R Code Chapter 4

Witold Wolski

December 23, 2016

4.4.1 Model building for the beetle data

```
getwd()
```

```
## [1] "C:/Users/wewol/Google Drive/Courses/Sheffield/MAS6003TaskSolutions"
```

```
load("data/MAS367-GLMs.RData", envir = e <- new.env())
grep("beetle",names(e))
```

```
## [1] 8
```

Fitting minimal model

```
beetle<-e$beetle
lapply(beetle,class)
```

```
## $conc
## [1] "numeric"
##
## $number
## [1] "integer"
##
## $dead
## [1] "integer"
##
## $propn.dead
## [1] "numeric"
```

```
head(beetle)
```

```
##     conc number dead propn.dead
## 1 1.6907     59    6  0.1016949
## 2 1.7242     60   13  0.2166667
## 3 1.7552     62   18  0.2903226
## 4 1.7842     56   28  0.5000000
## 5 1.8113     63   52  0.8253968
## 6 1.8369     59   53  0.8983051
```

```
null.glm <- glm(propn.dead ~ 1, family=binomial(logit), weights=number, data=beetle)
null.glm
```

```
##
## Call:  glm(formula = propn.dead ~ 1, family = binomial(logit), data = beetle,
##     weights = number)
##
## Coefficients:
## (Intercept)
##      0.4263
##
## Degrees of Freedom: 7 Total (i.e. Null);  7 Residual
## Null Deviance:       284.2
## Residual Deviance: 284.2     AIC: 312.4
```

Adding a log concentration term

```
linear.glm <- glm(propn.dead~1+conc, binomial(logit), weights = number, data=beetle)
summary(linear.glm)
```

```
##
## Call:
## glm(formula = propn.dead ~ 1 + conc, family = binomial(logit),
##     data = beetle, weights = number)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.5941  -0.3944   0.8329   1.2592   1.5940
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -60.717      5.181  -11.72   <2e-16 ***
## conc          34.270      2.912   11.77   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 284.202  on 7  degrees of freedom
## Residual deviance:  11.232  on 6  degrees of freedom
## AIC: 41.43
##
## Number of Fisher Scoring iterations: 4
```

Example of computing χ^2 values in R.

```
sum(residuals(linear.glm, type = "pearson")^2)
```

```
## [1] 10.02682
```

Numbers from point 3: Analysis of deviance in nested models

```
null.glm$deviance
```

```
## [1] 284.2024
```

```
linear.glm$deviance
```

```
## [1] 11.23223
```

```
null.glm$deviance - linear.glm$deviance
```

```
## [1] 272.9702
```

```
qchisq(0.95,1)
```

```
## [1] 3.841459
```

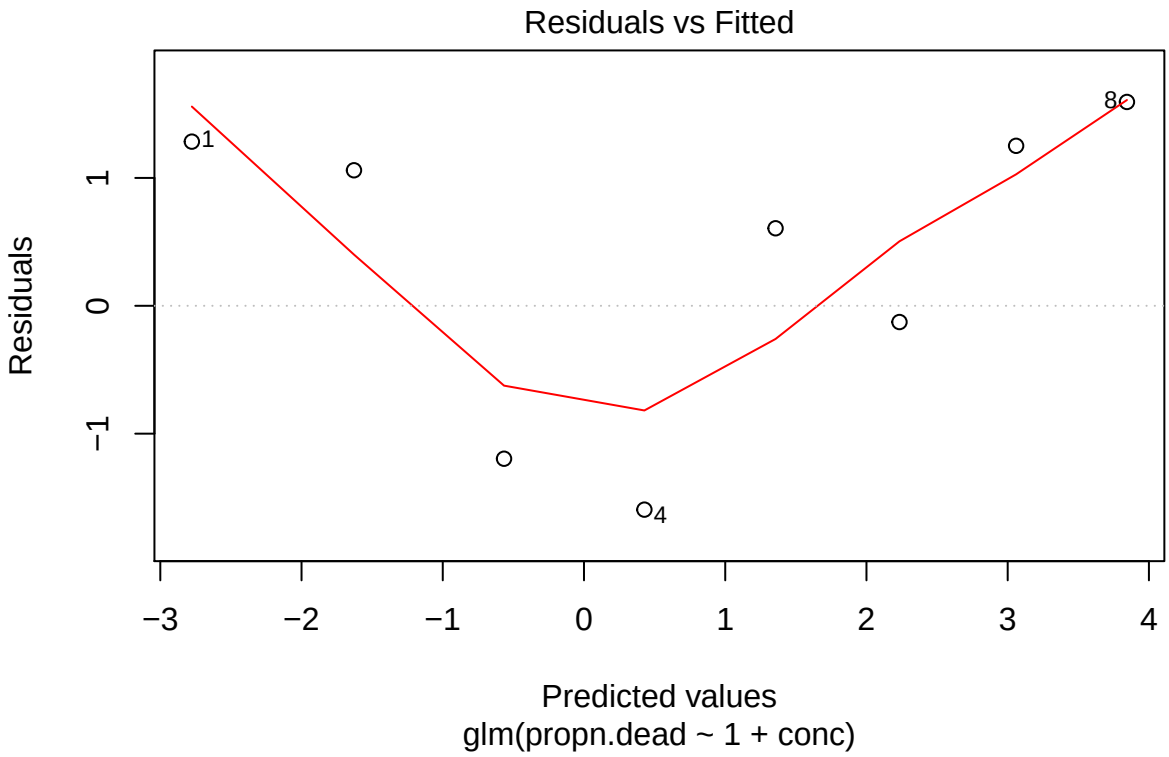
Is fit acceptable ?

```
qchisq(0.95,6)
```

```
## [1] 12.59159
```

