STAT 661: Project

LS v.s. EM

Jacob, Robin, Ryan & Shen

Dec, 2019

1 Least Square Method v.s. EM Method

Schmee, J., & Hahn, G. (1979). A Simple Method for Regression Analysis with Censored Data. Technometrics, 21(4), 417-432. doi:10.2307/1268280

Aitkin, M. (1981). A Note on the Regression Analysis of Censored Data. Technometrics, 23(2), 161-163. doi:10.2307/1268032

1.1 Introduction

Problems requiring regression analysis of censored data arise frequently in practice. For example, in accelerated testing one wishes to relate stress and average time to failure from data including unfailed units, i. e., censored observations. Maximum likelihood is one method for obtaining the desired estimates; in this paper, we propose an alternative approach. An initial least squares fit is obtained treating the censored values as failures. Then, based upon this initial fit, the expected failure time for each censored observation is estimated. These estimates are then used, instead of the censoring times, to obtain a revised least squares fit and new expected failure times are estimated for the censored values. These are then used in a further least squares fit. The procedure is iterated until convergence is achieved. This method is simpler to implement and explain to non-statisticians than maximum likelihood and appears to have good statistical and convergence properties. The method is illustrated by an example, and some simulation results are described. Variations and areas for further study also are discussed.

1.2 Least Square Method

Description of method for simple situation

$\mu_x = \beta_0 + \beta_1 x$

$\mu_x^\star = \mu_x + \frac{\sigma f(z)}{1-F(z)}$ where $z = \frac{(c_x - \mu_x)}{\sigma}$

- Iteration 0
  - Step 1:$\hat{\beta}_0^{(0)} = -4.9307$,$\hat{\beta}_1^{(0)} = 3.7471$,$\hat{\sigma}^{(0)} = 0.1572$.
  - Step 2:$x = \frac{1000}{170+273.2} = 2.256318$

$\hat{\mu}_{2.26}^{(0)} = -4.9307 + 3.7471\frac{1000}{170+273.2} = 3.523948$

$C_{2.26} = \log_{10}(5448) = 3.736237$

$z = \frac{C_{2.26} - \hat{\mu}_{2.26}^{(0)}}{\sigma} = \frac{3.736237-3.523948}{0.1572178} = 1.350286$

$\hat{\mu}_{2.26}^{\star(0)} = \hat{\mu}_{2.26}^{(0)} + \hat{\sigma}^{(0)}\frac{f(z)}{1-F(z)} = 3.8089$ Or 6440 hours

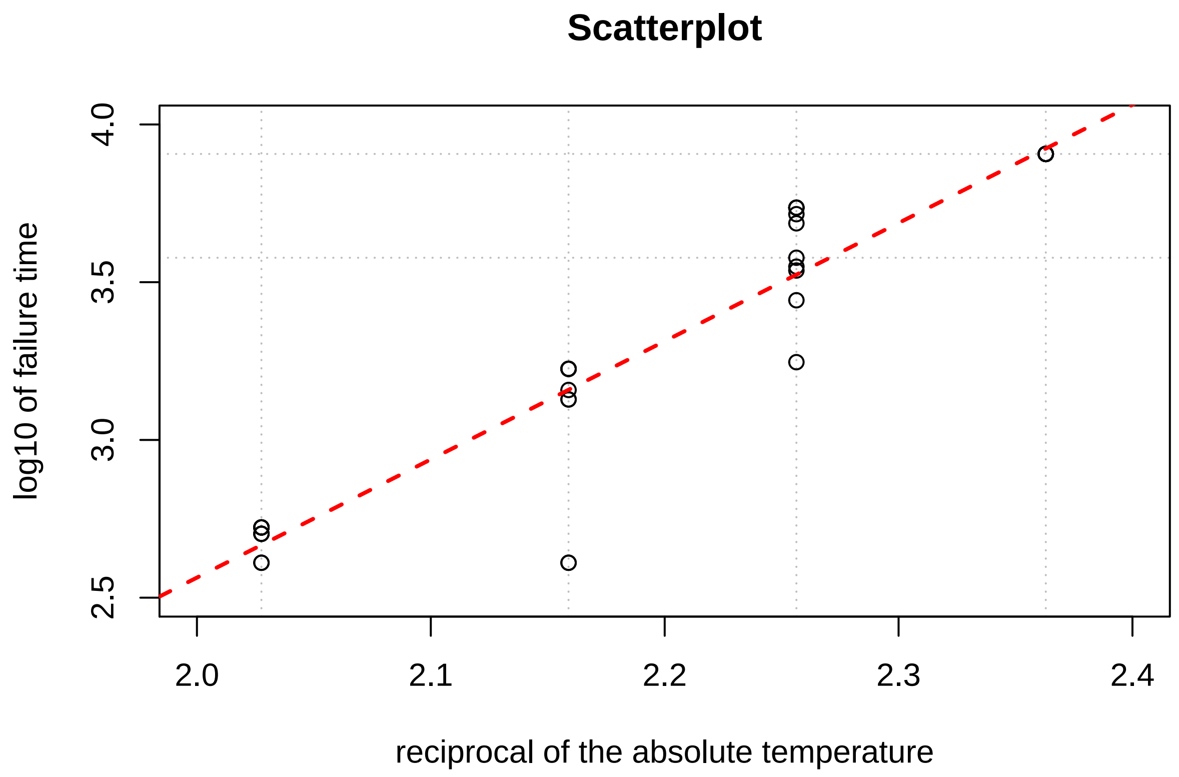
- Iteration 1
  - Step 1:$\hat{\beta}_0^{(1)} = -5.2603$,$\hat{\beta}_1^{(1)} = 3.9263$,$\hat{\sigma}^{(1)} = 0.1799$.
  - Step 2:$\hat{\mu}_{2.26}^{\star(1)} = 3.83972$
- Subsequent Iterations

