How is gravity Implemented in SubDyn

TestSubDyn()

InitInData%g = 9.80665

[p%FX, p%Fbar\_21 , p%FY] = SubDyn\_Init(Init%g)

Init%FG = AssembleKM(Init%g)

FGe = ElemG(g)

F(3) = -0.5\*L\*rho\*A\*g

F(9) = F(3)

TempCoeff = -1.0/12.0\*g\*L\*L\*rho\*A

F(4) = TempCoeff\*( DirCos(1, 3)\*DirCos(2, 1) - DirCos(1, 1)\*DirCos(2, 3) )

F(5) = TempCoeff\*( DirCos(1, 3)\*DirCos(2, 2) - DirCos(1, 2)\*DirCos(2, 3) )

F(10) = -F(4)

F(11) = -F(5)

! assemble element matrices to global matrices

DO J = 1, NNE

jn = nn(j)

Init%FG( (jn\*6-5):(jn\*6) ) = Init%FG( (jn\*6-5):(jn\*6) ) &

+ FGe( (J\*6-5):(J\*6) )

! add concentrated mass induced gravity force

DO I = 1, Init%NCMass

r = ( Init%CMass(I, 1) - 1 )\*6 + 3

Init%FG(r) = Init%FG(r) - Init%CMass(I, 2)\*Init%g

[p%FX, p%Fbar\_21 , p%FY] = Craig\_Bampton(Init%FG)

[FGR, FGL] = BreakSysMtrx(Init%FG)

DO I = 1, DOFR

II = IDR(I)

FGR(I) = Init%FG(II)

DO I = 1, DOFL

II = IDL(I)

FGL(I) = Init%FG(II)

FGRb = CBApplyConstr(FGR)

[p%FX, p%Fbar\_21 , p%FY] = SetParameters( FGRb, FGL)

p%FY = MATMUL( MBMt, MATMUL( TRANSPOSE(PhiM), FGL) ) &

- MATMUL( TRANSPOSE(TI), ( FGRb + MATMUL( TRANSPOSE(PhiRb), FGL) ) )

! Fbar\_21

p%Fbar\_21 = MATMUL( PhiM, MATMUL( TRANSPOSE(PhiM), FGL) )

! FX

p%FX = MATMUL( TRANSPOSE(PhiM), FGL )

SDOut\_Init()

ElemG()

SubDyn\_CalcContStateDeriv()

dxdt%qmdot= matmul(p%A\_21,x%qm) + matmul(p%A\_22,x%qmdot)+ matmul(p%B\_23,u%UFL(L2:L3-1)) +   
 matmul(p%B\_24,u%UFL(L3:L4)) + p%FX

SubDyn\_UpdateStates()

!Assign the acceleration to the x variable since it will be used for output file purposes for SSqmdd01-99, and dxdt will disappear

x%qmdotdot=dxdt%qmdot

SubDyn\_CalcOutput()

! Y1 = TP reaction Forces

y%Y1= matmul(p%C1\_11,x%qm) + matmul(p%C1\_12,x%qmdot) + matmul(p%D1\_11,u%UFL(1:p%TPdofL)) + &

matmul(p%D1\_13,u%UFL(L2:L3-1))+ matmul(p%D1\_14,u%UFL(L3:p%uL)) + p%FY

!Calculate accelerations even though they may not be necessary, in the future we may put

! a condition on the type of input to speed up, if forces are requested we need them

y%Udotdot(p%URbarL+1:p%URbarL+p%DOFL)= matmul(p%Abar\_21,x%qm) + matmul(p%Abar\_22,x%qmdot) + &

matmul(p%Bbar\_23,u%UFL(L2:L3-1)) + matmul(p%Bbar\_24,u%UFL(L3:p%uL)) + &

p%Fbar\_21

SDOut\_MapOutputs()

! Sum element-level loads, FM\_elm and FK\_elm into y%FM\_elm and y%FK\_elm for outputs ????

CALL CALC\_LOCAL( DIRCOS,p%MOutLst(I)%Me(:,:,J,1),p%MOutLst(I)%Ke(:,:,J,1),(/uddout( L : L+5 ),uddout( L2 : L2+5 )/), &

(/yout( L : L+5 ), yout( L2 : L2+5 )/),p%MoutLst(I)%Fg(:,J,1), K2,FM\_elm,FK\_elm)

FK\_elm = CALC\_LOCAL()

! TODO: Need to look at whether the following gravity component should be +Fg or -Fg. Since the calculation is in global coordinates, gravity

! acts in negative Z, but if the user has code uses gravity as -9.8 then this would already be negative. If the code uses gravity as positive

! then you would need to apply -Fg.

! As a note, HydroDyn expects the gravity variable to be >= 0. So we would use -Fg in that case.

Junk1=matmul(Ke,Y2) - Fg !GLOBAL REFERENCE

DO L=1,4

Junk4((L-1)\*3+1:L\*3) = matmul(DIRCOS, Junk1( (L-1)\*3+1:L\*3 ) )

ENDDO

FK\_elm=Junk4(6\*(k2-1)+1:k2\*6)

IF (p%OutAll) THEN !NEED TO CALCULATE TOTAL FORCES

CALL CALC\_LOCAL( DIRCOS,p%MOutLst2(I)%Me2(:,:,J),p%MOutLst2(I)%Ke2(:,:,J),(/uddout( L : L+5 ),uddout( L2 : L2+5 )/), &

(/yout( L : L+5 ), yout( L2 : L2+5 )/), p%MoutLst2(I)%Fg2(:,J), K2,FM\_elm,FK\_elm)

AllOuts( L:L2 ) =sgn\* (/FK\_elm,FM\_elm/)

ENDIF

Allouts(SSqmdd01:SSqmdd01+p%Nmodes-1)=x%qmdotdot

!Need to Calculate Reaction Forces Now, but only if requested

CALL CALC\_LOCAL( DIRCOS,p%MOutLst3(I)%Me(:,:,1,J),p%MOutLst3(I)%Ke(:,:,1,J),(/uddout( L : L+5 ),uddout( L2 : L2+5 )/), &

(/yout( L : L+5 ), yout( L2 : L2+5 )/), p%MoutLst3(I)%Fg(:,1,J), K2,FM\_elm,FK\_elm)

!transform back to global, need to do 3 at a time since K is 3x3

DO L=1,2

!sum forces at joint in GLOBAL REF

y%FK\_elm((L-1)\*3+1:L\*3) = y%FK\_elm((L-1)\*3+1:L\*3) + matmul(p%elemprops(K)%DirCos,FK\_elm((L-1)\*3+1:L\*3))

!signs may be wrong, we will fix that later

ENDDO

ENDDO

ReactNs((I-1)\*6+1:6\*I)= y%FK\_elm + y%FM\_elm !Accumulate reactions from all nodes in GLOBAL COORDINATES

AllOuts( ReactSS(1:P%TPdofL) ) = matmul(p%TIreact,ReactNs)