3 Thick-Walled Cylinder Subject to Internal Pressure

3.1 Problem Statement

This problem involves the plane strain elastic-plastic analysis of a thick-walled cylinder subjected to an internal pressure (see Figure 3.1) and is adapted from Wart et al. 1984*. The material is homogeneous and isotropic, and material failure is defined by the Tresca yield criterion.

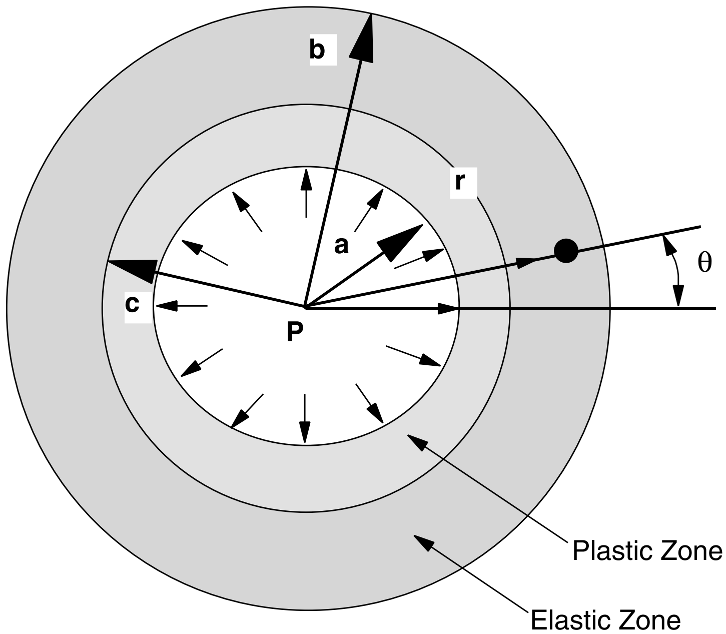

Figure 3.1 Elastic-plastic analysis of a thick-walled cylinder

The objective of this problem is to test the *UDEC* solution process for elastic-plastic material behavior against an analytic solution. This problem has finite boundaries, and thus the accuracy of the analysis should depend only on the fineness of the user-defined mesh. Specific aspects of the code tested by this problem include

(1) application of pressure boundary conditions; and

(2) computation of plastic stresses and deformations.

---

* This section was prepared for the U.S. Nuclear Regulatory Commission under U.S. NRC Contract No. 02-85-002.

3.2 Analytical Solution

The problem is axisymmetric (i.e., independent of θ) and, based on dimensional analysis, the solution (e.g., stresses and displacements) can be written in the form

$$\frac{\sigma_\alpha}{k_T} = \sigma_\alpha^* \left(\frac{r}{b};\ \frac{a}{b},\ \frac{P}{k_T} \right), \quad \alpha = r, \theta \tag{3.1}$$

$$\frac{u}{b} \frac{E}{k_T} = U^* \left(\frac{r}{b};\ \frac{a}{b},\ \frac{P}{k_T},\ \nu \right) \tag{3.2}$$

where r = radial coordinate;
σ_r, σ_θ = stresses;
u = radial displacement;
P = internal pressure;
a, b = inside and outside radii, respectively;
ν = Poisson's ratio;
E = modulus of elasticity; and
k_T = shear strength of Tresca material.

The analytical solution is given by Ford and Alexander (1977). The location of the elastic-plastic interface c/b can be obtained from the solution of the nonlinear equation

$$\frac{P}{k_T} = 2 \left(\ln \frac{c}{b} - \ln \frac{a}{b} \right) + \left(1 - \frac{c^2}{b^2} \right) \tag{3.3}$$

