Institute of Actuaries of India

Subject CS2B – Risk Modelling and Survival Analysis (Paper B)

July 2022 Examination

INDICATIVE SOLUTION

Introduction

The indicative solution has been written by the Examiners with the aim of helping candidates. The solutions given are only indicative. It is realized that there could be other points as valid answers and examiner have given credit for any alternative approach or interpretation which they consider to be reasonable.

```
options(warn=-1)
```

Solution.1

```
# (i)

fMLE <- function(params) {
    f<- 1/params[2]*(1+params[3]*(maxima-params[1])/params[2])^(-1-1/params[3
])*exp(- (1+params[3]*(maxima-params[1])/params[2])^(-1/params[3]))
    lnf<-log(f)
    sum(-lnf)
}
```

[3]

```
# (ii)

maxima <- c(8,8,9)
alpha<-6; beta<-4; gamma<-5
p<-c(alpha,beta,gamma)
fMLE(p)
```

```
## [1] 11.32428
```

```
MLE<-nlm(fMLE,p);
MLE
```

```
## $minimum
## [1] -2.456456
##
## $estimate
## [1] 8.707082 3.066028 4.333867
##
## $gradient
## [1]  470.02988 -101.66352   69.47203
##
## $code
## [1] 2
##
## $iterations
## [1] 26
```

[4]

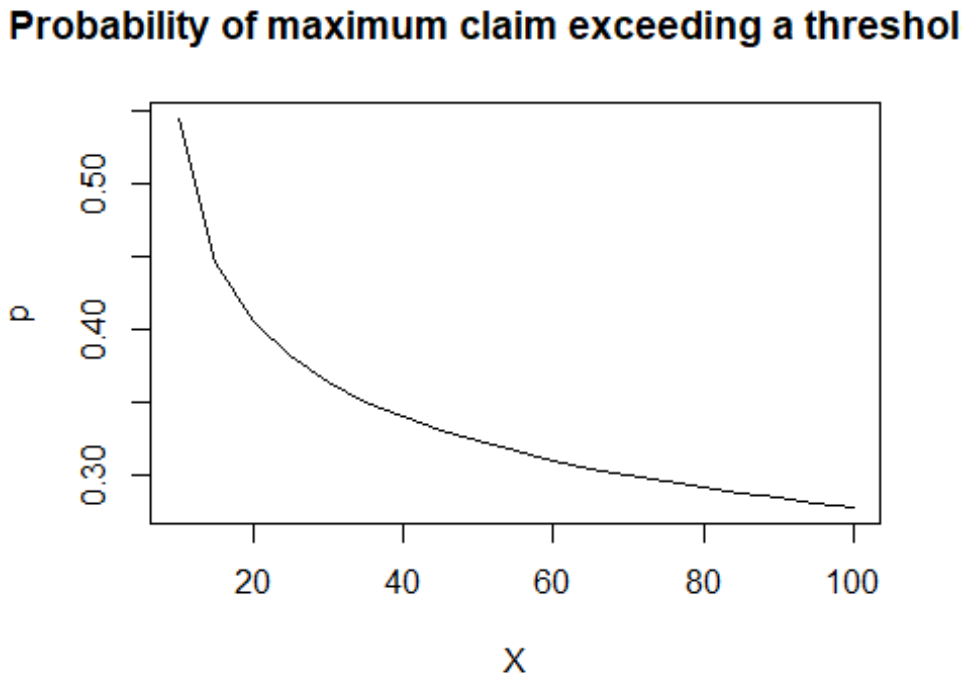
```
# (iii)

alpha <- MLE$estimate[1]
beta <- MLE$estimate[2]
gamma <- MLE$estimate[3]

probs<-c()
for (i in seq(10,100,5)) {
    m<-i
    p <- 1-exp(-(1+gamma*(m-alpha)/beta)^(-1/gamma))
    probs<-c(probs,p)
}

output <- matrix(NA, nrow = 19, ncol = 2)
output<-as.data.frame(output)
```

```
names(output)<-c("X","p")
output$X<-seq(10,100,5)
output$p<-probs
output
```

```
##      X         p
## 1   10 0.5446814
## 2   15 0.4452690
## 3   20 0.4056946
## 4   25 0.3813271
## 5   30 0.3639293
## 6   35 0.3505069
## 7   40 0.3396414
## 8   45 0.3305517
## 9   50 0.3227634
## 10  55 0.3159676
## 11  60 0.3099520
## 12  65 0.3045650
## 13  70 0.2996946
## 14  75 0.2952558
## 15  80 0.2911825
## 16  85 0.2874225
## 17  90 0.2839340
## 18  95 0.2806827
## 19 100 0.2776403
```

