Matriculation number:						

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: 31.08.2018, 11:59 pm

Study program	Begin of studies	Last name, First name
Information Engineering and Computer Science (M.Sc.)		

Assessment criteria and number of points that can be achieved:

Maximum number of points	Skills and Expertise	Systematic and scientific Quality	Quality of the results	Presentation of the results
100	45	15	30	10

Result:

Points	Mark	Skills and Expertise	Systematic and scientific Quality	Quality of the results	Presentation of the results

The assessment is only open to students who are enrolled in the study course "Information Engineering and Computer Science (M.Sc.)" and have successfully registered for the exam in Data Analysis and Statistics at the end of the summer semester 2018.

The assessment consists of the assignment that is given in this document. It involves a practical task and the subsequent preparation of a scientific report. Grading will be based on both parts. Carefully read the whole document before you start working on the assignment!

(Sections 1 and 2 have been adapted from Jason Matthiopoulos, 2011)

1. Introduction

Life history theory, a branch of evolutionary ecology, seeks to explain how natural selection and other evolutionary forces shape organisms to optimize their survival and reproduction in the face of ecological challenges posed by the environment (Stearns 1992, Roff 1992, Stearns 2000). Its principal aim is to explain the diversity in life histories among species, i.e. it is about the variations that organisms exhibit with respect to the sequence of events in their lives related to survival and reproduction that occur from birth through death.

It is based on the idea that for organisms' resources like energy and nutrition are limited and have to be split up among processes or events like growth, reproduction and health. Through natural selection marked variations in the way how this allocation problem is solved have evolved. Variations include breeding frequency, the age at which reproduction begins, the number of times an individual reproduces during its lifetime, the number of offspring produced at each reproductive episode (clutch or litter size), the ratio of male to female offspring produced, and whether reproduction is sexual or asexual. These differences in life history characteristics can have profound effects on the reproductive success of individuals and the dynamics, ecology, and evolution of populations.

The complex interactions between life history traits and the environment may favor or disadvantage single individuals or entire populations. These effects are difficult to intuit and their investigation may require the construction of stochastic, individual-based models. Here, we construct such a simulation, loosely based around the life history of female killer whales. We examine the implications on inclusive fitness of four different scenarios of resource richness and environmental variability.

.

1.1 The Study System

The Killer Whale or Orca (Orcinus orca), is the largest species of the dolphin family. It is found in all the world's oceans, from the frigid Arctic and Antarctic regions to warm, tropical seas. Some killer whale populations feed mostly on fish while others hunt sharks, marine mammals, including sea lions, seals, walruses and even large whales and Great white sharks.

There are up to five distinct killer whale types distinguished by geographical range, preferred prey items and physical appearance. Some of these may be separate races, subspecies or even species. Killer whales are highly social; some populations are composed of matrilineal family groups, which are the most stable of any animal species. Their sophisticated social behavior, hunting techniques, and vocal behavior have been described as manifestations of culture.

Females become reproductively active at the age of 15 and breed until they are 40. They have periods of polyestrous cycling with non-cycling periods of between three and sixteen

months. The gestation period varies from fifteen to 18 months. Calves nurse for about 2 years after birth, but remain closely associated with their mother for about four years. Mothers maintain a level of parental investment for that period, leading to an inter-birth interval of five years. During those years, calves suffer from an annual mortality of 20%. After their fifth year, female calves survive to recruitment with a probability of 0.98.

Females' life spans average 50 but may survive well into their 70-80s in exceptional cases. Males become sexually mature at the age of 15 but do not typically reproduce until age 21. Male killer whales generally do not live as long as females. In the wild, males average 30 years, with a maximum of 50–60 years in exceptional cases.

1.2 Ecological Theory

The interplay between life history strategies and environmental variation has the potential for rich dynamics. Life history traits evolve as a result of persistent environmental drivers but, in the short-term, the fitness of individuals and the fate of populations depends on environmental trends and the stochasticity around them. As scientists, we will rarely be able to predict a-priori which aspects of a life history strategy will be beneficial for an animal living in a particular environmental regime. Simulation is an invaluable tool for achieving this.

1.3 Stochastic Simulation - Modeling Random Phenomena

Many simple (random) phenomena can be modeled using named probability distributions. The approach is to identify a distribution whose assumptions are satisfied by the research question and then obtain the values for the parameters of that distribution. But what if the parameters are themselves random variables? For example, given a cohort of P_0 animals, a binomial distribution might be used to decide how many will be alive one month from today:

$$P_0 \sim B(P_0, p)$$

where p is the per capita probability of survival.

