Risk Modelling in Insurance - Final project

Klemen Hovnik and Manca Strgar

PART 1

1 Importing dataset

First, we need to import Insurance contract data from the file `NonFleetCo507` into R. The file contains 159.947 contracts with 11 variables. The column `Clm Count` shows how many claims were filed on the contract. This is the loss frequency data that we will use for model fitting. The column `TLength` shows the period of exposure during which the contract was active and the Clm Count was filed. Let's have a look at our dataset:

AgeInsured	SexInsured	Experience	TLength	Clm_Count	VAge	PrivateCar	NCD_0	Cover_C	VehCapCubic	VehCapTonn
32	M	11	0.4654346	0	10	1	0	1	1797	0
26	M	5	0.8076660	0	13	1	1	0	1590	0
32	M	5	0.3997262	0	0	1	0	1	1997	0
32	M	5	0.5831622	0	1	1	0	1	1997	0
41	M	14	0.7748118	0	9	1	0	0	1597	0
28	F	3	0.4928131	0	0	1	1	1	1587	0

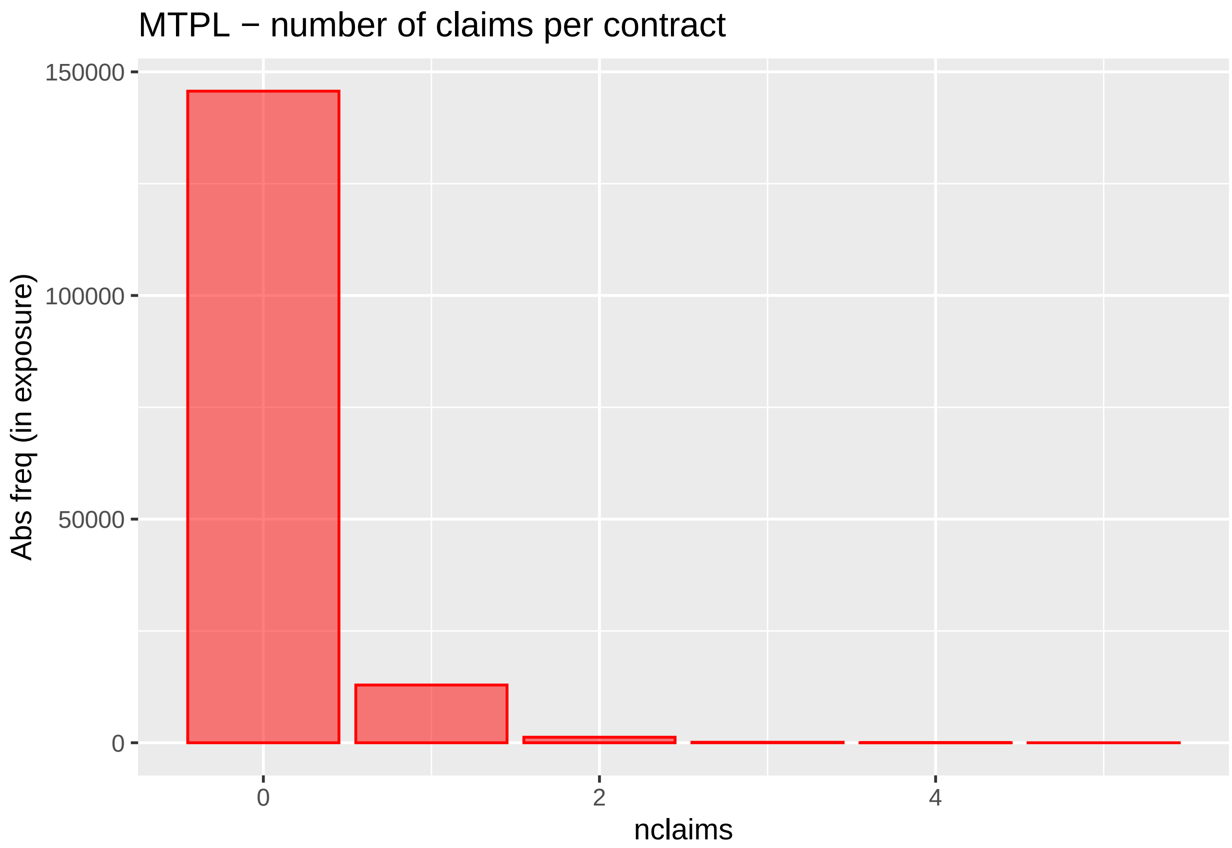
```
## tibble [159,947 x 11] (S3: tbl_df/tbl/data.frame)
##  $ AgeInsured : int [1:159947] 32 26 32 32 41 28 28 51 27 45 ...
##  $ SexInsured : chr [1:159947] "M" "M" "M" "M" ...
##  $ Experience : int [1:159947] 11 5 5 5 14 3 3 20 9 19 ...
##  $ TLength    : num [1:159947] 0.465 0.808 0.4 0.583 0.775 ...
##  $ Clm_Count  : int [1:159947] 0 0 0 0 0 0 0 0 0 0 ...
##  $ VAge       : int [1:159947] 10 13 0 1 9 0 1 2 29 21 ...
##  $ PrivateCar : int [1:159947] 1 1 1 1 1 1 1 1 1 1 ...
##  $ NCD_0      : int [1:159947] 0 1 0 0 0 1 1 1 0 0 ...
##  $ Cover_C    : int [1:159947] 1 0 1 1 0 1 1 1 0 0 ...
##  $ VehCapCubic: int [1:159947] 1797 1590 1997 1997 1597 1587 1587 1587 1285 1166 ...
##  $ VehCapTonn : int [1:159947] 0 0 0 0 0 0 0 0 0 0 ...
```

