fishR Vignette - Maturity Schedules

Dr. Derek Ogle, Northland College December 16, 2013

XXX THIS IS A WORK IN PROGRESS XXX

This vignette requires functions in the `FSA` and `FSAdata` packages maintained by the author and the `car` package available on CRAN. These packages are loaded into R with

```
> library(FSA)
> library(FSAdata)
> library(car)
```

1 Background

2 Maturity Data

Raw maturity data generally consists of a maturity statement (either "mature" or "immature"), size (length or weight), age, sex, and other variables as needed (e.g., capture date, capture location) recorded for individual fish. The maturity variable may need to be derived from more specific data about the "stage of maturation" recorded for each fish. Often, maturity will be recorded as a "0" for immature and a "1" for mature, though this is not required for most modern softwares. Sex is an important variable to record as maturity should be analyzed separately for each sex.

Summarized maturity data consists of the proportion of individuals that are mature within each age or length category. Age categories are generally the recorded ages whereas recorded lengths are often categorized into bins. Age or length categories should be as narrow as possible but include enough individuals such that the proportion mature in each bin can be reasonably computed.

In this vignette, the length-related maturity schedules for female Yelloweye Rockfish (*(Sebastes rubberimus*) collected from along the Oregon coast will be examined. These data can be found in **YERockFish** in the `FSAdata` package and are loaded with

```
> data(YERockfish)
> str(YERockfish)
'data.frame': 158 obs. of  5 variables:
 $ date    : Factor w/ 71 levels "10/1/2003","10/2/2003",..: 66 9 38 25 60 6 37 48 34 39 ...
 $ length  : int  31 32 32 32 32 33 33 34 34 34 ...
 $ age     : int  10 6 11 11 13 9 10 8 10 11 ...
 $ maturity: Factor w/ 2 levels "Immature","Mature": 1 1 1 1 1 1 1 1 1 1 ...
 $ stage   : Factor w/ 9 levels "1","2","3","4",..: 1 1 1 2 2 1 1 1 1 1 ...
> view(YERockfish)
          date length age maturity stage
26   6/16/2002     38  18   Mature     2
37    6/3/2002     40  15   Mature     3
50  10/25/2003     42  15   Mature     8
69   5/21/2001     44  20   Mature     6
144  7/28/2002     57  70   Mature     6
158  4/23/2001     70  66   Mature     4
```

Total length of the rockfish was measured to the nearest 1-cm. Length categories of 2-cm were chosen to summarize the data as these length bins provide reasonable sample sizes ($>$ 10 fish) in the length ranges

where the proportion of mature fish is changing most rapidly (Figure 1). The 2-cm length categories are added to the data frame with `lencat()` which requires a formula of the form `~len` as the first argument where `len` generically represents the length variable in the the original data frame, the name of the data frame in `data=`, the size of the starting length category in `startcat=`, and the width of length categories in `w=`. The result of `lencat()` is then saved to a dataframe. Adding the 2-cm length categories to the rockfish data frame is accomplished with

```
> YERockfish <- lencat(~length,data=YERockfish,startcat=30,w=2)
> view(YERockfish)
```

```
         date length age maturity stage LCat
13  7/17/2000     36  10 Immature     1   36
41  7/18/2000     41  11   Mature     2   40
65  6/11/2002     44  16     <NA>     2   44
119 10/7/2002     49  18   Mature     7   48
129 6/16/2002     52  30   Mature     6   52
156 8/18/2002     67  50   Mature     6   66
```

The frequency of mature and immature fish in each length category is computed with `table()` where the "row" variable (should be the length category) is provide first. The result of `table()` should be saved to an object name so that it can be submitted to `prop.table()` to compute the proportion of mature fish in each length category. The "row" proportions are calculated, which will be the proportion within each length category if the length categories were provided as rows in `table()`, by including `margin=1` to `prop.table()`. The proportion of rockfish that are mature in each 2-cm length category is computed with (only the first six rows are shown to save space)

```
> tblLen <- with(YERockfish,table(LCat,maturity))
> ptblLen <- prop.table(tblLen,margin=1)
> ptblLen[1:6,]   # only 1st 6 rows to save space
```

```
    maturity
LCat Immature Mature
  30   1.0000 0.0000
  32   1.0000 0.0000
  34   1.0000 0.0000
  36   0.5556 0.4444
  38   0.6250 0.3750
  40   0.3333 0.6667
```

