The Kaplan-Meier Estimate with R*

```
> rm(list=ls()); options(scipen=999)
> wdata = read.table("http://www.utstat.utoronto.ca/brunner/data/legal/Weibull.data2.txt")
> # head(wdata)
> Time = wdata$Time; Uncensored = wdata$Uncensored # Avoiding the attach() function
>
> # install.packages("survival",dependencies=TRUE) # Only need to do this once
> library(survival) # Do this every time
>
> y = Surv(Time,Uncensored); y[1:20] # A pre-processing step
 [1] 1.60+ 0.60+ 3.03  2.90+ 3.60  2.76  0.36+ 2.93  0.61  2.50+ 0.07+ 7.71  4.92+
[14] 0.08+ 0.01+ 2.04  2.16  2.97+ 2.81  4.90+
> km1 = survfit(y ~ 1) # Like a regression model with just an intercept: No x values
> km1
Call: survfit(formula = y ~ 1)

      n  events  median 0.95LCL 0.95UCL
 275.00  144.00    4.54    4.21    4.91
> # For comparison, MLE of the median was 4.466, 95% CI = (4.199 4.733)

> summary(km1) # Returns a matrix, first column t_j etc.
Call: survfit(formula = y ~ 1)
```

t_j n_j d_j $\widehat{S}(t_j)$

```
 time n.risk n.event survival std.err lower 95% CI upper 95% CI
 0.34    262       1   0.9962 0.00381      0.98874        1.000
 0.61    252       1   0.9922 0.00547      0.98156        1.000
 1.07    240       1   0.9881 0.00684      0.97479        1.000
 1.18    234       1   0.9839 0.00801      0.96831        1.000
 1.25    230       1   0.9796 0.00904      0.96203        0.997
 1.27    229       1   0.9753 0.00996      0.95598        0.995
   .      .       .      .      .            .              .
   .      .       .      .      .            .              .
   .      .       .      .      .            .              .

 7.71     4       1   0.0421 0.02220      0.01496        0.118
 8.14     2       1   0.0210 0.01856      0.00373        0.119
```

