5 Circular Tunnel Problems

5.1 Problem Statement

This problem concerns stress analysis of a long circular opening in an infinite medium under various boundary conditions and material properties (see Figure 5.1).* Three variations to this problem will be considered:

(1) Part A – tunnel in an elastic medium with a biaxial stress field;

(2) Part B – tunnel in an elastic-plastic medium with a hydrostatic stress field; and

(3) Part C – lined tunnel in an elastic medium with a biaxial stress field.

Upon excavation of a tunnel, the in-situ stresses within the rock or soil mass are redistributed from a uniform orthogonal stress field to a more complex stress distribution. Stress concentrations around a tunnel cause elastic deformations at the periphery and, if the yield strength of the material is exceeded, result in plastic deformations and redistribution of stresses due to yielding of the material. In the case of plastic yielding, a yield zone will develop around the tunnel, beyond which the stresses will be elastic. These processes are modeled by parts A and B of this problem.

Part C of this problem involves the interaction of a structural tunnel lining and an elastic medium. Although the actual design of a tunnel lining is more complex, this problem checks the basic interaction between the two types of material for non-axisymmetric loadings.

These problems have a closed-form analytical solution, and thus several aspects of the computer model can be tested:

(1) the ability of the code to simulate an infinite medium by boundary elements;

(2) the determination of displacements and stresses in a nonsymmetric problem in two dimensions;

(3) the computation of plastic stresses and deformations; and

(4) the interaction between structural lining and rock or soil mass.

---

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

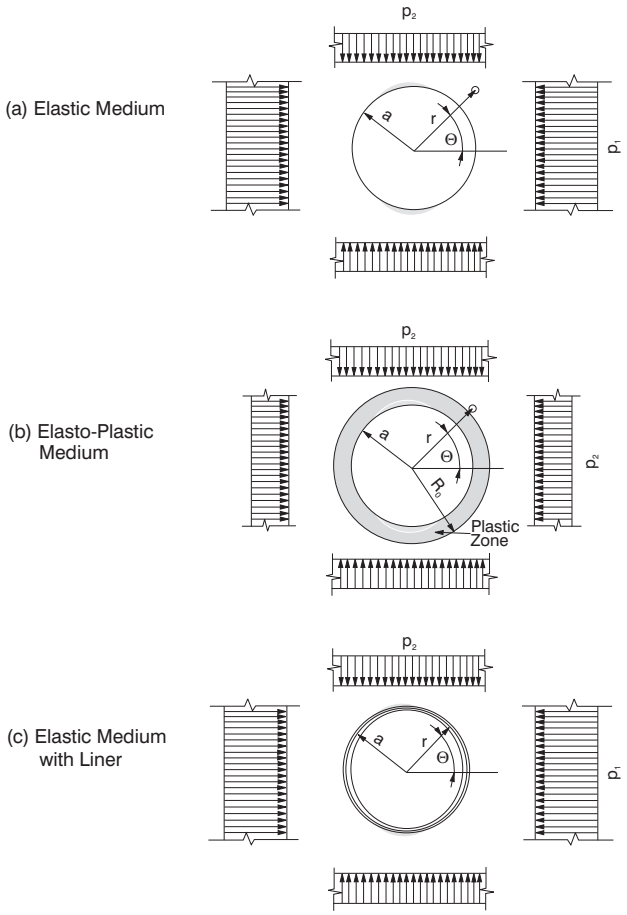


Figure 5.1 Three variations to the circular tunnel problem (after Wart et al. 1984)

5.2 Analytical Solutions

5.2.1 Cylindrical Hole in an Infinite Elastic Medium

For a cylindrical hole in an infinite, isotropic, elastic medium under plane-strain conditions, the radial and tangential stress distributions are given by the classical Kirsch solution (e.g., see Jaeger and Cook 1976).

A point located at polar coordinate (r, θ) near an opening with radius a (see Figure 5.1 (a)) has stresses σ_r, σ_θ, $\tau_{r\theta}$, given by

$$\frac{2\sigma_r}{p_1+p_2} = \sigma_r^*\left(\frac{r}{a},\theta;\frac{p_1-p_2}{p_1+p_2}\right) = 1-\frac{a^2}{r^2} + \frac{p_1-p_2}{p_1+p_2}\left(1-\frac{4a^2}{r^2}+\frac{3a^4}{r^4}\right)\cos 2\theta$$

$$\frac{2\sigma_\theta}{p_1+p_2} = \sigma_\theta^*\left(\frac{r}{a},\theta;\frac{p_1-p_2}{p_1+p_2}\right) = 1+\frac{a^2}{r^2} - \frac{p_1-p_2}{p_1+p_2}\left(1+\frac{3a^4}{r^4}\right)\cos 2\theta \qquad (5.1)$$

$$\frac{2\tau_{r\theta}}{p_1+p_2} = \tau_{r\theta}^*\left(\frac{r}{a},\theta;\frac{p_1-p_2}{p_1+p_2}\right) = -\frac{p_1-p_2}{p_1+p_2}\left(1+\frac{2a^2}{r^2}-\frac{3a^4}{r^4}\right)\sin 2\theta$$

The displacements can also be determined assuming conditions of plane strain:

$$\frac{4G}{p_1+p_2}\frac{u_r}{a} = U_r\left(\frac{r}{a},\theta;\nu,\frac{p_1-p_2}{p_1+p_2}\right) = \frac{a}{r} + \frac{p_1-p_2}{p_1+p_2}\frac{a}{r}\left[4(1-\nu)-\frac{a^2}{r^2}\right]\cos 2\theta$$

$$\frac{4G}{p_1+p_2}\frac{u_\theta}{a} = U_\theta\left(\frac{r}{a},\theta;\nu,\frac{p_1-p_2}{p_1+p_2}\right) = -\frac{p_1-p_2}{p_1+p_2}\frac{a}{r}\left[2(1-2\nu)+\frac{a^2}{r^2}\right]\sin 2\theta \qquad (5.2)$$

in which u_r is the radial outward displacement, and u_θ is the tangential displacement. G is the shear modulus, and ν is the Poisson's ratio.

5.2.2 Cylindrical Hole in an Infinite Mohr-Coulomb Medium

The yield zone radius, R_o (see Figure 5.1 (b)), is given analytically by a theoretical model based on the solution of Salencon (1969):

$$\frac{R_o}{a} = R_o^*\left(\frac{p_2}{q},\ \frac{p_i}{q},\ K_p\right) = \left(\frac{2}{K_p+1}\ \frac{\frac{p_2}{q}+\frac{1}{K_p-1}}{\frac{p_i}{q}+\frac{1}{K_p-1}}\right)^{1/(K_p-1)} \qquad (5.3)$$

where $K_p = (1+\sin\phi)/(1-\sin\phi)$;
$q = 2\,c\tan(45+\phi/2)$; and
p_i = internal pressure.

The radial stress at the elastic-plastic interface is

$$\frac{\sigma_{re}}{q} = -\frac{1}{K_p + 1}\left(2\,\frac{p_2}{q} - 1\right) \tag{5.4}$$

The stresses in the plastic zone are

$$\begin{aligned}
\frac{\sigma_r}{q} &= \sigma_r^*\left(\frac{p_i}{q},\ K_p\right) = \frac{1}{K_p - 1} - \left(\frac{p_i}{q} + \frac{1}{K_p - 1}\right)\cdot\left(\frac{r}{a}\right)^{K_p - 1} \\
\frac{\sigma_\theta}{q} &= \sigma_\theta^*\left(\frac{p_i}{q},\ K_p\right) = \frac{1}{K_p - 1} - K_p\left(\frac{p_i}{q} + \frac{1}{K_p - 1}\right)\cdot\left(\frac{r}{a}\right)^{K_p - 1}
\end{aligned} \tag{5.5}$$

The stresses in the elastic zone are

$$\begin{aligned}
\frac{\sigma_r}{q} &= \sigma_r^*\left(\frac{p_2}{q},\ \frac{p_i}{q},\ K_p\right) = -\frac{p_2}{q} + \left(\frac{p_2}{q} + \frac{\sigma_{re}}{q}\right)\cdot\left(\frac{R_o}{r}\right)^2 \\
\frac{\sigma_\theta}{q} &= \sigma_\theta^*\left(\frac{p_2}{q},\ \frac{p_i}{q},\ K_p\right) = -\frac{p_2}{q} - \left(\frac{p_2}{q} + \frac{\sigma_{re}}{q}\right)\cdot\left(\frac{R_o}{r}\right)^2
\end{aligned} \tag{5.6}$$

