Stats Practicals

2025-04-21

Stat Modelling course

Going through the Statistical modelling course from Cam. Begin with:

Linear Models

Classic is to use the ordinary least squares estimators. We have the model

$$Y = X\beta + \varepsilon$$

where Y is a our vector of dependent outcomes, X is the design matrix, β the vector of predictors. We have the common assumptions - $\mathbb{E}(\varepsilon) = 0$ - $\mathrm{Var}(\varepsilon) = \sigma^2 I$

We may also include a column of 1's in the design matrix if we wish to have an intercept, moreover the design matrix can take functions of $x_i j$ as elements - the model is linear in β.

Anyhoo - I'm not here to re-write the notes out. I'm doing the practicals.

Question 1

```r
N <- 100000000
z <- rnorm(N)

b =exp_given_geq1 <- mean(z[z>=1])
print(exp_given_geq1)
```

```
## [1] 1.524965
```

```r
sixth_moment = mean(z^6)
print(sixth_moment)
```

```
## [1] 15.00417
```

Question 2

```r
out <- qchisq(0.05, 6, FALSE)
print(out)
```

```
## [1] 1.635383
```

Question 3

```r
M <- matrix(c(3,4,-2,1,2,-1,7,-2,6,2,-1,1,1,6,-2,5), 4,4, byrow = TRUE)
#print(M)
b = c(9,13,11,27)

x <- solve(M,b)
print(x)
```

```
## [1] 1 2 3 4
```

Cool, that all seemed to go smoothly. Next, onto the next!!

This is concerned with writing functions in R.Consider

```
f <- function (x,y){
  z <- x^2 + y^2
  return(c(cos(z), sin(z)))
}

print(f(1,1))
```

```
## [1] -0.4161468  0.9092974
```

Cool, and we note that we can use scripts via the 'source' keyword in the console.

Next goal is to write a piece of R to simulate t-statistics from a linear model. Recall that

$$\frac{\hat{\beta}_i - \beta}{\sqrt{\hat{\sigma^2}(X^T X)^{-1}_{ii}}}$$

has has the t distribution when we scale by some amount depending on n and p. (Indeed, the MLE of σ^2 is proportional to a χ^2_{n-p} RV).

Cool, so our function will take in the design matrix X, a vector of coeffs β and a function for generating the errors (why not just generate some normal and chi squared random variables.)

Right, I'll write the function in a different script to see if I can call it within RMD

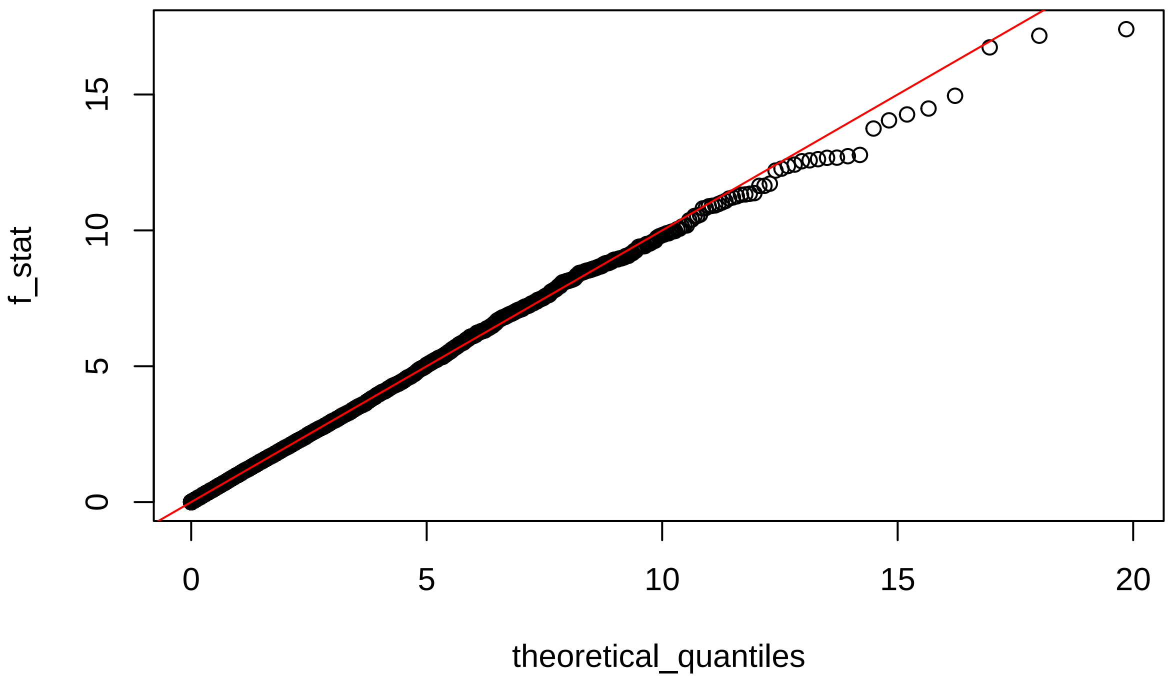
```
source('~/Docs/Stats_practicals/Lin_mod_sim.R')
#Okay - lets consider a case where $n=50$ and $p=2$
n<-50
p<-2
X <- matrix(rnorm(n*p),n,p)

#Then we can pass this to our function and see what happens (note that it will use the predetermined va
#In particular the initialised value of $beta$ is just $(0,0)$.

t_mat <- LinMod_sim(X)
```

Okay, so we've produced a number of t-distributed instances - perhaps we'd like to plot these to get an idea of the distribution of t_{n-p}. Ok - one way to check the correctness of our distribution (seeing as R has an inbuilt t distrib) is using QQ plots.

To do so - we sort the statistics we've calculated in order, then plot them against the relevant quartiles! - we should expect an approximately straight line.

Note that we're interested in a two sided t test (think - null being blah, alt being blah) and so it makes sense to look at the square of the t statistic. This is distrib $F_{1,d}$ where d is the df of the t-distrib in quesiton.

```
source('~/Docs/Stats_practicals/Lin_mod_sim.R')
perc <- qqplot_F((t_mat)^2,1,n-p)
```



```r
print(perc)
```

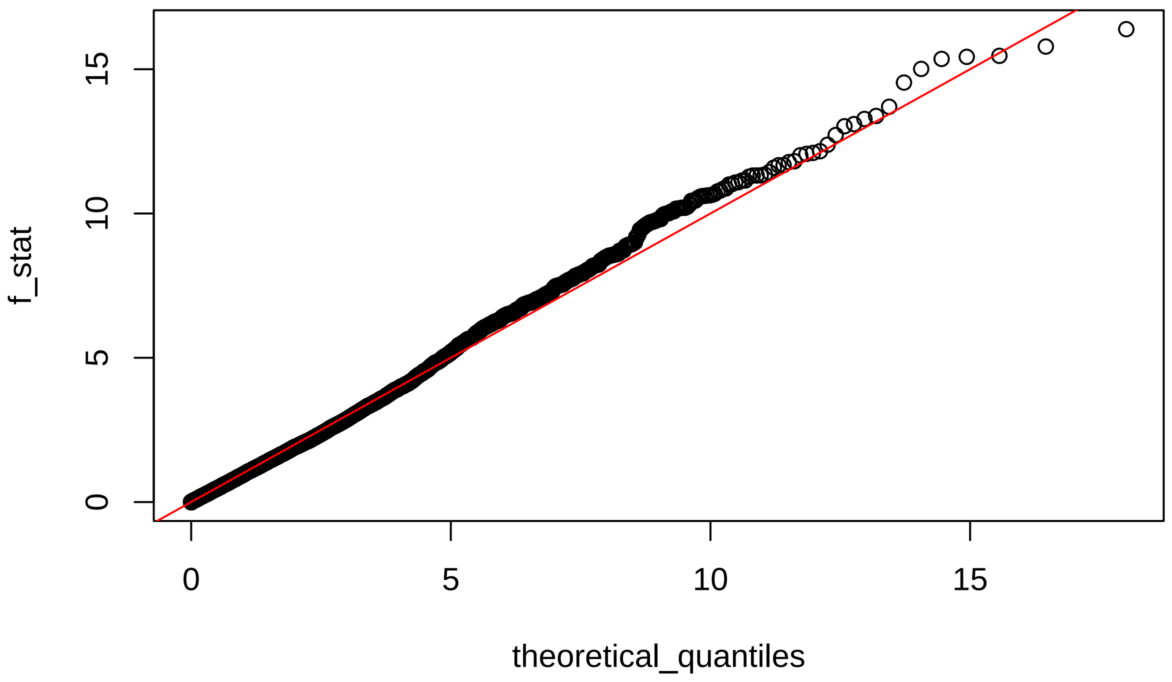
```
## [1] 0.9502
```

Okay, nice!

Let's try some different variations of N, p, and the random-distrib of our errors.

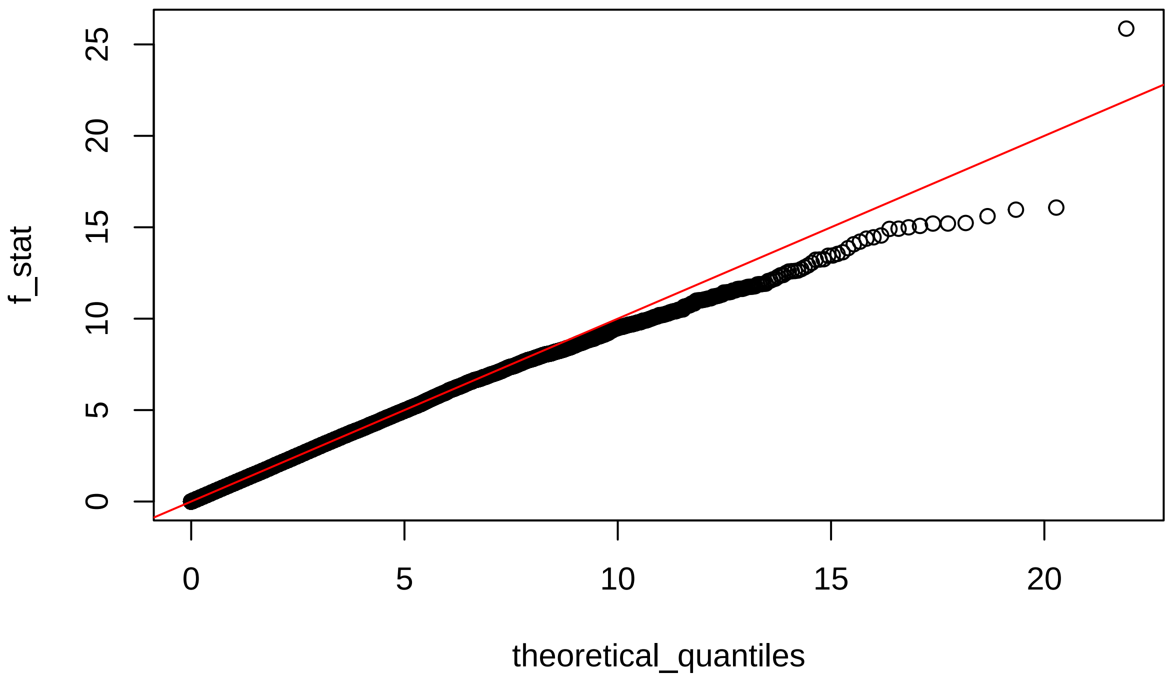
Note that we should expect that changing our random errors from a normal distribution will mean that our statistic isn't necessarily still t-distributed. Perhaps for large n there will be some CLT activity going on though that brings this back around

```r
source('~/Docs/Stats_practicals/Lin_mod_sim.R')
n<-100
p<-2
rand_gen <- rcauchy
X <- matrix(rnorm(n*p),n,p)

out2 <- LinMod_sim(X, errors_gen =rand_gen)
qqplot_F(out2^2, 1, n-p)
```



