Implementing loss models based on splicing

Vermeir Jellen

December 21, 2015

Contents

1 Introduction

In an insurance and financial context it is important to have a clear view on both the risk and return aspects of contingent claims. From a risk management point of view, an insurer must take worst case scenarios into account and anticipate their potential future realizations. As such, extreme historical events can not be discarded during the modelling process and they must be handled appropriately.

In this paper we review a body and tail splicing approach for modeling extreme events. The goal of this approach is to take tail events into account explicitly by combining separate data generating processes for both the body and the tail of the distribution. As a practical consideration, the resulting model can assist an insurance company in the pricing of an unlimited excess-loss layer above an operational priority R.

2 Severity modeling - Unique data generating process

In this chapter we investigate the properties of a loss distribution and attempt to model the corresponding data sample by using a single data generating process. We start with an overview on the data characteristics and then proceed by fitting an exponential and a lognormal model to the sample. We conclude with an analysis regarding the goodness of fit of the respective models.

2.1 Data - Secura Re

In this paper we investigate the properties of the famous 'Secura Re' data set. The data contains realizations of losses that lie in the excess-loss layer above above an operational priority $R = 1200000$. The top two plots in figure 1. below show the empirical distribution function and the histogram of the data. The bottom left graph illustrates the severity of the losses per year while the bottom right graph shows a boxplot for the complete data set. The latter two plots illustrate that the data distribution is skewed to the right, and that an appropriate model should be selected to accommodate for potential future occurances of these extreme loss events.

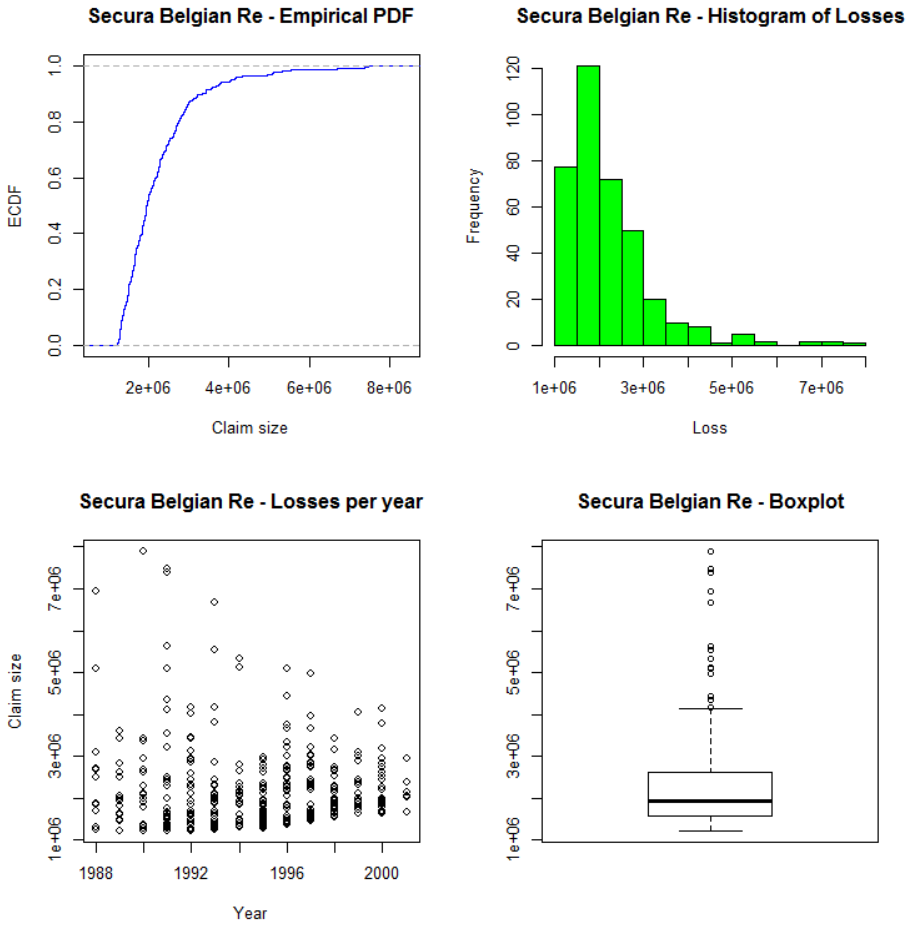

Figure 1: Secura Re - Empirical data

2.2 Unique data generating process approach

2.2.1 Exponential distribution

At first glance, the top two plots in figure 1. imply that an exponential distribution could potentially be a good fit for the data generating process of claim severity's that are greater than R. Hence, in this subsection we make the following assumption while modelling the data: $(X - R) \sim Exp(\lambda)$.

Appendix A.1 illustrates the calibration of the data to this exponential model. Here, we choose to calibrate the model via a maximum likelihood approach, for comparison reasons later on. Alternatively, the rate parameter λ of the exponential distribution can also be obtained via the method of moments. Figure 2. shows the results of the calibration process. The top two graphs

suggest that the model density and CDF are a relatively good fit for the empirical data. The bottom plot shows the QQ plot of the model quantiles versus the empirical quantiles. The plot implies that the model captures the loss data sufficiently well at the lower severity spectrum while data at the right tail is not adequately captured. We notice that most occurrences in the outer tail are slightly underestimated while the most severe occurences are understimated by the model. The AIC criterion for this model is 11017.52.

