Iterative Least Squares and EM Algorithm Approaches to Right Censored Linear Regression

Charlie Qu, April 15, 2016

1. Introduction

Schmee and Hahn introduced Iterative Least squares method as an alternative method to estimate parameters from a right censored dataset. The iterative least squares method that Schmee and Hahn proposed is easy to implement, as well as easy to explain to people who doesn't have strong statistical analysis background. The iterative least squares method also takes care of the problem of underestimating right-censored values from Maximum likelihood method. In our study, we will implement Schmee and Hahn's iterative least squares method in R to estimate the parameters of the motorettes example that was originally reported by Crawford. As well as using E-M algorithm to estimate parameters of the same example. We will conduct a comparison between the two methods.

2. Iterative Least Squares Method

Summary of iterative least squares method:

1. Convert failure hours to its $\log_{10}$ equivalent.
   Convert temperature to x = 1000/(T+273.2)
2. Iteration 0 step 1
   Fit a simple linear regression line with the motorette data treating censored data as if they fail at the censoring time to obtain:$\hat{\beta}_0^{(0)}$, $\hat{\beta}_1^{(0)}$ and $\hat{\sigma}^{(0)}$.
   Iteration 0 step 1
   Use the estimates obtained from last step $\hat{\beta}_0^{(0)}$, $\hat{\beta}_1^{(0)}$ and $\hat{\sigma}^{(0)}$ to estimate $\hat{\mu}_x^{*(0)}$ using the below equation:

   $$\hat{\mu}_x^* = \mu_x + \frac{\sigma f(z)}{[1-F(z)]} \quad \text{where } z = \frac{c_x - \mu}{\sigma} \text{ and } c_x \text{ denote the censoring time.}$$

3. Iteration 1 step 1

   Using $\hat{\mu}_x^{*(0)}$ as censored time to obtain a revised least squares regression line, and denote the new parameters as $\hat{\beta}_0^{(1)}$, $\hat{\beta}_1^{(1)}$ and $\hat{\sigma}^{(1)}$.
   Iteration 1 step 2
   Use the estimates $\hat{\beta}_0^{(1)}$, $\hat{\beta}_1^{(1)}$ and $\hat{\sigma}^{(1)}$ obtained from last step to estimate $\hat{\mu}_x^{*(1)}$ similar as what we did in iteration 0 step 1
4. Subsequent Iterations
   Continue the two steps in iteration 1 until the estimate converges.

Below is the R output from the:

Iteration	150℃	170℃	190℃	220℃	$\hat{\beta}_0$	$\hat{\beta}_1$	$\hat{\sigma}$
0	4.038379	3.808887	3.329470	2.829844	-4.9305	3.7470	0.1572
1	4.098339	3.839577	3.365457	2.858389	-5.2564	3.9246	0.1796
2	4.130353	3.85579	3.38185	2.8692	-5.481	4.038	0.1907863
3	4.148261	3.86467	3.3902	2.87412	-5.618	4.106	0.1965943
4	4.158478	3.86967	3.39473	2.87661	-5.699	4.146	0.1997713
10	4.172007	3.87623	3.4005	2.87963	-5.808	4.199	0.2038563
15	4.172528	3.87649	3.40072	2.87975	-5.812	4.202	0.2040121
17	4.172553	3.876498	3.400730	2.879751	-5.812	4.202	0.2040197
20	4.172564	3.876503	3.400735	2.879753	-5.813	4.202	0.2040228

Table 1. motorettes example.

We set our tolerance at 10^{-6} and ILS method reaches convergence at its 21st iteration (includes iteration 1 step and iteration 1 step).
Our finial parameter estimates are $\hat{\beta}_0 = -5.813$, $\hat{\beta}_1 = 4.202$, $\hat{\sigma} = 0.2040228$, and the finial expected time of failure for the censored observations are:

150℃: 4.172564 log hours (14878.67hrs)

170℃: 3.876503 log hours (7524.939hrs)

190℃: 3.400735 log hours (2516.141hrs)

220℃: 2.879753 log hours (758.1463hrs)

We conduct our analysis based on the following graphs.

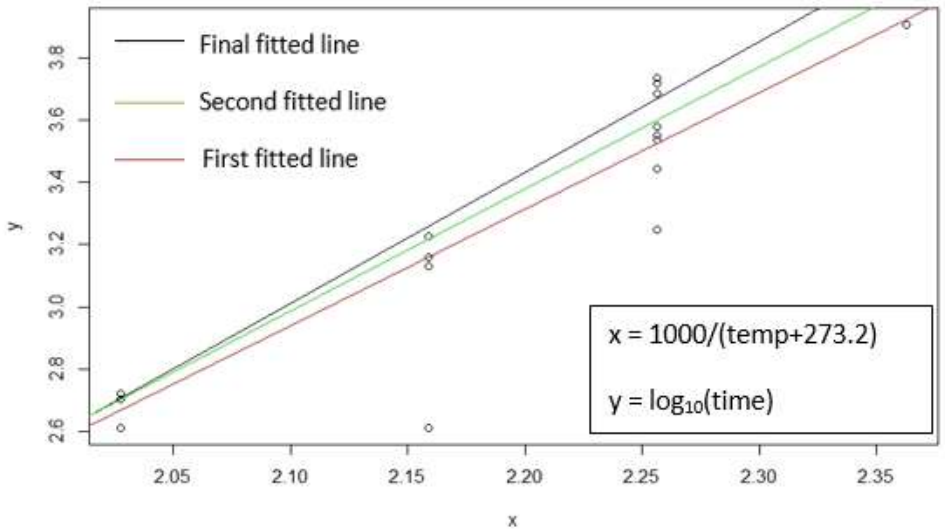

Figure 1. The figure above shows us the three regression lines from top to bottom in the order of Final fitted line, second fitted line (from iteration 1) and the First fitted line (from iteration 0):

