# Applied Statistics Assignment 3

Kyriacos Xanthos
CID: 01389741

December 10, 2021

## Question 1

### a)

```
> source(’glmxy.R’)
> x <- dfrm$x
> y <- dfrm$y
> myglm <- glm(y~x,family = poisson(link=’log’))
> print(myglm$coefficients)
(Intercept)           x
  0.8737919  -3.4809493
```

Here we can see that $\hat{\beta} = (0.8738, -3.4809)$.

### b)

Note that $g^{-1}(\hat{\eta}_i) = \hat{\lambda}_i = e^{X_i\hat{\beta}}$ and therefore $\hat{\beta} = X_i^{-1}\log(\hat{\lambda}_i)$. But note that with the canonical link function for a Poisson distribution we have:

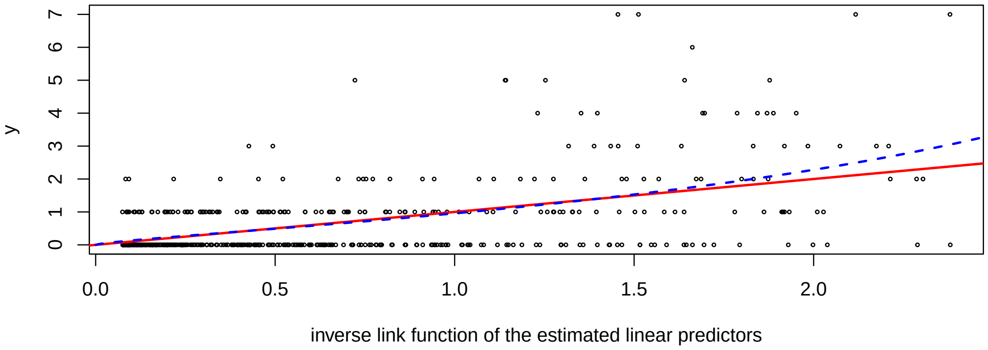
$$\hat{y_i} = e^{X_i\hat{\beta}} = \hat{y_i} = e^{X_i X_i^{-1}\log(\hat{\lambda}_i)} = \hat{\lambda}_i$$

which means that the fitted regression line on this axis is simply the line with intercept 0 and gradient 1. Now for the true regression we recall that for the canonical link function for the negative binomial distribution gives the regression:

$$\hat{y_i} = \frac{2}{\exp\left[-\hat{\beta}X_i\right] - 1} = \frac{2}{\exp\left[-\left(\beta_1 + \beta_2\frac{\log(\hat{\lambda}_i - \hat{\beta}_1)}{\hat{\beta}_2}\right)\right] - 1}$$

where we decomposed $\beta$ and $\hat{\beta}$ to their two components corresponding to the intercept and the gradient. Now plotting both of the lines on the required axis we observe that the two lines are very similar up to the value $\hat{\lambda}_i = 1.5$ but start to deviate after that. It seems like both lines are not able to capture the whole picture of the model since it seems that the weight of $y = 0$ (which is large since most of the points are in that category) is not fully incorporated in the model (both lines seem to deviate quite fast from it). The goodness of fit of the estimated regression line (red) is good at the start and can roughly follow the general trend of the data. The true regression line (blue, dashed) is slowly increasing after $\hat{\lambda}_i = 1.5$, something that is not captured from the Poisson model we have fitted but looking at the data it makes sense. The closeness of the two regression lines reinforces the idea of using a Poisson model as an approximation of a Negative Binomial model.

```
> true_beta <- c(-0.5,-2.5)
> est_beta <- c(myglm$coefficients[1],myglm$coefficients[2] )
> xs <- seq(0,2.5,by=0.01)
> plot(exp(myglm$coefficients[1] +myglm$coefficients[2]*x), y, cex=0.4)
> abline(0,1,col=’red’, lwd=2)
> lines(xs, 2/(exp(-(true_beta[1] +
+  true_beta[2]*(log(xs)-est_beta[1])/est_beta[2]))-1), col=’blue’,lwd=2,lty=2)
```



