```
% Axi-DDM: Displacement Discontinuity Method
% for Modeling Axisymmetric Cracks in an Elastic Half-Space.
% Available at http://purl.umn.edu/97344
%
% Authors: Elizaveta Gordeliy and Emmanuel Detournay
% Department of Civil Engineering, University of Minnesota, Minneapolis,
% USA
%
% Last modification: June 2011
%
% Reference:
% [1] E. Gordeliy and E. Detournay. Displacement Discontinuity Method
% for Modeling Axisymmetric Cracks in an Elastic Half-Space.
% International Journal of Solids and Structures (2011),
% doi:10.1016/j.ijsolstr.2011.05.009.
%
% This work was supported by the National Science Foundation under
% Grant No. 0600058.
%
% This work is licensed under the Creative Commons Attribution-NonCommercial-
% ShareAlike 3.0 License (CC BY-NC-SA 3.0). To view a copy of this license,
% visit http://creativecommons.org
% ------------------------------------------------------------------------
% Model: an axisymmetric crack in a half-space or full space.
%
% Crack geometry is defined in cylindrical coordinates (r,z) by specifying
% coordinates of endpoints for n elements of same length.
%
% The direction of travel on the crack surface is from element #1 towards
% element #n. If the crack intersects the z-axis, specify the corresponding
% endpoint of element #1 on the z-axis.
%
% The tangential unit vector on the crack surface is oriented
% in the direction of travel. The normal unit vector is oriented at
% an angle pi/2 counterclockwise from the tangential unit vector [1].
%
% Loading may include far-field stress, uniform pressure on crack faces,
% and user-defined tractions at crack faces as functions of (r,z).
%
% ------------------------------------------------------------------------
% AxiDDM.m uses MATLAB function elliptic3.m by Igor Moiseev, available at
% http://www.mathworks.com/matlabcentral/fileexchange/8805, for computation
% of the complete elliptic integral of the third kind.
%
% ------------------------------------------------------------------------
% User-defined parameters:
%
% E     = Young's modulus
% nu    = Poisson's ratio
%
% sigrr, sigrz, sigzz = far-field stresses
%
% press = uniform pressure on crack faces
% tn    = user-defined normal traction at crack faces: function of (r,z)
% ts    = user-defined shear traction at crack faces: function of (r,z)
%
% psip  = half-space indicator; psip = 0 for full space
%                               psip = 1 for half-space
```

```
%
% edge  = crack edge indicator; edge = 1 if crack has one edge
%                                    (e.g. spherical crack)
%                               edge = 2 if crack has two edges
%                                    (e.g. cylindrical crack in full space)
%
% n     = number of elements to discretize crack (n > 1)
% Re    = element endpoints r-coordinates: array of size (n+1,1);
%         for crack intersecting z-axis, element #1 must originate from z-axis
% Ze    = element endpoints z-coordinates: array of size (n+1,1);
%
% ------------------------------------------------------------------------
% Constants and variables:
%
% kappa = Kolosov's constant
% Ep    = plane strain Young's modulus
% alpha = edge correction coefficient
% cSIF  = SIF scaling coefficient
%
% dL    = element size
% Rm    = element midpoints r-coordinates
% Zm    = element midpoints z-coordinates
% Theta = element inclinations to r-axis
%
% UTN   = user-defined normal tractions at element midpoints
% UTS   = user-defined shear tractions at element midpoints
% TN    = total normal tractions at element midpoints
% TS    = total shear tractions at element midpoints
%
% Un    = crack opening at element midpoints for total tractions
% Us    = shear displacement jump at element midpoints for total tractions
%
% Mnn, Mns, Msn, Mss = influence matrices
%
% K1_n  = mode I stress intensity factor for crack edge at element #n
% K2_n  = mode II stress intensity factor for crack edge at element #n
% phi_n = deflection angle from crack edge at element #n
%
% K1_1  = mode I stress intensity factor for crack edge at element #1
% K2_1  = mode II stress intensity factor for crack edge at element #1
% phi_1 = deflection angle from crack edge at element #1
% ------------------------------------------------------------------------
% Results stored in file 'ResultsAxiDDM.dat'
%
% Output format:
%
% nu E Ep psip edge n
% sigrr sigrz sigzz press
% 1 Rm(1) Zm(1) UTN(1) UTS(1) TN(1) TS(1) Un(1) Us(1)
% ...