[6]

```
# (iv)

plot(output, type = "l", xlab = "X", ylab = "p", main = "Probability of maxim
um claim exceeding a threshold X")
```



[4]

```
# (v)
# If gamma is positive, the distribution is a Freschet distribution
# The Freschet distribution has a long, power-law tail that slowly converges
to 1. 1-CDF slowly converges to zero
# The probability of a maximum loss exceeding a given threshold always
increases inversely with Threshold
```

[3]

[20 Marks]

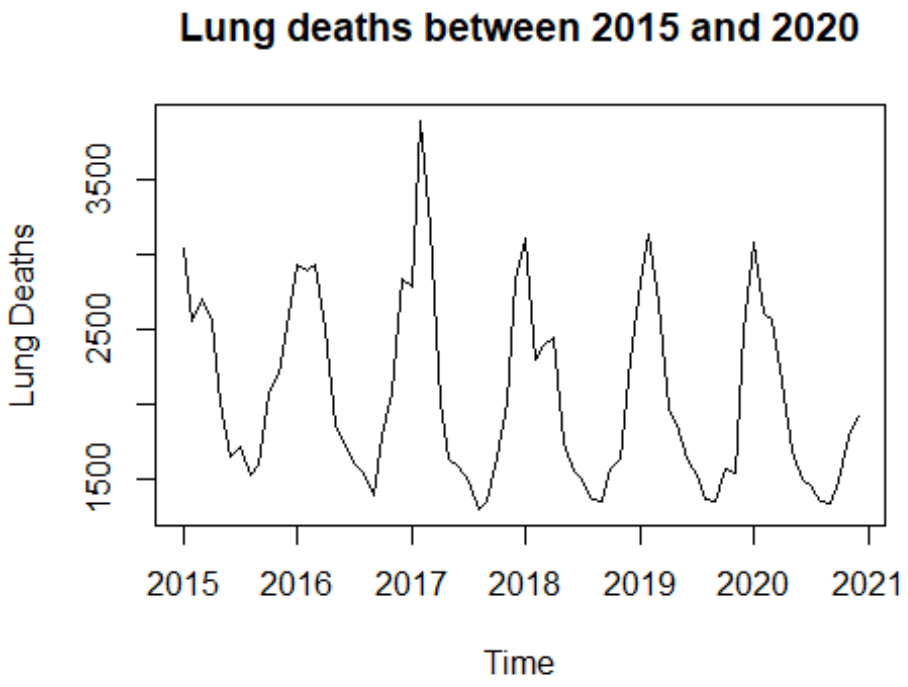
Solution 2

```
# (i)

Lung_Deaths <- read.csv("D:\\IAI Question Paper\\July 2022\\CS2B_Final\\Lung_
Deaths.csv")
Lung_Deaths<-ts(Lung_Deaths, start =c(2015,1), frequency = 12)
Lung_Deaths
```

```
##       Jan  Feb  Mar  Apr  May  Jun  Jul  Aug  Sep  Oct  Nov  Dec
## 2015 3035 2552 2704 2554 2014 1655 1721 1524 1596 2074 2199 2512
## 2016 2933 2889 2938 2497 1870 1726 1607 1545 1396 1787 2076 2837
## 2017 2787 3891 3179 2011 1636 1580 1489 1300 1356 1653 2013 2823
## 2018 3102 2294 2385 2444 1748 1554 1498 1361 1346 1564 1640 2293
## 2019 2815 3137 2679 1969 1870 1633 1529 1366 1357 1570 1535 2491
## 2020 3084 2605 2573 2143 1693 1504 1461 1354 1333 1492 1781 1915
```

