

Simple example of a tectonic Inverse Problem (inspired by the rifting crisis in Afar)

Solving the problem using the general probability formulation.

Albert Tarantola, February 2006

Forward Problem

These expressions solve the problem of evaluating the displacements caused when a fissure opens in a 2D "elastic" medium (see the PDF text for details).

```
In[537]:= Ux = x - X; Uy = y - Y; U = sqrt(Ux^2 + Uy^2);
Delta = Exp[delta];
f1 = Exp[-U/Delta];
fx = Sin[psi]; fy = Cos[psi];
beta = ArcCos[(fx Ux + fy Uy)/U];
alpha = pi/2 - beta;
f2 = Sin[alpha]^2;
Q = Exp[q];
ux = Q f1 f2 (Ux/U); uy = Q f1 f2 (Uy/U); uH = sqrt(ux^2 + uy^2); uz = uH/7;
```

The final expressions (for the three components of the displacements) are now used to define the functions solving the forward problem.

```
In[546]:= uxCAL[x_, y_, X_, Y_, delta_, psi_, q_] = ux;
uyCAL[x_, y_, X_, Y_, delta_, psi_, q_] = uy;
uzCAL[x_, y_, X_, Y_, delta_, psi_, q_] = uz;
```

Generating the synthetic data

We shall have (simulated) observations at six points, whose coordinates are as follows.

```
In[549]:= x1 = 11.0; y1 = 24.5; x2 = 13.0; y2 = 28.5; x3 = 14.0; y3 = 21.5;
x4 = 15.0; y4 = 17.5; x5 = 07.0; y5 = 16.5; x6 = 08.0; y6 = 24.5;
```

To simulate our observations, we arbitrarily take some "true model".

```
In[550]:= Xtrue = 10.0; Ytrue = 20.0; δtrue = 1.0; ψtrue = 0.5; qtrue = -0.5;
```

To the computed values, we shall add some random "errors", whose "size" is $\epsilon = 0.01$.

```
In[551]:= ε = 0.01;
random := Random[Real, {-ε, ε}]
SeedRandom[123]
```

We compute the synthetic data, and, to give some realism to the values, we round them leaving only three decimals.

```
In[554]:= K = 1000.;
```

```
In[555]:= ω = uxCAL[x1, y1, Xtrue, Ytrue, δtrue, ψtrue, qtrue];
uxOBS1 = Round[K (ω + random)] / K;
Print["ux1=", %%, " , ux1(rounded)=", %]
ω = uyCAL[x1, y1, Xtrue, Ytrue, δtrue, ψtrue, qtrue];
uyOBS1 = Round[K (ω + random)] / K;
Print["uy1=", %%, " , uy1(rounded)=", %]
ω = uzCAL[x1, y1, Xtrue, Ytrue, δtrue, ψtrue, qtrue];
uzOBS1 = Round[K (ω + random)] / K;
Print["uz1=", %%, " , uz1(rounded)=", %]
```

```
ux1=0.0222763 , ux1(rounded)=0.016
```

```
uy1=0.100243 , uy1(rounded)=0.099
```

```
uz1=0.0146698 , uz1(rounded)=0.022
```

```
In[564]:= ω = uxCAL[x2, y2, Xtrue, Ytrue, δtrue, ψtrue, qtrue];
uxOBS2 = Round[K (ω + random)] / K;
Print["ux2=", %%, " , ux2(rounded)=", %]
ω = uyCAL[x2, y2, Xtrue, Ytrue, δtrue, ψtrue, qtrue];
uyOBS2 = Round[K (ω + random)] / K;
Print["uy2=", %%, " , uy2(rounded)=", %]
ω = uzCAL[x2, y2, Xtrue, Ytrue, δtrue, ψtrue, qtrue];
uzOBS2 = Round[K (ω + random)] / K;
Print["uz2=", %%, " , uz2(rounded)=", %]
```

```
ux2=0.0071395 , ux2(rounded)=0.016
```

```
uy2=0.0202286 , uy2(rounded)=0.015
```

```
uz2=0.0030645 , uz2(rounded)=0
```

```
In[573]:=  $\omega$  = uxCAL[x3, y3, Xtrue, Ytrue,  $\delta$ true,  $\psi$ true, qtrue];
uxOBS3 = Round[K ( $\omega$  + random)] / K;
Print["ux3=", %%, " , ux3(rounded)=", %]
 $\omega$  = uyCAL[x3, y3, Xtrue, Ytrue,  $\delta$ true,  $\psi$ true, qtrue];
uyOBS3 = Round[K ( $\omega$  + random)] / K;
Print["uy3=", %%, " , uy3(rounded)=", %]
 $\omega$  = uzCAL[x3, y3, Xtrue, Ytrue,  $\delta$ true,  $\psi$ true, qtrue];
uzOBS3 = Round[K ( $\omega$  + random)] / K;
Print["uz3=", %%, " , uz3(rounded)=", %]
```

