Analysing binary data

Owen Petchey

2/17/2017

Introduction

Insect death

The wuestion is about the effects of three pesticides on the death of some insects. The pesticides were applied at a range of different doses to all of the individual insects in vial. This results in a dataset with the pesticide used(called "product"), the dose of that pesticide, the number of individuals at the start of the experiment, and the number that died. Hence its a dataset about who died and who lived, thus its a dataset about binary data.

This might seem like a quite complex study (i.e., it has two explanatory variables, one is continuous and the other is categorical), and you might think that starting with a simpler study (e.g., one continuous explanatory variable) might help you understand better. The reason we start with this somewhat complex study is that the question is almost identical in type to the ANCOVA in the previous chapter. Hence you following and understanding this insect death example will required you to understand and remember the ANCOVA example from the previous chapter.

Import that data:

```
insect.death <- read.csv("new_pesticide_study.csv"), row.names=1)
```

```
## Warning: Missing column names filled in: 'X1' [1]
```

```
## Parsed with column specification:
## cols(
##   X1 = col_integer(),
##   product = col_character(),
##   logdose = col_double(),
##   n = col_integer(),
##   dead = col_integer()
## )
```

```
insect_death
```

```
## # A tibble: 63 × 5
##       X1 product logdose     n  dead
##    <int>   <chr>   <dbl> <int> <int>
## 1      1       A    -5.0     9     0
## 2      2       A    -4.5    10     0
## 3      3       A    -4.0    12     0
## 4      4       A    -3.5    14     0
## 5      5       A    -3.0     9     1
## 6      6       A    -2.5    14     0
## 7      7       A    -2.0    14     3
## 8      8       A    -1.5    12     2
## 9      9       A    -1.0    12     5
## 10    10       A    -0.5     8     3
## # ... with 53 more rows
```

Various ways to express binary data

Each row of the dataset contains the results of a number of independent trials carried out at a particular dose (the `logdose` value) with one of the three products (the `product` value). In the first row (`product = "A"` and `logdose = -0.5`), for exmample, `n = 9` and `dead = 0`. There were 9 independent trials each involving an individual insect, and in none of those trials did an insect die. By contrast, the data in the 15th row shows 13 trials, and death of the insect in 12 of those.

An alternate expression of this data is to have one column be the number of successes (insect dies [this is test of an insecticide!]) and another column with number of failures.

```
insect_death <- mutate(insect_death, successes=dead,
                       failures=n-dead)
```

We can also express the same data as proportion of successes and number of trials.

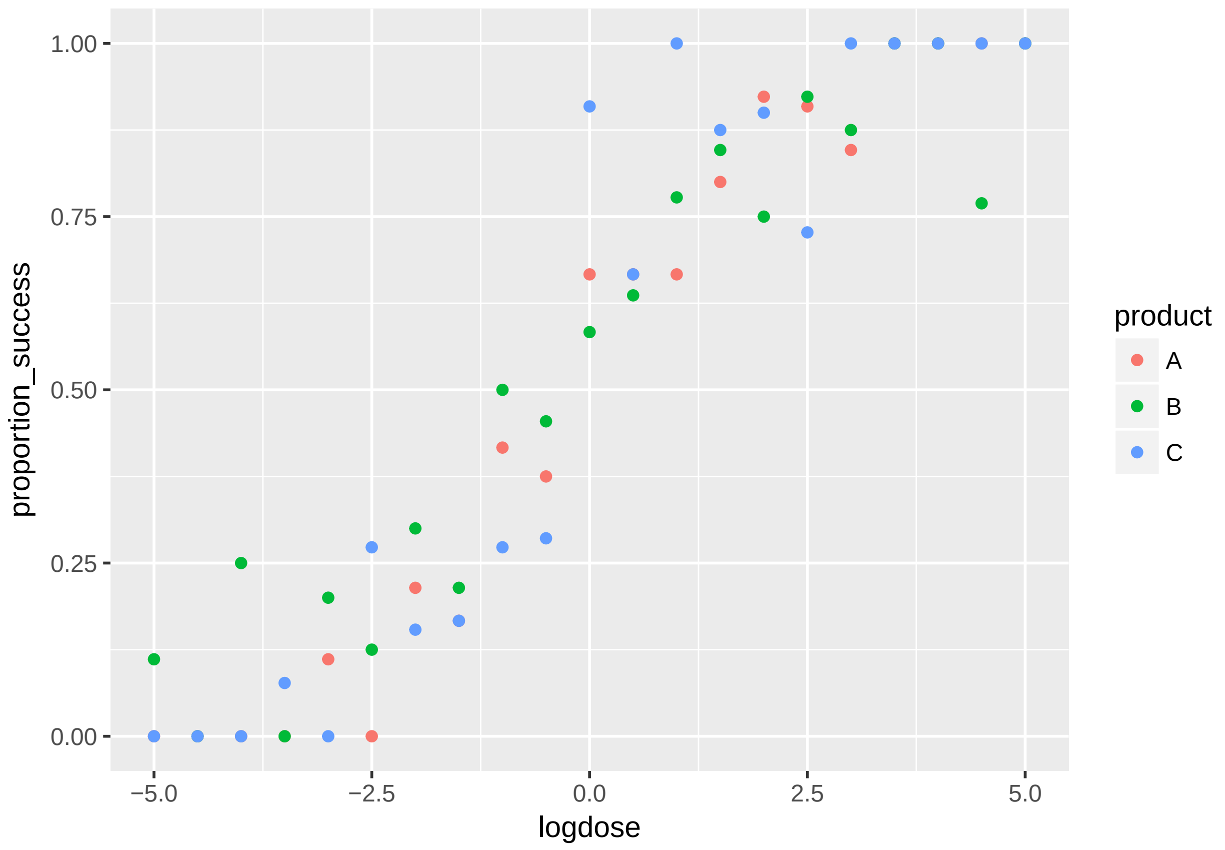
```
insect_death <- mutate(insect_death, proportion_success=successes/n)
```