We could rename colums in our dataset to make it easier to work with. We will rename the column `Clm_Count` into `nclaims` as the number of claims, and `TLength` into `expo` as exposure. For easier programming we will also rename `AgeInsued` and `SexInsured` into `age` and `sex`. We will also delete colums which are not important for our analysis rightnow.

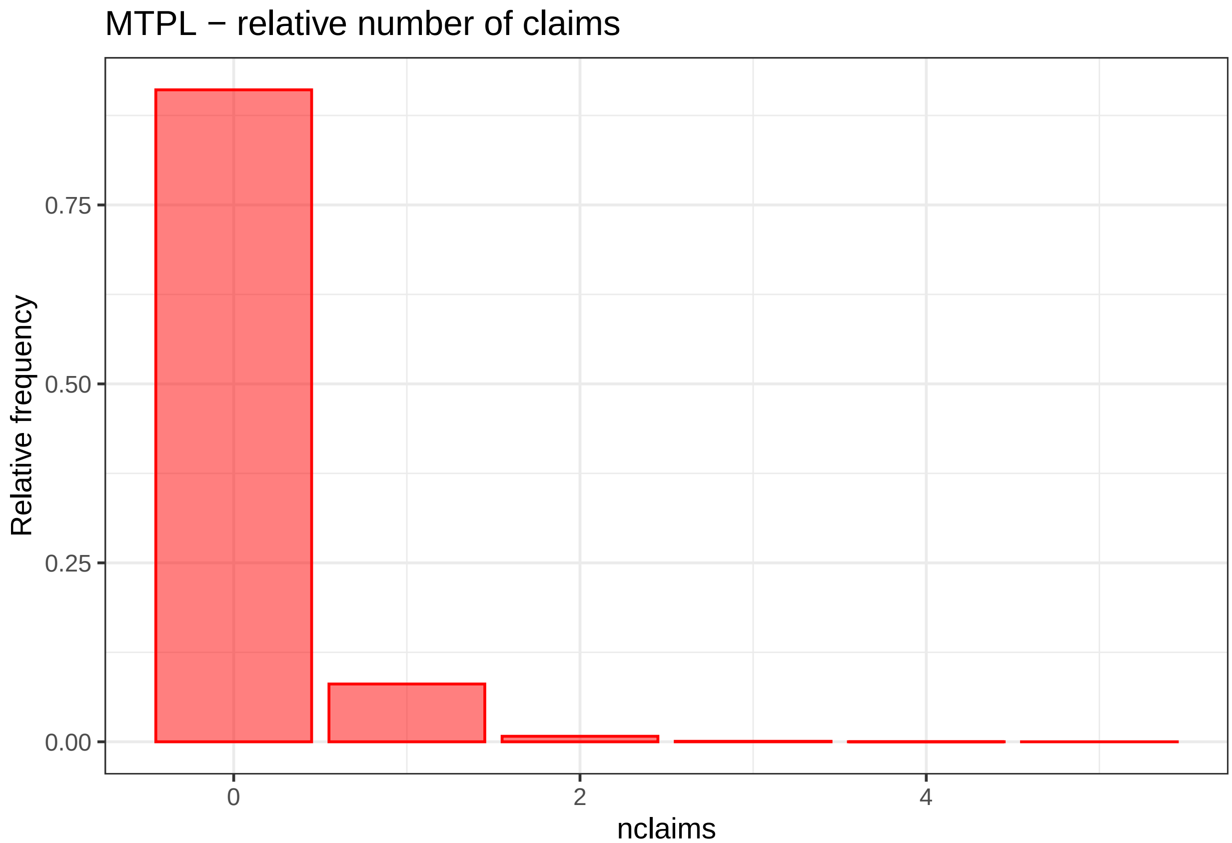
age	sex	experience	expo	nclaims	vage	ncd_0	cover_c
32	M	11	0.4654346	0	10	0	1
26	M	5	0.8076660	0	13	1	0
32	M	5	0.3997262	0	0	0	1
32	M	5	0.5831622	0	1	0	1
41	M	14	0.7748118	0	9	0	0
28	F	3	0.4928131	0	0	1	1