5.2.3 *Lined Tunnel in an Infinite Elastic Medium*

The analytical solution for an elastic liner embedded in an elastic solid with non-slipping interface (see Figure 5.1 (c)) is given by Einstein and Schwartz (1979). The thrust or axial force in the liner, N, and bending moment, M, are given in Eqs. (5.7) and (5.8), respectively:

$$\begin{aligned}
\frac{N}{P_o a} &= N^*\left(\theta;\ \frac{p_1}{p_2},\ \nu,\ \nu_s,\ \frac{E_s}{E}, \frac{d}{a}\right) \\
&= \frac{1}{2}\left(1 + \frac{p_1}{p_2}\right)(1 - a_0^*) + \frac{1}{2}\left(1 - \frac{p_1}{p_2}\right)(1 + 2a_2^*)\cos 2\theta
\end{aligned} \tag{5.7}$$

$$\begin{aligned}
\frac{M}{P_o a^2} &= M^*\left(\theta;\ \frac{p_1}{p_2},\ \nu,\ \nu_s,\ \frac{E_s}{E}, \frac{d}{a}\right) \\
&= \frac{1}{4}\left(1 - \frac{p_1}{p_2}\right)(1 - 2a_2^* + 2b_2^*)\cos 2\theta
\end{aligned} \tag{5.8}$$

where E = Young's modulus of the rock;

ν = Poisson's ratio of the rock;

E_s = Young's modulus of the liner;

ν_s = Poisson's ratio of the liner;

d = thickness of the liner;

A = d, cross-sectional area of the liner for a 1 m long section; and

I = $d^3/12$, liner moment of inertia;

$$a_0^* = \frac{C^*\ F^*\ (1-\nu)}{C^*+F^*+C^*\ F^*\ (1-\nu)};$$

$$a_2^* = \beta \cdot b_2^*;$$

$$\beta = \frac{(6+F^*)\ C^*\ (1-\nu) + 2F^*\nu}{3F^* + 3C^* + 2C^*\ F^*\ (1-\nu)};$$

$$b_2^* = \frac{C^*\ (1-\nu)}{2\ [C^*\ (1-\nu) + 4\nu - 6\beta - 3\beta\ C^*\ (1-\nu)]};$$

$$C^* = \frac{E\ a\ (1-\nu_s^2)}{E_s\ A\ (1-\nu^2)};\ \text{and}$$

$$F^* = \frac{E\ a^3\ (1-\nu_s^2)}{E_s\ I\ (1-\nu^2)}.$$

UDEC structural elements do not require a Poisson's ratio to be specified. In order to comply with the analytical solution, a plane-strain correction of $(1 - \nu_s^2)$ is applied to the Young's modulus of the liner.

5.3 *UDEC* Models

The following dimensionless parameters and values are used to describe the problems. Note that the density is not required by the analytical solution, but some value must be provided in *UDEC*. Since the solutions are independent of the choice of density, we used $\rho = 1$.

$$\begin{aligned} p_2/p_1 &= 0.5 \\ \frac{(p_1-p_2)}{(p_1+p_2)} &= \frac{1-p_2/p_1}{1+p_2/p_1} = 1/3 \\ p_2/q &= 0.75 \\ \nu &= 0.20 \\ E/q &= 300 \\ \phi &= 20^\circ \\ \psi &= 20^\circ \end{aligned}$$

$$\begin{aligned} \nu_s &= 0.20 \\ E_s/E &= 3 \\ d/a &= 0.1 \end{aligned}$$

For each part, two different discretizations are used, such that characteristic lengths, ℓ_z, of zones at the tunnel contour are (1) $a/\ell_z = 5$ (see Figure 5.2), and (2) $a/\ell_z = 10$ (see Figure 5.3). In both geometries, the inner and outer radii were 5.0 m and 30.0 m, respectively. Also, boundary elements were coupled to gridpoints in the outer boundary in both cases. "Glued" joints were used to provide the needed discretization in each case.

In Part C of the problem, interaction of a structural lining with the surrounding material is modeled. For this part, the lining was divided into 48 (for $a/\ell_z = 5$) and 96 (for $a/\ell_z = 10$) linear segments. To satisfy the conditions of perfect bonding between the lining and surrounding material, high interface stiffness and strength parameters were specified.

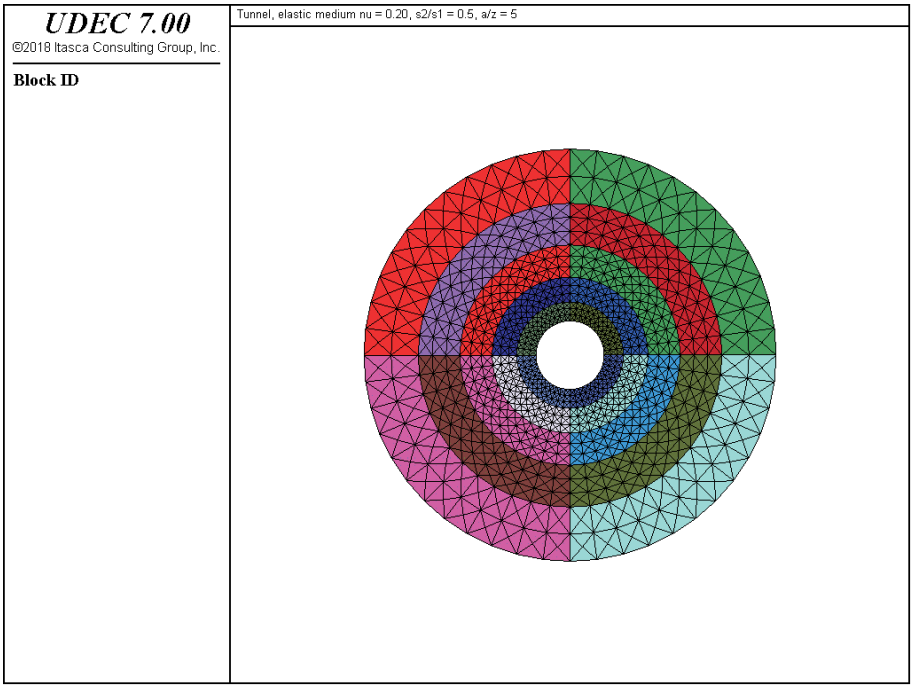


Figure 5.2 Coarse zoning used in circular tunnel problems

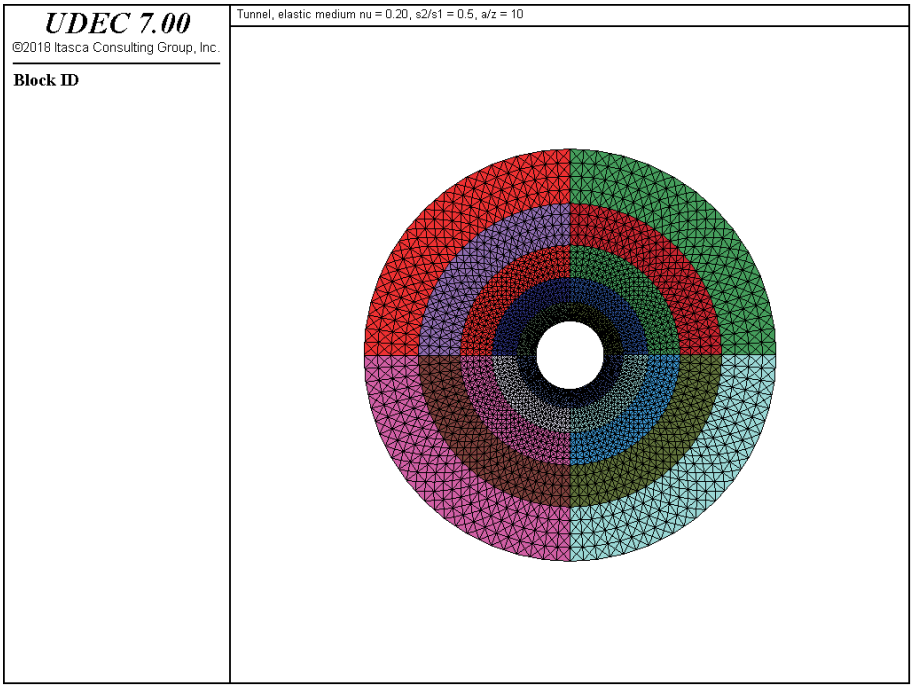


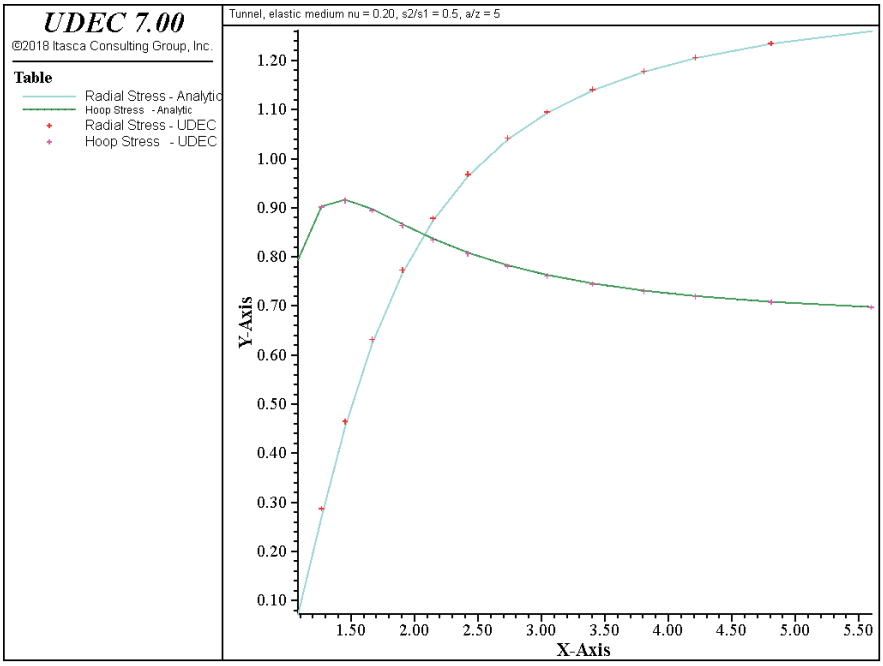
Figure 5.3 Finer zoning used in circular tunnel problems