Therefore, there is no plastic deformation when

$$\frac{P}{k_T} < 1 - \frac{a^2}{b^2} \tag{3.4}$$

while the whole cylinder is yielding, and never reaches equilibrium when

$$\frac{P}{k_T} > \ln \frac{b^2}{a^2} \tag{3.5}$$

Prior to initial yield, the stresses and displacements are

$$\frac{\sigma_r}{k_T} = -\frac{P}{k_T}\,\frac{[(b/r)^2 - 1]}{(b/a)^2 - 1} \tag{3.6}$$

$$\frac{\sigma_\theta}{k_T} = -\frac{P}{k_T}\,\frac{[(b/r)^2 + 1]}{(b/a)^2 - 1} \tag{3.7}$$

$$\frac{u}{b}\frac{E}{k_T} = \frac{P}{k_T}\,\frac{(1+\nu)}{[(b/a)^2 - 1]}\left[\,(1-2\nu) + \frac{b^2}{r^2}\right] \tag{3.8}$$

After initial yielding, the stresses in the elastic and plastic zones are

Plastic Zone (r < c)   Elastic Zone (r > c)

$$\frac{\sigma_r}{k_T} = 2\left(\ln\frac{r}{b} - \ln\frac{c}{b}\right) - \left(1 - \frac{c^2}{b^2}\right) \qquad \frac{\sigma_r}{k_T} = -\frac{c^2}{b^2}\left(\frac{b^2}{r^2} - 1\right) \tag{3.9}$$

$$\frac{\sigma_\theta}{k_T} = 2\left(\ln\frac{r}{b} - \ln\frac{c}{b}\right) + \left(1 + \frac{c^2}{b^2}\right) \qquad \frac{\sigma_\theta}{k_T} = \frac{c^2}{b^2}\left(\frac{b^2}{r^2} + 1\right) \tag{3.10}$$

Within the elastic zone, the displacements are given by

$$\frac{u}{b}\,\frac{E}{k_T} = (1+\nu)\,\frac{c^2}{b^2}\left[\,(1-2\nu) + \frac{b^2}{r^2}\,\right] \tag{3.11}$$

3.3 *UDEC* Model

The problem has been formulated in *UDEC* using *FISH* to allow calculation of σ_α^* and U^* as functions of independent variable r/b and the problem parameters a/b, P/k_T and ν. One simulation yields the solution corresponding to a single point in the parameter space of a/b, P/k_T and ν. In this verification test, the problem has been solved for

$$a/b = 0.65$$
$$P/k_T = 0.65,\ 0.70 \text{ and } 0.75$$
$$\nu = 0.20$$

Since *UDEC* is formulated in terms of dimensional variables, the scaling parameters used in generation of the particular *UDEC* model are

$$b = 6.0 \text{ m}$$
$$k_T = 170.0 \text{ MPa}$$
$$E/k_T = 250.0$$

The computer model consists of one-quarter of the cylinder, with symmetry conditions imposed on the horizontal and vertical surfaces. The quarter-cylinder was divided along the line $\theta = 45°$ by a "glued" discontinuity. The two blocks were discretized into grids with 4, 6 and 8 zones per thickness of the wall. Figure 3.2 shows the resulting discretization for 4 zones per thickness of the wall.

