$$\frac{\partial w}{\partial t} + \nabla(w u) = \nabla\left[\left(\nu + \frac{\nu_t}{\sigma_w}\right)\nabla w\right] + \gamma_2 P_k - \beta_2 w^2 + 2\frac{1}{\sigma_{w2}\, w}\frac{\partial k}{\partial x_k}\frac{\partial w}{\partial x_k}$$

I convection   II diffusion   III production   IV source

```
C#############################################################################
c                                                                            #
c                     SUBROUTINE PROGRAM                                     #
C                   VERSION 1.0 (16/02/2011)                                 #
C                   AUTHORIZED BY ZHANG JINGXIN                              #
C                                 SHANGHAI JIAO TONG UNIVERSITY              #
C                                 SHANGHAI, CHINA                            #
c----------------------------------------------------------------------------#
c      computes the advective,Coriolis,horizontal dispersive terms in the   #
c      momentum equation of u                                                #
c                                                                            #
c#############################################################################
      Subroutine ADVTKD(SGS,F1,S1,GRADX,GRADY,GRADZ)

      Include './Include/OCERM_INF'
      Common/VISW/VISSOURCE(IJM,KB),DISSMARK(IJM,KB)
    Parameter (TINT = 0.05, DESTYPE = 1, CW = 0.15)

    Parameter(BETASTAR=0.09,FKAPA=0.41)
    Parameter(SIGMAK1=0.85, SIGMAO1=0.5, BETA1=0.075, A1=0.31,
     &            GAMA1=BETA1/BETASTAR-SIGMAO1*FKAPA**2./Sqrt(BETASTAR))
    Parameter(SIGMAK2=1.0, SIGMAO2=0.856, BETA2=0.0828,
     &            GAMA2=BETA2/BETASTAR-SIGMAO2*FKAPA**2./Sqrt(BETASTAR))
      Parameter(CFW=3.5,AIFA=5./9.)

    Dimension HQ(IJE,KB),VQ(IJM,KB),SGS(IJM,KB),F1(IJM,KB),S1(IJM,KB)
    Dimension TEMP(KBM),VISCOE(IJM,KB),VAR_T(IJM,-1:KB+1)
    Dimension GRADX(IJM,KB),GRADY(IJM,KB),GRADZ(IJM,KB)
c===========================================================================c
c                    initialiing arrays                                     c
c===========================================================================c
!$OMP PARALLEL DEFAULT(SHARED) PRIVATE(I,K)
    Do K = 1, KB
!$OMP DO
       Do I = 1, IJM
           VISF(I,K) = 0.0
           VISCOE(I,K) = F1(I,K) * SIGMAO1 + (1. - F1(I,K)) * SIGMAO2
           VISSOURCE(I,K) = 0.0
           DISSMARK(I,K) = 1.0
       Enddo
!$OMP END DO NOWAIT
```

```fortran
!$OMP DO
        Do I = 1, IJE
           HQ(I,K) = 0.0
        Enddo
!$OMP END DO NOWAIT
!$OMP DO
        Do I = 1, IJM
           VQ(I,K) = 0.0
        Enddo
!$OMP END DO NOWAIT
     Enddo
!$OMP DO
       Do I = 1, IJM
        Do K = 1, KBM
           VAR_T(I,K) = TDISS(I,K)
        Enddo
        VAR_T(I,0) = 2. * TDISS(I,1) - TDISS(I,2)
        VAR_T(I,-1) = VAR_T(I,0)
C       VAR_T(I,KB) = TDISS(I,KBM)
        VAR_T(I,KB) =  60. * UMOL / BETA1 /
     &                  (DC(I) * DZZ(KBM)) ** 2.