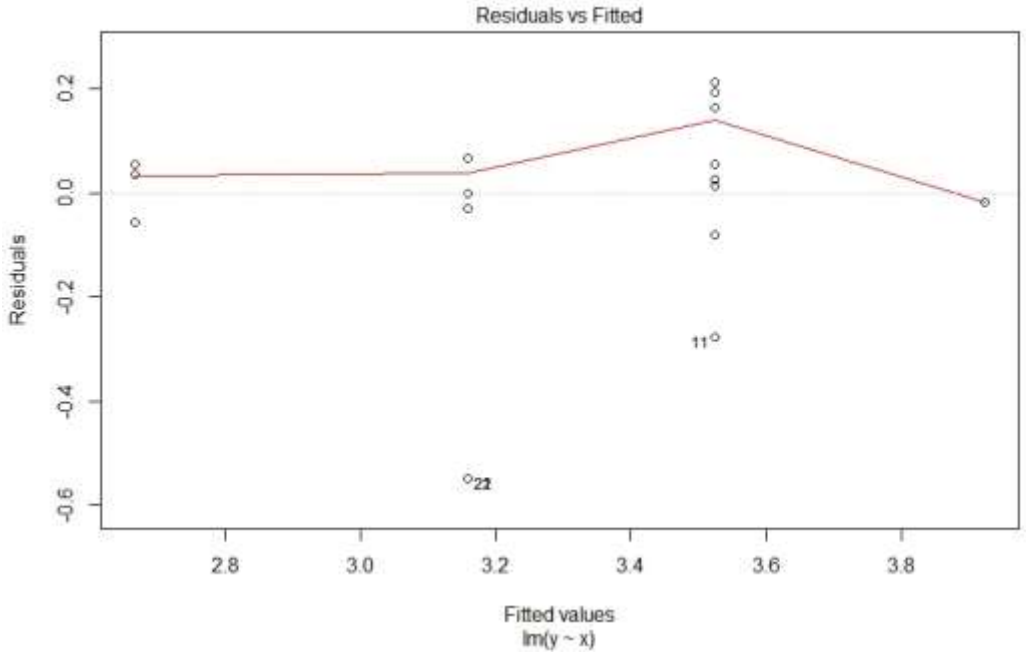

Figure 2. Residual plot from iteration 0.

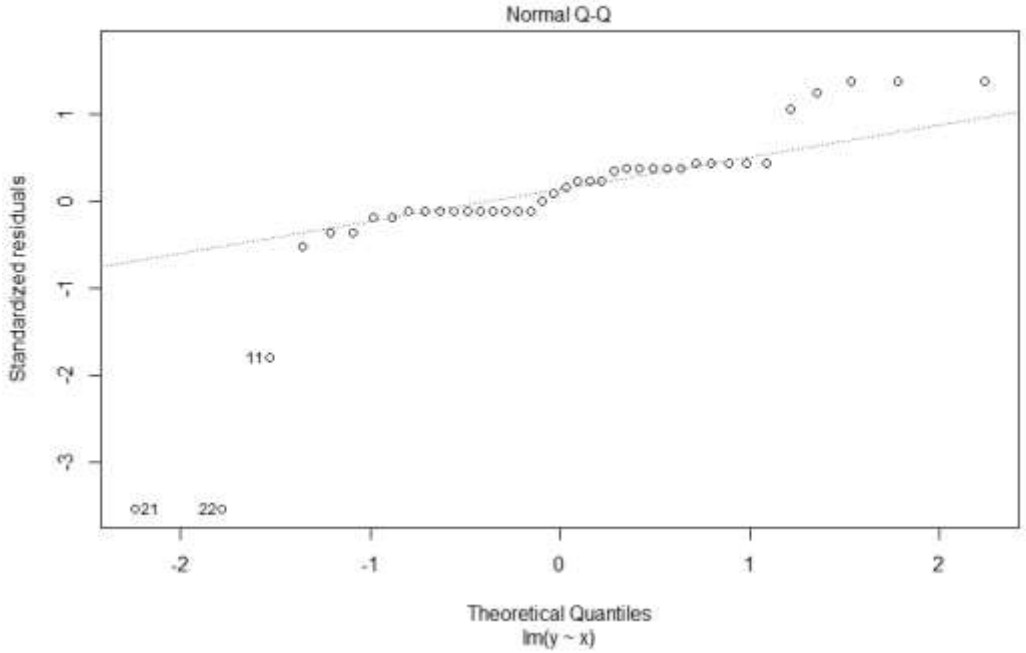

Figure 3. Iteration 0 QQ plot.

