STAT 661: Project

LS v.s. EM

Jacob, Robin, Ryan & Shen

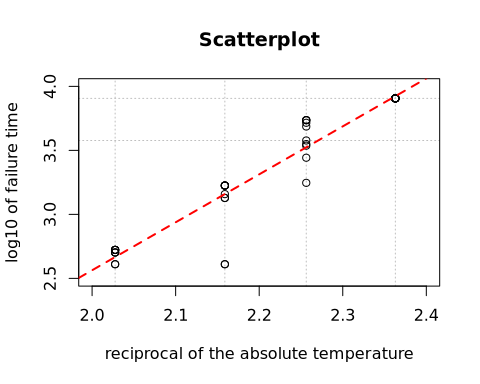
Dec, 2019

# Least Square Method v.s. EM Method

# Appendix

## 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(1,5),  
 408,408,504,504,504,rep(1,5)))  
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(logcen[4],5))  
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

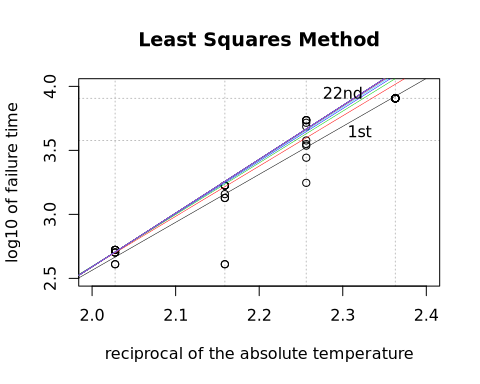


# Iteration 0   
sigma\_0 <- sigma(fit0) #standard error of residuals   
beta\_00 <- coef(fit0)[1] #intercept   
beta\_10 <- coef(fit0)[2] #slope   
mu\_0 <- beta\_00 + beta\_10\*trec #mean log time to failure   
z <- (logcen-mu\_0)/sigma\_0 #z-vector   
ex\_mu\_0 <- mu\_0 + sigma\_0\*dnorm(z)/(1-pnorm(z)) #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','Sigma','Iteration')))  
  
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   
 #new expected mean log times to failure  
 phi[1:4] <- mu + phi[7]\*dnorm(z)/(1-pnorm(z))  
 conv <- dist(rbind(PHI[phi[8]-1,1:4],phi[1:4]))   
 if(conv < delta) break   
 PHI[phi[8],]<-phi   
}

## LS Results

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 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 |

## LS figure



## EM Method

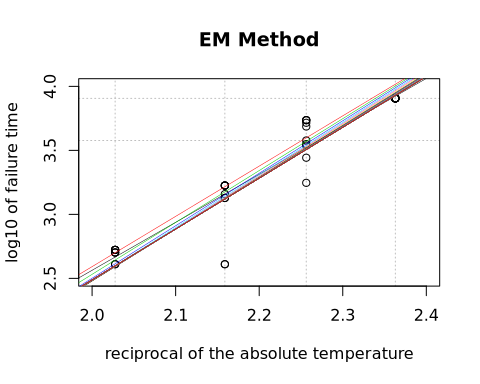
* E-step
* M-step

## the EM algorithm’s pseudo code

## 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))  
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(1,5),  
 408,408,504,504,504,rep(1,5)))  
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))  
  
y <- (y\_uncensored+y\_censored)  
  
index\_censored <- which(y %in% w)  
index\_uncensored <- which(!(y %in% w))  
  
m <- length(index\_uncensored)  
n<-length(index\_censored)+m  
  
# initial value  
fit0 <- lm(y~nu) #linear model   
sigma <- sigma(fit0) #standard error of residuals   
beta0 <- as.numeric(coef(fit0)[1]) #intercept   
beta1 <- as.numeric(coef(fit0)[2]) #slope   
  
nu\_i <- nu[index\_censored]  
nu\_j <- nu[index\_uncensored]  
  
mu\_i <- beta0+beta1\*nu\_i  
mu\_j <- beta0 + beta1 \* nu\_j  
  
H\_i <- dnorm((y[index\_censored]-mu\_i)/sigma)/(1-pnorm((y[index\_censored]-mu\_i)/sigma))  
# H\_i <-dnorm(0)/(1-pnorm(0))  
D\_j <- y[index\_uncensored]-mu\_j # difference between uncensored y and mean\_j  
ET <- unique(mu\_i+ sigma\*H\_i)   
  
