```
      implicit double precision (a-h,o-z)
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c     vdpdh1.in must be filled as follows:
c
c     .....
c
c     uses subroutines .....
c
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
      dimension yt(514)
      character filename*10
      common/param1/a(256,256),yx(256),g(256),
     ff       rha(256),x(256),d(256),ipvt(256),gme,rinp,
     ff       epst,deltax,tref,n
      double precision a,yx,g,rha,x,d,gme,rinp,
     f       epst,deltax,tref
      integer*4 ipvt,n
      common/geomet/rl,b,rh,rho,rmseff,resme,stifme,as,amred,gm
      double precision rl,b,rh,rho,rmseff,resme,stifme,as,amred,gm
      common/ref/iwrite
      open(10,iostat=ios,err=2000,file='vdpdhl.in',status='old')
      read(10,99993)iwrite
      print *,iwrite
      read (10,99993)nd
      print *,nd
      read (10,99993)k
      print *,k
      read (10,99993)kwrite
      print *,kwrite
      read (10,99994)et
      print *,et
      read (10,99993)istept
      print *,istept
      read (10,99994)st
      print *,st
      read(10,99992)filename
      print *,filename
      open(9,iostat=ios,err=2000,file=filename,form='unformatted')
      read(10,99992)filename
      print *,filename
      open(11,iostat=ios,err=2000,file=filename,form='unformatted')
      read(10,99992)filename
      print *,filename
      open(12,iostat=ios,err=2000,file=filename,form='unformatted')
      read(10,99992)filename
      print *,filename
      open(13,iostat=ios,err=2000,file=filename,form='unformatted')
      read(10,99992)filename
      print *,filename
      open(14,iostat=ios,err=2000,file=filename,form='unformatted')
      read(10,99992)filename
      print *,filename
      open(15,iostat=ios,err=2000,file=filename,form='unformatted')
      read(10,99993)isw
      print *,isw
      read(10,99993)ifill
      print *,ifill
      read(10,99994)epst
      print *,epst
      read(10,99994)rl
      print *,rl
      read(10,99994)b
      print *,b
      read(10,99994)rh
      print *,rh
      read(10,99994)rho
      print *,rho
      read(10,99994)rmsme
      print *,rmsme
      read(10,99994)resme
      print *,resme
      read(10,99994)stifme
      print *,stifme
      read(10,99994)gm
```

Original Fortran 77 by Rob Diependaal

```
      print *,gm
      read(10,99994)am
      print *,am
      read(10,99994)as
      print *,as
      nr=nd/2
c        read(11)adum,bdum,ndum,cdum,ddum,kdum
c        read(12)adum,bdum,ndum,cdum,ddum,kdum
c        read(14)adum,bdum,ndum,cdum,ddum,kdum
c        read(15)adum,bdum,ndum,cdum,ddum,kdum
c              kdum=1
c     do 500 jcount=1,kdum
c        read(11)(yt(j),j=2,nr)
c        read(12)(yt(nr+j),j=2,nr)
c        read(14)yt(1)
c        read(15)yt(nr+1)
c 500 continue
      rewind (11,iostat=ios,err=3000)
      rewind (12,iostat=ios,err=3000)
      rewind (14,iostat=ios,err=3000)
      rewind (15,iostat=ios,err=3000)
      do 1000 j=1,nd
         yt(j)=0d0
         if(j.le.nr-1)ipvt(j)=0
 1000 continue
      deltax=rl/dfloat(nr-1)
      t=st
      deltat=(et-st)/dfloat(k)
      amred=am/gm/as
      rmseff=2d0*rho*rl+rmsme
      call t3d1fm(nd,k,kwrite,deltat,istept,t,yt,isw,ifill)
      write(iwrite,99999)isw
      close(9)
      close(10)
      close(11)
      close(12)
      close(13)
      close(14)
      close(15)
      stop
 2000 write(iwrite,99996)ios
      stop
 3000 write(iwrite,99995)ios
      stop
99999 format(' isw= ',i2)
99996 format(' error in open statement, ios=',i5)
99995 format(' error in rewind statement, ios=',i5)
99994 format(d13.6)
99993 format(i5)
99992 format(a)
      end
!
