Comparison Performance of the Bayesian Approach with the Weibull and Birnbaum-Saunders Distributions in Imputation of Time-to-Event Censors

Parviz Shahmirzalou

2023-11-20

R and OpenBUGS Code for the paper:

1-Run the Bayesian Approach with the Weibull distribution.

```
    rm(list = ls())
  # Install packages:survival & R2openBUGS.
    library(survival)
    library(R2OpenBUGS)
    library(coda)
  # Set working directory and modelfile.
    getwd()
```

```
## [1] "C:/Users/novingostar/Documents/R-studio"
```

```
    bugswd = paste0(getwd(),"/bugswd"); bugswd
```

```
## [1] "C:/Users/novingostar/Documents/R-studio/bugswd"
```

```
    modelfile = paste0(bugswd,"/modelfile.txt"); modelfile
```

```
## [1] "C:/Users/novingostar/Documents/R-studio/bugswd/modelfile.txt"
```

```
  # Generate Data
    set.seed(12345)
    n = 200  # n=100; 200; 300
    x = rep(0:1, c(0.50*n, 0.50*n))  # Weibull scale parameter related to x.
    table(x)
```

```
## x
##   0   1 
## 100 100
```

```
    shape = 2  # Shape: 0.5; 1; 2
    b = c(-3, 0.3) # set b1 and b2 with table 2 in the paper.
    lambda = exp(b[1] + b[2]*x)   # Link the parameter to covariate x .
    summary(lambda)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
## 0.04979 0.04979 0.05850 0.05850 0.06721 0.06721
```

```
    scale = lambda^(-1/shape)   # Since weibull formulla in winbugs is different to R, we need t
o convert
                              # formula to get similar results.
    summary(scale)   # Mean scale parameter is near to 4.
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##   3.857   3.857   4.170   4.170   4.482   4.482
```

```
  #Generate Observed time
    y = rweibull(n,shape, scale )
    summary(y)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##  0.3144  1.9513  3.1116  3.4790  4.6081 11.6694
```

```
    range(y)
```

```
## [1]  0.3144349 11.6693801
```

```
  # Generate censored time
    delta1 = rep(1,n)   # to make censored data
    cen = rexp(n,0.06)             # Censored time
    delta = as.numeric(y < cen)
    cenper = 1 - mean(delta); cenper   # Get percent of censoring
```

```
## [1] 0.205
```

```
  # Merge observed and censored time.
    z = pmin(y,cen)  # to select observed time or censored time. Every one that is lesser than o
ther.
  # make variable "t" as observed time and variable "c" as censored time to use in BUGS.
    t <- ifelse(delta == 1, z, NA)
    c <- ifelse(delta == 1, 0, z)
  # Run model in BUGS.
      modeltext = "model {
      for(i in 1:n){
        t[i] ~ dweib(shape,lambda[i])C(c[i], )
        log(lambda[i]) <- b[1]+b[2]*x[i]
        cim[i] <- step(c[i]-1.0E-5)*pow(log(2)/lambda[i]+pow(c[i],shape), 1/shape)
        }
        # priors
        shape ~ dgamma(0.01,0.01)  # Non-informative prior
        for(j in 1:2) {b[j]~dnorm(0,0.01)}
      }
      "
      # write BUGS output into file.
      cat(modeltext, file = modelfile) #file.show(modelfile)
      modeldata = list(n = n, x = x, t = t, c = c)
      modelinit = list(list(b = rep(0,length(b)), shape = shape))
      param = c("shape","b","cim")
      # bugs ----------------------------------------
      bugsOut <- bugs(
        working.directory = bugswd,
        model.file = modelfile,
        data = modeldata,
        inits = modelinit,
        #inits = NULL,
        parameters.to.save = param,
        n.chains = 1,
        n.iter = 11000,
        n.burnin = 1000,
        n.thin = 20
        #, debug = TRUE
        #, codaPkg = TRUE
      )
  # output ----------------------------------------
    bugsOut$DIC
```

```
## [1] 680
```

```
    # Which records is censored:
    ic = which(delta==0); ic; length(ic)
```

```
##  [1]   2   7  14  15  18  22  24  33  38  44  46  53  54  73  74  77  83  85  86
## [20]  88  90  93 100 101 102 111 112 113 118 126 132 144 157 161 162 165 172 180
## [39] 187 190 196
## [1] 41
```

```
    # Dimension of output:
    dim(bugsOut$sims.array)
```

```
## [1] 10000     1   204
```

```
  # Describe censored simulations.
    bugsOut$summary[c(1:3,3+ic),c(1,2)]
```

```
##                mean        sd
## shape     1.805375 0.1092916
## b[1]     -2.734371 0.2141880
## b[2]      0.438219 0.1611971
## cim[2]    4.075876 0.2210809
## cim[7]    4.285282 0.2124492
## cim[14]   3.967729 0.2269040
## cim[15]   5.828252 0.1857823
## cim[18]   3.963575 0.2271502
## cim[22]   4.580935 0.2038736
## cim[24]   3.783857 0.2407708
## cim[33]   3.726482 0.2473757
## cim[38]   3.895677 0.2315628
## cim[44]   3.744611 0.2450177
## cim[46]   4.092529 0.2202821
## cim[53]   3.728120 0.2471460
## cim[54]   3.799501 0.2392711
## cim[73]   5.391367 0.1903302
## cim[74]   5.654796 0.1874578
## cim[77]   3.725392 0.2475333
## cim[83]   8.757314 0.1676079
## cim[85]   7.251766 0.1755527
## cim[86]   5.844715 0.1856372
## cim[88]   3.775350 0.2416208
## cim[90]   5.737278 0.1866526
## cim[93]   4.646813 0.2023413
## cim[100]  4.917894 0.1970473
## cim[101]  3.911482 0.1486143
## cim[102]  3.152224 0.1703693
## cim[111]  3.776528 0.1508518
## cim[112]  2.955111 0.1851738
## cim[113]  3.628099 0.1538315
## cim[118]  2.925447 0.1888321
## cim[126]  3.325505 0.1625607
## cim[132]  3.273161 0.1646200
## cim[144]  3.819013 0.1501085
## cim[157]  2.923859 0.1890650
## cim[161]  3.163178 0.1697829
## cim[162]  4.003277 0.1472827
## cim[165]  4.722583 0.1399990
## cim[172]  5.707235 0.1338404
## cim[180]  3.199393 0.1679400
## cim[187]  4.990766 0.1380773
## cim[190]  3.935618 0.1482457
## cim[196]  8.943724 0.1199796
```

```
    # Describe parameter simulations:
    parsim1 = bugsOut$sims.array[,1,1:3]   #parameter simulation
    parsim1[1:5,]  # Only five rows of 10.000 simulation for parameters.
```

```
##      shape   b[1]   b[2]
## [1,] 1.824 -2.771 0.5888
## [2,] 1.581 -2.200 0.3066
## [3,] 1.796 -2.566 0.3453
## [4,] 1.673 -2.460 0.2894
## [5,] 1.827 -2.790 0.5168
```

```
  # print median of simulations for every censor that replaced.
    bugsOut$median$cim[ic]
```

```
##  [1] 4.0630 4.2725 3.9560 5.8150 3.9520 4.5670 3.7750 3.7200 3.8840 3.7370
## [11] 4.0800 3.7210 3.7900 5.3780 5.6420 3.7190 8.7440 7.2380 5.8310 3.7660
## [21] 5.7240 4.6330 4.9040 3.9035 3.1460 3.7700 2.9510 3.6200 2.9210 3.3190
## [31] 3.2670 3.8120 2.9200 3.1570 3.9950 4.7140 5.6990 3.1930 4.9830 3.9270
## [41] 8.9330
```

```
  #
```

Convergence: Geweke diagnostics

```
  geweke.diag(parsim1, frac=0.10, frac2 = 0.50)   #Z-score
```

```
## 
## Fraction in 1st window = 0.1
## Fraction in 2nd window = 0.5 
## 
##     shape      b[1]      b[2] 
## -0.006788 -0.034990 -1.148750
```

Generate and save necessary files:

```
write.csv(parsim1, file = "matparsim1.csv")
mcmcparsim1 <- mcmc(as.matrix(parsim1))
```

ACF computations

```
autocorr.diag(mcmcparsim1)
```

```
##                 shape         b[1]         b[2]
## Lag 0   1.0000000000  1.000000000  1.000000000
## Lag 1   0.0930385007  0.088655107 -0.004490993
## Lag 5  -0.0003997621  0.003012278  0.011816211
## Lag 10 -0.0132304570 -0.011621838 -0.011823830
## Lag 50  0.0076258184  0.004488152 -0.008822238
```

Effective Sample Size (ESS)

```
  effectiveSize(parsim1)
```

```
##     shape      b[1]      b[2] 
##  8364.658  8413.885 10000.000
```

Figures 8 in the paper.

```
  # Kaplan-Meier Curve:
  curve1 = survfit(Surv(z,delta) ~ x); curve1
```

```
## Call: survfit(formula = Surv(z, delta) ~ x)
## 
##       n events median 0.95LCL 0.95UCL
## x=0 100     77   3.66    3.07    4.12
## x=1 100     82   2.94    2.32    3.71
```

```
  plot(curve1, mark.time = TRUE,lty = 1,conf.int = FALSE,  col = "black",
       main = paste("t~Weibull(2,4), c~Exp(0.06), p=0.20, n=200") )

  # Curve with Median of Simulated Times
    # output ---------------------------------------
    # imputation      h=hat
    bh = bugsOut$mean$b; bh
```

```
## [1] -2.734371  0.438219
```

```
    shapeh = bugsOut$mean$shape; shapeh
```

```
## [1] 1.805375
```

```
    lambdah = exp(bh[1] + bh[2]*x); lambdah #every person has specific lambda because it has spe
cific X.
```

```
##   [1] 0.06493486 0.06493486 0.06493486 0.06493486 0.06493486 0.06493486
##   [7] 0.06493486 0.06493486 0.06493486 0.06493486 0.06493486 0.06493486
##  [13] 0.06493486 0.06493486 0.06493486 0.06493486 0.06493486 0.06493486
##  [19] 0.06493486 0.06493486 0.06493486 0.06493486 0.06493486 0.06493486
##  [25] 0.06493486 0.06493486 0.06493486 0.06493486 0.06493486 0.06493486
##  [31] 0.06493486 0.06493486 0.06493486 0.06493486 0.06493486 0.06493486
##  [37] 0.06493486 0.06493486 0.06493486 0.06493486 0.06493486 0.06493486
##  [43] 0.06493486 0.06493486 0.06493486 0.06493486 0.06493486 0.06493486
##  [49] 0.06493486 0.06493486 0.06493486 0.06493486 0.06493486 0.06493486
##  [55] 0.06493486 0.06493486 0.06493486 0.06493486 0.06493486 0.06493486
##  [61] 0.06493486 0.06493486 0.06493486 0.06493486 0.06493486 0.06493486
##  [67] 0.06493486 0.06493486 0.06493486 0.06493486 0.06493486 0.06493486
##  [73] 0.06493486 0.06493486 0.06493486 0.06493486 0.06493486 0.06493486
##  [79] 0.06493486 0.06493486 0.06493486 0.06493486 0.06493486 0.06493486
##  [85] 0.06493486 0.06493486 0.06493486 0.06493486 0.06493486 0.06493486
##  [91] 0.06493486 0.06493486 0.06493486 0.06493486 0.06493486 0.06493486
##  [97] 0.06493486 0.06493486 0.06493486 0.06493486 0.10064542 0.10064542
## [103] 0.10064542 0.10064542 0.10064542 0.10064542 0.10064542 0.10064542
## [109] 0.10064542 0.10064542 0.10064542 0.10064542 0.10064542 0.10064542
## [115] 0.10064542 0.10064542 0.10064542 0.10064542 0.10064542 0.10064542
## [121] 0.10064542 0.10064542 0.10064542 0.10064542 0.10064542 0.10064542
## [127] 0.10064542 0.10064542 0.10064542 0.10064542 0.10064542 0.10064542
## [133] 0.10064542 0.10064542 0.10064542 0.10064542 0.10064542 0.10064542
## [139] 0.10064542 0.10064542 0.10064542 0.10064542 0.10064542 0.10064542
## [145] 0.10064542 0.10064542 0.10064542 0.10064542 0.10064542 0.10064542
## [151] 0.10064542 0.10064542 0.10064542 0.10064542 0.10064542 0.10064542
## [157] 0.10064542 0.10064542 0.10064542 0.10064542 0.10064542 0.10064542
## [163] 0.10064542 0.10064542 0.10064542 0.10064542 0.10064542 0.10064542
## [169] 0.10064542 0.10064542 0.10064542 0.10064542 0.10064542 0.10064542
## [175] 0.10064542 0.10064542 0.10064542 0.10064542 0.10064542 0.10064542
## [181] 0.10064542 0.10064542 0.10064542 0.10064542 0.10064542 0.10064542
## [187] 0.10064542 0.10064542 0.10064542 0.10064542 0.10064542 0.10064542
## [193] 0.10064542 0.10064542 0.10064542 0.10064542 0.10064542 0.10064542
## [199] 0.10064542 0.10064542
```

```
    scaleh = lambdah^(-1/shapeh); scaleh
```

```
##   [1] 4.547476 4.547476 4.547476 4.547476 4.547476 4.547476 4.547476 4.547476
##   [9] 4.547476 4.547476 4.547476 4.547476 4.547476 4.547476 4.547476 4.547476
##  [17] 4.547476 4.547476 4.547476 4.547476 4.547476 4.547476 4.547476 4.547476
##  [25] 4.547476 4.547476 4.547476 4.547476 4.547476 4.547476 4.547476 4.547476
##  [33] 4.547476 4.547476 4.547476 4.547476 4.547476 4.547476 4.547476 4.547476
##  [41] 4.547476 4.547476 4.547476 4.547476 4.547476 4.547476 4.547476 4.547476
##  [49] 4.547476 4.547476 4.547476 4.547476 4.547476 4.547476 4.547476 4.547476
##  [57] 4.547476 4.547476 4.547476 4.547476 4.547476 4.547476 4.547476 4.547476
##  [65] 4.547476 4.547476 4.547476 4.547476 4.547476 4.547476 4.547476 4.547476
##  [73] 4.547476 4.547476 4.547476 4.547476 4.547476 4.547476 4.547476 4.547476
##  [81] 4.547476 4.547476 4.547476 4.547476 4.547476 4.547476 4.547476 4.547476
##  [89] 4.547476 4.547476 4.547476 4.547476 4.547476 4.547476 4.547476 4.547476
##  [97] 4.547476 4.547476 4.547476 4.547476 3.567418 3.567418 3.567418 3.567418
## [105] 3.567418 3.567418 3.567418 3.567418 3.567418 3.567418 3.567418 3.567418
## [113] 3.567418 3.567418 3.567418 3.567418 3.567418 3.567418 3.567418 3.567418
## [121] 3.567418 3.567418 3.567418 3.567418 3.567418 3.567418 3.567418 3.567418
## [129] 3.567418 3.567418 3.567418 3.567418 3.567418 3.567418 3.567418 3.567418
## [137] 3.567418 3.567418 3.567418 3.567418 3.567418 3.567418 3.567418 3.567418
## [145] 3.567418 3.567418 3.567418 3.567418 3.567418 3.567418 3.567418 3.567418
## [153] 3.567418 3.567418 3.567418 3.567418 3.567418 3.567418 3.567418 3.567418
## [161] 3.567418 3.567418 3.567418 3.567418 3.567418 3.567418 3.567418 3.567418
## [169] 3.567418 3.567418 3.567418 3.567418 3.567418 3.567418 3.567418 3.567418
## [177] 3.567418 3.567418 3.567418 3.567418 3.567418 3.567418 3.567418 3.567418
## [185] 3.567418 3.567418 3.567418 3.567418 3.567418 3.567418 3.567418 3.567418
## [193] 3.567418 3.567418 3.567418 3.567418 3.567418 3.567418 3.567418 3.567418
```

```
    # Compute median of Simulations.
    zmed = qweibull(.5*pweibull(cen,shapeh,scaleh, lower.tail = FALSE),shapeh, scaleh, lower.tai
l = FALSE)

    zimp = rep(NA,n)
    zimp[ic] = zmed[ic]
    zimp[-ic] = z[-ic]  # zimp = failure times+imputed censored times

    curve2 = survfit(Surv(zimp,delta1) ~ x); curve2     # Bayesian Imputation
```

```
## Call: survfit(formula = Surv(zimp, delta1) ~ x)
##
##       n events median 0.95LCL 0.95UCL
## x=0 100    100   3.74    3.65    3.98
## x=1 100    100   3.11    2.59    3.53
```

```
    lines(curve2, mark.time = TRUE, col = "Blue", lty = 1)

  #Curve without Censored Times
    tOC = z[delta==1]   #time omitting censored
    deltaOC = rep(1, length(tOC))
    curve3 = survfit(Surv(tOC, deltaOC) ~ x[delta==1]); curve3       # Omitting_Censored
```

```
## Call: survfit(formula = Surv(tOC, deltaOC) ~ x[delta == 1])
##
##                    n events median 0.95LCL 0.95UCL
## x[delta == 1]=0 77     77   3.20    2.68    3.94
## x[delta == 1]=1 82     82   2.53    2.07    3.25
```

```
    lines(curve3, mark.time = TRUE, col = "Red", lty = 1)

    legend("topright", c("Kaplan-Meier Curve", "Curve with Median of Simulated Times", "Curve without Censored Times"),
           lty= 1, col = c("black", "Blue", "Red"), cex = 0.7)
```