One more way to express this data is to have one row for each individual insect, and a column giving the outcome of that trial, containing either a zero (for failure) or a one (for success). Its not so easy to convert our current data into that format, so we won't. Just be aware that if you have that type of binary data, its just fine like that.

Initial exploration

Now, of course, we make an exploratory graph. Lets make one with the proportion dead at each treatment combination on the y-axis, `logdose` on the x-axis and colour the points by the `product`.

```
ggplot(insect_death, aes(x=logdose, y=proportion_success, colour=product)) +
  geom_point() #+
```



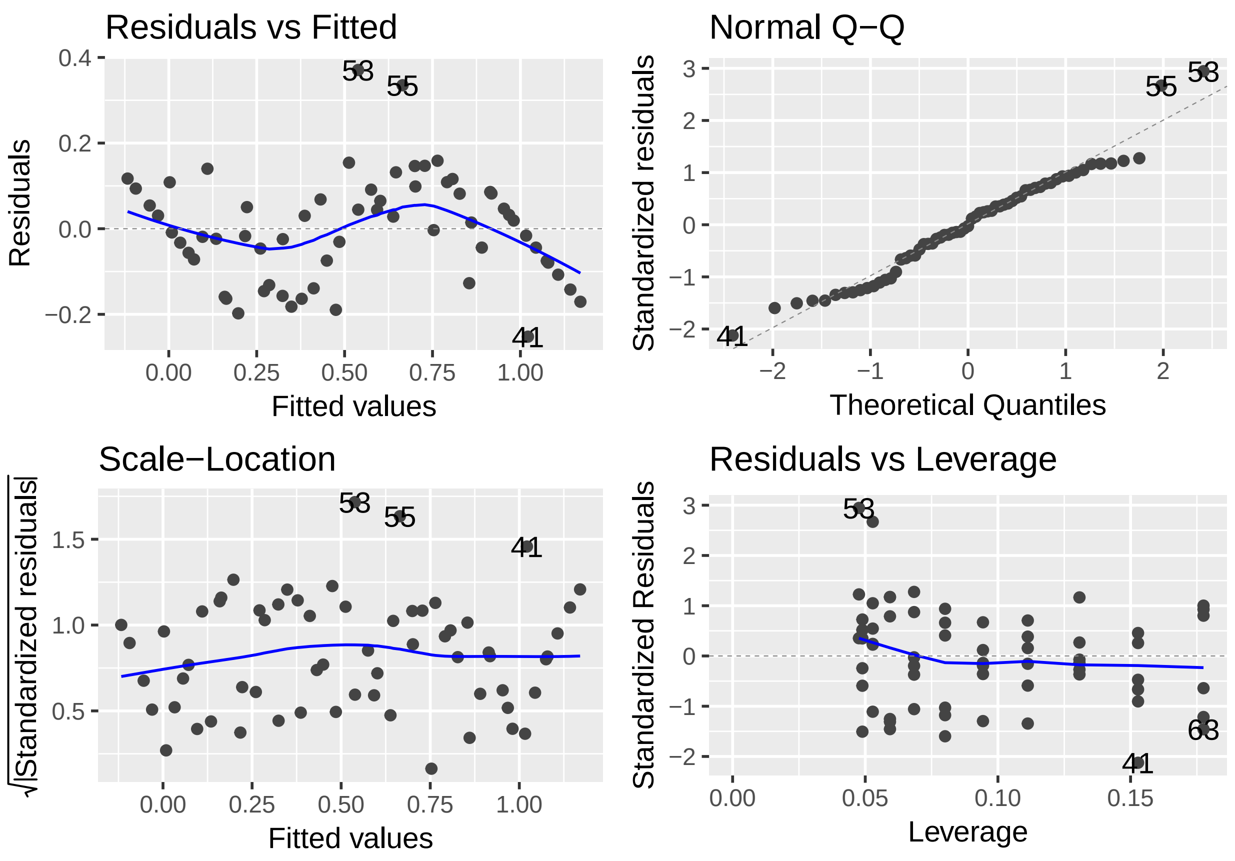
```r
#geom_smooth()
```

What can we conclude from this graph? Well, more pesticide equals greater death. Not so surprising. More interesting is the hint of relatively high mortality at low doses of product B, and relatively low mortality at high doses of product B. That is, it looks like the effect of dose is different for product B, compared to A and C. This means we expect a significant interaction between product and logdose in our statistical model.

Doing it wrong

