Assignment 2

2022-03-14

EDDA Assignment 2

1

1a.

```
data <- read.csv("nauseatable.txt", sep="")
chitest <- chisq.test(data)
chitest$residuals
```

```
##                        Incidence.of.no.nausea Incidence.of.Nausea
## Chlorpromazine                      1.0540926           -1.270001
## Pentobarbital(100mg)               -1.2179181            1.467383
## Pentobarbital(150mg)               -0.3282848            0.395527
```

From the residuals we can see that Chlorpromazine causes relatively less incidents of nausea while Pentobarbital(100mg) causes a lot more. While Pentobarbital(150mg) is also responsible for some cases of nausea, this is less so than Pentobarbital(100mg).

```
chitest
```

```
##
##  Pearson's Chi-squared test
##
## data:  data
## X-squared = 6.6248, df = 2, p-value = 0.03643
```

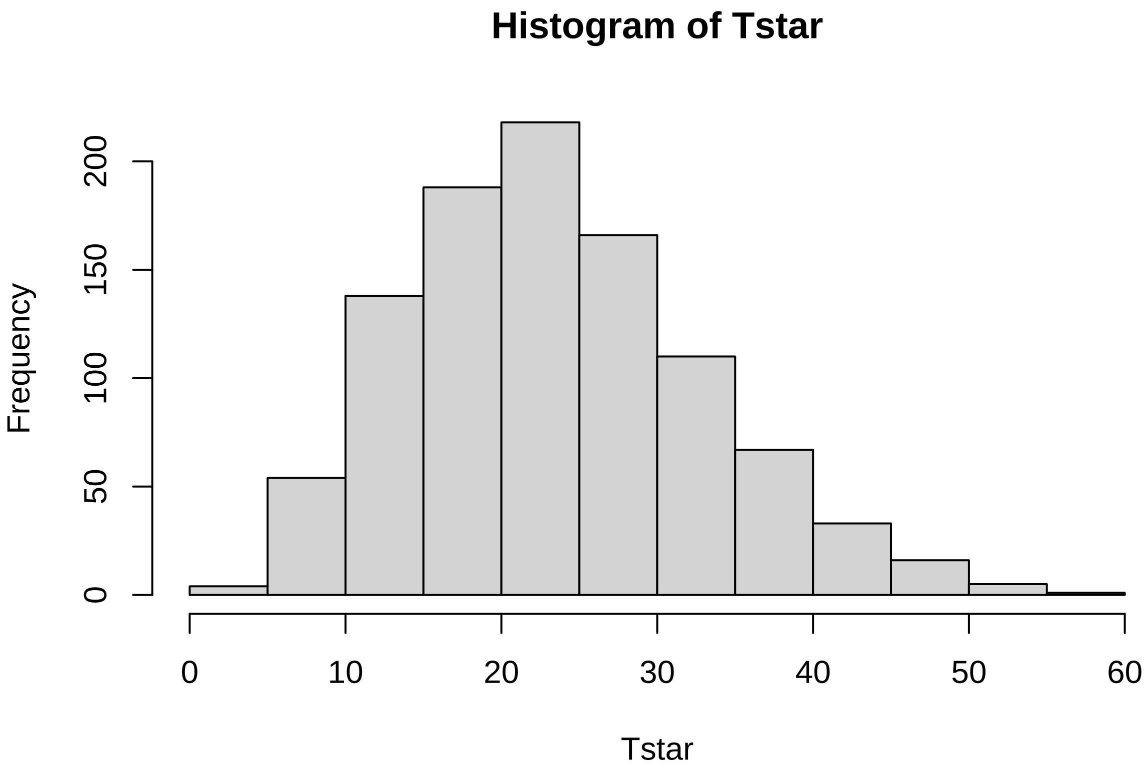
1b.

```
#data <- nauseatable
Nausea<-c()
medicine <- c()
for(i in 1:nrow(data)){
    Nausea <-c(Nausea, rep(1, data[i, 1]))
    Nausea <-c(Nausea, rep(0, data[i, 2]))
    medicine <- c(medicine, rep(i, (data[i, 1]+data[i, 2])))
}

df <- data.frame(medicine, Nausea)
B=1000
Tstar = c()
for (i in 1:B){
  Xstar = df[sample(nrow(df),1000,replace=TRUE),]
  test=chisq.test(table(Xstar))
  Tstar =  c(Tstar, test$statistic)
```

```
}

hist(Tstar)
```




```
chitest$p.value
```

```
## [1] 0.03642928
```

```
chitest$statistic
```

```
## X-squared
##  6.624765
```

```
pl=sum(Tstar<chitest$statistic)/B
pr=sum(Tstar>chitest$statistic)/B
pl
```

```
## [1] 0.013
```

```
pr
```

```
## [1] 0.987
```

```
p_value = 2*min(pl,pr)
p_value
```

```
## [1] 0.026
```

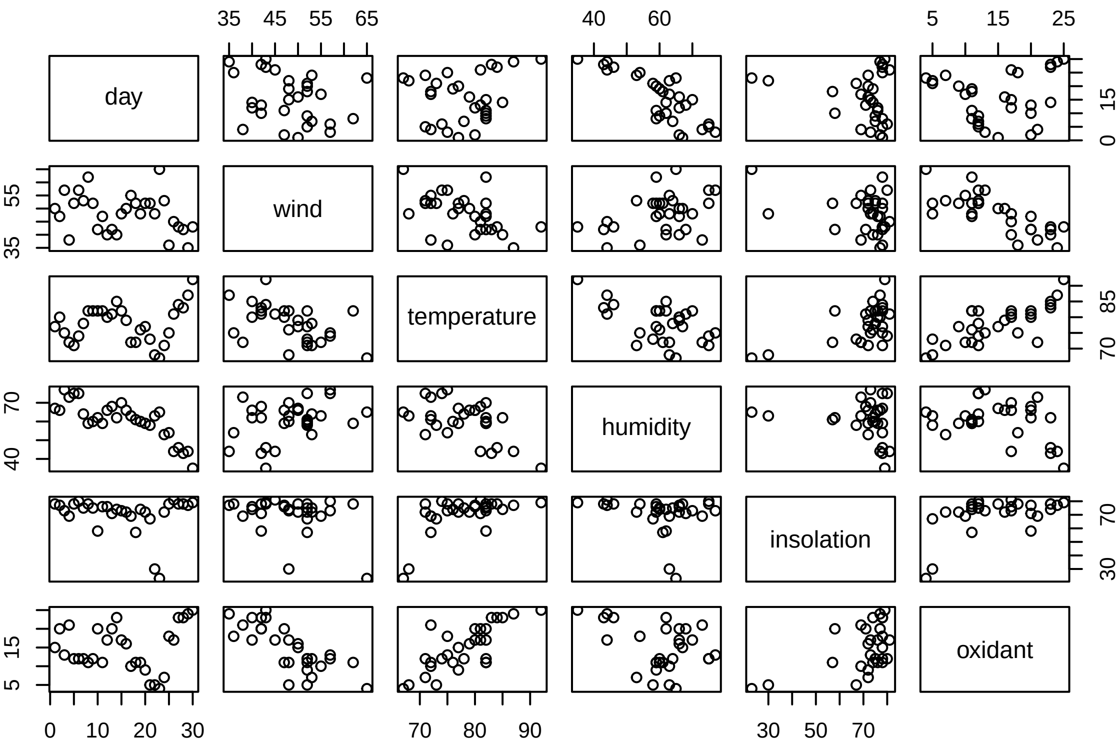
1c

The p-value obtained by the chisquare test for contingency tables is 0.03642928 and that for the permutation test is 0.034 which is slightly lower but by a negligable amount, making them around the same. This is what is to be expected as using different permutations does not change the distribution of the data and the use of the same test statistic in both test results in a similar outcome.

