The Abdus Salam
International Centre
for Theoretical Physics

United Nations
Educational, Scientific and
Cultural Organization

IAEA
International Atomic Energy Agency

2464-31

Earthquake Tectonics and Hazards on the Continents

17 - 28 June 2013

Practicals/exercises on geodetic methods and examples

T. J. Wright
Univ of Leeds, UK

```
function [ux,uy,uz,err] = dc3d3(alpha,x,y,dip,al1,al2,aw1,aw2,disl1,disl2)
%% function [ux,uy,uz,err] = dc3d3(alpha,x,y,dip,al1,al2,aw1,aw2,disl1,disl2)
%%
%% Direct Translation of Pete Clarke's dc3d3.c into octave code
%% dc3d3.c is itself a translation of Okada's dc3d.f subroutine into C code
%% ...which has been pruned to compute ux,uy,uz for z=0 only, no tensile
%% components, depth=0
%%
%% tjw 11-sept-02

%% Constants
     F0 = 0;
     F1 = 1;
     F2 = 2;
     PI2 = 2*pi;
     EPS = 1.0e-6;

     [woo,nx]=size(x);

%% Other vector gubbins

     disl1_3=[disl1; disl1; disl1];
     disl2_3=[disl2; disl2; disl2];

     u = zeros(3,nx);
     dub = zeros(3,nx);



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                               %
%%dccon0 "subroutine"
% Calculates medium and fault dip constants
     c0_alp3 = (F1-alpha)/alpha;

     pl8 = PI2/360;
     c0_sd = sin(dip*pl8);
     c0_cd = cos(dip*pl8);
     if (abs(c0_cd) < EPS)
       c0_cd = F0;
       if (c0_sd>F0)  c0_sd = F1; end
       if (c0_sd<F0)  c0_sd = -F1; end
     end
     c0_cdcd = c0_cd*c0_cd;
     c0_sdcd = c0_sd*c0_cd;
%                               %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
     p=y*c0_cd;
     q=y*c0_sd;

     jxi = (((x-al1).*(x-al2))<=F0);
     jet = (((p-aw1).*(p-aw2))<=F0);

     for k = 1:2
       if (k==1) et = p-aw1; else et = p-aw2; end
       for j = 1:2
         if (j==1) xi = x-al1; else xi = x-al2; end


%

%       disp([x(1) y(1) p(1) q(1) jxi(1) jet(1)])

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                               %
%%dccon2 "subroutine"
% calculates station geometry constants for finite source

       etq_max=max([abs(et); abs(q)]);
       dc_max = max([abs(xi); etq_max]);

%       dc_max = max(abs(xi),max(abs(et),abs(q)));
%       if ((abs(xi/dc_max) < EPS) || (abs(xi) < EPS)) xi = F0; end
%       if ((abs(et/dc_max) < EPS) || (abs(et) < EPS)) et = F0; end
%       if ((abs(q/dc_max) < EPS) || (abs(q) < EPS)) q = F0; end


       xi=xi-(abs(xi./dc_max) < EPS).*xi;
       xi=xi-(abs(xi) < EPS).*xi;
       et=et-(abs(et./dc_max) < EPS).*et;
       et=et-(abs(et) < EPS).*et;
       q=q-(abs(q./dc_max) < EPS).*q;
       q=q-(abs(q) < EPS).*q;


       dc_xi = xi; dc_et = et; dc_q = q;


 %      disp([dc_max(1) xi(1) et(1) q(1)])


       c2_r = sqrt(dc_xi.^2 + dc_et.^2 + dc_q.^2);

       if (sum((c2_r) == F0) > 0)
           disp(['singularity error']);
           ux=0;
           uy=0;
           uz=0;
```

```
            err=1;
            return;
        end

        c2_y = dc_et*c0_cd + dc_q*c0_sd;
        c2_d = dc_et*c0_sd - dc_q*c0_cd;

%        if (dc_q == F0) c2_tt = F0; else c2_tt = atan(dc_xi.*dc_et./(dc_q.*c2_r)); end

        c2_tt = atan(dc_xi.*dc_et./(dc_q.*c2_r));
        c2_tt = c2_tt-c2_tt.*(dc_q == F0);

%        if ((dc_xi < F0) && (dc_q == F0) && (dc_et == F0)) c2_x11 = F0;
%        else
%          rxi = c2_r + dc_xi;
%          c2_x11 = F1/(c2_r*rxi);
%        end

        c2_x11 = (ones(1,nx)*F1)./(c2_r.*(c2_r+dc_xi));
        c2_x11 = c2_x11-((dc_xi < F0).*(dc_q == 0).*(dc_et == 0)).*c2_x11;

%        if ((dc_et < F0) && (dc_q == F0) && (dc_xi ==F0))
%          if ((c2_r-dc_et) < 1e-14) disp(['dccon2 a']);disp([c2_3-dc_et, c2_r, dc_et, dc_q, dc_xi]); end
%          c2_ale = -log(c2_r-dc_et);
%          c2_y11 = F0 ;
%        else
%         ret = c2_r + dc_et;
%          if (ret < 1e-14) disp(['dccon2 b']); disp([ret,c2_r,dc_et,dc_q,dc_xi]); end
%          c2_ale = log(ret);
%          c2_y11 = F1/(c2_r*ret);
%        end


        c2_ale_msk1 = ((dc_et < F0) .* (dc_q == F0) .* (dc_xi ==F0));
        c2_ale_msk2 = 1 - c2_ale_msk1;

        ret1 = c2_r-dc_et;
        ret2 = c2_r+dc_et;

        c2_ale = (c2_ale_msk1.*-log(ret1)) + (c2_ale_msk2.*log(ret2));
        c2_y11 = (ones(size(ret2))*F1)./(c2_r.*ret2);

%        disp([c2_r(1) c2_y(1) c2_d(1) c2_tt(1) c2_x11(1) c2_y11(1) c2_ale(1)])

                                    %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%          if ( ( (q==F0) && (((jxi==1) && (et==F0)) || ((jet==1)&&(xi==F0))) ) || (c2_r==F0) )
%            ux = 0; uy = 0; uz = 0;
%           err = 2; % singular problems: return error code
%           return;
%          end

          if ( sum( (q == F0) .* ( ((jxi == 1).*(et == F0)) + ((jet == 1).*(xi == F0)) ) + (c2_r == F0) ) > 0)
            ux = 0;
            uy = 0;
            uz = 0;
            err = 2; % singular problems: return error code
            return;
          end


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                   %
%% ub "subroutine"
%  part B of displacement and strain at depth due to buried fauls in semi-infinite medium

        rd = c2_r + c2_d;

        if ( sum(rd < 1e-14) > 0)
          disp(['ub']);disp([rd ,c2_r, c2_d,xi,et,q]);end


        if (c0_cd ~= F0)

%          if (xi==F0) ai4 = F0;
%          else
%            xx=sqrt(xi.^2+q.^2); % xx replaces x in original subroutine
%            ai4 = F1/c0_cdcd * (xi/rd*c0_sdcd + F2*atan((et.*(xx+q*c0_cd) + xx*(c2_r+xx)*c0_sd) / (xi*(c2_r+xx)*c0_cd)) );
%          end
%          ai3 = (c2_y*c0_cd/rd - c2_ale + c0_sd*log(rd)) / c0_cdcd;

         xx=sqrt(xi.^2+q.^2);

         ai4 = (xi ~= 0).* (  (F1/c0_cdcd) * (xi./rd*c0_sdcd + F2*atan((et.*(xx+q*c0_cd) +...
              xx.*(c2_r+xx)*c0_sd) ./ (xi.*(c2_r+xx)*c0_cd)) )  );
         ai3 = (c2_y*c0_cd./rd - c2_ale + c0_sd*log(rd)) / c0_cdcd;

        else

          rd2 = rd.*rd;
          ai3 = (et./rd + c2_y.*q./rd2 - c2_ale) / F2; %%% HERE
          ai4 = (xi.*c2_y./rd2)/F2;

        end

        ai1 = -xi./rd*c0_cd - ai4*c0_sd;
        ai2 = log(rd) + ai3*c0_sd;
```

```
       qx = q.*c2_x11;
       qy = q.*c2_y11;

%       disp([ai1(1) ai2(1) ai3(1) ai4(1)])


%strike-slip contribution
       if (disl1 ~=0)
         du2(1,:) = - xi.*qy  - c2_tt  - c0_alp3*ai1*c0_sd;
         du2(2,:) = - q./c2_r          + c0_alp3*c2_y./rd*c0_sd;
         du2(3,:) =   q.*qy            - c0_alp3*ai2*c0_sd;

%         size(disl1)
%         size(du2)

         dub = (disl1_3.*du2)/PI2;
       else
         dub = zeros(3,nx);
       end

%       disp([du2(1,1) du2(2,1) du2(3,1)])


%dip-slip contribution
       if (disl2 ~=F0)
         du2(1,:) = - q./c2_r          + c0_alp3*ai3*c0_sdcd;
         du2(2,:) = - et.*qx - c2_tt   - c0_alp3*xi./rd*c0_sdcd;
         du2(3,:) =   q.*qx            + c0_alp3*ai4*c0_sdcd;
         dub = dub + (disl2_3/PI2).*du2;
       end

%       disp([du2(1,1) du2(2,1) du2(3,1)])


%                                 %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
         du(1,:) = dub(1,:);
         du(2,:) = dub(2,:)*c0_cd - dub(3,:)*c0_sd;
         du(3,:) = dub(2,:)*c0_sd + dub(3,:)*c0_cd;
         if((j+k)~=3) u = u+du; else u = u-du;end
       end
     end

%       disp([dub(1,1) dub(2,1) dub(3,1)])



     ux = u(1,:);
     uy = u(2,:);
     uz = u(3,:);