2 Empirical analysis

To have a better understanding of our data we can make a graph with number of claims per insurance contract. We can observe that in the majority of cases there is zero claims per contract.



Looking at this in relative terms, we can see that more than 90% of insurance contract did not have any claims.



Continuing our analysis, we can compute the mean and variance of the number of observed claims.

```
##       mean   variance
## 0.09848262 0.10925308
```

Because we have different exposures between policyholders, we could take exposure information `expo` into account as well. Then we can calculate empirical claim frequency, per unit of exposure and variance.

```
##      mean  variance
## 0.1545612 0.1675359
```

If we do the same for each gender we see that claim frequency is higher for males than females.

```
## # A tibble: 2 x 2
##   sex   emp_freq
##   <chr>    <dbl>
## 1 F        0.150
## 2 M        0.155
```

3 Fitting different distributions to the loss frequency data

We will fit different distribution to the observed claim count data. We will take exposure into account as well and will fit the distribution using Maximum Likelihood Estimation (MLE).

3.1 Poisson distribution

3.1.1 Numerical calculation

We start by fitting the Poisson distribution to the observed loss frequency data.

Since, not all policyholders are insured throughout the whole year (exposure = period of exposure during which the contract was active, is not equal to 1 for all the contracts), we assume that the claim intensity is proportional to the exposure. For Poisson distribution this means that the intensity is equal to $\lambda * expo$. For `expo=1` (contract active for a whole year), the expected number of claims equals to λ. For all other contracts, the expected number of claims equals $\lambda * expo$. We denote this with:

$$N_i \sim Poiss(\lambda * expo)$$

Let denote that m is the number of observations and n_i is the observed number of claims for `i`-th policyholder, then general definition of likelihood is given by:

$$L(\lambda) = \prod_{i=1}^{m} P(N_i = n_i) = \prod_{i=1}^{m} exp(-\lambda \cdot expo) \cdot \frac{(\lambda \cdot expo)^{n_i}}{n_i!}$$

$$l(\lambda) = \sum_{i=1}^{m} -\lambda \cdot expo + n_i \cdot log(\lambda \cdot expo) - log(n_i!)$$

We can maximize loglikelihood with respect to λ.

In practice, we minimize the negative log-likelihood so that we can use `nlm` (non-linear minimizer) function for finding the minimum.Minimizing the negative log-likelihood is equal to maximizing the log-likelihood.

$$l(\lambda) = \sum_{i=1}^{m} \lambda \cdot expo - n_i \cdot log(\lambda \cdot expo) + log(n_i!)$$

Since the parameter λ is strickly positive, we will reparametrize the likelihood and optimize for $\beta = log(\lambda)$ which can take values $(-\infty, \infty)$

Now we will use non-linear minimization function to get the minimum value. The starting value will be set to $\beta = 1$. The function returns the following lines:

- minimum: the value of estimated minimum of our function
- estimate: the point at which that minimum is obtained.
- gradient: first derivative at the estimated minimum
- hessian: second derivative at the estimated minimum

```
## $minimum
## [1] 50834.16
##
## $estimate
## [1] -1.867165
##
## $gradient
## [1] 0.007053196
##
## $hessian
##          [,1]
## [1,] 15753.57
##
## $code
## [1] 1
##
## $iterations
## [1] 11
```

From this we can get the estimation of λ which is identical to the expected value (empirical claim frequency per unit) that we have calculated before.

```
## [1] 0.1545612
```

In general, the covariance matrix of the maximum likelihood estimators can be estimated by the inverse of this hessian of minus the log-likelihood at the estimated minimum. By taking the square root of these diagonal elements, we obtain the corresponding standard error that equals:

```
## [1] 0.007967287
```

3.1.2 Akaike Information Criterion

The Akaike information criterion (AIC) is an estimator of prediction error and provides relative quality of statistical models for a given set of data. AIC estimates the quality of each model which we can relative compare to each of the other models.The preferred model is the one with the minimum AIC value.