A plot of the proportion of mature fish versus length category (Figure 1) is constructed with the following code

```
> lens <- as.numeric(rownames(ptblLen))
> plot(ptblLen[,"Mature"]~lens,pch=16,xlab="Total Length (cm)",ylab="Proportion Mature")
```

3 Modeling with Raw Data

3.1 Fitting the Logistic Regression Model

Raw maturity data is generally summarized with a logistic regression. A logistic regression is conducted with a binomial response variable and, generally, a quantitative explanatory variable. The logistic regression would like to use typical general linear model theory to model the probability of "success" (p) by the value of the explanatory variable. However, this relationship is generally not linear (primarily due to the constaint that the probability is between 0 and 1). Thus, the logistic regression procedure must transform p to form a linear equation. The required transformation is the so-called *logit* transformation,

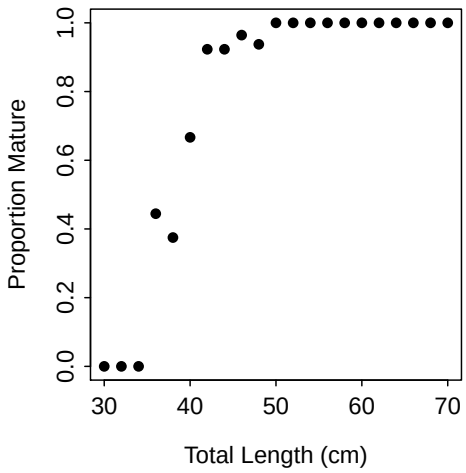

Figure 1. Proportion of female Yelloweye Rockfish that were mature at each 2-cm length category.

$$logit(p) = log\left(\frac{p}{1-p}\right)$$

where $1-p$ is the probability of " failure." In maturity analyses a "success" is defined as "being mature" and a "failure" is defined as "being immature." With this transformation a linear model is formed with

$$logit(p) = \alpha + \beta_1 X$$

or, more visually,

$$log\left(\frac{p}{1-p}\right) = \alpha + \beta_1 X \tag{1}$$

In maturity analyses, the explanatory variable, X, is often length or age such that the probability of being mature is being modelled as a function of length or age.

The logistic regression is fit in R with the general linear model procedure, or `glm()`, with a formula of the form `factor~quantitative` as the first argument followed by the data frame from which to extract the variables in `data=`. The `glm()` function is quite general but it is forced to fit a logistic regression by including the `family=binomial` argument. The results of the model fitting should be saved to an object so that specific information can be extracted from that object. For example, submitting the saved `glm()` object to `coef()` will produce estimates for the intercept and slope of the fitted model. The fit and extraction of parameter estimates is illustrated with

```
> glm1 <- glm(maturity~length,data=YERockfish,family=binomial)
> coef(glm1)
(Intercept)      length
   -16.9483      0.4372
```

Confidence intervals for the parameters of the logistic regression are best estiamted via bootstrapping rather than normal theory. The bootstrapping can be performed most easily with `bootCase()` from the `car` package. This function requires the saved `glm` object as the first argument and the number of bootstrap samples to construct in the `B=` argument. The number of bootstrap samples to take should be in the thousands. The matrix of results from `bootCase()` should be saved to an object. The 95% bootstrap confidence interval for a parameter is the values of the parameter estimate that have 2.5% of bootstrap samples smaller and

larger (i.e., the 2.5% and 97.5% quantiles of the parameter estimates). These values are most easily found by submitting the `bootCase` object to `confint()`. The bootstrapping and confidence interval extraction are illustrated with

```
> bcL <- bootCase(glm1,B=100)  # B should be nearer to 1000
> confint(bcL)
             95% LCI  95% UCI
(Intercept) -25.2732 -12.2391
length        0.3295   0.6423
```

