Expectation-Maximization

João Neto

October 2014

- The (Meta-)Algorithm
- EM for clustering
  - Eg: EM with mix of two linear models
  - Using R's mclust for classification
- Using R's Alice for missing value imputation
  - Diagnostics
  - Analysis Models

Refs:

- Data Mining Algorithms In R/Clustering/Expectation Maximization (EM) (http://en.wikibooks.org/wiki/Data_Mining_Algorithms_In_R/Clustering/Expectation_Maximization_(EM))
- Bishop, **Pattern Matching and ML**, chapter 9.

The EM algorithm is a methodology for algorithm construction, it is not a specific algorithm. Each problem is different, only the structure of the Expectation and Maximization steps are common. How exactly they are programmed is problem dependent.

The (Meta-)Algorithm

We observed data X and have a (possibly made up) set of latent variables Z. The set of model parameters is θ.

The goal of the EM algorithm is to find a maximum to the likelihood function $p(X|\theta)$ wrt parameter θ, when this expression or its log cannot be discovered by typical MLE methods.

Suppose for each observation $x^i \in X$ we get the corresponding value $z^i \in Z$. $\{X, Z\}$ is called the complete dataset. The likelihood for the complete set is $p(X, Z|\theta)$. However we do not know the complete set, only X. To proceed we need to construct the posterior $p(Z|X, \theta)$. Usually this is the part that requires some imagination/luck/experience.

With $p(Z|X, \theta)$ we can compute the likelihood for the complete set:

$$p(X, Z|\theta) = p(Z|X, \theta)p(X|\theta)$$

Assume also that we know an estimate θ_i for θ. This allows us to compute the posterior $p(Z|X, \theta_i)$. Initially $\theta_i = \theta_0$ is randomly assigned.

The log-likelihood is

$$\log p(X|\theta) = \log\{\sum_z p(X, Z = z|\theta)\} = \log\{\sum_z p(X|Z = z, \theta)p(Z = z|\theta)\}$$

as given by the marginal rule of probability. For continuous latent variables, the sum is replaced by an integral. The EM algorithm will maximize $\log p(X|\theta)$, but since $\log$ is a strict monotonous function, it will also maximize $p(X|\theta)$.

With these equations let's see what EM should do.

Given a current estimate θ_i we wish to find an update θ such that $p(X|\theta) > p(X|\theta_i)$ (remember, if that cannot be done, the algorithm found a maximum and we are over). If possible we want to maximize the difference between $p(X|\theta)$ and $p(X|\theta_i)$:

$$
\begin{aligned}
& \log p(X|\theta) - \log p(X|\theta_i) \\
= \; & \log\left\{\textstyle\sum_z p(X|Z=z,\theta)p(Z=z|\theta)\right\} - \log p(X|\theta_i) && \text{marginal rule} \\
= \; & \log\left\{\textstyle\sum_z p(X|Z=z,\theta)p(Z=z|\theta)\frac{p(Z=z|X,\theta_i)}{p(Z=z|X,\theta_i)}\right\} - \log p(X|\theta_i) \\
= \; & \log\left\{\textstyle\sum_z p(Z=z|X,\theta_i)\frac{p(X|Z=z,\theta)p(Z=z|\theta)}{p(Z=z|X,\theta_i)}\right\} - \log p(X|\theta_i) \\
\geq \; & \textstyle\sum_z p(Z=z|X,\theta_i)\log\left\{\frac{p(X|Z=z,\theta)p(Z=z|\theta)}{p(Z=z|X,\theta_i)}\right\} - \log p(X|\theta_i) && \log\textstyle\sum_i a_i x_i \geq \sum_i a_i \log x_i, \text{Jensen Ineq.} \\
= \; & \textstyle\sum_z p(Z=z|X,\theta_i)\log\left\{\frac{p(X|Z=z,\theta)p(Z=z|\theta)}{p(Z=z|X,\theta_i)p(X|\theta_i)}\right\} && \text{multiply } \textstyle\sum_z p(Z=z|X,\theta_i) = 1 \text{ by } p(X|\theta_i) \\
\equiv \; & \Delta(\theta|X,\theta_i) && \equiv : \text{by definition}
\end{aligned}
$$

So,

$$\log p(X|\theta) \geq \log p(X|\theta_i) + \Delta(\theta|X,\theta_i) \equiv Q(\theta|\theta_i)$$

This new function $Q(\theta|\theta_i)$ is bounded above by the likelihood $\log p(X|\theta)$. They are equal at $\theta = \theta_i$.

The goal of the EM algoritm is to choose θ_{i+1} as the value of θ such that $Q(\theta|\theta_i)$ is a maximum. Since $\log p(X|\theta_{i+1}) \geq Q(\theta_{i+1}|\theta_i)$ we ensure that the likelihood also rises as much as possible.

$$
\begin{aligned}
\theta_{i+1} &= \arg\max_\theta Q(\theta|\theta_i) \\
&= \arg\max_\theta \left\{\log p(X|\theta_i) + \textstyle\sum_z p(Z=z|X,\theta_i)\log\frac{p(X|Z=z,\theta)p(Z=z|\theta)}{p(Z=z|X,\theta_i)p(X|\theta_i)}\right\} \\
&= \arg\max_\theta \left\{\textstyle\sum_z p(Z=z|X,\theta_i)\log p(X|Z=z,\theta)p(Z=z|\theta)\right\} && \text{remove constants wrt } \theta \\
&= \arg\max_\theta \left\{\textstyle\sum_z p(Z=z|X,\theta_i)\log\frac{p(X,Z=z,\theta)p(Z=z,\theta)}{p(Z=z,\theta)p(\theta)}\right\} && p(X|Y) = \frac{p(X,Y)}{p(Y)} \\
&= \arg\max_\theta \left\{\textstyle\sum_z p(Z=z|X,\theta_i)\log p(X,Z=z|\theta)\right\} \\
&= \arg\max_\theta \left\{E_{Z|X,\theta_i}[\log p(X,Z|\theta)]\right\} && E_Z[g(Z)] = \textstyle\sum_z f(Z)g(Z)
\end{aligned}
$$

So, the EM algorithm is:

0. Have X and $p(Z|X,\theta)$.
1. $\theta_i \leftarrow \theta_0$, where θ_0 is a random assignment of values.
2. **E-step**: Compute $Q(\theta|X,\theta_i) = E_{Z|X,\theta_i}[\log p(X,Z|\theta)]$
3. **M-step**: Compute $\theta_{i+1} \leftarrow \text{argmax}_\theta\, Q(\theta|X,\theta_i)$
4. If θ_i and θ_{i+1} are not close enough, $\theta_i \leftarrow \theta_{i+1}$ and goto 2.

There is a convergence proof (http://seanborman.com/publications/EM_algorithm.ps.gz) that states $\Delta(\theta_{i+1}|X,\theta_i) \geq 0$, which means the EM algorithm stops eventually. However, there is no garantee that a global maximum is achieved. That's why it is sometimes needed to run the algorithm with different values for θ_0. The assigment of θ_0 is th eonly random element in the entire EM algorithm, which is why it is called an deterministic optimization technique.

An eg using EM with mixture of Gaussians can be found here (http://www.di.fc.ul.pt/~jpn/r/EM/GaussianMix.html).

EM for clustering

The EM algorithm can be seen an unsupervised clustering method based on mixture models (http://en.wikipedia.org/wiki/Mixture_model). It follows an iterative approach, sub-optimal, which tries to find the parameters of the probability distribution that has the maximum likelihood (http://en.wikipedia.org/wiki/Maximum_likelihood) of its attributes in the presence of missing/latent data.

The algorithm's input are the data set X, the total number of clusters/models K, the accepted error to converge ϵ and the maximum number of iterations.

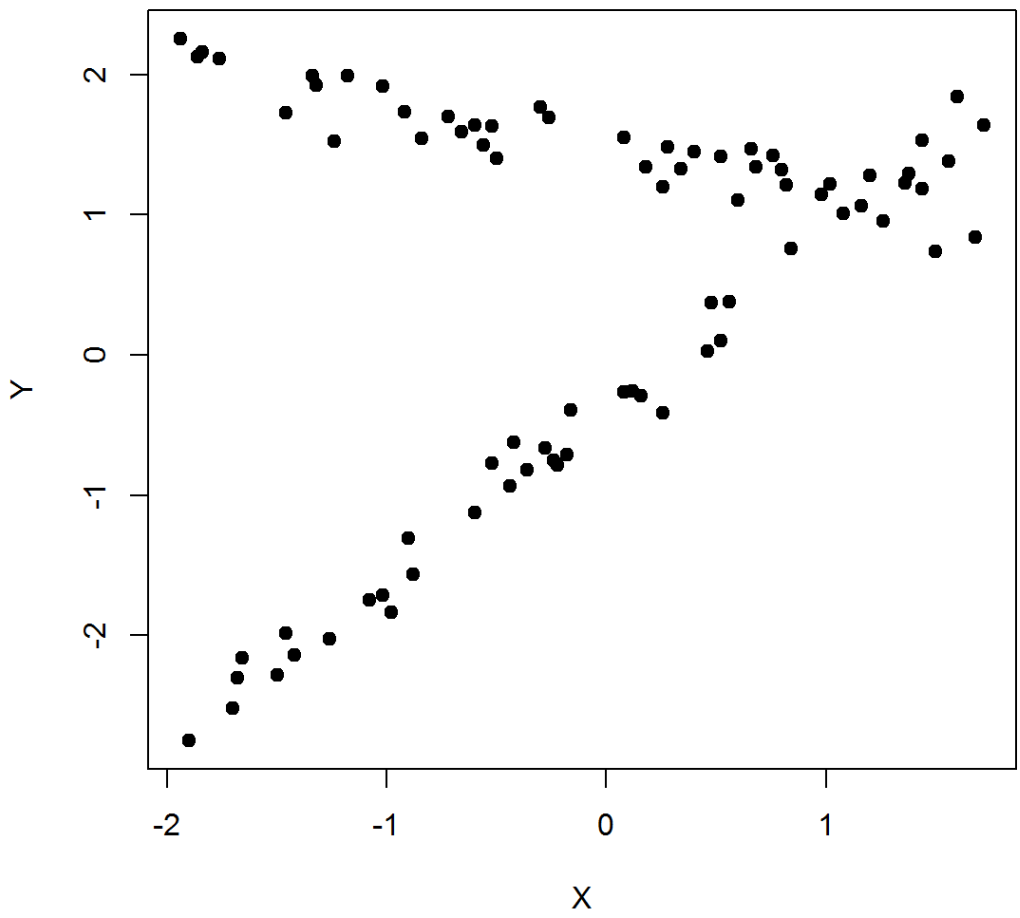
For each iteration, first it is executed what's called the Expectation Step (E-step), that estimates the probability of each point belonging to each model, followed by the Maximization step (M-step), that re-estimates the parameter vector of the probability distribution of each model. The algorithm finishes when the distribution parameters converges or reach the maximum number of iterations. Convergence is assured since the algorithm increases the likelihood at each iteration until it reaches the (eventually local) maximum.

Eg: EM with mix of two linear models

- Ref: (http://www.cs.huji.ac.il/~yweiss/emTutorial.pdf)http://www.cs.huji.ac.il/~yweiss/emTutorial.pdf (http://www.cs.huji.ac.il/~yweiss/emTutorial.pdf)

Let's have a dataset generated by either one of two linear processes C_1 and C_2:

```
set.seed(101)

slope1 <- -.3; intercept1 <- 1.5    # generating data from C_1
xs1 <- sample(seq(-2,2,len=201), 40)
ys1 <- intercept1 + slope1*xs1 + rnorm(length(xs1),0,.15) # add some noise

slope2 <- 1.2; intercept2 <- -.4    # generating data from C_2
xs2 <- sample(seq(-2,2,len=201), 40)
ys2 <- intercept2 + slope2*xs2 + rnorm(length(xs1),0,.15)

mydata <- rbind( cbind(xs1,ys1), cbind(xs2,ys2) )
plot(mydata, pch=19, xlab="X", ylab="Y")
```



We need to achieve two things:

- The parameters, slope and intersect, of the two models
- The assigment of each datapoint x^i to the modelled process that generated it

If we know one the other should be easy to find. If we know the parameters, we could apply a maximum likelihood estimation to find which model is more probable to have generated x^i, i.e., compute $p(x^i|C_j)$. If we know the classifications we could just perform two linear regressions in order to find the parameters.

