APPENDIX A. DIRECT SHEAR STRESS RESULTS CURVES [SHAHROUR AND REZAIE, 1997]

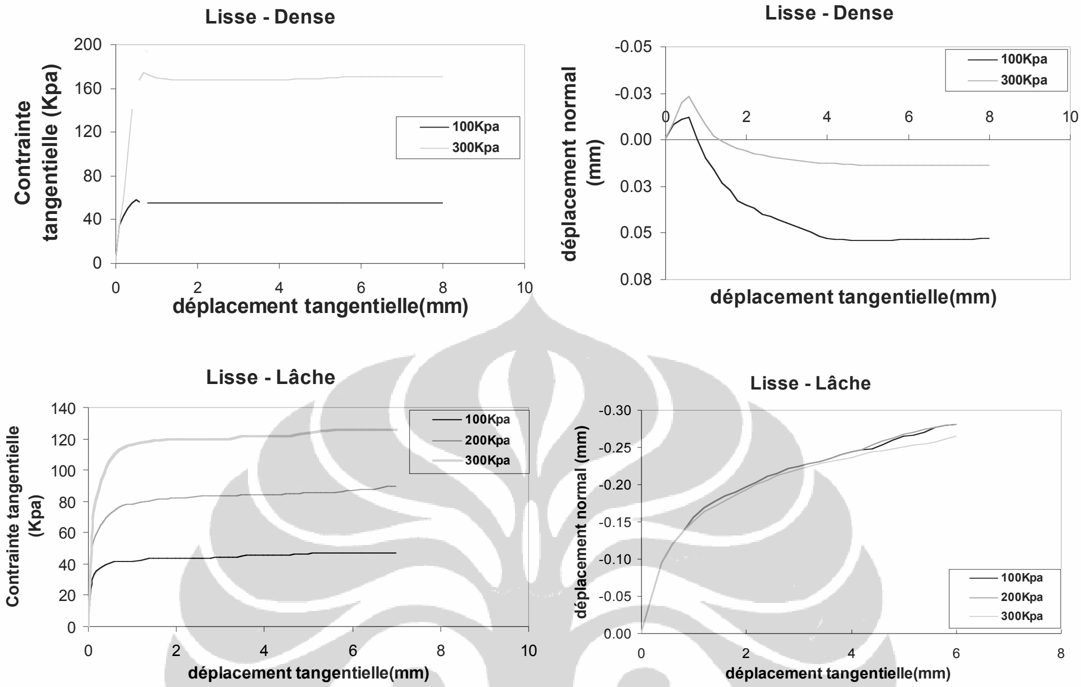

Figure 71. Results curves for a smooth interface [Shahrour and Rezaie, 1997]

