Institute of Actuaries of India

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

November 2020 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.

Solution 1:

i)

Mort_Inv <- read.csv("D: /Mortality_Investigation.csv")
Mort_Inv$DoB<-as.Date(Mort_Inv$DoB)
Mort_Inv$DoJ<-as.Date(Mort_Inv$DoJ)
Mort_Inv$DoE<-as.Date(Mort_Inv$DoE)
head(Mort_Inv) [2]

prop.table(table(Mort_Inv$Exit_Reason)) [2]

```
> head(Mort_Inv)
  Life        DoB        DoJ        DoE Exit_Reason
1   A1 1981-12-12 2018-11-13 2018-12-31    Survived
2   A2 1981-05-22 2017-10-06 2018-12-31    Survived
3   A3 1978-08-11 2018-01-30 2018-12-31    Survived
4   A4 1980-05-24 2016-05-12 2016-05-13  Withdrawal
5   A5 1979-04-03 2017-07-25 2018-12-31    Survived
6   A6 1979-11-08 2016-08-02 2017-04-14       Death
> prop.table(table(Mort_Inv$Exit_Reason))

     Death   Survived Withdrawal
      0.31       0.40       0.29
```

[4]

ii)

Mort_Inv$Age_At_Entry<-round((Mort_Inv$DoJ-Mort_Inv$DoB)/365.25,4)
Mort_Inv$Age_At_Exit<-round((Mort_Inv$DoE-Mort_Inv$DoB)/365.25,4)
tail(Mort_Inv)

```
> tail(Mort_Inv)
    Life        DoB        DoJ        DoE Exit_Reason Age_At_Entry  Age_At_Exit
95   A95 1981-03-28 2016-03-06 2019-11-21    Survived 34.9405 days 38.6502 days
96   A96 1981-01-17 2018-04-04 2018-09-14       Death 37.2101 days 37.6564 days
97   A97 1980-01-17 2016-08-29 2016-09-18       Death 36.6160 days 36.6708 days
98   A98 1978-04-17 2016-06-14 2016-07-07  Withdrawal 38.1602 days 38.2231 days
99   A99 1978-06-12 2017-09-05 2019-11-25    Survived 39.2334 days 41.4538 days
100 A100 1980-06-29 2018-03-27 2019-10-04    Survived 37.7413 days 39.2635 days
```

[5]

iii)

mean(Mort_Inv$Age_At_Entry[Mort_Inv$Exit_Reason == "Death"])

mean(Mort_Inv$Age_At_Exit[Mort_Inv$Exit_Reason == "Death"])

```
> mean(Mort_Inv$Age_At_Entry[Mort_Inv$Exit_Reason == "Death"])
Time difference of 37.01715 days
> mean(Mort_Inv$Age_At_Exit[Mort_Inv$Exit_Reason == "Death"])
Time difference of 37.89168 days
```

[3]

iv)

sum((Mort_Inv$Age_At_Entry)<37&Mort_Inv$Age_At_Exit>38)

```
[1] 14
```

[4]

v)

sum((Mort_Inv$Age_At_Entry)>38|Mort_Inv$Age_At_Exit<37)

```
[1] 49
```

[4]

vi)

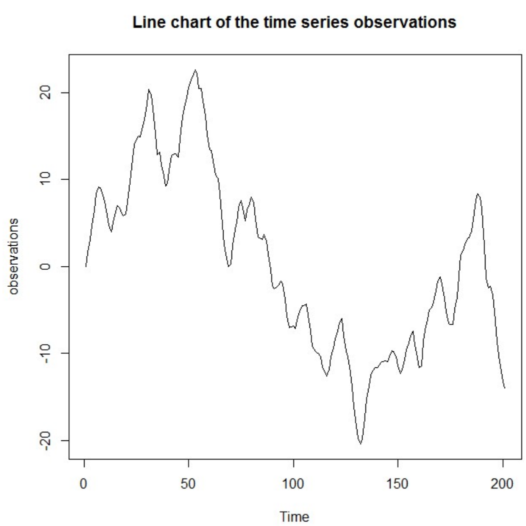
Mort_Inv$Contribution37<-ifelse((Mort_Inv$Age_At_Exit<37 |Mort_Inv$Age_At_Entry> 38),"No","Yes")
Mort_Inv$contribution37_Period<-ifelse(Mort_Inv$Contribution37 == "Yes",
(pmin(38,Mort_Inv$Age_At_Exit)- pmax(37,Mort_Inv$Age_At_Entry)),0)
sum(Mort_Inv$contribution37_Period)

```
[1] 27.4224
```

