THE UNIVERSITY OF MELBOURNE
Centre for Actuarial Studies. Department of Economics
ACTL30004 Actuarial Statistics

Assignment, COVER SHEET

Due by 12:00 PM on Friday 25 October 2019. Submission via LMS.

This assignment contributes 10% of the total university assessment of this subject.

Please attach this cover sheet on top of your answers to your submission. Include name and student ID number of all the students in the group. Write also your group number.

Declaration by Group
We declare that this assignment is our own work and does not involve plagiarism or collusion. We understand that penalties will be imposed if the instructions accompanying the assignment are not followed.

Student Number	Name in full	Signature	Group Number
913696	Valentina Cardinali	Valentina Cardinali	7
929715	Kelvin Lim Wan	Kelvin Lim	7
909762	Shervin Nastili	Shervin Nastili	7

Plagiarism and Collusion
Plagiarism is the presentation by a student of an assignment which has been copied in whole or in part from another students work, or from any other source without due acknowledgement.

Collusion is the presentation by a student of an assignment as his or her own which is the result, in whole or in part, of unauthorised collaboration, with another person(s). Allowing your work to be seen or used by other students outside of your group is also collusion, as is any form of discussion, before submission, with any other student outside your group. A student who assists another student outside their group in any way is also colluding.

(a) Let Y be the number of claims made in years 2004 and 2005.

For the Poisson distribution $Y \sim Pn\,(\lambda)$, we use the maxLik function in R to find the MLE of λ. The starting value is calcuted using MM: $\lambda_0 = \frac{1}{67856}\sum_{i=1}^{67856} y_i$. We get that $\hat{\lambda} = 0.072757$ with a standard error of 0.001035. Hence

$$Y \sim Pn\,(0.07276)$$

For the Negative Binomial distribution $Y \sim NB\,(r, \beta)$, we create a new function for the density function since the corresponding built-in function in R uses parameter p instead of β. We find the starting values by solving for the parameters using MM: $r_0\beta_0 = \frac{1}{67856}\sum_{i=1}^{67856} y_i$ and $r_0\beta_0(1+\beta_0) = \frac{1}{67856-1}\sum_{i=1}^{67856}(y_i - \bar{y})^2$. Using the maxLik function in R, we get that the MLE's are $\hat{r} = 1.157022$ and $\hat{\beta} = 0.062883$ with corresponding standard errors of 0.142181 and 0.007781. Thus,

$$Y \sim NB\,(1.157, 0.06288)$$

For the Zero-inflated Poisson distribution $Y \sim ZiPn\,(\theta, \psi)$, we need to use the multiroot function in R to find the MME's for the starting values: $\theta_0(1-\psi_0) = \frac{1}{67856}\sum_{i=1}^{67856} y_i$ and $\theta_0(1-\psi_0)(1+\psi_0\theta_0) = \frac{1}{67856-1}\sum_{i=1}^{67856}(y_i - \bar{y})^2$. We use the maxLik function in R with starting values θ_0 and ψ_0 to compute MLE's. This gives $\hat{\theta} = 0.132457$ and $\hat{\psi} = 0.450714$ and standard errors of 0.007524 and 0.030259 respectively. We get

$$Y \sim ZiPn\,(0.1325, 0.4507)$$

We select the best fitting model based on three model selection criteria:

Likelihood Ratio Test: Since the Poisson model is nested in both the Negative Binomial and Zero-inflated Poisson models, we can test H_0: Smaller model (Poisson) is a better fit vs H_1: Larger model (Negative Binomial or Zero-inflated Poisson) is a better fit. The maximum log-likelihood for each distributions are extracted from the MLE functions in R and are tabulated in Table 1. Then we compute the test statistics given as

$$2\,[max(\ell_{larger}) - max(\ell_{smaller})]$$

At a conventional 5% significance level, we reject H_0 if the test statistic is larger than $\chi^2_{k,0.05}$, where k is the difference between the dimensions of the two models. In both of our tests, $k = 1$, then $\chi^2_{1,0.05} = 3.841$.
The test statistic column in Table 1 shows the test statistics (1) 103.6395 and (2) 98.6043 corresponding to the tests (1) Poisson model vs Negative Binomial model and (2) Poisson model vs Zero-inflated Poisson model.
We cannot use the LRT to test the Negative Binomial model vs the Zero-inflated Poisson model since they are non-nested.

Akaike's Information Criterion: We calculate the AIC value for each model using

$$AIC = -2\,max(\ell) + 2p$$

where p is the number of parameters in the model, and tabulate it in Table 1. The model with the lowest AIC value is the model with the best fit.

Bayesian Information Criterion: We calculate the BIC value for each model using

$$BIC = -2\,max(\ell) + p\,log(67856)$$

which is then tabulated in Table 1. The model with the lowest BIC value is the model with the best fit.

Distribution	Maximum log-likelihood	Test statistic	AIC	BIC
Poisson	-18101.5	-	36205	36214.13
Negative Binomial	-18049.68	103.6395	36103.36	36121.61
Zero-inflated Poisson	-18052.2	98.6043	36108.4	36126.65

Table 1: Model selection criteria

For the LRT, the test statistic 103.6295 is larger than $\chi^2_{1,0.05} = 3.841$, hence Negative Binomial model is a better fit than the Poisson model and the test statistic 98.6043 is larger than $\chi^2_{1,0.05} = 3.841$, hence Zero-inflated model is a better fit than the Poisson model.
For the AIC, the Negative Binomial model has the lowest AIC value (36103.36), hence the Negative Binomial model is the best fit.
For the BIC, the Negative Binomial model has the lowest BIC value (36121.61), hence the Negative Binomial model is the best fit.

We conclude that the Negative Binomial model is the best fitting model.

(b) To create indicator variables for vehAge and drivAge, we use the following R codes:

```
vehage <- as.integer(ausprivauto0405$VehAge %in% c("old cars","oldest cars"))
drivage <- as.integer(ausprivauto0405$DrivAge %in% c("old people","older work
. people", "oldest people"))
```

Since the number of claims only take positive values, we should use a function of the mean (link function) that takes values only on the positive real line. Hence, we use the logarithmic link function to fit the Poisson GLM.

