AIFFD Chapter 6 - Mortality

Derek H. Ogle

Contents

This document contains R versions of the boxed examples from **Chapter 6** of the "Analysis and Interpretation of Freshwater Fisheries Data" book. Some sections build on descriptions from previous sections, so each section may not stand completely on its own. More thorough discussions of the following items are available in linked vignettes:

- the use of linear models in R in the preliminaries vignette,
- differences between and the use of type-I, II, and III sums-of-squares in the preliminaries vignette, and
- the use of "least-squares means" is found in the preliminaries vignette.

The following additional packages are required to complete all of the examples (with the required functions noted as a comment and also noted in the specific examples below).

```
> library(FSA)        # Summarize, Subset, binCI, rcumsum, catchCurve, chapmanRobson, mrOpen
> library(car)        # Anova
```

In addition, external tab-delimited text files are used to hold the data required for each example. These data are loaded into R in each example with `read.table()`. Before using `read.table()` the working directory of R must be set to where these files are located on **your** computer. The working directory for all data files on **my** computer is set with

```
> setwd("C:/aaaWork/Web/fishR/BookVignettes/aiffd2007")
```

In addition, I prefer to not show significance stars for hypothesis test output, reduce the margins on plots, alter the axis label positions, and reduce the axis tick length. In addition, contrasts are set in such a manner as to force R output to match SAS output for linear model summaries. All of these options are set with

```
> options(width=90,continue=" ",show.signif.stars=FALSE,
          contrasts=c("contr.sum","contr.poly"))
> par(mar=c(3.5,3.5,1,1),mgp=c(2.1,0.4,0),tcl=-0.2)
```

6.1 Basic Mortality Computations

The calculations in Box 6.1 do not rely on SAS- or R-specific code. However, I will reproduce the calculations below to illustrate how R can be used as a calculator.

Take, for example, a hypothetical fish population consisting of a single age-group. At the start of a 12-month interval, the age-group consists of 1000 individuals, and at the end it has been reduced by mortality to 700. For this example,

```
> N0 <- 1000                          # Initial population
> N12 <- 700                          # Final population
> ( d <- N0-N12 )                     # Interval absolute mortality (i.e., number of deaths)
```

```
[1] 300
```

```
> A12 <- d/N0                         # Interval (12 month) mortality rate
> 100*A12
```

```
[1] 30
```

```
> ( Z12 <- -log(1-A12) )              # Instantaneous mortality rate
```

```
[1] 0.3566749
```

Now, suppose we wish to know the fraction of the population remaining, the number of individuals, and the number of deaths at the end of 4 and 8 months intervals. For this, Z_{12} must be partitioned into 4-month (Z_4) and 8-month (Z_8) estimates as

```
> Z1 <- Z12/12                        # Monthly instantaneous mortality rate
> ( Z4 <- 4*Z1 )                      # Four-month Z estimate
```

```
[1] 0.1188916
```

```
> ( Z8 <- 8*Z1 )                      # Eight-month Z estimate
```

```
[1] 0.2377833
```

Interval mortality rates during the 4-month (A_4) and 8-month (A_8) intervals are then calculated as

```
> ( A4 <- 1-exp(-Z4) )                # Interval (4-month) mortality rate
```

```
[1] 0.112096
```

```
> ( A8 <- 1-exp(-Z8) )              # Interval (8-month) mortality rate
```

```
[1] 0.2116265
```

Note, below, the use of `round()` to round the results to a specified number of decimal places. This function requires two arguments – the value(s) to be rounded as the first argument and the number of decimal places as the second argument.

Numbers of individuals remaining (survival) after 4 months (N_4) and 8 months (N_8) are represented by

```
> d4 <- N0*A4                       # Deaths in first four-months
> round(d4,0)
```

```
[1] 112
```

```
> N4 <- N0-d4                       # Estimated survivors after 4-months
> round(N4,0)
```

```
[1] 888
```

```
> d8 <- N0*A8                       # Deaths in first eight-months
> round(d8,0)
```

```
[1] 212
```

```
> N8 <- N0-d8                       # Estimated survivors after 4-months
> round(N8,0)
```

```
[1] 788
```

```
> d8.4 <- d8-d4                     # Deaths in second four-months
> round(d8.4,0)
```

```
[1] 100
```

```
> d12.8 <- d-d8                     # Deaths in third four-months
> round(d12.8,0)
```

```
[1] 88
```

The code below is a check of our work.

```
> d4+d8.4+d12.8                     # Total deaths in the three four-month periods
```

```
[1] 300
```

```
> (d4+d8.4+d12.8) == d              # Does this equal the first deaths calculation?
```

```
[1] TRUE
```

6.2 Heincke's Method of Estimating Annual Mortality

From a reservoir, a large random sample of Spotted Bass (*Micropterus punctulatus*) was collected with electrofishing gear, and fish age was determined by inspecting otoliths. The number of fish in each age-class is given in the table in the text. There was some disagreement over the ages of the four largest and oldest fish, but they were all at least 7 years of age, so the data were coded accordingly. The low catch of age-1 fish relative to age-2 fish suggested that age-1 fish were not fully recruited to the electrofishing gear.

6.2.1 Preparing Data

The data for this box is very simple – number of fish in seven age-classes. Because of this simplicity the data can simply be entered into R using `c()` (for concatenation). The data could have been entered into an external file and read into R (I demonstrate this in Box 6.3). The data for this box are entered into two vectors below. Note that the colon on `1:7` constructs a sequence of all integers between 1 and 7 inclusive and that `c()` concatenates or combines the numbers in parentheses into a single vector.

```
> age <- 1:7
> catch <- c(257,407,147,32,17,5,4)
> rbind(age,catch)                          # for display purposes only.
```

```
      [,1] [,2] [,3] [,4] [,5] [,6] [,7]
age      1    2    3    4    5    6    7
catch  257  407  147   32   17    5    4
```

6.2.2 Heincke's Method

The author's of the box limited the first calculation to only those fish that were age-2 or older. With this constraint the annual mortality calculation requires the total number of fish that were age-2 and older. In this example, the catch of fish age-2 and older can be found in positions two through seven of the `catch` vector. Certain positions of a vector can be obtained by appending the desired positions within square brackets immediately after the name of the vector. Asking for certain positions of the `catch` vector are illustrated below.

```
> catch[2]                          # 2nd position
```

```
[1] 407
```

```
> catch[c(2,4)]                     # 2nd & 4th positions
```

```
[1] 407  32
```

```
> catch[2:4]                        # 2nd through 4th positions
```

```
[1] 407 147  32
```

```
> catch[-2]                          # all but the 2nd position
```

```
[1] 257 147  32  17   5   4
```

The total catch of age-2 and older fish is isolated below and submitted to `sum()` to find the total catch for ages 2 to 7. The annual mortality rate is then the number of age-2 fish divided by this total (i.e., the proportion of all fish age-2 and older that were age-2). The standard error for the estimate of A is computed given the formula in the text.

```
> ( N <- sum(catch[2:7]) )           # sum in positions 2 thru 7
```

```
[1] 612
```

```
> ( A <- 100*catch[2]/N )            # annual mortality estimate
```

```
[1] 66.50327
```

```
> ( se.A <- sqrt(A*(100-A)/N) )      # SE of A
```

```
[1] 1.907862
```

The Heincke method essentially computes the proportion of a certain age out of a total number in an age range. This is the same as computing a binomial proportion – in fact the SE computed above is based on binomial theory. A confidence interval for a binary proportion can be computed with `binCI()` from the `FSA` package. This function requires the number of "successes" (i.e., the age in question) as the first argument and the total number of trials (i.e., all fish in the age categories) as the second argument.

```
> 100*binCI(catch[2],N)              # default 95% CI for A
```

```
  95% LCI   95% UCI
 62.67124 70.12941
```