For any given month t, the survivors can be modeled as

$$P_{t+1} \sim B(P_1, p)$$

but any one of the two parameters of the distribution can be random. Certainly P_1, because it depends on the number of survivors from previous months, but also p, which may vary from month to month due to chance environmental effects.

When the parameters of one random variable have their own distribution then the variable is called compound. The distributions of some compound variables are tractable mathematically, but in most cases they are not. Then the only resource is stochastic simulation, i.e. the implementation of the system in a computer using random draws from the constituent distributions of the compound variable. Repeatedly running the simulation to obtain a large number of final outcomes leads to an empirical description of the compound distribution.

1.4 Mathematical Theory - Formalizing dependence on covariates

Quantifying the effects of particular causes is the stock-in-trade of most empirical scientists. Coming up with mathematical expressions for these relationships can be achieved either by mechanistic or empirical means. A mathematical model is called mechanistic when it has been derived from first principles. So, for example, a model for the global human population that takes into account births (by country) and deaths (by cause) is mechanistic at the level of demography, because it acknowledges the proximate causes of population change. In contrast, when we fit an exponential curve through the population data, we are using an empirical model, i.e. one that has the "right sort of form".

At their very simplest, empirical models can be defined on the basis of a function's domain, range and monotonicity. In other words, what kind of values a function takes as its input, what sort of values it produces as its output and whether it is purely increasing or decreasing. For example, if we know that the amount of weight (w) gained/lost by an individual organism is a simple, increasing function of the availability of two types of food (1 x and 2 x), we might express this with the empirical relationship

$$w = a_0 + a_1x_1 + a_2x_2 \tag{1}$$

Here, we have used a linear relationship in which w can take both negative and positive values, the intercept a_0 could be thought of as a negative value that has some association with basal metabolic costs and the coefficients a_1 and a_2 give the relationship between weight gain and food availability for each of the two types of food.

In certain cases, the response variable can only take non-negative values. For example, we might be interested in the average number of offspring (b) that can be produced by an animal as a function of food availability. Then, we might consider a log-linear model, in which the linear predictor $a_0 + a_1x_1 + a_2x_2$ is exponentiated,

$$b = \exp(a_0 + a_1x_1 + a_2x_2) \tag{2}$$

In other cases yet, we might be interested in modelling a probability or a proportion. For example, what is the annual probability of survival (p) of an animal as a function of food availability? This may be formalized by a so-called logit model which looks like this

$$p = \frac{\exp(a_0 + a_1x_1 + a_2x_2)}{1 + \exp(a_0 + a_1x_1 + a_2x_2)} \tag{3}$$

If you try very large and very small values for the variables 1 x and 2 x , you will notice that this function is constrained to take values between 0 and 1, which is what we require of a probability.

1.5 R Ingredients

Generating random numbers: You first need to decide which distribution you want these random numbers to come from (uniform, normal, binomial, Poisson etc). Then, you need to

specify how many numbers you want out of that distribution and finally, you need to specify the parameters of the distribution. Here are some useful examples:

R command	*Explanation*
`runif(2, 1,2)`	Generates two random numbers between 1 and 2 from the continuous uniform distribution.
`rnorm(100, 0, 10)`	Generates 100 draws from a normal distribution with mean zero and standard deviation 10.
`rpois(10, 5.3)`	Generates ten random numbers from a Poisson distribution with rate 5.3
`rbinom(10,1,0.5)`	Draws ten random numbers from a binomial distribution with 1 trial and a probability of success 0.5.
`rbinom(1,10,0.5)`	Draws one random number from a binomial distribution with 10 trials and a probability of success 0.5.
`rbinom(1,1,0.8)`	In order to determine if a random person survives a year, given the probability of survival is 0.8, draw one random number from a binomial distribution with 1 trial and a probability of success 0.8.
`x + rnorm(1,0,20)`	In order to determine a random increment for a variable x, given the increment is normally distributed with mean 0 and standard deviation 20, draw a random number from a normal distribution with mean zero and standard deviation 20 and add it to variable x.