Lets ignore that we know its a bad idea to make a linear model of binary data, just to see how bad things get. The most common method to make such a linear model is to use a proportion as the response variable (actually, this is not quite true, as probably few would make such a heinous mistake, and would instead transform the proportion, but lets continue anyway). We can use the proportion we already calculated, make the linear model, and then look at the model diagnostics.

```r
m1 <- lm(proportion_success ~ logdose * product, insect_death)
autoplot(m1)
```



Well, things look pretty bad in the residuals versus fitted values graph, showing that a linear model of the proportion data is not well justified. One solution is to apply a arcsine square root transformation to the proportions, but this is unneccessary in this case, because we have the binary data that underlies the proportions.

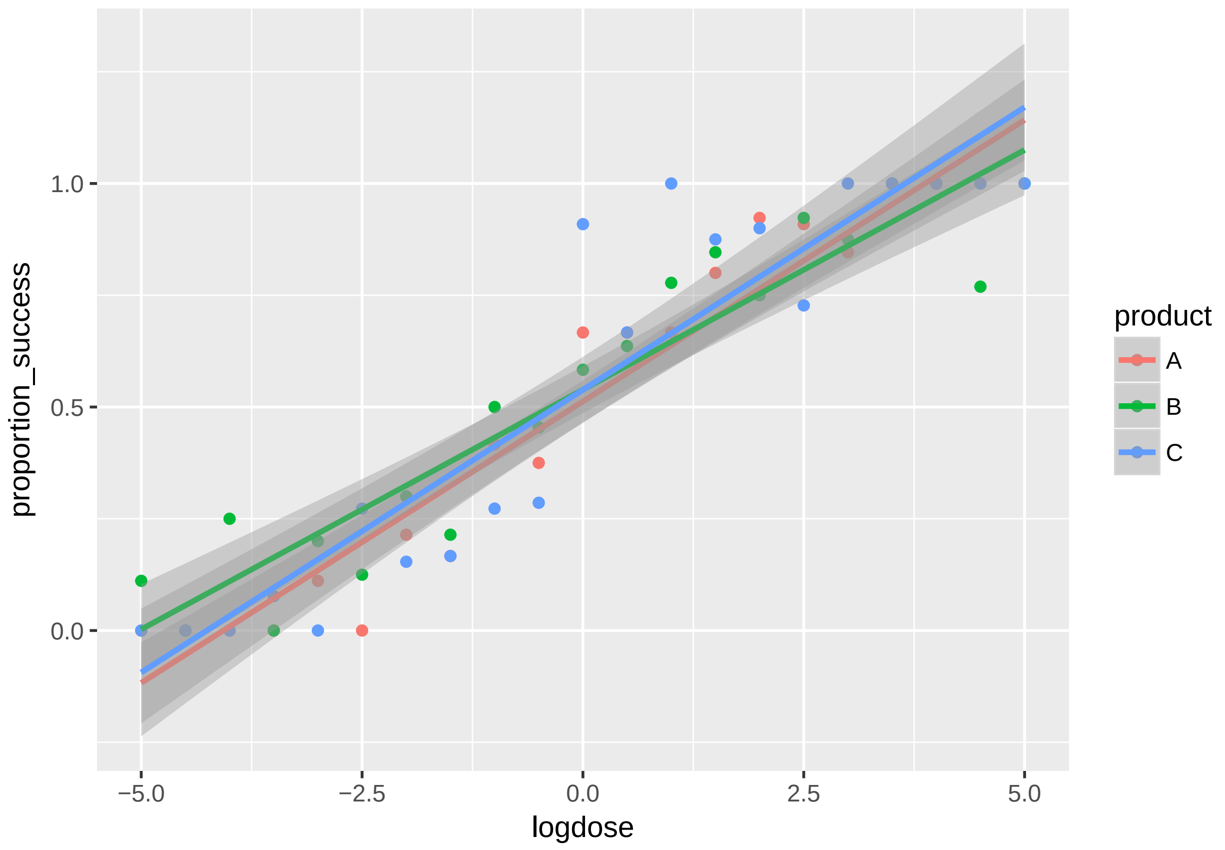
If we ignored the tcontinue and test the interaction term, we find it is not significant, in contrast to what we guessed:

```r
anova(m1)
```

```
## Analysis of Variance Table
##
## Response: proportion_success
##                 Df Sum Sq Mean Sq  F value Pr(>F)
## logdose          1 8.2977  8.2977 498.8793 <2e-16 ***
## product          2 0.0096  0.0048   0.2879 0.7509
## logdose:product  2 0.0461  0.0231   1.3861 0.2584
## Residuals       57 0.9481  0.0166
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

We can even make a version of the graph with the fitted lines.

