Final Report: Example 12.1.(p.397 of Klein & Moeschberber)

Ans. by Pan Chi-Hung

Consider a two-parameter Weibull model. The survival function is given by

$$S_X(x) = \exp(\,-\lambda x^{\alpha}\,),$$

where $\lambda > 0, \alpha > 0$.

Taking the log transform. Let $Y = \log X$.

$$\begin{aligned}
\Pr(\,Y > y\,) &= \Pr(\,\log X > y\,) = \Pr\{\,X > \exp(\,y\,)\,\} \\
&= \exp\{\,-\lambda \exp(\,\alpha y\,)\,\} \\
&= \exp\left[\,-\exp\left\{\frac{y-(-\log\lambda/\alpha\,)}{1/\alpha}\right\}\right] \\
&= \exp\left[\,-\exp\left\{\frac{y-\mu}{\sigma}\right\}\right],
\end{aligned}$$

where $\mu = -\log\lambda/\alpha, \sigma = 1/\alpha$.

The density function is given by

$$f_Y(\,y\,) = -\frac{\partial S_Y(\,y\,)}{\partial y} = \frac{1}{\sigma}\exp\left[\left\{\frac{y-\mu}{\sigma}\right\} - \exp\left\{\frac{y-\mu}{\sigma}\right\}\right].$$

The likelihood function for right-censored data is

$$\begin{aligned}
L(\,\mu,\sigma\,) &= \prod_i f_Y(\,y_i\,)^{\delta_i} S_Y(\,y_i\,)^{1-\delta_i} \\
&= \prod_i \frac{1}{\sigma}\exp\left[\left\{\frac{y_i-\mu}{\sigma}\right\} - \exp\left\{\frac{y_i-\mu}{\sigma}\right\}\right]^{\delta_i} \exp\left[\,-\exp\left\{\frac{y_i-\mu}{\sigma}\right\}\right]^{1-\delta_i},
\end{aligned}$$

where $\delta_i = I(\,Y_i \le C_{ri})$, C_{ri}: censoring time.

The log-likelihood function is

$$\begin{aligned}
l(\,\mu,\sigma\,) &= \log L(\,\mu,\sigma\,) \\
&= -\sum_i \delta_i \log\sigma + \sum_i \delta_i\left[\left\{\frac{y_i-\mu}{\sigma}\right\} - \exp\left\{\frac{y_i-\mu}{\sigma}\right\}\right] + \sum_i (\,1-\delta_i\,)\left[\,-\exp\left\{\frac{y_i-\mu}{\sigma}\right\}\right] \\
&= -\sum_i \delta_i \log\sigma + \sum_i \delta_i\left\{\frac{y_i-\mu}{\sigma}\right\} + \sum_i\left[\,-\exp\left\{\frac{y_i-\mu}{\sigma}\right\}\right].
\end{aligned}$$

Based data, we want to find the maximum likelihood estimator (MLE). We present the Newton-Raphson algorithm (NR) to find the MLE of (μ, σ). The following is the algorithm.

Step1.initial value (μ^0, σ^0)

Step2.Updated: $\begin{bmatrix} \mu^{k+1} \\ \sigma^{k+1} \end{bmatrix} = \begin{bmatrix} \mu^{k} \\ \sigma^{k} \end{bmatrix} - \begin{bmatrix} u_{\mu\mu}(\mu,\sigma) & u_{\mu\sigma}(\mu,\sigma) \\ u_{\mu\sigma}(\mu,\sigma) & u_{\sigma\sigma}(\mu,\sigma) \end{bmatrix}^{-1} \begin{bmatrix} u_{\mu}(\mu,\sigma) \\ u_{\sigma}(\mu,\sigma) \end{bmatrix} \Bigg|_{(\mu,\sigma)=(\mu^k,\sigma^k)}$,

where $u_\mu(\mu,\sigma) = \dfrac{\partial l(\mu,\sigma)}{\partial \mu}, u_\sigma(\mu,\sigma) = \dfrac{\partial l(\mu,\sigma)}{\partial \sigma}, u_{\mu\mu}(\mu,\sigma) = \dfrac{\partial^2 l(\mu,\sigma)}{\partial \mu^2}$,

$u_{\mu\sigma}(\mu,\sigma) = \dfrac{\partial^2 l(\mu,\sigma)}{\partial\mu\partial\sigma}$ and $u_{\sigma\sigma}(\mu,\sigma) = \dfrac{\partial^2 l(\mu,\sigma)}{\partial \sigma^2}$.

