MSMS 308 : Practical 07

Ananda Biswas

Exam Roll No. : 24419STC053

December 8, 2025

Question

Obtain maximum likelihood estimates of Weibull parameters for varying sample sizes and censoring proportions $p_c \in \{0.2, 0.3, 0.4\}$ under

(1) complete data (no censoring scheme),

(2) Type I censoring scheme,

(3) Type II censoring scheme.

and compare the estimates.

Theory and R Program

The probability density function of a lifetime T having Weibull distribution with shape $\alpha > 0$ and scale $\lambda > 0$ is given by

$$f(t) = \frac{\alpha}{\lambda}\left(\frac{t}{\lambda}\right)^{\alpha-1} \exp\left[-\left(\frac{t}{\lambda}\right)^{\alpha}\right], \qquad t \geq 0,$$

The Cumulative Distribution Function is given by

$$F(t) = 1 - \exp\left[-\left(\frac{t}{\lambda}\right)^{\alpha}\right], \qquad t \geq 0,$$

The Survival Function is given by

$$S(t) = \exp\left[-\left(\frac{t}{\lambda}\right)^{\alpha}\right], \qquad t \geq 0.$$

1 MLE of α and λ with no censoring

Let $T_1, \ldots, T_n \overset{\text{iid}}{\sim} \text{Weibull}(\alpha, \lambda)$ with density $f(t)$ and survival $S(t)$.

The likelihood function is given by

$$\begin{aligned} L(\alpha, \lambda) &= \prod_{i=1}^{n} \frac{\alpha}{\lambda}\left(\frac{t_i}{\lambda}\right)^{\alpha-1} e^{-(t_i/\lambda)^{\alpha}} \\ &= \left(\frac{\alpha}{\lambda}\right)^{n} \left(\prod_{i=1}^{n} \frac{t_i}{\lambda}\right)^{\alpha-1} \exp\left\{-\sum_{i=1}^{n}\left(\frac{t_i}{\lambda}\right)^{\alpha}\right\}. \end{aligned}$$

The log-likelihood function is given by

$$l(\alpha,\lambda) = n\log\alpha - n\log\lambda + (\alpha-1)\sum_{i=1}^{n}\log\left(\frac{t_i}{\lambda}\right) - \sum_{i=1}^{n}\left(\frac{t_i}{\lambda}\right)^{\alpha}.$$

Now,

$$\frac{\partial}{\partial\alpha}l(\alpha,\lambda) = \frac{n}{\alpha} + \sum_{i=1}^{n}\log\left(\frac{t_i}{\lambda}\right) - \sum_{i=1}^{n}\left(\frac{t_i}{\lambda}\right)^{\alpha}\log\left(\frac{t_i}{\lambda}\right) = u(\alpha,\lambda), \text{ say.} \tag{1}$$

and

$$\begin{aligned}\frac{\partial}{\partial\lambda}l(\alpha,\lambda) &= -\frac{n}{\lambda} + (\alpha-1)\sum_{i=1}^{n}\frac{\lambda}{t_i}\cdot\left(-\frac{t_i}{\lambda^2}\right) + \sum_{i=1}^{n}\frac{\alpha\cdot t_i^{\alpha}}{\lambda^{\alpha+1}}\\ &= -\frac{n}{\lambda} - (\alpha-1)\frac{n}{\lambda} + \sum_{i=1}^{n}\frac{\alpha\cdot t_i^{\alpha}}{\lambda^{\alpha+1}}\\ &= -\frac{n\alpha}{\lambda} + \sum_{i=1}^{n}\frac{\alpha\cdot t_i^{\alpha}}{\lambda^{\alpha+1}} = v(\alpha,\lambda), \text{ say.}\end{aligned}$$

Setting $v(\alpha,\lambda) = 0$ we get,

$$\begin{aligned}-\frac{n\alpha}{\lambda} + \sum_{i=1}^{n}\frac{\alpha\cdot t_i^{\alpha}}{\lambda^{\alpha+1}} &= 0\\ \Rightarrow \frac{n}{\lambda} &= \sum_{i=1}^{n}\frac{t_i^{\alpha}}{\lambda^{\alpha+1}}\\ \Rightarrow \frac{n}{\lambda} &= \frac{1}{\lambda^{\alpha+1}}\sum_{i=1}^{n}t_i^{\alpha}\\ \Rightarrow \lambda^{\alpha} &= \frac{1}{n}\sum_{i=1}^{n}t_i^{\alpha}\\ \therefore \lambda &= \left(\frac{1}{n}\sum_{i=1}^{n}t_i^{\alpha}\right)^{\frac{1}{\alpha}}\end{aligned} \tag{2}$$

Setting $u(\alpha,\lambda) = 0$ does not yield any closed form solution. So for getting the ML estimate of α, we resort to numerical methods (here Newton-Raphson method).