6.2.3 Heincke's Method (Alternative)

The authors of the box mention that the estimated annual mortality rate, using the method above, for age-3 and older fish is 72%. This begs the question of whether there is an efficient method of computing the estimated annual mortality rate for each age (and older). Fortunately, this can be accomplished fairly efficiently with `rcumsum()` from the `FSA` package. This function computes the reverse cumulative summation for a vector. In other words, it will compute the total number of fish age-1 and older, then the total number of fish age-2 and older, then age-3 and older, etc. The reverse cumulative sums can be computed and saved to an object so that the estimated annual mortality rates can be calculated by dividing the `catch` vector by these reverse cumulative sums vector and the CIs can be computed quickly with `binCI`.

```
> ( N <- rcumsum(catch) )
```

```
[1] 869 612 205  58  26   9   4
```

```
> 100*catch/N
```

```
[1]  29.57422  66.50327  71.70732  55.17241  65.38462  55.55556 100.00000
```

```
> 100*binCI(catch,N)
```

```
  95% LCI   95% UCI
 26.63515  32.69308
 62.67124  70.12941
 65.18616  77.42990
 42.45208  67.25015
 46.22057  80.58777
 26.66513  81.12215
 51.01092 100.00000
```

```
> data.frame(age,catch,N,A=100*catch/N,100*binCI(catch,N))  # combine for display only
```

```
  age catch   N         A X95..LCI  X95..UCI
1   1   257 869  29.57422 26.63515  32.69308
2   2   407 612  66.50327 62.67124  70.12941
3   3   147 205  71.70732 65.18616  77.42990
4   4    32  58  55.17241 42.45208  67.25015
5   5    17  26  65.38462 46.22057  80.58777
6   6     5   9  55.55556 26.66513  81.12215
7   7     4   4 100.00000 51.01092 100.00000
```

6.3 Robson and Chapman's Maximum-Likelihood Estimation of Survival

Assume that all the fish in a large sample were aged and the numbers of fish in each age-class were tallied, as shown in the table in the text. Along with constant (or near constant) recruitment and survival rates, the assumption of equal vulnerability to capture must be met. A cursory examination of the catch-at-age data suggests that the two youngest age-groups were not fully vulnerable, or recruited, to the gear (i.e., the curve does not truly begin to descend until age-3); therefore, the analysis will apply to only age-3 and older fish. The first step is to code each age, starting with zero for the youngest age considered fully recruited (i.e., age-3).

6.3.1 Preparing Data

The Box6_3.txt data file is read and observed below. Note that these data are so simple that they could have been entered directly into vectors in R as illustrated in Box 6.2.

```
> ( d3 <- read.table("data/Box6_3.txt",header=TRUE) )
```

```
  age catch
1   1    90
2   2   164
3   3   162
4   4   110
5   5    55
```

```
6    6    41
7    7    20
8    8    14
9    9     7
10  10     5
```

6.3.2 Chapman-Robson's Method

The maximum-likelihood method for estimating the rate of survival proposed by Chapman and Robson (1960) and Robson and Chapman (1961) is implemented with `chapmanRobson()` from the `FSA` package. This function requires an R formula of the form `catches~ages`, where `catches` generically represents the vector of catch numbers and `ages` generically represents the vector of ages, as the first argument, the data frame from which those vectors are drawn as the second (or `data=`) argument and a vector of the fully recruited ages as the third (or `ages2use=`) argument. The ages of fully recruited fish can be identified with a full range (e.g., `3:10`) or a concatenated list of ages (e.g., `c(3,4,5,6,7,8,9,10)`). The best estimates (with SEs) of the annual survival S and instantaneous mortality Z rates are extracted from the saved `chapmanRobson` object with `summary()`, the 95% confidence intervals for the two parameters are extracted with `confint()`, and a plot depicting the Chapman-Robson calculations is obtained with `plot()` (Note that the results shown here and those shown in the text differ slightly because more precision is used here on intermediate calculations).

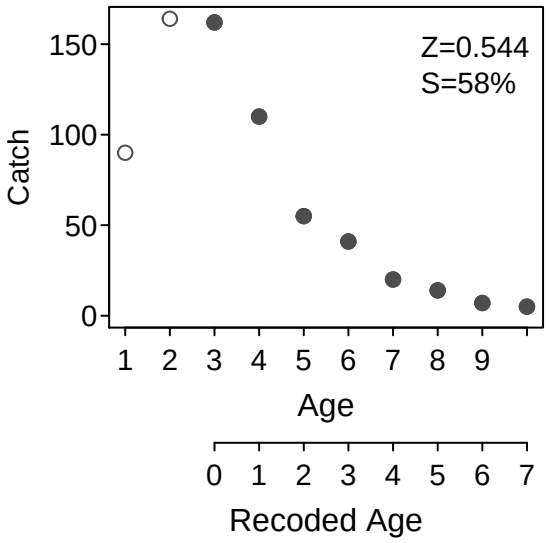
```
> cr1 <- chapmanRobson(d3$age,d3$catch,3:10)
> summary(cr1)
```

```
    Estimate Std. Error
S 57.9857579 1.57508209
Z  0.5442405 0.02468115
```

```
> confint(cr1)
```

```
     95% LCI    95% UCI
S 54.8986537 61.0728620
Z  0.4958664  0.5926147
```

```
> plot(cr1)
```



6.4 Mortality Rates from the Slope of Regression Line

The catch-at-age data shown in Box 6.3 and Figure 6.2 (in the text) is used again here. The catch-curve analysis is limited to those ages considered fully recruited to the gear (age-3 and older). At least five fish in the oldest age-class are present, so the mortality rate will apply to ages 3-10. Using least-squares regression, the slope of the line describing the relation between the natural log of number (y-variable) and age (x-variable) can be calculated longhand, by means of a spreadsheet, or with the R code below.

6.4.1 Preparing Data

The same data used in Box 6.3 are used here with the exception that a variable that is the natural log of the `catch` is added to the data frame.

```
> d3$logcatch <- log(d3$catch)
> str(d3)
```

```
'data.frame':   10 obs. of  3 variables:
 $ age     : int  1 2 3 4 5 6 7 8 9 10
 $ catch   : int  90 164 162 110 55 41 20 14 7 5
 $ logcatch: num  4.5 5.1 5.09 4.7 4.01 ...
```

6.4.2 The Catch-Curve Method – First Principles

6.4.2.1 Unweighted Regression Linear regressions in R are performed with `lm()` where the first argument is a formula of the form `y~x` and the second (`data=`) argument is the data frame in which the variables can be found. An optional third (`subset=`) argument can be used to efficiently create a subset of a data frame to be used for that particular linear model. This argument is particularly important in catch-curve analyses because it can be used to limit the analysis to only those ages that appear to be fully recruited to the gear. In this example, the authors have determined that all fish age-3 and older are fully recruited and should be used to estimate the mortality rate. The ANOVA table is extracted from the saved `lm` object with `anova()`, the parameter estimates are extracted with `summary()`, and default 95% CIs for the parameters are extracted with `confint()`.

```
> cc1 <- lm(logcatch~age,data=d3,subset=age>=3)
> anova(cc1)
```

```
Analysis of Variance Table

Response: logcatch
          Df  Sum Sq Mean Sq F value    Pr(>F)
age        1 10.9766 10.9766  1072.5 5.391e-08
Residuals  6  0.0614  0.0102
```

```
> summary(cc1)
```

```
Call:
lm(formula = logcatch ~ age, data = d3, subset = age >= 3)

Residuals:
     Min       1Q   Median       3Q      Max
```

```
-0.11342 -0.08876  0.01113  0.07263  0.12057

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  6.66033    0.10758   61.91 1.19e-09
age         -0.51122    0.01561  -32.75 5.39e-08

Residual standard error: 0.1012 on 6 degrees of freedom
Multiple R-squared:  0.9944,    Adjusted R-squared:  0.9935
F-statistic:  1073 on 1 and 6 DF,  p-value: 5.391e-08
```

```
> confint(cc1)
```

```
                2.5 %     97.5 %
(Intercept)  6.3970827  6.9235800
age         -0.5494179 -0.4730256
```

