CHAPTER 31

GAMMA ERRORS

With continuous response variables, the traditional models of regression and Anova assume constant variance and normal errors. In many circumstances, however, this structure is inappropriate because:

- variance often increases with the mean
- the distribution of errors is often highly skewed

One solution is to assume log-normal errors in *y* and to carry out the analysis on the log-transformed response variable, using least squares with normal errors and the identity link. An alternative is to use a **glm** with gamma errors.

The gamma distribution

The 2-parameter gamma distribution is described on p. xxx. It is particularly appropriate as an error structure in cases where the response variable has a constant coefficient of variation rather than a constant variance. The mean of the gamma distribution is $\alpha\beta$ and the variance is $\alpha\beta^2$. This means that the *variance mean ratio* is β and the *coefficient of variation* is

$$\text{CV} = \frac{s}{\mu} = \frac{\sqrt{\alpha\beta^2}}{\alpha\beta} = \frac{\sqrt{\alpha}\beta}{\alpha\beta} = \frac{\sqrt{\alpha}}{\alpha} = \frac{1}{\sqrt{\alpha}} = \text{constant}$$

In a plot of log variance against log mean, the graph would be linear with a slope = 2 (see Taylor's Power Law on p. xxx).

Inverse polynomials

An important class of models for which gamma errors are appropriate are the inverse polynomials. The simplest of these is the inverse linear:

$$\frac{1}{y} = a + b\frac{1}{x}$$

which is an asymptotic curve familiar to scientists under a variety of names including Briggs/Haldane, Michaelis/Menten and Holling's disk equation, to name just 3 from physics, biochemistry and ecology respectively (see p. xxx). Taking reciprocals gives

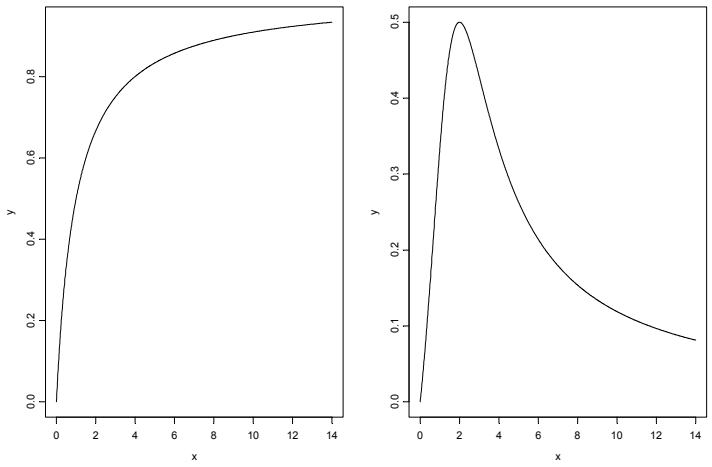
$$y = \frac{x}{b + ax}$$

The curve rises with an initial slope of $\frac{1}{b}$, and asymptotes at $\frac{1}{a}$. The general formulation of the inverse polynomial looks like this:

$$\frac{x}{y} = a + bx + cx^2 + dx^3 + \ldots + zx^n$$

and can have a wide range of shapes, including asymmetrical humped curves. Here are 2 contrasting inverse polynomials

```
par(mfrow=c(1,2))
x<-seq(0,14,.01)
y<-x/(1+x)
plot(x,y,type="l")
y<-1/(x-2+4/x)
plot(x,y,type="l")
```



It is clear that a reciprocal link function is appropriate for these models and, indeed, the reciprocal is the canonical link function for the gamma error distribution.

Deviance in glm's with gamma errors

With gamma errors, the lack of fit is measured by deviance rather than by SSE. The value of deviance is calculated for data *y* with mean μ like this:

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

Here are some data on time to death {3, 7, 11, 25}: they average 11.5 days, so their total deviance is:

```
2*(log(11.5/3)+(3-11.5)/11.5+log(11.5/7)+(7-11.5)/11.5+
                log(11.5/11)+(11-11.5)/11.5+(log(11.5/25)+(25-11.5)/11.5))

[1] 2.216189
```

We should check this by comparing it with the output from a **glm** with gamma errors

```
d<-c(3,7,11,25)
glm(d~1,family=Gamma)

Degrees of Freedom: 4 Total; 3 Residual
Residual Deviance: 2.216189
```

There you have it. Deviance is not mysterious; it is –2 times the log likelihood (see p. xxx).