ux3=0.0676067 , ux3(rounded)=0.075

uy3=0.0253525 , uy3(rounded)=0.028

uz3=0.0103149 , uz3(rounded)=0.006

```
In[582]:=  $\omega$  = uxCAL[x4, y4, Xtrue, Ytrue,  $\delta$ true,  $\psi$ true, qtrue];
uxOBS4 = Round[K ( $\omega$  + random)] / K;
Print["ux4=", %%, " , ux4(rounded)=", %]
 $\omega$  = uyCAL[x4, y4, Xtrue, Ytrue,  $\delta$ true,  $\psi$ true, qtrue];
uyOBS4 = Round[K ( $\omega$  + random)] / K;
Print["uy4=", %%, " , uy4(rounded)=", %]
 $\omega$  = uzCAL[x4, y4, Xtrue, Ytrue,  $\delta$ true,  $\psi$ true, qtrue];
uzOBS4 = Round[K ( $\omega$  + random)] / K;
Print["uz4=", %%, " , uz4(rounded)=", %]
```

ux4=0.000091652 , ux4(rounded)=0.003

uy4=-0.000045826 , uy4(rounded)=0.006

uz4=0.0000146386 , uz4(rounded)=-0.002

```
In[591]:=  $\omega$  = uxCAL[x5, y5, Xtrue, Ytrue,  $\delta$ true,  $\psi$ true, qtrue];
uxOBS5 = Round[K ( $\omega$  + random)] / K;
Print["ux5=", %%, " , ux5(rounded)=", %]
 $\omega$  = uyCAL[x5, y5, Xtrue, Ytrue,  $\delta$ true,  $\psi$ true, qtrue];
uyOBS5 = Round[K ( $\omega$  + random)] / K;
Print["uy5=", %%, " , uy5(rounded)=", %]
 $\omega$  = uzCAL[x5, y5, Xtrue, Ytrue,  $\delta$ true,  $\psi$ true, qtrue];
uzOBS5 = Round[K ( $\omega$  + random)] / K;
Print["uz5=", %%, " , uz5(rounded)=", %]
```

ux5=-0.0693041 , ux5(rounded)=-0.077

uy5=-0.0808548 , uy5(rounded)=-0.083

uz5=0.0152132 , uz5(rounded)=0.018

```
In[600]:=  $\omega$  = uxCAL[x6, y6, Xtrue, Ytrue,  $\delta$ true,  $\psi$ true, qtrue];
uxOBS6 = Round[K ( $\omega$  + random)] / K;
Print["ux6=", %%, " , ux6(rounded)=", %]
 $\omega$  = uyCAL[x6, y6, Xtrue, Ytrue,  $\delta$ true,  $\psi$ true, qtrue];
uyOBS6 = Round[K ( $\omega$  + random)] / K;
Print["uy6=", %%, " , uy6(rounded)=", %]
 $\omega$  = uzCAL[x6, y6, Xtrue, Ytrue,  $\delta$ true,  $\psi$ true, qtrue];
uzOBS6 = Round[K ( $\omega$  + random)] / K;
Print["uz6=", %%, " , uz6(rounded)=", %]
```

```
ux6=-0.0148412 , ux6(rounded)=-0.02
```

```
uy6=0.0333928 , uy6(rounded)=0.032
```

```
uz6=0.00522033 , uz6(rounded)=0.009
```