This is the basic structure of the EM algorithm:

1. Init-step: Assign random values to the model's parameters
2. E-step: assign points to the model that fits each one best (these assignments are continuous, not binary)
3. M-step: update the parameters using the points assigned in the previous step
4. Iterate until parameter values converge

Init Step

```
i1 <- s1 <- i2 <- s2 <- 0 # model parameters for slope and intersect
init_params <- function() {
  i1 <<- 2*runif(1)
  s1 <<- 2*runif(1)
  i2 <<- 2*runif(1)
  s2 <<- 2*runif(1)
  c(i1,s1,i2,s2)
}

params <- init_params()
```

E-step:

For each point x^i compute two weights (probabilities) w_1^i and w_2^i for the soft assigments of model 1 and 2.

We calculate the residuals r^i of the two models for each x^i:

$$r_j^i = |\text{intersect}_j + \text{slope}_j x^i - y^i|$$

So,

$$w_j^i = \frac{p(r_j^i|x^i \in C_j)}{p(r_1^i|x^i \in C_1) + p(r_2^i|x^i \in C_2)}$$

If we assign normal distributions to these probabilities

$$r_j^i|x^i \in C_j \sim \mathcal{N}(0, \sigma^2)$$

which means

$$p(r_j^i|x^i \in C_j) = \frac{1}{\sqrt{2\pi}\sigma}\exp\{-\frac{(r_j^i)^2}{2\sigma^2}\}$$

plugging the distributions into the w_j^i formula:

$$w_j^i = \frac{\exp\{-\frac{(r_j^i)^2}{2\sigma^2}\}}{\exp\{-\frac{(r_1^i)^2}{2\sigma^2}\} + \exp\{-\frac{(r_2^i)^2}{2\sigma^2}\}}$$

The σ is a free parameter. This corresponds to the expected noise level in the sample. Herein we'll assume $\sigma = 0.5$. Values too small might produce underflows in R, especially in the first iterations of the EM algorithm.

```
# params is [s1,i1,s2,i2]
e.step <- function(mydata, params, sigma=0.5) {
  w1 <- rep(NA, nrow(mydata))
  w2 <- rep(NA, nrow(mydata))

  for (i in 1:nrow(mydata)) {
    r1 <- abs(params[1] + params[2] * mydata[i,1] - mydata[i,2]) # residual for model 1
    r2 <- abs(params[3] + params[4] * mydata[i,1] - mydata[i,2]) # residual for model 2

    exp1 <- exp(-r1^2/sigma^2)
    exp2 <- exp(-r2^2/sigma^2)

    w1[i] <- exp1 / (exp1+exp2)
    w2[i] <- exp2 / (exp1+exp2)
  }

  cbind(w1,w2)
}

ws <- e.step(mydata, params)
head(ws)
```

```
##                w1        w2
## [1,] 1.958942e-04 0.9998041
## [2,] 1.040201e-30 1.0000000
## [3,] 2.546676e-01 0.7453324
## [4,] 5.015218e-01 0.4984782
## [5,] 4.996198e-12 1.0000000
## [6,] 7.438090e-08 0.9999999
```

M-step:

Here we assume the weights are given for each data point x^i.

To estimate the parameters of each process we use weighted least squares (https://onlinecourses.science.psu.edu/stat501/node/213).

In classic least squares, the parameters p_i for the i^{th} model would be given by:

$$\hat{p_i} = (X_i^T X_i)^{-1} X_i^T Y$$

To add weights w_i we make a diagonal matrix W_i with these values and compute:

$$\hat{p_i} = (X_i^T W_i X_i)^{-1} X_i^T W_i Y$$

```
# wls - weighted least squares
wls <- function(X,Y,W) {
  solve(t(X) %*% W %*% X) %*% t(X) %*% W %*% Y
}

m.step <- function(mydata, ws) {
  X <- cbind(rep(1, nrow(mydata)), mydata[,1])
  Y <- as.matrix(mydata[,2], ncol=1)
  p_1 <- wls(X,Y,diag(ws[,1]))
  p_2 <- wls(X,Y,diag(ws[,2]))

  c(p_1, p_2)
}

params <- m.step(mydata, ws)
```

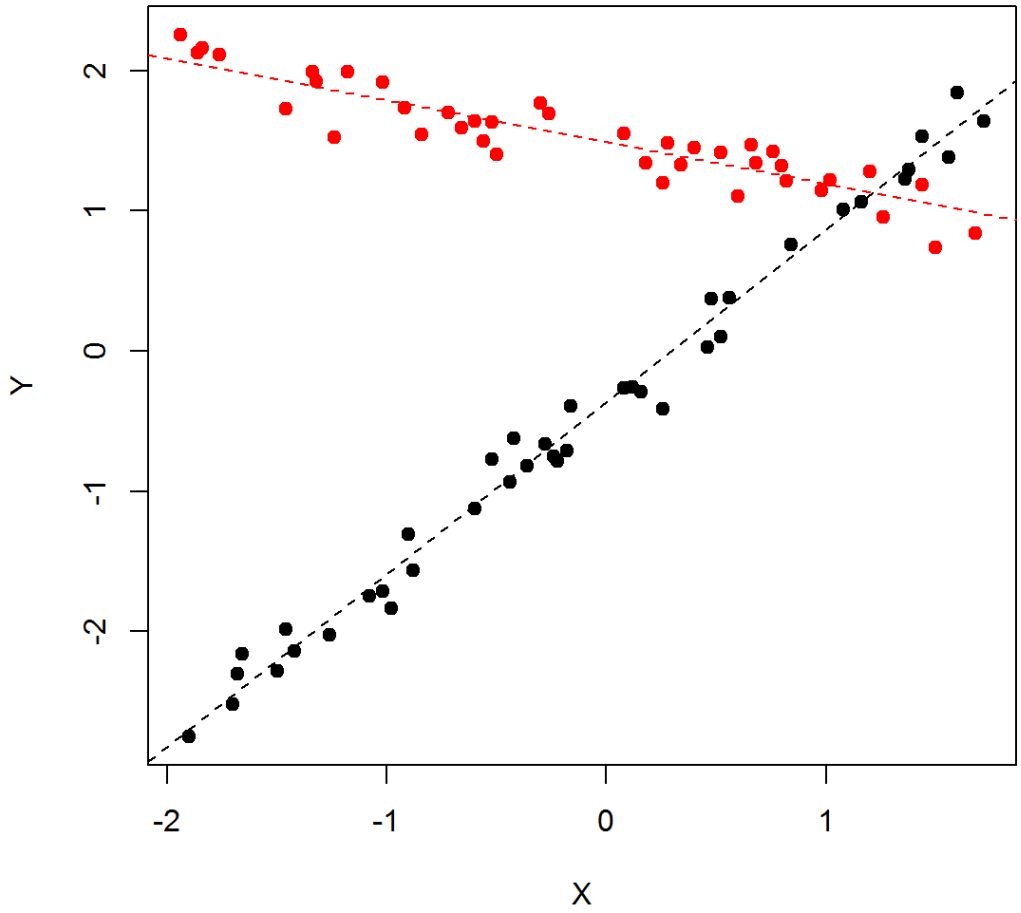
Let's put everything together for this problem:

```
em.2lines <- function(mydata, tol=1e-2, max.step=1e3) {
  step <- 0

  s1 <- i1 <- s2 <- i2 <- 0 # model parameters for slope and intersect
  params <- init_params()

  repeat {
    ws         <- e.step(mydata, params)
    old.params <- params
    params     <- m.step(mydata, ws)

    if (norm(as.matrix(old.params-params), type="F") < tol) # convergence achieved
      break

    step <- step +1
    if (step > max.step)
      break
  }

  list(params=params,    # the estimated parameters
       weights=ws,       # the weighs for each datapoint x^i
       class=apply(ws, 1, function(v) if (v[1]>v[2]) 1 else 2))  # the class for each data
point
}

report <- em.2lines(mydata)
report$params
```

```
## [1] -0.3648036  1.2281834  1.4905805 -0.2981542
```

Let's plot the results:

```
plot(mydata, pch=19, col=report$class, xlab="X", ylab="Y")
abline(a=report$params[1], b=report$params[2], col=1, lty=2) # draw 1st model with found p
arameters
abline(a=report$params[3], b=report$params[4], col=2, lty=2) # draw 2nd model with found p
arameters
```



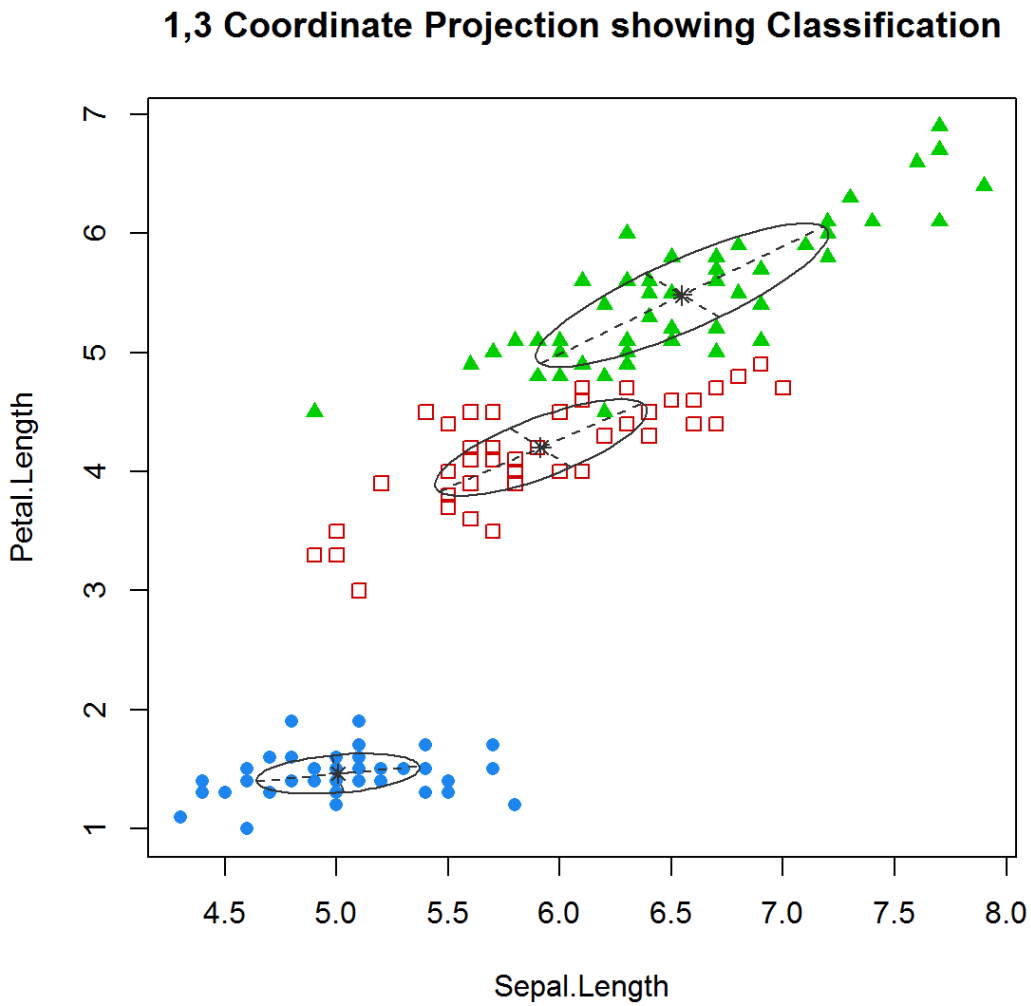
Using R's mclust for classification

Using `Mclust` from package `mclust` to find the optimal model for a given dataset and number of clusters based on EM:

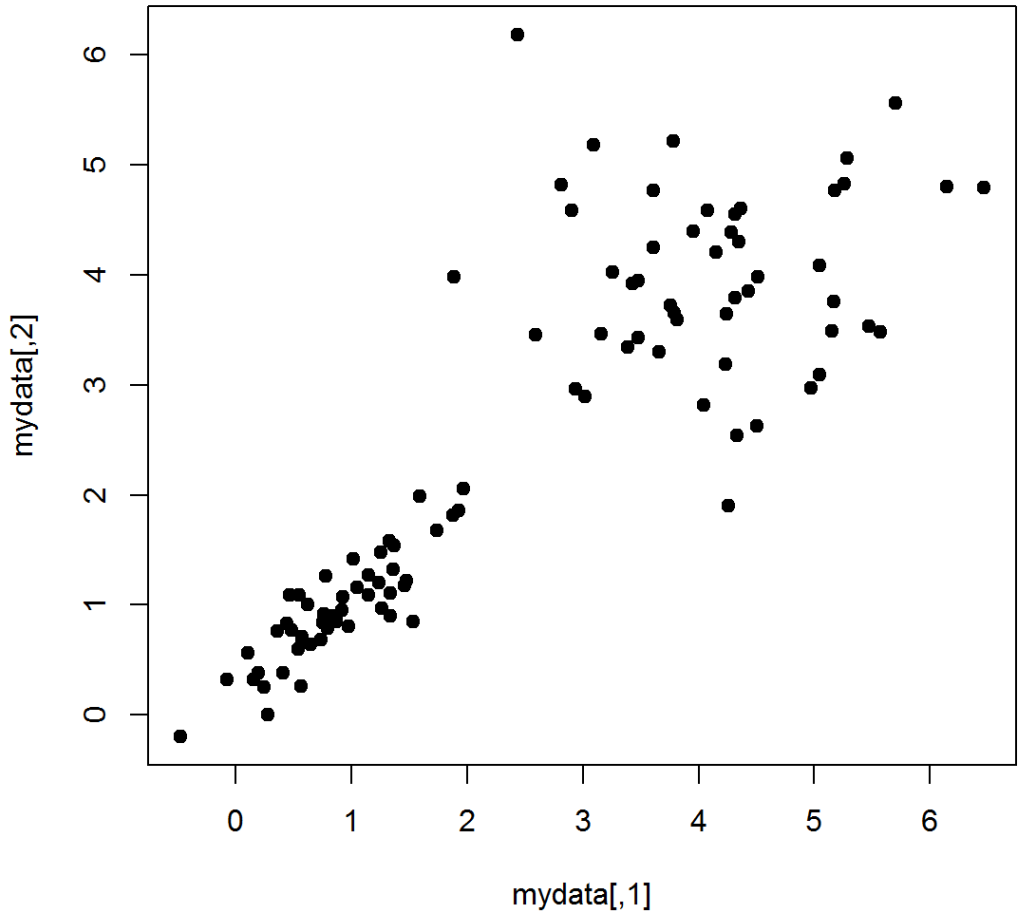
```
library(MASS)
library(mclust)
```

```
## Package 'mclust' version 5.1
## Type 'citation("mclust")' for citing this R package in publications.
```

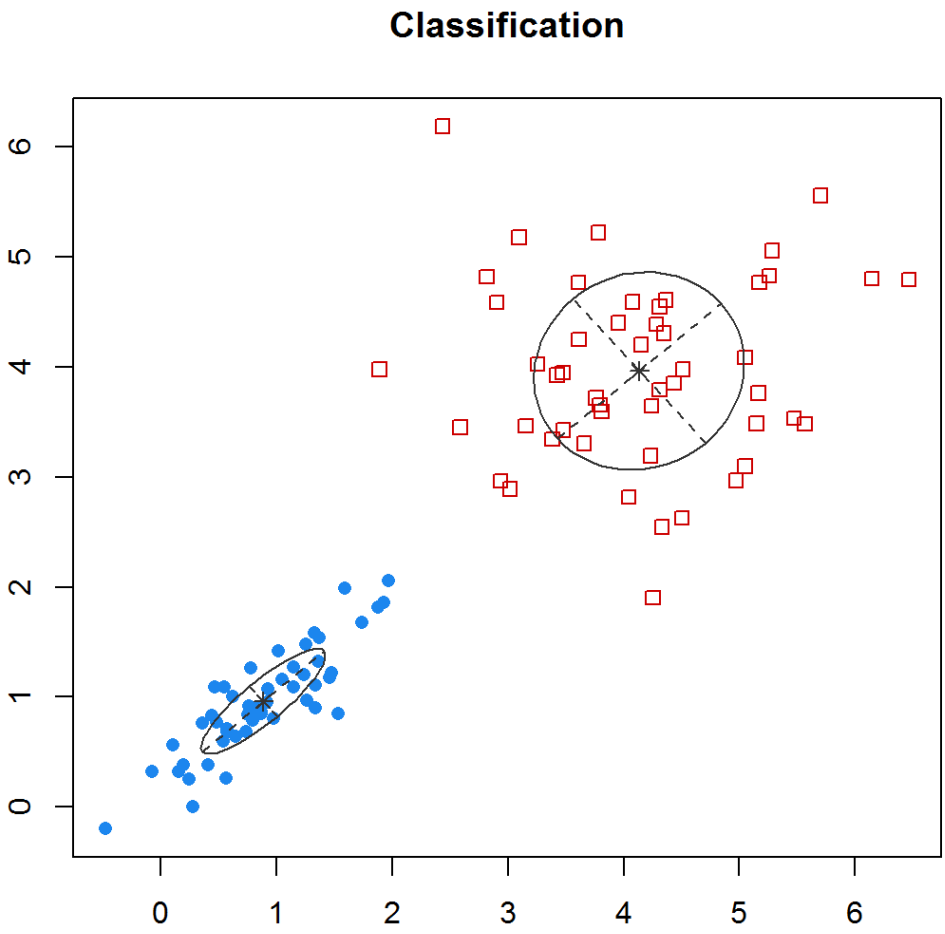
```
mc <- Mclust(iris[,1:4], G=3) # 3 clusters
plot(mc, what=c("classification"), dimens=c(1,3)) # using 1st and 3rd column of the iris d
ataset
```




```
# make dataset with points from two different multivariate normal densities
mydata <- rbind(mvrnorm(50, c(1,1), matrix(c(.5^2,.20,.20,.5^2), ncol=2)),
                mvrnorm(50, c(4,4), matrix(c(1,0,0,1), ncol=2)))
plot(mydata, pch=19)
```




```
mixclust <- Mclust(mydata)  # if G is not given, it will test between 1 to 9
plot(mixclust, what=c("classification"))  # plot the distinct clusters found
```



Using R's Alice for missing value imputation

Amelia assumes that the data follow a multivariate normal distribution, so all information about the relations in the data can be summarized by just means and covariances. When data are incomplete, Amelia uses the well-known EM algorithm to find corrected estimates of the means and covariances.

In their original form the EM estimates cannot be used to create multiple imputations, as the estimates do not reflect the fact that they have been estimated from a finite sample. In order to solve this, Amelia first takes m bootstrap samples, and applies the EM-algorithm to each of these bootstrap samples. The m estimates of means and variances will now be different. The first set of estimates is used to draw the first set of imputed values by a form of regression analysis, the second set is used to calculate the second set of imputed values, and so on.

> As Amelia assumes a multivariate normal distribution, it will work best when your data are approximately normally distributed (possibly after a transformation), and when the statistics you calculate from the data in your complete-data analysis are near the center of the distribution, like means, modes or regression weights. Ref (http://stats.stackexchange.com/questions/47247/multiple-imputation-with-the-amelia-package)

The following code was taken from the package's vignette

```
library(Amelia)
```

```
## Loading required package: Rcpp
## ##
## ## Amelia II: Multiple Imputation
## ## (Version 1.7.3, built: 2014-11-14)
## ## Copyright (C) 2005-2015 James Honaker, Gary King and Matthew Blackwell
## ## Refer to http://gking.harvard.edu/amelia/ for more information
## ##
```

```
data(freetrade)

summary(freetrade)
```

```
##       year        country             tariff           polity
##  Min.   :1981   Length:171         Min.   :  7.10   Min.   :-8.000
##  1st Qu.:1985   Class :character   1st Qu.: 16.30   1st Qu.:-2.000
##  Median :1990   Mode  :character   Median : 25.20   Median : 5.000
##  Mean   :1990                      Mean   : 31.65   Mean   : 2.905
##  3rd Qu.:1995                      3rd Qu.: 40.80   3rd Qu.: 8.000
##  Max.   :1999                      Max.   :100.00   Max.   : 9.000
##                                    NA's   :58       NA's   :2
##       pop                gdp.pc           intresmi          signed
##  Min.   : 14105080   Min.   :  149.5   Min.   :0.9036   Min.   :0.0000
##  1st Qu.: 19676715   1st Qu.:  420.1   1st Qu.:2.2231   1st Qu.:0.0000
##  Median : 52799040   Median :  814.3   Median :3.1815   Median :0.0000
##  Mean   :149904501   Mean   : 1867.3   Mean   :3.3752   Mean   :0.1548
##  3rd Qu.:120888400   3rd Qu.: 2462.9   3rd Qu.:4.4063   3rd Qu.:0.0000
##  Max.   :997515200   Max.   :12086.2   Max.   :7.9346   Max.   :1.0000
##                                        NA's   :13       NA's   :3
##      fiveop          usheg
##  Min.   :12.30   Min.   :0.2558
##  1st Qu.:12.50   1st Qu.:0.2623
##  Median :12.60   Median :0.2756
##  Mean   :12.74   Mean   :0.2764
##  3rd Qu.:13.20   3rd Qu.:0.2887
##  Max.   :13.20   Max.   :0.3083
##  NA's   :18
```

```
head(freetrade,12)
```

```
##    year  country tariff polity      pop   gdp.pc intresmi signed fiveop
## 1  1981 SriLanka     NA      6 14988000 461.0236 1.937347      0   12.4
## 2  1982 SriLanka     NA      5 15189000 473.7634 1.964430      0   12.5
## 3  1983 SriLanka   41.3      5 15417000 489.2266 1.663936      1   12.3
## 4  1984 SriLanka     NA      5 15599000 508.1739 2.797462      0   12.3
## 5  1985 SriLanka   31.0      5 15837000 525.5609 2.259116      0   12.3
## 6  1986 SriLanka     NA      5 16117000 538.9237 1.832549      0   12.5
## 7  1987 SriLanka   27.3      5 16361000 540.0475 1.422983      0   12.5
## 8  1988 SriLanka   27.3      5 16587000 545.8610 1.059624      1   12.6
## 9  1989 SriLanka     NA      5 16806000 551.1353 1.137557      0   12.6
## 10 1990 SriLanka   28.3      5 16993000 579.9548 1.663632      0   12.7
## 11 1991 SriLanka   26.9      5 17247000 597.6987 2.285213      1   12.8
## 12 1992 SriLanka   25.0      5 17405000 618.3329 2.877877     NA   13.1
##        usheg
## 1  0.2593112
## 2  0.2558008
## 3  0.2655022
## 4  0.2988009
## 5  0.2952431
## 6  0.2886563
## 7  0.2734092
## 8  0.2756469
## 9  0.2785387
## 10 0.2608332
## 11 0.2589872
## 12 0.2623017
```

