```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% 2D Example: solve strain field in pressurized cylindrical tube
%             elastic deformation only
%
% Wei Cai caiwei@stanford.edu
% Created       03/24/2013
% Last Modified 04/29/2013
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

mu = 100;
nu = 0.3;
lambda = 2*mu*nu/(1-2*nu);

a = 1; b = 2;
dr = 0.05;
r = [a:dr:b];
p0 = 1;

eps_rr = zeros(size(r));
eps_qq = zeros(size(r));

Niter = 10000;
plotfreq = 1000; % 50;

param = [a b dr mu nu p0];

F0 = eqns_cylind_tube_elast([eps_rr, eps_qq],param);
%options = optimset('Display','iter','TolFun',1e-8);
options = optimset('Display','off','TolFun',1e-8);
sol = fsolve('eqns_cylind_tube_elast', [eps_rr, eps_qq], options, param);
eps_rr = sol(1:end/2); eps_qq = sol(end/2+1:end);
F1 = eqns_cylind_tube_elast([eps_rr, eps_qq],param);

% (Alternative method using cgrelax, require gradient evaluation)
% [U dU_depsrrqq] = grad_cylind_tube_elast([eps_rr, eps_qq],param);
% acc = 1e-6; maxfn = 40000; dfpred = 1e-5; n = length([eps_rr, eps_qq]);
% [U,eps_rrqq_rlx] = cgrelax('grad_cylind_tube_elast',n,acc,maxfn,dfpred,...
%                            [eps_rr,eps_qq],param);
% eps_rr = eps_rrqq_rlx(1:end/2); eps_qq = eps_rrqq_rlx(end/2+1:end);

% stress field
sig_rr = (lambda + 2*mu) * eps_rr + lambda * eps_qq;
sig_qq = (lambda + 2*mu) * eps_qq + lambda * eps_rr;
sig_zz = lambda * (eps_rr + eps_qq);

pp=p0*a^2/(b^2-a^2); E = 2*mu*(1+nu);
sig_rr_anl = pp*(1-b^2./r.^2);
sig_qq_anl = pp*(1+b^2./r.^2);
eps_rr_anl = (1-nu^2)/E*sig_rr_anl - nu*(1+nu)/E*sig_qq_anl;
eps_qq_anl = (1-nu^2)/E*sig_qq_anl - nu*(1+nu)/E*sig_rr_anl;
```

```
% plot solution
fs = 17;
figure(1);
plot(r, eps_rr /p0*mu, '.', r, eps_qq/p0*mu, 'd', ...
     r, eps_rr_anl/p0*mu, r, eps_qq_anl/p0*mu);
xlim([0 max(r)*1.5]);
set(gca,'FontSize',fs);
xlabel('r'); ylabel('\epsilon  \mu / p_0');
legend('\epsilon_{rr}^{num}','\epsilon_{\theta\theta}^{num}',...
       '\epsilon_{rr}^{anl}','\epsilon_{\theta\theta}^{anl}');
figure(2);
plot(r, sig_rr/p0, '.', r, sig_qq/p0, 'o', r, sig_rr_anl/p0, r, sig_qq_anl/p0);
xlim([0 max(r)*1.5]);
set(gca,'FontSize',fs);
xlabel('r'); ylabel('\sigma / p_0');
legend('\sigma_{rr}^{num}','\sigma_{\theta\theta}^{num}',...