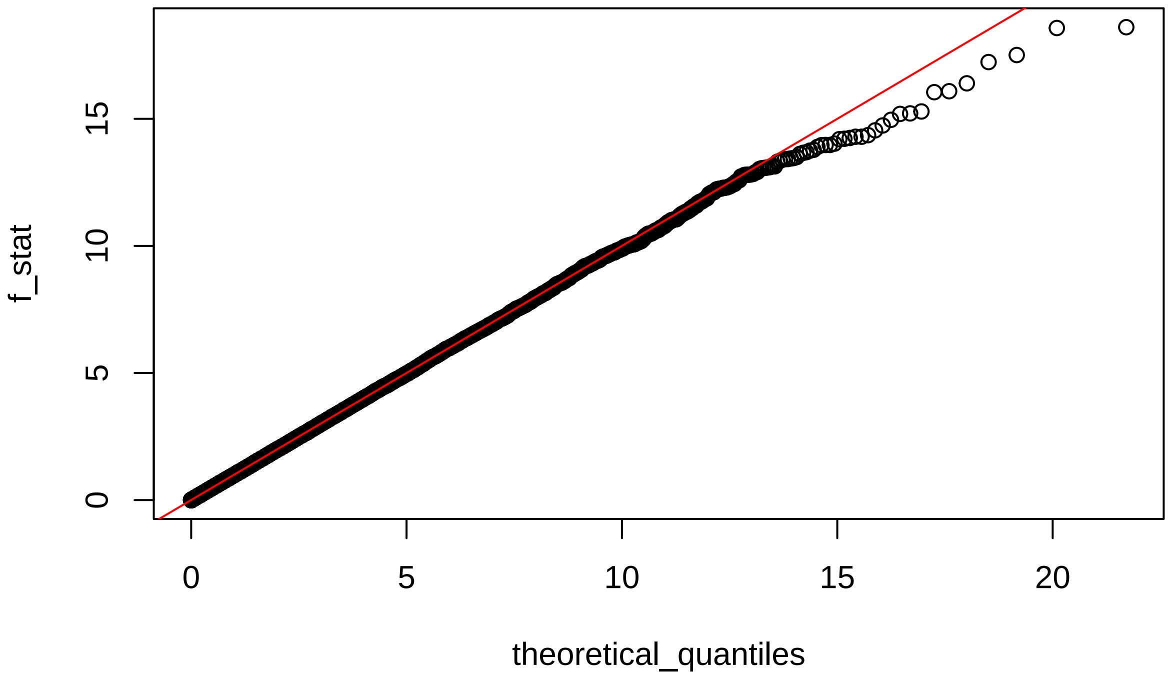
```
## [1] 0.9513
```

```
n<-100
p<-10
rand_gen <- rcauchy
#Note to self that cauchy distrib doesn't have a mean.
X <- matrix(rnorm(n*p),n,p)

out2 <- LinMod_sim(X, errors_gen =rand_gen)
qqplot_F(out2^2, 1, n-p)
```



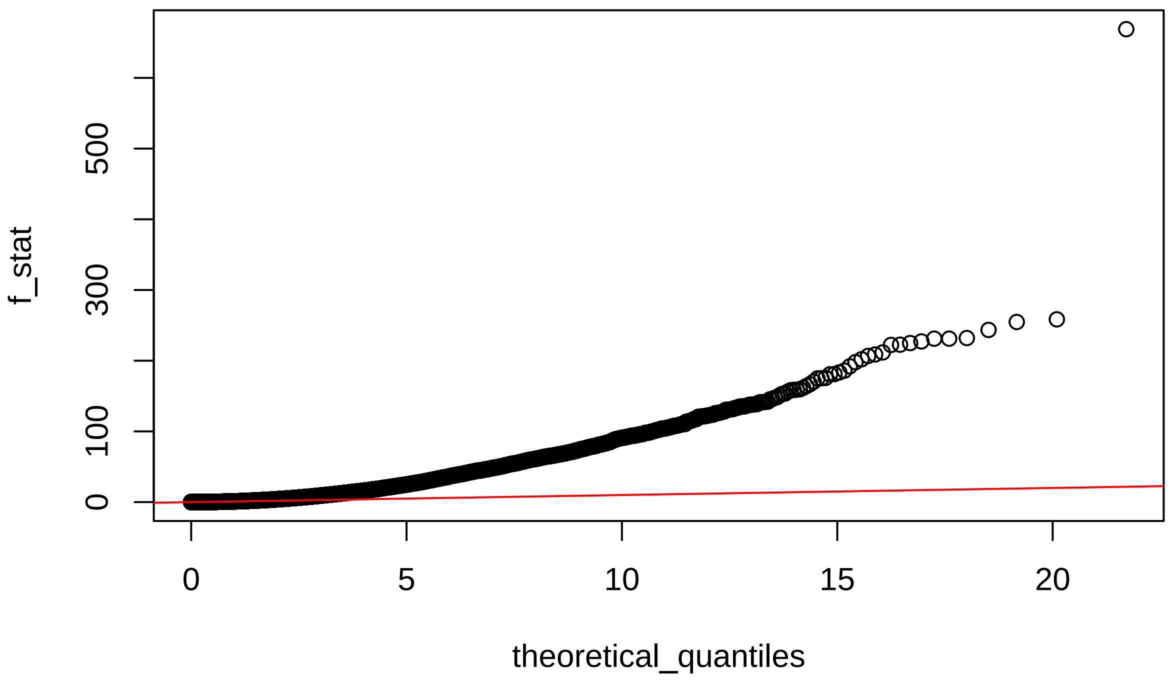
```
## [1] 0.95024
```

```r
n<-100
p<-2
rand_gen <- function(x) rexp(x)-1
out3 <- LinMod_sim(X,errors_gen = rand_gen)
qqplot_F(out3^2, 1, n-p)
```



```
## [1] 0.95019
```

```r
n<- 100
p<-2
rand_gen <- function(N) rgamma(N,shape =1/2, rate = 1/2) - 1
#Again we subtract off the 1 so that the mean is zero
out3 <-LinMod_sim(X,errors_gen = rand_gen)
qqplot_F(out2^4, 1, n-p)
```



```
## [1] 0.83701
```

Okay dokey, last thing is lists.

Allows us to make lists of items with different types.

```
empl <- list(employee = 'Ernie', spouse = "Adam", children = 2, child_ages = c(1,2))

print(empl)
```

```
## $employee
## [1] "Ernie"
##
## $spouse
## [1] "Adam"
##
## $children
## [1] 2
##
## $child_ages
## [1] 1 2
```

```
#THen use $ to access with keys

empl[2]
```

```
## $spouse
## [1] "Adam"
```

```
empl$employee
```

```
## [1] "Ernie"
```

Exercises

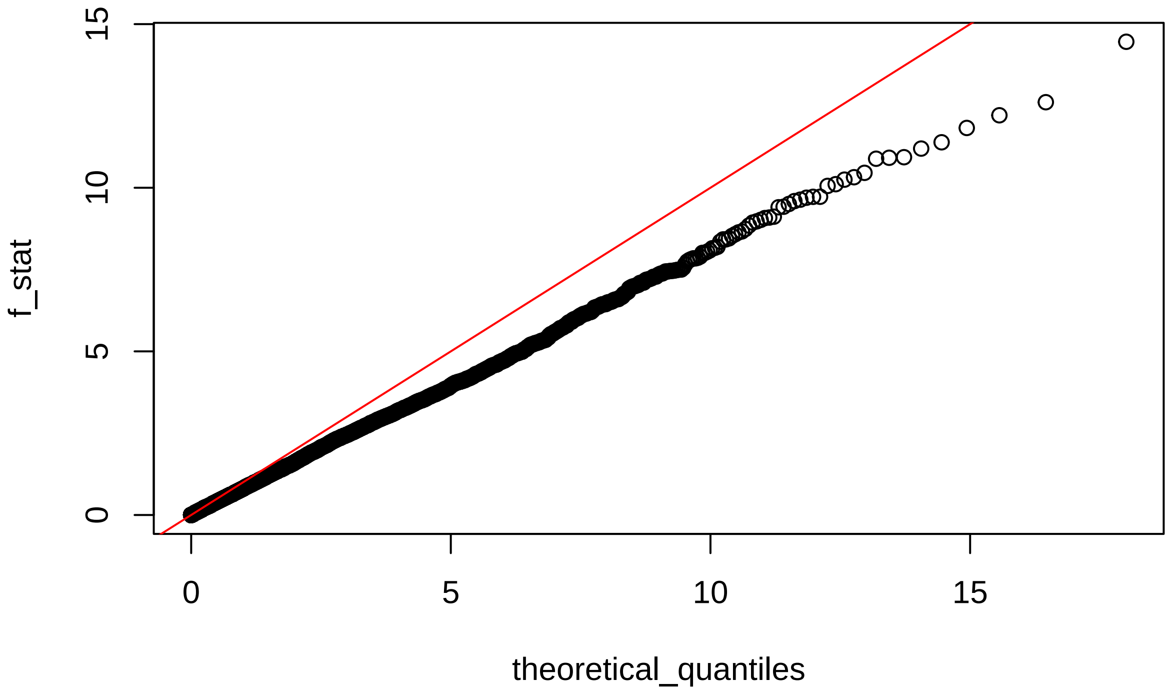
Q1: I mean - I suppose this changes quite a lot! Eg. we should no longer have an estimator $\hat{\sigma^2}$ as a scalar but rather a vector of length n. How should this be estimated? - intuitively we should just take the ith component to be $Y_i - \hat{Y}_i$ (eg. lets look at the MLE)

We have now that the error matrix epsilon is distributed MVN with variance being a diagonal matrix D, therefore the density function of Y is given by

$$f(y) = \frac{1}{(2\pi)^{n/2}\det(D)^{1/2}}\exp(-\frac{1}{2}(y - X\beta)^T D^{-1}(y - X\beta))$$

The MLEs are then $\hat{\beta} = (X^T X)^{-1} X^T Y$ and by using that D is diagonal, we have simply that the estimate for the ith variance is $(y_i - (X\beta)_i)^2$.

```
#Have made the modification in LinModSim file
#source('~/Docs/Stats_progamming/Lin_mod_sim.R')

set.seed(1)

n<-100
p<-2
X <- matrix(rnorm(n*p),n,p)

t_different_variances <- LinMod_sim_modified(X)
qqplot_F(t_different_variances^2, 1, n-p)
```



```
## [1] 0.9722
```

So the data lies below the diagonal. Here is an exerpt from the notes:

Note that if the points on the Q–Q plot lie above the diagonal it suggests that the usual t-test with nominal level α will have a size greater than α (why?). On the other hand, if the points lie below the diagonal, the t-test will be conservative and have size less than α.

