Generalized linear models

Etienne Low-Décarie

2017-12-18

Generalised Linear models in R

In this exercise (which has plenty of material to cover 2 sessions of practical work!), we will cover how to do logistic regression models in R (i.e. GLMs with binomial errors and a logit link), Poisson, over-dispersed Poisson and negative binomial models. There is a lot here, and it pays to go slowly through the examples. Don't worry if you don't finish in class, complete in your own time.

1. Load the `soay2.csv` data frame into R, call it `ss` for soay sheep.
2. Look at it, use `str` and `names`.
3. Make `year` a factor in the worksheet (this saves doing it within every model). Type: `ss$YEARf<-as.factor(ss$YEARf)`.
4. Survivorship is shown in this file by the variable `SURV1`. Animals that died are indicated with a zero (0) and those which lived by a one (1).

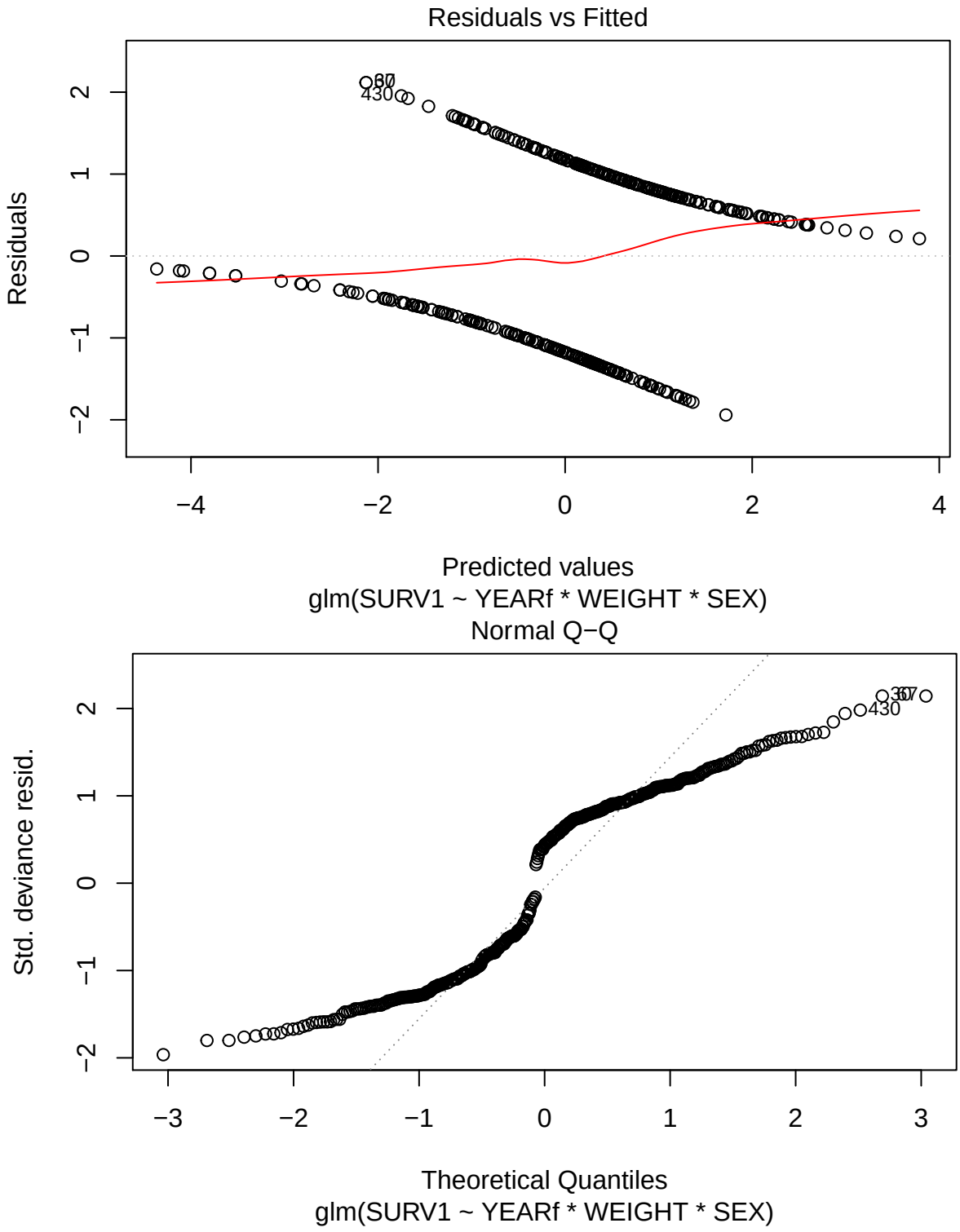
In the past, you would probably have analysed these data by calculating mean survivorships (% per year, for example) and then arcsine transforming the data to make them approximately normal before doing a linear regression with average survival as your dependent variable. With generalized linear modelling we do not need to do this! To analyse survival data (in which we have binary data of this sort) we perform a generalised linear model in which our error distribution is binomial and our link function is logit. This is conventionally known as logistic regression. The question is "what factors influence Soay sheep over-winter mortality?

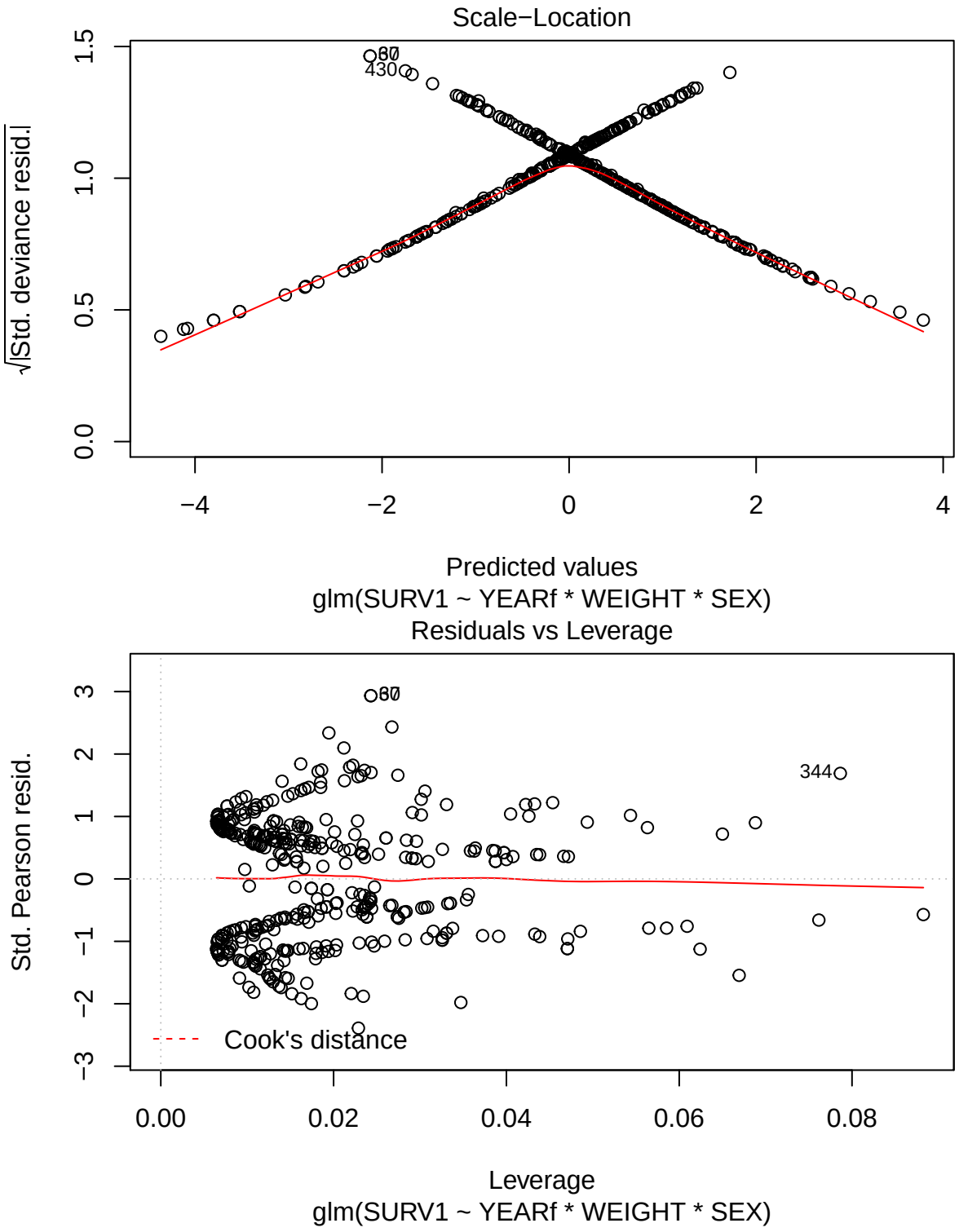
```
mm1<-glm(SURV1~YEARf*WEIGHT*SEX,
         family="binomial",
         data=ss)
```