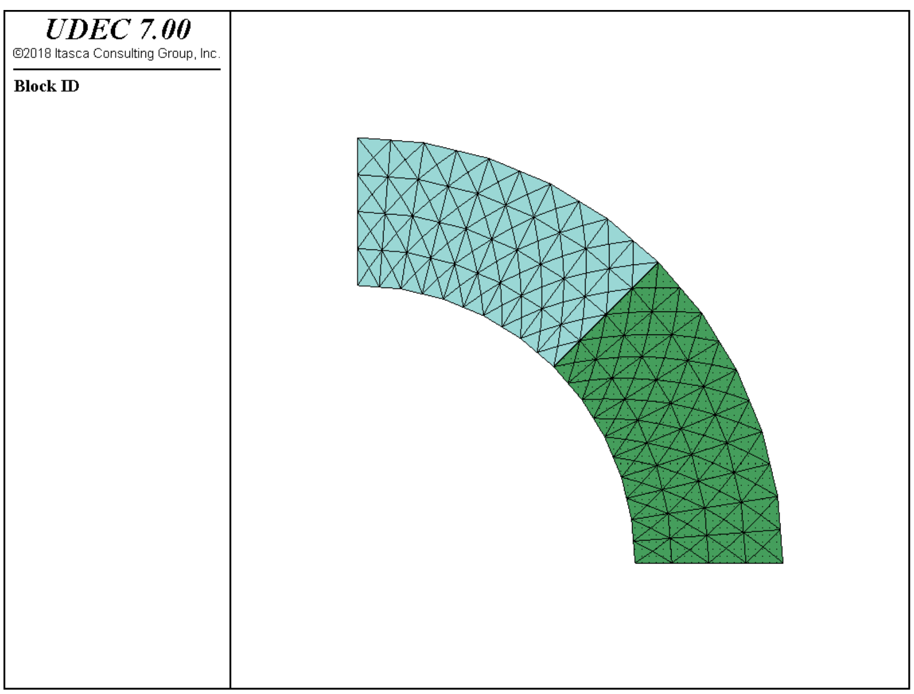

Figure 3.2 Discretization of thick-walled cylinder into constant strain finite difference triangles

The data files for this model are listed in Examples 3.1, 3.2 and 3.3. “CYL_IN.DAT” in Example 3.1 defines the input parameters for a series of 9 runs. “CONT_CYL.FIS” in Example 3.2 contains *FISH* routines used to prepare the *UDEC* parameters and perform the post-processing of results. “CYL.DAT” in Example 3.3 contains the *UDEC* commands to perform a pressurized cylinder test.

3.4 Results and Discussion

For each internal pressure, and each of the three discretizations, the *UDEC* results are compared to the analytic results in terms of normalized plastic radius, $c^* = c/b$, and normalized displacements of the outer wall, $U^* = (u/b)(E/k_T)$, as shown in Table 3.1.

Table 3.1 Comparison of UDEC and analytic results for thick-walled cylinder problems

	Analytic solution		*UDEC* results $n = 4$		*UDEC* results $n = 6$		*UDEC* results $n = 8$	
P/k_T	c^*	U^*	c^*	U^*	c^*	U^*	c^*	U^*
0.65	0.694	0.926	0.694	0.932	0.700	0.938	0.692	0.942
0.70	0.731	1.025	0.737	1.037	0.727	1.038	0.727	1.042
0.75	0.738	1.150	0.760	1.160	0.771	1.170	0.774	1.169

The normalized plastic radius, c^*, shown in Table 3.1 was calculated from *UDEC* results:

$$c^* = \left[\frac{A_p}{A_t} \left(1 - \frac{a^2}{b^2} \right) + \frac{a^2}{b^2} \right]^{1/2}$$

where

$$\frac{A_p}{A_t} = \frac{\pi \, (c^2 - a^2)}{\pi \, (b^2 - a^2)}$$

and A_p = total area of plastic zones, and A_t = total area of the model.

The normalized displacements of the outer wall, U^*, were determined by averaging displacements of all gridpoints on the outer wall.

Figure 3.3 shows a comparison of stresses along the radius calculated by *UDEC* versus the analytical solution. The numerical results are from the model with 8 zones per thickness of the wall. Each point on the graph is calculated by averaging stresses in zones with the center on the same radius r/b. Note the closer agreement between numerical and analytical solution in the elastic region, and the somewhat wider scatter of numerical results in the plastic region. This is largely due to the effects of using constant strain triangular elements; this is discussed in more detail in Section 4.

The results shown in Tables 3.1 and 3.3 indicate good agreement between the analytic solution and *UDEC* results for 4, 6 and 8 zones per thickness of the wall.

