m} \right) \frac{Q_m}{A_m} + \sum_{m=N_{sym}+1}^{N_r} \left(\frac{\delta_{nm}}{\epsilon_m} - F_{n-m} \frac{1-\epsilon_m}{\epsilon_m} \right) \frac{Q_m}{A_m} \tag{6/6/06-2}$$

$$= \sum_{m=1}^{N_{sym}} \sigma F_{n-m} (T_n^4 - T_m^4) + \sum_{m=N_{sym}+1}^{N_r} \sigma F_{n-m} (T_n^4 - T_m^4)$$

The following conditions result from the assumption of a plane of symmetry:

$$Q_n = Q_{N_{sym}+n}$$

$$A_n = A_{N_{sym}+n}$$

$$T_n = T_{N_{sym}+n}$$

$$\epsilon_n = \epsilon_{N_{sym}+n}$$

With these symmetry conditions, we can combine terms in Equation (6/6/06-2) as follows:

$$\sum_{m=1}^{N_{sym}} \left(\frac{\delta_{nm}}{\epsilon_m} - (F_{n-m} + F_{n-(N_{sym}+m)}) \frac{1-\epsilon_m}{\epsilon_m} \right) \frac{Q_m}{A_m} = \sum_{m=1}^{N_{sym}} \sigma (F_{n-m} + F_{n-(N_{sym}+m)}) (T_n^4 - T_m^4) \tag{6/6/06-3}$$

Now we have a set of equations for computing the radiation heat transfer from the set of N_{sym} surfaces all on the side of the symmetry plane that is being simulated. The effect of the

radiation exchange across the plane of symmetry are accounted for by the combination of the radiation configuration factors as shown. The problem, however, is that none of the surfaces N_{sym} to N_r are included in the geometric model for the problem so there is no way for the radiation module software to compute the corresponding configuration factors from the symmetry problem specifications.

The following steps are required to accommodate this in a simulation:

1. Compute the radiation configuration factors for the complete geometry:
 - a. Specify the full problem for both sides of the symmetry plane. Specify the radiation surfaces as if the complete problem is to be executed.
 - b. It is highly recommended that the mirror image pairs of surfaces be numbered such that their indices differ by the value of N_{sym} ($N_{sym} = N_r/2$).
 - c. Be sure to set `iusrcf_stg=1` so that the radiation module computes the configuration factors for all pairs of surfaces.
2. Execute the simulation for 2 time steps so that the radiation factors are computed.
 - a. The radiation factors are written to a file for review at the first time step in the simulation.
 - b. Use a simple text editor to arrange the configuration factors for spreadsheet calculations. Excel can also be used with its text manipulation functions.
3. Obtain the symmetry problem configuration factors.
 - a. Add the configuration factors between each surface and the pair of mirror image surfaces together as indicated in Equation 6/6/06-3:

$$(F_{n-m})_{sym} = F_{n-m} + F_{n-(N_{sym}+m)}$$
 where $(F_{n-m})_{sym}$ is the effective configuration factor for the radiation face in the symmetry problem. It required to do this for only the surfaces $1 \leq n \leq N_{sym}$.
 - b. Check to insure that

$$\sum_{m=1}^{N_{sym}} (F_{n-m})_{sym} \leq 1$$
 for all the $n \leq N_{sym}$ surfaces. This is a requirement for
 - c. Create a list of the configuration factor variable specifications [`cf_stg(n,m)`] suitable for the FLOW-3D input file.
4. Execute FLOW-3D with adjusted configuration factors input by the user for the symmetry geometry.
 - a. Create the input file for the symmetry problem. This should be a simple edit from the full non-symmetry problem.
 - b. Be sure to create only the N_{sym} radiation surfaces for the symmetry problem
 - c. Import the list of configuration factors for the symmetry problem.
 - d. Be sure to set `iusrcf_stg=0` so that the radiation module will use the user-specified values of the configuration factors.

These steps are demonstrated in the process of setting up the simulations for the full-scale Yucca Mountain drifts to be recorded in a later entry.

END OF ENTRY FOR 6/5/06

STG

8/6/06

STG

Full Scale Yucca Mountain Drift Cold Trap Simulations

References

Fedors, R., Green, S., Walter D., Adams. G., Farrell, D., and Svedeman, S., "Temperature and Relative Humidity Along Heat Drifts With and Without Drift Degradation," CNWRA 2004-04, Center for Nuclear Waste Regulatory Analysis, NRC Contract NRC-02-02-012, June 2004

Incropera, F. P., and Dewitt, D. P., *Fundamentals of Heat and Mass Transfer*, John Wiley and Sons, New York, 1985.

Background

Simulations of the cold trap effect for full scale YM drifts were described by Fedors, et al. (2004). The underlying technical work for those simulations is described in the entry for 3/3/04 in this notebook. At that time, the flow and heat transfer simulations neglected the effects of radiation in the drift. The temperature of the rock surrounding the drift was specified in accordance with a time of 109 yr after closure. Under those conditions it was found that the drift internal temperatures were predicted to be greater than the 1-atm boiling point of water of 100°C. It was decided to perform the simulations for the two limiting cases for the basic gas composition inside the drift. The first scenario was for the case of dry air and the second was for only water vapor. Any mixtures of air and water vapor are possible when the temperature is greater than the boiling point of water (at the local pressure), but these two cases only were considered for the 2004 simulations.

There has been a significant amount of effort expended to incorporate the effects of moisture transport and radiation heat transfer into the simulation of flow processes for the drift. This effort has been aimed at including these effects in the FLOW-3D software, but it could be translated for use in other CFD codes as well. This development effort is documented

extensively in this notebook and the validation and application of this work to in-drift processes are described in this notebook and in Notebook # _____ by Flavia Viana.

Objective

This entry documents the work performed specifically for the simulation of the cold trap process in a full scale YM drift including the effects of radiation heat transfer. It is expected that this work will be formally reported to the NRC in a status report to be delivered in September 2006. The objectives of these analyses are to directly compare the latest simulations to those described in the 2004 status report of Fedors, et al., (2004).

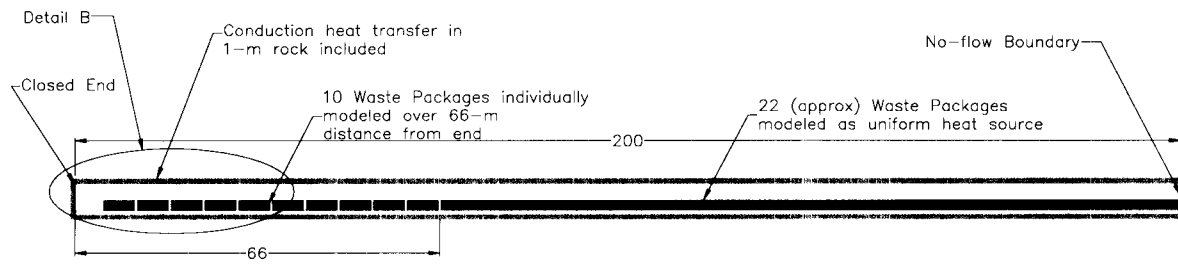
Simulation Geometry

The geometric configuration for the flow domain is described in Figure 8/6/06-1. The flow domain extends 200 m away from the closed end of the drift and 1 m into the rock from the end of the drift. A 1 m thickness of rock is included in the domain. The conduction heat transfer within this 1-m layer is to be simulated. The solid material in the flow domain includes a representation of the invert. The inside diameter of the cylindrical portion of the drift wall is 2.75 m and the top surface of the invert is located 2.03 m below the centerline.