- Calculation of Akaike Information Criterion (AIC): Let k be the number of estimated parameters in the model and let L be the maximum value of the likelihood function for the model. Then the AIC value of the model is:

  $$AIC = 2k - 2ln(L)$$

  For Poisson distribution AIC equals to:

```
## [1] 101670.3
```

3.1.3 Calculation with the help of Generalized Linear Models (GLMs)

We will now vertify our solution in the previous section for numerically computed log-likelihood with the `glm` function. We will put focus on the Poisson regression model. Under Poisson assumption in this model, the number of claims is distributed as follows:

$$N \sim Poiss(\mu),$$

$$\mu = expo \cdot exp(x^{`}\beta).$$

For `glm` function, we will take exposure into account as well. From the box below we can see the `glm` function that we used and the return that it gives. We get that the standrad error equals to 0,0079, AIC equals to 101.670, lambda estimate is -1,86. So we can conclude that both methods give the same results.

```
##
## Call:
## glm(formula = nclaims ~ 1, family = poisson(link = "log"), data = mtpl,
##     offset = log(expo))
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.5560  -0.5558  -0.4226  -0.2698   5.3747
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.867165   0.007968  -234.3   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 72260  on 159946  degrees of freedom
## Residual deviance: 72260  on 159946  degrees of freedom
```

```
## AIC: 101670
##
## Number of Fisher Scoring iterations: 6
```

3.2 Negative Binomial distirbution

3.2.1 Numerical calculation

The probability function for the negative binomal distribution is as follows:

$$P(N=k)=\frac{\Gamma(a+k)}{\Gamma(a)k!}(\frac{\mu}{\mu+a})^k(\frac{a}{\mu+a})^a$$

where μ is the expected number of claims for a policy holder who is insured for a full year. So for each policy holder we can define $\mu_i = expo \cdot \mu$.

The likelihood and loglikelihood function is:

$$L(\lambda)=\prod_{i=1}^{m} P(N_i=n_i)$$

$$l(\lambda)=\sum_{i=1}^{m} log(\Gamma(a+k)) - log(\Gamma(a)k!) + k \cdot log(\frac{\mu}{\mu+a}) + a \cdot log(\frac{a}{\mu+a}).$$

We know that $\mu = E(X)$ and $a = \frac{\mu^2}{Var(X)-\mu}$.

With the `nlm` function we get the following:

```
## $minimum
## [1] 50657.21
##
## $estimate
## [1] -1.8666036  0.3823959
##
## $gradient
## [1]  1.301921e-03 -1.455192e-05
##
## $hessian
##              [,1]       [,2]
## [1,] 14517.903037   7.601193
## [2,]     7.601193 233.583705
##
## $code
## [1] 1
##
## $iterations
## [1] 5
```

and we can calculate the corresponding standard error for μ and a which is equal to:

```
## [1] 0.008299497 0.065430830
```

- Akaike Information Criterion

For Negative Binomial distribution AIC equals to:

```
## [1] 101318.4
```

3.2.2 Calculation with the help of Generalized Linear Models (GLMs)

In the box below we can see what `glm` function returns for Negative Binomial distribution:

```
##
## Call:
## glm.nb(formula = nclaims ~ 1 + offset(log(expo)), link = log,
##     init.theta = 1.465793094)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.5423  -0.5421  -0.4165  -0.2682   4.5135
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.866603   0.008298    -225   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for Negative Binomial(1.4658) family taken to be 1)
##
##     Null deviance: 63325  on 159946  degrees of freedom
## Residual deviance: 63325  on 159946  degrees of freedom
## AIC: 101318
##
## Number of Fisher Scoring iterations: 1
##
##
##               Theta:  1.4658
##           Std. Err.:  0.0959
##
##  2 x log-likelihood:  -101314.4280
```

We can see that we get the same values of the estimate, and minimum.