2

```r
airpollution <- read.csv("airpollution.txt", sep="")
View(airpollution)
```

```r
pairs(airpollution)
```




```r
# temperature and oxidant might have a linear relation
```

```r
dogslr = lm(oxidant~wind+temperature+humidity+insolation,data=airpollution)
summary(dogslr)
```

```
##
## Call:
## lm(formula = oxidant ~ wind + temperature + humidity + insolation,
##     data = airpollution)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.5861 -1.0961  0.3512  1.7570  4.0712
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -15.49370   13.50647  -1.147  0.26219
## wind         -0.44291    0.08678  -5.104 2.85e-05 ***
## temperature   0.56933    0.13977   4.073  0.00041 ***
## humidity      0.09292    0.06535   1.422  0.16743
## insolation    0.02275    0.05067   0.449  0.65728
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.92 on 25 degrees of freedom
## Multiple R-squared:  0.798,  Adjusted R-squared:  0.7657
## F-statistic: 24.69 on 4 and 25 DF,  p-value: 2.279e-08
```

p-value for wind and temperature is less than 0.5. Therefore, they are significant.

Cook's Distance with only significant explanatory variables:

```r
dogsnlm = lm(oxidant~wind+temperature,data=airpollution)
round(cooks.distance(dogsnlm),2)
```

```
##    1    2    3    4    5    6    7    8    9   10   11   12   13   14   15   16
## 0.00 0.02 0.05 0.38 0.02 0.04 0.00 0.00 0.04 0.00 0.09 0.02 0.00 0.01 0.00 0.00
##   17   18   19   20   21   22   23   24   25   26   27   28   29   30
## 0.01 0.00 0.01 0.02 0.08 0.21 0.07 0.02 0.00 0.00 0.03 0.03 0.01 0.01
```

```r
max(round(cooks.distance(dogsnlm),2))
```

```
## [1] 0.38
```

No influence point here.

Cook's Distance for entire model :

```r
round(cooks.distance(dogslr),2)
```

```
##    1    2    3    4    5    6    7    8    9   10   11   12   13   14   15   16
## 0.00 0.01 0.03 0.20 0.01 0.02 0.00 0.00 0.03 0.00 0.06 0.04 0.00 0.00 0.01 0.00
##   17   18   19   20   21   22   23   24   25   26   27   28   29   30
## 0.01 0.01 0.01 0.02 0.06 0.33 0.83 0.01 0.00 0.00 0.04 0.08 0.00 0.08
```

```r
max(round(cooks.distance(dogslr),2))
```

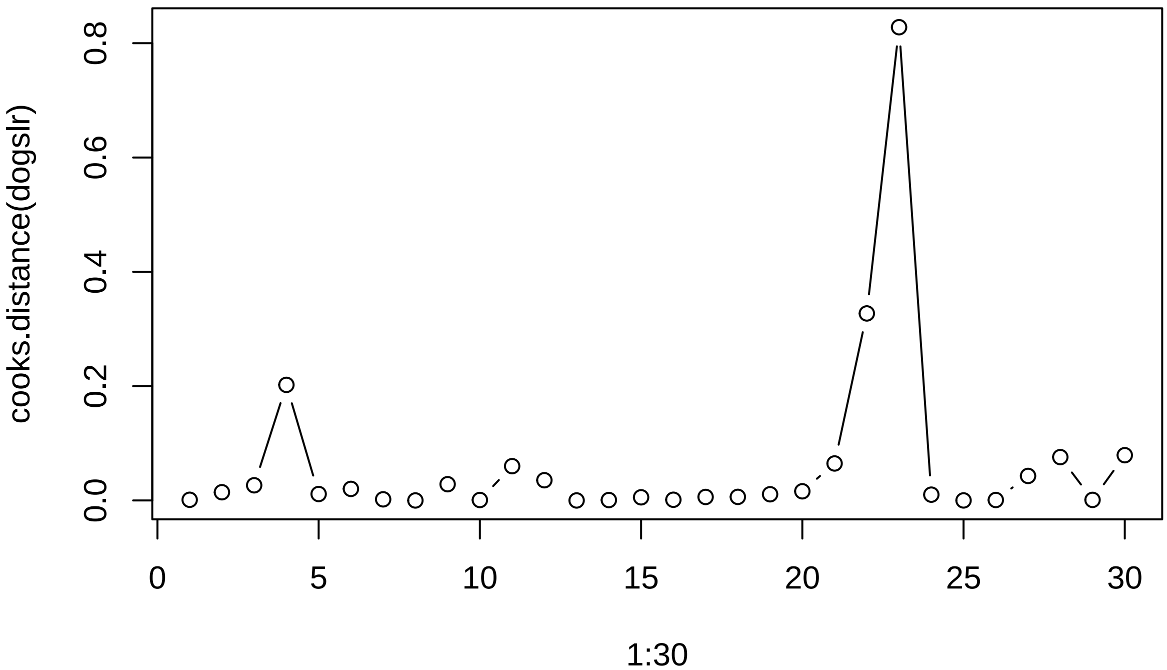
```
## [1] 0.83
```

```r
order(cooks.distance(dogslr))
```

```
##  [1]  8 13 25 14 26 10 29  1 16  7 15 17 18 24 19  5  2 20  6  3  9 12 27 11 21
## [26] 28 30  4 22 23
```

Cook's distance for 23rd data point is close to 1 so it can be an influence point.

```r
plot(1:30,cooks.distance(dogslr),type="b")
```



COLLINEARITY:

A.) Pairwise Linear Correlation

```
round(cor(airpollution),2)
```

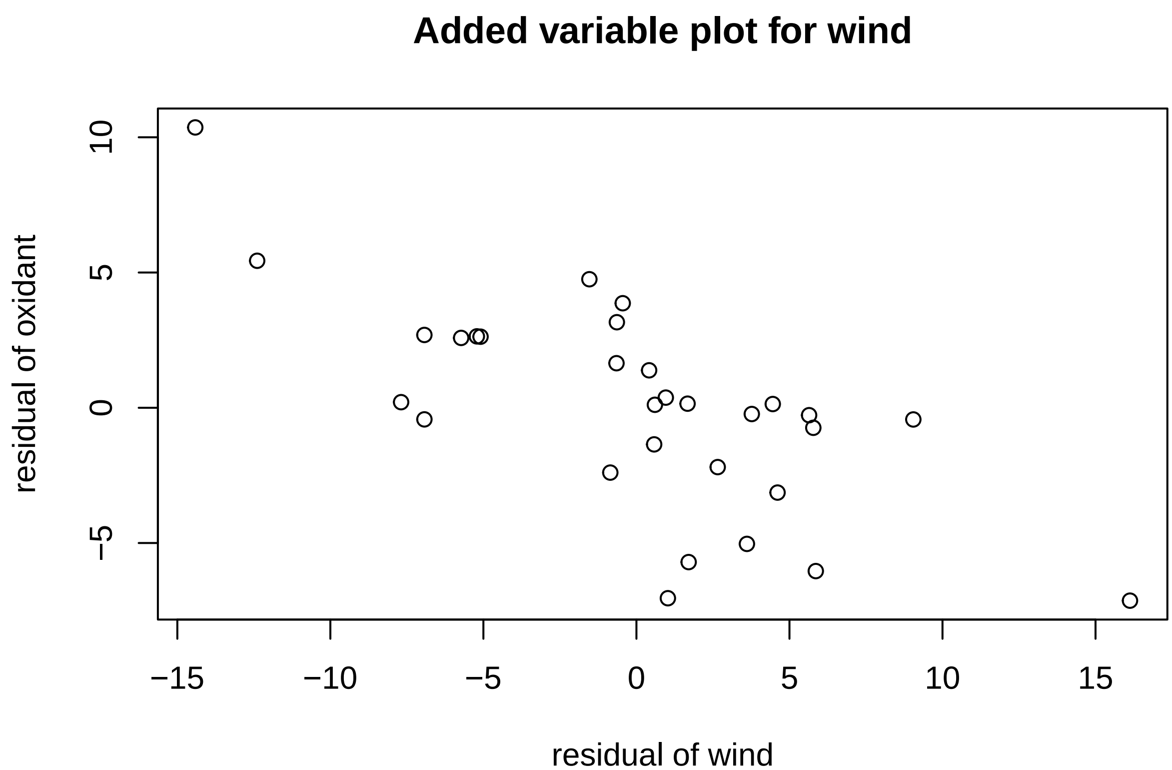
```
##                day  wind temperature humidity insolation oxidant
## day           1.00 -0.28        0.18    -0.81      -0.16    0.10
## wind         -0.28  1.00       -0.50     0.37      -0.32   -0.77
## temperature   0.18 -0.50        1.00    -0.54       0.57    0.76
## humidity     -0.81  0.37       -0.54     1.00      -0.18   -0.35
## insolation   -0.16 -0.32        0.57    -0.18       1.00    0.51
## oxidant       0.10 -0.77        0.76    -0.35       0.51    1.00
```