Gamma errors with continuous response variables

This experiment involves the estimation of the parameters of a functional response curve which shows mean feeding rate of different individual predators as a function of prey availability:

$$y = \frac{x}{b + ax}$$

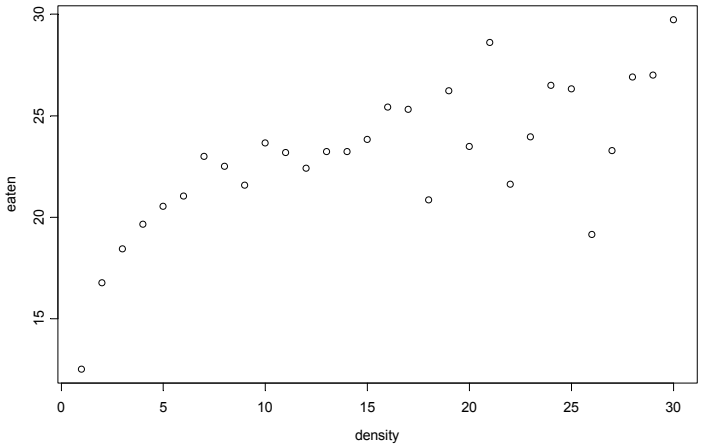
Taking reciprocals, we see that

$$\frac{1}{y} = \frac{b + ax}{x} = \frac{b}{x} + \frac{ax}{x} = a + b\frac{1}{x}$$

so the equation is linear with intercept a and slope b when $1/y$ is plotted against $1/x$.

```
attach(functionalresponse)
names(functionalresponse)
[1] "density" "eaten"
plot(density,eaten)
```

The scatterplot of feeding rates against food availability shows: (1) clear evidence of diminishing returns; (2) a pronounced increase in variance with increasing mean feeding rate. Both observations suggest that a glm with gamma errors might be an appropriate model structure.



The model is fitted like this:

```
model<-glm(eaten~I(1/density),family=Gamma)
```

Note the use of the I operator ("as is") in the model formula. It would be natural to write the transformation of the explanatory variable as 1/density, but the program would interpret the division as a slash, and hence think that we wanted to fit density nested within the intercept! Not what we intended at all. The estimated parameters look like this:

```
summary(model)

Coefficients:
                   Value   Std. Error  t value
 (Intercept) 0.03861653 0.0008645429 44.66699
I(1/density) 0.04404611 0.0056563131  7.78707

(Dispersion Parameter for Gamma family taken to be 0.0072898 )
```

The next step is to plot the fitted curve through the data. We could generate a vector of *x* values then use the coefficients to create a matching *y* vector, like this:

```
xv<-seq(0,30,.2)
yv<-xv/(0.04404611+0.03861653*xv)
lines(xv,yv)
```

With more complicated models, this transcription of coefficients is burdensome and runs the risk of introducing errors. The alternative is to use the **predict** function based on the current model fit. The trick here is to specify the vector of *x* values properly. In order to predict, the model needs to have values for all of the terms in the model (factors and continuous variables) and *they must have exactly the same names* in the new data frame as in the original data frame from which the model fit was carried out. This means that there are two vectors with the same name but with (often) different length attributes. To

avoid this problem, the name of the vector containing the *x* values for prediction needs to be hidden inside a list or a data frame.

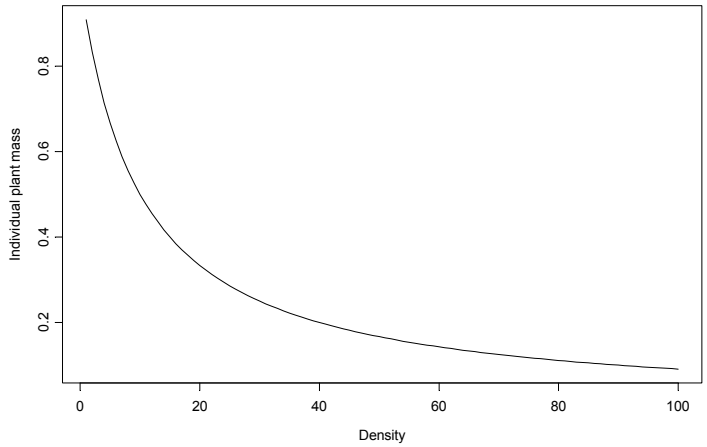
```
lines(xv,1/predict(model,list(density=xv)))
```

This is a much more elegant solution to the problem of plotting the fitted model. Note how the density vector for prediction is set to equal our generated series *xv* inside a list. Note also that it is assigned by = (equals) not <- (gets) or == (double equals). Because the model used a reciprocal link (the default for a glm with gamma errors), we need to transform the predicted values to obtain the feeding rates for plotting. The parameters used to generate the predicted curve are taken automatically from *model*, the values in *xv* are attributed to *density*, and the reciprocals of these predicted values are drawn against the generated series of *x* values using **lines**.