Step3. If $\left|\mu^{k+1} - \mu^k\right| < \varepsilon$ and $\left|\sigma^{k+1} - \sigma^k\right| < \varepsilon$ then stop.

In order the NR algorithm. First, we should calculate the score vector and hessian matrix. Score vector:

$$u_\mu(\mu,\sigma) = -\sum_i \frac{\delta_i}{\sigma} + \sum_i \left[\frac{1}{\sigma} \exp\left\{ \frac{y_i - \mu}{\sigma} \right\} \right]$$

$$u_\sigma(\mu,\sigma) = -\sum_i \frac{\delta_i}{\sigma} - \sum_i \delta_i \left\{ \frac{y_i - \mu}{\sigma^2} \right\} + \sum_i \left[\left\{ \frac{y_i - \mu}{\sigma^2} \right\} \exp\left\{ \frac{y_i - \mu}{\sigma} \right\} \right]$$

Hessian matrix:

$$u_{\mu\mu}(\mu,\sigma) = -\sum_i \left[\frac{1}{\sigma^2} \exp\left\{ \frac{y_i - \mu}{\sigma} \right\} \right]$$

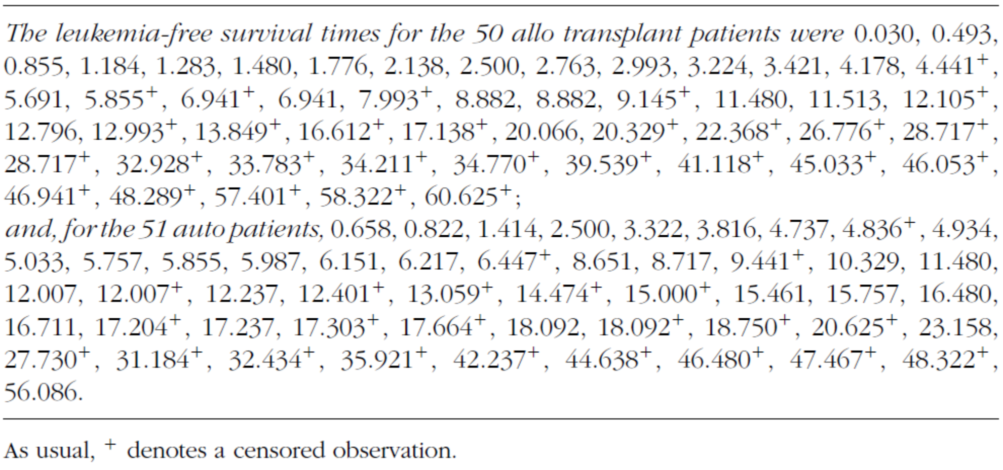
$$u_{\mu\sigma}(\mu,\sigma) = \sum_i \frac{\delta_i}{\sigma^2} + \sum_i \left[-\frac{1}{\sigma^2} \exp\left\{ \frac{y_i - \mu}{\sigma} \right\} - \left\{ \frac{y_i - \mu}{\sigma^3} \right\} \exp\left\{ \frac{y_i - \mu}{\sigma} \right\} \right]$$

$$u_{\sigma\sigma}(\mu,\sigma) = \sum_i \frac{\delta_i}{\sigma^2} + \sum_i 2\delta_i \left\{ \frac{y_i - \mu}{\sigma^3} \right\} - \sum_i \left[2\left\{ \frac{y_i - \mu}{\sigma^3} \right\} \exp\left\{ \frac{y_i - \mu}{\sigma} \right\} \right]$$
$$- \sum_i \left[\left\{ \frac{y_i - \mu}{\sigma^2} \right\}^2 \exp\left\{ \frac{y_i - \mu}{\sigma} \right\} \right]$$

⇨ Observed fisher information:

$$\begin{bmatrix} -u_{\mu\mu}(\mu,\sigma) & -u_{\mu\sigma}(\mu,\sigma) \\ -u_{\mu\sigma}(\mu,\sigma) & -u_{\sigma\sigma}(\mu,\sigma) \end{bmatrix}.$$

There are two data sets described in the textbook section 1.9.



