

Binary Data: Logistic Regression

Introduction

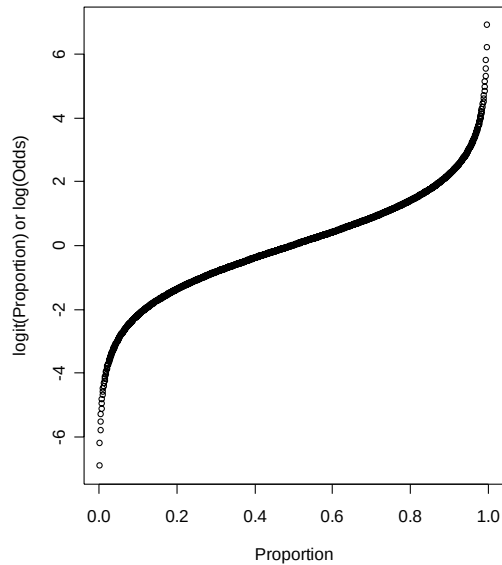
One type of data where the normal distribution does not apply and where GLMs prove very useful is with binary (0 or 1) responses such as presence versus absence of an organism, alive versus dead, male versus female and so on. Binary regression is appropriate when one or more of the explanatory variables is continuous with unique values for each sampling unit (i.e. every sampling unit, classified as 0 or 1, has a value for the explanatory variable). With categorical explanatory variables the data can instead be collapsed to a contingency table of counts and analysed with a Poisson error distribution (see *Count data and the Poisson distribution*).

The analysis is similar to that of proportion data but we have only a single column for the response variable consisting of zeros or ones so that the analysis is treated as a binomial trial with a sample size of one (see *Proportion Data*). This version of the binomial distribution is known as the Bernoulli distribution.

By definition, binary data can obviously only take two possible values - zero or one - and can therefore not be normally distributed. Similarly, the residuals will also not be normal distribution with a mean of zero and a constant variance. Instead, the so-called Pearson residuals have a mean of zero and variance of approximately one with the sum of the squares of these Pearson residuals giving the Pearson goodness of fit, χ^2 (see, for example, Crawley 2002, p.565 for details). Logistic regression deals with this non-normality by using the binomial distribution instead of the normal distribution.

One property of binary data that the general model has to deal with is the fact that the data is bounded at zero and one. This can be dealt with by using the logistic curve since this is a sigmoidal relationship that is asymptotic towards plus and minus infinity (see figure below). In logistic regression we work with odds which will probably be a familiar concept from betting. For example, when expressed in odds, a proportion of 0.666 (2 out of 3) becomes odds of 2-to-1 (2 successes to one failure). Rather than thinking of a probability of $p=0.66$ we need to think of odds of $p/q = 2/1$ (successes are usually termed p and failures q). We can see how the sigmoidal shape of the logistic regression using log-odds, or 'logits', can be used to deal with the bounded nature of binary data by looking at the relationship between proportions and their logits. As the proportion approaches zero or one the log-odds tend towards plus and minus infinity in an asymptotic way:

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



The advantage of using odds is that by taking the natural log we can make the sigmoidal logistic relationship linear for use in the regression as follows (for details see Crawley (2002) p.515):

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

We can make up a simple example to demonstrate how the sigmoidal logistic curve on the zero to one scale becomes linear on the logit scale. Imagine we have a binary response variable with more zeros at low levels of our explanatory variable, x , and more ones at high values. We can fit a binomial model (don't worry about the details of the model for now, they are explained below) and generate a logistic curve based on the fitted values from the model extracted with `predict` (as we did to draw curves in the *Statistics for Free I* course). The values are back-transformed (using `type = "response"`) onto the proportion scale:

```
x <- seq(1:30)
y <- c(0,0,0,0,0, 0,1,1,0,0,1, 0,0,1,0,1, 0,1,0,1, 0,1,1,1,1, 1,1,1,1,1)
par(mfrow = c(1, 2)) ; plot(y ~ x)
model <- glm(y ~ x, binomial) ; summary(model) ;
xseq <- seq(0, 30, 0.3) ; length(xseq)
lines(xseq, predict(model, list(x = xseq), type = "response"))
```