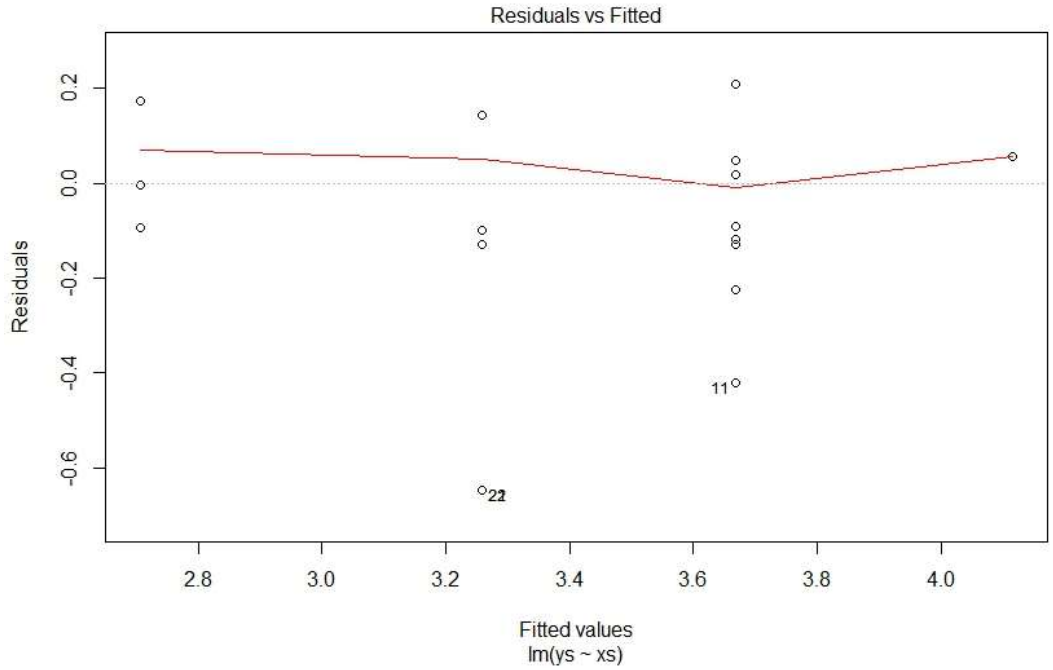

Figure 4. Final residual plot.

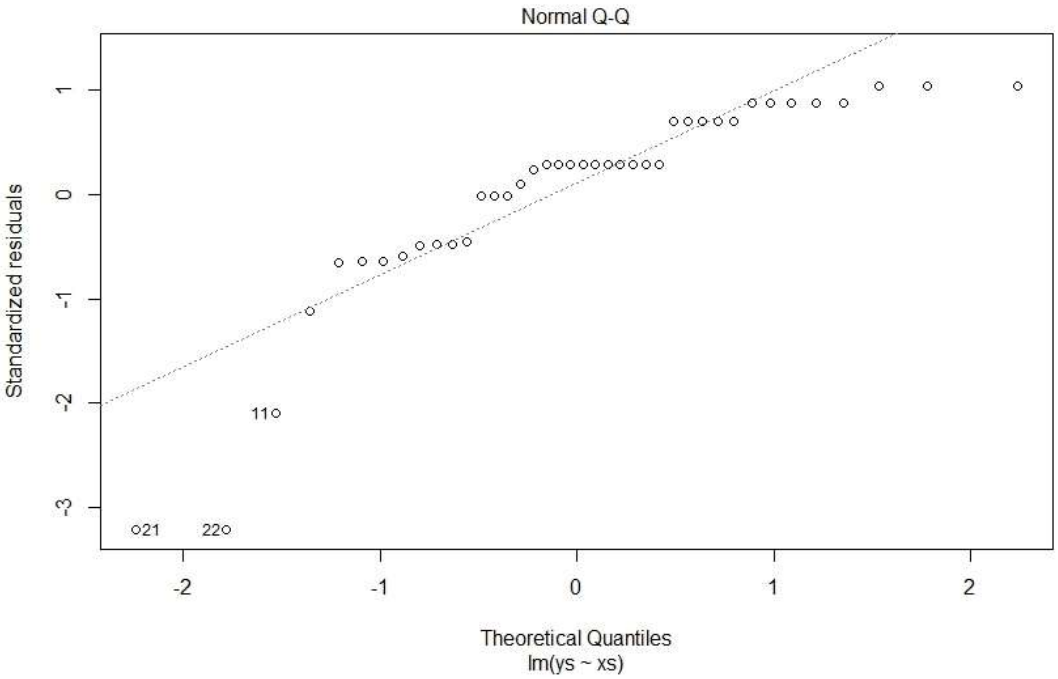

Figure 5. Final QQ plot.

