**ΑΣΚΗΣΗ 1 | ΕΡΓΑΣΙΑ 2**

**> library(actuar)**

**> probability<-1-1e-15**

**> lim <- floor(qgeom(probability,prob=1/3)) ;lim**

[1] 85

**> fN<-dgeom(0:85,prob=1/3)**

**> sum(fN)**

[1] 1

**> lim2<-floor(qgeom(probability,prob=1/4)) ;lim2**

[1] 120

**> fX<-dgeom(0:120,prob=1/4)**

**> sum(fX)**

[1] 1

**> Gs.conv<-aggregateDist("convolution",model.freq=fN,model.sev=fX)**

**> Gs.conv**

Aggregate Claim Amount Distribution

Exact calculation (convolutions)

Call:

aggregateDist(method = "convolution", model.freq = fN, model.sev = fX)

Data: ( 10201 obs. )

x[1:10201] = 0, 1, 2, ..., 1.02e+04, 1.02e+04

**> gs.conv<-diff(Gs.conv)**

**> Gs.rec<-aggregateDist("recursive",model.freq="geometric",model.sev=fX,prob=1/3)**

**> Gs.rec**

Aggregate Claim Amount Distribution

Recursive method approximation

Call:

aggregateDist(method = "recursive", model.freq = "geometric",

model.sev = fX, prob = 1/3)

Data: ( 128 obs. )

x[1:128] = 0, 1, 2, ..., 126, 127

**> gs.rec<-diff(Gs.rec)**

**> set.seed(19010)**

**> mfreq<-expression(data=rgeom(1/3))**

**> msev<-expression(data=rgeom(prob=1/4))**

**> Gs.simul<-aggregateDist("simulation",nb.simul=1000000,mfreq,msev)**

**> Gs.simul**

Aggregate Claim Amount Distribution

Approximation by simulation

Call:

aggregateDist(method = "simulation", model.freq = mfreq, model.sev = msev,

nb.simul = 1e+06)

Data: ( 112 obs. )

x[1:112] = 0, 1, 2, ..., 120, 122

**> gs.simul<-diff(Gs.simul)**

**> c<-20**

**> x<-seq(0,c)**

**> mat<-cbind(x,Gs.conv(x),Gs.rec(x),Gs.simul(x))**

**> colnames(mat)<-c("x","G(x)\_Convolution","G(x)\_Recursive","G(x)\_Simulation")**

**> rownames(mat)<-rep("",nrow(mat))**

**> mat**

x G(x)\_Convolution G(x)\_Recursive G(x)\_Simulation

0 0.4000000 0.4000000 0.400599

1 0.4600000 0.4600000 0.460583

2 0.5140000 0.5140000 0.514687

3 0.5626000 0.5626000 0.563184

4 0.6063400 0.6063400 0.606594

5 0.6457060 0.6457060 0.645530

6 0.6811354 0.6811354 0.681035

7 0.7130219 0.7130219 0.712960

8 0.7417197 0.7417197 0.741835

9 0.7675477 0.7675477 0.767617

10 0.7907929 0.7907929 0.790952

11 0.8117136 0.8117136 0.811951

12 0.8305423 0.8305423 0.831116

13 0.8474881 0.8474881 0.848041

14 0.8627392 0.8627392 0.863138

15 0.8764653 0.8764653 0.876763

16 0.8888188 0.8888188 0.889143

17 0.8999369 0.8999369 0.900398

18 0.9099432 0.9099432 0.910433

19 0.9189489 0.9189489 0.919373

20 0.9270540 0.9270540 0.927539

**> d<-c+1**

**> matg<-cbind(x,gs.conv[1:d],gs.rec[1:d],gs.simul[1:d])**

**> rownames(matg)<-rep("",nrow(mat))**

**> colnames(matg)<-c("x","g(x)\_Convolution","g(x)\_Recursive","g(x)\_Simulation")**

**> matg**

x g(x)\_Convolution g(x)\_Recursive g(x)\_Simulation

0 0.400000000 0.400000000 0.400599

1 0.060000000 0.060000000 0.059984

2 0.054000000 0.054000000 0.054104

3 0.048600000 0.048600000 0.048497

4 0.043740000 0.043740000 0.043410

5 0.039366000 0.039366000 0.038936

6 0.035429400 0.035429400 0.035505

7 0.031886460 0.031886460 0.031925

8 0.028697814 0.028697814 0.028875

9 0.025828033 0.025828033 0.025782

10 0.023245229 0.023245229 0.023335

11 0.020920706 0.020920706 0.020999

12 0.018828636 0.018828636 0.019165

13 0.016945772 0.016945772 0.016925

14 0.015251195 0.015251195 0.015097

15 0.013726075 0.013726075 0.013625

16 0.012353468 0.012353468 0.012380

17 0.011118121 0.011118121 0.011255

18 0.010006309 0.010006309 0.010035

19 0.009005678 0.009005678 0.008940

20 0.008105110 0.008105110 0.008166

**ΑΣΚΗΣΗ 2**

**Ερώτηση (i)**

**> y<-read.table("data.txt",header=TRUE)**

**> names(y)**

[1] "x"