     err = 0;

%       disp([ux(1) uy(1) uz(1)])
```

```
function [U,flag]=disloc3d3(m,x,lambda,mu)
% DISLOC3D    [U,flag]=disloc3d(m,x,lambda,mu)
%
% Returns the deformation at point 'x', given dislocation
% model 'm'.  lambda and mu are the Lame parameters
%
% Both 'm' and 'x' can be matrices, holding different models
% and observation coordinates in the columns.  In this case,
% the function returns the deformation at the points specified
% in the columns of 'x' from the sum of all the models in the
% columns of 'm'.  'x' must be 2xi (i = number of observation
% coordinates) and 'm' must be 9xj (j = number of models).
%
% The coordinate system is as follows: east = positive X,
% north = positive Y. Calculations assume Z=0.
%
% The main output is 'U', the three displacement components:
% east, north, and up (on the rows)).  This output has
% the same number of columns as 'x'.
%
% Output 'flag' is set for a singularity.
%
% The dislocation model is specified as: east,north,strike,dip,
% rake,slip,length,hmin,hmax. The coordinates (east,north)
% specify the midpoint of the surface projection of the fault
% plane.
%
% Modified to use cut down, octave version of dc3d.f: dc3d3.m.
% No Strains, No Tensile Components, Depth of observations = 0.
%
% Vectorised for optimal speed in MATLAB 17-may-2005

     DEG2RAD = (2*pi/360);
     nfaults = size(m,2);
     nx      = size(x,2);

%initiate matrix for output displacements, U
     U=zeros(3,nx);

% calculate alpha
     alpha = (lambda+mu)/(lambda+2*mu);

%loop over models
     for i=1:nfaults
%convert model into correct inputs for dc3d3.m
       flt_x = ones(1,nx)*m(1,i);
       flt_y = ones(1,nx)*m(2,i);
       strike = m(3,i);
       dip = m(4,i);
       rake = m(5,i);
       slip = m(6,i);
       length = m(7,i);
       hmin = m(8,i);
       hmax = m(9,i);

       rrake = (rake+90)*DEG2RAD;
       sindip = sin(dip*DEG2RAD);
       w = (hmax-hmin)/sindip;
       ud = ones(1,nx)*slip*cos(rrake);
       us = ones(1,nx)*-slip*sin(rrake);
       opening = ones(1,nx)*slip;
       halflen = length/2;
       al2 = ones(1,nx)*halflen;
       al1 = -al2;
       aw1 = ones(1,nx)*hmin/sindip;
       aw2 = ones(1,nx)*hmax/sindip;
%reject data which breaks the surface
       if (sum(hmin < 0)>0) disp(['ERROR: Fault top above ground surface']);end;
       hmin=hmin+(hmin == 0)*0.00001;

       sstrike = (strike+90)*DEG2RAD;
       ct = cos(sstrike);
       st = sin(sstrike);

%loop over points

         X=ct*(-flt_x+x(1,:))-st*(-flt_y+x(2,:));
         Y=ct*(-flt_y+x(2,:))+st*(-flt_x+x(1,:));
```

```
if (m(10,i) == 1)
    %REGULAR DOUBLE COUPLE FAULT
         [ux,uy,uz,err]=dc3d4(alpha,X,Y,-dip,al1,al2,aw1,aw2,us,ud);

elseif (m(10,i) == 2)
   % TENSILE ONLY COMPONENT SOLUTION
         [ux,uy,uz,err]=dc3d5(alpha,X,Y,-dip,al1,al2,aw1,aw2,0,0,opening);
else
    disp(['10th row in geometry file must be 1 (=double couple) or 2 (tensile)']);
end


         if (err~=0) disp(['error code in dc3d3... stopping!']);flag = err,return;else flag=0;end
         U(1,:) = U(1,:) + ct*ux + st*uy;
         U(2,:) = U(2,:) -st*ux + ct*uy;
         U(3,:) = U(3,:) + uz;

     end
```

```
%%%%% Script to produce simulated interferograms using okada formulations
% Tim J Wright, University of Leeds, 18 November 2008
% t.wright@see.leeds.ac.uk
%
% Note, the script produces wrapped and unwrapped simulations on a 50x50 km
% grid, with the fault located at the grid centre.
%
%
% Requires the following additional files:
%
% disloc3d3.m
% dc3d4.m
%
% change log:
%
% update graphics   22 November 2009   tjw
%
%%%%% Edit the following lines for each earthquake

Strike = 135;    %strike in degrees
Dip = 35;        %dip in degrees
Rake = -90;     %rake in degrees
Slip = 1;        %magnitude of slip vector in metres
Top_depth = 0.0 ;  %depth (measured vertically) to top of fault in kilometres
Bottom_depth = 7.5 ;  %depth (measured vertically) to bottom of fault in kilometres
Length = 13 ;  %fault length in kilometres
LOS_vector = [0.3523 -0.0768 0.9327];  %Unit vector in satellite line of sight

%%%%%%%%%%%%%%%%%%%%
% DO NOT EDIT ANY TEXT BELOW HERE
%%%%%%%%%%%%%%%%%%%%

%put model parameters into string.
model = [1;1;Strike;Dip;Rake;Slip;Length*1000;Top_depth*1000;Bottom_depth*1000;1];

%set up grid
x=[-25000:100:25000];
y=[-25000:100:25000];
[xx,yy]=meshgrid(x,y);
xx= reshape(xx,numel(xx),1);
yy= reshape(yy,numel(yy),1);
coords = [xx';yy'];

%define elastic parameters
lambda = 3.2e10;
mu     = 3.2e10;

%do elastic calculation
[U,flag]=disloc3d3(model,coords,lambda,mu);

%regrid and calculate LOS displacement
xgrid = reshape(U(1,:),numel(y),numel(x));
ygrid = reshape(U(2,:),numel(y),numel(x));
zgrid = reshape(U(3,:),numel(y),numel(x));

los_grid = xgrid*LOS_vector(1) + ygrid*LOS_vector(1) + zgrid*LOS_vector(3);
los_grid_wrap = mod(los_grid+10000,0.028333);

%calcualte fault parameters for plotting
 DEG2RAD = pi/180;
 end1x = model(1) + sin(model(3)*DEG2RAD).*model(7)/2;
 end2x = model(1) - sin(model(3)*DEG2RAD).*model(7)/2;
 end1y = model(2) + cos(model(3)*DEG2RAD).*model(7)/2;
 end2y = model(2) - cos(model(3)*DEG2RAD).*model(7)/2;
 c1x = end1x + sin((model(3)+90)*DEG2RAD).*cos(model(4)*DEG2RAD).*model(8);
 c2x = end1x + sin((model(3)+90)*DEG2RAD).*cos(model(4)*DEG2RAD).*model(9);
 c3x = end2x + sin((model(3)+90)*DEG2RAD).*cos(model(4)*DEG2RAD).*model(9);
 c4x = end2x + sin((model(3)+90)*DEG2RAD).*cos(model(4)*DEG2RAD).*model(8);
 c1y = end1y + cos((model(3)+90)*DEG2RAD).*cos(model(4)*DEG2RAD).*model(8);
 c2y = end1y + cos((model(3)+90)*DEG2RAD).*cos(model(4)*DEG2RAD).*model(9);
 c3y = end2y + cos((model(3)+90)*DEG2RAD).*cos(model(4)*DEG2RAD).*model(9);
 c4y = end2y + cos((model(3)+90)*DEG2RAD).*cos(model(4)*DEG2RAD).*model(8);

%% Plot interferograms
figure(1)
subplot(2,2,1)
imagesc(x/1000,y/1000,los_grid)
hold on
 plot([end1x;end2x]/1000,[end1y;end2y]/1000,'w')
 plot([end1x]/1000,[end1y]/1000,'wo')
 plot([c1x;c2x;c3x;c4x;c1x]/1000,[c1y;c2y;c3y;c4y;c1y]/1000,'w')
axis xy
axis image
title('Unwrapped simulation')
xlabel('easting (km)')
ylabel('northing (km)')
h=colorbar('vert');
ylabel(h,'metres')

subplot(2,2,2)
imagesc(x/1000,y/1000,los_grid_wrap/0.028333*2*pi-pi)
hold on
 plot([end1x;end2x]/1000,[end1y;end2y]/1000,'w')
 plot([end1x]/1000,[end1y]/1000,'wo')
 plot([c1x;c2x;c3x;c4x;c1x]/1000,[c1y;c2y;c3y;c4y;c1y]/1000,'w')
axis xy
axis image
title('Wrapped simulation')
xlabel('easting (km)')
ylabel('northing (km)')
h=colorbar('vert');
ylabel(h,'radians')


subplot(2,2,3)
```

```
imagesc(x/1000,y/1000,zgrid)
hold on
[xx,yy]=meshgrid(x,y);
step=25;
quiver(xx(1:step:end,1:step:end)/1000,yy(1:step:end,1:step:end)/1000,xgrid(1:step:end,1:step:end),ygrid(1:step:end,1:step:end),'w')
 plot([end1x;end2x]/1000,[end1y;end2y]/1000,'w')
 plot([end1x]/1000,[end1y]/1000,'wo')
 plot([c1x;c2x;c3x;c4x;c1x]/1000,[c1y;c2y;c3y;c4y;c1y]/1000,'w')
axis xy
axis image
title('3D deformation')
xlabel('easting (km)')
ylabel('northing (km)')
h=colorbar('vert');
ylabel(h,'Vertical Deformation (m)')

subplot(2,2,4)
hold off
plot(x/1000,zgrid(251,:),'b')
xlabel('easting (km)')
ylabel('displacement (m)')
hold on
plot(x/1000,xgrid(251,:),'r')
plot(x/1000,los_grid(251,:),'k')
legend1=legend ('vert','east', 'los');
title('Displacement profiles along y=0')
```

