HOCHSCHULE RHEIN-WAAL
Rhine-Waal University of Applied Sciences

Examination Assignment

Module:	Data Analysis and Statistics
Exam part:	Data Analysis and Statistics
Examiner:	Prof. Dr. Schwind, Dipl.-Biol. Ralf Darius
Deadline for the submission:	31.08.2017, 11:59 pm

Study program	Begin of studies	Last name, First name
Information Engineering and Computer Science (M.Sc.)	**S 2017**	**Russo, Matthew**

Table of Contents

1. Introduction

The orca's (Orcinus orca) hunting patterns have been likened to those of a wolf pack, cooperating with a family group of up to 40 individuals (Orca 2017). With such a strong emphasis on the family unit it is clear why looking into the reproduction is meaningful. This study will simulate the lives of one thousand female orca individuals while they are within the age range where reproduction occurs. Over each life cycle of the orcas simulated it will also look into the individual's offspring. It will look into how many recruits, offspring that survive until they have reached sexual maturity, each individual orca produces.

The orca can be inhabits each of the Earth's oceans, being found in from anywhere in the arctic and antarctic regions to tropical seas (Matthiopoulos 2011, p.1). The ability to thrive in varying habitats is extremely interesting. Going beyond just simulating female orca's life cycles, this study will look deeper into the effect of different environmental factors on the inclusive fitness, the ability to produce recruits of female orcas. The way in which environmental factors are examined is by defining four scenarios where richness and stability are changed for each scenario.

To achieve this simulation development will take place in the programming language R. A series of loops will be built to represent trials and life cycles, one where one loop means one trial. The other loop will be the life cycle loop where a single loop represents a year in a life cycle. Running one thousand trials will create a large data set from which trends and conclusions can be gleaned.

The structure of the study is relatively straightforward. The second chapter is an in depth look into the methods. The script that was used to create the simulation will be broken down line by line to demonstrate the function of each piece of code, why the piece of code was written, and in some cases explain computation. The third chapter is to display the results, for each scenario there will be a histogram for the age of when the orca stopped breeding, a histogram of the inclusive fitness, and the computed average of how many recruits were produced per individual. The fourth and final chapter will be the discussion, where the results will be interpreted.

2. Methods

To create the population models I used the programming language R and the integrated development environment RStudio. The language R specializes in statistics, whether it is modelling populations, general computation, or graphics such as plotting graphs, it can complete the task. RStudio was created specifically to facilitate development in R with extra tools, both built in and external add-ons, to make tasks easier and streamlined. To model the populations I wrote a script in R to simulate the lives of individual female orcas. The script follows a simple development pattern of a loop within a loop allowing it to run any number of trials of orca lifespans.

```
#############################################################################################
# trials of female orca whale populations and their offspring

# list of ages
ages <- c()
# list of recruits
recruits <- c()
# trial number
trials <- 0

while(trials < 1000) {

  # increase trial
  trials <- trials + 1
  #individual's age
  age <- 15
  # is the individual alive, 0: no, 1: yes
  alive <- 1
  # if the female has a calf set to 1, if not 0
  calf <- 0
  # the current calf's age
  calfage <- 0
  # the number of weaned offspring of an individual female
  offspring <- 0

  # count of recruits for an individual female
  recruitCount <- 0
```

Figure 1: First section of the script, displaying global variables and trials

To begin the script we run the standard command to clean up the environment, deleting any stored data values. After that there is a divider to signify the beginning of the section, seen in Figure 1, that contains the modeling of the life cycles. The first global variable defined is ages, an empty list meant to contain the age of each orca at its death per simulation. The next variable is recruits, similar to ages is it an empty list, it will contain the number of recruits an orca produces. The indices of these two lists will line up so say if you look at orca at index 30 of ages, and index 30 of recruits, you can piece the two parts of information together to find out the age of the whale at the end of breeding and how many recruits it produced. It is important that these two lists are declared

outside of the trial and life cycle loops because they should not be reset at the end of a trial or life cycle. Next we initialize the trial variable at 0, essentially this is arbitrary, but the end goal is to run 1000 trials, so it could be set to anything as long as the difference between it and the conditional in the while loop is 1000. The declaration of the while loop does just that, saying that as long as trials is less than 1000 it will run the contents of the loop.

Once we are within the trial while loop we begin to declare the variables we want to reset to the initialized value on each trial, yet stay present throughout the entire life cycle of the simulated orca. The first variable that is present is trials, we increment it by 1 to signify a trial has taken place. It is important to notice that it is referencing itself as we take the pre-existing value and add 1 to it, then reassigning this value to the variable, making sure we keep track of all trials that have taken place. After this we define age, the age of the individual whose life we are simulating. Age is initialized at 15 because it is the age a female orca begins breeding and this simulation functions under the assumption that each trial is the life cycle of an orca who has the ability to breed. After that I initialize the variable alive, which displays whether or not the orca is alive, where 0 means the orca is dead and 1 means the orca is alive. I start with the value at 1 because each individual we simulate starts its life cycle living.