Now,

$$u_{\alpha}(\alpha,\lambda) = \frac{\partial}{\partial\alpha}u(\alpha,\lambda) = -\frac{n}{\alpha^2} - \sum_{i=1}^{n}\left(\frac{t_i}{\lambda}\right)^{\alpha}\left[\log\left(\frac{t_i}{\lambda}\right)\right]^2; \tag{3}$$

At each iteration, with the present value of α we calculate λ by using (2); then we use the obtained value of λ in (1) and (3) to improve the estimate of α by Newton-Raphson method.

```
true_alpha <- 3; true_lambda <- 2
```

```r
sample_size <- c(50, 100, 150, 200, 250)
```

```r
estimate_lambda <- function(s, alpha) mean(s^alpha)^(1/alpha)
```

```r
u <- function(alpha, lambda, s){

  a <- length(s) / alpha

  b <- sum(log(s / lambda))

  c <- sum((s / lambda)^alpha * log(s / lambda))

  return(a + b - c)
}
```

```r
u_alpha <- function(alpha, lambda, s){

  a <- - length(s) / alpha^2

  b <- sum((s / lambda)^alpha * log(s / lambda)^2)

  return(a - b)
}
```

```r
estimate_alpha <- function(s, initial, epsilon = 0.0001, iterations = 100){

  alphas <- c(initial)

  for (i in 2:iterations) {
    l <- estimate_lambda(s, alphas[i-1])

    alphas[i] <- alphas[i-1] - u(alphas[i-1], l, s) / u_alpha(alphas[i-1], l, s)

    if(abs((alphas[i] - alphas[i-1])) < epsilon) break
  }

  return(alphas[length(alphas)])
}
```

```r
alpha_hat = lambda_hat = c()
```

```r
for(i in 1:length(sample_size)){

  x <- rweibull(sample_size[i], shape = true_alpha, scale = true_lambda)

  alpha_hat[i] <- estimate_alpha(x, 1)

  lambda_hat[i] <- estimate_lambda(x, alpha_hat[i])
}
```

Now we shall empirically calculate bias and MSE of the estimates for different sample sizes.

```
bias_and_MSE <- function(size){

  alpha_estimates = lambda_estimates = c()

  for (i in 1:100){

    x <- rweibull(size, shape = true_alpha, scale = true_lambda)

    alpha_estimates[i] <- estimate_alpha(x, 1)

    lambda_estimates[i] <- estimate_lambda(x, alpha_estimates[i])
  }

  alpha_bias <- mean(alpha_estimates) - true_alpha

  lambda_bias <- mean(lambda_estimates) - true_lambda

  alpha_MSE <- mean( (alpha_estimates - true_alpha)^2 )

  lambda_MSE <- mean( (lambda_estimates - true_lambda)^2 )

  return(c(alpha_bias, lambda_bias, alpha_MSE, lambda_MSE))
}
```

```
alpha_bias = lambda_bias = alpha_MSE = lambda_MSE = c()

for (i in 1:length(sample_size)) {

  temp <- bias_and_MSE(sample_size[i])

  alpha_bias[i] <- temp[1]

  lambda_bias[i] <- temp[2]

  alpha_MSE[i] <- temp[3]

  lambda_MSE[i] <- temp[4]
}
```

```
df1 <- data.frame(Sample.Size = sample_size,
                  alpha_hat = alpha_hat,
                  bias.alpha = alpha_bias,
                  MSE.alpha = alpha_MSE,
                  lambda_hat = lambda_hat,
                  bias.lambda = lambda_bias,
                  MSE.lambda = lambda_MSE)
```

```
stargazer(df1, summary = FALSE, rownames = FALSE)
```

Table 1:

Sample.Size	alpha_hat	bias.alpha	MSE.alpha	lambda_hat	bias.lambda	MSE.lambda
50	2.545	0.069	0.104	1.922	0.006	0.009
100	3.496	0.022	0.043	1.934	-0.009	0.005
150	3.283	0.025	0.032	1.932	0.001	0.004
200	3.202	0.037	0.047	2.050	-0.002	0.002
250	2.890	0.0002	0.025	1.979	-0.001	0.002

2 MLE of α and λ with Type I Right Censoring

Let us have a sample of size n under Type I Right censoring scheme from Weibull(α, λ) with density $f(t)$ and survival $S(t)$. $t_1, t_2, \ldots, t_r$ be r complete observations and rest $n-r$ are censored observations at C.