Ok - so we're using the T-test to get an interval for the parameter β_j some j. Note that a non-normal linear model - the t test wont be valid tho??

Anyway, if the data points lie below the diagonal - then this tells us something about how the actual distrib of our data (ie. the one we've set for the random errors) compares to the f-stat (ie f because we've taken t^2).

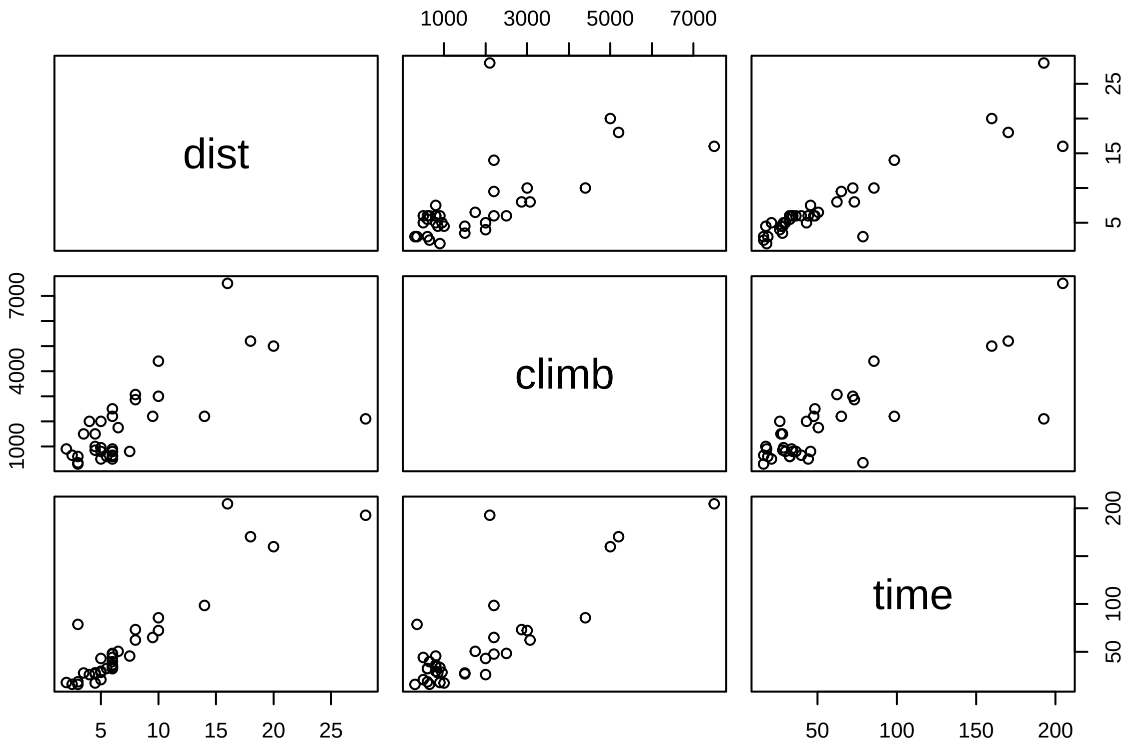
Since it's below the line, the actual quantiles tend to be smaller than the predicted quantiles - meaning that the data has lower variance essentially (ie. less probaility in the tail.)

Ok. Will look into this post ex 2

Exercise 2

Again, done in the script.

```
library(MASS)
?hills


Data <- hills

print(pairs(Data))
```



```
## NULL
```

```
print(Data[(Data$time >50) & (Data$dist<10),])
```

```
##             dist climb   time
## Ben Lomond   8.0  3070 62.267
## Goatfell     8.0  2866 73.217
## Lomonds      9.5  2200 65.000
## Knock Hill   3.0   350 78.650
## Criffel      6.5  1750 50.500
```

```
#Knock Hill seems to be the outlier - we want to replace the time value by subtracting off an hour.

#How to do so.


rownames = rownames(Data)
```

```
index = which(rownames == 'Knock Hill')
#print(typeof(index))

Data[index, 3] <- Data[index,3] -60


print(Data[(Data$time >50) & (Data$dist<10),])
```

```
##            dist climb   time
## Ben Lomond  8.0  3070 62.267
## Goatfell    8.0  2866 73.217
## Lomonds     9.5  2200 65.000
## Criffel     6.5  1750 50.500
```

Cool, so we got rid of the outlier. Question to self, what other good ways are there to remove outliers?

OK - lets try and do some linear regression with this data.

Model 1: Lets treat Time as the dependent variable, and distance and climb as the indep variables. I dont feel like there needs to be an intercept here. There shouldnt really be a kinda min benchmark time (if both distance and climb are zero, then time is also 0)

Model 2: We could try and consider something more complicated (eg quadratic in, say the climb), or we could try and do regression in another direction. Here we dont want to overfit though!

Model 3: Take logs! This is a reasonable idea if we suspect that the data is not linear - as if we have some proportionality to powers this will linearise the situation.

However we would need to include an intercept. If we are considering

$\log(t_i) = \alpha_i + \beta_1 \log(d_i) + \beta_2 \log(c_i) + \varepsilon_i$

Then for $d = 1$ and $c = 1$ we would not necessarily expect $t = 1$ also (which is what zero intercept would imply).

In the non-logged case, we indeed have no intercept because $d = c = 0$ does imply we should expect $t = 0$

```
attach(Data)

Correlation <- cor(Data)
print(Correlation)
```

```
##            dist     climb      time
## dist  1.0000000 0.6523461 0.9429552
## climb 0.6523461 1.0000000 0.8322600
## time  0.9429552 0.8322600 1.0000000
```

```
LinearModel1 = lm(time ~ 0 + dist + climb^2 )
summary(LinearModel1)
```

```
##
## Call:
## lm(formula = time ~ 0 + dist + climb^2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -17.818  -9.599  -5.538  -2.628  37.348
##
## Coefficients:
##       Estimate Std. Error t value Pr(>|t|)
## dist  5.464282   0.414256  13.191 1.04e-14 ***
## climb 0.010645   0.001592   6.685 1.30e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 11.58 on 33 degrees of freedom
## Multiple R-squared:  0.9775, Adjusted R-squared:  0.9761
## F-statistic: 716.1 on 2 and 33 DF,  p-value: < 2.2e-16
```

```
LinearModel2 = lm(log(time) ~ log(dist) + log(climb) )
summary(LinearModel2)
```

```
##
## Call:
## lm(formula = log(time) ~ log(dist) + log(climb))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.52624 -0.06273  0.00452  0.06846  0.31384
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.29359    0.27312   1.075     0.29
## log(dist)    0.91141    0.06534  13.949 3.76e-15 ***
## log(climb)   0.24889    0.04761   5.228 1.02e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1607 on 32 degrees of freedom
## Multiple R-squared:  0.9521, Adjusted R-squared:  0.9491
## F-statistic: 317.8 on 2 and 32 DF,  p-value: < 2.2e-16
```

```
LinearModel3 = lm(time ~ 0+ dist)
summary(LinearModel3)
```

```
##
## Call:
## lm(formula = time ~ 0 + dist)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -27.096 -10.701  -6.237   0.485  79.038
##
## Coefficients:
##      Estimate Std. Error t value Pr(>|t|)
## dist   7.8487     0.3185   24.64   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 17.51 on 34 degrees of freedom
## Multiple R-squared:  0.947,  Adjusted R-squared:  0.9454
## F-statistic: 607.2 on 1 and 34 DF,  p-value: < 2.2e-16
```

```
detach(Data)
```

```
#Ok - I like the first model the best
```

Yep, I like the look of the first model the best. So lets make a prediction with this -> assuming that the model is correct (ie. that T = beta_0 Dist + beta_1 Climb^2 + epsilon)

We make a new observation: Dist = 5.3 Miles, and Climb = 1100 ft.

Hence T = blah + eps, where blah is now a number. We can estimate T using our predictions for the values of beta, then get a confidence interval for it (which will be a t-test!)

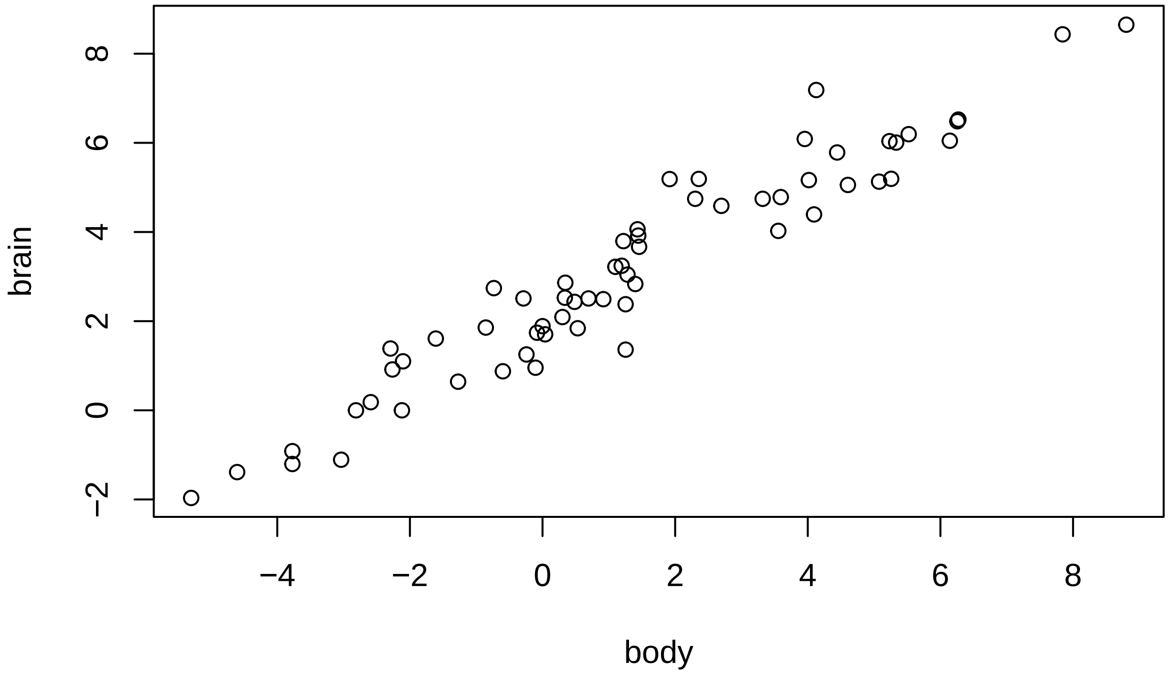
In R this is as follows:

```
newdata <- data.frame('dist' = 5.3, 'climb'=1100)

predict(LinearModel1, newdata,  interval='confidence', level=0.95 )
```

```
##        fit      lwr      upr
## 1 40.67066 38.38115 42.96018
```