5.4 Results

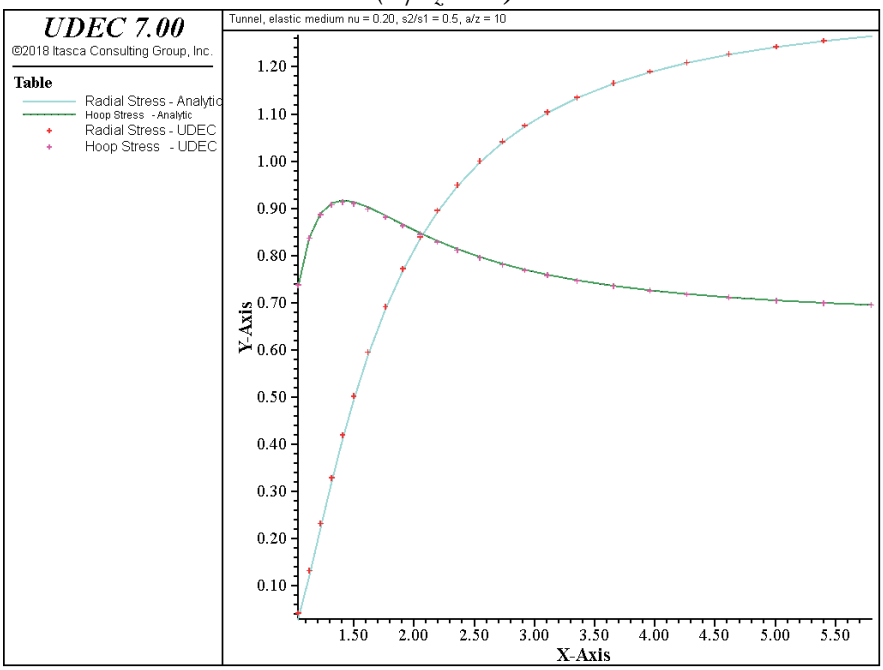
Part A – The results for Part A are compared graphically with the analytic solution in Figures 5.4 and 5.5. All results shown are for a line along the major principal stress direction. The finer zoning resulted in improved correspondence with the analytical solution.

Part B – The results of Part B are compared graphically with the analytic solution in Figure 5.6. The calculated radius of the elastic-plastic interface, R_o^*, based on the analytic solution is 1.164. For *UDEC*, the corresponding radius was determined following the procedure described in Section 3. The radius of the elastic-plastic interface was found to be 1.184 for both the coarse and fine zoning, or error of 1.72%.

Part C – The *UDEC* results for Part C are presented in terms of lining thrust, N^*, and moment, M^*, in Figures 5.7 and 5.8. Results shown are for the first quadrant. Results for the other quadrants are similar.

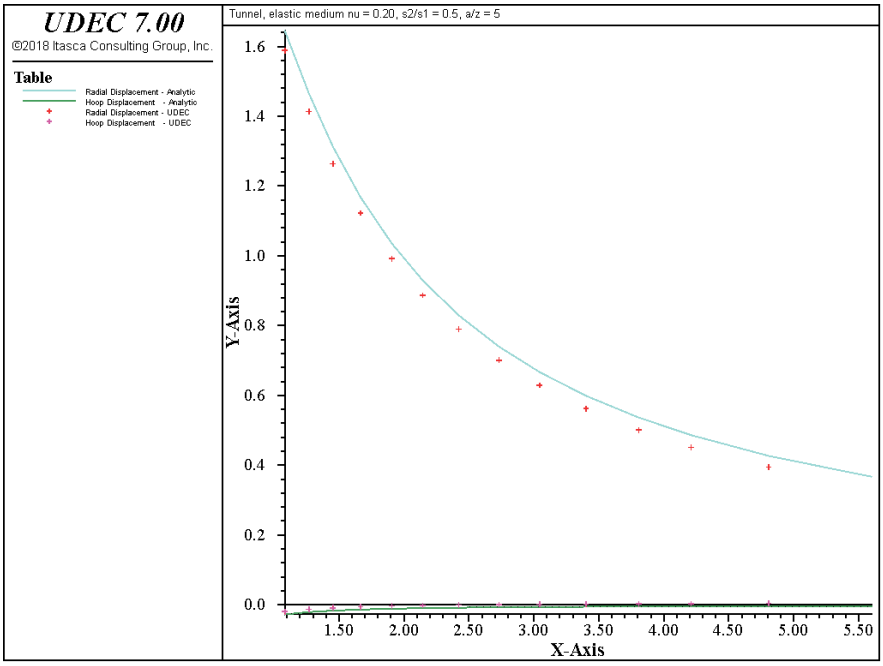


$(a/\ell_z = 5)$

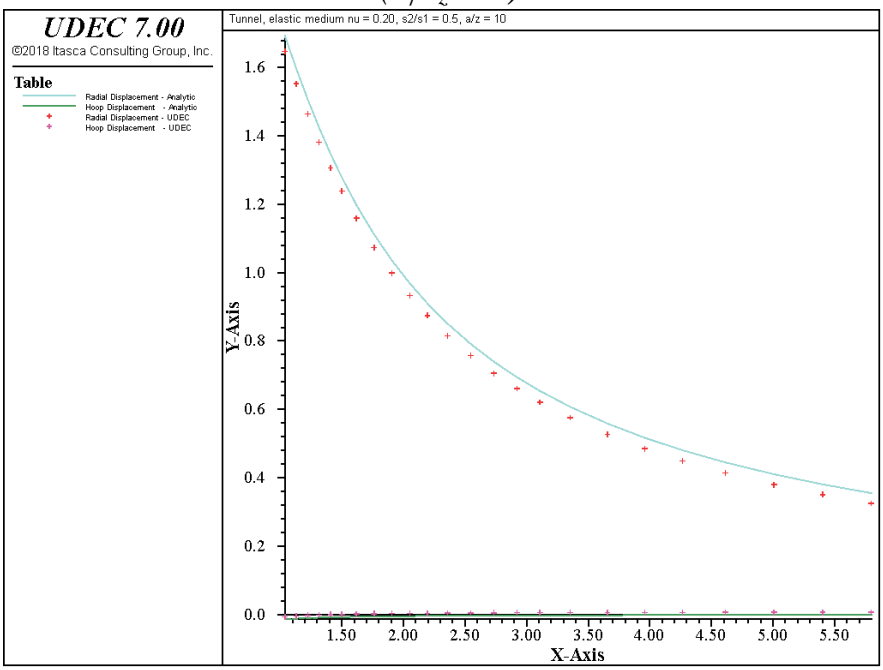


$(a/\ell_z = 10)$

Figure 5.4 Comparison of UDEC results of radial and tangential stresses versus radial distance along a line $\theta = 0°$, with analytical solution for the case of a tunnel in an elastic medium with a biaxial stress field