No significant linear correlation

check multicolinearity of variables :

```
#install.packages("car",dependencies=TRUE)
#library(car)
#vif(dogsnlm)
```

```
x=residuals(lm(wind~temperature+humidity+insolation,data=airpollution))
y=residuals(lm(oxidant~temperature+humidity+insolation,data=airpollution))
plot(x,y,main="Added variable plot for wind", xlab="residual of wind",ylab="residual of oxidant")
```




```
x_new=residuals(lm(wind~temperature,data=airpollution))
y_new=residuals(lm(oxidant~temperature,data=airpollution))
summary(lm(y_new~x_new))
```

```
##
## Call:
## lm(formula = y_new ~ x_new)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.3939 -1.8608  0.5826  1.9461  4.9661
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  6.968e-17  5.288e-01   0.000        1
## x_new       -4.271e-01  8.489e-02  -5.031 2.55e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.896 on 28 degrees of freedom
## Multiple R-squared:  0.4748, Adjusted R-squared:  0.456
## F-statistic: 25.31 on 1 and 28 DF,  p-value: 2.549e-05
```

```
summary(dogsnlm)
```

```
##
## Call:
## lm(formula = oxidant ~ wind + temperature, data = airpollution)
##
## Residuals:
```

```
##     Min      1Q  Median      3Q     Max 
## -6.3939 -1.8608  0.5826  1.9461  4.9661 
## 
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept) -5.20334   11.11810  -0.468    0.644    
## wind        -0.42706    0.08645  -4.940 3.58e-05 ***
## temperature  0.52035    0.10813   4.812 5.05e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 2.95 on 27 degrees of freedom
## Multiple R-squared:  0.7773, Adjusted R-squared:  0.7608 
## F-statistic: 47.12 on 2 and 27 DF,  p-value: 1.563e-09
```

Step-up model:

```
summary(lm(oxidant~wind,data=airpollution))
```

```
## 
## Call:
## lm(formula = oxidant ~ wind, data = airpollution)
## 
## Residuals:
##     Min      1Q  Median      3Q     Max 
## -9.9266 -2.5923  0.2065  2.6636  6.9077 
## 
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept)  45.3171     4.8976   9.253 5.19e-10 ***
## wind         -0.6331     0.1005  -6.300 8.20e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 3.948 on 28 degrees of freedom
## Multiple R-squared:  0.5863, Adjusted R-squared:  0.5715 
## F-statistic: 39.68 on 1 and 28 DF,  p-value: 8.205e-07
```

```
summary(lm(oxidant~temperature,data=airpollution))
```

```
## 
## Call:
## lm(formula = oxidant ~ temperature, data = airpollution)
## 
## Residuals:
##     Min      1Q  Median      3Q     Max 
## -6.9400 -2.2138  0.3775  2.5550 10.9099 
## 
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept) -46.4292     9.9542  -4.664 6.94e-05 ***
## temperature   0.7850     0.1273   6.168 1.17e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 3.997 on 28 degrees of freedom
```

```
## Multiple R-squared:  0.576,  Adjusted R-squared:  0.5609
## F-statistic: 38.04 on 1 and 28 DF,  p-value: 1.167e-06
```

```r
summary(lm(oxidant~humidity,data=airpollution))
```

```
##
## Call:
## lm(formula = oxidant ~ humidity, data = airpollution)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -10.3358  -4.0749   0.8782   4.7800   8.7957
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  27.4446     6.4368   4.264 0.000206 ***
## humidity     -0.2088     0.1049  -1.991 0.056317 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.745 on 28 degrees of freedom
## Multiple R-squared:  0.124,  Adjusted R-squared:  0.09273
## F-statistic: 3.964 on 1 and 28 DF,  p-value: 0.05632
```

```r
summary(lm(oxidant~insolation,data=airpollution))
```

```
##
## Call:
## lm(formula = oxidant ~ insolation, data = airpollution)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.9723 -4.4841 -0.3281  4.7631  8.2686
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.43279    5.32967  -0.269  0.79003
## insolation   0.22993    0.07424   3.097  0.00441 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.297 on 28 degrees of freedom
## Multiple R-squared:  0.2552, Adjusted R-squared:  0.2286
## F-statistic: 9.592 on 1 and 28 DF,  p-value: 0.004411
```

wind has the highest R-square value and a p-value < 0.05 . Therefore, we add this explanatory variable to or model.

```r
summary(lm(oxidant~wind+temperature,data=airpollution))
```

```
##
## Call:
## lm(formula = oxidant ~ wind + temperature, data = airpollution)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.3939 -1.8608  0.5826  1.9461  4.9661
```

```
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -5.20334   11.11810  -0.468    0.644
## wind        -0.42706    0.08645  -4.940 3.58e-05 ***
## temperature  0.52035    0.10813   4.812 5.05e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.95 on 27 degrees of freedom
## Multiple R-squared:  0.7773, Adjusted R-squared:  0.7608
## F-statistic: 47.12 on 2 and 27 DF,  p-value: 1.563e-09
```

```
summary(lm(oxidant~wind+humidity,data=airpollution))
```

```
##
## Call:
## lm(formula = oxidant ~ wind + humidity, data = airpollution)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.8120 -2.2808  0.3433  3.0476  5.8757
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 46.91570    5.68573   8.251 7.38e-09 ***
## wind        -0.60955    0.10971  -5.556 6.86e-06 ***
## humidity    -0.04516    0.07866  -0.574    0.571
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.996 on 27 degrees of freedom
## Multiple R-squared:  0.5913, Adjusted R-squared:  0.561
## F-statistic: 19.53 on 2 and 27 DF,  p-value: 5.674e-06
```

```
summary(lm(oxidant~wind+insolation,data=airpollution))
```

```
##
## Call:
## lm(formula = oxidant ~ wind + insolation, data = airpollution)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.2119 -2.7198  0.4815  2.8733  6.2012
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 32.32615    6.97098   4.637 8.07e-05 ***
## wind        -0.55639    0.09778  -5.690 4.81e-06 ***
## insolation   0.13161    0.05383   2.445   0.0213 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.638 on 27 degrees of freedom
## Multiple R-squared:  0.6613, Adjusted R-squared:  0.6362
```

```
## F-statistic: 26.36 on 2 and 27 DF,  p-value: 4.491e-07
```