As noted in the box, the estimate of the instantaneous mortality rate (Z) is obtained from the estimated slope. The parameters are isolated from the saved `lm` object with `coef()`, and, because the slope is the second value in the returned vector it can be isolated by appending an `[2]`. The confidence interval for the slope is in the second row of the matrix returned by `confint()`; thus, the CI for the slope can be isolated by appending an `[2,]`. Note the use of the negative sign to convert the negative slope into a positive Z value.

```
> ( Z <- -coef(cc1)[2] )        # 2nd coefficient of model results
```

```
      age
0.5112218
```

```
> ( ZCI <- -confint(cc1)[2,] )  # 2nd row of confint results
```

```
    2.5 %    97.5 %
0.5494179 0.4730256
```

The isolated instantaneous mortality results from above are used to effectively find estimates for the annual mortality rate (A) using the $A = 1 - e^{-Z}$ formula.

```
> ( A <- 100*(1-exp(-Z)) )
```

```
     age
40.02376
```

```
> ( ACI <- 100*(1-exp(-ZCI)) )
```

```
   2.5 %   97.5 %
42.27142 37.68859
```

6.4.2.2 Weighted Regression As noted in the box the predictions from the unweighted regression fit above can be stored and used as weights in the catch-curve analysis to minimize the effect of few fish in the older age classes on the regression results. Making these predictions is simple but using them is slightly complicated because of the subsetting required for the catch-curve regression. The easiest way to handle this slight complexity is to predict log catches for all ages, attach these to the original data frame, and then use the same subsetting routine. Predictions are made with `predict()` which requires the fitted model as the first argument and a data frame that has a variable that is named exactly as the variable in the model as the second argument. Thus, the predictions for all ages in the original data frame is made and appended to the original data frame as shown below.

```
> d3$W <- predict(cc1,d3)
> d3
```

```
   age catch logcatch        W
1    1    90 4.499810 6.149110
2    2   164 5.099866 5.637888
3    3   162 5.087596 5.126666
4    4   110 4.700480 4.615444
5    5    55 4.007333 4.104223
6    6    41 3.713572 3.593001
7    7    20 2.995732 3.081779
8    8    14 2.639057 2.570557
9    9     7 1.945910 2.059336
10  10     5 1.609438 1.548114
```

The second weighted regression is then fit with `lm()` exactly as before except that the weights just created are identified in the `weights=` argument.,

```
> cc2 <- lm(logcatch~age,data=d3,subset=age>=3,weights=W)
> anova(cc1)
```

```
Analysis of Variance Table

Response: logcatch
          Df  Sum Sq Mean Sq F value    Pr(>F)
age        1 10.9766 10.9766  1072.5 5.391e-08
Residuals  6  0.0614  0.0102
```

```
> summary(cc2)
```

```
Call:
lm(formula = logcatch ~ age, data = d3, subset = age >= 3, weights = W)

Weighted Residuals:
      Min        1Q    Median        3Q       Max
-0.196521 -0.153502 -0.006143  0.128350  0.228641

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  6.66128    0.10002   66.60 7.71e-10
age         -0.51139    0.01643  -31.12 7.31e-08
```

```
Residual standard error: 0.1822 on 6 degrees of freedom
Multiple R-squared:  0.9938,    Adjusted R-squared:  0.9928
F-statistic: 968.3 on 1 and 6 DF,  p-value: 7.314e-08
```

```
> ( Z <- -coef(cc2)[2] )
```

```
      age
0.5113879
```

```
> ( ZCI <- -confint(cc2)[2,] )
```

```
    2.5 %    97.5 %
0.5515999 0.4711759
```

6.4.3 The Catch-Curve Method – Simpler Function

The methods described above are implemented with `catchCurve()` from the `FSA` package. This function requires the same first three arguments as described for `chapmanRobson()` in Box 6.3. Note that the original, not logged, catches are given in this function. In addition, the weighted regression can be performed by including `use.weights=TRUE`. Specific information can be extracted from the results in the `catchCurve()` object as described for `chapmanRobson()` in Box 6.3.

```
> cc3 <- catchCurve(catch~age,d3,3:10)                    # unweighted catch-curve
> anova(cc3)
```

```
Analysis of Variance Table

Response: log.catch.e
          Df  Sum Sq Mean Sq F value    Pr(>F)
age.e      1 10.9766 10.9766  1072.5 5.391e-08
Residuals  6  0.0614  0.0102
```

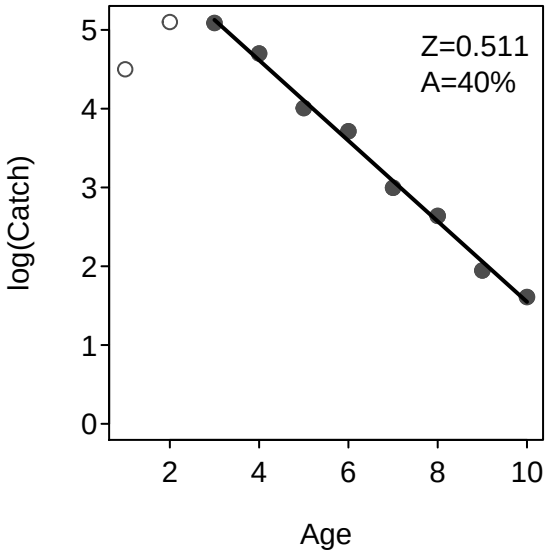
```
> summary(cc3)
```

```
    Estimate Std. Error  t value   Pr(>|t|)
Z  0.5112218 0.01560993 32.74978 5.3913e-08
A 40.0237632         NA       NA         NA
```

```
> confint(cc3)
```

```
     95% LCI    95% UCI
Z  0.4730256  0.5494179
A 37.6885900 42.2714237
```

```
> plot(cc3)
```



The weighted regression is fit as shown below.

```
> cc4 <- catchCurve(catch~age,d3,3:10,use.weights=TRUE)    # weighted catch-curve
> summary(cc4)
```

```
    Estimate Std. Error  t value   Pr(>|t|)
Z  0.5112218 0.01560993 32.74978 5.3913e-08
A 40.0237632         NA       NA         NA
```

```
> confint(cc4)
```

```
     95% LCI    95% UCI
Z  0.4730256  0.5494179
A 37.6885900 42.2714237
```

6.5 Adjusting Catch-at-Age Data for Unequal Recruitment

Trap-netting of White Crappies (*Pomoxis annularis*) in a Midwestern reservoir in the spring of 2001 provided information on the standing age structure for the 281 fish collected representing six age-classes. Previous studies indicated that all ages were recruited to the trap nets, so the goal was to calculate instantaneous mortality between ages 1 and 6. The catch-at-age data clearly indicated that recruitment varied widely among years, and therefore one of the assumptions of catch-curve analysis was violated. It was assumed that the trap-net catches accurately indexed recruitment variability, and these data were used to create an index of year-class strength (l_i) for each i year as $l_i = \frac{r_i}{r_L}$, where r_i = number of age-0 fish collected with trap nets in year i, and r_L = lowest number of age-0 fish collected during the time series. The index was used to adjust the representation of each year-class N_i in spring 2001 to a constant recruitment as $\frac{N_i}{l_i}$.

6.5.1 Preparing Data

The Box6_5.txt data file is read and the structure is observed below. The authors also added several variables to the original data frame. First, age is computed by subtracting the fish's `year.class` from the capture year (2001). As suggested in the box, an index of year-class strength is created by dividing the annual capture

of age-0 fish (i.e., `age0`) by the minimum annual capture of age-0 fish in all years (as found with `min()`). The catch of fish in each age-class is then adjusted by dividing the observed trap-net catch by the index of year-class strength.

```
> d5 <- read.table("data/Box6_5.txt",header=TRUE)
> str(d5)
```

```
'data.frame':   6 obs. of  3 variables:
 $ year.class: int  2000 1999 1998 1997 1996 1995
 $ catch     : int  150 28 5 69 12 17
 $ age0      : int  1665 556 111 2330 445 1220
```

```
> d5$age <- 2001-d5$year.class          # find age
> d5$ycs <- d5$age0/min(d5$age0)        # create year-class strength
> d5$adjcatch <- d5$catch/d5$ycs        # adjust catch for year-class strength
> d5                                    # now, what does the data frame look like
```

```
  year.class catch age0 age       ycs  adjcatch
1       2000   150 1665   1 15.000000 10.000000
2       1999    28  556   2  5.009009  5.589928
3       1998     5  111   3  1.000000  5.000000
4       1997    69 2330   4 20.990991  3.287124
5       1996    12  445   5  4.009009  2.993258
6       1995    17 1220   6 10.990991  1.546721
```

6.5.2 Fitting Catch-Curve Method

The table of the data (shown in the text and above) shows that all adjusted catches appear to be on the descending limb of the catch-curve and, thus, should be used in the regression to estimate the instantaneous mortality rate. The catch-curve model is fit with `catchCurve()` as described in BOX 6.4, but noting that leaving the third argument to `catchCurve()` blank tells R to use all ages in the analysis (These results are very slightly different from what is shown in Box 6.5 in the text. This is likely due to rounding but it is unclear if this is the reason as enough computational detail is not shown in Box 6.5 in the text).