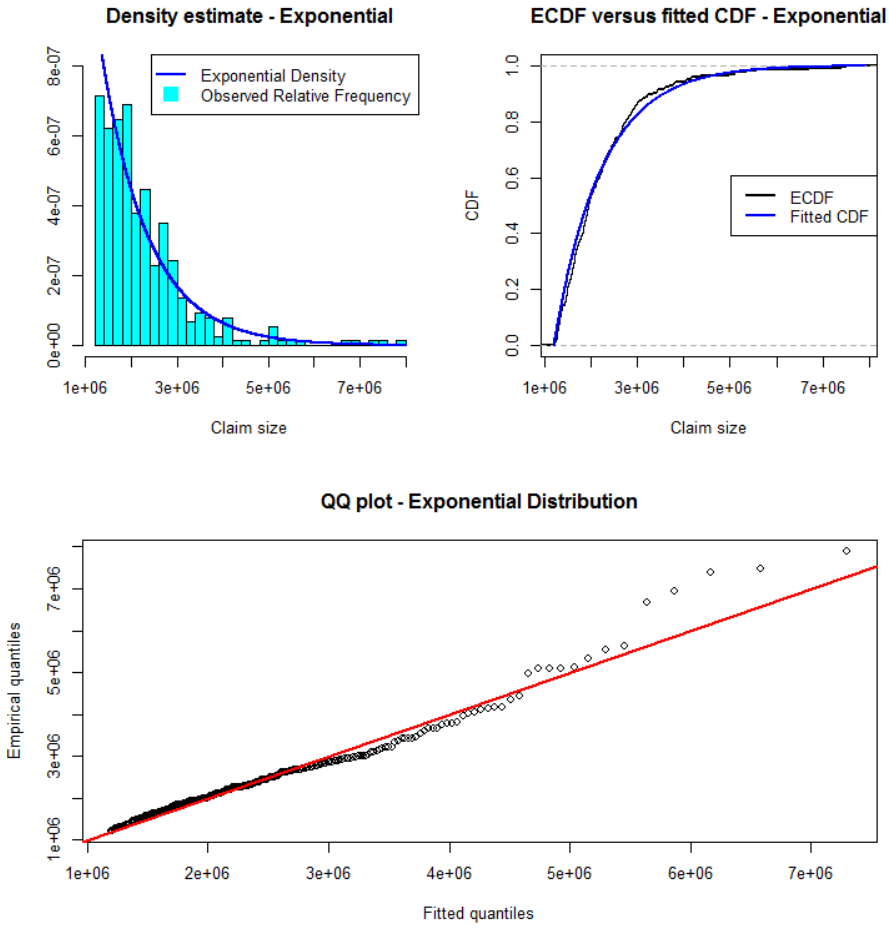

Figure 2: Secura Re - Exponential model fit

2.2.2 Lognormal distribution

In this subsection we repeat the fitting process for a lognormal distribution. Appendix A.2 illustrates the calibration of the data to the lognormal model;

Results are illustrated in figure 3. below. In comparison to the previous exponential model, we now again notice that the model fits the losses in the lower end of the severity spectrum adequately well. However, severity's of extreme losses are now strongly overestimated by the long right tail of the lognormal distribution. The AIC for this model is 11047.23.