        VAR_T(I,KB+1) = TDISS(I,KBM)
     Enddo
!$OMP END DO NOWAIT
!$OMP END PARALLEL
       If(IWENOSCHEME .NE. 0) Then
          Do K = 1, KBM
             Do I = -1, NUM_GHOST, -1
                QGHOST(I,K) = TDISS(INDEX_GHOST(I),K)
             Enddo
          Enddo
     Endif
C============================================================================c
c             TVD schemes for the calculation of convective fluxes           c
c============================================================================c
       Call TVDSCHEMEH(HQ,TDISS,GRADX,GRADY,IH_TVD)
       Call TVDSCHEMEV(VQ,TDISS,VAR_T,GRADZ,IV_TVD)
c============================================================================c
C                 Advection Descrization by 2nd Order TVD                    c
c============================================================================c
!$OMP PARALLEL DEFAULT(SHARED)
```

```
!$OMP&          PRIVATE(I,J,K,XX,FV1,FV2,FV3,FT2,D1,D2,HMAX,HWN,M,N,
!$OMP&          SLOWERPART1,SLOWERPART2,SLOWER,DTIDLE,STIDLE,R,DIVXVIS,
!$OMP&          DIVYVIS,DIVZVIS,AAMF,FLUX1,FLUX2,FD,RDT,FT,FL,FE2,
!$OMP&          ALF1,FE1,FE,FB,FDT,FDITDLE,G,FW,ID,IS,ZDES,D2MAX,TTTT,
!$OMP&          TW_STAR, UW_STAR, RKS_PLUS, TDISS_PLUS,YP,TDIFF_WALL)
      If (ADVECT.EQ.'NON-LINEAR') Then
c-----------------------------------------------------------------------------c
c                    horizontal advective terms                                c
c-----------------------------------------------------------------------------c
       Do K = 1, KBM
!$OMP DO
          Do I = 1, IJM
```

$I.\ \sum_i F_i \phi_i$

```
             If(CCM(I) .EQ. 1.0) Then
                Do J = 1, CELL_POLYGEN(I)
                   If(CFM(CELL_SIDE(I,J,1)) .EQ. 1.0) Then
                      VISF(I,K) = VISF(I,K) + DZ(K) *
     &                   HQ(CELL_SIDE(I,J,1),K) * CELL_CUV(I,J,6) *
     &                   (UN(CELL_SIDE(I,J,1),K) * CELL_CUV(I,J,7) +
     &                      VN(CELL_SIDE(I,J,1),K) * CELL_CUV(I,J,8))
                   Endif
                Enddo
             Endif
          Enddo
!$OMP END DO NOWAIT
       Enddo
!$OMP BARRIER
c-----------------------------------------------------------------------------c
c                    vertical advective terms                                  c
c-----------------------------------------------------------------------------c
       Do K = 1, KBM
!$OMP DO
          Do I = 1, IJM
             If(CCM(I) .EQ. 1.0) Then
                VISF(I,K) = -VISF(I,K) - AREA(I) *
     &                   (VQ(I,K) * W(I,K) - VQ(I,K+1) * W(I,K+1))
             Endif
          Enddo
!$OMP END DO NOWAIT
       Enddo
!$OMP BARRIER
    Endif
```

```
c=============================================================================c
C                    Horizontal Diffusion Descrization by CS                   c
c=============================================================================c
    Do K = 1, KBM
!$OMP DO
       Do I = 1, IJM
```

II $\sum_i D_i(\phi_c - \phi_0) + S_{D\text{-}cross}$

```
          If(CCM(I) .EQ. 1.0) Then
            Do J = 1, CELL_POLYGEN(I)
               FLUX1 = 0.0
               FLUX2 = 0.0
               If(CFM(CELL_SIDE(I,J,1)) .EQ. 1.0) Then
                  AAMF = UMOL + (AAM(I,K) + AAM(CELL_SIDE(I,J,2),K)) / 2. *
     &                     (VISCOE(I,K) + VISCOE(CELL_SIDE(I,J,2),K))/2.
                    FLUX1 = (DISCOE(I,J,1) - DISCOE(I,J,8))* AAMF *
     &                       (TDISS(CELL_SIDE(I,J,2),K) - TDISS(I,K))
                    FLUX2 = (DISCOE(I,J,7) - DISCOE(I,J,2)) * AAMF*
     &                       (TDISSV(CELL_SIDE(I,J,4),K) -
     &                        TDISSV(CELL_SIDE(I,J,3),K))
                    VISF(I,K) = VISF(I,K) + (FLUX1 + FLUX2) * DZ(K)
                Endif
                If(CFM(CELL_SIDE(I,J,1)) .EQ. 0.0 .OR.
     &               CFM(CELL_SIDE(I,J,1)) .EQ. -1.0) Then
                   If(ISLIP .EQ. 0) Then                 !Wall function
                      AAMF = UMOL + AAM(I,K) * VISCOE(I,K)
                      YP = D2D(I)
                      TDIFF_WALL = Sqrt(TKE(I,K))/Sqrt(0.3)/0.41/YP

                     FLUX1 = (DISCOE(I,J,1) - DISCOE(I,J,8)) * AAMF *
     &                    (TDIFF_WALL - TDISS(I,K))
                      VISF(I,K) = VISF(I,K) + FLUX1 * DZ(K)

C                    FLUX1 = (DISCOE(I,J,1) - DISCOE(I,J,8)) * AAMF *
C     &                   (60.*UMOL/BETA1/(AREA(I)/ACOS(-1.))-TDISS(I,K))
                      VISF(I,K) = VISF(I,K) + FLUX1 * DZ(K)
                   Endif
                Endif
                  If(CFM(CELL_SIDE(I,J,1)) .EQ. -2.0) Then
C-------------------------BC of TDISS at wall(by WangJian)------------------------
                     IF (IBC_TDISS_WALL == 1) THEN
                     TDIFF_WALL = 60.*UMOL/BETA1/(D2D(I)**2.0)
```

```
                    ELSEIF (IBC_TDISS_WALL == 2) THEN  !#Another formula#, WangJian,
2020-3-12・ !