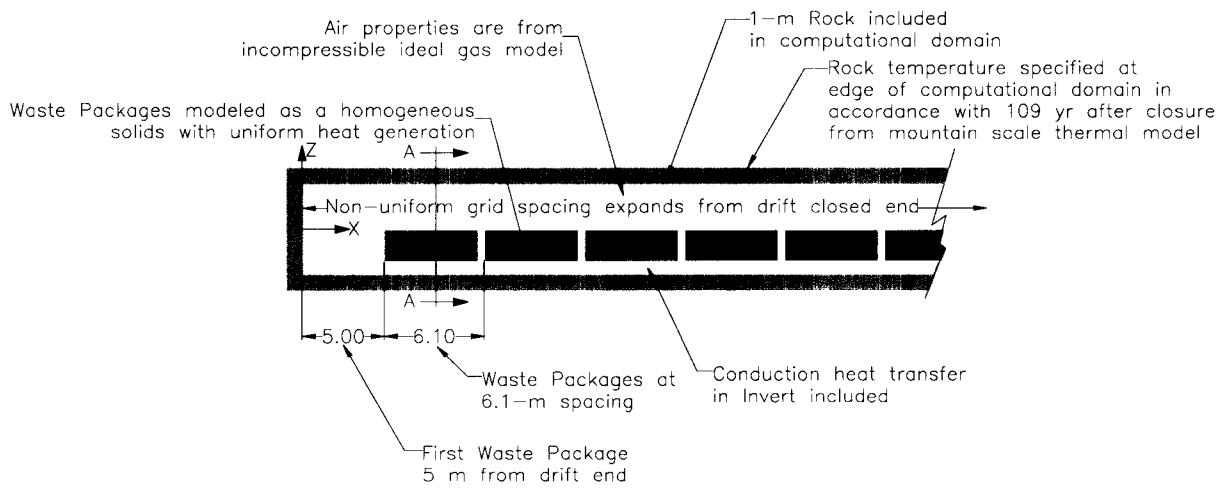
The waste packages are simulated as cylinders with a diameter of 1.8 m and a length of 5.6 m. The end of the first waste package is 5 m from the end of the drift. A total of 10 waste packages are individually modeled on 6.1-m intervals to cover the first 66 m of length from the end of the drift. The remaining 134 m of drift length contains a single unbroken cylinder to represent the remainder of the waste packages. This lumping of the waste packages is consistent with the axial expansion of the computational mesh in this region. The centerline of the waste packages is 1 m below the centerline of the drift.

TL

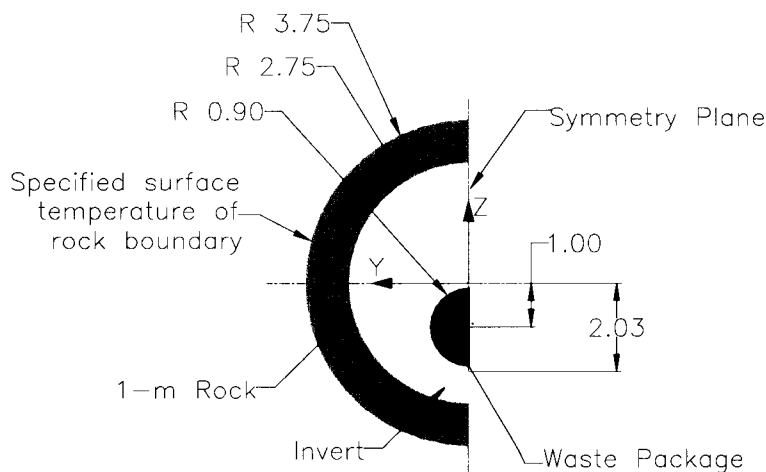
4/11/2008



(a)



(b)



Section A-A

(c)

Figure 8/6/06-1. Geometry for Full Scale Simulations.

The complete axial-vertical plane of the flow domain is seen in (a). The details of the closed end of the drift are seen in (b). The drift cross section is in (c). All dimensions are meters.

Material Properties

There are three sets of material properties that are required for the simulation. The property values are summarized in Table 8/6/06-1.

The thermal conductivities of the rock and waste package are the same as those used by Fedors, et al. The heat capacity of the rock and waste package used in the simulations is not the actual value for the respective materials. Instead a value is selected here to allow the thermal energy of these objects to reach a steady state condition as quickly as possible without affecting the time step of the solution algorithm. The surface emissivities of the rock and waste package are consistent with oxidized stainless steel and rocks (Incropera and Dewitt, 1985)

Table 8/6/06-1. Material Properties for YM Simulations

Property	Units	Value
Rock		
Thermal conductivity	W/(m*K)	1.61
Heat capacity (product of density and specific heat)	J/(m ³ *K)	300 [†]
Surface emissivity		0.9
Waste Package		
Thermal conductivity	W/(m*K)	10.0
Heat capacity (product of density and specific heat)	J/(m ³ *K)	3000 [†]
Surface emissivity		0.9
Air		
Gas constant	J/(kg*K)	289.0
Specific heat	J/(kg*K)	728.7
Thermal conductivity	W/(m*K)	0.03484
Dynamic Viscosity	Pa*sec	2.35×10 ⁻⁵
Nominal Density	kg/m ³	0.84368 [‡]

[†]Not actual value. This value is specified to allow solid materials to reach steady state as quickly as possible without affecting the time step required for fluid motion simulations

[‡]Nominal air density used for only the equation for conservation of thermal energy in accordance with Boussinesq assumption. Ideal incompressible gas model used for conservation of fluid momentum and conservation of mass

The air is modeled as an incompressible gas in accordance with the moisture transport module. It has been assumed for these simulations that there is no water vapor in the drift; however, the moisture transport module is invoked so that future simulations with water vapor can be readily compared to these. The thermal properties are consistent with a temperature of 417 K, the mean temperature of the rock temperature boundary condition. The constant volume specific heat, thermal conductivity, and dynamic viscosity are assumed here to be uniform and constant. The nominal density of the air is used only in the thermal energy equation. The incompressible ideal gas equation is used to compute the density for the momentum equation in keeping with the Boussinesq assumption for buoyancy-driven flows. The use of the incompressible equation of state is more robust than using the simple thermal expansion model for the variable density of a gas.

Computational Mesh

The grid used for these simulations is Cartesian orthogonal, but nonuniform. The coordinate system is defined as shown in Figure 8/6/06-1. The origin is at the center of the closed end of the drift with the x-axis extending along the centerline of the drift away from the closed end, the y-axis is horizontal from the plane of symmetry into the computational mesh. The z-axis is vertically upward.

The region $-1.25 < x < -1.00$ is covered by a single layer of cells that contain an obstacle for specifying the temperature at the surface of the 1-m thick layer of rock surrounding the drift. The region $-1.00 < x < 0.00$ is covered by 3 grid cells in the rock layer. The region $0.00 < x < 40.0$ is the portion of the drift nearest the closed end and contains the air and the first few waste packages. This region is covered by 140 uniformly spaced grid cells. The region $40.0 < x < 80.0$ has 70 grid cells that expand from a spacing of about 0.286 m at $x=40$ to a spacing of about 0.5714 at $x=80.0$. The remainder of the drift in the region $80.0 < x < 200$ is covered by a total of 30 cells with an expanding spacing from $x=80$ to $x=200$.

The grid in the y-direction has 30 cells with a uniform spacing of 0.098 m from the center of the drift to the farthest extent of the drift wall at $y=2.75$ m. The 1-m rock layer is defined by 3 grid cells. The last layer of grids on the y-axis is for the obstacle that specifies the rock layer temperature condition.

The grid in the z-direction is similar to that in the y-direction. The region $-4.25 < z < -3.75$ contains the rock layer boundary obstacle with a single grid layer. The region $-3.75 < z < -2.75$ uses 3 grid cells for the rock layer around the drift. Five grid cells define the thickest part of the invert in the region $-2.75 < z < -2.03$. The region $-2.03 < z < 2.75$ is the drift itself and is covered by 48 cells in the z-direction. The remainder of the z-axis contains 3 cells for $2.75 < z < 3.75$ for the rock layer and 1 grid cell for $3.75 < z < 4.25$ for the boundary obstacle.