We are using `glm` versus `lm`. A family must be specified, here `binomial`. Specifying `gaussian` would be the equivalent of `lm`.

- Let's look at diagnostic plots using:

```
plot(mm1)
```


Residuals vs Fitted
Residuals
Predicted values
glm(SURV1 ~ YEARf * WEIGHT * SEX)
Normal Q−Q
Std. deviance resid.
Theoretical Quantiles
glm(SURV1 ~ YEARf * WEIGHT * SEX)



should they make sense?

- Look at the output:

```
summary(mm1)
```

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

```
## glm(formula = SURV1 ~ YEARf * WEIGHT * SEX, family = "binomial",
##     data = ss)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.9411  -1.0528   0.4295   0.9519   2.1168
##
## Coefficients:
##                      Estimate Std. Error z value Pr(>|z|)
## (Intercept)       -146.91458   61.48837  -2.389   0.0169 *
## YEARf                1.50219    0.64849   2.316   0.0205 *
## WEIGHT               8.44691    4.57280   1.847   0.0647 .
## SEXM                50.92858   79.18240   0.643   0.5201
## YEARf:WEIGHT        -0.08516    0.04827  -1.764   0.0777 .
## YEARf:SEXM          -0.52020    0.83512  -0.623   0.5334
## WEIGHT:SEXM         -4.06199    5.56746  -0.730   0.4656
## YEARf:WEIGHT:SEXM    0.04108    0.05877   0.699   0.4845
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 580.87  on 419  degrees of freedom
## Residual deviance: 483.89  on 412  degrees of freedom
##   (81 observations deleted due to missingness)
## AIC: 499.89
##
## Number of Fisher Scoring iterations: 5
```

Most importantly at this point is to examine the Null and Residual deviance reporting

> Null deviance: the deviance associated with the null model, which has no explantory terms asssociated with it (except the grand mean). This is equivalent to the total sum of squares, the equivalent model would be `SURV1~1`
>
> Residual Deviance: the deviance associated witht the *current model* (here the maximal model).

```
glm(formula = SURV1 ~ factor(YEARf) * WEIGHT * SEX,
    family = "binomial",
    data=ss)
```

```
##
## Call:  glm(formula = SURV1 ~ factor(YEARf) * WEIGHT * SEX, family = "binomial",
##     data = ss)
##
## Coefficients:
##                 (Intercept)             factor(YEARf)93
##                    -9.72120                   -1.77586
##             factor(YEARf)94             factor(YEARf)95
##                     1.41897                    8.37798
##             factor(YEARf)96                     WEIGHT
##                     5.04929                    0.66367
##                        SEXM      factor(YEARf)93:WEIGHT
##                    -2.96953                    0.15991
##      factor(YEARf)94:WEIGHT      factor(YEARf)95:WEIGHT
##                    -0.07257                   -0.41811
##      factor(YEARf)96:WEIGHT        factor(YEARf)93:SEXM
```

```
##                     -0.28517                       8.08302
##        factor(YEARf)94:SEXM         factor(YEARf)95:SEXM
##                     10.30572                       7.86080
##        factor(YEARf)96:SEXM                   WEIGHT:SEXM
##                      3.45458                       0.05042
## factor(YEARf)93:WEIGHT:SEXM  factor(YEARf)94:WEIGHT:SEXM
##                     -0.47187                      -0.62239
## factor(YEARf)95:WEIGHT:SEXM  factor(YEARf)96:WEIGHT:SEXM
##                     -0.32053                      -0.14547
##
## Degrees of Freedom: 419 Total (i.e. Null);  400 Residual
##   (81 observations deleted due to missingness)
## Null Deviance:       580.9
## Residual Deviance: 411.6     AIC: 451.6
```

Note the line which says

(Dispersion Parameter for Binomial family taken to be 1)

This means that the model assumes that the variance is approximately constant relative to the mean. We can get an estimate of the empirical dispersion parameter (also known as the scale parameter) by calculating the ratio of residual deviance to residual degrees of freedom.

Question: What is the approximate dispersion parameter for our current model?

Tip: Use R as a simple calculator to divide the residual diveiance by its degrees of freedom

If the dispersion parameter is close to 1, then the model provides a good fit to the data. If the parameter is very small or large, a different model should be built.

Now let's look at the Analysis of Deviance table. You will remember that this is like an analysis of variance table, except that maximum-likelihood methods have been used to estimate deviances, rather than sums of squares. The default form of the analysis of deviance table printed by R does not calculate P values, because these are dependent on which test statistic is calculated (X^2, F-value, Cp statistic etc). Usually we would want to calculate X^2 (Chi-square) values. To print X^2 and P values:

```
anova(mm1, test="Chisq")
```

```
## Analysis of Deviance Table
##
## Model: binomial, link: logit
##
## Response: SURV1
##
## Terms added sequentially (first to last)
##
##
##                  Df Deviance Resid. Df Resid. Dev  Pr(>Chi)
## NULL                                419     580.87
## YEARf             1   24.766       418     556.11 6.472e-07 ***
## WEIGHT            1   53.255       417     502.85 2.929e-13 ***
## SEX               1    9.205       416     493.64  0.002414 **
## YEARf:WEIGHT      1    4.600       415     489.04  0.031970 *
## YEARf:SEX         1    0.560       414     488.48  0.454291
## WEIGHT:SEX        1    4.098       413     484.39  0.042936 *
## YEARf:WEIGHT:SEX  1    0.493       412     483.89  0.482607
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

You should obtain a print-out which indicates the different terms in the model, their associated degrees of freedom, their deviances and a progressive decline in the residual degrees of freedom and residual deviances, plus the associated probability values. > Notice that at the top of the table it says: > Terms added sequentially (first to last) > This means that the associated P-value is dependent on the position of each term in the model (equivalent to sequential sums of squares). Thus, if we have the model `SURV1~YEARf*WEIGHT*SEX`, then only the P-value for `YEARf:WEIGHT:SEX` is reliable. • Now we have to perform step-wise deletion test: first, we will do it the long way, then the cheat way. • As in previous exercises, we use the update function of R:

```
mm2 <- update(mm1,~.-YEARf:WEIGHT:SEX)
anova(mm1, mm2, test="Chisq")
```

```
## Analysis of Deviance Table
##
## Model 1: SURV1 ~ YEARf * WEIGHT * SEX
## Model 2: SURV1 ~ YEARf + WEIGHT + SEX + YEARf:WEIGHT + YEARf:SEX + WEIGHT:SEX
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1       412     483.89
## 2       413     484.39 -1 -0.49297   0.4826
```

We have performed an analysis of variance comparing the deviances of the original model (mm1) and the model that lacks the 3-way interaction (mm2). An analysis of deviance table is shown, comparing these 2 models, showing the Terms in the two models and their respective residual degrees of freedom and deviances. Test shows which terms differ between the two models and the difference in the models df, deviance, and the deviance (2) value and P value for difference between the two models. • Now we (you) can produce three more models, each one removing one of the remaining 2-way interaction • By repeating this process for all of the terms in the model, we should be able to produce a putative minimal model. • Test each of these deletions against the more substantial model with all 2-way interactions.

```
mm3a <- update(mm2,~.-WEIGHT:SEX)
anova(mm2, mm3a, test="Chisq")
```

```
## Analysis of Deviance Table
##
## Model 1: SURV1 ~ YEARf + WEIGHT + SEX + YEARf:WEIGHT + YEARf:SEX + WEIGHT:SEX
## Model 2: SURV1 ~ YEARf + WEIGHT + SEX + YEARf:WEIGHT + YEARf:SEX
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1       413     484.39
## 2       414     488.48 -1  -4.0979  0.04294 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

• Based on this exercise, which 2-way interactions remain in the model? • Now, recall the dropterm() function.

• In one fell swoop, you have all of the detail above. • Now, notice that with YEARf:WEIGHT and YEARf:SEX nominated for removal from the model, YEARf does not figure in any interactions. Thus, our candidate final, minimum adequate model should be easy to specify. * Write it out as `mm4<-glm(…)`

```
require(MASS)
```

```
## Loading required package: MASS
```

```
mm4<-glm(SURV1~YEARf+(WEIGHT*SEX),
         family="binomial",
         data=ss)

dropterm(mm4,test="Chisq")
```

```
## Single term deletions
##
## Model:
## SURV1 ~ YEARf + (WEIGHT * SEX)
##            Df Deviance    AIC     LRT   Pr(Chi)
## <none>          489.30 499.30
## YEARf       1   518.01 526.01 28.7182 8.371e-08 ***
## WEIGHT:SEX  1   493.64 501.64  4.3492   0.03703 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

- TIP:

```
mm4<-glm(SURV1~YEARf+(WEIGHT*SEX),
         family="binomial",
         data=ss)
```

> Question: does the removal of the 3-way interaction-term signifcantly reduce the explanatory power of our model? • Now, we appear to be at the minimal model • Use anova(mm4) to examine the empirical scale parameter. Is it still OK? • ??? How would you specifically test for the significance of YEARf? • ??? WHY does `dropterm` only report the WEIGHT:SEX interaction and the main effect of YEARf? WHY are the main effect of WEIGHT and SEX "unimportant"?

ADVANCED GRAPHICS MEETS GLM • It is quite difficult to understand whether your binomial glm is reporting sensible prediction based on binomial data. One way to handle this is to use a spreadsheet, or R, to calculate the % of different type of events. This would give you a rough estimate of your raw data, that you could plot against various explanatory variables. • With the output of your model, we can also plot the predicted, fitted response to our main factors and model covariates. These will be on the sensible % scale as well and produce an interpretation of your data that is hard to develop from raw data. * The first step is to produce a predicting data set.

```
require(broom)
```

```
## Loading required package: broom
```

```
?augment
mm4_augmented <- augment(mm4,
                         type.predict="response")
```

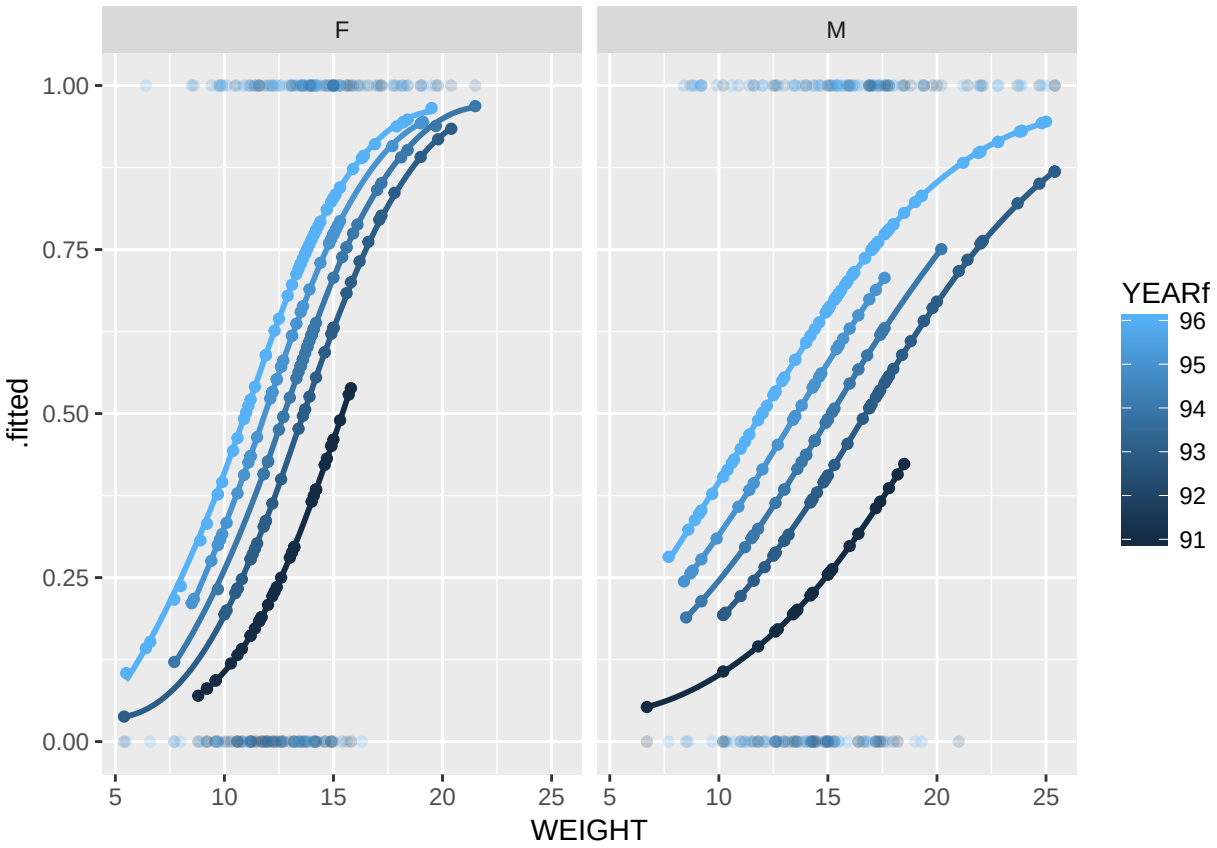
- This new dataset is compatible with ggplot2