Part 2: Using InSAR and elastic dislocation modelling to estimate earthquake source parameters.

Tim Wright, COMET, University of Leeds; t.j.wright@leeds.ac.uk

1. Theory / Introduction

Since the launch of ERS-1 in 1991, the coseismic displacements of more than 80 earthquakes have been captured by InSAR. These range in size from Magnitude ~5 to Magnitude ~8, and have a variety of different mechanisms. Typically, coseismic interferograms are interpreted by comparing the interferograms with the predictions of elastic dislocation models, in which slip occurs on one or more dislocation (fault) in an elastic half space.

In this practical class, you will examine 3 coseismic interferograms from continental earthquakes in different tectonic settings. For each one, you will try to find a simple, single-fault model that provides a reasonable fit to the observed interferograms, using some simple elastic dislocation modelling code (*leeds_okada*), based on the analytical expressions of (Okada, 1985, Okada, 1992).

2. Aims of the lab

1. To improve understanding of how to interpret coseismic interferograms.
2. To understand the coseismic displacement fields produced by different types of earthquake.
3. To learn how to produce elastic dislocation models of earthquakes
4. To gain an appreciation of the uncertainties involved with determining earthquake mechanisms from coseismic InSAR data.

3. Data sets

For each earthquake, there is an A3 printed sheet. It contains the following four panels:

(top left) Wrapped interferogram. Each interference fringe is a phase change of 2π radians, which corresponds to motion of 28.3 mm in the satellite line of sight (a unit vector is given to define this line of sight; the interferogram is the dot product of the 3D displacement field with the line of sight vector). A change in colour from red through yellow to blue, indicates motion away from the satellite, which can be interpreted as subsidence **only if the displacements are vertical.**

(top right) Unwrapped interferogram. This is included to help interpret the interferograms. Blue colours indicate motion away from the satellite (range increases); red colours indicate motion towards the satellite (range decreases).

(bottom left) Zoom in on the wrapped interferogram, to help you count fringes.

(bottom right) Coloured and shaded topographic map for the area of the zoomed interferogram. The topography comes from the shuttle radar topography mission 3 arcsecond elevation model (Farr et al., 2007).

The interferograms are formed from pairs of images acquired by the ERS (Quakes 1, 2) or Envisat (Quake 3) satellites. They were processed using the JPL/Caltech ROI_PAC software (Rosen et al., 2004).

4. Getting started

Download the zip file: http://see.leeds.ac.uk/~eartjw/data/trieste/lab.zip and unzip.

Open matlab and move into the lab_files directory.

Check that the matlab script, *leeds_okada.m* works, by opening it in the editor and running it, or by typing *leeds_okada*.

Unlike previous labs, you are not here building up a matlab script. There is more thinking involved instead. For each quake you will be editing the fault parameters in *leeds_okada.m* but the first stage is to engage your grey matter to work out as much as you can about the earthquakes from looking and thinking.

5. Main Task: Determine the best-fitting elastic dislocation model for three example earthquakes

Specifically, you will be attempting to find the best fit rectangular dislocation with uniform slip. You will be doing this by a process of induction and iterative forward modelling (i.e. educated trial and error).

For each earthquake, the following procedure should enable you to reach the best fitting model as quickly as possible:

1. Draw on a vector corresponding to the approximate horizontal projection of the vector from the satellite to the ground. This is perpendicular to the azimuth (flight direction) of the satellite, and on the handout I give you the Line of Sight unit vector that points from the ground to the satellite.

2. Count the fringes. How much motion is there towards and away from the satellite? Is it symmetrical (i.e. is the motion away from the satellite equal to that towards the satellite, or different). Note that dip slip earthquakes have a very asymmetrical displacement pattern. The subsidence will be larger than the uplift for normal faulting earthquakes and vice-versa for thrust faulting earthquakes.

3. Determine approximately the type of earthquake. Draw on the surface rupture (if there is one). Estimate the strike and determine the direction of dip.

4. Try to produce an elastic dislocation model for the earthquake that matches the observed interferograms.

i) Copy leeds_okada.m to quake1[or 2,3]_okada.m and edit this file.

ii) Edit parameters in the top of the file. (The line of sight vector is printed on the handout for each earthquake and must be typed into the script).

iii) Run quakeN_okada.m. This calculates the elastic dislocation model and plots the results in 4 different ways (see titles of plots).

iv) Compare the figures produced with the observed interferogram.

v) Repeat steps ii to iv until you have a satisfactory model.

If you wish, make a table of best-fitting parameters you find for each earthquake as well as a figure showing the best fit simulated interferogram. You may wish to indicate a range of values that fit the data or an approximate error bar for each parameter.

In the table include an estimate for the seismic moment (M_0) and moment magnitude (M_w) derived from the model parameters, assuming a typical shear modulus of 3.2×10^{10} Pa.

In the discussion, discuss how well you think you can fit each interferogram with these simple dislocation models. If the fit is not good, why not? What could be done to improve it?

What further steps would you need to do to write a real paper on this work?

You may also wish to discuss how the deformation in these earthquakes relates (or not) to features in the landscape visible in the topography.

References

FARR, T. G., ROSEN, P. A., CARO, E., CRIPPEN, R., DUREN, R., HENSLEY, S., KOBRICK, M., PALLER, M., RODRIGUEZ, E., ROTH, L., SEAL, D., SHAFFER, S., SHIMADA, J., UMLAND, J., WERNER, M., OSKIN, M., BURBANK, D. & ALSDORF, D. 2007. The shuttle radar topography mission. *Reviews of Geophysics,* 45.

OKADA, Y. 1985. Surface Deformation Due to Shear and Tensile Faults in a Half-Space. *Bulletin of the Seismological Society of America,* 75**,** 1135-1154.

OKADA, Y. 1992. Internal Deformation Due to Shear and Tensile Faults in a Half-Space. *Bulletin of the Seismological Society of America,* 82**,** 1018-1040.

ROSEN, P. A., HENSLEY, S., PELTZER, G. & SIMONS, M. 2004. Updated repeat orbit interferometry package released. *Eos, Transactions American Geophysical Union,* 85**,** 47.

Week 5 Problem: Forecasting Earthquakes in Turkey

The aim of this problem set is to estimate the date of the next earthquake to hit the town of Izmit, on the North Anatolian Fault in Turkey. The last big earthquake occurred in 1999. You will examine satellite measurements of deformation during the Izmit earthquake and determine how much slip occurred on the fault. Using the average motion of Turkey between earthquake you will then estimate the interval between earthquakes on this fault, and make a prediction for the date of the next earthquake.

1. Starting at the zero displacement fringes, count the maximum number of fringes on each side of the fault. Use the zoom tool to help you where the fringes are very close together.

Number of fringes south of the fault =
Number of fringes north of the fault =

2. What is the total offset in line-of-sight displacement (range change) at the fault (r)? Note that one side has moved away and one side towards the satellite… the offset is the step in range change from one side to the other.