$$\widehat{p}_j = \frac{n_j - d_j}{n_j} \qquad \widehat{S}(t) = \prod_{t_j \leq t} \widehat{p}_j$$

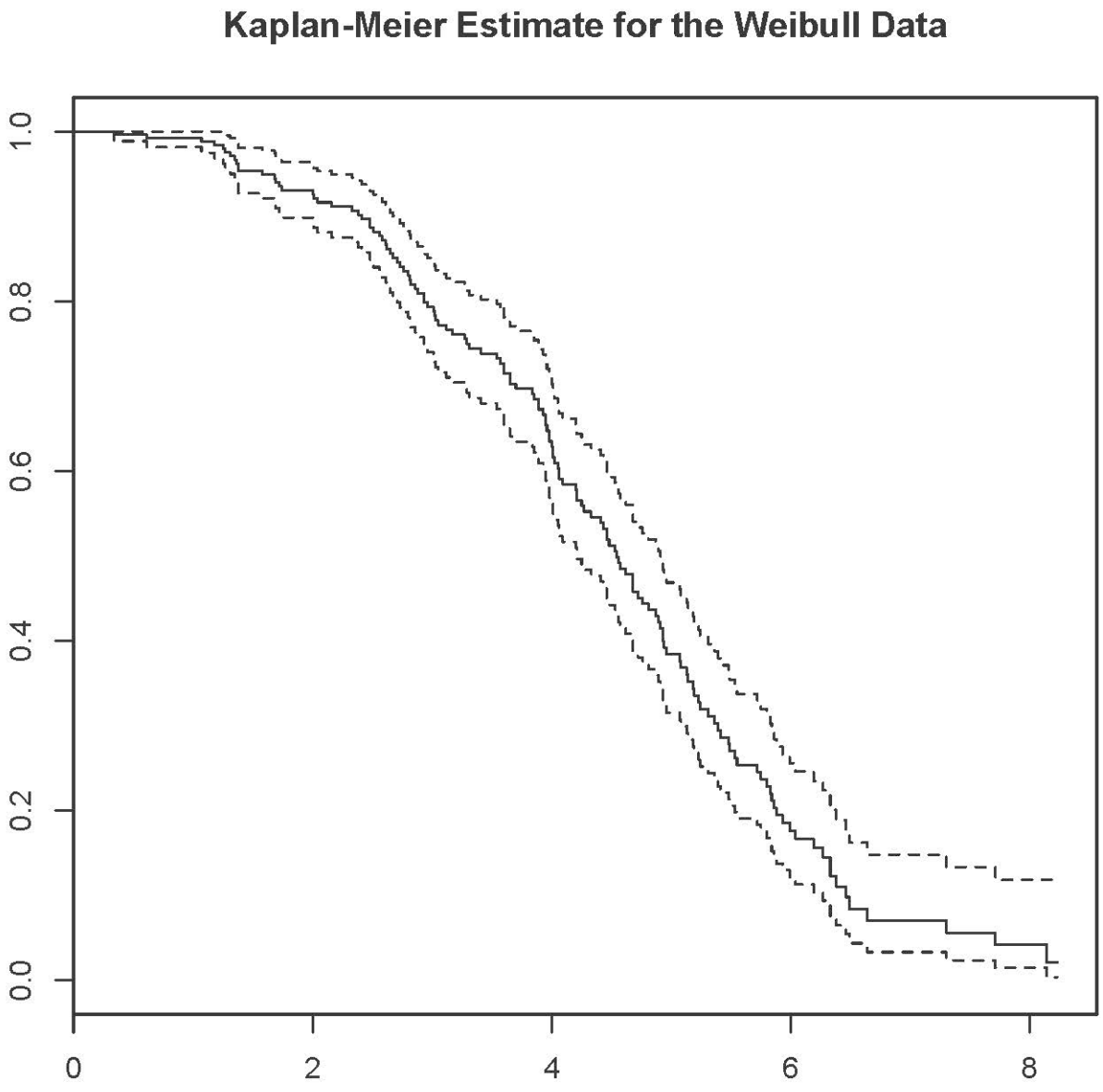
Estimated asymptotic variance is $\widehat{S}(t)^2 \sum_{t_j \leq t} \left(\frac{d_j}{n_j(n_j - d_j)} \right)$

```
> phat1=261/262; phat2=251/252; phat3=239/240
> Shat = phat1*phat2*phat3; Shat # Compare S-hat(1.07) = 0.9881
[1] 0.9880958
>
> # Get SE of S-hat(1.07):
> estvarlog = 1/(262*261) + 1/(252*251) + 1/(239*240)
> estvarShat = estvarlog * Shat^2 # One-var delta method
> seShat = sqrt(estvarShat); seShat # Compare 0.00684
[1] 0.006836256
> Shat + 1.96*seShat
[1] 1.001495
>  # Upper confidence limit was truncated to one.
```

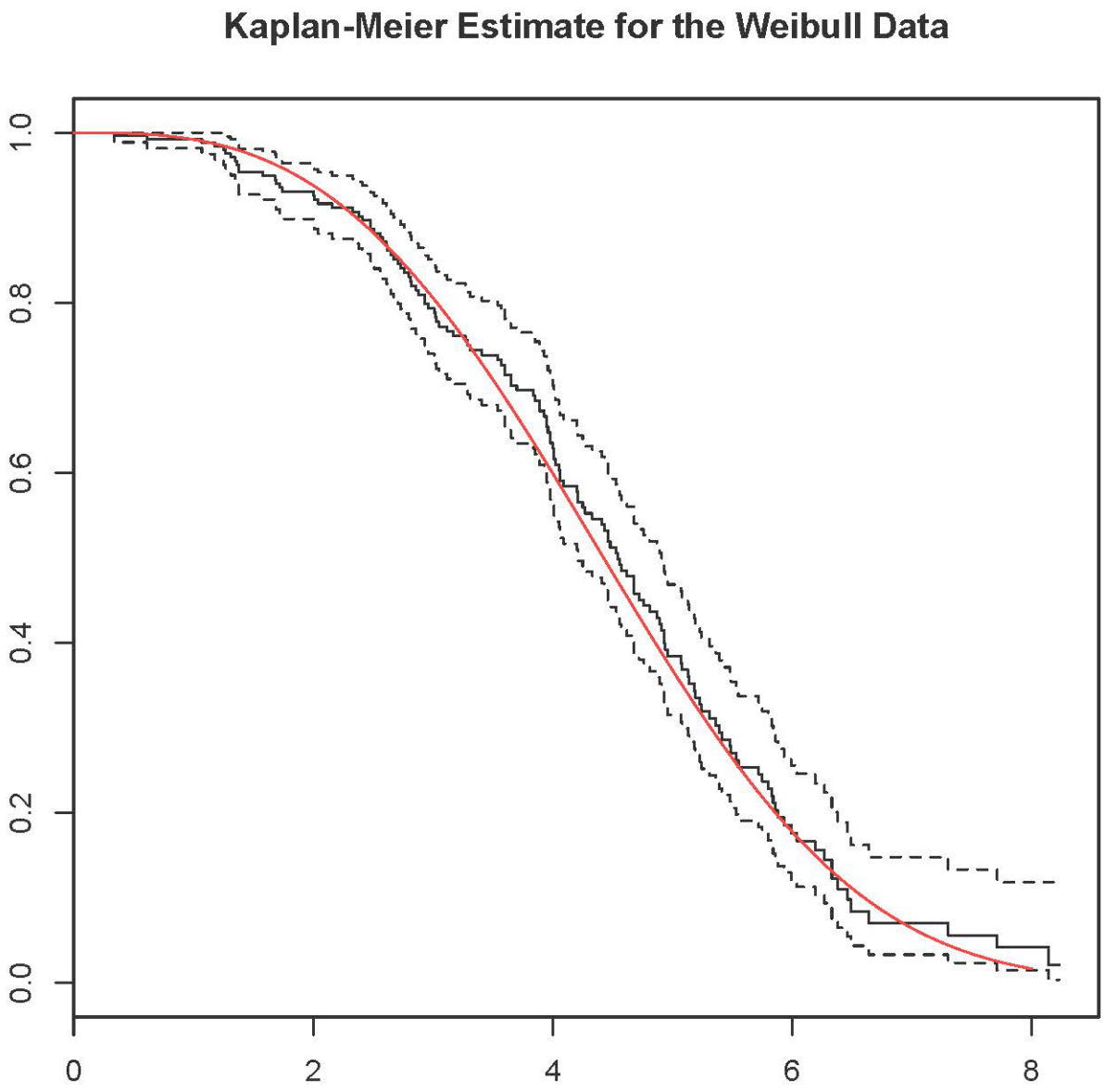
---

* This document is free and open source. See last page for copyright information.

```
> plot(km1)
> title('Kaplan-Meier Estimate for the Weibull Data')
```

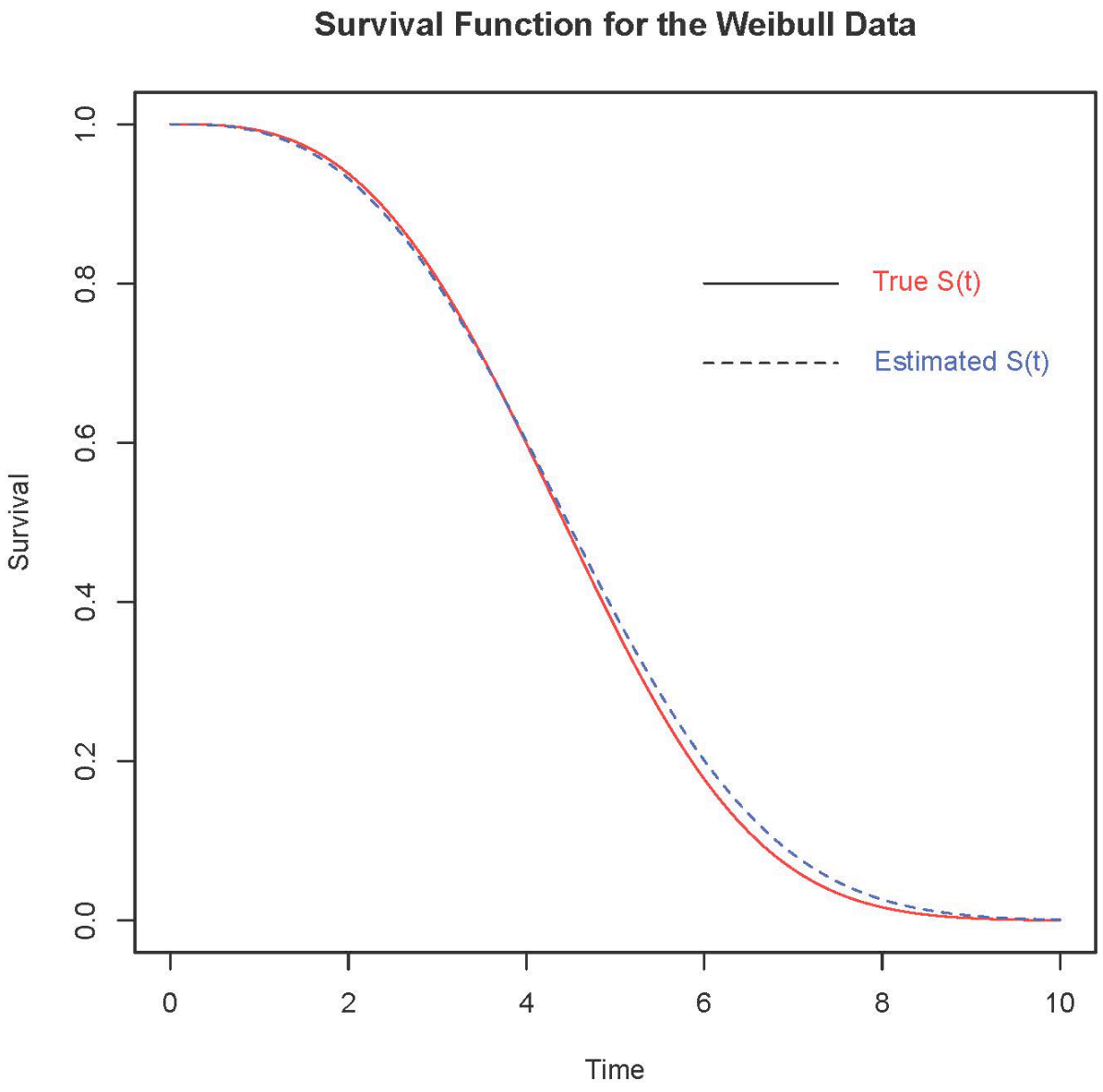