Plot of the Data

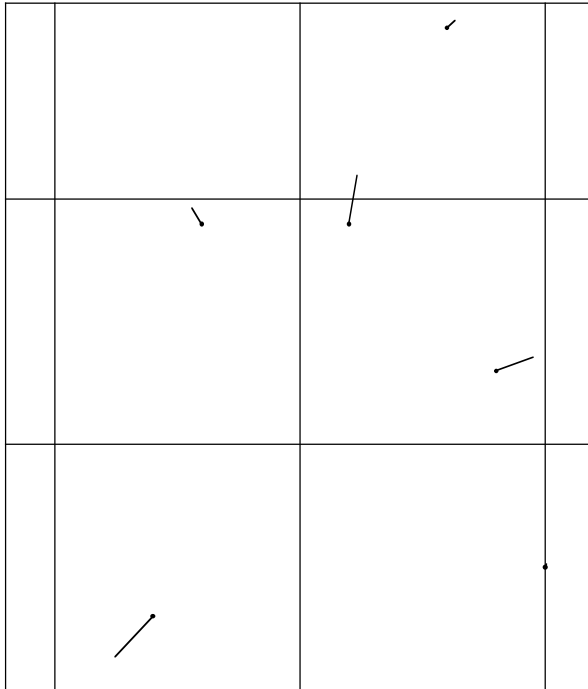
Note: the displacements have been multiplied by 10, to make them visible.

```
In[609]:= MeasuringPoints = Graphics[{
  Point[{x1, y1}], Point[{x2, y2}],
  Point[{x3, y3}], Point[{x4, y4}], Point[{x5, y5}], Point[{x6, y6}]
}];
Arrows = Graphics[{
  Line[{x1, y1}, {x1 + 10 uxOBS1, y1 + 10 uyOBS1}],
  Line[{x2, y2}, {x2 + 10 uxOBS2, y2 + 10 uyOBS2}],
  Line[{x3, y3}, {x3 + 10 uxOBS3, y3 + 10 uyOBS3}],
  Line[{x4, y4}, {x4 + 10 uxOBS4, y4 + 10 uyOBS4}],
  Line[{x5, y5}, {x5 + 10 uxOBS5, y5 + 10 uyOBS5}],
  Line[{x6, y6}, {x6 + 10 uxOBS6, y6 + 10 uyOBS6}]
}];
```

Let us also add some coordinate lines...

```
In[611]:= left = Line[{4, 15}, {4, 29}];
top = Line[{4, 29}, {16, 29}];
right = Line[{16, 29}, {16, 15}];
bottom = Line[{16, 15}, {4, 15}];
vert1 = Line[{5, 15}, {5, 29}];
vert2 = Line[{10, 15}, {10, 29}];
vert3 = Line[{15, 15}, {15, 29}];
hor1 = Line[{16, 20}, {4, 20}];
hor2 = Line[{16, 25}, {4, 25}];
Lines = Graphics[{left, top, right, bottom, vert1, vert2, vert3, hor1, hor2}];
```

In[621]:= Show[{MeasuringPoints, Arrows, Lines}, AspectRatio -> Automatic]



The a priori model and its uncertainties.

We assume that we have the following a priori information on the model parameters:

$$X = 12 \pm 20 ; Y = 12 \pm 20 ; \delta = 2 \pm 1 ; \psi = \text{ArcTan}[1] \pm 1 ; q = 0 \pm 2 .$$

This is a quite weak information, but it is useful for two reasons: i) sometimes - when we have low quality data - it may help obtaining reasonable solutions; a ii) the algorithm uses a metric in the model space, and the convergence is more rapid when this metric is reasonable.

In[622]:= **Xprior = 12.; Yprior = 12.; δ prior = 2.; ψ prior = ArcTan[1.]; qprior = 0;
 σ X = 20.; σ Y = 20.; σ δ = 1.; σ ψ = 1.; σ q = 2;**

Note that I use the von Mises distribution for the angle ψ .