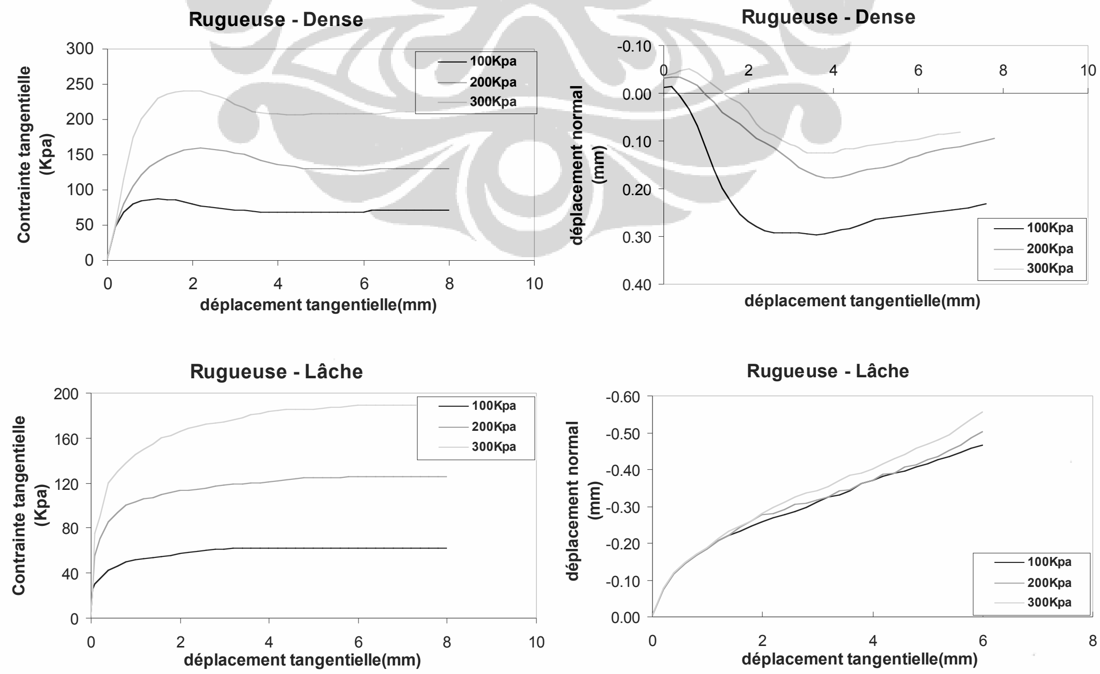

Figure 72. Results curves for a rough interface [Shahrour and Rezaie, 1997]

APPENDIX B. SHEAR STRESS CURVEC FOR A MONOTONIC SOLICITATION (WITH THE FINAL PARAMETERS)

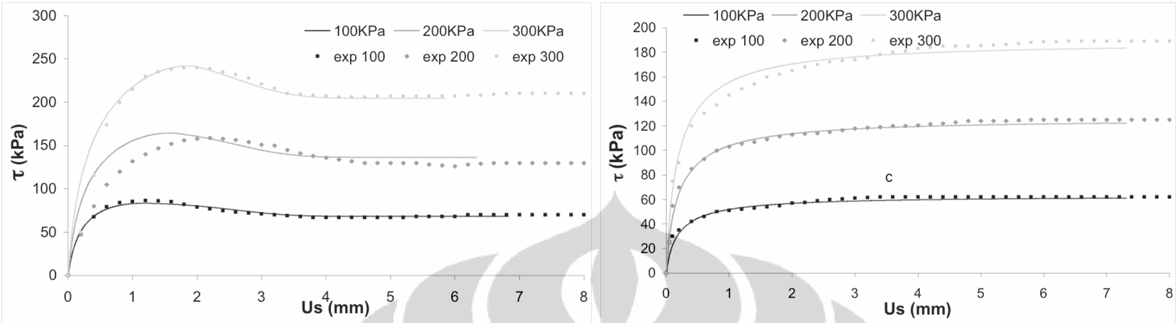

Figure 73. Shear stress curve for rough interface with : a) dense sand (*ID*=90%) ; b) loose sand (*ID*=15%)

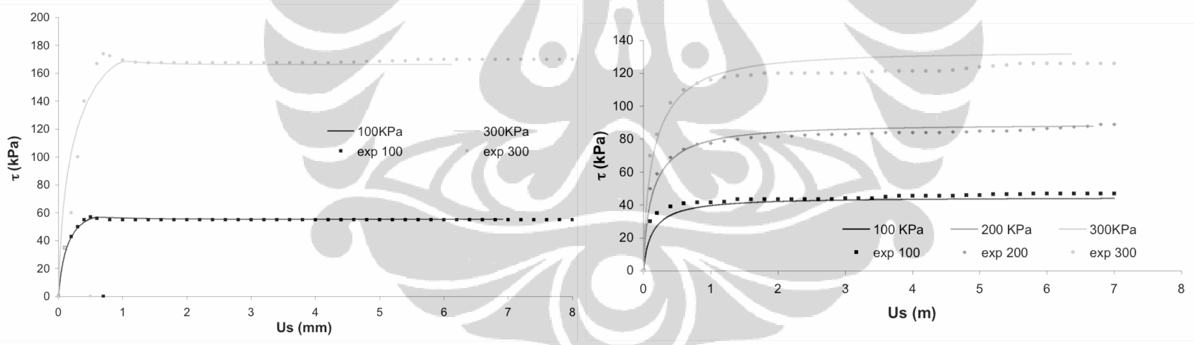

Figure 74. Shear stress curve for smooth interface with: a) dense sand (*ID*=90%) ; b) loose sand (*ID*=15%)

