Assignment 3 XY

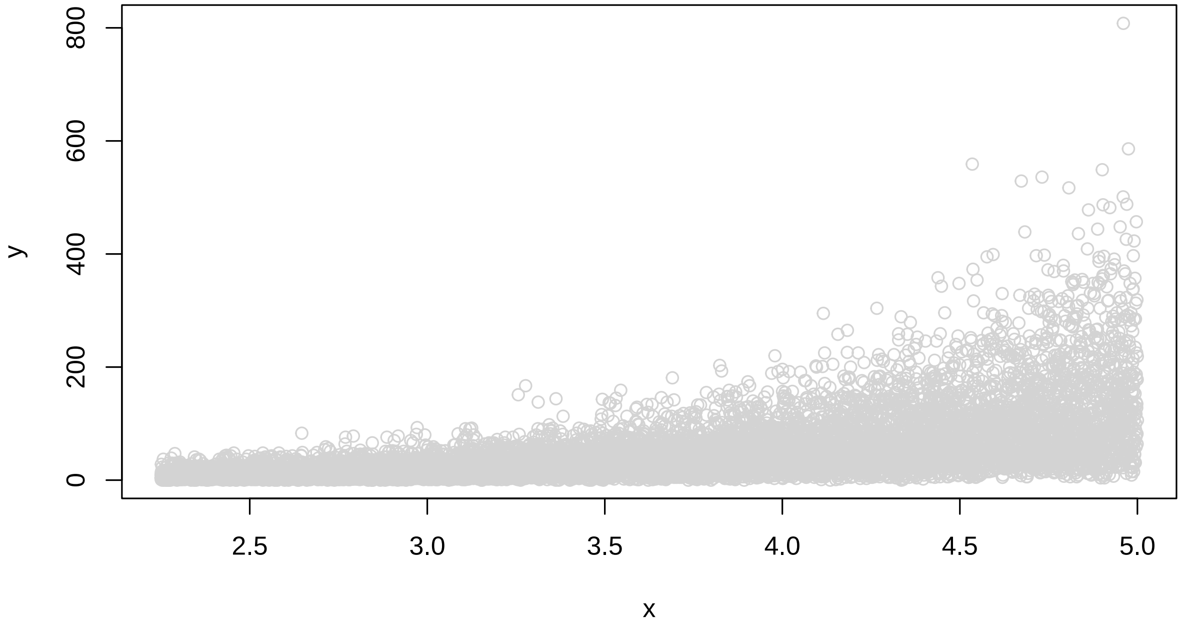
Ben Lu

2023-09-14

Question 1

(a)

```r
xy.df <- read.csv("XY.csv")
plot(y ~ x, data = xy.df, col="lightgrey")
```



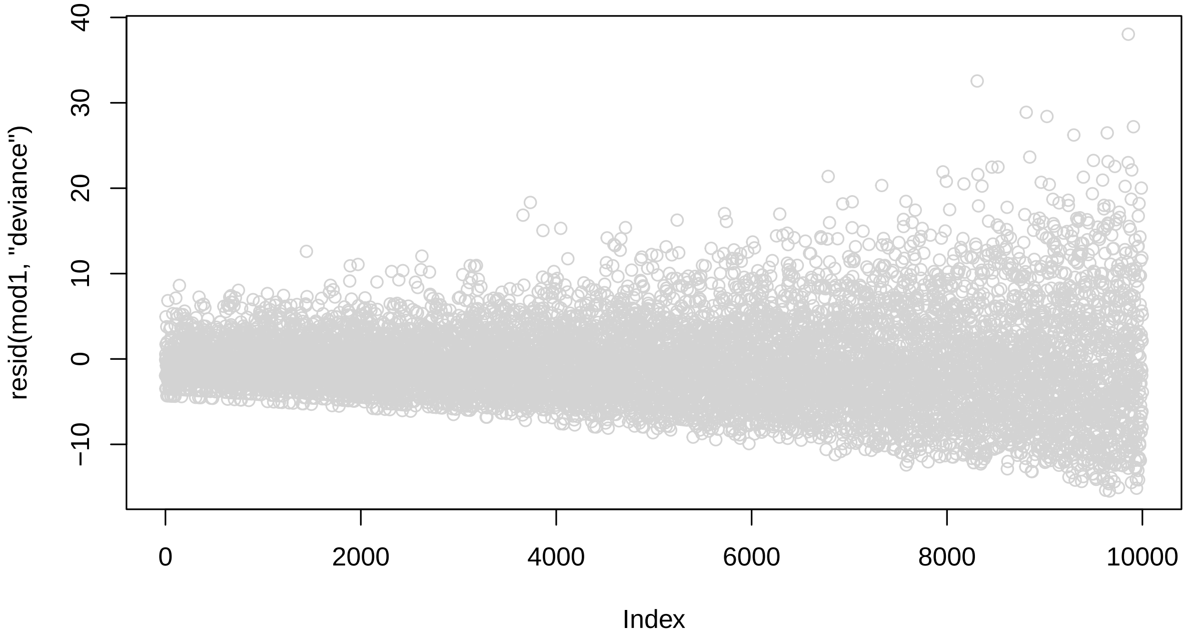
It appears that as the values of x increases, the values of y increases exponentially, which indicates non-linear relationship between the variables. We also observe that the variability of y increases as x increases.

