RSEISquake: Earthquakes in R

Jonathan M. Lees
University of North Carolina, Chapel Hill
Department of Geological Sciences
CB #3315, Mitchell Hall
Chapel Hill, NC 27599-3315
email: jonathan.lees@unc.edu
ph: (919) 962-1562

July 23, 2012

1 Introduction

Earthquake location is a nonlinear process: Usually it is accomplished by performing a sequence of linear inversions to converge to a location that minimizes the residual misfit. Event determination requires several pieces of information to successfully locate an earthquake: observed arrival time estimates, station locations, a velocity model and, in some cases, a set of station corrections.

Station locations should be provided in terms of Name, Latitude, Longitude and Depth. In some cases several stations may be located at the same location but at different depths (elevation) so these are usually distinguished by some naming convention. Location programs may involve projection of the datainto cartesion coordinates or, in some programs, the processing is done with lat-lon pairs instead. Usually the station locations are stored in a file on disk, as a table or in some other form that can be accessed and stored in memory.

The velocity model is commonly a one-dimensional, layered model, although recently three-dimnesional models are used. Since the location procedure must estimate the travel time from the event to the stations ray-tracing or some other method must be used to make the travel time predictions. In the case of one-dimensional velocity model estimating travel times is relatively simple and the RSEIS package provides routines to accomplish this. The velocity model is typically a table with depth and associated P-wave velocity for a given layer. Often S-wave velocities are provided, although if they are not available a standard approximation is $V_s = V_p/\sqrt{3}$ which is known as a *Poisson* solid. Package Rquake has several built in velocity models that can be used initially if a derived one is not available.

Phase arrival times are typically determined from displays of seismic data. Essentially all that is needed is the arrival time, in seconds relative to some origin of each station for each phase. Software in RSEIS can be used to extract arrival times. Additional information may include the polarity of the P-wave arrival used for focal mechanism determination.

2 Rquake

In this document I will illustrate how to Rquake, a non-linear earthquake location program.

3 Data Structures and Lists

3.1 Station File

Station location information can be stored in memory (in a list) or in a text file on disk. The station file is a table, with name, lat, lon, and elevation.

For example:

```
fsta = "/Users/lees/Mss/SEIS_BOOK/RQUAKE/data/staLLZ.txt"
###  system(paste(sep=" ", "cat", fsta), intern = TRUE )
```

```
CHAC0   -0.39377412    -78.15369741 3588
CHAC1   -0.366526404   -78.16962049  3606
CHAC2   -0.42485567    -78.2710065   4020
CHAC3   -0.4524493    -78.18676153    4328
CHAC4   -0.461317213   -78.21783387   4412
CHAC5   -0.351938598   -78.21809574   4000
CHAC6   -0.408928292   -78.20667762    3860
CHAC7   -0.39837847    -78.22075601    4109
CHAC8   -0.382639731  -78.2023599    3767
CHAC9   -0.323852103  -78.15061344  3762
```

These can be scanned in **R** with a simple command.

See **REIS** for more details on stations.

If the stations are in UTM coordinates, you may convert to Lat-Lon using the GEOmap package.

```
fsta = "/Users/lees/Mss/SEIS_BOOK/RQUAKE/data/staLLZ.txt"
###  system(paste(sep=" ", "cat", fsta), intern = TRUE )
```

```
stas = scan(file=fsta,what=list(name="", lat=0, lon=0, z=0))
stas$z = stas$z/1000
```

Units in Rquake are in km, so the meters are converted.

REIS has a function for reading in the stations:

```
stas = setstas("stas")
```

3.2 Velocity Structure

The one-dimensional velocity model is also stored in file (or stored in memory in an **R** session). See **REIS** for details.