APPENDIX C. RESULTS OF MONOTONIC SIMULATION FOR A ROUGH INTERFACE WITH DENSE SAND FOR DIFFERENTS VALUES OF ϕ_{car}

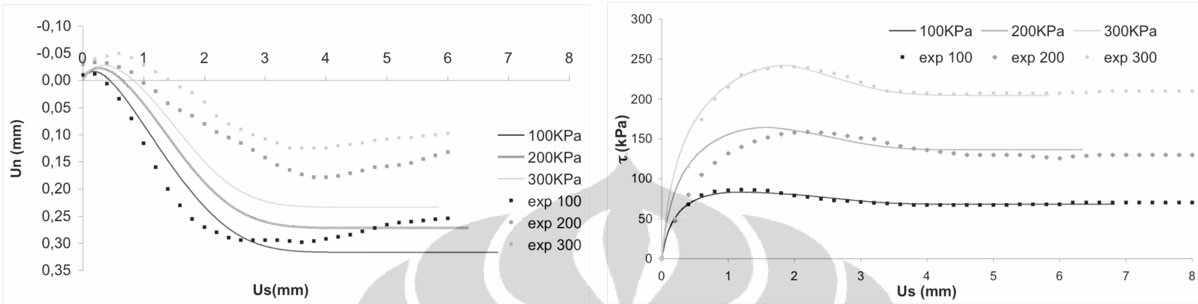

Figure 75. Displacement curve and shear stress curve of a CNL simulation for a rough interface with dense sand (ϕ_{car}= 26°)

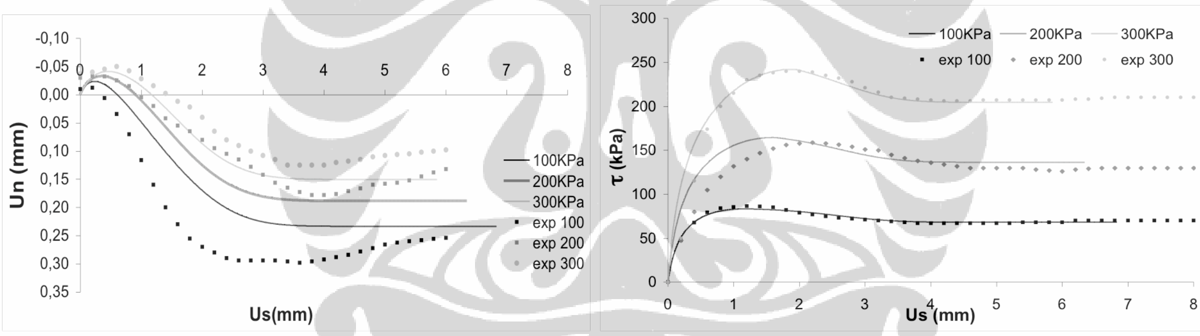

Figure 76. Displacement curve and shear stress curve of a CNL simulation for a rough interface with dense sand (ϕ_{car}= 29°)