3. EM algorithm method

3.1. Introduction

The expectation–maximization (EM) algorithm is an iterative method for finding maximum likelihood or maximum a posteriori (MAP) estimates of parameters in statistical models, where the model depends on unobserved latent variables. This section applies E-M algorithm to fit the linear regression model with right-censored data. By dropping the explicit likelihood function and simply treats the censored observations as if they were missing data, the algorithm alternated between estimating the missing data (E-step) and maximizing the log likelihood(M-step). There the lifetime of the motorettes that end accidentally on test or that still keep alive when the experiment was terminated is considered to be missing observations. The linear regression model with well estimated parameters by this algorithm, and a simple generalization are given finally, as a part of the final comparison with the previous ILS.

3.2. Deriviaion and procedure

The table gives the results of temperature accelerated life tests on electrical insulation in 40 motorettes, originally reported by Crawford [1].

Test Temperature			
150 °C ①	170 °C ②	190 °C ③	220 °C ④
8064	1764	408	408
8064	2772	408	408
8064	3444	1344	504
8064	3542	1344	504
8064	3780	1440	504
8064	4860	***1680***	***528***
8064	5196	***1680***	***528***
8064	***5448***	***1680***	***528***
8064	***5448***	***1680***	***528***
8064	***5448***	***1680***	***528***

Table 2. Temperature accelerated life tests.

- All 10 motorettes at 150 °C still on test without failure at 8064 hours.
- 3 motorettes at 170 °C still on test without failure at 5448 hours.
- 5 motorettes at 190 °C still on test without failure at 1680 hours.
- 5 motorettes at 220 °C still on test without failure at 528 hours.

Note that **italic data are censored, or more exactly, right censored.*

Ten motorettes were tested at each of four temperatures. Testing was terminated at different times at each temperature, resulting in a total of 17 failed units and 23 unfailed ones. The model used to analyze the data assumes that:

(i) for any temperature, the time to failure is lognormal distributed.

(ii) The standard deviation σ of the lognormal time to failure distribution is constant;

(iii) the mean of the logarithm of the time to failure a, is a linear function of the reciprocal x = 1000/(C + 273.2) of the absolute temperature C, that is

$$\mu_x = \beta_0 + \beta_1 x$$

where, β_0, β_1, and σ are unknown parameters.

(1) E-step: Determine the conditional expectation.

Denote the ten values (observed and censored) of at each temperature as T_i, $i = 1,2,\dots,10$, where $Y = (T_1, T_2, \dots, T_m)$ is the observed data, and $Z = (T_{m+1}, \dots, T_{10})$. Let $\omega_i = \min(T_i, c_i)$, where c_i is the censoring failure time. Since $T_i = \beta_0 + \beta_1 x_i + \sigma\varepsilon_i$, where is random error $\varepsilon_i \sim N(0,1)$. Therefore, $T_i \sim N(\beta_0 + \beta_1 x_i, \sigma^2)$. To estimate the parameters β_0, β_1, and σ^2, firstly find out the log likelihood equation

$$log\big(L(\beta_0, \beta_1, \sigma^2|y, z)\big) = -\frac{n}{2} log(2\pi) - n log\sigma - \frac{1}{2\sigma^2}\sum_{j=1}^{m}(t_j - \beta_0 - \beta_1 x_j)^2$$

By E-step. [1]

$$Q(\theta, \theta^*) = -\frac{n}{2} log(2\pi) - n log\sigma - \frac{1}{2\sigma^2}\sum_{j=1}^{m}(t_j - \beta_0 - \beta_1 x_j)^2 - \frac{1}{2\sigma^2}\sum_{i=m+1}^{n}\left\{\left[\mu_i^{*2} + \sigma^{*2} + \sigma^*(\omega_i + \mu_i^*)H\left(\frac{\omega_i - \mu_i^*}{\sigma^*}\right)\right] - 2(\beta_0 + \beta_1 x_i)\left(\mu_i^* + \sigma^* H\left(\frac{\omega_i - \mu_i^*}{\sigma^*}\right)\right) + \left(\mu_i^* + \sigma^* H\left(\frac{\omega_i - \mu_i^*}{\sigma^*}\right)\right)^2\right\} \quad (1)$$

(2) M-step: Maximize the expression with expect to β_0, β_1, σ^2, respectively [2]

The updating equation for β_0, β_1, σ^2, respectively

$$\widehat{\beta_0} = \frac{\sum_{j=1}^{m}(t_j - \beta_1 x_j) + \sum_{i=m+1}^{n}\left[\mu_i^{*2} + \sigma^* H\left(\frac{\omega_j - \mu_i^*}{\sigma^*}\right) - \beta_1 x_i\right]}{n} \quad (2)$$

$$\widehat{\beta_1} = \frac{\sum_{j=1}^{m}(x_j t_j - 2\beta_0 x_j) + \sum_{i=m+1}^{n} x_i\left(\mu_i^* + \sigma^* H\left(\frac{\omega_i - \mu_i^*}{\sigma^*}\right)\right)}{2\sum_{j=1}^{m} x_j^2} \quad (3)$$

$$\widehat{\sigma^2} = \frac{\sum_{j=1}^{m}(t_j - \beta_0 - \beta_1 x_j)^2 + \sum_{i=m+1}^{n}[\mu_i^{*2} + \sigma^{*2} + \sigma^* H\left(\frac{\omega_j - \mu_i^*}{\sigma^*}\right)(\omega_i + \mu_i^* - 2\mu_i) - 2\mu_i\mu_i^* + \mu_i^2]}{n} \quad (4)$$