[7]
[27 Marks]

Solution 2:

i)
set.seed(100)
observations<-arima.sim(list(order = c(1,1,1), ar = 0.7, ma = 0.3), n = 200)
plot(observations, main = "Line chart of the time series observations")



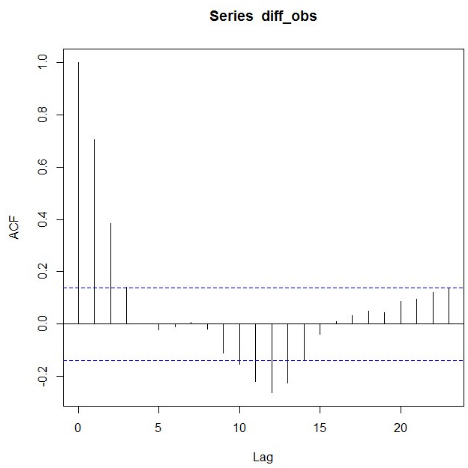
[3]

```
# The data is not stationary as we observe that the values are changing with time
# Upward Trend is observed in the data, which indicates the data being non stationary
# Mean and Standard Deviation are different at different points in time
```

[3]
[6]

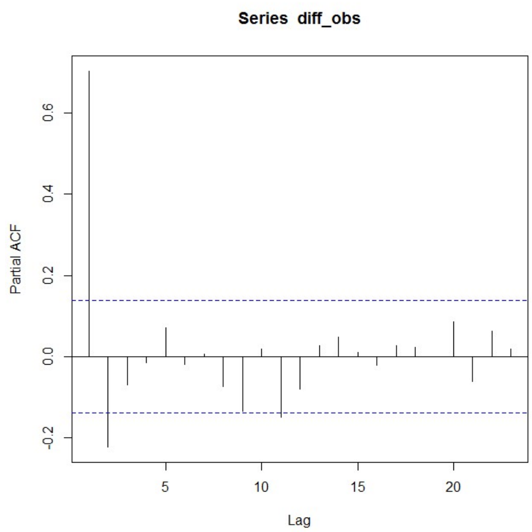
ii)

 # As the data is not stationary, we take the first difference of the observations
diff_obs<-diff(observations) [1]
acf(diff_obs)



[1.5]

pacf(diff_obs)



[1.5]

As both ACF and PACF are seen to have spikes only for the first two lags and they appear to tail off after that, ARMA(2,2) model appears to be the most appropriate model based on the ACF and PACF plots.

[1]

[5]

iii)

arima(diff_obs,order = c(1,0,0)) [2]

```
Call:
arima(x = diff_obs, order = c(1, 0, 0))

Coefficients:
         ar1  intercept
      0.7070    -0.0570
s.e.  0.0498     0.2284

sigma^2 estimated as 0.9177:  log likelihood = -275.55,  aic = 557.09
```

arima(diff_obs,order = c(2,0,0)) [1]

```
Call:
arima(x = diff_obs, order = c(2, 0, 0))

Coefficients:
         ar1      ar2  intercept
      0.8631  -0.2218    -0.0593
s.e.  0.0688   0.0693     0.1831

sigma^2 estimated as 0.8725:  log likelihood = -270.55,  aic = 549.1
```

arima(diff_obs,order = c(0,0,1)) [1]

```
Call:
arima(x = diff_obs, order = c(0, 0, 1))

Coefficients:
         ma1  intercept
      0.6436    -0.0655
s.e.  0.0434     0.1194

sigma^2 estimated as 1.06:  log likelihood = -289.89,  aic = 585.79
```

arima(diff_obs,order = c(1,0,1)) [1]

```
Call:
arima(x = diff_obs, order = c(1, 0, 1))

Coefficients:
         ar1     ma1  intercept
      0.5877  0.2533    -0.0578
s.e.  0.0754  0.0865     0.2002

sigma^2 estimated as 0.881:  log likelihood = -271.5,  aic = 550.99
```

```
# AIC is appearing the least for AR(2) model. The same is being by PACF graph
also.
```