Boundary Conditions

The air in the drift is completely enclosed by the drift wall except at the $x=200$ axial location. At the $x=200$ mesh boundary face (right face in FLOW-3D nomenclature) a symmetry condition is specified. Flow and heat do not cross this boundary.

The other five boundary faces are specified as symmetry faces as well; however, these faces do not interact with the flow field. Instead, a FLOW-3D obstacle is used to specify the temperature of the at a location of 1 m from the drift inner surface. The rock temperatures at 109 yr after closure are used here. The rock temperatures are listed in the Excel file

C:\Projects\div20\Full-Scale_May-2006\Rock_temps\Rock_Temps.xls

The curve fit used to express the temperatures at 1 m from the drift wall is found from the software package TableCurve (Version 5),

$$T = \frac{a + cx + ex^2 + gx^3}{1 + bx + dx^2 + fx^3} + T_{shift}$$

where a = 348.927587 K
 b = 0.07825616 m⁻¹
 c = 31.5828997 K·m⁻¹
 d = -4.36E-04 m⁻²
 e = -0.17356232 K·m⁻²
 f = 1.0512E-05 m⁻³
 g = 0.00420548 K·m⁻³
 T_{shift} = 0 K

The temperature is expressed in Kelvin using this curve fit expression. This temperature function for the boundary obstacle is communicated to FLOW-3D through the custom modification for specifying spatially varying obstacle temperatures.

Input File Listings

The FLOW-3D input file for the case in which radiation is neglected is

prepin.EXP_YM_109yr_norad

The contents of this file are listed below. This FLOW-3D file was executed for about 2700 sec of simulated time. The flow field and temperatures reached a reasonable steady condition in about 2500 sec.

Listing of the FLOW-3D input file prepin.EXP_YM_109yr_norad

YM Cold Trap - Full Scale 3D, No Radiation, No Moisture, Temp. Gradient Boundary, LES

8/6/06 prepin.YM_109yr_norad

Based on prepinr.Full-Scale-3D-1-r3c-air

- This is a fresh start at the YM scale simulation without radiation. The major change here is that the incompressible ideal gas model is used here to get the dry air properties. This is accomplished with the moisture module by setting the water fraction to zero at all times.
- Also, used correct values for kobs and rcobs for the phantom obstacle. These must be equal to the rock values for the spatial variation of the rock B.C. temperature to be correctly used.

\$xput

```

remark='units are mks',
itb=0,    remark='no free surface',
ifvis=-1, remark='LES',
ifenrg=2, remark='solve energy transport equation using first order',
ifrho=1,  remark='density as function of fluid fraction and local temperature',
gz=-9.8,  remark='gravity',
ipdis=1,  remark='hydrostatic pressure in z direction',
ihtc=2,   remark='evaluate heat transfer and solve the obstacle conduction
equation',
iwsh=1,
rmrhoe=1., remark='Density diffusion term coefficient',
rmrho=1.,  remark='Energy diffusion term coefficient',
          remark=' RMRHO, RMRHOE REQUIRED FOR VAPOR TRANSPORT MODEL ',
iadiy=1, iadiz=1,
iusrd=1,
twfin=1000., remark='finish time (sec)',
iadix=1,
delt=1.e-3,
sprtdt=1.e-1,
hpltdt=1.,
pltdt=100.,
imphtc=1,
$end

```

\$limits

\$end

\$props

```

remark=' equation of state parameters for air @ 417K (average rock wall temperature
from grid.xls',
remark=' Handbook of tables for Applied Engineering Science, 2nd edition, Table 1-2
(See AirProperties.pdf)',
cvl=728.7,    remark=' specific heat ',
rhof=.84638,  remark=' density',
thcl=0.03484, remark=' conductivity',
thexf1=2.432e-03, remark=' thermal expansion coefficient ',
tstar=417.,   remark=' reference temperature for thermal',
mul=2.3543e-05, remark=' viscosity',
$end

```

```

$scalar
  remark='Water vapor scalars not used for this problem but need to be defined to use
the ',
  remark=' radiation and water transport modules together',
  remark='Scalar 1 is the diffusing/advecting water',
  remark='Scalar 2 (non-diffusing) is for storing the surface phase change flux
values',
  remark='Scalar 3 (non-diffusing) is for storing the relative humidity values',
  remark='Scalar 4 (non-diffusing) is for storing the net surface liquid
accumulation',
  remark='Scalar 5 (non-diffusing) is for storing the vapor water concentration',
  remark='Scalar 6 (non-diffusing) is for storing the liquid water (mist)
concentration',
  remark='Scalar 7 (non-diffusing) is for storing the calculation iterations for the',
  remark='  wall mass flux',
  remark='Scalar 8 (non-diffusing) is for storing the calculation iterations for the',
  remark='  mist pahse change in the fluid interior',
  remark='Scalars 7 and 8 are helpful in tuning the value of vaprlx if necessary',
  nsc=8,
  isclr(1)=3, cmisc(1)=0.26e-04, scltit(1)='Tot.Water', rmisc=0.,
  isclr(2)=0, cmisc(2)=0., scltit(2)='Liq.Flux',
  isclr(3)=0, cmisc(3)=0., scltit(3)='Rel.Hum',
  isclr(4)=0, cmisc(4)=0., scltit(4)='Net.Liq',
  isclr(5)=0, cmisc(5)=0., scltit(5)='Vap.Wat',
  isclr(6)=0, cmisc(6)=0., scltit(6)='Liq.Wat',
  isclr(7)=0, cmisc(7)=0., scltit(7)='Itr.Wall',
  isclr(8)=0, cmisc(8)=0., scltit(8)='Itr.Mesh',
$end

```

```

$bcddata
  wl=1, Remark='Left boundary - wall, constant temp',
  wr=1, Remark='Right boundary - wall, constant temp (cold wall)',
  wf=1, Remark='Front boundary - Symmetry',
  wbk=1, Remark='Back boundary - Symmetry',
  wb=1, Remark='Bottom boundary - Symmetry',
  wt=1, Remark='Top boundary - Symmetry',
$end

```

```

$mesh
  nxcelt=244,
  px(1)=-1.25, nxcell(1)=1,
  px(2)=-1.0, nxcell(2)=3,
  px(3)=0.0, nxcell(3)=140,
  px(4)=40., nxcell(4)=70,
  px(5)=80., nxcell(5)=10,
  px(6)=120., nxcell(6)=10,
  px(7)=160., nxcell(7)=10,
  px(8)=199.,
  nycelt=32,
  py(1)=0., nycell(1)=28,
  py(2)=2.75, nycell(2)=3,
  py(3)=3.75, nycell(3)=1,
  py(4)=4.25,
  nzcelt=58,
  pz(1)=-4.25, nzcell(1)=1,
  pz(2)=-3.75, nzcell(2)=5,
  pz(3)=-2.03, nzcell(3)=48,
  pz(4)=2.75, nzcell(4)=3,
  pz(5)=3.75, nzcell(5)=1,