## d)

In order to calculate the 99% confidence interval for the mean response value for the covariate value $x = 0.5$ we will use the following formula:

$$\left(g^{-1}\left(\hat{\eta}_\star - Z_{\alpha/2}\sqrt{\boldsymbol{x}_\star^T\left(X^TWX\right)^{-1}\boldsymbol{x}_\star}\right), g^{-1}\left(\hat{\eta}_\star + Z_{\alpha/2}\sqrt{\boldsymbol{x}_\star^T\left(X^TWX\right)^{-1}\boldsymbol{x}_\star}\right)\right), \qquad (1)$$

where $g$ is the canonical link function, $\hat{\eta}_\star = \hat{\beta}\cdot\boldsymbol{x}_\star$, $\boldsymbol{x}_\star = (1, 0.5)^T$, $Z_{\alpha/2}$ indicates the critical value where the right-tailed area under a standard normal distribution is $\alpha/2$, and $W$ is the weights matrix [1].

Note that the observed information matrix and the expected information matrix coincide in the case of the canonical link function [2], so we can use the function `vcov()` in `R` to compute $\left(X^TWX\right)^{-1}$.

```
> beta <- myglm$coefficients
> x_star <- matrix(data = c(1,0.5), byrow = T, nrow = 2)
> invJ <- vcov(myglm)
> eta_star <- t(x_star)%*%beta
> eta_star_R <- predict(myglm, data.frame(x=x_star))
> z_alpha <- function(alpha) c(qnorm((alpha)/2), -qnorm((alpha)/2))
> CI <-c(exp(eta_star + z_alpha(0.01)[1]*sqrt(t(x_star)%*%invJ%*%x_star)),
+          exp(eta_star + z_alpha(0.01)[2]*sqrt(t(x_star)%*%invJ%*%x_star)))
> CI
[1] 0.3409977 0.5181523
```

So the 99% CI is (0.341, 0.518) for the covariate value $x = 0.5$.

## e)

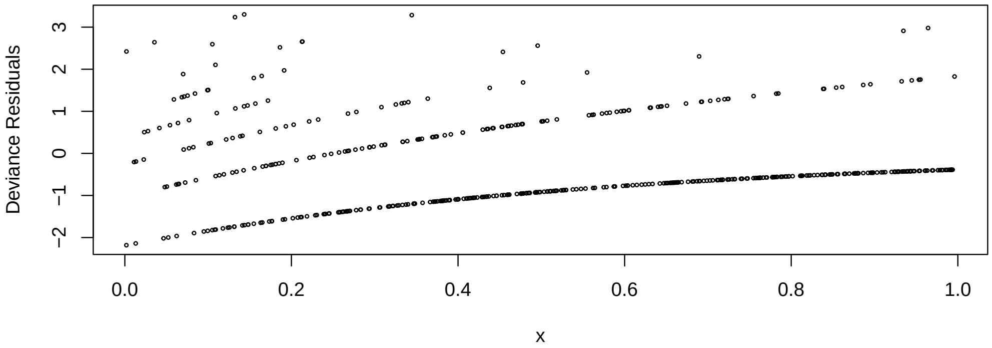
In this part we calculate the deviance residuals defined by:

$$r_{D,i} = \text{sign}\left(y_i - \hat{\mu}_i\right)\sqrt{2\left(y_i \log\left(y_i/\hat{\mu}_i\right) - \left(y_i - \hat{\mu}_i\right)\right)} \qquad (2)$$

where $\hat{\mu}_i = \hat{\beta}\cdot x_i$ [3] p.83. We can see this in `R` in a plot against the $x$:

```
> plot(x, residuals(myglm, type='deviance'), cex=0.4,
ylab = 'Deviance Residuals')
> mean(residuals(myglm, type='deviance')<0)
[1] 0.696
```