**> attach(y)**

**> library(fitdistrplus)**

**> weiMLE<-fitdist(x,"weibull")**

**> summary(weiMLE) #AIC**

Fitting of the distribution ' weibull ' by maximum likelihood

Parameters :

estimate Std. Error

shape 0.8701248 0.02089434

scale 34.0042902 1.30387923

Loglikelihood: -4583.441 AIC: 9170.883 BIC: 9180.698

Correlation matrix:

shape scale

shape 1.0000000 0.3208156

scale 0.3208156 1.0000000

**> lnoMLE<-fitdist(x,"lnorm")**

**> summary(lnoMLE)**

Fitting of the distribution ' lnorm ' by maximum likelihood

Parameters :

estimate Std. Error

meanlog 2.888899 0.04342519

sdlog 1.373225 0.03070617

Loglikelihood: -4624.999 AIC: 9253.998 BIC: 9263.814

Correlation matrix:

meanlog sdlog

meanlog 1 0

sdlog 0 1

**> logMLE<-fitdist(x,"logis")**

**> summary(logMLE)**

Fitting of the distribution ' logis ' by maximum likelihood

Parameters :

estimate Std. Error

location 28.48412 1.073503

scale 20.23604 0.557160

Loglikelihood: -5072.082 AIC: 10148.16 BIC: 10157.98

Correlation matrix:

location scale

location 1.0000000 0.1326477

scale 0.1326477 1.0000000

**> library(actuar)**

**> lloMLE<-fitdist(x,"llogis")**

**> summary(lloMLE)**

Fitting of the distribution ' llogis ' by maximum likelihood

Parameters :

estimate Std. Error

shape 1.317991 0.03483195

scale 19.762379 0.82274792

Loglikelihood: -4612.499 AIC: 9228.997 BIC: 9238.813

Correlation matrix:

shape scale

shape 1.00000000 0.04606971

scale 0.04606971 1.00000000

**> invMLE<-fitdist(x,"invgamma")**

**> summary(invMLE)**

Fitting of the distribution ' invgamma ' by maximum likelihood

Parameters :

estimate Std. Error

shape 0.4777136 0.01756201

scale 2.2507071 0.13209932

Loglikelihood: -4976.719 AIC: 9957.439 BIC: 9967.254

Correlation matrix:

shape scale

shape 1.0000000 0.6263612

scale 0.6263612 1.0000000

**> gamMLE<-fitdist(x,"gamma")**

**> summary(gamMLE)**

Fitting of the distribution ' gamma ' by maximum likelihood

Parameters :

estimate Std. Error

shape 0.82880252 0.032032743

rate 0.02261306 0.001171931

Loglikelihood: -4589.229 AIC: 9182.458 BIC: 9192.274

Correlation matrix:

shape rate

shape 1.0000000 0.7433266

rate 0.7433266 1.0000000

**> paretMLE<-fitdist(x,"pareto")**

**> summary(paretMLE) #AIC**

Fitting of the distribution ' pareto ' by maximum likelihood

Parameters :

estimate Std. Error

shape 4.539648 0.819902

scale 130.405437 28.306895

Loglikelihood: -4577.473 AIC: 9158.945 BIC: 9168.761

Correlation matrix:

shape scale

shape 1.0000000 0.9845525

scale 0.9845525 1.0000000

**Επιλέγοντας το κριτήριο AIC οι δύο κατανομές που προσαρμόζονται καλύτερα είναι η Weibull και η Pareto , έχουν μικρότερο AIC από τις υπόλοιπες.**

Αναλυτικά οι τύποι των συναρτήσεων είναι :

* Για Weibull (0.87, 34) έχουμε : f(x) =
* Για Pareto (4.54,130.40) έχουμε : f(x) = 4,54