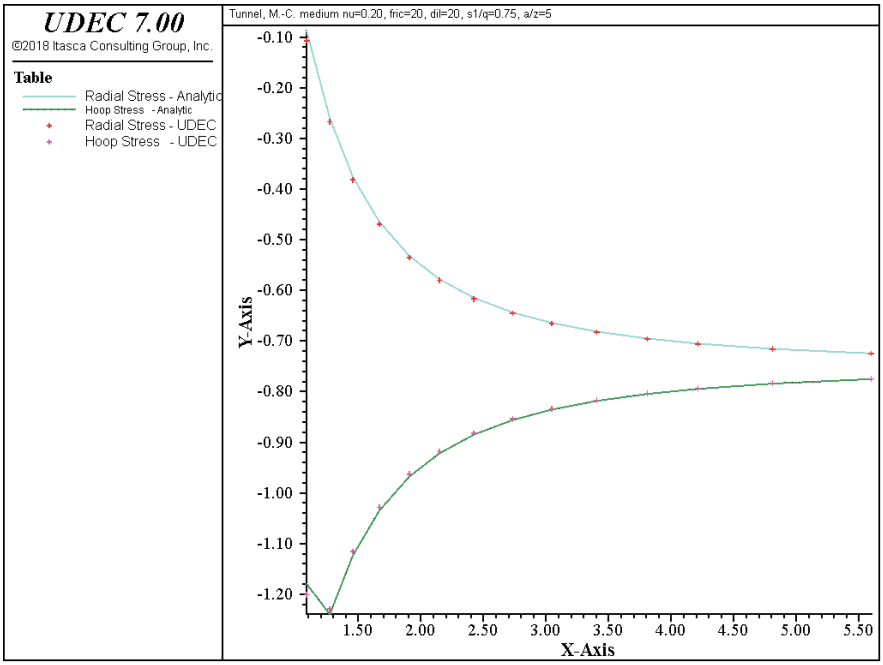


$(a/\ell_z = 5)$

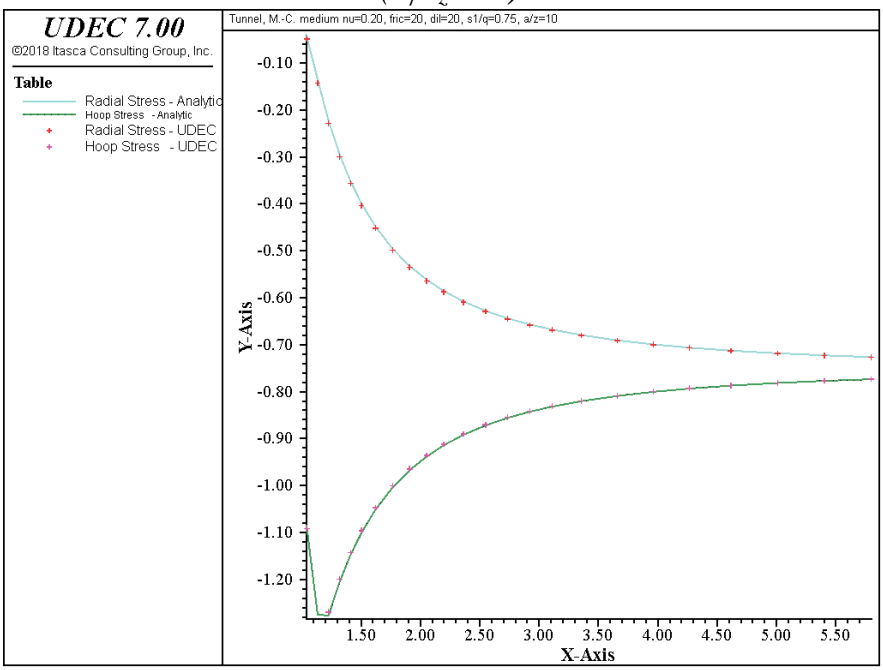


$(a/\ell_z = 10)$

Figure 5.5 Comparison of UDEC results of radial and tangential displacements versus radial distance along a line $\theta = 0°$, with analytical solution for the case of a tunnel in an elastic medium with a biaxial stress field

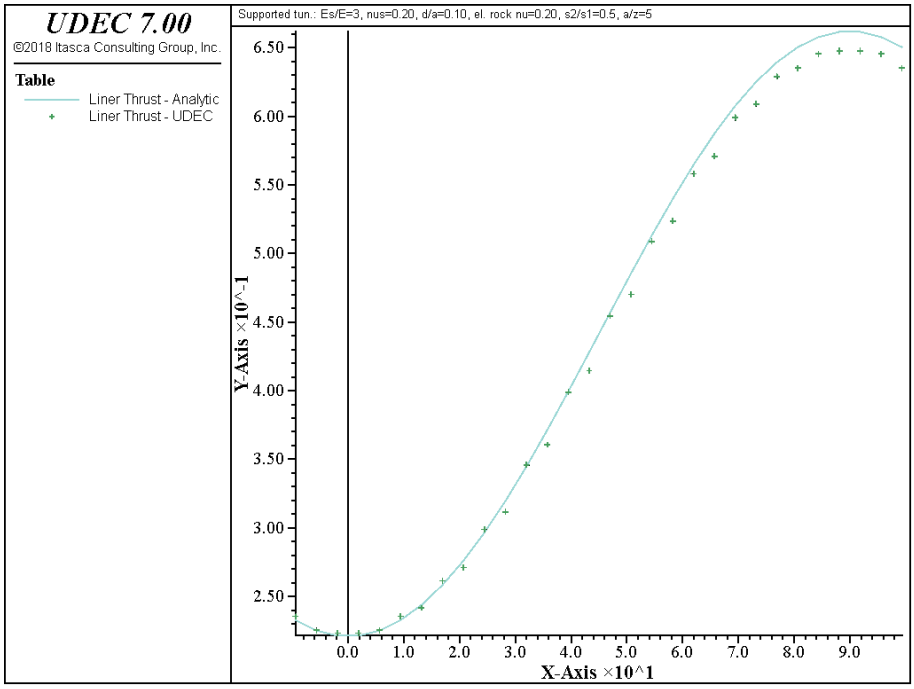


$(a/\ell_z = 5)$

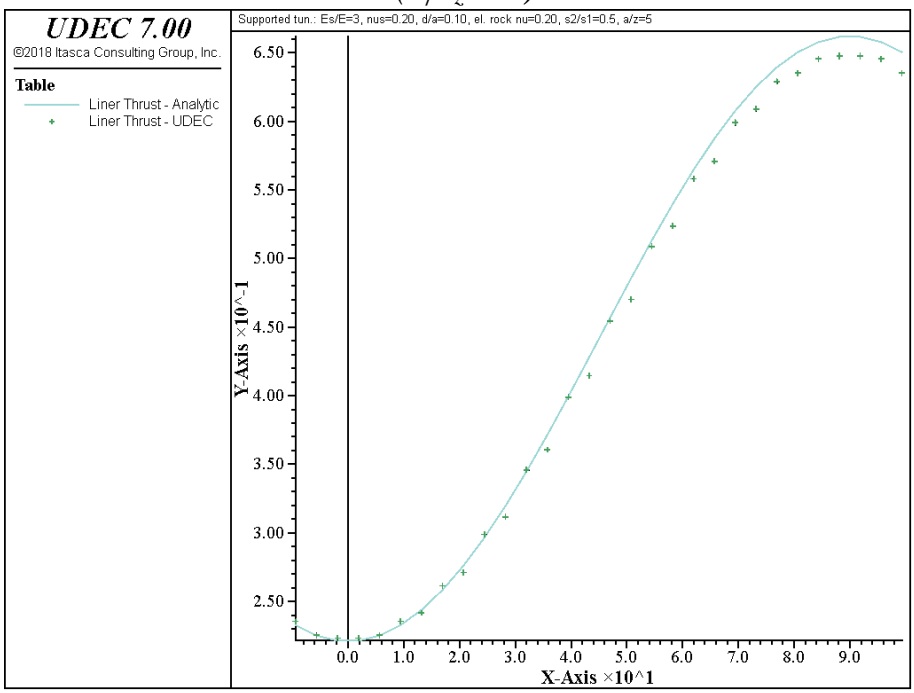


$(a/\ell_z = 10)$

Figure 5.6 Comparison of UDEC results of radial and tangential stresses versus radial distance along a line $\theta = 0°$, with analytical solution for the case of a tunnel in Mohr-Coulomb medium with a biaxial stress field