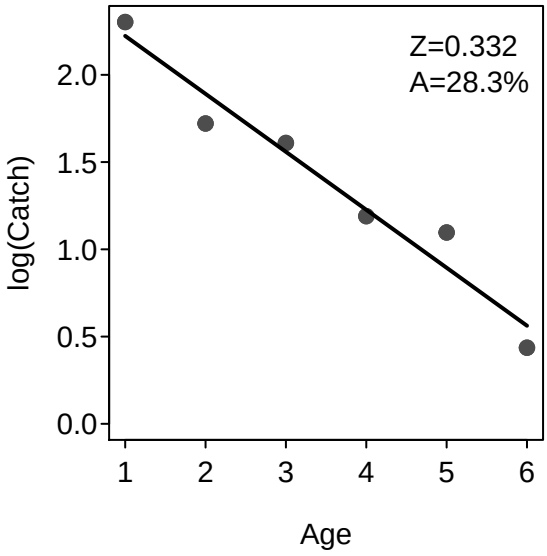
```
> cc1 <- catchCurve(adjcatch~age,data=d5)
> summary(cc1)
```

```
    Estimate Std. Error  t value     Pr(>|t|)
Z  0.3321564 0.03699358 8.978758 0.0008515291
A 28.2624898         NA       NA           NA
```

```
> confint(cc1)
```

```
     95% LCI    95% UCI
Z  0.2294458  0.4348671
A 20.5025925 35.2649284
```

```
> plot(cc1)
```



6.5.3 Ogle Comment

It makes more sense to me to compute a so-called *year-class correction factor* by dividing the maximum observed catch by each of the individual catches-at-age (as found with `max()`). These values then represent how many times smaller that year-class is then the maximum observed year-class. If each catch is then multiplied by this value, it inflates the smaller year-classes "up to" the size of the largest year-class.

```
> d5$yccf <- max(d5$age0)/d5$age0
> d5$adjcatch2 <- d5$catch*d5$yccf
> d5                                    # now, what does the data frame look like
```

```
  year.class catch age0 age       ycs  adjcatch      yccf adjcatch2
1       2000   150 1665   1 15.000000 10.000000  1.399399 209.90991
2       1999    28  556   2  5.009009  5.589928  4.190647 117.33813
3       1998     5  111   3  1.000000  5.000000 20.990991 104.95495
4       1997    69 2330   4 20.990991  3.287124  1.000000  69.00000
5       1996    12  445   5  4.009009  2.993258  5.235955  62.83146
6       1995    17 1220   6 10.990991  1.546721  1.909836  32.46721
```

The catch-curve model is then fit exactly as above (but with the new adjusted catch). A comparison of these results with the results from above shows that it does not make any difference which "adjustment" method you use. In fact, "re-scaling" to any constant value – minimum catch, maximum catch, mean catch, or 100 – will result in the same estimated instantaneous mortality rate because the slope of the regression is invariant to linear transformations of the data (all of these "scales" only differ by a constant ratio).

```
> cc2 <- catchCurve(adjcatch2~age,data=d5)
> summary(cc2)
```

```
    Estimate Std. Error  t value     Pr(>|t|)
Z  0.3321564 0.03699358 8.978758 0.0008515291
A 28.2624898         NA       NA           NA
```

As a final example of invariance property, examine the results below from scaling to the mean catch (as found with `mean()`).

```
> d5$yccf2 <- mean(d5$age0)/d5$age0
> d5$adjcatch3 <- d5$catch*d5$yccf2
> cc3 <- catchCurve(adjcatch3~age,data=d5)
> summary(cc3)
```

```
    Estimate Std. Error  t value     Pr(>|t|)
Z  0.3321564 0.03699358 8.978758 0.0008515291
A 28.2624898         NA       NA           NA
```

6.6 Comparing Instantaneous Mortality Rates from Catch Curves

Comparing instantaneous mortality rates (Z) for two or more populations is equivalent to comparing the slopes of the catch-curve regression lines. In the text are catch-at-age data for two populations that fully recruited to the gear at age-2. The R code to calculate and compare the slopes of the catch-curve regression lines is given below.

6.6.1 Preparing Data

The Box6_6.txt data file is read and the structure is observed below. The natural log of the catches, which is required for the catch curve regression, is appended to the data frame.

```
> d6 <- read.table("data/Box6_6.txt",header=TRUE)
> str(d6)
```

```
'data.frame':   17 obs. of  3 variables:
 $ age  : int  1 2 3 4 5 6 7 8 9 1 ...
 $ catch: int  433 818 243 67 48 5 30 42 22 305 ...
 $ lake : Factor w/ 2 levels "A","B": 1 1 1 1 1 1 1 1 1 2 ...
```

```
> d6$logcatch <- log(d6$catch)
```

6.6.2 ANCOVA to Determine if Z Differs Between Lakes

The method described in the box is the same as the ANCOVA discussed for Box 3.11 in the Chapter 3 Vignette and is fit with `lm()`. Also note that `subset=` argument is used to restrict the model to only age-2 and older fish (the authors note that fish age-2 and older are fully recruited to the gear). The type-I SS ANOVA table is extracted from the saved `lm` object with `anova()`, whereas the type-III and type-II SS ANOVA tables are extracted with `Anova()` (from the `car` package) with the appropriate `type=` argument. Because the interaction term is the last term in this model, all three types of SS provide the same p-value leading to the result that the interaction is insignificant. An insignificant interaction indicates that the slope does not differ between the groups. As the slope is directly related to the estimated instantaneous mortality rate (Z) this also suggests that there is no difference in Z between the two lakes. A plot of the fitted models to further illustrate this point is obtained with `fitPlot()` from the `FSA` package.

```
> lm1 <- lm(logcatch~age*lake,data=d6,subset=age>=2)
> anova(lm1)
```

```
Analysis of Variance Table

Response: logcatch
          Df  Sum Sq Mean Sq F value    Pr(>F)
age        1 20.0825 20.0825 26.0612 0.0003414
lake       1  0.3946  0.3946  0.5120 0.4891677
age:lake   1  0.6598  0.6598  0.8563 0.3746383
Residuals 11  8.4765  0.7706
```

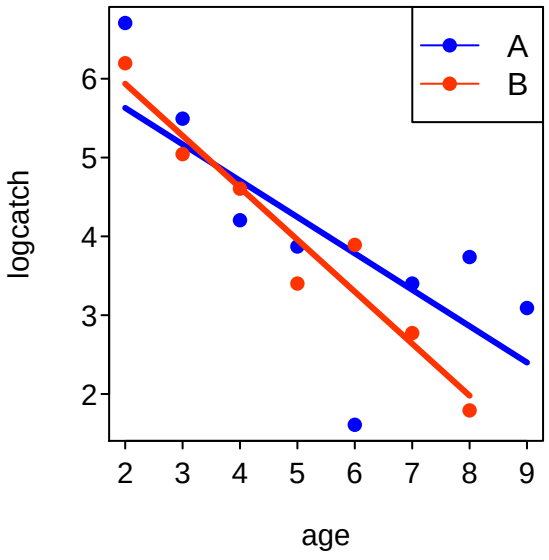
```
> Anova(lm1,type="III")
```

```
Anova Table (Type III tests)

Response: logcatch
            Sum Sq Df F value    Pr(>F)
(Intercept) 50.793  1 65.9140 5.677e-06
age          8.942  1 11.6047  0.005861
lake         0.263  1  0.3412  0.570899
age:lake     0.660  1  0.8563  0.374638
Residuals    8.477 11
```

```
> Anova(lm1,type="II")
```

```
Anova Table (Type II tests)

Response: logcatch
           Sum Sq Df F value    Pr(>F)
age       20.4650  1 26.5575 0.0003166
lake       0.3946  1  0.5120 0.4891677
age:lake   0.6598  1  0.8563 0.3746383
Residuals  8.4765 11
```

```
> fitPlot(lm1,main="")
```