```
require(ggplot2)
```

```
## Loading required package: ggplot2
```

```
p <- qplot(data=mm4_augmented,
           x=WEIGHT,
           y=.fitted,
           colour=YEARf)+
  facet_grid(.~SEX)+
  geom_smooth(aes(group=YEARf))+
  geom_point(aes(y=SURV1),alpha=0.2)
print(p)
```

```
## `geom_smooth()` using method = 'loess'
```



? DO HEAVIER ANIMALS HAVE A BETTER CHANCE OF SURVIVAL? ? ARE MALES OR FEMALES MORE SENSITIVE TO WEIGHT? ? WHICH YEAR WAS THE BEST YEAR? ? HOW COULD YOU CHANGE THE PLOT FUNCTION SO THAT YOU SEE THE MALE-FEMALE EFFECT IN EACH YEAR?

Weighted binomial data and dipsersion

The Issue: As Crawley points out in his books (from 1993 onwards), one problem with proportion data is that a 50% effect (e.g. survival) based on 2 samples is very different from a 50% effect based on 400 samples. The weighting of the effect is very different. In the first situation, you can only get 0, 50 or 100% survival. Crawley provides a good example with small sample sizes that shows how this can be mis-interpreted using standard anova and transformations. Here we provide an example of how to use the weighting syntax and simultaneously, how to deal with overdispersion in a binomial regression. The approach for dispersion is the same for a poisson regression, save for the use of different functions. We use an example from Crawley's book on the germination of two genotypes of a parasitic plant, stimulated with bean and cucumber extracts, to initiate germination. The data have estimates of the numbers of each genotype germinating, the sample size, the genotype and the extract treatment. As standard, we import the data and examine its structure.

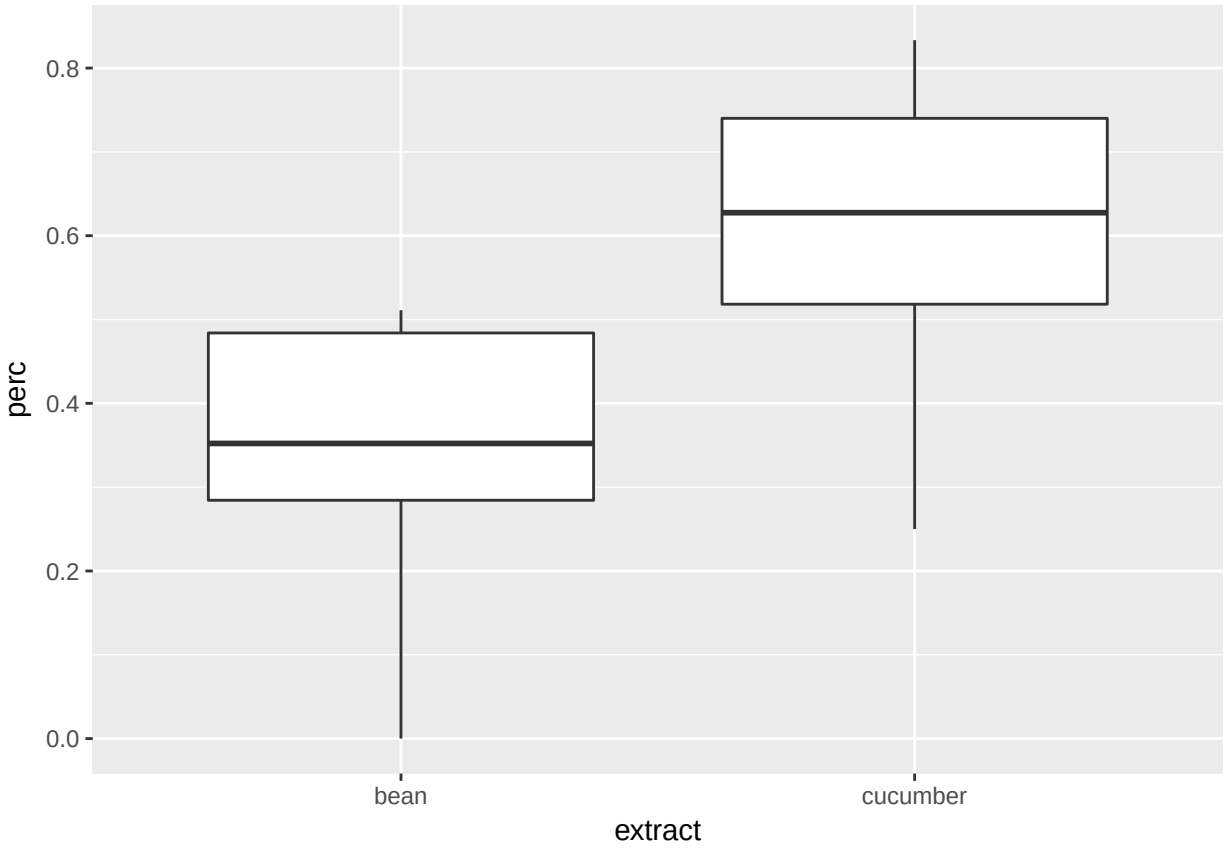
```
gg <- read.csv("./Data/germination.csv")
str(gg)
```

```
## 'data.frame':    21 obs. of  4 variables:
##  $ count    : int  10 23 23 26 17 5 53 55 32 46 ...
##  $ sample   : int  39 62 81 51 39 6 74 72 51 79 ...
##  $ Orobanche: Factor w/ 2 levels "a73","a75": 2 2 2 2 2 2 2 2 2 2 ...
##  $ extract  : Factor w/ 2 levels "bean","cucumber": 1 1 1 1 1 2 2 2 2 2 ...
```

Now, the data are counts per sample, which means we can calculate the number that have germinated and not, by subtracting the count data from the sample. Using R's vector based operations, we create a new object, a new response variable, which is a two column matrix of the number that germinate and the number that did not. Let's also calculate the percentages, so that we can have a quick first look at the data graphically.

```
gg <- within(gg,{
  non_germ <- sample-count
  perc <- count/sample
})
```

```
p <- qplot(data=gg,
      x=extract,
      y=perc,
      geom="boxplot")
print(p)
```



Now, remember, we are not using the percentage data because of the bias that we know is inherent in the distribution of this type of data. Not even with the transformations. We can simply do much better by dealing with the expected error distribution directly.

Notice how germ has the number of germinated and ungerminated plants. This could be presences and absence, success or failure, task done or not. All of these are binary data and are appropriate for logistic/binary regression. Now, let's look at the model. A binomial glm can obviously handle the 1,0 data as in the sheep example. It also handles this dual column response variable – in fact, it is preferred. It is efficient and removes the bias of small sample sizes by explicitly dealing with the weights of each observation.