Next we add temperature. It has the highest R square value and the variable is significant.

```
summary(lm(oxidant~wind+temperature+humidity,data=airpollution))
```

```
##
## Call:
## lm(formula = oxidant ~ wind + temperature + humidity, data = airpollution)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.5887 -1.1686  0.1978  1.9004  4.1544
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -16.60697   13.07154  -1.270    0.215
## wind         -0.44620    0.08513  -5.241 1.78e-05 ***
## temperature   0.60190    0.11764   5.117 2.47e-05 ***
## humidity      0.09850    0.06316   1.559    0.131
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.874 on 26 degrees of freedom
## Multiple R-squared:  0.7964, Adjusted R-squared:  0.7729
## F-statistic: 33.89 on 3 and 26 DF,  p-value: 3.904e-09
```

```
summary(lm(oxidant~wind+temperature+insolation,data=airpollution))
```

```
##
## Call:
## lm(formula = oxidant ~ wind + temperature + insolation, data = airpollution)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -6.407 -2.056  1.012  1.760  4.792
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -4.45496   11.26714  -0.395 0.695778
## wind        -0.42353    0.08737  -4.848 5.02e-05 ***
## temperature  0.47558    0.12564   3.785 0.000816 ***
## insolation   0.03646    0.05071   0.719 0.478636
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.976 on 26 degrees of freedom
## Multiple R-squared:  0.7816, Adjusted R-squared:  0.7565
## F-statistic: 31.02 on 3 and 26 DF,  p-value: 9.583e-09
```

None of the other explanatory variables - humidity and insolation are significant. Therefore our linear model has only two explanatory variables - wind and temperature.

```
lm_up = lm(oxidant~wind+temperature,data=airpollution)
summary(lm_up)
```

```
##
```

```
## Call:
## lm(formula = oxidant ~ wind + temperature, data = airpollution)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.3939 -1.8608  0.5826  1.9461  4.9661
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -5.20334   11.11810  -0.468    0.644
## wind        -0.42706    0.08645  -4.940 3.58e-05 ***
## temperature  0.52035    0.10813   4.812 5.05e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.95 on 27 degrees of freedom
## Multiple R-squared:  0.7773, Adjusted R-squared:  0.7608
## F-statistic: 47.12 on 2 and 27 DF,  p-value: 1.563e-09
```

oxidant = -5.20334 + (-0.42706)*wind + (0.52035)temperature + error

STEP DOWN :

```r
summary(lm(oxidant~wind+temperature+humidity+insolation,data=airpollution))
```

```
##
## Call:
## lm(formula = oxidant ~ wind + temperature + humidity + insolation,
##     data = airpollution)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.5861 -1.0961  0.3512  1.7570  4.0712
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -15.49370   13.50647  -1.147  0.26219
## wind         -0.44291    0.08678  -5.104 2.85e-05 ***
## temperature   0.56933    0.13977   4.073  0.00041 ***
## humidity      0.09292    0.06535   1.422  0.16743
## insolation    0.02275    0.05067   0.449  0.65728
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.92 on 25 degrees of freedom
## Multiple R-squared:  0.798,  Adjusted R-squared:  0.7657
## F-statistic: 24.69 on 4 and 25 DF,  p-value: 2.279e-08
```

Insolation has the largest p-value and it is greater than 0.05. Therefore, we remove Insolation variable from the linear model.

```r
summary(lm(oxidant~wind+temperature+humidity,data=airpollution))
```

```
##
## Call:
## lm(formula = oxidant ~ wind + temperature + humidity, data = airpollution)
##
```

```
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.5887 -1.1686  0.1978  1.9004  4.1544
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -16.60697   13.07154  -1.270    0.215
## wind         -0.44620    0.08513  -5.241 1.78e-05 ***
## temperature   0.60190    0.11764   5.117 2.47e-05 ***
## humidity      0.09850    0.06316   1.559    0.131
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.874 on 26 degrees of freedom
## Multiple R-squared:  0.7964, Adjusted R-squared:  0.7729
## F-statistic: 33.89 on 3 and 26 DF,  p-value: 3.904e-09
```

Largest p-value of humidity is > 0.05. Therefore, we remove humidity from the model.

```
summary(lm(oxidant~wind+temperature,data=airpollution))
```

```
##
## Call:
## lm(formula = oxidant ~ wind + temperature, data = airpollution)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.3939 -1.8608  0.5826  1.9461  4.9661
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -5.20334   11.11810  -0.468    0.644
## wind        -0.42706    0.08645  -4.940 3.58e-05 ***
## temperature  0.52035    0.10813   4.812 5.05e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.95 on 27 degrees of freedom
## Multiple R-squared:  0.7773, Adjusted R-squared:  0.7608
## F-statistic: 47.12 on 2 and 27 DF,  p-value: 1.563e-09
```

oxidant = -5.20334 + (-0.42706)*wind + (0.52035)temperature + error

We get the same model from both step up and step down approaches. Therefore, we don't need to compare the models.

```
linear_model = lm(oxidant~wind+temperature,data=airpollution)
x_newdata = data.frame(wind=33,temperature=54)
predict(linear_model,x_newdata,interval="confidence")
```

```
##       fit      lwr      upr
## 1 8.80281 1.656548 15.94907
```

```
predict(linear_model,x_newdata,interval="prediction")
```

```
##       fit        lwr      upr
## 1 8.80281 -0.5617877 18.16741
```

```
## what does negative value imply over here ?
```

3

Load fruitflies.txt and add loglongevity column

```
fruitflies = read.csv("fruitflies.txt", sep="")
fruitflies = fruitflies[sample(nrow(fruitflies)),]
loglong = log(fruitflies["longevity"])
fruitflies['loglongevity'] = loglong
```

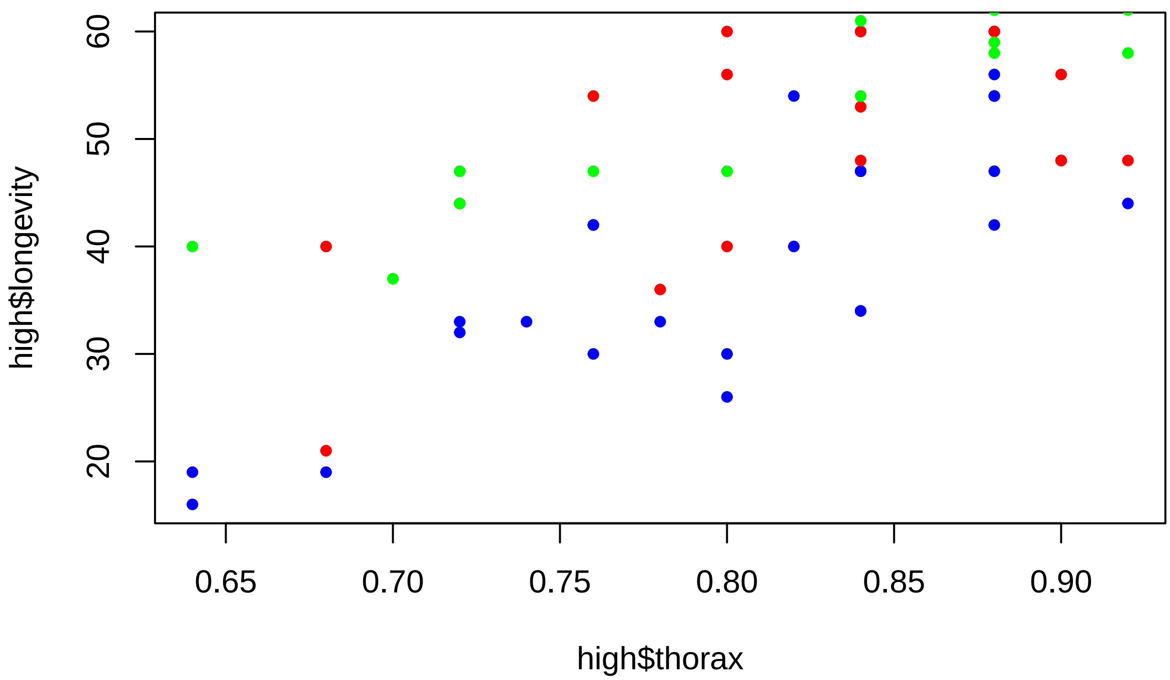
3a.