Let

$$Y_i \sim Pn\left(\lambda_i = exp\left(a + bz_{i1} + cz_{i2} + dz_{i3} + log(E_i)\right)\right)$$

since $\lambda_i = \mu_i$ and where z_{i1} is the explanatory variable vehValue, z_{i2} is the indicator variable vehAge, z_{i3} is the indicator variable drivAge and E_i is the exposure variable.

We fit a Poisson Linear Model to the data to get the suitable starting values a_0, b_0, c_0 and d_0

```
poislm <- lm(claimnb ~ vehvalue + vehage + drivage, offset=log(exposure))
```

So $a_0 = 1.250998$, $b_0 = -0.008722$, $c_0 = -0.074547$ and $d_0 = -0.052772$.

We use these as starting values and fit the Poisson GLM using

```
poisglm <- glm(claimnb ~ vehvalue + vehage + drivage, offset=log(exposure),
family=poisson(link=log), start=coef(poislm))
summary(poisglm)
```

The output gives

Parameters	a	b	c	d
Estimate	-1.76639	0.03049	-0.10558	-0.19378
Standard error	0.04030	0.01237	0.03211	0.02865
Test statistic	-43.834	2.464	-3.288	-6.763
P-value	<2e-16	0.01375	0.00101	1.35e-11

Table 2: Parameter estimates

Since the p-values for all the parameters are close to 0, we deduce that the effects: vehicle value, age of vehicle and age of driver are all statistically significant and should be included in the linear predictor. Also, the vehValue regressor a is positive, denoting that with increasing vehicle value, the number of claims is higher (because of higher repair cost). The parameter estimates c and d are negative, indicating a negative relationship between both age of vehicle and age of driver, with number of claims. The latter can be explained by the perception that older drivers are safer drivers. But decreasing number of claims with vehicle age is contrary to what we would expect.

Furthermore, if the Poisson model is a good fit, the deviance should be approximately distributed chi-squaredly with degrees of freedom of (dimension of the saturated model – dimension of the poisson model). Since the value that we get for deviance is significantly different to the degree of freedom of our model (expectation of chi-squared distribution = degrees of freedom), we see that deviance is not chi-squared distributed. This along with last point in the last paragraph suggests that the Poisson GLM is not appropriate for this data.

Hence, the Poisson GLM is in the form

$$Y_i \sim Pn\left(\lambda_i = exp\left(-1.766 + 0.03049 z_{i1} - 0.1056 z_{i2} - 0.1938 z_{i3} + log(E_i)\right)\right)$$

(c) For the negative binomial distribution, we have that $Y \sim NB\,(r, \beta)$ where $r, \beta > 0$. Its probability mass function is

$$\begin{aligned} f(y \mid r, \beta\,) &= \binom{y+r-1}{y}\left(\frac{1}{1+\beta}\right)^r\left(\frac{\beta}{1+\beta}\right)^y \\ &= exp\left[log\binom{y+r-1}{y} - r\,log\big(1+\beta\big) + y\,log\left(\frac{\beta}{1+\beta}\right)\right] \\ &= exp\left[\frac{y\,log\big(\frac{\beta}{1+\beta}\big) - r\,log\big(1+\beta\big)}{1} - \left(-log\binom{y+r-1}{y}\right)\right] \end{aligned}$$

This is in the form of the exponential family where

$$\theta = log\left(\frac{\beta}{1+\beta}\right) \Rightarrow \beta = \frac{e^\theta}{1-e^\theta} \quad , \qquad \phi = 1 \quad , \qquad w = 1 \quad ,$$

$$b(\theta) = r\,log(1+\beta) = -r\,log(1-e^\theta) \quad , \quad c(y,\phi) = -log\binom{y+r-1}{y}$$

Through some simple calculus, we get that $b'(\theta) = \frac{r\,e^\theta}{1-e^\theta}$ and $b''(\theta) = \frac{r\,e^\theta}{(1-e^\theta)^2}$

We know that $Var[Y] = \frac{\phi}{w}\,b''(\theta)$, therefore

$$Var[Y] = \frac{1}{1}\frac{r\,e^\theta}{(1-e^\theta)^2} = r\,\frac{e^\theta}{1-e^\theta}\,\frac{1}{1-e^\theta} = r\,\beta\,(1-\beta)$$

(d) Under the new parametrisation $\mu = E[Y] = r\beta$, it follows that $\beta = \frac{\mu}{r}$. Hence, we can rewrite the probability mass function as

$$\begin{aligned} f(y \mid r, \mu\,) &= \binom{y+r-1}{y}\left(\frac{1}{1+\frac{\mu}{r}}\right)^r\left(\frac{\frac{\mu}{r}}{1+\frac{\mu}{r}}\right)^y = \binom{y+r-1}{y}\left(\frac{r}{r+\mu}\right)^r\left(\frac{\mu}{r+\mu}\right)^y \\ &= exp\left[log\binom{y+r-1}{y} + r\,log\left(\frac{r}{r+\mu}\right) + y\,log\left(\frac{\mu}{r+\mu}\right)\right] \\ &= exp\left[\frac{y\,log\big(\frac{\mu}{r+\mu}\big) - \big(-r\,log\big(\frac{r}{r+\mu}\big)\big)}{1} - \left(-log\binom{y+r-1}{y}\right)\right] \end{aligned}$$

This is in the form of the exponential family where

$$\theta = log\left(\frac{\mu}{r+\mu}\right) \Rightarrow \mu = r\frac{e^{\theta}}{1-e^{\theta}} \quad , \qquad \phi = 1 \quad , \qquad w = 1 \quad ,$$
$$b(\theta) = -r\, log\left(\frac{r}{r+\mu}\right) = -r\, log(1-e^{\theta}) \quad , \quad c(y,\phi) = -log\binom{y+r-1}{y}$$

Also note: $V(\mu_i) = b''(\theta) = \frac{r\, e_i^{\theta}}{(1-e_i^{\theta})^2} = \frac{\mu_i(r+\mu_i)}{r}$ and $h(\eta_i) = \mu_i = e^{\eta_i} \Rightarrow h'(\eta_i) = e^{\eta_i} = \mu_i$.