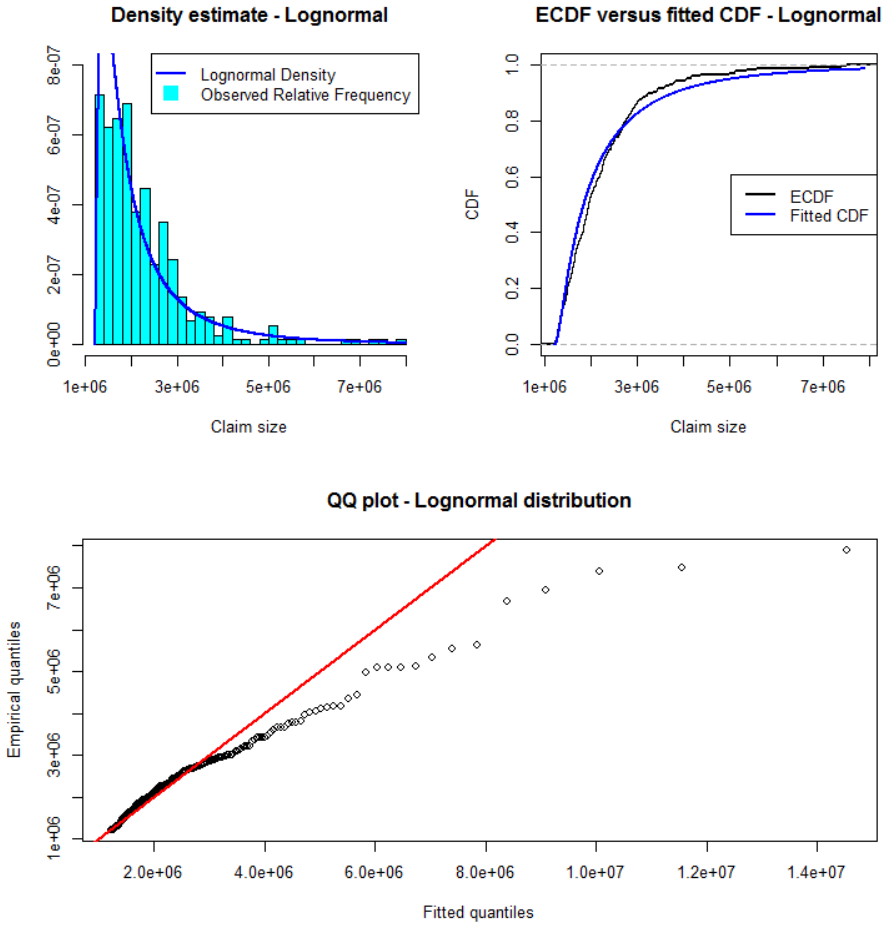

Figure 3: Secura Re - Lognormal model fit

3 Severity modeling - Splicing

In the previous chapter we illustrated that a single data generating process insufficiently captures the complete loss distribution spectrum. More precisely, the extreme losses in the tails are not captured adequately by these simple models. In this chapter we give a theoretical overview on splicing, which allows

us to overcome the latter problem. We then illustrate the concepts by calibrating two spliced distribution models to the loss data of the previous chapter.

3.1 Splicing - The theory

Splicing allows us to model a data set or data generating process by using multiple processes that are each responsible for generating data in distinct intervals. More concretely, a k-component spliced distribution has a density function that can be expressed as follows:

$$f_X(x) = \begin{cases} a_1 \cdot f_1(x), & c_0 \leq x < c_1 \\ a_2 \cdot f_2(x), & c_1 \leq x < c_2 \\ ... & ... \\ a_k \cdot f_k(x), & c_{k-1} \leq x < c_k \end{cases}$$

For $j = 1, ..., k$, each $a_j > 0$ and each $f_j(x)$ is a legitimate pdf with all probability on the interval $[c_{j-1}, c_j)$, and $\sum_j a_j = 1$. As an example, suppose that there are k pdfs, denoted by $g_1(x), ..., g_k(x)$ defined on the support $\Omega_x = [0, \text{inf})$. If we want $f_j(x)$ to be a legitimate pdf based on $g_j(x)$ restricted on the interval $[c_{j-1}, c_j)$ for $i = 1, ..., k$, then we can use $f_j(x) = \frac{g_j(x)}{G_j(c) - G_i(c_{j-1})}$. For the a_j we can use the empirical probabilities that x lies in the interval $[c_{j-1}, c_j)$.

In the context of this paper, splicing means that the density function of the losses can be modeled by using multiple data generating processes which correspond to different loss severity intervals.

3.2 Body-Tail approach

We investigate a body and tail splicing approach where a distribution for the body of the data is combined with a separate distribution for the tail. We start with a small overview on the generalized pareto distribution (GPD) because we will use this family of distributions to model the data in the tail. We then proceed with the calibration process and analyze the results in comparison to the single distribution approach.

3.2.1 Tail modeling - The generalized pareto distribution

The generalized pareto distribution is often used to model the tails of other distributions and is specified by 3 parameters: location μ, scale σ, and shape ε. The pdf of the distribution can be written as follows:

$f_{\varepsilon,\mu,\sigma}(x) = \frac{1}{\sigma}(1 + \frac{\varepsilon(x-\mu)}{\sigma})^{(-\frac{1}{\varepsilon}-1)}$

and corresponding CDF:

$$F_{\varepsilon,\mu,\sigma}(x) = \begin{cases} 1 - (1 + \frac{\varepsilon(x-\mu)}{\sigma})^{\frac{-1}{\varepsilon}}, & \text{for } \varepsilon \neq 0 \\ 1 - exp(-\frac{x-\mu}{\sigma}), & \text{for } \varepsilon = 0 \end{cases}$$

In this paper we use a GPD with $\mu = 0$ to model the losses above a certain threshold $x_b = 2580026$. The value of x_b was obtained via extreme value theory and its calculation falls outside the scope of this paper. When evaluating the data set we notice that the amount of losses greater than x_b corresponds to about $p_{gpd} = 25.6\%$

3.2.2 Exponential body, GPD tail

In this section we use the exponential distribution to model the body and the GPD distribution to model the tail of the loss distribution. Given our previous definitions of splicing and when taking the operational priority R into account, we arrive at the following model for the data generating process:

$$f(x) = \begin{cases} p_1 \cdot \frac{g(x)}{G(x_b)-G(R)}, & x < x_b \\ p_2 \cdot \frac{h(x)}{1-H(x_b)}, & x \geq x_b \end{cases}$$

Where $p_1 = 1 - p_{gdp}$ and $p_2 = p_{gdp}$. Here, $g(.)$ and $G(.)$ represent the density and cdf of the exponential distribution while $h(.)$ and $H(.)$ represent the density and cdf of the GPD distribution. This specification is equivalent to the model below, which we calibrate as illustrated in appendix A.3:

$$f(x) = \begin{cases} p_1 \cdot \frac{g(x-R)}{G(x_b)}, & x < x_b \\ p_2 \cdot h(x - x_b), & x \geq x_b \end{cases}$$

Figure 4. below illustrates the calibration results. In comparison to the single data generating processes we notice that the bulk of the data in the tail is now captured adequately well by the mixed distribution. Indeed, the AIC for the model is 11009.68, which is lower than the corresponding values of the previous models.