A predicted probability of "success" (i.e., being mature) at a given value of the explanatory variable is computed with

$$p = \frac{e^{\alpha+\beta_1 x}}{1+e^{\alpha+\beta_1}} \tag{2}$$

This prediction can be computed in R with `predict()`, which requires the saved `glm` object as the first argument, a data frame with the values of the explanatory variable for which to make the prediction as the second argument, and `type="response"` (which forces the prediction of the probability of being mature rather than the logit). For example, the predicted probabilities of being mature for female Yelloweye Rockfish that are 32- and 42-cm total length are computed with

```
> predict(glm1,data.frame(length=c(32,42)),type="response")
      1       2
0.04933 0.80428
```

Confidence intervals for the predicted probability cannot be found simply as a prediction must be made for each bootstrap sample and then values for the upper and lower 2.5% must be located. This can be accomplished, however, by creating a function that represents (2) and then applying that function to each row of the matrix containing the bootstrap samples. Applying this function to each row is accomplished with `apply()` using the matrix of bootstrap samples as the first argument, the number "1" as the second argument (denotes that the application is over the first margin of the matrix, or the rows), the prediction function as the third argument, and the value of "x" at which to make the prediction as the last argument. The result of `apply()` should be saved to an object so that the upper and lower 2.5% quantiles can be extracted with `quantile()`. This process is illustrated for the predicted probability of being mature for 32-cm long Yelloweye Rockfish with

```
> predP <- function(cf,x) exp(cf[1]+cf[2]*x)/(1+exp(cf[1]+cf[2]*x))
> p32 <- apply(bcL,1,predP,x=32)
> quantile(p32,c(0.025,0.975))
    2.5%    97.5%
0.007846 0.136294
```

Finally, one may construct a plot (Figure 2) showing the fitted logistic regression line with `fitPlot()` from the `FSA` package. In this plot, the fitted logistic regression is shown by a red line, the individual data are shown as gray-shaded dots where the darker the dot the more individuals plotted at that point, and the proportion that are mature for several categories of the x-axis variable are shown by blue "pluses." The number of points that are plotted on top of each other before the point turns completely black is controlled with the `pt.trans=` argument (larger numbers mean more points are needed to form a black dot). The number of categories of the x-axis at which the proportion of "successes" is calculated is controlled by the `p.ints=` argument (defaults to 25 levels). If `p.ints=NULL` is used then the proportions will be computed at each individual value of the x-axis variable, which may be appropriate for age data but is likely not appropriate for length data. Finally, the range of the x-axis variable can be controlled with the `xlim=` argument. The fitted line plot shown in Figure 2 was constructed with

```
> fitPlot(glm1,xlab="Total Length (cm)",ylab="Proportion Mature",main="",xlim=c(25,70))
```

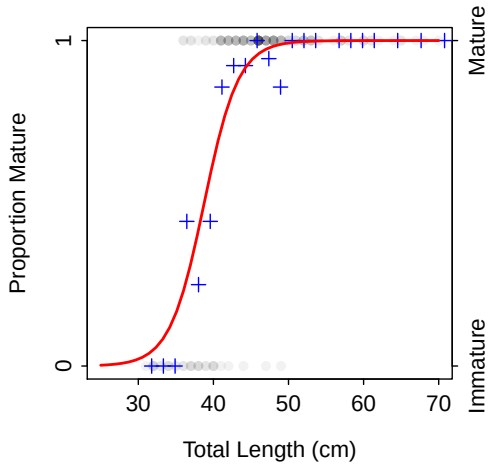

Figure 2. Fitted logistic regression for proportion of female Yelloweye Rockfish mature by total length.