Testing facts: Sometimes it is useful to test whether a fact is true or false. You may put the question to R as a mathematical statement. For example, you may ask if 2 is bigger than 3,

```
> 2>3
[1] FALSE
```

Equality is tested with the symbol `==`,

```
> 2==2
[1] TRUE
```

We can test the validity of combined statements by bolting together different questions with the AND, OR operators. In R, these are written `&&` and `||` respectively. Let's have a look at some examples:

Plain-language question	*R-version*	*R-response*
Is either of the following true? 1+1=2, 2+2=3	`1+1==2 || 2+2==3`	**TRUE**
Are both of the following true? 1+1=2, 2+2=3	`1+1==2 && 2+2=3`	**FALSE**

Conditional statements: Conditional statements are useful when you want to execute a set of commands, but only when a certain set of conditions hold. So, the sort of thing you might like to say in plain language is "*If a set of conditions is satisfied then do something* ". The programming structure `if` can be used for this purpose. Here is what it looks like in general

```
If (conditions)
{
Commands
}
```

The following example will first examine if the random number `r` is positive and then print a statement to say so.

```
r<-rnorm(1,0,1)
if (r>0)
{
print("positive")
}
```

There is an extension to this conditional structure leading to **compound conditional statements** which loosely translates to "*If a set of conditions is satisfied then do something, if they are not, then do something else*". The following example will report whether the random number `r` is positive or negative.

```
r<-rnorm(1,0,1)
if (r>0)
{
print("positive")
} else {
print("negative")
}
```

Iteration and loops: **Iteration** is the repeated application of a set of commands. In most occasions we can perform tasks iteratively by using the vectorisation capabilities of R. Sometimes, iterative tasks require a different programming device, called a **loop**, which comprises of 1) a loop declaration and 2) the main body of the loop. Here, we will introduce two types of loops, the `for` loop, performs a set of tasks for a predefined number of iterations. The `while` loop performs a set of tasks while a set of conditions hold.

In R, a typical `for` loop will look a bit like this:

```
for (counter in min:max) #  This line declares the loop counter and range.
      {
      #  The main body of loop - the commands to be repeated.
      }
```

For example, the following will produce the first 21 terms of the sequence $a_{t+1}=100+35a_t$

```
1 a<-c()
2 for (n in 0:20)
3 {
4 a<-c(a, 100+n*35)
5}
```

Here, line 1 creates an empty list `a`. Line 2 declares the loop i.e. it defines a counter `n` that will successively take the values 0 (min) to 20 (max). The brackets in lines 3 and 5 enclose the main body of the loop, the parts that are to be iterated. Several lines of code can be enclosed in the main body but, in this example, we only have one, line 4, which uses the concatenation command `c()`, to add to the current list `a`, the new value `100+n*35`. Since this value contains the counter `n`, it will be calculated anew with each iteration of the loop. To see how the loop operates, try inserting the command `print(a)` as a new line in-between lines 4 and 5. Also, note that the brackets and main body of the loop is

typeset towards the right. This makes it easier to identify on the screen or printed page where the loop begins and where it ends.

In R, a typical `while` loop will look a bit like this:

```
while (conditions) # This line declares the loop
conditions.
      {
      #  The main body of loop - the commands to be repeated.
      }
```

We can achieve the same result with a `while` loop as with a `for` loop. Here is an example

```
1 n<-0
2 a<-c()
3 while (n<=20)
4 {
5 a<-c(a, 100+n*35)
6 n<-n+1
7}
```

Can you see how this works? A counter is initialised outside of the loop and incremented by one inside the loop. When the counter reaches a pre-determined size, then the loop ends. In this example, the for implementation was more economical than the while version. However, the while loop, really comes into its own when the number of iterations is unknown. In the following example, we toss a fair coin as many times as are required to get 20 heads. At the end, the program reports the number of trials that were conducted. You will notice that this number may change if you re-run the program.

```
1 heads<-0
2 trials<-0
3 while (heads<20)
4 {
5 heads<-heads+rbinom(1,1,0.5)
6 trials<-trials+1
7 }
8 trials
```

Setting out initial conditions and parameters: It is generally good practice to represent parameters by symbols whose numerical assignments are collected together at the beginning of the code, separately from its main body. This is for three reasons: 1) it gives an overview of the kind of numerical information required for the model, 2) the statement of the model in the main body of the code is more reminiscent of its mathematical description which makes it easier to find mistakes and 3) in many models the same parameter is repeated several times. Rather than having to trawl through the entire code and change the parameter's numerical value in all these instances, it is only necessary to change its numerical assignment at the beginning.

1.6 References

Jason Matthiopoulos, How to be quantitative ecologist, Wiley 2011 .

Roff, D. A. *The Evolution of Life Histories. Theory and Analysis*. New York: Chapman and Hall, 1992.

Stearns, S. C. *The evolution of life histories*. Oxford: Oxford University Press, 1992.

Stearns, S. C. Life history evolution: successes, limitations, and prospects. *Naturwissenschaften* **87**, 476-486 (2000).

2. Practical Tasks

You will investigate how the availability and variability of resources affect the individual fitness, the total reproductive output and the age of female Orcas at death. The investigation seeks to support a better understanding of the effect of the environment on both the fitness of an individual Orca and the reproductive ability of an entire Orca population.