After these variables we initialize the ones regarding the orca's offspring. First we define calf, which distinguishes whether or not the individual orca has a calf, where 1 corresponds to yes and 0 to no. After calf comes calfage, which keeps track of the calf's age, both of these are started at 0 because the individual has not had a chance to breed until the life cycle while loop begins and if there is no calf it cannot have an age. After this is offspring, keeping track of how many calves an individual has given birth to whom have been weaned. Finally we define recruitCount, which records the number of recruits an individual has produced. Again you will see these variables are defined outside of the life cycle loop, which is extremely important because they are based on the life cycle as a whole and not reset on each year, or loop of the life cycle.

```
# loops over a single whale's life cycle while it is able to breed
while (alive == 1 && age < 40) {

  # represents condition of individual female
  co <- rnorm(1, 100, 10)

  # environment mean, represents quality of environment
  enMu <- 30
  # environment standard deviation, represents stability of environment
  enSD <- 2
  # random condition based on environment mean and standard deviation
  co2 <- rnorm(1, enMu, enSD)
  # add the random condition to the original condition
  co <- co + co2

  # one part of a linear predictor
  s0 <- -2
  # second part of a linear predictor
  s1 <- 0.05
  # linear predictor
  s <- s0+s1*co

  # survival rate
  sr <- exp(s)/(1+exp(s))

  # is the individual alive, 0: no, 1: yes
  alive <- rbinom(1,1,sr)

  #check is the individual is alive
  if (alive == 1) {
    # increment the individuals age by 1
    age <- age + 1
  }
```

Figure 2: Beginning of life cycle loop

This brings us to the second while loop, representing a life cycle of an individual female orca. Each pass of the loop represents a year in the life cycle of the individual. First I check the conditional within the while statement, whether or not the individual is alive, and then whether or not they are age 40 or under. The reason 40 is the maximum age allowed for the individual is because this study is looking at the female orca while it has the ability to breed, from age 15 to 40 (Matthiopoulos 2011, p.1), and if the individual reaches this maximum age we no longer care about how long it survives after due to the fact that it cannot produce any more recruits. Secondly I define co, this variable represents the condition of the individual, it is chosen randomly with a normal distribution,with a mean of 100 and a standard deviation of 10. This value gets chosen each year by random, hence why it is within the loop.

After this we want to determine if the individual has improved their condition in the year. To do this we are going to choose another normally distributed random value, as seen in co2. We choose this value using the variables enMu and enSD, where enMu represents a mean value for the environment. The mean value is a representation of the quality of the environment, if it is set high such as 30, the environment is rich, where as if it is set low, say at 2, the environment is poor. On

the other hand, the standard deviation value represents the stability of the environment where if the value is set high, say 30, the environment is very variable. If the standard deviation is set low, to 2, the environment is very stable. These two values are used together to create the second condition, which is added to the original condition, essentially telling whether or not the individual has improved or decreased its condition. For example if enMu is set to 30 and enSD is set to 2, the individual will be adding something around 30 to its condition each year. However if these numbers are reversed and the enMu is set to 2 and the enSD is set to 30, the individual could be adding large negative numbers to its condition regularly, because the standard deviation is so much larger than the mean, in turn decreasing the individual's condition.

Next we must calculate the survival rate of the individual to use and see if it survives the year. This is done by constructing a linear predictor, s, by using the values in s0, s1, and the individual's condition. Due to the fact that this linear predictor can output a negative value and the way use the survival rate requires it to be a positive value, we must exponentiate the value s when finding sr, the survival rate. To get the survival rate we must calculate probability with the linear predictor, we do this by using a logit-model, taking the exponentiated s and dividing it by the sum of 1 and another exponentiated s. This gives us the new survival rate for the given year, as it is reset each year of the life cycle. The survival rate is then used to simulate whether or not the individual has survived, this can be viewed in the newly updated alive variable. This is achieved by drawing one number with a binomial distribution between 0 and 1 with the success rate of the survival rate, essentially telling us whether or not the individual survived the year based on that newly calculated rate. Next we check with a simple if statement with the conditional being that if the individual did survive we must increase its age by one, as it has grown by another year.

```
# one part of a linear predictor
b0 <- -10
# second part of a linear predictor
b1 <- 0.1
# linear predictor
br <- b0+b1*co

# probability of breeding
b <- exp(br)/(1+exp(br))
# did the individual give birth, 0: no, 1: yes
calf <- rbinom(1,1, b)
# did the calf survive, 0: no, 1: yes
cdr <- rbinom(1, 1, .8)
```

Figure 3: Calculate variables related to calf birth and survival

In Figure 3, we show the updating and defining of variables related to whether or not the individual gives birth, and if the calf survives. I start by defining another linear predictor using the values in b0 and b1 as well as the condition of the individual to calculate br, the linear predictor. Then we must calculate b, the probability of breeding, in other words the probability the individual will have a calf. So as we did in sr, we exponentiate b due to the fact that we can only use a positive value as a probability and use a logit-model and calculate the probability, storing it in b. After that we simulate whether or not the individual has given birth by using a Bernoulli trial, in other words a random binomial trial with two distinct outcomes, 0 or 1, with a probability of success, b. Then we use this value to update the previously defined calf value to either 1 or 0, if it is one then the individual has successfully given birth this year.