% n Rm(n) Zm(n) UTN(n) UTS(n) TN(n) TS(n) Un(n) Us(n)
% n K1_n K2_n phi_n(degrees)
% 1 K1_1 K2_1 phi_1(degrees) % if crack has two edges
% CPU(sec) CPU(min)
% ------------------------------------------------------------------------
clear
cput=cputime;
global psip kappa
```

```
% ------------------------------------------------------------------------
% Define parameters
% ------------------------------------------------------------------------
% Elastic moduli
E=0.96;
nu=0.2;
% Far-field stress (tension positive)
sigrr=0;
sigrz=0;
sigzz=1;
% Uniform pressure on crack faces
press=0;
% User-defined tractions at crack faces in cylindrical coordinates (r,z):
% replace '(0*r.^2 + 0*z.^2)' by input function of vectors r and z
tn= @(r,z) (0*r.^2 + 0*z.^2); % normal traction as function of (r,z)
ts= @(r,z) (0*r.^2 + 0*z.^2); % shear traction as function of (r,z)
% ------------------------------------------------------------------------
% Define crack geometry and discretization:
% choose (uncomment) either one from Examples 1 - 4, or replace with other
% ------------------------------------------------------------------------
% Example 1: spherical cap crack in full space (Fig. 7 in [1])
psip=0;            % crack is in full space
edge=1;            % crack has one edge
n=50;              % define number of elements for crack discretization

c=100;               % c = radius of sphere, Fig. 7 in [1]
angle=asin(0.01);  % angle = asin(a/c), Fig. 7 in [1], where a is the
                   % largest value of r-coordinate on the crack surface

Re=c*sin(angle/n*[0:1:n]');  % define element endpoints r-coordinates
Ze=c-sqrt(c^2-Re.^2);        % define element endpoints z-coordinates
% ------------------------------------------------------------------------
% % Example 2: spherical cap crack in half-space
% psip=1;            % crack is in half-space
% edge=1;            % crack has one edge
% n=100;             % define number of elements for crack discretization
%
% c=2;               % c = radius of sphere, Fig. 7 in [1]
% angle=pi/6;        % angle = asin(a/c), Fig. 7 in [1], where a is the
%                    % largest value of r-coordinate on the crack surface
%
% Re=c*sin(angle/n*[0:1:n]');  % define element endpoints r-coordinates
% Ze=sqrt(c^2-Re.^2)-c/2*(sqrt(3)-1);  % define element endpoints z-coordinates
% ------------------------------------------------------------------------
% % Example 3: cylindrical crack in full space (Fig. 11 in [1])
% psip=0;            % crack is in full space
% edge=2;            % crack has two edges
% n=200;             % define number of elements for crack discretization
%
% R=1; L=1;          % R = crack radius, L = crack half-length
%
% Re=R*ones(n+1,1);  % define element endpoints r-coordinates
% Ze=2*L/n*[0:1:n]'; % define element endpoints z-coordinates
% ------------------------------------------------------------------------
% % Example 4: surface-breaking cylindrical crack in half-space (Fig. 11 in [1])
% psip=1;            % crack is in half-space
% edge=1;            % crack has one edge
% n=200;             % define number of elements for crack discretization
```

```
%
% R=1; L=1;          % R = crack radius, L = crack half-length
%
% Re=R*ones(n+1,1);  % define element endpoints r-coordinates
% Ze=2*L/n*[0:1:n]'; % define element endpoints z-coordinates
% ------------------------------------------------------------------------
% Solution
% ------------------------------------------------------------------------
% Initialize solution
kappa=3-4*nu;
Ep=E/(1-nu^2);
alpha=1/3;
dL=sqrt((Re(2)-Re(1))^2+(Ze(2)-Ze(1))^2);
Rm=0.5*(Re(1:n)+Re(2:n+1));
Zm=0.5*(Ze(1:n)+Ze(2:n+1));
Theta=atan2(Ze(2:n+1)-Ze(1:n),Re(2:n+1)-Re(1:n));
cSIF=Ep*sqrt(0.5*pi/dL)*3/8;
% Store input
FID1=fopen('ResultsAxiDDM.dat','w');
aux=[nu E Ep psip edge n];
fprintf(FID1,'%1.4f %1.4e %1.4e %1.0f %1.0f %3.0f\n',aux);
aux=[sigrr sigrz sigzz press];
fprintf(FID1,'%1.4e %1.4e %1.4e %1.4e\n',aux);
% ------------------------------------------------------------------------
% Compute tractions
% Compute tractions from far-field stress and applied pressure
TN=-(sigrr*sin(Theta).^2+sigzz*cos(Theta).^2-sigrz.*sin(2*Theta));
TS=-((sigzz-sigrr).*sin(2*Theta)/2+sigrz*cos(2*Theta));
TN=TN-press*ones(n,1);
% Add user-defined tractions
UTN=tn(Rm,Zm);
UTS=ts(Rm,Zm);
TN=TN+UTN;
TS=TS+UTS;
if max(abs(TN),abs(TS))==0
 warning('Total tractions are zero. Solution is stopped.')