!
      subroutine rhst1(t,yin,yout)
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c
c     routine rhst1
c
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c
c     first version: october, 24th - november, 1st 1985
c
c     author: rob diependaal
c
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c
c     computes right hand side of differential equation
c
c     input parameters:
c        t-time
c        yin-dependent variable
c
c     output parameters:
c        yout-right hand side
```

Original Fortran 77 by Rob Diependaal

```
c
c     common parameters: param1 (see routine sol1d1)
c
c     uses subroutines
c
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
      implicit double precision (a-h,o-z)
      dimension yin(514),yout(514),ak(256,1)
      common/param1/a(256,256),yx(256),g(256),
     f       rha(256),x(256),d(256),ipvt(256),gme,rinp,
     f       epst,deltax,tref,n
      double precision a,yx,g,rha,x,d,gme,rinp,
     f       epst,deltax,tref
      integer*4 ipvt,n
      common/geomet/rl,b,rh,rho,rmseff,resme,stifme,as,amred,gm
      double precision rl,b,rh,rho,rmseff,resme,stifme,as,amred,gm
c     stiffn(xx,uu,vv)=2d4*dexp(-3d-1*xx)
      stiffn(xx,uu,vv)=1d4*dexp(-3d-1*xx)
      presed(tt)=3.16d-5*dsin(8d0*datan(1d0)*tt)
      rmass(xx)=5d-1
c     beta(xx)=8d-2*dexp(5d-2*xx)
      beta(xx)=1d0
c     b2(xx)=85d-2+3d-3*xx
      b2(xx)=1d0
      b1(xx)=b2(xx)-beta(xx)
      pi=4d0*datan(1d0)
c
      if(t.eq.tref)goto 2000
      gme=resme*yin(n+1)+stifme*yin(1)
      g(1)=resist(0d0,yin(2),yin(n+2))*yin(n+2)+
     f   stiffn(0d0,yin(2),yin(n+2))*yin(2)
      rinp=presed(t)
      if(t.le.4d0)rinp=rinp*dexp((4d0-t)*(t-4d0)/2d0)
      coef2=(gme-amred*rinp)
      yx(1)=(g(1)-coef2*rha(1))/d(1)
      do 1000 i=3,n
         g(i-1)=resist(x(i-1),yin(i),yin(n+i))*yin(n+i)+
     f       stiffn(x(i-1),yin(i),yin(n+i))*yin(i)
         yx(i-1)=(g(i-1)-coef2*rha(i-1))/d(i-1)
 1000 continue
      call dgesl(a,256,n-1,ipvt,yx,0)
 2000 do 3000 i=2,n
         yout(i)=yin(n+i)
         yout(n+i)=yx(i-1)
 3000 continue
      yout(1)=yin(n+1)
      sum=deltax*beta(0d0)*rl*yx(1)
      do 4000 j=2,n-1
         x1=x(j)
         sum=sum+2d0*deltax*beta(x1)*(rl-x1)*yx(j)
 4000 continue
      yout(n+1)=(amred*rinp-gme-2d0*rho/pi/b/rh*sum)/rmseff
      return
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c
c     end of routine rhst1
c
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
      end
      subroutine t3d1fm(nd,k,kwrite,deltat,istept,t,yt,isw,ifill)
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c
c     routine t3d1fm
c
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c
c     first version: october 2nd, 1987
c
c     author: rob diependaal
c
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c
c     computes displacement, velocity, and pressure
c     in the time domain, based on a 3d-model with two boundary
```

Original Fortran 77 by Rob Diependaal

```
c     conditions in the place.