```
m1 <- glm(cbind(count,non_germ)~Orobanche*extract,
          family=binomial,
```

```
          data=gg)
summary(m1)
```

```
##
## Call:
## glm(formula = cbind(count, non_germ) ~ Orobanche * extract, family = binomial,
##     data = gg)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -2.01617  -1.24398   0.05995   0.84695   2.12123
##
## Coefficients:
##                             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:extractcucumber  0.7781     0.3064   2.539   0.0111 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (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
##
## Number of Fisher Scoring iterations: 4
```

Recall that our major diagnostic tool is the ratio between the residual deviance and the residual degrees of freedom: 33.2/17 is 1.9, nearly 2. This suggests a moderate level of overdispersion. What do we do? In R, we refit this model with a quasibinomial family. This allows the dispersion parameter to be estimated, rather than fixed at 1.

```
m2 <- glm(cbind(count,non_germ)~Orobanche*extract,
          family="quasibinomial",
          data=gg)
```

```
anova(m1)
```

```
## Analysis of Deviance Table
##
## Model: binomial, link: logit
##
## Response: cbind(count, non_germ)
##
## Terms added sequentially (first to last)
##
##
##                   Df Deviance Resid. Df Resid. Dev
## NULL                                 20     98.719
## Orobanche          1    2.544        19     96.175
## extract            1   56.489        18     39.686
## Orobanche:extract  1    6.408        17     33.278
```

```
anova(m2)
```

```
## Analysis of Deviance Table
##
## Model: quasibinomial, link: logit
##
## Response: cbind(count, non_germ)
##
## Terms added sequentially (first to last)
##
##
##                   Df Deviance Resid. Df Resid. Dev
## NULL                                 20     98.719
## Orobanche          1    2.544        19     96.175
## extract            1   56.489        18     39.686
## Orobanche:extract  1    6.408        17     33.278
```

```
m3 <- glm(cbind(count,non_germ)~extract,
          family="quasibinomial",
          data=gg)
```

```
m3_augmented <- augment(m3,
                        type.predict="response")
```

```
require(dplyr)
```

```
## Loading required package: dplyr
```

```
##
## Attaching package: 'dplyr'
```

```
## The following object is masked from 'package:MASS':
##
##     select
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
m3_augmented %>%
  group_by(extract) %>%
  summarise(mean_resp=mean(.fitted))
```

```
## # A tibble: 2 x 2
##    extract mean_resp
##     <fctr>     <dbl>
## 1     bean 0.3746835
## 2 cucumber 0.6330275
```

So, it should be clear that: a) you should use the cbind(success,fail) version of a dependent variable for binomial glms b) you need to examine the dispersion of a binomial glm c) if dispersion is present, you need to refit the model with quasibinomial family d) if the dispersion is near 1, use a chi-square test. If dispersion is large (or small), use an F-test with the quasi- family.

More Advanced GLMs!

Data import and exploration • read.csv the file bjobs.csv from the VLE

```
bjobs <- read.csv("./Data/bjobs.csv")
```

• The data represent a nationwide survey of the number of lesser-spotted spotty-birds (a little 'brown job') nesting in 127 representative districts in England, Scotland and Wales. • Our task is to determine whether there is any relationship between the area of the district and the number of lesser-spotted spotty birds nesting there, and whether this relationship is consistent between the three countries. • The data are as follows:

```
str(bjobs)
```

```
## 'data.frame':    127 obs. of  4 variables:
##  $ District: int  1 2 3 4 5 6 7 8 9 10 ...
##  $ Area    : num  30.6 52.8 56.2 40.8 66.4 ...
##  $ Number  : int  25 47 50 34 59 51 44 20 16 34 ...
##  $ Country : Factor w/ 3 levels "ENGLAND","SCOTLAND",..: 1 1 1 1 1 1 1 1 1 1 ...
```

Variable Description District District Number (1-127) Area Area of the district (ha) Number Number of pairs of birds nesting in the district Country ENGLAND, WALES, SCOTLAND

• Attach the file.. Get a feel for the data by producing Summary Statistics (stratified by Country) and Plots.

Q: Are the data normally distributed?

TIP: Why not produce a normality plot by typing qqnorm(Number) TIP: You can produce a normality plot for just England by typing the following: qqnorm(Number[Country==“ENGLAND”])

GLMs with Poisson errors

• The data look approximately normal, and in the past we would probably have analysed these data by assuming that they were normal (and continuous) and performing a linear regression with Number of birds as your dependent variable. • With generalised linear modelling we do not need to make this erroneous assumption! You will remember from the GLM lecture that counts like these are usually Poisson distributed, so now let's try to analyse the data using a GLM with Poisson errors. `Number~Area*Country` • Note that this model is effectively an analysis of covariance, with Area as the covariate and Country as the factor in the model, except that in this instance instead of assuming Normal errors we will be assuming Poisson errors. NB: Effectively, with this model we are asking the question: is there a significant relationship between Number of birds and Area, and is the slope of that regression the same for all 3 countries? Begin by plotting the data in various ways to see if you think there are relationships in the data

NB: In this particular instance, this simple model is also our maximal model

```
brown1<-glm(Number~Area*Country,
            family=poisson,
            data=bjobs,
            na.action=na.omit)
```

NB: What does na.action=na.omit do? If you have attach-ed the data sheet do you need the data=bjobs command? • explore the model output, Anodev table and graphics output for the model

