analysis_and_modelling

Edmund Liman

2023-08-11

```r
# Chunk setting up the workspace
set.seed(1)
setwd("C:\\Users\\edmun\\Documents\\GitHub Repositories\\GIM-ACTL30007")
library("readxl")
library("writexl")
```

```
## Warning: package 'writexl' was built under R version 4.3.1
```

```r
library("actuar")
```

```
##
## Attaching package: 'actuar'
```

```
## The following objects are masked from 'package:stats':
##
##     sd, var
```

```
## The following object is masked from 'package:grDevices':
##
##     cm
```

```r
library("fitdistrplus")
```

```
## Loading required package: MASS
```

```
## Loading required package: survival
```

```r
library("extRemes")
```

```
## Loading required package: Lmoments
```

```
## Loading required package: distillery
```

```
##
## Attaching package: 'extRemes'
```

```
## The following objects are masked from 'package:stats':
##
##     qqnorm, qqplot
```

```
library("evir")
```

```
##
## Attaching package: 'evir'
```

```
## The following object is masked from 'package:extRemes':
##
##     decluster
```

```
raw_data <- read_excel("src\\past_claims_data.xlsx")
```

```
sum(is.na(raw_data)) # Check how many policy id have zero claim
```

```
## [1] 593
```

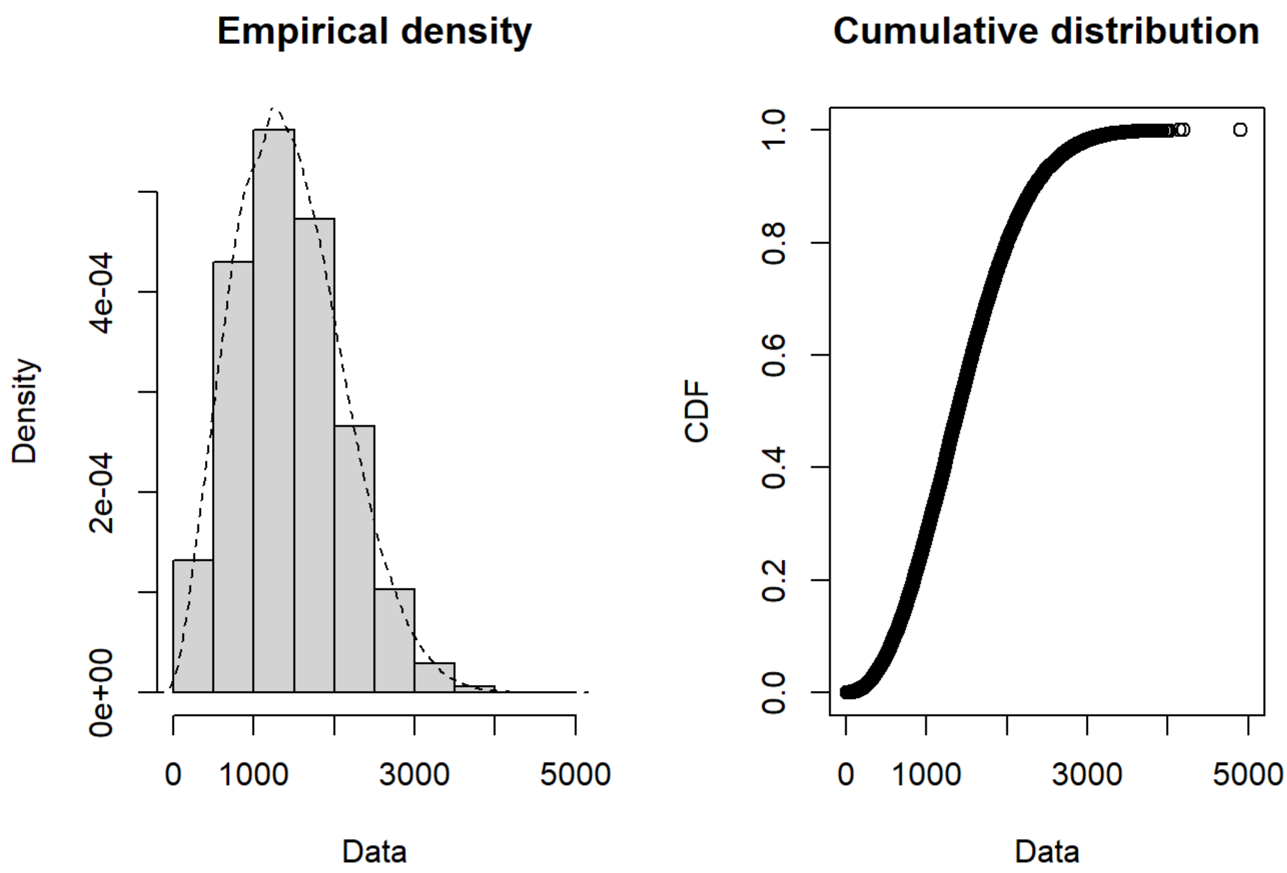
```
sum(raw_data$claim[claim = 0]) # Check claims of size 0
```

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

```
cleaned_data <- raw_data
cleaned_data$claim[is.na(cleaned_data$claim)] <- 0
write_xlsx(data.frame(raw_data, cleaned_data),"C:\\Users\\edmun\\Documents\\GitHub Repositori
es\\GIM-ACTL30007\\out\\comparison_data.xlsx")
attach(cleaned_data)
```

There are 593 policies with no claim. There are no claims of size 0.

```
plotdist(claim[claim>0], hist = TRUE, demp = TRUE)
```




```
min(claim[claim > 0])
```

```
## [1] 11.91567
```

```
max(claim[claim > 0])
```

```
## [1] 4907.379
```

```
format(actuar::emm(claim[claim>0], order = 1:3), scientific = FALSE)
```

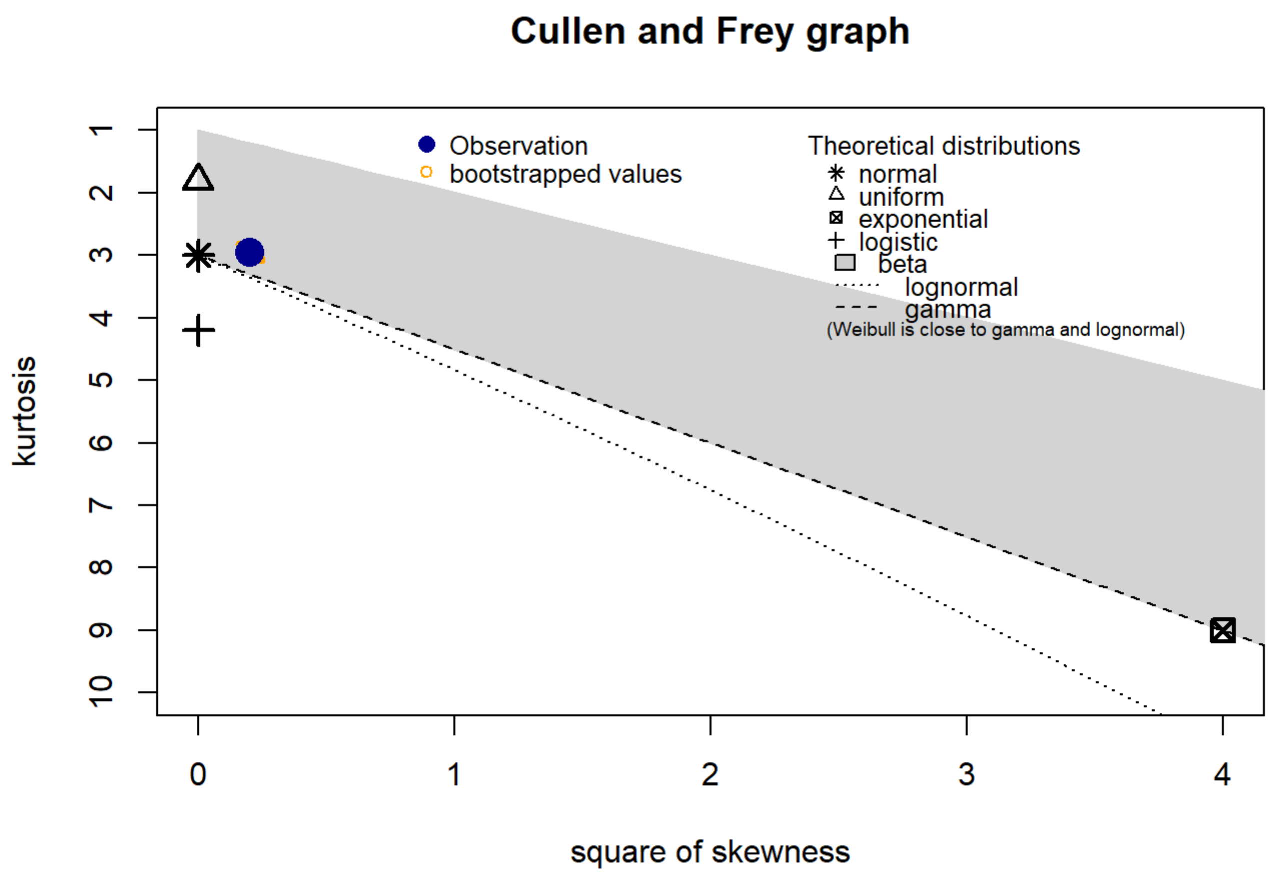
```
## [1] "      1443.124" "   2529736.699" "5075195351.327"
```

```
sd(claim[claim>0])/mean(claim[claim>0])
```

```
## [1] 0.4633643
```

Data looks nice. Do not need to perform any transformations.

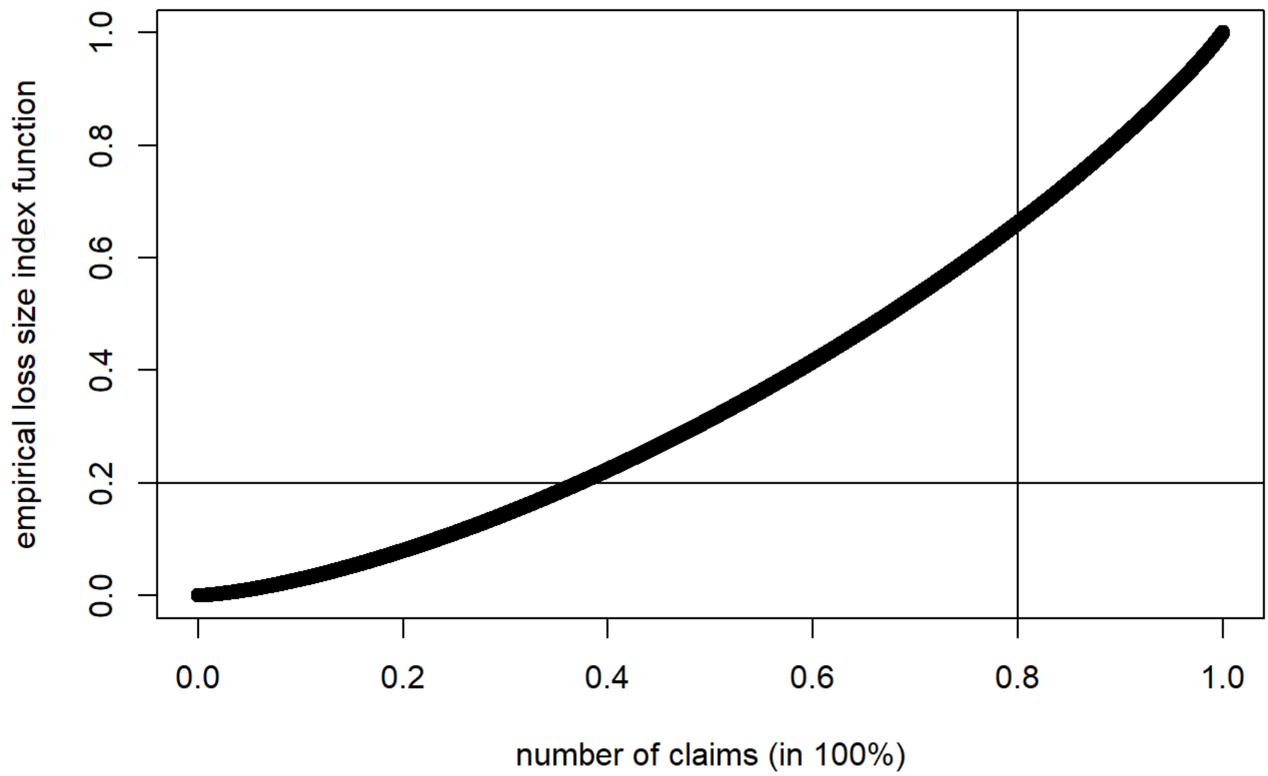
```
descdist(claim[claim>0], boot = 1000)
```




```
## summary statistics
## ------
## min:  11.91567   max:  4907.379
## median:  1389.93
## mean:  1443.124
## estimated sd:  668.692
## estimated skewness:  0.448016
## estimated kurtosis:  2.942086
```