(e) The Negative Binomial GLM with logarithmic link function is in the form

$$Y_i \sim NB\ (r,\ \mu_i = exp(a + bz_{i1} + cz_{i2} + dz_{i3} + log(E_i))$$

We set the initial values to be $r_0 = \hat{r}$ from (a) and take the parameter estimates of a, b, c, d from (b) for $\beta_0 = [a_0,\ b_0,\ c_0,\ d_0]^T$

The parameter estimation consists of two steps for each iteration:

(1) Fix parameter r_0 and estimate $\beta_1 = [a_1,\ b_1,\ c_1,\ d_1]^T$

For $j = 1, ..., 4$, the entries in the score vector (U_{4x1}) are given such that

$$U_j = \sum_{i=1}^{67856} \frac{(y_i - \mu_i) w_i\, h'(\eta_i)\, z_{ij}}{\phi\, V(\mu_i)} = \sum_{i=1}^{67856} \frac{r\,(y_i - \mu_i)\, z_{ij}}{r + \mu_i}$$

For $j, k = 1, ..., 4$, the entries in the Fisher information matrix (I_{4x4}) are given such that

$$I_{jk} = \sum_{i=1}^{67856} \frac{w_i\, h'(\eta_i)^2 z_{ij}\, z_{ik}}{\phi\, V(\mu_i)} = \sum_{i=1}^{67856} \frac{r\, \mu_i\, z_{ij} z_{ik}}{r + \mu_i}$$

β_1 is then estimated using some functions in R based on the Fisher-Scoring Algorithm given by

$$\beta_1 = \beta_0 + I(\beta_0)^{-1}\, U(\beta_0)$$

(2) Fix vector β_1 and estimate r_1

In R, we create a re-parametrised function (in terms of r and μ) for the Negative Binomial density function and use the maxLik function to generate a MLE for r_1.

Note: We use $\mu_i = exp(a_1 + b_1 z_{i1} + c_1 z_{i2} + d_1 z_{i3} + log(E_i))$.

Finally, we create a loop in R containing these recurring 2 steps, which stops iterating once the absolute value of each component of the score vector is smaller than 10^{-10} (Refer to Appendix B for this question's code). The output is given in the table below.

The maximum log-likelihood value is -17410.59.

We then get

$$Y_i \sim NB\ (2.147,\ \mu_i = exp(-1.765 + 0.03077 z_{i1} - 0.1040 z_{i2} - 0.1947 z_{i3} + log(E_i))$$

and the variance-covariance matrix associated to the estimates as

$$\begin{bmatrix} 0.001711 & -0.0004157 & -0.0009605 & -0.0004419 \\ -0.0004157 & 0.0001621 & 0.0001911 & 0.00002170 \\ -0.0009605 & 0.0001911 & 0.001081 & 0.00002534 \\ -0.0004419 & 0.00002170 & 0.00002534 & 0.0008575 \end{bmatrix}$$

MLE	SE	Iteration 1		Iteration 2		Iteration 3		Iteration 4	
$\hat{r}$	$se(\hat{r})$	2.140	0.5243	2.1470	0.3027	2.147	0.3707	2.147	0.3707
$\hat{a}$	$se(\hat{a})$	-1.764	0.04232	-1.765	0.04134	-1.765	0.04137	-1.765	0.04137
$\hat{b}$	$se(\hat{b})$	0.03093	0.01305	0.03078	0.01272	0.03077	0.01273	0.03077	0.01273
$\hat{c}$	$se(\hat{c})$	-0.1028	0.03356	-0.1040	0.03285	-0.1040	0.03287	-0.1040	0.03287
$\hat{d}$	$se(\hat{d})$	-0.1953	0.02985	-0.1947	0.02926	-0.1947	0.02928	-0.1947	0.02928
MLE	**SE**	**Iteration 5**		**Iteration 6**		**Iteration 7**		**Iteration 8**	
$\hat{r}$	$se(\hat{r})$	2.147	0.3707	2.147	0.3707	2.147	0.3707	2.147	0.3707
$\hat{a}$	$se(\hat{a})$	-1.765	0.04137	-1.765	0.04137	-1.765	0.04137	-1.765	0.04137
$\hat{b}$	$se(\hat{b})$	0.03077	0.01273	0.03077	0.01273	0.03077	0.01273	0.03077	0.01273
$\hat{c}$	$se(\hat{c})$	-0.1040	0.03287	-0.1040	0.03287	-0.1040	0.03287	-0.1040	0.03287
$\hat{d}$	$se(\hat{d})$	-0.1947	0.02928	-0.1947	0.02928	-0.1947	0.02928	-0.1947	0.02928
MLE	**SE**	**Iteration 9**		**Iteration 10**		**Iteration 11**		**Iteration 12**	
$\hat{r}$	$se(\hat{r})$	2.147	0.3707	2.147	0.3707	2.147	0.3707	2.147	0.3707
$\hat{a}$	$se(\hat{a})$	-1.765	0.04137	-1.765	0.04137	-1.765	0.04137	-1.765	0.04137
$\hat{b}$	$se(\hat{b})$	0.03077	0.01273	0.03077	0.01273	0.03077	0.01273	0.03077	0.01273
$\hat{c}$	$se(\hat{c})$	-0.1040	0.03287	-0.1040	0.03287	-0.1040	0.03287	-0.1040	0.03287
$\hat{d}$	$se(\hat{d})$	-0.1947	0.02928	-0.1947	0.02928	-0.1947	0.02928	-0.1947	0.02928

Table 3: Fisher-Scoring Algorithm values

(f) First, we fit a new Negative Binomial GLM with a linear predictor that excludes the indicator variable for the age of the driver, using the glm.nb function in R. Call this smaller model M2 and the larger model from (e) M1.

Note: For M2, $Y_i \sim NB\ (r,\ \mu_i = exp(a + bz_{i1} + cz_{i2} + log(E_i))$

To compare the two nested models, we use the Likelihood Ratio Test at the 5% significance level. We are testing $H_0 : d = 0$ vs $H_1 : d \neq 0$.

