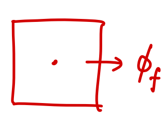
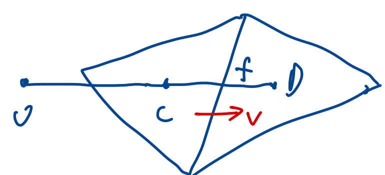
```
C###########################################################################
c                                                                          #
c                     SUBROUTINE PROGRAM                                   #
C                   VERSION 1.0 (30/04/2009)                               #
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 ADVU
      Include './Include/OCERM_INF'
      Include './Include/VORGEN_INF'

    Parameter(ZOWALL=1.E-5,I_METHOD = 1)
    Dimension HQ(IJE,KB),VQ(IJM,KB),UFHYD(IJM)
    Dimension VAR_EDGE(IJE,KB),VAR_T(IJM,-1:KB+1),GRADX(IJM,KB),
     &           GRADY(IJM,KB),GRADZ(IJM,KB),PNT(IJM,KB)
      Dimension FXE(IJM),FXH(IJM), TEMP(KB+1)
      Dimension REY_STRESS_UU(NUM_CELL,NUM_VER),
     &           REY_STRESS_UV(NUM_CELL,NUM_VER),
     &           REY_STRESS_UW(NUM_CELL,NUM_VER)
        Dimension FLU_SOURCE(NUM_CELL,NUM_VER)
C      Dimension QGHOST(NUM_GHOST:-1,KB)
c==========================================================================c
c                    initialiing arrays                                    c
c==========================================================================c
!$OMP PARALLEL DEFAULT(SHARED) PRIVATE(I,K,NUM_GHOST)
    Do K = 1, KB
!$OMP DO
       Do I = 1, IJM
            UF(I,K) = 0.0      ⇒ source term (对流&扩散)
c            PNT(I,K) = 0.0
       Enddo
!$OMP END DO
!$OMP DO
       Do I = 1, IJE
          HQ(I,K) = 0.0    ϕf (边通量)
       Enddo
```



```
!$OMP END DO
!$OMP DO
       Do I = 1, IJM
          VQ(I,K) = 0.0
       Enddo
!$OMP END DO
    Enddo
!$OMP END PARALLEL
      If(IWENOSCHEME .NE. 0) Then
        Do K = 1, KBM
           Do I = -1, NUM_GHOST, -1
              QGHOST(I,K) = U(INDEX_GHOST(I),K)
           Enddo
        Enddo
    Endif
C=============================================================================c
c             TVD schemes for the calculation of convective fluxes           c
c=============================================================================c
C---- Variables on the midpoint of the cell surfaces
!$OMP PARALLEL DEFAULT(SHARED) PRIVATE(I,J,K,IL,IR,UW)
!$OMP DO
      Do I = 1, IJM
       Do K = 1, KBM
          VAR_T(I,K) = U(I,K)
       Enddo
       VAR_T(I,0) = 2. * U(I,1) - U(I,2)
       VAR_T(I,-1) = VAR_T(I,0)
       VAR_T(I,KB) = 0.0
       VAR_T(I,KB+1) = 0.0
    Enddo
!$OMP END DO
!$OMP DO
      Do I = 1, IJM
         If(CCM(I) .EQ. 1.0) Then
            Do K = 1, KBM
               GRADX(I,K) = 0.0
               GRADY(I,K) = 0.0
               Do J = 1, CELL_POLYGEN(I)
                  If(CFM(CELL_SIDE(I,J,1)) .EQ. 1.0) Then
                  UW = (U(I,K) + U(CELL_SIDE(I,J,2),K)) / 2.
                  GRADX(I,K) = GRADX(I,K) + UW *
```