Sample velocity model stored on disk. In this case no estimates of error are provided, so they are set to zero. If S-wave velocity is not available, can use $V_s = V_p/\sqrt{3}$.

```
#MODEL WU COSO REGINAL FINE LAYERS REGIONAL VELOCITY MODEL
#P DEPTH   P VEL      PERR      S DEPTH    S VEL    SERR
   0.00     4.50      0.00      0.00        2.43    0.00
   0.50     4.51      0.00      0.50        2.59    0.00
   1.00     4.92      0.00      1.00        2.97    0.00
   1.50     4.92      0.00      1.50        2.97    0.00
   2.00     5.46      0.00      2.00        3.15    0.00
   2.50     5.46      0.00      2.50        3.15    0.00
   3.00     5.54      0.00      3.00        3.27    0.00
   3.50     5.54      0.00      3.50        3.27    0.00
   4.00     5.58      0.00      4.00        3.42    0.00
   5.50     5.58      0.00      5.50        3.42    0.00
  12.00     6.05      0.00     12.00        3.49    0.00
  20.00     7.20      0.00     20.00        4.15    0.00
```

The following is a constructor for making a 1D velocity model suitable for use in RSEIS and Rquake:

```
 VEL=list()
     VEL$'zp'=c(0,0.25,0.5,0.75,1,2,4,5,10,12)
     VEL$'vp'=c(1.1,2.15,3.2,4.25,5.3,6.25,6.7,6.9,7,7.2)
     VEL$'ep'=c(0,0,0,0,0,0,0,0,0,0)
     VEL$'zs'=c(0,0.25,0.5,0.75,1,2,4,5,10,12)
     VEL$'vs'=c(0.62,1.21,1.8,2.39,2.98,3.51,3.76,3.88,3.93,4.04)
```

```
VEL$'es'=c(0,0,0,0,0,0,0,0,0,0)
VEL$'name'='/data/wadati/lees/Site/Hengil/krafla.vel'
```

There are several default velocity models available in **REIS** . Function defaultVEL(i) will return one of 6 "standard" models used for different purposes.

If you have a velocity model on disk, you can read it in with **REIS** function, Get1Dvel.

To compare a set of different velocity models visually, try,

```
data(ASW.vel)
     data(wu_coso.vel)
     data(fuj1.vel)
     data(LITHOS.vel)
     Comp1Dvels(c("ASW.vel","wu_coso.vel",  "fuj1.vel" , "LITHOS.vel"  ))



dev.off()
```

3.3 Arrival Time List

At the most basic level, all that is required to estimate a hypocenter is relative time of arrival at each station. The arrival times, or the picks are stored in in list mode, i.e. a list of vectors each with attributes relating to the arrival time pick.

These vectors are described as:

tag character tag the should be unique

name character, station name

comp character, component name

c3 character, three-component station id sta.hhh.BHZ

phase character, phase name

err numeric, error

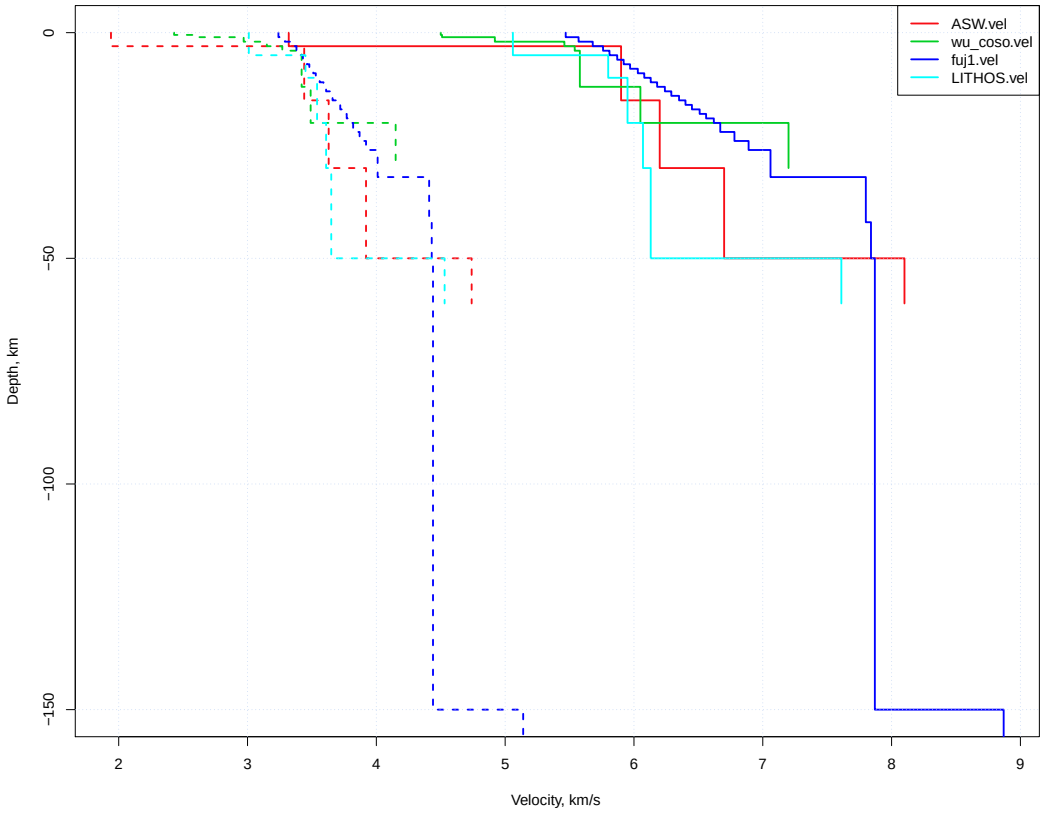

Figure 1: swig example with Reventador Data

pol character polarity, U, D, 0

flg numeric, flag, used in location

res numeric, travel time residual relative to model

dur numeric, duration

yr numeric, year

mo numeric, month

dom numeric, day-of-month

jd numeric, julian day

hr numeric, hour

mi numeric, minute

sec numeric, second

col numeric, or character, color for plotting in RSEIS

onoff numeric, less than 0 means do not use

A constructor for creating an empty pick list is cleanWPX. For many of the functions in RSEIS and Rquake the list must contain filled vectors for each element. use function repairWPX to fill out list elements that are deficient.

The arrival time list has one attribute, the "ID". This can be used to identify earthquake with a unique tag or identifaication number or name.

For **Rquake** , the elements that are absolutely *required* are: name, phase, err, sec. One can construct the input list from these elements by putting in arbitrary information for the other pasts of the list. Once the event and relative time shift is estiamted, these can be added or subtracted from guess origin time.

There are many different ways to store arrival time picks. It does not matter how these are stored, as long as they are read into R and formatted properly. By disassociating the input format from the analysis, we can simply write a short input, or conversion, routine to use all the codes as is.

We can thus store the data in any format we desire, perhaps for use in other non-R software.

3.3.1 Native (binary) R

The output of **swig** is binary R file, so the data can simply be loaded automatically.

3.3.2 UW format Pickfiles

loadUWpickfiles is a function that reads in a list of pickfiles stored on disk and returns a list of picked events.

Since UW pickfiles store the times relative to a common minute mark, and station information is not stored in the pickfile, this information is filled out in the code:

```
KF = vector(mode="list")
   for(i in 1:length(LF))
     {
       g1 = getpfile(LF[i])
       m1 = match(g1$STAS$name, stas$name)
       g1$STAS$lat = stas$lat[m1]
       g1$STAS$lon = stas$lon[m1]
       g1$STAS$z = stas$z[m1]
       w1 = which(!is.na(g1$STAS$lat))
       sec = g1$STAS$sec[w1]
       N = length(sec)

       Ldat = list(name = g1$STAS$name[w1],
         sec = g1$STAS$sec[w1],
         phase = g1$STAS$phase[w1],
         lat = g1$STAS$lat[w1],
         lon = g1$STAS$lon[w1],
         z = g1$STAS$z[w1],
         err = g1$STAS$err[w1],
         yr = rep(g1$LOC$yr, times = N),
         jd = rep(g1$LOC$jd, times = N),
         mo = rep(g1$LOC$mo, times = N),
         dom = rep(g1$LOC$dom, times = N),
         hr = rep(g1$LOC$hr, times = N),
         mi = rep(g1$LOC$mi, times = N))

       Ldat$err[Ldat$err <= 0] = 0.05
       Ksta = length(unique(Ldat$name))
      ###  cat(paste("#################      ", i, Ksta), sep = "\n")
```

```
    Ldat = LeftjustTime(Ldat)

    KF[[i]] = Ldat

  }
```

3.3.3 CSV Pickfiles

An example comma-separated-value file (csv), might look like this:

```
> cat 2011_11_21_12_14_20_683433.csv
"","tag","name","comp","c3","phase","err","pol","flg","res","dur","yr","mo","dom","jd","hr","mi","sec","col","onoff"
"1","CHAC1","CHAC1","V",0,"G",0,"_",0,0,0,2011,11,21,325,12,14,20.6834335327148,"#0000FF",1
"2","CHAC2","CHAC2","V",0,"G",0,"_",0,0,0,2011,11,21,325,12,14,50.691351890564,"#0000FF",1
"3","CHAC6","CHAC6","V",0,"G",0,"_",0,0,0,2011,11,21,325,12,15,14.6926865577698,"#0000FF",1
"4","CHAC8","CHAC8","V",0,"G",0,"_",0,0,0,2011,11,21,325,12,15,44.7056050300598,"#0000FF",1
```