We will check Gamma, Lognormal and Weibull

```
claim.nz.lif <- cumsum(sort(claim[claim>0]))/sum(claim[claim>0])
plot(1:length(claim[claim>0])/length(claim[claim>0]), claim.nz.lif,
  xlab = "number of claims (in 100%)", ylab = "empirical loss size index function")
abline(h = 0.2, v = 0.8)
```



Not heavy-tailed distribution. 80% of claims explains about 70% of loss.

```
mef <- function(x, u) {
  mefvector <- c()
  for (i in u) {
    mefvector <- c(mefvector, sum(pmax(sort(x) - i, 0))/length(x[x >
      i]))
  }
  return(mefvector)
}

mef(claim, c(0, 100, 1000, 2000, 4000, 4200, 4300))
```

```
## [1] 1443.1237 1345.8292  743.0078  435.5032  281.7552  707.3787  607.3787
```

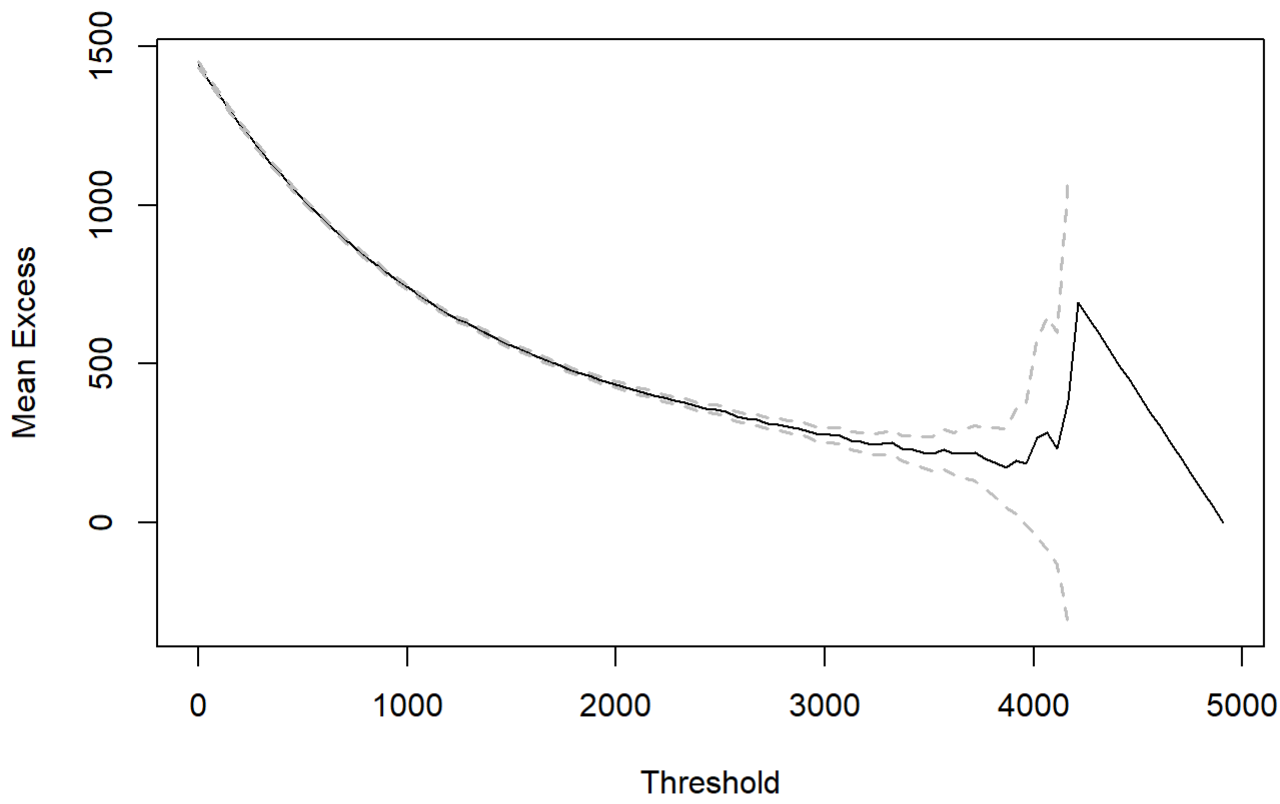
```
length(claim[claim>4000])
```

```
## [1] 5
```

```
length(claim[claim>4200])
```

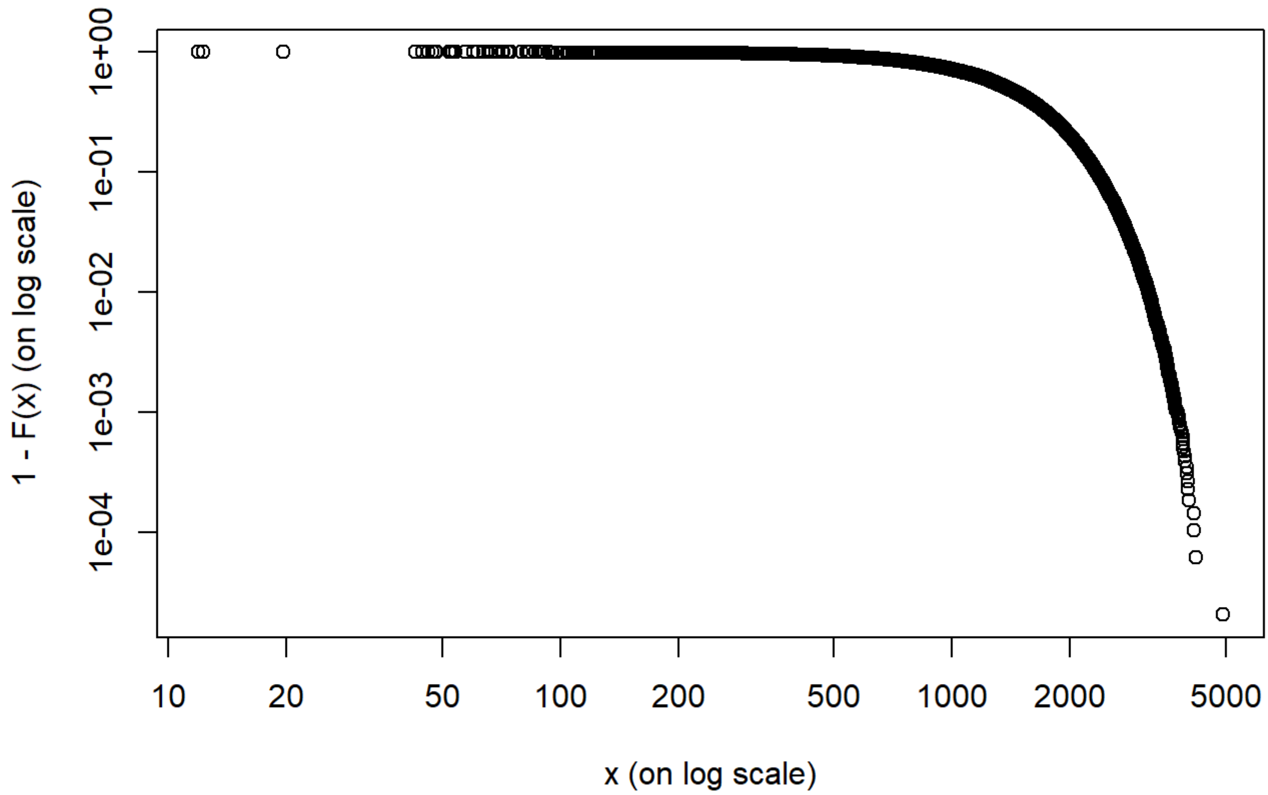
```
## [1] 1
```

```
mrlplot(claim)
```



The mean excess function tells us the average added value loss Y would take above a threshold. Decreasing mean excess function indicates a light-tailed distribution. One exception in my data is a high-values claim at the tail which leads to an increase in mean excess function. However, there are only 5 claims above 4000 and 1 claim above 4200 out of 25041 claims so it can be an outlier or a very rare event. Indicates that log-normal would not be a good fit as it is a heavy-tailed distribution. Weibull Shape parameter (tao in lectures) would not be between 0 and 1 or else it will be heavy-tailed. Gamma is light-tailed

```
emplot(claim[claim  > 0], alog = "xy", labels = TRUE)
```



The log-log is concave which supports that distribution is not heavy-tailed. 1-F(x) rapidly diminishes to 0 at the tail.

```
fit.gamma.mme <- fitdist(claim[claim > 0], "gamma", method = "mme", order = 1:2)
fit.gamma.mme$estimate
```

```
##       shape        rate
## 4.657711953 0.003227521
```

```
fit.gamma.mme$loglik
```

```
## [1] -193741.2
```

```
fit.lnorm.mme <- fitdist(claim[claim > 0], "lnorm", method = "mme", order = 1:2)
fit.lnorm.mme$estimate
```

```
##   meanlog     sdlog
## 7.1773177 0.4410161
```

```
fit.lnorm.mme$loglik
```

```
## [1] -197121.3
```

```
memp <- function(x, order) mean(x^order)
fit.weibull.mme <- fitdist(claim[claim > 0], "weibull", method = "mme", memp = memp, order =
1:2)
fit.weibull.mme$estimate
```