Plotting Pearson (?) residuals against predicted values

```
plot(linear.glm,which = 1)
```



Fitting quadratic model

```
quad.glm <- glm(propn.dead ~ 1 +conc+ I(conc^2), binomial(logit), weights= number, data= beetle)
quad.glm
```

```
##
## Call:  glm(formula = propn.dead ~ 1 + conc + I(conc^2), family = binomial(logit),
##     data = beetle, weights = number)
##
## Coefficients:
## (Intercept)         conc    I(conc^2)
##       431.1       -520.6        156.4
##
## Degrees of Freedom: 7 Total (i.e. Null);  5 Residual
## Null Deviance:       284.2
## Residual Deviance: 3.195     AIC: 35.39
```

Change in deviance

```
linear.glm$deviance - quad.glm$deviance
```

```
## [1] 8.037326
```

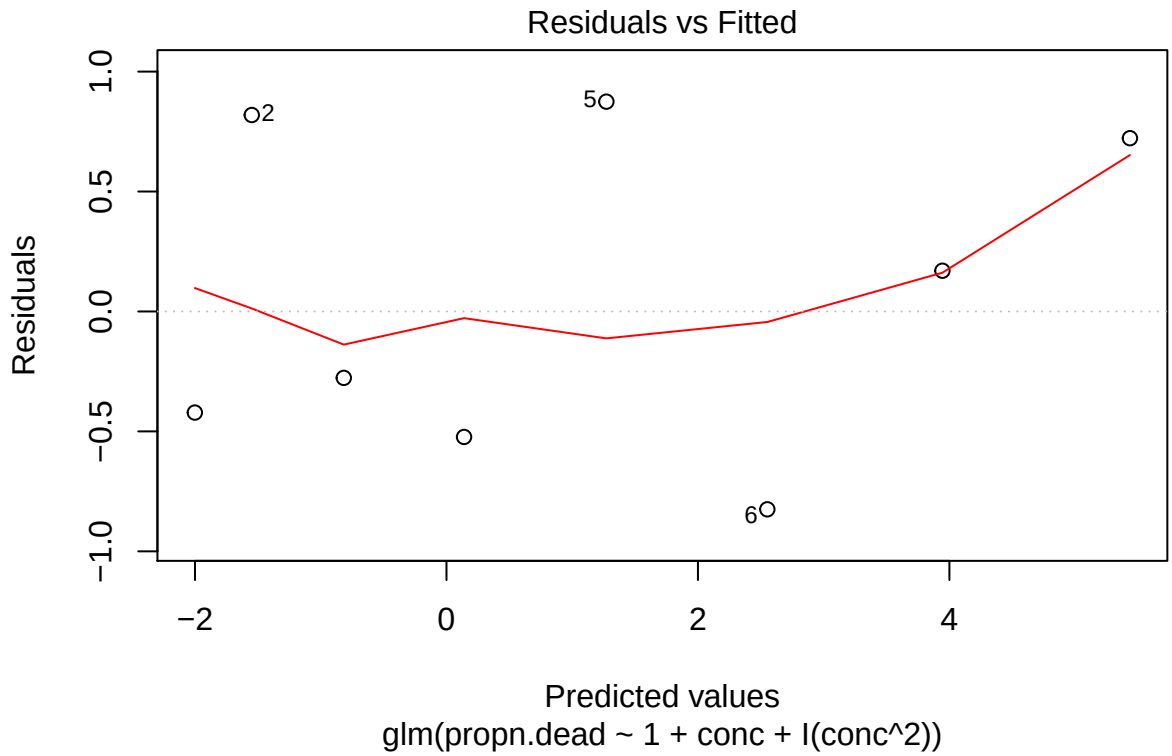
```r
qchisq(0.95,1)
```

```
## [1] 3.841459
```

```r
qchisq(0.95,5)
```

```
## [1] 11.0705
```

```r
plot(quad.glm,which = 1)
```