We use R to extract the maximum log-likelihoods for the two models and calculate the test statistic as

$$2\left[max(\ell_{M1}) - max(\ell_{M2})\right] = 2\left[-17410.59 + 17432.74\right] = 44.28583$$

Since the test statistic 44.29 is larger than $\chi^2_{1,0.05} = 3.841$, we reject H_0 and conclude that the model is better with the indicator variable for the age of the driver.

(g) We use the two model selection criteria:
Likelihood Ratio Test: Since Poisson GLM is nested in the Negative Binomial GLM, we use the LRT at a conventional 5% significance level. The test is H_0: Poisson GLM is a better fit vs Negative Binomial GLM is a better fit. Using R to get the maximum log-likelihoods for each model, we calculate the test statistic as

$$2\left[max(\ell_{NB}) - max(\ell_P)\right] = 2\left[-17410.59 + 17434.82\right] = 48.46094$$

Since the test statistic 48.46 is larger than $\chi^2_{1,0.05} = 3.841$, we reject H_0 and conclude that the Negative Binomial GLM is a better fit.

Aikake's Information Criterion: We use R to get each model's AIC value, given as

$$AIC_P = 34871.65 \quad , \quad AIC_{NB} = 34831.19$$

Since the Negative Binomial's AIC value 34831 is smaller than the Poisson's AIC value 34872, we conclude that the Negative Binomial GLM is a better fit.

(h) Poisson Model:
Its probability mass function can be written in the form of the exponential family where

$$\theta = log\,\lambda \quad , \quad \phi = 1 \quad , \quad w = 1 \quad , \quad b(\theta) = e^{\theta} \quad , \quad c(y,\phi) = log\,y!$$

It follows that $b''(\theta) = e^{\theta} = \lambda$, so that $Var(\mu) = \frac{\phi}{w} b''(\theta) = \lambda = \mu$.
Hence, the i^{th} Pearson's Residual has the following expression

$$\frac{y_i - \hat{\mu}_i}{\sqrt{Var(\hat{\mu}_i)}} = \frac{y_i - \hat{\mu}_i}{\sqrt{\hat{\mu}_i}}$$

The log-likelihood function is

$$\ell(\lambda) = \sum_{i=1}^{67856} log\, f(y_i|\lambda_i) = \sum_{i=1}^{67856} (-\lambda_i + y_i\, log\, \lambda_i - log\, y_i!)$$

and hence the residual scaled deviance is

$$D^*(\lambda|y) = 2[\ell_{SAT}(\hat{\lambda}) - \ell_{MOD}(\hat{\lambda})] = \sum_{i=1}^{67856} 2\Big[\hat{\lambda}_i - y_i\Big(1 - log\Big(\frac{y_i}{\hat{\lambda}_i}\Big)\Big)\Big]$$

Then $d_i = 2[\hat{\mu}_i - y_i[1 - log\big(\frac{y_i}{\hat{\mu}_i}\big)]$ since $\hat{\lambda} = \hat{\mu}$
Consequently, the i^{th} Deviance residual is

$$sign(y_i - \hat{\mu}_i)\sqrt{d_i} = sign(y_i - \hat{\mu}_i)\sqrt{2\Big[\hat{\lambda}_i - y_i\Big(1 - log\Big(\frac{y_i}{\hat{\lambda}_i}\Big)\Big)\Big]}$$

Negative Binomial Model:
Referring back to the derivations in (d), the i^{th} Pearson's residual is given by

$$\frac{y_i - \hat{\mu}_i}{\sqrt{Var(\hat{\mu}_i)}} = \frac{y_i - \hat{\mu}_i}{\sqrt{\frac{\phi}{w_i} V(\hat{\mu}_i)}} = \frac{y_i - \hat{\mu}_i}{\sqrt{\frac{\hat{\mu}_i(r+\hat{\mu}_i)}{r}}}$$

The log-likelihood function is

$$\ell(r,\mu) = \sum_{i=1}^{67856} log\, f(y_i|r,\mu_i) = \sum_{i=1}^{67856} \left[log\begin{pmatrix} y_i + r - 1 \\ y_i \end{pmatrix} + r\, log\left(\frac{r}{r+\mu_i}\right) + y_i\, log\left(\frac{\mu_i}{r+\mu_i}\right)\right]$$

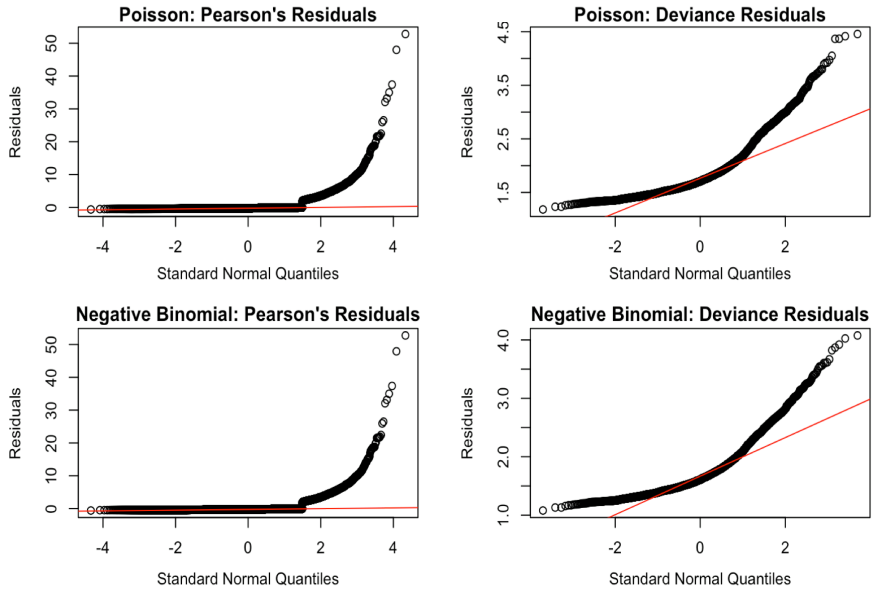
so the residual scaled deviance is

$$D^*(r,\mu|y) = 2[\ell_{SAT}(r,\hat{\mu}) - \ell_{MOD}(r,\hat{\mu})] = \sum_{i=1}^{67856} 2\left[r\, log\left(\frac{r+\hat{\mu}_i}{r+y_i}\right) + y_i\, log\left(\frac{y_i(r+\hat{\mu}_i)}{\hat{\mu}_i(r+y_i)}\right)\right]$$