3.2 Length or Age-at-Maturity

A common metric in fisheries science is to find the length or age at which a certain percentage of the fish are mature. For example, it is common to ask "what is the length or age at which 50% of the fish have reached maturity?" A general formula for computing this metric is found by solving (1) for X. This can be fairly quickly shown to be

$$x = \frac{log\left(\frac{p}{1-p}\right) - \alpha}{\beta_1} \tag{3}$$

In the common case of finding X for 50% maturity (i.e., $p = 0.5$), (3) reduces to the following simple formula

$$x = -\frac{\alpha}{\beta_1} \tag{4}$$

The age at which 50% of the fish are mature would be commonly symbolized as A_{50}. Similarly, the length at which 90% of the fish are mature would be L_{50}.

These calculations are simplified by creating a function to perform (3) as follows

```
> lrPerc <- function(cf,p) (log(p/(1-p))-cf[1])/cf[2]
```

Thus, the lengths at which 50% and 90% of the female Yelloweye Rockfish are mature is computed with

```
> ( L50 <- lrPerc(coef(glm1),0.5) )
(Intercept)
      38.77
> ( L90 <- lrPerc(coef(glm1),0.9) )
(Intercept)
      43.79
```

Confidence intervals for these values must also be constructed from the bootstrap sample results, as follows

```
> bL50 <- apply(bcL,1,lrPerc,p=0.5)
> ( L50ci <- quantile(bL50,c(0.025,0.975)) )
 2.5% 97.5%
37.21 40.17
> bL90 <- apply(bcL,1,lrPerc,p=0.9)
> ( L90ci <- quantile(bL90,c(0.025,0.975)) )
 2.5% 97.5%
 41.7  45.9
```

A fitted line plot with the L_{50} illustrated is constructed as follows

```
> fitPlot(glm1,xlab="Total Length (cm)",ylab="Proportion Mature",main="",xlim=c(25,70),
        plot.p=FALSE)
> lines(c(0,L50),c(0.5,0.5),lty=3,lwd=2,col="blue")
> lines(c(L50,L50),c(-0.2,0.5),lty=3,lwd=2,col="blue")
```

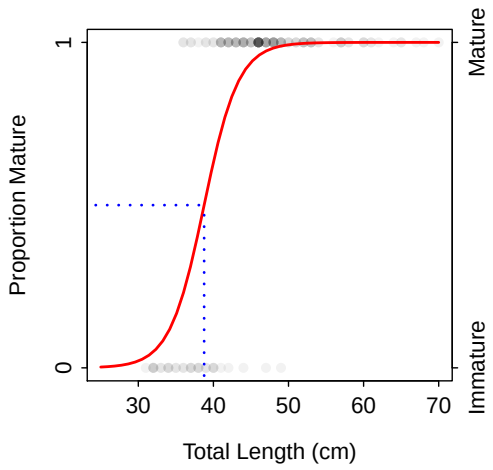

Figure 3. Fitted logistic regression for proportion of female Yelloweye Rockfish mature by total length with L_{50} shown.

4 Modeling with Summarized Data

It is also common to have the data in grouped or summarized format, where the proportion of "successes" and "trials" for each value of the explanatory variable are provided. For example, part of the summarized Yelloweye Rockfish data set (in a data frame called `YER2`) would look like that shown below

```
> YER2[1:10,]
   length pmat n
31     31 0.00 1
32     32 0.00 4
33     33 0.00 2
34     34 0.00 3
```

```
35     35 0.00 2
36     36 0.50 4
37     37 0.40 5
38     38 0.25 4
39     39 0.50 4
40     40 0.40 5
```

The appropriate logistic regression model can again be fit with `glm()` with the exception that the right-hand-side of the formula is the proportion of "successes" variable and a `weights=` argument should be set equal to the name of the sample size variable. This model is fit and the coefficients are extracted with the code below where it can be seen that the results are the same as those found with the raw data.

```
> glm2 <- glm(pmat~length,data=YER2,family=binomial,weights=n)
> coef(glm2)
(Intercept)      length
   -16.9483      0.4372
```

All of the same post model fit functions can be used on this result as shown below.

```
> bcL2 <- bootCase(glm2,B=100)
> confint(bcL2)
             95% LCI 95% UCI
(Intercept) -26.0311 -12.184
length        0.3174   0.667
> predict(glm1,data.frame(length=c(32,42)),type="response")
      1       2
0.04933 0.80428
> p32a <- apply(bcL2,1,predP,x=32)
> quantile(p32a,c(0.025,0.975))
    2.5%    97.5%
0.007408 0.114054
```