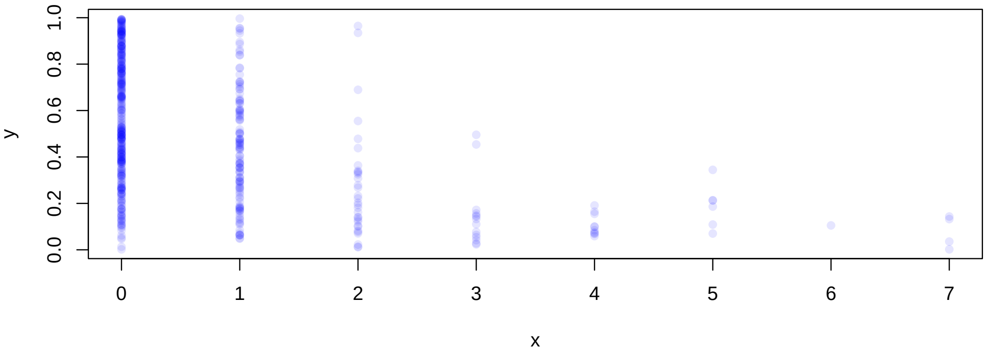
We also note that the proportion of the residuals that are negative is 69.6%.



## f)

Firstly we remove the intercept from the model so that we can directly see how each of the factors affect the model.

```
> lm1 <- lm(x ~ 0+as.factor(y))
> summary(lm1)

Call:
lm(formula = x ~ 0 + as.factor(y))

Residuals:
    Min      1Q  Median      3Q     Max
-0.5793 -0.1818 -0.0072  0.1991  0.6888

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
as.factor(y)0  0.58111    0.01417  41.017  < 2e-16 ***
as.factor(y)1  0.44035    0.02328  18.913  < 2e-16 ***
as.factor(y)2  0.27549    0.04452   6.188 1.29e-09 ***
as.factor(y)3  0.14995    0.06731   2.228   0.0263 *
as.factor(y)4  0.10678    0.07964   1.341   0.1806
as.factor(y)5  0.18924    0.10282   1.841   0.0663 .
as.factor(y)6  0.10505    0.25185   0.417   0.6768
as.factor(y)7  0.07825    0.12592   0.621   0.5346
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.2518 on 492 degrees of freedom
Multiple R-squared:  0.8094,        Adjusted R-squared:  0.8063
F-statistic: 261.1 on 8 and 492 DF,  p-value: < 2.2e-16
> plot(xall[1,],xall[3,],col = rgb(red = 0, green = 0, blue = 1, alpha = 0.1),
     pch = 16, xlab="x", ylab="y")
```



To help understand these p values we make a plot so that we can more clearly see how the data points are scattered between the different levels of y. It is clear that the smallest p values are for the estimates of levels $y = 0, 1, 2 \quad (< 10^{-9})$ and a larger p value for $y = 3$ (0.0263), but large p values from the rest of the levels suggesting that they are not significant enough to be included in the model. However, looking at the plot it is clear that the first 3 levels acquire most of the points of the dataset which is the reason they are significant in the regression. Therefore we infer that the p value is proportional to the size of each category. We do get a large Adjusted $R^2$ value which shows a good fit in general but as explained before this is because most of the data points are in level $y = 0$.