Line-of-sight offset, r = mm

[Remember: 1 fringe = 30 mm of motion in the satellite line of sight. Measurements are all relative to the "Zero displacement" fringe shown as a dashed white line in the figure.]

3. To convert this into slip on the fault plane we must take into account the look direction of the satellite. [The interferogram only measures the component of surface deformation in the satellite look direction].

We can assume in this case that the fault slip at the rupture is parallel to the rupture and horizontal. Let us say that the slip on the fault is s metres.

3.1 As a first step, give a simple formula for y, the component of horizontal slip (s) perpendicular to the satellite flight path, in terms of θ and s (Figure 1).

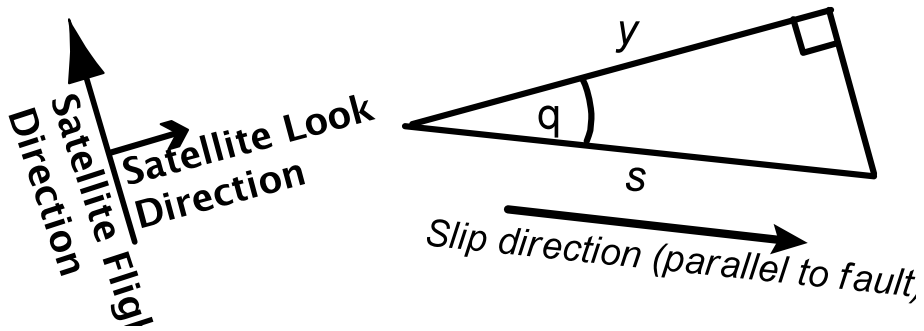


Figure 1. (Map View)

y =

3.2 Now, give a simple formula for *r*, the deformation in the satellite line of sight in terms of ϕ, the satellite incidence angle, and *y* (Figure 2).

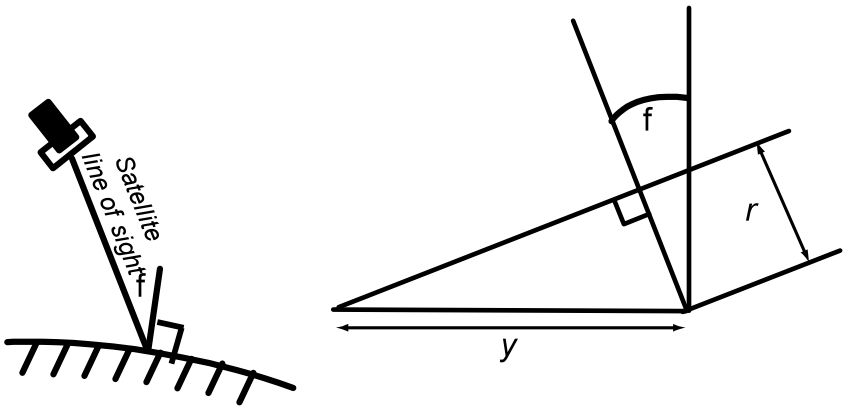


Figure 2.

r =

4.3 Finally, combine and rearrange your equations to give an expression for *s* in terms of *r,* θ and ϕ.

s =

Measure/estimate θ from the interferogram diagram using the central portion of the fault rupture to define the slip direction (as defined in figure 1). The satellite incidence angle at the centre of the interferogram, ϕ, is 23 degrees.

Therefore

Surface slip during the Izmit earthquake, s =

4. The long-term slip rate for the North Anatolian Fault is between 20 and 25 mm/yr. What is the repeat time for earthquakes at Izmit?

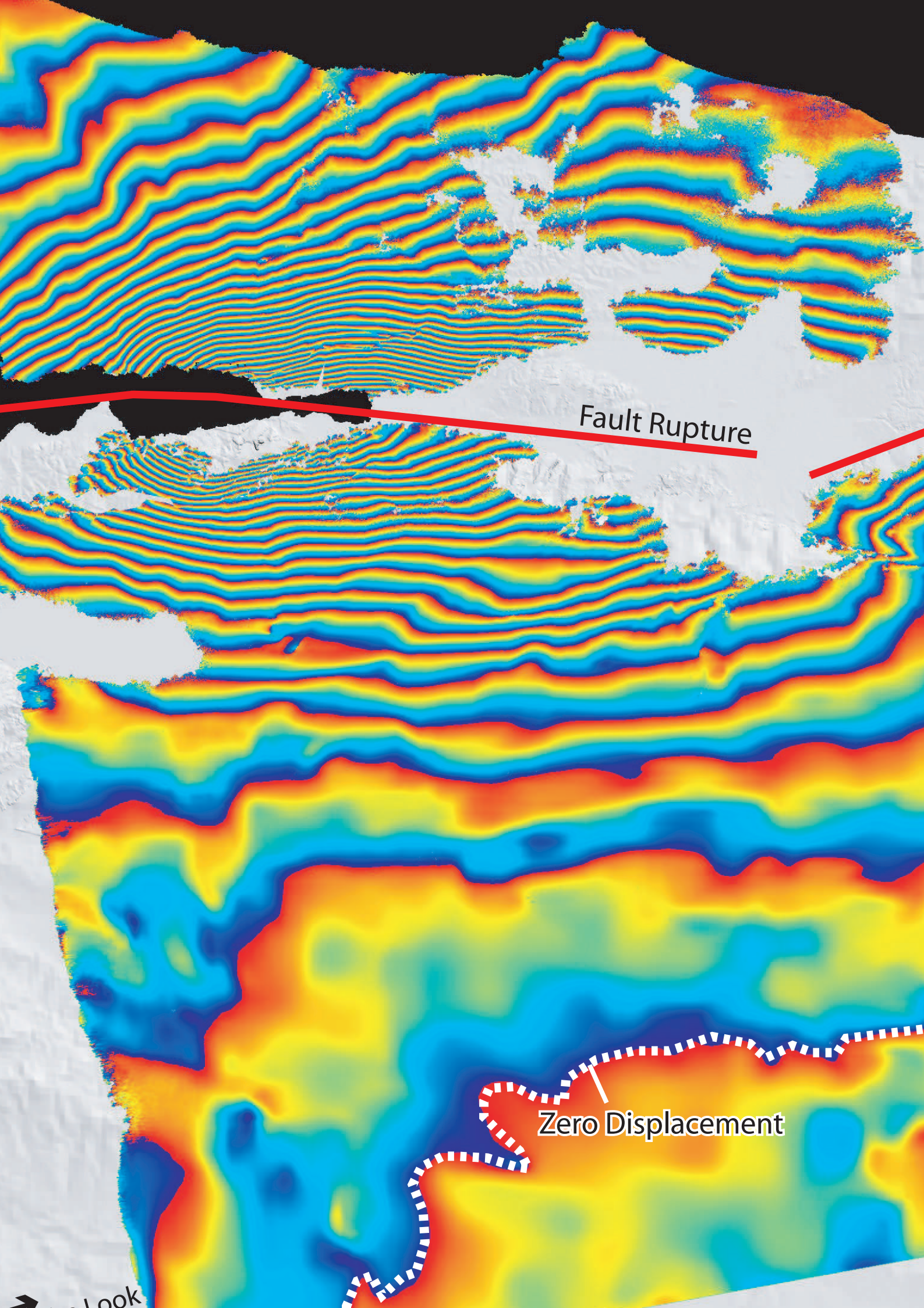
Repeat time **=** **years**

Therefore, when will be the next big earthquake in the Izmit area?