(b)

```
mod1 = glm(y~x, family = "poisson", data = xy.df)
summary(mod1)
```

```
##
## Call:
## glm(formula = y ~ x, family = "poisson", data = xy.df)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -15.462   -3.615   -0.933    2.021   38.035
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -0.059299   0.008938  -6.634 3.26e-11 ***
## x            1.015757   0.002104 482.694  < 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: 526516  on 9999  degrees of freedom
## Residual deviance: 249209  on 9998  degrees of freedom
## AIC: 300846
##
## Number of Fisher Scoring iterations: 5
```

```
plot(resid(mod1, "deviance"), col="lightgrey")
```



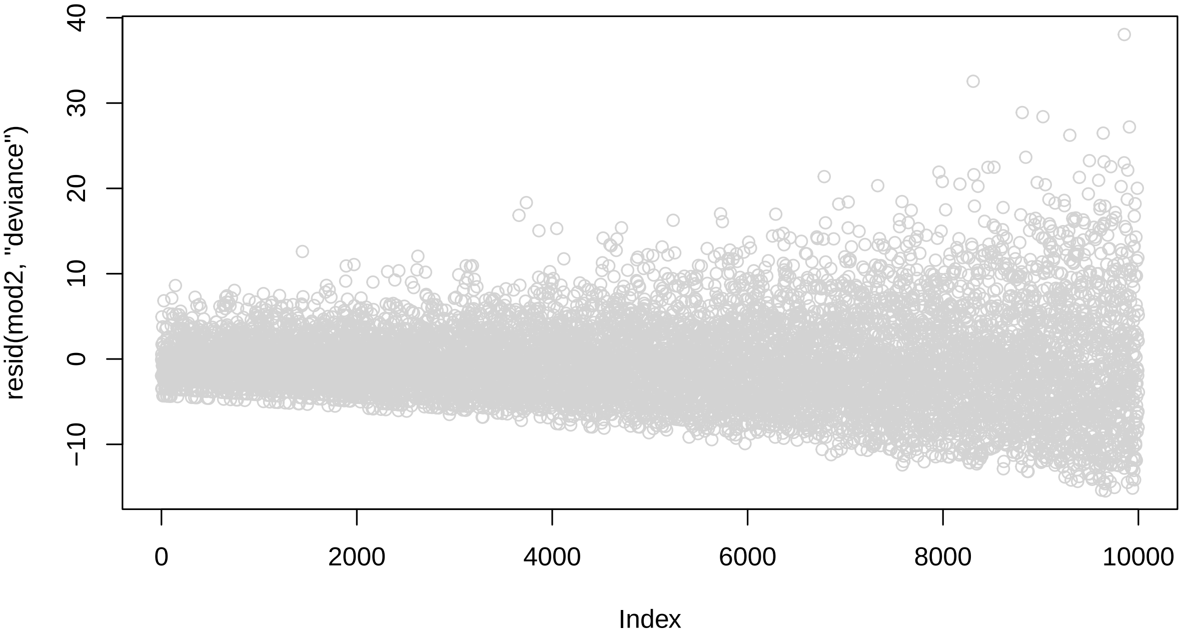
We initially fitted a Poisson model. The deviance residuals appear to be very large, much larger than the acceptable range of -2 to 2 standard deviations from the normal distribution. This suggests there is overdispersion in our data, which means our initial Poisson family model is inadequate.