Note the quotes surrounding the text strings. These are not strictly required, but if the funciton read.csv (next secion) is used they are required. The exact nature of the input format is not critical to RSEIS. Rather, leave the data in the format you prefer and write a short script to read it in and put the information in the proper list structure.

4 Example

Suppose you have a set of arrival at the stations of a network.

This solution is based on three non-linear inversions. Each non-linear inversion uses the SVD solution to iterate to a point where the residuals are minimized.

In each case the solution from the previous inversion used as the starting point for the next step.

1. inversion locates only the X-Y-T coordinates.
2. allows the depth to vary.
3. attempts a detailed inversion.

Next, extract information to make a plot.

```
 MLAT = median(Ldat$lat)
    MLON = median(Ldat$lon)
 EQ =AQ$EQ
 proj = setPROJ(type=2, LAT0=MLAT, LON0=MLON)
####   get station X-Y values in km
    XY = GLOB.XY(stas$lat, stas$lon, proj)
   eXY = GLOB.XY(EQ$lat, EQ$lon, proj)




plotEQ(Ldat, AQ, add=FALSE, prep=FALSE, proj=proj, xlim=NULL, ylim=NULL )
J = EmptyPickfile()
J$STAS = Ldat
col=tomo.colors(100)
contPFarrivals(J, stas, proj=proj,cont=TRUE, POINTS=FALSE,
                           image=FALSE , col=col,     phase="P", add=TRUE)



dev.off()
```