$\nabla_x \phi_c$



$$r = \frac{\phi_c - \phi_U}{\phi_D - \phi_c} = \frac{(\phi_D - \phi_U) - (\phi_D - \phi_c)}{\phi_D - \phi_c}$$

$$\phi_D - \phi_U = \nabla\phi_c \cdot 2 r_{cD}$$

$$\int_{CV} \nabla\phi_c \, dv = \int_{CS} \vec{n}\phi_c \, ds$$

$$\Rightarrow \sum \nabla\phi_c \, \Delta V = \sum \phi_{cf} n \, ds$$

$$\Rightarrow \nabla\phi_c \, A \cdot \Delta\sigma = \sum \phi_{cf} \Delta l \, \vec{n_i} \, \Delta\sigma$$

$$\Rightarrow \nabla\phi_c = \frac{\sum \phi_{cf} \Delta l \, \vec{n_i}}{A}$$

```
     &                          CELL_CUV(I,J,7) * CELL_CUV(I,J,6)
                   GRADY(I,K) = GRADY(I,K) + UW *
     &                          CELL_CUV(I,J,8) * CELL_CUV(I,J,6)
                   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. 1) Then
```

$(U\sin\alpha - V\cos\alpha)\sin\alpha$

```
                         UW = (U(I,K)*CELL_CUV(I,J,8) -
     &                         V(I,K)*CELL_CUV(I,J,7)) * CELL_CUV(I,J,8)
                         GRADX(I,K) = GRADX(I,K) + UW *
     &                                CELL_CUV(I,J,7) * CELL_CUV(I,J,6)
                        GRADY(I,K) = GRADY(I,K) + UW *
     &                                CELL_CUV(I,J,8) * CELL_CUV(I,J,6)
                      Endif
                   Endif
                   If(CFM(CELL_SIDE(I,J,1)) .EQ. -3.0) Then
                      UW = (U(I,K)*CELL_CUV(I,J,8) -
     &                      V(I,K)*CELL_CUV(I,J,7)) * CELL_CUV(I,J,8)
                      GRADX(I,K) = GRADX(I,K) + UW *
     &                             CELL_CUV(I,J,7) * CELL_CUV(I,J,6)
                   GRADY(I,K) = GRADY(I,K) + UW *
     &                             CELL_CUV(I,J,8) * CELL_CUV(I,J,6)
                   Endif
                Enddo
                GRADX(I,K) = GRADX(I,K) / AREA(I)
```

$\nabla_x \phi_c = \frac{U_w \Delta l \cos\alpha}{A}$

```
                GRADY(I,K) = GRADY(I,K) / AREA(I)
             Enddo
          Endif
       Enddo
!$OMP END DO
!$OMP DO
       Do I = 1, IJM
```

$\frac{\partial u}{\partial z} = \frac{U_{k-1} - U_k}{(h+\eta)\Delta\sigma}$

```
          If(CCM(I) .EQ. 1.0) Then
             Do K = 2, KBM
                GRADZ(I,K) = (U(I,K-1) - U(I,K)) / DC(I) / DZZ(K-1)
             Enddo
             GRADZ(I,1) = 0.0
             GRADZ(I,KB) = U(I,KBM) / (0.5 * DZ(KBM) * DC(I))
          Endif
       Enddo
!$OMP END DO
```

```
c!$OMP DO
c   Do I = 1, IJM
c      If(CCM(I) .EQ. 1.0) Then
c         Do K = 2, KBM
c            VAR_UD(I,K) = (U(I,K-1) + U(I,K)) / 2.0
c         Enddo
c         If(KB .GE. 3) Then
c            VAR_UD(I,1) = 2. * U(I,1) - U(I,2)
c            VAR_UD(I,KB) = 0.0
c         Else
c            VAR_UD(I,1) = U(I,1)
c         Endif
c      Endif
c   Enddo
c!$OMP END DO
!$OMP END PARALLEL
ccc Call GRAD_XY(VAR_EDGE,GRADX,GRADY)
ccc Call GRAD_Z(VAR_UD,GRADZ)
    Call TVDSCHEMEH(HQ,U,GRADX,GRADY,IH_TVD)
```