$\hat{\beta}_0 = -5.81829$,$\hat{\beta}_1 = 4.20426$,$\hat{\sigma} = 0.204322$.

$\hat{\mu}_{2.26}^{\star(17)} = 3.87676$

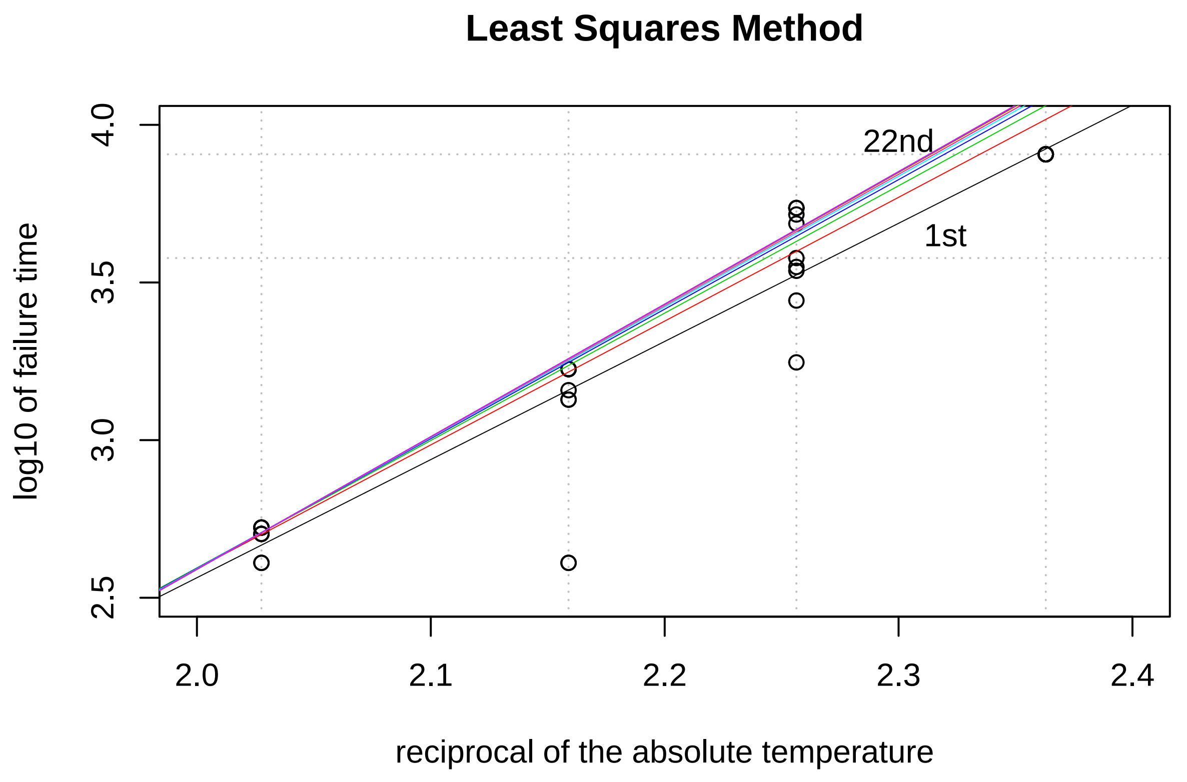
1.3 LS Code

```
temp <- c(150,170,190,220) #temperature levels
trec <- 1000/(temp+273.2) #reciprocal of the absolute temperature T
x <- c(rep(trec[1],10),rep(trec[2],10),rep(trec[3],10),rep(trec[4],10))
cen <- c(8064,5448,1680,528) #censoring times
logcen <- log10(cen) #log10 censoring times
y_uncensored <-log10(c(rep(1,10),1764,2772,3444,3542,3780,4860,5196,rep(1,3),408,408,1344,1344,1440,rep
y_censored <- c(rep(logcen[1],10),rep(0,7),rep(logcen[2],3),rep(0,5),rep(logcen[3],5),rep(0,5),rep(logce
S <- 23; Y<-matrix(nrow=S,ncol=40)
Y[1,] <- y_0 <- y_uncensored+y_censored
fit0 <- lm(y_0~x) #linear model between log10 of observed life time for different r eciprocal values
plot(x,y_0,main="Scatterplot",xlab="reciprocal of the absolute temperature",ylab="log10 of failure time
     xlim=c(2,2.4),ylim=c(2.5,4),
     panel.first=abline(h=c(3.577492,3.906551),v=c(2.027575,2.158895,2.256318,2.362949),lty=3,col="gray
abline(fit0,lwd=2,lty=2,col="red")
```



```