                TW_STAR = 2.*UMOL*1000.*(CELL_CUV(I,J,8)*(STRESS(I,K,1)
     &                        +0.5*STRESS(I,K,2) + 0.5*STRESS(I,K,4)
     &                        +0.5*STRESS(I,K,3) + 0.5*STRESS(I,K,7))
     &                        -CELL_CUV(I,J,7)*(STRESS(I,K,5)
     &                        +0.5*STRESS(I,K,2) + 0.5*STRESS(I,K,4)
     &                      +0.5*STRESS(I,K,6) + 0.5*STRESS(I,K,8)))
                    UW_STAR = SQRT(ABS(TW_STAR/1000.0))
                    RKS_PLUS = MAX(1.0, RKS*UW_STAR/UMOL)
                    IF (RKS_PLUS.LT.25.0)THEN
                     TDISS_PLUS = MIN(  (50.0/RKS_PLUS)**2 ,
     &                           6.0/0.09/(D2D(I)*UW_STAR/UMOL)**2)
                    ELSE
                     TDISS_PLUS = MIN(  100.0/RKS_PLUS ,
     &                           6.0/0.09/(D2D(I)*UW_STAR/UMOL)**2)
                    ENDIF
                    TDIFF_WALL = MAX(1.0E-10,
     &                           TW_STAR*TDISS_PLUS/(UMOL*1000.))
                    ENDIF
C--------------------------BC of TDISS at wall--------------------------
                    AAMF = UMOL + AAM(I,K) * VISCOE(I,K)
                  FLUX1 = (DISCOE(I,J,1) - DISCOE(I,J,8)) * AAMF*
     &                 (TDIFF_WALL - TDISS(I,K))
                    VISF(I,K) = VISF(I,K) + FLUX1 * DZ(K)
               Endif
           Enddo

          Endif
       Enddo
!$OMP END DO NOWAIT
    Enddo
!$OMP BARRIER
C=============================================================================C
c                        source and sink terms                                c
C=============================================================================C
!$OMP DO
    Do I = 1, IJM
       If(CCM(I) .EQ. 1.0) Then
         Do K = 1, KBM
!!!##################################################################!!!
```

```fortran
!!! Feature     :The crossdiffusion is computed implicitly.
!!! UpdatedBy   :WangJian
!!! UpdatedDate:2019-12-2 21:54:22
!!!##################################################################!!!
c-----S1/VIS is relataed to the equation, good for FUN3, bad for FUN1.
c-----S1/(VIS+UMOL) is treated numerically, good for FUN1, bad for FUN3.
c           VISF(I,K) = VISF(I,K) + ((F1(I,K)*GAMA1+(1.-F1(I,K))*GAMA2)
c    &                * S1(I,K) / (VIS(I,K)+umol) + SGS(I,K))
c    &                  * AREA(I) * DZ(K)+abs(SGS(I,K))* AREA(I) * DZ(K)
c           VISSOURCE(I,K) = (F1(I,K)*BETA1+(1.-F1(I,K))*BETA2) *
c    &                    TDISS(I,K)+abs(SGS(I,K))/(TDISS(I,K)+1.E-10)
c-----!The production of w is computed implicitly----------------------
            VISF(I,K) = VISF(I,K) + 2. * SGS(I,K) * AREA(I) * DZ(K)
```

III

```fortran
            VISSOURCE(I,K) = -(F1(I,K)*GAMA1+(1.-F1(I,K))*GAMA2) *
     &                      S1(I,K) / (TKE(I,K) + 1.E-10) +
     &                      (F1(I,K)*BETA1+(1.-F1(I,K))*BETA2) *
```

IV

```fortran
     &                      TDISS(I,K)+SGS(I,K))/(TDISS(I,K)+1.E-10)
```

cross-diffusion

$$2(1-F_1)\cdot \sigma_{w2}\cdot \frac{1}{\omega}\cdot \frac{\partial k}{\partial x_j}\cdot \frac{\partial \omega}{\partial x_j}$$

```fortran
          Enddo
        Endif
      Enddo
!$OMP END DO
      If(POREMODULE .EQ. 'INCLUDE' .AND. DEM .EQ. 'NEGLECT') Then
!$OMP DO
        Do I =1,IJM
          If(CCM(I).EQ.1.0)Then
            Do K=1,KBM
              If(PORE(I,K) .NE. 1.0)Then
                VISF(I,K) =  VISF(I,K) + AREA(I) * DZ(K)*
     &              TDISS(I,K)* CFW *
     &            1. / 2. * APU(I,K) * CDC*
     &          Sqrt(UR(I,K) ** 2. + VR(I,K) ** 2. + WR(I,K) ** 2.)