10 Wald t-test

```r
summary(quad.glm)
```

```
##
## Call:
## glm(formula = propn.dead ~ 1 + conc + I(conc^2), family = binomial(logit),
##     data = beetle, weights = number)
##
## Deviance Residuals:
##       1        2        3        4        5        6        7        8
```

```
## -0.4215   0.8189  -0.2769  -0.5232   0.8746  -0.8249   0.1698   0.7224
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   431.11     180.65   2.386  0.01702 *
## conc         -520.62     204.52  -2.546  0.01091 *
## I(conc^2)     156.41      57.86   2.703  0.00687 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 284.2024  on 7  degrees of freedom
## Residual deviance:   3.1949  on 5  degrees of freedom
## AIC: 35.393
##
## Number of Fisher Scoring iterations: 4
```

```
2*pnorm(-156.41/57.86)
```

```
## [1] 0.006866524
```

4.4.2

```
anova(linear.glm, quad.glm,test="Chi")
```

```
## Analysis of Deviance Table
##
## Model 1: propn.dead ~ 1 + conc
## Model 2: propn.dead ~ 1 + conc + I(conc^2)
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1         6    11.2322
## 2         5     3.1949  1   8.0373 0.004582 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

4.4.3

$AIC = -2l + 2p$

$\frac{l_min - l_mod}{l_min}$

```
quad.glm$aic
```

```
## [1] 35.39294
```

```
l_mod<-length(quad.glm$coefficients) - 0.5*quad.glm$aic
l_min<-length(null.glm$coefficients) - 0.5*null.glm$aic
(l_min-l_mod)/l_min
```

```
## [1] 0.9053064
```

4.4.4 Calculating residuals for the beetle quadratic model

```
residuals(quad.glm,type = "pearson")
```

```
##          1          2          3          4          5          6
## -0.4122545  0.8425072 -0.2754990 -0.5238566  0.8516013 -0.8716078
##          7          8
##  0.1654446  0.5113471
```

```
y=beetle$propn.dead
fitted = quad.glm$fitted
n = beetle$number
```

Lets define function:

```
pearson_resid <- function(y, fitted, n){
  (y-fitted)/sqrt(fitted * (1-fitted)/n)
}
```

and run it:

```
pearson_resid(y, fitted, n)
```

```
##          1          2          3          4          5          6
## -0.4122545  0.8425072 -0.2754990 -0.5238566  0.8516013 -0.8716078
##          7          8
##  0.1654446  0.5113471
```

Other residuals are available:

Def function to compute d_i deviance residuals:

```
d_i_deviance <- function(y, fitted, n){
  sign(y- fitted)*sqrt(-2*n*(y*log(fitted/y)+(1-y)*log((1-fitted)/(1-y+0.0000001))))
}

d_i_deviance(y, fitted, n)
```

```
##          1          2          3          4          5          6
## -0.4215475  0.8189611 -0.2769161 -0.5232272  0.8746384 -0.8249153
##          7          8
##  0.1698457  0.7223680
```

compare with r computed values

```
residuals(quad.glm,type="deviance")
```

```
##          1          2          3          4          5          6
## -0.4215335  0.8189537 -0.2768937 -0.5232165  0.8746312 -0.8249082
##          7          8
##  0.1698092  0.7223680
```

Task 12