$\Sigma \phi_f$

```
      Call TVDSCHEMEV(VQ,U,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,AAMF,FLUX1,FLUX2,FLUX3,D2,
!$OMP&                  CDWALL,UFHYD,UFDYN,TEMP,ID,CROSS_LENGTH)

      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
```

$$\frac{\sum F_i \phi_i}{\ }$$

$F_i = \rho u \Delta A_i$

```
             If(CCM(I) .EQ. 1.0) Then
                Do J = 1, CELL_POLYGEN(I)
                   If(CFM(CELL_SIDE(I,J,1)) .EQ. 1.0) Then
                      UF(I,K) = UF(I,K) +
     &                    HQ(CELL_SIDE(I,J,1),K) * CELL_CUV(I,J,6) *
```

$\rho(u\cos\alpha + v\sin\alpha)\,\Delta A_i \cdot \phi_i$

```
     &                      PORE_HF(CELL_SIDE(I,J,1),K) *
     &                    (UN(CELL_SIDE(I,J,1),K) * CELL_CUV(I,J,7) +
```

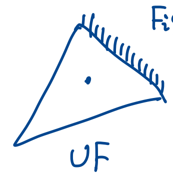
```
     &                       VN(CELL_SIDE(I,J,1),K) * CELL_CUV(I,J,8))
                       Endif
                    Enddo
                 Endif
                 UF(I,K) = UF(I,K) * DZ(K)
```

$\Delta l \cdot \Delta\sigma = \Delta A_i$

```
              Enddo
!$OMP END DO NOWAIT
           Enddo
!$OMP BARRIER
c==============================================================================c
c                         open boundary treatments                             c
c==============================================================================c
C-----   elevation boundary condition
!$OMP MASTER
          If(NUMEBC .NE. 0) Then
            Do N = 1, NUMEBC
               ID = IEBC(N)
               IS = IEBCINX(N)
               Do K = 1, KBM
```

$u\cos\alpha + v\sin\alpha = u$

$F_i\phi_i = \rho u \Delta A_i\, \phi_i$

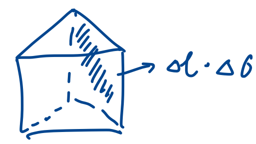



```
                  UNEBC = UR(ID,K) * CELL_CUV(ID,IS,7) +
     &                    VR(ID,K) * CELL_CUV(ID,IS,8)
                  If(UNEBC .LE. 0.0) Then
                     UF(ID,K) = 0.0
                  Else
```

ΔA
$\Uparrow$
$\Delta\sigma\Delta l\, u\cdot\phi$

```
                     UF(ID,K) = UF(ID,K) + DZ(K) * CELL_CUV(ID,IS,6) *
     &                          U(ID,K) * UNEBC
                  Endif
c                 UF(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
                  UNEBC = UR(ID,K) * CELL_CUV(ID,IS,7) +
     &                    VR(ID,K) * CELL_CUV(ID,IS,8)
                  If(UNEBC .LE. 0.0) Then
                     UF(ID,K) = 0.0
```

```
            Else
               UF(ID,K) = UF(ID,K) + DZ(K) * CELL_CUV(ID,IS,6) *
     &                        U(ID,K) * UNEBC
            Endif
          Enddo
        Enddo
      Endif
c-----   discharge boundary condition
      If(NUMQBC .NE. 0) Then
        Do N = 1, NUMQBC
          ID = IQBC(N)
          IS = IQBCINX(N)
          Do K = 1, KBM
            UF(ID,K) = UF(ID,K) + DZ(K) * DS(CELL_SIDE(ID,IS,1)) *
     &                 UN(CELL_SIDE(ID,IS,1),K) * CELL_CUV(ID,IS,6) *
     &                 (UN(CELL_SIDE(ID,IS,1),K) * CELL_CUV(ID,IS,7) +
     &                   VN(CELL_SIDE(ID,IS,1),K) * CELL_CUV(ID,IS,8))
          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
            UF(ID,K) = UF(ID,K) + DZ(K) * DS(CELL_SIDE(ID,IS,1)) *
     &                 UN(CELL_SIDE(ID,IS,1),K) * CELL_CUV(ID,IS,6) *
     &                 (UN(CELL_SIDE(ID,IS,1),K) * CELL_CUV(ID,IS,7) +
     &                   VN(CELL_SIDE(ID,IS,1),K) * CELL_CUV(ID,IS,8))
          Enddo
        Enddo
      Endif
!$OMP END MASTER
!$OMP BARRIER
C!$OMP FLUSH(UF)
c-----------------------------------------------------------------------------c
c                    vertical advective terms                                  c
c-----------------------------------------------------------------------------c
      Do K = 1, KBM
!$OMP DO
```