Thus, we have $d_i = 2\big[r\, log\big(\frac{r+\hat{\mu}_i}{r+y_i}\big) + y_i\, log\big(\frac{y_i(r+\hat{\mu}_i)}{\hat{\mu}_i(r+y_i)}\big)\big]$.
Hence, the i^{th} Deviance residual is in the form

$$sign(y_i - \hat{\mu}_i)\sqrt{d_i} = sign(y_i - \hat{\mu}_i)\sqrt{2\left[r\, log\left(\frac{r+\hat{\mu}_i}{r+y_i}\right) + y_i\, log\left(\frac{y_i(r+\hat{\mu}_i)}{\hat{\mu}_i(r+y_i)}\right)\right]}$$

The following QQ-plots are created in R using qqnorm:



From the graphs above, we can see that the Pearson's Residuals for both the Poisson and the Negative Binomial Distribution are not normally distributed. However, the deviance residuals for both models seem to be more closely normally distributed albeit far from being a perfect fit. One more observation that we can make is that there are more outlier deviance residuals for the Poisson model than for the Negative Binomial model.

(i) Now, we aim to compute r_i as defined by Dunn and Smyth (1996). Let $Y_i \sim NB\,(r, \mu_i)$. Since the negative binomial distribution is discrete, we compute

$$a_i = \lim_{y \to y_i^-} F(y) = F(y_i^-) = F(y_i - 1) \quad , \quad b_i = F(y_i)$$

Note: $F(y) = \sum_{x=0}^{y} \binom{x+r-1}{x} \left(\frac{r}{r+\mu}\right)^r \left(\frac{\mu}{r+\mu}\right)^x$ for $y = 0, 1, 2, ...$

Hence, the i^{th} randomised quantile residual for the negative binomial GLM defined as

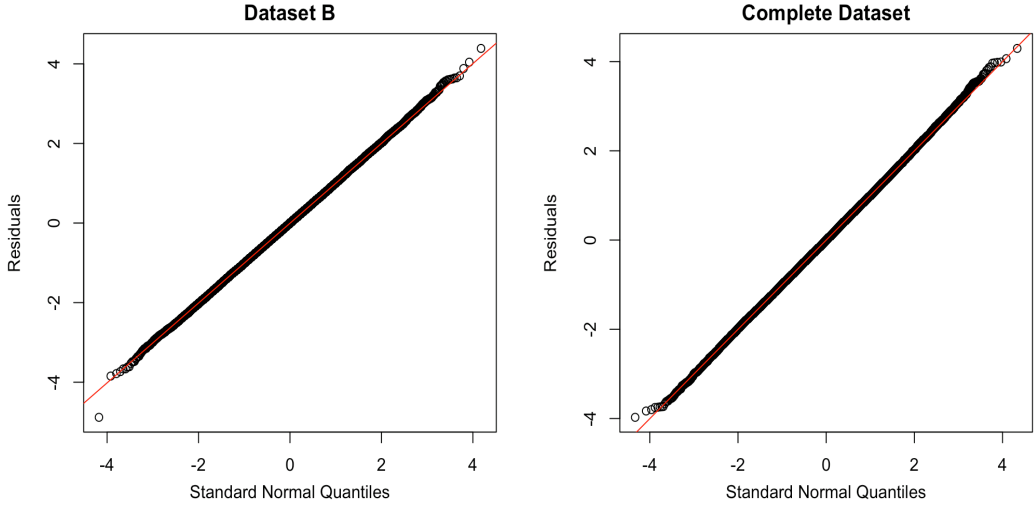
$$r_i = \Phi^{-1}(u_i)$$

where $u_i \sim Unif\left(F(y_i - 1), F(y_i)\right)$, and Φ^{-1} is defined to be the inverse cdf of a standard normal distribution.

(j) We use R to split the data into two partitions, where A takes the 1st, 3rd, 5th, ..., (n-1)th data point, and B takes the 2nd, 4th, ..., nth data point.
Then we use A to refit the negative binomial distribution using R's built-in function glm.nb, which results in:

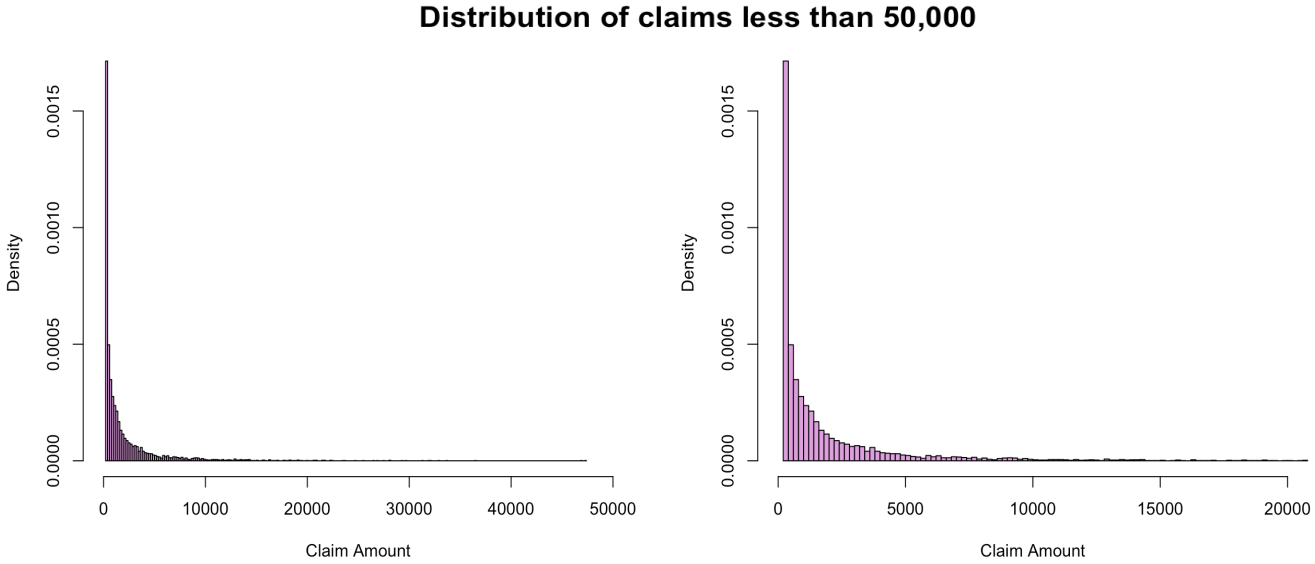
$$Y_i \sim NB\,(1.994,\ \mu_i = exp(-1.753 + 0.02789 z_{i1} - 0.1185 z_{i2} - 0.1958 z_{i3} + log(E_i))$$