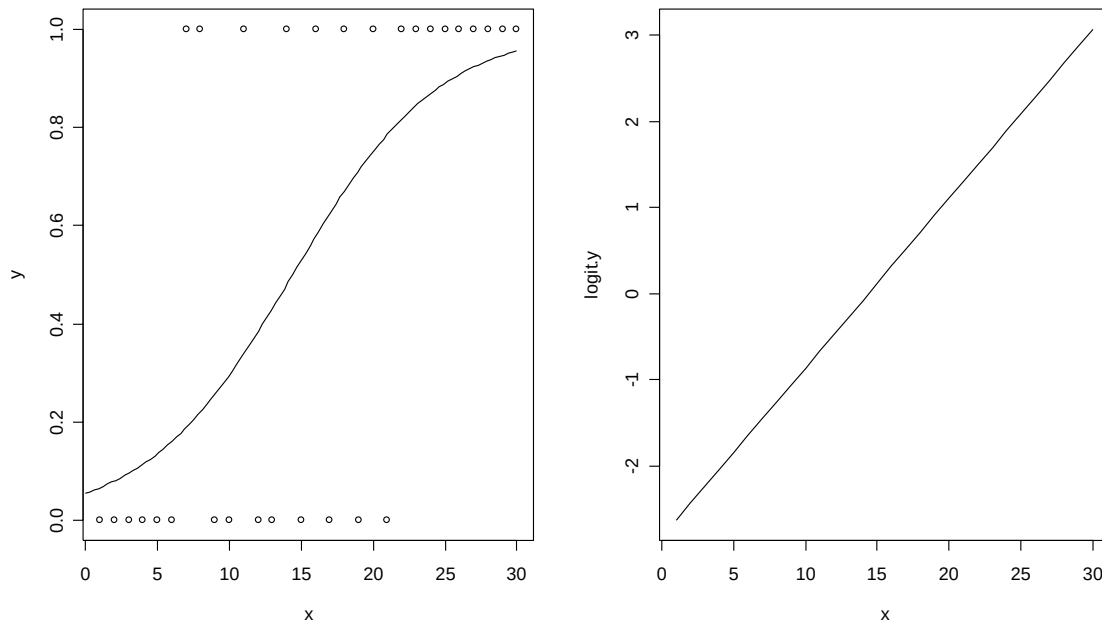
Call: `glm(formula = y ~ x, family = binomial)`

Coefficients:

	Estimate	Std. Error	z value	Pr(> z)	
(Intercept)	-2.83012	1.10957	-2.551	0.01075	*
x	0.19669	0.06826	2.881	0.00396	**

The figure is shown below next to one on the logit scale which is produced as follows. We extract the intercept, a , and slope, b , from the (edited) model summary and plug them into the regression equation given above to produce logit values for the sigmoid curve on the log-odds scale:

```
a <- -2.83012
b <- 0.19669
logit.y <- a + b * x
plot(logit.y ~ x, type="l")
```



Since we can get a linear relationship for use with binary (or 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 (see *Proportion Data*). One major advantage of logistic regression is that the generalized linear model with binomial error structure can deal with the lack of normality in the residuals.

