Geodetic Modeling

Jonathan M. Lees
University of North Carolina, Chapel Hill
Department of Geological Sciences

Sat Nov 27 07:57:26 EST 2010

1 Mogi Modeling: Introduction

Mogi Modeling is used to estimate the location and size of anomalies in the earth when there is a change in size or density. This approach is taken when trying to investigate the nature of injection of magma in the crust.

2 Theory

The equations derived by Mogi are approximations used in most mogi modeling programs. (see, Mogi, K., (1958). Relations between the Eruptions of Various Volcanoes and the Deformations of the Ground Surfaces around them. Bulletin of the Earthquake Research Institute, 36: 99-134.)

The original equations published in Mogi's paper were presented as,

$$\Delta d = \frac{3a^3P}{4\mu}\frac{d}{(f^2+d^2)^{3/2}} \tag{2.1}$$

$$\Delta h = \frac{3a^3P}{4\mu}\frac{f}{(f^2+d^2)^{3/2}} \tag{2.2}$$

where d is the surface radial distance from the source point, A, f is the depth to the center of the sphere, P is the change inhydor-static pressure in the sphere, a is the radius of the sphere, Δd is the displacement in the direction of the radial axis on the surface, and Δh is the vertical displacement on the surface.

Note that the approximation suggests that the shape of the curves depend only on the depth. if the data is normalized to the maximum value, then the only parameters that controls the fit to the data is the depth to the center of the anomaly.

These are compared to observations and models are extracted based on least squares type criteria.

3 Example

The example provided is from the a paper on the Phlegraean Fields, Italy. The data was digitized off the screen from a scanned figure extracted from the paper. There may thus be slight discrepancies from the original data set collected by the authors. (see, Berrino, G., Corrado, G., Luongo, G. and Toro, B., (1984). Ground deformation and gravity changes accompanying the 1982 Pozzuoli Uplift. Bulletin Volcanologique, 47(2): 187-200.)

It consists of a set of distances (X) and normalized values of vertical deformation recorded in the vicinity of the volcano.

```
> require(geophys)
> data(PXY)
```

```
> plot(PXY$x, PXY$y, xlab="Distance, km", ylab="Fractional Displacement")
```

4 Modeling

A single instance of the Mogi model is calculated with the program mogiM. mogiM computes radial and vertical displacements Ur and Uz, ground tilt Dt, radial and tangential strain Er and Et on surface, at a radial distance R from the top of the source due to a hydrostatic pressure inside a sphere of radius A at depth F, in a homogeneous, semi-infinite elastic body and approximation for $A \ll F$ (center of dilatation). Formula by Anderson [1936] and Mogi [1958].

Input parameters for the program:

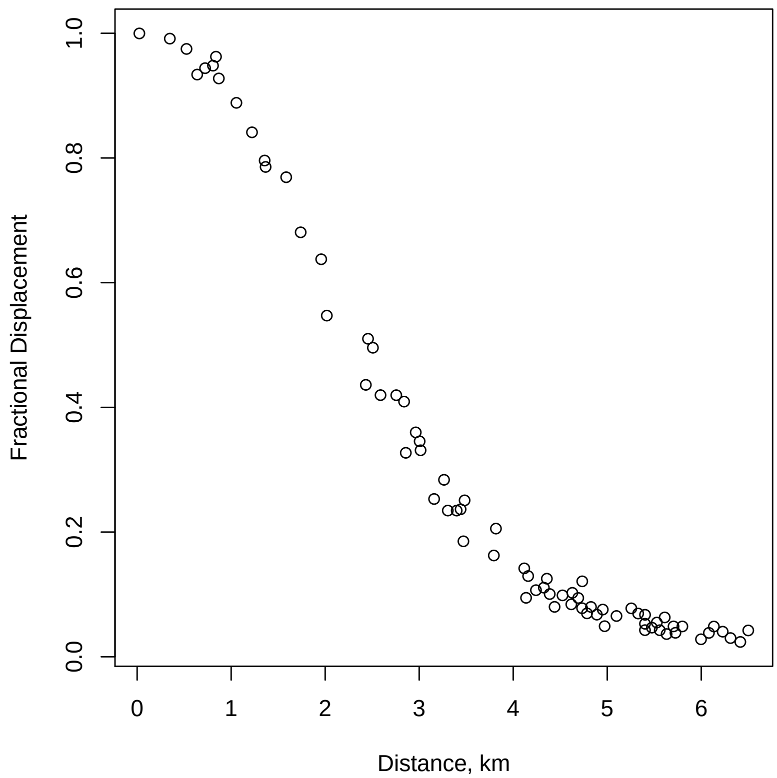

Figure 1: Mogi Data from the Phlegraean Fields, Italy

- R: Hoirizontal Distance frm source, m
- F: Depth below surface, m, positive down
- A: radius of magma chamber
- P: hydrostatic pressure change in the sphere
- E: elasticity (Young's modulus)
- ν: Poisson's ratio

Example:

```
>       delV = 2.3E13/(100^3)    ##### (convert to meter^3 from cm^3)
>       eF = 2.8E5/100      ##### (convert to meter from cm    )
>       EX = seq(from=0, by=100, to= 9000)
>       Atest = mogiM(R=EX,F=eF,A=delV)
>        plot(PXY, pch=6, col='purple', xlim=c(0,9), ylim=c(0, 1), xlab="Distance,
>           ###  model
>           lines(EX/1000, Atest$uz/max(Atest$uz))
```

Consider several models run with slightly different parameters.

```
> Ktest = list()
> eF = 2.8E5/100
> pct = eF*0.1
> testEFS = seq(from=eF-pct, to=eF+pct, length=10)
> for( i in 1:length(testEFS))
   {
    delV = 2.3E13/(100^3)    ##### (convert to meter^3 from cm^3)
      eF =  testEFS[i]     ##### (convert to meter from cm    )
      EX = seq(from=0, by=100, to= 9000)
      Ktest[[i]] = mogiM(R=EX,F=eF,A=delV)
  }
```

Plotting these with the original data:

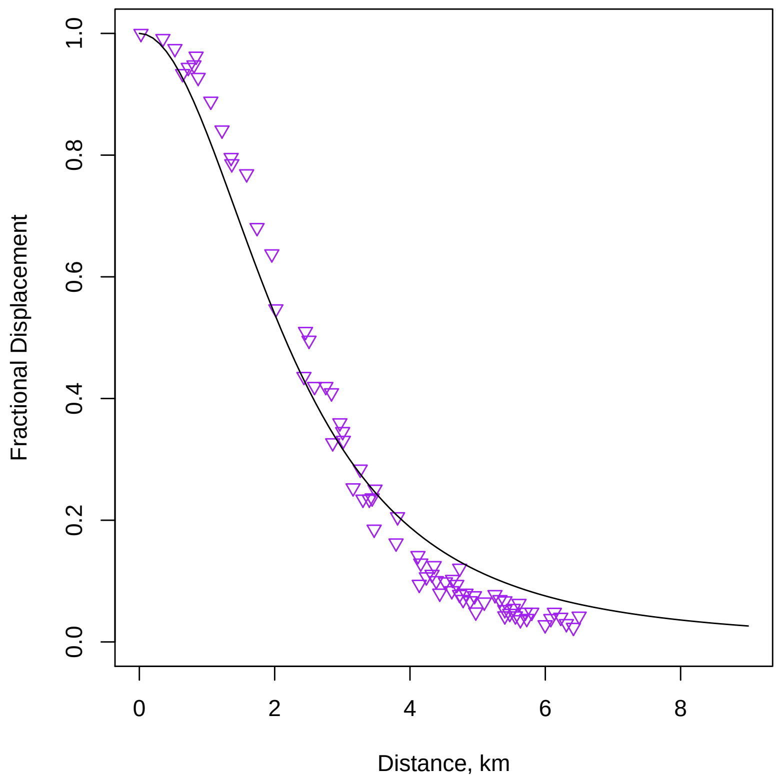

Figure 2: Mogi Model

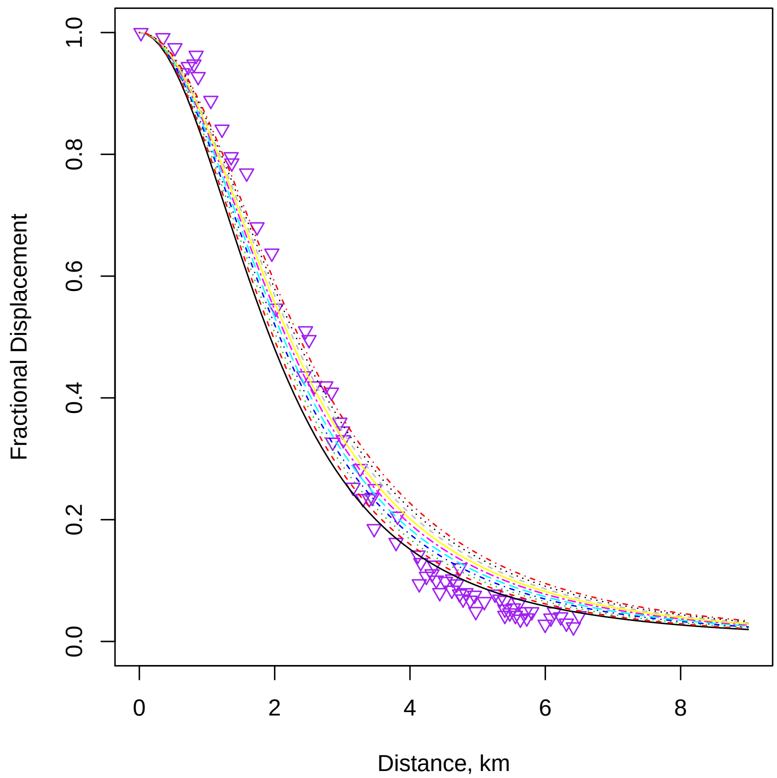

Figure 3: Mogi Models with Varying F(depth)

```
>    plot(PXY, pch=6, col='purple', xlim=c(0,9), ylim=c(0, 1), xlab="Distance, km
> for( i in 1:length(testEFS))
   {
    lines(EX/1000, Ktest[[i]]$uz/max(Ktest[[i]]$uz), col=i, lty=i)
    }


> Dtest = list()
>  eF = 2.8E5/100
>  delV = 2.3E13/(100^3)    ##### (convert to meter^3 from cm^3)
> pct = delV*0.1
> testDEL = seq(from=delV-pct, to=delV+pct, length=10)
>    EX = seq(from=0, by=100, to= 9000)
> for( i in 1:length(testDEL))
   {
     delV = testDEL[i]
      Dtest[[i]] = mogiM(R=EX,F=eF,A=delV)
  }
```

Plotting these with the original data:

```
>    plot(PXY, pch=6, col='purple', xlim=c(0,9), ylim=c(0, 1), xlab="Distance, km
> for( i in 1:length(testDEL))
   {
    lines(EX/1000, Dtest[[i]]$uz/max(Dtest[[i]]$uz), col=i, lty=i)
     }
```

These are all the same because the model output is normalized, so it does not depend on this parameter.

5 Grid Search

Sometimes it is possible to find the best model by exploring the full space of all possible models within a range of models. First we create a function that can be used to check a model against the observed data. The function calculates the misfit or predicted versus observed data by summing the root-mean-square (RMS) difference. This is then used as a score to assess the appropriateness of a model.

