

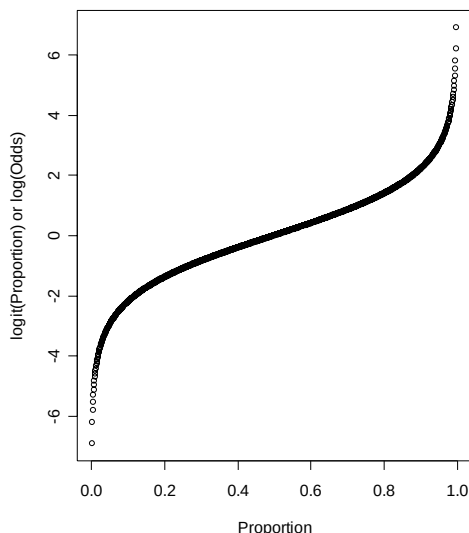
Proportion Data: Logistic Regression

Introduction

Another type of data that is often difficult to analyse with normal models are on proportions (or percentages). As with binary data the response is bounded at zero but now the upper limit is the size of each sample. A second difference is that rather than only two values being possible, zero or one, any intermediate value is possible between zero and the sample size. GLMs again prove very useful in dealing with the bounded nature of the data by using the S-shaped logistic curve that is asymptotic towards plus and minus infinity. Note that some types of proportion data are better analysed with least squares regression and a normal error distribution following transformation, percentage cover after arcsine transformation for example. Data on changes in a continuous response are often best analysed by either analysis of covariance - using the initial measurement as the covariate in an analysis of the final response - or by dividing the final measure by the initial measure and analysing the resulting derived variable after natural log transformation.

The analysis of proportions is similar to the analysis of binary data in that we can use the logistic curve, linearised by using logits, in combination with a binomial error structure. The difference is that rather than have only a single response variable consisting of zeros and ones we need two columns: one to specify how many times something was observed to happen (p) and one for how many times it did not (q = 1 - p). The advantage of this approach is that the sample size (the sum of the two columns) can be used to weight the regression to give more influence to larger, more reliable samples. The sample size, the sum of p and q, is termed the binomial denominator. We can see how the sigmoidal shape of the logistic regression using log odds can be used to deal with the bounded nature of binary data with the figure from the practical on binary data:

```
p <- (1:999) / 1000
logit.p <- log(p / (1 - p))
plot(logit.p ~ p, xlab = "Proportion", ylab = "logit(Proportion) or log(Odds)")
```



Recall that the advantage of using odds is that by taking the log of the odds, the logits, we can make the sigmoidal logistic relationship linear for use in the regression as follows:

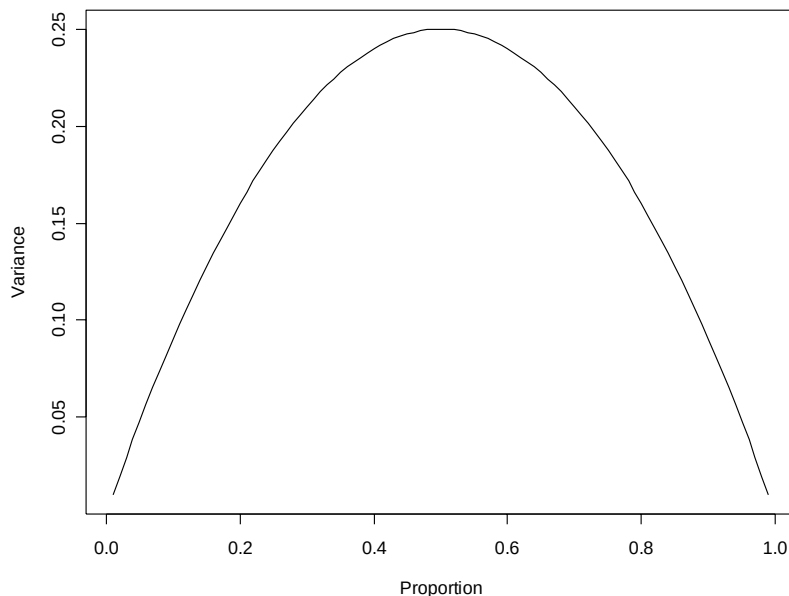
$$\ln\left(\frac{p}{q}\right) = a + bx$$