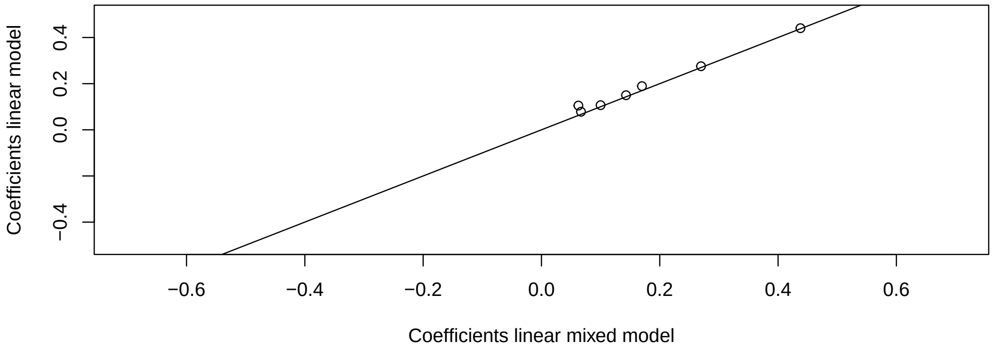
We now consider a linear mixed model with random effects for the levels of $y$.

```
> lm2 <- lmer(x ~ 0 + (1|as.factor(y)), REML = FALSE)
> summary(lm2)
Linear mixed model fit by maximum likelihood  [’lmerMod’]
Formula: x ~ 0 + (1 | as.factor(y))

     AIC      BIC   logLik deviance df.resid
    69.4     77.8    -32.7     65.4      498

Scaled residuals:
     Min       1Q   Median       3Q      Max
-2.29667 -0.71733 -0.00991  0.79620  2.75982

Random effects:
 Groups       Name        Variance Std.Dev.
 as.factor(y) (Intercept) 0.09273  0.3045
 Residual                 0.06334  0.2517
Number of obs: 500, groups:  as.factor(y), 8

> c2 <- ranef(lm2)
> c2
$‘as.factor(y)‘
  (Intercept)
0  0.57985727
1  0.43779251
2  0.26973189
3  0.14297720
4  0.09995146
5  0.16990034
6  0.06241671
7  0.06683813

with conditional variances for \as.factor(y)"
```

We can see that the numerical values of the estimates from the normal linear mixed model are very similar to the estimates from the simple linear model. We can plot them against each other and see how close they are to the line with gradient 1 and intercept 0:

```
plot(as.vector(unlist(c2)),c1, xlim=c(-0.7, 0.7), ylim =c(-0.5,0.5),
     ylab=’Coefficients linear model’, xlab=’Coefficients linear mixed model’)
abline(0,1)
```

We can indeed see from the plot that the estimates are very close to each other. The two approaches have very different assumptions. The assumptions of the Normal linear model is that the data have normally distributed errors which are the same for all categories. The Normal linear mixed model does not use this assumption but instead random effects allow us to induce structured correlation between observations which are drawn from the same group [3]. Looking at the plot of the data we infer that different levels indeed have different correlation between observations and they all have different variances. This means for the



data in this question a Normal linear Mixed model would be better in capturing the true nature of the observations. Consequently we will be more inclined to trust the estimates from the Normal linear Mixed model.

## Question 2

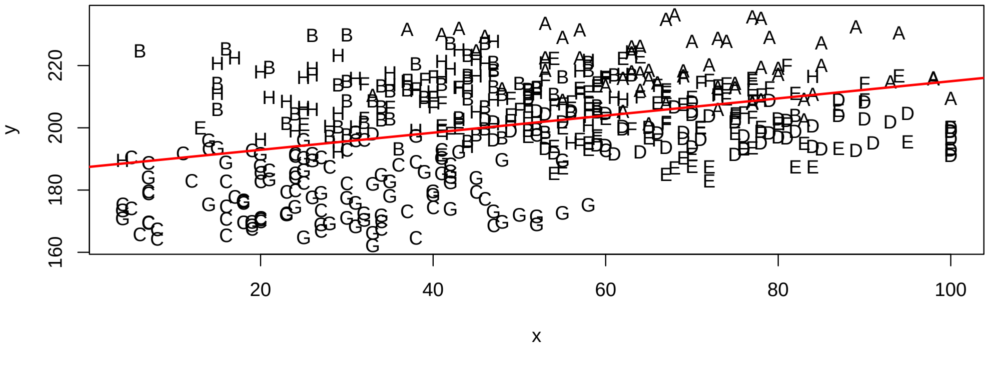
### a)

We begin by loading the dataset and storing it in the dataframe called `my_data`.

```
> my_data <- data.frame(source("xyz.R"))
> colnames(my_data) <- c(’x’,’y’,’z’)
```

We then fit a simple Normal linear model and plot the line of best fit on a scatterplot of response variable y against predictor variable x with each point labeled from the category label it is coming from.