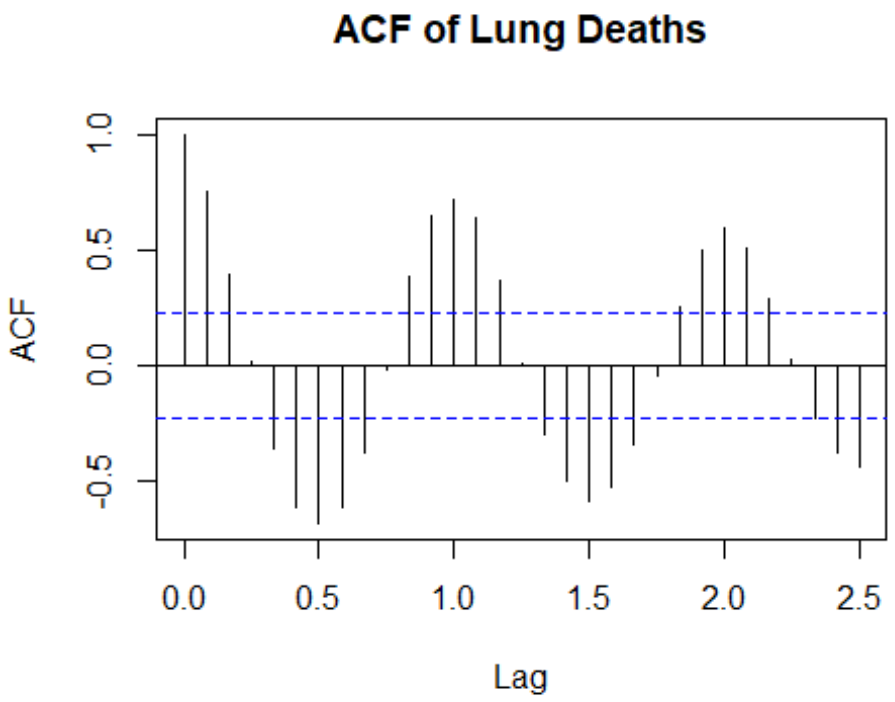
[2]

```
# (ii)

plot(Lung_Deaths, ylab = "Lung Deaths", main = "Lung deaths between 2015 and
2020")
```

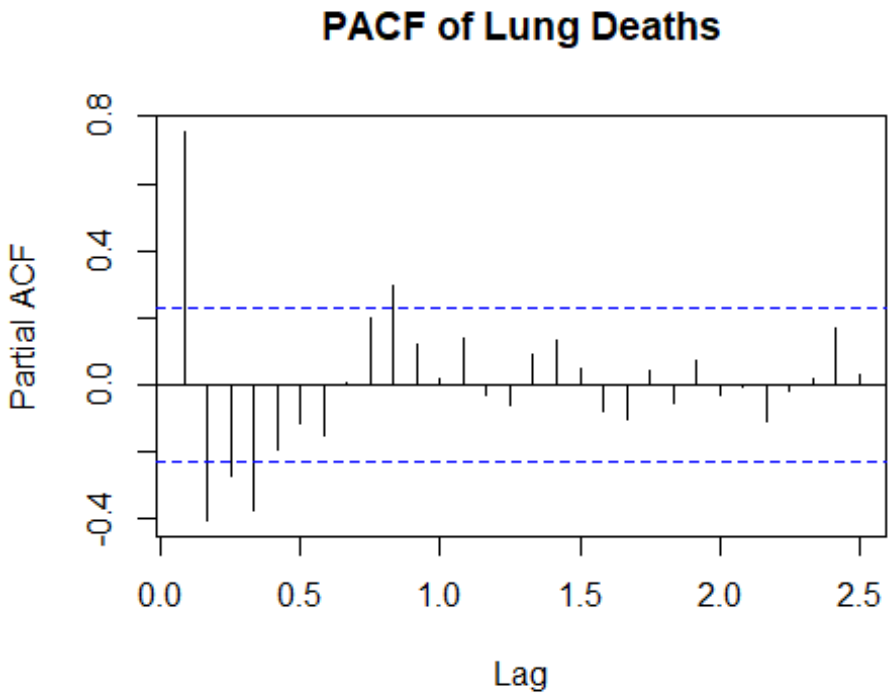


[3]

```r
# (iii)

acf(Lung_Deaths, lag.max = 30,main = "ACF of Lung Deaths")
```




```r
pacf(Lung_Deaths, lag.max = 30,main = "PACF of Lung Deaths")
```