After the calf is born, it must be simulated whether or not it will survive the year. We are able to do this because according Matthiopoulos, it is known that within the first five years of the calf's life it has a 20% chance of death (2011, p. 1). So we define this chance of death as cdr, where it will choose between 0 and 1 with a success rate of .8, the inverse of the mortality rate of the calves. If cdr returns 0, the calf will not survive this year, but if it returns 1 the calf will survive. Now it is possible to use further information along with these variables to simulate the lives of the offsprings and keep track of which make it to recruitment.

```
# check if the individual has had a new calf
if (calf == 1 && calfage == 0 && cdr == 1) {

  # increment calf age by 1
  calfage <- calfage + 1
  # negative effect on condition
  inv <- 10
  # decrement the individual's condition
  co <- co - inv

} else if (calfage > 0 && calfage < 5 && cdr == 1) { # check if the individual already has a calf and it survives the year

  # increment calf age by 1
  calfage <- calfage + 1
  # get negative effect on condition
  inv <- 10 * calfage
  # decrement the individual's condition
  co <- co - inv

} else {  # else the calf has died

  #reset calf and calf age
  calf <- 0
  calfage <- 0
}
```

Figure 4: Conditional statement to determine the status of the individual's offspring

This compoint if statement is split into three chained statements. This means the second will only ever be called if the first is found false and the third will only ever be called if both the first and second are found to be false. The first conditional is to check if the individual has had a new calf this year. It first checks if a birth has taken place by checking if calf is equal to one. The double ampersand in an important operator because it signifies and, as in condition one and condition two and condition three all must be true for this conditional to return true. After the first and operator we check if calfage is equal to zero, this indicates that the individual does not already have a calf. This is important to check because if calf is 1 and calf age is greater than 0, it does not matter that the event of the individual having a calf is true, because the orca does not breed if it currently has a calf who is not weaned. Then we must check cdr to see if the calf has survived, and if all of this is true, there is a birth, the calf's age is not greater than 0, and the calf has survived the year, then this if statement's conditions are met and we can move on to compute its contents.

First we must increase the calfage by one as the calf has survived a year. Then we set inv, the annual maternal investment by which we will decrease the individual's condition. Since the calf is new, the investment is just 10. After that the investment is removed from the condition, by subtracting inv from co. Once these tasks are completed the script moves on from the compound conditional and finishes out the life cycle loop.

If the conditions are not met it moved down the next conditional statement. You will notice that in this statement we do not check calf, because if these conditions are met it means the individual already has a calf and as mentioned previously it does not matter if the individual has the ability to give birth because it will not if it currently has an unweaned calf. To start it off we check if calfage is greater than 0, making sure that the calf is indeed not a new calf. We must check this because without it if an individual does not have a calf but cdr is equal to one the condition would be true, therefore checking that the calf age is at least 1 ensures that the individual has an unweaned living calf. After that calfage must be checked again to make sure it is not above 5, which may seem a bit redundant as we move into the next section of the loop, but is in place as a second level of defense to catch any calf that proceeds beyond age 5 because this would indicate a defect in the code and trials. After we finish checking that the age is within the appropriate range we must again check if the calf survived the year. If these conditions are met we run the contents, which are the same as the previous statement with one exception, the investment is now multiplied by the calfage to calculate

the correct investment as it is cumulative per year the calf is alive.

If the previous two conditions are unmet the else statement is hit. This statement means that the calf has died, essentially that cdr has been set to 0. When this is triggered the contents are evaluated. It resets both the calfage and calf to 0. This then allows the loop to finish and the individual can try again in the next simulated year.

```
# check if the calf has weaned
if (calfage == 5 && cdr == 1) {

  # add one to the count of weaned offspring
  offspring <- offspring +1
  # check if the offspring made it to recruitment, 0: no, 1: yes
  recruit <- rbinom(1, 1, .98)
  # add to this individual's recruit count
  recruitCount <- recruitCount + recruit

  #reset calf and calf age
  calf <- 0
  calfage <- 0

}
```

Figure 5: Independent conditional to check if the calf has weaned

After the previous compound conditional statement is complete to see if the status of the individual's calf, the simulation checks an independent conditional. It is necessary that this conditional is outside of the compound conditional because where each of the statements of it do not get checked each year depending on the previous conditions being met, this one must be checked each year. This statement is to see if the calf has been weaned and to see if it reaches recruitment. We start off by checking if calfage is 5 because as Matthiopoulos states, the calf will become independent at age 5 (2011, p. 1). Since this is separate from the compound conditional, it allows for the evaluation to take place the same year the calf turns 5 rather than checking in the following year, this is also why it is important this conditional statement is below the previous one in the script. Next we check cdr to make sure the calf is alive, again this may seem redundant, but it is a precaution to prevent defects in the data. If these conditions are met, the contents of the statement are evaluated.