- Exponential and GPD.png

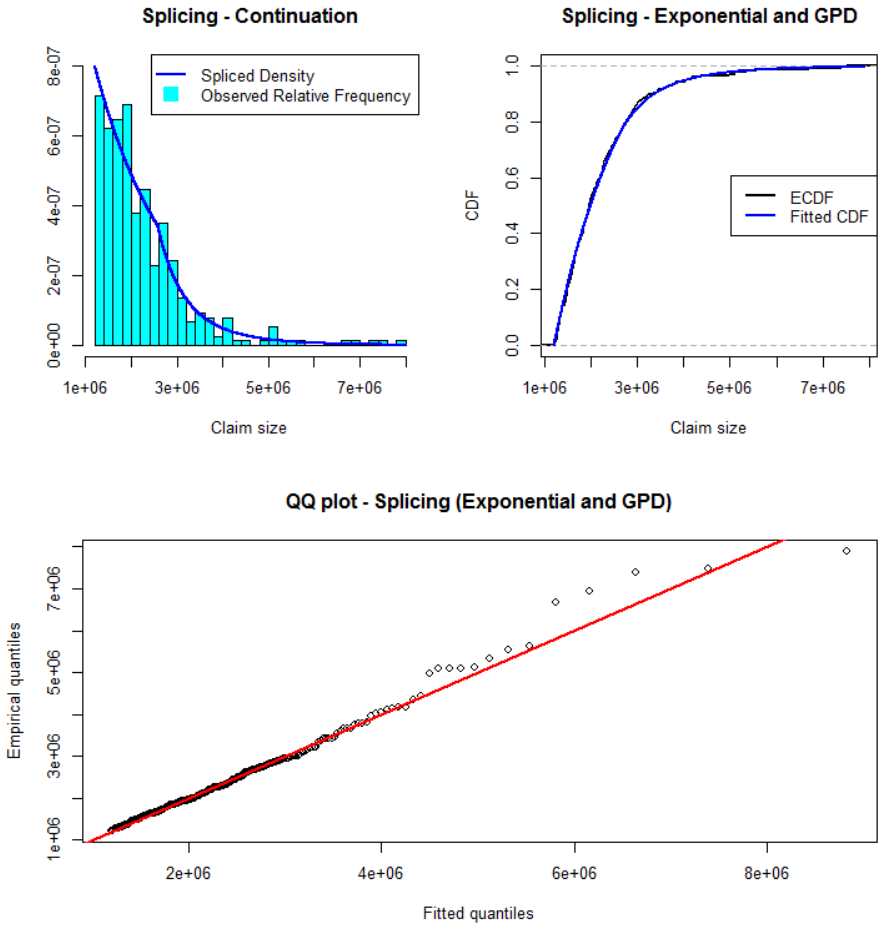

Figure 4: Splicing - Exponential body, GPD tail

Note that we also force $p_1 \cdot \frac{g(x_b-R)}{G(x_b-R)} = p_2 \cdot h(0)$ during the calibration process. This implies that $\sigma = \frac{p_2}{p_1} \cdot \frac{G(x_b-R)}{g(x_b-R)}$ and avoids discontinuity around x_b. The top plot in Figure 5. below illustrates what happens to the density fit when the continuity condition is not enforced: A discontinuity is clearly visible around the separation point of the two data generating processes, around $x = x_b$.