$\delta l \cdot \Delta\sigma$

$F = \rho U \Delta A_i$

ϕ_i

$(H+\eta)\ \Delta\sigma\Delta l \cdot u_n(U\cos\alpha + V\sin\alpha)$

$-\Delta A[w_k\phi_i - w_{k+1}\phi_i]$

```
           Do I = 1, IJM
              If(CCM(I) .EQ. 1.0) Then
                 UF(I,K) = -UF(I,K) - AREA(I) *
     &                        (VQ(I,K) * W(I,K) * PORE_VF(I,K) -
     &                         VQ(I,K+1) * W(I,K+1) * PORE_VF(I,K+1))
              Endif
           Enddo
!$OMP END DO
        Enddo
!$OMP BARRIER
     Endif
c==============================================================================c
C                    Horizontal Diffusion Descrization by CS                   c
c==============================================================================c
     Do K = 1, KBM
!$OMP DO
        Do I = 1, IJM
           If(CCM(I) .EQ. 1.0) Then
             Do J = 1, CELL_POLYGEN(I)
                FLUX1 = 0.0
                FLUX2 = 0.0
c                 wet cell side
                If(CFM(CELL_SIDE(I,J,1)) .EQ. 1.0) Then
```

$\sum D_i(\phi_c-\phi_D)+S_{D\text{-}cross}$

$D_i=\frac{\Gamma}{\Delta\xi}\frac{n\cdot n}{n\cdot e_\xi}\Delta A_i$

$S_{D\text{-}cross} = -\Gamma\frac{e_\xi-e_\eta}{n\cdot e_\xi}\Delta A_i\frac{\phi_b-\phi_a}{\Delta\eta}$

ν_t

Γ μ

```
                  AAMF = (AAM(I,K) + AAM(CELL_SIDE(I,J,2),K)) / 2. +UMOL

                  FLUX1 = (DISCOE(I,J,1) - DISCOE(I,J,8))* AAMF *
     &                    (U(CELL_SIDE(I,J,2),K) - U(I,K)) *
     &                     PORE_HF(CELL_SIDE(I,J,1),K)
```

$D_i(\phi_c-\phi_D)$ $\phi_c-\phi_D$

```
                  FLUX2 = (DISCOE(I,J,7) - DISCOE(I,J,2)) * AAMF*
     &                      (UV(CELL_SIDE(I,J,4),K) -
     &                       UV(CELL_SIDE(I,J,3),K)) *
     &                       PORE_HF(CELL_SIDE(I,J,1),K)
```

$S_{D\text{-}cross}$ $\phi_a-\phi_b \rightarrow S_{D\text{-}cross}$

```
                  UF(I,K) = UF(I,K) + (FLUX1 + FLUX2) * DZ(K)
                Endif
```