3.3. Comments on EM algorithm for this case

Recall that the updated estimate of the parameter, which maximizes the $Q(\theta^{(i+1)}, \theta^{(i)})$. Starting with the current estimate for θ^*, that is $Q(\theta^{(i+1)}, \theta^{(i+1)})=0$. Because θ is chosen to maximize $Q(\theta^{i+1}, \theta^i)$, then

$Q(\theta^{i+1}, \theta^{i}) \geq Q(\theta^{(i)}, \theta^{(i)}) = 0$, so for each iteration the likelihood is non-decreasing. When the algorithm reaches a fixed point for some $\theta^{(i)}$, which maximizes

the likelihood L. Since L and l are equal at $\theta^{(i)}$ if L and l are differentiable at $\theta^{(i)}$, then

$\theta^{(i)}$ must be a stationary point of L. The stationary point need not, however, be a local maximum.

Note: Right censoring – a data point is above a certain value but it is unknown by how much.

Type I censoring occurs if an experiment has a set number of subjects or items and stops the experiment at a predetermined time, at which point any subjects remaining are right-censored.

Type II censoring occurs if an experiment has a set number of subjects or items and stops the experiment when a predetermined number are observed to have failed; the remaining subjects are then right-censored.

3.4. R output from the motorettes example

Iterations	$\hat{\beta}_0$	$\hat{\beta}_1$	$\hat{\sigma}$
0	-4.93051	3.747043	0.157218
1	-5.26009	3.926205	0.199751
2	-5.51117	4.055823	0.217266
3	-5.67844	4.139847	0.228832
4	-5.78537	4.193345	0.237118
10	-5.98668	4.29473	0.255773
20	-6.01775	4.310485	0.259024
30	-6.01918	4.311211	0.259175
41	-6.01925	4.311246	0.259183

Table 3. Motorettes example.

We set our tolerance at 10^{-6} and EM method reaches convergence at its 42st iteration.

Our finial parameter estimates are $\hat{\beta}_0 = -6.01925$, $\hat{\beta}_1 = 4.311246$, $\hat{\sigma} = 0.259183$

Conclusion: Comparing the output from ILS and EM, they perform similarly for parameter estimation in this right censoring normal regression data set. But apparently, ILS converges faster that EM algorithm. ILS is easier to use than EM algorism since the E and M steps are complicated in EM algorithm.

4. Further discussion

The EM algorithm efficiently compute the maximum likelihood(ML) by iterative procedure in the presence of missing data. In ML estimation, we wish to estimatee the model parameters for which the observed data are the most likely. Theoretically speaking, because heavily censored observations contribute little information, which can be ignored necessarily, it can be predicted that the EM algorithm of the variance will be badly biased when the number of uncensored observations is small. Actually, the EM algorithm produces estimates which are an order of magnitude greater in bias than both ML and ILS.

On the contrary, the ILS will badly biased if the censoring is very heavy.

Reference

[1] CRAWFORD, D.E. (1970). Analysis of incomplete life test data on motorettes. Insulation/Circuits 16, 10, 43-48

[2] JOSEF SCHMEE and GERALD J. HAHN. A simple method for regression analysis with censored data. Technometrics, Vol. 21, No.4 (Nov., 1979), pp. 417-432

[3] MURRAY AITKIN. A note on the regression analysis of censored data. Technometrics, Vol. 23, No. 2 (May,1981), pp. 161-163

Appendix. Mathematical Inference.

[1]

$$Q(\theta,\theta^*) = E\left[-\frac{n}{2}\,log(2\pi) - n\,log\sigma\; - \frac{1}{2\sigma^2}\sum_{j=1}^{m}\left(t_j-\beta_0-\beta_1 x_j\right)^2 - \frac{1}{2\sigma^2}\sum_{i=m+1}^{n}(t_i-\beta_0-\beta_1 x_i)^2\,|_{t_1,\dots t_m,\omega_{m+1},\dots,t_n,\theta^*}\right]$$

$$= -\frac{n}{2}log(2\pi) - n\,log\sigma - \frac{1}{2\sigma^2}\sum_{j=1}^{m}\left(t_j-\beta_0-\beta_1 x_j\right)^2 - \frac{1}{2\sigma^2}\sum_{i=m+1}^{n}E[(t_i-\beta_0-\beta_1 x_i)^2 \mid t_i>\omega_i,\theta^*]$$

For the last part

$$E[(t_i-\beta_0-\beta_1 x_i)^2 \mid t_i>\omega_i,\theta^*]$$

$$= E[{t_i}^2 \mid t_i>\omega_i,\theta^*] - 2E[t_i(\beta_0+\beta_1 x_i) \mid t_i>\omega_i,\theta^*] + E[(\beta_0+\beta_1 x_i)^2 \mid t_i>\omega_i,\theta^*]$$