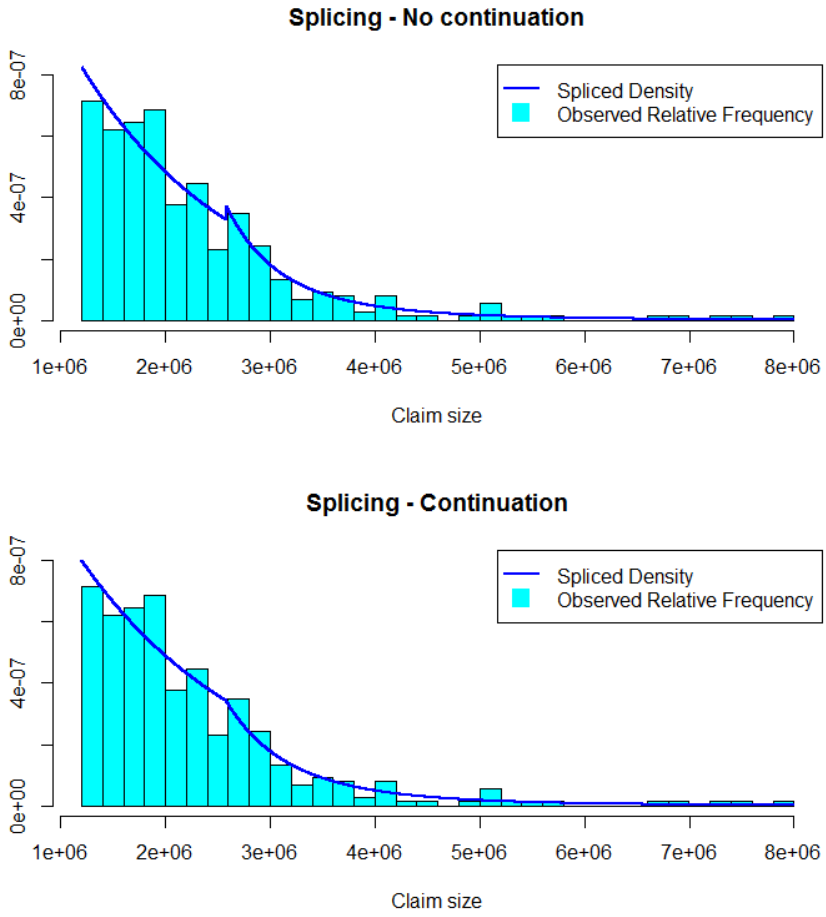

Figure 5: Splicing - Density continuation

3.2.3 Lognormal body, GPD tail

In this subsection we repeat the spliced body and tail calibration process but we now use a lognormal data generating process for the body. Results are illustrated in Figure 6. below. We notice that the results do not differ much from the previous spliced model. The lognormal distribution provides a good fit for the body of the distribution while the extreme outliers are again fitted by the GPD. The AIC for the model is 11008.3, which implies that it is the best out of the 4 models under consideration in this paper.