# Iteration 0
sigma_0 <- sigma(fit0) #standard error of residuals
beta_00 <- coef(fit0)[1] #intercept
beta_10 <- coef(fit0)[2] #slope
mu_0 <- beta_00 + beta_10*trec #mean log time to failure
z <- (logcen-mu_0)/sigma_0 #z-vector
ex_mu_0 <- mu_0 + sigma_0*dnorm(z)/(1-pnorm(z)) #new expected mean log times to failure
delta = 1e-006; iteration <- 1
PHI<-matrix(nrow=S,ncol=8,dimnames=list(NULL, c('mu150','mu170','mu190','mu220','Intercept','Slope','Si
PHI[1,]<-phi<-c(ex_mu_0, beta_00, beta_10,sigma_0,iteration)
# Subsequent iteration
repeat {
 phi[8] <- phi[8]+1
 y_censored <- c(rep(phi[1],10),rep(0,7),rep(phi[2],3),rep(0,5),rep(phi[3],5),rep(0,5),rep(phi[4],5))
 y<- y_uncensored+y_censored
 Y[phi[8],]<-y   # Replace the new censored values
 fit <- lm(y~x)   # fit a new model
 phi[5] <- coef(fit)[1] #intercept
 phi[6] <- coef(fit)[2] #slope
 phi[7] <- sigma(fit) #standard error of residuals
 mu <- phi[5] + phi[6]*trec
 z <- (logcen-mu)/phi[7] #z-vector
 phi[1:4] <- mu + phi[7]*dnorm(z)/(1-pnorm(z)) #new expected mean log times to failure
 conv <- dist(rbind(PHI[phi[8]-1,1:4],phi[1:4]))
 if(conv < delta) break
  PHI[phi[8],]<-phi
}
```