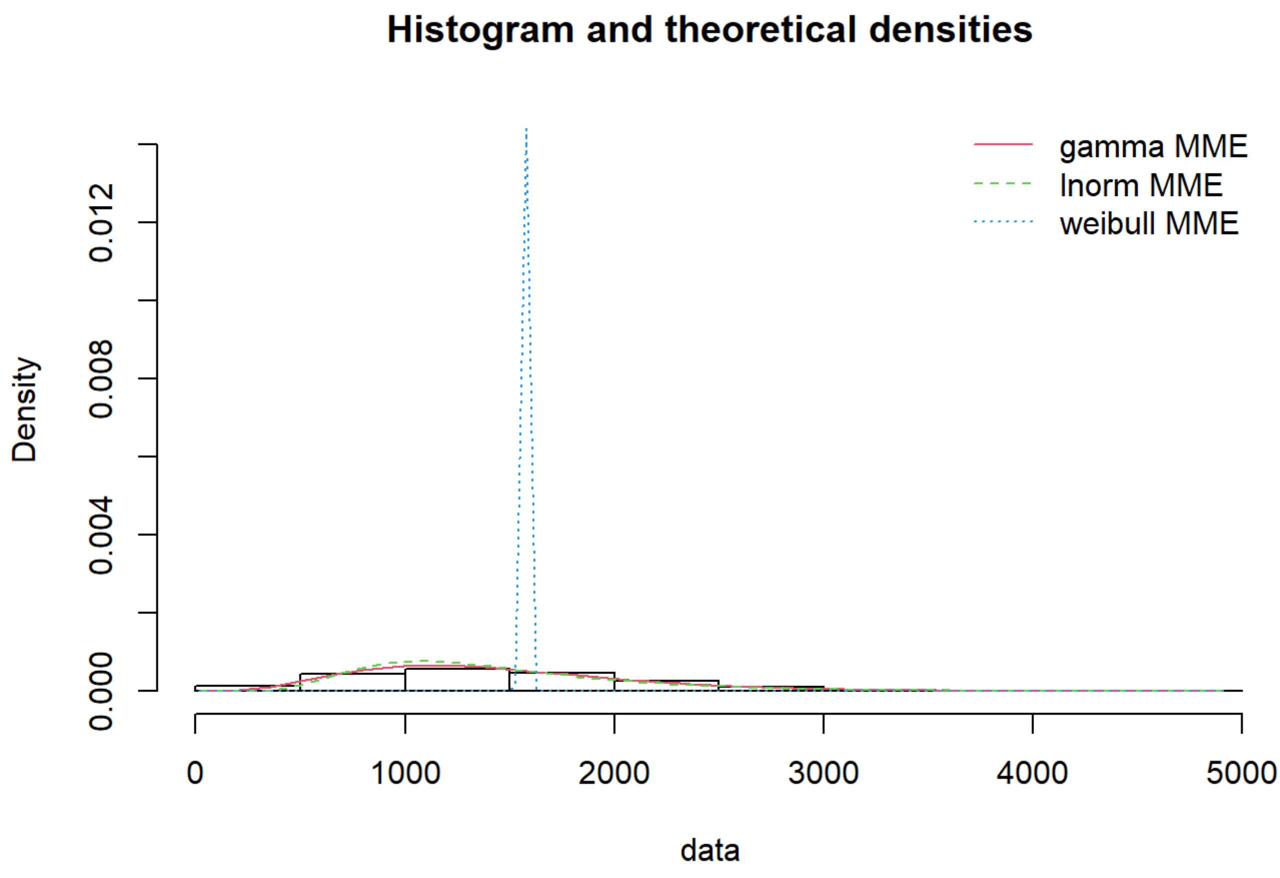
```
##     shape     scale
##  157.8534 1596.2348
```

```
fit.weibull.mme$loglik
```

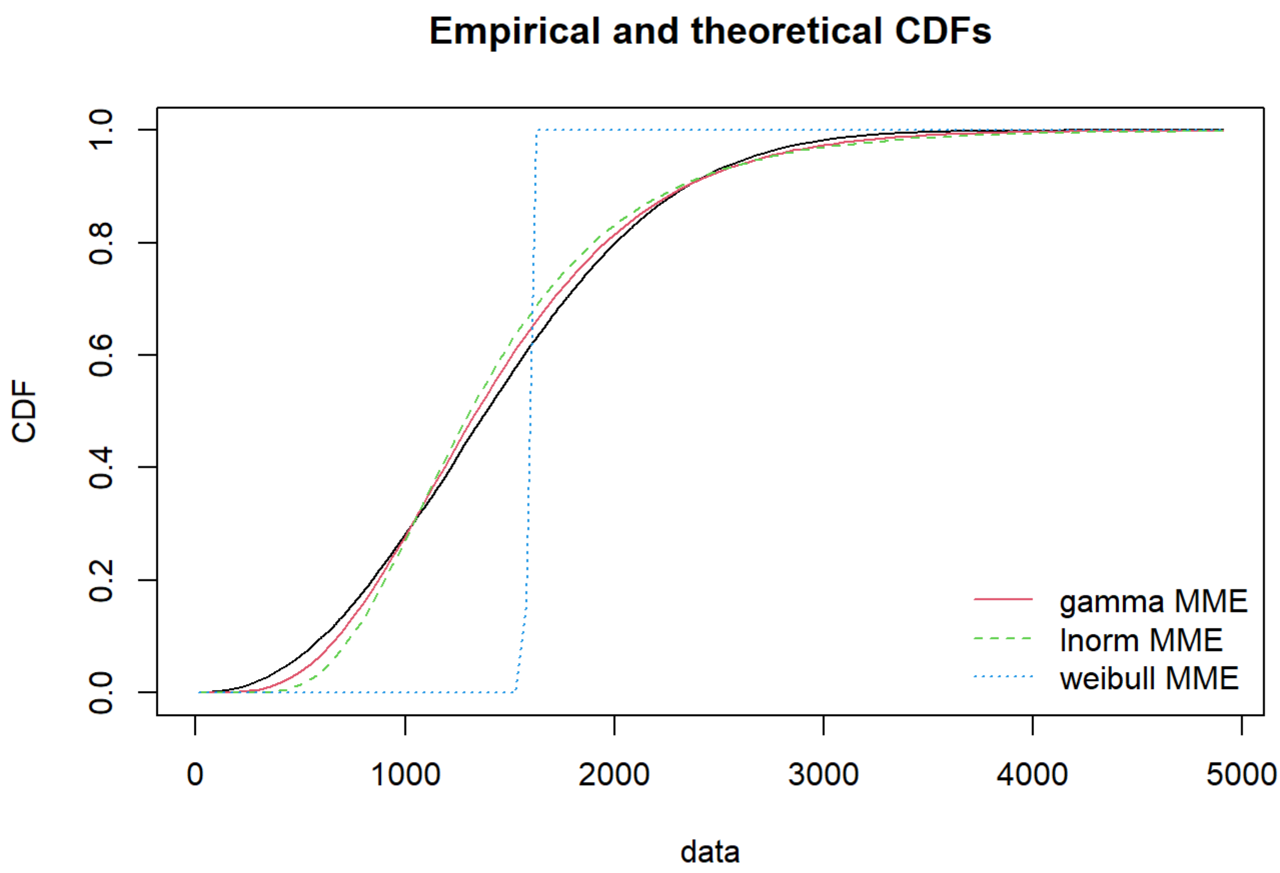
```
## [1] -Inf
```

Fitting the data using method of moments.

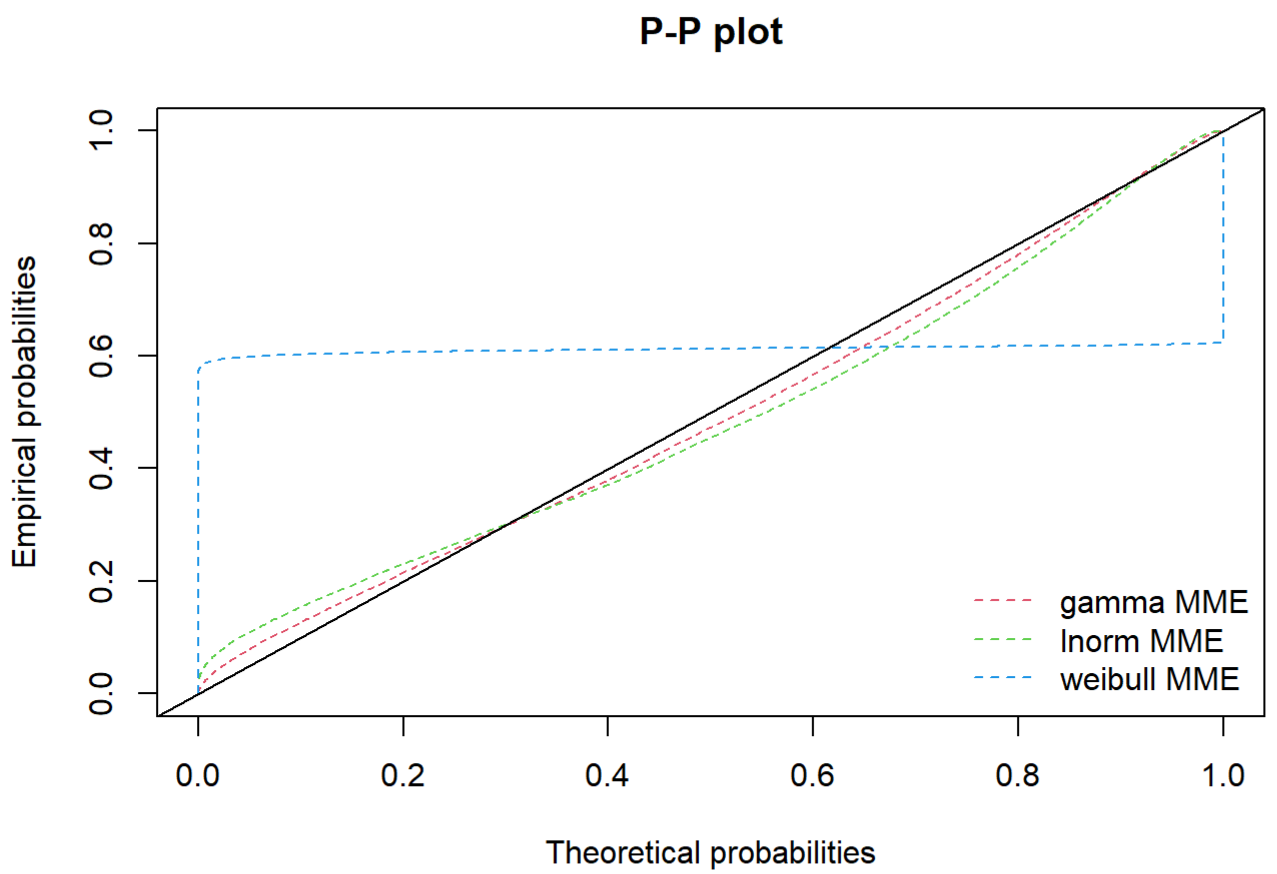
```
plot.legend <- c("gamma MME", "lnorm MME", "weibull MME")
fitdistrplus::denscomp(list(fit.gamma.mme, fit.lnorm.mme, fit.weibull.mme),
  legendtext = plot.legend, fitlwd = 1)
```