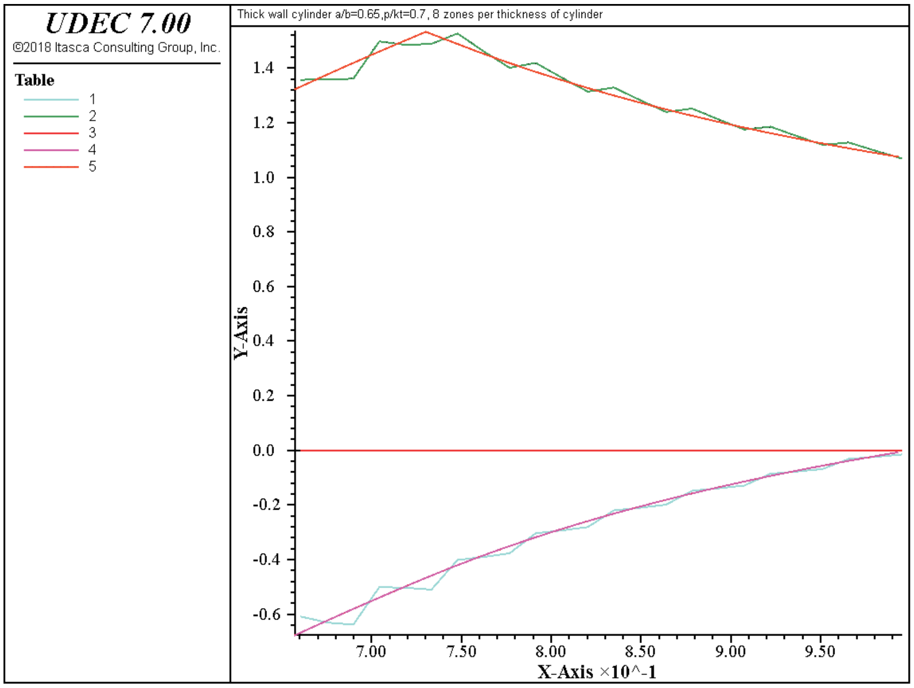

Figure 3.3 Comparison between UDEC and analytical solutions

3.5 References

Ford, H., and J. M. Alexander. ***Advanced Mechanics of Materials***. New York: Halsted Press (1977).

Wart, R. J., E. L. Skiba and R. H. Curtis. "Benchmark Problems for Repository Design Models," NUREG/CR-3636 (February 1984).

3.6 Listing of Data Files

Example 3.1 CYL_IN.DAT

```
model new
;-------------------------------------------------------
; Verification test:
; Thick-walled cylinder subjected to internal pressure
;
; Input data
;-------------------------------------------------------
;
table 1 delete
table 2 delete
table 3 delete
table 4 delete
table 5 delete
table 6 delete
call 'cont_cyl.fis'
;
; --- inner radius normalized with outer radius ---
;
fish set @a_nor 0.65
;
; --- pressure normalized with kt ---
;
fish set @p_nor 0.65
;
; --- elastic parameters; E normalized with kt ---
;
fish set @e_nor 250.
fish set @poisson 0.20
;
; --- number of zones per thickness ---
;
fish set @z_n 4
call 'cyl.dat'
;
model new
call 'cont_cyl.fis'
fish set @a_nor 0.65
fish set @p_nor 0.65
fish set @e_nor 250.
fish set @poisson 0.20
fish set @z_n 6
call 'cyl.dat'
```

```
;
model new
call 'cont_cyl.fis'
fish set @a_nor 0.65
fish set @p_nor 0.65
fish set @e_nor 250.
fish set @poisson 0.20
fish set @z_n 8
call 'cyl.dat'
;
model new
call 'cont_cyl.fis'
fish set @a_nor 0.65
fish set @p_nor 0.7
fish set @e_nor 250.
fish set @poisson 0.20
fish set @z_n 4
call 'cyl.dat'
;
model new
call 'cont_cyl.fis'
fish set @a_nor 0.65
fish set @p_nor 0.7
fish set @e_nor 250.
fish set @poisson 0.20
fish set @z_n 6
call 'cyl.dat'
;
model new
call 'cont_cyl.fis'
fish set @a_nor 0.65
fish set @p_nor 0.7
fish set @e_nor 250.
fish set @poisson 0.20
fish set @z_n 8
call 'cyl.dat'
;
model new
call 'cont_cyl.fis'
fish set @a_nor 0.65
fish set @p_nor 0.75
fish set @e_nor 250.
fish set @poisson 0.20
fish set @z_n 4
call 'cyl.dat'
;
```

```
model new
call ’cont_cyl.fis’
fish set @a_nor 0.65
fish set @p_nor 0.75
fish set @e_nor 250.
fish set @poisson 0.20
fish set @z_n 6
call ’cyl.dat’
;
model new
call ’cont_cyl.fis’
fish set @a_nor 0.65
fish set @p_nor 0.75
fish set @e_nor 250.
fish set @poisson 0.20
fish set @z_n 8
call ’cyl.dat’
;
ret
```