 break
end
% ------------------------------------------------------------------------
% Construct influence matrices
[Mnn,Mns,Msn,Mss] = Axi_DDmatrix(n,1,Rm/dL,Zm/dL,Re/dL,Ze/dL,Theta,edge);
Mnn=Ep/dL*Mnn;  Mns=Ep/dL*Mns;
Msn=Ep/dL*Msn;  Mss=Ep/dL*Mss;
Mnn(n,n)=Mnn(n,n)+Ep/(4*dL)*alpha; % correction for edge element #n
Mss(n,n)=Mss(n,n)+Ep/(4*dL)*alpha; % correction for edge element #n
if edge==2
  Mnn(1,1)=Mnn(1,1)+Ep/(4*dL)*alpha; % correction for edge element #1
  Mss(1,1)=Mss(1,1)+Ep/(4*dL)*alpha; % correction for edge element #1
end
% ------------------------------------------------------------------------
% Compute displacement jumps
X0=zeros(2*n,1);
[DD,flag]=bicgstab([[Mnn Mns];[Msn Mss]],[TN;TS],1e-6,1000,[],[],X0);
Un=-DD(1:n);
Us=-DD(n+1:2*n);
% ------------------------------------------------------------------------
% Compute SIFs and deflection angle for crack edge at element #n
K1_n=cSIF*Un(n);
```

```
K2_n=cSIF*Us(n);
phi1=2*atan((K1_n+sqrt(K1_n^2+8*K2_n^2))/(4*K2_n));
phi2=2*atan((K1_n-sqrt(K1_n^2+8*K2_n^2))/(4*K2_n));
SH1=cos(phi1/2)*(K1_n*(cos(phi1/2))^2-3/2*K2_n*sin(phi1));
SH2=cos(phi2/2)*(K1_n*(cos(phi2/2))^2-3/2*K2_n*sin(phi2));
if SH1>SH2
 phi_n=phi1;
else
 phi_n=phi2;
end
% ------------------------------------------------------------------------
% Compute SIFs and deflection angle for crack edge at element #1
% (if crack has two edges)
if edge==2
 K1_1=cSIF*Un(1);
 K2_1=cSIF*Us(1);
 phi1=2*atan((K1_1+sqrt(K1_1^2+8*K2_1^2))/(4*K2_1));
 phi2=2*atan((K1_1-sqrt(K1_1^2+8*K2_1^2))/(4*K2_1));
 SH1=cos(phi1/2)*(K1_1*(cos(phi1/2))^2-3/2*K2_1*sin(phi1));
 SH2=cos(phi2/2)*(K1_1*(cos(phi2/2))^2-3/2*K2_1*sin(phi2));
 if SH1>SH2
     phi_1=phi1;
 else
     phi_1=phi2;
 end
end
% ------------------------------------------------------------------------
% Store results and CPU in sec and min
for ie=1:1:n
 aux=[ie Rm(ie) Zm(ie) UTN(ie) UTS(ie) TN(ie) TS(ie) Un(ie) Us(ie)];
 fprintf(FID1,'%3.0f %1.4e %1.4e %1.4e %1.4e %1.4e %1.4e %1.6e %1.6e\n',aux);
end
aux=[n K1_n K2_n phi_n/pi*180];
fprintf(FID1,'%3.0f %1.6e %1.6e %3.3f\n',aux);
if edge==2
 aux=[1 K1_1 K2_1 phi_1/pi*180];
 fprintf(FID1,'%3.0f %1.6e %1.6e %3.3f\n',aux);
end
cput=cputime-cput
aux=[cput cput/60.0];
fprintf(FID1,'%12.2e %12.2f\n',aux);
fclose(FID1);
% ------------------------------------------------------------------------
% Plot displacement jumps at element midpoints as functions of r and z
fig=1; figure(fig); hold on;
subplot(2,1,1); plot(Rm,Un,'-r');
fig=1; figure(fig); hold on;
subplot(2,1,2); plot(Rm,Us,'-r');
fig=2; figure(fig); hold on;
subplot(2,1,1); plot(Zm,Un,'-r');
fig=2; figure(fig); hold on;
subplot(2,1,2); plot(Zm,Us,'-r');

fig=1; figure(fig); hold on; subplot(2,1,1);
hx=xlabel('$$r$$','fontsize',16); hy=ylabel('$$[u_n]$$','fontsize',16);
set(hx,'interpreter','latex'); set(hy,'interpreter','latex');
fig=1; figure(fig); hold on; subplot(2,1,2);
hx=xlabel('$$r$$','fontsize',16); hy=ylabel('$$[u_s]$$','fontsize',16);
```

```
set(hx,'interpreter','latex'); set(hy,'interpreter','latex');
fig=2; figure(fig); hold on; subplot(2,1,1);
hx=xlabel('$$z$$','fontsize',16); hy=ylabel('$$[u_n]$$','fontsize',16);
set(hx,'interpreter','latex'); set(hy,'interpreter','latex');
fig=2; figure(fig); hold on; subplot(2,1,2);
hx=xlabel('$$z$$','fontsize',16); hy=ylabel('$$[u_s]$$','fontsize',16);
set(hx,'interpreter','latex'); set(hy,'interpreter','latex');
% --------------------------------------------------------------------
% End of file AxiDDM.m
```