First we update the offspring variable, by adding 1 to the previou count, representing the total number of weaned calves this individual has produced. After that we must find out if this weaned calf will survive until recruitment. Using the assumption that calves survive until 98% of the times

after weaning (Matthiopoulos 2011, p. 1). Then I define recruit which randomly selects between 0 and 1 with a success rate of .98, where 1 is survival and 0 is deceased. After that I add the value of recruit to recruitCount, which is keeping track of all the recruits this individual has produced. After this it resets calf and calfage back to 0 allowing the individual to give birth to a new calf the following year.

```
# check if the calf has weaned
if (calfage == 5 && cdr == 1) {

  # sex of calf, 0: female, 1: male
  sex <- rbinom(1, 1, .5)
  # add one to the count of weaned offspring
  offspring <- offspring +1
  # check if the offspring made it to recruitment, 0: no, 1: yes
  recruit <- rbinom(1, 1, .98)
  # add to this individual's recruit count
  recruitCount <- recruitCount + ((recruit + sex) %% 2)

  #reset calf and calf age
  calf <- 0
  calfage <- 0

}
```

Figure 6: Edited conditional of a weaned calf to take sex into account

You may notice the code is making no distinction between male and female calves, this is due to the fact that both male and female reach sexual maturity at the age of 15 (Matthiopoulos 2011, p. 1). If one wanted to look specifically at one sex, all they would have to do is add another variable like in Figure 6. This shows an added variable called sex which determines the sex of the calf, under the assumption that there is a 50% chance the calf will be male and 50% chance it will be female, where 0 is female and 1 is male.

With this new piece of information some simple math can be done to see if it is a female recruit. In recuitCount, instead of simply adding the new recruit value, we add the sex value to it, in the case of looking for female it would be 0, resulting in 1. Then we take this value and perform the modulo operation on it with the value 2, and 1 mod 2 evaluates to one, leaving us with an increased recruit count when the recruit is female. Whereas if the recruit was male the computed value would be 0 due to the fact that 2 mod 2 is 0. To change this to only look at male recruits it would be as simple as changing it from adding the sex and recruit values to subtracting them. As -1 mod 2 is 1, which would be the case of male recruits and 0 mod 2 is 0, which is the case for female recruits.

```
# add the amount of recruits this individual has made to the overall list
recruits <- c(recruits, recruitCount)
# add this individuals age at end of breeding to the overall list
ages <- c(ages, age)
```

Figure 7: The last piece of code within the trial loop and outside the life cycle loop

Once the life cycle while loop is completed the final code outside of it is evaluated. These lines as seen in Figure 7, are updating the two empty lists we initialized at the beginning of the script. First recruits is called and the recruitCount is appended to the end of the previous recruits list, updating it to contain the value for the most recent individual. Similarly, ages is updated with the age of when the most recent individual stopped breeding. After these two lines are evaluated this trial is over and the while loop that keeps track of the number of trials is checked again and the it is either evaluated again or the code moves on to run the lines after this loop.

```
# average number of recruits
avgRecruits <- (cumsum(recruits)[1000] / 1000)
```

Figure 8: Finding the average number of recruits produced

To find the first result that is being calculated by the script, it must run the line of code seen in Figure 8. This computes the average number of recruits the females have and stores it in the variable avgRecruits. The computation happens by finding the cumulative sum of the list of recruits, then taking the last element of that list at index 1000 and dividing it by 1000 to find the average.

```
#########################################################################################################
# plot histograms

# histogram for ages at the end of breeding
hist(ages,
     main="Ages at End of Breeding",
     xlab="Age",
     border="black",
     ylim=c(0,1000),
     col="lightblue",
     breaks=25,
     las=1)

# histogram for inclusive fitness
hist(recruits,
     main="Number of Inclusive Fitness",
     xlab="Recruits",
     border="black",
     ylim=c(0,1000),
     col="lightgreen",
     las=1)
```

Figure 9: Section of the code which draws the histograms

After we find the average number of recruits reproduce, there is a divider to indicate where the section of plotting begins as displayed in Figure 9. The first histogram that is plotted displays the ages at the end of breeding. The second histogram displays inclusive fitness, or in other words the amount of recruits a female has produced. Both histograms are constructed in the same way, where the first parameter in the hist function is the data to be displayed on the graph. After that main is set to the desired title of the plot and xlab is set to the desired label of the x axis. The border is set the color that is to be the border of the bars. Then ylim is set to the y axis limit, in both cases (0, 1000), representing that the y axis should display from value 0 to value 1000. After that col is set to the desired color of the bars and finally we set las, which controls the rotation of the y axis labels, ensuring they are facing upwards rather than sideways. In the histogram for ages I also include breaks, which controls how many bars the plot should have and it is set to 25 so each year between 15 and 40 will have an individual bar.

3. Results

The results are split up into four scenarios which can be found below with corresponding histograms per scenario.