```
        summary(brown1)
```

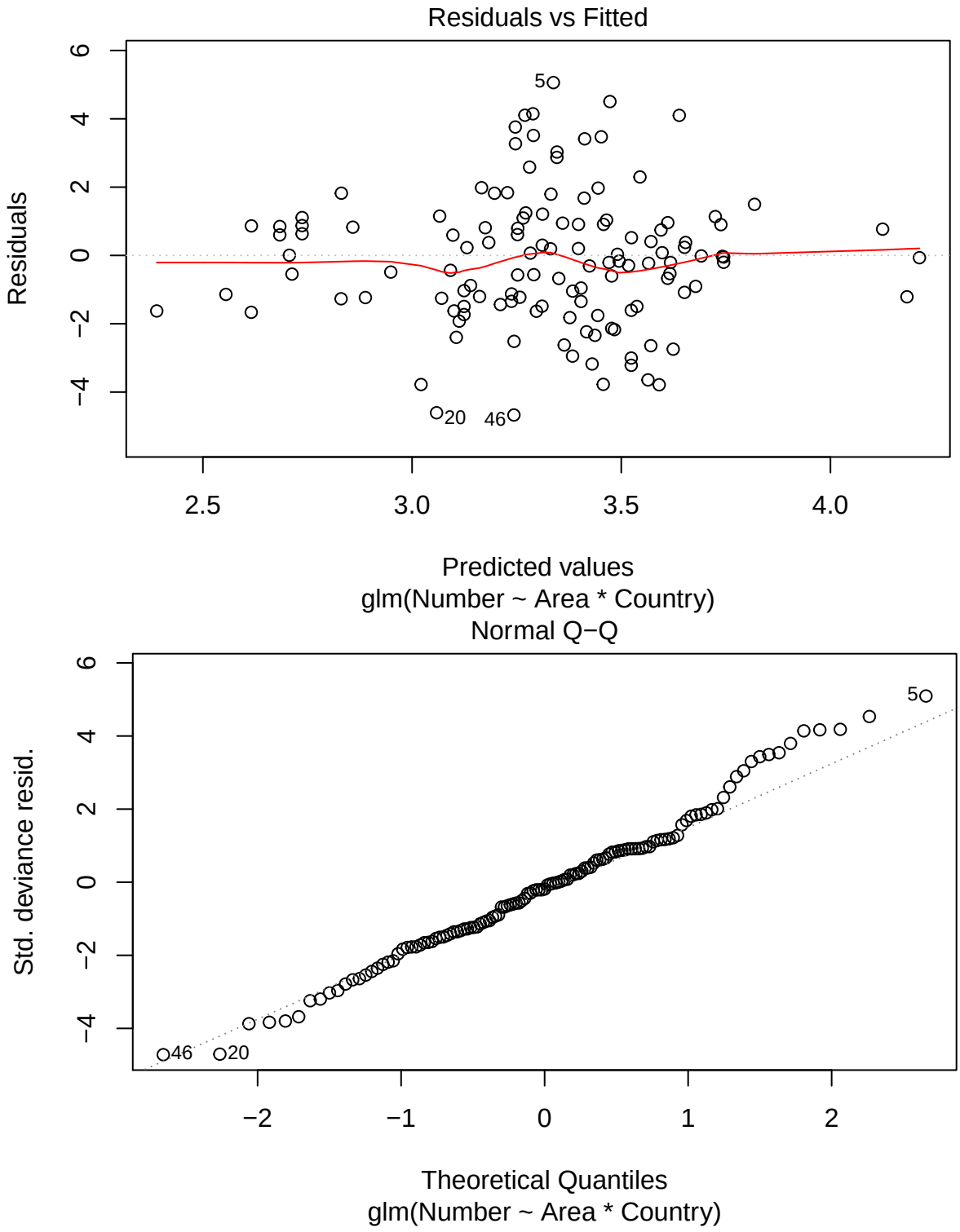
```
##
## Call:
## glm(formula = Number ~ Area * Country, family = poisson, data = bjobs,
##     na.action = na.omit)
```

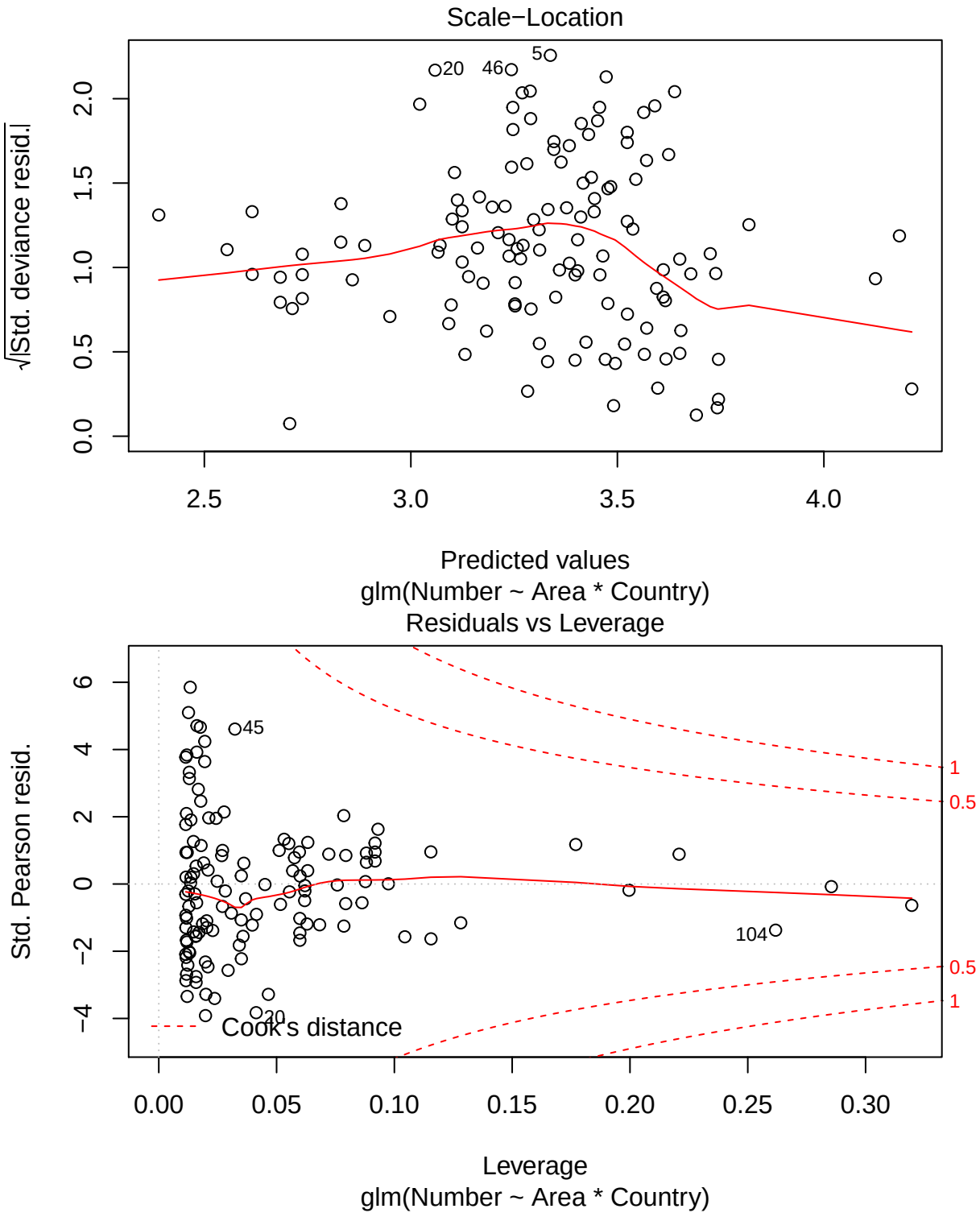
```
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -4.6713  -1.3925  -0.1664   0.9086   5.0601
##
## Coefficients:
##                         Estimate Std. Error z value Pr(>|z|)
## (Intercept)           2.8928679  0.0603126  47.965  < 2e-16 ***
## Area                  0.0066988  0.0007092   9.445  < 2e-16 ***
## CountrySCOTLAND      -4.1331518  0.6734718  -6.137 8.41e-10 ***
## CountryWALES         -3.4191929  0.3232254 -10.578  < 2e-16 ***
## Area:CountrySCOTLAND  0.0236321  0.0047137   5.014 5.34e-07 ***
## Area:CountryWALES     0.0227075  0.0023830   9.529  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 833.35  on 126  degrees of freedom
## Residual deviance: 484.52  on 121  degrees of freedom
## AIC: 1143.2
##
## Number of Fisher Scoring iterations: 4
```