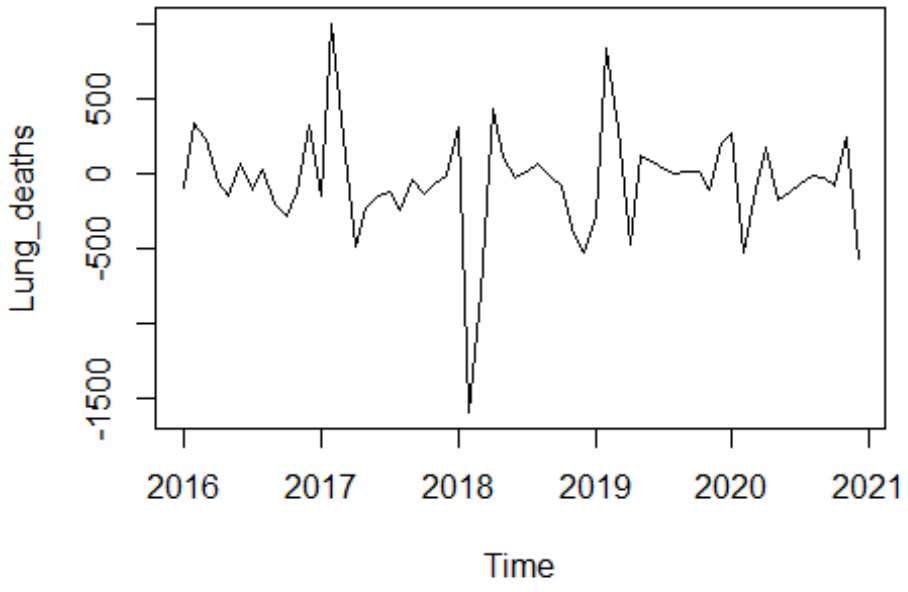


[6]

```
# (iv)

# There is seasonality in monthly data for which high values tend always to
occur in some particular months and low values tend always to occur in other
particular months. In this case, S = 12 (months per year) is the span of the
periodic seasonal behavior
# If significant correlation persists over many lags in the ACF functions, It
could either be a truly random autocorrelation in the series or there is a
fixed effect or trend that hasn't been removed (i.e. the series is not
stationary). Hence non stationary data
# Time Series of Lung Deaths is establishing seasonality which is visible
from highs and lows at uniform intervals of time
# ACF is exhibiting a high positive serial correlation at Lag 1 indicating
the presence of seasonality in the data
```

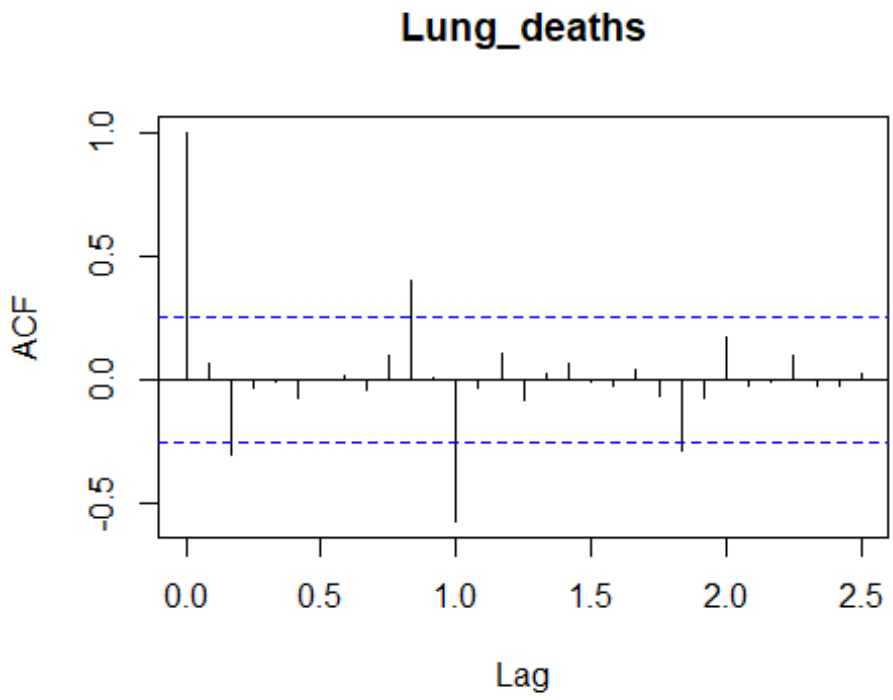
[3]

```
# (v)

diff12=diff(Lung_Deaths, lag = 12, differences=1)
plot(diff12)
```