The overall estimate of the common instantaneous mortality rate is found by fitting a model to the combined data set (i.e., ignoring `lake`). Thus, the overall instantaneous mortality rate is 0.53.

```
> lm2 <- lm(logcatch~age,data=d6,subset=age>=2)
> anova(lm2)
```

```
Analysis of Variance Table

Response: logcatch
          Df  Sum Sq Mean Sq F value    Pr(>F)
age        1 20.0825 20.0825  27.392 0.0001613
Residuals 13  9.5309  0.7331
```

```
> coef(lm2)
```

```
(Intercept)         age
  6.7901714  -0.5320886
```

```
> confint(lm2)
```

```
                2.5 %     97.5 %
(Intercept)  5.538711  8.0416322
age         -0.751722 -0.3124552
```

6.7 Cohort Catch Curves

Spring electrofishing samples at 40 sites in Normandy Reservoir indicated that recruitment by Spotted Bass (*Micropterus punctulatus*) varied more than twofold among years Sammons and Bettoli (1998); therefore, analysis of cohort catch curves was employed. The catch from the 1992 cohort in annual samples taken between 1993 and 1998 are shown in the text. These and other data suggested that fish were not fully recruited to the gear until age-2.

6.7.1 Preparing Data

The data were entered into a data frame using `data.frame()` below.

```
> ( d7 <- data.frame(age=1:6,catch=c(65,66,27,6,4,1)) )
```

```
  age catch
1   1    65
2   2    66
3   3    27
4   4     6
5   5     4
6   6     1
```

6.7.2 Chapman-Robson Method

The code below analyzes these data with `chapmanRobson()` as shown in Box 6.3. Note that the authors suggest that age-2 and older fish are recruited to the gear.

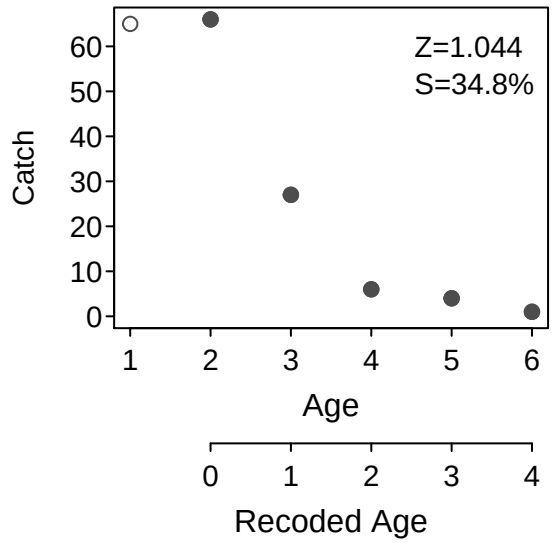
```
> cr1 <- chapmanRobson(catch~age,d7,2:6)
> summary(cr1)
```

```
   Estimate Std. Error
S 34.810127 3.80183307
Z  1.043845 0.06309479
```

```
> confint(cr1)
```

```
     95% LCI   95% UCI
S 27.3586707 42.261582
Z  0.9201811  1.167508
```

```
> plot(cr1)
```



6.7.3 Catch-Curve Method

The code below analyzes these data with `catchCurve()` as shown in Box 6.4. Note that the authors also note (correctly) that the last age should not be used in the catch curve analysis because of the very small sample size.

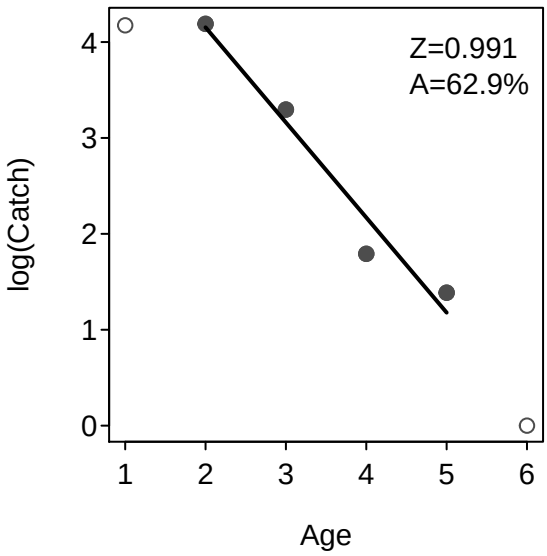
```
> cc1 <- catchCurve(catch~age,d7,2:5)
> summary(cc1)
```

```
    Estimate Std. Error  t value   Pr(>|t|)
Z  0.9914159  0.1433995 6.913662 0.02028661
A 62.8949035         NA       NA         NA
```

```
> confint(cc1)
```

```
     95% LCI   95% UCI
Z  0.3744175  1.608414
A 31.2310244 79.979516
```

```
> plot(cc1)
```



6.8 Mortality Estimation with Length-Based Models

We use a Largemouth Bass (*Micropterus salmoides*) data set from Columbus Lake, Mississippi, to illustrate mortality computations from length-based models. The von Bertalanffy model parameters (K and L_∞) were available from a parallel study in Columbus Lake and were K=0.226 and L_∞=636 mm (see Chapter 5 for calculations). All Largemouth Bass 150mm or longer were considered equally vulnerable to the collection gear (electrofishing); thus, L_x=150 mm. The mean and median length of fish 150 mm or longer in the data set were L_{mean}=260 mm and L_{median}=255 mm.

6.8.1 Ogle Comment

The raw data for the example in the box does not exist. Indeed enough information is given in that box to complete the calculations but most of that information in in summarized form. A reasonable approximation of the length data can be made from the results shown for the length frequency graph. The code here shows how I simulated those data. The age data that would be required to estimate L_∞ and K cannot be simulated from the given information.

The basic idea is to "expand" the counts of individuals in each of the 1-cm length bins to the value at the low end of the bin and then add a random "millimeter" component to simulate data recorded to millimeters. I will begin this process by entering the length bins (lower-value of the bin) and the frequency in the bins into two vectors (`bins`) and `freqs`). I then check the length of these vectors (i.e. number of numbers in the vector) to make sure that they at least match (I don't trust myself!)

```
> bins <- 9:52
> freqs <- c(5,39,43,55,64,86,106,99,82,81,56,45,27,36,43,52,65,73,63,42,44,40,37,31,36,
             15,19,18,13,8,20,19,9,12,9,11,13,13,4,10,6,3,3,4)
> length(bins)
```

```
[1] 44
```

```
> length(freqs)
```

```
[1] 44
```

A vector of the "floor" measurement for each fish (i.e., the lowest whole cm) is created by repeating each value in the `bins` vector the number of times shown in the `freqs` vector. These repetitions are constructed with `rep()`. A table of these values is then created to see if they match the summary information in the box. As they did, I then multiple the values by 10 to convert the cm to mm.

```
> len.cm <- rep(bins,times=freqs)
> table(len.cm)
```

```
len.cm
  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28
  5  39  43  55  64  86 106  99  82  81  56  45  27  36  43  52  65  73  63  42
 29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48
 44  40  37  31  36  15  19  18  13   8  20  19   9  12   9  11  13  13   4  10
 49  50  51  52
  6   3   3   4
```

```
> len.mm <- len.cm*10
```

I then set the random number seed so that you could reproduce the exact same results that I have produced.