Finally, the fitted plot (Figure 4) can be constructed similarly though it is suggested that the plotting of the proportion of "successess" for several categories of the explanatory variable (i.e., the "blue plussess" in Figure 2) be "shut off" given that the raw data in this case is exactly that.

```
> fitPlot(glm2,xlab="Total Length (cm)",ylab="Proportion Mature",main="",xlim=c(25,70),
          plot.p=FALSE)
```

5 Comparing Logistic Regressions Between Groups

It may be interesting to determine if the fit of the logistic regression differs between two groups. For example, with the Yelloweye Rockfish example, there may be a reason (e.g., a management change) to determine if the logistic regression differs between "pre-2002" and "post-2002." To construct this analysis a factor variable that represents the two groups needs to be computed. The code to construct this factor shown below is more complicated than I would like because the original "date" variable was a factor.

```
> YERockfish$year <- as.numeric(format(as.POSIXct(YERockfish$date,format="%m/%d/%Y"),"%Y"))
> YERockfish$pre02 <- "Yes"
> YERockfish$pre02[YERockfish$year<2003] <- "No"
> YERockfish$pre02 <- factor(YERockfish$pre02)
```

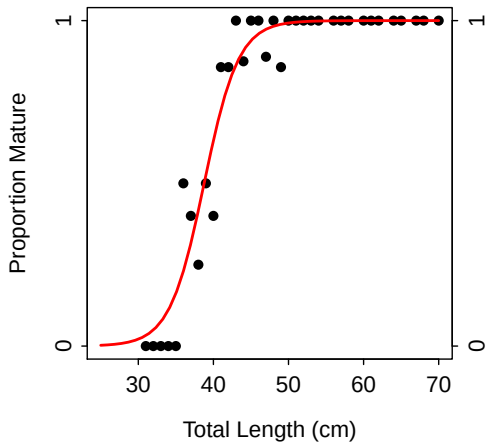

Figure 4. Fitted logistic regression for proportion of female Yelloweye Rockfish mature by total length (computed from the summarized data frame).

The model required in this case is similar to what was done previously except that the left-hand-side of the formula should contain the "covariate" "multiplied" by the factor variable that indicates the groups. With this construct, the `glm()` will fit the "covariate" "main" effect, the factor "main" effect, and the interaction between the "covariate" and the factor. In this case, the model is fith with

```
> glm3 <- glm(maturity~length*pre02,data=YERockfish,family=binomial)
```

Differences in the slope among the groups is determined by determining if the interaction variable is a significant term in the model. If the interaction variable is insignificant (i.e., the lines are parallel) then difference in intercepts among the groups is determined by determining if the factor variable is a significant term in the model. The significance of these terms is obtained from the `glm()` result with `drop1()` using the `glm()` object as the first argument, a formula of the form `~.` as the second term, and using the `test="Chisq"` argument. For example, the p-values for determining the signficance of these terms for the Yelloweye Rockfish data are obtained with

```
> drop1(glm3,~.,test="Chisq")
Single term deletions

Model:
maturity ~ length * pre02
            Df Deviance   AIC  LRT Pr(>Chi)
<none>           71.4  79.4
length       1    135.3 141.3 63.9  1.3e-15
pre02        1     71.9  77.9  0.5     0.49
length:pre02 1     72.1  78.1  0.7     0.41
```

From this it appears that the logistic regression models are parallel ($p = 0.4115$) with equal intercepts ($p = 0.4863$), which indicates that there is no significant difference in the logistic regressions between the pre- and post-2002 periods.

```
> ndf <- data.frame(length=c(32,42,32,42),pre02=c("Yes","Yes","No","No"))
> predict(glm3,ndf,type="response")
      1       2       3       4
0.05299 0.69749 0.03539 0.86945
```

```
> bcL3 <- bootCase(glm3,B=100)
> confint(bcL3)
                95% LCI  95% UCI
(Intercept)   -48.0124 -11.0865
length          0.2940   1.2003
pre021        -14.9954  21.3450
length:pre021  -0.4979   0.3875
```