```r
ggplot(insect_death, aes(x=logdose, y=proportion_success, colour=product)) +
  geom_point() +
  geom_smooth(method="lm")
```



This graph provides a good indication that a linear model is not a very good idea with binary data. At low and high doses, the predicted proportion of dead insects becomes less than 0 or greater than 1 (i.e., the lines go below 0 and above 1 on the y axis.). This is nonsense! Proportions can't be less than zero or greater than 1. Seeing the predictions of the statistical model straying outside the possible range of the response variable is a good indication that the model is inappropriate.

Doing it right

The binomial family

A much better model of the insect death data is a generalised linear model with a binomial distribution family and logit link function. The binomial distribution is just a distribution that describes the probability of something happening. Drawing numbers from a binomial distribution is just like flipping a coin:

```r
rbinom(10, 1, 0.5)
```

```
##  [1] 0 0 1 1 0 1 1 1 1 0
```

Each of these number is effectively the toss of a coin, with 0 meaning heads and 1 meaning tails (or the other way round).

The binomial distribution can have any probability, however. This would be like if a coin is less likely to come up heads (zeros) than tails (ones). In the following, we tell R that the coin comes up heads with probabiliy 0.1:

```r
set.seed(2)
```

```r
rbinom(10, 1, 0.1)
```

```
##  [1] 0 0 0 0 1 1 0 0 0 0
```

We can use another function in R to ask how likely it is that a coin comes up heads a specific number of times (call this H) when we toss it a certain number of times (call this N) (here we assume the coin is perfectly fair so p = 0.5).

Here we ask, if we toss a fair coin 100 times, what is the chance that we get exactly 50 heads and 50 tails? We answer this using R with:

```r
H <- 50
N = 100
p = 0.5
dbinom(H, N, p)
```

```
## [1] 0.07958924
```

So, if we toss a fair coin 100 times, the probability of getting exactly 50 heads is about 0.08. Though it is the most probable of all outcomes, it is neverthess quite unlikely that we get exactly 50 heads and 50 tails! Getting 49 heads and 51 tails, for example, is less probable:

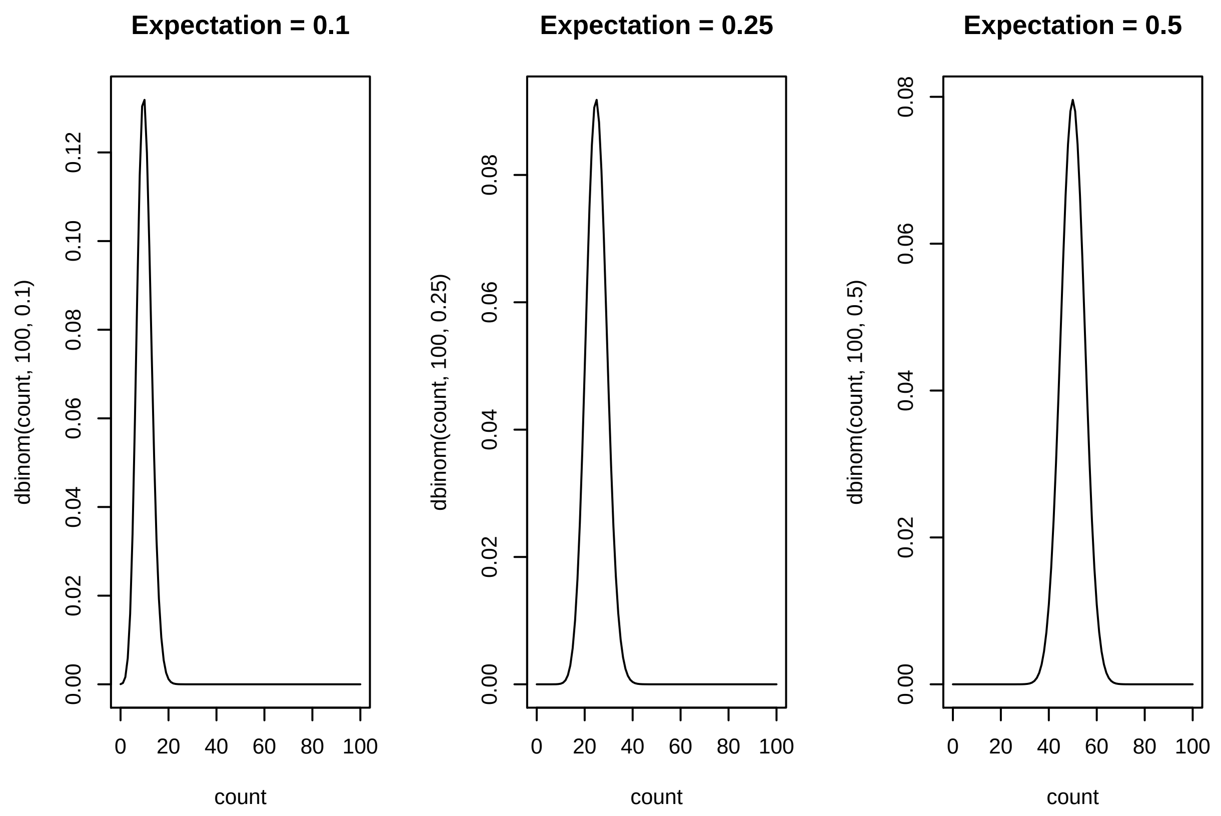
```r
dbinom(49, 100, 0.5)
```