In[624]:= **prior = Exp[- $\frac{1}{2} \frac{(X - X_{\text{prior}})^2}{\sigma X^2}$] Exp[- $\frac{1}{2} \frac{(Y - Y_{\text{prior}})^2}{\sigma Y^2}$]
 Exp[- $\frac{1}{2} \frac{(\delta - \delta_{\text{prior}})^2}{\sigma \delta^2}$] Exp[$\frac{\text{Cos}[\psi - \psi_{\text{prior}}]}{\sigma \psi^2}$] Exp[- $\frac{1}{2} \frac{(q - q_{\text{prior}})^2}{\sigma q^2}$];**

Random generation of prior models (printing the results)

```
In[662]:=  $\rho[X_, Y_, \delta_, \psi_, q_] = \text{prior};$ 
```

```
In[663]:=  $\Delta X = 40.; \Delta Y = 40.; \Delta \delta = 2.; \Delta \psi = 2.; \Delta q = 4.;$ 
perturb := {Xtest = Xcurrent + Random[Real, {- $\Delta X$ ,  $\Delta X$ ]},
  Ytest = Ycurrent + Random[Real, {- $\Delta Y$ ,  $\Delta Y$ ]},
   $\delta$ test =  $\delta$ current + Random[Real, {- $\Delta \delta$ ,  $\Delta \delta$ ]},
   $\psi$ test =  $\psi$ current + Random[Real, {- $\Delta \psi$ ,  $\Delta \psi$ ]},
  qtest = qcurrent + Random[Real, {- $\Delta q$ ,  $\Delta q$ ]},
   $\rho$ test =  $\rho$ [Xtest, Ytest,  $\delta$ test,  $\psi$ test, qtest]}
print := Print["i=", i, " ", {Xcurrent, Ycurrent,
   $\delta$ current,  $\psi$ current, qcurrent}, " ",  $\rho$ current]
renew := {Xcurrent = Xtest, Ycurrent = Ytest,  $\delta$ current =  $\delta$ test,
   $\psi$ current =  $\psi$ test, qcurrent = qtest,  $\rho$ current =  $\rho$ test};
work := {renew, print}
```

```
In[668]:= Xcurrent = 10.5; Ycurrent = 19.5;  $\delta$ current = 1.2;  $\psi$ current = 0.6; qcurrent = -0.6;
 $\rho$ current =  $\rho$ [Xcurrent, Ycurrent,  $\delta$ current,  $\psi$ current, qcurrent];
SeedRandom[321]
Do[{
  perturb,
  If[ $\rho$ test  $\geq$   $\rho$ current, work, If[Random[Real, {0, 1}]  $\leq$  ( $\rho$ test /  $\rho$ current), work]]
}, {i, 1, 50}]
```