Where,

$$E[t_i \mid t_i>\omega_i,\theta^*] = \mu_i^* + \sigma^* H\left(\frac{\omega_i-\mu_i^*}{\sigma^*}\right), \mu_i^* = \beta_0^* + \beta_1^* x_j, H\left(\frac{\omega_j-\mu_i^*}{\sigma^*}\right) = \frac{\Phi(\frac{\omega_i-\mu_i^*}{\sigma^*})}{1-\Phi(\frac{\omega_i-\mu_i^*}{\sigma^*})}$$

$$E[{t_i}^2 \mid t_i>\omega_i,\theta^*] = {\mu_i^*}^2 + {\sigma^*}^2 + \sigma^*(\omega_i+\mu_i^*)H\left(\frac{\omega_i-\mu_i^*}{\sigma^*}\right)$$

Therefore

$$Q(\theta,\theta^*) = -\frac{n}{2}\,log(2\pi) - n\,log\sigma\; - \frac{1}{2\sigma^2}\sum_{j=1}^{m}(t_j-\beta_0-\beta_1 x_j)^2 - \frac{1}{2\sigma^2}\sum_{i=m+1}^{n}\{[{\mu_i^*}^2 + {\sigma^*}^2 + \sigma^*(\omega_i+\mu_i^*)\,H(\tfrac{\omega_i-\mu_i^*}{\sigma^*})] - 2(\beta_0+\beta_1 x_i)(\mu_i^* + \sigma^* H(\tfrac{\omega_i-\mu_i^*}{\sigma^*})) + (\mu_i^* + \sigma^* H(\tfrac{\omega_i-\mu_i^*}{\sigma^*}))^2\} \quad (1)$$

[2]

Maximize the log likelihood by setting the partial derivatives as 0 and find the updating equation for the parameters

$$\frac{\partial Q}{\partial \beta_0} = \sum_{j=1}^{m}\left(t_j-\beta_0-\beta_1 x_j\right) - \sum_{i=m+1}^{n}\left(\mu_i^* + \sigma^* H\left(\frac{\omega_i-\mu_i^*}{\sigma^*}\right) - \beta_0 - \beta_1 x_i\right) = 0$$

$$\frac{\partial Q}{\partial \beta_1} = \sum_{j=1}^{m}(x_j t_j - 2\beta_0 x_j - 2\beta_1 {x_j}^2) + \sum_{i=m+1}^{n} x_i \left(\mu_i^* + \sigma^* H\left(\frac{\omega_{i-}\mu_i^*}{\sigma^*}\right)\right) = 0$$

$$\frac{\partial \mathrm{Q}}{\partial \sigma^2} = \sum_{\mathrm{j}=1}^{\mathrm{m}}(\mathrm{t_j} - \beta_0 - \beta_1 \mathrm{x_j})^2 - \sum_{\mathrm{i}=\mathrm{m}+1}^{\mathrm{n}} \mathrm{x_i}(\mu_\mathrm{i}^* + \sigma^* \mathrm{H}\left(\frac{\omega_{\mathrm{i}-}\mu_\mathrm{i}^*}{\sigma^*}\right) - (\beta_0 + \beta_1 \mathrm{x_i})\mathrm{x_i}) = 0$$