```



```
pz(6)=4.25,  
$end
```

```
$obs
```

```
avrck=-3.1,  
nobs=14,  
tobs(1)=0., tobs(2)=100000.,  
Remark='Obstacle 1 - Drift Wall (1m rock)',  
  iob(1)=1, ioh(1)=1,  
    rah(1)=3.75, z1(1)=-1., roty(1)=90.,  
  iob(2)=1, ioh(2)=0,  
    rah(2)=2.75, z1(2)=0., roty(2)=90.,  
  kobs(1)=1.61, Remark='From Fedor e-mail 12-9-03',  
  rcobs(1)=3.,  
  twobs(1,1)=395.,  
Remark='Obstacle 2 - Phantom Boundary',  
  iob(3)=2, ioh(3)=1,  
  iob(4)=2, ioh(4)=0,  
    rah(4)=3.75, z1(4)=-1., roty(4)=90.,  
  twobs(1,2)=395.,  
  kobs(2)=1.61, rcobs(2)=3., remark=' These must be equal to the neighbor rock',  
Remark='Obstacle 3 - Invert',  
  iob(5)=3, ioh(5)=1,  
    rah(5)=2.75, z1(5)=0., roty(5)=90.,  
  iob(6)=3, ioh(6)=0,  
    z1(6)=-2.03, Remark='From SK-0154 Rev02 1-28-00',  
  kobs(3)=1.61, Remark='From Fedor e-mail 12-9-03',  
  rcobs(3)=3.,  
  twobs(1,3)=395.,  
Remark='Obstacle 4 - Heater 1',  
  iob(7)=4, ioh(7)=1,  
    rah(7)=.9, z1(7)=-2.8, zh(7)=2.8, roty(7)=90.,  
    trnx(7)=7.8, trnz(7)=-1., Remark='From SK-0154 Rev02 1-28-00',  
  twobs(1,4)=395., kobs(4)=10., rcobs(4)=30.,  
  pobs(1,4)=1075., pobs(2,4)=1075.,  
Remark='Obstacle 5 - Heater 2',  
  iob(8)=5, ioh(8)=1,  
    rah(8)=.9, z1(8)=-2.8, zh(8)=2.8, roty(8)=90.,  
    trnx(8)=13.9, trnz(8)=-1., Remark='From SK-0154 Rev02 1-28-00',  
  twobs(1,5)=395., kobs(5)=10., rcobs(5)=30.,  
  pobs(1,5)=1075., pobs(2,5)=1075.,  
Remark='Obstacle 6 - Heater 3',  
  iob(9)=6, ioh(9)=1,  
    rah(9)=.9, z1(9)=-2.8, zh(9)=2.8, roty(9)=90.,  
    trnx(9)=20., trnz(9)=-1., Remark='From SK-0154 Rev02 1-28-00',  
  twobs(1,6)=395., kobs(6)=10., rcobs(6)=30.,  
  pobs(1,6)=1075., pobs(2,6)=1075.,  
Remark='Obstacle 7 - Heater 4',  
  iob(10)=7, ioh(10)=1,  
    rah(10)=.9, z1(10)=-2.8, zh(10)=2.8, roty(10)=90.,  
    trnx(10)=26.1, trnz(10)=-1., Remark='From SK-0154 Rev02 1-28-00',  
  twobs(1,7)=395., kobs(7)=10., rcobs(7)=30.,  
  pobs(1,7)=1075., pobs(2,7)=1075.,  
Remark='Obstacle 8 - Heater 5',  
  iob(11)=8, ioh(11)=1,  
    rah(11)=.9, z1(11)=-2.8, zh(11)=2.8, roty(11)=90.,  
    trnx(11)=32.2, trnz(11)=-1., Remark='From SK-0154 Rev02 1-28-00',  
  twobs(1,8)=395., kobs(8)=10., rcobs(8)=30.,  
  pobs(1,8)=1075., pobs(2,8)=1075.,  
Remark='Obstacle 9 - Heater 6',  
  iob(12)=9, ioh(12)=1,  
    rah(12)=.9, z1(12)=-2.8, zh(12)=2.8, roty(12)=90.,  
    trnx(12)=38.3, trnz(12)=-1., Remark='From SK-0154 Rev02 1-28-00',
```

```
twoobs(1,9)=395., kobs(9)=10., rcobs(9)=30.,
pobs(1,9)=1075., pobs(2,9)=1075.,
Remark='Obstacle 10 - Heater 7',
iob(13)=10, ioh(13)=1,
rah(13)=.9, zl(13)=-2.8, zh(13)=2.8, roty(13)=90.,
trnx(13)=44.4, trnz(13)=-1., Remark='From SK-0154 Rev02 1-28-00',
twoobs(1,10)=395., kobs(10)=10., rcobs(10)=30.,
pobs(1,10)=1075., pobs(2,10)=1075.,
Remark='Obstacle 11 - Heater 8',
iob(14)=11, ioh(14)=1,
rah(14)=.9, zl(14)=-2.8, zh(14)=2.8, roty(14)=90.,
trnx(14)=50.5, trnz(14)=-1., Remark='From SK-0154 Rev02 1-28-00',
twoobs(1,11)=395., kobs(11)=10., rcobs(11)=30.,
pobs(1,11)=1075., pobs(2,11)=1075.,
Remark='Obstacle 12 - Heater 9',
iob(15)=12, ioh(15)=1,
rah(15)=.9, zl(15)=-2.8, zh(15)=2.8, roty(15)=90.,
trnx(15)=56.6, trnz(15)=-1., Remark='From SK-0154 Rev02 1-28-00',
twoobs(1,12)=395., kobs(12)=10., rcobs(12)=30.,
pobs(1,12)=1075., pobs(2,12)=1075.,
Remark='Obstacle 13 - Heater 10',
iob(16)=13, ioh(16)=1,
rah(16)=.9, zl(16)=-2.8, zh(16)=2.8, roty(16)=90.,
trnx(16)=62.7, trnz(16)=-1., Remark='From SK-0154 Rev02 1-28-00',
twoobs(1,13)=395., kobs(13)=10., rcobs(13)=30.,
pobs(1,13)=1075., pobs(2,13)=1075.,
Remark='Obstacle 14 - Heater 11 (Long Heater - 22.5 Heaters)',
iob(17)=14, ioh(17)=1,
rah(17)=.9, zl(17)=-2.8, roty(17)=90.,
trnx(17)=68.8, trnz(17)=-1., Remark='From SK-0154 Rev02 1-28-00',
twoobs(1,14)=395., kobs(14)=10., rcobs(14)=30.,
pobs(1,14)=24200., pobs(2,14)=24200.,
$end

$fl
sclri(1)=0.00,
sclri(5)=0.00,
sclri(6)=0.,
presi=101325.,
$end

$bf
$end

$temp
tempi=395.,
$end

$motn
$end

$grafic
$end
jobsf=1,
jobsbk=2,
$parts
$end
$surdat

remark=' Obstacle Temperature Variation',
```

```

remark='Temperature (deg.K) profile for 109 yrs',
idrftw_stg(2)=-2,
rocktemp_a(2)=348.9275866,
rocktemp_b(2)=0.078256163,
rocktemp_c(2)=31.58289974,
rocktemp_d(2)=-4.36E-04,
rocktemp_e(2)=-0.17356232,
rocktemp_f(2)=1.05121E-05,
rocktemp_g(2)=0.004205476,
rocktemp_shift(2)=0.,