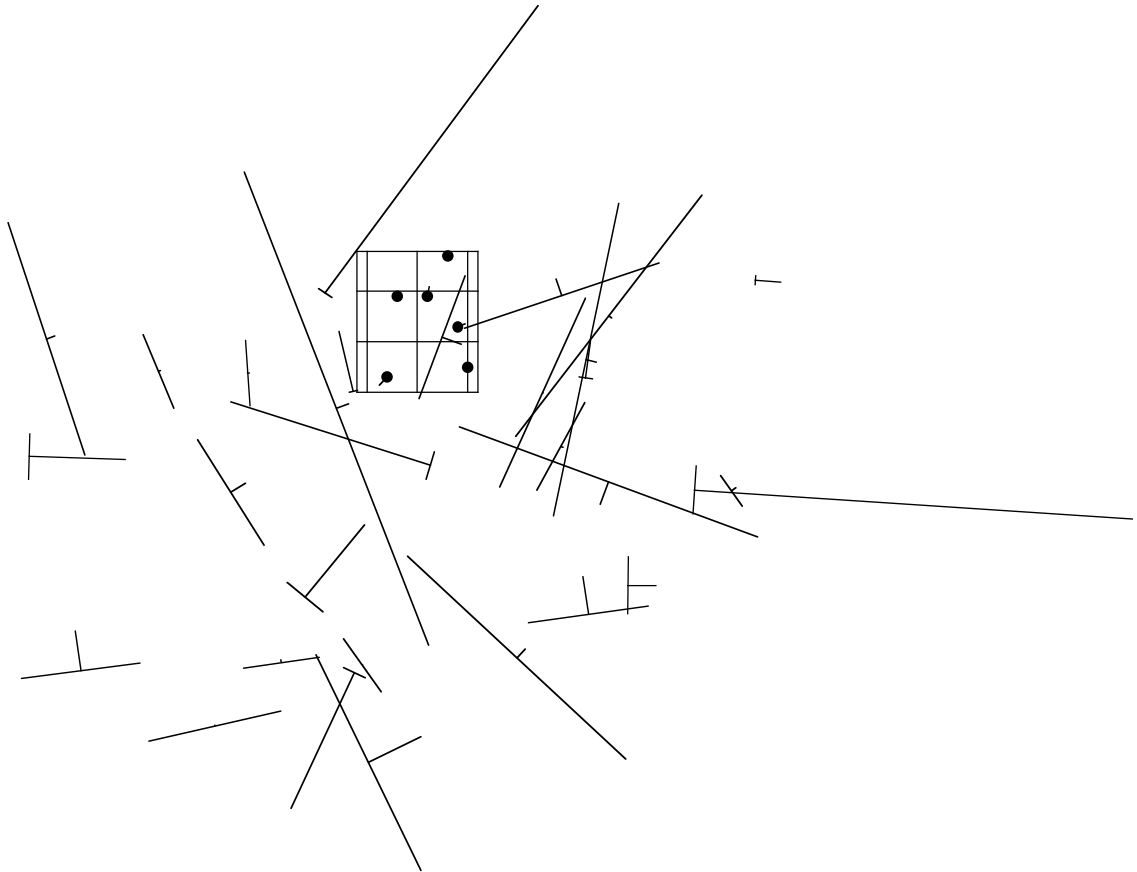
```
i=6 {14.5308, 9.45477, 2.42232, -0.199778, 1.04396} 1.36507
i=8 {-12.7354, 12.1844, 0.878071, 0.731394, -0.115123} 0.672117
i=15 {7.9971, 19.4707, 0.396735, 0.246066, 1.59822} 0.433329
i=16 {21.4877, 12.9049, 1.5292, -0.743342, 0.0999214} 0.832291
i=25 {13.1046, 23.5829, 3.44215, 1.15445, -2.78953} 0.286753
i=27 {-24.7503, 16.4941, 2.74893, 0.550132, 0.562506} 0.346102
i=29 {4.53459, 16.8109, 1.3478, 1.748, 3.69541} 0.2353
i=33 {1.96883, 6.86378, 0.588317, 2.80402, 4.30593} 0.0201245
i=35 {35.2883, 13.9126, 2.4193, 1.11061, 5.51372} 0.0267035
i=37 {9.04817, 41.8371, 2.63544, 2.88953, 4.77766} 0.00921182
i=42 {35.6828, 18.5181, 2.80149, 4.86454, 5.38026} 0.00506418
i=44 {-0.66537, 29.758, 3.01684, 3.4115, 2.30157} 0.0710845
i=45 {26.1404, 36.0893, 1.53059, 4.36636, 6.09656} 0.0013116
i=48 {5.06764, 26.4078, 3.3258, 4.90178, 2.89537} 0.0603455
i=49 {-13.8269, -4.49379, 1.54466, 6.78743, 3.0962} 0.219791
```