Plant competition models

A classic example of the use of gamma errors in ecological work is in the construction of response surface models of plant competition. Both elements of the canonical assumptions of a **glm** with gamma errors apply here: the variance increases more quickly than linearly with the mean, and the reciprocal in the natural link function, as mean plant size decreases hyperbolically as plant density increases.

If total plant biomass is roughly constant regardless of planting density, then individual



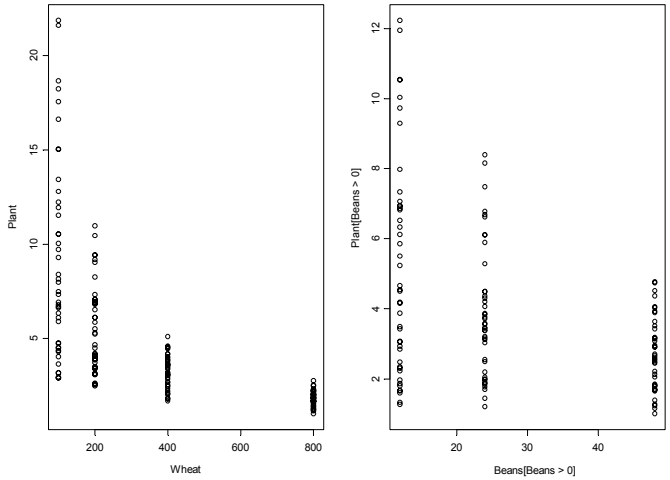
plant size should be inversely proportional to plant density. This suggests that a reciprocal link would be appropriate to the analysis of plant size / plant density relationships. The current example involves the growth of wheat plants in competition with different densities of bean plants and different densities of wheat.

```
attach(Density)
names(Density)
[1] "Plant" "Wheat" "Beans"
par(mfrow=c(1,2))
```

```
plot(Wheat,Plant)
plot(Beans[Beans>0],Plant[Beans>0])
```

The relationship between the mean size of wheat plants (y) and the density of both wheat (x) and bean plants (z) is hypothesised to be described by a model of this general form

$$\frac{1}{y} = a + bx + cx^2 + dz + ez^2$$

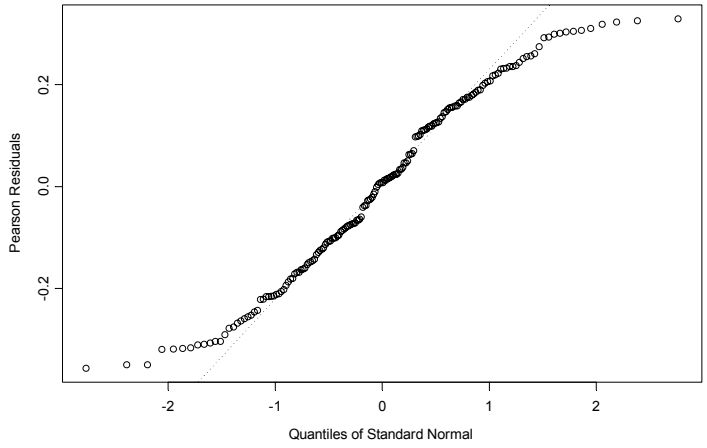


Both relationships look reasonably hyperbolic and the variance clearly increases as mean plant size increases.

```
model<-glm(Plant~Wheat*Beans,family=Gamma)

summary(model)
```

```
Coefficients:
                      Value    Std. Error     t value
(Intercept) -1.149190e-003 4.508787e-003  -0.2548778
      Wheat  6.022140e-004 2.467861e-005  24.4022691
      Beans  4.317402e-003 2.902596e-004  14.8742790
Wheat:Beans -3.712111e-007 1.178967e-006  -0.3148613

(Dispersion Parameter for Gamma family taken to be 0.0351985 )

Null Deviance: 87.9176 on 175 degrees of freedom
Residual Deviance: 6.232948 on 172 degrees of freedom
```

There is no suggestion of any interaction effect (t = 0.315)



The deviance residuals are OK, showing no tendency to increase with the fitted values. The errors, on the other hand, are distinctly non-normal (the qq plot is S-shaped rather than linear). What about curvature in the response surface ? We might try fitting quadratic terms for Wheat and Bean densities to cure the non-normality of errors.

```
model2<-glm(Plant~poly(Wheat,2)*poly(Beans,2),family=Gamma)

anova(model2,model,test="F")[,-1]
```

```
Resid. Df Resid. Dev    Test Df    Deviance   F Value     Pr(F)
1       167   6.208940
2       172   6.232948 1 vs. 2 -5 -0.02400815 0.1332373 0.9845322
```