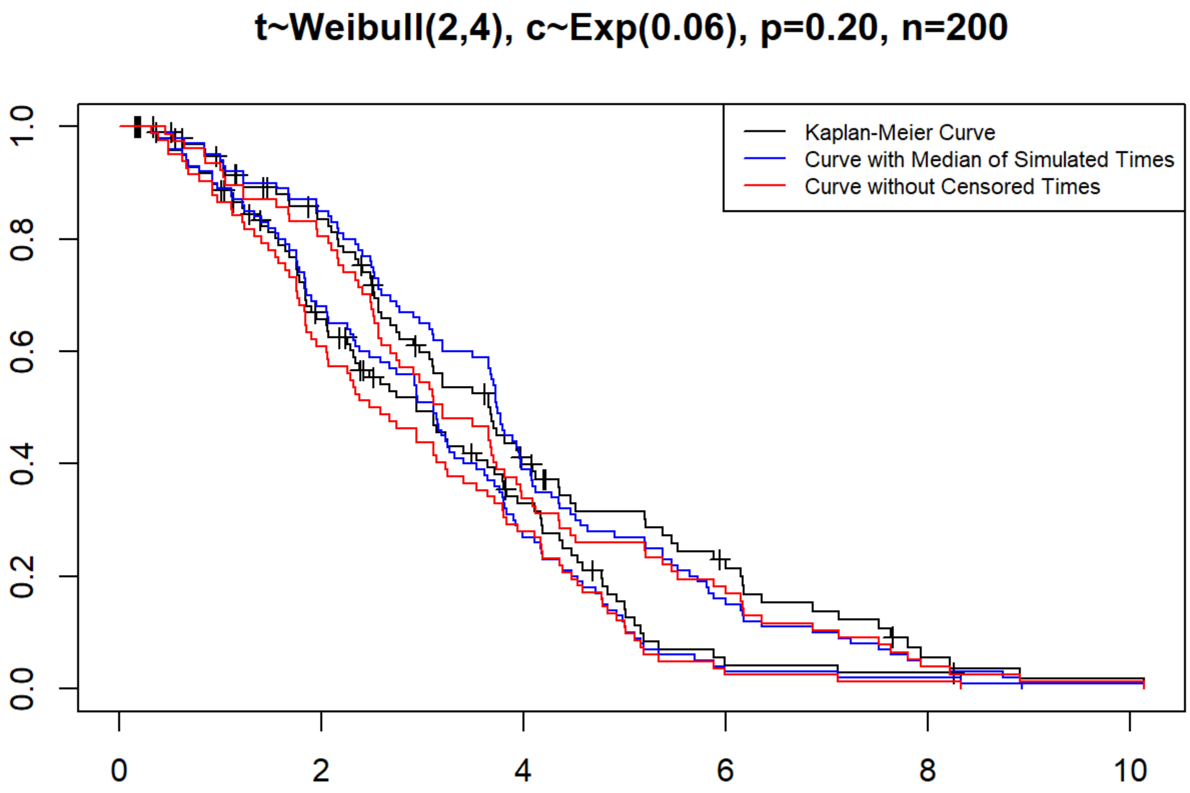
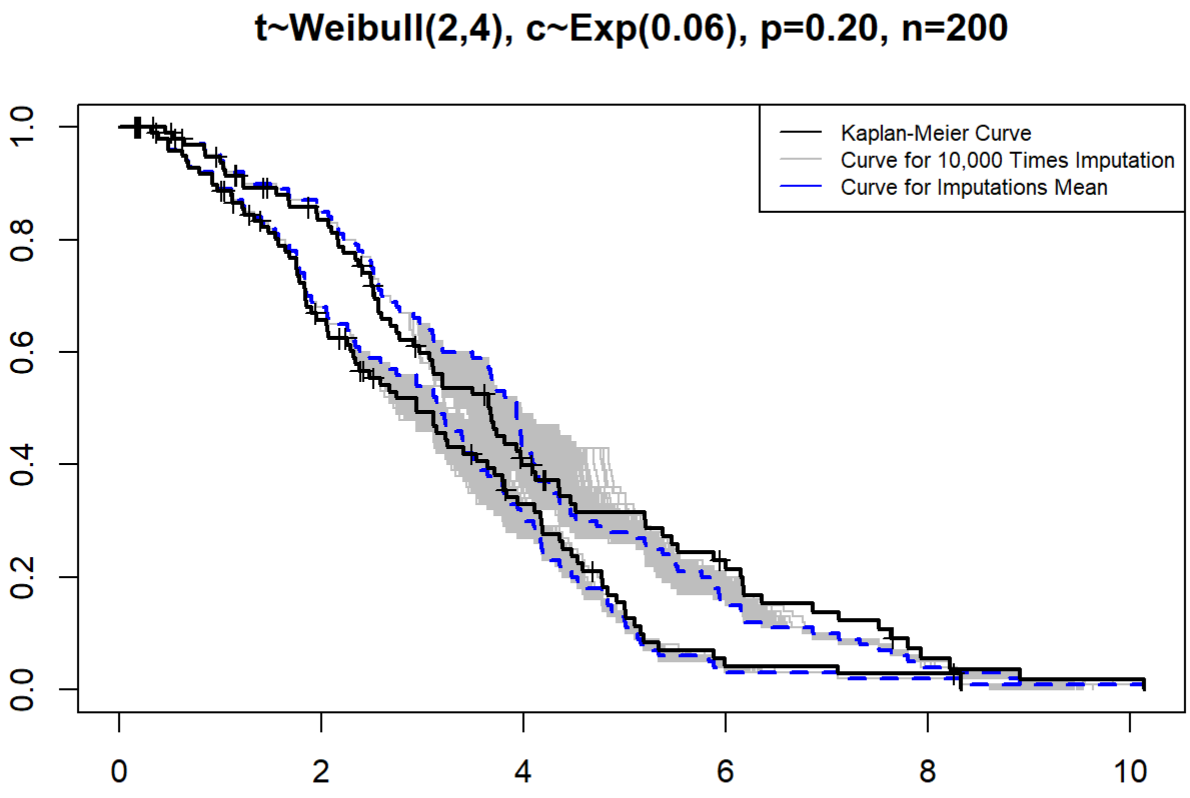


Figure 9 in the paper.

```
  # Kaplan-Meier Curve:
  km1 = survfit(Surv(z,delta) ~ x); km1
```

```
## Call: survfit(formula = Surv(z, delta) ~ x)
##
##       n events median 0.95LCL 0.95UCL
## x=0 100     77   3.66    3.07    4.12
## x=1 100     82   2.94    2.32    3.71
```

```
plot(km1, mark.time = TRUE,lty = 1, lwd =2, col = "black",
     main = paste("t~Weibull(2,4), c~Exp(0.06), p=0.20, n=200"))  #KM_Estimation

# Curve for 10,000 Times Imputation.
timp=t
impsim = bugsOut$sims.array[,1,3+ic]
for (i in 1:nrow(impsim)) {
  timp[ic] <- impsim[i,]
  kmi = survfit(Surv(timp,delta1) ~ x)
  lines(kmi, mark.time = TRUE, col = "gray", lty = 1)    # n time Imputation

}
# Curve for Imputations Mean
timp[ic] <- colMeans(impsim)
kmmean = survfit(Surv(timp,delta1) ~ x)
lines(kmi, mark.time = TRUE, col = "blue", lty = 2, lwd = 2)  # Mean of n times Imputation

lines(km1, mark.time = TRUE,lty = 1, lwd =2, col = "black",
     main = paste("t~Weibull(2,4), c~Exp(0.20), p=0.50, n=200"))  #KM_Estimation


legend("topright",
       c("Kaplan-Meier Curve", "Curve for 10,000 Times Imputation", "Curve for Imputations Mea
n"),
       lty = 1, col = c("Black", "gray","blue"), cex = .7)
```



2-Run the Bayesian Approach with the Birnbaum-Saunders (BS) distribution.

```
    library(survival)
    library(R2OpenBUGS)
    library(coda)

    # Set working directory and modelfile.
    getwd()
```

```
## [1] "C:/Users/novingostar/Documents/R-studio"
```

```
    bugswd = paste0(getwd(),"/bugswd"); bugswd
```

```
## [1] "C:/Users/novingostar/Documents/R-studio/bugswd"
```

```
    modelfile = paste0(bugswd,"/modelfile.txt"); modelfile
```

```
## [1] "C:/Users/novingostar/Documents/R-studio/bugswd/modelfile.txt"
```

```
    # generate Data
    set.seed(12345)
    n = 200 # n=100; 200; 300
    x = rep(0:1, c(0.50*n, 0.50*n))  # BS scale parameter related to x.
    table(x)
```

```
## x
##   0   1
## 100 100
```

```
    shape = 2    # Shape: 0.5; 1; 2
    b = c(1.37, 0.15)  # set b1 and b2 with table 4 in the paper.
    lambda = exp(b[1] + b[2]*x)
    summary(lambda)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   3.935   3.935   4.254   4.254   4.572   4.572
```

```
    scale = lambda
    # Define rbn to generate numbers from BS distribution.
    rbn <- function(n, shape, scale){    # shape = a, scale = b
      x <- rnorm(n, 0, shape/2)
      t <- scale * (1 + 2 * x^2 + 2 * x * sqrt(1 + x^2))
      return(t)
    }
    #Generate Observed time
    y <- rbn(n, shape, lambda)
    # Generate censored time
    delta1 = rep(1,n)
  ## Important Note: Censoring Percent: try to set >21%
    cen = rexp(n,0.022)
    delta = as.numeric(y < cen)
    cenper = 1 - mean(delta); cenper   # % censoring
```

```
## [1] 0.21
```

```
# Merge observed and censored time.
    z = pmin(y,cen)
    # make variable "t" as observed time and variable "c" as censored time to use in BUGS.
    t <- ifelse(delta == 1, z, NA)
    c <- ifelse(delta == 1, 0, z)
    # Run model in BUGS.
    modeltext = "model {
  for(i in 1:n){
    t[i] ~ dbs(shape, lambda[i])C(c[i], )
    log(lambda[i]) <- b[1]+b[2]*x[i]
    cim[i] <- step(c[i]-1.0E-5)*lambda[i]

    }
    # priors
    shape ~ dgamma(0.01,0.01)
    for(j in 1:2) {b[j]~dnorm(0,0.01)}
  }
  "
    # write BUGS output into file.
    cat(modeltext, file = modelfile) #file.show(modelfile)
    modeldata = list(n = n, x = x, t = t, c = c)
    modelinit = list(list(b = rep(0,length(b)), shape = shape))
    param = c("shape","b","cim")
    # bugs ----------------------------------------
    bugsOut <- bugs(
      working.directory = bugswd,
      model.file = modelfile,
      data = modeldata,
      inits = modelinit,
      #inits = NULL,
      parameters.to.save = param,
      n.chains = 1,
      n.iter = 11000,
      n.burnin = 1000,
      n.thin = 20
      #, debug = TRUE
      #, codaPkg = TRUE
    )
    # output ----------------------------------------
    bugsOut$DIC
```

```
## [1] 862.8
```

```
    # Which records is censored:
    ic = which(delta==0); ic
```

```
##  [1]   5   7  12  13  21  22  31  32  34  39  44  46  52  58  61  68  75  84  86
## [20]  93  94 103 108 119 122 124 131 135 142 148 152 157 162 166 168 169 172 174
## [39] 177 185 187 193
```

```
    # Dimension of output:
    dim(bugsOut$sims.array)
```

```
## [1] 10000     1   204
```

```
    # Describe censored simulations.
    bugsOut$summary[c(1:3,3+ic),c(1,2)]
```

```
##                  mean        sd
## shape    2.2002689000 0.1315064
## b[1]     1.6681904000 0.1425496
## b[2]     0.0006914509 0.2169084
## cim[5]   5.3567396000 0.7681659
## cim[7]   5.3567396000 0.7681659
## cim[12]  5.3567396000 0.7681659
## cim[13]  5.3567396000 0.7681659
## cim[21]  5.3567396000 0.7681659
## cim[22]  5.3567396000 0.7681659
## cim[31]  5.3567396000 0.7681659
## cim[32]  5.3567396000 0.7681659
## cim[34]  5.3567396000 0.7681659
## cim[39]  5.3567396000 0.7681659
## cim[44]  5.3567396000 0.7681659
## cim[46]  5.3567396000 0.7681659
## cim[52]  5.3567396000 0.7681659
## cim[58]  5.3567396000 0.7681659
## cim[61]  5.3567396000 0.7681659
## cim[68]  5.3567396000 0.7681659
## cim[75]  5.3567396000 0.7681659
## cim[84]  5.3567396000 0.7681659
## cim[86]  5.3567396000 0.7681659
## cim[93]  5.3567396000 0.7681659
## cim[94]  5.3567396000 0.7681659
## cim[103] 5.3822043000 0.9174185
## cim[108] 5.3822043000 0.9174185
## cim[119] 5.3822043000 0.9174185
## cim[122] 5.3822043000 0.9174185
## cim[124] 5.3822043000 0.9174185
## cim[131] 5.3822043000 0.9174185
## cim[135] 5.3822043000 0.9174185
## cim[142] 5.3822043000 0.9174185
## cim[148] 5.3822043000 0.9174185
## cim[152] 5.3822043000 0.9174185
## cim[157] 5.3822043000 0.9174185
## cim[162] 5.3822043000 0.9174185
## cim[166] 5.3822043000 0.9174185
## cim[168] 5.3822043000 0.9174185
## cim[169] 5.3822043000 0.9174185
## cim[172] 5.3822043000 0.9174185
## cim[174] 5.3822043000 0.9174185
## cim[177] 5.3822043000 0.9174185
## cim[185] 5.3822043000 0.9174185
## cim[187] 5.3822043000 0.9174185
## cim[193] 5.3822043000 0.9174185
```

```
    # Describe parameter simulations:
    parsim2 = bugsOut$sims.array[,1,1:3]    #parameter simulation
    parsim2[1:5,]  # Only five rows of 10.000 simulation for parameters.
```

```
##      shape  b[1]     b[2]
## [1,] 2.462 1.736  0.28230
## [2,] 2.294 1.631  0.04748
## [3,] 2.397 1.879  0.10590
## [4,] 2.316 1.573  0.06742
## [5,] 2.160 1.616 -0.18890
```