The likelihood function is given by

$$\begin{aligned} L(\alpha,\lambda) &= \left[\prod_{i=1}^{r} f(t_i)\right] [S(C)]^{n-r} \\ &= \prod_{i=1}^{r}\left[\frac{\alpha}{\lambda}\left(\frac{t_i}{\lambda}\right)^{\alpha-1}\exp\left(-\left(\frac{t_i}{\lambda}\right)^{\alpha}\right)\right]\left[\exp\left(-\left(\frac{C}{\lambda}\right)^{\alpha}\right)\right]^{n-r} \\ &= \alpha^r \lambda^{-\alpha r}\left(\prod_{i=1}^{r} t_i^{\alpha-1}\right)\exp\left[-\sum_{i=1}^{r}\left(\frac{t_i}{\lambda}\right)^{\alpha} - (n-r)\left(\frac{C}{\lambda}\right)^{\alpha}\right]. \end{aligned}$$

The log-likelihood function is given by

$$\ell(\alpha,\lambda) = r\log\alpha - \alpha r\log\lambda + (\alpha-1)\sum_{i=1}^{r}\log t_i - \sum_{i=1}^{r}\left(\frac{t_i}{\lambda}\right)^{\alpha} - (n-r)\left(\frac{C}{\lambda}\right)^{\alpha}.$$

Now,

$$\frac{\partial}{\partial\alpha}\ell(\alpha,\lambda) = \frac{r}{\alpha} - r\log\lambda + \sum_{i=1}^{r}\log t_i - \sum_{i=1}^{r}\left(\frac{t_i}{\lambda}\right)^{\alpha}\log\left(\frac{t_i}{\lambda}\right) - (n-r)\left(\frac{C}{\lambda}\right)^{\alpha}\log\left(\frac{C}{\lambda}\right) = u(\alpha,\lambda), \text{ say } \quad (4)$$

and

$$\begin{aligned} \frac{\partial}{\partial\lambda}\ell(\alpha,\lambda) &= \frac{\partial}{\partial\lambda}\left[-\alpha r\log\lambda - \sum_{i=1}^{r}\left(\frac{t_i}{\lambda}\right)^{\alpha} - (n-r)\left(\frac{C}{\lambda}\right)^{\alpha}\right] \\ &= -\frac{\alpha r}{\lambda} - \sum_{i=1}^{r}\frac{\partial}{\partial\lambda}\left(t_i^{\alpha}\lambda^{-\alpha}\right) - (n-r)\frac{\partial}{\partial\lambda}\left(C^{\alpha}\lambda^{-\alpha}\right) \\ &= -\frac{\alpha r}{\lambda} - \sum_{i=1}^{r} t_i^{\alpha}\left(-\alpha\lambda^{-\alpha-1}\right) - (n-r)C^{\alpha}\left(-\alpha\lambda^{-\alpha-1}\right) \end{aligned}$$

$$= -\frac{\alpha r}{\lambda} + \alpha \sum_{i=1}^{r} t_i^{\alpha} \lambda^{-\alpha-1} + \alpha(n-r) C^{\alpha} \lambda^{-\alpha-1}$$

$$= \frac{\alpha}{\lambda} \left[-r + \sum_{i=1}^{r} t_i^{\alpha} \lambda^{-\alpha} + (n-r) C^{\alpha} \lambda^{-\alpha} \right]$$

$$= \frac{\alpha}{\lambda} \left[-r + \sum_{i=1}^{r} \left(\frac{t_i}{\lambda} \right)^{\alpha} + (n-r) \left(\frac{C}{\lambda} \right)^{\alpha} \right] = v(\alpha, \lambda), \text{ say.}$$