We want to know if there is a statistically significant difference between three groups, based on one variable (log longevity). Therefore, we use a one-way ANOVA test.

Firs, we create an informative graph, plotting the thorax variable against the log longevity variable for each type of activity

```
high = subset(fruitflies, activity == 'high')
low = subset(fruitflies, activity == 'low')
isolated = subset(fruitflies, activity == 'isolated')

plot(high$thorax, high$longevity, col='blue', pch=20)
points(low$thorax, low$longevity, col='red', pch=20)
points(isolated$thorax, isolated$longevity, col='green', pch=20)
```



Then, we perform the anova test:

```
fruitflies_model <- aov(loglongevity ~ activity, data=fruitflies)
fruitflies_model
```

```
## Call:
##    aov(formula = loglongevity ~ activity, data = fruitflies)
```

```
##
## Terms:
##                 activity Residuals
## Sum of Squares  3.666493  6.796579
## Deg. of Freedom        2        72
##
## Residual standard error: 0.3072408
## Estimated effects may be unbalanced
```

```
summary(fruitflies_model)
```

```
##             Df Sum Sq Mean Sq F value  Pr(>F)
## activity     2  3.666  1.8332   19.42 1.8e-07 ***
## Residuals   72  6.797  0.0944
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

We can see that the p-value for activity is well below 0.05. Therefore, the sexual activity *influences* the longevity.

To estimate the longevities for each of the three conditions, we take the means:

```
mean(high$longevity)
```

```
## [1] 38.72
```

```
mean(low$longevity)
```

```
## [1] 56.76
```

```
mean(isolated$longevity)
```

```
## [1] 63.56
```

Therefore, the longevity for *high activity* is 38.72, for *low activity* is 56.76, for *isolated activity* is 63.56

This means that the higher the sexual activity, the shorter the fruitflies live.

3b.

We now include the thorax variable, which is a numerical variable. That means we now have a factor variable, a numerical variable and a numerical outcome, which means that an ANCOVA-test is appropriate. We use the drop1 function to make sure the result is correct even if the ANCOVA test is not balanced

```
fruitflies$activity = as.factor(fruitflies$activity)
fruitflies_model2 = drop1(lm(loglongevity ~ activity + thorax, data=fruitflies), test="F")
fruitflies_model2
```

```
## Single term deletions
##
## Model:
## loglongevity ~ activity + thorax
##          Df Sum of Sq    RSS     AIC F value    Pr(>F)
## <none>                2.9180 -235.50
## activity  2    2.1129 5.0309 -198.64  25.705 4.000e-09 ***
## thorax    1    3.8786 6.7966 -174.08  94.374 1.139e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
summary(fruitflies_model2)
```

```
##        Df         Sum of Sq          RSS              AIC        
##  Min.   :1.00   Min.   :2.113   Min.   :2.918   Min.   :-235.5  
##  1st Qu.:1.25   1st Qu.:2.554   1st Qu.:3.974   1st Qu.:-217.1  
##  Median :1.50   Median :2.996   Median :5.031   Median :-198.6  
##  Mean   :1.50   Mean   :2.996   Mean   :4.915   Mean   :-202.7  
##  3rd Qu.:1.75   3rd Qu.:3.437   3rd Qu.:5.914   3rd Qu.:-186.4  
##  Max.   :2.00   Max.   :3.879   Max.   :6.797   Max.   :-174.1  
##  NA's   :1      NA's   :1                                       
##     F value          Pr(>F)  
##  Min.   :25.71   Min.   :0   
##  1st Qu.:42.87   1st Qu.:0   
##  Median :60.04   Median :0   
##  Mean   :60.04   Mean   :0   
##  3rd Qu.:77.21   3rd Qu.:0   
##  Max.   :94.37   Max.   :0   
##  NA's   :1       NA's   :1
```

From the p-values, we can see that both the thorax variable and the activity variable influence the longevity, as these p-values are both well below 0.05.

If we take the maximum and minimum thorax lengths, we get the following thorax lengths and longevities:

```r
max_thorax = max(fruitflies['thorax'])
min_thorax = min(fruitflies['thorax'])
max_thorax_ff = subset(fruitflies, thorax == max_thorax)
min_thorax_ff = subset(fruitflies, thorax == min_thorax)

high_max = subset(high, thorax == max(high['thorax']))
high_min = subset(high, thorax == min(high['thorax']))
low_max = subset(low, thorax == max(low['thorax']))
low_min = subset(low, thorax == min(low['thorax']))
isolated_max = subset(isolated, thorax == max(isolated['thorax']))
isolated_min = subset(isolated, thorax == min(isolated['thorax']))

print("High")
```

```
## [1] "High"
```

```r
mean(high_max$longevity)
```

```
## [1] 44
```

```r
mean(high_min$longevity)
```

```
## [1] 17.5
```

```r
print("Low")
```

```
## [1] "Low"
```

```r
mean(low_max$longevity)
```

```
## [1] 58
```

```r
mean(low_min$longevity)
```

```
## [1] 30.5
```

```
print("Isolated")
```

```
## [1] "Isolated"
```

```
mean(isolated_max$longevity)
```

```
## [1] 75
```

```
mean(isolated_min$longevity)
```

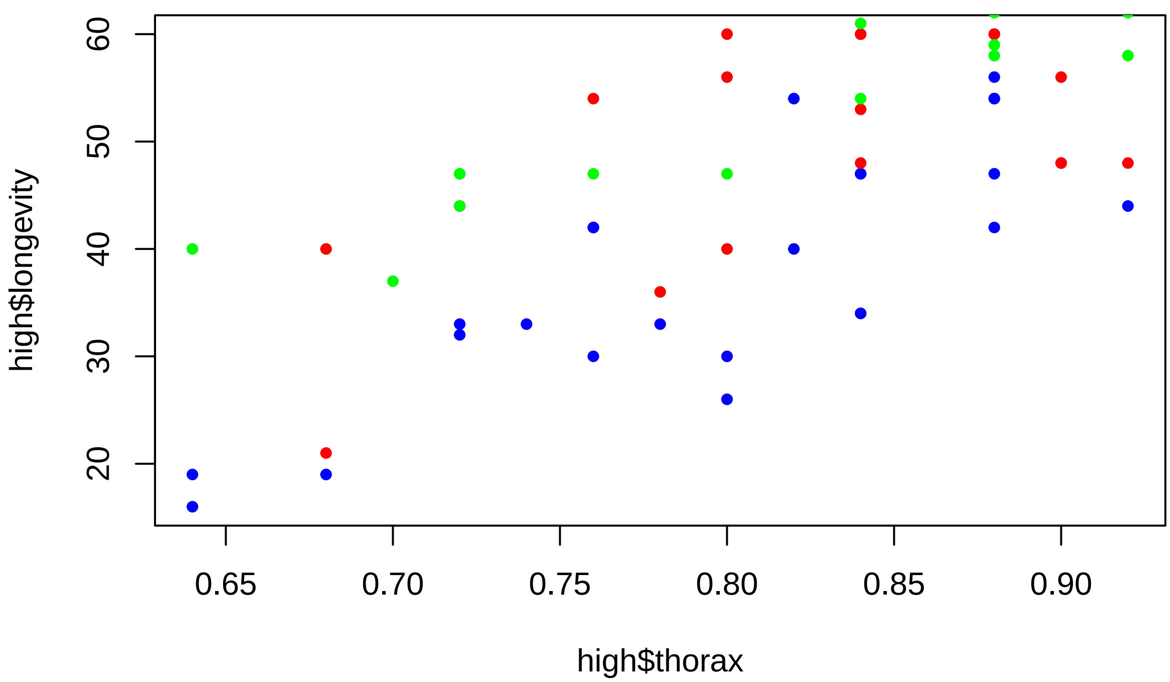
```
## [1] 40
```

We can see that for both the maximum and minimum values of thorax length, the longevity is highest for *isolated activity*, lower for *low activity* and lowest for *high activity*. Even so, the longevity for the maximum thorax value is still higher for high activity than it is for the minimum thorax value for isolated activity. This means that sexual activity decreases longevity.

3c.

We once again plot the longevity against the thorax length:

```
plot(high$thorax, high$longevity, col='blue', pch=20)
points(low$thorax, low$longevity, col='red', pch=20)
points(isolated$thorax, isolated$longevity, col='green', pch=20)
```



Here, we can see a clear increase in longevity when the thorax length increases, for each of the three activity types. This indicates that the the higher the thorax length, the higher the longevity. To check if this is actually statistically significant for each of the three activity types, we use a one-way ANOVA test for each of the activities:

```
high_model <- aov(loglongevity ~ thorax, data=high)
summary(high_model)
```

```
##             Df Sum Sq Mean Sq F value   Pr(>F)    
## thorax       1 2.0760  2.0760   53.61 1.89e-07 ***
## Residuals   23 0.8906  0.0387
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
print('-------------------------------------------')
```

```
## [1] "-------------------------------------------"
```

```
low_model <- aov(loglongevity ~ thorax, data=low)
summary(low_model)
```

```
##             Df Sum Sq Mean Sq F value   Pr(>F)
## thorax       1  1.006  1.0057   19.81 0.000183 ***
## Residuals   23  1.168  0.0508
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
print('-------------------------------------------')
```

```
## [1] "-------------------------------------------"
```

```
isolated_model <- aov(loglongevity ~ thorax, data=isolated)
summary(isolated_model)
```

```
##             Df Sum Sq Mean Sq F value   Pr(>F)
## thorax       1 0.9511  0.9511      31 1.15e-05 ***
## Residuals   23 0.7055  0.0307
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

In the results, we can see that the p-values for each activity type are below 0.05. Therefore, the difference is statistically significant for each activity type.

3d.

Since both the thorax length and the activity type influence the longevity, we prefer the analysis with thorax length, as this is a more complete analysis. However, since the thorax length influences the longevity the same way for each activity type, the analysis without the thorax length is still correct. If the thorax length would influence the longevity differently for each activity type, then it would be incorrect to analyse the data without the thorax length.

3e.

We now perform the ANCOVA analysis, but with the longevity as the response variable instead of the log longevity:

```
fruitflies_model2 = drop1(lm(longevity ~ activity + thorax, data=fruitflies), test="F")
fruitflies_model2
```

```
## Single term deletions
##
## Model:
## longevity ~ activity + thorax
##          Df Sum of Sq   RSS    AIC F value    Pr(>F)
## <none>                 7673 355.10
## activity  2    4966.7 12640 388.53  22.979 2.016e-08 ***
## thorax    1    7686.8 15360 405.15  71.127 2.624e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(fruitflies_model2)
```

```
##        Df         Sum of Sq          RSS              AIC            F value     
##  Min.   :1.00   Min.   :4967   Min.   : 7673   Min.   :355.1   Min.   :22.98  
##  1st Qu.:1.25   1st Qu.:5647   1st Qu.:10156   1st Qu.:371.8   1st Qu.:35.02  
##  Median :1.50   Median :6327   Median :12640   Median :388.5   Median :47.05  
##  Mean   :1.50   Mean   :6327   Mean   :11891   Mean   :382.9   Mean   :47.05  
##  3rd Qu.:1.75   3rd Qu.:7007   3rd Qu.:14000   3rd Qu.:396.8   3rd Qu.:59.09  
##  Max.   :2.00   Max.   :7687   Max.   :15360   Max.   :405.2   Max.   :71.13  
##  NA's   :1      NA's   :1                                      NA's   :1      
##      Pr(>F)  
##  Min.   :0  
##  1st Qu.:0  
##  Median :0  
##  Mean   :0  
##  3rd Qu.:0  
##  Max.   :0  
##  NA's   :1  
```

The p-values are still well below 0.05, so the results are still statistically significant. However, the Sum Sq and Mean Sq values are now so high that they are meaningless. Therefore, it is wise to use the log longevity as the response variable and not the longevity.

4

PART a :

```
psidata <- read.csv("psi.txt", sep="")

psidata$gpa=as.numeric(psidata$gpa)
psidata$psi=as.factor(psidata$psi)
is.factor(psidata$psi)
```

```
## [1] TRUE
```

```
glm_model=glm(passed~psi*gpa,data=psidata,family=binomial)
anova(glm_model,test="Chisq") # only the last p value is relevant
```

```
## Analysis of Deviance Table
##
## Model: binomial, link: logit
##
## Response: passed
##
## Terms added sequentially (first to last)
##
##
##         Df Deviance Resid. Df Resid. Dev Pr(>Chi)   
## NULL                       31     41.183            
## psi      1   5.8418        30     35.342 0.015650 * 
## gpa      1   9.0885        29     26.253 0.002572 **
## psi:gpa  1   1.8725        28     24.381 0.171189   
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

p-value of interaction between psi and gpa is greater than 0.05. Therefore, the interaction is not significant between factor psi and predictor grade.

```
glm2 = glm(passed~psi+gpa,data=psidata,family=binomial)
drop1(glm2,test="Chisq")
```

```
## Single term deletions
##
## Model:
## passed ~ psi + gpa
##        Df Deviance    AIC    LRT Pr(>Chi)
## <none>      26.253 32.253
## psi     1   32.418 36.418 6.1647 0.013033 *
## gpa     1   35.342 39.342 9.0885 0.002572 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

p-value for psi and gpa is less than 0.05. Therefore, they are significant.

```
summary(glm2)
```

```
##
## Call:
## glm(formula = passed ~ psi + gpa, family = binomial, data = psidata)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.8396  -0.6282  -0.3045   0.5629   2.0378
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -11.602      4.213  -2.754  0.00589 **
## psi1           2.338      1.041   2.246  0.02470 *
## gpa            3.063      1.223   2.505  0.01224 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 41.183  on 31  degrees of freedom
## Residual deviance: 26.253  on 29  degrees of freedom
## AIC: 32.253
##
## Number of Fisher Scoring iterations: 5
```

-11.602 + 2.338 + 3.063*x psi has a positive effect on probability of success(i.e. pass = 1). Therefore, it works. It increases the odds by e^2.338 = 10.36

```
is.factor(psidata$psi)
```

```
## [1] TRUE
```

PART b:

```
newdata=data.frame(psi=1,gpa=3)
newdata$psi <- as.factor(newdata$psi)
predict(glm2,newdata,type="response")
```

```
##         1
```

```
## 0.4815864
```

```
newdata2=data.frame(psi=0,gpa=3)
newdata2$psi <- as.factor(newdata2$psi)
predict(glm2,newdata2,type="response")
```

```
##          1
## 0.08230274
```

Therefore, higher grade of 3 is more probable with psi than without psi.

PART c:

-11.602 + 2.338 + 3.063*x psi has a positive effect on probability of success(i.e. pass = 1). Therefore, it works. It increases the odds by e^2.338 = 10.36 Therefore, it increases the success probability(of passing the exam) by 10 times with psi as compared to without psi.This number is not depenedent on gpa.