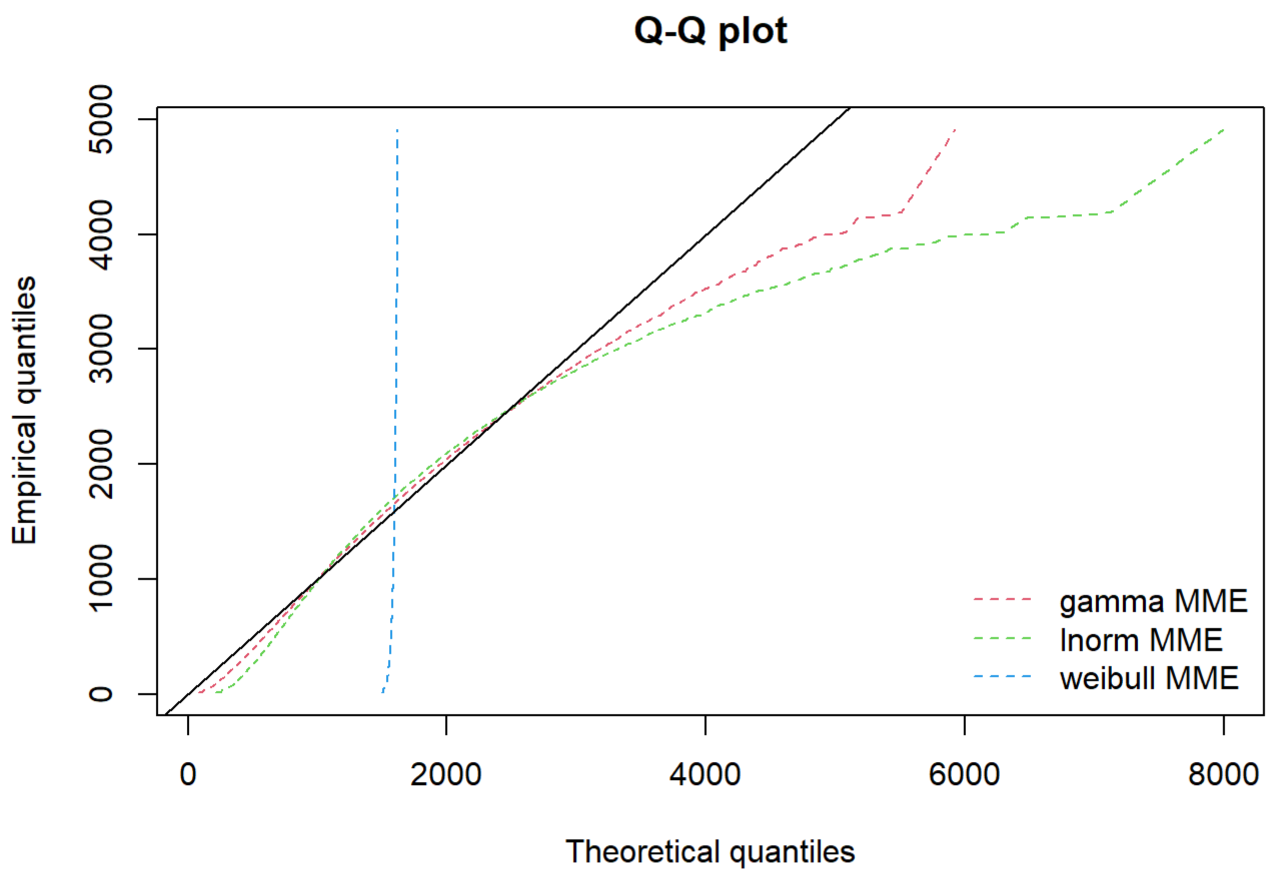

```
fitdistrplus::cdfcomp(list(fit.gamma.mme, fit.lnorm.mme, fit.weibull.mme),
  legendtext = plot.legend, fitlwd = 1, datapch = 10)
```




```
fitdistrplus::ppcomp(list(fit.gamma.mme, fit.lnorm.mme, fit.weibull.mme),
  legendtext = plot.legend, fitpch = 20)
```



```
fitdistrplus::qqcomp(list(fit.gamma.mme, fit.lnorm.mme, fit.weibull.mme),
  legendtext = plot.legend, fitpch = 20)
```




```
fit.gamma.mle <- fitdist(claim[claim > 0], "gamma", method = "mle")
fit.gamma.mle$estimate
```

```
##       shape        rate
## 3.948323712 0.002736376
```

```
fit.gamma.mle$loglik
```

```
## [1] -193551.3
```

```
fit.lnorm.mle <- fitdist(claim[claim > 0], "lnorm", method = "mle")
fit.lnorm.mle$estimate
```

```
##   meanlog     sdlog
## 7.1426474 0.5636995
```

```
fit.lnorm.mle$loglik
```

```
## [1] -195299.2
```

```
fit.weibull.mle <- fitdist(claim[claim > 0], "weibull", method = "mle")
fit.weibull.mle$estimate
```

```
##       shape       scale
##    2.286708 1628.721687
```

```
fit.weibull.mle$loglik
```

```
## [1] -192994.1
```

```
summary(fit.gamma.mle)
```

```
## Fitting of the distribution ' gamma ' by maximum likelihood
## Parameters :
##          estimate   Std. Error
## shape 3.948323712 2.093273e-02
## rate  0.002736376 1.293633e-05
## Loglikelihood:  -193551.3   AIC:  387106.6   BIC:  387122.8
## Correlation matrix:
##           shape      rate
## shape 1.0000000 0.8221299
## rate  0.8221299 1.0000000
```

```
summary(fit.lnorm.mle)
```

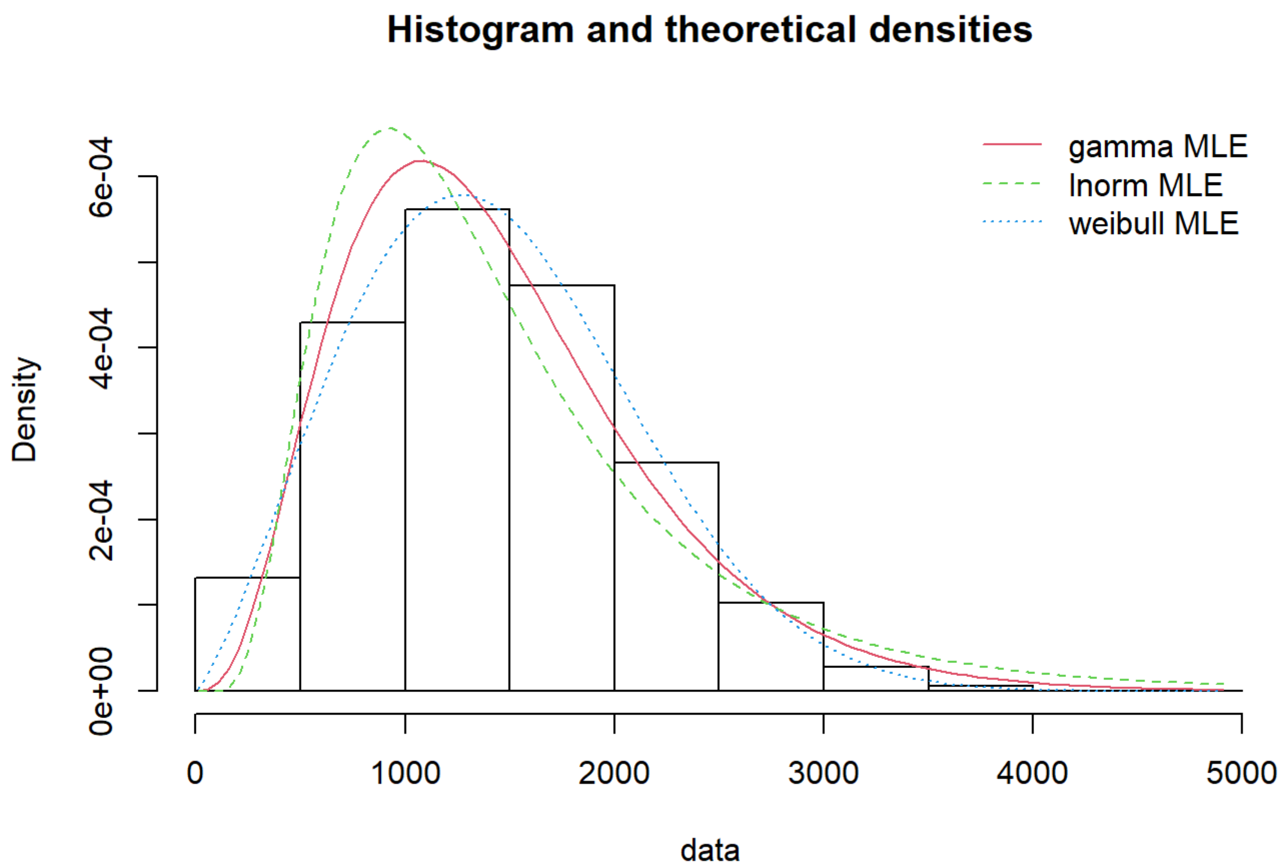
```
## Fitting of the distribution ' lnorm ' by maximum likelihood
## Parameters :
##          estimate  Std. Error
## meanlog 7.1426474 0.003605172
## sdlog   0.5636995 0.002549205
## Loglikelihood:  -195299.2   AIC:  390602.5   BIC:  390618.7
## Correlation matrix:
##         meanlog sdlog
## meanlog       1     0
## sdlog         0     1
```

```
summary(fit.weibull.mle)
```

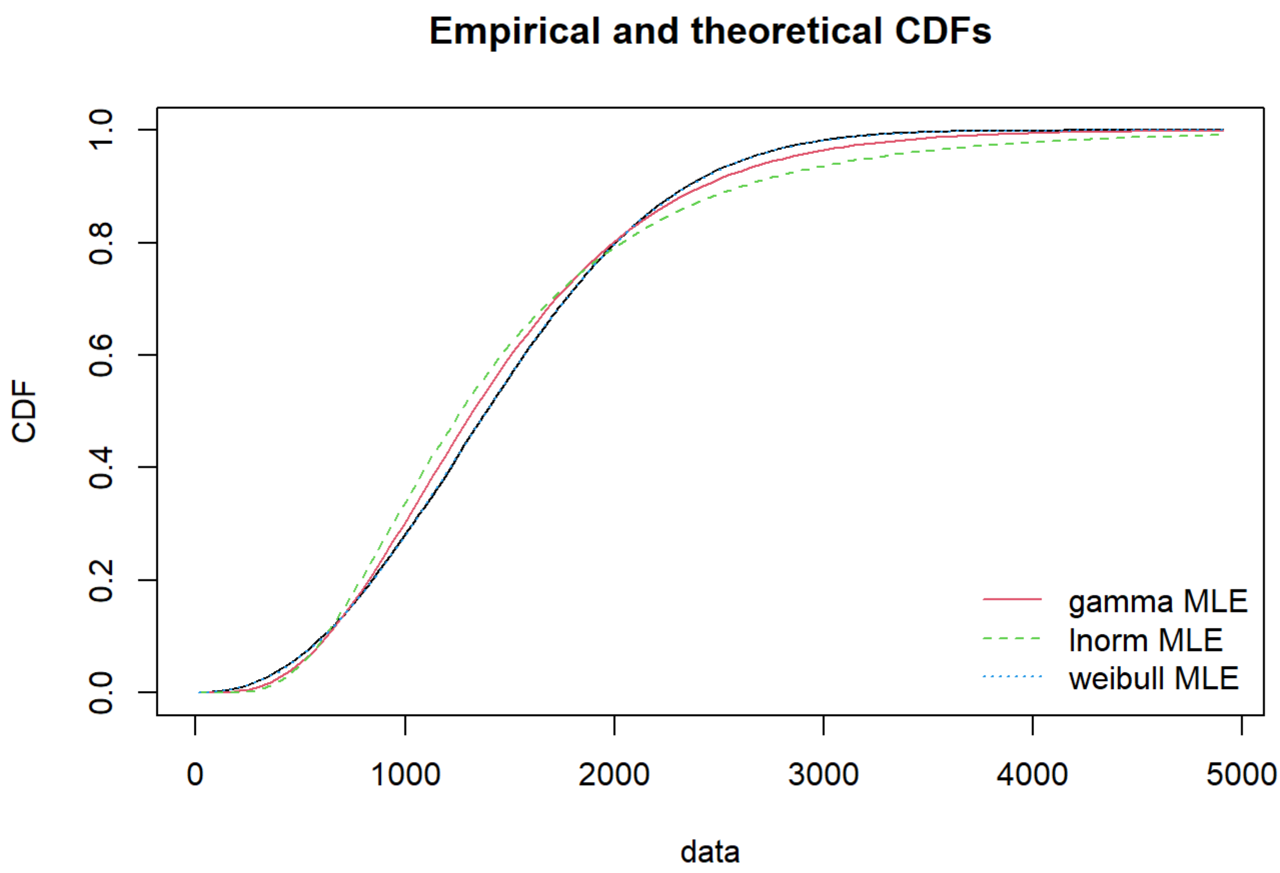
```
## Fitting of the distribution ' weibull ' by maximum likelihood
## Parameters :
##          estimate Std. Error
## shape    2.286708 0.01142025
## scale 1628.721687 4.79510241
## Loglikelihood:  -192994.1   AIC:  385992.3   BIC:  386008.5
## Correlation matrix:
##           shape     scale
## shape 1.0000000 0.3128634
## scale 0.3128634 1.0000000
```

Fitting the data using maximum likelihood.