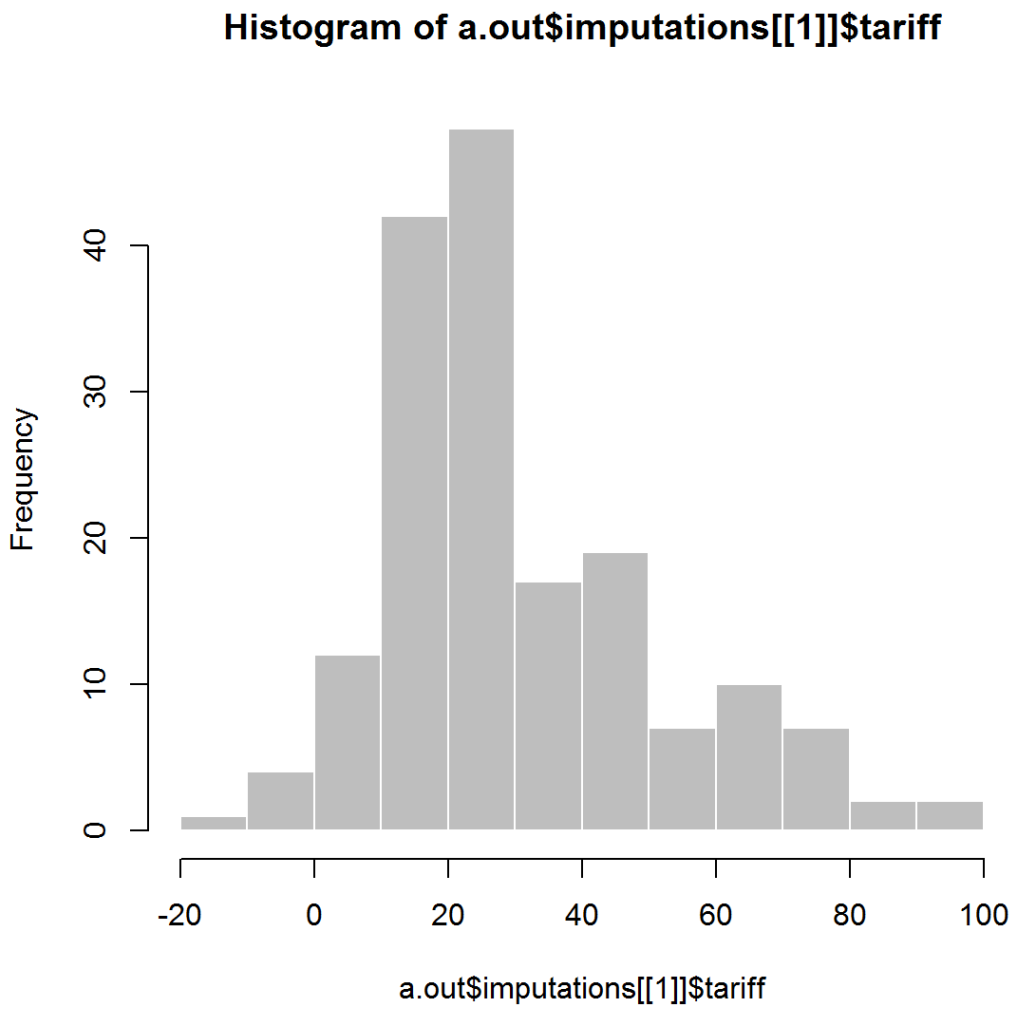
```
a.out <- amelia(freetrade, m = 5, idvars=2) # idvars informs the columns that are identifi
cation variables
```

```
## -- Imputation 1 --
##
##   1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18
##
## -- Imputation 2 --
##
##   1  2  3  4  5  6  7  8  9 10 11 12 13 14
##
## -- Imputation 3 --
##
##   1  2  3  4  5  6  7  8  9 10 11 12 13 14 15
##
## -- Imputation 4 --
##
##   1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18
##
## -- Imputation 5 --
##
##   1  2  3  4  5  6  7  8  9 10 11 12 13 14 15
```

```
head(a.out$imputations[[1]], 12)
```

```
##    year  country   tariff polity      pop   gdp.pc intresmi    signed
## 1  1981 SriLanka 48.93841      6 14988000 461.0236 1.937347 0.0000000
## 2  1982 SriLanka 47.44507      5 15189000 473.7634 1.964430 0.0000000
## 3  1983 SriLanka 41.30000      5 15417000 489.2266 1.663936 1.0000000
## 4  1984 SriLanka 48.57989      5 15599000 508.1739 2.797462 0.0000000
## 5  1985 SriLanka 31.00000      5 15837000 525.5609 2.259116 0.0000000
## 6  1986 SriLanka 56.17484      5 16117000 538.9237 1.832549 0.0000000
## 7  1987 SriLanka 27.30000      5 16361000 540.0475 1.422983 0.0000000
## 8  1988 SriLanka 27.30000      5 16587000 545.8610 1.059624 1.0000000
## 9  1989 SriLanka 46.04678      5 16806000 551.1353 1.137557 0.0000000
## 10 1990 SriLanka 28.30000      5 16993000 579.9548 1.663632 0.0000000
## 11 1991 SriLanka 26.90000      5 17247000 597.6987 2.285213 1.0000000
## 12 1992 SriLanka 25.00000      5 17405000 618.3329 2.877877 0.4811737
##    fiveop     usheg
## 1    12.4 0.2593112
## 2    12.5 0.2558008
## 3    12.3 0.2655022
## 4    12.3 0.2988009
## 5    12.3 0.2952431
## 6    12.5 0.2886563
## 7    12.5 0.2734092
## 8    12.6 0.2756469
## 9    12.6 0.2785387
## 10   12.7 0.2608332
## 11   12.8 0.2589872
## 12   13.1 0.2623017
```

```
hist(a.out$imputations[[1]]$tariff, col="grey", border="white")
```




```
########

a.out <- amelia(freetrade, m = 1, idvars=2,
                ords = "polity", noms='signed') # define 'polity' as nominal, and signed'
 as ordinal
```

```
## -- Imputation 1 --
##
##   1  2  3  4  5  6  7  8  9 10 11 12
```

```
# nominal var with p categories is translated to p-1 variables

head(a.out$imputations[[1]], 12)
```

```
##    year  country   tariff polity      pop   gdp.pc intresmi signed fiveop
## 1  1981 SriLanka 40.06983      6 14988000 461.0236 1.937347      0   12.4
## 2  1982 SriLanka 41.56279      5 15189000 473.7634 1.964430      0   12.5
## 3  1983 SriLanka 41.30000      5 15417000 489.2266 1.663936      1   12.3
## 4  1984 SriLanka 32.20819      5 15599000 508.1739 2.797462      0   12.3
## 5  1985 SriLanka 31.00000      5 15837000 525.5609 2.259116      0   12.3
## 6  1986 SriLanka 36.07042      5 16117000 538.9237 1.832549      0   12.5
## 7  1987 SriLanka 27.30000      5 16361000 540.0475 1.422983      0   12.5
## 8  1988 SriLanka 27.30000      5 16587000 545.8610 1.059624      1   12.6
## 9  1989 SriLanka 80.13476      5 16806000 551.1353 1.137557      0   12.6
## 10 1990 SriLanka 28.30000      5 16993000 579.9548 1.663632      0   12.7
## 11 1991 SriLanka 26.90000      5 17247000 597.6987 2.285213      1   12.8
## 12 1992 SriLanka 25.00000      5 17405000 618.3329 2.877877      0   13.1
##        usheg
## 1  0.2593112
## 2  0.2558008
## 3  0.2655022
## 4  0.2988009
## 5  0.2952431
## 6  0.2886563
## 7  0.2734092
## 8  0.2756469
## 9  0.2785387
## 10 0.2608332
## 11 0.2589872
## 12 0.2623017
```

```
##########
# if a column indicates a time sequence, use parameter ts
# Amelia will add covariates to the model that correspond to time
# and its polynomials. These covariates will help better predict the missing values.
a.out <- amelia(freetrade, m = 5, idvars=2, ts="year",
                ords = "polity", noms='signed')
```

```
## -- Imputation 1 --
##
##   1  2  3  4  5  6  7  8  9 10 11 12 13 14
##
## -- Imputation 2 --
##
##   1  2  3  4  5  6  7  8  9 10 11 12 13 14 15
##
## -- Imputation 3 --
##
##   1  2  3  4  5  6  7  8  9 10 11 12
##
## -- Imputation 4 --
##
##   1  2  3  4  5  6  7  8  9 10 11 12 13
##
## -- Imputation 5 --
##
##   1  2  3  4  5  6  7  8  9 10 11 12 13 14 15
```

```
head(a.out$imputations[[1]], 12)
```

```
##    year  country   tariff polity      pop   gdp.pc intresmi signed fiveop
## 1  1981 SriLanka 47.98727      6 14988000 461.0236 1.937347      0   12.4
## 2  1982 SriLanka 44.89338      5 15189000 473.7634 1.964430      0   12.5
## 3  1983 SriLanka 41.30000      5 15417000 489.2266 1.663936      1   12.3
## 4  1984 SriLanka 27.57339      5 15599000 508.1739 2.797462      0   12.3
## 5  1985 SriLanka 31.00000      5 15837000 525.5609 2.259116      0   12.3
## 6  1986 SriLanka 26.44747      5 16117000 538.9237 1.832549      0   12.5
## 7  1987 SriLanka 27.30000      5 16361000 540.0475 1.422983      0   12.5
## 8  1988 SriLanka 27.30000      5 16587000 545.8610 1.059624      1   12.6
## 9  1989 SriLanka 41.13784      5 16806000 551.1353 1.137557      0   12.6
## 10 1990 SriLanka 28.30000      5 16993000 579.9548 1.663632      0   12.7
## 11 1991 SriLanka 26.90000      5 17247000 597.6987 2.285213      1   12.8
## 12 1992 SriLanka 25.00000      5 17405000 618.3329 2.877877      1   13.1
##        usheg
## 1  0.2593112
## 2  0.2558008
## 3  0.2655022
## 4  0.2988009
## 5  0.2952431
## 6  0.2886563
## 7  0.2734092
## 8  0.2756469
## 9  0.2785387
## 10 0.2608332
## 11 0.2589872
## 12 0.2623017
```

```
############
# If cross-sectional units are specified these polynomials can be interacted with the
# cross-section unit to allow the patterns over time to vary between cross-sectional
# units. Unless you strongly believe all units have the same patterns over time in all
# variables (including the same constant term), this is a reasonable setting

# if a column indicates a time sequence, use parameter ts
# Amelia will add covariates to the model that correspond to time
# and its polynomials. These covariates will help better predict the missing values.
a.out <- amelia(freetrade, m = 5, ts="year", cs="country",
                ords = "polity", noms='signed')
```

```
## -- Imputation 1 --
##
##   1  2  3  4  5  6  7  8  9 10 11 12 13
##
## -- Imputation 2 --
##
##   1  2  3  4  5  6  7  8  9 10 11 12
##
## -- Imputation 3 --
##
##   1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
##  21
##
## -- Imputation 4 --
##
##   1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16
##
## -- Imputation 5 --
##
##   1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
```

```
head(a.out$imputations[[1]], 12)
```

```
##    year  country   tariff polity      pop   gdp.pc intresmi signed fiveop
## 1  1981 SriLanka 65.24997      6 14988000 461.0236 1.937347      0   12.4
## 2  1982 SriLanka 35.46509      5 15189000 473.7634 1.964430      0   12.5
## 3  1983 SriLanka 41.30000      5 15417000 489.2266 1.663936      1   12.3
## 4  1984 SriLanka 25.93887      5 15599000 508.1739 2.797462      0   12.3
## 5  1985 SriLanka 31.00000      5 15837000 525.5609 2.259116      0   12.3
## 6  1986 SriLanka 28.35509      5 16117000 538.9237 1.832549      0   12.5
## 7  1987 SriLanka 27.30000      5 16361000 540.0475 1.422983      0   12.5
## 8  1988 SriLanka 27.30000      5 16587000 545.8610 1.059624      1   12.6
## 9  1989 SriLanka 19.96578      5 16806000 551.1353 1.137557      0   12.6
## 10 1990 SriLanka 28.30000      5 16993000 579.9548 1.663632      0   12.7
## 11 1991 SriLanka 26.90000      5 17247000 597.6987 2.285213      1   12.8
## 12 1992 SriLanka 25.00000      5 17405000 618.3329 2.877877      0   13.1
##        usheg
## 1  0.2593112
## 2  0.2558008
## 3  0.2655022
## 4  0.2988009
## 5  0.2952431
## 6  0.2886563
## 7  0.2734092
## 8  0.2756469
## 9  0.2785387
## 10 0.2608332
## 11 0.2589872
## 12 0.2623017
```

```
##########

a.out <- amelia(freetrade, m = 5, ts = "year", cs = "country")
```

```
## -- Imputation 1 --
##
##   1  2  3  4  5  6  7  8  9 10
##
## -- Imputation 2 --
##
##   1  2  3  4  5  6  7  8  9 10 11 12 13
##
## -- Imputation 3 --
##
##   1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
##  21
##
## -- Imputation 4 --
##
##   1  2  3  4  5  6  7  8  9 10 11 12 13 14
##
## -- Imputation 5 --
##
##   1  2  3  4  5  6  7  8  9 10 11 12 13 14 15
```

```
# intercs is a logical variable indicating if the time effects of polytime should vary acr
oss the cross-section.
# polytime is what power of polynomial should be included in the imputation model to accou
nt for the effects of time.
a.out.time <- amelia(freetrade, m = 5, ts="year", cs="country", intercs=T, polytime = 2)
```