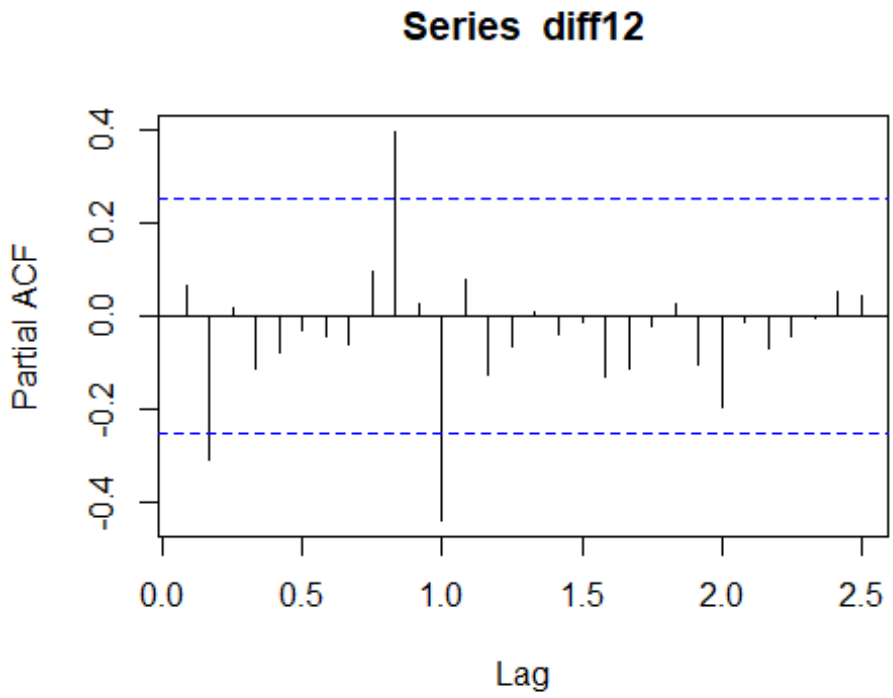


[4]

```
# (vi)
acf(diff12,lag.max = 30)
```




```
pacf(diff12,lag.max = 30)
```



[4]

```
# (vii)

# One summary statistic of a stationary time series is the auto-correlation f
unction, or the ACF
```

```
# If significant correlation persists over many lags in the ACF functions, It
could either be a truly random autocorrelation in the series or there is a fi
xed effect or trend that hasn't been removed (i.e. the series is not stationa
ry). Here that is not the case and hence stationary data
# Even after the seasonal difference, we observe significant correlation with
lag 12 both in the ACF and the PACF, indicating seasonal AR first order could
be an appropriate model for the data
#The blue dotted lines on the ACF and PACF indicate cut-offs for significance
.
#For a stationary time-series the ACF should decay to zero quickly and displa
y no signs of oscillation.
#The ACF looks to cut out at lag 2 and does not contain any periodic oscillat
ion so this would indicate stationarity.
#The PACF shows no significance past lag 2. This again, indicates stationarit
y.
```

[3]

[25 Marks]