```
> # These data were simulated, so I know the true parameter values.
> # Add true S(t) to the plot
> truealpha = 3; truelambda = 1/5
> x = seq(from=0,to=8,length=101)
> trueS = exp(-(truelambda*x)^truealpha)
> lines(x,trueS,lty=1, col = "red1")
```

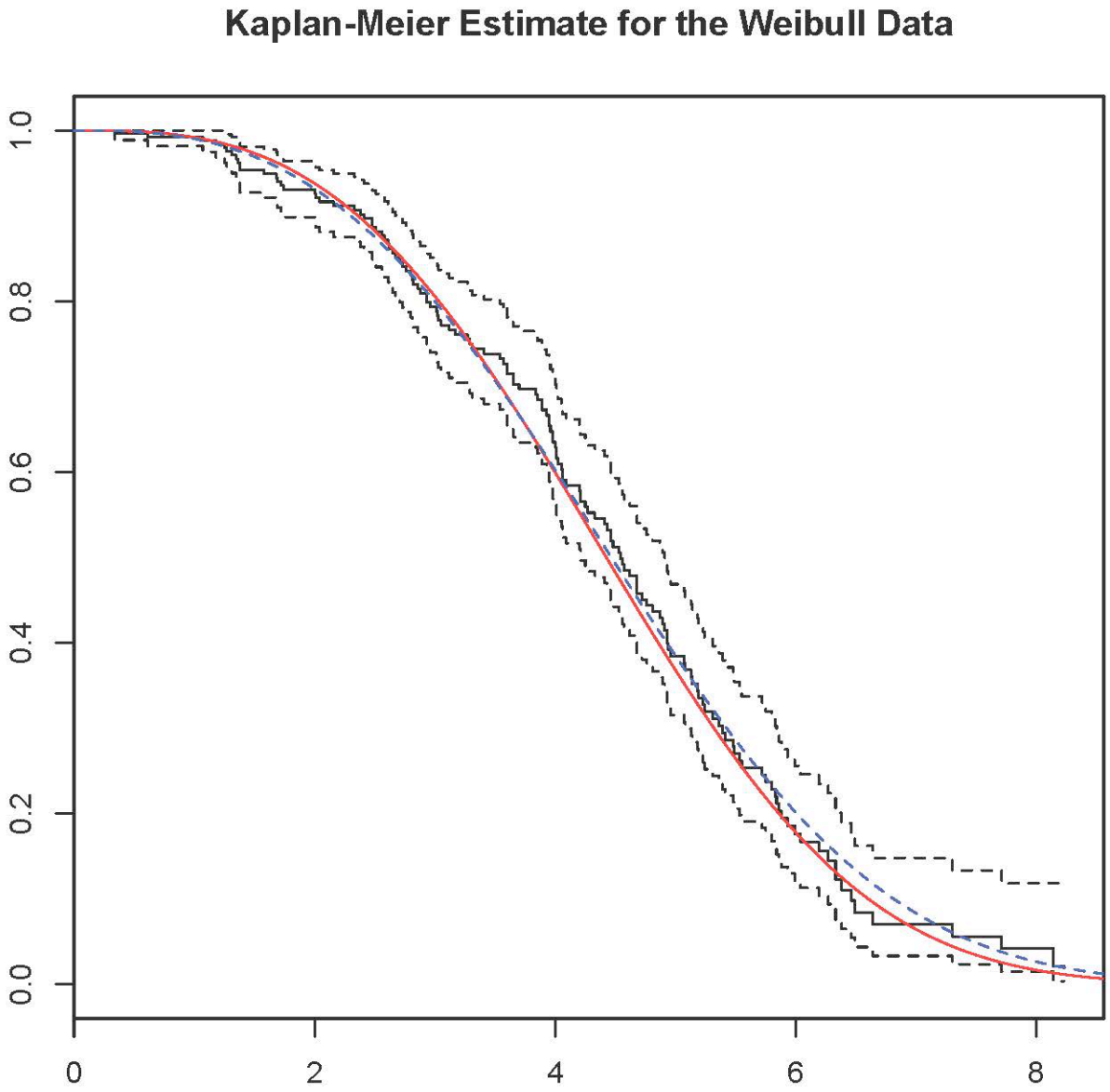


```
> # Parametric estimate of S(t) and median, for comparison.
>
> mloglike = function(theta,t,delta)
+     { # Minus log likelihood function
+     alpha = theta[1]; lambda = theta[2]
+     # logf and logS will be of length n
+     logf = log(alpha)+log(lambda)+(alpha-1)*log(lambda*t) + -(lambda*t)^alpha
+     logS = -(lambda*t)^alpha
+     value = -sum(logf*delta) - sum(logS*(1-delta))
+     return(value)
+     } # End of function mloglike
>
> ############
> # Find MLE #
> ############
>
> startvals = c(1,1/2) # I tried a few values
> search1 = optim(par=startvals, fn=mloglike, t=Time,delta=Uncensored,
+                   hessian=TRUE, lower=c(0,0), method='L-BFGS-B')
> # search1
> alphahat = search1$par[1]; lambdahat = search1$par[2]
>
> # Compare true and estimated median
> truealpha = 3; truelambda = 1/5
> H = search1$hessian
> Vhat = solve(H) # Solve returns the inverse.
>
> ##############################################################################
> # Point estimate and confidence interval for the median
> # Median = log(2)^(1/alpha) / lambda
> ##############################################################################
>
> # Point estimate of median
> medhat = 1/lambdahat * log(2)^(1/alphahat); medhat
[1] 4.466034
> # Compare the truth
> truemedian = log(2)^(1/truealpha) / truelambda; truemedian
[1] 4.424985
>
> # Confidence interval for median
> # Need gdot
> # D[b^(1/a),a] works in Wolfram Alpha, as a check on hand calculation.