Setting $v(\alpha, \lambda) = 0$ we get,

$$\frac{\alpha}{\lambda} \left[-r + \sum_{i=1}^{r} \left(\frac{t_i}{\lambda} \right)^{\alpha} + (n-r) \left(\frac{C}{\lambda} \right)^{\alpha} \right] = 0$$

$$\Rightarrow -r + \sum_{i=1}^{r} \left(\frac{t_i}{\lambda} \right)^{\alpha} + (n-r) \left(\frac{C}{\lambda} \right)^{\alpha} = 0$$

$$\Rightarrow -r + \frac{1}{\lambda^{\alpha}} \left[\sum_{i=1}^{r} t_i^{\alpha} + (n-r) C^{\alpha} \right] = 0$$

$$\Rightarrow \frac{1}{\lambda^{\alpha}} \left[\sum_{i=1}^{r} t_i^{\alpha} + (n-r) C^{\alpha} \right] = r$$

$$\Rightarrow \lambda^{\alpha} = \frac{1}{r} \left[\sum_{i=1}^{r} t_i^{\alpha} + (n-r) C^{\alpha} \right]$$

$$\therefore \ \widehat{\lambda} = \left\{ \frac{1}{r} \left[\sum_{i=1}^{r} t_i^{\alpha} + (n-r) C^{\alpha} \right] \right\}^{\frac{1}{\alpha}} \tag{5}$$

Setting $u(\alpha, \lambda) = 0$ does not yield any closed form solution. So for getting the ML estimate of α, we resort to numerical methods (here Newton-Raphson method).

Now,

$$u_{\alpha}(\alpha, \lambda) = \frac{\partial}{\partial \alpha} u(\alpha, \lambda) = -\frac{r}{\alpha^2} - \sum_{i=1}^{r} \left(\frac{t_i}{\lambda} \right)^{\alpha} \left[\log\left(\frac{t_i}{\lambda} \right) \right]^2 - (n-r) \left(\frac{C}{\lambda} \right)^{\alpha} \left[\log\left(\frac{C}{\lambda} \right) \right]^2. \tag{6}$$

At each iteration, with the present value of α we calculate λ by using (5); then we use the obtained value of λ in (4) and (6) to improve the estimate of α by Newton-Raphson method.

Only what is remaining is how to choose C. Censoring proportions $p_c \in \{0.2, 0.3, 0.4\} \Rightarrow$ proportion of complete observations $p_o \in \{0.8, 0.7, 0.6\}$. For our simulation purpose we set C such that

$$P(T \leq C) = p_o$$

$$\Rightarrow F_T(C) = p_o$$

$$\Rightarrow 1 - \exp\left\{ -\left(\frac{C}{\lambda} \right)^{\alpha} \right\} = p_o$$

$$\Rightarrow \exp\left\{-\left(\frac{C}{\lambda}\right)^{\alpha}\right\} = 1 - p_o$$

$$\Rightarrow \ln\left[\exp\left\{-\left(\frac{C}{\lambda}\right)^{\alpha}\right\}\right] = \ln(1 - p_o)$$

