```
function [d,u_neg,u_pos,syz,UZ,SXZ] = ...
easy_anti(G,pyz,xb,yb,xe,ye,PXZ,PYZ,X,Y)
%  Two-dimensional displacement discontinuity boundary element
%  script. For anti-plane strain elements parallel to x-axis
with
%  stress boundary conditions.
%  Reference frame: x = vertical, y = horizontal, z = "into
page"
%  G = shear modulus                                  constant
%  pyz = shear stress on boundary elements,  [1xn] row vector
%  xb = starting element endpt x-coordinate, [1xn] row vector
%  yb = starting element endpt y-coordinate, [1xn] row vector
%  xe = ending element endpt x-coordinate,   [1xn] row vector
%  ye = ending element endpt y-coordinate,   [1xn] row vector
%  PXZ, PYZ = ambient shear stress,     [1xn] row vector or a
constant
%  X,Y = gridpoints (found using "meshgrid"),     [pxq] array
%  Last revised on 2/22/03

%  FORMAT BOUNDARY ELEMENT ARRAYS AS ROW VECTORS
     xb = (xb(:))'; yb = (yb(:))'; xe = (xe(:))'; ye = (ye(:))';
%  CALCULATE BOUNDARY ELEMENT MIDPOINTS AND HALF-LENGTHS
%  (lower case)
     xm   = (xb + xe)/2; % element midpoints,     [1xn] row
vector
     ym   = (yb + ye)/2; % element midpoints,     [1xn] row
vector
     a    =  xe-xm;      % half-lengths,          [1xn] row
vector
%  ADJUST STRESS BOUNDARY CONDITIONS TO ACCOUNT FOR AMBIENT
FIELD
%  (The remote stress will be added back at the end)
     bc = (pyz - PYZ)*ones(size(xb(:)));         % bc = [nx1]
vector
%  COMPUTE INFLUENCE COEFFICIENTS FOR ELEMENT-ELEMENT
INTERACTONS
%  A(i,j) = effect at obs pt i due to a unit load at element j,
%  A = [nxn] arrays
     [a_uz,a_sxz,a_syz] = easy_anti_coeff(G,xm,ym,xm,ym,a);
% Set self-influence displacement coefficients (on main diagonal
% of a_uz) to -0.5
     a_uz(find(eye(length(a_uz)))) = -0.5;
%  SOLVE [a_syz][d] = [bc] FOR DISPLACEMENT DISCONTINUITIES [d],
%  [nxn]*[nx1]=[nx1]
     d = a_syz\bc;
%  FIND TRACTIONS AND DISPLACEMENTS AT THE BOUNDARY ELEMENTS
     syz = a_syz * d + PYZ;   % Note that PYZ is added back in;
     u_neg = a_uz * d;   % Displacement on (-) side of elements;
```

```
     u_pos = u_neg + d;  % Displacement on (+) side of elements;
%   PLOT SLIP FIGURE
     figure(1);     plot(xm,d,'r',xm,u_pos,'g',xm,u_neg,'b');
legend('Slip','u+','u-');
     xlabel('x');   ylabel('Slip or displacement');
     title('Slip and Displacements');

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    if isempty(X), UZ = [], SXZ = [], return, end
%  COMPUTE DISPLACEMENTS AND STRESSES AT SPECIFIED POINTS IN
%  BODY (CAPS)
%  Calculate influence coefficient arrays for gridpoint-element
%  pairs, A = [pxq,n]
%  Gridpoint arrays (X,Y) are sent to anti_coeff as column
%  vectors.  X(:) = [pxq,1] = [m,1]
     [p,q] = size(X);
     [A_UZ,A_SXZ,A_SYZ] = easy_anti_coeff(G,X(:),Y(:),xm,ym,a);
%  Calculate displacements and stress perturbation at gridpoints
     UZ  = A_UZ * d;                   % [pxq,n]*[n,1]=[pxq,1]
     SXZ = A_SXZ * d + PXZ;        % [pxq,n]*[n,1]=[pxq,1]
     SYZ = A_SYZ * d + PYZ;        % [pxq,n]*[n,1]=[pxq,1]
%  Reshape the arrays into the matrices the size of X (i.e.,
pxq)
     UZ = reshape(UZ,p,q);
     SXZ = reshape(SXZ,p,q);
     SYZ = reshape(SYZ,p,q);
%   PLOT FIGURES FOR OBSERVATION POINTS
     figure(2);     c1=contour (X,Y,UZ);     clabel(c1);
     xlabel('x');   ylabel('y');   axis('equal'); title ('UZ');
     figure(3);     c2=contour(X,Y,SXZ);     clabel(c2);
     xlabel('x');   ylabel('y');   axis('equal'); title('SXZ');
     figure(4);     c3=contour(X,Y,SYZ);     clabel(c3);
     xlabel('x');   ylabel('y');   axis('equal'); title('SYZ');
```