Question 9, Es 3

```
library(MASS)
attach(mammals)

plot(log(mammals))
```

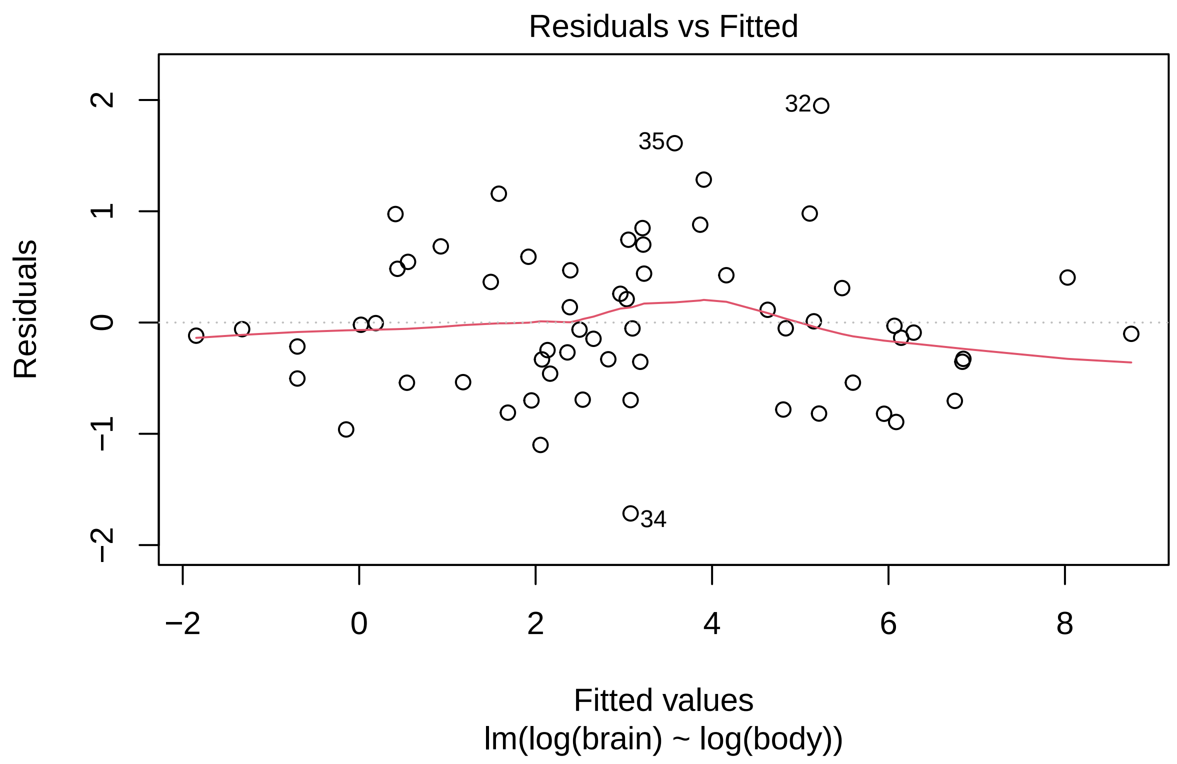


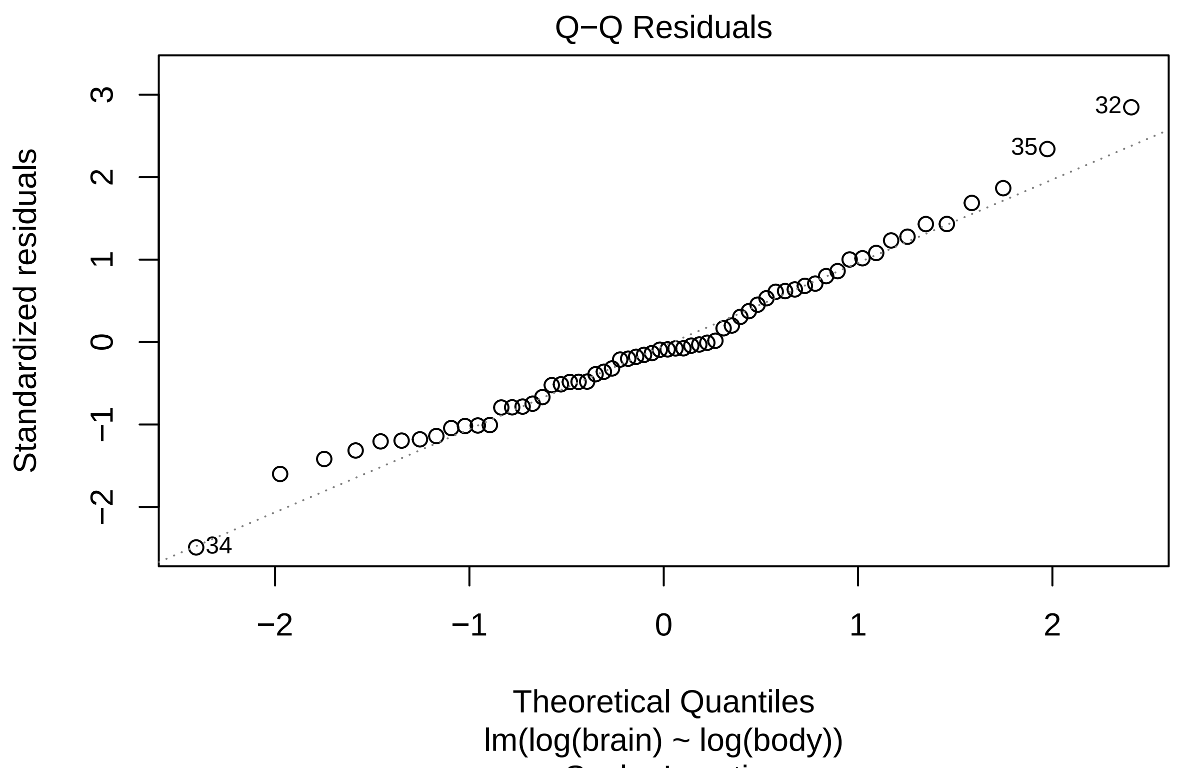
```
index = which(rownames(mammals) == 'Human')


LinModel = lm(log(brain) ~ log(body)) #Note have included an intercept for same reason as above.

summary(LinModel)
```