$$\Rightarrow -\left(\frac{C}{\lambda}\right)^{\alpha} = \ln(1 - p_o)$$

$$\Rightarrow \left(\frac{C}{\lambda}\right)^{\alpha} = -\ln(1 - p_o)$$

$$\Rightarrow \frac{C}{\lambda} = [-\ln(1 - p_o)]^{\frac{1}{\alpha}}$$

$$\Rightarrow C = \lambda\,[-\ln(1 - p_o)]^{\frac{1}{\alpha}}.$$

```
true_alpha <- 3; true_lambda <- 2
```

```
sample_size <- c(50, 100, 150, 200, 250)
```

```
estimate_lambda <- function(failures, C, n, alpha){

  r <- length(failures)

  a <- sum(failures^alpha)

  b <- (n-r) * C^alpha

  return( ( (a + b) / r)^(1/alpha))
}
```

```
u <- function(alpha, lambda, failures, C, n) {

  r <- length(failures)

  a <- r / alpha

  b <- - r * log(lambda) + sum(log(failures))

  c <- sum((failures / lambda)^alpha * log(failures / lambda))

  d <- (n - r) * (C / lambda)^alpha * log(C / lambda)

  return(a + b - c - d)
}
```

```
u_alpha <- function(alpha, lambda, failures, C, n) {
```

```
  r <- length(failures)

  a <- - r / alpha^2

  b <- sum((failures / lambda)^alpha * (log(failures / lambda))^2)

  c <- (n - r) * (C / lambda)^alpha * (log(C / lambda))^2

  return(a - b - c)
}
```

```
estimate_alpha <- function(failures, C, n, initial, epsilon = 0.0001, iterations = 100) {

  alphas <- c(initial)

  for (i in 2:iterations) {

    lambda <- estimate_lambda(failures, C, n, alphas[i - 1])

    score <- u(alphas[i - 1], lambda, failures, C, n)
    score_der <- u_alpha(alphas[i - 1], lambda, failures, C, n)

    alphas[i] <- alphas[i - 1] - score / score_der

    if (abs(alphas[i] - alphas[i - 1]) < epsilon) break
  }

  return(alphas[length(alphas)])
}
```

```
alpha_hat = lambda_hat = c()
```

```
prop_censored <- c(0.2, 0.3, 0.4)
prop_observed <- c(0.8, 0.7, 0.6)
```

```
for(i in 1:length(sample_size)){

  for(j in 1:length(prop_observed)){

    x <- rweibull(sample_size[i], shape = true_alpha, scale = true_lambda)

    C <- true_lambda * (- log(1 - prop_observed[j]))^(1 / true_alpha)

    failures <- x[x <= C]

    new_alpha <- estimate_alpha(failures, C, sample_size[i], 1)

    alpha_hat <- append(alpha_hat, new_alpha)
```

```
    lambda_hat <- append(lambda_hat,
                         estimate_lambda(failures, C, sample_size[i], new_alpha))
  }
}
```

Now we shall empirically calculate bias and MSE of the estimates for different sample sizes.

```
bias_and_MSE <- function(size, prop_observed){

  alpha_estimates = lambda_estimates = c()

  for (i in 1:100){

    x <- rweibull(size, shape = true_alpha, scale = true_lambda)

    C <- true_lambda * (- log(1 - prop_observed))^(1 / true_alpha)

    failures <- x[x <= C]

    new_alpha <- estimate_alpha(failures, C, size, 1)

    alpha_estimates <- append(alpha_estimates, new_alpha)

    lambda_estimates <- append(lambda_estimates,
                         estimate_lambda(failures, C, size, new_alpha))
  }

  alpha_bias <- mean(alpha_estimates) - true_alpha

  lambda_bias <- mean(lambda_estimates) - true_lambda

  alpha_MSE <- mean( (alpha_estimates - true_alpha)^2 )

  lambda_MSE <- mean( (lambda_estimates - true_lambda)^2 )

  return(c(alpha_bias, lambda_bias, alpha_MSE, lambda_MSE))
}
```

```
alpha_bias = lambda_bias = alpha_MSE = lambda_MSE = c()

for (i in 1:length(sample_size)) {

  for(j in 1:length(prop_observed)){

    temp <- bias_and_MSE(sample_size[i], prop_observed[j])

    alpha_bias <- append(alpha_bias, temp[1])

    lambda_bias <- append(lambda_bias, temp[2])

    alpha_MSE <- append(alpha_MSE, temp[3])
```

```
    lambda_MSE <- append(lambda_MSE, temp[4])
  }
}
```

```
df2 <- data.frame(Sample.Size = rep(sample_size, rep(3, length(sample_size))),
                  p_c = rep(prop_censored, length(sample_size)),
                  alpha_hat = alpha_hat,
                  bias.alpha = alpha_bias,
                  MSE.alpha = alpha_MSE,
                  lambda_hat = lambda_hat,
                  bias.lambda = lambda_bias,
                  MSE.lambda = lambda_MSE)
```

```
stargazer(df2, summary = FALSE, rownames = FALSE)
```

Table 2:

Sample.Size	p_c	alpha_hat	bias.alpha	MSE.alpha	lambda_hat	bias.lambda	MSE.lambda
50	0.200	2.266	0.045	0.129	1.990	-0.006	0.010
50	0.300	4.256	0.060	0.193	1.943	0.004	0.013
50	0.400	3.615	0.140	0.462	1.839	-0.004	0.019
100	0.200	3.260	0.023	0.056	1.875	0.001	0.005
100	0.300	3.659	0.014	0.113	1.932	0.005	0.008
100	0.400	2.875	0.074	0.154	2.104	0.007	0.007
150	0.200	2.995	-0.043	0.062	2.012	0.0001	0.003
150	0.300	2.794	0.048	0.070	2.031	0.004	0.004
150	0.400	3.006	0.056	0.095	1.978	0.011	0.004
200	0.200	2.897	0.039	0.050	2.052	0.0002	0.003
200	0.300	2.970	0.031	0.050	1.982	0.001	0.005
200	0.400	2.842	0.034	0.058	2.059	-0.004	0.004
250	0.200	3.044	-0.004	0.031	2.030	0.001	0.002
250	0.300	2.991	0.006	0.032	1.982	0.0005	0.003
250	0.400	2.548	0.002	0.044	2.162	0.0002	0.003

