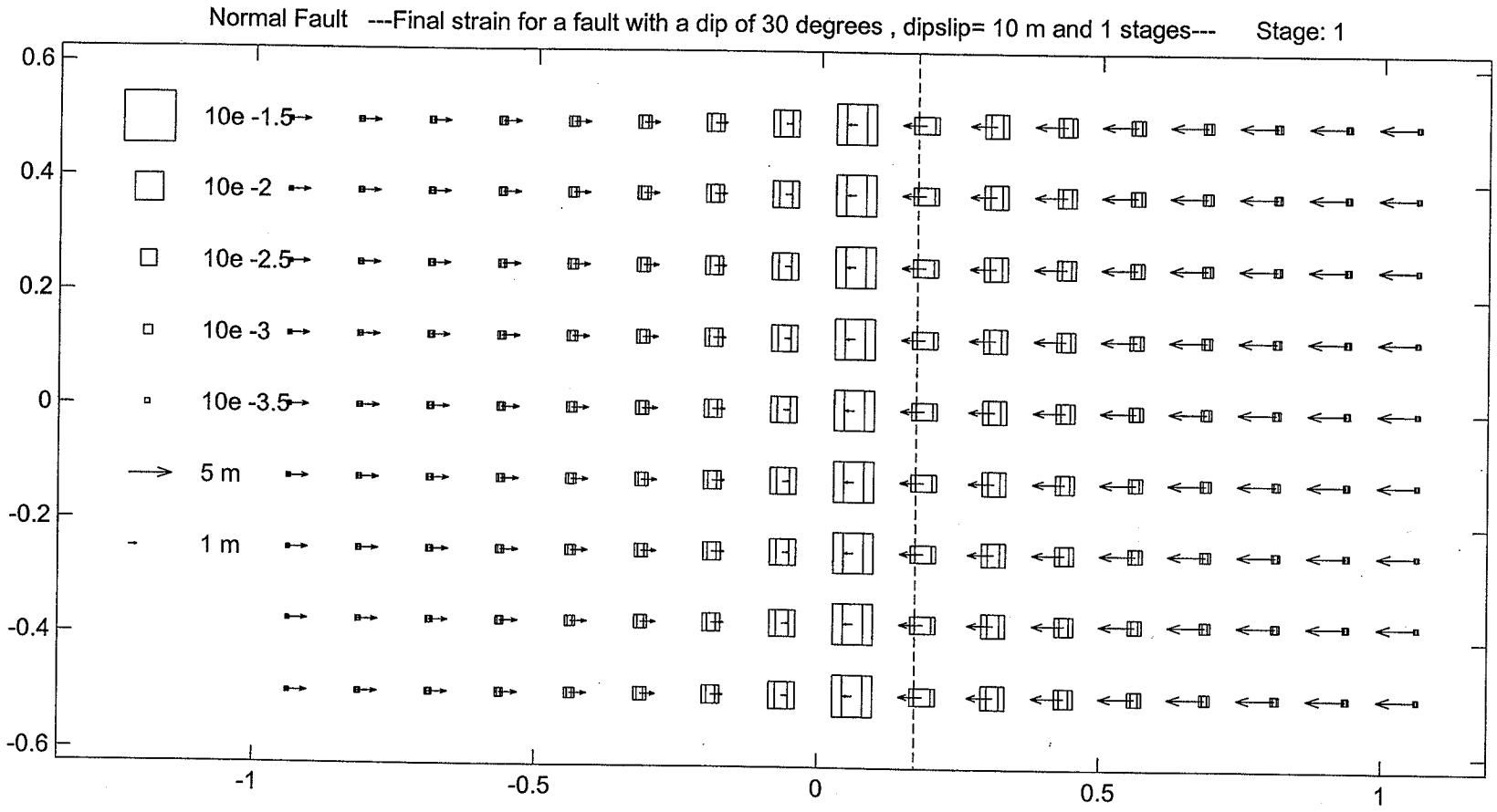
Normal Fault ---Final strain for a fault with a dip of 30 degrees , dipslip= 10 m and 1 stages--- Stage: 1
10e -1.5
10e -2
10e -2.5
10e -3
10e -3.5
5 m
1 m
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-1
-0.5
0
0.5
1

```
%This program is a main program which combines dipSlipfaulting and
%strikeSlipfaulting  for plotting strain figures on the ground surface
%1.It allows the fault gradually slips along a existing fault plane towards to the go↙
und surface.
%2.It plots the strain figures in red which have experienced plastic deformation.

%Author: Wen-Jeng Huang
%Date: Jan, 15, 2006
%The latest revison: Feburay, 13, 2006

global aa bb xl yb w
global Nhalf_width times Icrit
global small x3 theta
global factorI
% close all figures
close all
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Here are the parameters you can play with.