$\Delta l \times \Delta\sigma = \Delta A_i$

```
c               dry side or solid boundary
                  If(CFM(CELL_SIDE(I,J,1)) .EQ. 0.0 .OR.
     &               CFM(CELL_SIDE(I,J,1)) .EQ. -1.0) Then
c-----               refering to the rough side wall
```

```
                  AAMF = AAM(I,K) + UMOL

                  FLUX1 = -U(I,K) * DISCOE(I,J,1) * AAMF * PORE(I,K)

c                 FLUX2 = -V(I,K) * DISCOE(I,J,5) * AAMF

                  If(Abs(CELL_CUV(I,J,8)) .LE. 1.E-6) Then
                     FLUX3 = 0.0
                  Else
                       If(ISLIP .EQ. 1) Then
                        FLUX3 = 0.0
                     Else
c-----   distance to the wall
C                       D2 = Abs((PXY(CELL_SIDE(I,J,3),1) - CXY(I,1))*
C     &                             (PXY(CELL_SIDE(I,J,4),2) - CXY(I,2))-
C     &                             (PXY(CELL_SIDE(I,J,4),1) - CXY(I,1))*
C     &                             (PXY(CELL_SIDE(I,J,3),2)-CXY(I,2)))/
C     &                         CELL_CUV(I,J,6)
                         D2 = D2D(I)
c-----   drag coefficient
C                          CDWALL = 0.4 ** 2. / (Dlog(D2 / Z0WALL)) ** 2.
                         If(VERTMIX .EQ. 'SSTMODEL  ') Then
                            ZSTAR = Dmax1(Sqrt(0.3)*
     *                               Sqrt(TKE(I,K)) * D2 / 1.E-6,15.0)
                            CDWALL = 0.41 * Sqrt(0.3)*
     *                               Sqrt(TKE(I,K))/Log(9.81*ZSTAR)
                         Endif
                         FLUX3 = -DISCOE(I,J,8) * D2 * CDWALL * U(I,K) *
     *                            PORE(I,K)
C                      FLUX3 = -DISCOE(I,J,8) * D2 / CELL_CUV(I,J,8)*
C     &                            CDWALL * Abs(U(I,K)*CELL_CUV(I,J,8))*
C     &                            U(I,K) * CELL_CUV(I,J,8) / DC(I)
                     Endif
                  Endif
                    UF(I,K) = UF(I,K) + (FLUX1 - FLUX3) * DZ(K)
               Endif
c              non-slip boundary
                 If(CFM(CELL_SIDE(I,J,1)) .EQ. -2.0) Then
                    FLUX1 = -(UMOL + AAM(I,K)) * (DISCOE(I,J,1)-DISCOE(I,J,8)) *
     &                      U(I,K) * PORE(I,K)
                    UF(I,K) = UF(I,K) + FLUX1 * DZ(K)
```

```
                 Endif
c------          slip wall
                 If(CFM(CELL_SIDE(I,J,1)) .EQ. -3.0) Then
                    AAMF = AAM(I,K) + UMOL
                  FLUX1 = -U(I,K) * DISCOE(I,J,1) * AAMF * PORE(I,K)
                    UF(I,K) = UF(I,K) + FLUX1 * DZ(K)
                 Endif

            Enddo
          Endif
       Enddo
!$OMP END DO NOWAIT
    Enddo
!$OMP BARRIER
c============================================================================c
c                          source terms treatments                           c
c============================================================================c
C                    Coriolis Terms and POREMODULE
c
c----------------------------------------------------------------------------c
    If(CORLIS .EQ. 'INCLUDE') Then
!$OMP DO
      Do I = 1, IJM
         If(CCM(I) .EQ. 1.0) Then
           Do K = 1, KBM
              UF(I,K) = UF(I,K) + PORE(I,K) * COR(I) * V(I,K) *
     &                        AREA(I) * DZ(K)
           Enddo
         Endif
      Enddo
!$OMP END DO