Since we can get a linear relationship for use with proportion data by using the log-odds $\ln(p/q)$ why not simply analyse the logit transformed data with a normal linear model? The problem is that the residual error variation is not normally distributed. The variance of a binomial distribution with a mean np is:

$$s^2 = npq$$

where p are the successes, q the failures ($1 - p$) and n the total sample size (the binomial denominator). Imagine we conduct a series of 100 replicate trials which range from being nearly all 'failures' (e.g. $p = 1\%$ death) to nearly all 'successes' ($p = 99\%$). We can plot the variance as follows:

```
p <- seq(0.01, 0.99, 0.01)
q <- 1-p
n <- rep(1, 99)
s2 <- n * p * q
plot(s2 ~ p, typ="l", xlab="Proportion", ylab="Variance")
```



The variance is low at the extremes - when p is low and q or when p is high and q is low - highest at intermediate values. When p is low and q approaches 1 the binomial distribution becomes close to the Poisson distribution since variance becomes close to the mean.

In binomial GLMs the estimation of the deviance can be illustrated with this simple example from Crawley (2002). For proportion data the deviance is:

$$2 \sum y \log \left[\frac{y}{\hat{y}} \right] + (n - y) \log \left[\frac{n - y}{n - \hat{y}} \right]$$

where, y is the observed number of successes, $\hat{y}$ is the predicted fitted value and n the binomial denominator (the total from which we observed y successes). Imagine an experiment with two treatments, each with two replicates, where we got 3 out of 6 and 4 out of 10 for one treatment and 5 out of 9 and 8 out of 12 for the other. First, calculate the average of two proportions by summing the successes and dividing by the total number of attempts:

$$3 + 4 + 5 + 8 = 20 / 6 + 10 + 9 + 12 = 37 = 0.5405$$

Next we calculate the four expected counts, $\hat{y}$:

```
y <- c(3, 4, 5, 8)
n <- c(6, 10, 9, 12)
n*20/37
```

```
[1] 3.243243 5.405405 4.864865 6.486486
```

The total deviance, calculated in two portions, is:

```
p1 <- 2*(3*log(3/3.243)+3*log(3/(6-3.243))+4*log(4/5.405)+6*log(6/(10-5.405)))
p2 <- 2*(5*log(5/4.865)+4*log(4/(9-4.865))+8*log(8/6.486)+4*log(4/(12-6.486)))
total.deviance <- p1+p2 ; total.deviance
```

```
[1] 1.629673
```

To get the residual deviance we need to calculate a new set of fitted values based on the two treatment means. The mean proportion in the first treatment was:

```
mean1 <- 7/16 ; mean1
mean2 <- 13/21 ; mean2
```

```
[1] 0.4375
```

```
[1] 0.6190476
```

so the fitted values are:

```
p <- c(7/16, 7/16, 13/21, 13/21)
```

p*n

```
[1] 2.625000 4.375000 5.571429 7.428571
```

and the residual deviance, again calculated in two parts, is:

```
p1 <- 2*(3*log(3/2.625)+3*log(3/(6-2.625))+4*log(4/4.375)+6*log(6/(10-4.375)))
p2 <- 2*(5*log(5/5.571)+4*log(4/(9-5.571))+8*log(8/7.429)+4*log(4/(12-7.429)))
residual.deviance <- p1+p2 ; residual.deviance
```

```
[1] 0.4201974
```

We can now check our calculated deviance against that from R:

```
treatment <- factor( c(1, 1, 2, 2))
pq <- cbind(y, n-y)
treat.model <- glm(pq ~ treatment, family = binomial) ; summary(treat.model)
total.dev.model <- glm(pq ~ 1, family = binomial) ; summary(total.dev.model)

Call: glm(formula = pq ~ treatment, family = binomial)
```

Coefficients:

	Estimate	Std. Error	z value	Pr(> z)
(Intercept)	-0.2513	0.5040	-0.499	0.618
treatment2	0.7368	0.6752	1.091	0.275