(c)

```
mod2 = glm(y~x, family = "quasipoisson", data = xy.df)
# quasipoisson coefficients
summary(mod2)$coeff
```

```
##                Estimate Std. Error   t value  Pr(>|t|)
## (Intercept) -0.05929935 0.04653262 -1.274361 0.2025652
## x            1.01575702 0.01095522 92.718959 0.0000000
```

```
plot(resid(mod2, "deviance"), col="lightgrey")
```



```
summary(mod1)
```

```
##
## Call:
## glm(formula = y ~ x, family = "poisson", data = xy.df)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -15.462   -3.615   -0.933    2.021   38.035
```

```
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -0.059299   0.008938  -6.634 3.26e-11 ***
## x            1.015757   0.002104 482.694  < 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: 526516  on 9999  degrees of freedom
## Residual deviance: 249209  on 9998  degrees of freedom
## AIC: 300846
##
## Number of Fisher Scoring iterations: 5
```

```
summary(mod2)
```

```
##
## Call:
## glm(formula = y ~ x, family = "quasipoisson", data = xy.df)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -15.462   -3.615   -0.933    2.021   38.035
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.05930    0.04653  -1.274    0.203
## x            1.01576    0.01096  92.719   <2e-16 ***
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## (Dispersion parameter for quasipoisson family taken to be 27.10232)
##
##     Null deviance: 526516  on 9999  degrees of freedom
## Residual deviance: 249209  on 9998  degrees of freedom
## AIC: NA
##
## Number of Fisher Scoring iterations: 5
```

```
# poisson standard error
(summary(mod1)$coeff[1,2]) #poisson intercept standard error
```

```
## [1] 0.008938286
```

```
(summary(mod1)$coeff[2,2]) #poisson mean standard error
```

```
## [1] 0.00210435
```

```
# quasipoisson standard error calculations
summary(mod1)$coeff[1,2] * sqrt(summary(mod2)$disp) #quasipoisson intercept stanadrd error
```

```
## [1] 0.04653262
```

```
summary(mod1)$coeff[2,2] * sqrt(summary(mod2)$disp) #quasipoisson mean standard error
```

```
## [1] 0.01095522
```

As there is overdispersion, we changed our model to QuasiPoisson to investigate the summary outputs and deviance residuals. It appears that the estimate coefficients remain the same, but the standard errors have increased because our QuasiPoisson model accounts for the overdispersion in our data.

The standard errors for our quasipoisson model are calculated by multiplying the standard error of our initial Poisson model by the square root of the dispersion parameter of our QuasiPoisson model, which results in larger standard errors. This is because there is overdispersion in our data and our dispersion parameter is significantly greater than 1.

However, our deviance residuals remain unchanged from our Poisson model, which are still very large and this implies our secondary model is inadequate.