    Endif

      If(POREMODULE .EQ. 'INCLUDE') Then
        If(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
                      UF(I,K) = UF(I,K) - U(I,K) * AREA(I) * DZ(K) *
```

```fortran
     &                     SQRT(UR(I,K)**2.+VR(I,K)**2.+WR(I,K)**2.)*
     &                     1. / 2. * APU(I,K) * CDC
                  Endif
              Enddo
              Endif
           Enddo
!$OMP END DO
           Endif
       Endif
c-----------------------------------------------------------------------------c
C                     DEMFORCE                                                c
c-----------------------------------------------------------------------------c
         If(DEM .NE. 'NEGLECT' .AND. I_PTF .NE. 0) Then
!$OMP DO
           Do I = 1 , IJM
              If( CCM(I) .EQ. 1.0) Then
                 Do K = 1 ,KBM
                    UF(I,K) = UF(I,K) + DEMFORCEX(I,K) * AREA(I) * DZ(K)
                 Enddo
              Endif
           Enddo
!$OMP END DO
         Endif
c-----------------------------------------------------------------------------c
C                     Atmaspher Pressure and wind stress                      c
c-----------------------------------------------------------------------------c
!$OMP DO
       Do I = 1, IJM
          If(CCM(I) .EQ. 1.0) Then
               UF(I,1) = UF(I,1) +WUSURF(I) / RMEAN(I,1)
          Endif
       Enddo
!$OMP END DO
c-----------------------------------------------------------------------------c
C    Explicited part of Dynamic pressure and Hydrostatic pressrure            C
C-----------------------------------------------------------------------------C
!$OMP DO
       Do I = 1, IJM
          If(CCM(I) . EQ. 1.0) Then
             UFHYD(I) = 0.0
             If(I_METHOD .EQ. 1) Then
```

```
                 Do J = 1, CELL_POLYGEN(I)
                If(CFM(CELL_SIDE(I,J,1)) .EQ. 1.0) Then
                   UFHYD(I) = UFHYD(I) + CELL_CUV(I,J,6) *
     &                        (ELF(CELL_SIDE(I,J,2)) + ELF(I)) / 2. *
     &                        CELL_CUV(I,J,7)
                 Else
                   UFHYD(I) = UFHYD(I) + CELL_CUV(I,J,6) *
     &                        ELF(I) * CELL_CUV(I,J,7)
                 Endif
             Enddo
             Endif
           If(I_METHOD .EQ. 2) Then
                 Do J = 1, CELL_POLYGEN(I)
                  If(CFM(CELL_SIDE(I,J,1)) .EQ. 1.0) Then
                   UFHYD(I) = UFHYD(I) +
     &                        WIX(I,J) * (ELF(CELL_SIDE(I,J,2)) - ELF(I))
                 Endif
               Enddo
                 UFHYD(I) = UFHYD(I) * AREA(I)
             Endif
c          UFHYD(I) = UFHYD(I) * AREA(I)
C          Do K = 1, KBM
CC            UF(I,K) = UF(I,K) - GRAV * DC(I) * (1.0 - THITA) * DZ(K)*
CC     &                        UFHYD
CC         Enddo
        Endif
     Enddo
!$OMP END DO
!$OMP MASTER
     If(NUMEBC .NE. 0) Then
          Do N = 1, NUMEBC
           ID = IEBC(N)
           UFHYD(ID) = 0.0
           Do J = 1, CELL_POLYGEN(ID)
              If(CFM(CELL_SIDE(ID,J,1)) .EQ. 1.0) Then
                 UFHYD(ID) = UFHYD(ID) +
     &                        WIX(ID,J) * (ELF(CELL_SIDE(ID,J,2)) - ELF(ID))
              Endif
           Enddo
           UFHYD(ID) = UFHYD(ID) * AREA(ID)
        Enddo
```