The leukemia-free survival times for the 50 allo transplant patients were 0.030, 0.493, 0.855, 1.184, 1.283, 1.480, 1.776, 2.138, 2.500, 2.763, 2.993, 3.224, 3.421, 4.178, 4.441$^+$, 5.691, 5.855$^+$, 6.941$^+$, 6.941, 7.993$^+$, 8.882, 8.882, 9.145$^+$, 11.480, 11.513, 12.105$^+$, 12.796, 12.993$^+$, 13.849$^+$, 16.612$^+$, 17.138$^+$, 20.066, 20.329$^+$, 22.368$^+$, 26.776$^+$, 28.717$^+$, 28.717$^+$, 32.928$^+$, 33.783$^+$, 34.211$^+$, 34.770$^+$, 39.539$^+$, 41.118$^+$, 45.033$^+$, 46.053$^+$, 46.941$^+$, 48.289$^+$, 57.401$^+$, 58.322$^+$, 60.625$^+$;
and, for the 51 auto patients, 0.658, 0.822, 1.414, 2.500, 3.322, 3.816, 4.737, 4.836$^+$, 4.934, 5.033, 5.757, 5.855, 5.987, 6.151, 6.217, 6.447$^+$, 8.651, 8.717, 9.441$^+$, 10.329, 11.480, 12.007, 12.007$^+$, 12.237, 12.401$^+$, 13.059$^+$, 14.474$^+$, 15.000$^+$, 15.461, 15.757, 16.480, 16.711, 17.204$^+$, 17.237, 17.303$^+$, 17.664$^+$, 18.092, 18.092$^+$, 18.750$^+$, 20.625$^+$, 23.158, 27.730$^+$, 31.184$^+$, 32.434$^+$, 35.921$^+$, 42.237$^+$, 44.638$^+$, 46.480$^+$, 47.467$^+$, 48.322$^+$, 56.086.

As usual, $^+$ denotes a censored observation.

Because the data is follows the Weibull distribution. But now our model is extreme value distribution. We transform the data from Weibull distribution to extreme value distribution by taking log. Let $\varepsilon = 10^{-4}$ Using the first 50 patients to estimate the MLE of (μ, σ) $(\hat{\mu}_{allo}, \hat{\sigma}_{allo}) = (4.254250, 1.944376)$. The variance-covariance matrix for $\hat{\mu}_{allo}, \hat{\sigma}_{allo}$

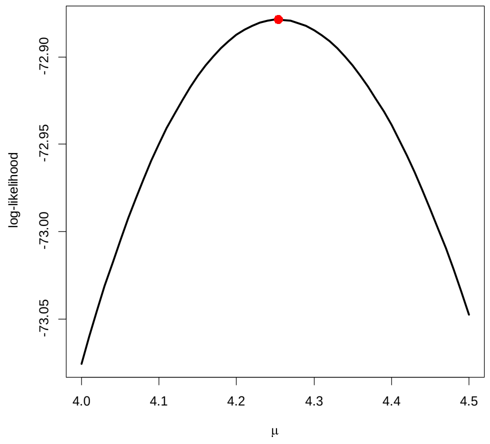
is $\begin{bmatrix} -u_{\mu\mu}(\mu,\sigma) & -u_{\mu\sigma}(\mu,\sigma) \\ -u_{\mu\sigma}(\mu,\sigma) & -u_{\sigma\sigma}(\mu,\sigma) \end{bmatrix}^{-1} = \begin{bmatrix} 0.22862405 & 0.08765807 \\ 0.08765807 & 0.13533152 \end{bmatrix}$.

Using the last 51 patients to estimate the MLE of (μ, σ)

$(\hat{\mu}_{auto}, \hat{\sigma}_{auto}) = (3.451869, 1.110957)$ The variance-covariance matrix for

$\hat{\mu}_{allo}, \hat{\sigma}_{allo}$

is $\begin{bmatrix} 0.04760507 & 0.01042011 \\ 0.01042011 & 0.03079733 \end{bmatrix}$. Also I check the MLE by the following figure1,2.