>
> gdot = cbind( - log(2)^(1/alphahat)*log(log(2))/(lambdahat*alphahat^2),
+               - log(2)^(1/alphahat)/lambdahat^2 )
> v_medhat = as.numeric( gdot %*% Vhat %*% t(gdot) ); se_medhat = sqrt(v_medhat)
> lower95 = medhat - 1.96*se_medhat; upper95 = medhat + 1.96*se_medhat
> c(lower95,upper95)
[1] 4.199471 4.732597
> # For comparison, Kaplan-Meier estimate was
> #      n  events  median 0.95LCL 0.95UCL
> # 275.00  144.00    4.54    4.21    4.91
```

```
> # Estimate the survival function
> x = seq(from=0,to=10,length=101)
> Shat = exp(-(lambdahat*x)^alphahat)
> trueS = exp(-(truelambda*x)^truealpha)
> tstring = 'Survival Function for the Weibull Data'
> plot(x,trueS,type='l',xlab='Time',ylab='Survival',ylim=c(0,1), main=tstring, col = "red1")
> lines(x,Shat,lty=2, col = "blue1")
> # Annotate the plot (Make the legend)
> x1 = c(6,7.5); y1 = c(0.8,0.8)
> lines(x1,y1,lty=1)
> text(8.5,0.8,'True S(t)', col = "red1")
> x2 = x1; y2 = c(0.7,0.7)
> lines(x2,y2,lty=2)
> text(8.9,0.7,'Estimated S(t)', col = "blue1")
```



```
> # Add MLE to Kaplan-Meier plot
> plot(km1)
> title("Kaplan-Meier Estimate for the Weibull Data")
> lines(x,trueS,lty=1, col = "red1")
> lines(x,Shat,lty=2, col = "blue1")
```

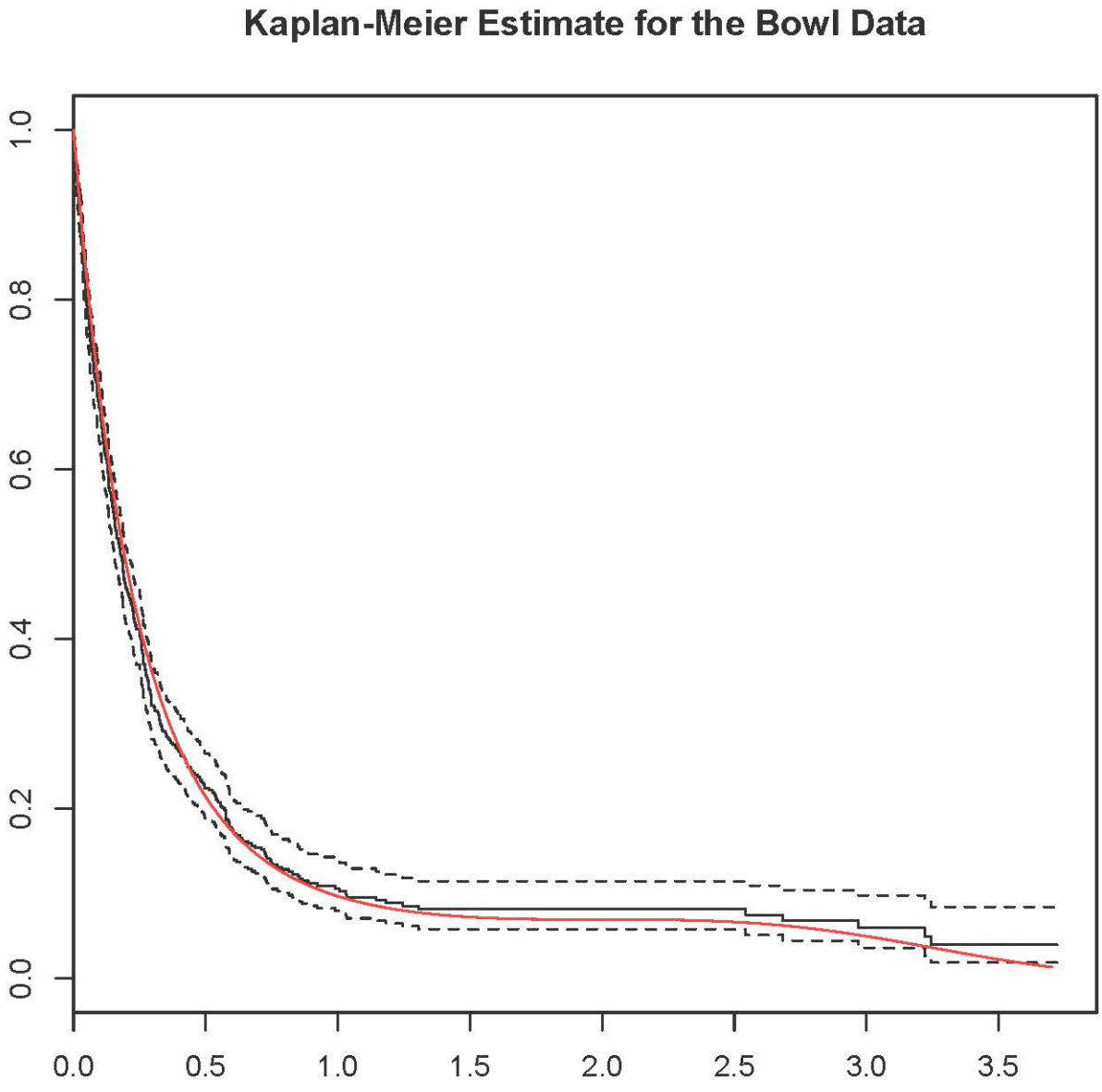


```
> # What if the (Weibull) model is wrong? Bowl (shaped hazard) data
> rm(list=ls()); options(scipen=999)
> bowldat = read.table("http://www.utstat.toronto.edu/brunner/data/legal/bowlhaz.data.txt")
> head(bowldat); summary(bowldat); attach(bowldat)
        Time Uncensored
1 0.72275893          1
2 0.12004774          0
3 0.53171197          1
4 0.05997346          1
5 0.35008144          1
6 0.65936362          1
      Time            Uncensored
 Min.   :0.000144   Min.   :0.000
 1st Qu.:0.062032   1st Qu.:1.000
 Median :0.158124   Median :1.000
 Mean   :0.352768   Mean   :0.862
 3rd Qu.:0.376218   3rd Qu.:1.000
 Max.   :3.722165   Max.   :1.000
>
>
> # install.packages("survival",dependencies=TRUE) # Only need to do this once