Scenario 1: Rich and Stable Environment

To simulate an environment that is both rich and stable we set the environment mean, enMu to 20, and the environment standard deviation, enSD to 1. After this we run the full script we find the average number of recruits produced to be 1.558.

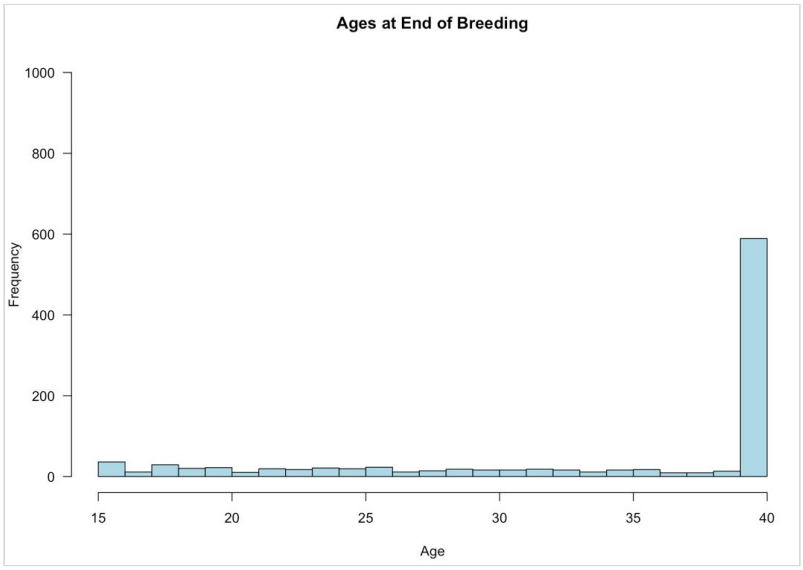

Figure 10: Age histogram for scenario 1

By looking at Figure 10, it is obvious that the majority of the individuals end breeding at age 40. There is a pretty constant spread between the ages between 15 and 39, with it slightly decreasing as the age approaches 39.

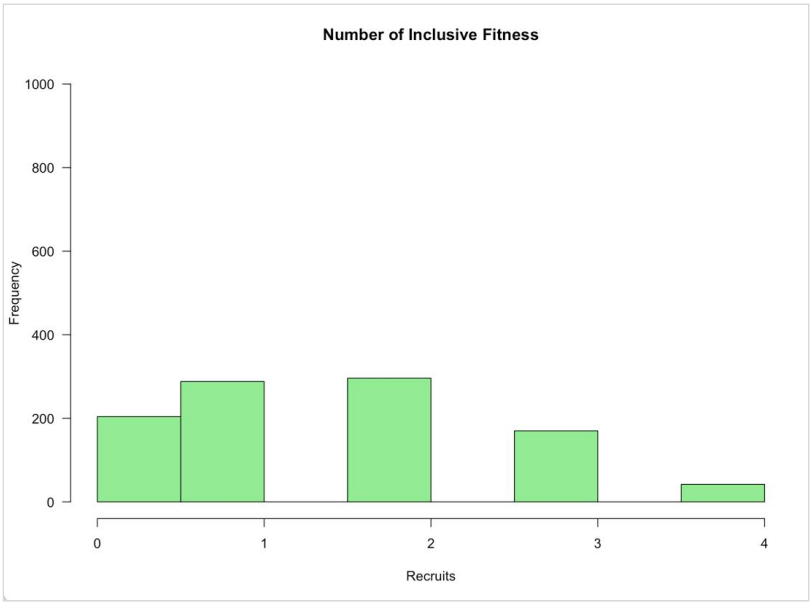

Figure 11: Inclusive fitness histogram for scenario 1

Inspecting Figure 11 shows the breakdown of the distribution of recruits. More individuals are producing 1 or 2 recruits than individuals producing 0 recruits. The amount of individuals

producing three recruits nearly matches the amount of individuals producing no recruits and the amount producing 4 recruits is not negligible, as it looks to be somewhere between 5-10% of the total individuals.

Scenario 2: Poor and Stable Environment

The next scenario is simulated as a poor and stable environment by setting enMu to 2 and enSD to 1. Then the script is run, subsequently the average number of recruits produced is found to be 0.938.