```
residuals(linear.glm, "pearson")[1]
```

```
##        1
## 1.409296
```

```
residuals(linear.glm, "deviance")[1]
```

```
##        1
## 1.283678
```

```
linear.glm$fitted.values[1]
```

```
##          1
## 0.05860103
```

```
pearson_resid(linear.glm$y, linear.glm$fitted.values, linear.glm$prior.weights)
```

```
##          1          2          3          4          5          6
##  1.4092960  1.1011003 -1.1762596 -1.6123815  0.5944454 -0.1281090
##          7          8
##  1.0914228  1.1331102
```

```
d_i_deviance(linear.glm$y, linear.glm$fitted.values, linear.glm$prior.weights)
```

```
##          1          2          3          4          5          6
##  1.2836823  1.0596957 -1.1961175 -1.5941279  0.6061509 -0.1272048
##          7          8
##  1.2510760  1.5939850
```

4.4.5 Example 2 of model building : plant anthers

```
anthers <- e$anthers
```

Plot data

```
head(anthers)
```

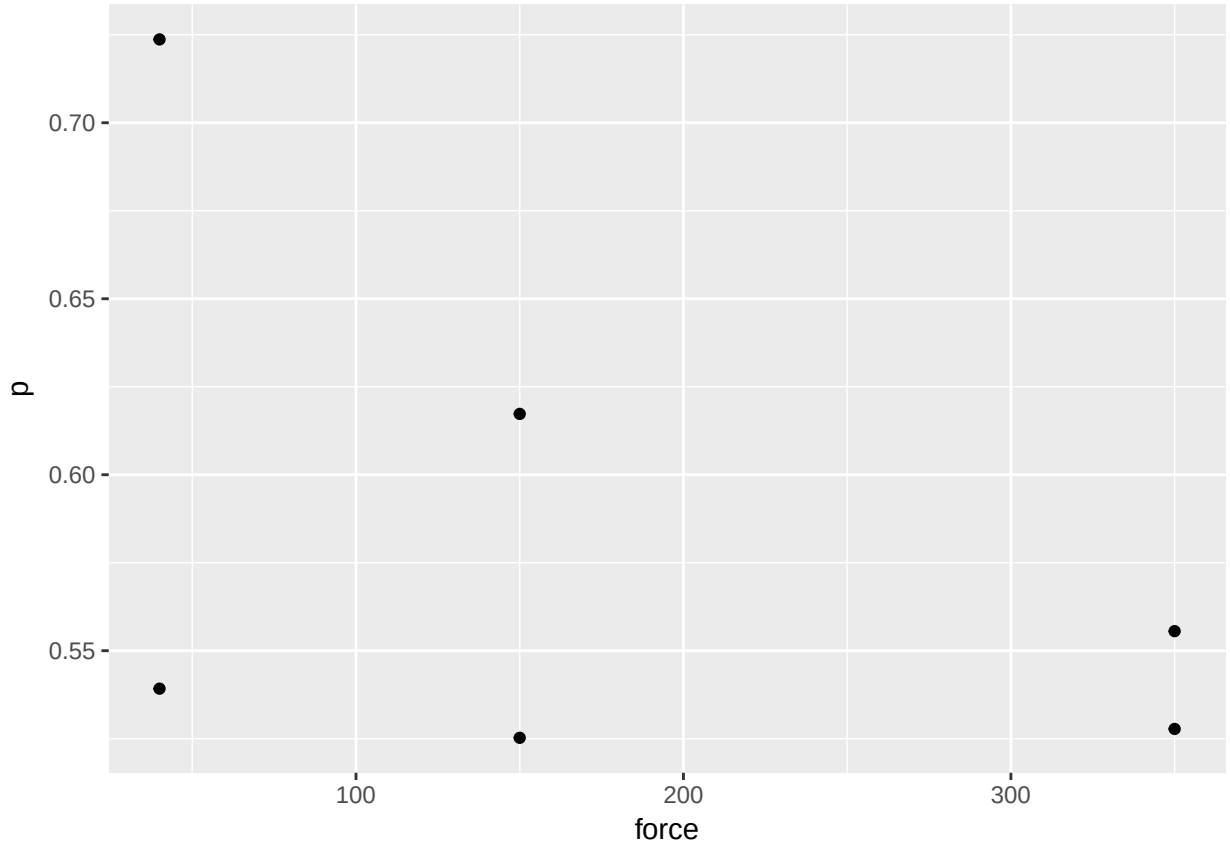
```
##       group force prepared obtained
## 1   control    40      102       55
## 2   control   150       99       52
## 3   control   350      108       57
## 4 treatment    40       76       55
## 5 treatment   150       81       50
## 6 treatment   350       90       50
```