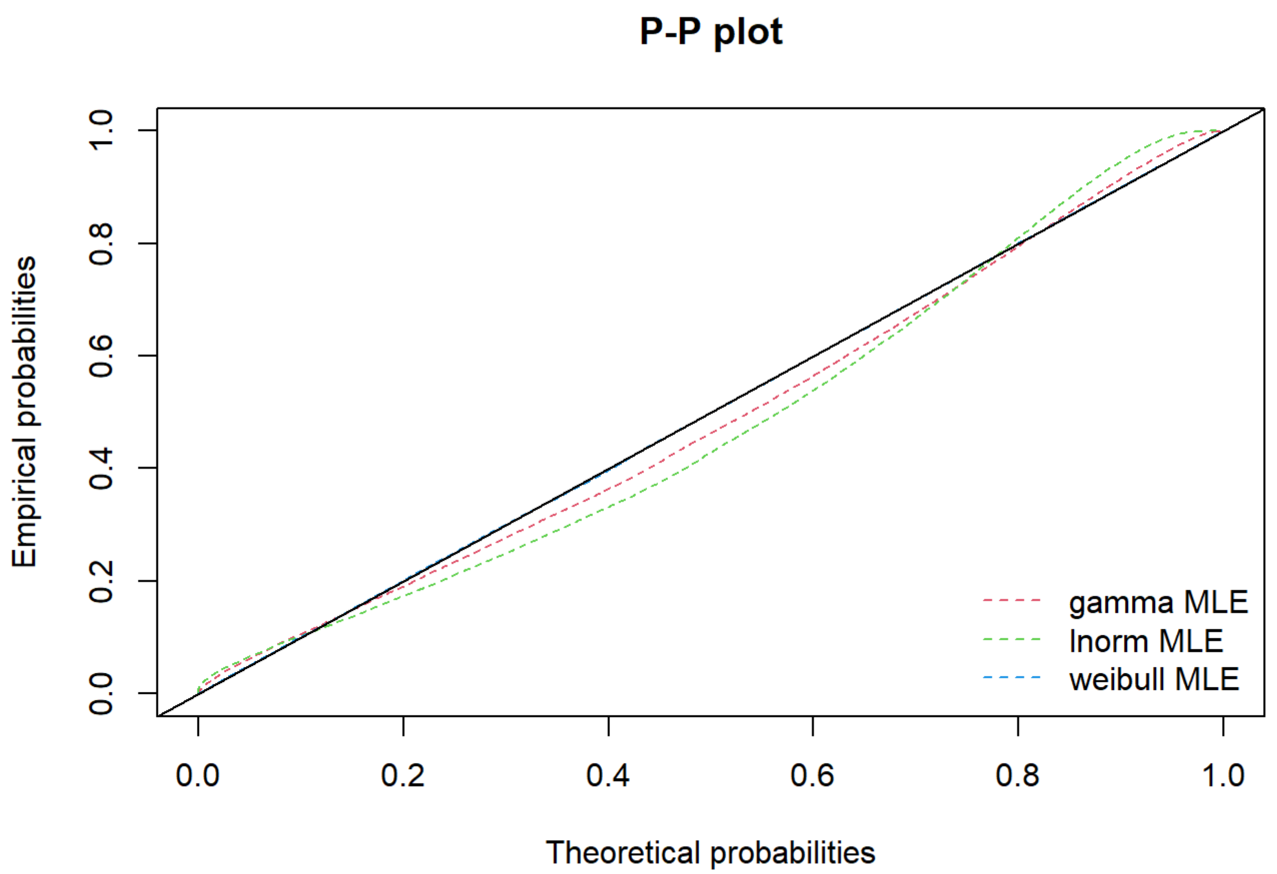
```
plot.legend <- c("gamma MLE", "lnorm MLE", "weibull MLE")
fitdistrplus::denscomp(list(fit.gamma.mle, fit.lnorm.mle, fit.weibull.mle),
  legendtext = plot.legend, fitlwd = 1)
```



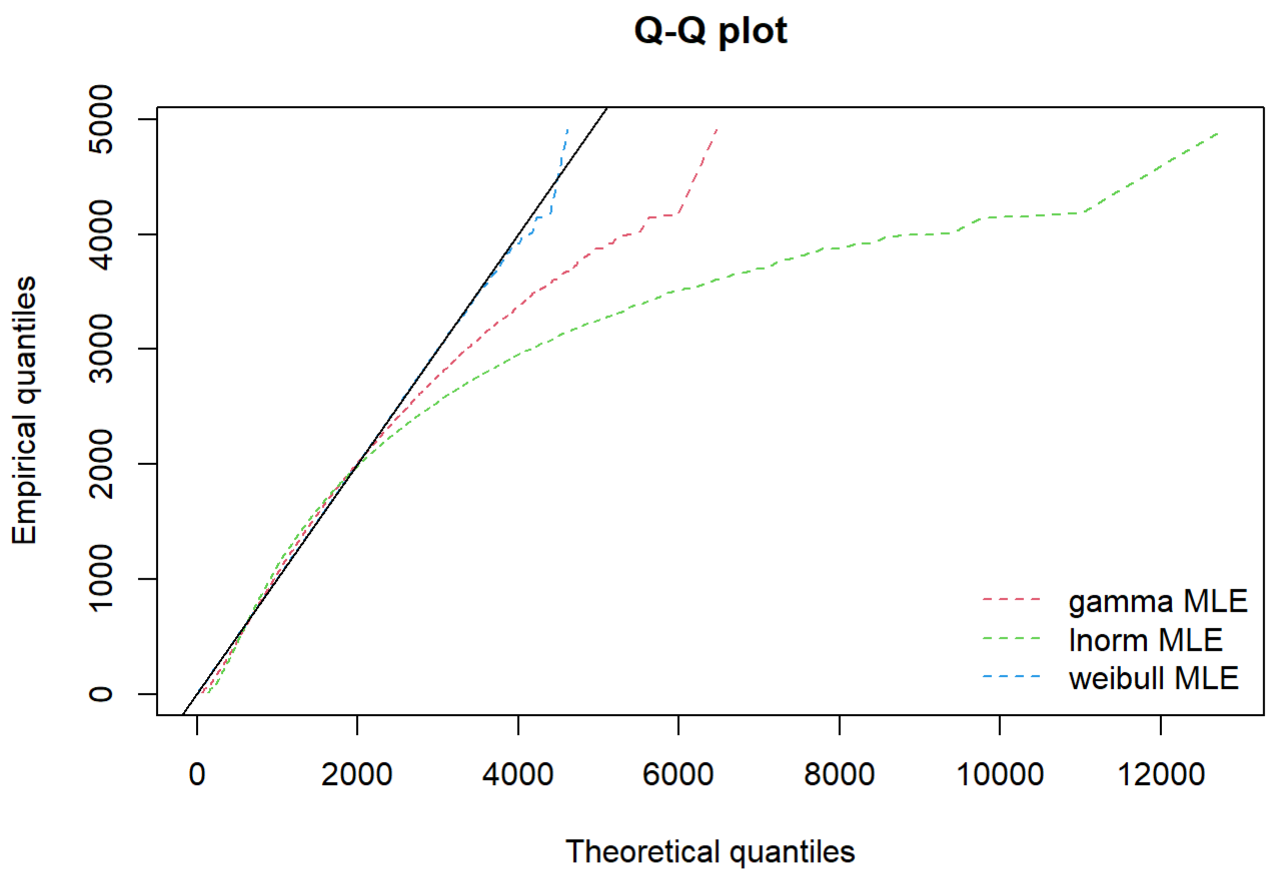

```
fitdistrplus::cdfcomp(list(fit.gamma.mle, fit.lnorm.mle, fit.weibull.mle),
  legendtext = plot.legend, fitlwd = 1, datapch = 10)
```




```
fitdistrplus::ppcomp(list(fit.gamma.mle, fit.lnorm.mle, fit.weibull.mle),
  legendtext = plot.legend, fitpch = 20)
```



```
fitdistrplus::qqcomp(list(fit.gamma.mle, fit.lnorm.mle, fit.weibull.mle),
  legendtext = plot.legend, fitpch = 20)
```




```
gofstat(list(fit.gamma.mle, fit.lnorm.mle, fit.weibull.mle, fit.gamma.mme, fit.lnorm.mme, fi
t.weibull.mme),
  fitnames = c("gamma MLE", "lnorm MLE", "weibull MLE", "gamma MME", "lnorm MME", "weibull MM
E"))
```

```
## Goodness-of-fit statistics
##                               gamma MLE    lnorm MLE weibull MLE    gamma MME
## Kolmogorov-Smirnov statistic  0.03943959   0.07520581 0.004271387   0.03328785
## Cramer-von Mises statistic   13.68946898  50.44231939 0.039274503  11.52012644
## Anderson-Darling statistic   81.68017684 309.01849236 0.218687383 102.63166339
##                                lnorm MME  weibull MME
## Kolmogorov-Smirnov statistic   0.06013151    0.5808157
## Cramer-von Mises statistic    41.12659777 2252.3165817
## Anderson-Darling statistic   473.20968512          Inf
##
## Goodness-of-fit criteria
##                               gamma MLE lnorm MLE weibull MLE gamma MME
## Akaike's Information Criterion  387106.6  390602.5    385992.3  387486.4
## Bayesian Information Criterion  387122.8  390618.7    386008.5  387502.6
##                               lnorm MME weibull MME
## Akaike's Information Criterion  394246.5         Inf
## Bayesian Information Criterion  394262.7         Inf
```

```
claimgof <- gofstat(list(fit.gamma.mle, fit.lnorm.mle, fit.weibull.mle, fit.gamma.mme, fit.ln
orm.mme, fit.weibull.mme), fitnames = c("gamma MLE", "lnorm MLE", "weibull MLE", "gamma MME",
"lnorm MME", "weibull MME"), chisqbreaks = c(0, 1000, 2000, 3000, 4000, 5000))
```

```
## Warning in cbind(Chi2$chisqtable, Chi2temp$chisqtable[, 2]): number of rows of
## result is not a multiple of vector length (arg 2)
```

```
claimgof$chisqpvalue
```

```
##   gamma MLE   lnorm MLE weibull MLE   gamma MME   lnorm MME weibull MME
##         NaN         NaN         NaN         NaN         NaN         NaN
```

```
claimgof$adtest
```

```
##      gamma MLE      lnorm MLE    weibull MLE      gamma MME      lnorm MME
##     "rejected" "not computed" "not rejected" "not computed" "not computed"
##    weibull MME
## "not computed"
```

```
claimgof$kstest
```

```
##      gamma MLE      lnorm MLE    weibull MLE      gamma MME      lnorm MME
##     "rejected"     "rejected" "not rejected"     "rejected"     "rejected"
##    weibull MME
##     "rejected"
```

```
claimgof$chisqtable
```

```
##         obscounts theo gamma MLE theo lnorm MLE theo weibull MLE theo gamma MME
## <= 0            0        0.00000         0.0000     0.000000e+00       0.000000
## <= 1000      6863     7428.28593      8274.4223     6.832498e+03    6814.235176
## <= 2000     12646    12187.21028     11085.3223     1.267619e+04   13105.072858
## <= 3000      4511     3979.00802      3554.5782     4.509988e+03    3892.294303
## <= 4000       423      736.11440      1031.3457     4.193486e+02     571.384613
## <= 5000         5      103.50753       322.0162     9.924350e+00      59.533949
## > 5000          0       13.87383       180.3153     5.531823e-02       5.479101
##         theo lnorm MME theo weibull MME
## <= 0           0.00000     0.000000e+00
## <= 1000     6613.76087     2.131748e-28
## <= 2000    13716.99540     2.444800e+04
## <= 3000     3382.25429     0.000000e+00
## <= 4000      596.42310     0.000000e+00
## <= 5000      109.46941     0.000000e+00
## > 5000        29.09694     0.000000e+00
```

We will be using Weibull distribution with shape = 2.286708 and scale = 1628.721687 to model claims severity.

```
weibull.shape = as.numeric(fit.weibull.mle$estimate["shape"])
weibull.shape
```

```
## [1] 2.286708
```

```
weibull.scale = as.numeric(fit.weibull.mle$estimate["scale"])
weibull.scale
```

```
## [1] 1628.722
```

```
mean(claim[claim > 0]) # empirical mean
```

```
## [1] 1443.124
```

```
weibull.scale * gamma(1+1/weibull.shape) # theoretical mean
```

```
## [1] 1442.817
```

```
var(claim[claim > 0]) # empirical variance
```

```
## [1] 447149
```