Random generation of prior models (drawing the results)

```
In[635]:= Fissure = Table[0, {100}];
j = 0;
store := {j = j + 1,
  center = {Xcurrent, Ycurrent},
  Δ = Exp[δcurrent],
  point1 = {Xcurrent + (Δ/2) Cos[ψcurrent], Ycurrent - (Δ/2) Sin[ψcurrent]},
  point2 = {Xcurrent - (Δ/2) Cos[ψcurrent], Ycurrent + (Δ/2) Sin[ψcurrent]},
  Q = Exp[qcurrent],
  pointQ = {Xcurrent + Q Sin[ψcurrent], Ycurrent + Q Cos[ψcurrent]},
  lineP = Line[{point1, point2}],
  lineQ = Line[{center, pointQ}],
  Fissure[[j]] = Graphics[{lineP, lineQ}]
}
work := {renew, store}

In[639]:= Do[{
  perturb,
  If[ρtest ≥ ρcurrent, work, If[Random[Real, {0, 1}] ≤ (ρtest / ρcurrent), work]]
}, {i, 1, 150}]
```

```
In[640]:= FissureSet = {
  Fissure[[01]], Fissure[[02]], Fissure[[03]], Fissure[[04]], Fissure[[05]],
  Fissure[[06]], Fissure[[07]], Fissure[[08]], Fissure[[09]], Fissure[[10]],
  Fissure[[11]], Fissure[[12]], Fissure[[13]], Fissure[[14]], Fissure[[15]],
  Fissure[[16]], Fissure[[17]], Fissure[[18]], Fissure[[19]], Fissure[[20]],
  Fissure[[21]], Fissure[[22]], Fissure[[23]], Fissure[[24]], Fissure[[25]],
  Fissure[[26]], Fissure[[27]], Fissure[[28]], Fissure[[29]], Fissure[[30]]};
Show[{MeasuringPoints, Arrows, FissureSet, Lines},
  PlotRange -> All, AspectRatio -> Automatic]
```



Resolution of the inverse problem using the general probability formulation

The posterior probability distribution.

Warning: for some strange reason, the developers of *Mathematica* use the same symbol (a dot) to designate a product of two matrices and the (elementary) scalar product of two vectors. The expression below should be written $S2 = \text{Transpose}[(\text{tcal-tobs}).\text{Inverse}[\text{CD}].(\text{tcal-tobs})]$, but *Mathematica* would misunderstand the expression.

```

In[642]:= likelihood = Exp[- $\frac{\text{Abs}[\text{uxCAL}[x1, y1, X, Y, \delta, \psi, q] - \text{uxOBS1}]}{\epsilon}$ ]
Exp[- $\frac{\text{Abs}[\text{uyCAL}[x1, y1, X, Y, \delta, \psi, q] - \text{uyOBS1}]}{\epsilon}$ ]
Exp[- $\frac{\text{Abs}[\text{uzCAL}[x1, y1, X, Y, \delta, \psi, q] - \text{uzOBS1}]}{\epsilon}$ ]
Exp[- $\frac{\text{Abs}[\text{uxCAL}[x2, y2, X, Y, \delta, \psi, q] - \text{uxOBS2}]}{\epsilon}$ ]
Exp[- $\frac{\text{Abs}[\text{uyCAL}[x2, y2, X, Y, \delta, \psi, q] - \text{uyOBS2}]}{\epsilon}$ ]
Exp[- $\frac{\text{Abs}[\text{uzCAL}[x2, y2, X, Y, \delta, \psi, q] - \text{uzOBS2}]}{\epsilon}$ ]
Exp[- $\frac{\text{Abs}[\text{uxCAL}[x3, y3, X, Y, \delta, \psi, q] - \text{uxOBS3}]}{\epsilon}$ ]
Exp[- $\frac{\text{Abs}[\text{uyCAL}[x3, y3, X, Y, \delta, \psi, q] - \text{uyOBS3}]}{\epsilon}$ ]
Exp[- $\frac{\text{Abs}[\text{uzCAL}[x3, y3, X, Y, \delta, \psi, q] - \text{uzOBS3}]}{\epsilon}$ ]
Exp[- $\frac{\text{Abs}[\text{uxCAL}[x4, y4, X, Y, \delta, \psi, q] - \text{uxOBS4}]}{\epsilon}$ ]
Exp[- $\frac{\text{Abs}[\text{uyCAL}[x4, y4, X, Y, \delta, \psi, q] - \text{uyOBS4}]}{\epsilon}$ ]
Exp[- $\frac{\text{Abs}[\text{uzCAL}[x4, y4, X, Y, \delta, \psi, q] - \text{uzOBS4}]}{\epsilon}$ ]
Exp[- $\frac{\text{Abs}[\text{uxCAL}[x5, y5, X, Y, \delta, \psi, q] - \text{uxOBS5}]}{\epsilon}$ ]
Exp[- $\frac{\text{Abs}[\text{uyCAL}[x5, y5, X, Y, \delta, \psi, q] - \text{uyOBS5}]}{\epsilon}$ ]
Exp[- $\frac{\text{Abs}[\text{uzCAL}[x5, y5, X, Y, \delta, \psi, q] - \text{uzOBS5}]}{\epsilon}$ ]
Exp[- $\frac{\text{Abs}[\text{uxCAL}[x6, y6, X, Y, \delta, \psi, q] - \text{uxOBS6}]}{\epsilon}$ ]
Exp[- $\frac{\text{Abs}[\text{uyCAL}[x6, y6, X, Y, \delta, \psi, q] - \text{uyOBS6}]}{\epsilon}$ ]
Exp[- $\frac{\text{Abs}[\text{uzCAL}[x6, y6, X, Y, \delta, \psi, q] - \text{uzOBS6}]}{\epsilon}$ ];
posterior = prior * likelihood;