```
anthers$p <- anthers$obtained/anthers$prepared

library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 3.3.2
```

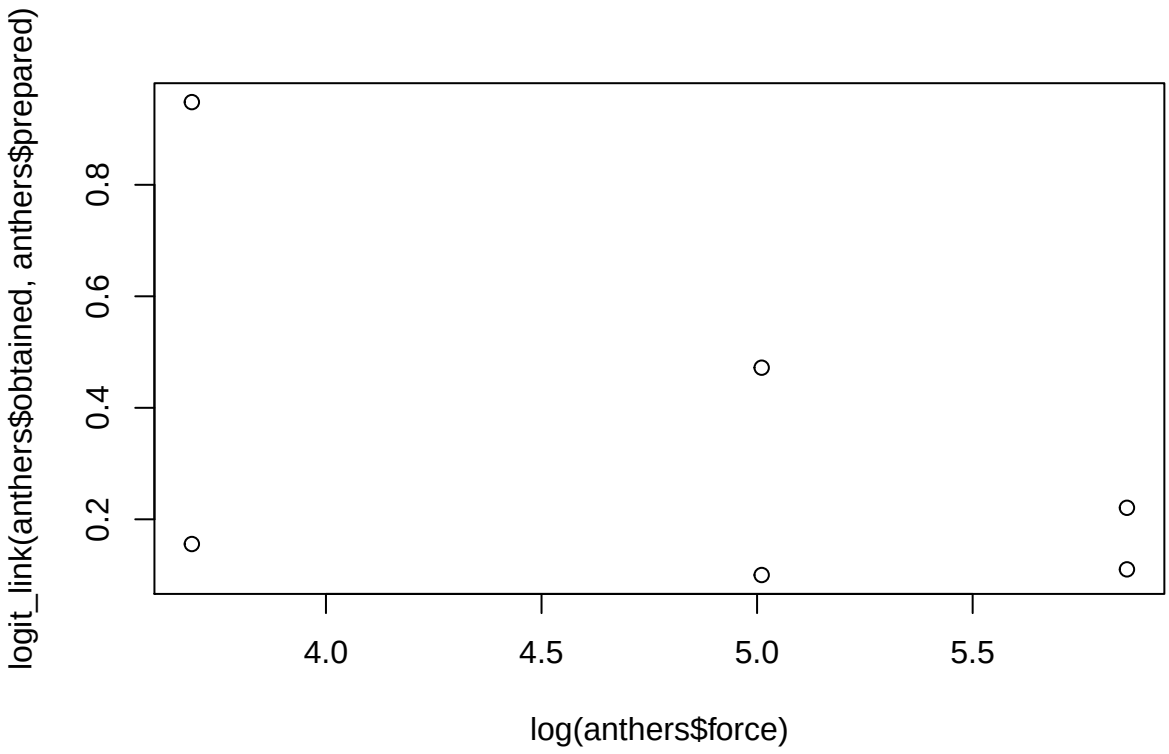
```
ggplot(anthers, aes(force,p), colour= group) + geom_point()
```



2 consider logit link

```
logit_link <- function(s,n){
  y <- (s + 1/2)/(n+1)
  log(y/(1-y))
}
```

```
plot(log(anthers$force),logit_link(anthers$obtained,anthers$prepared))
```



Minimal Model

```
glm.0 <-glm(p ~ 1, weights = prepared, data = anthers, binomial(logit))
summary(glm.0)
```

```
##
## Call:
## glm(formula = p ~ 1, family = binomial(logit), data = anthers,
##     weights = prepared)
##
## Deviance Residuals:
##       1        2        3        4        5        6
## -0.7029  -0.9716  -0.9622   2.7078   0.7965  -0.3483
```

```
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.29713    0.08576   3.465 0.000531 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 10.452  on 5  degrees of freedom
## Residual deviance: 10.452  on 5  degrees of freedom
## AIC: 42.02
##
## Number of Fisher Scoring iterations: 3
```

Model 1

$\eta = \alpha$. Dobson is refernced, to make the lecture notes more confusing model 1 in dobson is model 3 in notes.