[2]

[7]

iv)

model<-arima(diff_obs,order = c(2,0,0))
predict(model,n.ahead = 3)

```
$`pred`
Time Series:
Start = 202
End = 204
Frequency = 1
[1] -0.22671820 -0.05765193 -0.02073163

$se
Time Series:
Start = 202
End = 204
Frequency = 1
[1] 0.9340953 1.2338975 1.3271154
```

[4]

[22 Marks]

<u>Solution 3:</u>

i)

set.seed(100)
freq<-rpois(10000,0.75)
table(freq)

```
freq
   0    1    2    3    4    5
4761 3499 1328  327   70   15
```

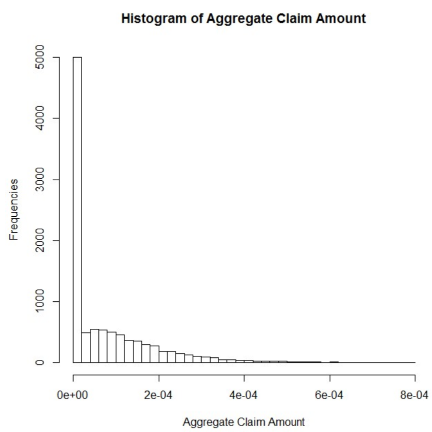
[4]

```
# There was a typo in the question. It should have been rate parameter is 1/20000 or scale parameter should have been 20000
# In case the students follow either one of the approaches, full marks will be awarded.
```

#Solution Assuming rate parameter = 20000

ii)

```
AggclaimAmount<-c()
for (i in 1:10000) {
  claimAmount<-sum(rgamma(freq[i],shape = 2, rate  = 20000))
  AggclaimAmount<-c(AggclaimAmount,claimAmount)
}
hist(AggclaimAmount, breaks =30, main = "Histogram of Aggregate Claim Amount", xlab = "Aggregate Claim Amount", ylab = "Frequencies")
```



[6]

iii)

```
mean_poisson<-0.75
mean_gamma<-2/(20000)
var_poisson<-0.75
var_gamma<-2/((20000)^2)

mean_aggregate<-mean_poisson*mean_gamma
mean_aggregate
```

```
[1] 7.5e-05
```

[3]

```
var_aggregate<-mean_poisson*var_gamma+var_poisson*mean_gamma^2
var_aggregate
```

```
[1] 1.125e-08
```

[2]

[5]

iv)

```
mean_claims_I<-c()
mean_claims_R<-c()
for (i in seq(50000,100000,5000)) {
  mean_claims_R<-c(mean_claims_R,mean(pmax(AggclaimAmount-i,0)))
  mean_claims_I<-c(mean_claims_I,mean(pmin(AggclaimAmount,i)))
}

# Mean Costs to the Insurers
mean_claims_I

[1] 7.395992e-05 7.395992e-05 7.395992e-05 7.395992e-05 7.395992e-05 7.395992e-05
 [7] 7.395992e-05 7.395992e-05 7.395992e-05 7.395992e-05 7.395992e-05

# Mean Costs to the Reinsurer
mean_claims_R

[1] 0 0 0 0 0 0 0 0 0 0 0
```