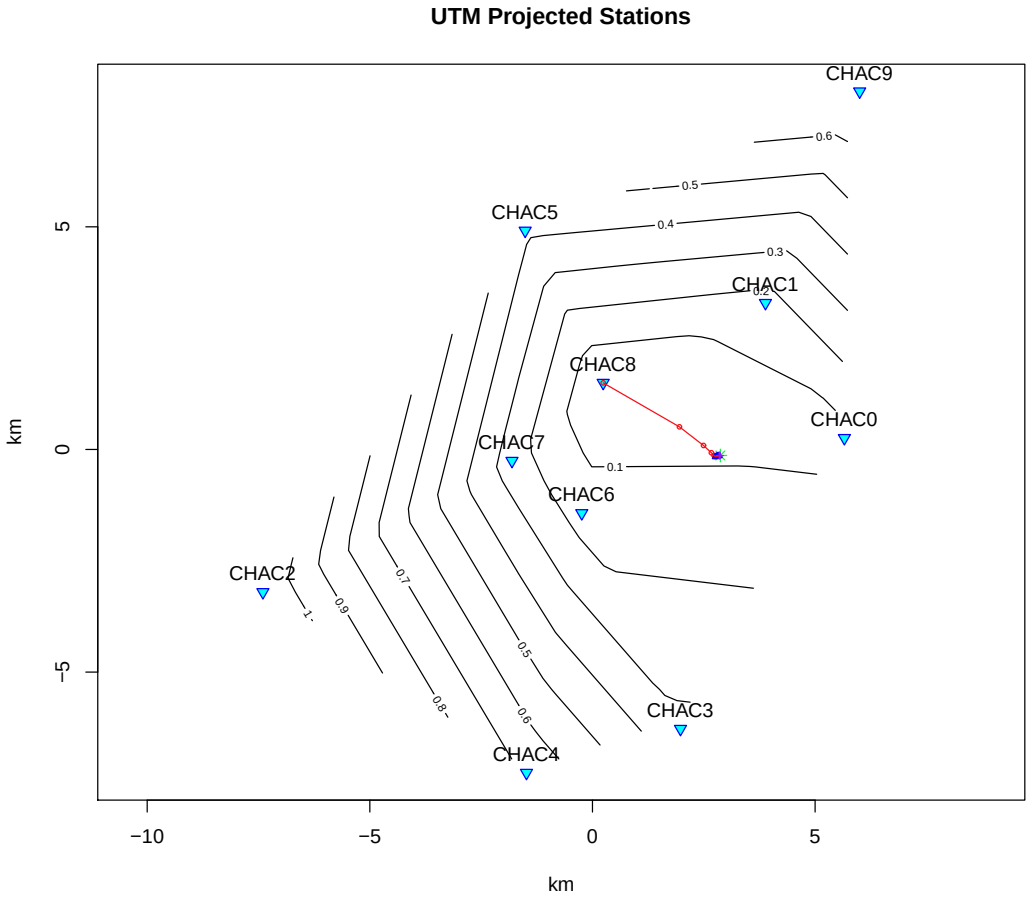

Figure 2: Earthquake location example

4.1 Jackknifing the Event Location

In many cases a station may have an undue influence on the earthquake location. In this case we are interested in estimating what the influence is and how it might affect the hypocenter determination.

A technique called the "Jackknife" is commonly used to provide such an estimate. The Jackknife was created to provide estimates of parameters that do not have a simple aalytical solution. This may be especially important for estimating the variance of a parameter. This technique was named Jacknife because of its power and usefulness in exploratory data analysis.

In the estimated model, errors arise from inaccurate measurements, incomplete coverage of rays over the target area, over simplified parameterization and mislocation of the earthquakes. In classical least squares discussed above, one can show that an estimate for the error in the model is found in the covariance matrix, $cov(\boldsymbol{s}) = \boldsymbol{s}\boldsymbol{s}^T$. If the estimated variances of the weighted observations is a constant value σ^2, then it can be shown formally that $cov(\boldsymbol{s}) = \sigma^2 \left(\boldsymbol{A}^T\boldsymbol{A}\right)^{-1}$ [Menke, 1984]. As before, the matrix $\left(\boldsymbol{A}^T\boldsymbol{A}\right)^{-1}$ is generally not available due to limitations in computer storage and space, thus leaving us with no estimate for the uncertainties inherent in the slowness structures we derive.

The approach used in this research to estimate the uncertainties in the inversion images is derived from techniques developed for statistical applications and is commonly known as the jackknife, named so for its rough and ready usefulness [Schucany et al., 1971]. The method is described thus: suppose we are given n identically independent random variables $\{X_1, X_2, \cdots, X_n\}$ and we wish to estimate the value of a parameter, $\hat{\theta} = \hat{\theta}\,(X_1, X_2, \cdots, X_n)$, e.g. a statistic on the data such as the mean, median or correlation coefficient ($\hat{\theta}$ represents an estimate of the true value θ). We form a set of intermediate estimates by considering the values of $\hat{\theta}^{(i)}$ calculated by leaving out the $i-th$ datum, i.e.

$$\hat{\theta}^{(i)} = \theta\,(X_1, X_2, \cdots, X_{i-1}, X_{i+1}, \cdots, X_n) \tag{4.1}$$

A set of 'pseudo-values' is created by considering the weighted sum,

$$\tilde{\theta}^{(i)} = n\hat{\theta} - (n-1)\hat{\theta}^{(i)} \tag{4.2}$$

and the jackknife estimate of the parameter th is the arithmetic mean of the pseudo-values:

$$\theta_{JACK} = \frac{1}{n}\sum_{i=1}^{n} \tilde{\theta}^{(i)} \tag{4.3}$$

The historical motivation for this formulation stems from an attempt to estimate the bias for many common statistics, particularly quadratic functionals [Quenouille, 1949; Efron, 1982]. Only later did researchers [Tukey, 1958] realize the more important use of the jackknife in estimating variability of model parameters. Heuristically, the form of Eqn 4.3 stems from noting that each pseudo-value measures the influence a particular datum has on estimating the model parameter by forming the linear combination of the estimates with and without the particular datum.