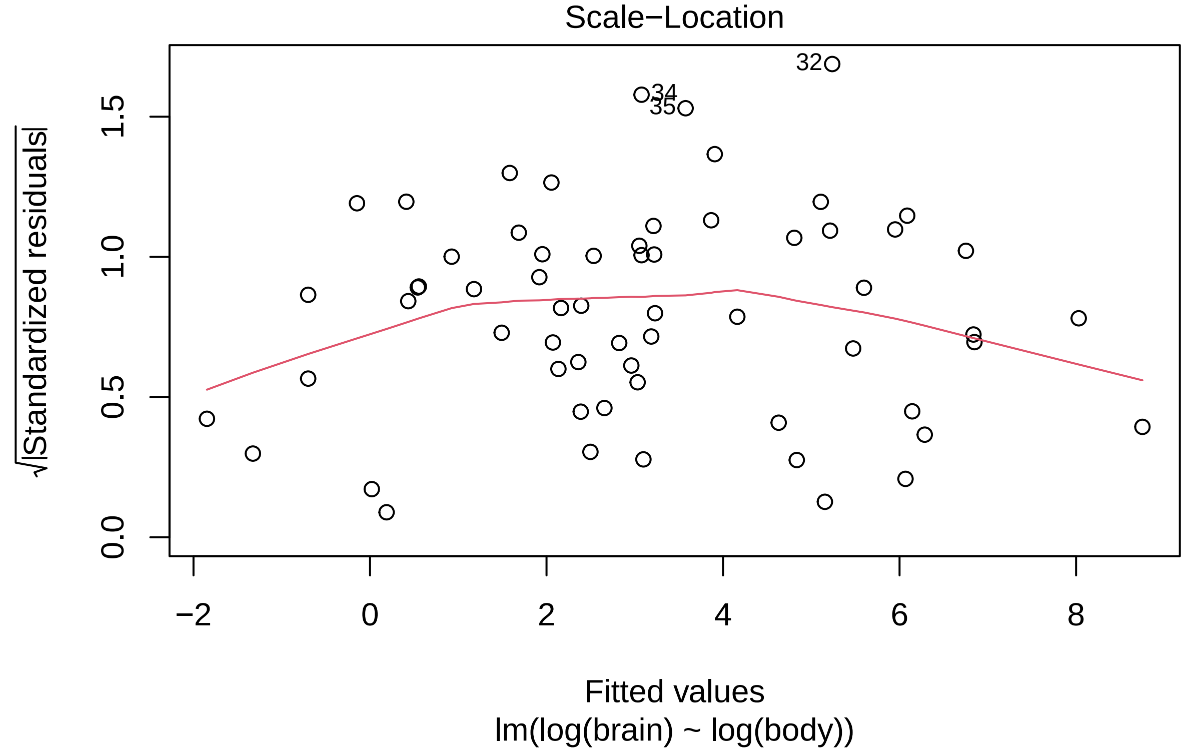
```
##
## Call:
```

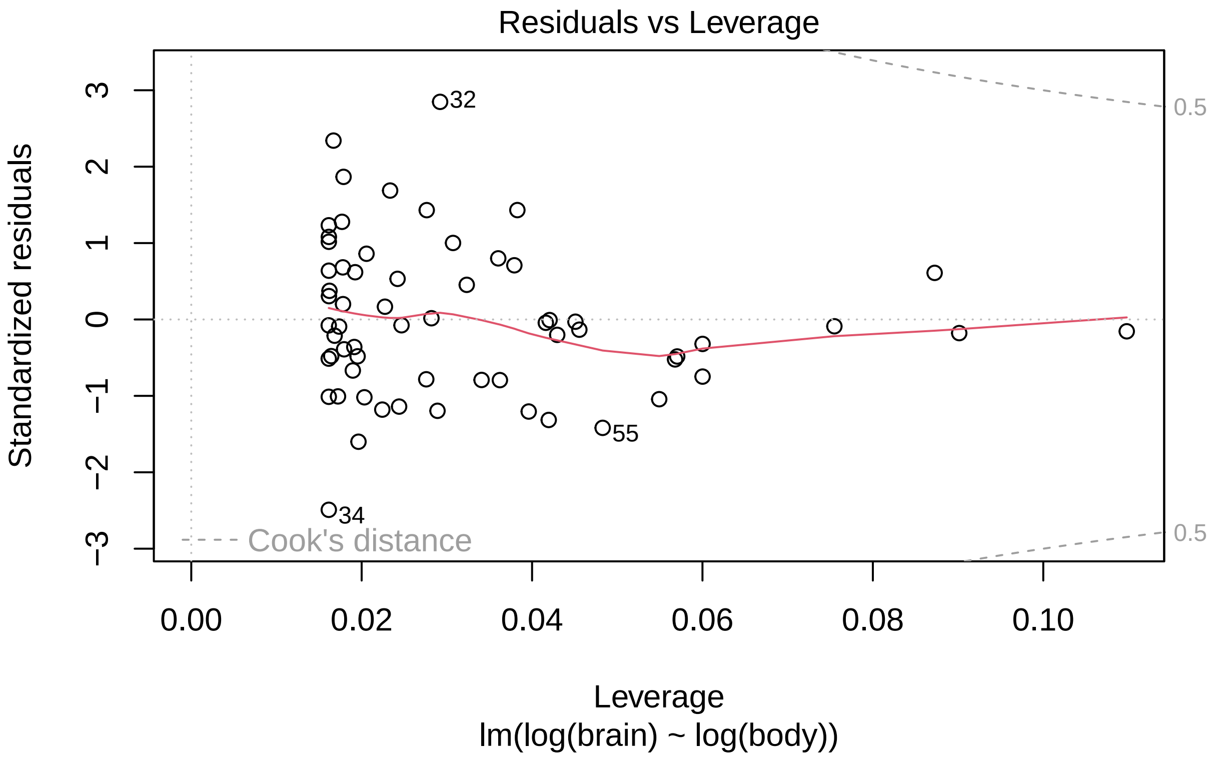
```
## lm(formula = log(brain) ~ log(body))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.71550 -0.49228 -0.06162  0.43597  1.94829
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.13479    0.09604   22.23   <2e-16 ***
## log(body)    0.75169    0.02846   26.41   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6943 on 60 degrees of freedom
## Multiple R-squared:  0.9208, Adjusted R-squared:  0.9195
## F-statistic: 697.4 on 1 and 60 DF,  p-value: < 2.2e-16
```

```r
plot(LinModel)
```




Q−Q Residuals
Standardized residuals
Theoretical Quantiles
lm(log(brain) ~ log(body))
32
35
34
−2
−1
0
1
2
3


Scale−Location
√|Standardized residuals|
Fitted values
lm(log(brain) ~ log(body))
32
34
35
0.0
0.5
1.0
1.5
−2
0
2
4
6
8



```r
tval <- rstudent(LinModel)[index]
n <- length(brain)
p=2 #Have intercept and log(body)


print(pt(tval,n-p-1 ))
```

```
##        32
## 0.9982223
```

```r
detach(mammals)
```

A one sided test is appropriate if we think that humans are cracked - ie if the alt hypoth is that our brains are larger than they should be!

Ok onto the third sheet.

```r
library(MASS)
attach(cabbages)

#So want to kinda do ANOVA on this!


model = lm(HeadWt ~ Date)

summary(model)
```

```
##
## Call:
## lm(formula = HeadWt ~ Date)
##
## Residuals:
```

```
##    Min     1Q Median     3Q    Max
## -1.105 -0.520 -0.280  0.405  2.095
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   2.7200     0.1831  14.853   <2e-16 ***
## Dated20       0.2350     0.2590   0.907    0.368
## Dated21      -0.6150     0.2590  -2.375    0.021 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.819 on 57 degrees of freedom
## Multiple R-squared:  0.1678, Adjusted R-squared:  0.1386
## F-statistic: 5.745 on 2 and 57 DF,  p-value: 0.005335
```

```
model2 = lm(log(HeadWt) ~ Date)

summary((model2))
```

```
##
## Call:
## lm(formula = log(HeadWt) ~ Date)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.66069 -0.19069 -0.07219  0.17865  0.77440
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.95589    0.07004  13.648  < 2e-16 ***
## Dated20      0.10883    0.09905   1.099  0.27648
## Dated21     -0.29520    0.09905  -2.980  0.00423 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3132 on 57 degrees of freedom
## Multiple R-squared:  0.2382, Adjusted R-squared:  0.2114
## F-statistic:  8.91 on 2 and 57 DF,  p-value: 0.0004296
```

```
detach(cabbages)
```

Sheet 4

Question 3

```
n <- c(9,10,15,25,32,33,37,46,46)
i <- 1:9

model <- glm(n ~ i , family = poisson(link = 'log'))

summary(model
       )
```

```
##
## Call:
```

```
## glm(formula = n ~ i, family = poisson(link = "log"))
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  2.25882    0.17444   12.95  < 2e-16 ***
## i            0.19159    0.02617    7.32 2.47e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 63.9421  on 8  degrees of freedom
## Residual deviance:  6.3512  on 7  degrees of freedom
## AIC: 55.624
##
## Number of Fisher Scoring iterations: 4
```

Question 8

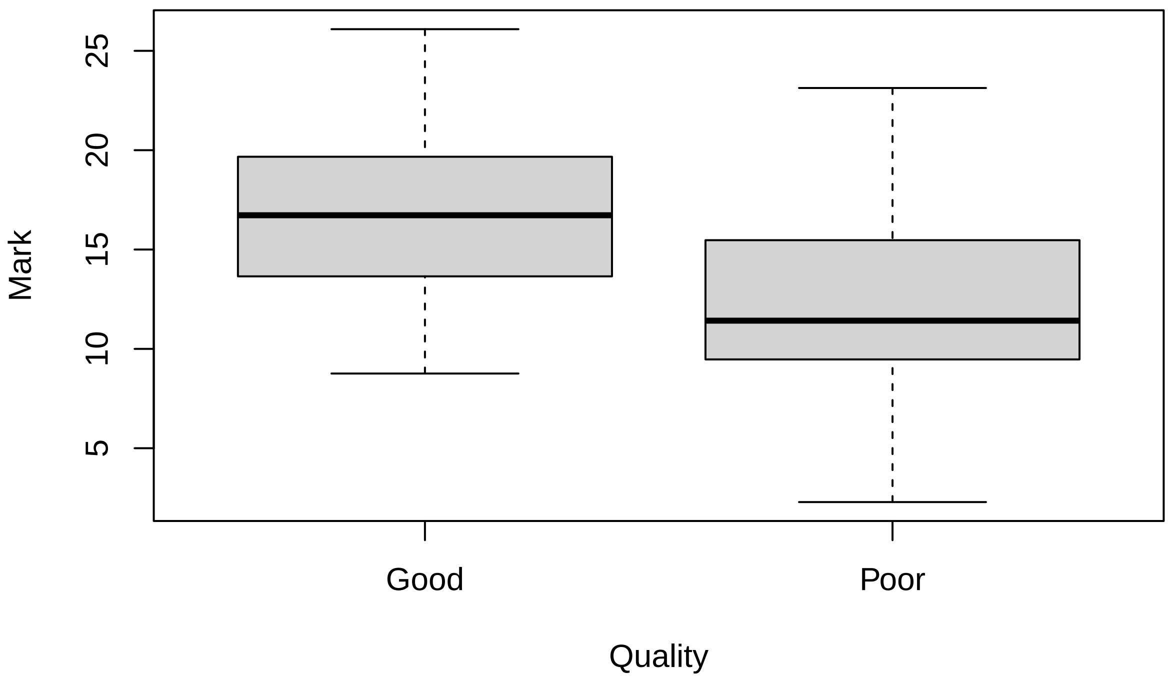
```
M = matrix(c(1,0,-1,1) ,2 )
S = matrix(c(0.6784^2, 0.59*0.7871*0.6784, 0.59*0.7871*0.6784, 0.7871^2 ),2)


sd = (M%*%S%*%t(M))[1,1]

zscore = (1.9328-1.5331)/sd
```

Practical 5: ANOVA and ANCOVA

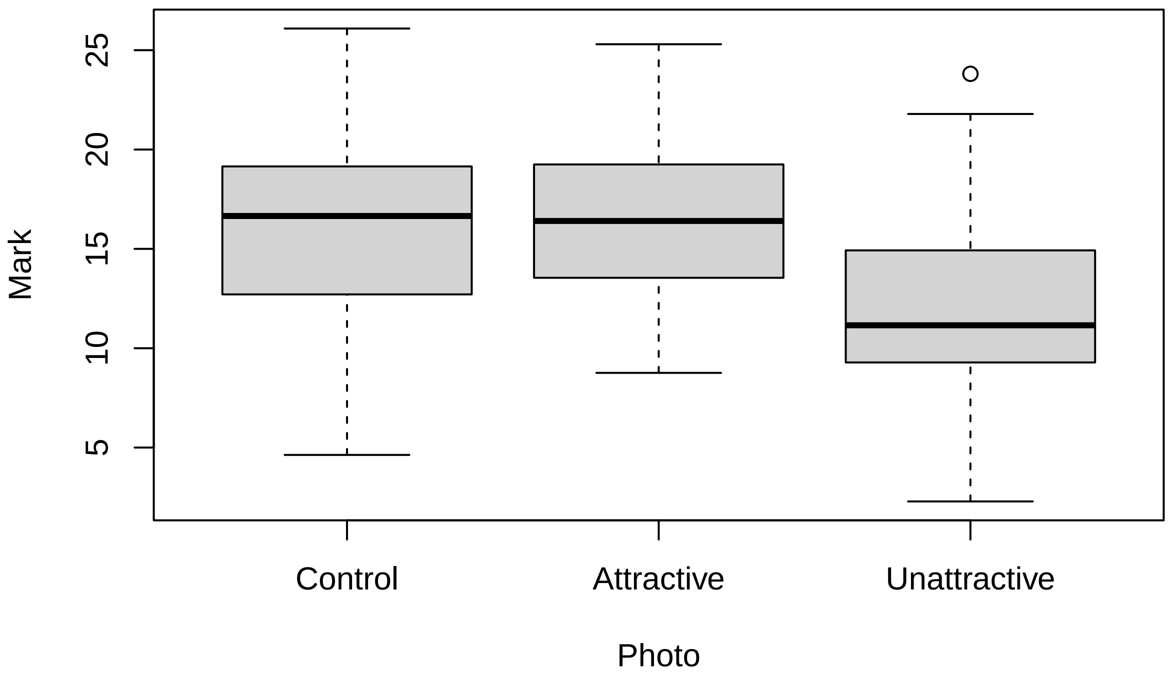
```
file_path <- "http://www.statslab.cam.ac.uk/~rds37/teaching/statistical_modelling/"
EssayMarks <- read.csv(paste0(file_path, "EssayMarks.csv"))



fdata <- factor(c(1,2,3,4,3,2))

levels(fdata) <- c("Fisher", "Bayes", "Neyman", "Pearson")


attach(EssayMarks)
Quality <- factor(Quality)
Photo <- factor(Photo)

Photo <-  relevel(Photo, 'Control')
levels(Photo)
```

```
## [1] "Control"      "Attractive"   "Unattractive"
```

```
#Yep, just doing this as we do corner point constraint on control (which makes the most sense) rather t

plot(Mark~ Quality)
```



```r
plot(Mark~ Photo)
```



So there seems to do something going on, and we want to do some hyptohesis testing to see whether there is indeed. (ie. want to test the null, that there is no effect (ie. Alpha = 0), agains the alt that there is an effect)

So, we have 2 differenct types of category, with 2,3 cats within them.