```
## [1] 0.07802866
```

The figure below shows on the y-axis the probability of getting numbers of outcomes given on the x-axis.

Figure: Examples of the binomial distribution for 100 coin tosses, with three coins. The first coin is very biased (p = 0.1), the second less so (p = 0.25), and the third is fair (p = 0.5).

```r
count <- seq(0, 100, by=1)
layout(matrix(1:3, 1, 3))
plot(count, dbinom(count, 100, 0.1), main="Expectation = 0.1", type="l")
#arrows(count, rep(0, length(count)), count, dpois(count, 2), code=0)
plot(count, dbinom(count, 100, 0.25), main="Expectation = 0.25", type="l")
#arrows(count, rep(0, length(count)), count, dpois(count, 4), code=0)
plot(count, dbinom(count, 100, 0.5), main="Expectation = 0.5", type="l")
```

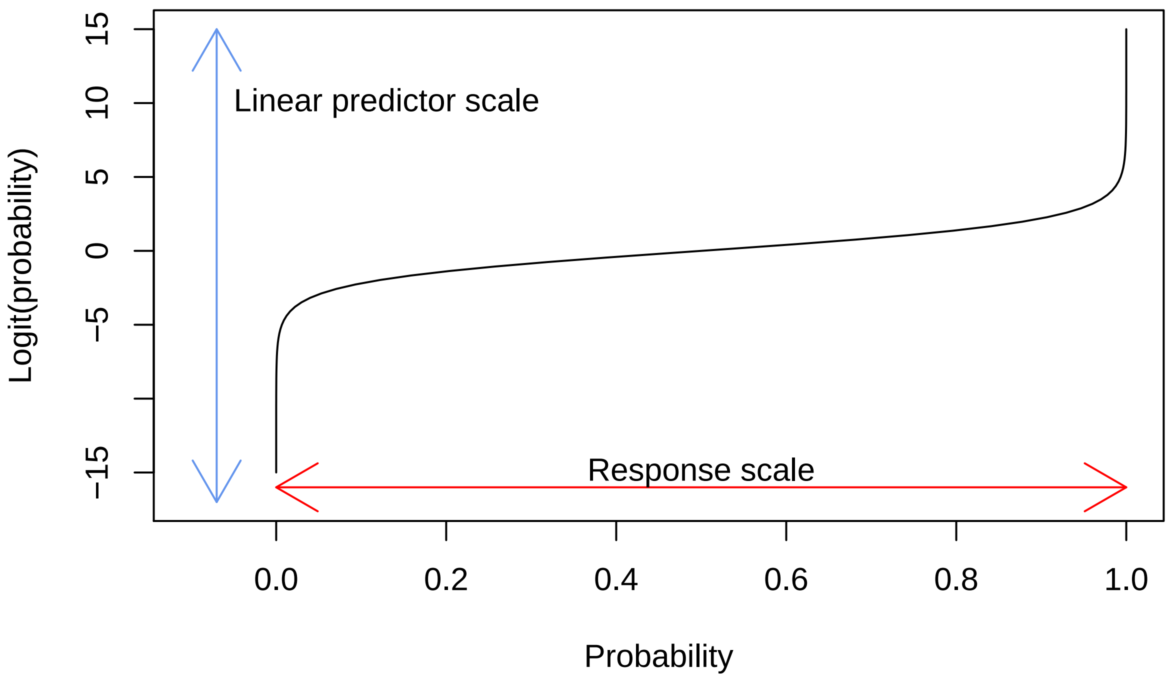