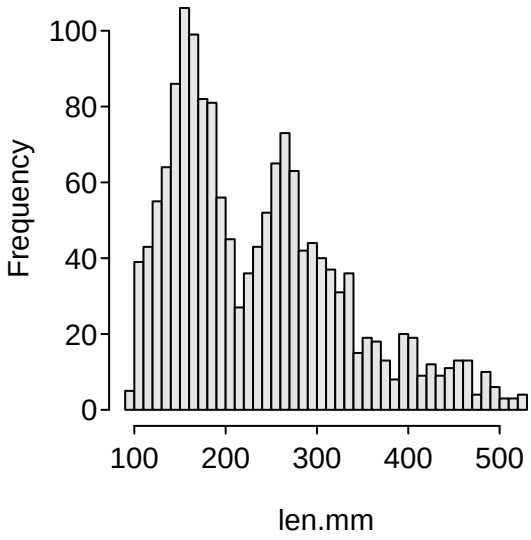
```
> set.seed(2101)
```

The `sample()` function is then used to construct as many random integers from between 0 and 9 (first argument) as there are values in the `len.mm` vector (note that sampling from 0:9 must be done with replacement; i.e., use of `replace=TRUE`). and then added to the `len.mm` vector,

```
> rands <- sample(0:9,length(len.mm),replace=TRUE)
> len.mm <- len.mm + rands
```

The values in `len.mm` now contain random data that appears to have been measured to the nearest mm, but when summarized in a length-frequency graph by 10-mm bins should match the results shown in the box.

```
> hbins <- seq(90,530,by=10)
> h <- hist(len.mm,breaks=hbins,right=FALSE,col="gray90",main="")
```




```
> h$mids
```

```
 [1]  95 105 115 125 135 145 155 165 175 185 195 205 215 225 235 245 255 265 275
[20] 285 295 305 315 325 335 345 355 365 375 385 395 405 415 425 435 445 455 465
[39] 475 485 495 505 515 525
```

```
> h$counts
```

```
 [1]   5  39  43  55  64  86 106  99  82  81  56  45  27  36  43  52  65  73  63
[20]  42  44  40  37  31  36  15  19  18  13   8  20  19   9  12   9  11  13  13
[39]   4  10   6   3   3   4
```

Finally, these simulated data were written out to a text file for use below as if the data had been entered and read from an external data file and not simulated in this manner.

```
> write.table(data.frame(len=len.mm),"Box6_8.txt",quote=FALSE,row.names=FALSE,col.names=TRUE)
```

6.8.2 Preparing Data

The Box6_8.txt data file is read and the structure is observed below. Note that the authors restricted their analysis to fish 150-mm and larger because those fish were fully recruited to the gear. A new data frame with just these fish is constructed with `Subset()` where the first argument is the original data frame name and the second argument is a conditioning statement for the subset.

```
> d8 <- read.table("data/Box6_8.txt",header=TRUE)
> str(d8)
```

```
'data.frame':   1559 obs. of  1 variable:
 $ len: int  94 91 91 95 99 101 107 102 109 109 ...
```

```
> d8a <- Subset(d8,len>=150)
> str(d8a)
```

```
'data.frame':   1267 obs. of  1 variable:
 $ len: int  154 154 158 152 150 153 156 159 151 152 ...
```

6.8.3 Beverton-Holt Method for Estimating Z from Mean Length

The authors illustrate the use of equation 6.8 to estimate Z. This method requires knowledge of L_∞ and K from the von Bertalanffy growth model (see Box 5.4 in the Chapter 5 vignette for how to fit a von Bertalanffy model to length-at-age data). In this box, the given summary information is stored into two objects and the mean (using `mean()`) and minimum (using `min()`) lengths are found and stored into two objects. Thus, the instantaneous mortality (Z) using equation 6.8 is estimated in the last line below.

```
> Linf <- 636
> K <- 0.226
> ( Lmean <- mean(d8a$len) )
```

```
[1] 259.6803
```

```
> ( Lmin <- min(d8a$len) )
```

```
[1] 150
```

```
> ( Z1 <- K*(Linf-Lmean)/(Lmean-Lmin) )
```

```
[1] 0.7754191
```

6.8.4 Hoenig Method for Estimating Z from Median Length

Hoenig (1983) modified Beverton and Holt's method to produce equation 6.9 in the text. This method still requires L_∞, K, and $L_{minimum}$ but it also requires the median length (as found with `median()`. The instantaneous mortality (Z) using equation 6.9 is estimated in the last line below using two intermediate calculations in the previous two lines.

```
> ( Lmedian <- median(d8a$len) )
```

```
[1] 251
```

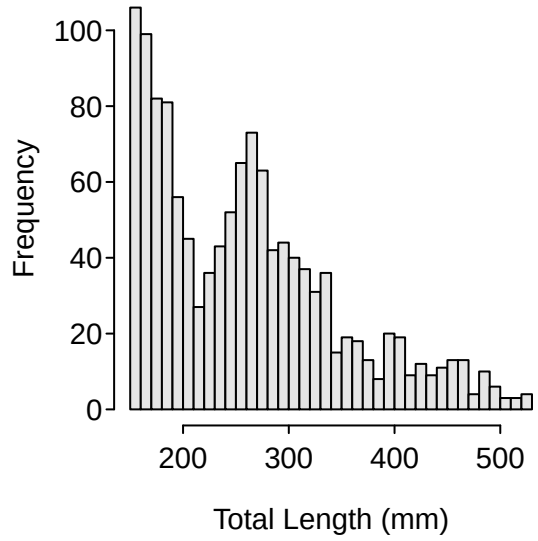
```
> Ymedian <- -log(1-Lmedian/Linf)
> Ymin <- -log(1-Lmin/Linf)
> ( Z2 <- 0.693*K / (Ymedian-Ymin) )
```

```
[1] 0.6722804
```

6.8.5 Estimating Z from Length-Converted Catch Curve

Estimating Z from a length-converted catch curve as described by Pauly (1984) requires finding the midpoint of length bins and the number of fish in each of those bins. In essence, one needs to know the basic calculations for constructing a length-frequency histogram. In R, histograms are constructed with `hist()`. This function only requires a vector of values to construct the histogram as the first argument. However, you can control the bins used by the histogram algorithm by sending break points for the bins to the `breaks=` argument. In addition, `hist()` defaults to the bins being right-inclusive – i.e., a value of 10 would be included in the 9-10 bin rather than the 10-11 bin – whereas most fisheries biologists are used to using left-inclusive bins. Left-inclusive bins can be used by including the `right=FALSE` argument to `hist()`. In the example below, left-inclusive bins that start at 150-mm, end at 530-mm, and are 10-mm wide were created. The histogram was then constructed, but it was also saved to an object (`h`). The midpoint values and the counts in each bin are extracted by appending `$mids` and `$counts`, respectively, to the saved `hist` object, as illustrated below.

```
> bins <- seq(150,530,by=10)
> h <- hist(d8a$len,breaks=bins,right=FALSE,xlab="Total Length (mm)",main="",col="gray90")
```




```
> h$mids
```

```
 [1] 155 165 175 185 195 205 215 225 235 245 255 265 275 285 295 305 315 325 335
[20] 345 355 365 375 385 395 405 415 425 435 445 455 465 475 485 495 505 515 525
```

```
> h$counts
```

```
 [1] 106  99  82  81  56  45  27  36  43  52  65  73  63  42  44  40  37  31  36
[20]  15  19  18  13   8  20  19   9  12   9  11  13  13   4  10   6   3   3   4
```

The midpoint values are converted to t-prime values as illustrated in the text and the counts were log-transformed.

```
> tprime <- -log(1-h$mids/Linf)
> round(tprime,3)                # rounded only to compare to values in Box 6.8
```

```
 [1] 0.279 0.300 0.322 0.344 0.366 0.389 0.413 0.437 0.461 0.486 0.512 0.539
[13] 0.566 0.594 0.623 0.653 0.684 0.715 0.748 0.782 0.817 0.853 0.891 0.930
[25] 0.970 1.013 1.057 1.103 1.152 1.203 1.257 1.314 1.374 1.438 1.506 1.580
[37] 1.659 1.746
```

```
> logN <- log(h$counts)
```

The linear regression denoted by equation 6.10 in the book is then fit with `lm()`. The model coefficients are extracted with `coef()` and the slope (i.e., second coefficient in `coef()`) is isolated and saved to an object (`b`). Finally, the instantaneous mortality is estimated from these values with the last line of code below.

```
> lm1 <- lm(logN~tprime)
> coef(lm1)
```

```
(Intercept)      tprime
   4.973924   -2.228366
```

```
> ( b <- coef(lm1)[2] )
```

```
   tprime
-2.228366
```

```
> ( Z3 <- K*(1-b) )
```

```
   tprime
0.7296106
```

6.8.6 Empirical Estimates of L_∞ and K

The authors (on page 249) provide three methods for estimating L_∞ and K if length-at-age data do not exist. These methods are illustrated here.