%%Specify the fault geometry%%%%%%%%%%%%%%%%%%%%%%%%
D = 0.1 ; % depth in km
W = 10 ;%width  in km
U1 = 0.01 ;%dip slip  in km
U2 =0.00; % strike slip in km
delta =  30; %fault angle in degree
%%%%%%%%%%%%
%% adjustable parameters
Nhalf_width = 0.025; %  a factor which can change the size of the strain figures
factorI = 15; % a factor which can change the size of the displacment arrows
WidthofPlot = 2;% a factor to change the plotting area
%%% permanent deformation
Icrit = 1e-5; % critical value of I2  for determining if the element will yield.
%%%%%fault slipping
times = 1; % the number of stages
ratio = 0.9; % the initial width / the actual width
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%The parameters above are the one you can play with
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

small = 0.000001; % a factor for calculating strains in kilometer which is 1 cm in th↙
is case
L=0;% L was the length of the fault but is redundant for this program
x3=0;%The program is only for calculating the points on the ground surface
if delta == 90 % avoid 90 degress
    delta = 89.9;
else
end
if D <= 0
    D=small;
end
```

```
48  theta  = delta / 180 * pi; %delta : fault angle (degree -> radian
49  W1 = D/ sin(theta); % the extended width betweeen fault tip and ground surface
50  %x3 = elevation; km
51  % incremental slip for stages
52  incrementU1 = U1/ times;
53  incrementU2 = U2/ times;
54
55  % If there is only one stage, we avoid a mistake which we might make
56  if times == 1
57      segmentWidth = 0;
58      ratio = 1;
59  else
60  segmentWidth = ( W - W * ratio ) / (times -1);
61  end
62
63  % to make sure that x3 is not zero because we want to get strains in 3D.
64  if x3 == 0
65      x3 = small;
66  else
67  end
68
69  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
70
71  %%some parameters for the map of strain figures at a certain scale
72  aa  = WidthofPlot; % km the length of the map in y axis
73  bb = 2 * aa; % km the length of the map in x axis
74  w = aa/8 ; % the interval of adjacent strain figures
75  xl = -(aa/2) ; % y coordinate starting from the bottom
76  shift = 1/2* w; % to avoid to plot the strain figure along the fault trace
77  %projectedX =  round(D * (1 / tan(theta)) * 10)/10 ;
78  projectedX = 0;
79  yb = -(bb/2) +  projectedX + shift; % x coordinate starting from the left
80
81          aaa = round(aa / w + 1);
82          bbbI = round(bb / w + 1);
83          dataStorageI = zeros(aaa* bbbI, 6);
84          dataStorageIbTotal = zeros(aaa * bbbI * 6, 6);% for the total strain
85          dataStorageIbElastic = zeros(aaa * bbbI * 6, 6);% for the elastic strain
86          dataStorageIbPerm = zeros(aaa * bbbI * 6, 6);% for the Permenant strain
87          permanRecord = zeros(aaa * bbbI , 1);% for tracking permanent deformation
88          I2Elastic = zeros(aaa * bbbI , 1);% for tracking I2
89
90  % calculate strains by using a differential method
91  for n = 1 : times % A1
92   figure(n) %fugere(1) is here because of function "posiDispTotalDispI" use function "↙
    quiver" to plot displacement
93  %To let more fault area to slip
94        width = ratio * W + segmentWidth * ( n -1) ;
95        Depth = D * ( (W1 + W- width ) / W1 );
```

```
 96
 97    [NmatrixX,NmatrixY]=matrixXandY;
 98  % This function is to create two matrixs of x and y positions.