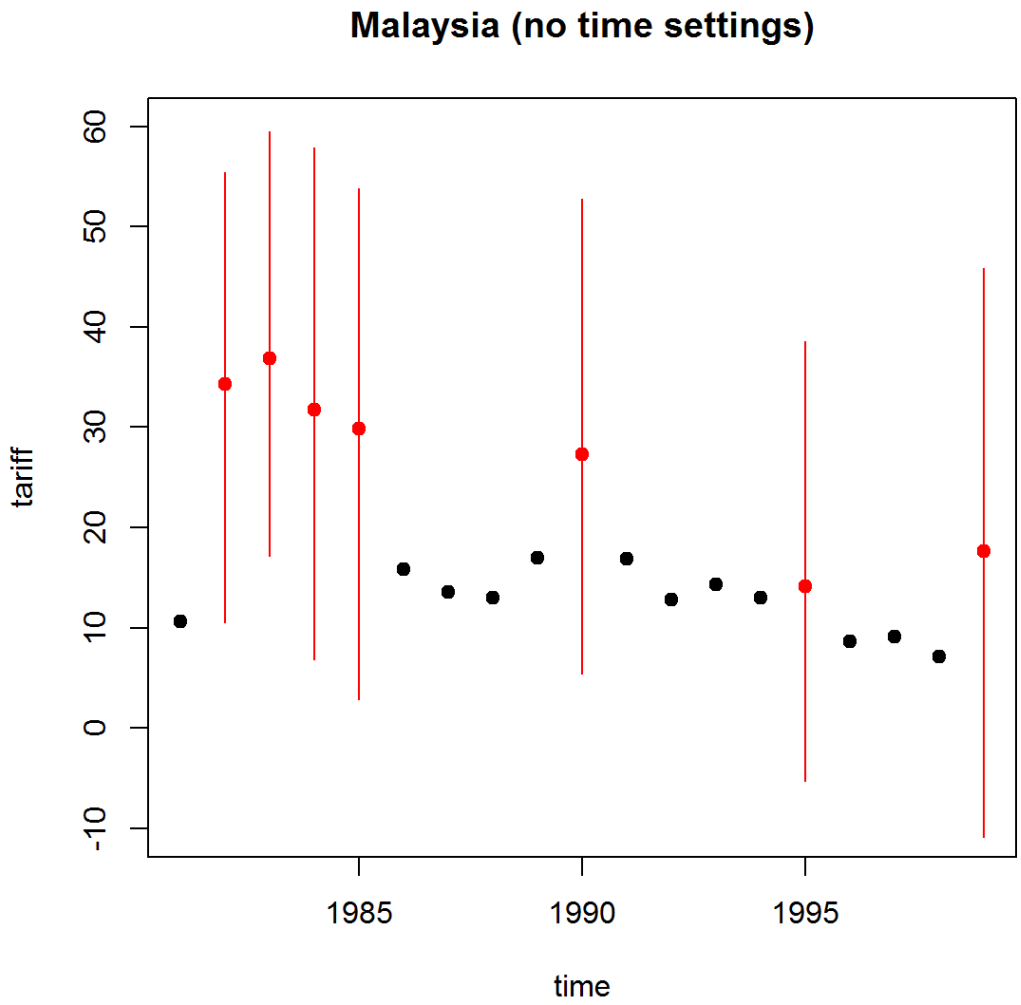
```
## -- Imputation 1 --
##
##   1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
##  21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40
##  41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60
##  61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80
##  81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100
##  101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120
##  121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 139 140
##  141 142 143 144
##
## -- Imputation 2 --
##
##   1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
##  21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40
##  41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60
##  61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80
##  81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100
##  101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120
##  121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 139 140
##  141 142 143 144 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160
##  161 162 163 164 165 166 167 168 169 170 171 172 173 174 175 176 177 178 179 180
##  181 182 183 184 185 186 187 188
##
## -- Imputation 3 --
##
##   1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
##  21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40
##  41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60
##  61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80
##  81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100
##  101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120
##  121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 139 140
##  141 142 143 144 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160
##  161 162 163 164 165 166 167 168 169 170 171 172 173 174 175 176 177 178 179 180
##  181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 198 199 200
##  201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216 217 218 219 220
##  221 222 223 224 225 226 227 228 229 230 231 232 233 234 235 236 237 238 239 240
##  241 242 243 244 245 246 247 248 249 250 251 252 253 254 255 256 257 258 259 260
##  261 262 263 264 265 266 267 268 269 270 271 272 273 274 275 276 277 278 279 280
##  281 282 283 284 285 286 287 288 289 290 291 292 293 294 295 296 297 298 299 300
##  301 302 303 304 305 306 307 308 309 310 311 312 313 314 315 316 317 318 319 320
##  321 322 323 324 325 326 327 328 329 330 331 332 333 334 335 336 337 338 339 340
##  341 342 343 344 345 346 347 348 349 350 351 352 353 354 355 356 357 358 359 360
##  361 362 363 364 365 366 367 368 369 370 371 372 373 374 375 376 377 378 379 380
##  381 382 383 384 385 386 387 388 389 390 391 392 393 394 395 396 397 398 399 400
##  401 402 403 404 405 406 407 408 409 410 411 412 413 414 415 416 417 418 419 420
##  421 422 423 424 425 426 427 428 429 430 431 432 433 434 435 436 437 438 439 440
##  441 442 443 444 445 446 447 448 449 450 451 452 453 454 455 456 457 458 459 460
##  461 462 463 464 465 466 467 468 469 470 471 472 473 474 475 476 477 478 479 480
##  481 482 483 484 485
##
## -- Imputation 4 --
##
##   1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
##  21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40
##  41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60
##  61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80
##  81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100
##  101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120
```

```
##  121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 139 140
##  141 142 143 144 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160
##  161 162 163 164 165 166 167 168 169 170 171 172 173 174 175 176 177 178 179 180
##  181 182 183 184 185
##
## -- Imputation 5 --
##
##   1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
##  21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40
##  41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60
##  61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80
##  81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100
##  101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120
##  121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 139 140
##  141 142 143 144 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160
##  161 162 163 164 165 166 167 168 169 170 171 172 173 174 175 176 177 178 179 180
##  181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197
```

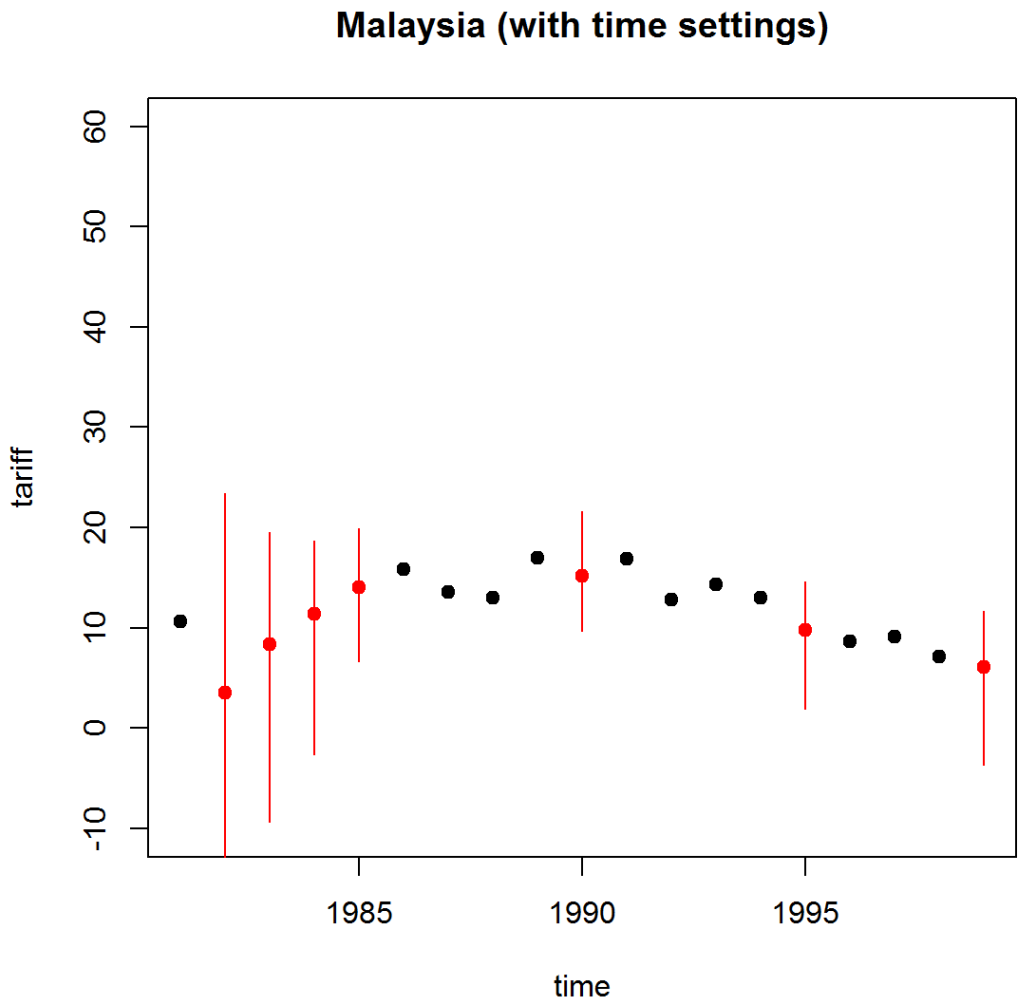
```
head(a.out.time$imputations[[1]], 12)
```

```
##    year  country    tariff polity      pop   gdp.pc intresmi     signed
## 1  1981 SriLanka -1.947011      6 14988000 461.0236 1.937347  0.0000000
## 2  1982 SriLanka 10.174298      5 15189000 473.7634 1.964430  0.0000000
## 3  1983 SriLanka 41.299999      5 15417000 489.2266 1.663936  1.0000000
## 4  1984 SriLanka 19.551316      5 15599000 508.1739 2.797462  0.0000000
## 5  1985 SriLanka 31.000000      5 15837000 525.5609 2.259116  0.0000000
## 6  1986 SriLanka 24.976304      5 16117000 538.9237 1.832549  0.0000000
## 7  1987 SriLanka 27.299999      5 16361000 540.0475 1.422983  0.0000000
## 8  1988 SriLanka 27.299999      5 16587000 545.8610 1.059624  1.0000000
## 9  1989 SriLanka 25.694741      5 16806000 551.1353 1.137557  0.0000000
## 10 1990 SriLanka 28.299999      5 16993000 579.9548 1.663632  0.0000000
## 11 1991 SriLanka 26.900000      5 17247000 597.6987 2.285213  1.0000000
## 12 1992 SriLanka 25.000000      5 17405000 618.3329 2.877877 -0.3546901
##    fiveop     usheg
## 1    12.4 0.2593112
## 2    12.5 0.2558008
## 3    12.3 0.2655022
## 4    12.3 0.2988009
## 5    12.3 0.2952431
## 6    12.5 0.2886563
## 7    12.5 0.2734092
## 8    12.6 0.2756469
## 9    12.6 0.2785387
## 10   12.7 0.2608332
## 11   12.8 0.2589872
## 12   13.1 0.2623017
```

```
# we have a much better prediction about the missing values when incorporating time
# than when we omit it:

tscsPlot(a.out, cs = "Malaysia", main = "Malaysia (no time settings)",
         var = "tariff", ylim = c(-10, 60))
```




```
tscsPlot(a.out.time, cs = "Malaysia", main = "Malaysia (with time settings)",
         var = "tariff", ylim = c(-10, 60))
```




```
##################
# The incorporation of priors follows basic Bayesian analysis where the imputation
# turns out to be a weighted average of the model-based imputation and the prior
# mean, where the weights are functions of the relative strength of the data and prior

# For instance, suppose that we had some expert prior information about tariff
# rates in Thailand.

freetrade[freetrade$country == "Thailand", c("year","country","tariff")]
```

```
##     year  country tariff
## 153 1981 Thailand   32.3
## 154 1982 Thailand     NA
## 155 1983 Thailand     NA
## 156 1984 Thailand     NA
## 157 1985 Thailand   41.2
## 158 1986 Thailand     NA
## 159 1987 Thailand     NA
## 160 1988 Thailand     NA
## 161 1989 Thailand   40.8
## 162 1990 Thailand   39.8
## 163 1991 Thailand   37.8
## 164 1992 Thailand     NA
## 165 1993 Thailand   45.6
## 166 1994 Thailand   23.3
## 167 1995 Thailand   23.1
## 168 1996 Thailand     NA
## 169 1997 Thailand     NA
## 170 1998 Thailand   20.1
## 171 1999 Thailand   17.1
```

```
# that tariff rates were roughly 40% in
# Thailand between 1986 and 1988 with about a 6% margin of error with 95% confidence. This
 corresponds
# to a standard deviation of about 3 (the interval of 2-sigma is 6).
# In order to include this information, we must form the priors matrix:

pr <- matrix(c(158, 3, 40, 3,
               159, 3, 40, 3,
               160, 3, 40, 3), ncol=4, byrow=T)

# The first column of this matrix corresponds to the row numbers of Thailand in
# these three years, the second column refers to the column number of tariff in the
# data and the last two columns refer to the actual prior.

a.out.pr <- amelia(freetrade, ts = "year", cs = "country", priors = pr)
```

```
## -- Imputation 1 --
##
##   1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16
##
## -- Imputation 2 --
##
##   1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
##  21
##
## -- Imputation 3 --
##
##   1  2  3  4  5  6  7  8  9 10
##
## -- Imputation 4 --
##
##   1  2  3  4  5  6  7  8  9 10 11 12
##
## -- Imputation 5 --
##
##   1  2  3  4  5  6  7  8  9 10 11 12
```

```
a.out.pr$imputations[[1]][a.out.pr$imputations[[1]]$country == "Thailand", c("year","country","tariff")]
```

```
##     year  country   tariff
## 153 1981 Thailand 32.30000
## 154 1982 Thailand 35.15511
## 155 1983 Thailand 51.91600
## 156 1984 Thailand 42.73209
## 157 1985 Thailand 41.20000
## 158 1986 Thailand 38.09353
## 159 1987 Thailand 39.57465
## 160 1988 Thailand 37.19743
## 161 1989 Thailand 40.80000
## 162 1990 Thailand 39.80000
## 163 1991 Thailand 37.80000
## 164 1992 Thailand 36.61743
## 165 1993 Thailand 45.60000
## 166 1994 Thailand 23.30000
## 167 1995 Thailand 23.10000
## 168 1996 Thailand 19.34810
## 169 1997 Thailand 12.31609
## 170 1998 Thailand 20.10000
## 171 1999 Thailand 17.10000
```

```
# Another way is to present a confidence interval:

pr <- matrix(c(158, 3, 34, 46, 0.95, # [34,46] with 95% confidence
               159, 3, 34, 46, 0.95,
               160, 3, 34, 46, 0.95), ncol=5, byrow=T)

a.out.pr <- amelia(freetrade, ts = "year", cs = "country", priors = pr)
```

```
## -- Imputation 1 --
##
##   1  2  3  4  5  6  7  8  9 10 11 12 13 14
##
## -- Imputation 2 --
##
##   1  2  3  4  5  6  7  8  9 10 11 12 13
##
## -- Imputation 3 --
##
##   1  2  3  4  5  6  7  8  9 10
##
## -- Imputation 4 --
##
##   1  2  3  4  5  6  7  8  9 10 11 12 13 14 15
##
## -- Imputation 5 --
##
##   1  2  3  4  5  6  7  8  9 10
```