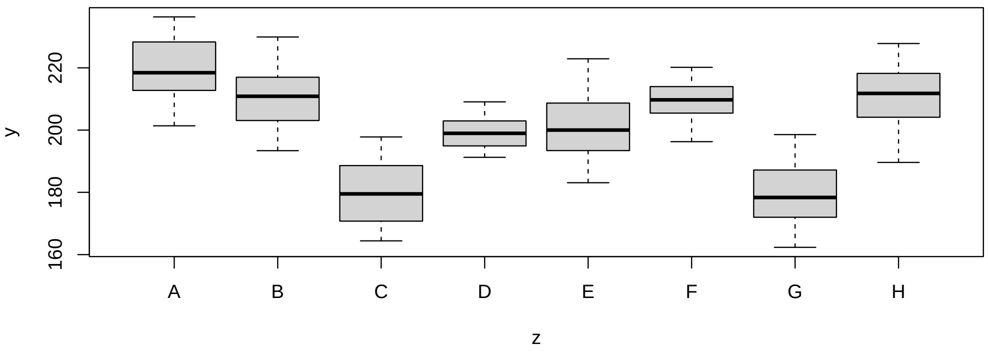
```
> my_lm <- lm(y˜x, data=my_data)
> plot(y ˜ x, data=my_data, pch=as.character(z))
> abline(my_lm$coefficients[1], my_lm$coefficients[2], col=’red’, lwd=2)
```



Note that the intercept from the fitted linear model is 187.41 and the gradient 0.28. It is clear from the plot that a simple linear model appears to be inapropriate for this dataset since the categories seem to be linearly seperable. For example most of category A is in the top right, most of the category C is in the bottom left, category D is concentrated on a small rectangle and in general we can see that the categories of each observation play a large role with what y and x values they acquire. Another problem with a simple linear model is that the data is spread out and even if we fit a linear model, there is a very large gap between the fitted line and many of the observations. In other words, we have very small $R^2 = 0.15$. The data also seem to be right-cencored as there are a lot of values at exactly $x = 100$.

### b)

```
boxplot(y˜z, data=my_data)
```



We can calculate the means of each group in the following way:

```
> means <- aggregate(y ~  z, my_data, mean)
> means
  z        y
1 A 219.7284
2 B 210.6880
3 C 179.8769
4 D 199.2856
5 E 200.7198
6 F 209.2756
7 G 179.8393
8 H 211.1180
```

It is clear that the response variable $y$ is indeed dependent on the category since we can see from the boxplot that each category is distributed differently with different means. We can see from the numerical means that category $C$ and $G$ are very similar, and $A$ has the largest mean value from all the categories. Categories $D$ and $E$ also have very similar means but the variance of $E$ is larger than $D$.

### c)

Assuming now a normal linear mixed model for $y$ assuming a fixed effect for $x$ together with an intercept, and random effect intercept terms corresponding to membership of the categories $A - H$ we can see the summary of such a model below:

```
> my_lmer <- lmer(y ~ 1+x+(1|z), data = my_data)
> summary(my_lmer)
Linear mixed model fit by REML [’lmerMod’]
Formula: y ~ 1 + x + (1 | z)
   Data: my_data

REML criterion at convergence: 3472.3

Scaled residuals:
     Min       1Q   Median       3Q      Max
-2.45717 -0.75415 -0.06246  0.72526  2.56690

Random effects:
 Groups   Name        Variance Std.Dev.
 z        (Intercept) 213.03   14.595
 Residual              74.73    8.645
Number of obs: 480, groups:  z, 8

Fixed effects:
             Estimate Std. Error t value
```

```
(Intercept) 2.010e+02  5.349e+00  37.579
x           5.947e-03  2.680e-02   0.222

Correlation of Fixed Effects:
  (Intr)
x -0.253
```

The summary reports a value of 213.03 for the variance of the random effects and a value of 74.73 for the error variance.

### d)

We explore now a reduced dataset with only the categories $B, F, H$. We use Unrestricted Maximum Likelihood this time.

```
> my_lmer2 <- lmer(y ˜ 1+x+(1|z), data = my_data_2,  REML=FALSE)
> logLik(my_lmer2)
'log Lik.' -634.7416 (df=4)
```

We obtain the value of -634.7416 for the maximum log-likelihood function from the reduced dataset.