```

```

remark=' Moisture Module Inputs&Controls',
istwtf_stg='tw',      remark='Energy for phase change comes from wall',
isvap_stg = 1,        remark='Define scalar index for water concentration DO NOT
CHANGE',
isliq_stg = 2,        remark='Define scalar index for liquid flux          DO NOT
CHANGE',
isrh_stg = 3,         remark='Define scalar index for rel. hum.              DO NOT
CHANGE',
istliq_stg = 4,       remark='Define scalar index for tot.liq. accum.          DO NOT
CHANGE',
isywv_stg = 5,        remark='Define scalar index for vapor water              DO NOT
CHANGE',
isywl_stg = 6,        remark='Define scalar index for liquid water (mist) DO NOT
CHANGE',
hvvap_stg=2300000.,  remark='Heat of vaporization',
cvvap_stg=1411.,     remark='Water vapor specific heat (const. vol.)',
cvliq_stg=4186.,     remark='Water liquid specific heat (const. vol.)',
rvap_stg=461.,       remark='Gas constant for water vapor',
rgas_stg=289.,       remark='Gas constant for air',
vaprlx_stg=0.8,      remark='Relaxation factor for phase change iterations',
rhlim_stg='y',       remark='Limit rel humidity to 100 percent',

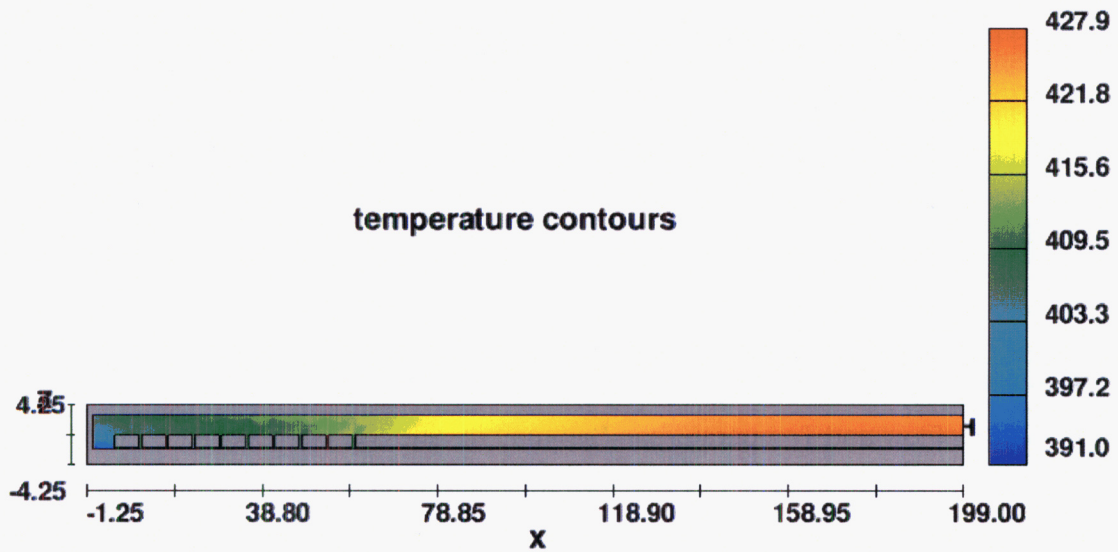
```

\$end

Documentation: general comments, background, expectations, etc.

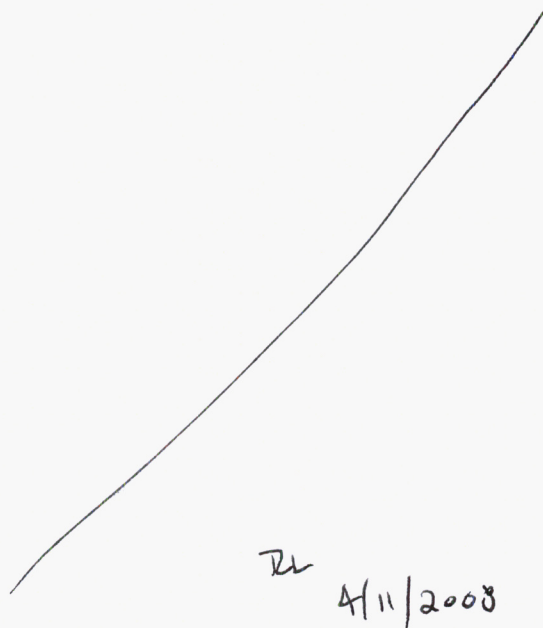
End of listing of the FLOW-3D input file prepin.YM_109yr_norad

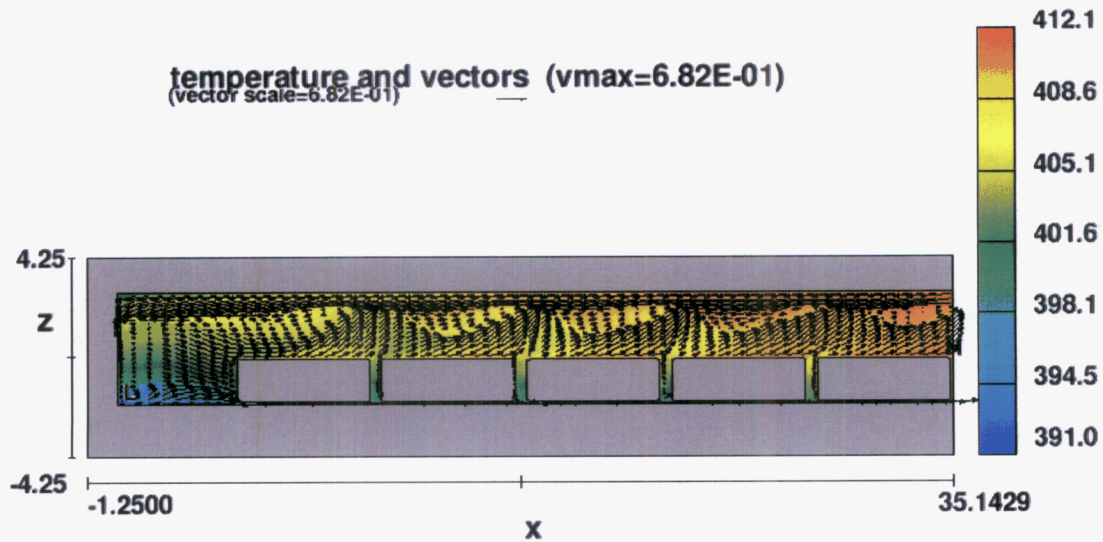
The temperature contours in the x-z symmetry plane are shown in Figure 8/6/06-2. There is an overall gradient along the axis of the drift due to the axial gradient in the rock as well as the flow of heat due to convection. The air velocity vectors are not shown here for the sake of clarity. The convection pattern is shown more clearly in Figure 8/6/06-3 for the area in the first 35 m from the end of the drift. There is a convection cell that circulates between the waste packages and the end of the drift. The flow is toward the cold end of the drift along the top of the drift and the air flows away from the drift along the sides of the waste packages and the invert. There is an upward flow of air in the gaps between waste packages that causes undulations in the axial convection cell.



FLOW-3D t=2.736E+03 y=4.911E-02 (ix=2 to 245 kz=2 to 59)
19:05:47 08/18/2006 khva hydr3d: version 9.0 win32-cvf 2005
M Cold Trap - Full Scale 3D, No Radiation, No Moisture, Temp. Gradient Boundary,

Figure 8/6/06-2. Temperature in Symmetry Plane, No Radiation.





FLOW-3D t=2.736E+03 y=4.911E-02 (ix=2 to 128 kz=2 to 59)
19:05:47 08/18/2006 khva hydr3d: version 9.0 win32-cvf 2005
M Cold Trap - Full Scale 3D, No Radiation, No Moisture, Temp. Gradient Boundary,

Figure 8/6/06-3 Flow and Temperature in Symmetry Plane No Radiation.

The flow patterns in the cross sections of the drift at $x \sim 30\text{m}$ and $x \sim 34\text{m}$ are shown in Figure 8/6/06-4. These locations are in the gap between the 4th and 5th waste packages and near the center of the 5th waste package, respectively.