> library(survival) # Do this every time
>
> y = Surv(Time,Uncensored) # A pre-processing step
> km2 = survfit(y ~ 1) # A regression model with just an intercept: No x values
> km2
Call: survfit(formula = y ~ 1)

      n  events  median 0.95LCL 0.95UCL
500.000 431.000   0.178   0.154   0.206
> # What is the true median?
> # From HW4, S(t) = exp(-1/3 ((t-2)^3 + 8)
> # Set exp(-1/3 ((t-2)^3 + 8) = 1/2, get t = 0.190935

> # Plot true S(t) and MLE for comparison to K-M
>
> # Weibull minus log likelihood again
> mloglike = function(theta,t,delta)
+     { # Minus log likelihood function for Weibull
+     alpha = theta[1]; lambda = theta[2]
+     # logf and logS will be of length n
+     logf = log(alpha)+log(lambda)+(alpha-1)*log(lambda*t) + -(lambda*t)^alpha
+     logS = -(lambda*t)^alpha
+     value = -sum(logf*delta) - sum(logS*(1-delta))
+     return(value)
+     } # End of function mloglike
>
> # Find MLE
>
> startvals = c(1,1/2)
> search = optim(par=startvals, fn=mloglike, t=Time,delta=Uncensored,
+                     hessian=TRUE, lower=c(0,0), method='L-BFGS-B')
> alphahat = search$par[1]; lambdahat = search$par[2]
>
> # MLE of median
> medhat = 1/lambdahat * log(2)^(1/alphahat); medhat
[1] 0.2030915
> # Compared to truth of 0.191 and K-M estimate of 0.178
> 0.203-0.191 # Error of MLE
[1] 0.012
> 0.203-0.178 # Error of Kaplan-Meier
[1] 0.025
```

```
> plot(km2)
> # Add title and other estimates to Kaplan-Meier plot
> title('Kaplan-Meier Estimate for the Bowl Data')
> lines(x,trueS,lty=1, col = "red1")
```



```
> lines(x,Shat,lty=2, col = "blue1")
>
> # Annotate the plot (Make the legend)
> x1 = c(2,2.75); y1 = c(0.8,0.8)
> lines(x1,y1,lty=1, col = "red1")
> text(3,0.8,'True S(t)', col = "red1")
> x2 = x1; y2 = c(0.7,0.7)
> lines(x2,y2,lty=1)
> text(3.15,0.7,'Kaplan-Meier')
> x3 = x1; y3 = c(0.6,0.6)
> lines(x3,y3,lty=2,  col = "blue1")
> text(3.15,0.6,'Weibull MLE', col = "blue1")
```