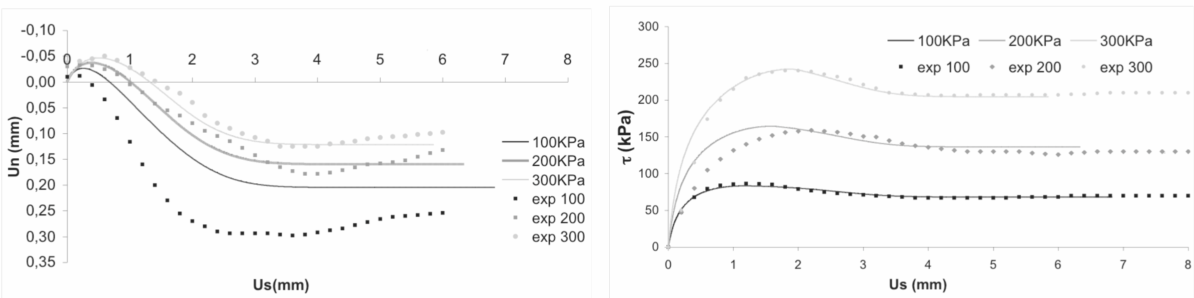

Figure 77. Displacement curve and shear stress curve of a CNL simulation for a rough interface with dense sand (ϕ_{car}= 30°)

APPENDIX D. RESULTS OF A MONOTONIC SIMULATION FOR ROUGH INTERFACE WITH DENSE SAND FOR DIFFERENTS VALUES OF PARAMETRE *n*

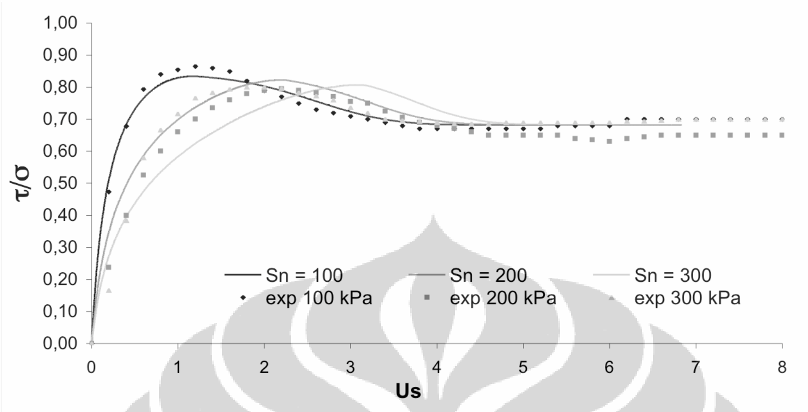

Figure 78. Stress ratio of a CNL simulation for a rough interface rough with dense sand (*n* =0,65)

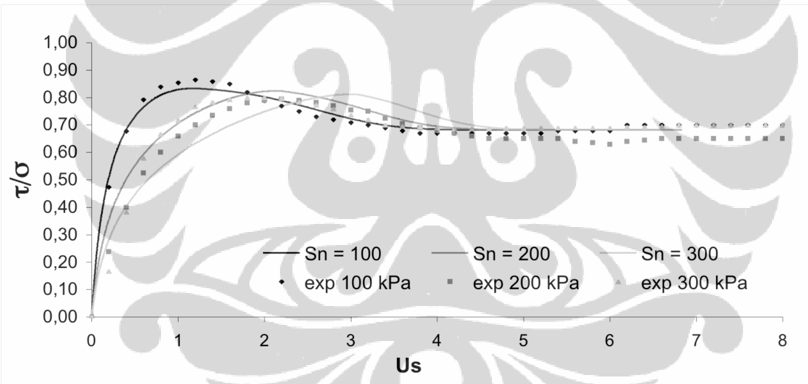

Figure 79. Stress ratio of a CNL simulation for a rough interface rough with dense sand (*n* =0,8)

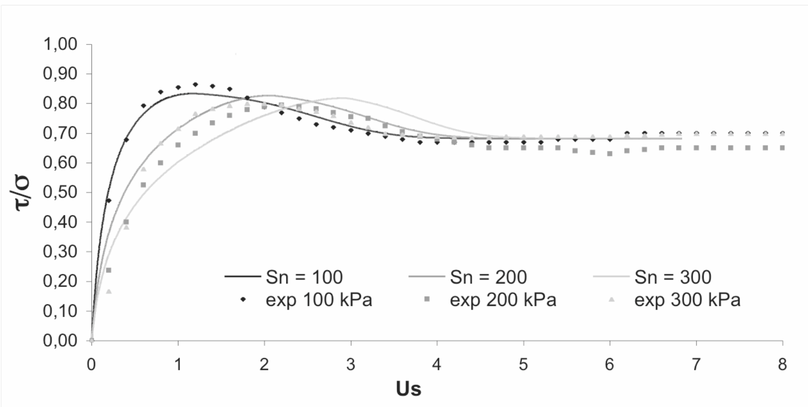

Figure 80. Stress ratio of a CNL simulation for a rough interface rough with dense sand (*n* =1)

