```
##############################################################################################
## Data set provided in Gisler (slides Aggregate Loss Modeling)
##############################################################################################



library(MASS)
T=10
vt=c(240755,255571,269739,281708,306888,320265,323481,334753,340265,344757)
Nt=c(13153,14186,14207,13461,21261,19934,15796,15157,17483,19185)
x=Nt/vt
data=round(x*100,2)

t=1982:1991
plot(t,x, xlab="time", ylab="observed frequencies")
abline(h=mean(x))

#Poisson
lambda=fitdistr(x,"Poisson")
lambda

vco=(lambda$estimate*mean(vt))^(-1/2)
vco

abline(h=lambda$estimate+lambda$estimate*vco,col="green")
abline(h=lambda$estimate-lambda$estimate*vco,col="green")


#goodness-of-fit
X2=sum(vt*(Nt/vt-lambda$estimate)^2)/lambda$estimate
X2

qchisq(0.99,T-1)

#X2>>qchisq hence reject H0


# library(vcd)
# gf=goodfit(Nt,type="poisson", method="MinChisq")
# summary(gf)
# plot(gf)

#Negative-binomial
#fitdistr(x,"negative binomial")

lambda=1/(sum(vt))*sum(Nt)
lambda
Vt=1/(T-1)*sum(vt*(Nt/vt-lambda)^2)
Vt
#larger than 5.43
gamma=lambda^2/(Vt-lambda)*1/(T-1)*(sum(vt)-sum(vt^2)/sum(vt))
gamma

vco=sqrt(1/(lambda*mean(vt))+1/gamma)
vco

#lambda estimate same as before!
plot(t,x, xlab="time", ylab="observed frequencies")
abline(h=mean(x))
abline(h=lambda+lambda*vco,col="green")
abline(h=lambda-lambda*vco,col="green")




##############################################################################################
## Exercise Natural Hazards in Switzerland (slides Aggregate Loss Modeling)
##############################################################################################

Y=c(52.8,135.2,55.9,138.6,122.9,55.8,368.2,83.8,78.5,75.3,178.3,182.8,54.4,365.3,1051.1)
sum(log(Y))
#a)
alpha_M=1/(mean(log(Y))-log(50))*14/15
alpha_M
alpha_bias=15/14*alpha_M
alpha_bias

#b)
```

```
alfa=alpha_M
#alfa=alpha_bias
15/20*((1-(2000/50)^(-alfa+1))*(50*alfa)/(alfa-1)+2000*(2000/50)^(-alfa))

15/20*1/(alfa-1)*(alfa*50-50^alfa*2000/(2000^alfa))

15/20*50/(alfa-1)*(alfa-(50/2000)^(alfa-1))


#c)
(2000/50)^(-alfa)


(2*10^9/10^6/50)^(-alpha_M)
```