We can also calculate correspoding standard deviation, which equals to 0.0082976. Also the value of AIC stays the same. From this we can conclude, that both methods return the same results.

3.3 Zero Inflated Poisson distirbution

3.3.1 Numerical calculation

This is a Poisson distribution where the probability of having zero claims is increased by p.

$$P(N^{ZI} = k) = \begin{cases} p + (1-p) \cdot P(N=0); k = 0, \\ (1-p) \cdot P(N=k); k > 0 \end{cases}$$

where N represents Poisson distribution. The parameter p takes valued in $[0, 1]$, so we can transform in to the real line $(-\infty, \infty)$ with logarithm:

$$logit(p) = log(\frac{p}{1-p}) = \beta$$

$$p = \frac{exp(\beta)}{1 + exp(\beta)}$$

```
## $minimum
## [1] 50663.24
```

```
##
## $estimate
## [1] -1.3754366 -0.4551083
##
## $gradient
## [1] -0.02995685  0.02011802
##
## $hessian
##           [,1]      [,2]
## [1,] 14606.513 -5622.753
## [2,] -5622.753  2422.831
##
## $code
## [1] 1
##
## $iterations
## [1] 13
```

With `nlm` function we can compute the estimated values for λ and p.

```
##    lambda         p
## 0.2527292 0.3881469
```

The corresponding standard error under Zero Inflated distribution for both parameters equals to:

```
## [1] 0.02533814 0.06221376
```

- Akaike Information Criterion

For Zero Inflated Poisson distribution AIC equals to:

```
## [1] 101330.5
```

3.3.2 Calculation with the help of Generalized Linear Models (GLMs)

```
##
## Call:
## zeroinfl(formula = nclaims ~ 1, offset = log(expo), dist = "poisson")
##
## Pearson residuals:
##     Min      1Q  Median      3Q     Max
## -0.3753 -0.3751 -0.2908 -0.1886 18.8920
##
## Count model coefficients (poisson with log link):
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.37544    0.02535  -54.26   <2e-16 ***
##
## Zero-inflation model coefficients (binomial with logit link):
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -0.45512    0.06225  -7.312 2.64e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Number of iterations in BFGS optimization: 14
## Log-likelihood: -5.066e+04 on 2 Df
```

We can see that the `glm` function returns the same values of the estimate, and minimum.

We can also calculate correspoding standard deviation, which equals to 0.0253508, 0.0622471. The value of AIC also stays the same.

so we can conclude, that both methods return the same results.

3.4 Hurdle Poisson distirbution

3.4.1 Numerical calculation

In Hurlde Poisson the probability of observing zero claims is set to p. The probability of observing k claims equals:

$$P(N^H = k) = \begin{cases} p; k = 0, \\ (1-p) \cdot \frac{(N=k)}{1-P(N=0)}; k > 0 \end{cases}$$

Here the probability p does not depend on the exposure, while intensity λ is proportional to exposure as: $\lambda_i = expo \cdot \lambda$.

```
## $minimum
## [1] 52955.24
##
## $estimate
## [1] -1.381171  2.323694
##
## $gradient
## [1] 4.741167e-05 1.087780e-02
##
## $hessian
##               [,1]          [,2]
## [1,]  1.540939e+03 -3.131203e-04
## [2,] -3.131203e-04  1.298946e+04
##
## $code
## [1] 1
##
## $iterations
## [1] 12
```

With `nlm` function we can compute the estimated values for λ and p.

```
##    lambda         p
## 0.2512842 0.9108204
```

The corresponding standard error for both parameters equals:

```
## [1] 0.025474597 0.008774137
```

- Akaike Information Criterion

For Hurdle Poisson distribution AIC equals to:

```
## [1] 105914.5
```

3.4.2 Calculation with the help of Generalized Linear Models (GLMs)

```
##
## Call:
## hurdle(formula = nclaims ~ 1, offset = (log(expo)), dist = "poisson",
##     zero.dist = c("binomial"))
##
```

```
## Pearson residuals:
##     Min      1Q  Median      3Q     Max
## -0.3128 -0.3062 -0.2997 -0.2961 15.2003
##
## Count model coefficients (truncated poisson with log link):
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.38117    0.02548  -54.22   <2e-16 ***
## Zero hurdle model coefficients (binomial with logit link):
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -2.323694   0.008773  -264.9   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Number of iterations in BFGS optimization: 8
## Log-likelihood: -5.296e+04 on 2 Df
```

We can see that we get the same values of the estimate, and minimum.