 99
100
101  for  ii =  1 : aaa   %A2
102
103  y = xl  + (ii-1) * w; %xl: y in the cartesion coordinate system start from the bottom
      (negitive side)
104
105     for jj =  1 :bbbI   %A3
106
107      x = yb + (jj-1 ) * w ; % yb: x in the cartesian coordinater syetem start from th
    e left(negative side) and avoid x =0;
108
109   if n == 1 % A4
110
111         for counterI = 1 : 6 %A5
112  dataStorage((ii-1) * bbbI * 6 + (jj-1 ) * 6 + counterI, 1) =  x;% X coordinate in km
113  dataStorage((ii-1) * bbbI * 6 + (jj-1 ) * 6 + counterI, 2) =  y;% Y coordinate in km
114  dataStorage((ii-1) * bbbI * 6 + (jj-1 ) * 6 + counterI, 3) = -x3;% Z coordinate in km
115  end %A5
116  else %A4
117  end  %A4
118
119    % check which type of deformation
120    if permanRecord((ii-1) * bbbI + jj) == 0 %A6
121
122       [dataStorageIbTotal, dataStorageIbElastic,dataStorageIbPerm, permanRecord,I2Ela
    stic] = dispforStrainEI(L, Depth, width, incrementU1,incrementU2,n,y,x,ii,jj,bbbI,dat
    aStorageIbTotal, dataStorageIbElastic,dataStorageIbPerm, permanRecord,I2Elastic);
123    elseif permanRecord((ii-1) * bbbI + jj) == 1 %A6
124       [dataStorageIbTotal, dataStorageIbElastic,dataStorageIbPerm, permanRecord,I2Ela
    stic] = dispforStrainPI(L, Depth, width, incrementU1,incrementU2,n,y,x,ii,jj,bbbI,dat
    aStorageIbTotal, dataStorageIbElastic,dataStorageIbPerm, permanRecord,I2Elastic);
125    else %A6
126       [dataStorageIbTotal, dataStorageIbElastic,dataStorageIbPerm, permanRecord,I2Ela
    stic] = dispforStrainPermEI(L, Depth, width, incrementU1,incrementU2,n,y,x,ii,jj,bbbI
    ,dataStorageIbTotal, dataStorageIbElastic,dataStorageIbPerm, permanRecord,I2Elastic);
127
128    end  %A6
129
130  end % A3
131  end % A2
132
133  %% Plotting figures
134
135   [dataStorageI] = posiDispTotalDispISeries(L,  Depth, width, incrementU1,incrementU2,
     n, dataStorageI, permanRecord);
```

```
136  % This function is to derive the displacements in meter and plot the total displaceme↙
     nts
137  % the coordinate and the displacement are in cartesion coordinate system
138
139
140  xr = xl + aa; % xl:  y coordinate in the left edge; xr: in the right
141  yt = yb + bb + 1/2* w; % yb:  x coodinate in the bottom edge; yt: in the top
142  %%figure(1)
143  %%Total displacment and strain
144  % This function is to plot the projected line of the fault tip in the ground surface
145  faultTip(2*bb,D,W,ratio,segmentWidth,W1,n);
146
147
148  [NmatrixE1,NmatrixE2, Nmatrixtheta, Nscale,LegendScale] =strainScale(dataStorageIbTot↙
     al );
149  %The difference between Nscale and legendScale could be 1 or 0.5 depending
150  %on the maximum strain.
151
152  Nflag =2;
153  strainsquareNewpermTotal(NmatrixX,NmatrixY,NmatrixE1,NmatrixE2,Nmatrixtheta, Nscale,N↙
     flag,permanRecord);
154  %% Legend
155  [legendx,legendy,strainX,strainY,textX,textY] =strainlegend(LegendScale,3*shift);
156  Nflag = 1;
157  strainsquareLegend(legendx,legendy,strainX,strainY,Nscale,LegendScale,Nflag,textX,tex↙
     tY);
158  Ud = U1*1000;
159  Us = U2*1000;
160  if Ud~= 0 & Us~= 0
161    title(['Oblique Fault   ','---Final strain for a fault with a dip of ',int2str(delt↙
     a),' degrees ',',dipslip= ',num2str(abs(Ud)),' m',' ,strikeslip= ',num2str(abs(Us)),' ↙
      m',' and ',int2str(times),' stages---','      Stage: ',int2str(n)]);
162
163  elseif Ud == 0 & Us > 0
164
165       title(['Left-lateral Strike Fault   ','---Final strain for a fault with a dip o↙
     f ',int2str(delta),' degrees ',' ,strikeslip= ',num2str(abs(Us)),' m',' and ',int2str↙
     (times),' stages--- ','      Stage: ',int2str(n)]);
166    elseif Ud == 0 & Us < 0
167       title(['Right-lateral Strike Fault   ','---Final strain for a fault with a dip ↙
     of ',int2str(delta),' degrees ',' ,strikeslip= ',num2str(abs(Us)),' m',' and ',int2st↙
     r(times),' stages---','      Stage: ',int2str(n)]);
168    elseif Ud < 0
169       title(['Reverse Fault   ','---Final strain for a fault with a dip of ',int2str(↙
     delta),' degrees ',', dipslip= ',num2str(abs(Ud)),' m',' and ',int2str(times),' stage↙
     s---','      Stage: ',int2str(n)]);
170    else
171           title(['Normal Fault   ','---Final strain for a fault with a dip of ',int2↙
     str(delta),' degrees ',', dipslip= ',num2str(abs(Ud)),' m',' and ',int2str(times),' s↙
```

```
    tages---','      Stage: ',int2str(n)]);
172         end
173   axis equal;
174
175   axis( [ (yb -(3.25)* w) yt+0.5*w xl-w xr+w]);
176
177   end % A1
178
179
```