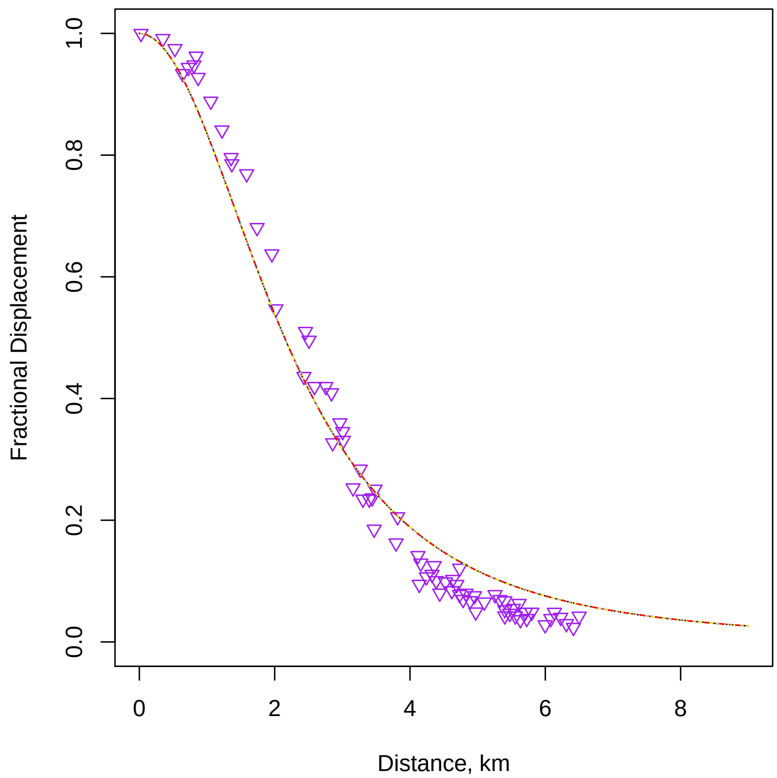

Figure 4: Mogi Models with Varying DelV

```
> fr<-function(x)
   {
     Atest = mogiM(R=PXY$x*1000 ,F=x[1],A=x[2])
     rms = sum ( (PXY$y - Atest$uz/max(Atest$uz))^2 )
    return(rms)
   }
```

Since the model does not depend on any other parameter other than the depth, we can search solely on this parameter. If the data provided were absolute deformation values, rather than normalized to maximum values, We could try to extract other parameters from the grid search.

```
>  eF = 2.8E5/100
> pct = eF*0.1
>  testEFS = seq(from=eF-pct, to=eF+pct, length=100)
> delV = 2.3E13/(100^3)
> fgrid = rep(0, length=length(testEFS))
>    EX = seq(from=0, by=100, to= 9000)
> for( j in 1:length(testEFS))
   {

     eF =  testEFS[j]
      fgrid[j] = fr(c(eF, delV))
   }
```

The minimum of this curve is the solution we are seeking.

```
> fbest = testEFS[which.min(fgrid)]
> print(paste("Min Solution at=", fbest))
[1] "Min Solution at= 2785.85858585859"
```

To see how this results in best solution, plot fgrid,

```
> plot(testEFS , fgrid)
> abline(v=fbest, col='red')
```

Of course this solution depends on the steps of the grid search, and details of the choice of steps. Furthermore, we are calculating many models, perhaps unnecessarily. In the next section we will use a program to find the optimal choice of F(depth) without requiring us to choose that grid spacing and in a much more efficient manner.

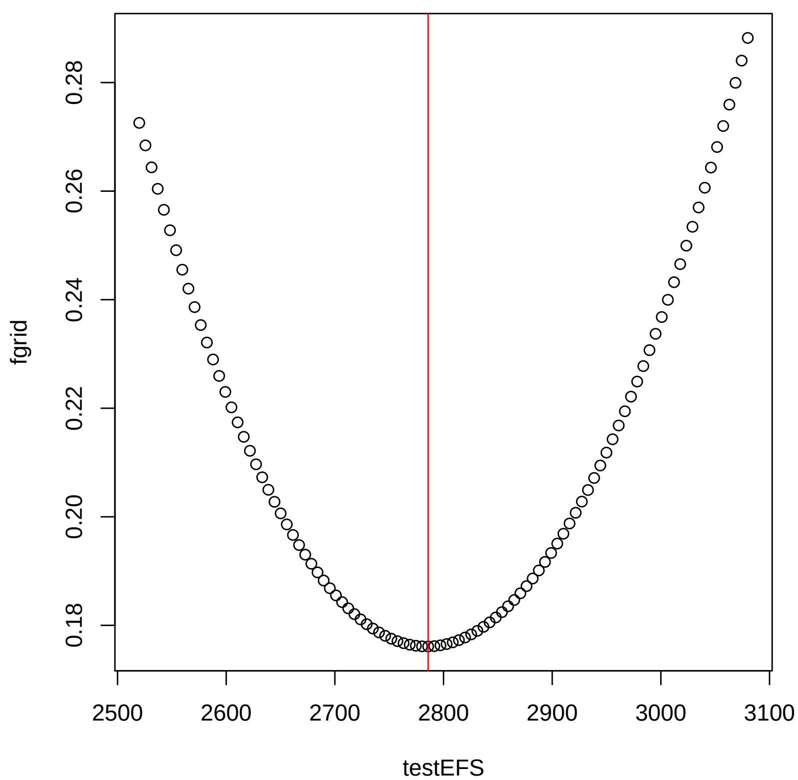

Figure 5: Grid search for minimum model

6 Least Squares Modeling

How can we find the best model that fits the data? we need to use some modeling tools. there are a number of tools available in R. In the following I use a program from the packages stats. It is called optim.