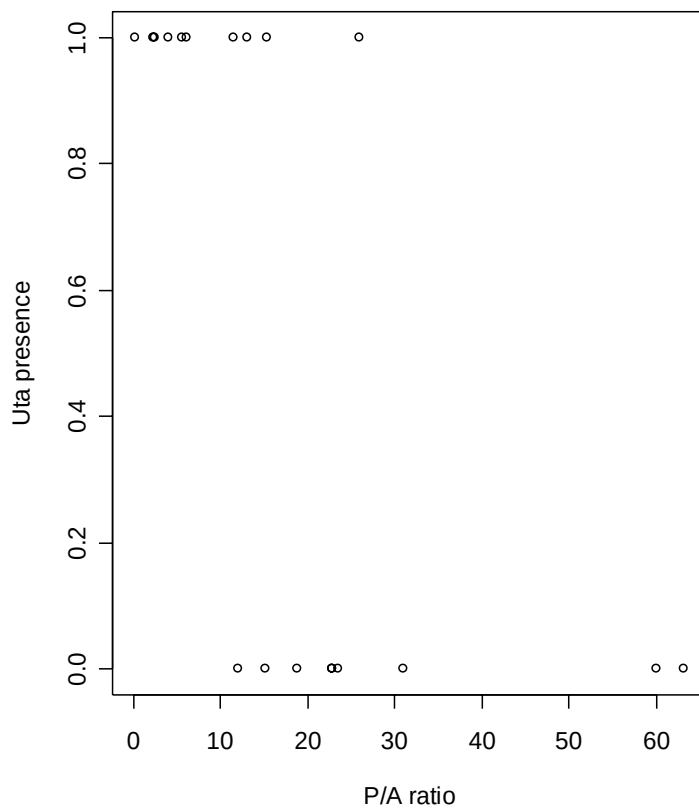
Simple logistic regression of binary data in R

This first example comes from Quinn and Keough (2002) from an original paper by Polis et al. (1998). Polis and colleagues studied spider populations on islands in the Gulf of California in relation to several factors including presence of predators including lizards. A subset of the data are used to examine the presence/absence of a species of lizard, *Uta*, on islands in relation to the perimeter-to-area ratio of the islands (used as a surrogate for the input of marine detritus). First, load the data and plot the presence or absence of *Uta* as a function of the perimeter-to-area ratio:

```
lizards.binary <-  
read.table("D:\\Andy\\_Teaching\\Statistics\\_R.GLMs\\logistic.regression.polis.txt",  
header=T) ; attach(lizards.binary) ; lizards.binary
```

	island	pa.ratio	uta
1	Bota	15.41	1
2	Cabeza	5.63	1
3	Cerraja	25.92	1
4	Coronadito	15.17	0
...			
18	Angeldlg	0.21	1
19	Mejia	2.55	1
20	Peninsula	NA	1

```
plot(uta ~ pa.ratio, ylab="Uta presence", xlab="P/A ratio")
```



Uta occurs on 10 of the 19 islands. The plot suggests it may occur more often on islands with a low ratio of perimeter to area - we can test this using logistic regression as follows:

```
uta.model <- glm(uta ~ pa.ratio, family = binomial (link = logit)) ;
summary(uta.model)
```

```
Call:
glm(formula = uta ~ pa.ratio, family = binomial(link =
logit))
```

Deviance Residuals:

Min	1Q	Median	3Q	Max
-1.6067	-0.6382	0.2368	0.4332	2.0986

Coefficients:

	Estimate	Std. Error	z value	Pr(> z)
(Intercept)	3.6060	1.6899	2.134	0.0329 *
pa.ratio	-0.2196	0.1002	-2.191	0.0284 *

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

(Dispersion parameter for binomial family taken to be 1)

Null deviance: 26.287 on 18 degrees of freedom
Residual deviance: 14.221 on 17 degrees of freedom
AIC: 18.221