mu150	mu170	mu190	mu220	Intercept	Slope	Sigma	Iteration
4.038	3.809	3.329	2.83	-4.931	3.747	0.1572	1
4.099	3.84	3.366	2.858	-5.26	3.926	0.1799	2
4.131	3.856	3.382	2.869	-5.486	4.04	0.1911	3
4.149	3.865	3.39	2.874	-5.623	4.108	0.1969	4
4.159	3.87	3.395	2.877	-5.704	4.148	0.2001	5
4.165	3.873	3.397	2.878	-5.752	4.172	0.2019	6
4.168	3.874	3.399	2.879	-5.779	4.185	0.2029	7
4.17	3.875	3.4	2.879	-5.795	4.193	0.2035	8
4.171	3.876	3.4	2.879	-5.805	4.198	0.2039	9
4.172	3.876	3.401	2.88	-5.81	4.2	0.2041	10
4.172	3.877	3.401	2.88	-5.814	4.202	0.2042	11
4.173	3.877	3.401	2.88	-5.815	4.203	0.2042	12
4.173	3.877	3.401	2.88	-5.817	4.203	0.2043	13
4.173	3.877	3.401	2.88	-5.817	4.204	0.2043	14
4.173	3.877	3.401	2.88	-5.818	4.204	0.2043	15
4.173	3.877	3.401	2.88	-5.818	4.204	0.2043	16
4.173	3.877	3.401	2.88	-5.818	4.204	0.2043	17
4.173	3.877	3.401	2.88	-5.818	4.204	0.2043	18
4.173	3.877	3.401	2.88	-5.818	4.204	0.2043	19
4.173	3.877	3.401	2.88	-5.818	4.204	0.2043	20
4.173	3.877	3.401	2.88	-5.818	4.204	0.2043	21
4.173	3.877	3.401	2.88	-5.818	4.204	0.2043	22
NA	NA	NA	NA	NA	NA	NA	NA



1.4 EM Method

- E-step

$$Q(\vec{\theta},\vec{\theta}^{\star}) = -\frac{n}{2}\ln(2\pi) - n\ln(\sigma) - \frac{1}{2\sigma^2}\sum_{j=1}^{m}(t_j-\beta_0-\beta_1\nu_j)^2 - \frac{1}{2\sigma^2}\sum_{i=m+1}^{n}E[(T_i-\beta_0-\beta_1\nu_i)^2|T_i>w_i,\vec{\theta}^{\star}]$$

$$E[T_i|T_i>w_i,\vec{\theta}^{\star}] = \mu_i^{\star} + \sigma^{\star}H(\frac{w_i-\mu_i^{\star}}{\sigma^{\star}})$$

$$E[T_i^2|T_i>w_i,\vec{\theta}^{\star}] = \mu_i^{\star 2} + \sigma^{\star 2} + \sigma^{\star}(w_i+\mu_i^{\star})H(\frac{w_i-\mu_i^{\star}}{\sigma^{\star}})$$

$$E[(T_i-\beta_0-\beta_1\nu_i)^2|T_i>w_i,\vec{\theta}^{\star}] = \mu_i^{\star 2}+\sigma^{\star 2}+\sigma^{\star}(w_i+\mu_i^{\star})H(\frac{w_i-\mu_i^{\star}}{\sigma^{\star}})-2(\beta_0+\beta_1\nu_i)[\mu_i^{\star}+\sigma^{\star}H(\frac{w_i-\mu_i^{\star}}{\sigma^{\star}})]+(\beta_0+\beta_1\nu_i)^2$$

- M-step

$$\frac{\partial Q}{\partial \beta_0} = -\frac{1}{\sigma^2}\left\{\sum_{j=1}^{m}[t_j-\beta_0-\beta_1\nu_j] + \sum_{i=m+1}^{n}[\mu_i^{\star}+\sigma^{\star}H(\frac{w_i-\mu_i^{\star}}{\sigma^{\star}})-\beta_0-\beta_1\nu_i]\right\} = 0$$

$$\frac{\partial Q}{\partial \beta_1} = -\frac{1}{\sigma^2}\left\{\sum_{j=1}^{m}[t_j-\beta_0-\beta_1\nu_j]\nu_j + \sum_{i=m+1}^{n}[\mu_i^{\star}+\sigma^{\star}H(\frac{w_i-\mu_i^{\star}}{\sigma^{\star}})-\beta_0-\beta_1\nu_i]\nu_i\right\} = 0$$

$$\frac{\partial Q}{\partial \sigma^2} = \frac{1}{2\sigma^2}\left\{-n + \frac{1}{\sigma^2}\sum_{j=1}^{m}[t_j - \beta_0 - \beta_1\nu_j]^2 + \frac{1}{\sigma^2}\sum_{i=m+1}^{n} E[(T_i - \beta_0 - \beta_1\nu_i)^2|T_i > w_i, \vec{\theta}^\star]\right\} = 0$$

$$\sum_{j=1}^{m}[t_j - \beta_0 - \beta_1\nu_j]^2 + \sum_{i=m+1}^{n}\left\{\mu_i^{\star 2} + \sigma^{\star 2} + \sigma^\star(w_i + \mu_i^\star - 2\mu_i)H(\frac{w_i - \mu_i^\star}{\sigma^\star}) - 2\mu_i\mu_i^\star + \mu_i^2\right\} = n\sigma^2$$