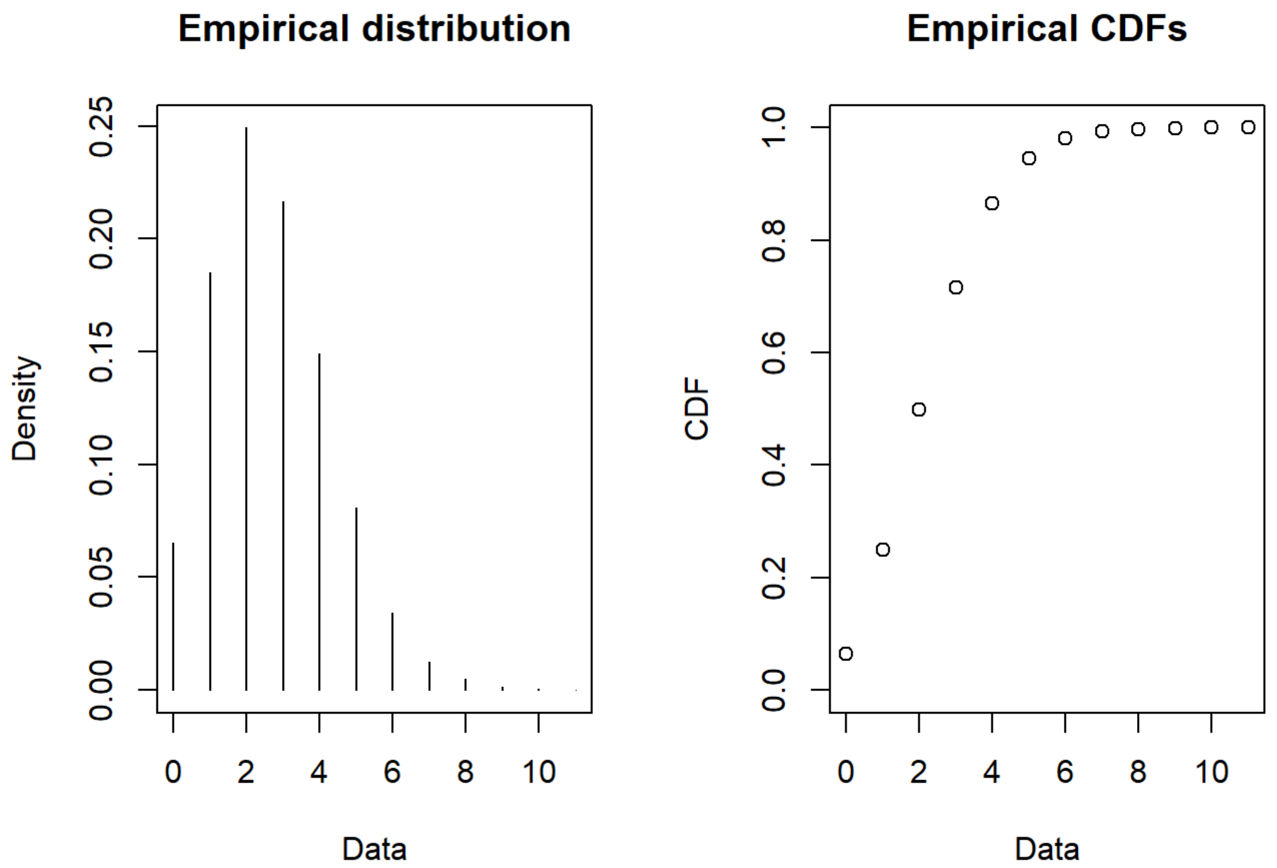
```
(weibull.scale ^ 2) * (gamma(1 + 2/weibull.shape) - (gamma(1 + 1/ weibull.shape)) ^ 2) # theo
retical variance
```

```
## [1] 447191.3
```

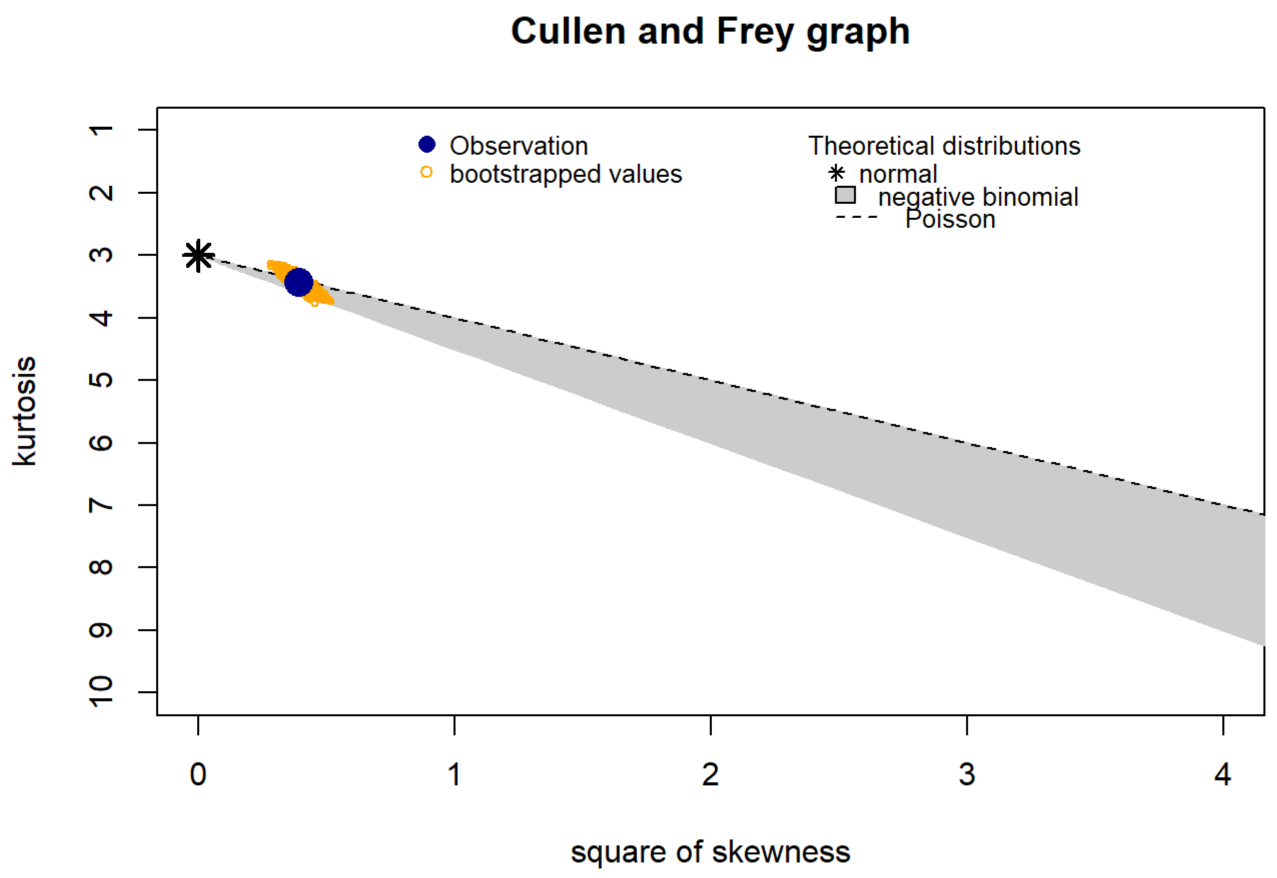
Checking fitness looks good!

Finding the distribution of the claims frequency:

```
claimfreq <- as.data.frame(table(cleaned_data$polind[claim > 0]))
claimfreq <- rbind(claimfreq, data.frame(Var1 = rep("manual", 593), Freq = rep(0, 593))) # ad
ding 593 polind which had no claim
plotdist(claimfreq$Freq, hist = TRUE, demp = TRUE, discrete = TRUE)
```




```
descdist(claimfreq$Freq, boot = 1000, discrete = TRUE)
```




```
## summary statistics
## ------
## min:  0   max:  11
## median:  3
## mean:  2.686889
## estimated sd:  1.632845
## estimated skewness:  0.6265714
## estimated kurtosis:  3.430748
```

We will try negative binomial and poisson

```
fit.nb.mme <- fitdist(claimfreq$Freq, "nbinom", method = "mme")
fit.nb.mme$estimate
```

```
##     size       mu
##      NaN 2.686889
```

```
fit.nb.mme$loglik
```

```
## [1] NaN
```

```
fit.pois.mle <- fitdist(claimfreq$Freq, "pois", method = "mle")
fit.pois.mle$estimate
```

```
##    lambda
## 2.686889
```

```
fit.pois.mle$loglik
```

```
## [1] -16980.86
```

```
fit.nb.mle <- fitdist(claimfreq$Freq, "nbinom", method = "mle")
fit.nb.mle$estimate
```

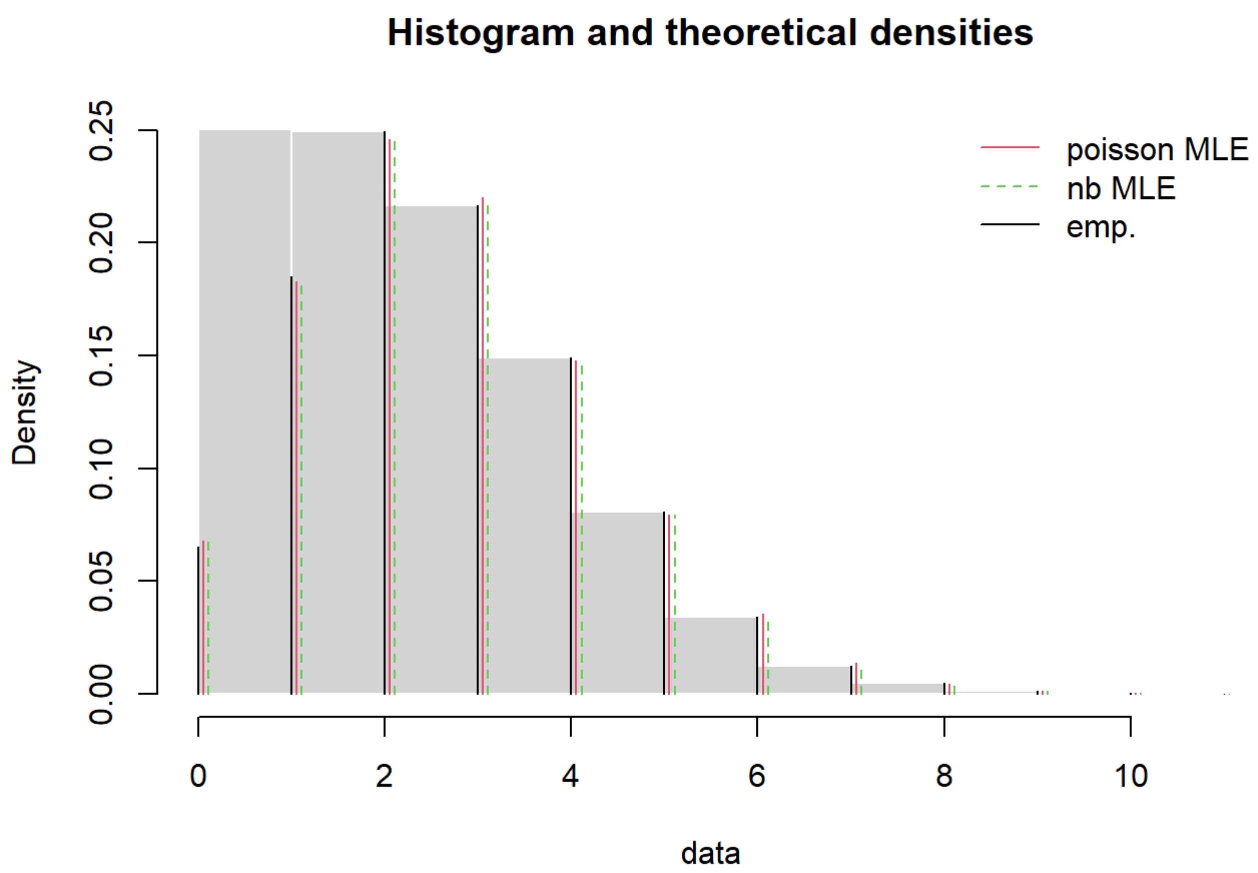
```
##         size           mu
## 38080.966228     2.687037
```

```
fit.nb.mle$loglik
```