Number of Fisher Scoring iterations: 5

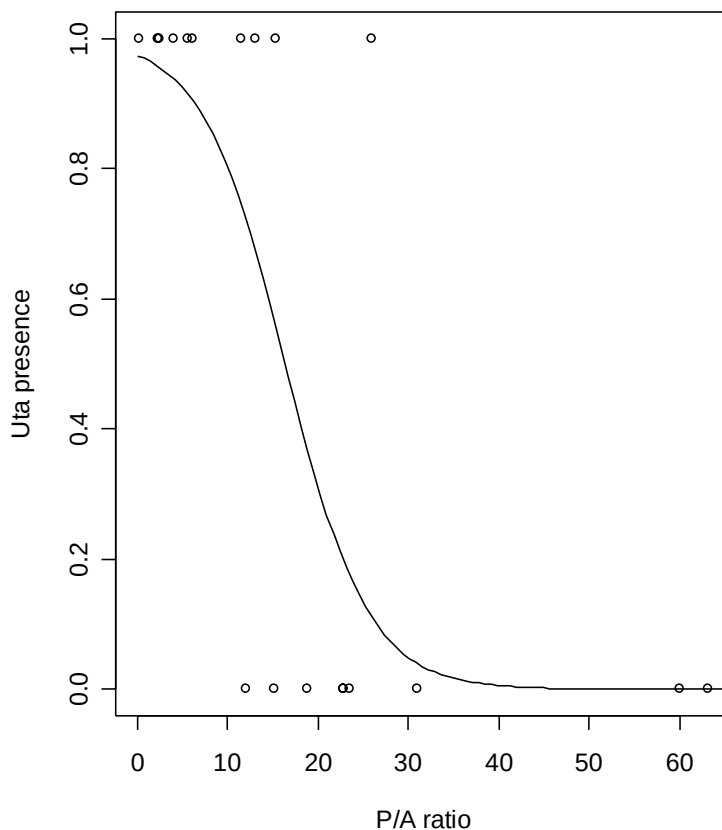
The glm specifies the binomial error structure and the logit transformation is the default option. The output is of a similar to that produced when using the linear model (lm) function for normally distributed data. R gives a summary of the model, the residuals, the maximum likelihood estimates of the coefficients for the logistic regression and a summary of the explained and residual deviance 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 . Returning to the analysis, taking the exponent (anti-log) of the log-odds estimate for the perimeter:area ratio shows its odds ratio to be 0.803:

```
exp(-0.2196)
```

```
[1] 0.8028399
```

That is, for an increase of one unit of the perimeter:area ratio an island has a 0.803 chance of having Uta compared to not having Uta. We can add the logistic regression to the plot with the code given below. First, we make a sequence (called 'xvalues') that divides the range of the X axis (pa.ratio ranging from about 0-70), into at least a hundred steps to give sufficient definition to draw a smooth curve. The **predict** function generates fitted values from the statistical model (uta.model). We can use xvalues within **list** to produce fitted value for each of hundred points in the created sequence. One complexity is that to produce 100 fitted values for 'xvalues' the sequence in a sense becomes our explanatory variable. To get the hundred values we have to specify, pa.ratio = xvalues, within **list**. The line is drawn, using **lines**, based on the fitted value from predict for each of the hundred points. The type = "response" argument in the **lines** function back-transforms the logits onto the scale of zero to one.

```
xvalues <- seq(0, 70, 0.7)
lines(xvalues, predict(uta.model, list(pa.ratio = xvalues), type = "response"))
```



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(uta.model, test="Chi")
```

Analysis of Deviance Table

Model: binomial, link: logit

Response: uta

Terms added sequentially (first to last)

	Df	Deviance	Resid.Df	Resid.Dev	P(> Chi)
NULL			18	26.2869	
pa.ratio	1	12.0662	17	14.2207	0.0005

The analysis of deviance table first reports differences between models beginning with the null model. The deviances are assumed to approximately follow a Chi-squared distribution so that the P-value is that of a Chi-squared value given by the deviance. In this case the test shows a significant effect of pa.ratio on the probability of the presence or absence of the lizard species.

With GLMs the residual deviance is usually assumed to be equal to the residual degrees of freedom. Sometimes the residual deviance is greater than the residual degrees of freedom - termed overdispersion - which we need to correct for. However, with binary data the values are all either zero or one and we therefore do not have to worry about overdispersion in the same way as when analysing proportion data with a logistic regression, counts with Poisson errors and so on. Some more advanced topics on diagnostics for binary logistic regressions are given in Fox (2002, p.227), Venables and Ripley (2002, p.208), Krause & Olsen (2002, p.263) and Neter, Kuter, Nachtsheim & Wasserman (1996, p.591). For more extensive coverage there are whole books dedicated to the analysis of binary data (Collett 1991; Cox, D.R. & Snell, E.J.).