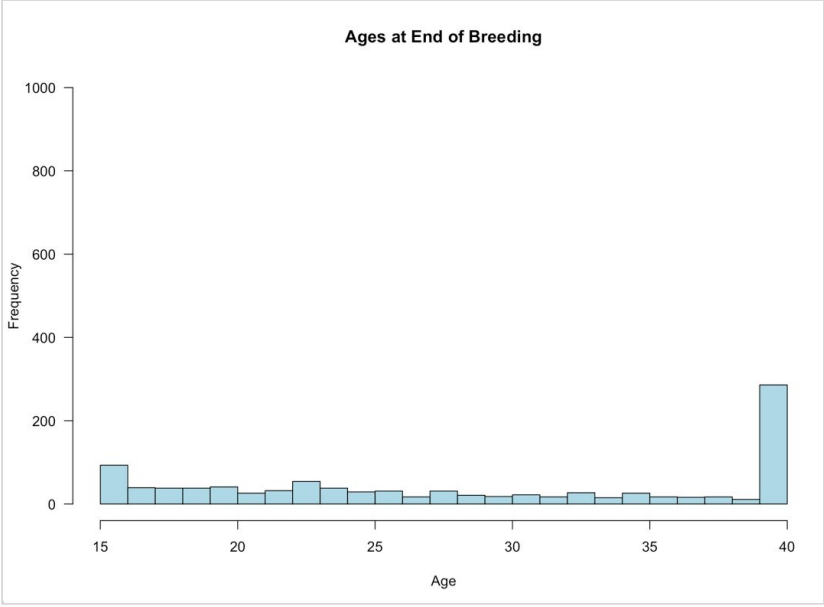

Figure 12: Age histogram for scenario two

In Figure 12 we can see a that the age most individuals stop breeding is 40. The second most common age which is 15, which seems to be about one third of the size of the bar representing 40. The rest of the ages look to follow a negative linear trend as they go from 16 to 39.

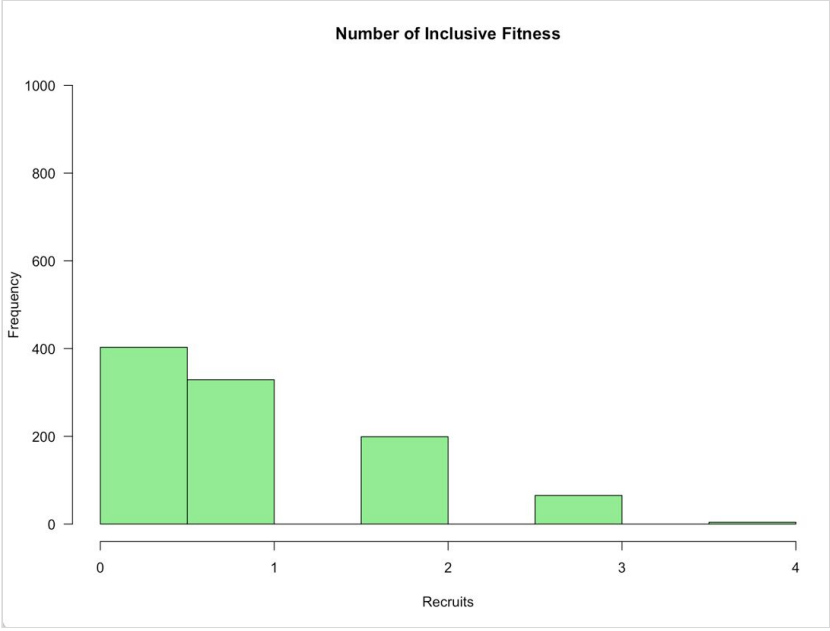

Figure 13: Inclusive fitness histogram for scenario two

As seen in figure 13 the majority of individuals do not produce any recruits. This is closely followed by the amount of individuals who produced one recruit. Then individuals who produced more follow a strong linear negative trend, to the point where the amount who produced 4 recruits seems to be insignificantly small.

Scenario 3: Rich and Variable Environment

The rich and variable environment is simulated by setting enMu to 20 and enSD to 30 and then running the script. This results in the average number of recruits produced as 1.032

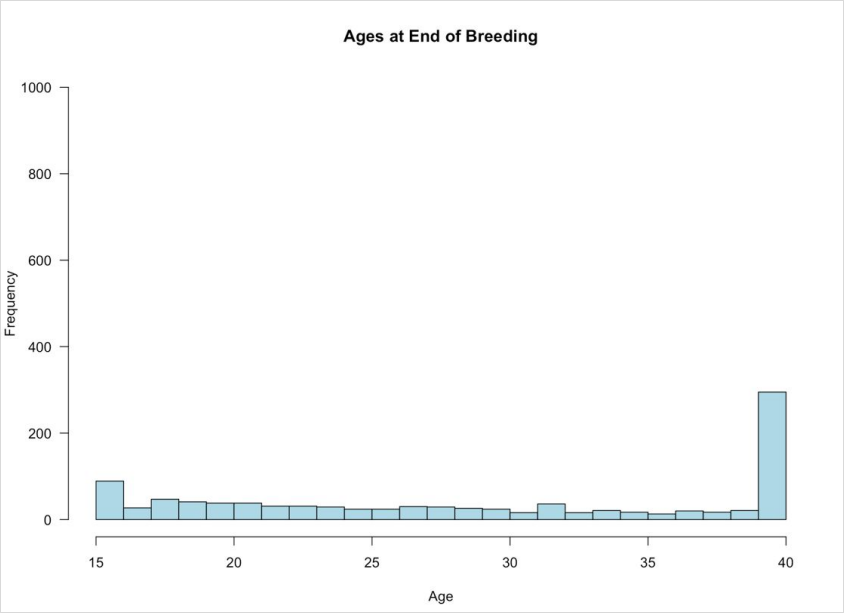

Figure 14: Age histogram for scenario three

When looking at figure 14 it is easy to see the majority of individuals conclude breeding at age 40. Other than that there is a pretty distinct negative linear trend from 15 to 39.