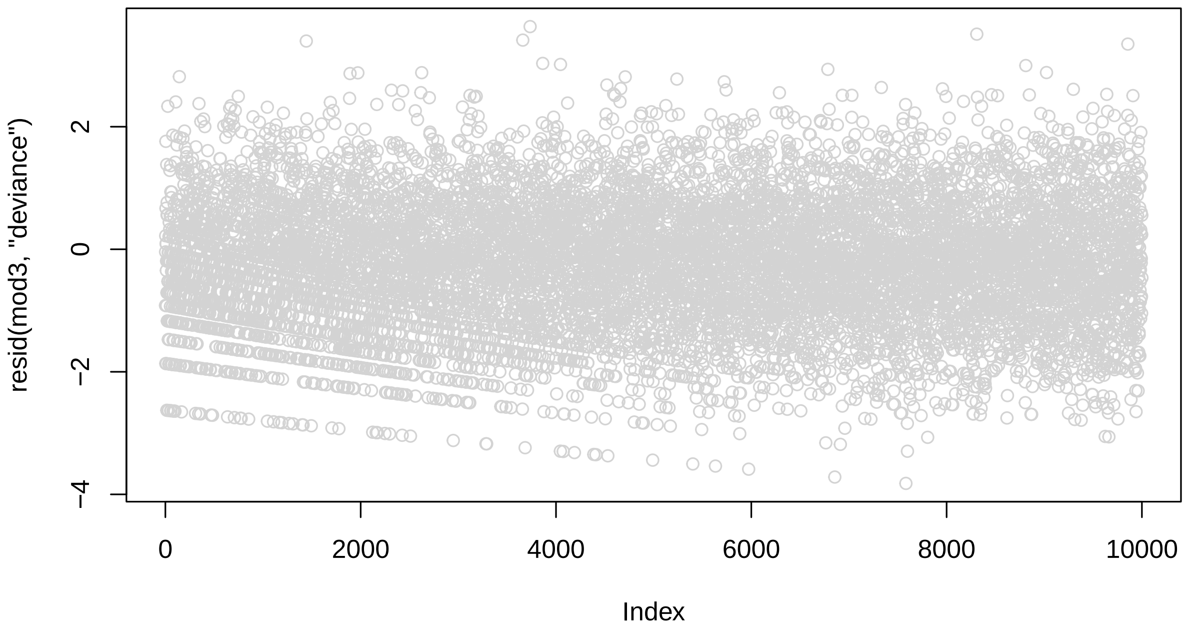
(d)

```
mod3 = glm.nb(y~x, data = xy.df)
summary(mod3)
```

```
## 
## Call:
## glm.nb(formula = y ~ x, data = xy.df, init.theta = 1.981162147, 
##     link = log)
## 
## Deviance Residuals: 
##     Min       1Q   Median       3Q      Max  
## -3.8211  -0.9383  -0.2273   0.4436   3.6340  
## 
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)    
## (Intercept) -0.048794   0.034914  -1.398    0.162    
## x            1.013098   0.009321 108.691   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## (Dispersion parameter for Negative Binomial(1.9812) family taken to be 1)
## 
##     Null deviance: 22336  on 9999  degrees of freedom
## Residual deviance: 10843  on 9998  degrees of freedom
## AIC: 90854
## 
## Number of Fisher Scoring iterations: 1
## 
```

```
##
##                Theta:  1.9812
##            Std. Err.:  0.0286
##
##  2 x log-likelihood:  -90847.8710
```

```
plot(resid(mod3, "deviance"),col="lightgrey")
```