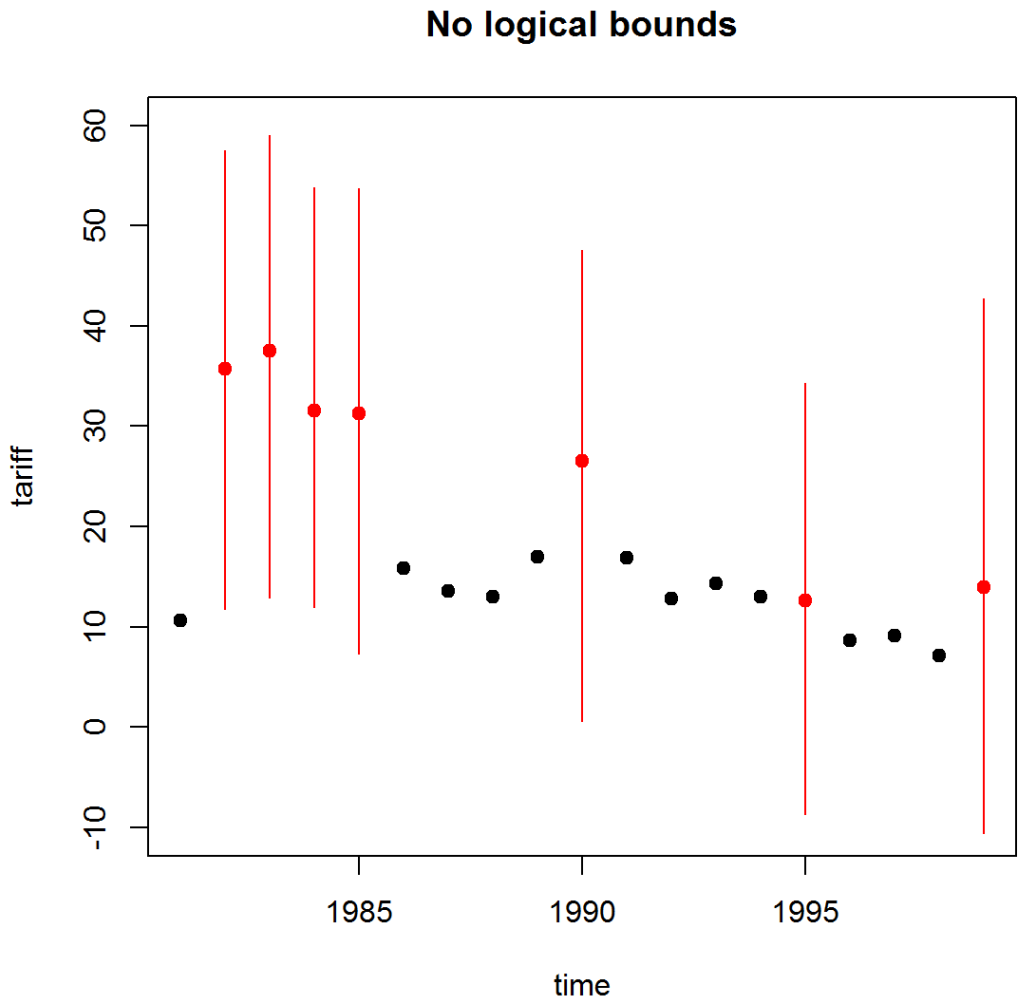
```
a.out.pr$imputations[[1]][a.out.pr$imputations[[1]]$country == "Thailand", c("year","country","tariff")]
```

```
##     year  country   tariff
## 153 1981 Thailand 32.30000
## 154 1982 Thailand 25.18532
## 155 1983 Thailand 39.37597
## 156 1984 Thailand 45.73508
## 157 1985 Thailand 41.20000
## 158 1986 Thailand 36.10108
## 159 1987 Thailand 40.62394
## 160 1988 Thailand 42.43548
## 161 1989 Thailand 40.80000
## 162 1990 Thailand 39.80000
## 163 1991 Thailand 37.80000
## 164 1992 Thailand 13.04487
## 165 1993 Thailand 45.60000
## 166 1994 Thailand 23.30000
## 167 1995 Thailand 23.10000
## 168 1996 Thailand 24.31571
## 169 1997 Thailand 18.41202
## 170 1998 Thailand 20.10000
## 171 1999 Thailand 17.10000
```

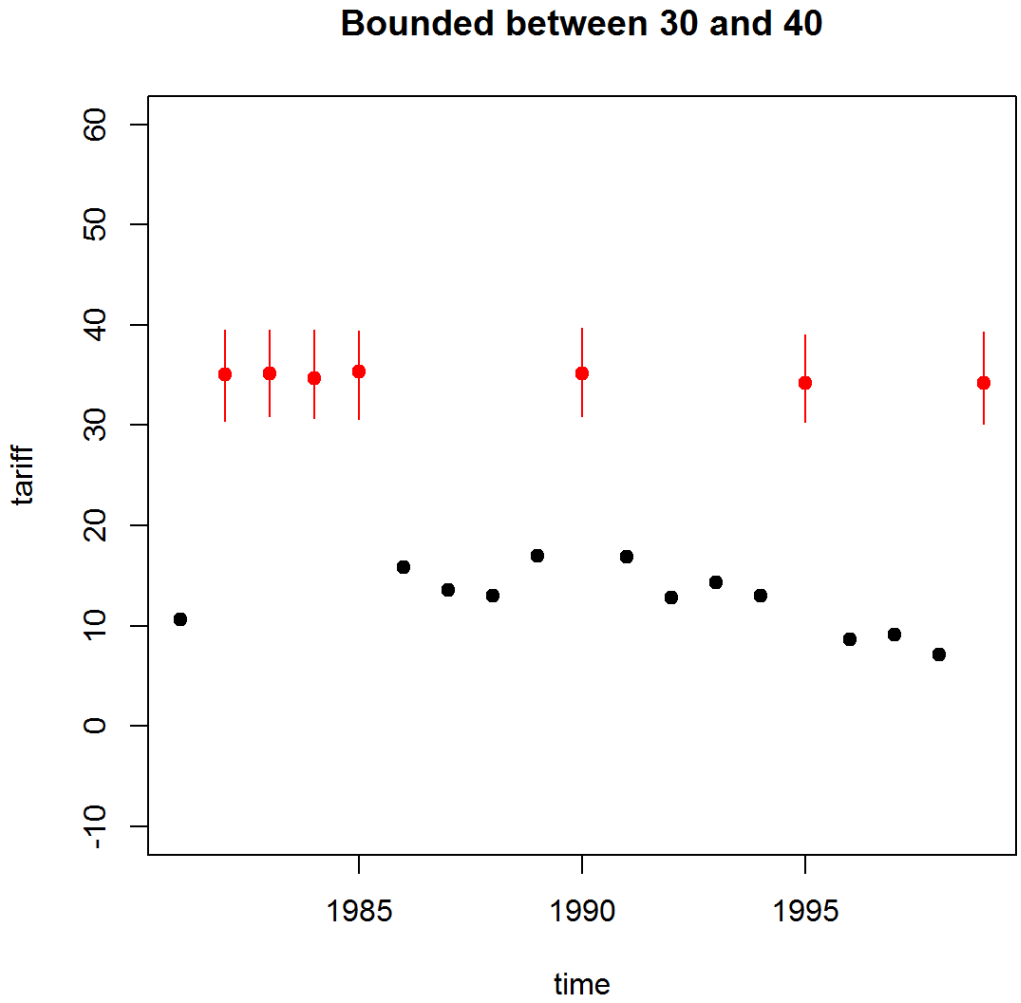
```
# If a prior has the value 0 in the first column, this prior will be applied to all
# missing values in this variable, except for explicitly set priors. T

pr <- matrix(c(158, 3, 34, 46, 0.95, # [34,46] with 95% confidence
               159, 3, 34, 46, 0.95,
               160, 3, 34, 46, 0.95,
                0 , 3, 30, 50, 0.90), ncol=5, byrow=T)

#########
# it's possible to define bounds to the imputations

bds <- matrix(c(3, 30, 40), nrow = 1, ncol = 3) # 3rd column can only have values between
 [30,40]

a.out.bds <- amelia(freetrade, ts = "year", cs = "country", bounds = bds, max.resample = 1
000)
```

```
## -- Imputation 1 --
##
##   1  2  3  4  5  6  7  8  9 10 11 12 13
##
## -- Imputation 2 --
##
##   1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
##
##
## -- Imputation 3 --
##
##   1  2  3  4  5  6  7  8  9 10 11 12 13 14 15
##
## -- Imputation 4 --
##
##   1  2  3  4  5  6  7  8  9 10 11
##
## -- Imputation 5 --
##
##   1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19
```

```
tscsPlot(a.out, cs = "Malaysia", main = "No logical bounds", var =
           "tariff", ylim = c(-10,60))
```