PART d: CONTINGENCY TABLES: Check if psi and passed is independent or not.

```
tot=xtabs(~psi+passed,data=psidata)
tot
```

```
##    passed
## psi  0  1
##   0 15  3
##   1  6  8
```

```
z=chisq.test(tot); z
```

```
## Warning in chisq.test(tot): Chi-squared approximation may be incorrect
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  tot
## X-squared = 4.0657, df = 1, p-value = 0.04376
```

```
chisq.test(tot,simulate.p.value=TRUE)
```

```
##
##  Pearson's Chi-squared test with simulated p-value (based on 2000
##  replicates)
##
## data:  tot
## X-squared = 5.7192, df = NA, p-value = 0.02649
```

they are not independent as p-value is less than 0.05. Therefore, passed and psi are dependent.

It is a 2*2 table, therefore we can also use fischer's exact test. From this test we can get the exact p-value.

```
fisher.test(tot)
```

```
##
##  Fisher's Exact Test for Count Data
##
## data:  tot
## p-value = 0.0265
## alternative hypothesis: true odds ratio is not equal to 1
## 95 percent confidence interval:
##   1.047057 49.595860
## sample estimates:
```

```
## odds ratio
##   6.227408
```

Therefore, exact p-value is 0.0265. Fischer test approach is valid here.

```r
ratio = (6/15)/(8/3)
print(ratio)
```

```
## [1] 0.15
```

For every one student with psi and who has passed there is 0.15 that failed.

PART e:

Fischer test approach is valid here.

Advantage of logistic but disadvantage of contingency table : By second approach, we show there is dependency but it doesn't quanitfy it. Whereas we can numerically express the relation between psi and passed from logistic regression. We can't make predictions with the results of the fisher's exact test.

Fischer test is more suited for small sample size.

Advantage of contigency but disadvantage of logistic

5

Load awards.txt

```r
awards = read.csv("awards.txt", sep="")
```

5a.

We perform Poisson regression using the variable program

```r
poisson_awards <- glm(num_awards ~ prog, family="poisson", data=awards)
#poisson_awards
summary(poisson_awards)
```

```
##
## Call:
## glm(formula = num_awards ~ prog, family = "poisson", data = awards)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.4974  -1.2833  -0.1165   0.1881   3.4500
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -0.3485     0.2311  -1.508    0.131
## prog          0.1543     0.1047   1.474    0.141
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 228.83  on 199  degrees of freedom
## Residual deviance: 226.65  on 198  degrees of freedom
## AIC: 520.97
##
## Number of Fisher Scoring iterations: 5
```

Here, the p-value is greater than 0.05, so the variable program alone does not influence the number of awards.

Next, we estimate the number of awards of each type of program, according to the Poisson model. We create new dataframes that contain each of the program type, and use the predict() function to apply the model to the awards type.

```
new_data = data.frame(prog=1)
predict(poisson_awards, newdata=new_data, type = "response")
```

```
##         1
## 0.8234681
```

```
new_data = data.frame(prog=2)
predict(poisson_awards, newdata=new_data, type = "response")
```

```
##         1
## 0.9608369
```

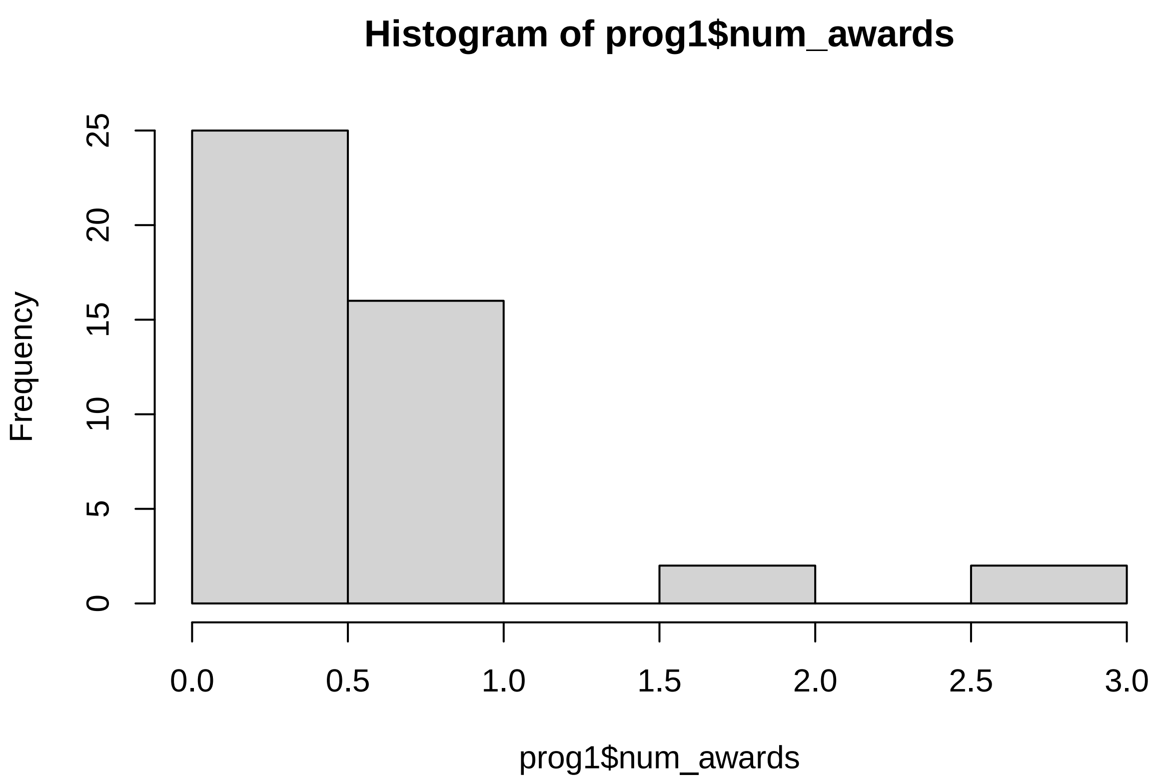
```
new_data = data.frame(prog=3)
predict(poisson_awards, newdata=new_data, type = "response")
```

```
##        1
## 1.121121
```

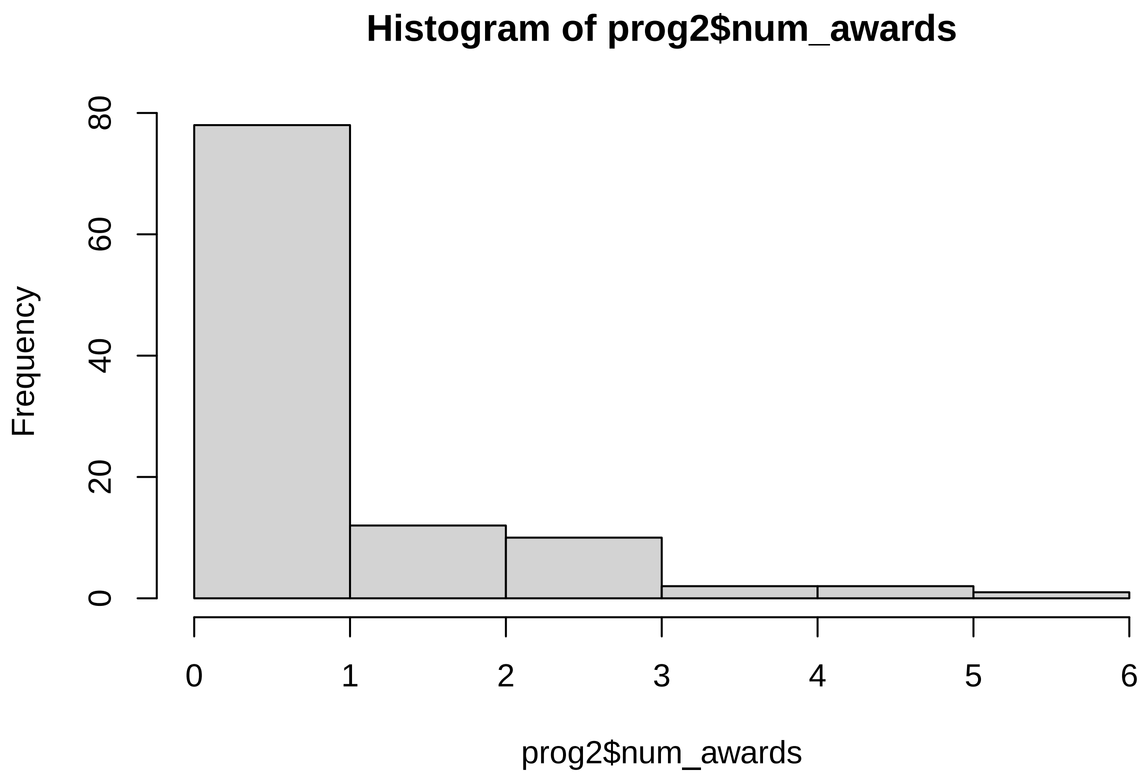
According to the results, program *academic* (3) results in the highest number of awards, which is 1.12.