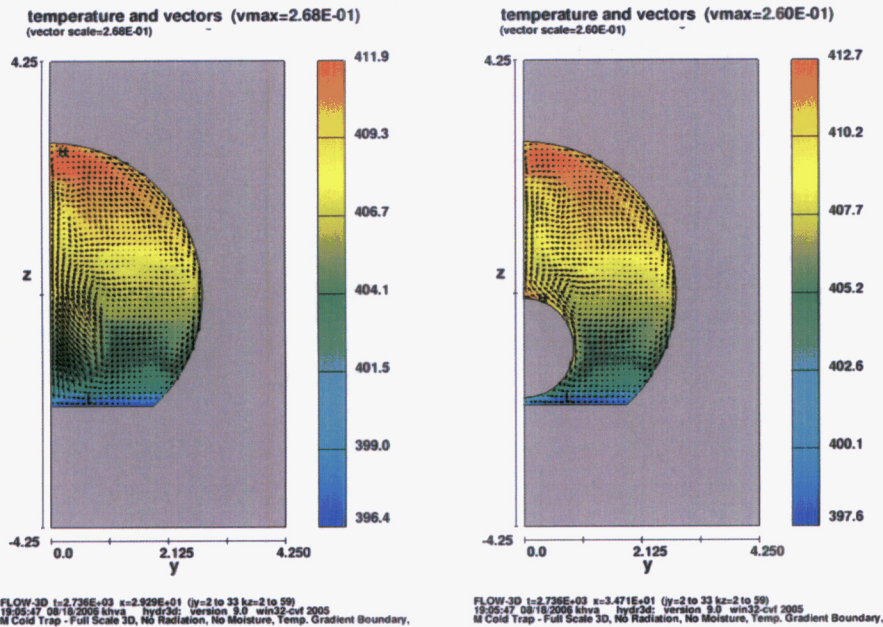
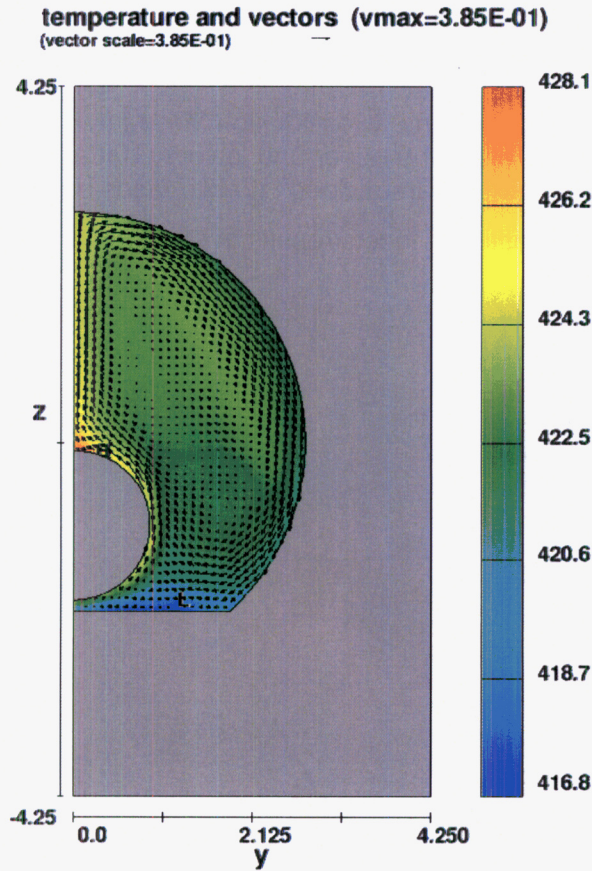


Figure 8/6/06-4 Flow and Temperature in Cross Section, Gap Between Waste Packages, No Radiation.

These graphs show that there is also convection between the waste packages and the drift wall in the drift cross section as well. The overall flow pattern is a complicated 3-dimensional flow field that carries heat from the waste packages to the drift wall.

The cross section flow pattern at $x=180$ m, well away from the end of the drift is shown in Figure 8/6/06-5. This flow field is similar to what one would expect from a purely two-dimensional flow between a cylinder and the drift wall.



FLOW-3D t=2.736E+03 x=1.814E+02 (jy=2 to 33 kz=2 to 59)
19:05:47 08/18/2006 khva hydr3d: version 9.0 win32-cvf 2005
M Cold Trap - Full Scale 3D, No Radiation, No Moisture, Temp. Gradient Boundary,

Figure 8/6/06-5 Flow and Temperature in Cross Section, Far From End Wall, No Radiation.

For the case in which radiation is included, the procedure described in the entry for 6/5/06 was followed. The waste packages and drift wall surfaces were divided into the regions defined in Figure 8/6/06-6. Around the cylindrical portion of the drift wall, there are three surfaces: 'Drift Upper', 'Drift Lower', and 'Invert.' The cylindrical surface of the waste package is divided into two surfaces: 'Waste package Upper' and 'Waste Package Lower'.

The radiation surfaces on the drift wall and the invert are divided at the same spacing as the waste packages. For the portion of the drift in the region $80.0 < x < 200$ the drift wall, invert and the waste package model are divided into 5 equally spaced axial divisions.

The closed end of the drift is divided into two radiation faces in which the split is at the same vertical location as the split between the upper and lower surfaces of the drift wall. Like wise, the two ends of each waste package are divided into two faces each.

Using these divisions for the radiation facets results in 99 radiation surfaces being generated for the half-model of the drift.

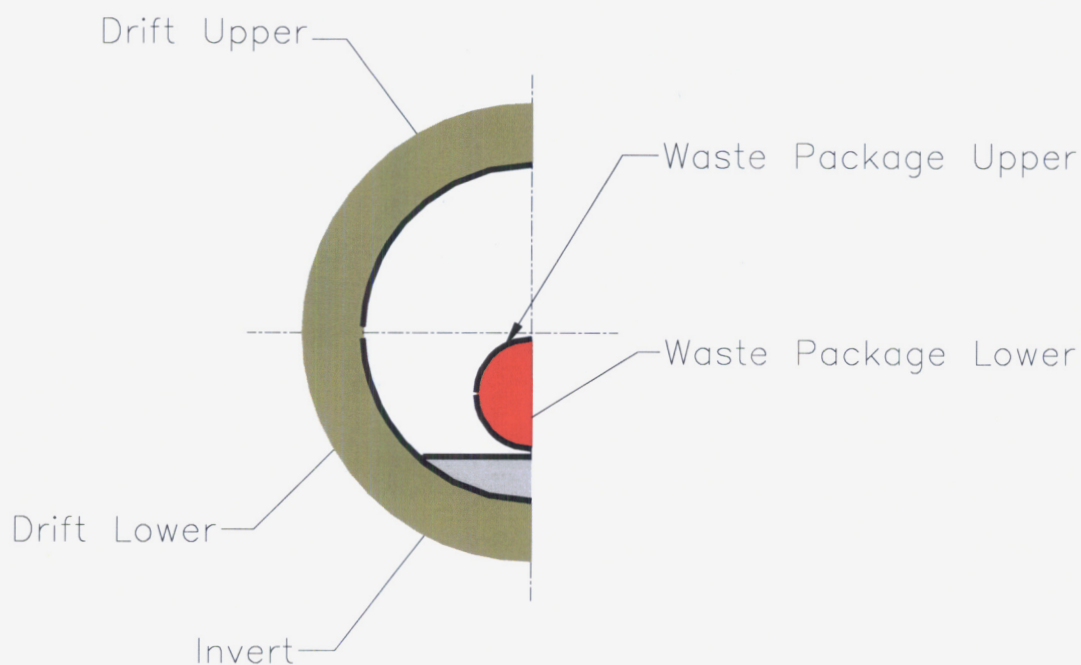


Figure 8/6/06-6. Geometry for Radiation Surfaces.
The radiation surfaces in the y-z cross section are seen here.