Residual deviance: **0.4206** on 2 degrees of freedom

```
Call: glm(formula = pq ~ 1, family = binomial)
```

Coefficients:

	Estimate	Std. Error	z value	Pr(> z)
(Intercept)	0.1625	0.3299	0.493	0.622

Residual deviance: **1.6297** on 3 degrees of freedom

We will look at the output in detail below but it shows us that the residual deviance and the total deviance (also called residual in this case, but from a model where we have only fitted the grand mean, parameter 1) match our calculated values. The calculations get more complex for more complex examples but this simple example gives us an idea of what is going on inside R and where the outputted deviances come from.

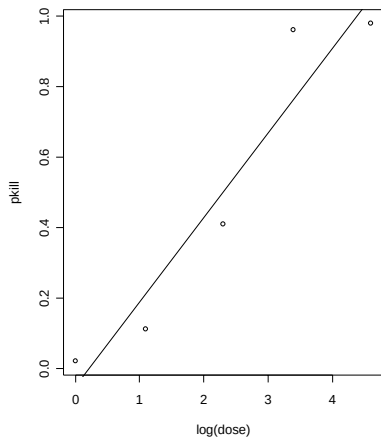
Simple logistic regression of proportion data in R

This example of simple analysis of proportion data in relation to a continuous variable comes from Crawley (2002). Different size batches of insects were exposed to different doses of insecticide and the number of dead insects recorded. The example is so small we can enter the numbers by hand:

```
dose <- c(1, 3, 10, 30, 100)
dead <- c(2, 10, 40, 96, 98)
batch <- c(100, 90, 98, 100, 100)
```

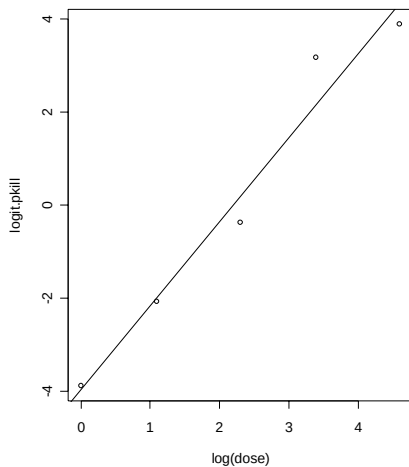
We can now calculate the proportion of each batch that were killed, 'pkill', and plot it as a function of dose on a logarithmic scale:

```
pkill <- dead/batch
plot(pkill ~ log(dose))
abline(lm(pkill ~ log(dose)))
```



The limitations of the linear regression are clear: at high doses it predicts greater than 100% mortality and at low doses negative proportions (spontaneous generation!). Taking the logit of the proportions (the log odds) solves this problem:

```
logit.pkill <- log(pkill / (1 - pkill))
plot(logit.pkill ~ log(dose))
abline(lm(logit.pkill ~ log(dose)))
```



Using the logits linearises the relationship but we still need to deal with the non-normal error structure. The proportions are estimated from different size samples (batches) and the regression can also take care of this by weighting by sample size. The GLM analyses the proportion killed with a logit link function (to get the log odds), a binomial distribution (to model the non-normal errors) and weights by the batch size (to give greater consideration to proportions estimated from larger samples). We can write the GLM out in full as follows:

```
model1 <- glm(pkill ~ log(dose), family = binomial (link = logit), weights = batch) ;
summary(model1)
```

```
Call:
glm(formula = pkill ~ log(dose), family =
binomial("logit"), weights = batch)
```

Deviance Residuals:

1	2	3	4	5
0.8125	0.8446	-1.7808	2.3780	-0.7926

Coefficients:

	Estimate	Std. Error	z value	Pr(> z)
(Intercept)	-4.532	0.438	-10.35	<2e-16 ***
log(dose)	1.964	0.175	11.22	<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: 408.353 on 4 degrees of freedom
Residual deviance: 10.828 on 3 degrees of freedom
AIC: 32.287
```

Number of Fisher Scoring iterations: 4

More usually we can write it in an alternative shorter formulation. Instead of specifying the weights argument R will do this automatically if we bind together two columns (using cbind) giving the 'successes' (dead) and 'failures' (alive) and use this object as the response variable in our model. We can also omit the optional 'family=' and the logit argument since that is the default with binomial errors:

```
pq <- cbind(dead, batch - dead)
model2 <- glm(pq ~ log(dose), binomial) ; summary(model2)
```

```
Call:
glm(formula = pq ~ log(dose), family = binomial)
```

Statistics for free: Generalised Linear Modelling using R

Deviance Residuals:

```
[1] 0.8125 0.8446 -1.7808 2.3780 -0.7926
```

Coefficients:

	Estimate	Std. Error	z value	Pr(> z)	
(Intercept)	-4.532	0.438	-10.35	<2e-16	***
log(dose)	1.964	0.175	11.22	<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: 408.353 on 4 degrees of freedom
Residual deviance: 10.828 on 3 degrees of freedom
AIC: 32.287

Number of Fisher Scoring iterations: 4

The output from the two alternative ways of writing the model is the same. It is similar to that produced when using the linear model (`lm`) function for normally distributed data and many of the features of analysis of deviance are analogous to those already encountered in analysis of variance. R gives a summary of the model, the residuals (whether the observation was larger or smaller than expected), the maximum likelihood estimates of the coefficients for the logistic regression and a summary of the explained and residual deviance (equivalent to the residual sums of squares) and degrees of freedom. Recall from the *Statistics for Free I* course that the Akaike Information Criteria combines the variance (or deviance) explained with the degrees of freedom used to provide a measure of model 'efficiency'. The parameter estimates are given approximate standard errors and probabilities from the z distribution. Some statistics packages produce a Wald statistics. Both can be seen as similar to the T-tests in the tables of estimates from analyses with normal errors. However, these tests of parameter estimates may be unreliable with small sample sizes. Therefore, it can be better to rely on tests done by model comparisons using the deviances and Chi-squared tests (in a similar way to preferring F tests to T-tests with analyses of normal variables). (As an aside, just as with simple analyses with normal errors F equals T squared, so the value of Chi-squared is approximately z^2 .) Remember that the P values are approximate because there is no exact solution. Instead, the maximum likelihood fitting procedure is iterative. An initial guess is used to get a first approximate solution, which is then used as a starting point for the next iteration. The iterations repeat until the solution becomes sufficiently stable with the number of iterations given at the bottom of the output. Sometimes the estimates will not converge on a stable solution and R stops the estimation after a certain number of iterations (usually 10, but this can be altered). Another feature of the GLM maximum likelihood estimation is that there is no so-called 'scale parameter' like the variance of normally distributed variables. Instead the method assumes the variance is a function of the mean. R reminds us that the, "Dispersion parameter for binomial family taken to be 1". One difference from analysing binary data

is that for proportions we have to check this assumption is not too badly infringed (see below).

The parameter estimates from the logistic regression are interpreted as describing the relationship between $\log(\text{dose})$ and deaths as:

$$\ln\left(\frac{p}{1-p}\right) = -4.531819 + 1.94314 \times \ln(\text{dose})$$

We use **anova** to get the equivalent of an ANOVA table (or more accurately an analysis of deviance table) using the test argument to specify a chi-squared test of this relationship:

```
anova(model2, test="Chi")
```

Analysis of Deviance Table

Model: binomial, link: logit

Response: pq

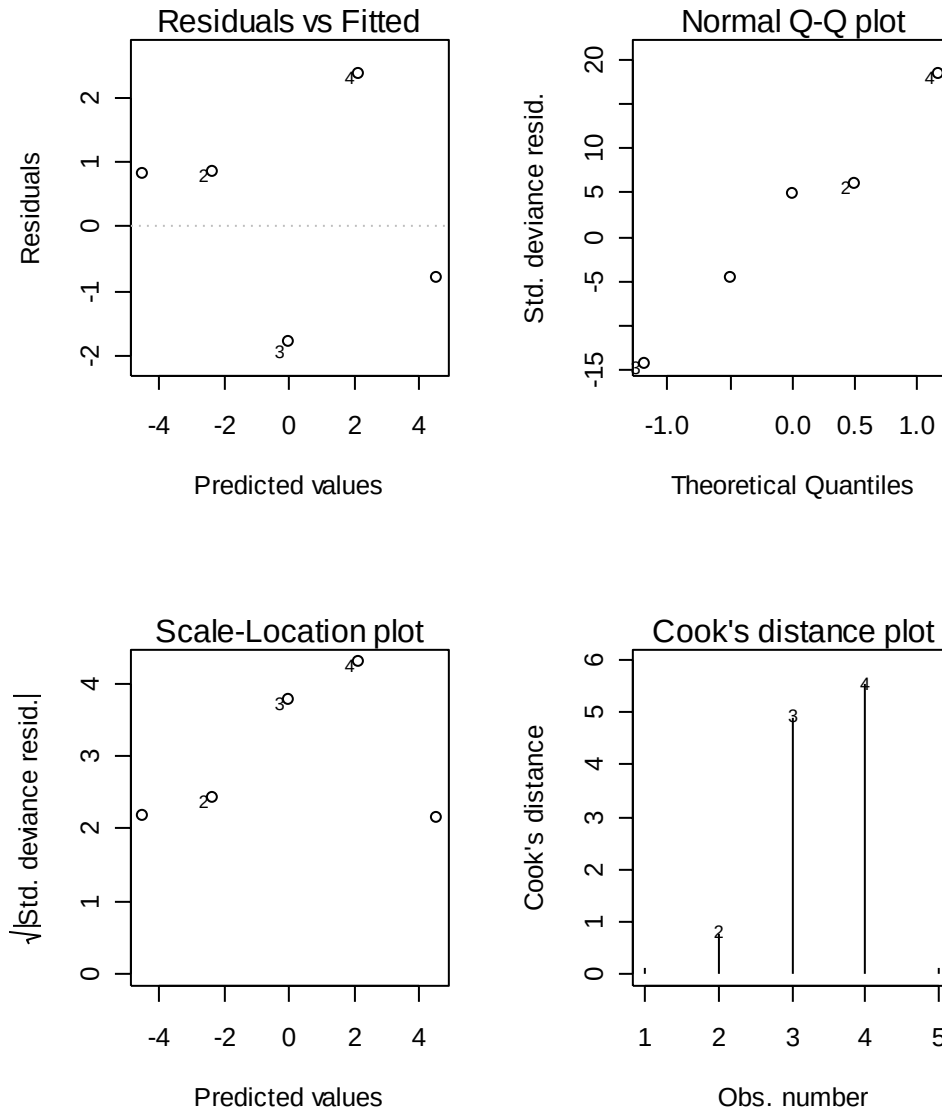
Terms added sequentially (first to last)

	Df	Deviance	Resid.Df	Resid.Dev	P(> Chi)
NULL			4	408.35	
log(dose)	1	397.52	3	10.83	1.904e-88

The analysis of deviance table first reports differences between models beginning with the null model. One difference to the analysis of variance table is that rather than have a column of variances and a column of F-value derived from them, there is just a single column for the deviance since it is assumed to approximately follow a Chi-squared distribution.

We can check the assumption that the "dispersion parameter for binomial family taken to be 1" by comparing the residual deviance with the residual degrees of freedom. If the binomial distribution is a good model then they should be roughly equal. Here the residual deviance is over three times the size of the residual degrees of freedom - a condition termed 'over-dispersion'. As usual, there is no magic number that gives an absolute boundary for over-dispersion but examples from text book consider values of 1.5 times and greater to be over-dispersed. We can also use the model checking plots:

```
par(mfrow=c(2,2)) ; plot(model2)
```



The residuals look reasonably normal (after controlling for their binomial nature) and the QQ plot (using the deviance residuals that are often recommended for GLMs) are reasonably straight. The Cook's distances are large and the plot of the standardised deviance residuals versus the predicted values suggests there may be some pattern remaining but it is hard to be sure with such a small (and fictitious!) example dataset. Instead, we will examine over-dispersion in greater detail in the following example from Collett (1991).

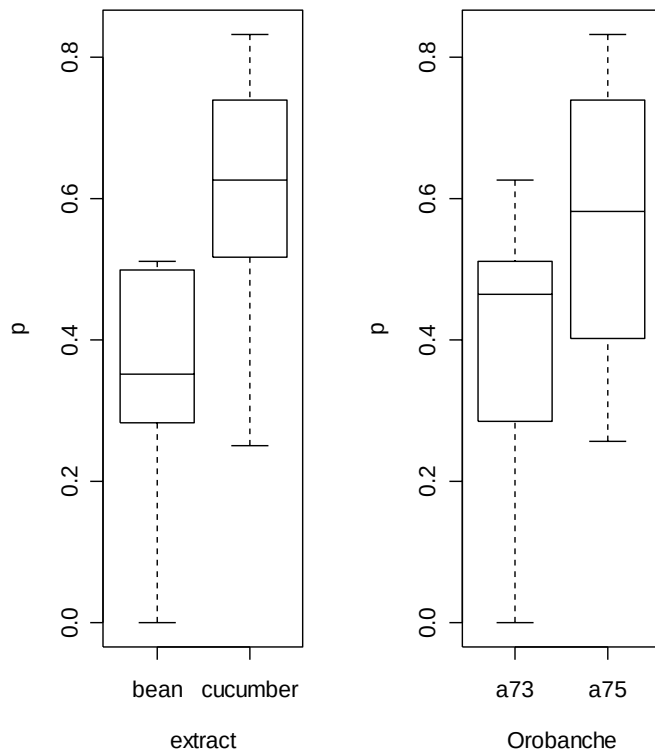
Overdispersion with proportion data

In this example from Collett (1991), extracts from bean and cucumber plants were tested for stimulatory effects on the germination of two genotypes of the parasitic plant *Orobanch*:

```
germination <-  
read.table("D:\\Andy\\_Teaching\\Statistics\\_R.GLMs\\binomial.germination.collet  
t.txt", header=T) ; attach(germination) ; summary(germination) ;
```

First calculate the proportion germinated and plot the means:

```
p <- count / sample  
par(mfrow=c(1,2)) ; plot(p ~ extract)  
plot(p ~ Orobanche)
```



There is an effect of extract - germination is higher with cucumber extract than with bean extract - and a smaller effect of *Orobanche* genotype. With these plots we cannot tell whether there is also an interaction between extract and genotype. For the analysis of the proportions we need to calculate the number germinating (count) and the number that did not germinate (sample - count):

```
pq <- cbind(count, sample - count)  
model1 <- glm(pq ~ Orobanche * extract, binomial) ; summary(model1)
```

```
Call: glm(formula = pq ~ Orobanche * extract, family =  
binomial)
```

Coefficients:

Statistics for free: Generalised Linear Modelling using R

	Estimate	Std. Error	z value	Pr(> z)	
(Intercept)	-0.4122	0.1842	-2.238	0.0252	*
Orobanchea75	-0.1459	0.2232	-0.654	0.5132	
extractcucumber	0.5401	0.2498	2.162	0.0306	*
Orobanchea75:extract	0.7781	0.3064	2.539	0.0111	*

(Dispersion parameter for binomial family taken to be 1)

Null deviance: 98.719 on 20 degrees of freedom
 Residual deviance: 33.278 on 17 degrees of freedom
 AIC: 117.87

As always, before getting too carried with the results we need to check the model. R reminds us of the assumption that the residual deviance is assumed to be equal to the residual degrees or freedoms. However, in this case the model is quite badly over-dispersed:

```
33.278 / 17
[1] 1.957529
```

The variance of a binomial distribution with mean np is:

$$S^2 = npq$$

When this is the case the assumption that the dispersion parameter equals one will hold. However, in this example the ratio is close to two instead of one. Either the model is mis-specified or the the probability of success, p , is not constant within a given treatment level. One solution is to assume the variance is instead:

$$S^2 = npqs$$

where s is an unknown scale parameter that is greater than one. The scale parameter is estimated as Pearson's Chi-square divided by the degrees of freedom. We can use the empirical scale parameter, rather than assume a dispersion parameter of one, by specifying a quasi-binomial distribution:

```
model2 <- glm(pq ~ Orobanche * extract, quasibinomial) ; summary(model2)
```

```
Call:
glm(formula=pq~Orobanche*extract, family=quasibinomial)
```

```
Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.01617  -1.24398   0.05995   0.84695   2.12123
```

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	-0.4122	0.2513	-1.640	0.1193
Orobancha75	-0.1459	0.3045	-0.479	0.6379
extractcucumber	0.5401	0.3409	1.584	0.1315
Orobancha75:extract	0.7781	0.4181	1.861	0.0801 .

(Dispersion parameter for quasibinomial family taken to be 1.861804)

Null deviance: 98.719 on 20 degrees of freedom
Residual deviance: 33.278 on 17 degrees of freedom
AIC: NA

Number of Fisher Scoring iterations: 3

The new test shows that once we have taken the over-dispersion into account the interaction term is no longer significant. We can produce an analysis of deviance table as follows. Now that we are using a quasi-binomial approach based on Pearson's Chi-square we use specify an F-test (be careful to specify the quasi-binomial family as well as the F-test; specifying an F-test alone will not produce the desired test):

```
anova(model2, test="F")
```

Analysis of Deviance Table

Model: quasibinomial, link: logit

Response: pq

Terms added sequentially (first to last)

	Df	Deviance	Res.Df	Res.Dev	F	Pr(>F)
NULL	20	98.719				
Orobancha	1	2.544	19	96.175	1.3665	0.25854
extract	1	56.489	18	39.686	30.3412	3.839e-05 ***
Oro:ext	1	6.408	17	33.278	3.4419	0.08099 .

We can check whether the dataset is non-orthogonal and the order-dependent by fitting a new model with the reverse sequence:

```
model3 <- glm(pq ~ extract * Orobancha, quasibinomial) ;  
anova(model3, test="F") ;
```

Analysis of Deviance Table

	Df	Deviance	Resid.Df	Resid. Dev	F	Pr(>F)
NULL	20	98.719				
extract	1	55.969	19	42.751 30.0615	4.043e-05	***
Orobanche	1	3.065	18	39.686 1.6462	0.21669	
ext:Oro	1	6.408	17	33.278 3.4419	0.08099	.

The data is non-orthogonal and dependent on the order of fitting but the qualitative outcome remains unchanged: the effect of extract is highly significant but genotype is not. We could present the analysis like this - retaining non-significant terms in the model - or we could use a model simplification approach and proceed to a minimal adequate model containing only the effect of extract:

```
mam <- glm(pq ~ extract, quasibinomial) ; summary(mam)
```

```
Call: glm(formula = pq ~ extract, family = quasibinomial)
```

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	-0.5122	0.1531	-3.345	0.0034 **
extractcucumber	1.0574	0.2118	4.992	8.1e-05 ***

To convert the logit estimates back to proportions requires us to reverse the logit transformation of $\ln(p / [1-p])$ as follows. To get the intercept and slope:

```
1 / (1 + 1 / (exp(-0.5122)))
[1] 0.3746779
```

```
1 / (1 + 1 / (exp(-0.5122 + 1.0574)))
[1] 0.6330212
```

The mean germination rate of the seed with the first extract was 37% and with the second extract (remember we have to add the *difference* to the first treatment level) 63%. An alternative way to get these numbers is to use the predict function - that generates fitted values from the model - with the "response" argument to backtransform onto the original scale:

```
tapply(predict(mam, type="response"), extract, mean)
```

```
bean cucumber
0.3746835 0.6330275
```

```
detach(germination)
```

References

- Collett, D. (1991) Modelling Binary Data Chapman & Hall, London.
- Cox, D.R. & Snell, E.J. (1989) Analysis of Binary Data Chapman & Hall, London.
- Fox, J. (2002) An R and S-Plus companion to applied regression Sage Publications, Thousand Oaks.
- Krause, A. & Olsen, M. (2002) The basics of S-Plus Springer-Verlag, Berlin.
- Venables, W.N. & Ripley, B.D. (2002) Modern applied statistics with S Springer-Verlag, Berlin.