APPENDIX E. RESULTS OF A MONOTONIC SIMULATION FOR A SMOOTH INTERFACE WITH DENSE SAND FOR ϕ_{car} = 29°

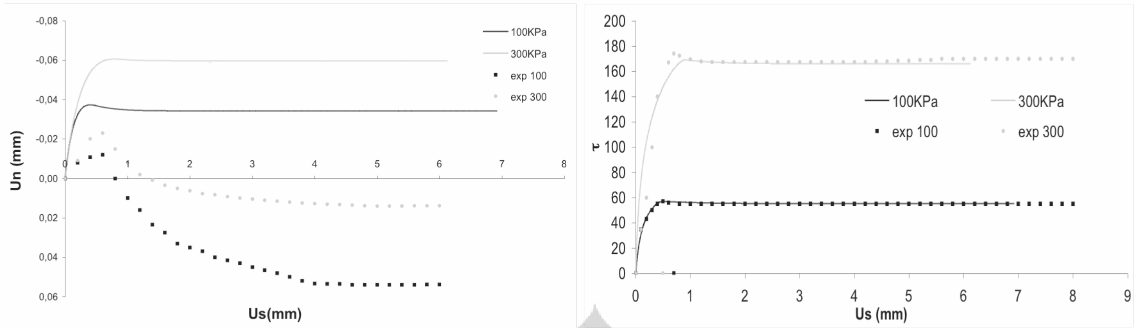

Figure 81. Displacement curve and shear stress curve of a monotonic simulation for a interface with dense sand (ϕ_{car} =29°)

APPENDIX F. THE USED BYRNE FONCTION FOR THE CYCLIC SIMULATION

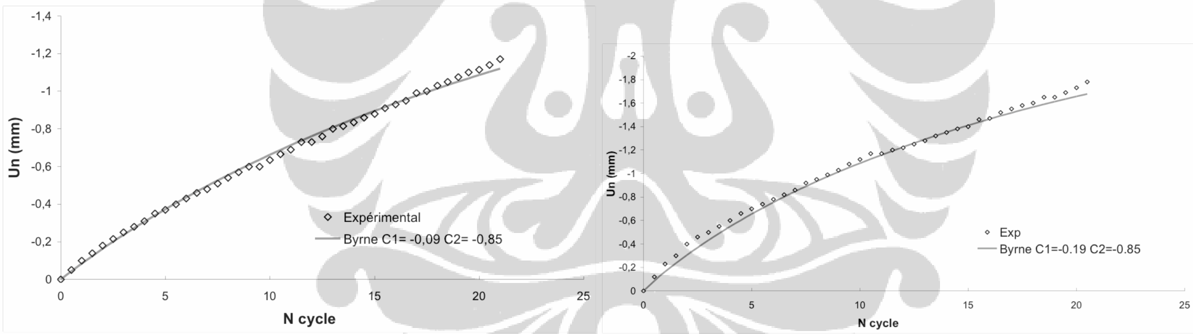

Figure 82. Byrne fonction for cyclic solicitation for rough interfaces with: a) dense sand (*ID*=90%); b) loose sand (*ID*=15%)