We can also calculate correspoding standard deviation, which equals to 0.0254759, 0.0087733. The value of AIC also stays the same.

We can conclude, that both methods return the same results.

Now that we have calculated AIC for all distributions we can compare the values. The best AIC value is the lowest one.

```
##    AIC POI     AIC NB    AIC ZIP AIC HURDLE
##     101670     101318     101330     105914
```

We can observe that the lowest AIC value is achieved with the Negative Binomial distribution and equals to 101.318, followed by Zero Inflated Poisson. We can also see that Hurdle distribution has by far the worst AIC.

4 Comaring frequency models

We will now compare the frequency models by comparing the expected number of zeros with the actually observed number of zero claims. First we will calculate the actual number of zero claims per contracts in our dataset.

We compute that there is 145683 insurance contracts that did not have any claims, out of 159947 contracts.

- Poisson distribution

The predicted number of zero claims per contract under Poisson distribution is equal to:

```
## [1] 145141
```

This means that our model returns pretty accurate number of expected zeros. The difference is 542 which represents only 0.37 % deviation.

- Negative Binomial distribution The predicted number of zero claims per contract under Negative Binomial distribution is equal to:

```
## [1] 145690
```

This means that our model returns very accurate number of expected zeros. The difference is only 7 zeros.

- Zero Inflated Poisson distribution

The predicted number of zero claims per contract under ZIP is equal to:

```
## [1] 145692
```

This means that our model returns very accurate number of expected zeros. The difference is only 9 zeros.

- Hurdle distribution

The predicted number of zero claims per contract under Hurdle distribution is equal to:

```
## [1] 145692
```

Which means that our model gives us the same number of expected zeros as there are actual zeros in our dataset.

From accuracy prespective the Hurdle distribution estimates the number of zeros the most accurately, while Poisson distribution returns the biggest (yet very small) deviation.

PART 2

1 Fitting exponential function

First we need to import our data from the file `SeverityCensoring` into R. The file contains 9062 claims, with columns representing policy id, claim id, rc (iz claim is censored - number or uncensored - value NA), deductible (always 100) and the claim amount.

	policyId	claimId	rc	deductible	claimAmount
3	600003	900003	41639	100	41639
50	600045	900050	NA	100	16965
51	600045	900051	NA	100	363
58	600052	900058	5949	100	5949
77	600071	900077	NA	100	148

Before we start fitting exponential distribution to our data we will modify our rc column, so that it is a logical colum (indicator), where TRUE means the claim is censores and FALSE means that the claim is not censored. This is how we do this:

```
Severity$rc <- !is.na(Severity$rc)
```

Now we can finally fit exponential distribution to our model.We will compute negative log-likelihood function, taking trucation and censoring into account. We will do this with the help of `dexp` and `pexp` function that represent density and distribution functions of exponential distribution. As our paramether $\lambda > 0$ we use $\lambda = e^{\beta}$ and now it can take all values.

```
claimAmount <- Severity$claimAmount
rc <- Severity$rc
deductible <- Severity$deductible

exp.negloglikelihood <- function(par)
  {lambda <- exp(par)
  -sum(dexp(claimAmount[!rc],lambda,log=T)) - #here we take values of claimAmount if rc false
    sum(pexp(claimAmount[rc],lambda,log.p=T,lower.tail=F)) + #claimamount if rc true
    sum(pexp(deductible,lambda,log.p=T,lower.tail=F)) #adds deductible
}
```

To get the initial estimation of our parameter, we use the moment estimation. As we know

$$E(X) = \frac{1}{\lambda}$$

and from here we get that

$$\lambda = \frac{1}{m_1}.$$

Where m_1 is first moment derived from our data. Now we can fit exponential distribution and we get the MLE estimation of λ:

```
## [1] 0.0004694256
```

The value of AIC is equal to:

```
## [1] 127986.6
```

2 Fitting other distributions

Now for the second pard we need to also fit three other distributions to our data: lognormal, inverse Gaussian and Burr distribution

2.1 Fitting lognormal distribution

We repeat our process from above. But we need different inital estimations for our two parameters μ and σ^2. We need to use the first and the second moments:

$$E(X) = e^{\mu + \frac{\sigma^2}{2}}$$

$$E(X^2) = e^{2\mu + 2\sigma^2}$$

Now we get first and the second moment from our data and from that our estimations. We fit the model and get our estimates for μ and σ^2:

```
## [1] 6.154313 1.955911
```