- Lognormal.png

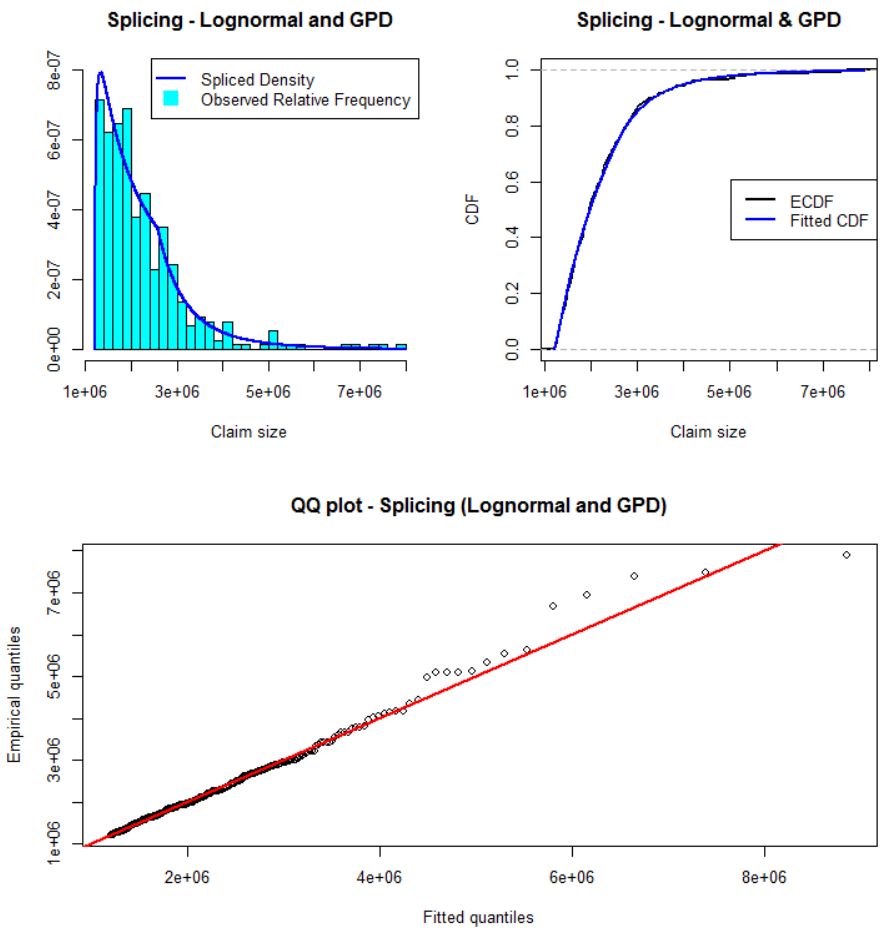

Figure 6: Splicing - Lognormal body, GPD tail

Appendices

A Code fractions

A.1 Secura Re - Exponential model fit

```
# 1. Exponential likelihood
Exp.lik <- function(data,p)
{
```

```
  Exp.lambda <- exp(p[1])
  Exp.Lik <- dexp(data,rate=Exp.lambda,log=TRUE)
  return(-sum(Exp.Lik))
}

fit.nlm <- nlm(Exp.lik, p = c(log(1/mean(Loss-trunclower))),
               data=Loss-trunclower,print.level=2)
Exp.lambda <- exp(fit.nlm$estimate[1])

layout(matrix(c(1,3,2,3),nrow=2,ncol=2))
truehist(Loss, nbins = 40, ylim = c(0, 8e-07),xlab="Claim size",
         main = "Density estimate - Exponential")
curve(dexp(x-trunclower,rate=Exp.lambda),
      from = trunclower, to = 8e+06, n = 10000,
      add = TRUE, col = "blue", lwd = 2)
legend('topright', legend = c("Exponential Density",
            "Observed Relative Frequency"), col = c("blue", "cyan"),
       pch=c(NA,15), pt.cex=2, lty = c(1, NA), lwd = c(2, NA))

plot(ecdf(Loss), do.points = FALSE, xlab = 'Claim size', ylab = 'CDF',
     main = 'ECDF versus fitted CDF - Exponential',
     xlim = c(trunclower, max(Loss)), lwd = 2)
curve((x >= trunclower) * pexp(x-trunclower,rate=Exp.lambda),
      n = 1000, col = "blue", lwd = 2, add = TRUE)
legend('right', c('ECDF', 'Fitted CDF'), col = c(1, 4), lwd = 2)


qExp <- function(pinput,data){
  # solve F_X(x) = p numerically by minimizing (F_X(x) - p)^2
  objective <- function(xi){
    x <- exp(xi)
    F_X=0
    if(x >= 0)
      F_X <- pexp(x,rate=Exp.lambda)
    return((F_X-pinput)^2)
  }
  optim(par = log(qexp(pinput, 1/mean(data))),
        objective, method = "L-BFGS-B")$par
}

n <- length(Loss)
p <- 1:n/(n+1)
q <- exp(sapply(p,qExp,data=Loss-trunclower) )+ trunclower
q <- qexp(p,rate=Exp.lambda)+trunclower
plot(q, sort(Loss),
     xlab = "Fitted quantiles", ylab = "Empirical quantiles", pch=1,
```

```
      main = "QQ plot - Exponential Distribution")
abline(a = 0, b = 1, lwd = 2, col = "red")

AIC.EXP <- (+2)*fit.nlm$minimum + 2*length(fit.nlm$estimate)
AIC.EXP
```

A.2 Secura Re - Lognormal model fit

```
#2. lognormal likelihood
LN.lik <- function(data,p)
{
  LN.mu <- p[1]
  LN.sigma <- exp(p[2])
  LN.Lik <- dlnorm(data, meanlog=LN.mu, sdlog=LN.sigma, log=TRUE)
  # LN.Lik <- -log(data) - log(LN.sigma) -
  #                    0.5*log(2*pi)-0.5*(log(data)-LN.mu)^2/LN.sigma^2

  return(-sum(LN.Lik))
}

fit.nlm <- nlm(LN.lik, p = c(mean(log(Loss-trunclower)),
               log(sd(log(Loss-trunclower)))),data=Loss-trunclower,
                                                    print.level=2)
LN.mu <- fit.nlm$estimate[1]
LN.sigma <- exp(fit.nlm$estimate[2])

truehist(Loss, nbins = 40, ylim = c(0, 8e-07),
         main = "Density estimate - Lognormal",
                                      xlab = "Claim size")
curve(dlnorm(x-trunclower,meanlog=LN.mu, sdlog=LN.sigma),
                  main = "Density estimate - Lognormal",
                  from = trunclower, to = 8e+06, n = 10000,
                        add = TRUE, col = "blue", lwd = 2)
legend('topright', legend = c("Lognormal Density",
          "Observed Relative Frequency"), col = c("blue", "cyan"),
            pch=c(NA,15), pt.cex=2, lty = c(1, NA), lwd = c(2, NA))

plot(ecdf(Loss), do.points = FALSE, xlab = 'Claim size', ylab = 'CDF',
      main = "ECDF versus fitted CDF - Lognormal",
                        xlim = c(trunclower, max(Loss)), lwd = 2)
curve((x >= trunclower) * plnorm(x-trunclower,meanlog=LN.mu, sdlog=LN.sigma),
       n = 1000, col = "blue", lwd = 2, add = TRUE)
legend('right', c('ECDF', 'Fitted CDF'), col = c(1, 4), lwd = 2)


qLN1 <- function(pinput,data){
```

```
  # solve F_X(x) = p numerically by minimizing (F_X(x) - p)^2
  objective <- function(xi){
    x <- exp(xi)
    F_X=0
    if(x >= 0)
      F_X <- plnorm(x, meanlog=LN.mu, sdlog=LN.sigma)
    return((F_X-pinput)^2)
  }
  optim(par = log(qlnorm(pinput, mean(log(data)), sd(log(data)))),
                          objective, method = "L-BFGS-B")$par
}

n <- length(Loss)
p <- 1:n/(n+1)
q <- exp(sapply(p,qLN1,data=Loss-trunclower)) + trunclower
q <- qlnorm(p, meanlog=LN.mu, sdlog=LN.sigma)+trunclower
plot(q, sort(Loss), main = "QQ plot - Lognormal distribution",
     xlab = "Fitted quantiles", ylab = "Empirical quantiles", pch=1)
abline(a = 0, b = 1, lwd = 2, col = "red")


AIC.LN <- (+2)*fit.nlm$minimum + 2*length(fit.nlm$estimate)
AIC.LN
```