[6]

v)

```
mean_agg_cost<-mean(AggclaimAmount)

# 75% of the Aggregate claims cost

mean_Cost_Insurer<-mean_agg_cost*0.75

# 75% of the Aggregate claims cost = 5.546994e-05
# Retention limits should be much lesser than the limits specified in part (iv)
# Can be recalculated by considering a different range from 0.0001 to 0.0002
# If the student does not compute this range but mentions that no values from the range are applicable, then full marks should be awarded.

mean_claims_I<-c()
mean_claims_R<-c()
for (i in seq(0.0001,0.0002,0.00001)) {
  mean_claims_R<-c(mean_claims_R,mean(pmax(AggclaimAmount-i,0)))
  mean_claims_I<-c(mean_claims_I,mean(pmin(AggclaimAmount,i)))
}

mean_claims_I
[1] 4.201910e-05 4.482882e-05 4.740955e-05 4.979510e-05 5.198747e-05 5.400000e-05
 [7] 5.584620e-05 5.751952e-05 5.904900e-05 6.044518e-05 6.170202e-05

mean_claims_R
[1] 3.194081e-05 2.913110e-05 2.655037e-05 2.416482e-05 2.197244e-05 1.995992e-05
 [7] 1.811371e-05 1.644040e-05 1.491091e-05 1.351474e-05 1.225790e-05

Retention_Limit<-0.00015
Reinsurer_Claims<-pmax(AggclaimAmount-Retention_Limit,0)
#Proportion of Claims to be taken up by the reinsurer
sum(Reinsurer_Claims>0)/10000

[1] 0.1929
```

[5]

vi)

If for the part (v), the student identifies that no values from the range are applicable and stops here, full marks should be awarded

SD_Retention<-sd(Reinsurer_Claims)
SD_Retention

```
> SD_Retention
[1] 5.991976e-05
```

[2]

Skew_Retention<-mean((Reinsurer_Claims-mean(Reinsurer_Claims))^3)/(sd(Reinsurer_Claims))^3
Skew_Retention

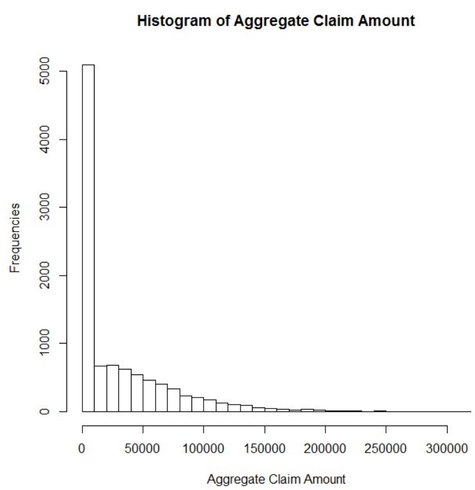
```
> Skew_Retention
[1] 4.305918
```

[2]
[4]

#Alternative Solution Assuming rate parameter = 1/20000

ii)

```
AggclaimAmount<-c()
for (i in 1:10000) {
  claimAmount<-sum(rgamma(freq[i],shape = 2, rate  = 1/20000))
  AggclaimAmount<-c(AggclaimAmount,claimAmount)
}
hist(AggclaimAmount, breaks =30, main = "Histogram of Aggregate Claim Amount", xlab = "Aggregate Claim Amount", ylab = "Frequencies")
```



[6]

iii)

```
mean_poisson<-0.75
mean_gamma<-2/(1/20000)
var_poisson<-0.75
var_gamma<-2/((1/20000)^2)

mean_aggregate<-mean_poisson*mean_gamma
```

mean_aggregate

```
[1] 30000
```

[3]

*var_aggregate<-mean_poisson*var_gamma+var_poisson*mean_gamma^2*
var_aggregate

```
[1] 1.8e+09
```

[2]
[5]

iv)

```
mean_claims_I<-c()
mean_claims_R<-c()
for (i in seq(50000,100000,5000)) {
  mean_claims_R<-c(mean_claims_R,mean(pmax(AggclaimAmount-i,0)))
  mean_claims_I<-c(mean_claims_I,mean(pmin(AggclaimAmount,i)))
}
```

```
# Mean Costs to the Insurers
mean_claims_I
```

```
[1] 19450.96 20582.04 21600.00 22511.81 23319.69 24043.04 24680.81 25247.09 257
58.25
[10] 26210.38 26608.72
```

```
# Mean Costs to the Reinsurer
mean_claims_R
```

```
[1] 10133.006  9001.921  7983.967  7072.158  6264.271  5540.923  4903.159  4336
.875
 [9]  3825.712  3373.590  2975.244
```