6.8.6.1 Three Largest Fish Method Estimate of L_∞ Finding the mean lengths of the three largest fish in the length vector is a bit of work, which is best shown in several steps – (1) order the vector positions from smallest to largest fish, (2) find the positions of three largest fish, (3) find the lengths of those three fish, and (4) find the mean of those lengths.

```
> ord.len <- order(d8a$len)
> ord.top3 <- ord.len[(length(d8a$len)-2):length(d8a$len)]
> ( len.top3 <- d8a$len[ord.top3] )
```

```
[1] 523 525 525
```

```
> ( Lmean.top3 <- mean(len.top3) )
```

```
[1] 524.3333
```

Pauly (1984) then suggests dividing this mean by 0.95 to produce an estimate of L_∞. This is dramatically lower then the L_∞ provided by the authors.

```
> ( Linf1 <- Lmean.top3/0.95 )
```

```
[1] 551.9298
```

6.8.6.2 Froese and Binohlan (2000) Empirical Equation Method Estimate of L_∞ Froese and Binohlan (2000) developed an empirical equation for estimating L_∞ from the maximum length observed for the population. The equation is $log(L_\infty) = 0.044 + 0.984log(L_{max})$. The maximum value in these data and the estimate of L_∞ using this equation are shown below. This is even more dramatically lower then the L_∞ provided by the authors.

```
> ( Lmax <- max(d8a$len) )
```

```
[1] 525
```

```
> ( Linf2 <- exp(0.044+0.984*log(Lmax)) )
```

```
[1] 496.3016
```

6.8.6.3 Wetherall et al. (1987) Method Estimate of L_∞ The method of Wetherall et al. (1987) requires knowing the lower limit of each length interval, which can be obtained by appending `$breaks` to the saved `hist` object from above. In addition, the method requires knowing the mean lengths of all fish in each interval and all larger intervals. For example, we must find the mean length of fish in the 150-mm and all larger intervals, in the 160-mm and all larger intervals, the 170-mm and all larger intervals, and so on. This type of calculation is best handled with the loop shown below. Finally, the difference between these means and the lower limit of the length intervals is computed and saved to an object.

```
> ( Lx <- h$breaks )                                  # lower limit of each TL interval
```

```
 [1] 150 160 170 180 190 200 210 220 230 240 250 260 270 280 290 300 310 320 330
[20] 340 350 360 370 380 390 400 410 420 430 440 450 460 470 480 490 500 510 520
[39] 530
```

```
> Lmeans <- NULL                                      # initiate the Lmeans vector
> for (i in 1:length(Lx)) {                           # loop through length intervals
    Lmeans <- c(Lmeans,mean(d8a$len[d8a$len>=Lx[i]]))
  }
> Lmeans                                              # mean for each interval & higher
```

```
 [1] 259.6803 269.3075 279.0584 287.7990 297.1046 303.9597 309.5902 312.9092
 [9] 317.2531 322.3931 328.7188 337.0522 347.6036 358.0661 365.8841 374.8102
[17] 383.8019 393.1051 401.7347 413.3062 418.6237 425.6686 432.5987 437.7778
[25] 440.9559 449.0345 457.8557 462.3750 468.3289 472.9851 478.6786 486.0465
[33] 495.4000 498.6923 507.5000 515.0000 519.7143 523.2500      NaN
```

```
> Ldiff <- Lmeans-Lx
```

The linear regression of these differences on the lower limits of the length intervals is then fit with `lm()` as usual. The y-intercept and slope coefficients are isolated from the `coef()` similar to what was shown above. These values are used to estimate L_∞ as shown in the text below equation 6.11. Again, this estimate is dramatically lower then the L_∞ provided by the authors.

```
> lm2 <- lm(Ldiff~Lx)
> a <- coef(lm2)[1]      # isolate intercept
> b <- coef(lm2)[2]      # isolate slope
> ( Linf3 <- -a/b )
```

```
(Intercept)
   565.1053
```

6.9 Total Mortality Estimation from Marked Recaptures

A 5-year tagging program was completed to monitor mortality (as well as population size and recruitment which are not shown in this example) of Largemouth Bass (*Micropterus salmoides*) in Lake Travesty. Fish were marked and recaptured annually during a 2-week collection period in spring each year, and results were analyzed with the Jolly-Seber Method. The five year-history of marking and recaptures (preadjusted for tag loss) are shown in the table in the box in the text.

6.9.1 Ogle Comment

The Jolly-Seber method is implemented with `mrOpen()` from the `FSA` package. (The `mrOpen()` function and the Jolly-Seber method are discussed in much detail in Chapter 8 of the Introductory Fisheries Analysis with R vignettes. The presentation below will assume that you have read that vignette). The biggest differences between the use of `mrOpen()` and what was described in the box is the use of Seber's modified formulas (see bottom of page 253 in AIFFD), the allowance for possible accidental deaths during the sampling process in `mrOpen()`, the transposition of the summary table shown in Table 6.2 of the chapter, and the use of different terminology. The following table shows the equivalencies in terminology between `mrOpen()` and the box.

AIFFD	`mrOpen()`	Meaning
m_i	n_i	Number of fish captured in the ith sample
$r_{\cdot i}$	m_i	Number of marked fish recaptured in the ith sample.
–	u_i	Number of UNmarked fish captured in the ith sample.
r_{ii}	m_{ji}	Number of marked fish in the ith sample last caught in the jth sample.
–	R_i	Number of marked fish returned to the population after the ith sample.
$r_{i\cdot}$	r_i	Number of the fish released in the ith sample that are subsequently recaptured.
k_i	z_i	Number of fish captured before the ith sample, not captured in the ith sample, but captured subsequ
B_i^*	M_i	Estimated number of marked fish in the population just prior to the ith sample.

6.9.2 Preparing Data

The summarized catch data is provided in the box. These results are entered into two matrices below. For example, the r_{ii} portion of the table on page 255 in the book is entered below. Note that `rep()` is used to repeat the first argument by the number of times in the second argument and `rbind()` is used to "bind" together multiple vectors of the same length in a row-wise manner (i.e., each vector is a row in the resulting matrix). Also note that the matrix has been entered as a square 5x5 matrix where the last column, not shown in the book table, containing all blanks is needed (i.e., this column is all blanks because the last sample is such that the fish released cannot be subsequently recaptured). As noted above, this portion of the table is a transposition of the "top" portion of the "Method B" table required by `mrOpen()`. In other words, the rows of this table should be the columns of the required table and vice versa. Fortunately, the entire matrix can be easily transposed with `t()`.

```
> s1 <- rep(NA,5)
> s2 <- c(43,rep(NA,4))
> s3 <- c(28,31,NA,NA,NA)
> s4 <- c(12,16,48,NA,NA)
> s5 <- c(3,9,19,37,NA)
> ( aiffd.top <- rbind(s1,s2,s3,s4,s5) )
```

```
   [,1] [,2] [,3] [,4] [,5]
s1   NA   NA   NA   NA   NA
s2   43   NA   NA   NA   NA
s3   28   31   NA   NA   NA
s4   12   16   48   NA   NA
s5    3    9   19   37   NA
```

```
> ( mb.top <- t(aiffd.top) )
```

```
     s1 s2 s3 s4 s5
[1,] NA 43 28 12  3
[2,] NA NA 31 16  9
[3,] NA NA NA 48 19
[4,] NA NA NA NA 37
[5,] NA NA NA NA NA
```