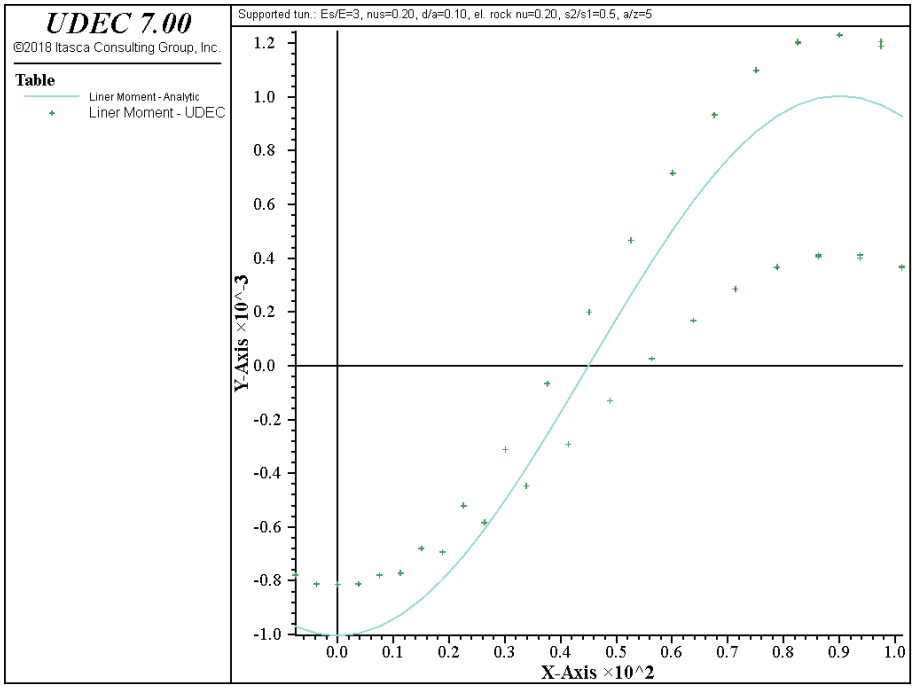


$(a/\ell_z = 5)$

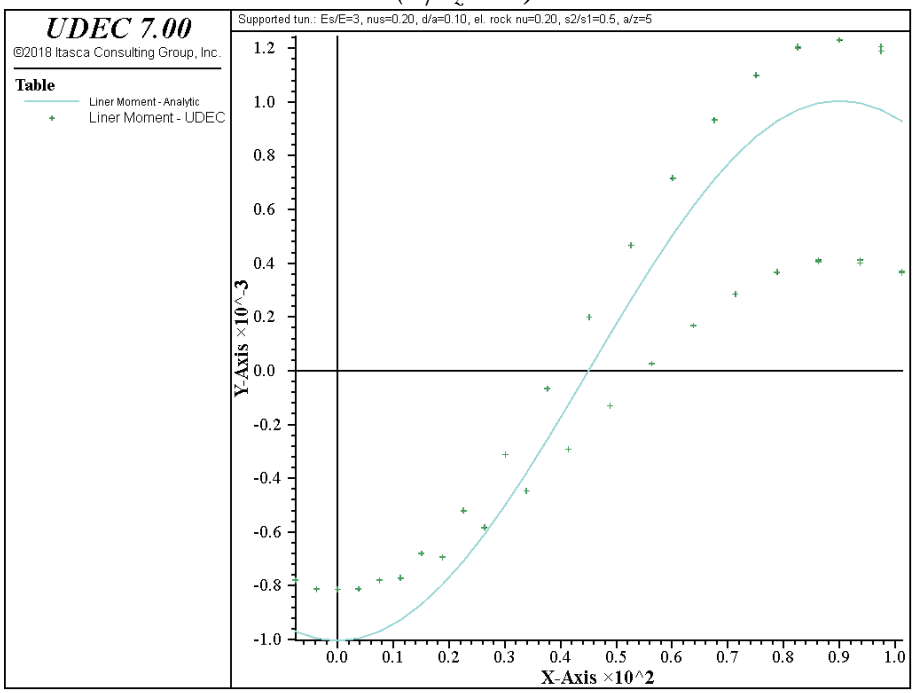


$(a/\ell_z = 10)$

Figure 5.7 Comparison of UDEC results for lining thrust with analytical solution for the case of a lined tunnel in an elastic medium with a biaxial stress field



$(a/\ell_z = 5)$



$(a/\ell_z = 10)$

Figure 5.8 Comparison of UDEC results for lining moment with analytical solution for the case of a lined tunnel in an elastic medium with a biaxial stress field

5.5 References

Einstein, H. H., and C. W. Schwartz. "Simplified Analysis for Tunnel Supports," *J. of Geotech. Eng.*, 499-518 (April 1979).

Jaeger, J. C., and N. G. W. Cook. ***Fundamentals of Rock Mechanics***, 3rd Ed. London: Chapman and Hall (1976).

Salencon, J. "Contraction Quasi-Statique d'une Cavite a Symmetrie Spherique ou Cylindrique dans un Milieu Elastoplastique," *Annales des Ponts et Chaussees*, ***4***, 213-216 (1969).

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

5.6 Listing of Data Files

Example 5.1 CYLBE_IN.DAT

```
model new
;-----------------------------------------------------------
; Verification test:
; Circular tunnel problems using boundary elements
;
; Input data
;-----------------------------------------------------------
call 'cont_cb.fis'
;
; --- friction angle ---
;
fish set @phi 20.
;
; --- dilation angle ---
;
fish set @ksi 20.
;
; --- Poisson's ratio ---
;
fish set @poisson 0.2
;
; --- lateral stress coefficient ---
;
fish set @k_min  0.5
;
; --- vertical stress ---
;
fish set @sigma_q  0.75
;
; --- support - thickness normalized with radius, Young's modulus
; --- normalized with Young's modulus of medium and Poisson's ratio
;
fish set @thick_a  0.10
fish set @young_s   3.33333
fish set @poisson_s  0.2
;
; --- zone size and number of support elements ---
;
fish set @a_z 10.
fish set @ndiv 96
;
; --- convergence criterion ---
```

```
;
fish set @conv_rat 2.e-6
;
call 'cylbe.dat'
model new

call 'cont_cb.fis'
fish set @phi 20.
fish set @ksi 20.
fish set @poisson 0.2
fish set @k_min  0.5
fish set @sigma_q  0.75
fish set @thick_a  0.10
fish set @young_s   3.33333
fish set @poisson_s  0.2
fish set @ndiv 48
fish set @a_z 5.
fish set @conv_rat 2.e-6
call 'cylbe.dat'
ret
```

Example 5.2 CONT_CB.FIS

```
;-----------------------------------------------------------
; Verification test:
; Circular tunnel problems umath.sin(g boundary elements
;
; Model preparation, control and post-procesmath.sin(g
;-----------------------------------------------------------
;
fish define sconv
;
; --- converts real numbers into strings ---
; --- arguments: number - realn and number of decimal digits - ndig ---
;
  irealn = int(realn)
  srealn = string(irealn)
  if ndig = 0 then
    exit
  end_if
  srealn = string(irealn)+'.'
  realn  = realn - irealn
  loop i (1,ndig)
    realn = 10.*realn
    irealn = int(realn)
```

```
    srealn = srealn+string(irealn)
    realn  = realn - irealn
  end_loop
end
;
fish define setup
;
; --- titles of simulations ---
;
  realn  = a_z
  ndig   = 0
  sconv
  sa_z = srealn
;
  realn  = k_min
  ndig   = 1
  sconv
  sk_min = srealn
;
  realn  = poisson
  ndig   = 2
  sconv
  spoisson = srealn
;
  realn  = poisson_s
  ndig   = 2
  sconv
  spoisson_s = srealn
;
  realn  = thick_a
  ndig   = 2
  sconv
  sthick_a = srealn
;
  realn  = young_s
  ndig   = 0
  sconv
  syoung_s = srealn
;
  realn  = phi
  ndig   = 0
  sconv
  sphi   = srealn
;
  realn  = ksi
  ndig   = 0
```

```
  sconv
  sksi   = srealn
;
  realn  = sigma_q
  ndig   = 2
  sconv
  ssigma_q = srealn
;
  run_t1  = 'Tunnel, elastic medium nu = '+spoisson
  run_t1  = run_t1+', s2/s1 = '+sk_min
  run_t1  = run_t1+', a/z = '+sa_z
;
  run_t2  = 'Supported tun.: Es/E='+syoung_s
  run_t2  = run_t2+', nus='+spoisson_s+', d/a='+sthick_a
  run_t2  = run_t2+', el. rock nu='+spoisson
  run_t2  = run_t2+', s2/s1='+sk_min
  run_t2  = run_t2+', a/z='+sa_z
;
  run_t3  = 'Tunnel, M.-C. medium nu='+spoisson
  run_t3  = run_t3+', fric='+sphi
  run_t3  = run_t3+', dil='+sksi
  run_t3  = run_t3+', s1/q='+ssigma_q
  run_t3  = run_t3+', a/z='+sa_z
;
  run_t4  = 'Supported tun. el.-pl. rock nu='+spoisson
  run_t4  = run_t3+', fric='+sphi
  run_t4  = run_t3+', dil='+sksi
  run_t4  = run_t3+', s1/q='+ssigma_q
  run_t4  = run_t3+', a/z='+sa_z
;
  nam_t1  = 'ctel'+sa_z+'z'
  nam_t1g = nam_t1+'g.sav'
  nam_t1e = nam_t1+'e.sav'
  nam_t1f = nam_t1+'.sav'
  nam_t2  = 'ctsu'+sa_z+'z'
  nam_t2e = nam_t2+'e.sav'
  nam_t3  = 'ctep'+sa_z+'z'
  nam_t3e = nam_t3+'e.sav'
  nam_t3f = nam_t3+'.sav'
  nam_t4  = 'cteps'+sa_z+'z'
  nam_t4f = nam_t4+'.sav'
;
; --- UDEC parameters ---
;
  a   = 5.
  r_a = 6.
```

```
  q   = 20.
  young_q = 300.
  young = young_q*q
  shear_m = 0.5*young/(1.+poisson)
  bulk_m = 0.333333*young/(1.-2.*poisson)
  sinphi = math.sin(phi*math.degrad)
  cosphi = math.cos(phi*math.degrad)
  sinksi = math.sin(ksi*math.degrad)
  cosksi = math.cos(ksi*math.degrad)
  coh_v  = 0.5*q*(1.-sinphi)/cosphi
;
  sigma_2 = -sigma_q*q
  sigma_1 = sigma_2/k_min
;
; --- principal stresses are 30 deg clockwise from the horizontal ---
;
  alpha = 30
  cosal = math.cos(alpha*math.degrad)
  sinal = math.sin(alpha*math.degrad)
  sigma_xx = sigma_1*cosal*cosal+sigma_2*sinal*sinal
  sigma_xy = (sigma_2-sigma_1)*sinal*cosal
  sigma_yy = sigma_2*cosal*cosal+sigma_1*sinal*sinal
  sigma_zz = 0.5*sigma_yy
;
; --- support properties ---
;
  thick_s = thick_a*a
  young_s = young_s*young
  ndiv    = ndiv
  shear_s   = 0.5*young_s/(1.+poisson_s)
  bulk_s    = 0.333333*young_s/(1.-2.*poisson_s)
  young_sp  = young_s/(1.-poisson_s*poisson_s)
  poisson_sp = poisson_s/(1.-poisson_s)
;
; --- parameters used in generation of model geometry ---
;
  conv_rat = conv_rat
  a2 = 2.*a
  a4 = 4.*a
  large = bulk_m*1e10
  if a_z < 5. then
    a_z = 5.
  end_if
  if r_a < 5. then
    r_a = 5.
  end_if
```

```
  zone = a/a_z
  j_stiff = 10.*(bulk_m+4.*shear_m/3.)/zone
  c_stiff = j_stiff*2.*math.pi*a/ndiv
  mod_size = a*r_a
  ref_poin = 3.*mod_size
  _ref_poin = -ref_poin
  ahalf = 0.5*a
  small_len = 0.2*zone
  n_small_len = -small_len
  a_l = a - small_len
  a_u = a + small_len
  mod_size_u = 1.02 * mod_size
  mod_size_l = 0.98 * mod_size
  rrat   = 1.3
  conv_rat_10 = 0.1*conv_rat
end
;
fish define rings
;
; --- generation of circular rings ---
;
  d1 = 0.5*a_z*zone
  if rrat # 1. then
    nrings = math.ln(1.-(mod_size-a)*(1.-rrat)/d1)/math.ln(rrat)
    nrings = int(nrings)
    coef = nrings*math.ln(rrat)
    coef = math.exp(coef)
    d1 = (mod_size-a)*(1.-rrat)/(1.-coef)
    pow = math.exp(nrings*math.ln(rrat))
    mod_size = a + d1*(1.-pow)/(1.-rrat)
  else
    nrings = int((mod_size-a)/d1)
    d1 = (mod_size-a)/nrings
  end_if
  _mod_size = -1*mod_size
  command
    block create circle 0 0 @mod_size 48
    block cut crack begin @_mod_size 0 end @mod_size 0
    block cut crack begin 0 @_mod_size end 0 @mod_size
  end_command
  di = d1
  dr = a
  loop i (1,nrings)
    command
      block cut tunnel 0 0 @dr 48
    end_command
```

```
    dr = dr + di
    di = di*rrat
  end_loop
end
;
fish define gzones
;
; --- zoning of the model ---
;
  zone0 = zone
  dr  = a
  di  = d1
  dri = dr + di
  jump = 1
  loop i (1,nrings)
    command
      block zone gen quad @zone range ann center 0 0 rad @dr @dri
    end_command
    zone = jump*dri*zone/dr
    dr = dri
    di = di * rrat
    dri = dri + di
    if zone > 0.5*(dri-dr) then
      zone = 0.5*(dri-dr)
    end_if
  end_loop
  command
    block zone gen edge @zone0
  end_command
end
```