$\int_{cv} \nabla\xi \, dv = \int_{cs} \vec{n}\xi \, ds$

$\Rightarrow \nabla\xi \, \Delta A \, \Delta\sigma = \xi_f \, \vec{n} \, \Delta l \, \Delta\sigma$

$\Rightarrow \nabla_x\xi \, \Delta A \, \Delta\sigma = \underbrace{\xi_f \, \Delta l \cos\alpha \, \Delta\sigma}_{UFHYD}$

Δl

ξ_f

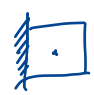
$\cos\alpha$

Here is $\int_{cv} \nabla_x \xi \, dv = \nabla_x \xi_i \, \Delta V$

$= \frac{\Delta\xi}{\Delta x} A \, \Delta\sigma$

```
     Endif
C-----   astrotidal boundary condition
     If(NUMAST .NE. 0) Then
        Do N = 1, NUMAST
           ID = IABC(N)
           UFHYD(ID) = 0.0
           Do J = 1, CELL_POLYGEN(ID)
              If(CFM(CELL_SIDE(ID,J,1)) .EQ. 1.0) Then
                 UFHYD(ID) = UFHYD(ID) +
     &                    WIX(ID,J) * (ELF(CELL_SIDE(ID,J,2)) - ELF(ID))
              Endif
           Enddo
           UFHYD(ID) = UFHYD(ID) * AREA(ID)
        Enddo
     Endif
c-----   discharge boundary condition
     If(NUMQBC .NE. 0) Then
        Do N = 1, NUMQBC
```

$$\frac{\partial \zeta}{\partial x} = \frac{\Delta \zeta}{\Delta x}$$



```
           ID = IQBC(N)
           UFHYD(ID) = 0.0
           Do J = 1, CELL_POLYGEN(ID)
              If(CFM(CELL_SIDE(ID,J,1)) .EQ. 1.0) Then
                 UFHYD(ID) = UFHYD(ID) +
     &                  WIX(ID,J) * (ELF(CELL_SIDE(ID,J,2)) - ELF(ID))
              Endif
           Enddo
           UFHYD(ID) = UFHYD(ID) * AREA(ID)
        Enddo
     Endif
C-----   velocity boundary condition
     If(NUMVBC .NE. 0) Then
        Do N = 1, NUMVBC
           ID = IVBC(N)
           UFHYD(ID) = 0.0
           Do J = 1, CELL_POLYGEN(ID)
             If(CFM(CELL_SIDE(ID,J,1)) .EQ. 1.0) Then
                UFHYD(ID) = UFHYD(ID) +
     &                    WIX(ID,J) * (ELF(CELL_SIDE(ID,J,2)) - ELF(ID))
             Endif
           Enddo
           UFHYD(ID) = UFHYD(ID) * AREA(ID)
```

```
          Enddo
       Endif
!$OMP END MASTER
!$OMP BARRIER
!$OMP DO
        Do I = 1, IJM
         If(CCM(I) . EQ. 1.0) Then
           Do K = 1, KBM
```

$$\int_{cv} gD(1-\theta)\left(\frac{\partial \zeta^n}{\partial x}\right)_i dv$$

```
             UF(I,K) = UF(I,K) - GRAV * DC(I) * (1.0 - THITA) * DZ(K)*
     &                       UFHYD(I) * PORE(I,K)
           Enddo
         Endif
        Enddo
```

$$\int_{cv}\left(\frac{\partial \zeta^n}{\partial x}\right)_i dv$$

```
!$OMP END DO
C---------------------------------------------------------------------------C
c                     wave radiation stress                                 C
C---------------------------------------------------------------------------C
      If(WAVEDYN.NE.'NEGLECT') Then

      Endif
C---------------------------------------------------------------------------C
C                    Baroclinic pressure term                               C
C---------------------------------------------------------------------------C
        If(TOR.EQ.'BAROCLINIC') Then

      Endif
C---------------------------------------------------------------------------C
C                    Synthetize eddy in ZDES model                          c
C---------------------------------------------------------------------------C
!$OMP BARRIER
C!$OMP MASTER
        If(DES .EQ. 'SAZDES ') Then
!$OMP DO
           Do I = 1, NUM_CELL
              ID = ID_CELL(I)
              Do K = 1, NUM_VER
                 REY_STRESS_UU(I,K) = -UDIS(I,K) * UDIS(I,K)
     &                                -UR(ID,K)  * UDIS(I,K)
     &                                -UR(ID,K)  * UDIS(I,K)
                 REY_STRESS_UV(I,K) = -UDIS(I,K) * VDIS(I,K)
     &                                -UR(ID,K)  * VDIS(I,K)
```

```fortran
     &                              -VR(ID,K)   * UDIS(I,K)
                REY_STRESS_UW(I,K) = -UDIS(I,K) * WDIS(I,K)
     &                              -UR(ID,K)   * WDIS(I,K)
     &                              -WR(ID,K)   * UDIS(I,K)
             Enddo
             CROSS_LENGTH = Sqrt(AREA(ID))
             Do K = 1, NUM_VER - 1
cc               FLU_SOURCE(I,K) = DZ(K) * CROSS_LENGTH * 2. *
cc     &              (UDIS(I,K) + VDIS(I,K)) +
cc     &               AREA(ID) / DC(ID) * WDIS(I,K)
                 FLU_SOURCE(I,K) = DZ(K) * CROSS_LENGTH * DC(ID) *
     &                       (REY_STRESS_UU(I,K) + REY_STRESS_UV(I,K))+
     &                        AREA(ID) * REY_STRESS_UW(I,K)
             Enddo
             Do K = 1, NUM_VER - 1
C               UF(ID,K) = UF(ID,K) + U(ID,K) * FLU_SOURCE(I,K)
                UF(ID,K) = UF(ID,K) + FLU_SOURCE(I,K)
             Enddo
          Enddo
!$OMP END DO
       Endif
C!$OMP END MASTER
!$OMP BARRIER
C============================================================================C
C                     Step forward in time                                   C
C============================================================================C
     Do K = 1, KBM
!$OMP DO
        Do I = 1, IJM
           If(CCM(I) .EQ. 1.0) Then
             UF(I,K) = U(I,K) * AREA(I) * DZ(K) * PORE(I,K) + DTI * UF(I,K)
           Endif
        Enddo
!$OMP END DO
     Enddo
!$OMP END PARALLEL
C============================================================================C
C              Sponge layer setting for numerical wave flume                 C
C============================================================================C
     If(IFLUME .EQ. 1) Then
        XL = -1.0
```