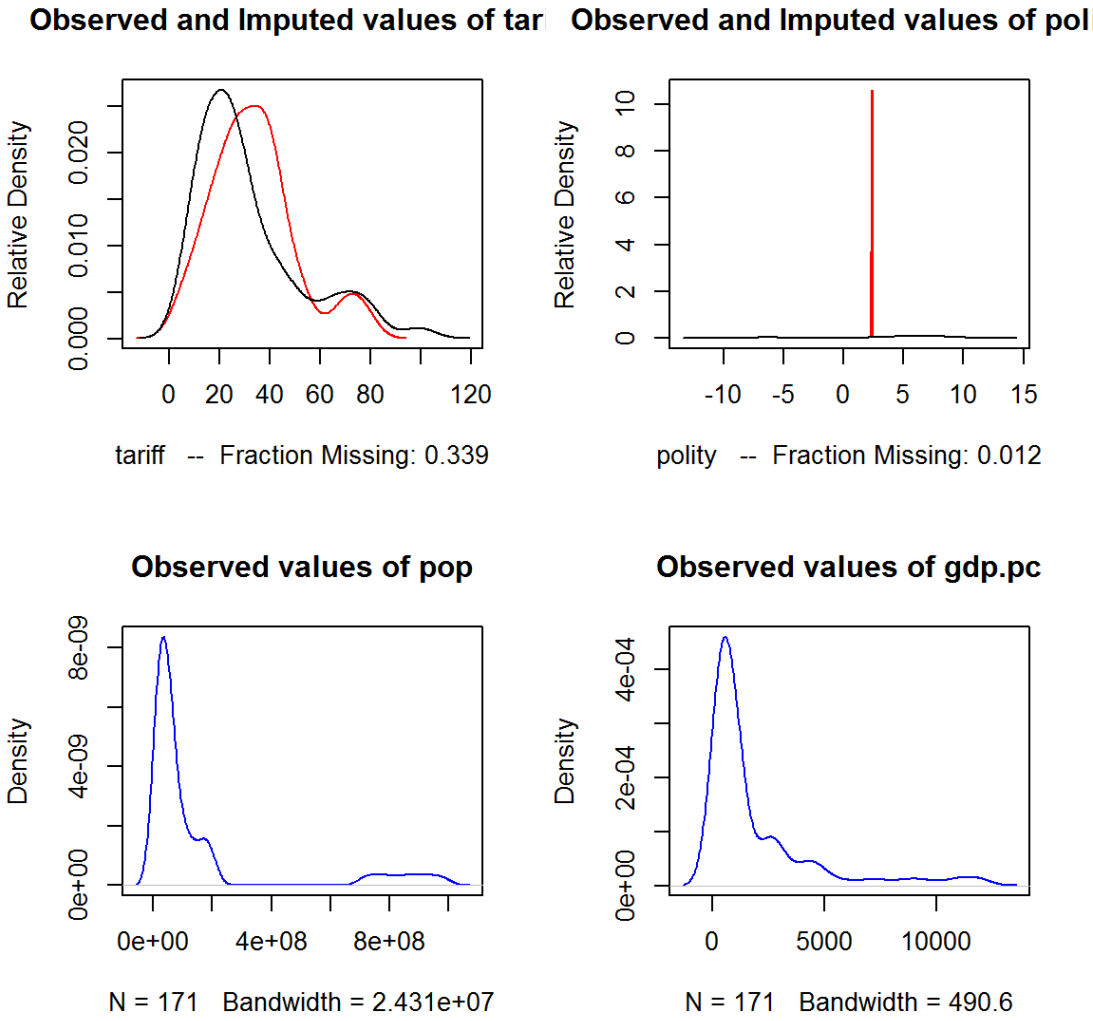

```
tscsPlot(a.out.bds, cs = "Malaysia", main = "Bounded between 30 and 40", var =
             "tariff", ylim = c(-10,60))
```



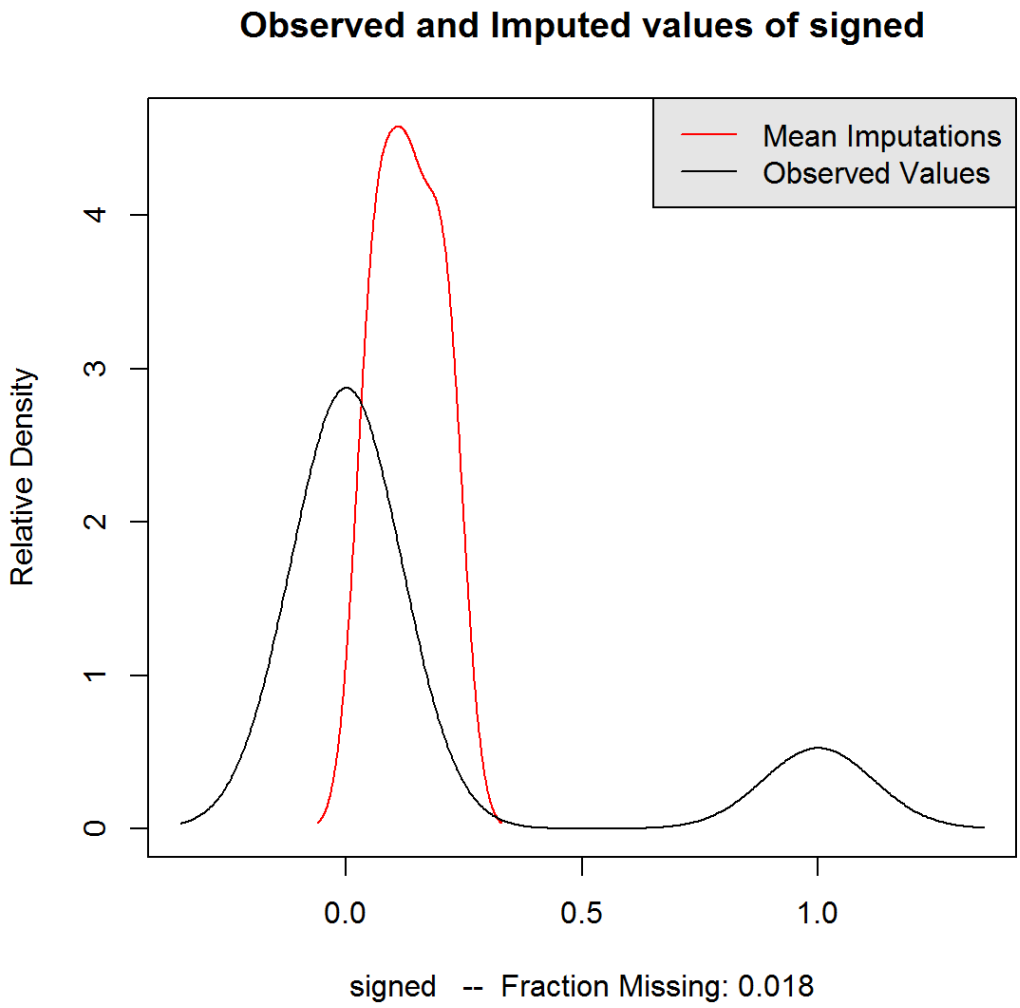

```
# Analysts should be extremely cautious when using these bounds as they
# can seriously affect the inferences from the imputation model, as shown in this
# example. Even when logical bounds exist, we recommend simply imputing variables
# normally, as the violation of the logical bounds represents part of the true uncertainty
# of imputation.
```

Diagnostics

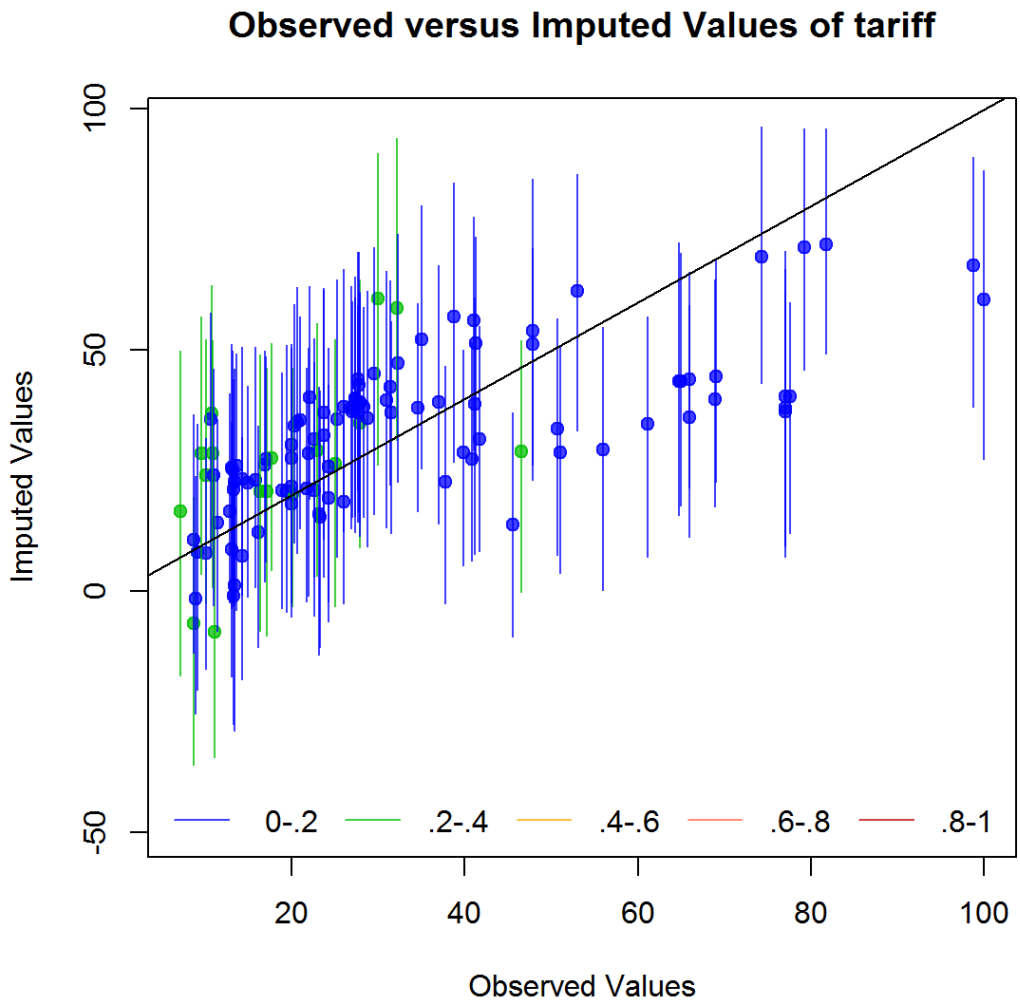
```
# comparing densities

# plot the density of the mean imputation over the m datasets.
# That is, for each cell that is missing in the variable, the diagnostic will find
# the mean of that cell across each of the m datasets and use that value for the density
# plot (in red). The black distributions are the those of the observed data. When variable
s
# are completely observed, their densities are plotted in blue
plot(a.out, which.vars = 3:6)
```



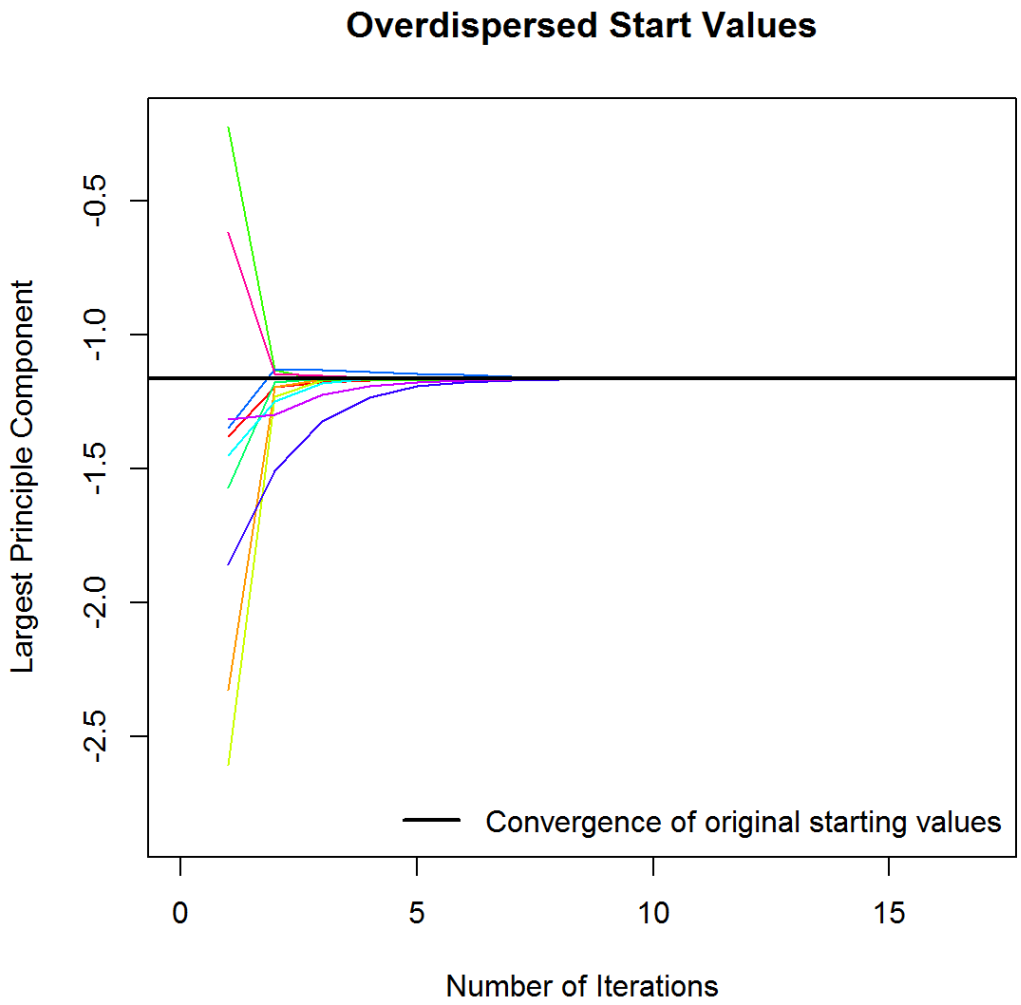

```
par(mfrow=c(1,1))

# Imputations with strange distributions or those that are far from the
# observed data may indicate that imputation model needs at least some investigation
# and possibly some improvement.

compare.density(a.out, var = "signed") # to plot just one variable
```