To correctly accommodate the parts of the drift that are on the other side of the symmetry plane, the complete drift geometry (with a total of 198 radiation facets) was modeled for a single time step. The radiation configuration factors for the surfaces on the +y side of the symmetry plane that is defined by the half-model of the drift were adjusted in accordance with the method described in the entry for 6/6/06. The calculations were performed in the Excel file,

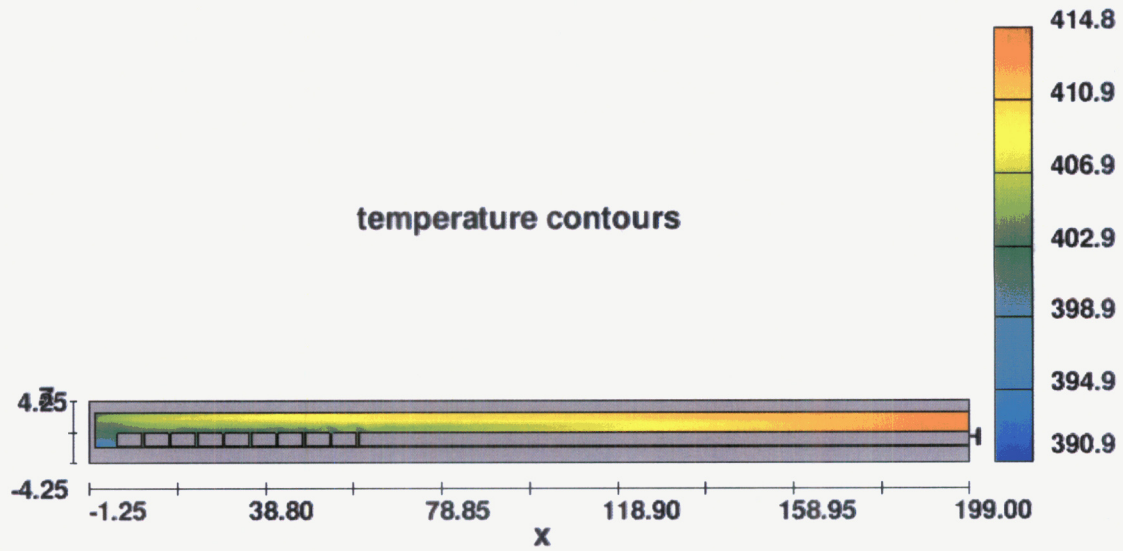
cfid_nosym-to-symm.xls

The adjusted configuration factors were copied into the FLOW-3D input file. Because all the configuration factors are specified in the input file, the listing is too long to include here. This input file is

prepinr.rad_restart-from-norad-2000

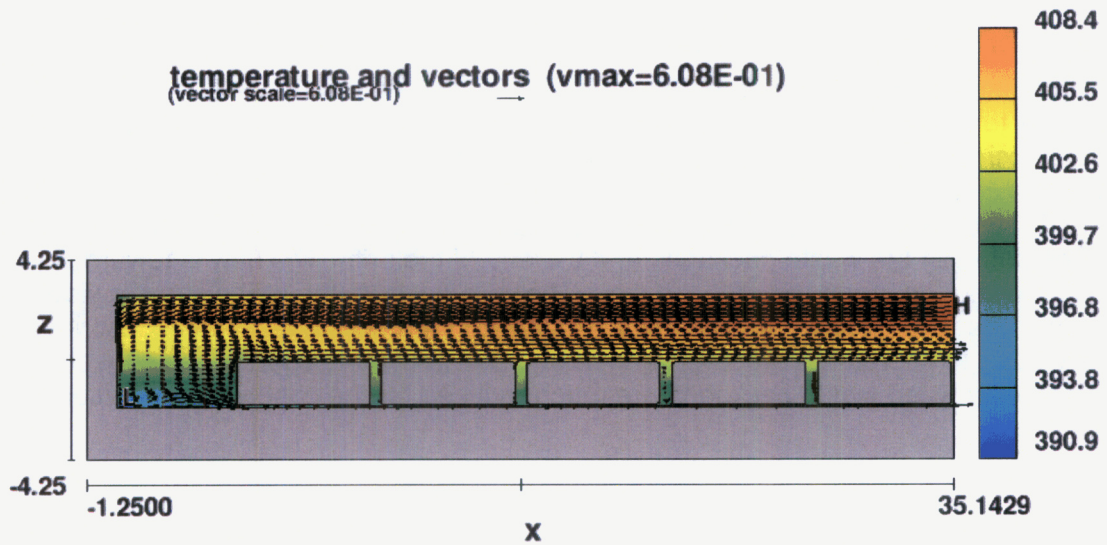
and was set up to be a restart from the no-radiation case described above. The file was executed from a time of 2000 sec (from a restart at $t=2000$ sec) to a time of 5000 sec. Execution was terminated before reaching completely a steady condition. Probably another 1000 sec would be needed, but this was considered to be sufficient for demonstrating the effects of including radiation on the heat transfer and flow.

The temperature contours in the x-z symmetry plane are shown in Figure 8/6/06-7. There is an overall gradient along the axis of the drift due to the axial gradient in the rock as well as the flow of heat due to convection. This is similar to that shown above for the no-radiation case, but note that the maximum temperature is now about 415 K whereas the maximum temperature for the no-radiation case was about 428 K. The convection pattern is shown more clearly in Figure 8/6/06-8 for the area in the first 35 m from the end of the drift. There is a convection cell that circulates between the waste packages and the end of the drift. The flow is toward the cold end of the drift along the top of the drift and the air flows away from the drift along the sides of the waste packages and the invert. Again, this is similar to the no-radiation case described above, but the overall temperatures are lower and more importantly, the undulations caused by the upward flow between waste packages are much reduced for the case when radiation is included.



FLOW-3D t=5.000E+03 y=4.911E-02 (ix=2 to 245 kz=2 to 59)
 16:14:33 08/22/2006 cjno hydr3d: version 9.0 win32-cvf 2005
 YM Cold Trap - Full Scale 3D, Radiation, No Moisture, Temp. Gradient Boundary, L

Figure 8/6/06-7. Temperature in Symmetry Plane, Radiation.



FLOW-3D t=5.000E+03 y=4.911E-02 (ix=2 to 128 kz=2 to 59)
 16:14:33 08/22/2006 cjno hydr3d: version 9.0 win32-cvf 2005
 YM Cold Trap - Full Scale 3D, Radiation, No Moisture, Temp. Gradient Boundary, L

Figure 8/6/06-8 Flow and Temperature in Symmetry Plane, Radiation.

The temperature predictions of the two scenarios are compared more directly in Figure 8/6/06-9 and Figure 8/6/06-10. Figure 8/6/06-9 shows that cross-section-averaged temperature of the two sets of predictions. When radiation effects are included in the simulations, the average temperature of the air in the drift is significantly less than when radiation is neglected. This is

because a large amount of heat is transferred directly from the waste packages to the drift walls via radiation. The air does not receive and transport this energy; therefore, the air temperature is reduced.