Example 5.3 CYLBE.DAT

```
;-----------------------------------------------------------
; Verification test:
; Circular tunnel problems using boundary elements
;
; UDEC commands
;-----------------------------------------------------------
;
; --- problem setup ---
;
@setup
model title @run_t1
block tolerance corner-round-length = 0.002
```

```
;
; --- generation of geometry
;
@rings
@gzones
block contact join by-contact
model save @nam_t1g
;
; --- set stresses - the stress field makes the major principal
; --- stress orientated 30 degrees clockwise from the horizontal axis
;
block edge apply stress @sigma_xx @sigma_xy @sigma_yy
block insitu stress @sigma_xx @sigma_xy @sigma_yy stress-ZZ @sigma_zz
;
; --- define material properties ---
;
block property material 1 dens 1 bulk @bulk_m shear @shear_m
;
block domain update 10000
;
; --- auto damping ---
;
block mechanical damping global
;
; --- histories ---
;
;todo hist vmax
;
; --- displacement and stress histories ---
;
history interval 20
block gridpoint history disp-x @a 0
block gridpoint history disp-x @a2 0
block gridpoint history disp-x @a4 0
block gridpoint history disp-y @a 0
block gridpoint history disp-y @a2 0
block gridpoint history disp-y @a4 0
block zone history stress-xx @a 0
block zone history stress-xx @a2 0
block zone history stress-xx @a4 0
block zone history stress-yy @a 0
block zone history stress-yy @a2 0
block zone history stress-yy @a4 0
block zone history stress-xy @a 0
block zone history stress-xy @a2 0
block zone history stress-xy @a4 0
```

```
;
; --- cycle until new equilibrium ---
;
block solve ratio @conv_rat
;
; --- save 'the equilibrium state ---
;
model save @nam_t1e
;
; --- excavate ---
;
block delete range ann cente 0 0 rad 0 @a
;
; --- set boundary elements ---
;
block boundary-element gen
block boundary-element material 1
block boundary-element fix 0 @_ref_poin @ref_poin 0
block boundary-element stiff
;
; --- cycle until new equilibrium ---
;
block solve ratio @conv_rat
;
; --- check equilibrium ---
;
;pl pe his 0
;pl pe his 1
;pl pe his 2 3 4 5 6 7
;
call 'comp_a.fis'
model save @nam_t1f
;
;-------------------------------------------------------------
; Part C:restart from first equilibrium (before excavation)
; Select the appropriate filename.
;-------------------------------------------------------------
;
model restore @nam_t1e
model title @run_t2
;
; --- excavate ---
;
block delete range ann center 0 0 rad 0 @a
;
; --- set support and its properties ---
```

```
;
block struct liner create by-angle center 0 0 angle-begin=0 ...
  point=@ndiv mat-liner=2
block struct liner prop m=2 den = 1 ...
  coupling-stiffness-normal=@c_stiff ...
  coupling-stiffness-shear=@c_stiff ...
  coupling-coh=@large  coupling-tens=@large thick=@thick_s
;
; --- plane stress values ---
;
;bl st lin prop m=2 st_den=1 st_ymod=@young_s st_prat=@poisson_s
;
; --- plane strain values ---
;
bl st lin prop m=2 den=1 youngs=@young_sp poisson=@poisson_sp
;
; --- set boundary elements ---
;
block boundary-element gen
block boundary-element material 1
block boundary-element fix 0 @_ref_poin @ref_poin 0
block boundary-element stiff
;
; --- cycle until new equilibrium ---
;
block solve ratio @conv_rat
;
; --- check equilibrium ---
;
;pl pe his 0
;pl pe his 1
;pl pe his 2 3 4 5 6 7
;
; --- save 'the supported elastic solution state.
;
model save @nam_t2e
call 'comp_c.fis'
model save @nam_t2e
;---------------------------------------
;
; Start afresh for elastic-plastic model
;
;---------------------------------------
; Circular tunnel. (Part B and Part D)
;---------------------------------------
;
```

```
; --- use the saved geometry state ---
;
model restore @nam_t1g
model title @run_t3
;
; --- Mohr-Coulumb behavior ---
;
block change model 3
block mechanical damping global
;
; --- give state of stresses (hydrostat.) ---
;
block edge apply stress @sigma_2 0 @sigma_2
block insitu stress @sigma_2 0 @sigma_2 stress-ZZ @sigma_zz
;
; --- set material properties ---
;
bl prop mat 1 density 1 bulk @bulk_m shear @shear_m
bl prop mat 1 cohesion @coh_v tension @large friction @phi dilation @ksi
;
; --- glue joints ---
;
bl cont prop mat 1 st-n @j_stiff st-s @j_stiff coh @large ten @large
;
; --- histories ---
;
history interval 20
block gridpoint history velocity-maximum
block gridpoint history disp-x @a 0
block gridpoint history disp-x @a2 0
block gridpoint history disp-x @a4 0
block gridpoint history disp-y @a 0
block gridpoint history disp-y @a2 0
block gridpoint history disp-y @a4 0
block zone history stress-xx @a 0
block zone history stress-xx @a2 0
block zone history stress-xx @a4 0
block zone history stress-yy @a 0
block zone history stress-yy @a2 0
block zone history stress-yy @a4 0
block zone history stress-xy @a 0
block zone history stress-xy @a2 0
block zone history stress-xy @a4 0
;
; --- cycle until new equilibrium ---
;
```

```
block solve ratio @conv_rat_10
;
; --- check equilibrium ---
;
;pl pe his 0
;pl pe his 1
;pl pe his 2 3 4 5 6 7
;
; --- save 'the equilibrium state ---
;
model save @nam_t3e
;
; --- excavate ---
;
block delete range ann center 0 0 rad 0 @a
;
; --- set boundary elements ---
;
block boundary-element gen
block boundary-element material 1
block boundary-element fix 0 @_ref_poin @ref_poin 0
block boundary-element stiff
;
; --- cycle until new equilibrium ---
;
block solve ratio @conv_rat_10
call 'comp_b.fis'
model save @nam_t3f
;
; --- check equilibrium ---
;
;pl pe his 0
;pl pe his 1
;pl pe his 2 3 4 5 6 7
;-------------------------------------------------------------
; Part D:restart from first equilibrium (before excavation)
; Select the appropriate filename.
;-------------------------------------------------------------
model restore @nam_t3e
model title @run_t4
;
; --- excavate ---
;
block delete range ann center 0 0 rad 0 @a
;
; --- set support and its properties ---
```

```
;
block struct liner create by-angle center 0,0 angle-begin=0 ...
  points=@ndiv mat-liner=2
block struct liner prop m=2 den = 1 ...
  coupling-stiffness-normal=@c_stiff ...
  coupling-stiffness-shear=@c_stiff ...
  coupling-coh=@large coupling-tens=@large thick=@thick_s
;
; --- plane stress values ---
;
;bl st lin prop m=2 den=1 youngs=young_s poisson=poisson_s
;
; --- plane strain values ---
;
bl st lin prop m=2 den=1 youngs=@young_sp poisson=@poisson_sp
;
; --- set boundary elements ---
block boundary-element gen
block boundary-element material 1
block boundary-element fix 0 @_ref_poin @ref_poin 0
block boundary-element stiff
;
; --- cycle until new equilibrium ---
;
block solve ratio @conv_rat_10
;
; --- check equilibrium ---
;
;pl pe his 0
;pl pe his 1
;pl pe his 2 3 4 5 6 7
model save @nam_t4f
return
```