Example 3.2 CONT_CYL.FIS

```
;-----------------------------------------------------
; Verification test:
; Thick-walled cylinder subjected to internal pressure
;
; Model preparation and post-processing
;-----------------------------------------------------
;
fish define setup
;
; --- title of simulation ---
;
run_t   = ’Thick wall cylinder a/b=’+string(a_nor)+’,p/kt=’+string(p_nor)
run_t   = run_t+’, ’+string(z_n)+’ zones per thickness of cylinder’
ta       = int(100.*a_nor+.5)
pa       = int(100.*p_nor+.5)
nam_t   = ’a’+string(ta)+’p’+string(pa)+’z’+string(z_n)+’.sav’
nam_ps  = ’a’+string(ta)+’p’+string(pa)+’z’+string(z_n)+’.png’
;
; --- check yield conditions at boundaries of cylinder ----
;
f_b     = p_nor+2*math.ln(a_nor)
f_a     = p_nor-1.+a_nor*a_nor
if f_b > 0. then
```

```
  msg = io.out(' The whole cylinder is yielding ')
end_if
if f_a < 0. then
  msg = io.out(' The whole cylinder is elastic ')
end_if
;
; --- UDEC parameters ---
;
b_rad   = 6.
kt       = 170.
z_nor   = 1./z_n+0.01
z_len   = z_nor*(1.-a_nor)*b_rad
a_rad   = a_nor*b_rad
p        = -p_nor*kt
youngs   = e_nor*kt
shear_m = youngs/(2*(1+poisson))
bulk_m  = youngs/(3*(1-2*poisson))
joint_s = 20.*bulk_m
inf      = 10000.*bulk_m
;
; --- parameters used in generation of model geometry ---
;
mid_rad = 0.5*(a_rad+b_rad)
inr_rad = a_rad*math.cos(45*math.degrad)
out_rad = b_rad*math.cos(45*math.degrad)
delp     = 0.1*z_len
delm     = -0.1*z_len
a1       = a_rad+delm
a2       = a_rad+delp
b1       = b_rad+delm
b2       = b_rad+delp
end
;
fish define analit
;
; --- calculation of radius of plastic zone by solution of
;     non-linear equation using Newton's method
;
if f_a > 0 then
  if f_b < 0 then
    delx = 1.-a_nor
    cx   = a_nor
    loop while delx > 0.0001*(1.-a_nor)
      f_x  = p_nor-2.*(math.ln(cx/a_nor))-(1-cx*cx)
      fp_x = -2.*(1./cx-cx)
      cxnew = cx-f_x/fp_x
```

```
      delx  = cxnew-cx
      cx    = cxnew
    end_loop
  end_if
end_if
;
; --- calculates stresses at the points corresponding to the
;     locations of zone centroids
;
loop i (1,3*z_n)
  x_cur = a_nor+(2.0*i-1)*(1.-a_nor)/(6.*z_n)
  if x_cur > cx then
    sig_r = -cx*cx*(1./(x_cur*x_cur)-1.)
    sig_t =  cx*cx*(1./(x_cur*x_cur)+1.)
  else
    sig_r = 2.0*math.ln(x_cur/cx)-(1-cx*cx)
    sig_t = 2.0*math.ln(x_cur/cx)+(1+cx*cx)
  end_if
  table.x(4,i) = x_cur
  table.x(5,i) = x_cur
  table.y(4,i) = sig_r
  table.y(5,i) = sig_t
end_loop
if a_nor > cx then
  sig_ra = -cx*cx*(1./(a_nor*a_nor)-1.)
  sig_ta =  cx*cx*(1./(a_nor*a_nor)+1.)
else
  sig_ra = 2.0*math.ln(a_nor/cx)-(1-cx*cx)
  sig_ta = 2.0*math.ln(a_nor/cx)+(1+cx*cx)
end_if
if 1.0 > cx then
  sig_rb = 0.
  sig_tb = 2.0*cx*cx
  u_b    = 2.0*(1.0-poisson*poisson)*cx*cx
end_if
end
;
fish define summary
;
; --- summary of numerical calculation ---
;
tsz           = 3*z_n
table.x(1,tsz) = 0.
table.x(2,tsz) = 0.
table.x(3,tsz) = 0.
table.x(6,tsz) = 0
```

```
table.y(1,tsz) = 0.
table.y(2,tsz) = 0.
table.y(3,tsz) = 0.