And AIC for this fit is:

```
## [1] 121207.2
```

2.2 Fitting inverse Gaussian distribution

Here our first two moments, that are used for the initial estimation, are equal to:

$$E(X) = \mu$$

$$E(X^2) = \mu^2 + \frac{\mu^3}{\theta}$$

Whit this we get our estiamtions for μ and θ:

```
## [1] 5512.4128  427.6825
```

AIC for inverse Gaussian fit is:

```
## [1] 120651.7
```

2.3 Fitting Burr distribution

Here we have three paramethers, so we need first three moments, for initial estimation.

$$E(X) = \frac{\theta * \Gamma(1 + \frac{1}{\gamma}) * \Gamma(\alpha - \frac{1}{\gamma})}{\Gamma(\alpha)}$$

$$E(X^2) = \frac{\theta^2 * \Gamma(1 + \frac{2}{\gamma}) * \Gamma(\alpha - \frac{2}{\gamma})}{\Gamma(\alpha)}$$

$$E(X^3) = \frac{\theta^3 * \Gamma(1 + \frac{3}{\gamma}) * \Gamma(\alpha - \frac{3}{\gamma})}{\Gamma(\alpha)}$$

Whit this we get our estiamtions for α, θ and γ:

```
## [1] 0.163914852 3.631583521 0.004719882
```

Corresponding AIC is:

```
## [1] 119870.7
```

3 Fitting Erlang mixture distribution

Here we use the felp of EM algorithm for Erlang mixtures which we source from file `EM_MixedErlang`. We need to define loss and nrc vector so that they meet the conditions form the insutructions. This is how we did this:

```
loss <- claimAmount ;
nrc <- claimAmount
for (i in 1:9062) {
  if (rc[i] == TRUE){
    nrc[i] <- NA
  }
}
```

Here are the MLE estimations for Erlang mixture distribution:

```
## M =  5 ,  AIC  =  119843.4 , shape =  1 6 14 27 96
##  theta =  612.2609 , alpha =  0.8228129 0.04624449 0.01072555 0.02000244 0.1002147

## $alpha
## [1] 0.82281285 0.04624449 0.01072555 0.02000244 0.10021467
##
## $beta
## [1] 0.79773427 0.05278980 0.01224362 0.02283353 0.11439878
##
## $shape
## [1]  1  6 14 27 96
##
## $theta
## [1] 612.2609
##
## $loglikelihood
## [1] -59911.69
##
## $AIC
## [1] 119843.4
##
## $BIC
## [1] 119914.5
##
## $M
## [1] 5
##
## $M_initial
## [1] 5
##
```