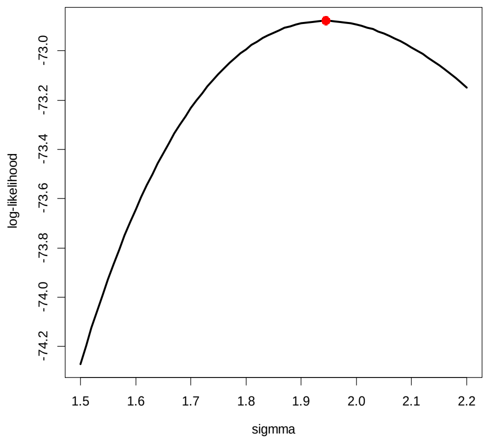


Figure1

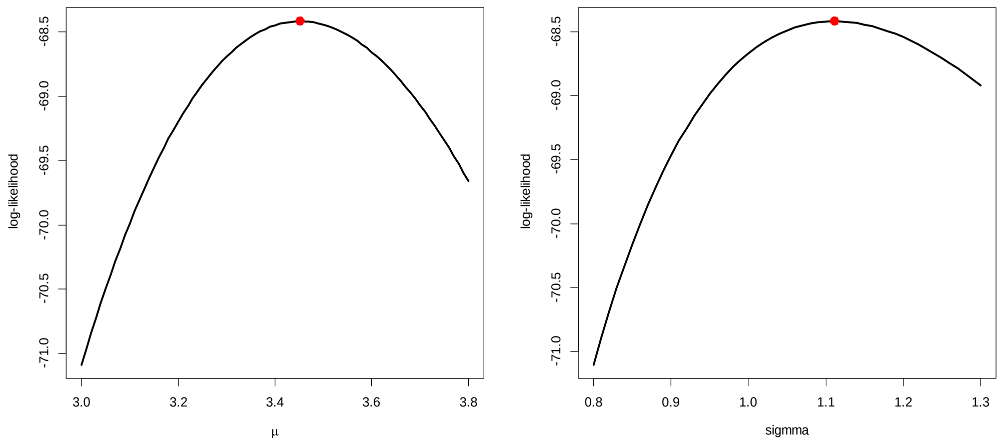


Figure2

Because the invariance principle of MLE. We can solve the equations:

$\hat{\mu} = -\log\lambda / \alpha, \hat{\sigma} = 1/\alpha$ to find the MLE of (λ, α)

Result:

$$(\hat{\lambda}_{auto}, \hat{\alpha}_{auto}) = (0.04472989, 0.9001248),\ (\hat{\lambda}_{allo}, \hat{\alpha}_{allo}) = (0.1121434, 0.5143037).$$

By the two dimensional delta method we can derive the variance-covariance matrix:

For $(\hat{\lambda}_{auto}, \hat{\alpha}_{auto})$ is $\begin{bmatrix} 0.0004541815 & -0.002781671 \\ -0.002781671 & 0.02021734 \end{bmatrix}$

For $(\hat{\lambda}_{allo}, \hat{\alpha}_{allo})$ is $\begin{bmatrix} 0.001639633 & -0.003179966 \\ -0.003179966 & 0.009468422 \end{bmatrix}$.

To test the hypothesis that the exponential model provides as good a fit to the data as the Weibull model. $\Leftrightarrow H_0 : \sigma = 1$

By the LR-test:

$$-2\{ l(\hat{\lambda}_{auto}, 1) - l(\hat{\lambda}_{auto}, \hat{\alpha}_{auto}) \} = -2(-68.67027 + 68.41986) = 0.5008186$$

$$-2\{ l(\hat{\lambda}_{allo}, 1) - l(\hat{\lambda}_{allo}, \hat{\alpha}_{allo}) \} = -2(-83.65852 + 72.87873) = 21.55957$$

Under $H_0\ -2\{ l(\hat{\lambda}, 1) - l(\hat{\lambda}, \hat{\alpha}) \} \approx \chi_1^2$

$$-2\{ l(\hat{\lambda}_{auto}, 1) - l(\hat{\lambda}_{auto}, \hat{\alpha}_{auto}) \} = 0.5008186 < \chi^2_{1,0.05} = 7.879$$

⇨ Not reject H_0.Exponential distribution may suitable for auto transplants.

$$-2\{ l(\hat{\lambda}_{allo}, 1) - l(\hat{\lambda}_{allo}, \hat{\alpha}_{allo}) \} = 21.55957 > \chi^2_{1,0.05} = 7.879$$

reject H_0. Weibull distribution may suitable for allo transplants.