[3] **R code and R output:**

```
###########Iterative least squares example:#############

data<-read.csv("c:/Users/Shayne/Desktop/Project661/Moterettesdata.csv",header=T)

y<-log(data$time,10)    #use log base 10 to match witht the paper

x<-1000/(data$temp+273.2)

fit0<-lm(y~x)    #iteration 0 step 1

m0<-summary(fit0)

sigma0hat<-m0$sigma


#iteration 0 step 2

mu0hat<-fit0$coefficients[1]+fit0$coefficients[2]*x

z=(y-mu0hat)/sigma0hat

mu0hatstar<-mu0hat+sigma0hat*dnorm(z)/(1-pnorm(z))

resulti0<-mu0hatstar*data$censorind

eceft00<-c(resulti0[1],resulti0[18],resulti0[26],resulti0[36]) #"estmate of expected failure time


#iteration 1 step 1

newdata<-cbind(data$time*abs(data$censorind-1)+10^(resulti0),data$temp,data$censorind)

newdata<-data.frame(newdata)

colnames(newdata)<-c("time","temp","censorind")

y1<-log10(newdata$time)
```

```
x1<-1000/(newdata$temp+273.2)
fit1<-lm(y1~x1)          #iteration 1 step 1
m1<-summary(fit1)
sigma1hat<-m1$sigma

#iteration 1 step 2
beta01hat<-fit1$coefficients[1]
beta11hat<-fit1$coefficients[2]
beta0hat<-beta01hat
beta1hat<-beta11hat
mu1hat<-beta0hat+beta1hat*x
z1=(y-mu1hat)/sigma1hat
mu1hatstar<-mu1hat+sigma1hat*dnorm(z1)/(1-pnorm(z1))
resulti1<-mu1hatstar*data$censorind
eceft1<-c(resulti1[1],resulti1[18],resulti1[26],resulti1[36]) #"estmate of expected failure time

#subsequent iterations
e=1
i=0
resultis<-resulti1
while (e>10^-6){
newdata<-cbind(data$time*abs(data$censorind-1)+10^(resultis),data$temp,data$censorind)
newdata<-data.frame(newdata)
colnames(newdata)<-c("time","temp","censorind")
ys<-log10(newdata$time)
xs<-1000/(newdata$temp+273.2)
```

```
fits<-lm(ys~xs) #iteration 1 step 1
ms<-summary(fits)
sigmashat<-ms$sigma
beta0hat<-fits$coefficients[1]
beta1hat<-fits$coefficients[2]
mushat<-beta0hat+beta1hat*x
zs=(y-mushat)/sigmashat
mushatstar<-mushat+sigmashat*dnorm(zs)/(1-pnorm(zs))
resultis<-mushatstar*data$censorind
ecefts<-c(resultis[1],resultis[18],resultis[26],resultis[36])
i=i+1
print(i)
print(ecefts)
print(fits)
print(sigmashat)
e<-max(abs(ecefts-eceft0))
eceft0<-ecefts
}
plot(x,y)
abline(fits)
abline(fit0,col="red")
abline(fit1,col="green")
plot(data$temp,data$time*data$censorind,ylim=c(10,100000))
```

```
> summary(fit0)

Call:
lm(formula = y ~ x)
```

```
Residuals:
    Min      1Q  Median      3Q     Max
-0.54830 -0.01701  0.01914  0.05838  0.21223

Coefficients:
            Estimate Std. Error t value
(Intercept)  -4.9305     0.4433  -11.12
x             3.7470     0.2011   18.64
            Pr(>|t|)
(Intercept) 1.64e-13 ***
x            < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.1572 on 38 degrees of freedom
Multiple R-squared:  0.9014,        Adjusted R-squared:  0.8988
F-statistic: 347.3 on 1 and 38 DF,  p-value: < 2.2e-16
```

```
> summary(fit1)

Call:
lm(formula = y1 ~ x1)

Residuals:
    Min      1Q  Median      3Q     Max
-0.60474 -0.05153  0.02109  0.11294  0.21008

Coefficients:
            Estimate Std. Error t value
(Intercept)  -5.2564     0.5065  -10.38
x1            3.9246     0.2297   17.08
            Pr(>|t|)
(Intercept) 1.21e-12 ***
x1           < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.1796 on 38 degrees of freedom
Multiple R-squared:  0.8848,        Adjusted R-squared:  0.8818
F-statistic: 291.8 on 1 and 38 DF,  p-value: < 2.2e-16
```