S <- 60  
THETA<-matrix(nrow=S,ncol=8,dimnames=list(NULL,  
 c('Intercept','Slope','Sigma','mu150','mu170','mu190','mu220','Iteration')))  
THETA[1,]<-c(beta0,beta1,sigma,unique(ET),1)  
Q <- 0  
k=1  
delta = 1e-4  
repeat {   
beta0<- THETA[k,1]  
beta1<- THETA[k,2]  
sigma<- THETA[k,3]  
  
# M step   
mu\_i <- beta0+beta1\*nu\_i  
# H\_i<- dnorm((y[index\_censored]-mu\_i)/sigma)/(1-pnorm((y[index\_censored]-mu\_i)/sigma))  
  
beta0\_star <- 7.012488+3.601233\*sum((mu\_i+sigma\*H\_i)\*(2.208376-nu\_i))  
beta1\_star <- -2.586712-1/.6113\*sum((mu\_i+sigma\*H\_i)\*(2.201434-nu\_i))  
  
mu\_j <- beta0\_star+beta1\_star\*nu\_j  
 D\_j <- y[index\_uncensored]-mu\_j   
   
mu\_i\_star <- beta0\_star+beta1\_star\*nu\_i   
# H\_i\_star<- dnorm((y[index\_censored]-mu\_i\_star)/sigma)/(1-pnorm((y[index\_censored]-mu\_i\_star)/sigma))  
sigma\_star <- (sum((y[index\_uncensored]-beta0\_star-beta1\_star\*nu\_j)^2)+sum(sigma^2+H\_i\*sigma\*2\*(mu\_i-mu\_i\_star)+(mu\_i-mu\_i\_star)^2))/40  
  
# E step  
# Get Q  
k <- k+1   
Q[k] <-   
 -n\*log(2\*pi)/2-  
 n\*log(sigma\_star)-  
 (1/(2\*sigma\_star^2))\*sum((D\_j)^2)- # 1/2sigma^2\*sum(j uncensored) -  
 (1/(2\*sigma\_star^2))\*sum( # 1/2sigma^2\*sum(i censored   
 mu\_i^2+ # mu\_i\_star^2+  
 sigma^2+ # sigma\_star^2+   
 sigma\*( # sigma\_star\*(  
 y[index\_censored]+mu\_i-2\*mu\_i\_star # w\_i+mu\_i\_star-2mu\_i  
 )\*H\_i- # )\*H)-  
 2\*mu\_i\*mu\_i\_star+ # 2\*mu\_i\*mu\_i\_star+  
 mu\_i\_star^2 # mu\_i^2  
 ) # )  
# Update THETA  
 THETA[k,1] <- beta0\_star  
 THETA[k,2] <- beta1\_star  
 THETA[k,3] <- sigma\_star  
 THETA[k,4:7]<-unique(mu\_i\_star+ sigma\_star\*H\_i)   
 THETA[k,8] <- k  
  