```

Random generation of posterior models (printing the results)

```

In[672]:=  $\sigma[X_, Y_, \delta_, \psi_, q_] = 10^7$  posterior;

```

In[673]:=

```

ΔX = 0.5; ΔY = 0.5; Δδ = 0.5; Δψ = 0.5; Δq = 0.5;
perturb := {Xtest = Xcurrent + Random[Real, {-ΔX, ΔX}],
  Ytest = Ycurrent + Random[Real, {-ΔY, ΔY}],
  δtest = δcurrent + Random[Real, {-Δδ, Δδ}],
  ψtest = ψcurrent + Random[Real, {-Δψ, Δψ}],
  qtest = qcurrent + Random[Real, {-Δq, Δq}],
  σtest = σ[Xtest, Ytest, δtest, ψtest, qtest]}
print := Print["i=", i, " ", {Xcurrent, Ycurrent,
  δcurrent, ψcurrent, qcurrent}, " ", σcurrent]
renew := {Xcurrent = Xtest, Ycurrent = Ytest, δcurrent = δtest,
  ψcurrent = ψtest, qcurrent = qtest, σcurrent = σtest};
work := {renew, print}

```

In[678]:=

```

Xcurrent = 10.5; Ycurrent = 19.5; δcurrent = 1.2; ψcurrent = 0.6; qcurrent = -0.6;
σcurrent = σ[Xcurrent, Ycurrent, δcurrent, ψcurrent, qcurrent];
SeedRandom[123]
Do[{
  perturb,
  If[σtest ≥ σcurrent, work, If[Random[Real, {0, 1}] ≤ (σtest / σcurrent), work]]
}, {i, 1, 2000}]