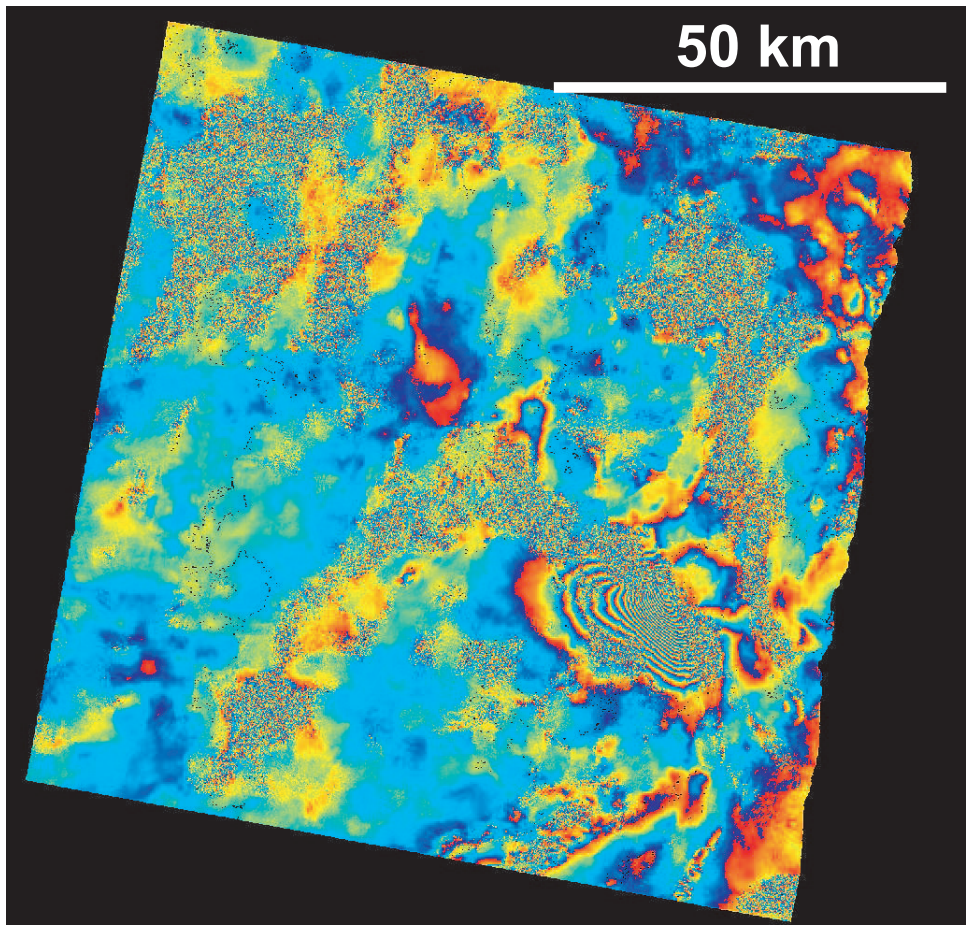
Date of next Izmit earthquake **=**

5. What assumptions have you made in coming to this conclusion?

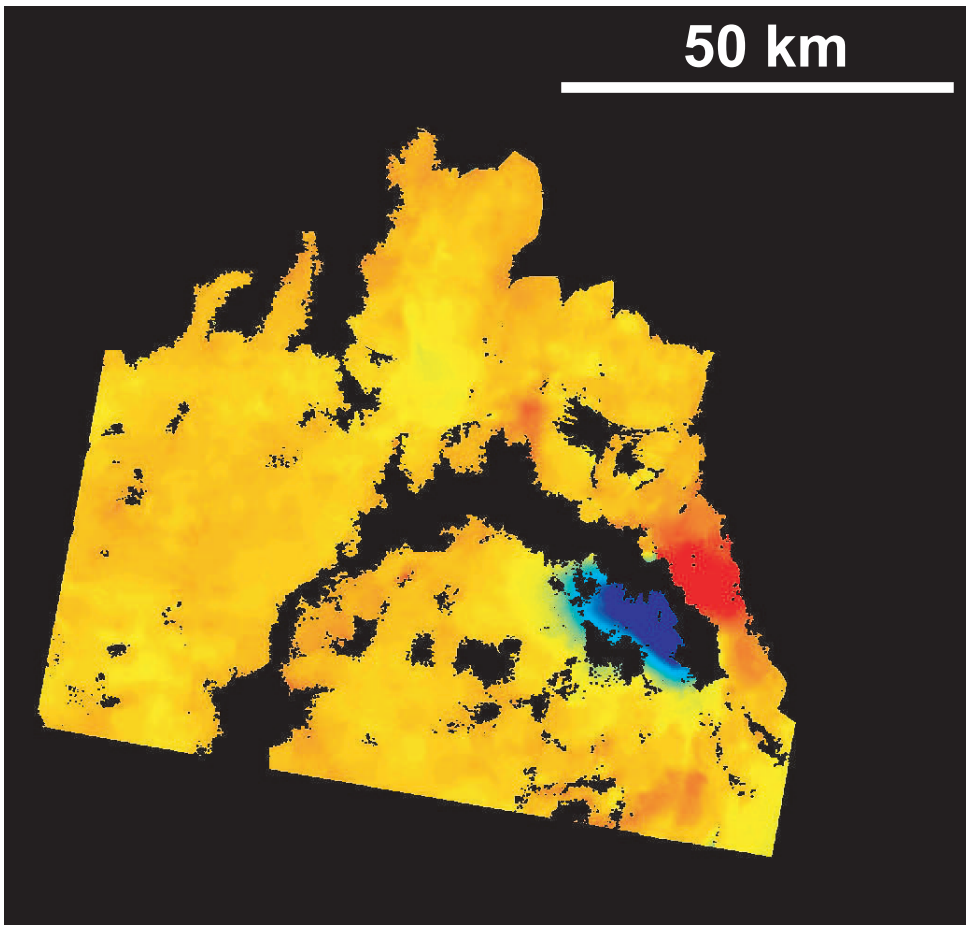
Assumptions:


Fault Rupture
Zero Displacement
Look

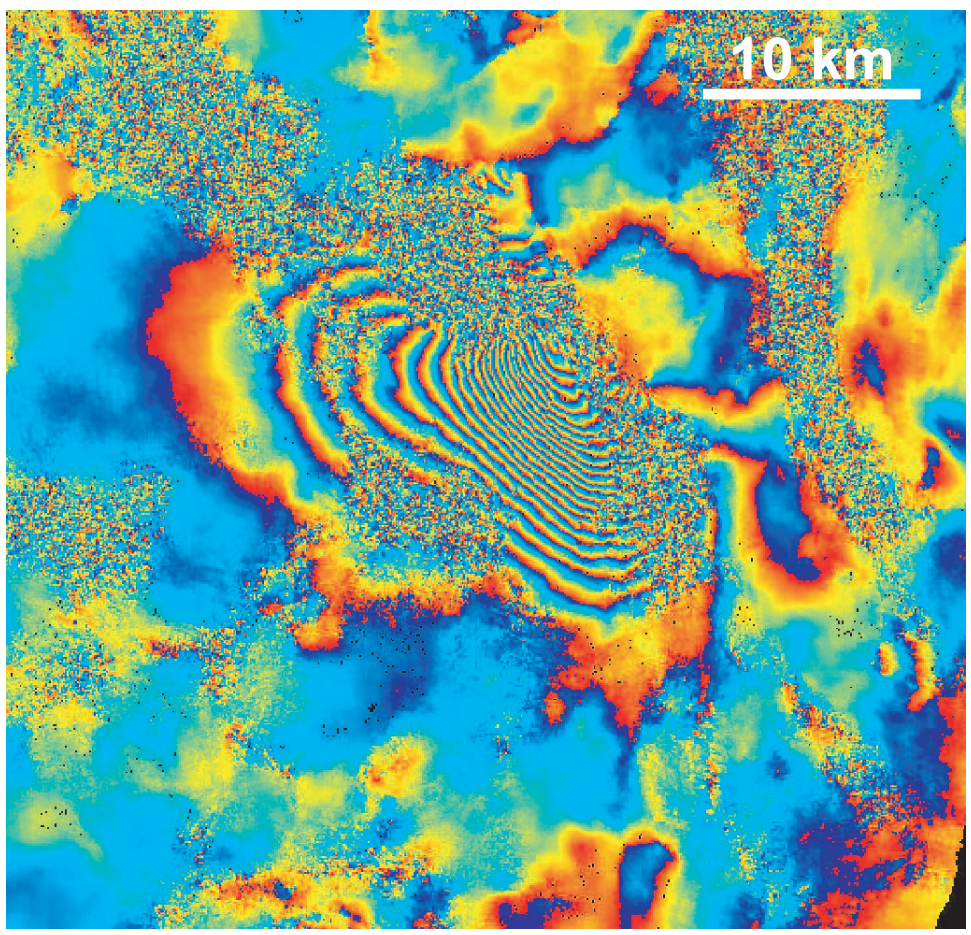
Earthquake 1



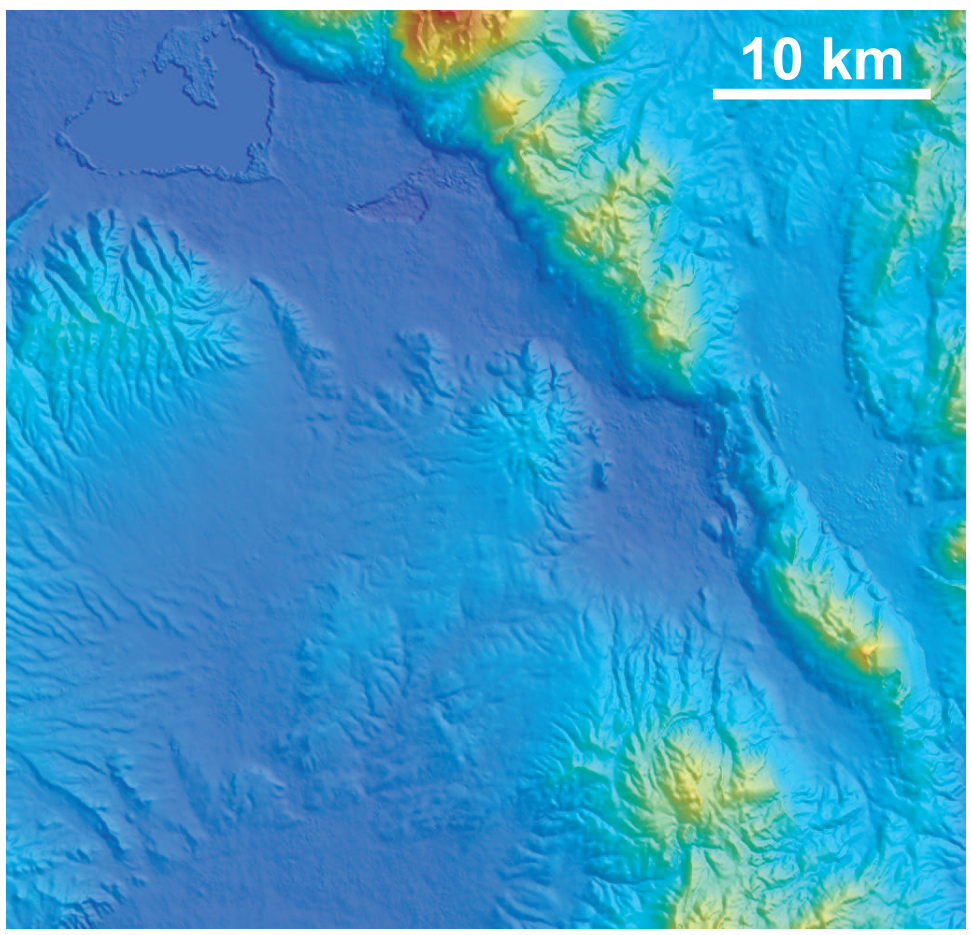
Whole image, wrapped



Whole image, unwrapped



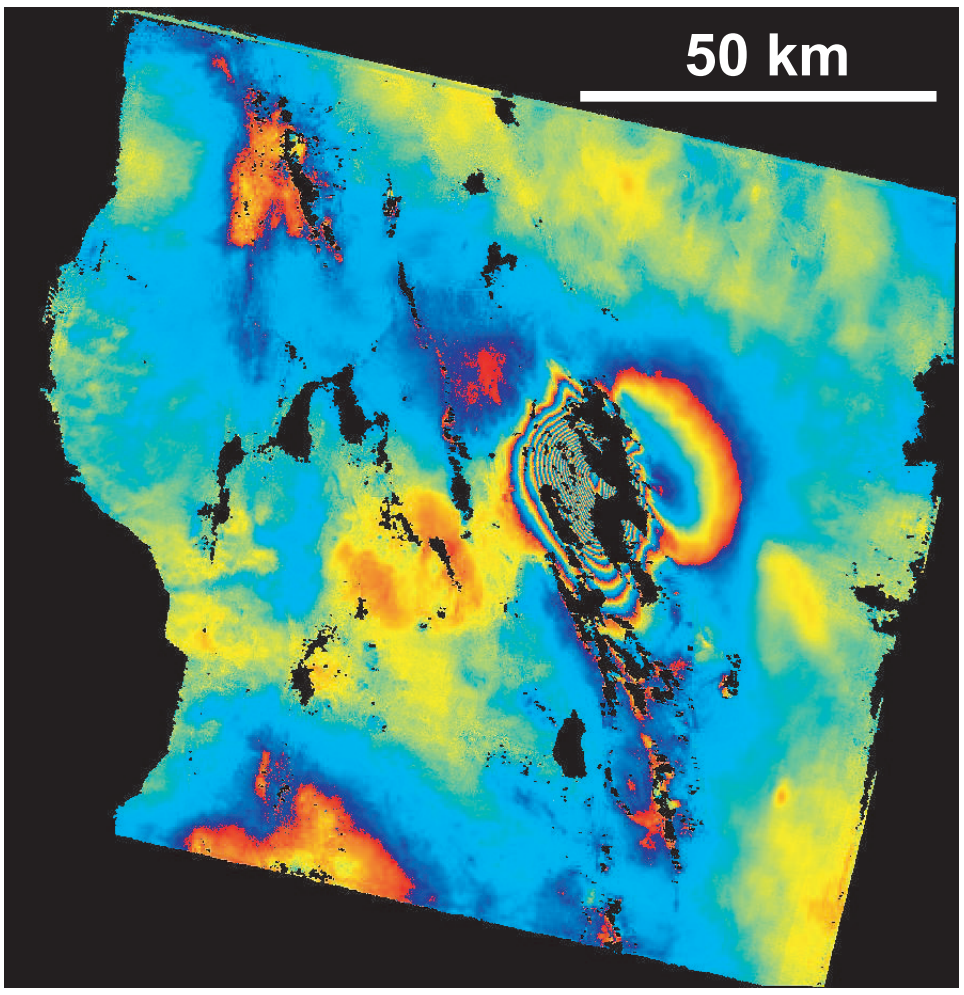
Zoom, wrapped



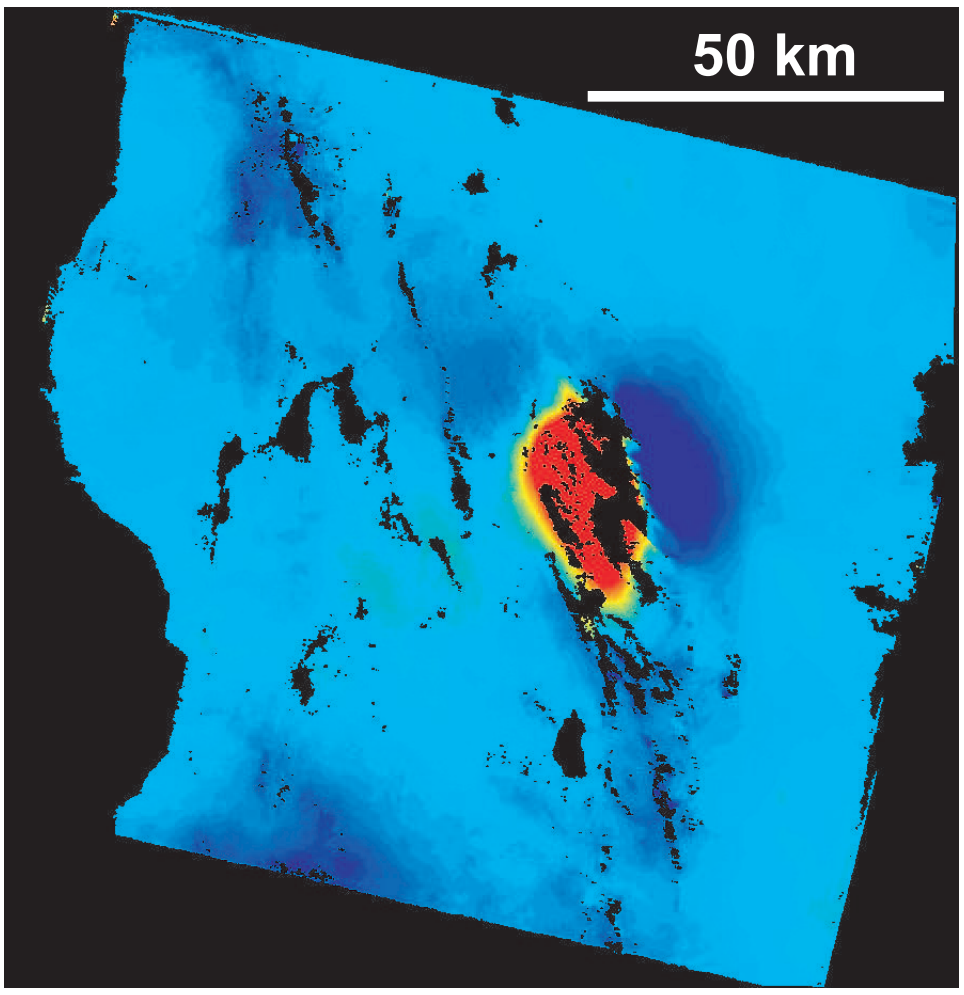
Zoom topography (blue to red ~ 1500m)

Descending interferogram; LOS Vector at earthquake location [East, north, up] ~ [0.3523, -0.0768, 0.9327]