c     writes the computed quantities to background memory
c
c     input parameters:
c        nd:   dimension of yt
c        k: number of time steps
c        kwrite:  time step from which onwards the quantities
c                 will be written to memory
c        deltat:  time step
c        istept:  =1 -integration step (in the time) is variable
c                 =2 -integration step is constant
c        t: time
c        yt:   contains displacement and velocity
c        isw:  =1 -extra yx (see common)computations will be done
c              =0 -no extra computations will be done
c        ifill:=1 -a and x (see common) will be filled
c              =0 -a must be filled and decomposed
c                 and x must be filled
c
c     output parameters:
c        t,yt
c        isw:  =0 -no error has occurred
c              =2 -error in input parameters
c              =3 -error in integration in the time
c     common parameters:
c        param1:  a,yx,g,x,epst,deltax,tref,n
c           a-matrix containing information about
c              discretisation in the place
c           yx-contains accelleration
c           g-contains pressure minus mass term
c           x-contains points along x-axis
c           epst-accuracy for integration in the time
c           deltax-place step
c           tref-reference time
c           n-number of discretisation intervals in the place
c
c     uses subroutines
c
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
      implicit double precision (a-h,o-z)
      dimension yt(nd),yp(256),work(256)
      common/param1/a(256,256),yx(256),g(256),
     f   rha(256),x(256),d(256),ipvt(256),gme,rinp,
     f   epst,deltax,tref,n
      double precision a,yx,g,rha,x,d,gme,rinp,
     f   epst,deltax,tref
      integer*4 ipvt,n
      common/parlu/alu(256,3),blu(256,1),int(256),yxlu(256)
      common/geomet/rl,b,rh,rho,rmseff,resme,stifme,as,amred,gm
      double precision rl,b,rh,rho,rmseff,resme,stifme,as,amred,gm
      common/ref/iwrite
      common/comsum/asi(1024),aco(1024),rnum(100,256),den(100,256)
      double precision asi,aco,rnum,den
      external rhst1
c     stiffn(xx,uu,vv)=2d4*dexp(-3d-1*xx)
      stiffn(xx,uu,vv)=1d4*dexp(-3d-1*xx)
!      presed(tt)=3.16d-5*dsin(8d0*datan(1d0)*tt)
      presed(tt)=3.16d-7*dsin(8d0*datan(1d0)*tt)
      rmass(xx)=5d-1
c     beta(xx)=8d-2*dexp(5d-2*xx)
      beta(xx)=1d0
c     b2(xx)=85d-2+3d-3*xx
      b2(xx)=1d0
      b1(xx)=b2(xx)-beta(xx)
      dcoth(xx)=dcosh(xx)/dsinh(xx)
      epssum=1d-2
      epsmod=1d-4
      maxsum=1000
      maxmod=100
      pi=4d0*datan(1d0)
      alpha=8d0*rho/(pi*pi*b)*deltax
      pddh=pi*deltax/rh
      tpddh=2d0*pddh
      pdd2h=pddh/2d0
```

Original Fortran 77 by Rob Diependaal

```
      pdd4h=pdd2h/2d0
      t4=pi*rl/rh
      coef=8d0*rho/pi/rmseff
      grme=4d0*rho/rh/rmseff
c
c     tests
c
      if(nd.gt.514)isw=2
      if(isw.lt.0.or.isw.gt.1)isw=2
      if(k.le.0.or.k.lt.kwrite.or.kwrite.le.0)isw=2
      n=nd/2
      if(2*n.ne.nd)isw=2
      if(isw.eq.2)return
c
c     initiations
c
      if(ifill.eq.0)goto 2000
      do 1000 j=1,4*(n-1)
         asi(j)=dsin(dble(j)*pi/2d0/dble(n-1))
         aco(j)=dcos(dble(j)*pi/2d0/dble(n-1))
 1000 continue
c
c     fills a, rha and x
c