Constructing a fitted line plot with the separate logistic regression models is a bit more work. First, extract the coefficients from the fitted line, noticing that the "Pre02Yes" coefficient is the difference in intercepts between the two groups and the "length:pre02Yes" coefficient is the difference in slopes between the two groups

```
> (cf <- coef(glm3))
  (Intercept)        length        pre021 length:pre021
    -17.36633       0.44600      -2.58366       0.07414
```

Second, create a vector of total length values over which to evaluate the two models. Then predict the probability of being mature for the "pre-02" and "post-02" groups.

```
> x <- seq(30,70,0.1)
> ppre <- exp(cf[1]+cf[2]*x)/(1+exp(cf[1]+cf[2]*x))
> ppost <- exp(cf[1]+cf[3]+(cf[2]+cf[4])*x)/(1+exp(cf[1]+cf[3]+(cf[2]+cf[4])*x))
```

Finally, plot the "pre-02" relationship before including the "post-02" relationship and a legend.

```
> plot(ppre~x,type="l",lwd=2,xlab="Total Length (cm)",ylab="Proportion Mature")
> lines(ppost~x,type="l",lwd=2,col="red")
> legend("bottomright",c("Pre-2002","Post-2002"),lwd=2,col=c("black","red"),bty="n")
```

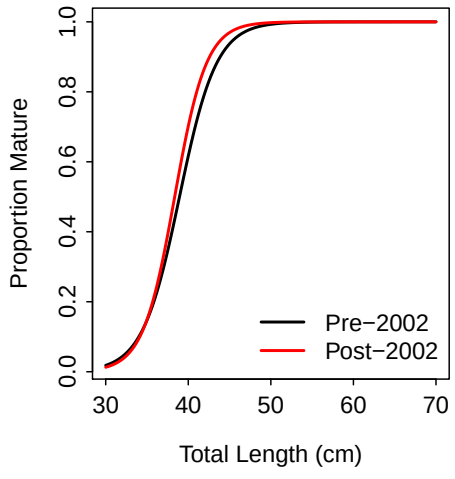

Figure 5. Fitted logistic regression for proportion of female Yelloweye Rockfish mature by total length separated by the "pre-2002" and "post-2002" periods.

XXX Need more work to show how to predict with CIs for the two groups XXX

XXX Also need an example with some differences XXX

Reproducibility Information

Version Information

- **Compiled Date:** Mon Dec 16 2013
- **Compiled Time:** 9:56:32 PM
- **Code Execution Time:** 4.46 s

R Information

- **R Version:** R version 3.0.2 (2013-09-25)
- **System:** Windows, i386-w64-mingw32/i386 (32-bit)
- **Base Packages:** base, datasets, graphics, grDevices, methods, stats, utils
- **Other Packages:** car_2.0-19, FSA_0.4.3, FSAdata_0.1.4, gdata_2.13.2, knitr_1.5.15
- **Loaded-Only Packages:** bitops_1.0-6, caTools_1.16, cluster_1.14.4, evaluate_0.5.1, formatR_0.10, Formula_1.1-1, gplots_2.12.1, grid_3.0.2, gtools_3.1.1, highr_0.3, Hmisc_3.13-0, KernSmooth_2.23-10, lattice_0.20-24, MASS_7.3-29, multcomp_1.3-1, mvtnorm_0.9-9996, nlme_3.1-113, nnet_7.3-7, plotrix_3.5-2, quantreg_5.05, sandwich_2.3-0, sciplot_1.1-0, SparseM_1.03, splines_3.0.2, stringr_0.6.2, survival_2.37-4, tools_3.0.2, zoo_1.7-10
- **Required Packages:** FSA, FSAdata, car and their dependencies (gdata, gplots, graphics, Hmisc, knitr, MASS, multcomp, nlme, nnet, plotrix, quantreg, sciplot, stats)