Convergence: Geweke diagnostics

```
  geweke.diag(parsim2, frac=0.10, frac2 = 0.50)   #Z-score
```

```
## 
## Fraction in 1st window = 0.1
## Fraction in 2nd window = 0.5 
## 
##   shape    b[1]    b[2] 
## -0.8658  0.3071  0.3815
```

Generate and save necessary files:

```
write.csv(parsim2, file = "matparsim2.csv")
mcmcparsim2 <- mcmc(as.matrix(parsim2))
```

ACF computations

```
autocorr.diag(mcmcparsim2)
```

```
##                shape         b[1]          b[2]
## Lag 0   1.000000000  1.000000000  1.0000000000
## Lag 1   0.015139853  0.028866847  0.0065451146
## Lag 5  -0.008310886  0.005970384 -0.0053163050
## Lag 10 -0.001456997 -0.002174590  0.0003295209
## Lag 50  0.015130165  0.002798546  0.0131302866
```

Effective Sample Size (ESS)

```
  effectiveSize(parsim2)
```

```
##     shape      b[1]      b[2] 
##  9700.749  9437.917 10000.000
```

Figures 10 in the paper.

```
    # Kaplan-Meier Curve:
    curve1 = survfit(Surv(z,delta) ~ x); curve1
```

```
## Call: survfit(formula = Surv(z, delta) ~ x)
## 
##       n events median 0.95LCL 0.95UCL
## x=0 100     79  10.62    4.38   13.26
## x=1 100     79   4.91    3.68    8.27
```

```
    plot(curve1, mark.time = TRUE,lty = 1,conf.int = FALSE,  col = "black",
         main = paste("t~BS(2,4), c~Exp(0.02), p=0.20, n=200") )  #KM_Estimation
    # Curve with Median of Simulated Times
    # output ----------------------------------------
    # imputation      h=hat
    bh = bugsOut$mean$b; bh
```

```
## [1] 1.6681904000 0.0006914509
```

```
    shapeh = bugsOut$mean$shape; shapeh
```

```
## [1] 2.200269
```

```
    lambdah = exp(bh[1] + bh[2]*x); lambdah #every person has specific lambda because it has specific X.
```

```
##   [1] 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564
##   [9] 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564
##  [17] 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564
##  [25] 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564
##  [33] 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564
##  [41] 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564
##  [49] 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564
##  [57] 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564
##  [65] 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564
##  [73] 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564
##  [81] 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564
##  [89] 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564
##  [97] 5.302564 5.302564 5.302564 5.302564 5.306231 5.306231 5.306231 5.306231
## [105] 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231
## [113] 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231
## [121] 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231
## [129] 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231
## [137] 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231
## [145] 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231
## [153] 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231
## [161] 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231
## [169] 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231
## [177] 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231
## [185] 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231
## [193] 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231
```

```
    scaleh = lambdah; scaleh
```

```
##   [1] 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564
##   [9] 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564
##  [17] 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564
##  [25] 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564
##  [33] 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564
##  [41] 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564
##  [49] 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564
##  [57] 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564
##  [65] 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564
##  [73] 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564
##  [81] 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564
##  [89] 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564 5.302564
##  [97] 5.302564 5.302564 5.302564 5.302564 5.306231 5.306231 5.306231 5.306231
## [105] 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231
## [113] 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231
## [121] 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231
## [129] 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231
## [137] 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231
## [145] 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231
## [153] 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231
## [161] 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231
## [169] 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231
## [177] 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231
## [185] 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231
## [193] 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231 5.306231
```

```
    #install.packages("extraDistr")
    library(extraDistr)
    # Compute median of Simulations.
    zmed = qfatigue(.5*pfatigue(cen,shapeh,scaleh, mu = 0, lower.tail = FALSE),shapeh, scaleh,mu
= 0, lower.tail = FALSE)
    zmed
```

```
##   [1] 134.205944  64.874889  27.055870 104.381854  24.950928 136.464762
##   [7]  17.738646  42.401502 171.116799 218.608897  45.096489  48.381503
##  [13]  17.044121  84.457400  61.019161  60.220565 118.161954  56.864627
##  [19]  84.652535  56.277845  26.689440  30.935836 166.957613 173.698546
##  [25]  22.513073 191.402215  16.879631  52.412156  52.607853 239.972084
##  [31]  38.462174  70.608788 257.120302  47.226316  86.785065  90.108624
##  [37]  54.054967   6.510887  39.990704  25.938337 146.524839  15.324049
##  [43]  23.874841  22.315169  86.285126  28.274805  82.150338  53.079238
##  [49]  65.345132  70.940246  90.086957  49.011204  33.132254  58.795896
##  [55]  69.480867  44.450148  48.986426  27.328536 132.325739  63.900188
##  [61]  15.939196  86.170517  33.599669  98.532420  35.429713  49.425857
##  [67]  53.300658  29.690226  46.970480  48.830017  23.390886  56.068628
##  [73]  44.374546 134.408960  32.480211  85.876032  45.894622  44.969426
##  [79]  81.233232  74.767943  37.179975  71.538304  38.546656  95.445427
##  [85]  50.048458  47.418871 118.578517  34.797140  66.695114  67.577085
##  [91]  23.582286  34.461744  78.134789  11.183721  34.636065  38.708878
##  [97]  54.115477 132.702028  87.971306  59.276524  70.972659 109.230380
## [103]  16.876086 194.090515  30.824539  28.040675  47.741876  34.824050
## [109]  26.043102 108.436798 142.434142  38.309425  78.680370 126.351196
## [115] 128.516202  22.168812  38.314223 116.970974  59.874117  38.074413
## [121] 112.164076  17.506382  99.684282  55.330007 108.706912  73.817769
## [127]  49.306269 102.578566  40.221771  35.035486  42.996920 106.369720
## [133] 185.792528  23.453782 124.956375  60.085985  38.088675 113.893946
## [139]  49.651949  40.356362 236.445019   6.100654  44.028064  72.929669
## [145]  53.842948  35.249913  62.512407  22.897771 176.291528  81.118232
## [151]  28.171615  21.628120  24.385252  55.076828  18.933582  30.531713
## [157]  63.380091 127.475452 137.477517  60.692961  98.702834   8.448115
## [163]  43.618042  70.153808  56.550769   7.093004  38.374958   6.573409
## [169]  54.283954  72.584013  38.396037  30.116215  34.590061  24.745177
## [175]  55.708054  99.065436  15.046242  33.814951  58.781941  35.254390
## [181]  24.998819  33.490179 217.469909 125.842397  22.690373  72.939882
## [187]  37.768278 132.157468  18.121001  54.805270  36.955311  40.784095
## [193]  35.626071  60.886967 179.736178 264.891750  47.075563  49.690388
## [199]  69.298235  66.714592
```

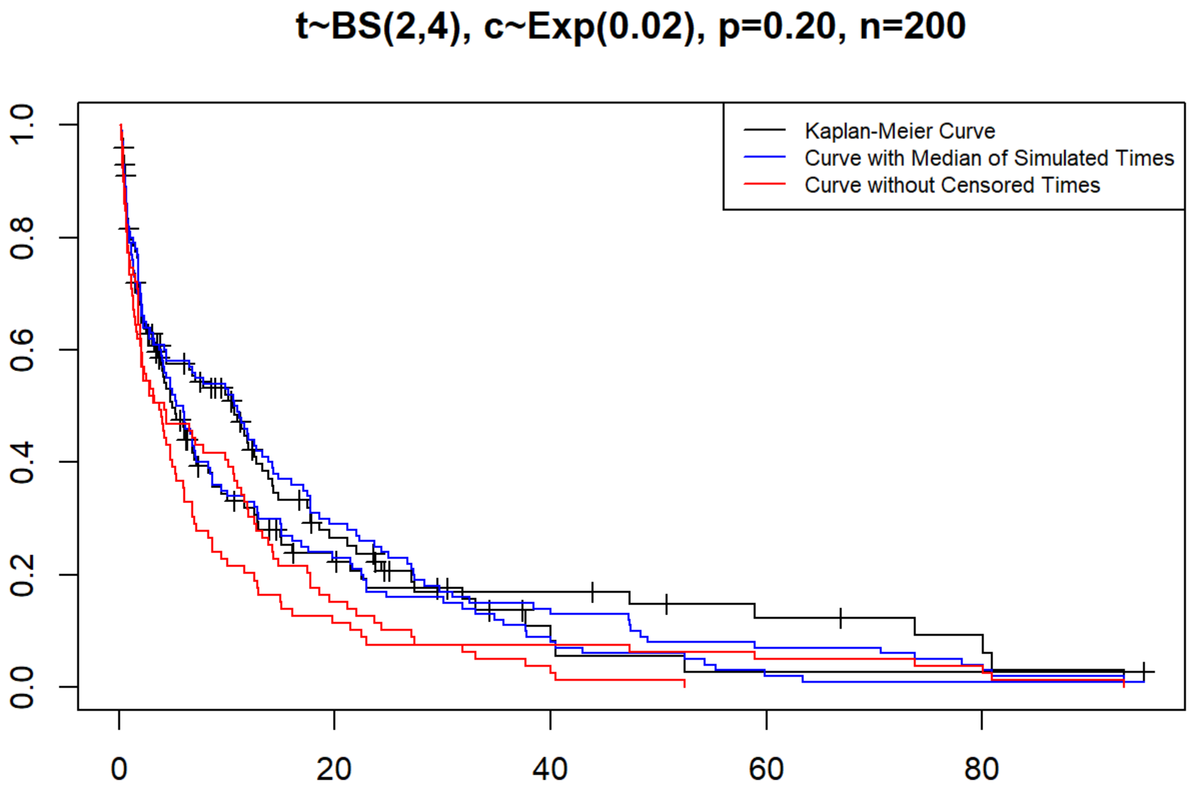
```
    # Make a variable include median of simulations.
    zimp = rep(NA,n)
    zimp[ic] = zmed[ic]
    zimp[-ic] = z[-ic]  # zimp = failure times+imputed censored times



    #
    curve2 = survfit(Surv(zimp,delta1) ~ x); curve2
```

```
## Call: survfit(formula = Surv(zimp, delta1) ~ x)
##
##       n events median 0.95LCL 0.95UCL
## x=0 100    100  10.84    4.39   14.22
## x=1 100    100   5.62    3.98    8.27
```

```
    lines(curve2, mark.time = TRUE, col = "Blue", lty = 1)

    #Curve without Censored Times
    tOC = z[delta==1]
    deltaOC = rep(1, length(tOC))
    curve3 = survfit(Surv(tOC, deltaOC) ~ x[delta==1]); curve3
```

```
## Call: survfit(formula = Surv(tOC, deltaOC) ~ x[delta == 1])
##
##                   n events median 0.95LCL 0.95UCL
## x[delta == 1]=0 79     79   4.14    2.08   10.68
## x[delta == 1]=1 79     79   3.68    2.01    5.33
```

```
    lines(curve3, mark.time = TRUE, col = "Red", lty = 1)

    legend("topright", c("Kaplan-Meier Curve", "Curve with Median of Simulated Times", "Curve wi
thout Censored Times"),
           lty= 1, col = c("black", "Blue", "Red"), cex = 0.7)
```



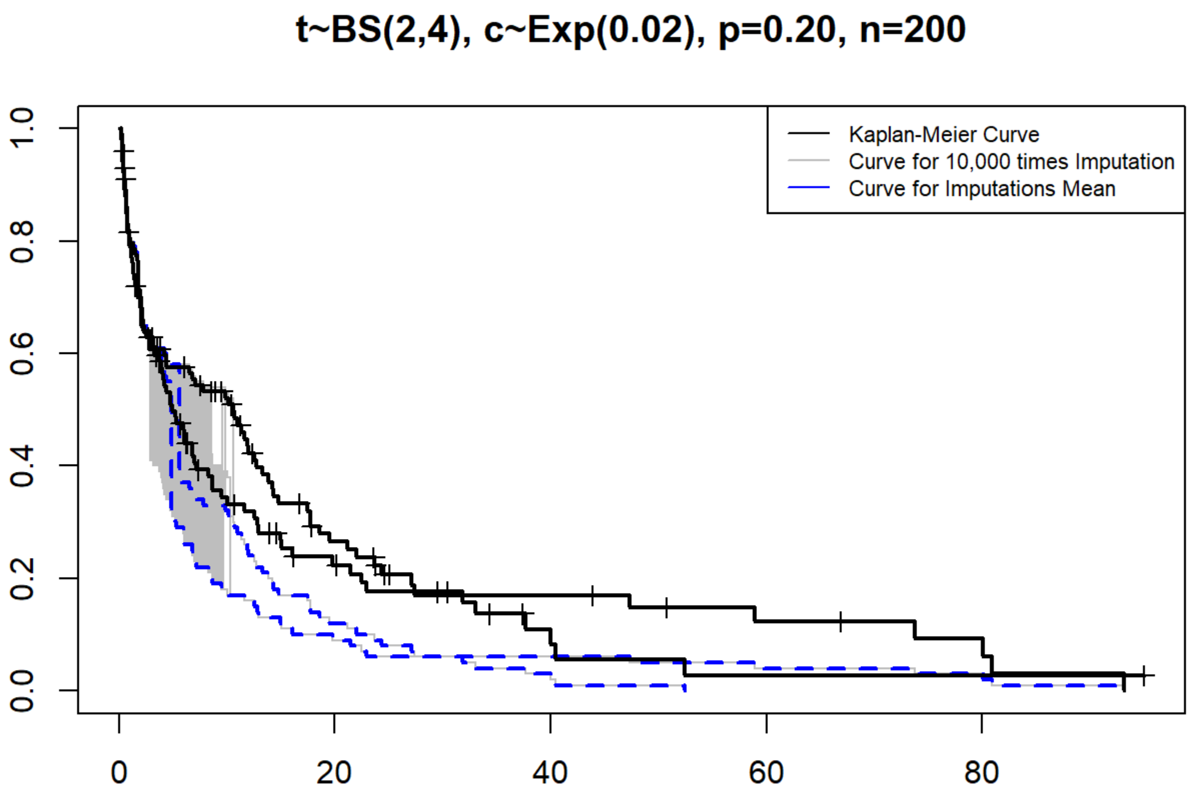
t~BS(2,4), c~Exp(0.02), p=0.20, n=200

Figure 11 in the paper.

```
    # Kaplan-Meier Curve:
    curve1 = survfit(Surv(z,delta) ~ x); curve1
```

```
## Call: survfit(formula = Surv(z, delta) ~ x)
##
##       n events median 0.95LCL 0.95UCL
## x=0 100     79  10.62    4.38   13.26
## x=1 100     79   4.91    3.68    8.27
```

```
    plot(curve1, mark.time = TRUE,lty = 1, lwd = 2, col = "black",
         main = paste("t~BS(2,4), c~Exp(0.02), p=0.20, n=200"))  #KM_Estimation

    # Curve for 10,000 Times Imputation
    timp=t
    impsim = bugsOut$sims.array[,1,3+ic]
    for (i in 1:nrow(impsim)) {
      timp[ic] <- impsim[i,]
      kmi = survfit(Surv(timp,delta1) ~ x)
      lines(kmi, mark.time = TRUE, col = "gray", lty = 1)    # n time Imputation

    }
    # Curve for Imputations Mean
    timp[ic] <- colMeans(impsim)
    kmmean = survfit(Surv(timp,delta1) ~ x)
    lines(kmi, mark.time = TRUE, col = "blue", lty = 2, lwd = 2)  # Mean of n times Imputation

    lines(curve1, mark.time = TRUE,lty = 1, lwd = 2, col = "black",
         main = paste("t~BS(2,4), c~Exp(0.01), p=0.10, n=200"))  #KM_Estimation



    legend("topright",
           c("Kaplan-Meier Curve", "Curve for 10,000 times Imputation", "Curve for Imputations M
ean"),
           lty = 1, col = c("Black", "gray","blue"), cex = .7)
```