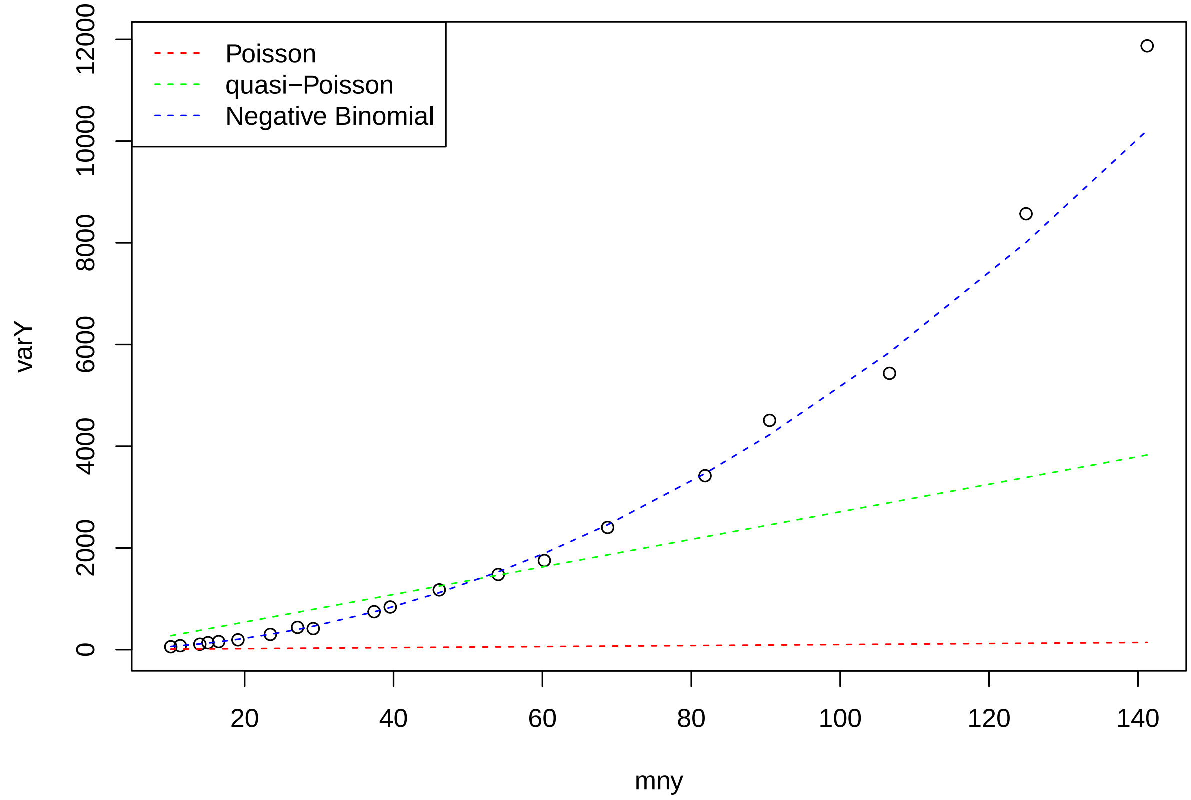


As our previous analysis suggests a much greater degree of overdispersion may be present in our data, we changed our model family to Negative Binomial. Observations of the deviance residuals from our Negative Binomial model appears to be much smaller than our previous models, with majority of the deviance residuals falling within the acceptable range of -2 to 2 standard deviations, which indicates no concerns about our final model.

(e)

```
# cutting data into 20 bins
fs=cut(xy.df$x,20)
mny=tapply(xy.df$y, fs, mean)
varY=tapply(xy.df$y, fs, var)
# plot mean-variance relation for mod1
plot(mny, varY)
lines(mny, mny, lty=2, col="red")
# superimpose two more lines on the plot above
lines(mny, mny*27.1, lty=2, col="green") #quasipoisson
lines(mny, mny + (mny^2)/1.9812, lty=2, col="blue") #negative binomial
# legend
```

```r
legend("topleft", legend=c("Poisson", "quasi-Poisson", "Negative Binomial"),lty=rep(2,3),
col=c("red", "green", "blue"))
```



We created a variance-mean plot of our data, in which we are modelling dispersion. In our observation, it appears as the mean increases, the rate of increase of our variance also increases, which results in a non-linear increasing trend. This suggests over-dispersion in our data, which was best fitted by the Negative Binomial model as shown by the blue dashed line. This is because the Negative Binomial model accounts for a greater degree of over-dispersion as suggested by our dispersion plot. It is apparent that the Poisson and quasi-Poisson models were not adequate as they did not fit the variance-mean relationship of our data.

(f)

Our mathematical model is:

$$log(\mu_i) = \beta_0 + \beta_1 \times x$$

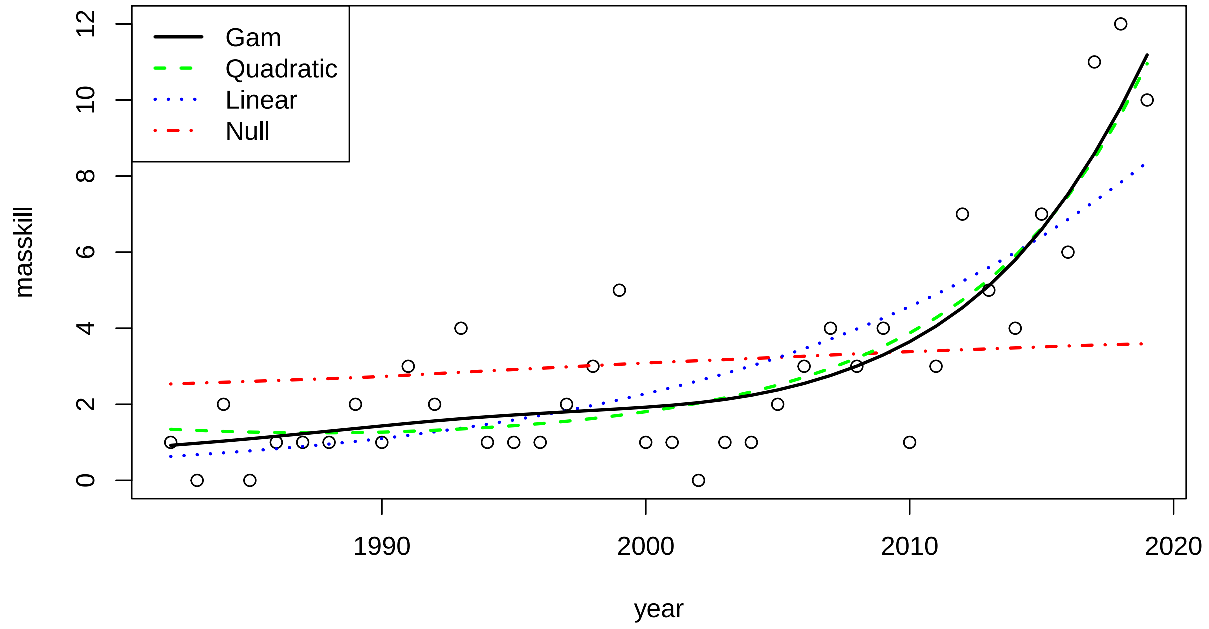
$$Y_i \sim Negative\ Binomial(\mu_i, \theta)$$

Where μ_i is the mean counts and θ is the dispersion parameter.

Question 2

(a)

```r
masskill.df <- read.csv("masskill.csv")[1:38,]
plot(masskill ~ year, data = masskill.df)

# null model
masskill.null <- glm(masskill ~ 1, family="poisson",
offset=log(popn/100), data=masskill.df)

# linear model
masskill.lin <- glm(masskill ~ I(year-1982), family="poisson",
offset=log(popn/100), data=masskill.df)

# poisson quadratic model
masskill.quad <- glm(masskill ~ I(year-1982) + I((year-1982)^2),
family="poisson", offset=log(popn/100),
data=masskill.df)

# poisson GAM model
masskill.gam <- gam(masskill ~ s(I(year-1982))+offset(log(popn/100)),
family="poisson", data=masskill.df)

# plot the model lines
lines(masskill.df$year, exp(predict(masskill.null)), col = "red", lty = 4, lwd = 2)
lines(masskill.df$year, exp(predict(masskill.lin)), col = "blue", lty = 3, lwd = 2)
lines(masskill.df$year, exp(predict(masskill.quad)), col = "green", lty= 2, lwd = 2)
lines(masskill.df$year, exp(predict(masskill.gam)), col = "black", lty = 1, lwd = 2)

legend("topleft", legend = c("Gam", "Quadratic", "Linear", "Null"), col = c("black"
, "green", "blue", "red"), lty = c(1,2,3,4), lwd = 2)
```



(b)

```
# anova
anova(masskill.null, masskill.lin, masskill.quad, test = "Chisq")
```

```
## Analysis of Deviance Table
##
## Model 1: masskill ~ 1
## Model 2: masskill ~ I(year - 1982)
## Model 3: masskill ~ I(year - 1982) + I((year - 1982)^2)
##   Resid. Df Resid. Dev Df Deviance  Pr(>Chi)
## 1        37     79.515
## 2        36     36.821  1   42.693 6.403e-11 ***
## 3        35     31.273  1    5.548    0.0185 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# AIC
AIC(masskill.null, masskill.lin, masskill.quad, masskill.gam)
```

```
##                     df      AIC
## masskill.null 1.000000 179.5839
## masskill.lin  2.000000 138.8905
## masskill.quad 3.000000 135.3427
## masskill.gam  3.669223 133.7228
```

In our nested models, it appears that the residual deviance reduced substantially from 79.5 to 36.8 when a linear term was added to the null model. The residual deviance was further reduced from 36.8 to 31 by adding a quadratic term.