      x(1)=0d0
      rha(1)=coef*rl
      aux=alpha*beta(0d0)
      x(2)=deltax
      x1=deltax
      t1=dlog(2d0*(dcosh(pddh)-1d0))
      t7=pdd2h*dcoth(pdd2h)
      t5=t7/2d0
      gr=t1-t4-t7-1d0
      do 1100 j=1,2
         bet=beta(dble(j-1)*deltax)
         bb1=b1(dble(j-1)*deltax)
         bb2=b2(dble(j-1)*deltax)
         do 1110 i=1,maxmod
            rnum(i,j)=dcos(dble(i)*pi*bb2/b)+
     f                dcos(dble(i)*pi*bb1/b)
            help=dble(i)*bet/b
            den(i,j)=1d0-help*help
            if(dabs(den(i,j)).lt.1d-10)then
               den(i,j)=1d0
               rnum(i,j)=pi/2d0*dsin(pi*bb2/bet)
            endif
 1110    continue
 1100 continue
      next=idint(b/beta(0d0))
      nex1=next
      call sum3d(1,1,epssum,epsmod,gr,maxsum,maxmod,next,
     f     deltax,n-1,gr3d)
      a(1,1)=aux*(gr/pi-gr3d+grme/2d0*rl*rl)-rmass(0d0)
      d(1)=dabs(a(1,1))
c      write(iwrite,*)a(1,1)
      rha(2)=coef*(rl-x1)
      t2=dlog(2d0*(dcosh(tpddh)-1d0))
      t6=pdd2h*dcoth(pddh)
      gr=t2-t4-t6-t7-5d-1
      next=max(nex1,idint(b/beta(x1)))
      call sum3d(2,1,epssum,epsmod,gr,maxsum,maxmod,next,
     f     deltax,n-1,gr3d)
      a(2,1)=aux*(gr/pi-gr3d+grme/2d0*rl*(rl-x1))
      d(2)=dabs(a(2,1))
c       write(iwrite,*)a(2,1)
      a1=pddh
      do 100 i=3,n-1
         x1=x(i-1)+deltax
         x(i)=x1
         rha(i)=coef*(rl-x1)
         a1=a1+pddh
         t8=dlog(2d0*(dcosh(a1)-1d0))
         gr=t8-t4
         bet=beta(x1)
```

Original Fortran 77 by Rob Diependaal

```
          bb1=b1(x1)
          bb2=b2(x1)
          do 1120 ii=1,maxmod
             rnum(ii,i)=dcos(dble(ii)*pi*bb2/b)+
     f                  dcos(dble(ii)*pi*bb1/b)
             help=dble(ii)*bet/b
             den(ii,i)=1d0-help*help
             if(dabs(den(ii,i)).lt.1d-10)then
                den(ii,i)=1d0
                rnum(ii,i)=pi/2d0*dsin(pi*bb2/bet)
             endif
 1120     continue
          next=max(nex1,idint(b/bet))
          call sum3d(i,1,epssum,epsmod,gr,maxsum,maxmod,next,
     f         deltax,n-1,gr3d)
          a(i,1)=aux*(gr/pi-gr3d+grme/2d0*rl*(rl-x1))
          d(i)=dabs(a(i,1))
c           write(iwrite,*)a(i,1)
  100 continue
      do 200 j=2,n-1
          x2=x(j)
          a4=pi*x2/rh
          aux=alpha*beta(x2)
          nex1=idint(b/beta(x2))
          do 210 i=1,n-1
             x1=x(i)
             a1=pi*x1/2d0/rh
             a2=2d0*a1
             a3=2d0*a2
             next=max(nex1,idint(b/beta(x1)))
             if(iabs(i-j).ge.2)then
                if(i.eq.1)then
                   t1=2d0*dlog(2d0*(dcosh(a4)-1d0))
                else
                   t1=2d0*dlog(2d0*dabs(dcosh(a2)-dcosh(a4)))
                endif
                gr=t1-2d0*t4
                call sum3d(i,j,epssum,epsmod,gr,maxsum,maxmod,next,
     f                deltax,n-1,gr3d)
                a(i,j)=aux*(gr/pi-gr3d+grme*(rl-x1)*(rl-x2))
             else
                if(iabs(i-j).eq.1)then
                   if(i.eq.1)then
                      gr=2d0*(t1-t4)-t7-1d0
                      call sum3d(i,j,epssum,epsmod,gr,maxsum,maxmod,
     f                     next,deltax,n-1,gr3d)
                      a(i,j)=aux*(gr/pi-gr3d+grme*rl*(rl-x2))
                   else