       '\sigma_{rr}^{anl}','\sigma_{\theta\theta}^{anl}');
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% 2D Example: solve strain field in pressurized cylindrical tube
%             elastic deformation only
% Construct the equations to be solved by cylind_tube_elast.m
%
% Wei Cai caiwei@stanford.edu
% Created       03/24/2013
% Last Modified 04/29/2013
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function F = eqns_cylind_tube_elast(vars, param)
% F(1:N-1): equilibrium equation
% F(N:2N-2): compatibility equation
% F(2N-1:2N): boundary condition

a = param(1);
b = param(2);
dr = param(3);
mu = param(4);
nu = param(5);
p0 = param(6);

N = length(vars)/2; F = zeros(1,2*N);
eps_rr = vars(1:N); eps_qq = vars(N+1:end);

lambda = 2*mu*nu/(1-2*nu);
r = [a:dr:b]; r_avg = (r(2:end) + r(1:end-1)) / 2;

% stress field
sig_rr = (lambda + 2*mu) * eps_rr + lambda * eps_qq;
sig_qq = (lambda + 2*mu) * eps_qq + lambda * eps_rr;
sig_zz = lambda * (eps_rr + eps_qq);

dsigrrdr = (sig_rr(2:end) - sig_rr(1:end-1)) / dr;
sig_rr_avg = (sig_rr(2:end) + sig_rr(1:end-1)) / 2;
sig_qq_avg = (sig_qq(2:end) + sig_qq(1:end-1)) / 2;

% equilibrium equation
F(1:N-1) = dsigrrdr + (sig_rr_avg-sig_qq_avg)./r_avg;

% strain field
depsqqdr = (eps_qq(2:end) - eps_qq(1:end-1)) / dr;
eps_qq_avg = (eps_qq(2:end) + eps_qq(1:end-1)) / 2;
eps_rr_avg = (eps_rr(2:end) + eps_rr(1:end-1)) / 2;

% compatibility condition
F(N:2*N-2) = depsqqdr - (eps_rr_avg-eps_qq_avg)./r_avg;

% boundary condition
F(2*N-1:2*N) = [sig_rr(1) + p0, sig_rr(end)];
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% 2D Example: solve strain field in pressurized cylindrical tube
%              pressure large enough to cause plastic deformation
%
% Wei Cai caiwei@stanford.edu
% Created       03/24/2013
% Last Modified 04/29/2013
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

mu = 100;
nu = 0.3;
k = 4;                      % yield stress / sqrt(3)
E = 2*mu*(1+nu);            % Young's modulus
K = 2*mu*(1+nu)/3/(1-2*nu); % bulk modulus

a = 1; b = 2;
dr = 0.02;  drho = dr;
r = [a:dr:b]; rho = r;
p0 = 1;

S_rr_sol   = zeros(length(r), length(rho));
S_qq_sol   = zeros(length(r), length(rho));
sig_rr_sol = zeros(length(r), length(rho));
sig_qq_sol = zeros(length(r), length(rho));
sig_zz_sol = zeros(length(r), length(rho));
eps_rr_sol = zeros(length(r), length(rho));
eps_qq_sol = zeros(length(r), length(rho));

% analytic solution in elastic region
for i=1:length(r),
    for j=1:length(rho),
        % limit to elastic region
        if r(i) >= rho(j),
            ppp=k*((1-2*nu)^2/3 + b^4/rho(j)^4)^(-1/2);
            sig_rr_1 = ppp*(1-b^2./r(i).^2);
            sig_qq_1 = ppp*(1+b^2./r(i).^2);
            sig_zz_1 = nu*(sig_rr_1 + sig_qq_1);
            S_rr_1 = sig_rr_1 - (sig_rr_1+sig_qq_1+sig_zz_1)/3;
            S_qq_1 = sig_qq_1 - (sig_rr_1+sig_qq_1+sig_zz_1)/3;
            eps_rr_1 = (1-nu^2)/E*sig_rr_1 - nu*(1+nu)/E*sig_qq_1;
            eps_qq_1 = (1-nu^2)/E*sig_qq_1 - nu*(1+nu)/E*sig_rr_1;
            S_rr_sol(i,j)   = S_rr_1;
            S_qq_sol(i,j)   = S_qq_1;
            sig_rr_sol(i,j) = sig_rr_1;
            sig_qq_sol(i,j) = sig_qq_1;
            sig_zz_sol(i,j) = sig_zz_1;
            eps_rr_sol(i,j) = eps_rr_1;
            eps_qq_sol(i,j) = eps_qq_1;
        end
    end
end
```

```
% find solution in plastic region
disp('find solution in plastic region...');
for j=2:length(rho),
    disp(sprintf('j = %d / %d',j, length(rho)));
    for i=j-1:-1:1,
        % find solution at (r, rho+drho)
        eps_rr_1 = eps_rr_sol(i,j-1); eps_qq_1 = eps_qq_sol(i,j-1);
        S_rr_1   = S_rr_sol(i,j-1);
        eps_rr_2 = eps_rr_sol(i+1,j); eps_qq_2 = eps_qq_sol(i+1,j);
        S_rr_2   = S_rr_sol(i+1,j);
        param = [r(i), dr, mu, nu, k, eps_rr_1, eps_qq_1, ...