The investigation will be done by means of a simulation programmed in R. You will gradually build up and then run a simulation of the reproductive life histories of thousands of female Orcas that allows you to generate data that can be analyzed with respect to the given research objective.

A) Read carefully the material in sections 1- 4 above.

B) Read through the tasks in this section.

C) Use the numbered list below (*List 1: Building the simulation - Instructions*) to step by step build and run the simulation in R. The list consists of different parts that should be done one after another. The first three sections will cover the modeling part:

- Modelling a single female without a calf
- Modelling a single female with a calf
- Modelling several females

The instructions in these sections will have to be translated into R code.

The code then will have to be run with different parameter sets in order to simulate different real world scenarios. The scenarios are covered in the following sections:

- Scenario 1: Rich and stable environment
- Scenario 2: Poor and stable environment
- Scenario 3: Rich and variable environment
- Scenario 4: Poor and variable environment

Run each scenario at least once.

D) **Visualize the generated data appropriately and obtain suitable statistics by means of R in order to describe and analyse the output of your simulation.**

E) **Read through the assessment components in the next section before you carry out any tasks.**

List 1: Building the simulation - Instructions:

Modelling a single female without a calf:

First create and initialize the global variables that will be used in the simulation (see also section 1.5, Setting out initial conditions and parameters):

1) Use the variable *co*, measured in some arbitrary unit, to track the condition of each individual female. At recruitment (i.e. when they become reproductively active), females have a normal distribution of conditions with mean 100 and standard deviation 10. Write a line of code that selects a random value for *co*, from such a distribution (*see section 1.5, Generating random numbers*).

2) Define the variable *age* to track the age of a single female. Set *age* to the appropriate value for a female that has just become recruited (*see section 1.1*).

3) The variable *alive* will take the value 1 if a female is still alive in any given year, and zero if it has died. Set it to 1, to begin with.

4) The quality and variability of the environment will determine the ability of an individual to improve its condition. The influence of the environment will be represented by the variables *enMu* and *enSD.* A high value of *enMu* will indicate a rich environment and a high value of *enSD* will indicate a variable environment. To begin with, set these values to 0 and 20 respectively.

5) The variable *sr* will represent the survival rate within the population of reproductive female whales (*i.e. probability of a randomly selected whale from that population of surviving a year*). To begin with, set *sr* to 0.8.

6) The variables *s0* and *s1* are used to describe the relationship between the condition of a whale and its probability of survival and hence will be used to model the influence of the whales condition on its probability of survival. Set *s0* to -2 and *s1* to 0.05.

Write a loop (see section 1.5, Iteration and loops) that allows to model survival and death of the whale. Each iteration will represent a year in the life of the female whale:

7) Write a loop that runs while a female is alive and of a reproductive age. Within each iteration of the loop, do the following two things:

   (i) allow the whale to survive with the probability *sr* (*see section 1.5, Generating random numbers*) and
   (ii) if it does, increment its *age* by 1.

   Run the loop and check the age-at-end-of-reproduction.

Add elements to that loop that are supposed to represent influences on the whales condition and probability of survival:

8) In step 4 we introduced variables (*enMu* and *enSD)* that would represent the influence of the environment on the condition of an individual whale without specifying how. We will make up for that now.

   The quality and variability of the environment will determine the ability of an individual to improve its condition during any given year. We assume that these annual condition increments are normally distributed and independent between successive years. *enMu* represents the mean and *enSD* represents the standard deviation of that distribution. Adjust the code in the loop in order to implement an annual increment of the individual condition accordingly (*see section 1.5, Generating random numbers*).

9) Let the condition of an individual affect its probability of survival. Do this by making the probability of survival *sr* depend on the following linear predictor:

   *s0+s1*co* (see step 6, *s0 = -2* and *s1 = 0.05*).

   See section 1.4 for more information. Notice that in this case, we are interested in modeling a probability (section 1.4 – equation (3)).

Modelling a single female with a calf:

The following instructions add elements to the code that has been created so far and make it a representation of a female with a calf. First create and initialize additional variables at the beginning of your code (see section 1.5, Setting out initial conditions and parameters):

10) Define the variable *calf* that can take the value 1, if a female currently has a calf and 0 if she doesn't. Also, define the variables *calfage* to track the age of a calf and *offspring* to track the number of calves that become independent of their mothers. Initialize all of these at zero, outside your main loop.

11) The variable *b* will represent the breeding probability of a reproductive female whales. To begin with, set *b* to 0.