A.3 Splicing - Exponential body, GPD tail

```
# threshold specified using EVT technique
threshold = 2580026
# determine splicing weights (using bernouilli MLE)
n = length(Loss)
k = length(Loss[Loss > threshold])
# body component
pn <- (n-k)/n
# tail component
k/n

# 3. Body-Tail: Exponential + GPD
EXPGDP.lik = function(p){
  EXP.lambda <- exp(p[1])

  GDP.shape <- p[2]
  # GDP.sigma <- exp(p[3])

  GDP.sigma <- (1-pn)*pexp(threshold-trunclower,rate=EXP.lambda)/
    (pn*dexp(threshold-trunclower,rate=EXP.lambda))
```

```
  seqThreshold <- which(Loss <= threshold)
  gtThreshold <- which(Loss > threshold)

  L1 <- pn * dexp(Loss[seqThreshold]-trunclower,
                                  rate=EXP.lambda) /
              pexp(threshold-trunclower, rate=EXP.lambda)
  L2 <- (1-pn) * (1/GDP.sigma)*(1 + (GDP.shape*
                          (Loss[gtThreshold]-threshold)) /
                            GDP.sigma)^(-1-1/GDP.shape)

  # Log Likelihood
  logL <- sum(c(log(c(L1,L2))))
  return(-logL)
}

# fit.nlm <- nlm(EXPGDP.lik, p = c(log(1/mean(Loss-trunclower)),
#                                  1/4,log(748329)),print.level=2)
fit.nlm <- nlm(EXPGDP.lik, p = c(log(1/mean(Loss-trunclower)),
                                   1/4),print.level=2)
Exp.lambda = exp(fit.nlm$estimate[1])
GDP.shape <- fit.nlm$estimate[2]
# GDP.sigma <- exp(fit.nlm$estimate[3])

GDP.sigma <- (1-pn)*pexp(threshold-trunclower,rate=Exp.lambda)/
  (pn*dexp(threshold-trunclower,rate=Exp.lambda))

layout(matrix(c(1,3,2,3),nrow=2,ncol=2))
truehist(Loss, nbins = 40, ylim = c(0, 8e-07),
         xlab="Claim size", main="Splicing - Continuation")
curve(I(x <= threshold) * pn * dexp(x-trunclower,rate=Exp.lambda) /
        pexp(threshold-trunclower, rate=Exp.lambda) +
        (1-I(x <= threshold)) * (1-pn) * (1/GDP.sigma)*
        (1 + (GDP.shape*(x-threshold))/GDP.sigma)^(-1-1/GDP.shape),
      from = trunclower, to = 8e+06, n = 10000,
      add = TRUE, col = "blue", lwd = 2)
legend('topright', legend = c("Spliced Density",
                              "Observed Relative Frequency"),
             col = c("blue", "cyan"), pch=c(NA,15), pt.cex=2,
                               lty = c(1, NA), lwd = c(2, NA))

plot(ecdf(Loss), do.points = FALSE,
           xlab = 'Claim size', ylab = 'CDF',
             main = 'Splicing - Exponential and GPD',
               xlim = c(trunclower, max(Loss)), lwd = 2)
# Fitted CDF
curve((x >= trunclower) * ((x <= threshold) * pn *
```

```
                pexp(x-trunclower,rate=Exp.lambda) /
              pexp(threshold-trunclower, rate=Exp.lambda) +
            (x > threshold)*(pn + (1-pn)*(1-(1+ GDP.shape *
                  (x-threshold)/GDP.sigma)^(-1/GDP.shape)))),
                    n = 1000, col = "blue", lwd = 2, add = TRUE)
legend('right', c('ECDF', 'Fitted CDF'), col = c(1, 4), lwd = 2)

## QQ plot
# this funtion takes a probablity p as argument
# and returns the quantile x for which F_X(x) = p
qsplicing <- function(p){
  # solve F_X(x) = p numerically by minimizing (F_X(x) - p)^2
  objective <- function(xi){
    x <- exp(xi)
    F_X=500
    if(x >= trunclower & x <= threshold)
      F_X <- pn * pexp(x-trunclower,rate=Exp.lambda) /
      pexp(threshold-trunclower, rate=Exp.lambda)
    if(x > threshold)
      F_X <- (pn + (1-pn)*
        (1-(1+ GDP.shape*(x-threshold)/GDP.sigma)^(-1/GDP.shape)))
    return((F_X-p)^2)

  }
  if(p < pn)
    init <- log(qexp(p,rate=Exp.lambda)+trunclower)
  else
    init <- log(qlnorm(p,mean(log(Loss)),sd(log(Loss))))
  optim(par = init, objective, method = "L-BFGS-B")$par
}

n <- length(Loss)
p <- 1:n/(n+1)
q <- exp(sapply(p, qsplicing))
plot(q, sort(Loss),
     main = "QQ plot - Splicing (Exponential and GPD)",
     xlab = "Fitted quantiles", ylab = "Empirical quantiles",
                                                    pch=1)
abline(a = 0, b = 1, lwd = 2, col = "red")

AIC.EXPGPD <- (+2)*fit.nlm$minimum + 2*length(fit.nlm$estimate)
AIC.EXPGPD
```