The "bottom" portion of the "Method B" table required by `mrOpen()` is constructed from some of the marginal values of the table shown in the text. However, one must be very careful to note the different definitions as outlined in the table above. Also note that the book table does not show what the sample size was in the last sample. Thus, the only item known from the last sample is the number of observed marked (i.e., recaptured) fish. Fortunately, this information is not required in the Jolly-Seber calculations. However, `mrOpen()` requires non-negative values in the entire matrix which is accomplished below by entering m for n for the last sample. This is purely a work-around for dealing with the missing information from the table in the box and the requirements of `mrOpen()`.

```
> n <- c(643,489,712,630,68)     # m_i_ from book
> m <- c(0,43,59,76,68)          # r_.i_ from book
> u <- n-m
> R <- n                         # assumes no accidental deaths, thus m_i_ from book
> ( mb.bot <- rbind(m,u,n,R) )
```

```
  [,1] [,2] [,3] [,4] [,5]
m    0   43   59   76   68
u  643  446  653  554    0
n  643  489  712  630   68
R  643  489  712  630   68
```

6.9.3 Performing Jolly-Seber Calculations

The Jolly-Seber calculations are computed in R by submitting the top and bottom portions of the Method B table as the first two arguments to `mrOpen()` and submitting the return object to `summary()`. If `verbose=TRUE` is used then the complete set of "observables" will be printed first. The first three columns of this output is simply repeated from the bottom portion of the Method B table. The last two columns are computed from the top portion of the Method B table and correspond to marginal columns of the table shown in the box (but make sure to note the different terminology).

```
> mr1 <- mrOpen(mb.top, mb.bot)
> summary(mr1,verbose=TRUE)
```

```
Observables
   m   n   R  r  z
1  0 643 643 86 NA
2 43 489 489 56 43
3 59 712 712 67 40
4 76 630 630 37 31
5 68  68  68 NA NA

Estimates
Standard error of phi includes sampling and individual variability.

      M  M.se      N   N.se   phi phi.se      B  B.se
1    NA    NA     NA     NA 0.642  0.112     NA    NA
2 412.6  70.8 4595.4 1010.1 0.557  0.105 3124.7 945.2
3 478.4  80.1 5685.1 1158.8 0.522  0.115 1872.7 770.4
4 590.8 122.4 4841.2 1113.8    NA     NA     NA    NA
5    NA    NA     NA     NA    NA     NA     NA    NA
```

It was noted previously that `mrOpen()` uses the Seber (1965) modification to estimate the number of marks extant in the population (however, it does not add 1 to the m_i term as shown in the text). This is not the method illustrated in the box. For comparisons purpose, the estimated number of extant marks can be calculated "by hand" using the formula for B_i^* (without the additional 1 on the m_i term) on the bottom of page 253,

- B_3^* = 59+(712+1)*40/(67+1) = 478.4118 = 478 = M_3
- B_2^* = 43+(489+1)*43/(56+1) = 412.6491 = 413 = M_2
- A_2 = 1 - 478/(413-43+489) = 0.444

Estimated survival rates are obtained by submitting the saved `mrOpen` object to `summary()` using the `parm="phi"` argument. The mortality rates shown in the text are found by subtracting the "phi" estimates from 1. Thus, for example A_2 = 1-ϕ_2 = 1-0.557 = 0.443. Confidence intervals for the survival estimates (phi) are obtained by submitting the `mrOpen` object to `confint()` using the `parm="phi"`.

```
> summary(mr1,parm="phi")
```

```
    phi phi.se
1 0.642  0.112
2 0.557  0.105
3 0.522  0.115
4    NA     NA
5    NA     NA
```

```
> confint(mr1,parm="phi")
```

```
  phi.lci phi.uci
1   0.423   0.861
2   0.351   0.764
3   0.296   0.748
4      NA      NA
5      NA      NA
```

```
Reproducibility Information
  Compiled Date: Wed May 13 2015
  Compiled Time: 7:01:57 PM
  Code Execution Time: 1.09 s

  R Version: R version 3.2.0 (2015-04-16)
  System: Windows, i386-w64-mingw32/i386 (32-bit)
  Base Packages: base, datasets, graphics, grDevices, methods, stats, utils
  Required Packages: FSA, car and their dependencies (dplyr, gdata, gplots,
    graphics, Hmisc, knitr, lmtest, MASS, mgcv, multcomp, nnet, pbkrtest,
    plotrix, quantreg, relax, sciplot, stats)
  Other Packages: car_2.0-25, FSA_0.6.13, knitr_1.10.5, rmarkdown_0.6.1
  Loaded-Only Packages: acepack_1.3-3.3, assertthat_0.1, bitops_1.0-6,
    caTools_1.17.1, cluster_2.0.1, codetools_0.2-11, colorspace_1.2-6,
    DBI_0.3.1, digest_0.6.8, dplyr_0.4.1, evaluate_0.7, foreign_0.8-63,
    formatR_1.2, Formula_1.2-1, gdata_2.16.1, ggplot2_1.0.1, gplots_2.17.0,
    grid_3.2.0, gridExtra_0.9.1, gtable_0.1.2, gtools_3.4.2, highr_0.5,
    Hmisc_3.16-0, htmltools_0.2.6, KernSmooth_2.23-14, lattice_0.20-31,
    latticeExtra_0.6-26, lme4_1.1-7, lmtest_0.9-33, magrittr_1.5,
    MASS_7.3-40, Matrix_1.2-0, mgcv_1.8-6, minqa_1.2.4, multcomp_1.4-0,
    munsell_0.4.2, mvtnorm_1.0-2, nlme_3.1-120, nloptr_1.0.4, nnet_7.3-9,
    parallel_3.2.0, pbkrtest_0.4-2, plotrix_3.5-11, plyr_1.8.2,
    proto_0.3-10, quantreg_5.11, RColorBrewer_1.1-2, Rcpp_0.11.6,
    relax_1.3.15, reshape2_1.4.1, rpart_4.1-9, sandwich_2.3-3,
    scales_0.2.4, sciplot_1.1-0, SparseM_1.6, splines_3.2.0, stringi_0.4-1,
    stringr_1.0.0, survival_2.38-1, TH.data_1.0-6, tools_3.2.0,
    yaml_2.1.13, zoo_1.7-12
```

References

Chapman, D. G., and D. S. Robson. 1960. The analysis of a catch curve. Biometrics 16:354–368.

Froese, R., and C. Binohlan. 2000. Empirical relationships to estimate asymptotic length, length at first maturity and length at miximum yield per recruits in fishes, with a simple method to evaluate length frequency data. Journal of Fish Biology 56:758–773.

Hoenig, J. M. 1983. Empirical use of longeity data to estimate mortality rates. U.S. National Marine Fisheries Service Fishery Bulletin 81:898–903.

Pauly, D. 1984. Fish population dynamics in tropical waters: A manual for use wtih programmable calculators. ICLARM (International Center for Living Aquatic Resources Management) Studies; Reviews 8, Manila, Philippines.

Robson, D. S., and D. G. Chapman. 1961. Catch curves and mortality rates. Transactions of the American Fisheries Society 90:181–189.

Sammons, S. M., and P. W. Bettoli. 1998. Influence of water levels and habitat manipulations on fish recruitment in normandy reservoir. Fisheries Report, Tennessee Wildlife Resources Agency, Nashville.

Seber, G. A. F. 1965. A note on the multiple recapture census. Biometrika 52:249–259.

Wetherall, J. A., J. J. Polovina, and S. Ralston. 1987. Estimating growth and mortality in steady-state fish stocks from length-frequency data. *in* ICLARM (international center for living aquatic resources management) conference proceedings 13, manila, philippines and kuwait institute for scientific research, safat.