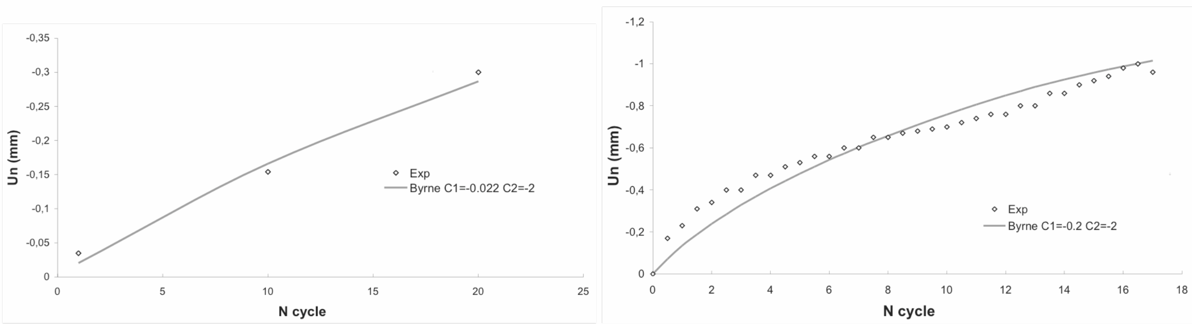

Figure 83. Byrne fonction for cyclic solicitation for smooth interfaces with: a) dense sand (*ID*=90%); b) loose sand (*ID*=15%)