1.5 EM Code

```
temp <- c(150,170,190,220) #temperature levels
trec <- 1000/(temp+273.2) #reciprocal of the absolute temperature T
nu <- c(rep(trec[1],10),rep(trec[2],10),rep(trec[3],10),rep(trec[4],10))
index_nu <- c(rep(-2,7),rep(-3,5),rep(-4,5),rep(1,10),rep(2,3),rep(3,5),rep(4,5))
n<-length(nu); m <- length(nu_uncensored)
cen <- c(8064,5448,1680,528) #censoring times
w <- log10(cen) #log10 censoring times: last time still working
y_uncensored <-log10(c(rep(1,10),1764,2772,3444,3542,3780,4860,5196,rep(1,3),408,408,1344,1344,1440,rep
y_censored <- c(rep(w[1],10),rep(0,7),rep(w[2],3),rep(0,5),rep(w[3],5),rep(0,5),rep(w[4],5))
# which(index_x %in% 1)
S <- 23; Y<-matrix(nrow=S,ncol=40)
Y[1,] <- y_0 <- (y_uncensored+y_censored)
index <- which(y_0 %in% w)

fit0 <- lm(y_0~nu) #linear model
sigma_0 <- sigma(fit0) #standard error of residuals
beta_00 <- coef(fit0)[1] #intercept
beta_10 <- coef(fit0)[2] #slope
mu_0 <- beta_00 + beta_10*trec #mean log time to failure
z <- (w-mu_0)/sigma_0 #z-vector
H <- dnorm(z)/(1-pnorm(z))
Test <- mu_0 + sigma_0*H #new expected mean log times to failure

delta = 1e-006; iteration <- 1
THETA<-matrix(nrow=S,ncol=8,dimnames=list(NULL, c('mu150','mu170','mu190','mu220','Intercept','Slope','
THETA[1,]<-theta<-c(Test, beta_00, beta_10,sigma_0,iteration)
# Subsequent iteration
repeat {
 theta[8] <- theta[8]+1
 y_censored <- c(rep(theta[1],10),rep(theta[2],3),rep(theta[3],5),rep(theta[4],5))
 y<- y_uncensored+y_censored
 Y[theta[8],]<-y   # Replace the new censored values
 fit <- lm(y~trec)    # fit a new model
 theta[5] <- coef(fit)[1] #intercept
 theta[6] <- coef(fit)[2] #slope
 theta[7] <- sigma(fit) #standard error of residuals

 mu <- theta[5] + theta[6]*trec
 z <- (w-mu)/theta[7] #z-vector
 theta[1:4] <- mu + theta[7]*dnorm(z)/(1-pnorm(z)) #new expected mean log times to failure
 conv <- dist(rbind(THETA[theta[8]-1,1:4],theta[1:4]))
 if(conv < delta) break
```

```
  THETA[theta[8],]<-theta
}
```

2 The EM algorithm

2.1 Introduction

For many models in machine learning and statistics, computing the ML or MAP parameter estimate is easy provided we observe all the values of all the relevant random variables, i.e., if we have complete data. However, if we have missing data and/or latent variables, then computing the ML/MAP estimate becomes hard.

One approach is to use a generic gradient-based optimizer to find a local minimum of the NLL$(\vec{\theta})$. However, we often have to enforce constraints, such as the fact that covariance matrices must be positive definite, mixing weights must sum to one, etc., which can be tricky. In such cases, it is often much simpler (but not always faster) to use an algorithm called **expectation maximization**,or **EM** for short (Dempster et al. 1977; Meng and van Dyk 1997; McLachlan and Krishnan 1997). This is is an efficient iterative algorithm to compute the ML or MAP estimate in the presence of missing or hidden data, often with closed-form updates at each step. Furthermore, the algorithm automatically enforce the required constraints.