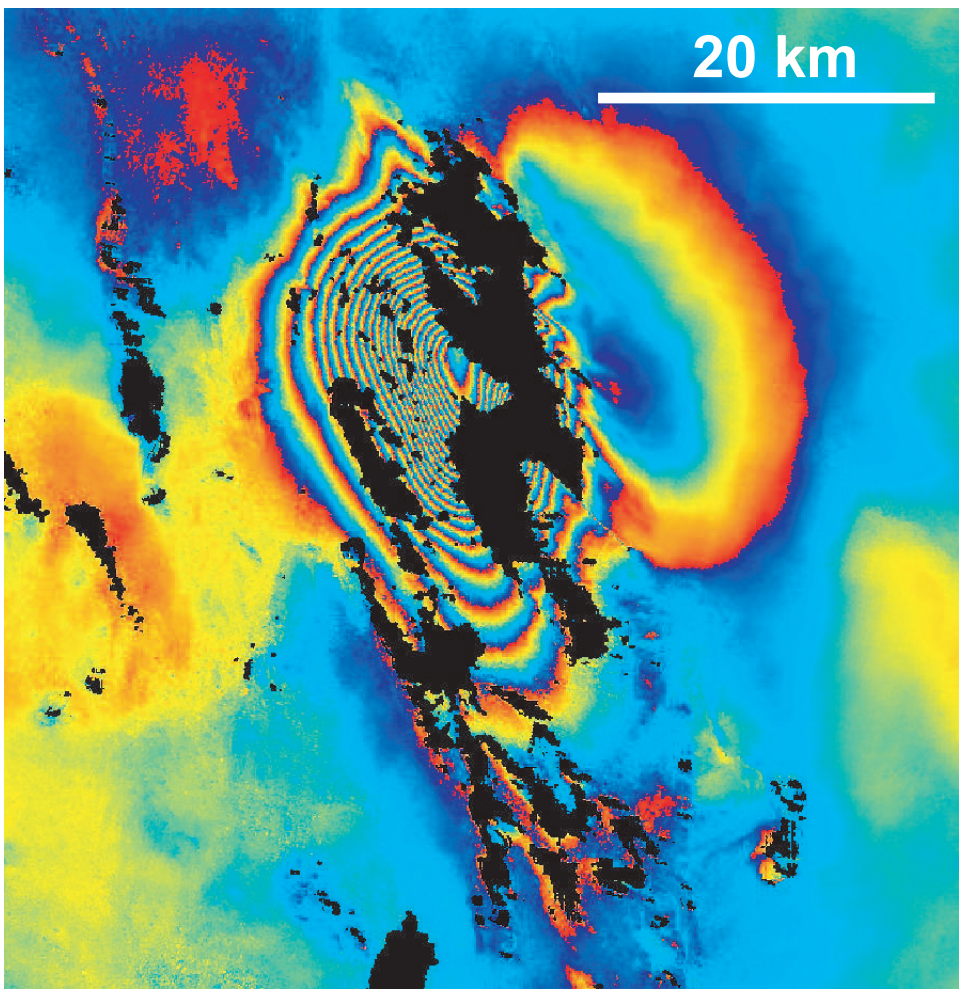
Earthquake 2



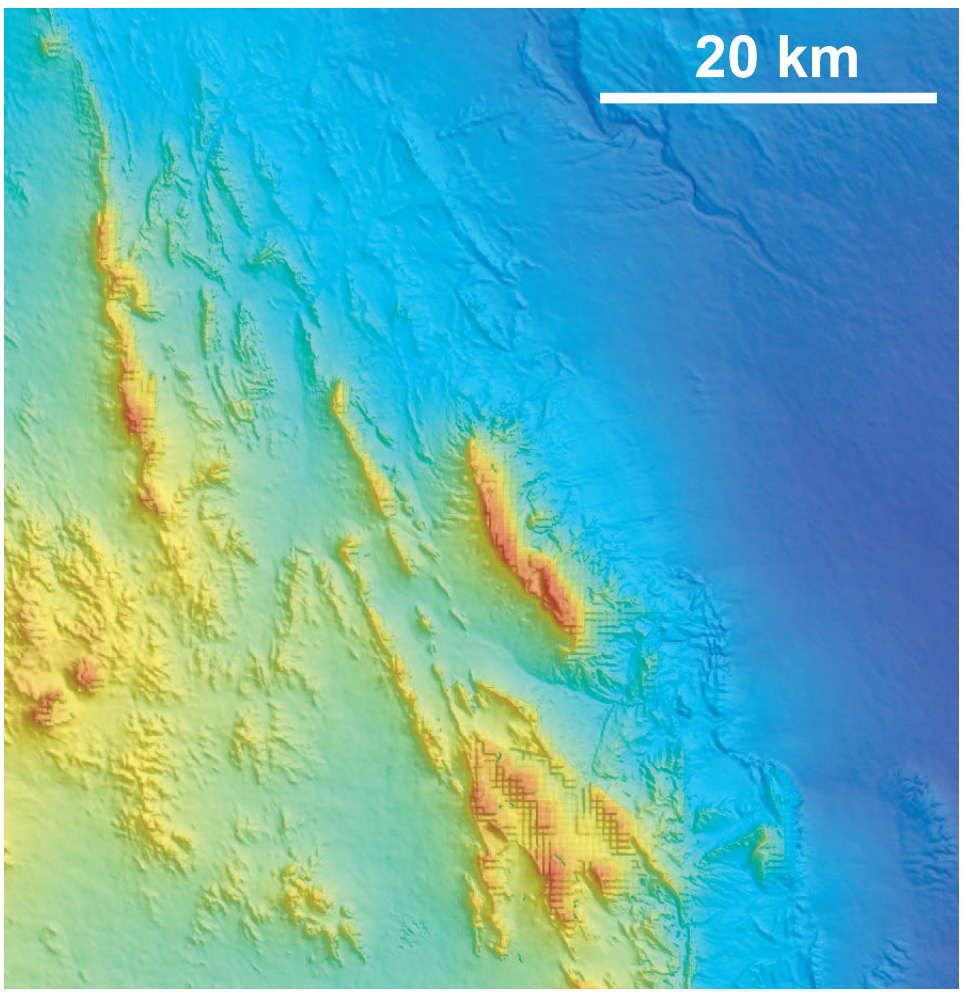
Whole image, wrapped



Whole image, unwrapped



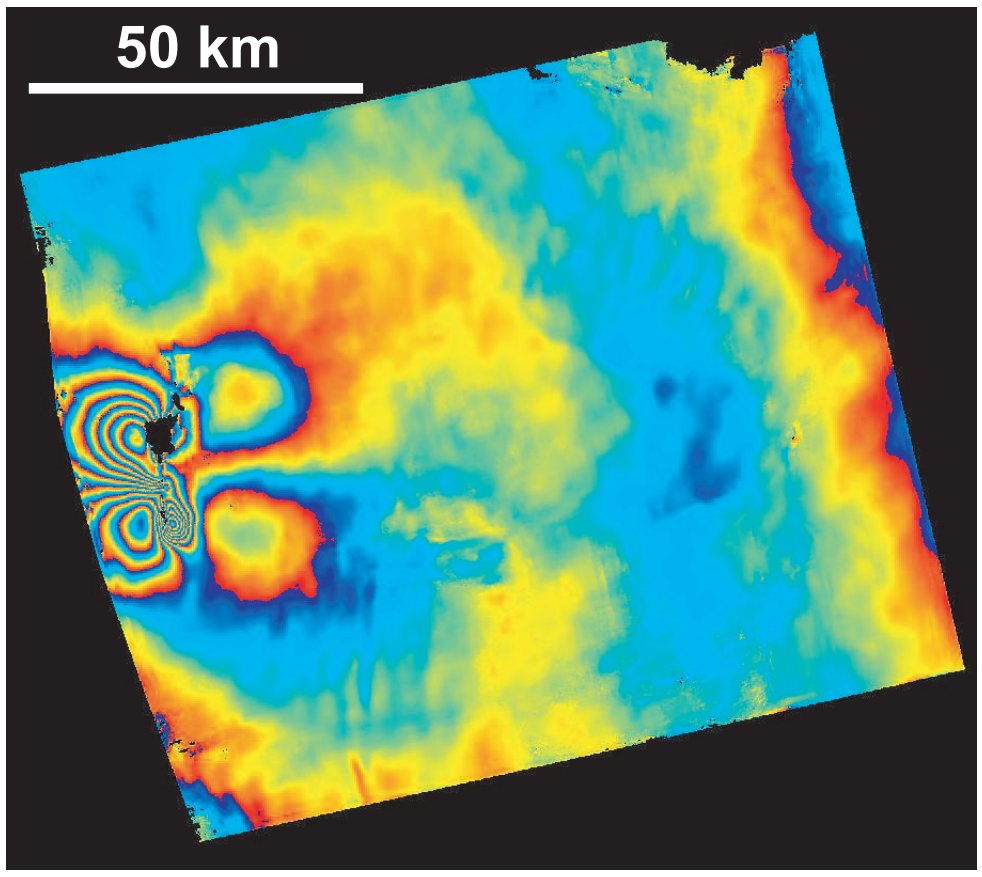
Zoom, wrapped



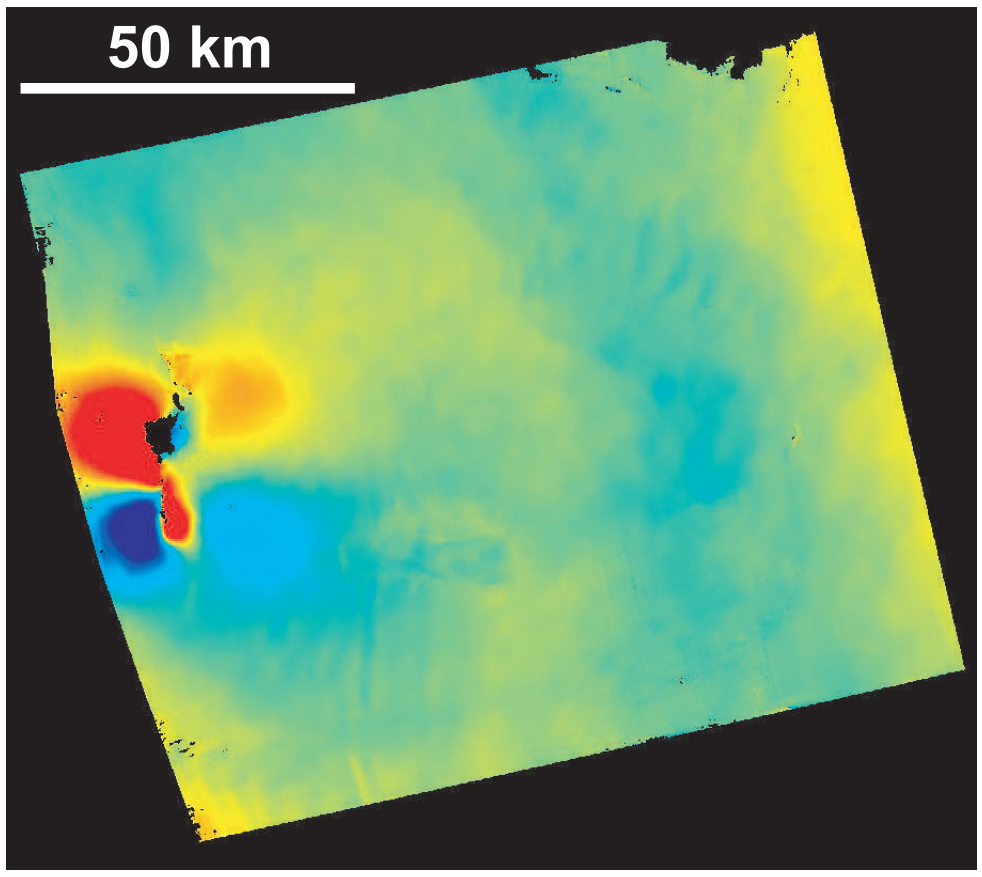
Zoom topography (blue to red ~ 1500m)

Descending interferogram; LOS Vector at earthquake location [East, north, up] ~ [0.3512, -0.0715, 0.9336]