Code

```
allo_data =
c(0.030,0.493,0.855,1.184,1.283,1.480,1.776,2.138,2.500,2.763,2.993,3.224,3.421,4.
178,4.441,
5.691,5.855,6.941,6.941,7.993,8.882,8.882,9.145,11.480,11.513,12.105,
12.796,12.993,13.849,16.612,17.138,20.066,20.329,22.368,26.776,28.717,
28.717,32.928,33.783,34.211,34.770,39.539,41.118,45.033,46.053,
46.941,48.289,57.401,58.322,60.625)
censor_allo = c(rep(1,14),0,1,0,0,1,0,1,1,0,1,1,0,1,0,0,0,0,1,rep(0,18))

score_vector_fun = function(mu,sig){
c(sum(-d/sig)+sum(1/sig*exp((y-mu)/sig)),
-sum(d/sig)+sum(d*(-(y-mu)/sig^2))+sum((y-mu)/sig^2*exp((y-mu)/sig)))
}

hessian_matrix_fun = function(mu,sig){
matrix(c(-sum(1/sig^2*exp((y-mu)/sig)),

sum(d/sig^2)-sum(1/sig^2*exp((y-mu)/sig)+(y-mu)/sig^3*exp((y-mu)/sig)),

sum(d/sig^2)-sum(1/sig^2*exp((y-mu)/sig)+(y-mu)/sig^3*exp((y-mu)/sig)),
            sum(d/sig^2)+sum(2*d*(y-mu)/sig^3-2*(y-mu)/sig^3*exp((y-mu)/sig)-
            (y-mu)^2/sig^4*exp((y-mu)/sig))
),2,2)
}

y=log(allo_data)
d=censor_allo
mu=-log(mean(allo_data)^-1)
sig=1
n=length(d)
```

```
par_old=c(mu,sig)
AI=1

repeat{
score_vector = score_vector_fun(par_old[1],par_old[2])
hessian_matrix = hessian_matrix_fun(par_old[1],par_old[2])
par_new = par_old-solve(hessian_matrix)%*%score_vector

error1 = abs(par_new[1]-par_old[1])
error2 = abs(par_new[2]-par_old[2])
if( (error1 < 10^-4) && (error2 < 10^-4) ){break}
par_old = par_new
mu = par_old[1]
sig = par_old[2]
AI = AI+1
}

par_new
solve(-hessian_matrix_fun(par_new[1],par_new[2]))

auto_data = c(0.658,0.822,1.414,2.500,3.322,3.816,4.737,4.836,4.934,
5.033,5.757,5.855,5.987,6.151,6.217,6.447,8.651,8.717,9.441,10.329,11.480,
12.007,12.007,12.237,12.401,13.059,14.474,15.000,15.461,15.757,16.480,
16.711,17.204,17.237,17.303,17.664,18.092,18.092,18.750,20.625,23.158,
27.730,31.184,32.434,35.921,42.237,44.638,46.480,47.467,48.322,56.086)

censor_auto = c(1, 1, 1, 1, 1, 1, 1,0,1,
1, 1, 1, 1, 1, 1,0,1,1,0,1,1,1,0,1,
0,0,0,0,1,1,1,1,0,1,0,0,1,0,0,0,1,rep(0,9),1)

y=log(auto_data)
d=censor_auto
mu=-log(mean(auto_data)^-1)
sig=1
n=length(d)
par_old=c(mu,sig)
AI=1
```

```
repeat{
score_vector = score_vector_fun(par_old[1],par_old[2])
hessian_matrix = hessian_matrix_fun(par_old[1],par_old[2])
par_new = par_old-solve(hessian_matrix)%*%score_vector

error1 = abs(par_new[1]-par_old[1])
error2 = abs(par_new[2]-par_old[2])
if( (error1 < 10^-4) && (error2 < 10^-4) ){break}
par_old = par_new
mu = par_old[1]
sig = par_old[2]
AI = AI+1
}

par_new
-solve(hessian_matrix_fun(par_new[1],par_new[2]))

ll = c()
a = seq(3,3.8,by = 0.01)
for( i in 1:length(a) ){
mu = a[i]
ll[i] = log_likeli_fun(mu,par_new[2])
}
plot(a,ll,type = "l",xlab = expression(mu),ylab = "log-likelihood",lwd = 3)
points(par_new[1],log_likeli_fun(par_new[1],par_new[2]),cex=1.5,col=2,pch=16)

ll = c()
a = seq(0.8,1.3,by = 0.01)
for( i in 1:length(a) ){
sig = a[i]
ll[i] = log_likeli_fun(par_new[1],sig)
}
plot(a,ll,type = "l",xlab = expression(sigmma),ylab = "log-likelihood",lwd = 3)
points(par_new[2],log_likeli_fun(par_new[1],par_new[2]),cex=1.5,col=2,pch=16)
```