Ricker and Larkin Models - Assignment 4

Christian Carson

03/12/2024

Read in the data and set up the procedure section. Data can be read from github at: https://github.com/christiancarson/REM661/blob/main/Assingment%204/Sockeye.data (https://github.com/christiancarson/REM661/blob/main/Assingment%204/Sockeye.data)

Import Data from github

```
data <- read.table("https://raw.githubusercontent.com/christiancarson/REM661/main/Assing
ment%204/Sockeye.data", header=TRUE, sep="\t")
stocks    <- unique( data$Stock )
nstock    <- length( stocks )
```

Load libraries

Load required libraries and install if necessary.

```
#Load required libraries
library(ggplot2)
library(gridExtra)
library(dplyr)
```

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

```
## The following object is masked from 'package:gridExtra':
##
##     combine
```

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

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

```
library(gt)
```

Ricker Model

The Ricker model is written as:

$$\ln\left(\frac{R_t}{S_t}\right) = \alpha - \beta S_t + \omega_t$$

where: - R_t is the number of recruits in year t - S_t is the number of spawners in year t - α is the intercept - β is the slope - ω_t is the error term

Here we will fit a linear model to estimate alpha and beta in log(R/S) = alpha - beta * S

```r
Ricker <- function() {
  #Here we create a list to iterate over each stock
  rick <- list()

  #Now we create a dataframe to store the outputs of that iteration in the list
  results <- data.frame(
    Stock = character(),
    Alpha = numeric(),
    Beta = numeric(),
    stringsAsFactors = FALSE
  )

  #Then we create a loop to iterate over each stock
  for (i in 1:nstock) {
    #We match the stock from the list to the stock in the observation data
    stock.id <- data$Stock == stocks[i]
    #We then create a variable for the recruits and spawners for that stock
    R <- data$Recruits[stock.id]
    S <- data$Spawners[stock.id]
    #Here we create a LM that predicts the logarithm of the ratio of recruits to spawners (log(R / S)) based on the values of spawners (S).
    fit <- lm(log(R / S) ~ S)
    #We store the output of each stock in the rick list. The [[i]] notation indicates that we are storing the output at index i in the list. The output is a list containing two elements: fit and r. fit stores the linear regression model object, and r stores the predicted values of recruits (exp(fit$fitted.values) * S).
    rick[[i]] <- list(fit = fit, r = exp(fit$fitted.values) * S)
    #then, we use rbind to store the results of the iteration (stock name, first coef, second coef) in the dataframe, with the stock name, alpha, and beta as the columns.
    results <- rbind(
      results,
      data.frame(
        Stock = stocks[i],
        Alpha = coef(fit)[1],
        Beta = coef(fit)[2]
      )
    )
  }
#Finally, we return the list of models and the dataframe of coefficients together
  return(list(models = rick, coefficients = results))
}
```

Larkin Model

Next, we will create a function to estimate parameters and fitted values based on the Larkin model. The Larkin model is written as:

$$\ln\left(\frac{R_t}{S_t}\right) = \alpha - \beta_0 S_t - \beta_1 S_{t-1} - \beta_2 S_{t-2} - \beta_3 S_{t-3} + \omega_t$$

where: - R_t is the number of recruits in year t - S_t is the number of spawners in year t - α is the intercept - β_0 is the slope for the current year - β_1 is the slope for the previous year - β_2 is the slope for the year before the previous year - β_3 is the slope for the year before the year before the previous year - ω_t is the error term

It is largley similar to the Ricker model, but we need to add lagged spawner beta estimates to the data to account for overdispersion.