Example 5.4 COMP_A.FIS

```
; fish file to compare the radial, and hoop (tangential) stresses
; and displacements between UDEC and an analytic solution.
; Elastic solution.
; use for ctel5z.sav and ctel10z.sav
;
fish define compare
;
; --- find zones along the requested section line ---
;
```

```
n_max = 50
array sec_zo(50)
diff_min = 1e30
iblock = bl.head
loop while iblock # 0
  izone = bl.zone(iblock)
  loop while izone # 0
    x = block.zone.pos.x(izone)
    y = block.zone.pos.y(izone)
    if x > 0 then
      if y < 0 then
        theta = math.atan2(y,x)
        diff  = math.abs(theta*180./math.pi+alpha)
        if diff<diff_min then
          diff_min = diff
          theta_min = theta
        end_if
      end_if
    end_if
    izone = block.zone.next(izone)
  end_loop
  iblock = bl.next(iblock)
end_loop
sinth = math.sin(theta_min)
costh = math.cos(theta_min)
i = 0
iblock = block.head
loop while iblock # 0
  izone = bl.zone(iblock)
  loop while izone # 0
    x = block.zone.pos.x(izone)
    y = block.zone.pos.y(izone)
    if x > 0 then
      if y < 0 then
       theta = math.atan2(y,x)
       diff  = math.abs(theta-theta_min)
       if diff<0.1/a_z then
         i = i+1
         if i < n_max then
           sec_zo(i) = izone
         end_if
       end_if
      end_if
    end_if
    izone = block.zone.next(izone)
  end_loop
```

```
    iblock = bl.next(iblock)
  end_loop
  n_sec = i
;
; --- sort zones with increasing radius ---
;
  loop j (1,n_sec-1)
    loop i (1,n_sec-1)
      izone = sec_zo(i)
      x = block.zone.pos.x(izone)
      y = block.zone.pos.y(izone)
      rad = math.sqrt(x*x+y*y)
      izone1 = sec_zo(i+1)
      x1 = block.zone.pos.x(izone1)
      y1 = block.zone.pos.y(izone1)
      rad1 = math.sqrt(x1*x1+y1*y1)
      if rad1<rad then
        sec_zo(i)   = izone1
        sec_zo(i+1) = izone
      end_if
    end_loop
  end_loop
;
; --- calculate analytic solution and retrieve numerical solution
;
  loop i (1,n_sec)
    izone = sec_zo(i)
    x = block.zone.pos.x(izone)
    y = block.zone.pos.y(izone)
    rad = math.sqrt(x*x+y*y)/a
    a_rad = 1/rad
    theta        = math.atan2(y,x)
    costh        = math.cos(theta)
    sinth        = math.sin(theta)
    theta        = theta+alpha*math.degrad
    cos_2th      = math.cos(2*theta)
    sin_2th      = math.sin(2*theta)
    part1 = (1.-a_rad*a_rad)
    part2 = (1.-4.*a_rad*a_rad+3.*a_rad*a_rad*a_rad*a_rad)*cos_2th
    table.y(1,i) = part1+part2*(1.-k_min)/(1.+k_min)
    part1 = (1.+a_rad*a_rad)
    part2 = (1.+3.*a_rad*a_rad*a_rad*a_rad)*cos_2th
    table.y(2,i) = part1-part2*(1.-k_min)/(1.+k_min)
    part2 = (1.+2.*a_rad*a_rad-3.*a_rad*a_rad*a_rad*a_rad)*sin_2th
    table.y(3,i) = -part2*(1.-k_min)/(1.+k_min)
    part1 = a_rad
```

```
    part2 = a_rad*(4.*(1.-poisson)-a_rad*a_rad)*cos_2th
    dis_r = part1+(1.-k_min)*part2/(1.+k_min)
    table.y(4,i) = dis_r
    part2 = a_rad*(2.*(1.-2.*poisson)+a_rad*a_rad)*sin_2th
    dis_t = -(1.-k_min)*part2/(1.+k_min)
    table.y(5,i) = dis_t
    table.y(6,i) = math.sqrt(dis_r*dis_r+dis_t*dis_t)
    table.x(1,i) = rad
    table.x(2,i) = rad
    table.x(3,i) = rad
    table.x(4,i) = rad
    table.x(5,i) = rad
    table.x(6,i) = rad
;
    str_sc = 0.5*(sigma_1+sigma_2)
    dis_sc = 0.5*str_sc*a/shear_m
;
    sig_x  = bl.zo.str.xx(izone)/str_sc
    sig_y  = bl.zo.str.yy(izone)/str_sc
    sig_xy = bl.zo.str.xy(izone)/str_sc
;
    igp1 = bl.zone.gp(izone,1)
    igp2 = bl.zone.gp(izone,2)
    igp3 = bl.zone.gp(izone,3)
;
    u_x  = 0.33333*(bl.gp.disp.x(igp1)+bl.gp.disp.x(igp2)+ ...
           bl.gp.disp.x(igp3))/dis_sc
    u_y  = 0.33333*(bl.gp.disp.y(igp1)+bl.gp.disp.y(igp2)+ ...
           bl.gp.disp.y(igp3))/dis_sc
;
    sig_r = sig_x*costh*costh+2.*sig_xy*sinth*costh+sig_y*sinth*sinth
    sig_t = sig_x*sinth*sinth-2.*sig_xy*sinth*costh+sig_y*costh*costh
    sig_rt = (sig_y-sig_x)*sinth*costh+sig_xy*(costh*costh-sinth*sinth)
;
    u_r = u_x*costh+u_y*sinth
    u_t = u_y*costh-u_x*sinth
;
    table.x(11,i) = rad
    table.y(11,i) = sig_r
    table.x(12,i) = rad
    table.y(12,i) = sig_t
    table.x(13,i) = rad
    table.y(13,i) = sig_rt
    table.x(14,i) = rad
    table.y(14,i) = u_r
    table.x(15,i) = rad
```

```
    table.y(15,i) = u_t
    table.x(16,i) = rad
    table.y(16,i) = math.sqrt(u_r*u_r+u_t*u_t)
  end_loop
end
@compare
table 1 label 'Radial Stress - Analytic'
table 2 label 'Hoop Stress   - Analytic'
table 3 label 'Shear Stress  - Analytic'
table 4 label 'Radial Displacement - Analytic'
table 5 label 'Hoop Displacement   - Analytic'
table 11 label 'Radial Stress - UDEC'
table 12 label 'Hoop Stress   - UDEC'
table 13 label 'Shear Stress  - UDEC'
table 14 label 'Radial Displacement - UDEC'
table 15 label 'Hoop Displacement   - UDEC'
ret
```