There is no hint of any need for quadratic terms, so we shall remove the interaction term and stick with the simplest model:

```
model<-glm(Wt~Wheat+Beans,family=Gamma)

summary(model)
```

Coefficients:

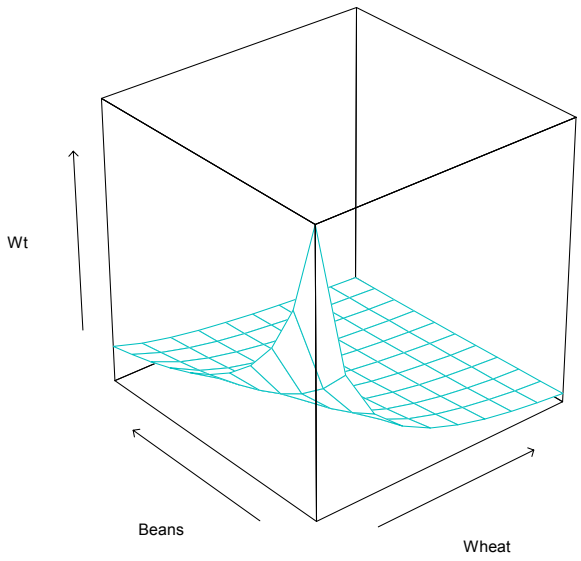
```
                    Value    Std. Error     t value
(Intercept) -0.0003781824 0.00378133613  -0.1000129
      Wheat  0.0005971468 0.00001857947  32.1401425
      Beans  0.0042477867 0.00018719763  22.6914559
```

We finish by inspecting the shape of the response surface:

```
xyz<-expand.grid(Wheat=seq(100,1000,100),Beans=seq(0,60,6))
xyz$Wt<-as.vector(1/predict(model,xyz))
```

Note that we need to have to back-transform the prediction from the reciprocal link on which the gamma errors work. Note also that the prediction grid starts with a minimum wheat density of 100 plants per square metre, because there are no values of the response variable (mean wheat weight) when there are zero plants per unit area.

```
wireframe(Wt~Wheat*Beans,xyz)
```



Analysis of covariance with Gamma errors

This example involves an analysis of covariance, where multiple predicted lines are to be plotted on the same graph. The yield of a chemical reaction is related to the concentration of reactant and is hypothesised as varying from batch to batch of reactant:

```
attach(gammaplot)
names(gammaplot)
[1]  "conc" "yield" "Batch"
```

The variance increases rapidly with the mean, so a sensible choice of model might be a glm with gamma errors and a reciprocal link. We start by fitting a maximal model with different regression lines for each batch:

```
model<-glm(yield~Batch*conc,family=Gamma)
```

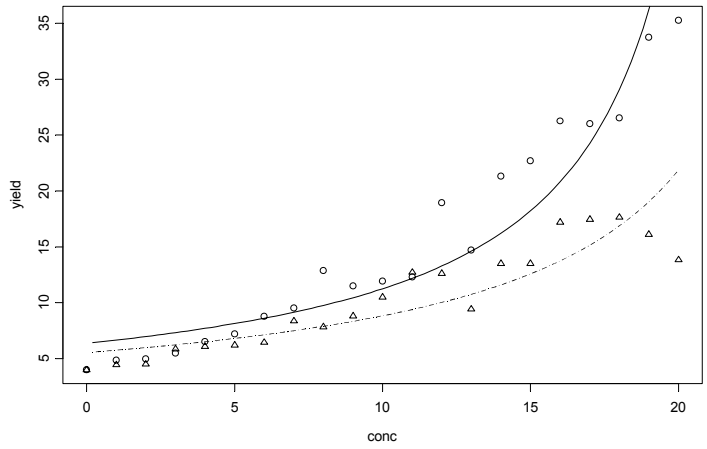
Prediction from an Ancova requires a data frame containing both *x* values and factor levels (see p. xxx):

```
newx<-
data.frame(conc=rep(seq(0.2,20,.2),2),Batch=c(rep("new",100),rep("old",100)))
```

We can now use the fitted model and the new data frame of values for predicting yield. Note that we take the reciprocal of the predictions because the **glm** is fitted with a reciprocal link:

```
newx$yield<-1/predict(model,newx)
```

Finally, we can plot the data and the fitted model on the same axes:




```
plot(conc,yield,type="n")
points(conc[Batch=="old"],yield[Batch=="old"],pch=2)
points(conc[Batch=="new"],yield[Batch=="new"],pch=1)
lines(newx$conc[newx$Batch=="new"],newx$yield[newx$Batch=="new"])
lines(newx$conc[newx$Batch=="old"],newx$yield[newx$Batch=="old"],lty=3)
```