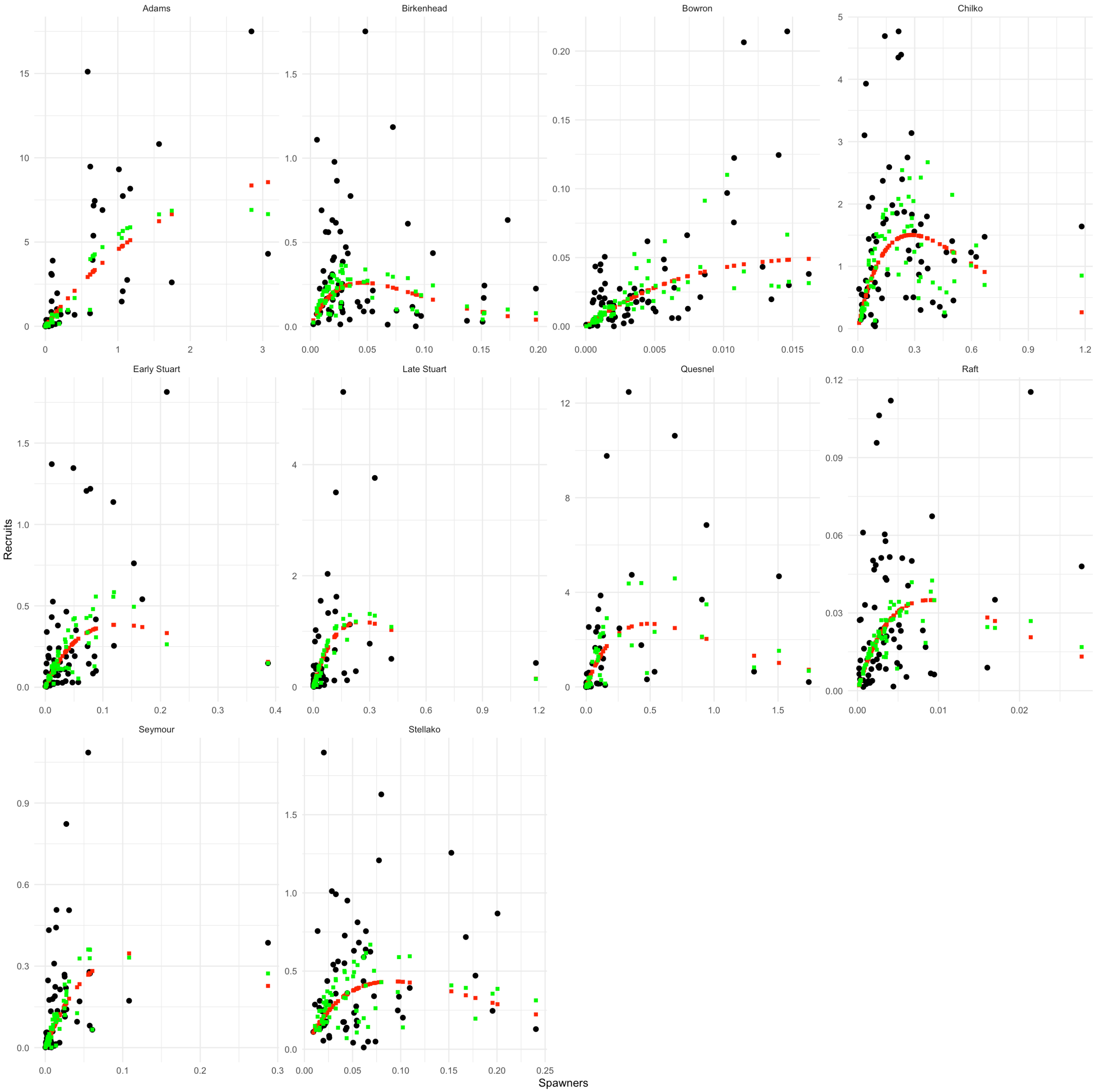
```
Larkin <- function(){
  #Create a list to fill with outputs from each stock
  lark <- list()
  #create a dataframe to store the outputs of the iteration, with the structure we want them in
  results <- data.frame(Stock=character(), Alpha=numeric(), Beta0=numeric(), Beta1=numeric(), Beta2=numeric(), Beta3=numeric(), stringsAsFactors=FALSE)

  #Loop over stocks
  for(i in 1:nstock){
    #matches the stock from the list to the stock in the observation data
    stock.id <- data$Stock == stocks[i]
    #create a variable for the recruits and spawners for that stock
    R <- data$Recruits[stock.id]
    S <- data$Spawners[stock.id]
    #create a variable for the number of years
    nyr <- length(R)

    #ensure enough data points for model fitting, for the Larkin model we need at least 4 years of data so we use an if function for >3 years
    if(nyr > 3){
      #now we create variables for the recruits and spawners for that stock, but we select recruits from the 4th year onwards
      Rt <- R[4:nyr]
      St <- S[4:nyr]
      #for the spawner variables, we need to start from the third element of S and go to the second to last element of S, this would make a 1 year lag
      Sm1 <- S[3:(nyr-1)]
      #for the two year lag, we start from the second element of S and go to the third to last element of S
      Sm2 <- S[2:(nyr-2)]
      #for the three year lag, we start from the first element of S and go to the fourth to last element of S
      Sm3 <- S[1:(nyr-3)]
#now we store those outputs in a dataframe
      dat <- data.frame(Rt=Rt, St=St, Sm1=Sm1, Sm2=Sm2, Sm3=Sm3)
      #from the dataframe, we make a glm that predicts the logarithm of the ratio of recruits to spawners (log(Rt / St)) based on the values of spawners (St), Sm1, Sm2, and Sm3
      fit <- glm(log(Rt/St) ~ St + Sm1 + Sm2 + Sm3, data=dat)
#we then store the output of each stock in the lark list. The [[i]] notation indicates that we are storing the output at index i in the list. The output is a list containing two elements: fit and r. fit stores the glm model object, and r stores the predicted values of recruits (exp(fit$fitted.values) * St).
      lark[[i]] <- list(fit=fit, r=exp(fit$fitted.values) * St)
    #Now we take the coefficients from the model and store them to be pulled immediately next
      coefs <- coef(fit)
      #then, we use rbind to store the results of the iteration (stock name, alpha, beta0, beta1, beta2, beta3) in the dataframe
      results <- rbind(results, data.frame(Stock=stocks[i], Alpha=coefs[1], Beta0=coefs[2], Beta1=coefs[3], Beta2=coefs[4], Beta3=coefs[5]))
    }
  }
  #finally, we return the list of models and the dataframe of coefficients together
  return(list(models=lark, coefficients=results))
}
```