table.y(6,tsz) = 0
io.out('table 6 x '+string(table.x(6,1)))
plarea        = 0.
toarea        = 0.
ibl           = block.head
loop while ibl # 0
  izo = block.zone(ibl)
  loop while izo # 0
    gp1  = block.zone.gp(izo,1)
    gp2  = block.zone.gp(izo,2)
    gp3  = block.zone.gp(izo,3)
    x1   = block.gp.pos.x(gp1)
    x2   = block.gp.pos.x(gp2)
    x3   = block.gp.pos.x(gp3)
    y1   = block.gp.pos.y(gp1)
    y2   = block.gp.pos.y(gp2)
    y3   = block.gp.pos.y(gp3)
    r_1 = math.sqrt(x1*x1+y1*y1)/b_rad
    r_2 = math.sqrt(x2*x2+y2*y2)/b_rad
    r_3 = math.sqrt(x3*x3+y3*y3)/b_rad
    if r_1 < 1.01 then
      if r_1 > 0.99 then
        uk     = uk + 1
        xdis   = block.gp.disp.x(gp1)
        ydis   = block.gp.disp.y(gp1)
        u_cal = e_nor*math.sqrt(xdis*xdis+ydis*ydis)/b_rad
        u_bn  = (u_bn*(uk-1.)+u_cal)/uk
      end_if
    end_if
    if r_2 < 1.01 then
      if r_2 > 0.99 then
        uk     = uk + 1
        xdis   = block.gp.disp.x(gp2)
        ydis   = block.gp.disp.y(gp2)
        u_cal = e_nor*math.sqrt(xdis*xdis+ydis*ydis)/b_rad
        u_bn  = (u_bn*(uk-1.)+u_cal)/uk
      end_if
    end_if
    if r_3 < 1.01 then
      if r_3 > 0.99 then
        uk     = uk + 1
        xdis   = block.gp.disp.x(gp3)
        ydis   = block.gp.disp.y(gp3)
```

```
        u_cal = e_nor*math.sqrt(xdis*xdis+ydis*ydis)/b_rad
        u_bn  = (u_bn*(uk-1.)+u_cal)/uk
      end_if
    end_if
    x_mid = 0.33333*(x1+x2+x3)
    y_mid = 0.33333*(y1+y2+y3)
    r_mid = math.sqrt(x_mid*x_mid+y_mid*y_mid)
    cs     = x_mid/r_mid
    sn     = y_mid/r_mid
    r_mid = r_mid/b_rad
;
; --- averaging stresses along the same radius ---
;
;io.out('table '+string(table.x(6,1)))
    loop i (1,3*z_n)
      if r_mid < 1.01*(a_nor+(2*i-1)*(1.-a_nor)/(6.*z_n)) then
        if r_mid > 0.99*(a_nor+(2*i-1)*(1.-a_nor)/(6.*z_n)) then
          table.x(6,i) = table.x(6,i)+1
          ii          = table.x(6,i)
          ;io.out(string(ii))
          nsxx = block.zone.stress.xx(izo)/kt
          nsxy = block.zone.stress.xy(izo)/kt
          nsyy = block.zone.stress.yy(izo)/kt
          nsr  = nsxx*cs*cs+nsyy*sn*sn+2.*nsxy*sn*cs
          nst  = nsxx*sn*sn+nsyy*cs*cs-2.*nsxy*sn*cs
          nsrt = (nsyy-nsxx)*sn*cs+nsxy*(cs*cs-sn*sn)
          table.x(1,i) = (table.x(1,i)*(ii-1.)+r_mid)/ii
          table.x(2,i) = table.x(1,i)
          table.x(3,i) = table.x(1,i)
          table.y(1,i) = (table.y(1,i)*(ii-1.)+nsr)/ii
          table.y(2,i) = (table.y(2,i)*(ii-1.)+nst)/ii
          table.y(3,i) = (table.y(3,i)*(ii-1.)+nsrt)/ii
        end_if
      end_if
    end_loop
    zoarea = math.abs(0.5*((x2-x1)*(y3-y1)-(x3-x1)*(y2-y1)))
;
; --- calculating total area of the wall and area of yielding zones ---
;
    if block.zone.state(izo) # 0 then
      plarea = plarea+zoarea
    end_if
    toarea = toarea+zoarea
    izo = block.zone.next(izo)
  end_loop
  ibl = block.next(ibl)
```

```
end_loop
cxn = math.sqrt(plarea*(1.-a_nor*a_nor)/toarea+a_nor*a_nor)
end
```