3-Run the Bayesian Approach on the Breast Cancer Data distributed as the Weibull.

```
# Install packages:survival & R2openBUGS.
library(survival)
library(R2OpenBUGS)
# Set working directory and modelfile.
getwd()
```

```
## [1] "C:/Users/novingostar/Documents/R-studio"
```

```
bugswd = paste0(getwd(),"/bugswd"); bugswd
```

```
## [1] "C:/Users/novingostar/Documents/R-studio/bugswd"
```

```
modelfile = paste0(bugswd,"/modelfile.txt"); modelfile
```

```
## [1] "C:/Users/novingostar/Documents/R-studio/bugswd/modelfile.txt"
```

```
  # Import and define variables in Data.
  breast <- read.table("Data_Paper1.txt", header = TRUE)
  t <- breast$t
  c <- breast$c
  x <- breast$AgeC
  length(t[t == "NA"])/length(t)   # Percent of Censoring, 88 Censor, 40%
```

```
## [1] 0.4
```

```
  length(c[c == "0"])/length(c)   # Percent of Observed
```

```
## [1] 0.6
```

```
  n = length(t); n
```

```
## [1] 220
```

```
  z = breast$z   # Composed from Observed and Censored data
  delta = breast$delta   # delta=0 means Censoring
  ic = which(delta == "0")   # indicator censor
  age <- breast$AgeC
  # Run model in BUGS.
  modeltext = "model {
    for(i in 1:n){
    t[i] ~ dweib(shape,lambda)C(c[i], )
    cim[i]<-step(c[i]-1.0E-5)*pow(log(2)/lambda+pow(c[i],shape),1/shape)
    }
    # priors
    shape ~ dgamma(0.01,0.01)
    lambda ~ dgamma(0.01, 0.01)
    }
  "
  # write BUGS output into file.
  cat(modeltext, file = modelfile) #file.show(modelfile)
  modeldata = list(n = n, t = t, c = c)
  modelinit = list(list(shape = 1, lambda = 1 ))
  param = c("shape","lambda", "cim")
  # bugs ----------------------------------------
  bugsOut <- bugs(
    working.directory = bugswd,
    model.file = modelfile,
    data = modeldata,
    inits = modelinit,
    #inits = NULL,
    parameters.to.save = param,
    n.chains = 1,
    n.iter = 11000,
    n.burnin = 1000,
    n.thin = 20
    #, debug = TRUE
    #, codaPkg = TRUE
  )

# output ----------------------------------------
  bugsOut$DIC
```

```
## [1] 1697
```

```
  # Dimension of output:
  dim(bugsOut$sims.array)   #composed: alpha, lambda, 88 simulation,deviance = 91 columns.
```

```
## [1] 10000     1    91
```

```
  # Describe censored simulations.
  bugsOut$sims.array[1:5,1,3:90]         # Head
```

```
##      cim[1] cim[4] cim[5] cim[8] cim[10] cim[16] cim[19] cim[20] cim[25]
## [1,]  416.1  215.3  195.0  212.8   392.5   252.5   192.6   180.2   192.6
## [2,]  414.1  223.7  205.8  221.5   391.2   257.7   203.8   193.5   203.8
## [3,]  380.8  185.3  166.5  182.9   357.4   220.6   164.4   153.6   164.4
## [4,]  406.2  214.1  195.8  211.8   383.1   248.5   193.8   183.3   193.8
## [5,]  383.3  191.4  173.5  189.1   360.2   225.6   171.5   161.3   171.5
##      cim[26] cim[27] cim[29] cim[30] cim[32] cim[33] cim[35] cim[39] cim[41]
## [1,]   215.3   248.9   238.4   230.6   249.8   569.5   222.0   397.2   507.8
## [2,]   223.7   254.5   244.8   237.6   255.3   564.0   229.8   395.8   503.5
## [3,]   185.3   217.2   207.1   199.7   218.0   532.8   191.6   362.1   471.6
## [4,]   214.1   245.2   235.4   228.1   246.0   556.7   220.2   387.7   496.0
## [5,]   191.4   222.3   212.5   205.3   223.1   534.2   197.5   364.8   473.3
##      cim[46] cim[47] cim[49] cim[50] cim[51] cim[58] cim[62] cim[64] cim[65]
## [1,]   290.2   268.5   419.9   183.6   419.0   198.9   212.8   248.0   329.8
## [2,]   293.2   272.7   417.8   196.3   416.9   209.2   221.5   253.6   330.8
## [3,]   257.1   236.0   384.5   156.5   383.6   170.1   182.9   216.3   295.7
## [4,]   284.3   263.7   409.9   186.1   408.9   199.3   211.8   244.4   322.3
## [5,]   261.3   240.7   387.0   164.1   386.1   176.9   189.1   221.4   299.2
##      cim[66] cim[76] cim[80] cim[81] cim[83] cim[84] cim[87] cim[89] cim[90]
## [1,]   253.3   315.0   271.2   310.4   265.8   273.9   243.7   186.6   258.7
## [2,]   258.6   316.7   275.2   312.3   270.2   277.8   249.6   198.7   263.5
## [3,]   221.4   281.2   238.6   276.7   233.4   241.3   212.1   159.0   226.5
## [4,]   249.4   308.0   266.2   303.6   261.1   268.8   240.3   188.6   254.4
## [5,]   226.4   285.0   243.2   280.6   238.1   245.8   217.3   166.4   231.4
##      cim[91] cim[93] cim[96] cim[97] cim[98] cim[99] cim[100] cim[101] cim[104]
## [1,]   232.3   536.7   471.4   266.7   275.7   305.8    268.5    438.0    237.5
## [2,]   239.2   531.8   467.9   271.0   279.5   307.9    272.7    435.4    244.0
## [3,]   201.3   500.2   435.5   234.3   243.0   272.2    236.0    402.4    206.3
## [4,]   229.7   524.4   460.2   262.0   270.5   299.2    263.7    427.5    234.6
## [5,]   206.9   501.8   437.5   239.0   247.5   276.1    240.7    404.7    211.7
##      cim[105] cim[109] cim[110] cim[111] cim[112] cim[113] cim[114] cim[115]
## [1,]    360.5    261.4    245.4    284.8    194.2    282.9    208.6    526.1
## [2,]    360.3    266.0    251.2    288.0    205.2    286.3    217.8    521.4
## [3,]    325.9    229.1    213.8    251.8    165.8    250.0    179.1    489.7
## [4,]    352.0    256.9    241.9    279.1    195.2    277.4    208.0    514.0
## [5,]    329.0    233.9    219.0    256.1    172.8    254.3    185.4    491.4
##      cim[120] cim[121] cim[126] cim[127] cim[129] cim[131] cim[132] cim[136]
## [1,]    461.8    226.3    180.8    259.6    295.7    259.6    267.6    311.3
## [2,]    458.6    233.7    194.1    264.4    298.3    264.4    271.9    313.1
## [3,]    426.0    195.6    154.1    227.4    262.4    227.4    235.2    277.6
## [4,]    450.9    224.2    183.8    255.2    289.5    255.2    262.8    304.5
## [5,]    428.1    201.3    161.9    232.2    266.5    232.2    239.8    281.4
##      cim[137] cim[138] cim[140] cim[142] cim[144] cim[145] cim[146] cim[149]
## [1,]    203.7    232.3    181.5    202.1    439.9    293.9    266.7    221.2
## [2,]    213.5    239.2    194.6    212.0    437.2    296.6    271.0    229.1
## [3,]    174.5    201.3    154.7    173.0    404.3    260.6    234.3    190.8
## [4,]    203.6    229.7    184.4    202.2    429.4    287.8    262.0    219.5
## [5,]    181.1    206.9    162.4    179.6    406.6    264.8    239.0    196.7
##      cim[158] cim[161] cim[164] cim[173] cim[177] cim[180] cim[182] cim[187]
## [1,]    212.8    384.9    321.4    288.4    226.3    520.3    293.9    352.1
## [2,]    221.5    383.9    322.8    291.4    233.7    515.8    296.6    352.2
## [3,]    182.9    350.0    287.5    255.3    195.6    484.0    260.6    317.6
## [4,]    211.8    375.7    314.2    282.6    224.2    508.3    287.8    343.9
## [5,]    189.1    352.8    291.2    259.5    201.3    485.7    264.8    320.9
##      cim[196] cim[197] cim[200] cim[201] cim[202] cim[205] cim[211] cim[212]
## [1,]    334.4    277.5    742.4    394.4    865.6    202.1    265.8    465.6
## [2,]    335.2    281.2    734.4    393.0    856.3    212.0    270.2    462.3
## [3,]    300.3    244.8    705.0    359.3    827.9    173.0    233.4    429.8
## [4,]    326.7    272.2    727.7    384.9    849.9    202.2    261.1    454.6
## [5,]    303.7    249.2    705.5    362.0    828.0    179.6    238.1    431.8
##      cim[215] cim[218] cim[220]
## [1,]    212.8    900.9    305.8
## [2,]    221.5    891.3    307.9
## [3,]    182.9    863.1    272.2
## [4,]    211.8    884.9    299.2
## [5,]    189.1    863.1    276.1
```

```
bugsOut$sims.array[9996:10000,1,3:90]  # Tail
```

```
##      cim[1] cim[4] cim[5] cim[8] cim[10] cim[16] cim[19] cim[20] cim[25]
## [1,]  401.5  223.0  207.8  221.0   379.3   253.3   206.2   198.2   206.2
## [2,]  393.3  201.4  183.4  199.2   370.2   235.7   181.4   171.1   181.4
## [3,]  375.4  180.3  161.7  177.9   352.0   215.4   159.6   148.9   159.6
## [4,]  411.0  211.0  190.8  208.5   387.4   247.9   188.5   176.3   188.5
## [5,]  401.2  201.2  181.1  198.7   377.6   238.0   178.8   166.7   178.8
##      cim[26] cim[27] cim[29] cim[30] cim[32] cim[33] cim[35] cim[39] cim[41]
## [1,]   223.0   250.3   241.5   235.2   251.1   547.8   228.3   383.7   488.5
## [2,]   201.4   232.4   222.6   215.4   233.2   544.0   207.5   374.8   483.3
## [3,]   180.3   212.0   202.0   194.6   212.8   527.4   186.5   356.7   466.2
## [4,]   211.0   244.4   233.9   226.2   245.2   564.1   217.7   392.1   502.6
## [5,]   201.2   234.5   224.0   216.3   235.4   554.4   207.8   382.3   492.8
##      cim[46] cim[47] cim[49] cim[50] cim[51] cim[58] cim[62] cim[64] cim[65]
## [1,]   286.0   267.0   405.0   200.3   404.1   210.7   221.0   249.6   321.5
## [2,]   271.4   250.8   397.0   173.9   396.1   186.8   199.2   231.6   309.4
## [3,]   251.8   230.8   379.1   151.8   378.2   165.2   177.9   211.2   290.4
## [4,]   285.4   263.8   414.8   179.7   413.9   194.7   208.5   243.5   324.8
## [5,]   275.6   254.0   405.0   170.1   404.0   185.0   198.7   233.6   315.0
##      cim[66] cim[76] cim[80] cim[81] cim[83] cim[84] cim[87] cim[89] cim[90]
## [1,]   254.0   308.1   269.4   303.9   264.7   271.7   245.9   202.2   258.6
## [2,]   236.5   295.1   253.4   290.7   248.3   255.9   227.5   176.3   241.5
## [3,]   216.2   275.9   233.4   271.4   228.2   236.0   207.0   154.3   221.3
## [4,]   248.8   310.1   266.5   305.5   261.2   269.2   239.1   182.6   254.1
## [5,]   238.9   300.2   256.7   295.6   251.3   259.3   229.3   172.9   244.2
##      cim[91] cim[93] cim[96] cim[97] cim[98] cim[99] cim[100] cim[101] cim[104]
## [1,]   236.6   516.2   453.7   265.5   273.3   299.8    267.0    422.1    240.8
## [2,]   217.0   511.7   447.4   249.1   257.6   286.3    250.8    414.7    221.8
## [3,]   196.2   494.8   430.1   229.1   237.8   266.9    230.8    397.0    201.2
## [4,]   227.9   531.4   466.2   262.1   271.0   300.9    263.8    432.8    233.0
## [5,]   218.0   521.6   456.4   252.2   261.1   291.1    254.0    423.0    223.2
##      cim[105] cim[109] cim[110] cim[111] cim[112] cim[113] cim[114] cim[115]
## [1,]    349.6    260.9    247.4    281.2    207.3    279.6    217.9    506.0
## [2,]    339.1    244.1    229.1    266.2    182.7    264.5    195.4    501.2
## [3,]    320.6    223.9    208.6    246.5    161.0    244.8    174.1    484.3
## [4,]    355.5    256.7    240.9    280.0    190.1    278.2    204.4    520.8
## [5,]    345.7    246.8    231.0    270.1    180.3    268.3    194.6    511.1
##      cim[120] cim[121] cim[126] cim[127] cim[129] cim[131] cim[132] cim[136]
## [1,]    444.7    231.7    198.6    259.3    290.8    259.3    266.2    304.8
## [2,]    438.1    211.4    171.6    242.4    276.7    242.4    250.0    291.6
## [3,]    420.6    190.5    149.5    222.2    257.1    222.2    229.9    272.3
## [4,]    456.6    221.9    177.0    254.9    290.9    254.9    262.9    306.4
## [5,]    446.8    212.1    167.3    245.1    281.0    245.1    253.1    296.6
##      cim[137] cim[138] cim[140] cim[142] cim[144] cim[145] cim[146] cim[149]
## [1,]    214.2    236.6    199.0    213.0    423.9    289.2    265.5    227.6
## [2,]    191.1    217.0    172.2    189.6    416.6    274.9    249.1    206.8
## [3,]    169.6    196.2    150.1    168.1    398.9    255.4    229.1    185.8
## [4,]    199.5    227.9    177.6    197.9    434.7    289.1    262.1    216.8
## [5,]    189.7    218.0    168.0    188.2    424.9    279.2    252.2    207.0
##      cim[158] cim[161] cim[164] cim[173] cim[177] cim[180] cim[182] cim[187]
## [1,]    221.0    372.3    313.9    284.4    231.7    500.5    289.2    341.9
## [2,]    199.2    362.9    301.3    269.7    211.4    495.6    274.9    331.0
## [3,]    177.9    344.6    282.2    250.1    190.5    478.6    255.4    312.3
## [4,]    208.5    379.9    316.5    283.6    221.9    515.0    289.1    347.1
## [5,]    198.7    370.1    306.7    273.8    212.1    505.3    279.2    337.3
##      cim[196] cim[197] cim[200] cim[201] cim[202] cim[205] cim[211] cim[212]
## [1,]    325.7    274.8    715.9    381.1    836.7    213.0    264.7    448.3
## [2,]    313.9    259.4    715.2    372.1    837.5    189.6    248.3    441.8
## [3,]    294.9    239.5    699.6    353.9    822.5    168.1    228.2    424.4
## [4,]    329.5    272.8    736.9    389.3    860.1    197.9    261.2    460.4
## [5,]    319.6    262.9    727.3    379.5    850.5    188.2    251.3    450.7
##      cim[215] cim[218] cim[220]
## [1,]    221.0    871.4    299.8
## [2,]    199.2    872.6    286.3
## [3,]    177.9    857.7    266.9
## [4,]    208.5    895.3    300.9
## [5,]    198.7    885.8    291.1
```

```
  bugsOut$summary[1:2, c(1:2)]    # mean & sd parameters: alpha & lambda
```

```
##               mean           sd
## shape  1.169883950 0.0763082405
## lambda 0.001913478 0.0008124746
```

```
  # Describe parameter simulations:
  parsim3 = bugsOut$sims.array[,1,1:2]   #parameter simulation
  impsim = bugsOut$sims.array[,1,3:90]  # imputation simulation
  timp = t
```