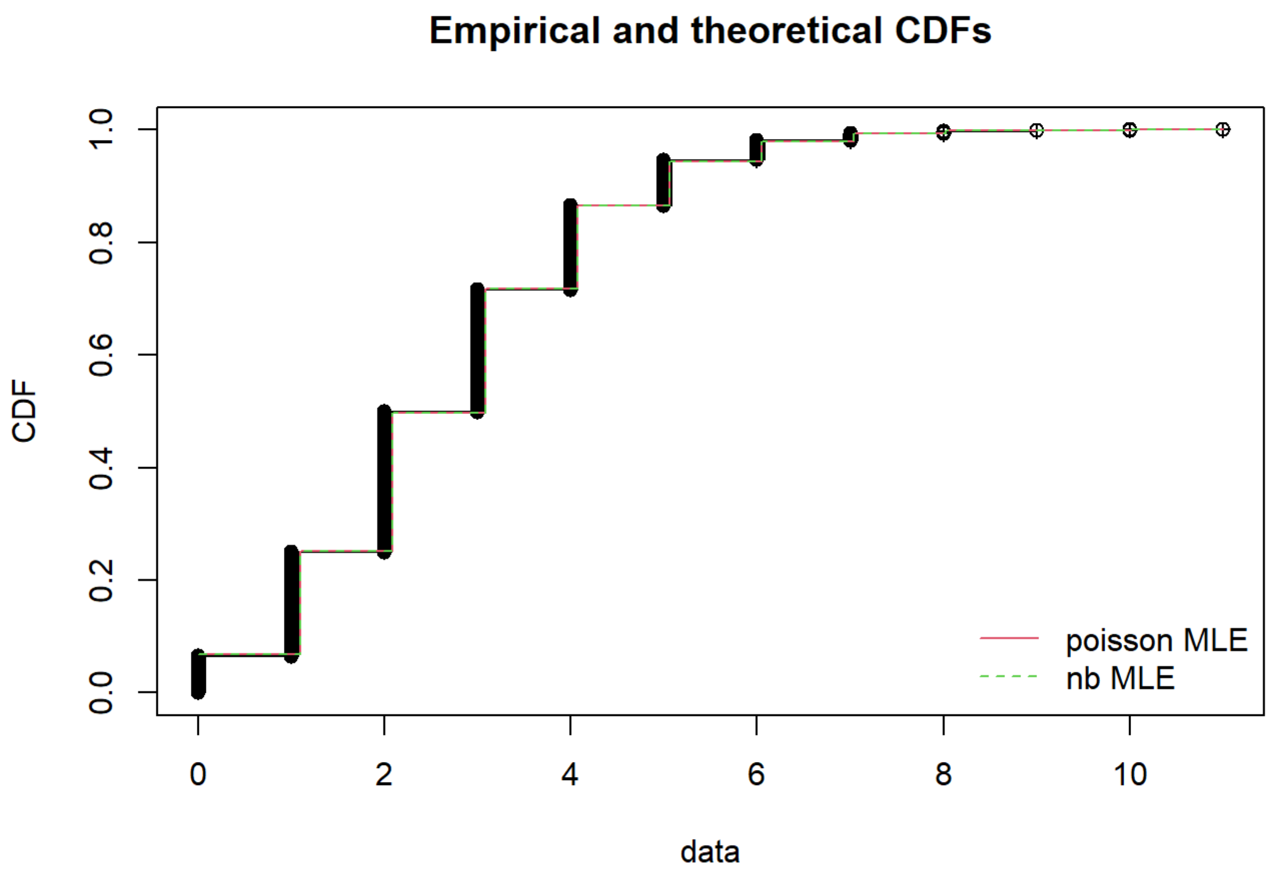
```
## [1] -16980.86
```

MLE and MME for Poisson distribution is the same. Since MME for negative binomial is incomplete, we will be comparing MLE for Poisson and Negative Binomial.

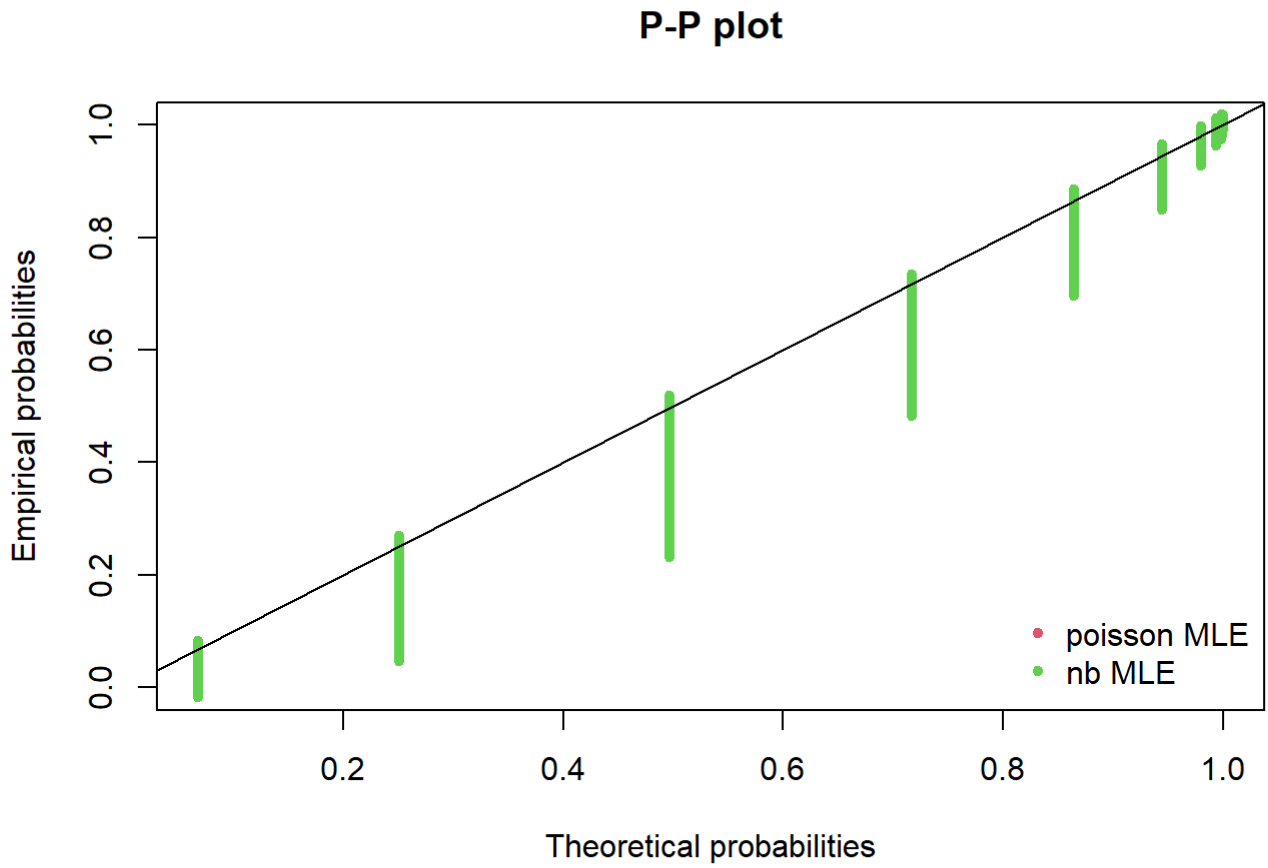
```
plot.legend <- c("poisson MLE", "nb MLE")
fitdistrplus::denscomp(list(fit.pois.mle, fit.nb.mle),
  legendtext = plot.legend, fitlwd = 1)
```



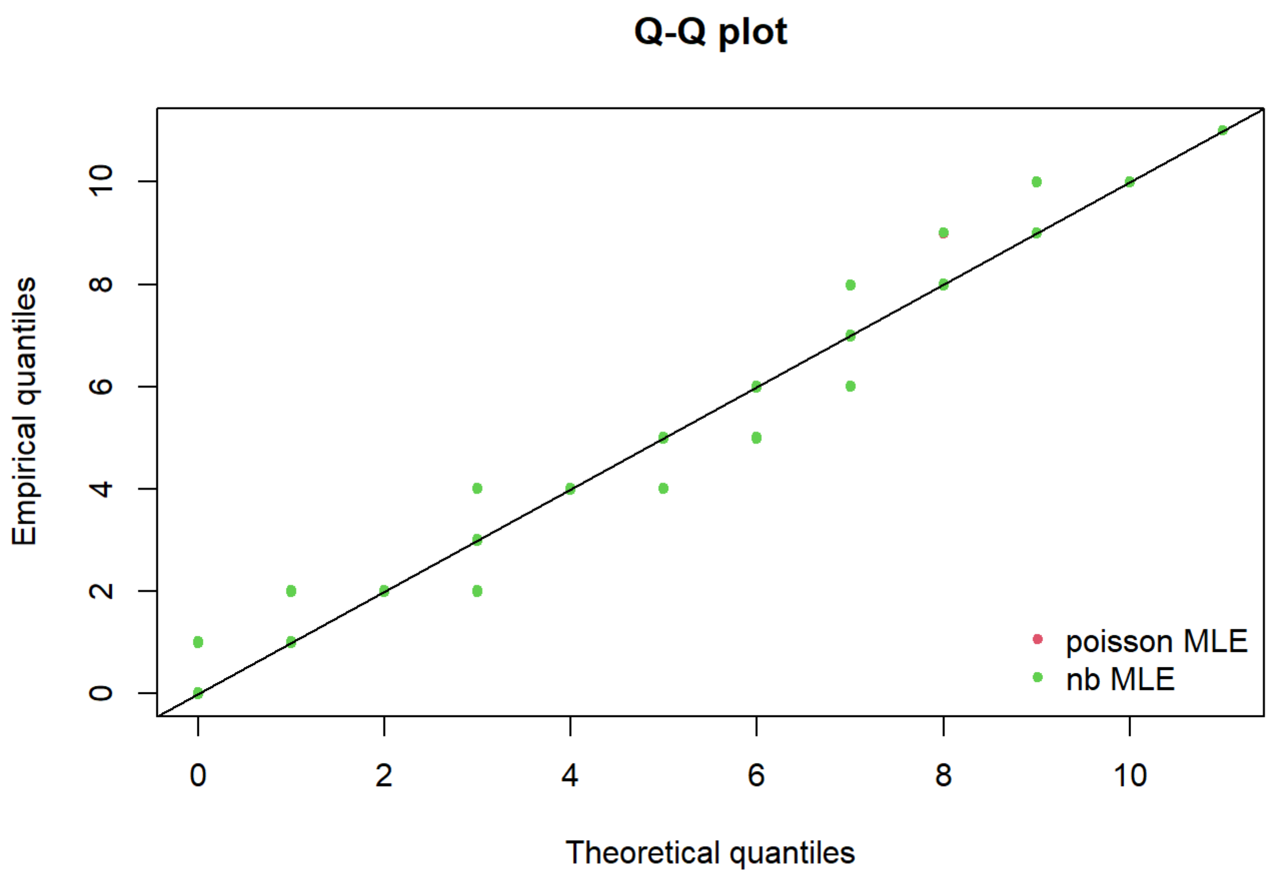
```
fitdistrplus::cdfcomp(list(fit.pois.mle, fit.nb.mle),
  legendtext = plot.legend, fitlwd = 1, datapch = 10)
```




```
fitdistrplus::ppcomp(list(fit.pois.mle, fit.nb.mle),
  legendtext = plot.legend, fitpch = 20)
```




```
fitdistrplus::qqcomp(list(fit.pois.mle, fit.nb.mle),
  legendtext = plot.legend, fitpch = 20)
```



```
gofstat(list(fit.pois.mle, fit.nb.mle),
  fitnames = c("poisson MLE", "nb MLE"))
```

```
## Chi-squared statistic:  3.420304 3.428835
## Degree of freedom of the Chi-squared distribution:  6 5
## Chi-squared p-value:  0.7545409 0.6341821
## Chi-squared table:
##      obscounts theo poisson MLE theo nb MLE
## <= 0       593         619.5734    619.5403
## <= 1      1683        1664.7247   1664.6102
## <= 2      2268        2236.4649   2236.3354
## <= 3      1971        2003.0441   2003.0023
## <= 4      1359        1345.4891   1345.5463
## <= 5       737         723.0359    723.1314
## <= 6       310         323.7862    323.8664
## > 6        178         182.8818    182.9676
##
## Goodness-of-fit criteria
##                                poisson MLE   nb MLE
## Akaike's Information Criterion    33963.71 33965.72
## Bayesian Information Criterion    33970.83 33979.95
```

```
claimgof <- gofstat(list(fit.pois.mle, fit.nb.mle), fitnames = c("poisson MLE", "nb MLE"), ch
isqbreaks = c(2, 4, 6, 8, 10))
```

We will use Poisson distribution with lambda = 2.686889 due to lower AIC and BIC scores.

```
pois.lambda <- as.numeric(fit.pois.mle$estimate)
pois.lambda # theoretical mean and variance
```

```
## [1] 2.686889
```

```
mean(claimfreq$Freq) # empirical mean
```

```
## [1] 2.686889
```

```
var(claimfreq$Freq) # empirical variance
```

```
## [1] 2.666184
```

Good Fitting!