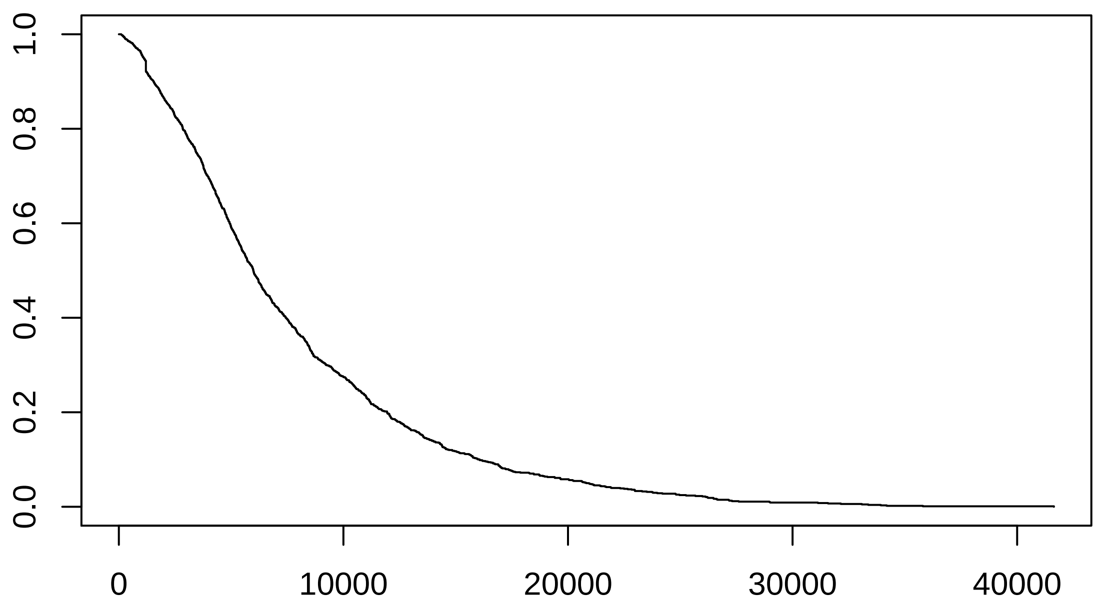
```
## $s
## [1] 3
```

4 Plotting the Kaplan-Meier estimate of the survival function

Here we need to define three vectors: deds, loss and full and then plot the fit. We do this like this:

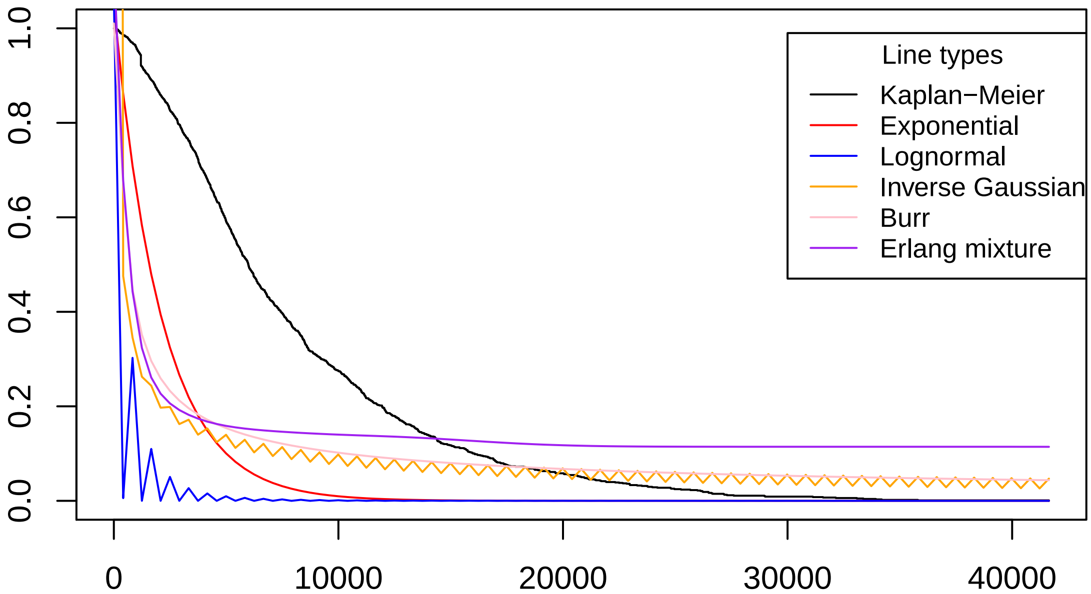
```
deds <- deductible ; loss <- claimAmount ;
full <- rc
for (i in 1:9062) {
  if (rc[i] == TRUE){
    full[i] == FALSE
  }else{full[i] == TRUE}
}


fit <- survfit(Surv(deds, loss, full) ~ 1)
plot(fit, mark.time=F, conf.int=F)
```



5 Plots

Now we will plot all the survival functions for all the different distributions onto one graph, so that we can compare them. The graph looks like this:



The model that seems closest to Kaplan-Meier estimate is the Burr model in our opinion.

6 Comparison of AIC values

The Akaike information criterion (AIC) is an estimator of prediction error and thereby relative quality of statistical models for a given set of data. Given a collection of models for the data, AIC estimates the quality of each model, relative to each of the other models. Thus, AIC provides a means for model selection. Given a set of candidate models for the data, the preferred model is the one with the minimum AIC value. So lets again write down AIC values for our models:

```
##                       AIC
## Erlang mixture   119843.4
## Burr             119870.7
## Inverse Gaussian 120651.7
## Lognormal        121207.2
## Exponential      127986.6
```

As we can see that the best choice of model is the one with Exlang mixture distribution. The most inappropriate among this models is the one with Exponential distribution, according to the AIC parameter. In exercise 5 we thought that the Burr distribution fits best to the Kaplan-Meier estimate, but AIC parameter tells us that Exlang mixture model is the best, so our anwser is not consistent.