Therefore we want to model the data Y_{ijk}, where this is the kth student (of 10) in the group given essay of qual type i and photo type j.

Our Linear Model's are then as follows

$$
\begin{aligned}
Y_{ijk} &= \mu + \alpha_i + \varepsilon_{ijk} && (1)\\
Y_{ijk} &= \mu + \alpha_i + \beta_j + \varepsilon_{ijk} && (2)\\
Y_{ijk} &= \mu + \alpha_i + \beta_j + \gamma_{ij} + \varepsilon_{ijk} && (3)
\end{aligned}
$$

And we impose some corner point constraints, ie. alpha_1, beta_1, and gamma_{1j} gamma_{i1} are all 0.

So the first model is where the beta and gamma are taken to be zero - we can do F-test (ie. ANOVA) against the models where they are not zero to test if they are significant enough to say they're nonzero.

First, why have we included an intercept? This comes with the fact that we're using the corner point constraint.

In the first model, the intercept is the mean quality when i =0 (ie, the essay is 'good'). In the second model and third, the intercept is the mean qual when i=j=0 (ie. essay good, control photo.)

The third model has these interaction terms between the types of categories. This captures dependence between them.

For example: perhaps it is the case that when the text is good *and* the photo is attractive, there is more of a change in perception than if the text is bad and the photo is attractive - there could be some dependence between the two types of category.

```r
EssayMarksLM1 <- lm(Mark ~ Quality)
EssayMarksLM2 <- lm(Mark ~ Quality + Photo)
EssayMarksLM3 <- lm(Mark ~ Quality*Photo)

summary(EssayMarksLM1)
```

```
##
## Call:
## lm(formula = Mark ~ Quality)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -10.0433  -3.0099  -0.5215   2.6003  10.7967
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  17.0997     0.8695  19.666  < 2e-16 ***
## QualityPoor  -4.7663     1.2297  -3.876 0.000273 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.762 on 58 degrees of freedom
## Multiple R-squared:  0.2057, Adjusted R-squared:  0.1921
## F-statistic: 15.02 on 1 and 58 DF,  p-value: 0.0002729
```

```r
summary(EssayMarksLM2)
```

```
##
## Call:
## lm(formula = Mark ~ Quality + Photo)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -10.0227  -3.0141  -0.1066   1.8968   9.8632
```

```
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)        18.0332     1.1467  15.726  < 2e-16 ***
## QualityPoor        -4.7663     1.1467  -4.157 0.000112 ***
## PhotoAttractive     0.7495     1.4044   0.534 0.595673
## PhotoUnattractive  -3.5500     1.4044  -2.528 0.014327 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.441 on 56 degrees of freedom
## Multiple R-squared:  0.3331, Adjusted R-squared:  0.2974
## F-statistic: 9.325 on 3 and 56 DF,  p-value: 4.269e-05
```

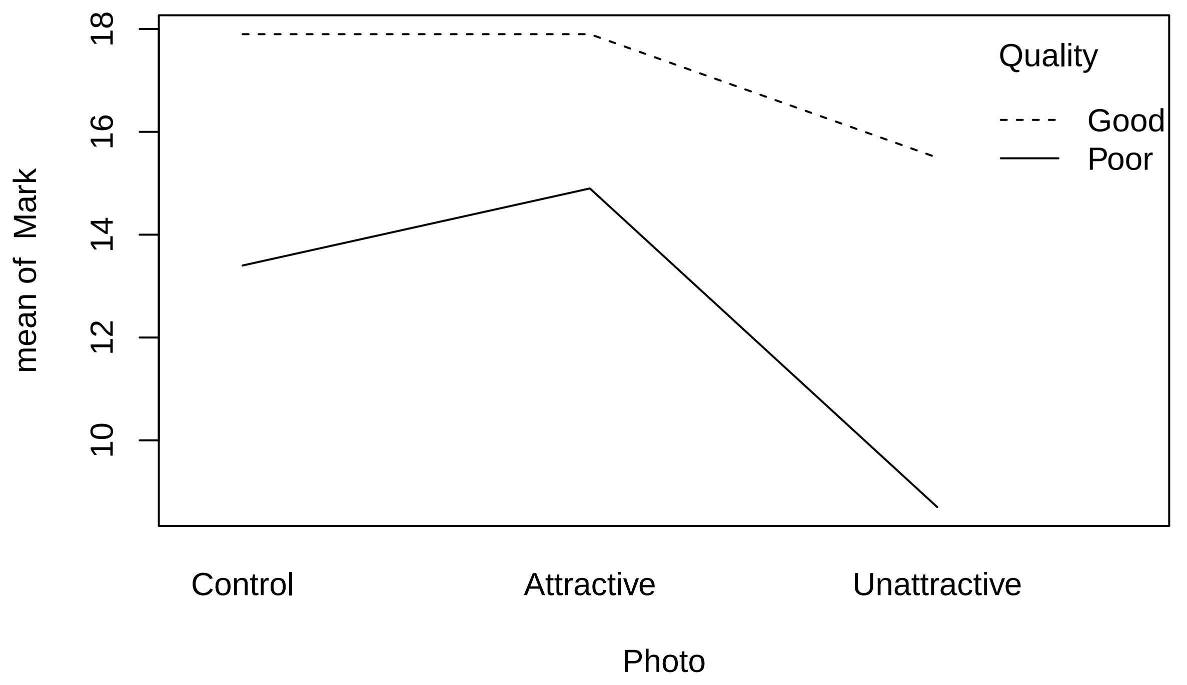
```
summary(EssayMarksLM3)
```

```
##
## Call:
## lm(formula = Mark ~ Quality * Photo)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -9.140 -3.074 -0.055  2.332  9.730
##
## Coefficients:
##                                Estimate Std. Error t value Pr(>|t|)
## (Intercept)                   1.790e+01  1.406e+00  12.729   <2e-16 ***
## QualityPoor                  -4.500e+00  1.989e+00  -2.263   0.0277 *
## PhotoAttractive               4.540e-16  1.989e+00   0.000   1.0000
## PhotoUnattractive            -2.401e+00  1.989e+00  -1.207   0.2326
## QualityPoor:PhotoAttractive   1.499e+00  2.813e+00   0.533   0.5962
## QualityPoor:PhotoUnattractive -2.298e+00  2.813e+00  -0.817   0.4175
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.447 on 54 degrees of freedom
## Multiple R-squared:  0.3552, Adjusted R-squared:  0.2955
## F-statistic:  5.95 on 5 and 54 DF,  p-value: 0.000188
```