First set up a functiont hat evaluates the mogi model, given a set of parameters, and returns a score (root-mean-square difference) between the thoeretical and observed values.

```
>    library(stats)
> fr<-function(x)
   {
     Atest = mogiM(R=PXY$x*1000 ,F=x[1],A=x[2])
     rms = sum ( (PXY$y - Atest$uz/max(Atest$uz))^2 )
    return(rms)
   }
```

next, set up an initial input and let optim find the best model that converges to the the minimum solution.

```
>      xin = c(2600, 2.0e+07)
>      FOUT = optim(xin , fr)
>     print(FOUT$par)
```

```
[1]     2784.536 15999907.732
```

Note the difference between this estiamte and the one found previously by grid search.

```
> FOUT$par[1]
```

```
[1] 2784.536
```

```
> fbest
```

```
[1] 2785.859
```

```
> FOUT$par[1] - fbest
```

```
[1] -1.322606
```

There is a slight discrepancy. How much better is the optimum solution? We can check:

```
>  xin = FOUT$par
>  T1 = fr(xin)
>  xin = c(fbest,FOUT$par[2])
>  T2 = fr(xin)
> T1 - T2
[1] -2.306972e-06
```

Next use the optimum solution to estimate the deformation and plot together:

```
>        Btest = mogiM(R=EX,F=FOUT$par[1] ,A=FOUT$par[2])
```

Finally, print out the result and plot the optimum model:

```
>         plot(PXY, pch=6, col='purple', xlim=c(0,9), ylim=c(0, 1) , xlab="Distan
> lines(EX/1000, Atest$uz/max(Atest$uz), col="green" , lty=2)
>       lines(EX/1000, Btest$uz/max(Btest$uz))
```

The plots of these two models look nearly identical.

7 Okada Modeling

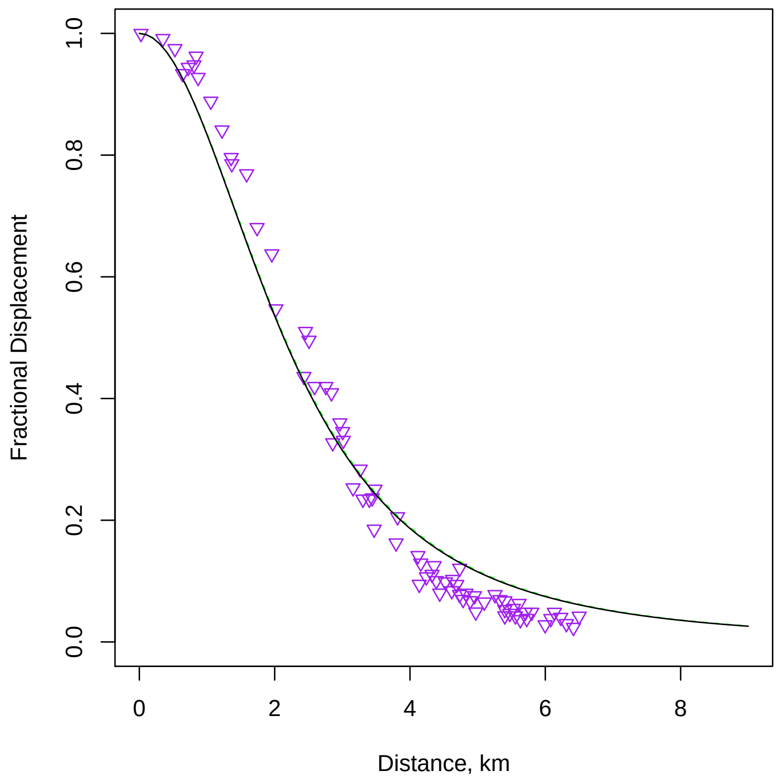

Figure 6: Optimal Mogi Model