In the tomographic setting the jackknife involves partitioning the set of observations into subsets, performing inversions on the subsets, and calculating a standard error from the set of image vectors that result. Instead of leaving out one datum per inversion, we divide the data into k subsets with each group containing $n - \frac{n}{k}$ data values where the $\frac{n}{k}$ values omitted are chosen without replacement from the original data set. Each subset then has a different portion of the full data set missing. We then perform an inversion for each of the k subsets and denote the slowness image derived from such an inversion $\hat{s}_j$. From these 'mini-inversions' a 'pseudo-inversion' is formed following equation Eqn 4.3:

$$\tilde{s}_j = k\hat{s}_{all} - (k-1)\hat{s}_j \tag{4.4}$$

The jackknifed estimate of the slowness is simply the average of the pseudo-inversions:

$$\tilde{s} = \frac{1}{k}\left(\sum_{j=1}^{k} \tilde{s}_j\right) \tag{4.5}$$

which has variance,

$$\boldsymbol{v} = \frac{\sum \tilde{s}_j^2 - \frac{1}{k}\left(\sum \tilde{s}_j\right)^2}{k(k-1)} \tag{4.6}$$

Given the variance, the standard error is $\boldsymbol{E}_\sigma = \sqrt{\boldsymbol{v}}$. This will be an estimate of the variability of the model due to the variability and distribution of the data. Presently there is no clear cut way to determine the optimal choice of k, the number of mini-inversions to perform. One would guess the larger k is the better, but very large k implies performing large numbers of inversions, which would be extremely time consuming and provide little advantage over computing errors in the classical fashion. A compromise can be struck if we assume that the variability in the pseudo-inversions will be represented in far fewer partitions of the data. In this study I have found $k = 30$ to be a reasonable for estimation of errors on the real data. In order to avoid the influence of any special ordering, the partitions are chosen randomly for each subset.

To get the jackknife estiamte for the location parameters lat-lon-z we have a routine in the RQUAKE package. A list of earthquake pickfiles are provided to the code and each event is jackknifed individually. the results aresaved and summarized so that station influence can be estiamted.

First the locations are jackknifed,

Then they are plotted with two plots, one showing the boxplots of the results fro each station, the second showing a controur of the median values for each station distrbuted in map view.

```
plotJACKLLZ(COSOjack, sta, proj, PLOT=1, PS=TRUE)
```

```
plotJACKLLZ(COSOjack, sta, proj, PLOT=2, PS=TRUE)
```