2.2 Basic idea

EM exploits the fact that if the data were fully observed, then the ML/ MAP estimate would be easy to compute. In particular, each iteration of the EM algorithm consists of two processes: The E-step, and the M-step.

- In the **E-step**, the missing data are inferred given the observed data and current estimate of the model parameters. This is achieved using the conditional expectation, explaining the choice of terminology.
- In the **M-step**, the likelihood function is maximized under the assumption that the missing data are known. The missing data inferred from the E-step are used in lieu of the actual missing data.

Let $\vec{x}_i$ be the visible or observed variables in case i, and let $\vec{z}_i$ be the hidden or missing variables. The goal is to maximize the log likelihood of the observed data:

$$\ell(\vec{\theta}) = \log p(\mathcal{D}|\vec{\theta}) = \sum_{i=1}^{N} \log p(\vec{x}_i|\vec{\theta}) = \sum_{i=1}^{N} \log \sum_{\vec{z}_i} p(\vec{x}_i, \vec{z}_i|\vec{\theta}) \tag{1}$$

Unfortunately this is hard to optimize, since the log cannot be pushed inside the sum.

EM gets around this problem as follows. Define the **complete data log likelihood** to be

$$\ell_c(\vec{\theta}) = \sum_{i=1}^{N} \log p(\vec{x}_i, \vec{z}_i|\vec{\theta}) \tag{2}$$

This cannot be computed, since $\vec{z}_i$ is unknown. So let us define the **expected complete data log likelihood** as follows:

$$Q(\vec{\theta}, \vec{\theta}^{t-1}) \triangleq \mathbb{E}_{\vec{z}|\mathcal{D},\theta^{t-1}} \left[\ell_c(\vec{\theta})\right] = \mathbb{E}\left[\ell_c(\vec{\theta})|\mathcal{D}, \theta^{t-1}\right] \tag{3}$$

where t is the current iteration number. Q is called the **auxiliary function**(see Section **??** for derivation). The expectation is taken wrt the old parameters, $\vec{\theta}^{t-1}$, and the observed data $\mathcal{D}$. The goal of the E-step is to compute $Q(\vec{\theta}, \vec{\theta}^{t-1})$, or rather, the parameters inside of it which the MLE(or MAP) depends on; these are known as the **expected sufficient statistics** or **ESS**. In the M-step, we optimize the Q function wrt $\vec{\theta}$:

$$\vec{\theta}^t = \arg\max_{\vec{\theta}} Q(\vec{\theta}, \vec{\theta}^{t-1}) \tag{4}$$

To perform MAP estimation, we modify the M-step as follows:

$$\vec{\theta}^t = \arg\max_{\vec{\theta}} Q(\vec{\theta}, \vec{\theta}^{t-1}) + \log p(\vec{\theta}) \tag{5}$$

The E step remains unchanged.

In summary, the EM algorithm's pseudo code is as follows

Algorithm 1: EM algorithm

```
input  : observed data $\mathcal{D} = \{\vec{x}_1, \vec{x}_2, \cdots, \vec{x}_N\}$,joint distribution $P(\vec{x}, \vec{z}|\vec{\theta})$
output: model's parameters $\vec{\theta}$
// 1. identify hidden variables $\vec{z}$, write out the log likelihood function $\ell(\vec{x}, \vec{z}|\vec{\theta})$
$\vec{\theta}^{(0)}$ = ... // initialize
while (!convergency) do
    // 2. E-step: plug in $P(\vec{x}, \vec{z}|\vec{\theta})$, derive the formula of $Q(\vec{\theta}, \vec{\theta}^{t-1})$
    $Q(\vec{\theta}, \vec{\theta}^{t-1}) = \mathbb{E}\left[\ell_c(\vec{\theta})|\mathcal{D}, \theta^{t-1}\right]$
    // 3. M-step: find $\vec{\theta}$ that maximizes the value of $Q(\vec{\theta}, \vec{\theta}^{t-1})$
    $\vec{\theta}^t = \arg\max_{\vec{\theta}} Q(\vec{\theta}, \vec{\theta}^{t-1})$
```