3 MLE of α and λ with Type II Right Censoring

Let us have a sample of size n under Type II Right censoring scheme from Weibull(α, λ) with density $f(t)$ and survival $S(t)$. $t_{(1)}, t_{(2)}, \ldots, t_{(r)}$ be r complete observations and rest $n-r$ are censored observations at $t_{(r)}$.

The likelihood function is given by

$$L(\alpha,\lambda) = \frac{n!}{(n-r)!}\left[\prod_{i=1}^{r} f(t_{(i)})\right][S(t_{(r)})]^{n-r}$$

$$= \frac{n!}{(n-r)!}\prod_{i=1}^{r}\left[\frac{\alpha}{\lambda}\left(\frac{t_{(i)}}{\lambda}\right)^{\alpha-1}\exp\left(-\left(\frac{t_{(i)}}{\lambda}\right)^{\alpha}\right)\right]\left[\exp\left(-\left(\frac{t_{(r)}}{\lambda}\right)^{\alpha}\right)\right]^{n-r}$$

$$= \frac{n!}{(n-r)!}\,\alpha^r \lambda^{-\alpha r}\left(\prod_{i=1}^{r} t_{(i)}^{\alpha-1}\right)\exp\left[-\sum_{i=1}^{r}\left(\frac{t_{(i)}}{\lambda}\right)^{\alpha} - (n-r)\left(\frac{t_{(r)}}{\lambda}\right)^{\alpha}\right].$$

The log-likelihood function is given by

$$\ell(\alpha,\lambda) = \log\left(\frac{n!}{(n-r)!}\right) + r\log\alpha - \alpha r\log\lambda + (\alpha-1)\sum_{i=1}^{r}\log t_{(i)} - \sum_{i=1}^{r}\left(\frac{t_{(i)}}{\lambda}\right)^{\alpha} - (n-r)\left(\frac{t_{(r)}}{\lambda}\right)^{\alpha}.$$

Now,

$$\frac{\partial}{\partial\alpha}\ell(\alpha,\lambda) = \frac{r}{\alpha} - r\log\lambda + \sum_{i=1}^{r}\log t_{(i)} - \sum_{i=1}^{r}\left(\frac{t_{(i)}}{\lambda}\right)^{\alpha}\log\left(\frac{t_{(i)}}{\lambda}\right) - (n-r)\left(\frac{t_{(r)}}{\lambda}\right)^{\alpha}\log\left(\frac{t_{(r)}}{\lambda}\right) = u(\alpha,\lambda),\ \text{say.} \tag{7}$$

and

$$\begin{aligned}
\frac{\partial}{\partial\lambda}\ell(\alpha,\lambda) &= \frac{\partial}{\partial\lambda}\left[-\alpha r\log\lambda - \sum_{i=1}^{r}\left(\frac{t_{(i)}}{\lambda}\right)^{\alpha} - (n-r)\left(\frac{t_{(r)}}{\lambda}\right)^{\alpha}\right] \\
&= -\frac{\alpha r}{\lambda} - \sum_{i=1}^{r} t_{(i)}^{\alpha}\left(-\alpha\lambda^{-\alpha-1}\right) - (n-r)t_{(r)}^{\alpha}\left(-\alpha\lambda^{-\alpha-1}\right) \\
&= -\frac{\alpha r}{\lambda} + \alpha\sum_{i=1}^{r} t_{(i)}^{\alpha}\lambda^{-\alpha-1} + \alpha(n-r)t_{(r)}^{\alpha}\lambda^{-\alpha-1} \\
&= \frac{\alpha}{\lambda}\left[-r + \sum_{i=1}^{r} t_{(i)}^{\alpha}\lambda^{-\alpha} + (n-r)t_{(r)}^{\alpha}\lambda^{-\alpha}\right] \\
&= \frac{\alpha}{\lambda}\left[-r + \sum_{i=1}^{r}\left(\frac{t_{(i)}}{\lambda}\right)^{\alpha} + (n-r)\left(\frac{t_{(r)}}{\lambda}\right)^{\alpha}\right] = v(\alpha,\lambda),\ \text{say.}
\end{aligned}$$