Without reinsurance, Profit = Revenue - Total Loss

```
nopolind <- 9099 # total number of polind
# Loss = Mean loss per claim * # policies * Expected # claim per policy
Total_loss_insurer <- weibull.scale * gamma(1+1/weibull.shape) * nopolind * pois.lambda
Total_loss_insurer
```

```
## [1] 35273993
```

Reinsurance Conclusions: Type 1: Profit = Revenue - min(0, 1.5E[Y]) * claim - Total Price_1 Where Y is Loss Type 2: Profit = Revenue - min(0, 1.5E[S]) * policy - Total Price_2 Where S = Aggregate per Policyholder Assumption: Policyholder can only hold 1 Policy - Disadvantage! We can cross out Revenue since both cases have the same number.

Calculating min(0, 1.5E[Y]) - Type 1 Reinsurance

```
upper_bound1 = 1.5 * weibull.scale * gamma(1+1/weibull.shape)

sweibull <- function(x) {1 - pweibull(x, shape = weibull.shape, scale = weibull.scale)}

minfuncvalue <- integrate(sweibull, 0, upper_bound1)$value
minfuncvalue
```

```
## [1] 1383.099
```

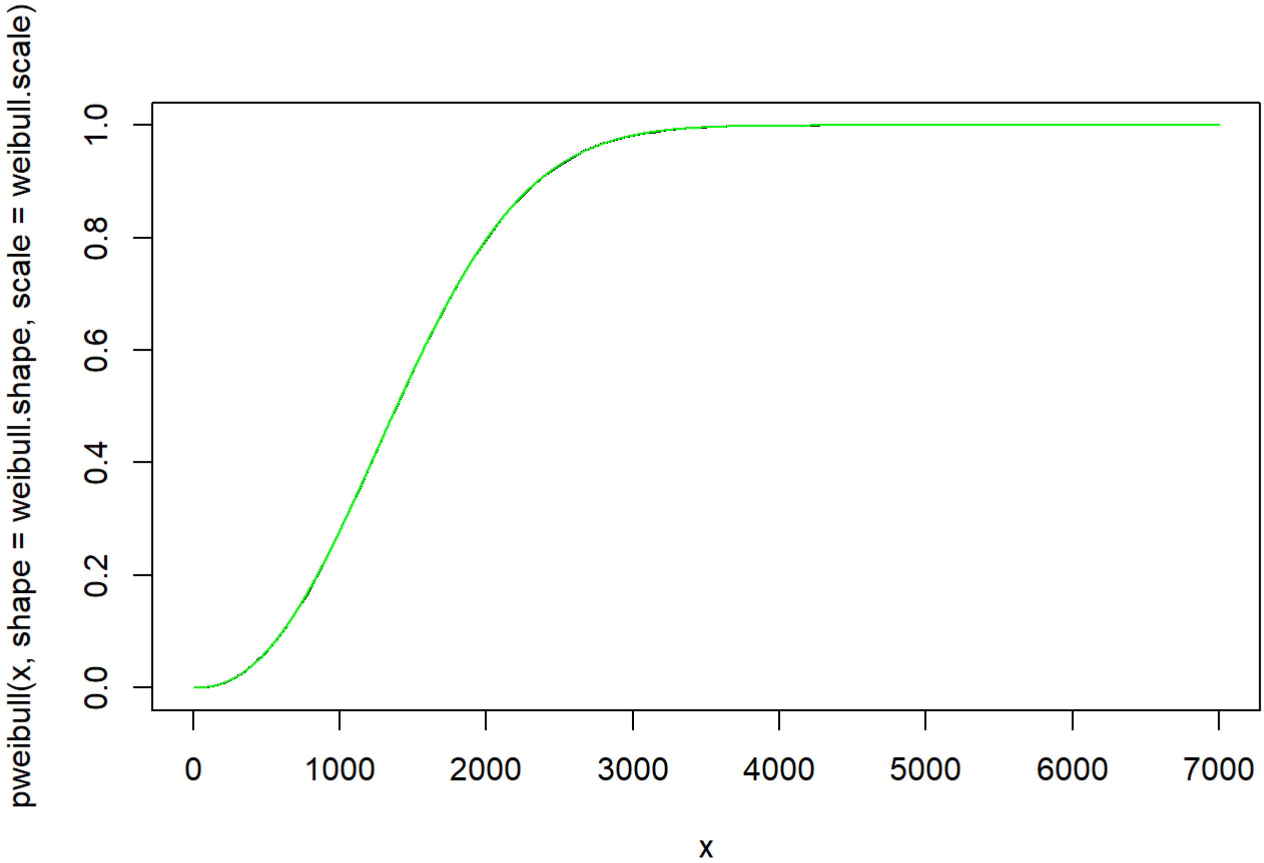
```
total_loss_insurer_1 <- minfuncvalue * nopolind * pois.lambda
total_loss_insurer_1
```

```
## [1] 33814016
```

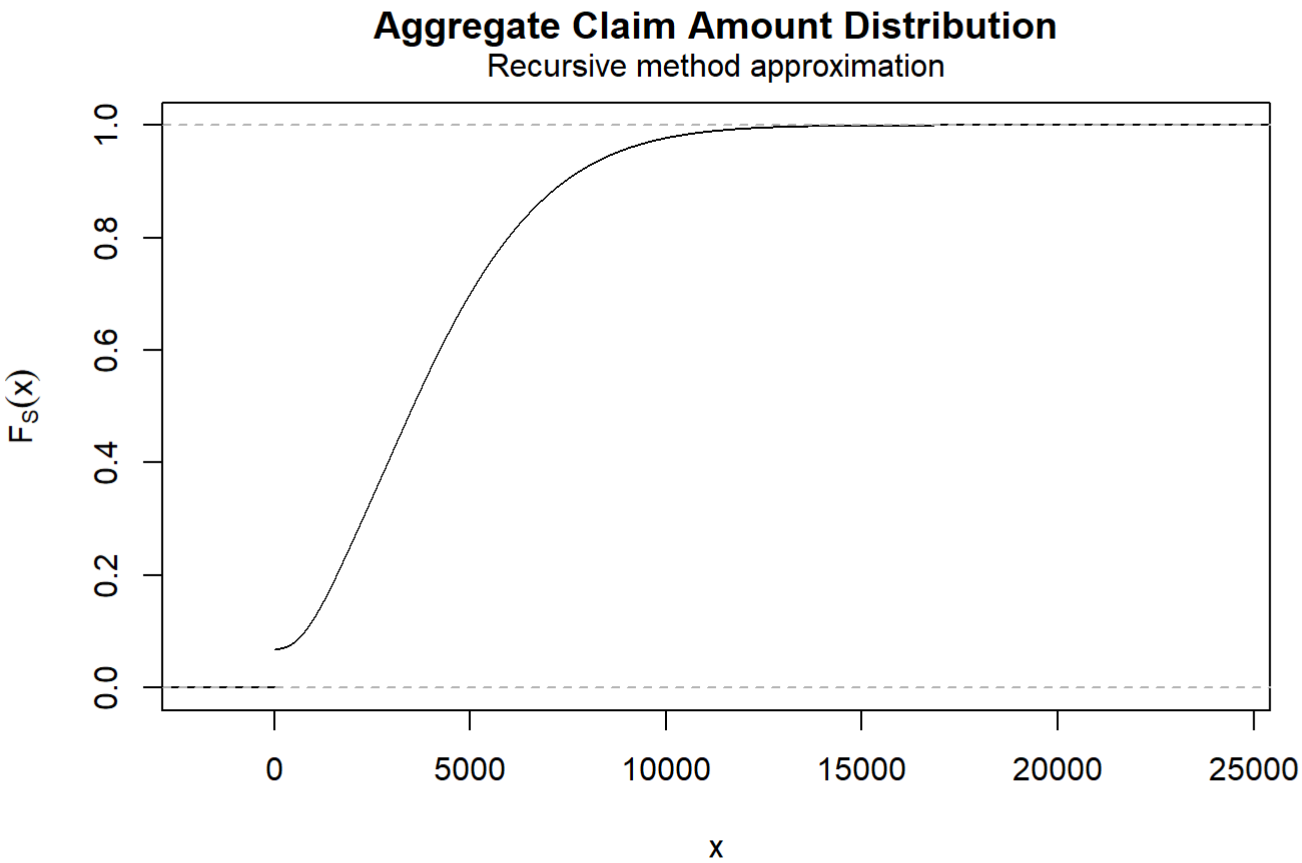
Calculating min(0, 1.5E[S]) - Type 2 Reinsurance Since per Policyholder Severity ~ Weibull and Frequency ~ Pois, We can model S ~ CompoundPois(lambda * v, Y ~ Weibull) this is per policy

```
expected_S = pois.lambda * weibull.scale * gamma(1+1/weibull.shape)
upper_bound2 = 1.5 * expected_S

stp = 1
final = 7000
# First, we discretise the proposed Weibull distribution
weibull.discr.unbia <- discretise(pweibull(x, shape = weibull.shape, scale = weibull.scale),
from = 0,
  to = final, step = stp, method = "unbiased", lev = levweibull(x,
    shape = weibull.shape, scale = weibull.scale))
weibull.discr.unbia.cdf <- cumsum(weibull.discr.unbia)
curve(pweibull(x, shape = weibull.shape, scale = weibull.scale), from = 0, to = final)
lines((0:(final/stp)) * stp, weibull.discr.unbia.cdf, type = "s",
  pch = 20, col = "green")
```




```r
S.unbia.cdf <- aggregateDist(method = "recursive", model.freq = "poisson",
  lambda = pois.lambda, model.sev = weibull.discr.unbia, x.scale = stp,
  maxit = 100000000)
plot(S.unbia.cdf, pch = 20, col = "black", cex = 0.5)
```




```
S.unbia.sdf <- function(x){1 - S.unbia.cdf(x)}

minfuncvalue2 <- integrate(S.unbia.sdf, 0, upper_bound2)$value
minfuncvalue2
```

```
## [1] 3463.925
```

```
total_loss_insurer_2 <- minfuncvalue2 * nopolind
total_loss_insurer_2
```

```
## [1] 31518257
```

In conclusion, insurer loss under reinsurance 1 is more than when under reinsurance 2. Without reinsurance, insurer loss is the greatest.