APPENDIX G. DIRECT SHEAR TEST ON FLAC

```
;-----------------------------------
; Modelisation d'interface elastic non linear avec le critere de mohr coulomb
; Type d'essai : Sollicitation monotone CNL
; 01/2009
; Editeur: Irma Almanya
;-----------------------------------
config
;--------------------------------------------
; Generation de grid et definition de materiaux
;--------------------------------------------
grid 30,30
gen (0.0,0.0) (0.0,0.3) (0.3,0.3) (0.3,0.0) ratio 1.0,1.0 i=1,31 j=1,31
model elastic
group 'Steel:Hard Steel' notnull
model elastic notnull group 'Steel:Hard Steel'
prop density=7000.0 bulk=1.67E11 shear=7.69E10 notnull group 'Steel:Hard Steel'
model null j 6
group 'null' j 6
group delete 'null'
model null i 1 5 j 7 30
group 'null' i 1 5 j 7 30
group delete 'null'
model null i 26 30 j 7 30
group 'null' i 26 30 j 7 30
group delete 'null'
model null i 6 25 j 9 30
group 'null' i 6 25 j 9 30
group delete 'null'
; Initialise un petit collision entre les materiaux
ini x add .0107 j 7 9
ini y add -.0107 j 7 9
; Definir les properties de sols
group 'User:silica sand' i 6 25 j 7 8
model elastic group 'User:silica sand'
prop density=1300.0 bulk=1.87E7 shear=8.6E6 group 'User:silica sand'
;------------------------------------------
; Definition d'interface
; parametre interface pour le "smooth dense"
;------------------------------------------
interface 1 aside from 1,6 to 31,6 bside from 6,7 to 26,7
interface 1 kn=5.0E8 ks=4.0e8 cohesion=0.0 dil=0.0 friction = 30
;----------------------
; Conditions aux limites
;----------------------
fix x y j 1
fix x i 31 j 2 6
fix x i 1 j 2 6
fix x i 6 j 7 14
fix x i 26 j 7 14
fix y i 6 26 j 11 14
;----------------
; pression initiale
```

```
;-----------------
apply pressure 200000.0 from 6,9 to 26,9
hist unbal
solve
call int.fin
call app.fin
;-------------------------------------------------------------------
; Initiation de déplacement dû à la confinement initial
;-------------------------------------------------------------------
def ini_jdisp
        njdisp0 = 0.0
        sjdisp0 = 0.0
        pnt = int_pnt
        loop while pnt # 0
                pa = imem(pnt+$kicapt)
                loop while pa # 0
                        sjdisp0 = sjdisp0 + abs(fmem(pa+$kidasd))
                        njdisp0 = njdisp0 + fmem(pa+$kidand)
                        pa = imem(pa)
                end_loop
                pa = imem(pnt+$kicbpt)
                loop while pa # 0
                        sjdisp0 = sjdisp0 + abs(fmem(pa+$kidasd))
                        njdisp0 = njdisp0 + fmem(pa+$kidand)
                        pa = imem(pa)
                end_loop
                pnt = imem(pnt)
        end_loop
end
ini_jdisp
;-------------------------------------------------------------------------------
; Fonction pour stocker la pression & la rigidité tangentielle initiale
;-------------------------------------------------------------------------------
def cher_inint
        pnt = app_pnt
        nstav_init = -1 * fmem(pnt+$kapv4)
        pnt = int_pnt
        ks_init = fmem(pnt+$kicks)
end
cher_init
;---------------------------------------
; function pour stocker phi_max et phi_c
;---------------------------------------
def ini_fric
        pnt = int_pnt
        phi_m = fmem(pnt+$kicfri)
        phi_mrad = phi_m * degrad

        phi_c = 29
        phi_crad = phi_c*degrad
end
ini_fric
;-------------------------------------------------------------------------------------
; Fonction d'évolution de phi_max afin de modéliser le radoucissement
;-------------------------------------------------------------------------------------
```

```
def phi_f
        Radou = (1-(2.71828^(-2.5*(sstrain_p^4))))
        Mper = tan(phi_mrad) + (Radou *( tan(phi_crad) - tan(phi_mrad)))
        phi_frad = atan(Mper)
        phi_f = phi_frad /degrad
end
def chang
        pnt=int_pnt
        fmem(pnt+$kicfri)=phi_f
end
;----------------------------------------------------------------------
; Fonction qui calcule le moyen de contrainte de l'interface
;----------------------------------------------------------------------
def av_str
        whilestepping
        sstav = 0.0
        nstav =0.0
        njdisp = 0.0
        sjdisp = 0.0
        sjdisp_p = 0.0
        sjdisp_e = 0.0
        ncon = 0.0
        jlen = 0.0
        pnt = int_pnt
        ;-----------------------------------------
        ; Injection de la dégradation de rigidité
        ;-----------------------------------------
        ks = ks_init*((1.0-ttyield)^2)*(2.71828^(-0.01*ttyield))
        fmem(pnt+$kicks)=ks
        ;-----------------------------------------
        ; Calculation de njdisp_irreversible
        mcar = 22 * degrad
        njdispirr = incsjdisp*0.7*(ratstav-(tan(mcar)))
        ;-----------------------------------------
        loop while pnt # 0
                pa = imem(pnt+$kicapt)
                loop while pa # 0
                        sstav = sstav + fmem(pa+$kidfs)
                        nstav = nstav + fmem(pa+$kidfn)
                        jlen = jlen + fmem(pa+$kidlen)
                        sjdisp = sjdisp + abs(fmem(pa+$kidasd))
                        njdisp = njdisp + fmem(pa+$kidand)
                        ; -- Injection de la partie irréversible  --
                        fmem(pa+$kidand)=fmem(pa+$kidand)+njdispirr
                        pa = imem(pa)
                end_loop
                pa = imem(pnt+$kicbpt)
                loop while pa # 0
                        ncon = ncon + 1
                        sstav = sstav + fmem(pa+$kidfs)
                        nstav = nstav + fmem(pa+$kidfn)
                        jlen = jlen + fmem(pa+$kidlen)
                        sjdisp = sjdisp + abs(fmem(pa+$kidasd))
                        njdisp = njdisp + fmem(pa+$kidand)
```

```
                    ; Injection de la partie irréversible  --
                    fmem(pa+$kidand)=fmem(pa+$kidand)+njdispirr
                    pa = imem(pa)
                end_loop
                pnt = imem(pnt)
        end_loop
        if ncon # 0
                sstav = sstav / jlen
                nstav = nstav / jlen
                sjdisp = (sjdisp-sjdisp0) / (2.0 * ncon)
                njdisp = (njdisp-njdisp0) / (2.0 * ncon)
                sstrain_e = sstav / ks
                sstrain = sjdisp / (10*0.35e-3)
                sstrain_p = sstrain - sstrain_e
        endif
        ttyield = sstav/(nstav*tan(phi_mrad))
        ratstav = sstav/nstav
        mrat = (atan(ratstav))/degrad
        incsjdisp = abs(sjdisp-as)
        as = sjdisp
end
av_str
;
hist sstav nstav sjdisp njdisp
ini xvel 5e-8 i= 1,31 j 1,6
fix x i= 1,31 j 1,6
hist nstep 100
ini xdis 0.0 ydis 0.0
def main
        fois = 1500
        loop while fois # 0
                chang
                fois = fois - 1
                command
                        step 100
                end_command
        end_loop
end
main
ret
```