```
anthers$p <- anthers$obtained/anthers$prepared
anthers$logForce <- log(anthers$force)
glm.1 <-glm(p ~ group, weights = prepared, data = anthers, family = binomial(logit))
summary(glm.1)
```

```
##
## Call:
## glm(formula = p ~ group, family = binomial(logit), data = anthers,
##     weights = prepared)
##
## Deviance Residuals:
##        1         2         3         4         5         6
##  0.17150  -0.10947  -0.06177   1.77208  -0.19040  -1.39686
##
## Coefficients:
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept)      0.1231     0.1140   1.080   0.2801
## grouptreatment   0.3985     0.1741   2.289   0.0221 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 10.452  on 5  degrees of freedom
## Residual deviance:  5.173  on 4  degrees of freedom
## AIC: 38.741
##
## Number of Fisher Scoring iterations: 3
```

```
summary.lm(glm.1)
```

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

```
## glm(formula = p ~ group, family = binomial(logit), data = anthers,
##     weights = prepared)
##
## Weighted Residuals:
##       1       2       3       4       5       6
##  0.17144 -0.10949 -0.06178  1.73385 -0.19074 -1.41234
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)      0.1231     0.1285   0.958    0.392
## grouptreatment   0.3985     0.1963   2.030    0.112
##
## Residual standard error: 1.127 on 4 degrees of freedom
## Multiple R-squared:  0.05731,    Adjusted R-squared:  -0.1784
## F-statistic: 0.2432 on 1 and 4 DF,  p-value: 0.6478
```

Model 2

```
anthers$p <- anthers$obtained/anthers$prepared
anthers$logForce <- log(anthers$force)
glm.2 <-glm(p ~ group + logForce, weights = prepared, data = anthers, family = binomial(logit))
summary(glm.2)
```

```
##
## Call:
## glm(formula = p ~ group + logForce, family = binomial(logit),
##     data = anthers, weights = prepared)
##
## Deviance Residuals:
##       1         2         3         4         5         6
## -0.74964  -0.00509   0.72746   0.99006  -0.13512  -0.72744
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)     0.87673    0.48701   1.800   0.0718 .
## grouptreatment  0.40684    0.17462   2.330   0.0198 *
## logForce       -0.15459    0.09702  -1.593   0.1111
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 10.4520  on 5  degrees of freedom
## Residual deviance:  2.6188  on 3  degrees of freedom
## AIC: 38.187
##
## Number of Fisher Scoring iterations: 3
```

```
summary.lm(glm.2)
```

```
##
```

```
## Call:
## glm(formula = p ~ group + logForce, family = binomial(logit),
##     data = anthers, weights = prepared)
##
## Weighted Residuals:
##         1         2         3         4         5         6
## -0.752149 -0.005092  0.727404  0.975233 -0.135287 -0.730611
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)     0.87673    0.45321   1.934   0.1485
## grouptreatment  0.40684    0.16250   2.504   0.0874 .
## logForce       -0.15459    0.09029  -1.712   0.1854
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9306 on 3 degrees of freedom
## Multiple R-squared:  0.1477, Adjusted R-squared:  -0.4204
## F-statistic:  0.26 on 2 and 3 DF,  p-value: 0.7868
```

Model 3

```
lapply(anthers, class)
```

```
## $group
## [1] "factor"
##
## $force
## [1] "integer"
##
## $prepared
## [1] "integer"
##
## $obtained
## [1] "integer"
##
## $p
## [1] "numeric"
##
## $logForce
## [1] "numeric"
```

```
glm.3 <-glm(p ~ group * logForce , weights = prepared, data = anthers, family = binomial(logit))
summary(glm.3)
```

```
##
## Call:
## glm(formula = p ~ group * logForce, family = binomial(logit),
##     data = anthers, weights = prepared)
##
## Deviance Residuals:
```

```
##        1         2         3         4         5         6
##  0.03611  -0.09370   0.05466   0.04305  -0.09855   0.05560
##
## Coefficients:
##                         Estimate Std. Error z value Pr(>|z|)
## (Intercept)              0.23389    0.62839   0.372   0.7097
## grouptreatment           1.97711    0.99802   1.981   0.0476 *
## logForce                -0.02274    0.12685  -0.179   0.8577
## grouptreatment:logForce -0.31862    0.19888  -1.602   0.1091
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 10.451974  on 5  degrees of freedom
## Residual deviance:  0.027728  on 2  degrees of freedom
## AIC: 37.596
##
## Number of Fisher Scoring iterations: 3
```

Task 13