Our null model has the highest AIC value of 179.6 which suggests the least suitability of fit, adding a linear term reduced the AIC value significantly from 179.6 down to 138.9. The AIC value was reduced further down slightly by adding a quadratic term from 138.9 to 135.3. In comparison to the previously mentioned models, the GAM model had the lowest AIC value of 133.7 which suggests the most suitable fit for our data.

(c)

In summary, it appears that for our nested models, adding more explanatory variables such as linear and quadratic terms to our null model decreases the residual deviance and AIC value. This makes sense because adding more explanatory terms can improve our model's ability to explain the variability in our data, indicating a better fit. However, despite the improvements of our nested models, our GAM model still had the lowest residual deviance and AIC value indicating the best fit overall.

Question 3

(a)

```
Moons.df <- read.csv("Moons.csv",stringsAsFactors = TRUE)
# Density var
Moons.df=within(Moons.df,{Density=Mass/Diameter^3})
names(Moons.df)
```

```
## [1] "Name"     "Distance" "Diameter" "Mass"     "Moons"    "Density"
```

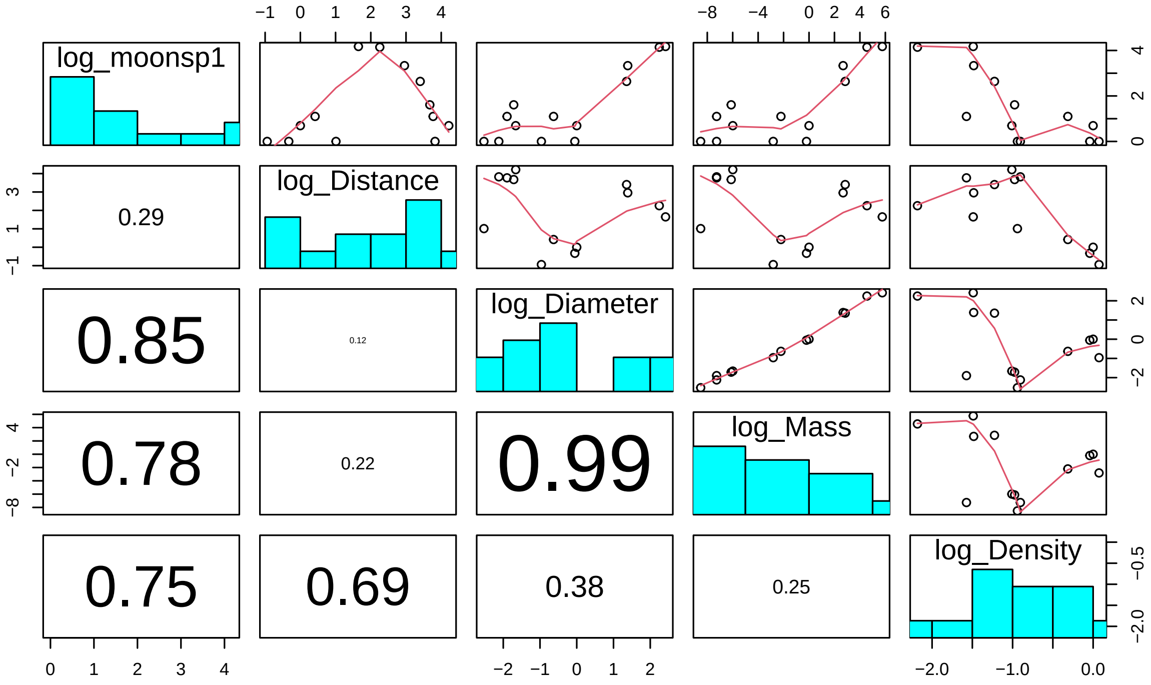
```
# new variables
names(Moons.df)
```

```
## [1] "Name"     "Distance" "Diameter" "Mass"     "Moons"    "Density"
```

```
LogMoons.df=cbind(Moons.df[,c("Name","Moons")],
log(Moons.df[,"Moons"]+1),log(Moons.df[,c(2:4,6)]))
names(LogMoons.df)=c("Name", "Moons", "log_moonsp1",
paste("log", names(Moons.df[c(2:4,6)]), sep="_"))
names(LogMoons.df)
```

```
## [1] "Name"         "Moons"        "log_moonsp1"  "log_Distance" "log_Diameter"
## [6] "log_Mass"     "log_Density"
```

```
pairs20x(LogMoons.df[,-(1:2)])
```



```
Moons.df$log_moonsp1
```

```
## NULL
```