                      t1=2d0*dlog(2d0*(dabs(dcosh(a2)-dcosh(a4))))
                      t6=a1*dcoth(a2)
                      if(i.lt.j)then
                         t3=2d0*x1/deltax*dlog(dsinh(a2+pdd2h)/dsinh(a2))
                         t8=(a1+pdd4h)*dcoth(a2+pdd2h)
                      else
                         t3=2d0*x1/deltax*dlog(dsinh(a2)/dsinh(a2-pdd2h))
                         t8=(a1-pdd4h)*dcoth(a2-pdd2h)
                      endif
                      gr=t1+t3-t5-t6-t8-2d0*t4-5d-1
                      call sum3d(i,j,epssum,epsmod,gr,maxsum,maxmod,
     f                     next,deltax,n-1,gr3d)
                      a(i,j)=aux*(gr/pi-gr3d+grme*(rl-x1)*(rl-x2))
                   endif
                else
                   t1=dlog(4d0*(dcosh(pddh)-dcosh(a3))+2d0*
     f                  (dcosh(a3+pddh)+dcosh(a3-pddh)-
     f                  dcosh(tpddh)-1d0))
                   t3=2d0*x1/deltax*dlog(dsinh(a2+pdd2h)/dsinh(a2-pdd2h))
                   t6=a2*dcoth(a2)
                   t8=(a1+pdd4h)*dcoth(a2+pdd2h)
                   t9=(a1-pdd4h)*dcoth(a2-pdd2h)
                   gr=t1+t3-t7-t6-t8-t9-2d0*t4-1d0
                   call sum3d(i,j,epssum,epsmod,gr,maxsum,maxmod,
     f                  next,deltax,n-1,gr3d)
                   a(i,j)=aux*(gr/pi-gr3d+grme*(rl-x1)**2)-rmass(x1)
```

Original Fortran 77 by Rob Diependaal

```
                endif
              endif
c               write(iwrite,*)a(i,j)
              d(i)=d(i)+dabs(a(i,j))
  210       continue
  200     continue
           print *,'A is filled'
          write(9)n-1
          write(9)((a(l1,l2),l1=1,n-1),l2=1,n-1)
          write(9)(rha(l1),l1=1,n-1)
          write(9)(x(l1),l1=1,n-1)
          write(9)(d(l1),l1=1,n-1)
c
c         equilibrates the rows of a
c
          do 300 i=1,n-1
             equil=d(i)
             do 310 j=1,n-1
                a(i,j)=a(i,j)/equil
c               write(iwrite,*)a(i,j)
  310        continue
  300     continue
c
c         decomposes a in lower and upper triangle
c
          call dgeco(a,256,n-1,ipvt,rcond,work)
          write(9)((a(l1,l2),l1=1,n-1),l2=1,n-1)
          write(9)(ipvt(l1),l1=1,n-1)
          rewind (9)
 2000     if(ifill.eq.0)then
             read(9)nn
             read(9)((a(l1,l2),l1=1,nn),l2=1,nn)
             read(9)(rha(l1),l1=1,nn)
             read(9)(x(l1),l1=1,nn)
             read(9)(d(l1),l1=1,nn)
             read(9)((a(l1,l2),l1=1,nn),l2=1,nn)
             read(9)(ipvt(l1),l1=1,nn)
          endif
          ht=deltat
          tout=t
          k1=k+1
          jhelp=0
c
          do 3000 j=1,k1
c
c            do-loop over time variable
c
                write(iwrite,33101)j
33101           format(' time step number: ',i7)
             if(isw.eq.0)goto 3300    !3200
             gme=resme*yt(n+1)+stifme*yt(1)
             g(1)=resist(0d0,yt(2),yt(n+2))*yt(n+2)+
     f          stiffn(0d0,yt(2),yt(n+2))*yt(2)
             rinp=presed(t)
             if(t.le.4d0)rinp=rinp*dexp((4d0-t)*(t-4d0)/2d0)
             coef2=gme-amred*rinp
             yx(1)=(g(1)-coef2*rha(1))/d(1)
             do 3100 i=3,n
                i1=i-1
                xhelp=x(i1)
                g(i1)=resist(xhelp,yt(i),yt(n+i))*yt(n+i)+