!     &           (UR(I,K)** 2. + VR(I,K)**2 +WR(I,K)** 2.)

              Endif
            Enddo
          Endif
        Enddo
!$OMP END DO
      Endif

      If(DEM .NE. 'NEGLECT' .AND. I_PTF .NE. 0) Then
```

```
!$OMP DO
          Do I =1,IJM
            If(CCM(I) .EQ. 1.0)Then
              Do K=1,KBM
                 !If (PORE(I,K) .NE. 1.0) Then
c               VISF(I,K) =  VISF(I,K) + AREA(I) * DZ(K) * STDISSDEM(I,K) ! STDISSDEM
                  !Endif
              Enddo
            Endif
          Enddo
!$OMP END DO
       Endif
!$OMP BARRIER
c============================================================================c
c                    open boundary treatments                                c
c============================================================================c
!$OMP MASTER
C-----   elevation boundary condition
       If(NUMEBC .NE. 0) Then
         Do N = 1, NUMEBC
            ID = IEBC(N)
            IS = IEBCINX(N)
            Do K = 1, KBM
               VISF(ID,K) = 0.0
               UNEBC = UR(ID,K) * CELL_CUV(ID,IS,7) +
     &                 VR(ID,K) * CELL_CUV(ID,IS,8)
               Do J = 1, CELL_POLYGEN(ID)
                    If(CFM(CELL_SIDE(ID,J,1)) .EQ. 1.0) Then
                        VISF(ID,K) = VISF(ID,K) - DZ(K) *
     &                   HQ(CELL_SIDE(ID,J,1),K) * CELL_CUV(ID,J,6) *
     &                   (UN(CELL_SIDE(ID,J,1),K) * CELL_CUV(ID,J,7) +
     &                      VN(CELL_SIDE(ID,J,1),K) * CELL_CUV(ID,J,8))
                  Endif
               Enddo
               If(UNEBC .GT. 0.0) Then
                  VISF(ID,K) = VISF(ID,K) - DZ(K) * CELL_CUV(ID,IS,6)*
     &                       TDISS(ID,K) * UNEBC
               Else
                    VISF(ID,K) = VISF(ID,K) - DZ(K) * CELL_CUV(ID,IS,6)*
     &                       UNEBC * TDISSE(N)
                 Endif
```

```fortran
                VISSOURCE(ID,K) = 0.0
                DISSMARK(ID,K) = 0.0
            Enddo
         Enddo
      Endif
C-----   astrotidal boundary condition
      If(NUMAST .NE. 0) Then
        Do N = 1, NUMAST
           ID = IABC(N)
           IS = IABCINX(N)
           Do K = 1, KBM
              VISF(ID,K) = 0.0
              UNAST = UR(ID,K) * CELL_CUV(ID,IS,7) +
     &                VR(ID,K) * CELL_CUV(ID,IS,8)
              Do J = 1, CELL_POLYGEN(ID)
                 If(CFM(CELL_SIDE(ID,J,1)) .EQ. 1.0) Then
                    VISF(ID,K) = VISF(ID,K) - DZ(K) *
     &                 HQ(CELL_SIDE(ID,J,1),K) * CELL_CUV(ID,J,6) *
     &                 (UN(CELL_SIDE(ID,J,1),K) * CELL_CUV(ID,J,7) +
     &                   VN(CELL_SIDE(ID,J,1),K) * CELL_CUV(ID,J,8))
                 Endif
              Enddo
                If(UNAST .GT. 0.0) Then
                 VISF(ID,K) = VISF(ID,K) - DZ(K) * CELL_CUV(ID,IS,6)*
     &                        TDISS(ID,K) * UNAST
              Else
                   VISF(ID,K) = VISF(ID,K) - DZ(K) * CELL_CUV(ID,IS,6)*
     &                        UNAST * Sqrt(1.5) * DC(ID) * DZ(K) *
     &                          UNAST * TINT
                Endif
                VISSOURCE(ID,K) = 0.0
            Enddo
        Enddo
      Endif
c-----   discharge boundary condition