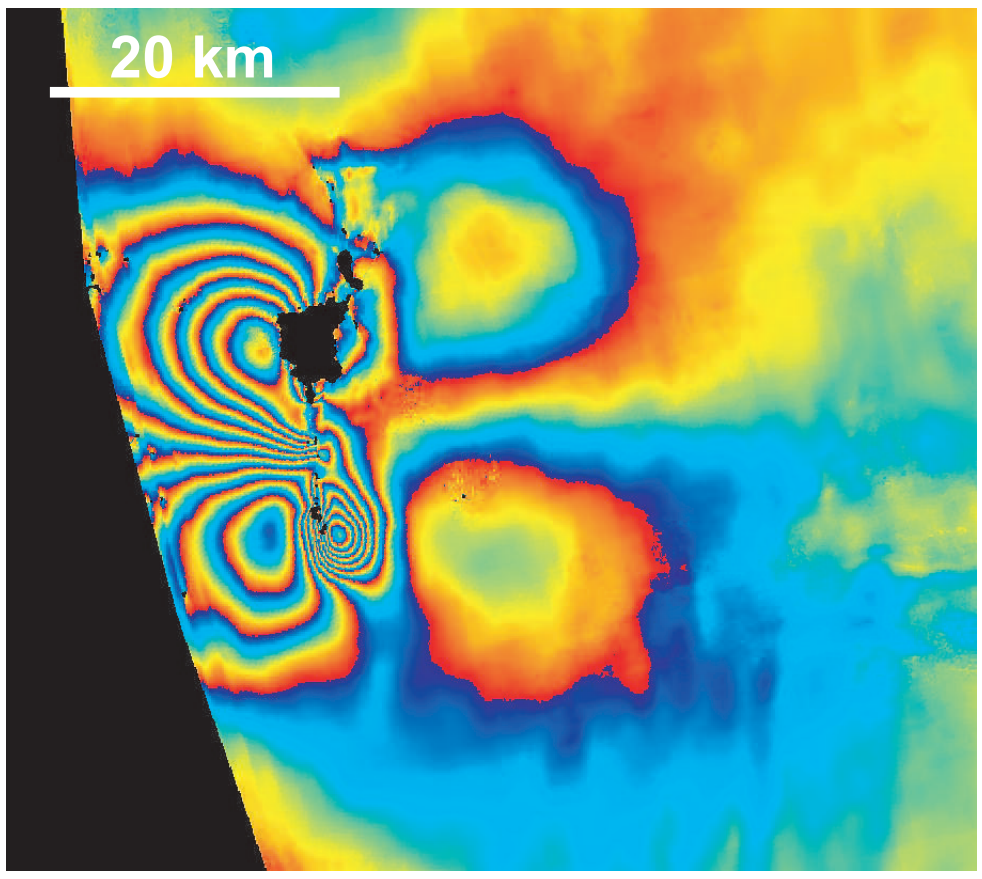
Earthquake 3



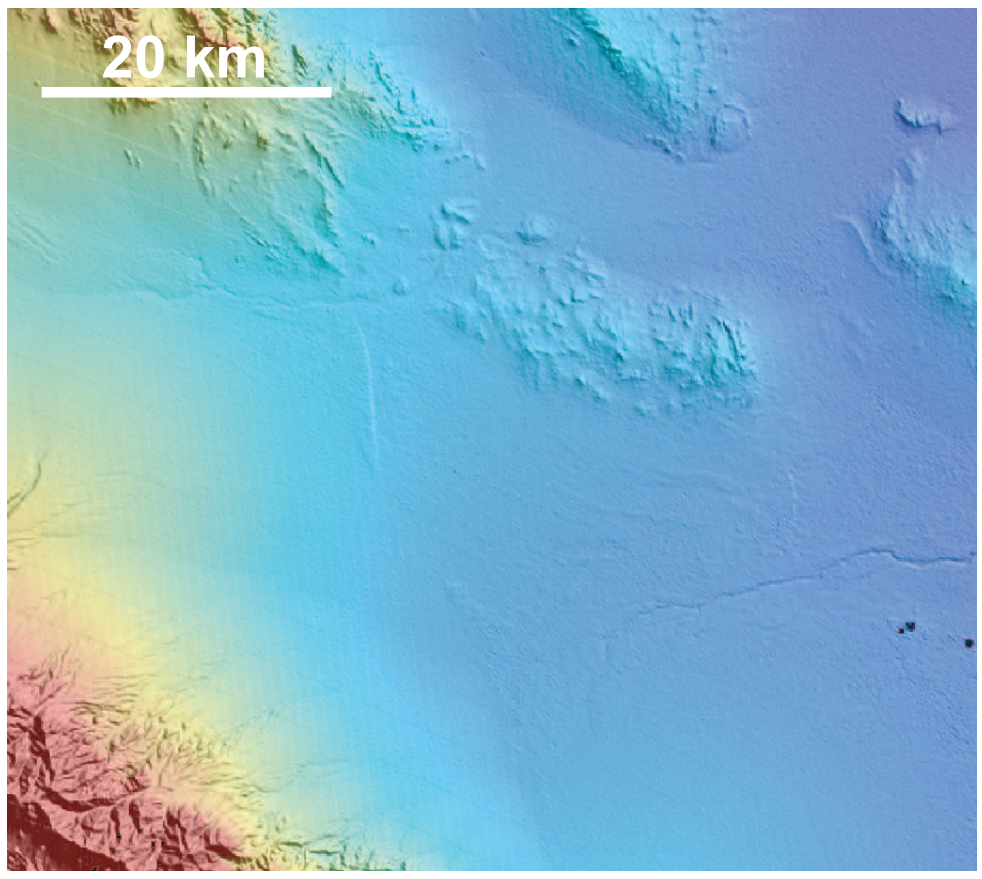
Whole image, wrapped



Whole image, unwrapped



Zoom, wrapped



Zoom topography (blue to red ~ 1500m)

Ascending interferogram; LOS Vector at earthquake location [East, north, up] ~ [-0.3186, -0.0671, 0.9455]