**Ερώτηση (ii)**

**Για την Weibull έχουμε :**

**> hist(x,prob=TRUE,xlim=c(0,500),ylim=c(0,0.02))**

**> u<-seq(0,500)**

**> lines(u,dweibull(u,shape=weiMLE$estimate[1],scale=weiMLE$estimate[2]),col="green",lwd=0.8)**

****

**Για την Pareto έχουμε :**

**> hist(x,prob=TRUE,xlim=c(0,500),ylim=c(0,0.02))**

**> lines(u,dpareto(u,shape=paretMLE$estimate[1],scale=paretMLE$estimate[2]),col="red",lwd=0.8)**

****

**Ερώτηση (iii)**

**> qqcomp(list(weiMLE,paretMLE),fitcol="black",main="QQ-plot on X",legendtext=c("Weibull","Pareto"),fitpch=1:4)**



**Τα δεδομένα δεν προσαρμόζονται απόλυτα στις 2 κατανομές (pareto και Weibull) αφού μετά το 170 αρχίζουν και παρεκκλίνουν.**

**Επίσης γίνεται αντιληπτό ότι τα δεδομένα προσαρμόζονται λίγο καλύτερα στην pareto αφού έχουν μικρότερη απόκλιση από την ευθεία y=x σε σχέση με τη Weibull.**

**ΑΣΚΗΣΗ 3**

**> library(actuar)**

**> h<-0.001**

**> fx<-discretize(punif(x,min=0,max=50),method="unbiased",lev=levunif(x,min=0,max=50),from=0,to=50,step=h)**

**> length(fx)**

[1] 50001

**> probability<- 1-1e-15**

**> n<-3 ; p<-0.6**

**> lim<- floor(qbinom(probability,size=n,prob=p))**

**> lim**

[1] 3

**> fN<-dbinom(0:3,size=n,prob=p)**

**> Gs.conv<-aggregateDist("convolution",model.freq=fN,model.sev=fx,x.scale=h)**

**> Gs.conv**

Aggregate Claim Amount Distribution

Exact calculation (convolutions)

Call:

aggregateDist(method = "convolution", model.freq = fN, model.sev = fx,

x.scale = h)

Data: ( 150001 obs. )

x[1:150001] = 0, 0.001, 0.002, ..., 150, 150

**> Gs.rec<-aggregateDist("recursive",model.freq="binom",model.sev=fx,size=3,prob=0.6,x.scale=0.001,maxit=1000000)**

**> Gs.rec**

Aggregate Claim Amount Distribution

Recursive method approximation

Call:

aggregateDist(method = "recursive", model.freq = "binom", model.sev = fx,

x.scale = 0.001, size = 3, prob = 0.6, maxit = 1e+06)

Data: ( 148487 obs. )

x[1:148487] = 0, 0.001, 0.002, ..., 148.5, 148.5

**> set.seed(19010)**

**> mfreq<-expression(data=rbinom(n,p))**

**> msev<- expression(data=runif(min=0,max=50))**

**> Gs.simul<- aggregateDist("simulation",nb.simul=1000000,mfreq,msev)**

**> Gs.simul**

Aggregate Claim Amount Distribution

Approximation by simulation

Call:

aggregateDist(method = "simulation", model.freq = mfreq, model.sev = msev,

nb.simul = 1e+06)

Data: ( 936534 obs. )

x[1:936534] = 0, 6.382e-05, 0.0003488, ..., 148.1, 148.5

**> x<-seq(0,150,10) :** στο 150 έχουμε G(X) = 1

**> mat2<-cbind(x,Gs.conv(x),Gs.rec(x),Gs.simul(x))**

**> colnames(mat2)<-c("x","G(x)\_Convolution","G(x)\_Recursive","G(x)\_Simulation")**

**> rownames(mat2)<-rep("",nrow(mat2))**

**> mat2**

x G(x)\_Convolution G(x)\_Recursive G(x)\_Simulation

0 0.06400288 0.06400288 0.063399

10 0.13053179 0.13053179 0.129850

20 0.21606878 0.21606878 0.215690

30 0.32234186 0.32234186 0.321791

40 0.45107903 0.45107903 0.450792

50 0.60400540 0.60400540 0.603092

60 0.70710888 0.70710888 0.706313

70 0.79811619 0.79811619 0.797525

80 0.87357133 0.87357133 0.873156

90 0.93001829 0.93001829 0.929664

100 0.96400108 0.96400108 0.963872

110 0.98156869 0.98156869 0.981554

120 0.99222439 0.99222439 0.992230

130 0.99769617 0.99769617 0.997651

140 0.99971204 0.99971204 0.999708

150 1.00000000 1.00000000 1.000000