12) The variables *b0* and *b1* are used to describe the relationship between the condition of a whale and its breeding probability and hence will be used to model the influence of the whales condition on its breeding probability. Set *b0* to -10 and *b1* to 0.1.

13) Add a variable *inv* that is supposed to represent effect of annual maternal investment on a female's condition. Set *inv* to 10.

Add elements to the loop that allow to represent influences on the whales breeding probability, breeding incidents, survival and death of a females calve:

14) Within your loop, write a compound conditional statement (*see section 1.5, Conditional statements*) that does the following:

   - If the female doesn't have a calf, then

- calculate its breeding probability *b* as a function that depends on the linear predictor

  b_0+b_1*co (see step 12, $b_0=-10$ and $b_1=0.1$)

  See section 1.4 for more information. Notice that in this case, we are interested in modeling a probability (section 1.4 – equation (3)).
- use this probability as part of a Bernoulli trial (i.e. a binomial experiment with one trial), to simulate the event of birth
- if a calf is born, then set *calf* to 1, otherwise leave it at zero

❖ Else, i.e. If the female has a calf, then
- decrement the female's condition by the amount *inv*, corresponding to annual maternal investment.
- decide randomly whether the calf is going to survive the year according to a fixed probability (*see section 1.1 and 1.5*)
- set *calf* to zero if calf has died
- set *calfage* to zero if calf has died, or increment it by one otherwise.

15) Within the loop, after the statement produced from step 14, write a simple conditional statement (*see section 1.5, Conditional statements*) that does the following:

❖ If the calf has reached the age of independence
- Increment the number of the mothers offspring
- reset *calf* and *calfage* to zero.

Run the entire code so far to make sure it works. Check out critical outputs such as age or offspring. For example, if your program generates more offspring than the final age of the female, then something is wrong. Go back over the code to find where the problem is.

Modelling several females:

In order to simulate a population and not only one random female add the following to the model that you created so far:

16) We will need to store the total number of offspring that reach recruitment from each female whale and the age at which females stop breeding. Create the empty lists *recruits* and *ages* for this purpose.

17) Use a loop to simulate 1000 female life histories, each time storing the female's final age and its total number of recruited offspring into the lists *ages* and *recruits* respectively. Note that the total number of recruited offspring is not the same as the number of offspring that become weaned.

Scenario 1: Rich and stable environment

18) We will interpret the number of recruits produced by each female as its **inclusive fitness**. Set the parameters of the environment to *enMu*=20 and *enSD*=1.

19) Generate a histogram of ages at end of reproduction

20) Generate a histogram of inclusive fitness

21) Calculate the average number of recruits produced by each female

Scenario 2: Poor and stable environment

22) Set *enMu*=2 and *enSD*=1 and repeat tasks 18-21.

Scenario 3: Rich and variable environment

23) Set *enMu*=20 and *enSD*=30 and repeat tasks 18-21.

Scenario 4: Poor and variable environment

24) Set *enMu*=2 and *enSD*=30 and repeat tasks 18-21.

Steps 1-24 of the list above are mandatory. Feel free to add whatever you consider reasonable with respect to the goals of the work and the method employed to achieve them!!!

3. Assessment

The results of the tasks given in section 2 – Practical Task - have to be compiled into a scientific report that is due on 31.08.2018, 11:59 pm! Your report should comprise the following elements:

- **Cover page**: The first page of this assignment paper has to be used.
- A signed statement of authorship. You can copy the following text into report and sign it:

  This report is the result of my own work. Material from the published or unpublished work of others, which is referred to in the report, is credited to the author in the text.

- Introduction to the overall subject of the report and the particular tasks covered in it (research question/s, motivation, goals, context, approach in brief).
- A description of the approach (in detail), the tools and methods you are going to use to solve the given tasks.
- A description of the results.
- An interpretation and discussion of your results and the methods used.
- A list of the references used in your report.
- Fully commented RStudio code

The report at the latest has to be turned in on Aug 31, 2018, 11:59 pm. A report that will not have turned in by then will automatically be graded as failed! **The date of the receipt applies!**

There are different options for the delivery of your report:

(1) Hand out a printed copy personally to Mr Ralf Darius (room 02 00 405).

(2) Post a printed copy to (**The date of the receipt applies!**):

Ralf Darius
Hochschule Rhein-Waal
Friedrich-Heinrich-Allee 25
D-47475 Kamp-Lintfort

Alternatively you can simply drop your report in the post office box on the left side of the entrance hall of building 02 when entering through the main entrance. Use the POB labelled “DARIUS”.

(3) Send a digital copy via Email to ralf.darius@hochschule-rhein-waal.de until Aug 31, 2018, 11:59 pm (**The date of the receipt applies!**).