### e)

```
> simple_lm <- lm(y˜ 1 + x, data=my_data_2)
> logLik(simple_lm)
'log Lik.' -634.7416 (df=3)
> d <- as.numeric(2*(logLik(my_lmer2)-logLik(simple_lm)))
> d
[1] 2.273737e-13
```

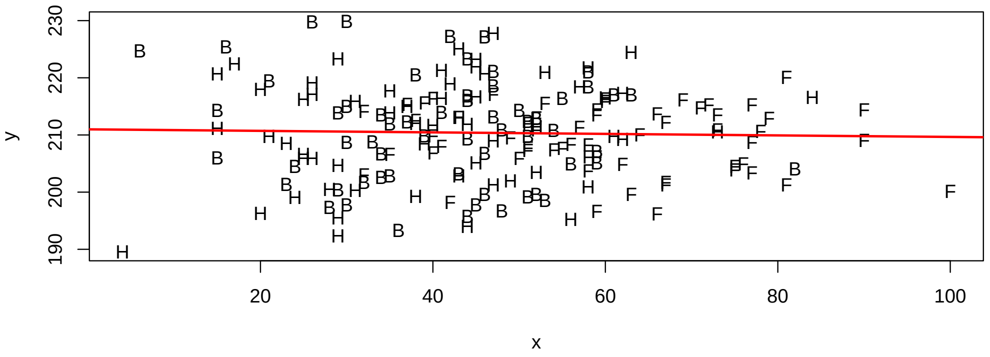
By fitting a simple linear model we get the exact same maximised log-likelihood of -634.7416 with three degrees of freedom instead of 4 this time since we have no random effects and a deviance of $2.27 \times 10^{-13}$. This shows that the two models are very similar under this reduced dataset. This makes sense since B, F, H have very similar average values, so the random effects do not affect the model this much because the groups do not have very different variances and means between them which is an implicit assumption when using Normal Linear Mixed Models.

### f)

```
> ds <- numeric(1000)
> for (i in 1:1000) {
+   y_new <- unlist(simulate(simple_lm))
+   nullmod <- lm(y_new˜ 1 + x, data=my_data_2)
+   altmod <-  suppressMessages(lmer(y_new˜ 1+x+(1|z),
+                                         data = my_data_2,  REML=FALSE))
+   ds[i] <- as.numeric(2 * (logLik(altmod) - logLik(nullmod)))
+ }
> phat <- mean(ds>d)
> phat
[1] 0.249
> sqrt(phat*(1-phat)/1000)
[1] 0.01367476
```

Using parametric bootstrap, we obtain a p value of 0.249 with a standard error of 0.0137. The p value is way above the 5% significance level, showing that we do not have enough evidence to reject the null hypothesis that the given data under the reduced categories B, F, H can be modeled using simple linear regression. Even with more repetitions the parametric bootstrap method always gives a large p value. As we mentioned in part e), this makes sense since the three categories we are comparing have very similar means and variances between them, which means the random effects in the normal linear mixed models do not make a difference on fitting a model. In such cases we stick with the simple normal linear model. We can make a plot of this reduced dataset and see that indeed categories B, F, H do mix well, however they show that there is not a very clear linear relationship between them which is why our line of best fit has almost a gradient of 0.

```
> plot(y ~ x, data=my_data_2, pch=as.character(z))
> abline(simple_lm$coefficients[1], simple_lm$coefficients[2],
col='red', lwd=2)
```



## References

[1] Din-Houn Lau. Applied statistics lecture notes. *Imperial College London (Lecture)*, October 2020.

[2] Wikipedia. Generalized linear model — Wikipedia, the free encyclopedia. http://en.wikipedia.org/w/index.php?title=Generalized%20linear%20model&oldid=1053374390, 2021. [Online; accessed 07-December-2021].

[3] Nick Heard. Applied statistics lecture notes. *Imperial College London (Lecture)*, October 2021.