APPENDIX H. FLAC FOLDER (.FIN)

INT.FIN

```
set echo off
def $int_fin
;Interface parameters:  include-file for FISH program
;Global pointer to list of control blocks:  INT_PNT
;Block sizes
  $nwinco       = 16   ; Control block (one for each interface)
  $nwindi       = 21   ; Interface node block
;Control block
;-------------
;                0     Link to next control block
  $kicext       = 1    ; Spare extension (can be used by FISH)
  $kicapt       = 2    ; Pointer to list of "A-side" nodes
  $kicbpt       = 3    ; Pointer to list of "B-side" nodes
  $kicatp       = 4    ; Type of A-side contact:  0 = grid; 1 = beam
  $kicbtp       = 5    ; Type of B-side contact:  0 = grid; 1 = beam
  $kicfri       = 6    ; Friction angle in degrees
  $kiccoh       = 7    ; Cohesion (stress units)
  $kicbon       = 8    ; Tensile bond strength
  $kicglu       = 9    ; Bit flags, as follows:
                         ;  bit 1 - 1 if glued; 0 if not
                         ;  bit 2 - 1 if slip allowed while bonded; 0 if not
  $kicid        = 10   ; ID number
  $kicks        = 11   ; Shear stiffness (stress/disp)
  $kickn        = 12   ; Normal stiffness
  $kictph       = 13   ; Tan(friction angle)
  $kicdil       = 14   ; Dilation angle
  $kicsbon      = 15   ; Ratio of shear bond strength to tensile bond strength
;Node block
;----------
;               0     Link to next node block
  $kidext       = 1    ; Spare extension (can be used by FISH)
;                        ---- grid connection -----     ---- beam connection -----
  $kidi         = 2    ; I index of associated g.p.     ID of structural node
  $kidj         = 3    ; J index of associated g.p.     = 0 for beam connection
  $kidadd       = 4    ; Address of g.p.              Address of structural node
  $kidfn        = 5    ; Normal force
  $kidfs        = 6    ; Shear force
  $kidun        = 7    ; Unit normal vector (2 words)
;               = 8      "
  $kidslf       = 9    ; Slip flag:  1 = slipping;  0 = not
  $kidseg       = 10   ; Pointer to nearest opposing node block
  $kidrlk       = 11   ; Reverse node pointer ("down," ir material is to right)
  $kidbfl       = 12   ; Bond flag: 1 set if tension bond unbroken;  0 if broken
  $kidrat       = 13   ; Ratio of contact position:
;             = 1.0 ...  exactly at position of opposing node
;             < 0.0 ...  to left of opposing node (material below)
;             > 0.0 ...  to right   "       "     "       "
  $kidlen       = 14   ; Effective length of contact
```

```
  $kidsxx        = 15   ; Initial xx-stress
  $kidsxy        = 16   ; Initial xy-stress
  $kidsyy        = 17   ; Initial yy-stress
  $kidx          = 18   ; X coordinate; for plotting purposes only - updated
;                         infrequently
  $kidy          = 19   ; Y coordinate; -ditto-
  $kidass        = 20   ; Accumulated shear slip (used for dilation).
  $kidasd        = 21   ; Accumulated relative shear displacement (marker only).
  $kidand        = 22   ; Accumulated relative normal displacement (marker only).
end
$int_fin
set echo on
```