```

```

i=3 {10.1385, 19.3947, 1.3222, 0.319747, -0.665161} 4.25995
i=8 {9.73575, 19.5604, 0.98987, 0.549839, -0.745089} 4.37289
i=14 {9.62412, 19.3538, 1.47788, 0.454551, -1.22263} 143.582
i=107 {9.80123, 19.4017, 1.34567, 0.531656, -0.977688} 349.123
i=137 {9.89126, 19.3308, 1.27536, 0.479469, -0.931388} 475.433
i=185 {10.2064, 19.3018, 1.12986, 0.354185, -0.502977} 434.238
i=214 {10.3461, 19.7006, 0.959783, 0.418338, -0.462676} 180.812
i=249 {10.1436, 19.5032, 1.25361, 0.375862, -0.691515} 344.433
i=264 {9.79152, 19.4805, 1.09397, 0.471018, -0.608648} 313.94
i=333 {9.99163, 19.6843, 1.38137, 0.434719, -1.04745} 748.883
i=514 {9.89335, 19.7044, 1.07985, 0.408568, -0.588876} 2562.07
i=609 {9.9584, 19.7345, 0.922737, 0.583944, -0.408811} 1901.38
i=615 {9.60747, 19.6246, 1.08341, 0.605817, -0.674141} 247.477
i=701 {9.31309, 19.4199, 1.19419, 0.486664, -0.88011} 5.67036
i=704 {9.75024, 19.4782, 1.47876, 0.619556, -1.36317} 9.02695
i=708 {9.86364, 19.7352, 1.40909, 0.469263, -0.882654} 21.8724
i=720 {9.50737, 19.8248, 1.29302, 0.550714, -0.860226} 92.9559
i=746 {9.79848, 20.234, 1.0042, 0.796076, -0.550487} 8.05655
i=782 {9.47435, 20.2755, 0.812687, 0.682606, -0.182851} 65.6328
i=811 {9.57232, 20.1523, 0.941889, 0.770742, -0.432691} 30.1477
i=887 {10.0404, 20.3608, 0.88409, 0.436153, 0.00454853} 0.355442
i=894 {10.4505, 19.9283, 0.496143, 0.407656, 0.46026} 0.659157

```

```

i=952 {10.0908, 20.0136, 0.434416, 0.43411, 0.853414} 616.204
i=1151 {10.0734, 20.0233, 0.604133, 0.363451, 0.372121} 60.8448
i=1253 {10.1196, 20.1214, 0.586559, 0.586879, 0.351912} 85.9509
i=1407 {10.2997, 20.0205, 0.707852, 0.407109, -0.00772368} 98.9812
i=1461 {10.1038, 20.0695, 0.798261, 0.499709, -0.232842} 96.8689
i=1485 {10., 19.6919, 1.04192, 0.459991, -0.501671} 6431.85
i=1653 {10.1113, 19.662, 0.874362, 0.370838, -0.286307} 809.44
i=1798 {10.2361, 20.036, 0.867758, 0.418774, -0.311048} 361.56
i=1838 {10.1017, 19.7942, 0.895955, 0.418621, -0.16955} 2939.41
i=1902 {10.1293, 19.594, 0.841565, 0.320719, -0.126821} 2761.23

```

Random generation of posterior models (drawing the results)

```

In[682]:= Fissure = Table[0, {100}];
j = 0;
store := {j = j + 1,
  center = {Xcurrent, Ycurrent},
  Δ = Exp[δcurrent],
  point1 = {Xcurrent + (Δ / 2) Cos[ψcurrent], Ycurrent - (Δ / 2) Sin[ψcurrent]},
  point2 = {Xcurrent - (Δ / 2) Cos[ψcurrent], Ycurrent + (Δ / 2) Sin[ψcurrent]},
  Q = Exp[qcurrent],
  pointQ = {Xcurrent + Q Sin[ψcurrent], Ycurrent + Q Cos[ψcurrent]},
  lineP = Line[{point1, point2}],
  lineQ = Line[{center, pointQ}],
  Fissure[[j]] = Graphics[{lineP, lineQ}]
}
work := {renew, store}

In[686]:= Do[{
  perturb,
  If[σtest ≥ σcurrent, work, If[Random[Real, {0, 1}] ≤ (σtest / σcurrent), work]]
}, {i, 1, 2000}]

In[687]:= FissureSet = {
  Fissure[[01]], Fissure[[02]], Fissure[[03]], Fissure[[04]], Fissure[[05]],
  Fissure[[06]], Fissure[[07]], Fissure[[08]], Fissure[[09]], Fissure[[10]],
  Fissure[[11]], Fissure[[12]], Fissure[[13]], Fissure[[14]]};
Show[{MeasuringPoints, Arrows, FissureSet, Lines},
  PlotRange → All, AspectRatio → Automatic]
Show[{FissureSet}, PlotRange → All, AspectRatio → Automatic]

```