Solution 3

```
# (i)
dpois(2,7.5)
## [1] 0.0155555
```

[2]

```
# (ii)
dpois(7,10)*dpois(15,20)

## [1] 0.004652489
```

[4]

```
# (iii)
probs<-c()
for (i in 0:20) {
    p<-dpois(i,5)
    probs<-c(probs,p)
}

output <- matrix(NA, nrow = 21, ncol = 2)
output<-as.data.frame(output)
names(output)<-c("Customers","Probability")
output$Customers<-0:20
output$Probability<-probs
output

##    Customers  Probability
## 1          0 6.737947e-03
## 2          1 3.368973e-02
## 3          2 8.422434e-02
## 4          3 1.403739e-01
## 5          4 1.754674e-01
## 6          5 1.754674e-01
## 7          6 1.462228e-01
## 8          7 1.044449e-01
## 9          8 6.527804e-02
```

```
## 10         9 3.626558e-02
## 11        10 1.813279e-02
## 12        11 8.242177e-03
## 13        12 3.434240e-03
## 14        13 1.320862e-03
## 15        14 4.717363e-04
## 16        15 1.572454e-04
## 17        16 4.913920e-05
## 18        17 1.445271e-05
## 19        18 4.014640e-06
## 20        19 1.056484e-06
## 21        20 2.641211e-07
```

[5]

[11 Marks]

Solution 4

```
# (i)
# Estimated Integrated Hazard:

# Estimated variance of the estimator of the integrated hazard is calculated
as:
```

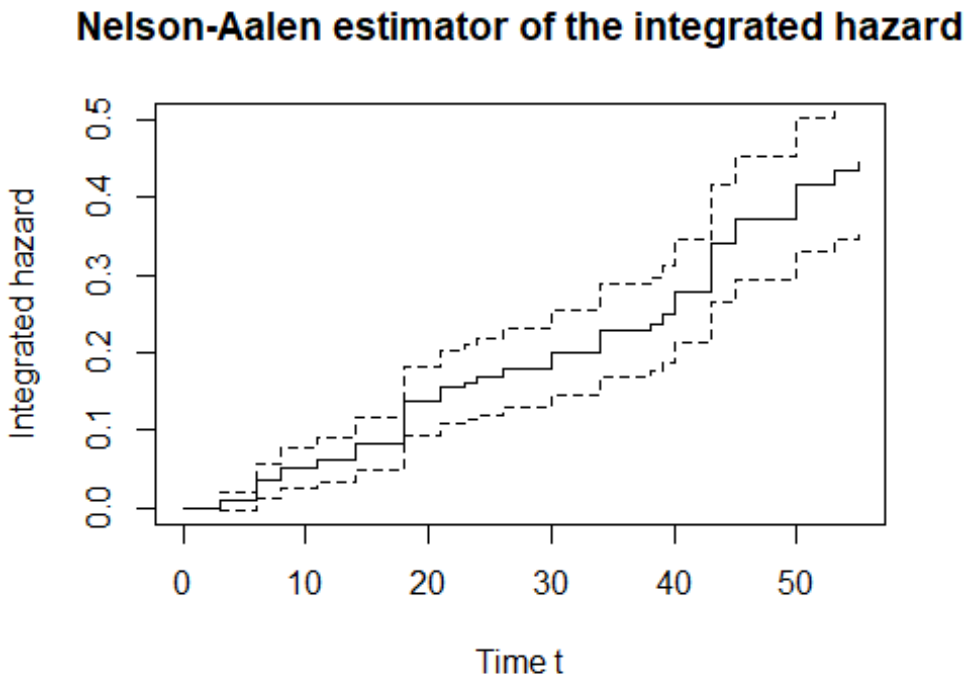
[3]

```
# (ii)

file <- read.csv("D:\\IAI Question Paper\\July 2022\\CS2B_Final\\data_1.csv")
dj <- file$dj
nj <- file$nj
tj <- file$tj
names(file)[5]<-"Censored"
Lambda=cumsum(dj/nj)
Var = cumsum(dj*(nj-dj)/nj^3)
file$Lambda = Lambda
file$Variance = Var
file
##     j tj  nj dj Censored     Lambda     Variance
## 1   1  3 200  2       NA 0.01000000 0.0000495000
## 2   2  6 198  5       NA 0.03525253 0.0001738173
## 3   3  8 193  3       NA 0.05079657 0.0002531045
## 4   4 11 190  2       NA 0.06132288 0.0003079230
## 5   5 14 188  4        5 0.08259948 0.0004186884
## 6   6 18 179 10       NA 0.13846540 0.0007133528
## 7   7 21 169  3       NA 0.15621688 0.0008165266
## 8   8 23 166  1       NA 0.16224098 0.0008525977
## 9   9 24 165  1       NA 0.16830158 0.0008891060
## 10 10 26 164  2       15 0.18049670 0.0009625597
## 11 11 30 147  3       NA 0.20090487 0.0010985574
## 12 12 34 144  4       NA 0.22868264 0.0012861003
## 13 13 38 140  1       NA 0.23582550 0.0013367563
## 14 14 39 139  2       NA 0.25021399 0.0014387812
## 15 15 40 137  4        2 0.27941107 0.0016456761
## 16 16 43 131  8       NA 0.34047977 0.0020833808
## 17 17 45 123  4        4 0.37300010 0.0023391756
## 18 18 50 115  5       NA 0.41647836 0.0027008095
```

```
## 19 19 53 110  2       NA 0.43466018 0.0028630935
## 20 20 55 108  1       NA 0.44391944 0.0029480336
```