     f             stiffn(xhelp,yt(i),yt(nd+i))*yt(i)
                yx(i1)=(g(i1)-coef2*rha(i1))/d(i1)
 3100        continue
             call dgesl(a,256,n-1,ipvt,yx,0)
             goto 3300
 3200        isw=1
 3300        jj=j-1
!            jhelp=jhelp+1
!            if(jj.lt.kwrite.and.jhelp.eq.128)then
!               do 3310 kw=11,12
!                  write(kw)x(n-1),deltax,n-1,t,deltat,1
! 3310          continue
!               do 3320 kw=14,15
```

Original Fortran 77 by Rob Diependaal

```
!               write(kw)0d0,0d0,1,t,deltat,1
! 3320         continue
!              write(11)(yt(l),l=2,n)
!              write(12)(yt(l),l=n+2,nd)
!              write(14)yt(1)
!              write(15)yt(n+1)
!              rewind (11,iostat=ios,err=5000)
!              rewind (12,iostat=ios,err=5000)
!              rewind (14,iostat=ios,err=5000)
!              rewind (15,iostat=ios,err=5000)
!              jhelp=0
!           endif
      if(jj.lt.kwrite)then
      goto 3600
!          if(jj.gt.kwrite)goto 3500
     else if(jj.eq.kwrite) then
          khelp=k1-kwrite
          tend=t+(khelp-1)*deltat
          do 3400 kw=11,13
             write(kw)x(n-1),deltax,n-1,tend,deltat,khelp
 3400     continue
          do 3450 kw=14,15
             write(kw) 0d0,0d0,1,tend,deltat,khelp
 3450     continue
     end if
c3500     do 3550 i=1,n-1
c            yp(i)=rmass(x(i))*yx(i)+g(i)
c3550     continue
 3500     write(11)(yt(l),l=2,n)
          write(12)(yt(l),l=n+2,nd)
c         write(13)(yp(l),l=1,n-1)
c         write(14)yt(1)
c         write(15)yt(n+1)
 3600     if(j.eq.k1)goto 4000
          iflag=istept
          tout=tout+deltat
          tref=t
 3700     call rkf4st(rhst1,t,tout,yt,ht,nd,epst,iflag)
             if(iflag.eq.2)write(iwrite,33001)iflag
33001        format(' iflag=',i2)
          if(iflag.eq.2)goto 3700
          if(iflag.eq.3)goto 3000
          write(iwrite,99999)j
          isw=4
          return
 3000 continue
 4000 isw=0
      return
! 5000 write(iwrite,99995)ios
!      return
99999 format(' error in integration in t-direction, j= ',i5)
99995 format(' t3d1fm: error in rewind statement, ios=',i5)
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c
c     end of routine t3d1fm
c
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
      end
      double precision function resist(xx,uu,vv)
      implicit double precision (a-h,o-z)
c
      if(dabs(vv).lt.1d-20)then
         r1=1d0
      else
         r1=1d-4*dsinh(vv*1d4)/vv
      endif
      if(dabs(vv).gt.170d0)then
         r2=0d0
      else
         r2=2d0/dcosh(vv*1d6)
      endif
      resist=25d-1*dsqrt(2d0)*(r1-r2)*dexp(-15d-2*xx)
c     resist=5d0*dexp(-15d-2*xx)
      return
```

Original Fortran 77 by Rob Diependaal

```
      end
      subroutine sum3d(i,j,epssum,epsmod,gr,maxsum,maxmod,next,
     f           deltax,ndim,gr3d)
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c
c     routine sum3d
c
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c
c     author: Rob Diependaal
c
c     first version: November 6th, 1987
c
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
      implicit double precision (a-h,o-z)
      common/comsum/asi(1024),aco(1024),rnum(100,256),den(100,256)