A.4 Splicing - lognormal body, GPD tail

```
# 4. Body-Tail: Lognormal + GPD
```

```
LNGDP.lik = function(p){
  LN.mu <- p[1]
  LN.sigma <- exp(p[2])

  GDP.shape <- p[3]
  GDP.sigma <- (1-pn)*
       plnorm(threshold-trunclower,meanlog=LN.mu,sdlog=LN.sigma)/
    (pn*dlnorm(threshold-trunclower,meanlog=LN.mu,sdlog=LN.sigma))

  seqThreshold <- which(Loss <= threshold)
  gtThreshold <- which(Loss > threshold)

  L1 <- pn * dlnorm(Loss[seqThreshold]-
                      trunclower,meanlog=LN.mu,sdlog=LN.sigma) /
       plnorm(threshold-trunclower, meanlog=LN.mu,sdlog=LN.sigma)
  L2 <- (1-pn) * (1/GDP.sigma)*
                  (1 + (GDP.shape*(Loss[gtThreshold]-threshold)) /
                                          GDP.sigma)^(-1-1/GDP.shape)

  # Log Likelihood
  logL <- sum(c(log(c(L1,L2))))
  return(-logL)
}

fit.nlm <- nlm(LNGDP.lik, p = c(mean(log(Loss)),
                    log(sd(log(Loss))),0.1),print.level=2)
LN.mu = fit.nlm$estimate[1]
LN.sigma = exp(fit.nlm$estimate[2])
GDP.shape <- fit.nlm$estimate[3]
GDP.sigma <- (1-pn)*
     plnorm(threshold-trunclower,meanlog=LN.mu,sdlog=LN.sigma)/
  (pn*dlnorm(threshold-trunclower,meanlog=LN.mu,sdlog=LN.sigma))

layout(matrix(c(1,3,2,3),nrow=2,ncol=2))
truehist(Loss, nbins = 40, ylim = c(0, 8e-07),
         xlab = "Claim size", main="Splicing - Lognormal and GPD")
curve(I(x <= threshold) * pn *
           dlnorm(x-trunclower,meanlog=LN.mu,sdlog=LN.sigma) /
    plnorm(threshold-trunclower, meanlog=LN.mu,sdlog=LN.sigma) +
                  (1-I(x <= threshold)) * (1-pn) * (1/GDP.sigma)*
        (1 + (GDP.shape*(x-threshold))/GDP.sigma)^(-1-1/GDP.shape),
                          from = trunclower, to = 8e+06, n = 10000,
                               add = TRUE, col = "blue", lwd = 2)
legend('topright', legend = c("Spliced Density",
    "Observed Relative Frequency"), col = c("blue", "cyan"),
        pch=c(NA,15), pt.cex=2, lty = c(1, NA), lwd = c(2, NA))
```

```
plot(ecdf(Loss), do.points = FALSE, xlab = 'Claim size',
                 ylab = 'CDF', main = "Splicing - Lognormal & GPD",
                                xlim = c(trunclower, max(Loss)), lwd = 2)
curve((x >= trunclower) * ((x <= threshold) * pn *
              plnorm(x-trunclower,meanlog=LN.mu,sdlog=LN.sigma) /
         plnorm(threshold-trunclower, meanlog=LN.mu,sdlog=LN.sigma) +
                        (x > threshold) * (pn + (1-pn)*(1-(1+ GDP.shape*
                              (x-threshold)/GDP.sigma)^(-1/GDP.shape)))),
                             n = 1000, col = "blue", lwd = 2, add = TRUE)
legend('right', c('ECDF', 'Fitted CDF'), col = c(1, 4), lwd = 2)

# this funtion takes a probablity p as argument
# and returns the quantile x
# for which F_X(x) = p
qsplicing <- function(p){
  # solve F_X(x) = p numerically by minimizing (F_X(x) - p)^2
  objective <- function(xi){
    x <- exp(xi)
    F_X=50000
    if(x >= trunclower & x <= threshold)
      F_X <- pn * plnorm(x-trunclower,meanlog=LN.mu,sdlog=LN.sigma) /
           plnorm(threshold-trunclower, meanlog=LN.mu,sdlog=LN.sigma)
    if(x > threshold)
      F_X <- (pn + (1-pn)*
        (1-(1+ GDP.shape*(x-threshold)/GDP.sigma)^(-1/GDP.shape)))
    return((F_X-p)^2)
  }
  if(p < pn)
    init <- log(qlnorm(p, LN.mu, LN.sigma)+trunclower)
  else
    init <- log(qlnorm(p, mean(log(Loss)), sd(log(Loss))))
  optim(par = init, objective, method="L-BFGS-B")$par
}
n <- length(Loss)
p <- 1:n/(n+1)
q <- exp(sapply(p, qsplicing))
plot(q, sort(Loss), main = "QQ plot - Splicing (Lognormal and GPD)",
     xlab = "Fitted quantiles", ylab = "Empirical quantiles", pch=1)
abline(a = 0, b = 1, lwd = 2, col = "red")


AIC.LNGPD <- (+2)*fit.nlm$minimum + 2*length(fit.nlm$estimate)
AIC.LNGPD
```

AIC . EXP
AIC . LN
AIC . EXPGPD
AIC . LNGPD