Setting $v(\alpha,\lambda) = 0$ we get,

$$\begin{aligned}
&\frac{\alpha}{\lambda}\left[-r + \sum_{i=1}^{r}\left(\frac{t_{(i)}}{\lambda}\right)^{\alpha} + (n-r)\left(\frac{t_{(r)}}{\lambda}\right)^{\alpha}\right] = 0 \\
&\Rightarrow -r + \sum_{i=1}^{r}\left(\frac{t_{(i)}}{\lambda}\right)^{\alpha} + (n-r)\left(\frac{t_{(r)}}{\lambda}\right)^{\alpha} = 0 \\
&\Rightarrow -r + \frac{1}{\lambda^{\alpha}}\left[\sum_{i=1}^{r} t_{(i)}^{\alpha} + (n-r)t_{(r)}^{\alpha}\right] = 0 \\
&\Rightarrow \frac{1}{\lambda^{\alpha}}\left[\sum_{i=1}^{r} t_{(i)}^{\alpha} + (n-r)t_{(r)}^{\alpha}\right] = r \\
&\Rightarrow \lambda^{\alpha} = \frac{1}{r}\left[\sum_{i=1}^{r} t_{(i)}^{\alpha} + (n-r)t_{(r)}^{\alpha}\right] \\
&\therefore\ \widehat{\lambda} = \left\{\frac{1}{r}\left[\sum_{i=1}^{r} t_{(i)}^{\alpha} + (n-r)t_{(r)}^{\alpha}\right]\right\}^{\frac{1}{\alpha}}.
\end{aligned} \tag{8}$$

Setting $u(\alpha, \lambda) = 0$ does not yield any closed form solution. So for getting the ML estimate of α, we resort to numerical methods (here Newton-Raphson method).

Now,

$$u_\alpha(\alpha, \lambda) = \frac{\partial}{\partial \alpha} u(\alpha, \lambda) = -\frac{r}{\alpha^2} - \sum_{i=1}^{r} \left(\frac{t_{(i)}}{\lambda}\right)^\alpha \left[\log\left(\frac{t_{(i)}}{\lambda}\right)\right]^2 - (n-r)\left(\frac{t_{(r)}}{\lambda}\right)^\alpha \left[\log\left(\frac{t_{(r)}}{\lambda}\right)\right]^2. \quad (9)$$

At each iteration, with the present value of α we calculate λ by using (8); then we use the obtained value of λ in (7) and (9) to improve the estimate of α by Newton-Raphson method.

```
true_alpha <- 3; true_lambda <- 2
```

```
sample_size <- c(50, 100, 150, 200, 250)
```

```
estimate_lambda <- function(failures, n, alpha) {

  r <- length(failures)

  t_r <- failures[r]

  a <- sum(failures^alpha)

  b <- (n - r) * t_r^alpha

  return( ((a + b) / r)^(1 / alpha) )
}
```

```
u <- function(alpha, lambda, failures, n) {

  r <- length(failures)

  t_r <- failures[r]

  a <- r / alpha

  b <- - r * log(lambda) + sum(log(failures))

  c <- sum((failures / lambda)^alpha * log(failures / lambda))

  d <- (n - r) * (t_r / lambda)^alpha * log(t_r / lambda)

  return(a + b - c - d)
}
```

```
u_alpha <- function(alpha, lambda, failures, n) {

  r <- length(failures)
```

```
  t_r <- failures[r]

  a <- - r / alpha^2

  b <- sum((failures / lambda)^alpha * (log(failures / lambda))^2)

  c <- (n - r) * (t_r / lambda)^alpha * (log(t_r / lambda))^2

  return(a - b - c)
}
```

```
estimate_alpha <- function(failures, n, initial, epsilon = 0.0001, iterations = 100) {

  alphas <- c(initial)

  for (i in 2:iterations) {

    lambda <- estimate_lambda(failures, n, alphas[i - 1])

    score <- u(alphas[i - 1], lambda, failures, n)
    score_der <- u_alpha(alphas[i - 1], lambda, failures, n)

    alphas[i] <- alphas[i - 1] - score / score_der

    if (abs(alphas[i] - alphas[i - 1]) < epsilon) break
  }

  return(alphas[length(alphas)])
}
```

```
alpha_hat = lambda_hat = c()
```

```
prop_censored <- c(0.2, 0.3, 0.4)
```

```
for(i in 1:length(sample_size)){

  for(j in 1:length(prop_censored)){

    x <- rweibull(sample_size[i], shape = true_alpha, scale = true_lambda)

    r <- floor( (1 - prop_censored[j]) * sample_size[i])

    x_ord <- sort(x)

    failures <- x_ord[1:r]

    new_alpha <- estimate_alpha(failures, sample_size[i], 1)

    alpha_hat <- append(alpha_hat, new_alpha)
```

```
    lambda_hat <- append(lambda_hat,
                         estimate_lambda(failures, sample_size[i], new_alpha))
  }
}
```