```
glm.1 <-glm(p ~ group, weights = prepared, data = anthers, family = binomial(logit))
glm.1
```

```
##
## Call:  glm(formula = p ~ group, family = binomial(logit), data = anthers,
##     weights = prepared)
##
## Coefficients:
##    (Intercept)  grouptreatment
##         0.1231          0.3985
##
## Degrees of Freedom: 5 Total (i.e. Null);  4 Residual
## Null Deviance:       10.45
## Residual Deviance: 5.173     AIC: 38.74
```

```
glm.2 <-glm(p ~ group + force, weights = prepared, data = anthers, family = binomial(logit))
glm.2
```

```
##
## Call:  glm(formula = p ~ group + force, family = binomial(logit), data = anthers,
##     weights = prepared)
##
## Coefficients:
##    (Intercept)  grouptreatment           force
##       0.306643        0.405554       -0.000997
##
## Degrees of Freedom: 5 Total (i.e. Null);  3 Residual
## Null Deviance:       10.45
## Residual Deviance: 2.922     AIC: 38.49
```

The conlusions od not change at all. The residual deviances are for both models are rather similar.

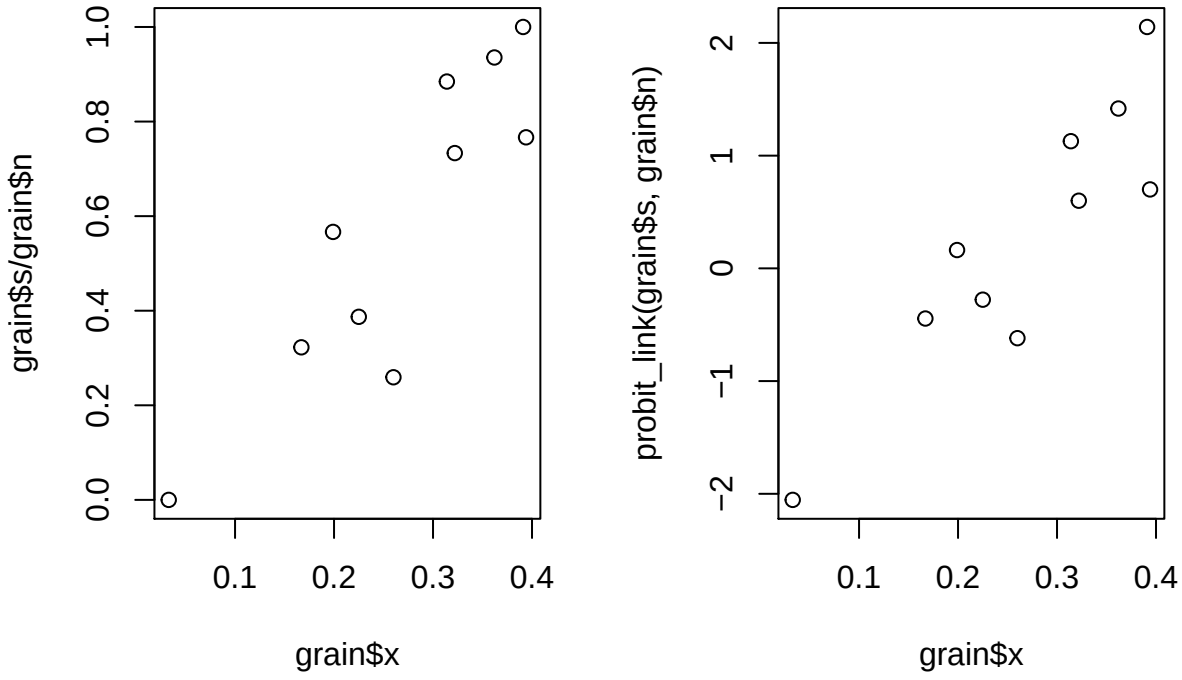
4.5.1 (Task 14)

```
grain <- e$grain



probit_link<- function(s,n){
  y<-(s+0.5)/(n+1)
  (qnorm(y))
}

par(mfrow=c(1,2))
plot(grain$x, grain$s/ grain$n)
plot(grain$x, probit_link(grain$s, grain$n))
```