5b.

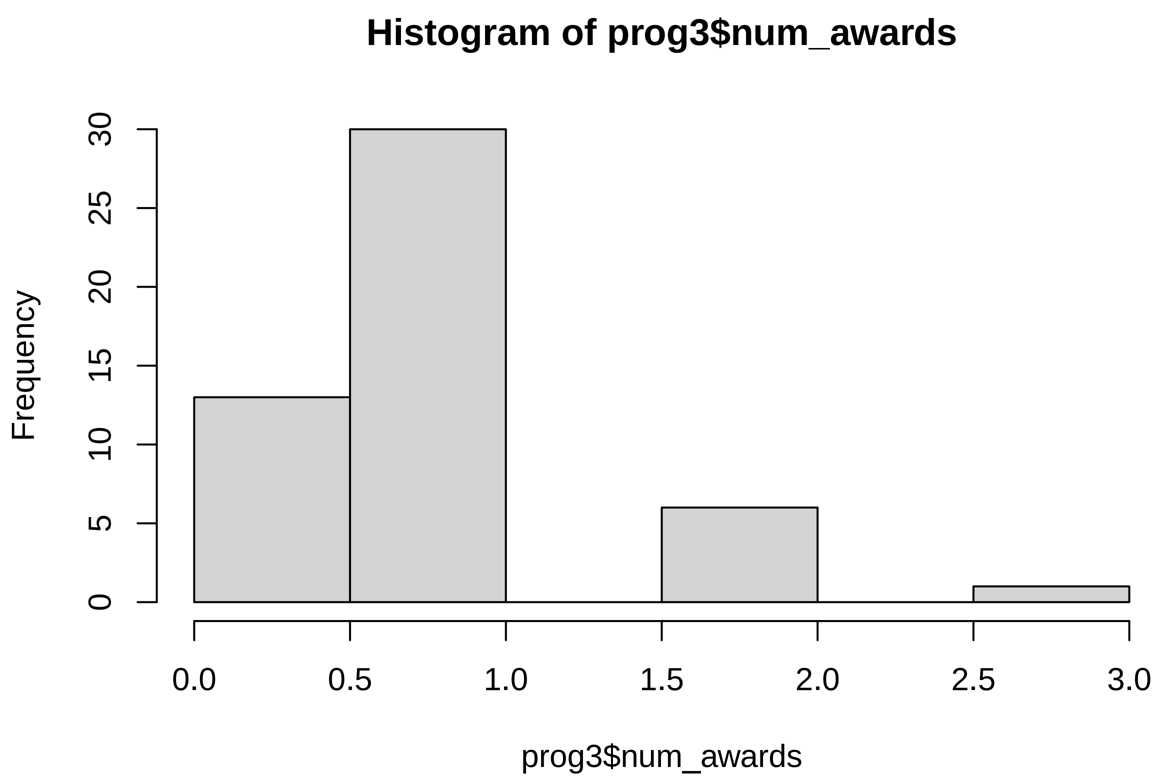
```
prog1 = subset(awards, prog==1)
hist(prog1$num_awards)
```




```
prog2 = subset(awards, prog==2)
hist(prog2$num_awards)
```




```
prog3 = subset(awards, prog==3)
hist(prog3$num_awards)
```



The distributions have the same shape, the response variable is ordinal and we assume independence. Therefore, we *can* apply the Kruskal-Wallis test. Performing this test:

```
kruskal_wallis<-kruskal.test(num_awards ~ prog, data=awards)
kruskal_wallis
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  num_awards by prog
## Kruskal-Wallis chi-squared = 10.755, df = 2, p-value = 0.00462
```

Here, the p-value is lower than 0.05, which implies that the program *does* influence the number of awards.

5c

We perform Poisson regression using both the variables math and program:

```
poisson_awards <- glm(num_awards ~ prog+math, family="poisson", data=awards)
#poisson_awards
summary(poisson_awards)
```

```
##
## Call:
## glm(formula = num_awards ~ prog + math, family = "poisson", data = awards)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -1.98567  -1.14535  -0.05993   0.33887   2.55070
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -2.721628   0.524702  -5.187 2.14e-07 ***
## prog         0.263565   0.117082   2.251   0.0244 *
## math         0.039541   0.007455   5.304 1.13e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 228.83  on 199  degrees of freedom
## Residual deviance: 199.08  on 197  degrees of freedom
## AIC: 495.4
##
## Number of Fisher Scoring iterations: 5
```

When using math and program, the p-values for both variables are below 0.05, and are thus significant. This is interesting, as creating the model using only the program variable yields that program is not significant in influencing the number of awards. This means that the model using only the program variable is insufficient.

To see which program is most effective, we use the model to predict the number of awards for each program, using both the maximum and minimum values of the math variable

```
print("Program 1")
```

```
## [1] "Program 1"
```

```
new_data = data.frame(prog=1, math=max(awards$math))
predict(poisson_awards, newdata=new_data, type = "response")
```

```
##        1
```

```
## 1.661149
```

```r
new_data = data.frame(prog=1, math=min(awards$math))
predict(poisson_awards, newdata=new_data, type = "response")
```

```
##         1
## 0.3156216
```

```r
print("Program 2")
```

```
## [1] "Program 2"
```

```r
new_data = data.frame(prog=2, math=max(awards$math))
predict(poisson_awards, newdata=new_data, type = "response")
```

```
##        1
## 2.162087
```

```r
new_data = data.frame(prog=2, math=min(awards$math))
predict(poisson_awards, newdata=new_data, type = "response")
```

```
##         1
## 0.4108009
```

```r
print("Program 3")
```

```
## [1] "Program 3"
```

```r
new_data = data.frame(prog=3, math=max(awards$math))
predict(poisson_awards, newdata=new_data, type = "response")
```

```
##       1
## 2.81409
```

```r
new_data = data.frame(prog=3, math=min(awards$math))
predict(poisson_awards, newdata=new_data, type = "response")
```

```
##         1
## 0.5346826
```

In the results, we can see that for both the maximum and minimum values of the math variable, the number of awards is the highest for program 3. Therefore program 3 is three is the best for the number of awards.

The number of awards for the vocational program (1) and the math score of 55 is estimated by:

```r
new_data = data.frame(prog=1, math=55)
predict(poisson_awards, newdata=new_data, type = "response")
```

```
##         1
## 0.7532863
```

Thus, the number of awards for the vocational program and math score 55 is 0.75.