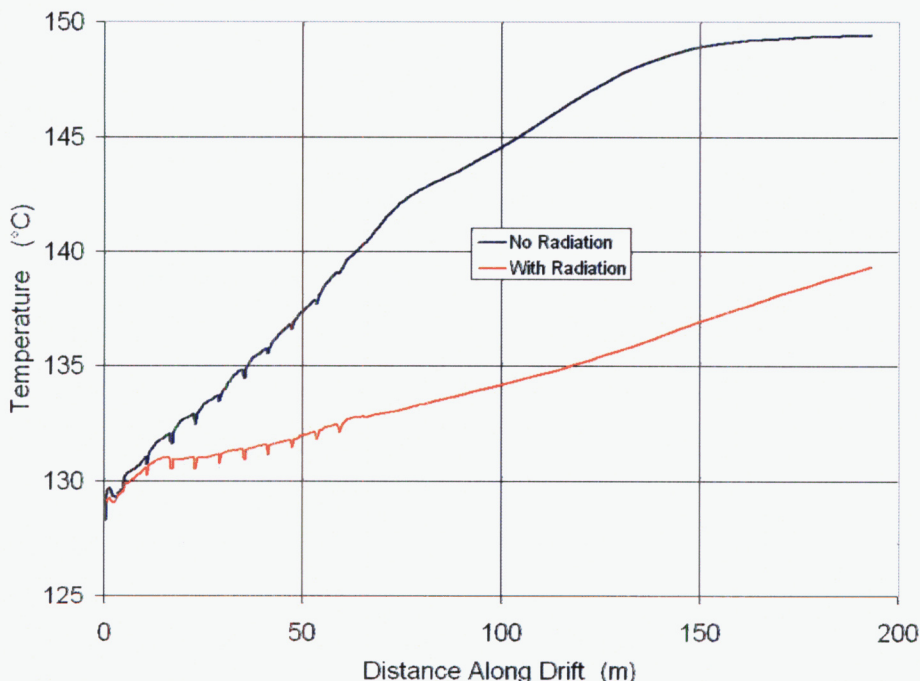
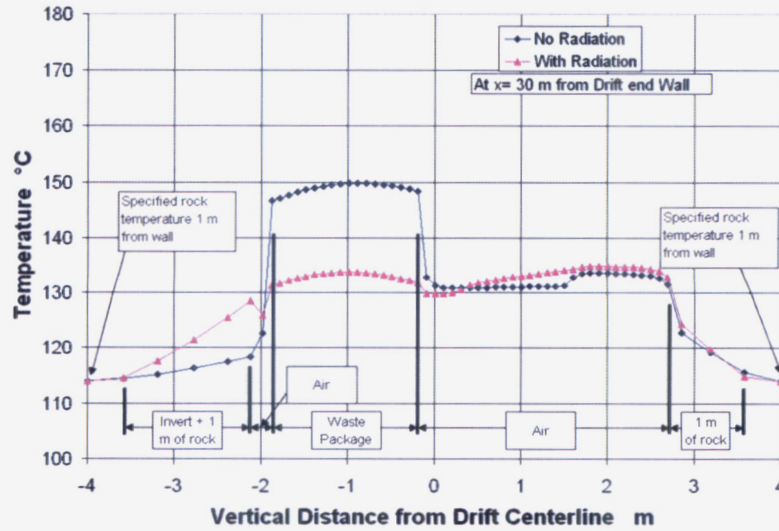
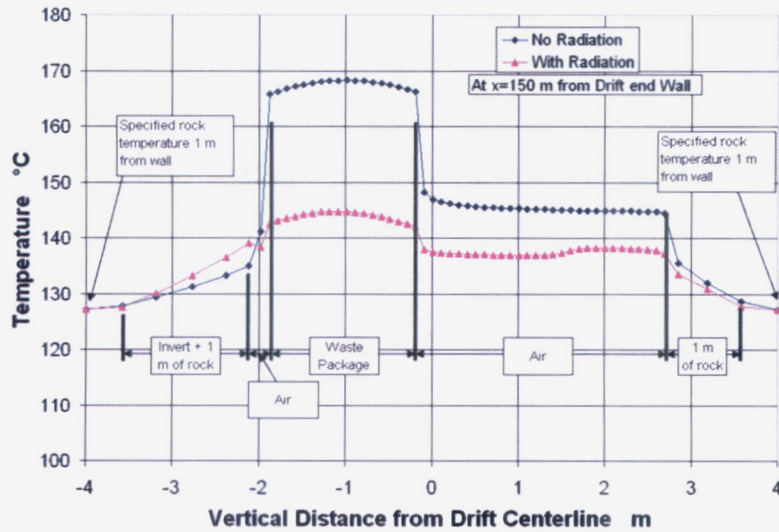


Figure 8/6/06-9. Cross-Section-Averaged Temperature Along Drift

Figure 8/6/06-10 shows the temperature profile along a line from the bottom of the simulation domain to the top of the domain in the plane of symmetry at two locations. Figure 8/6/06-10a show the profile at x=32 m near the center of the 4th waste package. When radiation effects are included, the waste package temperature is about 12°C [22°F] lower than in the no-radiation case. The invert surface temperature is greater than the nearby air temperature when radiation effects are included. This phenomenon is not observed at the top of the drift because 1) the radiation heat flux is reduced when the surfaces are farther apart, and 2) the air circulation is stronger at the top of the drift and partially mitigates the radiation effects. The predicted temperature profile at the 32 m position (Figure 8/6/06-10a), also shows that the air temperature near the top drift wall is slightly greater than the waste package temperature because of the heat carried from the hotter section of the drift. This feature is not seen in the profile at 150 m, Figure 8/6/06-10b, because the overall axial air circulation is much less at this location than at locations closer to the drift end



(a)



(b)

Figure 8/6/06-10. Vertical Temperature Profile.
(a) $x=32$ m, (b) $x=150$ m

The axial air circulation rate is computed by integrating the fluid velocity of the cross section for all locations where the gas is moving away from the closed end of the drift. It is assumed that the drift wall is impermeable in these simulations and the drift is non-porous. So, air circulation rate is the volumetric flow of air exchanged between volumes on either side of a plane at a specified axial location. The calculated air circulation rates are shown in Figure 8/6/06-11. The circulation rate is strongest near the closed end of the drift where the air circulates between the hot waste package and the relatively cooler end wall. The axial air circulation shows numerous perturbations in this region because of the plumes rising from the waste package gaps. Overall,

the air circulation rate is significantly greater when radiation effects are included in the simulations compared to the case when radiation is neglected. The air circulation decreases to a small value before the no-flow boundary at $x = 200$ m is reached in the no-radiation simulation and is not consistent with the values observed in the simulations when radiation is included. This indicates that additional drift length should be included in the simulations to account for the effects of processes. That is, the no-flow boundary should be greater than 200 m from the closed end.

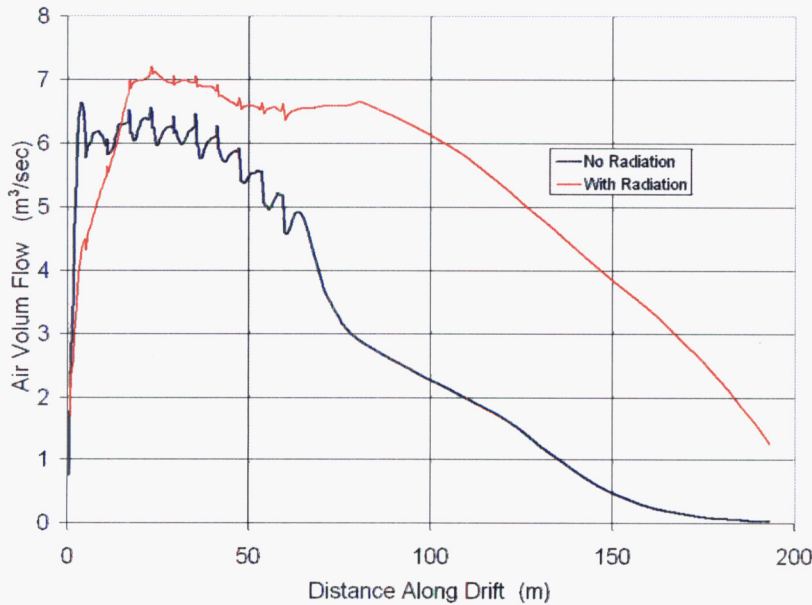


Figure 8/6/06-11. Air Circulation Rate Along Drift

END OF ENTRY FOR 8/6/06 *STG*

=====

Entries made into Scientific Notebook #536E for the period April 1, 2006 to September 30, 2006, have been made by Steven Green (October 2, 2006).

No original text or figures entered into this Scientific Notebook has been removed

STG 10/2/2006

=====

3/14/08 *STG*

THE PURPOSE OF THIS ENTRY IS TO MAKE A CORRECTION AND TO CLOSE THE NOTEBOOK.

+++++

CORRECTION TO ENTRY FOR 3/16/06

On p. 252 of this notebook, the filename listed for the 1-D radiation analysis summary is incorrectly listed as

1-D_rad_Analysis-FLOW-3D.xls

The correct file name is

Compare_Analysis-FLOW-3D.xls

+++++

This notebook is to be closed per this entry.

END OF ENTRY FOR 3/14/08 *STG*

=====

Entries made into Scientific Notebook #536E for the period October 1, 2006 to March 15, 2008, have been made by Steven Green (March 14, 2006).

No original text or figures entered into this Scientific Notebook has been removed

STG 3/14/2008

=====