[3]

```
# (iii)
sdLambda=sqrt(Var)
plot(c(0,tj),c(0,Lambda),xlim=c(0,55),ylim=c(0,0.5),type="s",
         main="Nelson-Aalen estimator of the integrated hazard",
         xlab="Time t",ylab="Integrated hazard")
lines(tj,Lambda-1.64485*sdLambda, type="s",lty=2)
lines(tj,Lambda+1.64485*sdLambda, type="s",lty=2)
```



[6]

[12 Marks]

Solution 5

```
# (i)

trunc_logN <- function(k, mu, sig, L, U){

    Lk <- (log(L) - mu)/sig - k*sig
    Uk <- (log(U) - mu)/sig - k* sig
    cumulate <- exp(k*mu+.5*k^2*sig^2) * (pnorm(Uk) - pnorm(Lk))
    print(paste("First term:", exp(k*mu+.5*k^2*sig^2)))
    print(paste("Phi(Uk):  ",  pnorm(Uk)))
    print(paste("Phi(Lk):  ",  pnorm(Lk)))

    print(paste("Final Value:", cumulate))

}
```

[4]

```
# (ii)

trunc_logN(1, 1.2, 0.7, 10,Inf)

## [1] "First term: 4.24185214282043"
## [1] "Phi(Uk):   1"
## [1] "Phi(Lk):   0.809246116780569"
## [1] "Final Value: 0.809149768285661"

trunc_logN(2, 1.2, 0.7, 10,Inf)

## [1] "First term: 29.3707711132894"
## [1] "Phi(Uk):   1"
## [1] "Phi(Lk):   0.569507941751428"
## [1] "Final Value: 12.6438837089077"
```

[3]

```
# (iii)

# First moment of a normal distribution = mu = 1.2
# Second moment of a normal distribution = 0.7^2+1.2^2 = 1.93

# The question should have been "Log normal distribution" instead of normal d
istribution

# Assuming Lognormal distribution

# Moments of Lognormal distribution are

exp(1.2+0.5*0.7^2)

## [1] 4.241852

exp(2*1.2+0.7^2)*(exp(0.7^2)-1)+(exp(1.2+0.5*0.7^2))^2

## [1] 29.37077

trunc_logN(1, 1.2, 0.7, 0,Inf)

## [1] "First term: 4.24185214282043"
## [1] "Phi(Uk):   1"
## [1] "Phi(Lk):   0"
## [1] "Final Value: 4.24185214282043"

trunc_logN(2, 1.2, 0.7, 0,Inf)

## [1] "First term: 29.3707711132894"
## [1] "Phi(Uk):   1"
## [1] "Phi(Lk):   0"
## [1] "Final Value: 29.3707711132894"
```

[2]

```
# (iv)

# The results are not comparable as (ii) is associated with Lognormal distrib
ution while (iii) is associated with normal distribution
```

```
# Assuming (iii) is log normal distribution
# The mean and variance of a truncated distribution are very much lower than
the non-truncated because
# (i) Beyond 10, the probabilities are very less and the actual range of valu
es in excess of 10 is very limited
# (ii) The consistency improves because of a narrower range and hence lower v
ariance and second order movement
```