      double precision asi,aco,rnum,den
      common/geomet/rl,b,rh,rho,rmseff,resme,stifme,as,amred,gm
      double precision rl,b,rh,rho,rmseff,resme,stifme,as,amred,gm
      common/ref/iwrite
c
      pi=4d0*datan(1d0)
      m=4*ndim
      i1=i-1
      j1=j-1
      gr3d=0d0
      grref=gr*rl/pi
c
      do 1000 n=1,maxmod
         t1sq=dble(n)*pi/b
         t1sq=t1sq*t1sq
         sum=0d0
         do 1100 k=0,maxsum
            coef=(dble(k)+5d-1)*pi/rl
            coefsq=coef*coef
            gamma=dsqrt(coefsq+t1sq)
            if(j.eq.1)then
               if(i.ge.3)then
                  ind=mod((2*k+1)*i1,m)
                  term=aco(ind)
               else
                  if(i.eq.2)then
                     ind=mod(4*k+2,m)
                     term=asi(ind)/2d0/coef/deltax
                  else
                     ind=mod(2*k+1,m)
                     term=asi(ind)/coef/deltax
                  endif
               endif
            else
               if(iabs(i-j).ge.2)then
                  ind=mod((2*k+1)*(i1+j1),m)
                  term=aco(ind)
                  if(i.eq.1)then
                     term=2d0*term
                  else
                     ind=mod((2*k+1)*iabs((j-i)),m)
                     term=term+aco(ind)
                  endif
               else
                  if(iabs(i-j).eq.1)then
                     ind=mod(2*k+1,m)
                     t1=asi(ind)/coef/deltax+aco(ind)
                     if(i.ne.1)then
                        ind=mod((4*k+2)*i1,m)
                        t2=asi(ind)/coef/deltax
                        if(i.gt.j)then
                           ind=mod((2*k+1)*(2*i1-1),m)
                           term=(t1+t2-asi(ind)/coef/deltax+
     f                          aco(ind))/2d0
                        else
                           ind=mod((2*k+1)*(2*i1+1),m)
                           term=(t1-t2+asi(ind)/coef/deltax+
     f                          aco(ind))/2d0
```

Original Fortran 77 by Rob Diependaal

```
                  endif
               else
                  term=t1
               endif
            else
               ind=mod(2*k+1,m)
               term=asi(ind)/coef/deltax
               ind=mod((4*k+2)*i1,m)
               term=term*(1d0+aco(ind))
            endif
         endif
      endif
      term=term/gamma
      sum=sum+term
c     if(dabs(term/sum).lt.epssum)goto 1200
 1100 continue
c     write(iwrite,99999)n,epssum
 1200 sum=sum*rnum(n,i)*rnum(n,j)/den(n,i)/den(n,j)
      gr3d=gr3d+sum
      if(n.gt.next.and.dabs(sum/(gr3d+grref)).lt.epsmod)goto 2000
 1000 continue
      write(iwrite,99998)maxmod,epsmod,i,j
 2000 gr3d=gr3d/rl
      ratio=gr3d*pi/gr
      write(iwrite,99997)ratio,i,j
      return
99999 format(' sum does not converge that rapidly, n=',i12,/
     f       ' within tolerance',d20.13)
99998 format(' mode number',i12,' is still significant'/
     f       ' within tolerance',d20.13,' for entry',2i12)
99997 format(' ratio of 3d correction on 2d term is',d20.13,/
     f       ' for entry',2i12)
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c
c     end of routine sum3d
c
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
      end
```

Original Fortran 77 by Rob Diependaal