Convergence: Geweke diagnostics.

```
  geweke.diag(parsim3, frac=0.10, frac2 = 0.50)   #Z-score
```

```
## 
## Fraction in 1st window = 0.1
## Fraction in 2nd window = 0.5
## 
##   shape  lambda 
##  0.2636 -0.4964
```

Generate and save necessary files:

```
write.csv(parsim3, file = "matparsim3.csv")
mcmcparsim3 <- mcmc(as.matrix(parsim3))
```

ACF computations

```
autocorr.diag(mcmcparsim3)
```

```
##                shape       lambda
## Lag 0  1.0000000000  1.000000000
## Lag 1  0.6168381593  0.590070397
## Lag 5  0.0705969645  0.073112900
## Lag 10 0.0042004795  0.012212819
## Lag 50 0.0008355669 -0.000427442
```

Effective Sample Size (ESS)

```
effectiveSize(parsim3)
```

```
##    shape   lambda
## 2369.585 2577.802
```

Fiqure 12 in the paper.

```
  # Kaplan-Meier Curve:
  curve1 = survfit(Surv(z,delta) ~ age); curve1
```

```
## Call: survfit(formula = Surv(z, delta) ~ age)
##
##         n events median 0.95LCL 0.95UCL
## age=1  66     37    174     119     308
## age=2 154     95    134     112     166
```

```
  plot(curve1, mark.time = TRUE,lty = 1,conf.int = FALSE,  col = "black",
       main = paste("Posterior Estimate: Shape=1.24,Scale=0.001,DIC=1698"))  #KM_Estimation
  # Curve with Median of Simulated Times
  # output ----------------------------------------
  # imputation      h=hat
  shapeh = bugsOut$mean$shape; shapeh
```

```
## [1] 1.169884
```

```
  lambdah = bugsOut$mean$lambda; lambdah
```

```
## [1] 0.001913478
```

```
  cen=c
  # Compute median of Simulations.
  library(miscTools)
```

```
##
## Attaching package: 'miscTools'
## The following object is masked from 'package:extraDistr':
##
##     ddnorm
```

```
  zmed = colMedians(impsim)
  #
  ic = which(delta==0); ic      #index censor to count number of censored case.
```

```
##  [1]   1   4   5   8  10  16  19  20  25  26  27  29  30  32  33  35  39  41  46
## [20]  47  49  50  51  58  62  64  65  66  76  80  81  83  84  87  89  90  91  93
## [39]  96  97  98  99 100 101 104 105 109 110 111 112 113 114 115 120 121 126 127
## [58] 129 131 132 136 137 138 140 142 144 145 146 149 158 161 164 173 177 180 182
## [77] 187 196 197 200 201 202 205 211 212 215 218 220
```

```
  length(ic)
```

```
## [1] 88
```

```
  zimp = rep(NA,n)
  zimp[ic] = zmed[ic]
  zimp[-ic] = z[-ic]  # zimp = failure times+imputed censored times
  delta1 = rep(1,n)  # after impute, all of times are observed then we made delta1.
  #
  km2 = survfit(Surv(zimp,delta1) ~ x); km2     # Bayesian Imputation
```

```
## Call: survfit(formula = Surv(zimp, delta1) ~ x)
##
##    54 observations deleted due to missingness
##        n events median 0.95LCL 0.95UCL
## x=1  49     49    141     110     200
## x=2 117    117    112      88     145
```

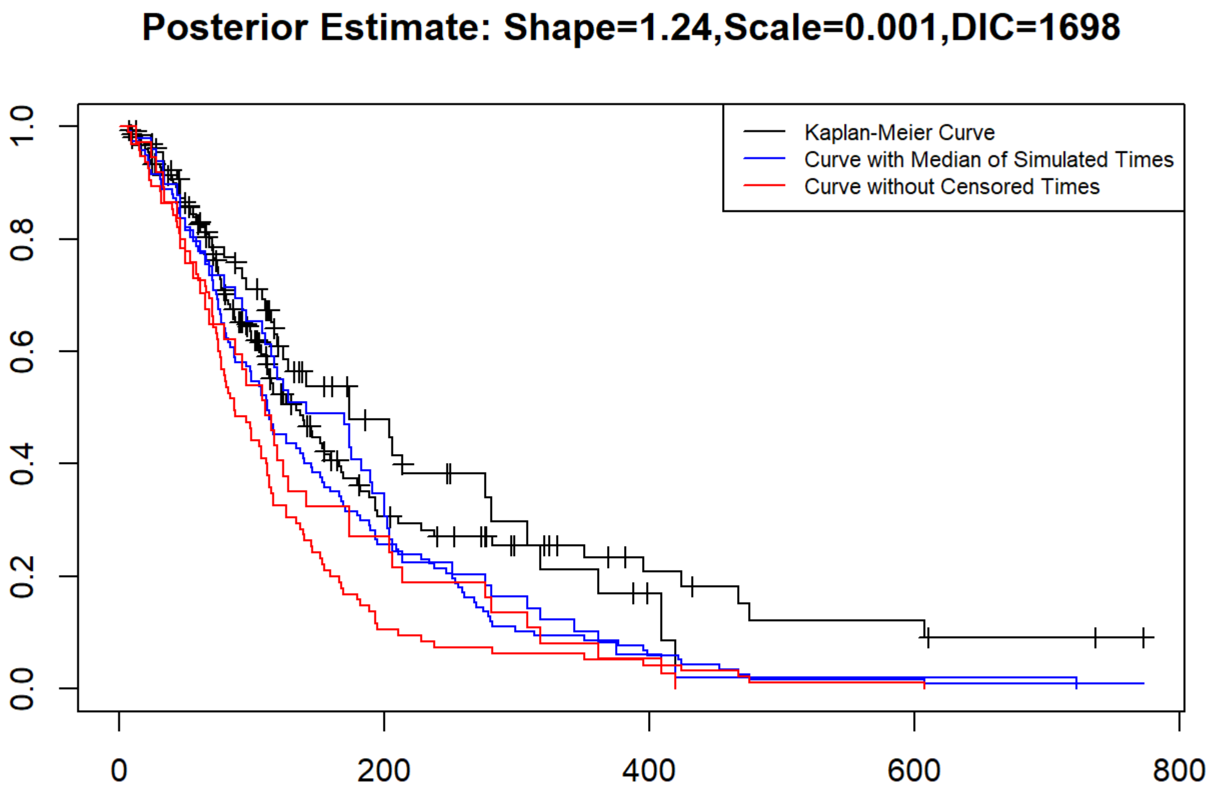
```
  lines(km2, mark.time = TRUE, col = "Blue", lty = 1)

  # Curve without Censored Times
  tOC = z[delta==1]  # number of observed times
  deltaOC = rep(1, length(tOC))
  length(deltaOC)
```

```
## [1] 132
```

```
  km3 = survfit(Surv(tOC, deltaOC) ~ x[delta==1]); km3       # Omitting_Censored
```

```
## Call: survfit(formula = Surv(tOC, deltaOC) ~ x[delta == 1])
##
##                    n events median 0.95LCL 0.95UCL
## x[delta == 1]=1 37     37    110      79     174
## x[delta == 1]=2 95     95     87      76     112
```

```
  lines(km3, mark.time = TRUE, col = "Red", lty = 1)

  legend("topright", c("Kaplan-Meier Curve", "Curve with Median of Simulated Times", "Curve with
out Censored Times"),
         lty= 1, col = c("black", "Blue", "Red"), cex = 0.7)
```

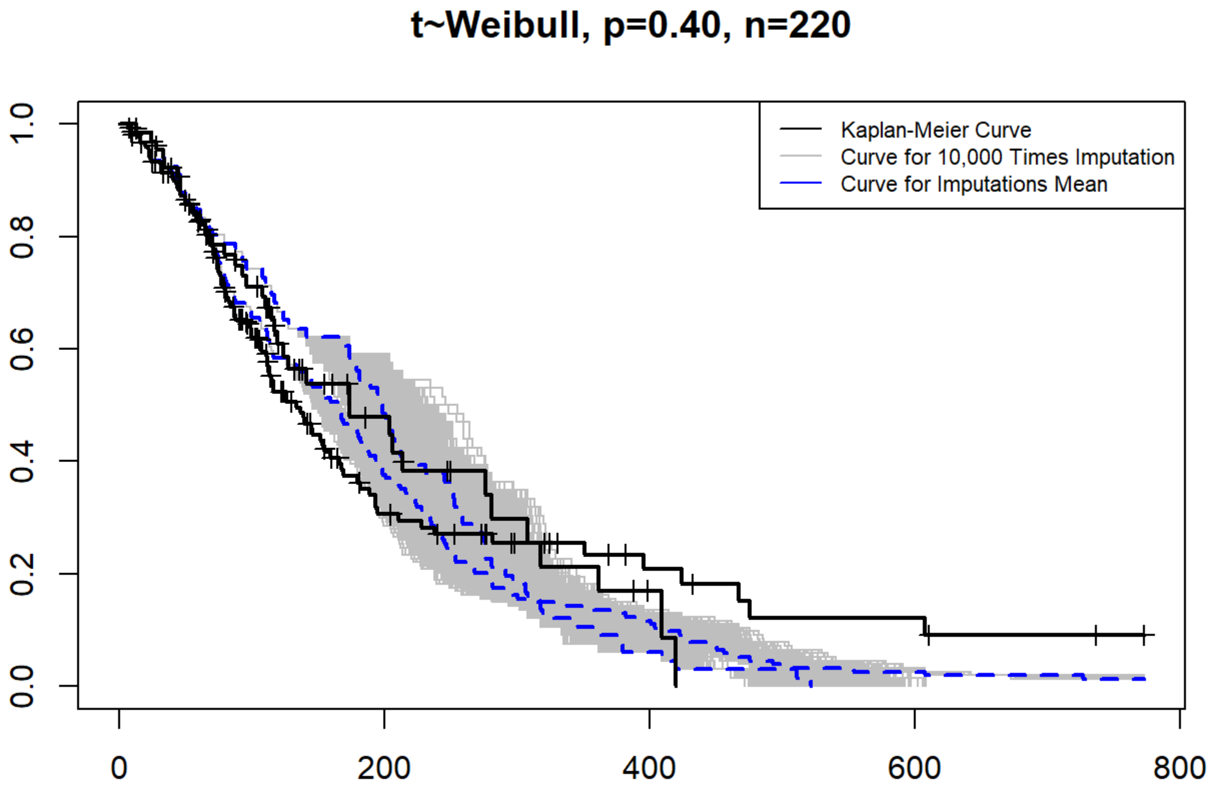


Fiqure 13 in the paper.

```
  # Kaplan-Meier Curve
  curve1 = survfit(Surv(z,delta) ~ x); curve1
```

```
## Call: survfit(formula = Surv(z, delta) ~ x)
##
##        n events median 0.95LCL 0.95UCL
## x=1   66     37    174     119     308
## x=2  154     95    134     112     166
```

```
  plot(curve1, mark.time = TRUE,lty = 1, lwd =2, col = "black",
       main = paste("t~Weibull, p=0.40, n=220"))  #KM_Estimation

  # Curve with Median of Simulated Times
  # simulation
  for (i in 1:nrow(impsim)) {
    timp[ic] <- impsim[i,]
    kmi = survfit(Surv(timp,delta1) ~ x)
    lines(kmi, mark.time = TRUE, col = "gray", lty = 1)    # n time Imputation
    #Sys.sleep(.5)
  }
  # Curve for Imputations Mean
  lines(kmi, mark.time = TRUE, col = "blue", lty = 2, lwd = 2)  # Mean of n times Imputation

  lines(curve1, mark.time = TRUE,lty = 1, lwd =2, col = "black",
       main = paste("t~Weibull, p=0.40, n=220"))  #KM_Estimation

  legend("topright",
         c("Kaplan-Meier Curve", "Curve for 10,000 Times Imputation", "Curve for Imputations Mea
n"),
         lty = 1, col = c("Black", "gray","blue"), cex = .7)
```