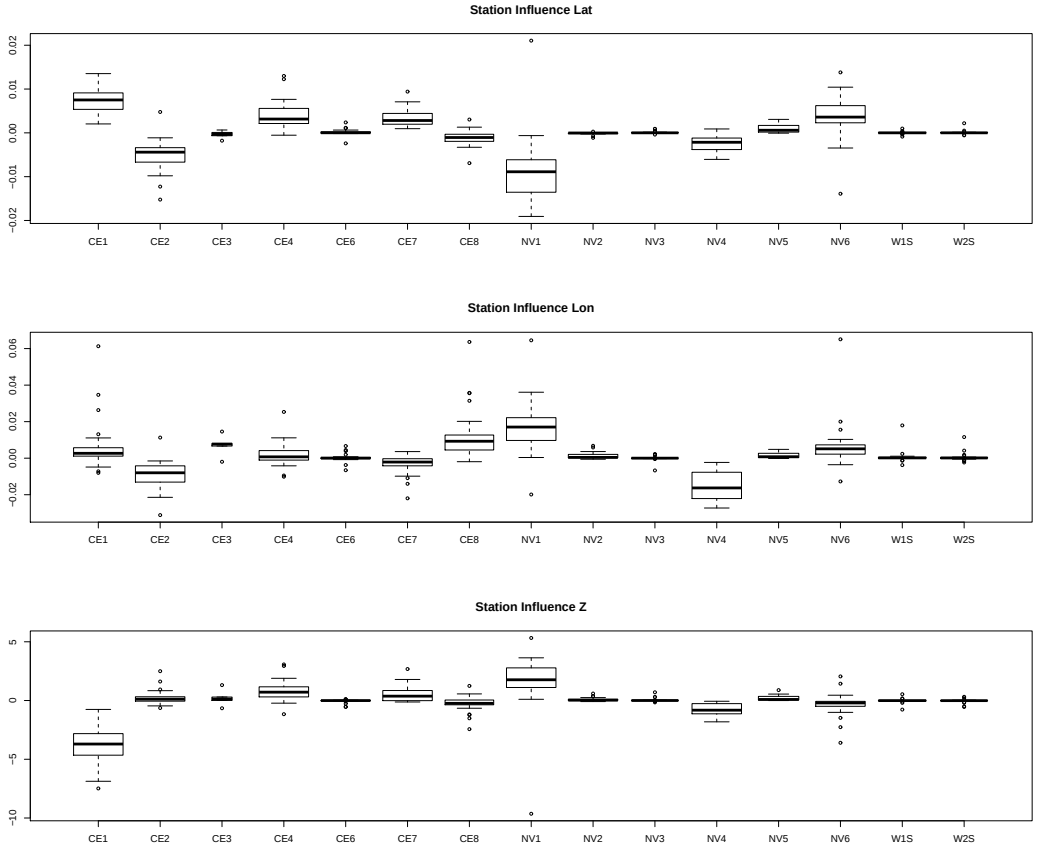

Figure 3: Jackknife estimate of earthquake location parameters, LAT-LON-Z

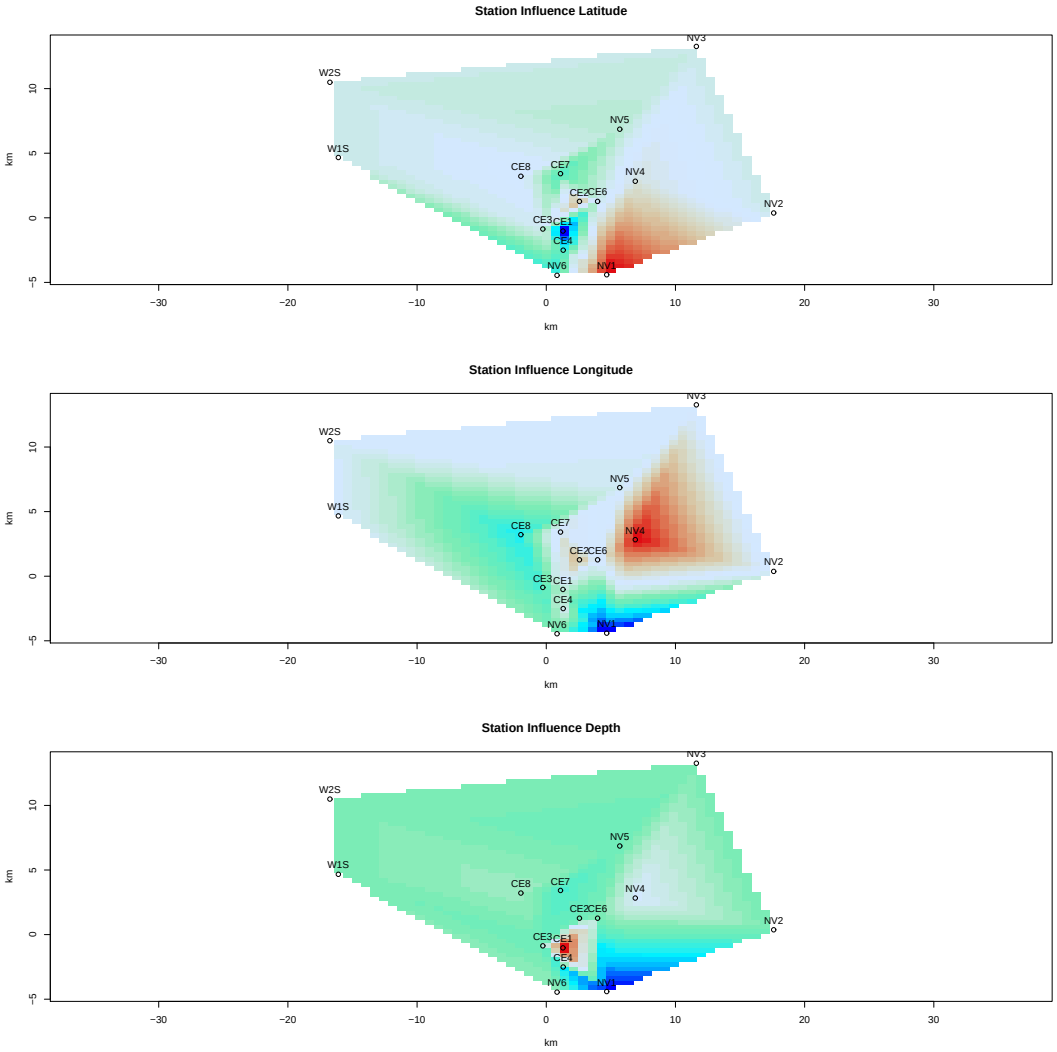

Figure 4: Jackknife estimate of earthquake location parameters, LAT-LON-Z

References Cited