```
detach(lizards.binary)
```

Using logistic regression to model zeros in continuous data

A frequent complexity in analysis of continuous variables is the presence of zero values. Zeros are difficult to deal with when using many transformations. More fundamentally, can the difference between, say, 10 and 20 be thought of in the same way as the difference between 10 and 0? One approach is to view zeros as a different type of response that should be analysed separately. In *Statistics for Free I* course we used ANCOVA to analyse the relationship between densities of nine specialist insect herbivores and the biomass of their host plants:

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

The summary output immediately makes one complexity clear: many samples have densities of zero insects. Earlier, we excluded the zeros and analysed the relationship between host plant biomass and insect densities. Now we can return to zeros and use this approach of viewing them as a different sort of information from the non-zero values. We can use a GLM with a binomial error structure to analyse the occurrence of specialists in relation to host plant biomass. First we need to make a factor of the insect species and create a binary variable for the presence (density > 0) of the different insect species:

```
INSECT<-factor(INSECT) ;  
ins.present<-insect.density>0 ; ins.present
```

Now we can fit a full model including the effects of plant mass, insect species and their interaction:

```
binary.insects <- glm(ins.present ~ INSECT + plant.biomass +  
INSECT:plant.biomass, binomial) ; anova(binary.insects, test="Chi") ;
```

There are significant effects of both plant mass and species identity and their interaction term is marginally significant. As the interaction is marginal we could choose a final model with or without the interaction term. For simplicity we will remove the interaction term here, although we would clearly want to discuss it in our write up since it was close to being significant:

```
no.interaction <- glm(ins.present ~ INSECT + plant.biomass, binomial) ;  
anova(no.interaction, test="Chi") ;
```

We can now use this minimal adequate model to produce a figure of the relationship between insect presence and plant biomass. One approach is to first plot the relationship for one species and plot the graph exactly the way we want it (remembering it will fill only 1/9 of a page once the other species are added). Then we can copy and paste the text and search and replace the species names. The only drawback with this approach is that the code can get very long. An alternative is to use a 'for loop'. In the data file we have the species names listed in a column called INSECT. However, the loop needs a column coding the insect species numerically that it will loop around for species 1 to 9. We are also going to need the numeric column in factor form [# for what?]. We can use **as.numeric** to create a new column 'species' and then convert it to a new factor 'SPECIES' in the usual way:

```
species <- as.numeric(INSECT)  
SPECIES <- factor(species)
```

However, we also want to be able to match the numbers back to the species names so we know which is which. We could print the long numeric and text columns but to make things easier we can use unique to summarise the species names and their corresponding numbers:

```
insect.names <- unique(INSECT) ; insect.names
```

```
[1] Apion.lotii      Craspedolepta.nervosa  Apion.dichroum
Rhinonchus.spp    Apion.curtirostre
Ceutorhynchidus.troglodytes  Gymnetron.pascuorum
Mecinus.pyraster  Errastunus.occelaris
```

```
9 Levels: Apion.curtirostre Apion.dichroum Apion.lotii
Ceutorhynchidus.troglodytes ... Rhinonchus.spp
```

```
insect.nos <- unique(species) ; insect.nos ;
```

```
[1] 3 5 2 9 1 4 7 8 6
```

So, the loop will begin with species number three - which from `insect.names` we can see is *Apion dichroum* - then species five, *Apion curtirostre*, and so on. One last stage of preparation: we are going to want to plot the fitted regression slopes on the graphs and to do that we need to create a sequence of at least a hundred values - `xseq` - to use in the `lines` command:

```
xseq <- seq(0,600,6) ; length(xseq) ;
```

```
[1] 101
```

As we have done before when plotting curves, the (101) Y values to go with `xseq` will be generated using `predict` which will generate fitted values for the 101 values of `xseq` from the model the plot is based on. However, we do need to generate a new factor to link each of the X and Y sequences to the appropriate species. We can use `concatenate` to generate a factor with nine levels, with each level composed of a species name 101 times:

```
sp.codes <- factor(c(rep("Apion.curtirostre",101), rep("Apion.dichroum",101),
rep("Apion.lotii",101), rep("Ceutorhynchidus.troglodytes",101),
rep("Craspedolepta.nervosa",101), rep("Errastunus.occelaris",101),
rep("Gymnetron.pascuorum",101), rep("Mecinus.pyraster",101),
rep("Rhinonchus.spp",101))) ; length(sp.codes) ;
```

Finally, we can run the for loop!