[6]

v)

```
mean_agg_cost<-mean(AggclaimAmount)

# 75% of the Aggregate claims cost

mean_Cost_Insurer<-mean_agg_cost*0.75
mean_Cost_Insurer

# 75% of the Aggregate claims cost = 22187.97
# Retention limit accordingly is 60000

Retention_Limit<-60000
Reinsurer_Claims<-pmax(AggclaimAmount-Retention_Limit,0)
#Proportion of Claims to be taken up by the reinsurer
sum(Reinsurer_Claims>0)/10000
```

```
[1] 0.1929
```

[5]

vi)

```
SD_Retention<-sd(Reinsurer_Claims)
SD_Retention
```

```
> SD_Retention
[1] 23967.9
```

[2]

Skew_Retention<-mean((Reinsurer_Claims-mean(Reinsurer_Claims))^3)/(sd(Reinsurer_Claims))^3
Skew_Retention

```
> Skew_Retention
[1] 4.305918
```

[2]
[4]
[30 Marks]

Solution 4:

i)

covid19 <- read.csv("/Covid_2019.csv")

missingvalues<-sapply(covid19,FUN = function(x)sum(is.na(x)))
missingvalues [2]

```
> missingvalues
                 Continent                    Country                total_case
s
                         0                          0
0
              total_deaths    total_cases_per_million   total_deaths_per_millio
n
                         0                          0
0
                population         population_density                 median_ag
e
                         0                         11                         2
4
              aged_65_older             gdp_per_capita      cardiovasc_death_rat
e
                        27                         27                         2
4
       diabetes_prevalence              female_smokers               male_smoker
s
                        17                         69                         7
1
hospital_beds_per_thousand            life_expectancy                     Sever
e
                        45                          3
0
```

covid19_1<-na.omit(covid19) [2]
[4]

ii)

Covid_Cluster<-covid19_1[,c("population_density","median_age","aged_65_older","gdp_per_capita",
"cardiovasc_death_rate","diabetes_prevalence","female_smokers",
"male_smokers","hospital_beds_per_thousand","life_expectancy")]
Covid_Cluster<-scale(Covid_Cluster)

[3]

iii)

set.seed(100)
cluster1<-kmeans(Covid_Cluster,centers = 5)
cluster1$size

```
> cluster1$size
[1] 21 28 26 21 30
```

[4]

iv)

covid19_1$cluster<-cluster1$cluster
table(covid19_1$cluster,covid19_1$Severe)
prop.table(table(covid19_1$cluster,covid19_1$Severe),margin = 1)

```
> table(covid19_1$cluster,covid19_1$Severe)

    No Yes
  1 16   5
  2 27   1
  3 10  16
  4 15   6
  5 15  15
> prop.table(table(covid19_1$cluster,covid19_1$Severe),margin = 1)

            No        Yes
  1 0.76190476 0.23809524
  2 0.96428571 0.03571429
  3 0.38461538 0.61538462
  4 0.71428571 0.28571429
  5 0.50000000 0.50000000
```

[5]

v)

aggregate(total_cases~cluster,data = covid19_1, FUN = "sum")

aggregate(total_deaths~cluster,data = covid19_1, FUN = "sum")

```
> aggregate(total_cases~cluster,data = covid19_1, FUN = "sum")
  cluster total_cases
1       1     1467904
2       2      658083
3       3     6589530
4       4     1205188
5       5     6799810
> aggregate(total_deaths~cluster,data = covid19_1, FUN = "sum")
  cluster total_deaths
1       1        34662
2       2        10849
3       3       317888
4       4        30973
5       5       228663
```

[5]
[21 Marks]

****