                 S_rr_1, eps_rr_2, eps_qq_2, S_rr_2];
        %options = optimset('Display','iter','TolFun',1e-8);
        options = optimset('Display','off','TolFun',1e-8);
        %F0 = eqns_cylind_tube_plast([eps_rr_1, eps_qq_1, S_rr_1], param);
        %sol = fsolve('eqns_cylind_tube_plast', [eps_rr_1, eps_qq_1, S_rr_1], ...
        %             options, param);
        [sol,Fval,exitflag] = fsolve('eqns_cylind_tube_plast', ...
                       [eps_rr_1, eps_qq_1, S_rr_1], options, param);
        switch exitflag
            case 0,  fprintf(['Solution incorrect: ' ...
                 'Maximum number of function evaluations or iterations reached.\n']);
            case -1, fprintf(['Solution incorrect: ' ...
                 'Algorithm terminated by the output function.\n']);
            case -2, fprintf(['Solution incorrect: ', ...
                 'Algorithm seems to be converging to a point that is not a root.\n']);
            case -3, fprintf(['Solution incorrect: ', ...
                 'Trust region radius became too small.\n']);
            case -4, fprintf(['Solution incorrect: ' ...
                 'Line search cannot sufficiently decrease the residual '...
                 'along the current search direction.\n']);
        end
        %F1 = eqns_cylind_tube_plast(sol, param);
        eps_rr_sol(i,j) = sol(1); eps_qq_sol(i,j) = sol(2); S_rr_sol(i,j) = sol(3);
        S_qq_sol(i,j) = (-S_rr_sol(i,j) + sqrt(4*k^2-3*S_rr_sol(i,j)^2))/2;
        sig_rr_sol(i,j) = S_rr_sol(i,j) + K*(eps_rr_sol(i,j)+eps_qq_sol(i,j));
        sig_qq_sol(i,j) = S_qq_sol(i,j) + K*(eps_rr_sol(i,j)+eps_qq_sol(i,j));
        sig_zz_sol(i,j) = -S_rr_sol(i,j)-S_qq_sol(i,j) ...
                        + K*(eps_rr_sol(i,j)+eps_qq_sol(i,j));
    end
end

% verify solution
disp('verify solution in plastic region...');
for j=2:length(rho),
    for i=j-1:-1:1,
        % find solution at (r, rho+drho)
        eps_rr_1 = eps_rr_sol(i,j-1); eps_qq_1 = eps_qq_sol(i,j-1);
        S_rr_1   = S_rr_sol(i,j-1);
        eps_rr_2 = eps_rr_sol(i+1,j); eps_qq_2 = eps_qq_sol(i+1,j);
```

```
        S_rr_2   = S_rr_sol(i+1,j);
        param = [r(i), dr, mu, nu, k, eps_rr_1, eps_qq_1, ...
                 S_rr_1, eps_rr_2, eps_qq_2, S_rr_2];
        F = eqns_cylind_tube_plast([eps_rr_sol(i,j), eps_qq_sol(i,j), ...