We also use B to fit a negative binomial distribution using glm.nb and then use the function qres.nbinom to get the randomised quantile residuals for B. The QQ-plots are shown below

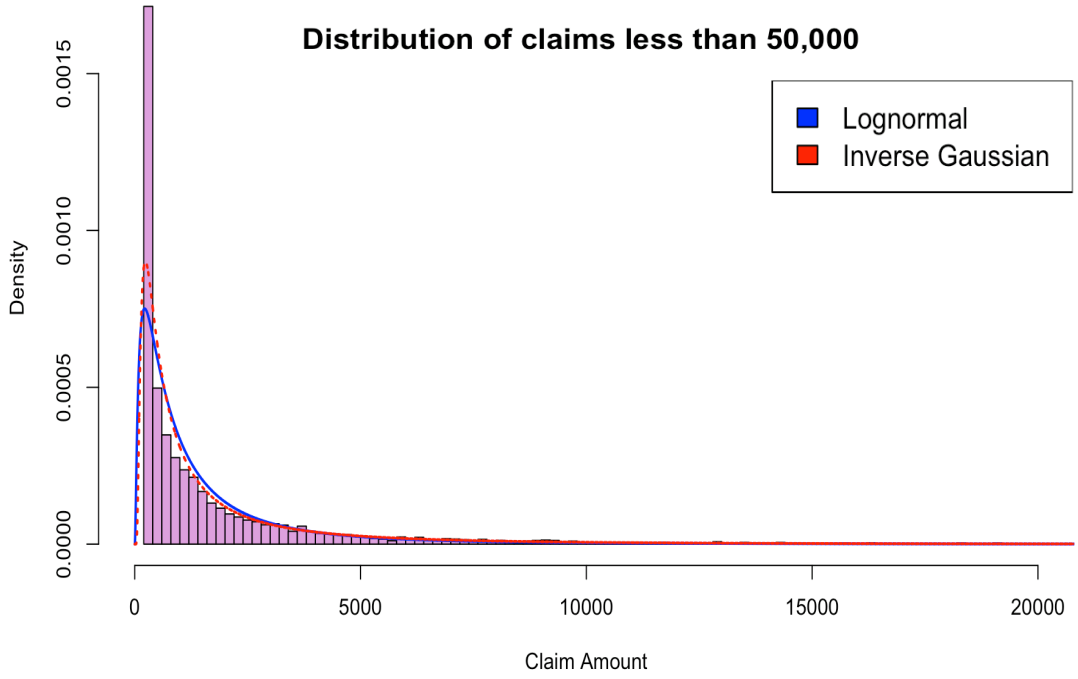


We use the same procedure to produce the QQ-plots of the randomised quantile residuals for the complete dataset. From these plots, we can see the randomised quantile residuals from our negative binomial glm fit the normal distribution quite well, especially when compared to the Pearson's residuals and deviance residuals.

(k) The density histogram of the claims which are positive, but less than 50,000 is plotted below. As the densities for values greater than 20,000 are considerably close to 0, we also show the plot for values smaller than 20.000.



The negative binomial and inverse gaussian distributions are fit to this data set using the maxLik package in R. This gives a fit of $LN(\mu = 6.809189, \sigma = 1.187762)$ and $IG(\mu = 2002.743, \lambda = 719.0556)$ for the lognormal distribution and inverse gaussian distribution respectively, where both of their probability density functions are defined in the Appendix A. Now we superimpose both of their densities to the histogram as shown below.



We would not be able to apply the likelihood ratio test to compare the fits of these two distributions to the data as neither distribution is a nested model of the other one.

(l) Now that we are including the greatest observation in the dataset, we refit the lognormal and inverse gaussian distributions to the new dataset using the same method as part (k). The new fits are $LN(\mu = 6.810081, \sigma = 1.189179)$ and $IG(\mu = 2014.4041, \lambda = 717.7727)$. Next, we are able to compare the applications of our model

fits in terms of Value at Risk (VaR). VaR is a risk measure used to prevent companies from becoming technically insolvent. In our case, it will correspond to the 100-pth percentile of the positive claims distribution. The VaR is now able to be computed as follows.

```
security.levels = c(0.9, 0.95, 0.99)
lnorm.var <- qlnorm(security.levels, meanlog = mle.lognormal[1],
sdlog = mle.lognormal[2])
invgauss.var <- qinvgauss(security.levels, mean = mle.invgauss[1],
shape = mle.invgauss[2])
obs.var <- quantile(claims.subset, security.levels)
```

Which give us the following results:

	Security Levels		
	0.90	**0.95**	**0.99**
Lognormal	4163.350	6413.155	14422.22
Inverse Gaussian	4866.604	7766.625	16813.93
Observed	4952.976	8124.660	17937.13

Table 4: VaRs at each security level

As we can see in the above table, The inverse gaussian fit is more conservative than the lognormal fit across all security levels, however neither distribution is conservative enough when compared to the observed values. This implies that if we are going to assume either model fits our positive claims distribution, we may be increasing our chances of insolvency by underestimating our VaRs.