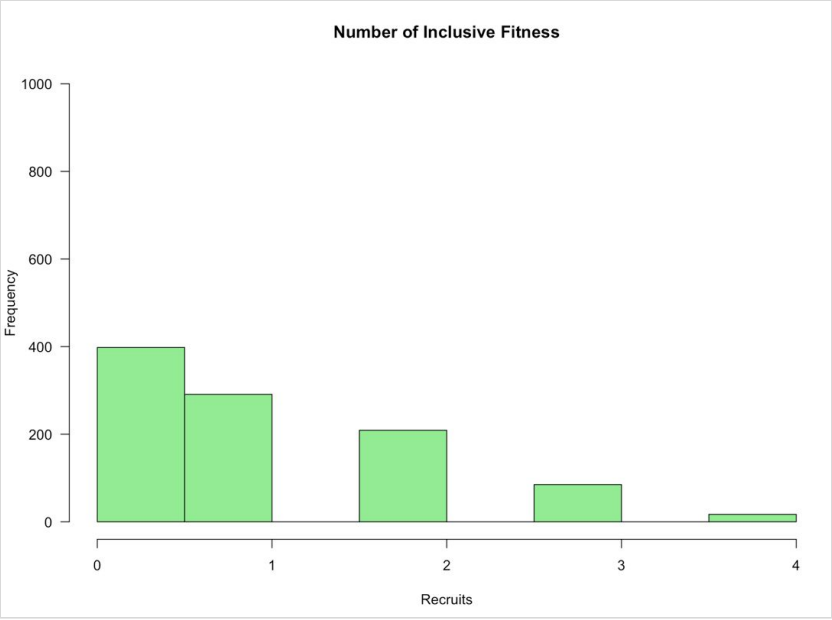

Figure 15: Inclusive fitness histogram for scenario three

Figure 15 clearly shows that the majority of individuals produce 0 recruits with a very strong linear negative trend from 0 to 4.

Scenario 4: Poor and Variable Environment

The fourth scenario is of an environment that is poor and variable. This means setting enMu to 2 and enSD to 30. After running the simulation the average number of recruits produced is 0.573.

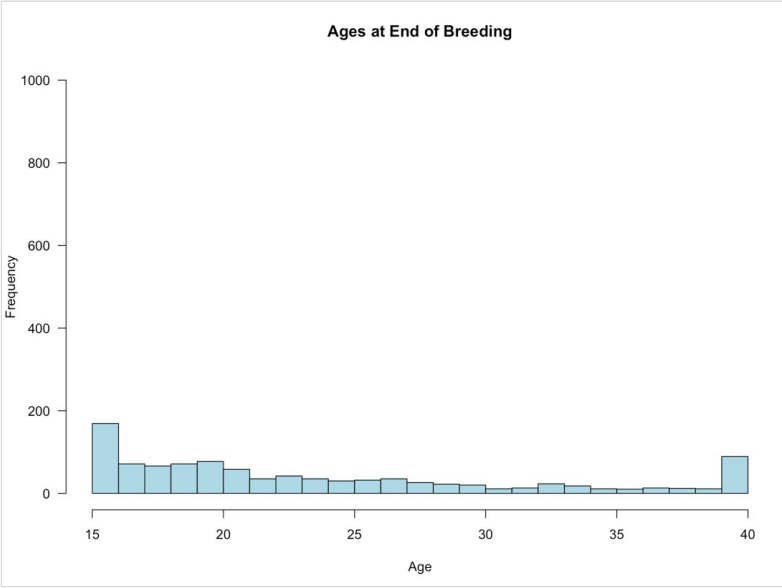

Figure 16: Age histogram for scenario four

One can easily discern from Figure 16 that most individuals end breeding at 15. Between 16 and 39 there seems to be a negative linear trend until 40 where there is a significant positive increase.

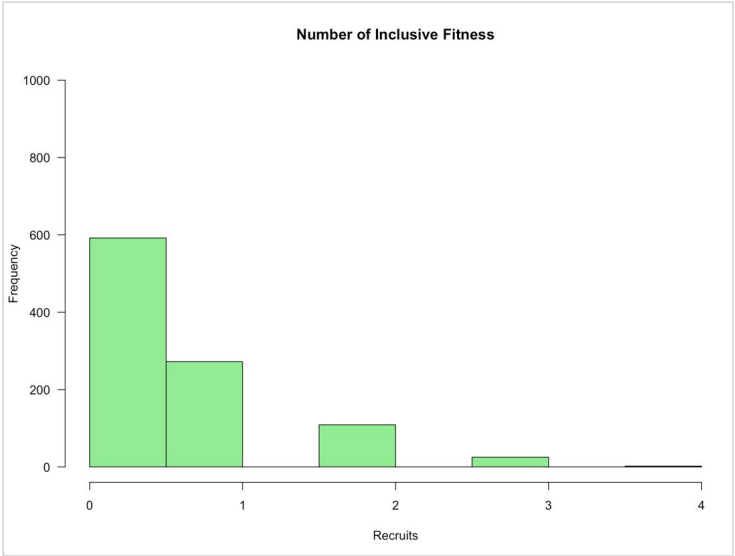

Figure 17: Inclusive fitness histogram for scenario four

By looking at Figure 17 one can see a significant majority of individuals have produced 0 recruits. The graph seems to follow a negative exponential trend, to the point of where individuals who produced 4 recruits seems to be nearly 0.