```
anova(brown1, test="Chisq")
```

```
## Analysis of Deviance Table
##
## Model: poisson, link: log
##
## Response: Number
##
## Terms added sequentially (first to last)
##
##
##              Df Deviance Resid. Df Resid. Dev  Pr(>Chi)
## NULL                            126     833.35
## Area          1   33.257       125     800.10 8.076e-09 ***
## Country       2  202.677       123     597.42 < 2.2e-16 ***
## Area:Country  2  112.902       121     484.52 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
plot(brown1)
```


Residuals vs Fitted
Residuals
Predicted values
glm(Number ~ Area * Country)
Normal Q−Q
Std. deviance resid.
Theoretical Quantiles
glm(Number ~ Area * Country)



• Note the line which says: (Dispersion Parameter for Poisson family taken to be 1) • This means that the model assumes that the variance is approximately constant relative to the mean [in fact for Poisson distributions, the variance equals the mean]. • We can get an estimate of the dispersion parameter by calculating the ratio of residual deviance to residual degrees of freedom. Q: What is the approximate dispersion parameter for our current model? TIP: Use R as a simple calculator to divide the residual deviance by its d.f.

NB: If the dispersion parameter is close to 1 (and certainly <2 and >0.5), then our model provides a good fit to the data. If the dispersion parameter is very large or small, then we need to construct a different kind of model. • For our model, the dispersion parameter is much greater than 1 and in fact is around 4. Thus, our data are overdispersed (i.e. aggregated) and one of the assumptions of our model is violated and we must consider a different model. • You should obtain a print-out showing the different terms in the model, their associated degrees of freedom, their deviances and a progressive decline in the residual degrees of freedom and residual deviances, plus the associated probability values. • Normally at this point, we would start to perform step-wise deletion tests. However, we have already determined that our dispersion parameter is too large. There are a number of ways we can get round this problem. How do we cope with overdispersion? Quasi-likelihood • One way of getting round the problem of overdispersion is to construct a quasi-likelihood model. This is similar to standard GLMs, except it can be used to estimate regression relationships without fully knowing the error distribution of the response variable. Simply type the following: brown2<-glm(Number~Area*Country, family=quasi(link="log", var="mu"), data=bjobs, na.action=na.omit) or the equivalent brown2<-glm(Number~Area*Country, family=quasipoisson, data=bjobs, na.action=na.omit) • This command is exactly the same as that for brown1, except that instead of using the Poisson family, we have used the Quasi-likelihood family. Inside the inner brackets, we have told R that we want to use a log link function (as we did for the Poisson model) and that the variance should be a function of the mean (mu), as it was for the Poisson. NB: The details required for this part of the function will vary depending on what the response variable is (e.g. for the logistic regression we did in the last Exercise, we tried quasi-likelihood with a logit link function and binomial errors: family=quasi(link="logit", var="mu(1-mu)"). Note: quasipoisson is a subset of the more general quasi() command, the latter's flexibility requiring you to specify the mean-variance relationship, [quasi(link=log, var="mu") is equivalent of quasipoisson] TIP: If you want to know what the default values are for particular families, simply type the following: family(quasi),or family(poisson), or family(binomial), etc

• Compare the means and standard errors for your parameter estimates from brown1 (your poisson model) and brown2 (quasi-likelihood model). TIP: Copy and paste both sets of estimates into a Script file, if it makes it easier to compare.

Q: Which model gives the larger standard errors? What about the means?

NB: Notice that in the quasi-likelihood model, the overdispersion is implicitly taken care of by indicating an empirical dispersion parameter, as indicated by the line: (Dispersion Parameter for Quasi-likelihood family taken to be 4.018633)

• We can also perform an analysis of variance on this model, but this time, because we are using a quasi-likelihood model and an empirical dispersion parameter, we test for significance using F-values! Simply type: anova(brown2, test="F")

Q: How do you interpret this Anova?

GLMs with Negative Binomial errors

• A better way of coping with overdispersion in count data is to model it using negative binomial errors (and for parasite data, this would probably be your starting point).

But first, a bit of theory…. • For the Poisson distribution, the variance is equal to the mean, i.e. variance = mean [1] • For the negative binomial distribution, the variance changes with the mean as follows: variance = mean + mean2/k [2] • where k is inversely proportional to the skew in the distribution, so that if our distribution is very skewed, k will be small. • Notice that when k is very large, mean2/k becomes very small, and equation 2 becomes: variance ~ mean, which describes the Poisson distribution (equation 1) ! • Notice also that equation 2 can be re-written to: variance = mean * (1 + mean/k) [3] • Which can also be re-written as: variance = mean * [4] • where = (1 + mean/k). • Thus, when we use the quasi-likelihood method, we quantify the variance by multiplying the mean by , an empirical estimate of the dispersion parameter. • Hopefully, it will now be clear that: (a) when k is large, the negative binomial distribution is equivalent to the Poisson distribution, and, (b) when we do quasi-likelihood we are gaining only an estimate

of the true relationship between the variance and the mean (because we are using an empirical estimate of the dispersion parameter)!

• To construct a model with negative binomial errors, we use the glm.nb function (acknowledgements to Venables & Ripley), but first we need to load the appropriate library of functions: • library(MASS)

NB: You will need to type this at the beginning of every session in which you use the glm.nb function. NB: To check what is in the MASS library (an abbreviation of Modern Applied Statistics with Splus, by Venables & Ripley), instead of clicking on Load library, click on Description and OK.

• Now simply type:

```
brown3<-glm.nb(Number~Area*Country,
               data=bjobs,
               na.action=na.omit)
```

NB: Notice that we do not need to add a family= argument, because the family is automatically negative binomial (as indicated by glm.nb) • Now, generate a summary of brown3 Q: What is the dispersion parameter for this model? • As the dispersion parameter is around 1, we have a good fitting model.Thus, the relationship between the variance and the mean is adequately described by the negative binomial. • If you check the output from summary(brown3) you will notice that at the end it says: Theta: 9.78049 Std. Err.: 1.68657 • Theta is the R way of describing k of the negative binomial. Thus, we have an estimate of the degree of skew in our data. Relative to most parasite data, k = 9.78 is relatively large, suggesting that there is relatively little skew (for a Poisson distribution the appropriate value is at least k = 20). • If you now check out the standard errors of your parameter estimates in the summary of model brown3, you should find that they lie somewhere between those generated by the ill-fitting Poisson model (brown1) and the more-rigorous, but parsimonious, quasi-likelihood model (brown2). • Now let's construct an analysis of deviance table, by typin

```
anova(brown3, test="Chisq")
```

```
## Warning in anova.negbin(brown3, test = "Chisq"): tests made without re-
## estimating 'theta'
```

```
## Analysis of Deviance Table
##
## Model: Negative Binomial(9.7805), link: log
##
## Response: Number
##
## Terms added sequentially (first to last)
##
##
##              Df Deviance Resid. Df Resid. Dev  Pr(>Chi)
## NULL                           126     220.65
## Area          1    8.461       125     212.19  0.003627 **
## Country       2   53.134       123     159.06 2.898e-12 ***
## Area:Country  2   28.214       121     130.84 7.473e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

NB: Notice that we are again testing our model using the chi-square test, rather than the F-test, because our dispersion parameter is close to 1 and so we do not need to use an empirical estimate Q: How do you interpret this output? • You can also check the residual plots, by plot(brown3) Q: How do you interpret this analysis?