4-Run the Bayesian Approach on the Breast Cancer Data distributed as the Birnbaum-Saunders.

```
# Install packages:survival & R2openBUGS.
library(survival)
library(R2OpenBUGS)
# Set working directory and modelfile.
getwd()
```

```
## [1] "C:/Users/novingostar/Documents/R-studio"
```

```
bugswd = paste0(getwd(),"/bugswd"); bugswd
```

```
## [1] "C:/Users/novingostar/Documents/R-studio/bugswd"
```

```
modelfile = paste0(bugswd,"/modelfile.txt"); modelfile
```

```
## [1] "C:/Users/novingostar/Documents/R-studio/bugswd/modelfile.txt"
```

```
# Import and define variables in Data.
breast <- read.table("Data_Paper1.txt", header = TRUE)
x <- breast$AgeC
t <- breast$t  #time based on month
c <- breast$c
length(t[t == "NA"])/length(t)   # Percent of Censoring, 88 Censor, 40%
```

```
## [1] 0.4
```

```
length(c[c == "0"])/length(c)   # Percent of Observed
```

```
## [1] 0.6
```

```
n = length(x); n
```

```
## [1] 220
```

```
z = breast$z   # Composed from Observed and Censored data
delta = breast$delta   # delta=0 means Censoring
ic = which(delta == 0)   # indicator censor
length(ic)
```

```
## [1] 88
```

```
age <- breast$AgeC
# Run model in BUGS.
modeltext = "model {
for(i in 1:n){
t[i] ~ dbs(shape,lambda)C(c[i], )
cim[i] <- step(c[i]-1.0E-5)*lambda    #tmed
}
# priors
shape ~ dgamma(0.01,0.01)
lambda ~ dgamma(0.01, 0.01)
}
"
# write BUGS output into file.
cat(modeltext, file = modelfile) #file.show(modelfile)
modeldata = list(n = n, t = t, c = c)
modelinit = list(list(shape = 4, lambda = 4))
param = c("shape","lambda", "cim")
# bugs ----------------------------------------
bugsOut <- bugs(
  working.directory = bugswd,
  model.file = modelfile,
  data = modeldata,
  inits = modelinit,
  #inits = NULL,
  parameters.to.save = param,
  n.chains = 1,
  n.iter = 11000,
  n.burnin = 1000,
  n.thin = 20
  #, debug = TRUE
  #, codaPkg = TRUE
)

# output ----------------------------------------
bugsOut$DIC
```

```
## [1] 1510
```

```
# Dimension of output:
dim(bugsOut$sims.array)   #composed: alpha, lambda, 88 simulation,deviance = 91 columns.
```

```
## [1] 10000     1    91
```

```
# Describe censored simulations.
bugsOut$sims.array[1:5,1,3:90]  # report 1 till 5 from 100 times censored times simulations.
```

```
##      cim[1] cim[4] cim[5] cim[8] cim[10] cim[16] cim[19] cim[20] cim[25]
## [1,]  119.4  119.4  119.4  119.4   119.4   119.4   119.4   119.4   119.4
## [2,]  142.5  142.5  142.5  142.5   142.5   142.5   142.5   142.5   142.5
## [3,]  143.6  143.6  143.6  143.6   143.6   143.6   143.6   143.6   143.6
## [4,]  130.4  130.4  130.4  130.4   130.4   130.4   130.4   130.4   130.4
## [5,]  130.3  130.3  130.3  130.3   130.3   130.3   130.3   130.3   130.3
##      cim[26] cim[27] cim[29] cim[30] cim[32] cim[33] cim[35] cim[39] cim[41]
## [1,]   119.4   119.4   119.4   119.4   119.4   119.4   119.4   119.4   119.4
## [2,]   142.5   142.5   142.5   142.5   142.5   142.5   142.5   142.5   142.5
## [3,]   143.6   143.6   143.6   143.6   143.6   143.6   143.6   143.6   143.6
## [4,]   130.4   130.4   130.4   130.4   130.4   130.4   130.4   130.4   130.4
## [5,]   130.3   130.3   130.3   130.3   130.3   130.3   130.3   130.3   130.3
##      cim[46] cim[47] cim[49] cim[50] cim[51] cim[58] cim[62] cim[64] cim[65]
## [1,]   119.4   119.4   119.4   119.4   119.4   119.4   119.4   119.4   119.4
## [2,]   142.5   142.5   142.5   142.5   142.5   142.5   142.5   142.5   142.5
## [3,]   143.6   143.6   143.6   143.6   143.6   143.6   143.6   143.6   143.6
## [4,]   130.4   130.4   130.4   130.4   130.4   130.4   130.4   130.4   130.4
## [5,]   130.3   130.3   130.3   130.3   130.3   130.3   130.3   130.3   130.3
##      cim[66] cim[76] cim[80] cim[81] cim[83] cim[84] cim[87] cim[89] cim[90]
## [1,]   119.4   119.4   119.4   119.4   119.4   119.4   119.4   119.4   119.4
## [2,]   142.5   142.5   142.5   142.5   142.5   142.5   142.5   142.5   142.5
## [3,]   143.6   143.6   143.6   143.6   143.6   143.6   143.6   143.6   143.6
## [4,]   130.4   130.4   130.4   130.4   130.4   130.4   130.4   130.4   130.4
## [5,]   130.3   130.3   130.3   130.3   130.3   130.3   130.3   130.3   130.3
##      cim[91] cim[93] cim[96] cim[97] cim[98] cim[99] cim[100] cim[101] cim[104]
## [1,]   119.4   119.4   119.4   119.4   119.4   119.4    119.4    119.4    119.4
## [2,]   142.5   142.5   142.5   142.5   142.5   142.5    142.5    142.5    142.5
## [3,]   143.6   143.6   143.6   143.6   143.6   143.6    143.6    143.6    143.6
## [4,]   130.4   130.4   130.4   130.4   130.4   130.4    130.4    130.4    130.4
## [5,]   130.3   130.3   130.3   130.3   130.3   130.3    130.3    130.3    130.3
##      cim[105] cim[109] cim[110] cim[111] cim[112] cim[113] cim[114] cim[115]
## [1,]    119.4    119.4    119.4    119.4    119.4    119.4    119.4    119.4
## [2,]    142.5    142.5    142.5    142.5    142.5    142.5    142.5    142.5
## [3,]    143.6    143.6    143.6    143.6    143.6    143.6    143.6    143.6
## [4,]    130.4    130.4    130.4    130.4    130.4    130.4    130.4    130.4
## [5,]    130.3    130.3    130.3    130.3    130.3    130.3    130.3    130.3
##      cim[120] cim[121] cim[126] cim[127] cim[129] cim[131] cim[132] cim[136]
## [1,]    119.4    119.4    119.4    119.4    119.4    119.4    119.4    119.4
## [2,]    142.5    142.5    142.5    142.5    142.5    142.5    142.5    142.5
## [3,]    143.6    143.6    143.6    143.6    143.6    143.6    143.6    143.6
## [4,]    130.4    130.4    130.4    130.4    130.4    130.4    130.4    130.4
## [5,]    130.3    130.3    130.3    130.3    130.3    130.3    130.3    130.3
##      cim[137] cim[138] cim[140] cim[142] cim[144] cim[145] cim[146] cim[149]
## [1,]    119.4    119.4    119.4    119.4    119.4    119.4    119.4    119.4
## [2,]    142.5    142.5    142.5    142.5    142.5    142.5    142.5    142.5
## [3,]    143.6    143.6    143.6    143.6    143.6    143.6    143.6    143.6
## [4,]    130.4    130.4    130.4    130.4    130.4    130.4    130.4    130.4
## [5,]    130.3    130.3    130.3    130.3    130.3    130.3    130.3    130.3
##      cim[158] cim[161] cim[164] cim[173] cim[177] cim[180] cim[182] cim[187]
## [1,]    119.4    119.4    119.4    119.4    119.4    119.4    119.4    119.4
## [2,]    142.5    142.5    142.5    142.5    142.5    142.5    142.5    142.5
## [3,]    143.6    143.6    143.6    143.6    143.6    143.6    143.6    143.6
## [4,]    130.4    130.4    130.4    130.4    130.4    130.4    130.4    130.4
## [5,]    130.3    130.3    130.3    130.3    130.3    130.3    130.3    130.3
##      cim[196] cim[197] cim[200] cim[201] cim[202] cim[205] cim[211] cim[212]
## [1,]    119.4    119.4    119.4    119.4    119.4    119.4    119.4    119.4
## [2,]    142.5    142.5    142.5    142.5    142.5    142.5    142.5    142.5
## [3,]    143.6    143.6    143.6    143.6    143.6    143.6    143.6    143.6
## [4,]    130.4    130.4    130.4    130.4    130.4    130.4    130.4    130.4
## [5,]    130.3    130.3    130.3    130.3    130.3    130.3    130.3    130.3
##      cim[215] cim[218] cim[220]
## [1,]    119.4    119.4    119.4
## [2,]    142.5    142.5    142.5
## [3,]    143.6    143.6    143.6
## [4,]    130.4    130.4    130.4
## [5,]    130.3    130.3    130.3
```

```
  bugsOut$summary[1:2, c(1:2)]    # mean & sd parameters: alpha & lambda
```

```
##              mean          sd
## shape    1.207428  0.07906746
## lambda 142.837480 12.23087735
```

```
  # Describe parameter simulations:
  parsim4 = bugsOut$sims.array[,1,1:2]   #parameter simulation: 10000*2
  impsim = bugsOut$sims.array[,1,3:90]  # imputation simulation: 10000*88
  timp = t
```

Convergence: Geweke dignostics.

```
  geweke.diag(parsim4, frac=0.10, frac2 = 0.50)   #Z-score
```

```
## 
## Fraction in 1st window = 0.1
## Fraction in 2nd window = 0.5 
## 
##  shape lambda 
## 0.8888 0.3807
```

Generate and save necessary files:

```
write.csv(parsim4, file = "matparsim4.csv")
mcmcparsim4 <- mcmc(as.matrix(parsim4))
```

ACF computations

```
autocorr.diag(mcmcparsim4)
```

```
##                shape       lambda
## Lag 0   1.000000000  1.000000000
## Lag 1   0.004794630 -0.002319400
## Lag 5   0.007802455  0.002563156
## Lag 10  0.010724258  0.002714407
## Lag 50 -0.012052864 -0.008207074
```

Effective Sample Size (ESS)

```
  effectiveSize(parsim4)
```

```
##  shape lambda
##  10000  10000
```

Fiqure 12 in the paper.

```
  # Kaplan-Meier Curve:
  curve1 = survfit(Surv(z,delta) ~ age); curve1
```

```
## Call: survfit(formula = Surv(z, delta) ~ age)
##
##         n events median 0.95LCL 0.95UCL
## age=1  66     37    174     119     308
## age=2 154     95    134     112     166
```

```
  plot(curve1, mark.time = TRUE,lty = 1,conf.int = FALSE,  col = "black",
       main = paste("Posterior Estimate: Shape=1.22, Scale=145.21, DIC=1510"))  #KM_Estimation

  # Curve with Median of Simulated Times
  # output ----------------------------------------
  # imputation      h=hat
  shapeh = bugsOut$mean$shape; shapeh
```

```
## [1] 1.207428
```

```
  lambdah = bugsOut$mean$lambda; lambdah
```

```
## [1] 142.8375
```

```
  scaleh = lambdah; scaleh
```

```
## [1] 142.8375
```

```
  cen=c
  # Compute median of Simulations.
  #install.packages("extraDistr")
  library(extraDistr)
  # How calculate median times in Birnbaum-Saunders distribution:
  zmed = qfatigue(.5*pfatigue(cen,shapeh,scaleh, mu = 0, lower.tail = FALSE),shapeh, scaleh,mu =
0, lower.tail = FALSE)
  #
  ic = which(delta==0); ic      #index censor to count number of censored case.
```

```
##  [1]   1   4   5   8  10  16  19  20  25  26  27  29  30  32  33  35  39  41  46
## [20]  47  49  50  51  58  62  64  65  66  76  80  81  83  84  87  89  90  91  93
## [39]  96  97  98  99 100 101 104 105 109 110 111 112 113 114 115 120 121 126 127
## [58] 129 131 132 136 137 138 140 142 144 145 146 149 158 161 164 173 177 180 182
## [77] 187 196 197 200 201 202 205 211 212 215 218 220
```

```
  zimp <- rep(NA, n)
  zimp[ic] <- zmed[ic]
  zimp[-ic] <- z[-ic]  # zimp = failure times+imputed censored times
  delta1 = rep(1,n)  # after impute, all of times are observed then we made delta1.
  #
  curve2 = survfit(Surv(zimp,delta1) ~ x); curve2     # Bayesian Imputation
```

```
## Call: survfit(formula = Surv(zimp, delta1) ~ x)
##
##       n events median 0.95LCL 0.95UCL
## x=1  66     66    188     144     274
## x=2 154    154    152     126     188
```

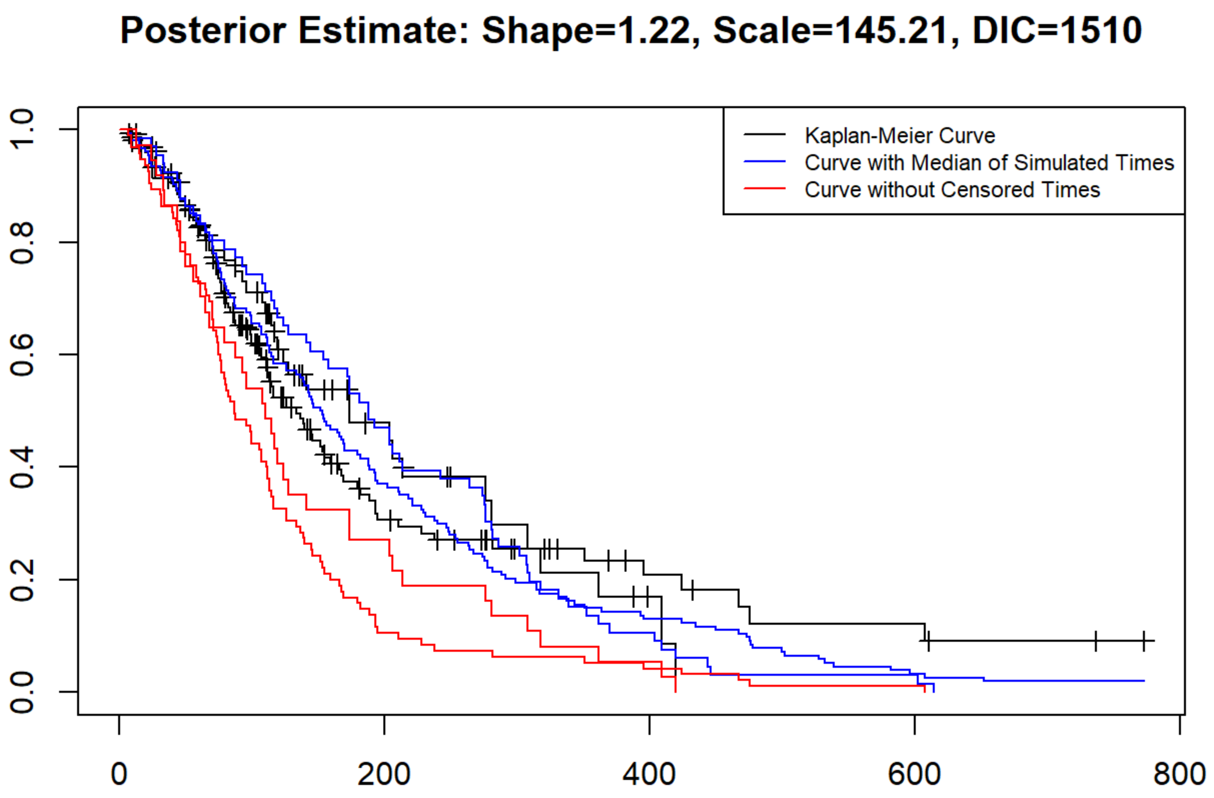
```
  lines(curve2, mark.time = TRUE, col = "Blue", lty = 1)

  # Curve without Censored Times
  tOC = z[delta==1]  # number of observed times
  deltaOC = rep(1, length(tOC))
  length(deltaOC)
```

```
## [1] 132
```

```
  curve3 = survfit(Surv(tOC, deltaOC) ~ x[delta==1]); curve3       # Omitting_Censored
```

```
## Call: survfit(formula = Surv(tOC, deltaOC) ~ x[delta == 1])
##
##                   n events median 0.95LCL 0.95UCL
## x[delta == 1]=1 37     37    110      79     174
## x[delta == 1]=2 95     95     87      76     112
```

```
  lines(curve3, mark.time = TRUE, col = "Red", lty = 1)

  legend("topright", c("Kaplan-Meier Curve", "Curve with Median of Simulated Times", "Curve with
out Censored Times"),
         lty= 1, col = c("black", "Blue", "Red"), cex = 0.7)
```