Now we shall empirically calculate bias and MSE of the estimates for different sample sizes.

```
bias_and_MSE <- function(size, prop_censored){

  alpha_estimates = lambda_estimates = c()

  for (i in 1:100){

    x <- rweibull(size, shape = true_alpha, scale = true_lambda)

    r <- floor( (1 - prop_censored) * size)

    x_ord <- sort(x)

    failures <- x_ord[1:r]

    new_alpha <- estimate_alpha(failures, size, 1)

    alpha_estimates <- append(alpha_estimates, new_alpha)

    lambda_estimates <- append(lambda_estimates,
                         estimate_lambda(failures, size, new_alpha))
  }

  alpha_bias <- mean(alpha_estimates) - true_alpha

  lambda_bias <- mean(lambda_estimates) - true_lambda

  alpha_MSE <- mean( (alpha_estimates - true_alpha)^2 )

  lambda_MSE <- mean( (lambda_estimates - true_lambda)^2 )

  return(c(alpha_bias, lambda_bias, alpha_MSE, lambda_MSE))
}
```

```
alpha_bias = lambda_bias = alpha_MSE = lambda_MSE = c()

for (i in 1:length(sample_size)) {

  for(j in 1:length(prop_censored)){

    temp <- bias_and_MSE(sample_size[i], prop_censored[j])

    alpha_bias <- append(alpha_bias, temp[1])

    lambda_bias <- append(lambda_bias, temp[2])
```

```
    alpha_MSE <- append(alpha_MSE, temp[3])

    lambda_MSE <- append(lambda_MSE, temp[4])
  }
}
```

```
df3 <- data.frame(Sample.Size = rep(sample_size, rep(3, length(sample_size))),
                  p_c = rep(prop_censored, length(sample_size)),
                  alpha_hat = alpha_hat,
                  bias.alpha = alpha_bias,
                  MSE.alpha = alpha_MSE,
                  lambda_hat = lambda_hat,
                  bias.lambda = lambda_bias,
                  MSE.lambda = lambda_MSE)
```

```
stargazer(df3, summary = FALSE, rownames = FALSE)
```

Table 3:

Sample.Size	p_c	alpha_hat	bias.alpha	MSE.alpha	lambda_hat	bias.lambda	MSE.lambda
50	0.200	2.294	0.099	0.163	1.980	-0.016	0.009
50	0.300	4.351	0.131	0.216	1.935	-0.013	0.011
50	0.400	4.065	0.194	0.470	1.777	-0.021	0.015
100	0.200	3.295	0.041	0.059	1.870	-0.002	0.005
100	0.300	3.690	0.053	0.121	1.928	-0.004	0.007
100	0.400	2.924	0.106	0.158	2.087	-0.002	0.007
150	0.200	2.978	-0.030	0.060	2.016	-0.003	0.003
150	0.300	2.811	0.065	0.067	2.026	-0.0001	0.004
150	0.400	3.004	0.078	0.096	1.979	0.004	0.004
200	0.200	2.939	0.051	0.054	2.042	-0.002	0.003
200	0.300	2.947	0.044	0.046	1.988	-0.003	0.004
200	0.400	2.759	0.059	0.062	2.089	-0.011	0.004
250	0.200	3.062	0.007	0.031	2.026	-0.001	0.002
250	0.300	3.018	0.018	0.032	1.976	-0.002	0.003
250	0.400	2.651	0.014	0.049	2.116	-0.003	0.003