 if(abs(Q[k]-Q[k-1])<=delta) break  
}  
# THETA

## EM figure



## The results of THETA

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Intercept | Slope | Sigma | mu150 | mu170 | mu190 | mu220 | Iteration |
| -4.931 | 3.747 | 0.1572 | 4.038 | 3.809 | 3.329 | 2.83 | 1 |
| -5.261 | 3.927 | 0.02861 | 4.038 | 3.651 | 3.247 | 2.73 | 2 |
| -5.576 | 4.054 | 0.02264 | 4.02 | 3.612 | 3.201 | 2.667 | 3 |
| -5.7 | 4.101 | 0.02017 | 4.004 | 3.589 | 3.175 | 2.635 | 4 |
| -5.729 | 4.108 | 0.01916 | 3.992 | 3.575 | 3.161 | 2.62 | 5 |
| -5.715 | 4.098 | 0.01869 | 3.982 | 3.565 | 3.153 | 2.614 | 6 |
| -5.684 | 4.082 | 0.01843 | 3.974 | 3.559 | 3.148 | 2.611 | 7 |
| -5.649 | 4.065 | 0.01827 | 3.968 | 3.555 | 3.145 | 2.611 | 8 |
| -5.616 | 4.049 | 0.01817 | 3.963 | 3.551 | 3.144 | 2.611 | 9 |
| -5.587 | 4.035 | 0.01811 | 3.959 | 3.549 | 3.143 | 2.612 | 10 |
| -5.563 | 4.023 | 0.01806 | 3.956 | 3.547 | 3.142 | 2.613 | 11 |
| -5.543 | 4.013 | 0.01802 | 3.954 | 3.545 | 3.141 | 2.613 | 12 |
| -5.526 | 4.005 | 0.01799 | 3.952 | 3.544 | 3.141 | 2.614 | 13 |
| -5.512 | 3.999 | 0.01797 | 3.95 | 3.543 | 3.14 | 2.614 | 14 |
| -5.502 | 3.994 | 0.01796 | 3.949 | 3.542 | 3.14 | 2.615 | 15 |
| -5.493 | 3.99 | 0.01795 | 3.948 | 3.542 | 3.14 | 2.615 | 16 |
| -5.486 | 3.986 | 0.01794 | 3.947 | 3.541 | 3.14 | 2.615 | 17 |
| -5.48 | 3.984 | 0.01793 | 3.946 | 3.541 | 3.14 | 2.616 | 18 |
| -5.475 | 3.981 | 0.01792 | 3.946 | 3.541 | 3.14 | 2.616 | 19 |
| -5.472 | 3.98 | 0.01792 | 3.945 | 3.54 | 3.14 | 2.616 | 20 |
| -5.469 | 3.978 | 0.01791 | 3.945 | 3.54 | 3.139 | 2.616 | 21 |
| -5.466 | 3.977 | 0.01791 | 3.945 | 3.54 | 3.139 | 2.616 | 22 |
| -5.465 | 3.976 | 0.01791 | 3.944 | 3.54 | 3.139 | 2.616 | 23 |
| -5.463 | 3.976 | 0.01791 | 3.944 | 3.54 | 3.139 | 2.616 | 24 |
| -5.462 | 3.975 | 0.01791 | 3.944 | 3.54 | 3.139 | 2.616 | 25 |
| -5.461 | 3.975 | 0.0179 | 3.944 | 3.54 | 3.139 | 2.617 | 26 |
| -5.46 | 3.974 | 0.0179 | 3.944 | 3.54 | 3.139 | 2.617 | 27 |
| -5.459 | 3.974 | 0.0179 | 3.944 | 3.539 | 3.139 | 2.617 | 28 |
| -5.459 | 3.974 | 0.0179 | 3.944 | 3.539 | 3.139 | 2.617 | 29 |
| -5.459 | 3.974 | 0.0179 | 3.944 | 3.539 | 3.139 | 2.617 | 30 |
| -5.458 | 3.973 | 0.0179 | 3.944 | 3.539 | 3.139 | 2.617 | 31 |
| -5.458 | 3.973 | 0.0179 | 3.944 | 3.539 | 3.139 | 2.617 | 32 |
| -5.458 | 3.973 | 0.0179 | 3.944 | 3.539 | 3.139 | 2.617 | 33 |