Subsequent interations:

```
[1] 1
[1] 4.130353 3.855788 3.381853 2.869201

Call:
lm(formula = ys ~ xs)

Coefficients:
(Intercept)           xs
     -5.481        4.038
```

```
[1] 0.1907863
[1] 2
[1] 4.148261 3.864666 3.390204 2.874116

Call:
lm(formula = ys ~ xs)

Coefficients:
(Intercept)          xs
     -5.618       4.106

[1] 0.1965943
[1] 3
[1] 4.158478 3.869665 3.394729 2.876607

Call:
lm(formula = ys ~ xs)

Coefficients:
(Intercept)          xs
     -5.699       4.146

[1] 0.1997713
[1] 4
[1] 4.164367 3.872529 3.397268 2.877956

Call:
lm(formula = ys ~ xs)

Coefficients:
(Intercept)          xs
     -5.746       4.169

[1] 0.2015635
[1] 5
[1] 4.167783 3.874186 3.398720 2.878714

Call:
lm(formula = ys ~ xs)

Coefficients:
(Intercept)          xs
     -5.774       4.183

[1] 0.2025921
[1] 6
[1] 4.169771 3.875149 3.399560 2.879148

Call:
lm(formula = ys ~ xs)

Coefficients:
(Intercept)          xs
     -5.790       4.191

[1] 0.203188
```

```
[1] 7
[1] 4.170932 3.875712 3.400049 2.879400

Call:
lm(formula = ys ~ xs)

Coefficients:
(Intercept)           xs
     -5.799        4.195

[1] 0.2035351
[1] 8
[1] 4.171610 3.876040 3.400334 2.879547

Call:
lm(formula = ys ~ xs)

Coefficients:
(Intercept)           xs
     -5.805        4.198

[1] 0.2037377
[1] 9
[1] 4.172007 3.876232 3.400501 2.879633

Call:
lm(formula = ys ~ xs)

Coefficients:
(Intercept)           xs
     -5.808        4.199

[1] 0.2038563
[1] 10
[1] 4.172239 3.876345 3.400598 2.879683

Call:
lm(formula = ys ~ xs)

Coefficients:
(Intercept)           xs
      -5.81         4.20

[1] 0.2039256
[1] 11
[1] 4.172375 3.876411 3.400655 2.879712

Call:
lm(formula = ys ~ xs)

Coefficients:
(Intercept)           xs
     -5.811        4.201

[1] 0.2039662
[1] 12
```

```
[1] 4.172454 3.876449 3.400689 2.879730

Call:
lm(formula = ys ~ xs)

Coefficients:
(Intercept)          xs
     -5.812        4.201

[1] 0.20399
[1] 13
[1] 4.172501 3.876472 3.400708 2.879740

Call:
lm(formula = ys ~ xs)

Coefficients:
(Intercept)          xs
     -5.812        4.201

[1] 0.2040039
[1] 14
[1] 4.172528 3.876485 3.400720 2.879746

Call:
lm(formula = ys ~ xs)

Coefficients:
(Intercept)          xs
     -5.812        4.202

[1] 0.2040121
[1] 15
[1] 4.172544 3.876493 3.400727 2.879749

Call:
lm(formula = ys ~ xs)

Coefficients:
(Intercept)          xs
     -5.812        4.202

[1] 0.2040169
[1] 16
[1] 4.172553 3.876498 3.400730 2.879751

Call:
lm(formula = ys ~ xs)

Coefficients:
(Intercept)          xs
     -5.812        4.202

[1] 0.2040197
[1] 17
[1] 4.172559 3.876500 3.400733 2.879752
```

```
Call:
lm(formula = ys ~ xs)

Coefficients:
(Intercept)          xs
    -5.813       4.202

[1] 0.2040213
[1] 18
[1] 4.172562 3.876502 3.400734 2.879753

Call:
lm(formula = ys ~ xs)

Coefficients:
(Intercept)          xs
    -5.813       4.202

[1] 0.2040223
[1] 19
[1] 4.172564 3.876503 3.400735 2.879753

Call:
lm(formula = ys ~ xs)

Coefficients:
(Intercept)          xs
    -5.813       4.202

[1] 0.2040228
```

```
############E-M algorithm ##############

t<-data$temp

time<-data$time

h <- function(x){ dnorm(x)/(1-pnorm(x))}

v <- 1000/(t+273.2)

x<-log10(time)

n<-length(data$temp)

m<-n-sum(data$censorind)

e<-1

emreg <- function(v, x, n, m){    #x: uncensored +censored

a <- lm(x~v)                      #fits linear model as if no censoring
```

```
beta <- coef(a)              #use as initial vaues
sigma <- sqrt(deviance(a)/(n-2))
output <- c(beta,sigma)
beta0<-beta
while(e>10^-6) {
mu <- beta[1]+beta[2]*v
y <- mu+sigma*h((x-mu)/sigma)          #estimated values
y[1:m] <- x[1:m]                    #x[1:m] not censored so use them
a <- lm(y~v)                        #fit model again
beta <- coef(a)                      #new coefficients
xx <- (x[(m+1):n]-mu[(m+1):n])/sigma
ss <- sum((y[1:m]-mu[1:m])^2)+sigma^2*sum((1+xx*psi(xx)))
sigma <- sqrt(ss/n)                  #new sigma
e<-max(abs(beta-beta0))
beta0<-beta
output <- c(output,beta,sigma)
}
return(matrix(output,ncol=3,byrow=T))
}
```

Output

```
         [,1]     [,2]      [,3]
 [1,] -4.930507 3.747043 0.1572178
 [2,] -5.260094 3.926205 0.1997510
 [3,] -5.511173 4.055823 0.2172663
 [4,] -5.678436 4.139847 0.2288319
 [5,] -5.785372 4.193345 0.2371179
```

[6,] -5.855179 4.228364 0.2430911

[7,] -5.902304 4.252084 0.2474183

[8,] -5.935001 4.268584 0.2505660

[9,] -5.958128 4.280275 0.2528627

[10,] -5.974700 4.288663 0.2545424

[11,] -5.986678 4.294730 0.2557730

[12,] -5.995384 4.299143 0.2566758

[13,] -6.001738 4.302364 0.2573387

[14,] -6.006386 4.304722 0.2578259

[15,] -6.009794 4.306450 0.2581841

[16,] -6.012296 4.307719 0.2584476

[17,] -6.014133 4.308651 0.2586415

[18,] -6.015484 4.309337 0.2587843

[19,] -6.016478 4.309841 0.2588893

[20,] -6.017209 4.310212 0.2589667

[21,] -6.017747 4.310485 0.2590236

[22,] -6.018143 4.310686 0.2590655

[23,] -6.018435 4.310834 0.2590964

[24,] -6.018650 4.310943 0.2591192

[25,] -6.018808 4.311023 0.2591359

[26,] -6.018924 4.311082 0.2591482

[27,] -6.019010 4.311126 0.2591573

[28,] -6.019073 4.311158 0.2591640

[29,] -6.019120 4.311181 0.2591689

[30,] -6.019154 4.311199 0.2591726

[31,] -6.019179 4.311211 0.2591752

[32,] -6.019198 4.311221 0.2591772

[33,] -6.019211 4.311228 0.2591787

[34,] -6.019221 4.311233 0.2591797

[35,] -6.019229 4.311237 0.2591805

[36,] -6.019234 4.311239 0.2591811

[37,] -6.019238 4.311241 0.2591815

[38,] -6.019241 4.311243 0.2591818

[39,] -6.019244 4.311244 0.2591821

[40,] -6.019245 4.311245 0.2591822

[41,] -6.019246 4.311245 0.2591824

[42,] -6.019247 4.311246 0.2591825

>