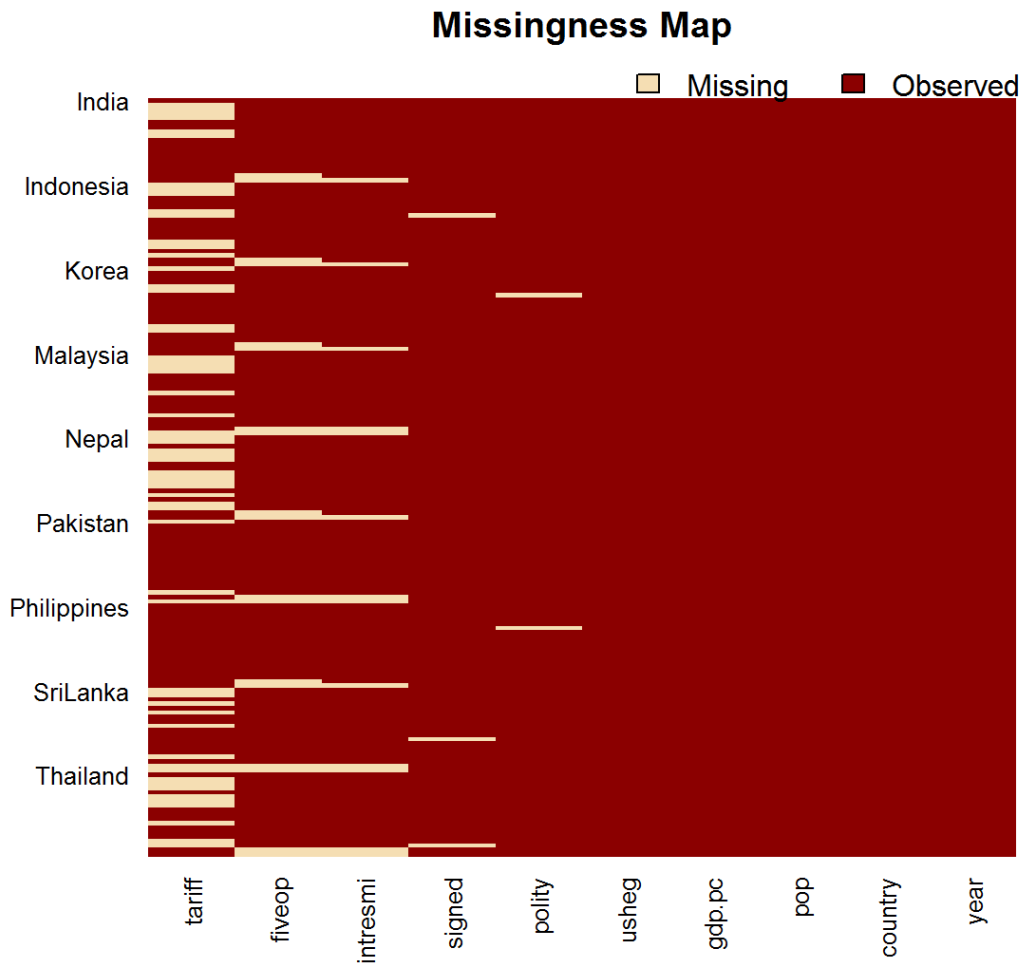

```
# overimputing

# Overimputing involves sequentially treating each of the observed values as if
# they had actually been missing. For each observed value in turn we then generate
# several hundred imputed values of that observed value, as if it had been missing.
# While m = 5 imputations are sufficient for most analysis models, this large number
# of imputations allows us to construct a confidence interval of what the imputed
# value would have been, had any of the observed data been missing. We can then
# graphically inspect whether our observed data tends to fall within the region where
# it would have been imputed had it been missing

# On this graph, a y = x
# line indicates the line of perfect agreement; that is, if the imputation model was a
# perfect predictor of the true value, all the imputations would fall on this line. For
# each observation, Amelia also plots 90% confidence intervals that allows the user to
# visually inspect the behavior of the imputation model.

# As the amount of missing information
# in a particular pattern of missingness increases, we expect the width of the
# confidence interval to increase. The color of the confidence interval reflects the perce
nt
# of covariates observed in that pattern of missingness, as reflected in the legend at
# the bottom.

overimpute(a.out, var = "tariff")
```




```
# Because the EM algorithm is deterministic,
# the point in the parameter space where you start it can impact where it ends, though
# this is irrelevant when the likelihood has only one mode. However, if the starting
# values of an EM chain are close to a local maximum, the algorithm may find this
# maximum, unaware that there is a global maximum farther away. To make sure that
# our imputations do not depend on our starting values, a good test is to run the EM
# algorithm from multiple, dispersed starting values and check their convergence.

# Amelia includes a diagnostic to run the EM chain from multiple starting values
# that are overdispersed from the estimated maximum. The overdispersion diagnostic
# will display a graph of the paths of each chain. Since these chains move through
# spaces that are in an extremely high number of dimensions and can not be graphically
# displayed, the diagnostic reduces the dimensionality of the EM paths by showing
# the paths relative to the largest principle components of the final mode(s) that are
# reached.

disperse(a.out, dims = 1, m = 10)
```




```
# One useful tool for exploring the missingness in a dataset is a missingness map. This
# is a map that visualizes the dataset a grid and colors the grid by missingness status.
# The column of the grid are the variables and the rows are the observations.

# The missmap function arrange the columns so that the
# variables are in decreasing order of missingness from left to right. If the cs argument
# was set in the amelia function, the labels for the rows will indicate where each of
# the cross-sections begin.

missmap(a.out)
```



Analysis Models

```
library(Zelig)
```

```
## Loading required package: boot
## Loading required package: sandwich
## ZELIG (Versions 4.2-1, built: 2013-09-12)
##
## +-----------------------------------------------------------------+
## |  Please refer to http://gking.harvard.edu/zelig for full        |
## |  documentation or help.zelig() for help with commands and       |
## |  models support by Zelig.                                       |
## |                                                                 |
## |  Zelig project citations:                                       |
## |    Kosuke Imai, Gary King, and Olivia Lau.  (2009).             |
## |    ``Zelig: Everyone's Statistical Software,''                  |
## |    http://gking.harvard.edu/zelig                               |
## |   and                                                           |
## |    Kosuke Imai, Gary King, and Olivia Lau. (2008).              |
## |    ``Toward A Common Framework for Statistical Analysis         |
## |    and Development,'' Journal of Computational and              |
## |    Graphical Statistics, Vol. 17, No. 4 (December)              |
## |    pp. 892-913.                                                 |
## |                                                                 |
## |   To cite individual Zelig models, please use the citation      |
## |   format printed with each model run and in the documentation.  |
## +-----------------------------------------------------------------+
##
##
##
## Attaching package: 'Zelig'
##
## The following object is masked from 'package:mclust':
##
##     sim
##
## The following object is masked from 'package:utils':
##
##     cite
```

```
z.out <- zelig(tariff ~ polity + pop + gdp.pc + year +country, data = freetrade, model =
"ls")
```

```
##
##
##  How to cite this model in Zelig:
##   Kosuke Imai, Gary King, and Olivia Lau. 2015.
##   "ls: Least Squares Regression for Continuous Dependent Variables"
##   in Kosuke Imai, Gary King, and Olivia Lau, "Zelig: Everyone's Statistical Software,"
##   http://gking.harvard.edu/zelig
##
```

```
summary(z.out)
```

```
##
## Call:
## lm(formula = formula, weights = weights, model = F, data = data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -30.7640  -3.2595   0.0868   2.5983  18.3097
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)         1.973e+03  4.016e+02   4.912 3.61e-06 ***
## polity             -1.373e-01  1.821e-01  -0.754    0.453
## pop                -2.021e-07  2.542e-08  -7.951 3.23e-12 ***
## gdp.pc              6.096e-04  7.442e-04   0.819    0.415
## year               -8.705e-01  2.084e-01  -4.176 6.43e-05 ***
## countryIndonesia   -1.823e+02  1.857e+01  -9.819 2.98e-16 ***
## countryKorea       -2.204e+02  2.078e+01 -10.608  < 2e-16 ***
## countryMalaysia    -2.245e+02  2.171e+01 -10.343  < 2e-16 ***
## countryNepal       -2.163e+02  2.247e+01  -9.629 7.74e-16 ***
## countryPakistan    -1.554e+02  1.982e+01  -7.838 5.63e-12 ***
## countryPhilippines -2.040e+02  2.088e+01  -9.774 3.75e-16 ***
## countrySriLanka    -2.091e+02  2.210e+01  -9.460 1.80e-15 ***
## countryThailand    -1.961e+02  2.095e+01  -9.358 2.99e-15 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6.221 on 98 degrees of freedom
##   (60 observations deleted due to missingness)
## Multiple R-squared:  0.9247, Adjusted R-squared:  0.9155
## F-statistic: 100.3 on 12 and 98 DF,  p-value: < 2.2e-16
```

```
# applying the model considering all imputations
z.out.imp <- zelig(tariff ~ polity + pop + gdp.pc + year +country, data = a.out$imputation
s, model = "ls")
```

```
##
##
##  How to cite this model in Zelig:
##   Kosuke Imai, Gary King, and Olivia Lau. 2015.
##   "ls: Least Squares Regression for Continuous Dependent Variables"
##   in Kosuke Imai, Gary King, and Olivia Lau, "Zelig: Everyone's Statistical Software,"
##   http://gking.harvard.edu/zelig
##
```

```
summary(z.out.imp)
```

```
##
##   Model: ls
##   Number of multiply imputed data sets: 5
##
## Combined results:
##
## Call:
## lm(formula = formula, weights = weights, model = F, data = data)
##
## Coefficients:
##                            Value   Std. Error      t-stat    p-value
## (Intercept)         2.512420e+03 8.468627e+02  2.96673876 0.01391078
## polity             -3.852609e-02 4.117430e-01 -0.09356828 0.92673299
## pop                -1.046568e-07 5.343014e-08 -1.95875889 0.07096090
## gdp.pc             -4.419923e-04 1.306792e-03 -0.33822685 0.73605591
## year               -1.186506e+00 4.393638e-01 -2.70050846 0.02195043
## countryIndonesia   -1.055059e+02 3.770986e+01 -2.79783301 0.01471201
## countryKorea       -1.251940e+02 4.363263e+01 -2.86927522 0.01325183
## countryMalaysia    -1.299523e+02 4.644113e+01 -2.79821617 0.01645913
## countryNepal       -1.257024e+02 4.871590e+01 -2.58031572 0.02566069
## countryPakistan    -8.052342e+01 4.154902e+01 -1.93803412 0.07564534
## countryPhilippines -1.203099e+02 4.370354e+01 -2.75286365 0.01698879
## countrySriLanka    -1.201115e+02 4.598395e+01 -2.61203043 0.02193629
## countryThailand    -1.134698e+02 4.447349e+01 -2.55140280 0.02567604
##
## For combined results from datasets i to j, use summary(x, subset = i:j).
## For separate results, use print(summary(x), subset = i:j).
```