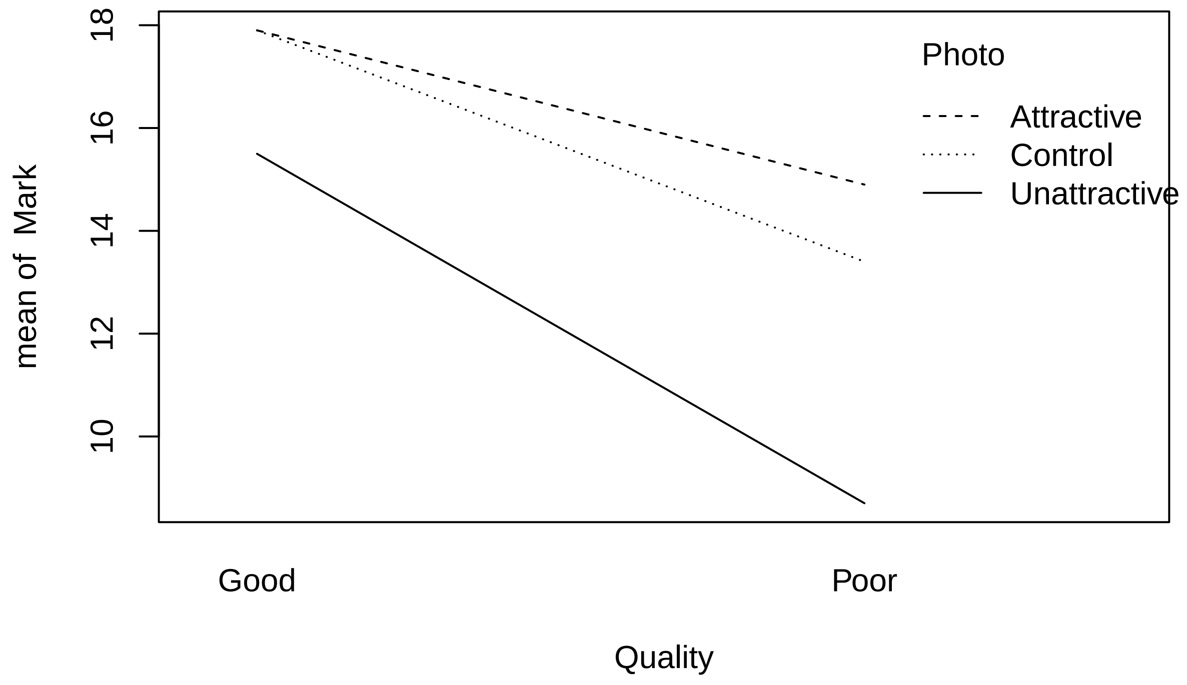
```
anova(EssayMarksLM1, EssayMarksLM2)
```

```
## Analysis of Variance Table
##
## Model 1: Mark ~ Quality
## Model 2: Mark ~ Quality + Photo
##   Res.Df    RSS Df Sum of Sq     F   Pr(>F)
## 1     58 1315.5
## 2     56 1104.5  2       211 5.349 0.007483 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

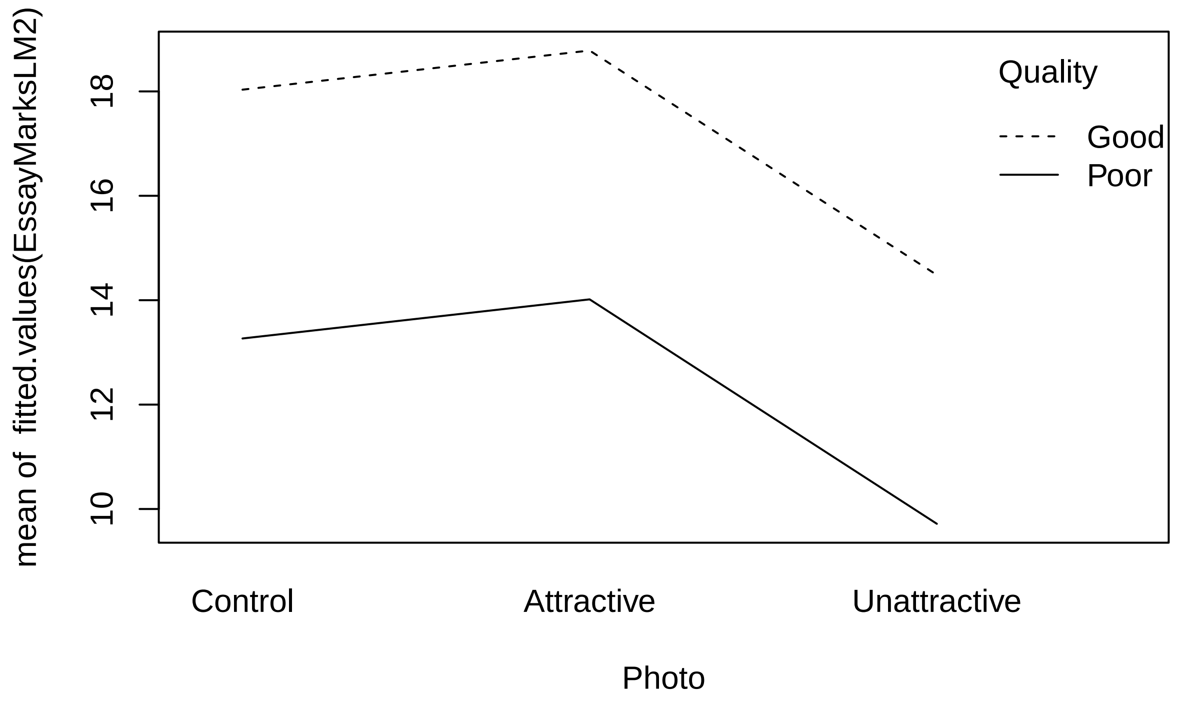
```
interaction.plot(Photo,Quality, Mark)
```



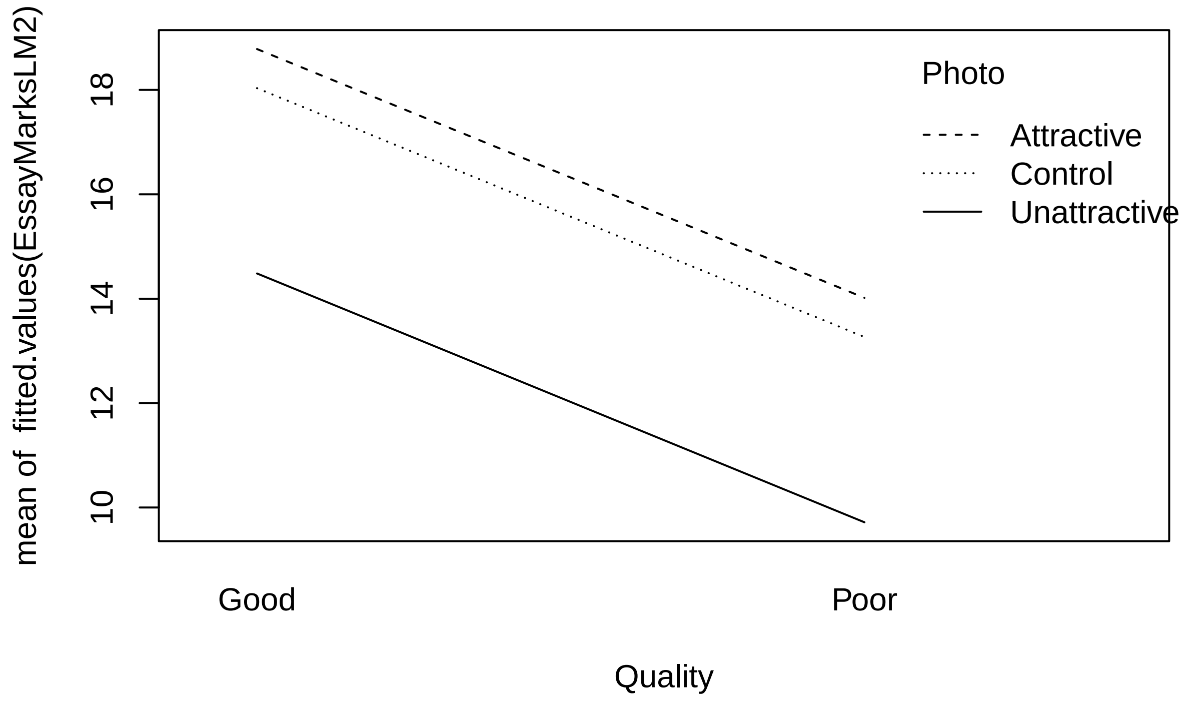
```r
interaction.plot(Quality, Photo, Mark)
```



```r
interaction.plot(Photo, Quality, fitted.values(EssayMarksLM2))
```



```r
interaction.plot(Quality, Photo, fitted.values(EssayMarksLM2))
```



```r
anova(EssayMarksLM2, EssayMarksLM3)
```

```
## Analysis of Variance Table
##
## Model 1: Mark ~ Quality + Photo
## Model 2: Mark ~ Quality * Photo
##   Res.Df    RSS Df Sum of Sq      F Pr(>F)
## 1     56 1104.5
## 2     54 1067.9  2    36.575 0.9247 0.4028
```

```
Photo_grp <- Photo
levels(Photo_grp) <- c( "Control+Attr", "Control+Atrr", "Unattractive")

EssayMarksLM4 <- lm(Mark ~ Quality*Photo_grp)
EssayMarksLM5 <- lm(Mark ~ Quality + Photo + Quality:Photo_grp)

AIC(EssayMarksLM1, EssayMarksLM2, EssayMarksLM3, EssayMarksLM4, EssayMarksLM5)
```

```
##                df      AIC
## EssayMarksLM1  3 361.5300
## EssayMarksLM2  5 355.0405
## EssayMarksLM3  7 357.0200
## EssayMarksLM4  7 357.0200
## EssayMarksLM5  7 357.0200
```

Binomial Regression

Here we look at doing logistic regression.

```
file_path <- "http://www.statslab.cam.ac.uk/~rds37/teaching/statistical_modelling/"
Myopia <- read.csv(paste0(file_path, "Myopia.csv"))
Myopia[1:3, ]
```

```
##   myopic gender sportHR readHR compHR studyHR TVHR mumMyopic dadMyopic
## 1      1 female      45      8      0       0   10       Yes       Yes
## 2      0 female       4      0      1       1    7       Yes       Yes
## 3      0 female      14      0      2       0   10        No        No
```

```
attach(Myopia)
```

```
MyopiaLogReg1 <- glm(myopic ~ ., data = Myopia, family = binomial)
summary(MyopiaLogReg1)
```

```
##
## Call:
## glm(formula = myopic ~ ., family = binomial, data = Myopia)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -2.670761   0.409491  -6.522 6.93e-11 ***
## gendermale  -0.365467   0.260908  -1.401 0.161288
## sportHR     -0.042545   0.018199  -2.338 0.019397 *
## readHR       0.095053   0.038217   2.487 0.012875 *
## compHR       0.030289   0.037993   0.797 0.425326
## studyHR     -0.064828   0.064246  -1.009 0.312949
## TVHR         0.008328   0.022853   0.364 0.715548
## mumMyopicYes  0.868446   0.261778   3.317 0.000908 ***
## dadMyopicYes  0.988224   0.262875   3.759 0.000170 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 480.08  on 617  degrees of freedom
## Residual deviance: 439.60  on 609  degrees of freedom
## AIC: 457.6
##
## Number of Fisher Scoring iterations: 5
```

```
MyopiaLogReg0 <- glm(myopic ~ 1, family = binomial)
anova(MyopiaLogReg0, MyopiaLogReg1, test = "LR")
```

```
## Analysis of Deviance Table
##
## Model 1: myopic ~ 1
## Model 2: myopic ~ gender + sportHR + readHR + compHR + studyHR + TVHR +
##     mumMyopic + dadMyopic
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1       617     480.08
## 2       609     439.60  8   40.478 2.61e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
MyopiaLogReg2 <- glm(myopic ~ . + mumMyopic:dadMyopic, data = Myopia, family = binomial)
summary(MyopiaLogReg2)
```

```
##
## Call:
## glm(formula = myopic ~ . + mumMyopic:dadMyopic, family = binomial,
##     data = Myopia)
##
## Coefficients:
##                              Estimate Std. Error z value Pr(>|z|)
## (Intercept)                 -3.026872   0.534178  -5.666 1.46e-08 ***
## gendermale                  -0.380252   0.260553  -1.459  0.14445
## sportHR                     -0.041641   0.018121  -2.298  0.02157 *
## readHR                       0.088813   0.038498   2.307  0.02106 *
## compHR                       0.028953   0.037755   0.767  0.44316
## studyHR                     -0.065668   0.064611  -1.016  0.30945
## TVHR                         0.007574   0.022877   0.331  0.74058
## mumMyopicYes                 1.392168   0.518563   2.685  0.00726 **
## dadMyopicYes                 1.508416   0.517159   2.917  0.00354 **
## mumMyopicYes:dadMyopicYes -0.744954   0.607457  -1.226  0.22007
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 480.08  on 617  degrees of freedom
## Residual deviance: 438.01  on 608  degrees of freedom
## AIC: 458.01
##
## Number of Fisher Scoring iterations: 6
```

Note to self, the fact that we're doing LR tests here (apporx) and not F-tests is becasue F tests are quite specific to the case where we have a normal LM!!!!

Think about what an F test is, is actually just a LR test for the normal Linear model, but is exact because things are so nice!

```
#Anova for the two models using LR (and Wilks)
anova(MyopiaLogReg1, MyopiaLogReg2, test="LR")
```

```
## Analysis of Deviance Table
##
## Model 1: myopic ~ gender + sportHR + readHR + compHR + studyHR + TVHR +
##     mumMyopic + dadMyopic
## Model 2: myopic ~ gender + sportHR + readHR + compHR + studyHR + TVHR +
##     mumMyopic + dadMyopic + mumMyopic:dadMyopic
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1       609     439.60
## 2       608     438.01  1   1.5918   0.2071
```

```r
#Q2

MyopiaLogReg3 <- glm(myopic~ .-compHR - TVHR, data = Myopia, family=binomial)

#Then we test the null that they're not signif against them being signif (ie. MyopiaLogReg1)

anova(MyopiaLogReg3, MyopiaLogReg1, test = 'LR')
```

```
## Analysis of Deviance Table
##
## Model 1: myopic ~ (gender + sportHR + readHR + compHR + studyHR + TVHR +
##     mumMyopic + dadMyopic) - compHR - TVHR
## Model 2: myopic ~ gender + sportHR + readHR + compHR + studyHR + TVHR +
##     mumMyopic + dadMyopic
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1       611     440.46
## 2       609     439.60  2  0.86043   0.6504
```

```r
#The test is not significant - hence we cant conclude that they are collectively significant.