Example 3.3 CYL.DAT

```
;-----------------------------------------------------
; Verification test:
; Thick-walled cylinder subjected to internal pressure
;
; UDEC commands
;-----------------------------------------------------
log on
;
; --- problem setup and analytic solution ---
;
@setup
@analit
model title @run_t
;
; --- generation of geometry ---
;
block tolerance corner-round-length .01
block create polygon 0 0 0 @b_rad @b_rad @b_rad @b_rad 0
block cut arc 0 0 @a_rad 0 90 10
block cut arc 0 0 @b_rad 0 90 10
block delete range position-x 0 @inr_rad position-y 0 @inr_rad
block cut crack 0 0 @mid_rad @mid_rad
block delete range pos-x @out_rad @b_rad pos-y @out_rad @b_rad
;
; --- material properties ---
;
block property material 1 density 1 bulk @bulk_m shear @shear_m
block property material 1 cohesion @kt friction 0 tension @inf
block contact property material 1 stiffness-normal @joint_s ...
  stiffness-shear @joint_s cohesion @kt friction 0 tension @inf
;
; --- generate finite difference zones for fully-deformable blocks ---
;
block zone gen quad @z_len
;
; --- Tresca model ---
;
block change model 3
;
```

```
;
; --- histories ---
;
history interval 100
block gridpoint history displacement-x @a_rad 0
block gridpoint history displacement-x @b_rad 0
block gridpoint history displacement-y 0 @a_rad
block gridpoint history displacement-y 0 @b_rad
;
; --- boundary conditions ---
;
block edge apply stress @p 0 @p range ann center 0 0. radius @a1 @a2
block grid apply vel-y 0 range pos-x @a1 @b2 pos-y @delm @delp
block grid apply vel-x 0 range pos-x @delm @delp pos-y @a1 @b2
;
; --- cycle to equilibrium ---
;
block solve ratio 1e-5
;
; --- summary of numerical results ---
;
@summary
;
; --- save 'state ---
;
model save @nam_t
log on
;
; --- radius of plastic zone ---
; --- analytical ---
;
fish list @cx
;
; --- numerical ---
;
fish list @cxn
;
; --- displacement of outer boundary ---
; --- analytical ---
;
fish list @u_b
;
; --- numerical ---
;
fish list @u_bn
log off
```

```
table 1 label 'Radial Stress - UDEC'
table 2 label 'Hoop Stress   - UDEC'
table 3 label 'Shear Stress  - UDEC'
table 4 label 'Radial Stress - Analytic'
table 5 label 'Hoop Stress   - Analytic'
return
```