Example 5.5 COMP_B.FIS

```
; fish file to compare the radial and hoop stresses between
; UDEC and an analytic solution.
; Plastic solution.
; use for  ctep5z.sav  and ctep10z.sav
;
fish define compare
;
; --- find zones along the requested section line ---
;
  n_max = 50
  array sec_zo(50)
  diff_min = 1e30
  iblock = bl.head
  loop while iblock # 0
    izone = bl.zone(iblock)
    loop while izone # 0
      x = bl.zone.pos.x(izone)
      y = bl.zone.pos.y(izone)
      if x > 0 then
        if y < 0 then
          theta = math.atan2(y,x)
          diff  = math.abs(theta*180./math.pi+alpha)
          if diff<diff_min then
            diff_min = diff
            theta_min = theta
          end_if
        end_if
      end_if
      izone = bl.zone.next(izone)
    end_loop
    iblock = bl.next(iblock)
  end_loop
  sinth = math.sin(theta_min)
  costh = math.cos(theta_min)
  i = 0
  iblock = bl.head
  loop while iblock # 0
    izone = bl.zone(iblock)
    loop while izone # 0
      x = bl.zone.pos.x(izone)
      y = bl.zone.pos.y(izone)
      if x > 0 then
        if y < 0 then
```

```
            theta = math.atan2(y,x)
            diff  = math.abs(theta-theta_min)
            if diff<0.1/a_z then
              i = i+1
              if i < n_max then
                sec_zo(i) = izone
              end_if
            end_if
          end_if
        end_if
        izone = bl.zone.next(izone)
      end_loop
      iblock = bl.next(iblock)
    end_loop
    n_sec = i
;
; --- sort zones with increasing radius ---
;
    loop j (1,n_sec-1)
      loop i (1,n_sec-1)
        izone = sec_zo(i)
        x = bl.zone.pos.x(izone)
        y = bl.zone.pos.y(izone)
        rad = math.sqrt(x*x+y*y)
;
        izone1 = sec_zo(i+1)
        x1 = bl.zone.pos.x(izone1)
        y1 = bl.zone.pos.y(izone1)
        rad1 = math.sqrt(x1*x1+y1*y1)
        if rad1<rad then
  sec_zo(i)   = izone1
  sec_zo(i+1) = izone
        end_if
      end_loop
    end_loop
;
; --- calculate analytic solution and retrieve numerical solution
;
    sinphi = math.sin(phi*math.degrad)
    cosphi = math.cos(phi*math.degrad)
    Kp = (1.+sinphi)/(1.-sinphi)
    plas_a = 2.*(sigma_q*(Kp-1.)+1.)/(Kp+1.)
    plas_a = math.exp(math.ln(plas_a)/(Kp-1.))
    sig_o  = -(2.*sigma_q-1.)/(Kp+1.)
;
    loop i (1,n_sec)
```

```
    izone = sec_zo(i)
    x = bl.zone.pos.x(izone)
    y = bl.zone.pos.y(izone)
    rad_a = math.sqrt(x*x+y*y)/a
    table.x(1,i) = rad_a
    table.x(2,i) = rad_a
;
    if rad_a < plas_a then
      pow_r = (Kp-1)*math.ln(rad_a)
      pow_r = math.exp(pow_r)
      table.y(1,i) = (1.-pow_r)/(Kp-1.)
      table.y(2,i) = (1.-Kp*pow_r)/(Kp-1.)
    else
      plas_r = plas_a/rad_a
      table.y(1,i) = -sigma_q+(sigma_q+sig_o)*plas_r*plas_r
      table.y(2,i) = -sigma_q-(sigma_q+sig_o)*plas_r*plas_r
    end_if
;
    sig_x  = bl.zo.str.xx(izone)/q
    sig_y  = bl.zo.str.yy(izone)/q
    sig_xy = bl.zo.str.xy(izone)/q
;
    sig_r = sig_x*costh*costh+2.*sig_xy*sinth*costh+sig_y*sinth*sinth
    sig_t = sig_x*sinth*sinth-2.*sig_xy*sinth*costh+sig_y*costh*costh
    sig_rt = (sig_y-sig_x)*sinth*costh+sig_xy*(costh*costh-sinth*sinth)
;
    table.x(11,i) = rad_a
    table.y(11,i) = sig_r
    table.x(12,i) = rad_a
    table.y(12,i) = sig_t
  end_loop
;
  plarea = 0.
  toarea = 0.
  iblock = bl.head
  loop while iblock # 0
    izone = bl.zone(iblock)
    loop while izone # 0
      gp1 = bl.zone.gp(izone,1)
      gp2 = bl.zone.gp(izone,2)
      gp3 = bl.zone.gp(izone,3)
      x1  = bl.gp.pos.x(gp1)
      x2  = bl.gp.pos.x(gp2)
      x3  = bl.gp.pos.x(gp3)
      y1  = bl.gp.pos.y(gp1)
      y2  = bl.gp.pos.y(gp2)
```

```
      y3  = bl.gp.pos.y(gp3)
      zoarea = math.abs(0.5*((x2-x1)*(y3-y1)-(x3-x1)*(y2-y1)))
;
; --- calculating total area of the model and area of yielding zones ---
;
      if bl.zone.state(izone) # 0 then
        plarea = plarea+zoarea
      end_if
      toarea = toarea+zoarea
      izone = bl.zone.next(izone)
    end_loop
    iblock = bl.next(iblock)
  end_loop
  aplas_a = math.sqrt(plarea*(mod_size*mod_size/(a*a)-1.)/toarea+1.)
end
@compare
table 1  label 'Radial Stress - Analytic'
table 2  label 'Hoop Stress   - Analytic'
table 11 label 'Radial Stress - UDEC'
table 12 label 'Hoop Stress   - UDEC'
ret
```

Example 5.6 COMP_C.FIS

```
; fish file to compare the Axial force and bending moment between
; UDEC and an analytic solution.
; use for ctsu5ze.sav and ctsu10ze.sav
;
; --- Define the constants ---
fish define cons
   br = bulk_m
   sr = shear_m
   E  = young
   nu = poisson
   nu_s = poisson_s
   Es = young_s
   Ar = thick_s
   II = Ar * Ar * Ar / 12.
   cs = E * a  * (1. - nu_s*nu_s)/ (Ar * Es * (1. - nu*nu))
   fs = E * a*a*a * (1. - nu_s*nu_s) / (Es * II * (1. - nu*nu))
   a0 = cs * fs * (1. - nu) / (cs + fs + cs * fs * (1.-nu))
   be = ((6.+fs)*cs*(1.-nu)+2.*fs*nu)/(3.*fs+3.*cs+2.*cs*fs*(1.-nu))
   b2 = cs*(1.-nu)*0.5/(cs*(1.-nu)+4.*nu-6.*be-3.*be*cs*(1.-nu))
   a2 = be*b2
   c1 = 0.5*(1.+1./k_min)*(1.-a0)
```

```
   c2 = 0.5*(1.-1./k_min)*(1.+2.*a2)
   c3 = 0.25*(1.-1./k_min)*(1.-2.*a2+2.*b2)
   n0 = sigma_2 * a
   m0 = n0 * a
end
@cons
call 'str.fin'
fish define compare
   nin = bl.str.elem.head
   i = 0
   ern = 0.0
   erm = 0.0
   loop while nin # 0
      p1 = xmem(index(nin + $SELN1))
      p2 = xmem(index(nin + $SELN2))
      tanteta = (fmem(p2+$SNDY)+fmem(p1+$SNDY))
      tanteta = tanteta/(fmem(p2+$SNDX)+fmem(p1+$SNDX))
      teta = math.atan(tanteta)+alpha*math.degrad
      if teta >= -3*math.pi/ndiv then
        if teta <=(ndiv/2+3)*math.pi/ndiv then
  i = i + 1
          tetad = teta / math.degrad
          temp1 = c1 + c2 * math.cos(2.*teta)
  temp2 = fmem(nin+$SELFAX) / n0
          table.x(1,i) = tetad
          table.y(1,i) = temp1
          table.x(11,i) = tetad
          table.y(11,i) = temp2
          temp3 = 100. * math.abs(temp1 - temp2)
          ern = math.max(temp3,ern)
  tanteta =  fmem(p1+$SNDY) / fmem(p1+$SNDX)
          teta = math.atan(tanteta)+alpha*math.degrad
          tetad = teta / math.degrad
          temp1 = c3 * math.cos(2.*teta)
  temp2 = -fmem(nin+$SELM1) / m0
          table.x(2,i) = tetad
          table.y(2,i) = temp1
          table.x(12,i) = tetad
          table.y(12,i) = temp2
          if math.abs(tetad - 45.) > 45.e-3 then
             temp3 = 100. * math.abs(temp1 - temp2)
  else
     temp3 = 0.0
  end_if
          erm = math.max(temp3,erm)
end_if
```

```
      end_if
      nin = xmem(index(nin+$SELNEXT))
   end_loop
;
; --- sort tables ---
;
   loop j (1,i-1)
     loop k (1,i-1)
       if table.x(1,k) > table.x(1,k+1) then
         x1  = table.x(1,k+1)
         x2  = table.x(2,k+1)
         x11 = table.x(11,k+1)
         x12 = table.x(12,k+1)
         y1  = table.y(1,k+1)
         y2  = table.y(2,k+1)
         y11 = table.y(11,k+1)
         y12 = table.y(12,k+1)
;
         table.x(1,k+1)  = table.x(1,k)
         table.x(2,k+1)  = table.x(2,k)
         table.x(11,k+1) = table.x(11,k)
         table.x(12,k+1) = table.x(12,k)
         table.y(1,k+1)  = table.y(1,k)
         table.y(2,k+1)  = table.y(2,k)
         table.y(11,k+1) = table.y(11,k)
         table.y(12,k+1) = table.y(12,k)
;
         table.x(1,k)  = x1
         table.x(2,k)  = x2
         table.x(11,k) = x11
         table.x(12,k) = x12
         table.y(1,k)  = y1
         table.y(2,k)  = y2
         table.y(11,k) = y11
         table.y(12,k) = y12
       end_if
     end_loop
   end_loop
   ern = ern / table.y(1,1)
   erm = erm / table.y(2,1)
end
@compare
table 1  label 'Liner Thrust - Analytic'
table 2  label 'Liner Moment - Analytic'
table 11 label 'Liner Thrust - UDEC'
table 12 label 'Liner Moment - UDEC'
```

```
ret
```