      If(NUMQBC .NE. 0) Then
         Call BCOND(3)
         Do N = 1, NUMQBC
            ID = IQBC(N)
            IS = IQBCINX(N)
            ISS = CELL_SIDE(ID,IS,1)
```

```
          Do K = 1, KBM
             VISF(ID,K) = 0.0
             UNQBC = UN(ISS,K) * CELL_CUV(ID,IS,7) +
     &                VN(ISS,K) * CELL_CUV(ID,IS,8)
              Do J = 1, CELL_POLYGEN(ID)
                  If(CFM(CELL_SIDE(ID,J,1)) .EQ. 1.0) Then
                   VISF(ID,K) = VISF(ID,K) - DZ(K) *
     &                  HQ(CELL_SIDE(ID,J,1),K) * CELL_CUV(ID,J,6) *
     &                 (UN(CELL_SIDE(ID,J,1),K) * CELL_CUV(ID,J,7) +
     &                   VN(CELL_SIDE(ID,J,1),K) * CELL_CUV(ID,J,8))
                Endif
             Enddo
             If(UNQBC .GT. 0.0) Then
               VISF(ID,K) = VISF(ID,K) - DZ(K) * CELL_CUV(ID,IS,6)*
     &                         TDISS(ID,K) * UNQBC
             Else
               VISF(ID,K) = VISF(ID,K) - DZ(K) * CELL_CUV(ID,IS,6)*
     &                        UNQBC * TDISSQ(N,K)
C            PRINT*, TDISSQ(N,K),UNQBC
C     &              Sqrt(1.5) * Abs(UNQBC) * TINT /
C     &              (DC(ID) * (1. + ZZ(K))) / Sqrt(0.3)
               Endif
               VISSOURCE(ID,K) = 0.0
               DISSMARK(ID,K) = 0.0
          Enddo
        Enddo
      Endif
C-----   velocity boundary condition
      If(NUMVBC .NE. 0) Then
        Do N = 1, NUMVBC
          ID = IVBC(N)
          IS = IVBCINX(N)
          Do K = 1, KBM
             VISF(ID,K) = 0.0
             UNVBC = UN(CELL_SIDE(ID,IS,1),K) * CELL_CUV(ID,IS,7) +
     &                VN(CELL_SIDE(ID,IS,1),K) * CELL_CUV(ID,IS,8)
               Do J = 1, CELL_POLYGEN(ID)
                  If(CFM(CELL_SIDE(ID,J,1)) .EQ. 1.0) Then
                     VISF(ID,K) = VISF(ID,K) - DZ(K) *
     &                  HQ(CELL_SIDE(ID,J,1),K) * CELL_CUV(ID,J,6) *
     &                 (UN(CELL_SIDE(ID,J,1),K) * CELL_CUV(ID,J,7) +
```

```
     &                        VN(CELL_SIDE(ID,J,1),K) * CELL_CUV(ID,J,8))
                 Endif
              Enddo
              If(UNVBC .GT. 0.0) Then
                VISF(ID,K) = VISF(ID,K) - DZ(K) * CELL_CUV(ID,IS,6)*
     &                         TDISS(ID,K) * UNVBC
              Else
                VISF(ID,K) = VISF(ID,K) - DZ(K) * CELL_CUV(ID,IS,6)*
     &                           UNVBC *
     &                  Sqrt(1.5) * Abs(UNVBC) * TINT /
     &                  (DC(ID) * (1. + ZZ(K))) / Sqrt(0.3)

                Endif
                VISSOURCE(ID,K) = 0.0
           Enddo
        Enddo
     Endif
!$OMP END MASTER
!$OMP BARRIER
C=============================================================================C
C                     Step forward in time                                    C
C=============================================================================C
C      PRINT*, (TDISS(33634,K),K=1,KBM)
C      STOP

    Do K = 1, KBM
!$OMP DO
       Do I = 1, IJM
          If(CCM(I) .EQ. 1.0) Then
             VISF(I,K) = TDISS(I,K) * AREA(I) * DZ(K) + DTI * VISF(I,K)
          Endif
        Enddo
!$OMP END DO
    Enddo
!$OMP END PARALLEL
c======================  end subroutine program  ===========================C
      Return
      End
```