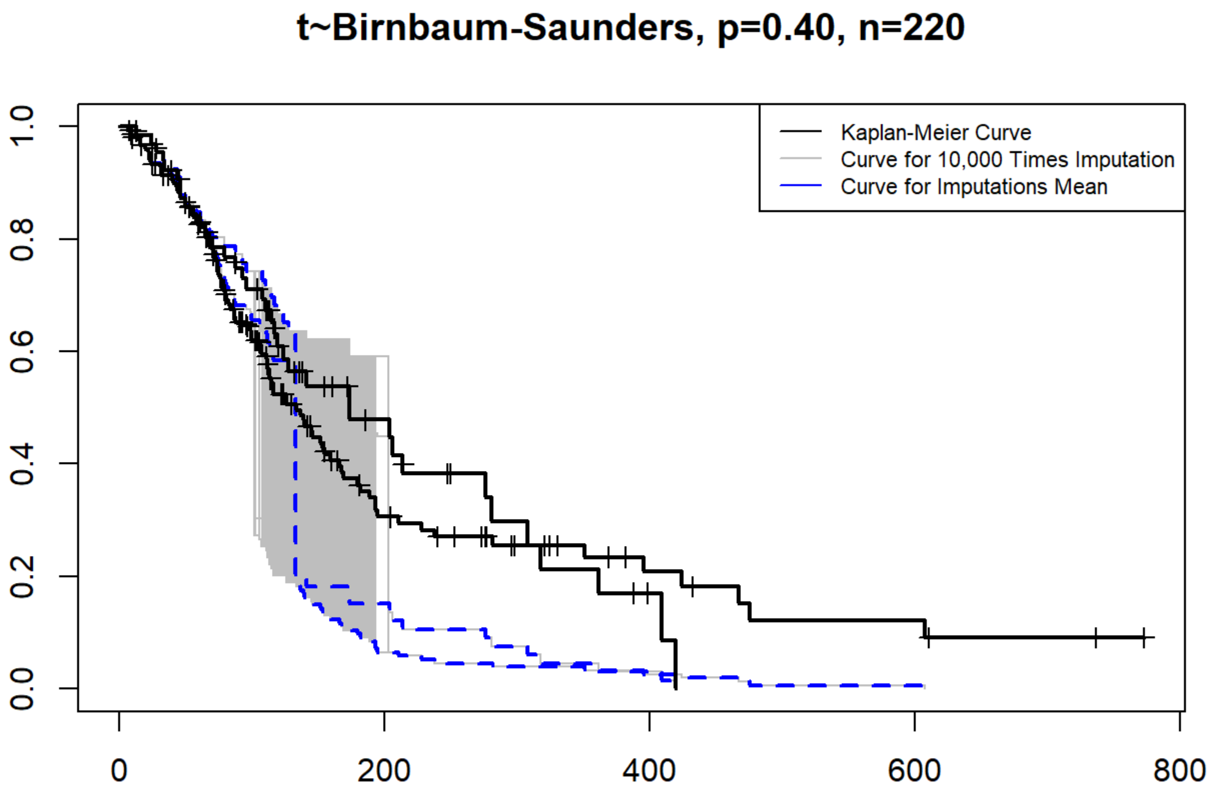


Fiqure 13 in the paper.

```
  # Kaplan-Meier Curve
  curve1 = survfit(Surv(z,delta) ~ x); curve1
```

```
## Call: survfit(formula = Surv(z, delta) ~ x)
##
##       n events median 0.95LCL 0.95UCL
## x=1  66     37    174     119     308
## x=2 154     95    134     112     166
```

```
  plot(curve1, mark.time = TRUE,lty = 1, lwd=2, col = "black",
       main = paste("t~Birnbaum-Saunders, p=0.40, n=220"))  #KM_Estimation
  # Curve with Median of Simulated Times
  # simulation
  for (i in 1:nrow(impsim)) {
    timp[ic] <- impsim[i,]
    kmi = survfit(Surv(timp,delta1) ~ x)
    lines(kmi, mark.time = TRUE, col = "gray", lty = 1)     # n time Imputation
    #Sys.sleep(.5)
  }
  # Curve for Imputations Mean
  timp[ic] <- colMeans(impsim)
  kmmean = survfit(Surv(timp,delta1) ~ x)
  lines(kmi, mark.time = TRUE, col = "blue", lty = 2, lwd = 2)  # Mean of n times Imputation

  lines(curve1, mark.time = TRUE,lty = 1, lwd=2, col = "black",
       main = paste("t~Birnbaum-Saunders, p=0.40, n=220"))  #KM_Estimation

  legend("topright",
         c("Kaplan-Meier Curve", "Curve for 10,000 Times Imputation", "Curve for Imputations Mea
n"),
         lty = 1, col = c("Black", "gray","blue"), cex = .7)
```



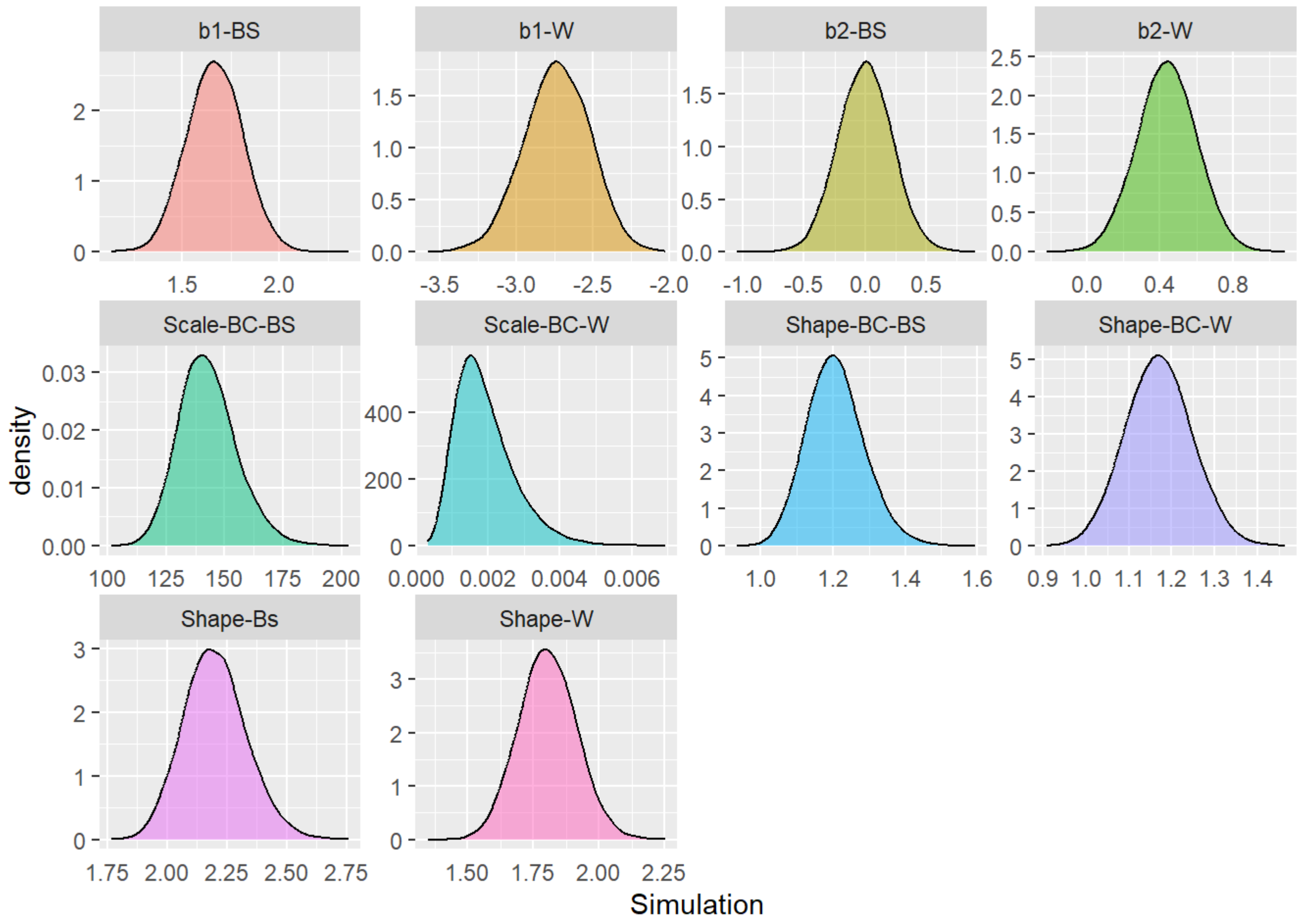
5- Convergence Geweke Diagnostics: Fig 4 in the paper.

5-1 Posterior Density Plot

```
library(coda)
library(ggplot2)
```

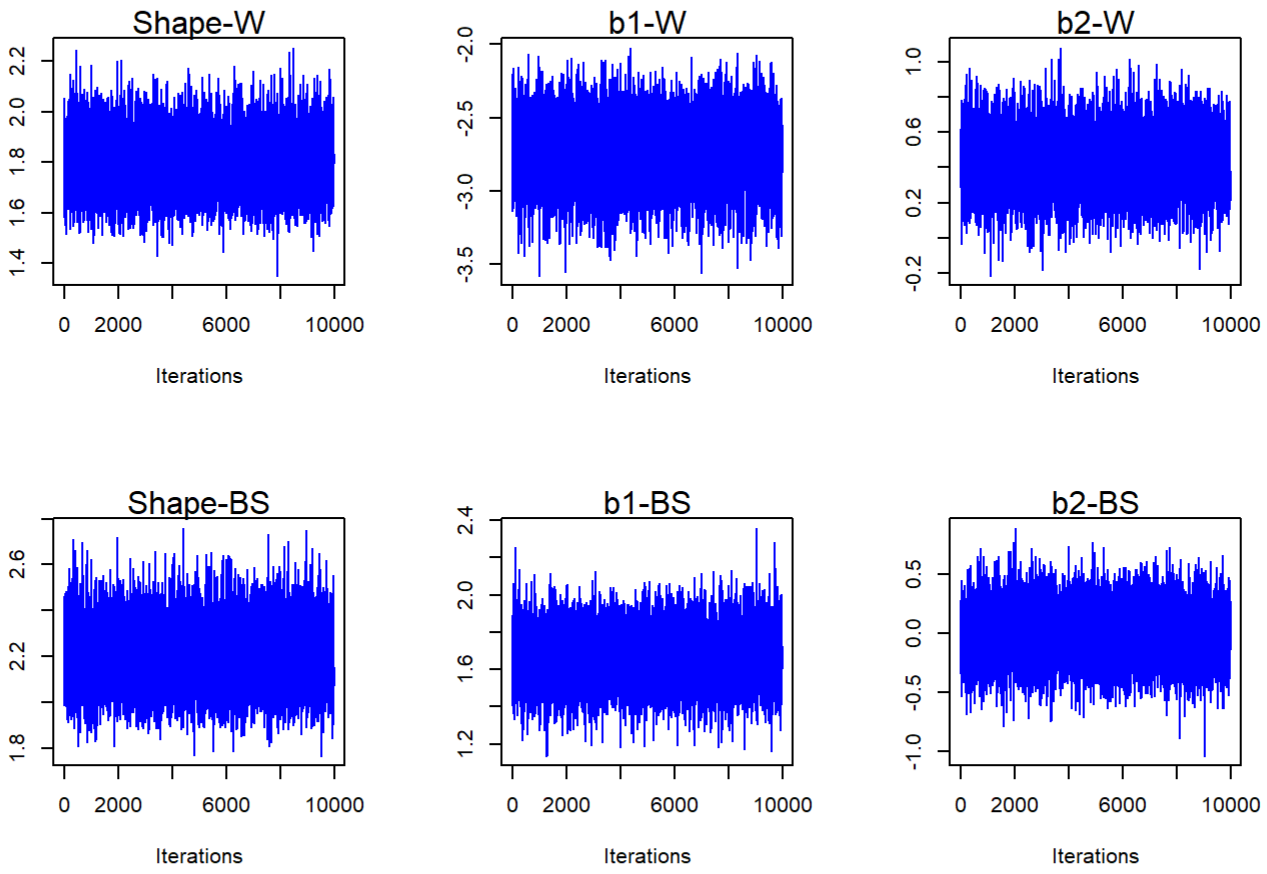
```
## Warning: package 'ggplot2' was built under R version 4.3.2
```

```
Shape_W <- c(read.csv("matparsim1.csv")[,2])
b1_W <- c(read.csv("matparsim1.csv")[,3])
b2_W <- c(read.csv("matparsim1.csv")[,4])
Shape_BS <- c(read.csv("matparsim2.csv")[,2])
b1_BS <- c(read.csv("matparsim2.csv")[,3])
b2_BS <- c(read.csv("matparsim2.csv")[,4])
Shape_BC_W <- c(read.csv("matparsim3.csv")[,2])
Scale_BS_W <- c(read.csv("matparsim3.csv")[,3])
Shape_BC_BS <- c(read.csv("matparsim4.csv")[,2])
Scale_BC_BS <- c(read.csv("matparsim4.csv")[,3])

simulation <- c(Shape_W, b1_W, b2_W,
                Shape_BS, b1_BS, b2_BS,
                Shape_BC_W,Scale_BS_W,
                Shape_BC_BS, Scale_BC_BS
                )
tot_matparsim <- data.frame(Simulation = simulation,
                            Parameter = rep(c("Shape-W", "b1-W","b2-W",
                                              "Shape-Bs", "b1-BS", "b2-BS",
                                              "Shape-BC-W", "Scale-BC-W",
                                              "Shape-BC-BS", "Scale-BC-BS"),
                            each = 10000))

Dens <- ggplot(data=tot_matparsim, aes(x=Simulation, group = Parameter, fill = Parameter)) +
                geom_density(alpha = 0.5, adjust = 1.5) + theme_gray() +
                theme(legend.position="none", panel.spacing = unit(0.1, "lines"),
                axis.ticks.x=element_blank())  + facet_wrap(~Parameter, scales = "free")

Dens
```