```
#arrows(count, rep(0, length(count)), count, dpois(count, 6), code=0)
```

The logit link function

When we ask for a binomial glm, we automatically use the logit-link function. But what is this. Simply put, its a mathematical function that changes a probability (say p) into a number that can take any value, from - to + infinity. The function is $logit(p) = log(\frac{p}{1-p})$ and you can see it in the figure below. In the insect mortality study, this link function changes the probability of death (which ranges from 0 to 1) to a linear predictor scale that goes from minus to plus infinity.

```
library(boot)
y=seq(-15,15,length=100)
plot(inv.logit(y), y, type="l", xlab="Probability", ylab="Logit(probability)",
     xlim=c(-0.1, 1), ylim=c(-17, 15))
arrows(0.00, -16, 1, -16, code=3, col="red")
text(0.5,-15, "Response scale")
arrows(-0.07, -17, -0.07, 15, code=3, col="cornflowerblue")
text(-0.05, 10, "Linear predictor scale", adj=0)
```



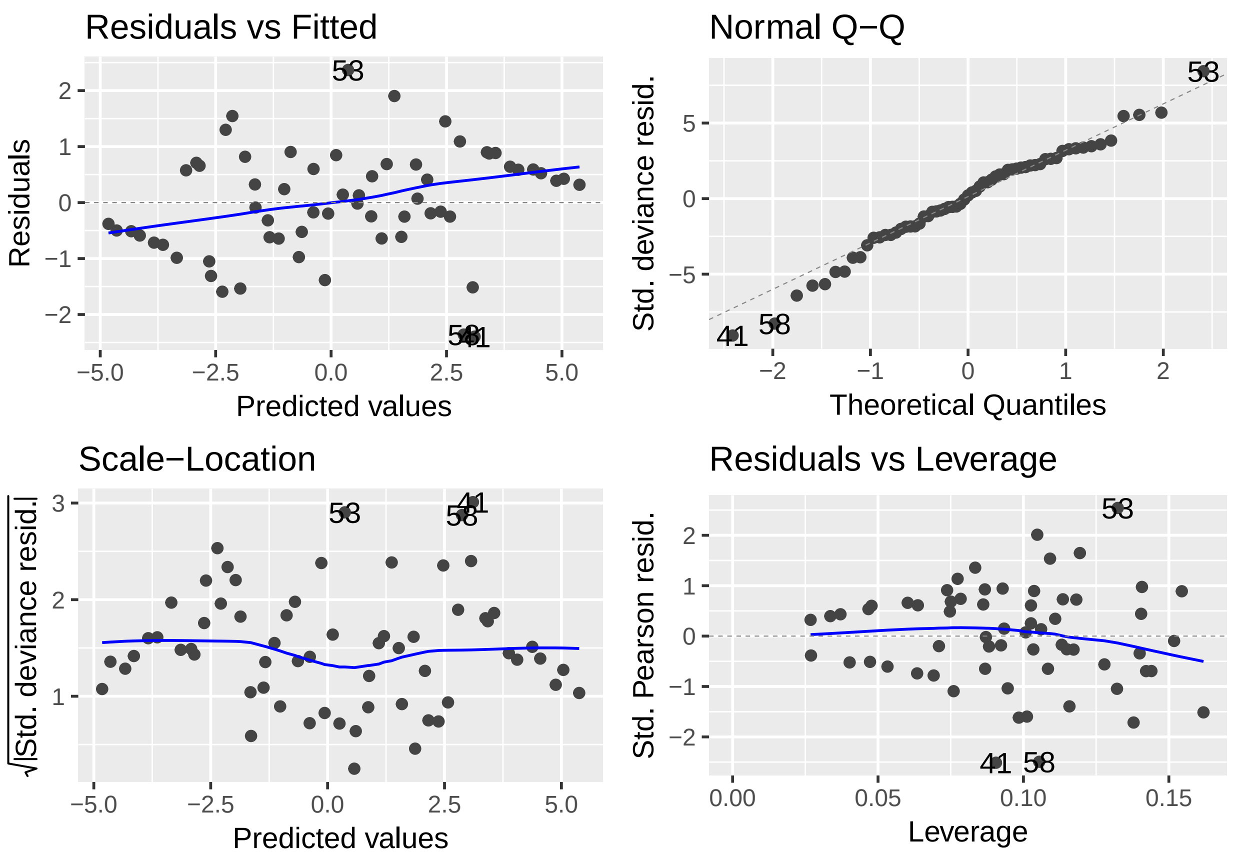
Making and interpreting the binomial

Then we make the general linear model with binomial distribution family:

```
m1 <-glm(cbind(successes, failures) ~ logdose * product,
         data = insect_death, family = "binomial")
```

Note that when we ask for the binomial distribution, glm automatically uses the logit link function.