```
ylabel<-"Probability of occurrence" ;
xlabel<-expression(paste("Rumex acetosa (g m-2,")) ;
y.lim<-c(0,1.1) ;
par(mfrow=c(3,3), mar=c(4,4,2,1)+0.1, bty="l", pty="s", cex.main=0.9,
cex.axis=0.8, cex.lab=0.9) ;
for (i in 1:9)
  {main.title <- levels(insect.names)[i]
  plot(ins.present[species == i] ~ plant.biomass[species == i],
```

```

type="n", ylab=ylabel, xlab=xlabel, ylim=y.lim, main=main.title) ;
points(plant.biomass[species == i], ins.present[species == i]) ;
lines(xseq, predict(no.interaction, type="response",
data.frame(plant.biomass=c(xseq,xseq,xseq,xseq,xseq,xseq,xseq,xseq,xseq),
INSECT = sp.codes)) [sp.codes == levels(insect.names)[i]]) } ;

```

Each graph has the same Y axis limits and X and Y labels. These are defined before the loop to shorten the code inside. The parameters for all panels are set using **par** as we have done many times before.

The loop begins with 'for (i in 1:9)': the code within the loop will be repeated 9 times. All the subsequent commands are then held within curly brackets (or braces: {}) as a compound expression. The loop would also work with 'for 1:9' but adding 'i' as a counter enables us to know which repetition of the loop we are on and to do different things different times around the loop. For example, we have set the main title to be the level of insect.names equal to i. So, the first time around the loop the main title is the first level:

```
insect.names[1]
```

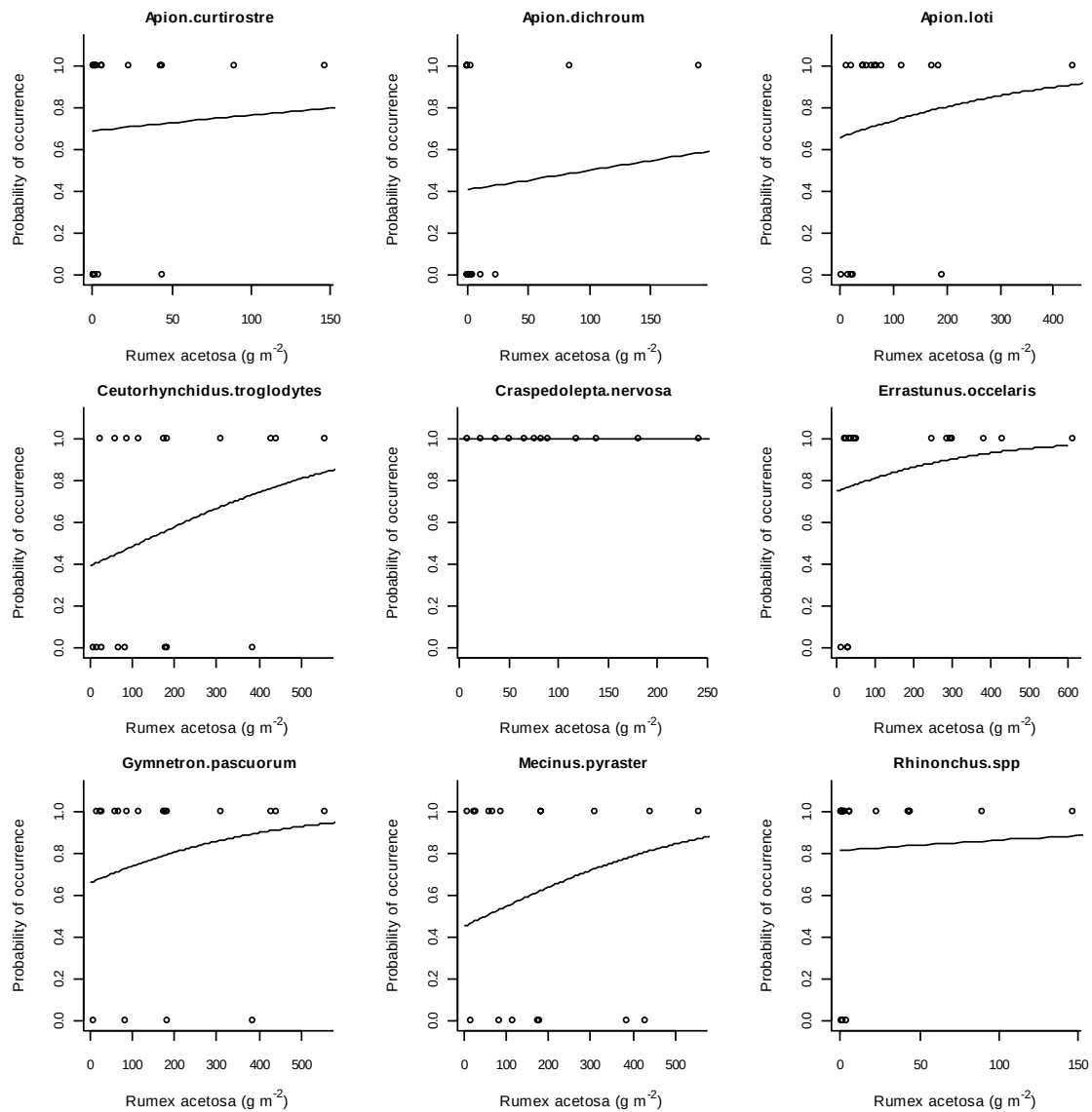
```
[1] Apion.lotii
```

The **plot** command works exactly as we have used it before. However, for **points** we can again use 'species = i' to subset the dataset to species 1 to 9 in turn. The most complex bit is the **lines** command. The line is plotted using the 101 values in xseq. The **predict** command produces a fitted value for each values of xseq generated from the no.interaction model. The type=response argument back-transforms the fitted values from the logit scale onto a zero to one scale. The trouble is we want to produce 9 lots of fitted values, one for each species. To get these we need to supply predict with the xseq nine times. Each repeat of xseq needs to be classified by one of the species. We can do this by putting the nine repeats of xseq into a dataframe with sp.codes - remember that sp.codes is already 909 rows long so we don't need to do anything extra to it. Finally, because predict generates fitted values based on a named statistical model we must use the names of the explanatory variables in the model. That is why we have to say, 'INSECT = sp.codes' and, 'plant.mass=c(xseq,xseq,xseq,xseq,xseq,xseq,xseq,xseq,xseq)'.

If this seems complicated it's because it is! We can generate a relatively complex figure with a short piece of code but the cost is that it is quite hard to understand. If this seems too complex to understand the same figure can be generated without a loop one panel at a time but the resulting code will of course get quite long.

```
detach(specialists)
```

Statistics for free: Generalised Linear Modelling using R



Exercises

Logistic regression: Survival of the passengers on the Titanic.

Source: Encyclopaedia Titanica (www.encyclopedia-titanica.org), Harrell (2001)

Data file: binary.titanic.harrell.txt

Exercise: Did the passengers on the Titanic follow the maritime rule of, "women and children first"? If they did, what was the relationship between survival and age for the men; at what ages was a male still considered a child rather than an adult man?

Solution: Harrell (2001) pages 299 - 326.

Logistic regression: presence/absence of rodents in habitat fragments

Source: Bolger et al. (1997)

Data file: binary.rodents.bolger.txt

Exercise: Box 13.2

Solution: Quinn & Keough pages 365 - 367.

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.
- Harrell, F.E. Jr. (2001) Regression modeling strategies: with applications to linear models, logistic regression, and survival analysis.
- Krause, A. & Olsen, M. (2002) The basics of S-Plus. Springer-Verlag, Berlin.
- Neter, Kuter, Nachtsheim, and Wasserman. 1996. Applied Linear Statistical Models. Irwin, Chicago.
- Venables, W.N. & Ripley, B.D. (2002) Modern applied statistics with S. Springer-Verlag, Berlin.