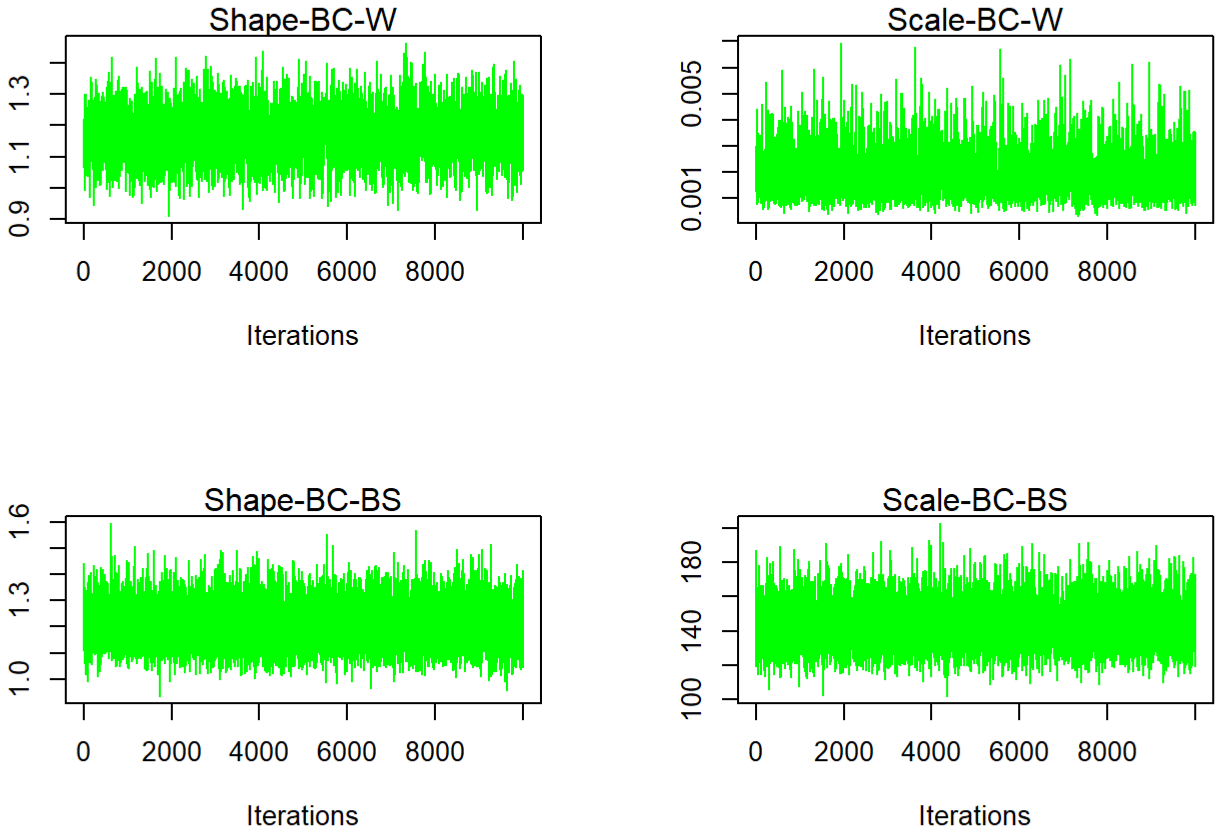


5-2 Trace Plot for all of Senrios. Fig 5 in the paper.

```
tracedata <- data.frame(read.csv('matparsim1.csv')[,2:4],read.csv('matparsim2.csv')[,2:4],
                        read.csv('matparsim3.csv')[,2:3],read.csv('matparsim4.csv')[,2:3])
names(tracedata) <- c('Shape_W', 'b1_W', 'b2_W',
                      'Shape_BS', 'b1_BS','b2_BS',
                      'Shape_BC_W','Scale_BC_W',
                      'Shape_BC_BS', 'Scale_BC_BS')

layout(matrix(c(1, 2, 3, 4, 5, 6), ncol= 3, nrow = 2, byrow = TRUE))
traceplot(as.mcmc(tracedata[,1]), col = "blue")
mtext("Shape-W", side = 3)
traceplot(as.mcmc(tracedata[,2]), col = "blue")
mtext("b1-W", side = 3)
traceplot(as.mcmc(tracedata[,3]), col = "blue")
mtext("b2-W", side = 3)
traceplot(as.mcmc(tracedata[,4]), col = "blue")
mtext("Shape-BS", side = 3)
traceplot(as.mcmc(tracedata[,5]), col = "blue")
mtext("b1-BS", side = 3)
traceplot(as.mcmc(tracedata[,6]), col = "blue")
mtext("b2-BS", side = 3)
```

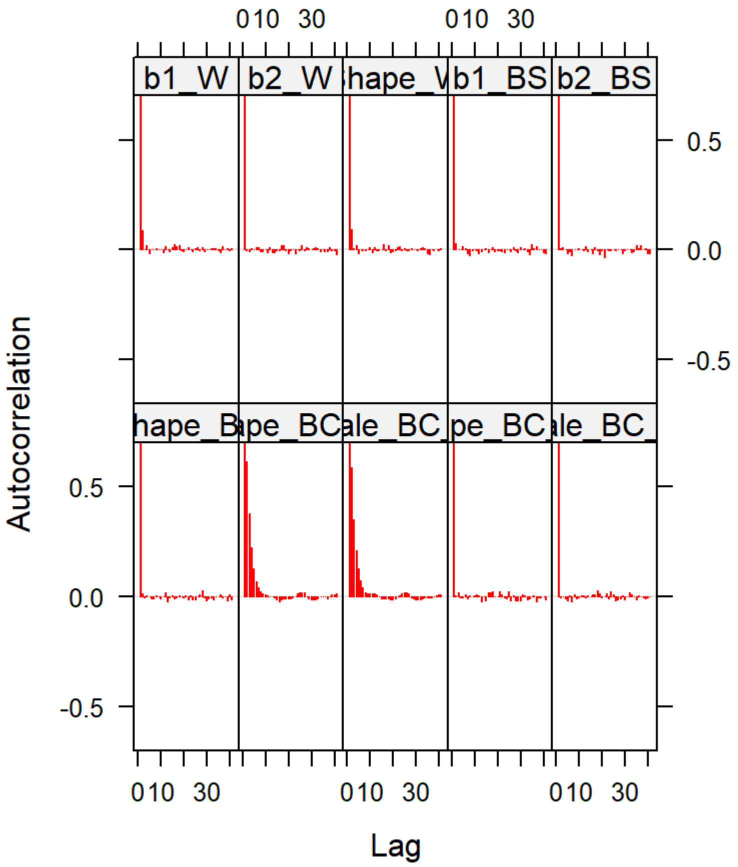


```
layout(matrix(c(1, 2, 3, 4), ncol= 2, nrow = 2, byrow = TRUE))
traceplot(as.mcmc(tracedata[,7]), col = "green")
mtext("Shape-BC-W", side = 3)
traceplot(as.mcmc(tracedata[,8]), col = "green")
mtext("Scale-BC-W", side = 3)
traceplot(as.mcmc(tracedata[,9]), col = "green")
mtext("Shape-BC-BS", side = 3)
traceplot(as.mcmc(tracedata[,10]), col = "green")
mtext("Scale-BC-BS", side = 3)
```



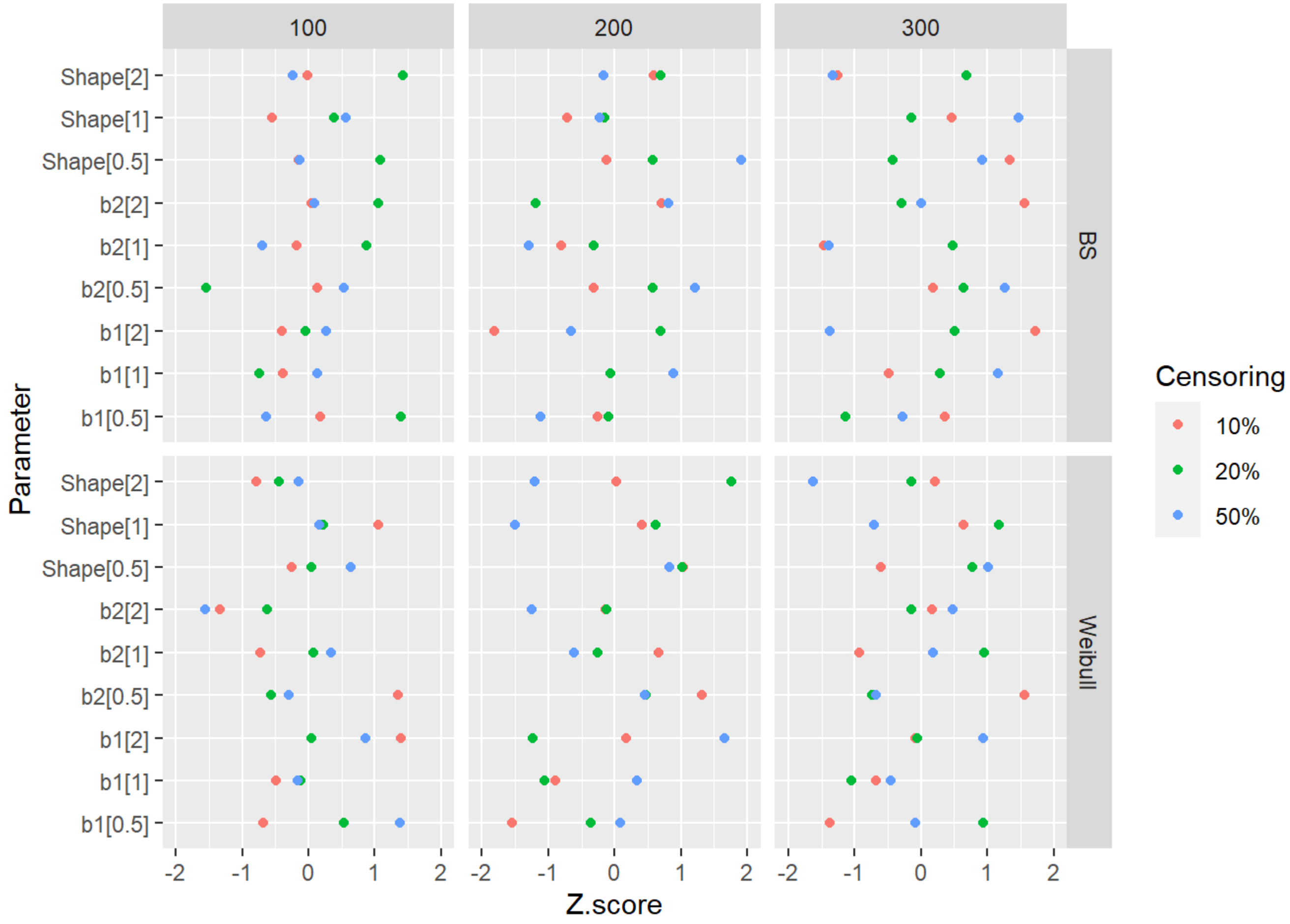
5-3 ACF PLOT for all of Scenarios. Fig 3 in the paper.

```
 tracedata1 <- data.frame(read.csv('matparsim3.csv')[,2:3],
                          read.csv('matparsim4.csv')[,2:3],
                          read.csv('matparsim1.csv')[,2:4],
                          read.csv('matparsim2.csv')[,2:4])
 names(tracedata1) <- c('Shape_BC_W','Scale_BC_W',
                        'Shape_BC_BS', 'Scale_BC_BS',
                        'Shape_W', 'b1_W', 'b2_W',
                        'Shape_BS', 'b1_BS','b2_BS')
tracedata2 <- tracedata1[, c(8,1,2,3,4,6,7,5,9,10)]
acfplot(as.mcmc(tracedata2), col = "red")
```

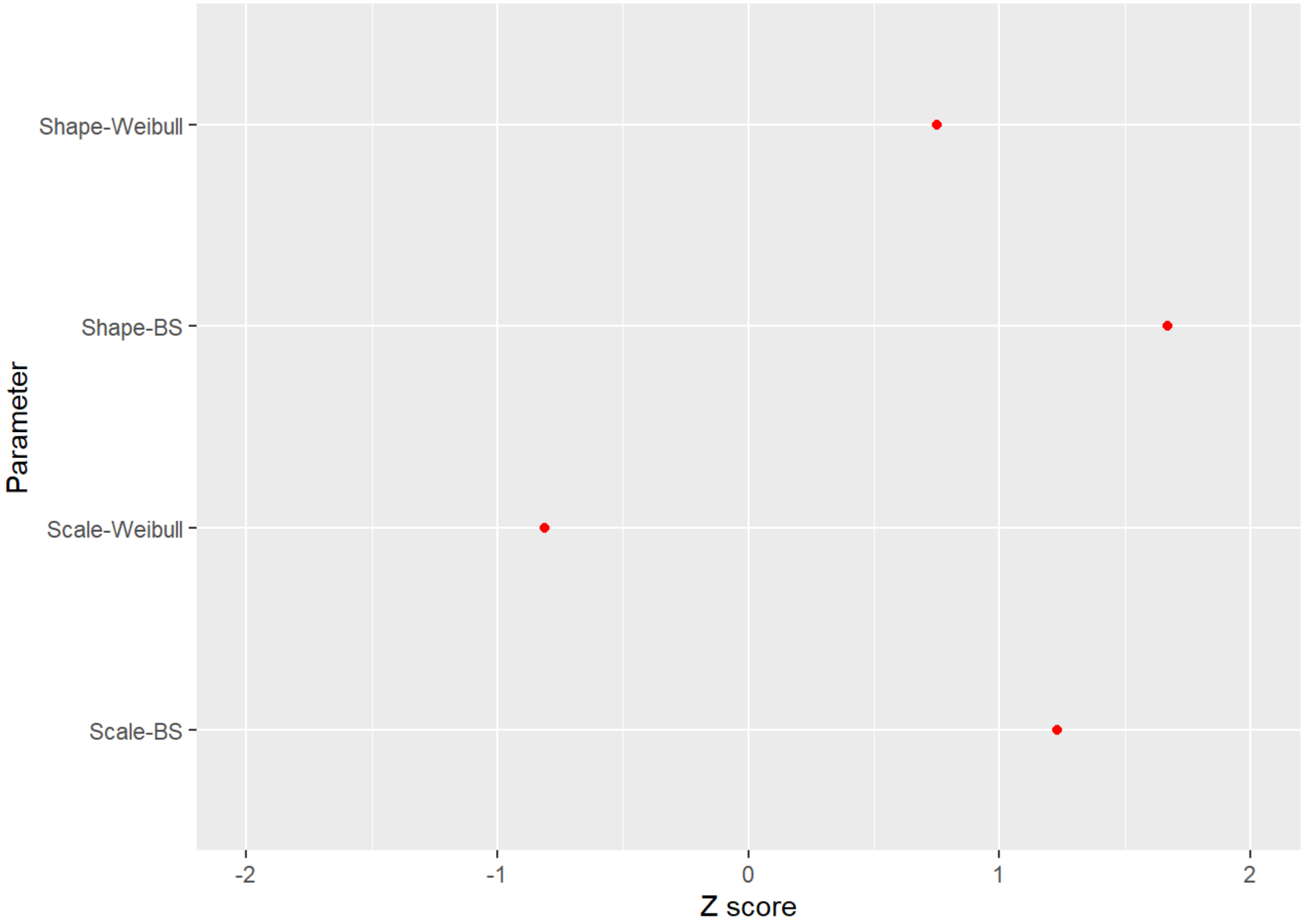


5-4 Geweke Diagnostics. Figures 6 and 7 in the paper.

```
## Weibull & BS Scenarios:
  matparsim1 <- read.csv("Convergence-Total.csv")
  matparsim1$Censoring <- as.factor(matparsim1$Censoring)

 library(ggplot2)
 library(gridExtra)
 ggplot(data = matparsim1, aes(x = Z.score, y = Parameter, color = Censoring)) +
       geom_point() + xlim(-2,2) + facet_grid(Scenarios~Sample.Size)
```



```
# Convergence Plot for Breast Cancer dataset.
 library(ggplot2)
 conv.data <- read.csv("convergence data - BC.csv")
 graph1 <- ggplot(data = conv.data, aes(x = Zscore, y = Parameter)) +
           geom_point(color = 'red')
 graph2 <- graph1 + xlim(-2,+2) + labs(x = "Z score" , y = "Parameter")
 graph2
```


Shape-Weibull
Shape-BS
Scale-Weibull
Scale-BS
Parameter
-2
-1
0
1
2
Z score