```
autoplot(m1)
```



The diagnostic plots look fine. So we can continue with the interpretation of the model. First the analysis of deviance table:

```r
anova(m1, test="Chisq")
```

```
## Analysis of Deviance Table
##
## Model: binomial, link: logit
##
## Response: cbind(successes, failures)
##
## Terms added sequentially (first to last)
##
##
##                  Df Deviance Resid. Df Resid. Dev Pr(>Chi)
## NULL                                62     541.46
## logdose           1   475.91        61      65.55  < 2e-16 ***
## product           2     0.71        59      64.85  0.70131
## logdose:product   2     9.19        57      55.65  0.01009 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

We are reminded of the model (binomial with a logit link), of the response variable (`cbind(dead, alive)`), and that terms are added sequentially, which means that logdose is added to the null model, then product is added to the model containing logdose, and then the interaction between logdose and product is added to this.

Adding logdose to the model explains 475.91 units of deviance, out of the total of 541.46. This is a lot, and confirms only that higher doses of pesticides tend to kill more insects; this is not so surprising or interesting! (Adding `logdose` is like adding a single regression line to the graph.)

Adding the product term to the model explains 0.71 units of deviance, a very small amount. (Adding product to a model with logdose is like putting three regression lines with the same slope.)

Adding the interaction between logdose and product (which allow each of the three regression lines to have a different slope) explains 9.19 units of deviance, which is a reasonable amount. This suggests that allowing different slopes is appropriate / necessary. The biological interpretation of this is that the effect of logdose on mortality differs among pesticides. This is what we guessed when we first visualised the data.

Recall that these deviances are log likelihoods, and we can test the difference in deviance between two models by looking at the ratio of their likelihoods. First we make a model without the interaction:

```r
m2 <-glm(cbind(successes, failures) ~ logdose + product,
         data = insect_death, family = "binomial")
```

And then use the anova function to compare them:

```r
anova(m1, m2, test="Chisq")
```

```
## Analysis of Deviance Table
##
## Model 1: cbind(successes, failures) ~ logdose * product
## Model 2: cbind(successes, failures) ~ logdose + product
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1        57     55.652
## 2        59     64.845 -2  -9.1932  0.01009 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

(Technical detail: using the difference in log likelihoods is equivalent to using the log of the ratio of the likelihoods. log(A) - log(B) = log(A / B))

And now lets interpret the coefficients table:

```r
summary(m1)
```

```
##
## Call:
## glm(formula = cbind(successes, failures) ~ logdose * product,
##     family = "binomial", data = insect_death)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -2.39568  -0.60044   0.07022   0.64900   2.36789
##
## Coefficients:
##                  Estimate Std. Error z value Pr(>|z|)
## (Intercept)       0.10953    0.21497   0.510   0.6104
## logdose           0.98622    0.11835   8.333   <2e-16 ***
## productB          0.14427    0.27899   0.517   0.6051
## productC          0.25828    0.30986   0.834   0.4045
## logdose:productB -0.35196    0.14265  -2.467   0.0136 *
## logdose:productC  0.01627    0.17436   0.093   0.9257
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
```

```
##
##     Null deviance: 541.462  on 62  degrees of freedom
## Residual deviance:  55.652  on 57  degrees of freedom
## AIC: 162.38
##
## Number of Fisher Scoring iterations: 5
```

There is the usual reminder of the model and information about the residuals. Then comes the coefficients.

The first is the intercept, which is for product A (because A is before B and C in the alphabet – R works alphabetically with factors, unless we tell it otherwise). The estimate for this intercept is negative, which may seem odd as the response is about proportion dead so should be bounded between 0 and 1. But remember the logit link function is being used and the intercept coefficient is on the scale of the linear predictor, which is the logit of the proportion.

Next is the the logdose coefficient is the slope of the relationship between proportion dead and logdose for product A.

As in the ANCOVA example in the previous chapter (the limpet fecundity study) the productB coefficient and the productC coefficients are about the difference between the intercept of these products from the intercept for product A. And the logdose:productB and logdose:productC coefficients are the difference between the slope between product A and product B, and product A and product C, respectively. The table confirms what we suspected, that the relationship between mortality and logdose is different for product B, and the negative coefficient tells us the relationship is weaker (again we already guessed this).

Just as in the poisson example, we need to check if the data is overdispersed relative to the binomial distribution assumption, and do this in the same way, by dividing the residual deviance by the residual degrees of freedom:

```r
summary(m1)$deviance / summary(m1)$df.residual
```

```
## [1] 0.97635
```

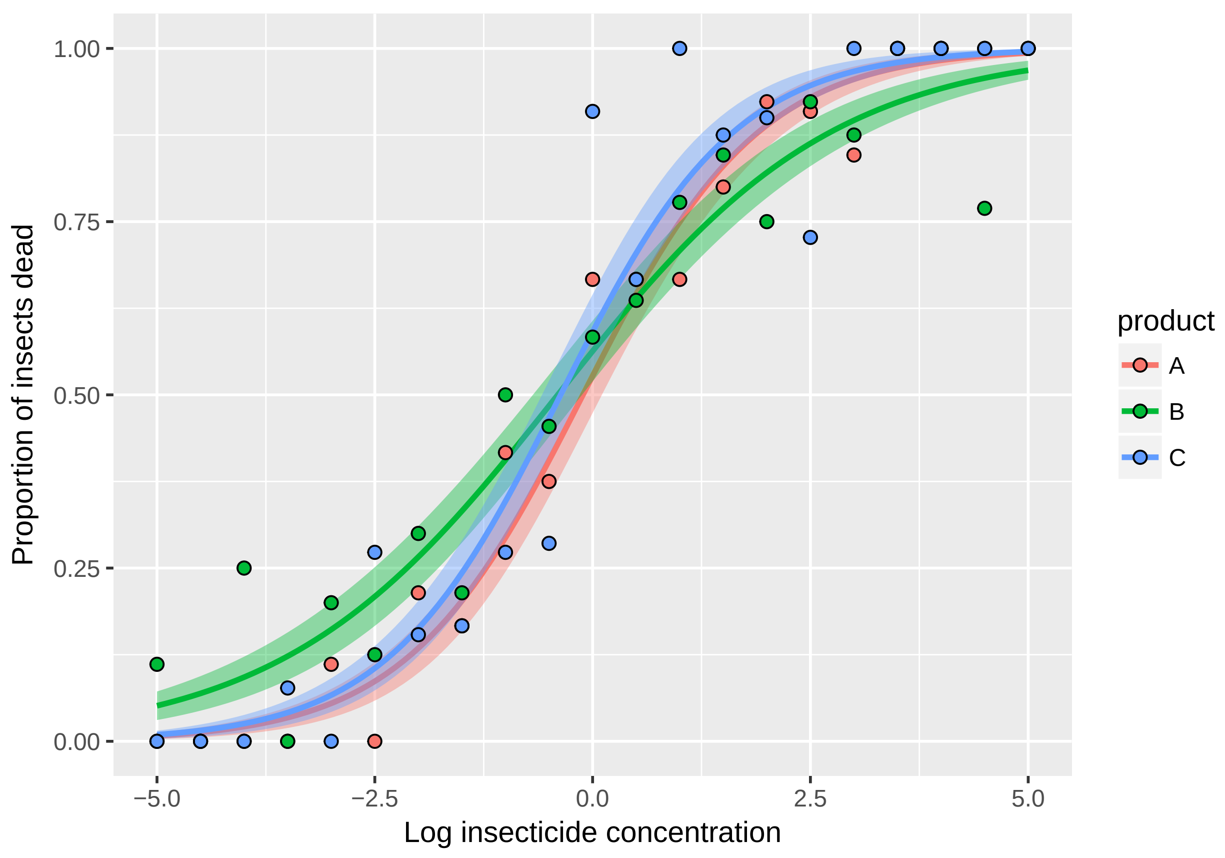
This is within the 0.9 to 1.1 limit, so we can be happy the data is not overdispersed.

Making a beautiful graph

Lets use our usual workflow to add the model fits to the graph, and also the summary statistics.

```r
new_data <- expand.grid(logdose=seq(min(insect_death$logdose),
                                    max(insect_death$logdose), length=100),
                    product=unique(insect_death$product))
p1 <- predict(m1, newdata = new_data, type="response", se.fit=TRUE)
n1 <- cbind(new_data, p1)
ggplot(insect_death, aes(x=logdose, y=proportion_success, fill=product)) +
  xlab("Log insecticide concentration") +
  ylab("Proportion of insects dead") +
  geom_smooth(data=n1, aes(y=fit, ymin=fit-se.fit, ymax=fit+se.fit, colour=product),
            stat="identity") +
  geom_point(pch=21, size=2)
```

```
## Warning: Ignoring unknown aesthetics: ymin, ymax
```



Proportion of insects dead
1.00
0.75
0.50
0.25
0.00
−5.0
−2.5
0.0
2.5
5.0
Log insecticide concentration
product
A
B
C