[3]

[12 Marks]

Solution 6

```
Cricket <- read.csv("D:\\IAI Question Paper\\July 2022\\CS2B_Final\\Cricket.c
sv")

# (i)

runs_avg <- aggregate(Runs~Init_group, data = Cricket, FUN = "mean")
runs_avg
```

```
##   Init_group      Runs
## 1  Alrounder 143.66667
## 2    Batsman 213.91892
## 3     Bowler  16.44444
```

```
wickets_avg <- aggregate(Wickets~Init_group, data = Cricket, FUN = "mean")
wickets_avg
```

```
##   Init_group   Wickets
## 1  Alrounder 4.3095238
## 2    Batsman 0.2432432
## 3     Bowler 9.0138889
```

[3]

```
# (ii)
Cricket1 <-Cricket
Cricket1<-Cricket1[,4:7]
```

[2]

```
# (iii)
Cricket1 = as.data.frame(scale (Cricket1))
set.seed(100)
clust_Cricket<-kmeans(Cricket1,3)
clust_Cricket
```

```
## K-means clustering with 3 clusters of sizes 39, 62, 87
##
## Cluster means:
##         Runs    Ave_Bat    Wickets   Econonmy
## 1 -0.4626058 -0.3628813  1.7012202 -0.8011925
## 2  1.1193502  1.0687167 -0.6221781  0.8735751
## 3 -0.5903229 -0.5989433 -0.3192247 -0.2633925
##
## Clustering vector:
##   [1] 3 2 3 3 3 2 1 3 2 3 2 3 2 3 3 2 2 1 3 3 2 1 3 1 2 3 3 3 3 3 2 3 2 2
```

```
3 1 2
##  [38] 3 3 2 1 2 2 3 3 2 3 3 2 3 1 2 3 2 3 3 1 2 2 1 1 3 3 1 3 3 2 1 1 3 3
2 3 3
##  [75] 3 2 1 3 1 2 1 3 3 3 1 2 1 3 2 3 3 3 3 1 1 1 1 3 2 2 2 1 3 2 2 2 3 3
3 2 1
## [112] 3 2 3 3 3 2 3 2 3 1 1 2 1 3 3 2 3 3 3 2 3 3 2 3 1 3 1 1 2 2 2 2 3 2
3 2 2
## [149] 2 3 3 3 2 3 3 2 3 3 2 3 2 1 2 3 1 2 3 2 2 1 1 1 2 3 3 3 1 1 2 3 3 3
3 3 2
## [186] 1 2 1
##
## Within cluster sum of squares by cluster:
## [1]  50.32514 141.94279 108.10393
##  (between_SS / total_SS =  59.8 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withi
nss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

[4]

```
# (iv)
Cricket$Clust_Membership <-clust_Cricket$cluster
```

[2]

```
# (v)
Cricket$Clust_Membership <-factor(Cricket$Clust_Membership, labels = c("Bowle
r","Batsman","Alrounder"))
```

[3]

```
# (vi)
table(Cricket$Clust_Membership,Cricket$Init_group)

##
##             Alrounder Batsman Bowler
##   Bowler             7       0     32
##   Batsman           11      51      0
##   Alrounder         24      23     40

error<- (32+11+24+23)/nrow(Cricket)
error

## [1] 0.4787234
```

[3]

```
# (vii)

# Possible reasons

# 1. Many batsman in realty might have played very few matches and scored ver
y few runs. They might not have got classified as batsmen by the cluster
# 2. Similarly many bowlers also
```

```
# 3. When there were only 42 players who were alrounders in the real data, cl
uster considered 87 as alrounders because the players who do not fall into ba
tsmen and bowers were treated as alrounders
# 4. No batsman gor classified as a bowler and vice versa indicating the mis-
classification is with respect to alrounders
# Rather than considering all players, if we would have considered players wh
o played a minimum number of matches, the clustering accuracy would have impr
oved significantly
```

[3]

[20 Marks]

****