From inspecting our data, it appears that the number of moons for the planets varies greatly from 0 to as large as 64. The variability of the moon counts is such that it could be as low as 2 or 4, and then escalates to 13, 27 and 64. This is described as a non-linear and exponential increase, which necessitates logging the number of moons to establish linearity, hence log_moonsp1 was created.

In observing our paired plots, there appears to be a quadratic relationship between the log number of the moons and log of distance. The plot of log moons and log diameter appears to be non-linear for the first half of the plot, but increases linearly for the second half, we also observe a similar relationship between log moons and log mass. The relationship between log moons and log density appears to be constant for the left side of the plot, decreases linearly in the middle section and remains constant on the right side of the plot.

We obtained a correlation value of 0.99 between log diameter and log mass, which indicates high collinearity between both variables. As they are highly correlated, their effects on log number of moons are indistinguishable due to multicollinearity. This means we can either drop log mass or log diameter, and we decided to drop log diameter in favor of keeping log mass.

(b)

```
fit.all=glm(Moons~(log_Distance+
log_Mass+log_Density)^2+I(log_Distance^2) +I(log_Mass^2),
family=poisson, data=LogMoons.df)

options(na.action = "na.fail")
all.model.fits <- dredge(fit.all)
```

```
## Fixed term is "(Intercept)"
```

```
head(all.model.fits)
```

```
## Global model call: glm(formula = Moons ~ (log_Distance + log_Mass + log_Density)^2 + 
##     I(log_Distance^2) + I(log_Mass^2), family = poisson, data = LogMoons.df)
## ---
## Model selection table
##       (Int) log_Dns log_Dst log_Dst^2 log_Mss log_Mss^2 log_Dns:log_Mss
## 15  -0.2092           3.076   -0.6943  0.1997                          
## 143 -0.4715           3.198   -0.7056  0.3013                          
## 16  -0.1015 -0.1440   2.719   -0.6156  0.2046                          
## 31  -0.3306           3.020   -0.6711  0.2093  0.003577                
## 10   0.8647 -0.9289                    0.3030                          
## 74   0.4610 -1.3780                    0.4817                    0.1478
##     log_Dst:log_Mss df  logLik AICc delta weight
## 15                   4 -22.118 57.2  0.00  0.696
## 143        -0.04383  5 -20.721 60.0  2.78  0.174
## 16                   5 -22.008 62.6  5.35  0.048
## 31                   5 -22.035 62.6  5.40  0.047
## 10                   3 -27.818 64.3  7.07  0.020
## 74                   4 -25.900 64.8  7.56  0.016
## Models ranked by AICc(x)
```

It appears that model number 15 was the best variation of our model with the lowest AIC value. In this variant, the effects of every unit increase in log distance increases the log number of moons by 3.076, every unit increase in log distance squared decreases the log number of moons by 0.6943 and every unit increase in log mass increases the log number of moons by 0.1997.

(c)

Our most parsimonious model has kept the effects of log distance, log distance squared and log mass. In our paired plot, there was a quadratic relationship between log number of moons and log distance, hence a quadratic term for log distance was kept. As the relationship between log moon and log mass doesn't appear to be quadratic, it makes sense that the term for log mass squared was excluded, and only log mass however, was included. Finally, we did not include the effects of log diameter as we observed a high collinearity between log mass and log diameter.