Using Akaike weights

Modern statistics are increasingly turning to the use of information theoretic criteria to assess model fits, recognising that for many situations, there may be a range of competing alternative models, all with some

support. To illustrate it, here is a bit of code – that is not very elegant – you can generalise it and turn it into a function as an exercise!

Akaike weights are simply a weighted measure of AIC (or AICc for small samples). They are calculated from the differences in AIC between two models, with the "best" model having the smallest AIC. The relative likelihood of a model is exp(-0.5 * AIC score for that model). The Akaike weight for a model is this value divided by the sum of these values across all models.

Go back to soay2 (reload and re-attach as necessary)

```
s2<-read.csv("./Data/soay2.csv")
```

Set up a range of example models to illustrate:

```
m1<-glm(WEIGHT~factor(AGE)*STR*SEX, data=s2)
m2<-glm(WEIGHT~factor(AGE)*SEX+STR, data=s2)
m3<-glm(WEIGHT~factor(AGE)*SEX, data=s2)
```

Now some code to create a data.frame to hold results, called aics. Put this code in a script and execute it step by step – and look at the object aics after each step - so you can work out what each line is doing.

```
aics<-data.frame(paste("m",1:3,sep=""),c(m1$aic,m2$aic,m3$aic),row.names=NULL)
colnames(aics)<-c("model","AIC")
aics<-aics[order(aics$AIC),]
for(i in 1:dim(aics)[1]){aics$diff[i]<-aics$AIC[1]-aics$AIC[i]}
aics$wi<-2.71828182845904523536^(0.5*aics$diff)
aics$aic.weights<-aics$wi/sum(aics$wi)
```

Now when you type `aics` you should have a frame with the models ranked in order of AIC, with the wi and weights as separate columns. If the AIC are small, the models should be seen as equivalent (and so you can average coefficients in "model averaging". Try playing around with a range of models and seeing how the weights change.

Finally A note on GAMs – Generalised Additive Models that incorporate non-parametric smoothing terms.

Although the data isn't really appropriate to fitting a GAM here is some code to do a model for the `bjobs` bird counts, ignoring the fact that the counts come from different countries, and just concentrating on Numbers~Area.

A simple plot of the data doesn't indicate any curvature in the data. However, when you look at the mean values for area rounded to the nearest 10 you can see that the underlying pattern isn't a straight line!

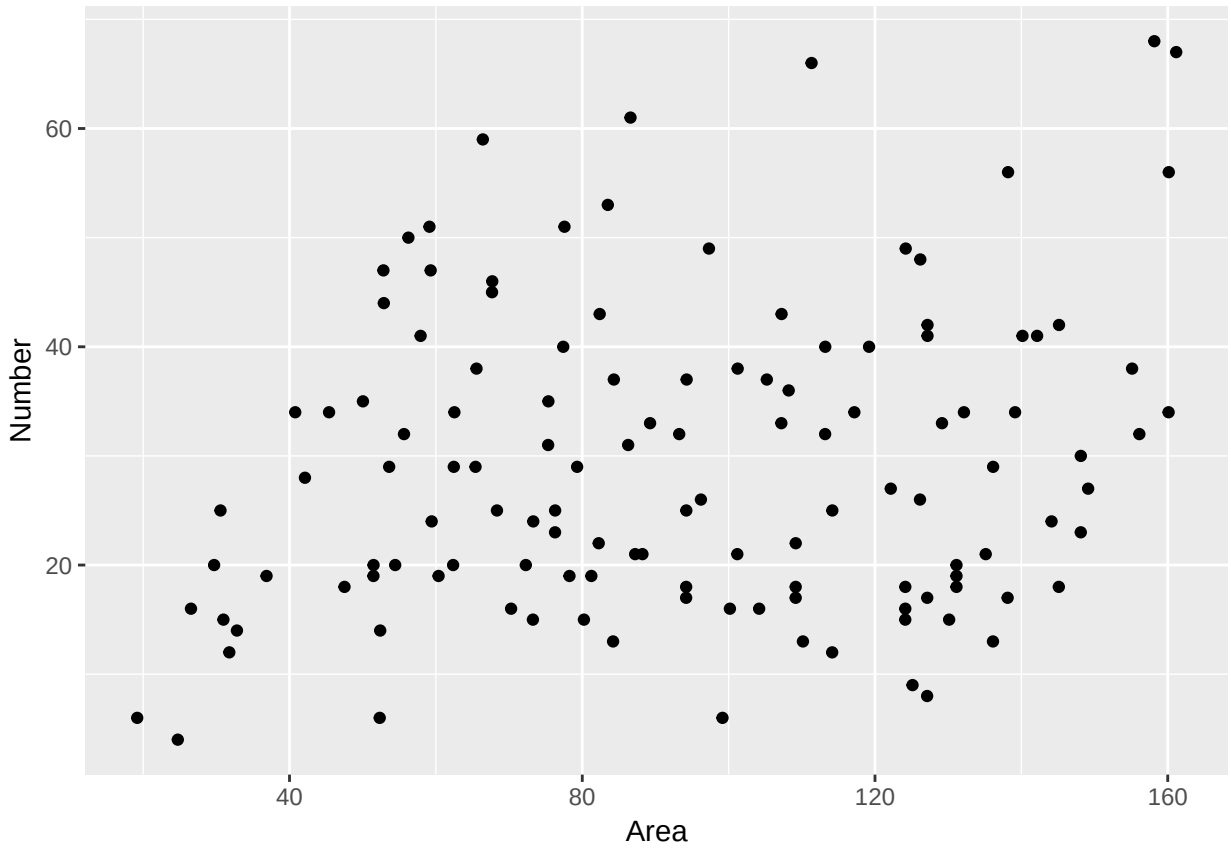
```
bjobs <- within(bjobs,{
  area2<-round(Area/10,0)*10
  })
```

this creates new col with area rounded to nearest 10

calculates means for each area rounded to nearest 10:

```
bjobs <- bjobs %>%
  group_by(area2)%>%
  mutate(aves=mean(Number))
```

```
p <- qplot(data=bjobs,
      y=Number,
      x=Area)
print(p)
```



Now do the GAM

```
library(mgcv)
```

```
## Loading required package: nlme
```

```
##
## Attaching package: 'nlme'
```

```
## The following object is masked from 'package:dplyr':
##
##     collapse
```

```
## This is mgcv 1.8-22. For overview type 'help("mgcv-package")'.
```

```
brown5<-gam(Number~s(Area,k=10)+Country,
            family=poisson,
            data=bjobs) #gam model
anova.gam(brown5)
```

```
##
## Family: poisson
## Link function: log
##
## Formula:
## Number ~ s(Area, k = 10) + Country
##
```

```
## Parametric Terms:
##         df Chi.sq p-value
## Country  2  238.9  <2e-16
##
## Approximate significance of smooth terms:
##           edf Ref.df Chi.sq p-value
## s(Area) 5.705  6.839  348.9  <2e-16
```

```
plot(brown5)    # gam plot of model showing (partial) GAM fit
```