(m) To perform our Kolmogorov-Smirnov (KS) tests, we begin with the lognormal distribution. We are testing
H_0 : the sample of the positive claim amounts comes from the probability distribution function associated with $LN(\mu = 6.810081, \sigma = 1.189179)$ vs
H_1 : the sample of positive claim amounts does *not* come from the probability distribution function associated with $LN(\mu = 6.810081, \sigma = 1.189179)$. This test will be determined using the KS statistic which is defined as $D = max(D^+, D^-)$, where

$$D^+ = \max_{1 \le j \le N} \{\frac{j}{N} - \hat{F}(x_{(j)})\}, D^- = \max_{1 \le j \le N} \{\hat{F}(x_{(j)}) - \frac{j-1}{N}\}$$

with N = the number of data points in our dataset, and $\hat{F}(x_{(j)})$ represents the cumulative density function of the fitted lognormal distribution, at the j^{th} ordered data point of our claims dataset.
This can be easily calculated using a for loop, as shown below.

```
sorted.claims = sort(claims.subset)
n = length(sorted.claims)

#CDF at y[i]th data point from claims distribution
F.claims <- function(i) {
  plnorm(sorted.claims[i], meanlog = mle.lognormal[1], sdlog = mle.lognormal[
2])
}

#get observed statistic
for (i in 2:n) {
  ks = max(F.claims(i)-(i-1)/n, i/n-F.claims(i))
  if (ks > ks.lnorm) {
    ks.lnorm = ks
  }
}
```

Using this code, we compute the test statistic $D = 0.1021038$. Now, we use Monte Carlo

simulation to simulate a p-value used to conclude our KS test. This is done by simulating a data set from $LN(\mu = 6.810081, \sigma = 1.189179)$, calculating the observed KS statistic for said dataset, and repeating this process 10,000 times. Our simulated p-value is therefore the proportion of simulated KS statistics which are greater than or equal to the observed statistic, D. After executing the following code,

```
ITER = 10000
ks.mat.lnorm = rep(0, ITER)

#CDF at y[i]th data point from simulated distribution
F.data <- function(j) {
  plnorm(sorted.data.lnorm[j], meanlog = mle.lognormal[1], sdlog = mle.lognor
mal[2])
}

for(i in 1:ITER) {
  #simulate and sort data
  sorted.data.lnorm = sort(rlnorm(n, meanlog = mle.lognormal[1], sdlog = mle.
lognormal[2]))

  #calculate ks statistic for each simulation, store in vector
  for (j in 2:n) {
    ks.lnorm.stat = max(F.data(j)-(j-1)/n, j/n-F.data(j))
    if (ks.lnorm.stat > ks.mat.lnorm[i]) {
      ks.mat.lnorm[i] = ks.lnorm.stat
    }
  }
}

#simulated p val
pval.ks.lnorm <- sum(ks.mat.lnorm>=ks.lnorm)/ITER
```

the simulated p-value is 0.00.
Note: this does not necessarily mean the exact p-value is zero, but rather, the exact p-value is likely to be less than $\frac{1}{10{,}000}$, and thus is too small to be approximated with 10,000 iterations. Nonetheless, at a conventional significance level of $\alpha = 0.05$, we reject H_0 and conclude that the sample of the positive claim amounts does not come from the distribution $LN(\mu = 6.810081, \sigma = 1.189179)$.

To perform our KS test for the fitted inverse gaussian distribution, we make the following adjustments.
H_0 : the sample of the positive claim amounts comes from the probability distribution function associated with $IG(\mu = 2014.4041, \lambda = 717.7727)$ vs
H_1 : the sample of positive claim amounts does *not* come from the probability distribution function associated with $IG(\mu = 2014.4041, \lambda = 717.7727)$.
D is defined the same way, however $\hat{F}()$ now represents the cumulative density function of the fitted inverse gaussian distribution. We repeat the same method as described before, where we replace the lognormal distribution with the fitted inverse gaussian distribution. This provides us with $D = 0.09489285$, which also results in a simulated p-value of 0.00. Therefore, we reject H_0, and conclude that the sample of the positive claim amounts does not come from the distribution $IG(\mu = 2014.4041, \lambda = 717.7727)$.

(n) We also perform our Anderson-Darling (AD) tests on the fitted distributions.
The AD test may seem favourable when compared to the KS test, as the AD test statistic makes use of the weighted square difference between the empirical cumulative density function (CDF) and the theoretical CDF, whereas the KS statistic only makes use of the absolute difference between the empirical and theoretical CDFs. Because of this, the AD test is more sensitive at the tails of the distribution compared to the KS test. This may be particularly useful when testing our positive claims distribution, as the histogram plotted

in part (k) shows that there is a considerable difference between the theoretical and empirical distribution at the left end of the range of data. Because of this, we expect a more accurate conclusion when performing the AD test.
These tests will be performed with the same hypotheses as stated in (m). For the lognormal distribution, we are testing
H_0 : the sample of the positive claim amounts comes from the probability distribution function associated with $LN(\mu = 6.810081, \sigma = 1.189179)$ vs
H_1 : the sample of positive claim amounts does *not* come from the probability distribution function associated with $LN(\mu = 6.810081, \sigma = 1.189179)$.
The AD test statistic is calculated as

$$A^2 = -N - \frac{1}{N}\sum_{j=1}^{N}[(2j-1)\log(\hat{F}(x_{(j)})) + (2n+1-2j)\log(1-\hat{F}(x_{(j)}))]$$

with notation consistent to what was defined in (m). We again calculate our observed test statistic and use Monte Carlo simulation to estimate our p-value, as described in part (m).
Similarly, for the inverse gaussian distribution, our hypotheses are
H_0 : the sample of the positive claim amounts comes from the probability distribution function associated with $IG(\mu = 2014.4041, \lambda = 717.7727)$ vs
H_1 : the sample of positive claim amounts does *not* come from the probability distribution function associated with $IG(\mu = 2014.4041, \lambda = 717.7727)$.
The same process is followed again, replacing the lognormal distribution with the fitted inverse gaussian distribution. The results from both tests are presented below:

	A	A^2	$p-value$
Lognormal	8.514396	72.49494	0.00
Inverse Gaussian	7.320299	53.58678	0.00

Table 5: AD test results

Consequently, we reject both null hypotheses and conclude that the sample of the positive claim amounts does not come from the distribution $LN(\mu = 6.810081, \sigma = 1.189179)$, nor does it come from the distribution $IG(\mu = 2014.4041, \lambda = 717.7727)$.

Reference:
Dunn, P. and Smyth, G. (1996). Randomized quantile residuals. *Journal of Computational and Graphical Statistics*, 5(3):236-244.