                 S_rr_sol(i,j)], param);
        disp(sprintf('i = %d  j = %d  F = %e %e %e',i,j,F(1),F(2),F(3)));
    end
end

% plot solution
fs = 17;
figure(1);
skip = find(r/r(1)==1.2,1,'First')-1;
plot(r/a, sig_rr_sol(:,1:skip:end)/(2*k), '.-');
set(gca,'FontSize',fs);
xlabel('r / a');
ylabel('\sigma_{rr} / (2k)');
legend('\rho/a = 1','1.2','1.4','1.6','1.8','2.0','Location','SouthEast');

figure(2);
plot(r, sig_qq_sol(:,1:skip:end)/(2*k), '.-');
set(gca,'FontSize',fs);
xlabel('r / a');
ylabel('\sigma_{\theta\theta} / (2k)');
legend('\rho/a = 1','1.2','1.4','1.6','1.8','2.0','Location','NorthWest');

figure(3);
plot(r, sig_zz_sol(:,1:skip:end)/(2*k), '.-');
set(gca,'FontSize',fs);
xlabel('r / a');
ylabel('\sigma_{zz} / (2k)');
legend('\rho/a = 1','1.2','1.4','1.6','1.8','2.0','Location','SouthEast');
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% 2D Example: solve strain field in pressurized cylindrical tube
%              pressure large enough to cause plastic deformation
% Construct the equations to be solved by cylind_tube_plast.m
%
% Wei Cai caiwei@stanford.edu
% Created       03/24/2013
% Last Modified 04/29/2013
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function F = eqns_cylind_tube_plast(vars, param)
% F(1): equilibrium equation
% F(2): compatibility equation
% F(3): plastic flow rule

eps_rr = vars(1);
eps_qq = vars(2);
S_rr   = vars(3);

r  = param(1);
dr = param(2); drho = dr;
mu = param(3);
nu = param(4);
k  = param(5);
eps_rr_1 = param(6);
eps_qq_1 = param(7);
S_rr_1   = param(8);
eps_rr_2 = param(9);
eps_qq_2 = param(10);
S_rr_2   = param(11);

K = 2*mu*(1+nu)/3/(1-2*nu); % bulk modulus

F = [0 0 0]';

S_qq   = (-S_rr   + sqrt(4*k^2-3*S_rr^2  ))/2;
S_qq_1 = (-S_rr_1 + sqrt(4*k^2-3*S_rr_1^2))/2;
S_qq_2 = (-S_rr_2 + sqrt(4*k^2-3*S_rr_2^2))/2;

dSrr_dr = (S_rr_2-S_rr)/dr;
dSqq_dr = (S_qq_2-S_qq)/dr;
depsrr_dr = (eps_rr_2-eps_rr)/dr;
depsqq_dr = (eps_qq_2-eps_qq)/dr;

depsrr_drho = (eps_rr - eps_rr_1)/drho;
depsqq_drho = (eps_qq - eps_qq_1)/drho;
dSrr_drho   = (S_rr - S_rr_1)/drho;
dSqq_drho   = (S_qq - S_qq_1)/drho;
```

```
% equilibrium condition
F(1) = dSrr_dr + K*( depsrr_dr + depsqq_dr  ) ...
       + ( (S_rr_2+S_rr)/2 - (S_qq_2+S_qq)/2 ) / (r+dr/2);

% compatibility condition
F(2) = depsqq_dr - ( (eps_rr_2+eps_rr)/2 - (eps_qq_2+eps_qq)/2 ) / (r+dr/2);

% plastic flow rule
F(3) = 2*mu*( (2/3*depsrr_drho - 1/3*depsqq_drho)*(S_qq+S_qq_1)/2   ...
             -(2/3*depsqq_drho - 1/3*depsrr_drho)*(S_rr+S_rr_1)/2 ) ...
         - dSrr_drho*(S_qq+S_qq_1)/2 + dSqq_drho*(S_rr+S_rr_1)/2 ;
```