| -5.458 | 3.973 | 0.0179 | 3.944 | 3.539 | 3.139 | 2.617 | 34 |
| -5.458 | 3.973 | 0.0179 | 3.944 | 3.539 | 3.139 | 2.617 | 35 |
| -5.457 | 3.973 | 0.0179 | 3.944 | 3.539 | 3.139 | 2.617 | 36 |
| -5.457 | 3.973 | 0.0179 | 3.944 | 3.539 | 3.139 | 2.617 | 37 |
| -5.457 | 3.973 | 0.0179 | 3.944 | 3.539 | 3.139 | 2.617 | 38 |
| -5.457 | 3.973 | 0.0179 | 3.944 | 3.539 | 3.139 | 2.617 | 39 |
| -5.457 | 3.973 | 0.0179 | 3.944 | 3.539 | 3.139 | 2.617 | 40 |
| -5.457 | 3.973 | 0.0179 | 3.944 | 3.539 | 3.139 | 2.617 | 41 |
| -5.457 | 3.973 | 0.0179 | 3.944 | 3.539 | 3.139 | 2.617 | 42 |
| -5.457 | 3.973 | 0.0179 | 3.944 | 3.539 | 3.139 | 2.617 | 43 |
| -5.457 | 3.973 | 0.0179 | 3.944 | 3.539 | 3.139 | 2.617 | 44 |
| -5.457 | 3.973 | 0.0179 | 3.944 | 3.539 | 3.139 | 2.617 | 45 |
| -5.457 | 3.973 | 0.0179 | 3.944 | 3.539 | 3.139 | 2.617 | 46 |
| -5.457 | 3.973 | 0.0179 | 3.944 | 3.539 | 3.139 | 2.617 | 47 |
| -5.457 | 3.973 | 0.0179 | 3.944 | 3.539 | 3.139 | 2.617 | 48 |
| -5.457 | 3.973 | 0.0179 | 3.944 | 3.539 | 3.139 | 2.617 | 49 |
| -5.457 | 3.973 | 0.0179 | 3.944 | 3.539 | 3.139 | 2.617 | 50 |
| -5.457 | 3.973 | 0.0179 | 3.944 | 3.539 | 3.139 | 2.617 | 51 |
| -5.457 | 3.973 | 0.0179 | 3.944 | 3.539 | 3.139 | 2.617 | 52 |
| -5.457 | 3.973 | 0.0179 | 3.944 | 3.539 | 3.139 | 2.617 | 53 |
| -5.457 | 3.973 | 0.0179 | 3.944 | 3.539 | 3.139 | 2.617 | 54 |
| -5.457 | 3.973 | 0.0179 | 3.944 | 3.539 | 3.139 | 2.617 | 55 |
| -5.457 | 3.973 | 0.0179 | 3.944 | 3.539 | 3.139 | 2.617 | 56 |
| -5.457 | 3.973 | 0.0179 | 3.944 | 3.539 | 3.139 | 2.617 | 57 |
| -5.457 | 3.973 | 0.0179 | 3.944 | 3.539 | 3.139 | 2.617 | 58 |
| NA | NA | NA | NA | NA | NA | NA | NA |
| NA | NA | NA | NA | NA | NA | NA | NA |

## The results of Q

*0*, *-754.7*, *-771.6*, *-896.1*, *-959.3*, *-993*, *-1013*, *-1025*, *-1033*, *-1039*, *-1043*, *-1046*, *-1049*, *-1050*, *-1052*, *-1053*, *-1054*, *-1054*, *-1055*, *-1055*, *-1056*, *-1056*, *-1056*, *-1056*, *-1057*, *-1057*, *-1057*, *-1057*, *-1057*, *-1057*, *-1057*, *-1057*, *-1057*, *-1057*, *-1057*, *-1057*, *-1057*, *-1057*, *-1057*, *-1057*, *-1057*, *-1057*, *-1057*, *-1057*, *-1057*, *-1057*, *-1057*, *-1057*, *-1057*, *-1057*, *-1057*, *-1057*, *-1057*, *-1057*, *-1057*, *-1057*, *-1057* and *-1057*

# Reference

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>