4. Discussion

As the results have shown, there is a relationship between environmental factors and inclusive fitness. Starting off I am making the classification that rich is a positive factor and stable is also a positive factor. Therefore if the environment is poor, that is negative, as well as if it is variable that is negative. Looking at scenario one, both of the environmental factors richness and stability, enMu and enSD are positive; the environment is both rich and stable. As a direct outcome of this we see that a majority of the orcas are producing some amount of recruits, as the bar for zero is the smallest in Figure 11, and the smallest of any of the inclusive fitness histograms. While looking at Figure 10, we see the majority of the individuals stop breeding at age 40, the natural end of breeding for the species. This scenario has the highest number of individuals coming to a natural stop in breeding out of the four.

When looking at both scenarios two and three, we notice their histograms are very similar. Comparing Figure 12 and Figure 14, they unquestionably resemble each other. Furthermore when comparing Figure 13 and Figure 15 they are nearly identical, with some discrepancy with the bars for 1 and 4. Further inspecting the environmental factors that are controlling these two scenarios we see each one has a positive factor and a negative factor. Scenario two is poor and stable and scenario three is the inverse of that, rich and variable. What can be deduced from this is that the fact that these factors are almost interchangeable. As long as one of the factors is positive and the other is negative, the will give you similar results, this is also backed up by how close the average recruits each scenario produced are in number: 0.938 for scenario two and 1.032 for scenario three.

Upon reviewing scenario four we see the most negative histograms and average recruits per individual. This falls in line with the idea that the ratio between positive and negative factors weighs in more on the outcome rather than the actual factor itself. Both of this scenarios factors are negative, it is poor and variable. This leads to the smallest amount of individuals making it to the natural end of breeding at age 40 out of any scenarios as seen in Figure 16. It is also the scenario

that has the most individuals ending breeding at age 15, signifying that they do not even survive the first year of being able to reproduce. This is also the first scenario where the majority of the individuals produce zero recruits, and has the most extreme negative trend as displayed in Figure 17. Coincidently it has the lowest average recruits produced per individual at 0.537.

Another thing to take into consideration is the way in which this study defines a life cycle. A life cycle is ended at the age when an orca can no longer reproduce or when it dies prior to reaching that age. Therefore every individual who survives to be at least 40 years of age stops breeding at the natural stopping point and the age recorded is 40. This leads to a graph with a spike at the end to represent the natural end of the individual's ability to give birth. To improve the study one could look into defining the age at the end of breeding in a more concrete way, such as the age at which the orca had its last calf. Then the question is, even an individual reaches age 40, if it did give birth to another calf after age 25, then do these years after 25 still count as time breeding? This could add another layer of specificity to the data the is taken away from the simulations.

Expanding on the life cycle, allowing the loop to go beyond 40 and see the age at which individuals die would be interesting complementary data. When experimenting with the script and allowing this to happen there seemed to be many cases especially in scenario one of orcas living anywhere from 100 to 500 years of age, whereas the natural maximum is around 80 in exceptional cases (Matthiopoulos 2011, p. 1). This happens because there is not adequate information on the yearly effect of growing old on the individual's condition, so when they die it is purely by the same survival rate at age 400 as it would be at age 20. If the necessary data was added to the script to decrement the individual's condition with age it would be interesting to see how the scenarios affect overall life span.

As stated in the methods the simulations do not take offspring sex into account. The code I provide if one would like to delineate the sexes of the calves functions under the assumption that the sex ratio of birth is 50% male and 50% female. However a study found within a certain population of orcas that were monitored over 26 years, that from the 65 recorded sexed births the ratio skewed towards male offspring, with 57% being male, leaving the other 43% female (Taylor 2000, p. 11). If one wanted to get more accurate results for sexed births in the simulations based on this report one would have to replace the success rate in the variable sex as seen in Figure 6 with 0.57.

An interesting area for future research would be to look at specific orca populations in the world to see if the scenarios simulated in this report match up with real life concrete data.

References

Matthiopoulos, Jason (2011) *How to be a Quantitative Ecologist*, Hoboken, NJ: Wiley.

National Geographic (2017) Orca, Available from: <http://www.nationalgeographic.com/animals/mammals/o/orca/>; [Accessed 31 August 2017]

Taylor, Martin. "Population viability analysis for the southern resident population of the killer whale (Orcinus orca)" *Tucson, Ariz. : Center for Biological Diversity* (2000): 11.