Appendix A

- The probability mass function of the zero inflated Poisson distribution:

$$f(y|\psi,\theta) = \begin{cases} \psi + (1-\psi)\,\exp\{-\theta\} & y = 0 \\ (1-\psi)\,\dfrac{\exp\{-\theta\}\,\theta^y}{y!} & y = 1, 2, \ldots \end{cases}$$

  with $\theta > 0$ and $0 \leq \psi \leq 1$.

- The probability mass function of the negative binomial distribution:

$$f(y|r,\beta) = \binom{y+r-1}{y} \left(\frac{1}{1+\beta}\right)^r \left(\frac{\beta}{1+\beta}\right)^y \quad y = 0, 1, \ldots \; r, \beta > 0.$$

- Probability density function of the lognormal distribution:

$$f(x|\mu,\sigma) = \frac{1}{x\,\sigma\sqrt{2\pi}}\,\exp\left\{-\frac{1}{2}\left(\frac{\log x - \mu}{\sigma}\right)^2\right\} \quad \text{with } \mu \in \mathbb{R},\ \sigma > 0 \text{ and } x > 0.$$

- Probability density function of the inverse gaussian distribution:

$$f(x|\mu,\lambda) = \left(\frac{\lambda}{2\,\pi\,x^3}\right)^{1/2}\,\exp\left\{-\frac{\lambda\,(x-\mu)^2}{2\mu^2 x}\right\} \quad \text{with } \lambda > 0,\ \mu > 0 \text{ and } x > 0.$$

```
intercept <- c(rep(1, 67856))
negbin.repar <- function(y,r,mu){
  factorial(y + r - 1)/(factorial(y)*factorial(r - 1))*(r/(r + mu))^r*(mu/(r
+ mu))^y
}

#Initial values
new.a <- coef(summary(poisglm))[1, 1]; new.b <- coef(summary(poisglm))[2, 1]
new.c <- coef(summary(poisglm))[3, 1]; new.d <- coef(summary(poisglm))[4, 1]
new.r <- coef(summary.negbin)[1, 1]
new.mu <- exp(new.a + new.b*vehvalue + new.c*vehage + new.d*drivage + log(exp
osure))

#Fisher-scoring algorithm
U1 <- U2 <- U3 <- U4 <- i <- 1
tol.level <- 1e-10

while((abs(U1) > tol.level) | (abs(U2) > tol.level) | (abs(U3) > tol.level) |
(abs(U4) > tol.level)){
  r0 <- new.r; a0 <- new.a; b0 <- new.b; c0 <- new.c; d0 <- new.d; mu0 <- new.mu

  I11 <- sum(r0*mu0*intercept*intercept/(r0 + mu0))
  I12 <- sum(r0*mu0*intercept*vehvalue/(r0 + mu0))
  I13 <- sum(r0*mu0*intercept*vehage/(r0 + mu0))
  I14 <- sum(r0*mu0*intercept*drivage/(r0 + mu0))
  I22 <- sum(r0*mu0*vehvalue*vehvalue/(r0 + mu0))
  I23 <- sum(r0*mu0*vehvalue*vehage/(r0 + mu0))
  I24 <- sum(r0*mu0*vehvalue*drivage/(r0 + mu0))
  I33 <- sum(r0*mu0*vehage*vehage/(r0 + mu0))
  I34 <- sum(r0*mu0*vehage*drivage/(r0 + mu0))
  I44 <- sum(r0*mu0*drivage*drivage/(r0 + mu0))
  fim <- matrix(c(I11, I12, I13, I14, I12, I22, I23, I24, I13, I23, I33, I34,
I14, I24, I34, I44), 4, 4)
  inv.fim=solve(fim)

  U1 <- sum((r0*(claimnb - mu0))*intercept/(r0 + mu0))
  U2 <- sum((r0*(claimnb - mu0))*vehvalue/(r0 + mu0))
  U3 <- sum((r0*(claimnb - mu0))*vehage/(r0 + mu0))
  U4 <- sum((r0*(claimnb - mu0))*drivage/(r0 + mu0))

  a1 <- a0+inv.fim[1,1]*U1+inv.fim[1,2]*U2+inv.fim[1,3]*U3+inv.fim[1,4]*U4
  b1 <- b0+inv.fim[2,1]*U1+inv.fim[2,2]*U2+inv.fim[2,3]*U3+inv.fim[2,4]*U4
  c1 <- c0+inv.fim[3,1]*U1+inv.fim[3,2]*U2+inv.fim[3,3]*U3+inv.fim[3,4]*U4
  d1 <- d0+inv.fim[4,1]*U1+inv.fim[4,2]*U2+inv.fim[4,3]*U3+inv.fim[4,4]*U4
  mu1 <- exp(a1 + b1*vehvalue + c1*vehage + d1*drivage + log(exposure))

  loglikfun.r <- function(param){
    r1 <- param[1]
    sum(log(negbin.repar(claimnb, r1, mu1)))
  }
  mle.r <- maxLik(logLik=loglikfun.r, start=c(r1=r0))
```

```
new.mu <- mu1; new.r <- coef(summary(mle.r))[1, 1]
new.a <- a1; new.b <-b1; new.c <-c1; new.d <- d1
sd.r <- coef(summary(mle.r))[1, 2]
sd.a <- sqrt(inv.fim[1, 1]); sd.b <- sqrt(inv.fim[2, 2])
sd.c <- sqrt(inv.fim[3, 3]); sd.d <- sqrt(inv.fim[4, 4])

print(paste0("Iteration number:", i))
print("Maximum likelihood estimates:")
print(paste0("r", i, "=", new.r))
print(paste0("a", i, "=", new.a))
print(paste0("b", i, "=", new.b))
print(paste0("c", i, "=", new.c))
print(paste0("d", i, "=", new.d))
print("Standard errors:")
print(paste0("r", i, "=", sd.r))
print(paste0("a", i, "=", sd.a))
print(paste0("b", i, "=", sd.b))
print(paste0("c", i, "=", sd.c))
print(paste0("d", i, "=", sd.d))
print("Maximum log-likelihood:")
print(summary.negbinglm1$loglik)
i <- i + 1}
```