Plotting

To account for the first three years of missing data in the Larkin model, we will offset the predictions to account for the missing lagged years.

```r
#call both Ricker and Larkin functions
rick_results <- Ricker()
lark_results <- Larkin()

#start with the original data frame and assign to adjust for missing years
combined_data <- data
#set with NAs
combined_data$Rick_Pred <- NA
#match the number of rows in combined_data by adding NAs to the Rick_Pred and Lark_Pred columns
for(i in 1:length(rick_results$models)) {
  #match the stock from the list to the stock in the observation data
  stock_id <- combined_data$Stock == stocks[i]
  combined_data$Rick_Pred[stock_id] <- rick_results$models[[i]]$r
}

#do the same for Larkin, but offset the predictions to account for missing lagged years
combined_data$Lark_Pred <- NA
for(i in 1:length(lark_results$models)) {
  #match the stock from the list to the stock in the observation data
  stock_id <- combined_data$Stock == stocks[i]
#now offset the predictions to account for missing lagged years, we do this by adding NAs to the Lark_Pred column from the 4th element onwards
  combined_data$Lark_Pred[stock_id][4:length(combined_data$Lark_Pred[stock_id])] <- lark_results$models[[i]]$r
}

#plot using ggplot2, initalizing the now cleaner combined_data and setting the aesthetics
p <- ggplot(combined_data, aes(x = Spawners, y = Recruits)) +
#add observations
  geom_point(color = "black", size = 2) +
#add predictions for Ricker and Larkin
  geom_point(aes(y = Rick_Pred), color = "red", shape = 15) +
  geom_point(aes(y = Lark_Pred), color = "green", shape = 15) +
  #facet by stock
  facet_wrap(~Stock, scales = "free") +
  #set labels
  labs(x = "Spawners", y = "Recruits") +
  #set theme
  theme_minimal()


# Print the plot
print(p)
```

```
## Warning: Removed 30 rows containing missing values (`geom_point()`).
```



```
#there will be a plotting error as there are NA values for the first three years across the 10 stocks, but this is only for the Larkin model predictions. It ensures that the predictions are offset to account for the missing lagged years
```