$$\int_{cv}\frac{q_i^* - q_i^n}{\Delta t} = \int_{cv} Fq_i^n - gD(1-\theta)\left(\frac{\partial\zeta^n}{\partial x}\right)_i - gD\theta\left(\frac{\partial\zeta^*}{\partial x}\right)_i + \frac{\nu_t}{D^2}\frac{q_{i+1}^* - 2q_i^* + q_{i-1}^*}{(\Delta\sigma)^2} + IBF_i^n$$

$$\int_{cv}\frac{q_{xi}^* - q_{xi}^n}{\Delta t}\,dv = \frac{q_{xi}^* - q_{xi}^n}{\Delta t}\Delta V$$

$$= (q_{xi}^* - q_{xi}^n) A\Delta\sigma \cdot \frac{1}{\cancel{\Delta t}}$$

$$= q_{xi}^* \cancel{A}\Delta\sigma - q_{xi}^n A\Delta\sigma$$

$$\int_{cv} D\,dv = D\cancel{A}\Delta\sigma$$

$$\int_{cv} F_{2B,i}^n\,dv = F_{2B,i}^n A\Delta\sigma$$

```fortran
        XR = 26.0
          XLL = -5.
          XRR = 30.0
        Do I = 1, IJM
           If(CXY(I,1) .LE. XL) Then
              Do K = 1, KBM
                 DIFFCOE = 10.0 * (XL - CXY(I,1)) / 4.
                 UF(I,K) = UF(I,K) -
     &                    DTI * DIFFCOE * U(I,K) * DZ(K) * AREA(I)
              Enddo
           Endif
           If(CXY(I,1) .GE. XR) Then
              Do K = 1, KBM
                 DIFFCOE = 8.0 * (CXY(I,1) - XR) / (30.0 - XR)
                 UF(I,K) = UF(I,K) -
     &                    DTI * DIFFCOE * U(I,K) * DZ(K) * AREA(I)
              Enddo
           Endif
        Enddo
       Endif

c============================================================================c
c                        immersed boundary force                             c
c============================================================================c
       If ((IBMETHOD .NE. 'NEGLECT') .AND. (IBFSCHEME.EQ.2)) Then
       Do K = 1, KBM
           Do I = 1, IJM
               UF(I,K) = UF(I,K) + BFX(I,K) * DTI * DZ(K) * AREA(I)
           Enddo
       Enddo
       Endif

c=====================  end subroutine program  ===========================C
       Return
       End
```