#Q3
#Want to test whether dad and mum myopic are the same.

mumORdadMyopic <- (dadMyopic == 'Yes') | (mumMyopic == 'Yes')
mumORdadMyopic <- factor(mumORdadMyopic, labels = c("No", "Yes"))


MyopiaLogReg4 = glm(myopic ~ . - mumMyopic + mumORdadMyopic, data = Myopia, family = binomial)
```

```
## Warning in terms.formula(formula, data = data): 'varlist' has changed (from
## nvar=9) to new 10 after EncodeVars() -- should no longer happen!
```

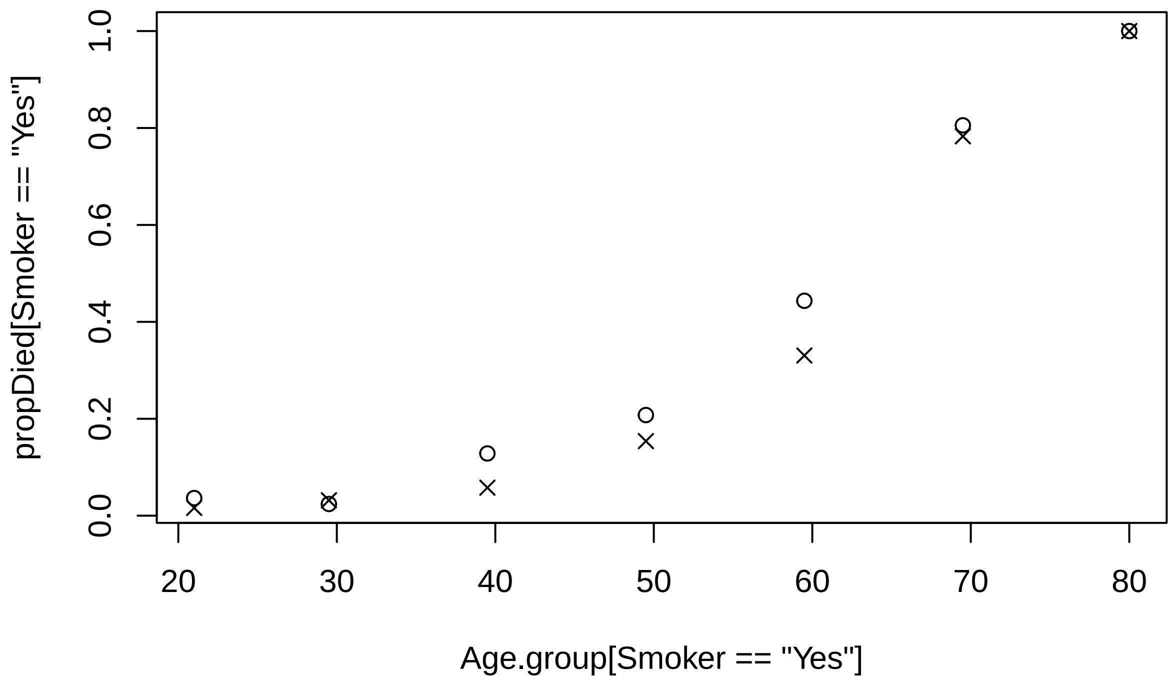
```r
AIC(MyopiaLogReg1, MyopiaLogReg4)
```

```
##               df      AIC
## MyopiaLogReg1  9 457.5992
## MyopiaLogReg4  9 460.4233
```

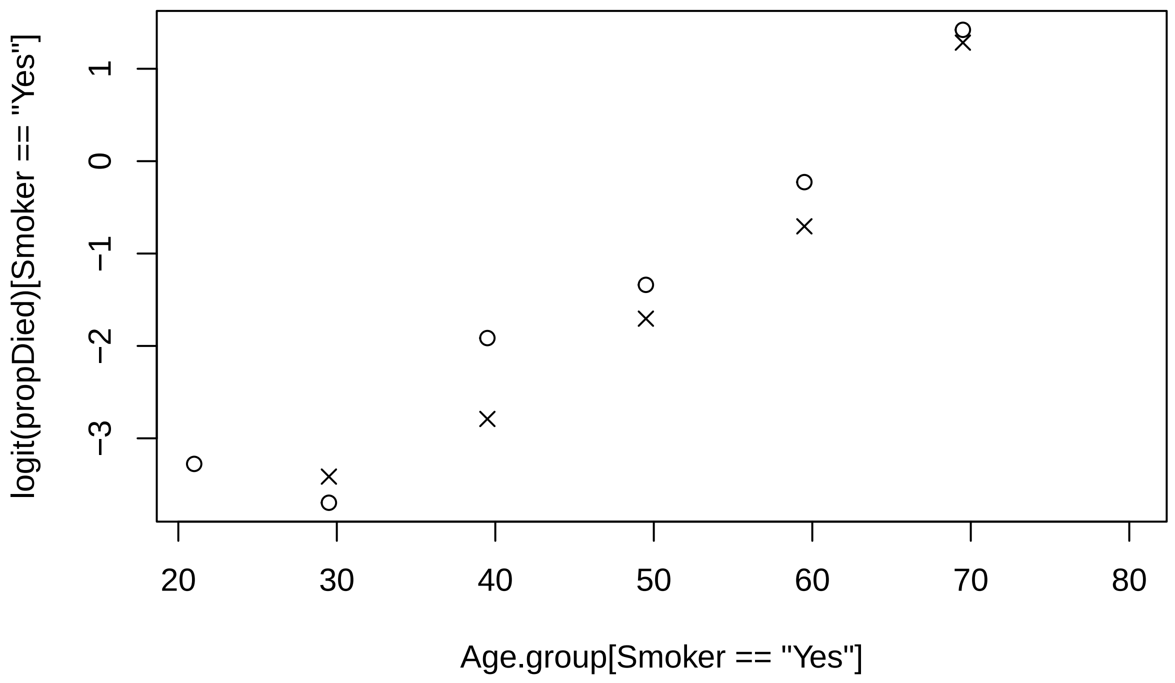
Second part of the practical.

```r
Smoking <- read.csv(paste(file_path, "Smoking.csv", sep =""))
attach(Smoking)

total <- Survived + Died
#I suppose we're doing binomial regression here, this time will be with n_i not all equal to one, and w

propDied <- Died/total
```

```r
plot( propDied[Smoker == 'Yes'] ~ Age.group[Smoker  == 'Yes'])
points(propDied[Smoker == "No"] ~ Age.group[Smoker == "No"], pch = 4)
```



```r
logit <- function(p) log(p/(1-p))
plot(logit(propDied)[Smoker =="Yes"] ~ Age.group[Smoker == "Yes"])
points(logit(propDied)[Smoker == "No"] ~ Age.group[Smoker == "No"], pch = 4)
```



```r
SmokingLogReg1 <- glm(propDied ~ Age.group + Smoker, family = binomial, weights = total)

summary(SmokingLogReg1)
```

```
##
## Call:
## glm(formula = propDied ~ Age.group + Smoker, family = binomial,
##     weights = total)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -7.687751   0.447646 -17.174   <2e-16 ***
## Age.group    0.124957   0.007274  17.178   <2e-16 ***
## SmokerYes    0.266053   0.168702   1.577    0.115
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 641.496  on 13  degrees of freedom
## Residual deviance:  32.572  on 11  degrees of freedom
## AIC: 85.568
##
## Number of Fisher Scoring iterations: 5
```

```
SmokingLogReg2 <- glm(propDied ~ Age.group + I(Age.group^2) + Smoker, family =  binomial , weights = to
SmokingLogReg3 <- glm(propDied ~ factor(Age.group) + Smoker, family = binomial, weights = total)

SmokingLogReg4 <- glm(propDied ~ Age.group + Smoker, family = binomial(link=probit), weights = total)
SmokingLogReg5 <- glm(propDied ~ Age.group + Smoker, family = binomial(link=cloglog), weights = total)


summary(SmokingLogReg2)
```

```
##
## Call:
## glm(formula = propDied ~ Age.group + I(Age.group^2) + Smoker,
##     family = binomial, weights = total)
##
## Coefficients:
##                  Estimate Std. Error z value Pr(>|z|)
## (Intercept)    -2.8658262  1.0224461  -2.803  0.00506 **
## Age.group      -0.0772521  0.0426067  -1.813  0.06981 .
## I(Age.group^2)  0.0019587  0.0004256   4.602 4.18e-06 ***
## SmokerYes       0.4306573  0.1764364   2.441  0.01465 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 641.496  on 13  degrees of freedom
## Residual deviance:  11.625  on 10  degrees of freedom
## AIC: 66.621
##
## Number of Fisher Scoring iterations: 4
```

```
summary(SmokingLogReg3)
```

```
##
```

```
## Call:
## glm(formula = propDied ~ factor(Age.group) + Smoker, family = binomial,
##     weights = total)
##
## Coefficients:
##                         Estimate Std. Error z value Pr(>|z|)
## (Intercept)              -3.8601     0.5939  -6.500 8.05e-11 ***
## factor(Age.group)29.5     0.1201     0.6865   0.175 0.861178
## factor(Age.group)39.5     1.3411     0.6286   2.134 0.032874 *
## factor(Age.group)49.5     2.1134     0.6121   3.453 0.000555 ***
## factor(Age.group)59.5     3.1808     0.6006   5.296 1.18e-07 ***
## factor(Age.group)69.5     5.0880     0.6195   8.213  < 2e-16 ***
## factor(Age.group)80      27.8073 11293.1430   0.002 0.998035
## SmokerYes                 0.4274     0.1770   2.414 0.015762 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 641.4963  on 13  degrees of freedom
## Residual deviance:   2.3809  on  6  degrees of freedom
## AIC: 65.377
##
## Number of Fisher Scoring iterations: 20
```

```r
summary(SmokingLogReg4)
```

```
##
## Call:
## glm(formula = propDied ~ Age.group + Smoker, family = binomial(link = probit),
##     weights = total)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -4.173847   0.215932 -19.329   <2e-16 ***
## Age.group    0.068313   0.003575  19.108   <2e-16 ***
## SmokerYes    0.117996   0.094145   1.253     0.21
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 641.496  on 13  degrees of freedom
## Residual deviance:  42.796  on 11  degrees of freedom
## AIC: 95.792
##
## Number of Fisher Scoring iterations: 5
```

```r
summary(SmokingLogReg5)
```

```
##
## Call:
## glm(formula = propDied ~ Age.group + Smoker, family = binomial(link = cloglog),
##     weights = total)
##
```

```
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -6.692406   0.352901 -18.964   <2e-16 ***
## Age.group    0.101179   0.005388  18.779   <2e-16 ***
## SmokerYes    0.226478   0.125238   1.808   0.0705 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 641.50  on 13  degrees of freedom
## Residual deviance:  11.31  on 11  degrees of freedom
## AIC: 64.306
##
## Number of Fisher Scoring iterations: 5
```

Prac 7 Bin and Poi

Let's first try and do some plotting!