```
head(grain)
```

```
##       x  n  s
## 1 0.394 30 23
## 2 0.391 30 30
```

```
## 3 0.362 31 29
## 4 0.322 30 22
## 5 0.314 26 23
## 6 0.260 27  7
```

```
grain$p_star <- (grain$s + 0.5)/ (grain$n+1)
grain$p <- (grain$s)/ (grain$n)
```

2 Null deviance

```
glm(p ~ 1 , weights = grain$n, data= grain, family=binomial(link= 'probit'))
```

```
##
## Call:  glm(formula = p ~ 1, family = binomial(link = "probit"), data = grain,
##     weights = grain$n)
##
## Coefficients:
## (Intercept)
##      0.2444
##
## Degrees of Freedom: 9 Total (i.e. Null);  9 Residual
## Null Deviance:       138
## Residual Deviance: 138   AIC: 167.6
```

Can reproduce results from script only with p_start. But so far we only fitted models with $p = s/n$. Using p_star also produces a warning

```
#glm(p_star ~ 1 + x , weights = grain$n, data= grain, family=binomial(link= 'probit'))
glin <- glm(p ~ 1 + x , weights = grain$n, data= grain, family=binomial(link= 'probit'))
names(glin)
```

```
##  [1] "coefficients"      "residuals"         "fitted.values"
##  [4] "effects"           "R"                 "rank"
##  [7] "qr"                "family"            "linear.predictors"
## [10] "deviance"          "aic"               "null.deviance"
## [13] "iter"              "weights"           "prior.weights"
## [16] "df.residual"       "df.null"           "y"
## [19] "converged"         "boundary"          "model"
## [22] "call"              "formula"           "terms"
## [25] "data"              "offset"            "control"
## [28] "method"            "contrasts"         "xlevels"
```

```
glin$deviance
```

```
## [1] 35.47649
```

```
glin$null.deviance
```

```
## [1] 138.0006
```

```
glin$null.deviance-glin$deviance
```

```
## [1] 102.5241
```

The change in deviance (94 on 1 df) is clearly very high showing an improvement of the model with a linear term over the null model (overwhelming evidence to reject $H0 : \beta_1 = 0$)

But the actual deviance for the model is high 35.4764889 providing evidence against this Binomial model $\chi^2_{8,0.9995}$:

```
qchisq(0.9995,8 )
```

```
## [1] 27.86805
```

5

Binomial model estimating ϕ as $D(y, \hat{\mu})/(n - p)$.

```
glin$deviance / (glin$df.residual)
```

```
## [1] 4.434561
```

```
glq <- glm(p ~ 1 + x , weights = grain$n, data= grain, family=quasi)
glq$deviance
```

```
## [1] 5.633424
```

6

Standard errors for parameter estimates are then multiplied by:

```
sqrt(glin$deviance / glin$df.residual)
```

```
## [1] 2.10584
```

7

Improvement over the null model on 1 df.

```
(glin$null.deviance-glin$deviance)/ (glin$deviance / glin$df.residual)
```

```
## [1] 23.11934
```

The scaled deviance for this model:

```
glin$deviance/(glin$deviance / glin$df.residual)
```

```
## [1] 8
```

```
qchisq(0.95,7)
```

```
## [1] 14.06714
```

```
sglin <- summary(glin)
```

with $e.s.e(\hat{\beta}_1) =$

```
sglin$coefficients[2,"Std. Error"]
```

```
## [1] 0.9810898
```

```
sglin$coefficients[2,"Std. Error"] * sqrt(glin$deviance / glin$df.residual)
```

```
## [1] 2.066018
```

```
ese_b=sglin$coefficients[2,"Std. Error"] * sqrt(glin$deviance / glin$df.residual)
```

So the wald test statistics

```
coefficients(glin)[2]
```

```
##        x
## 8.641606
```

```
coefficients(glin)[2] / ese_b
```

```
##        x
## 4.182735
```

```
t = coefficients(glin)[2] / ese_b
```

The slope is highly significant.

```
1-pt(t, df=7)
```

```
##           x
## 0.00206159
```

using quasi (same as subsection 5)

```r
glq <- glm(p_star ~ 1 + x , weights = grain$n, data= grain, family=quasi())
glq$deviance
```

```
## [1] 5.254139
```

Somehow the numbers obtained are not equal to those in the lecture notes (3.62): Also trying to run the glm with `quasi(link = "probit", variance = "constant")` does fail.