Simulating Turtle Movements

Tati Micheletti

29 June 2025

Simulating Turtle Movements

The present game-exercise intends to give you a hands-on experience on how ecological modelers use data, statistics, and probability to forecast changes in our planet. To play the game, you were previously divided in three groups:
1. the **Turtle** group
2. the **Bird** group, and
3. the **Tree** group.

If you are reading this, you belong to the Turtle group, which we will rename to ***Habitat Selection Simulation*** group. Fancier, right? :)

You will be responsible for simulating movement of an animal species across the landscape and, ultimately, forecasting how their populations will change through time based on landscape changes.

You have received a small Lego set, which you can use to simulate the movements of your turtle population. This Lego set brings the aerial view of your landscape and you can see how the animal moves through different habitats through time.

1. Getting started

Before you start, you will need to install two libraries to help with the simulations, if you haven't done it already. The first library is called `Require`, and is an R package (or library) dedicated to improve code reproducibility on the setup side. The second library we will need is called `data.table`, which improves efficiency of data manipulation.

```r
if (!require("Require")) {install.packages("Require"); require("Require")}
Require("data.table")
```

The next thing we will do is to `source` (or in other words, *read and be aware of it*) all `functions` that will help us simulate the movement of the turtle. These functions are described in the file `turtleSimulation.R`, which is a typical coding file for the R environment. Functions are the basis for programming. They are generally generic commands to explain to the software what to do. If you are interested, you can read more about:

R and functions, in this great and simple tutorial from Norm Matloff: https://github.com/matloff/fasteR and in a cool YouTube video from R Programming https://www.youtube.com/watch?v=BvKETZ6kr9Q
data.table and efficient data manipulation in this tutorial: https://www.machinelearningplus.com/data-manipulation/datatable-in-r-complete-guide/
RMarkdown and other awesome packages from Yihui Xie here: https://bookdown.org/yihui/rmarkdown/ and https://bookdown.org/yihui/rmarkdown-cookbook/
RStudio in a fun and easy way: https://moderndive.netlify.app/1-getting-started.html

The present `turtleSimulation.Rmd`, on the other hand, is a mixed-file that allows for both documentation (normal text) and code to be written. When the file is done, this document can be `rendered` into a PDF with a really neat format, which you can also see in the folder, as `turtleSimulation.pdf`.

2. Functions' Sourcing

To `source` the functions we will using, we can use a function in R called `source()` (I agree, not too creative, but easy to remember). To run the line below, you can press the little green arrow on the right side of the `code chunck`, where the instructions for R to source the file with functions are written.

```r
source("turtleSimulation.R")
```

If everything goes according to the plan, you should have seen in the `Console` below this window the following sentence:

```
All functions were correctly sourced! You are ready to start.
```

Now the real fun begins!

3. Habitat preferences

The little turtle will move across the landscape based on the probabilities that a habitat is good for it. In general, it will choose the following available habitats, in this order of preference: A. Water (light blue) B. Old forest that is near water (dark green) C. Middle-aged forest near water (middle green) D. Young forest near water (light green) E. Old forest (dark green) F. Middle-aged forest (middle green) G. Young forest (light green) H. Recently burned patch (light yellow) I. Human disturbed patch (grey or black)

The movement is divided in three parts: *direction*, *distance*, and *patch choice*. All three are primarily independent of each other.

3.1. Direction

The first thing to determine is the direction that the turtle will move. All directions have the same probability of being chosen.

3.2. Distance

Once you know the direction, you need to determine the steps. The turtle is more likely to travel short distances than long ones.

3.3. Patch choice

Once the turtle has moved in the determined direction and distance, we need to define exactly where it will land. The choice will depend on the availability of habitat around the **destination patch** it arrived at.

4. Simulating Habitat Selection

As explained before, the simulation will happen in three moments, in this order: 1. Direction 2. Distance 3. Patch choice

4.1. Direction

To know which direction the turtle will travel, you can use the function:

```r
simulateDirection()
```

If the turtle reaches the limits of landscape, just turn back in the exact direction it came from.

4.2. Distance

To know how far the turtle will travel, you can use the function:

```
simulateDistance()
```

4.3. Patch choice

Finally, to know which patch the turtle actually chose to spend the year, you can use the function: `simulatePatchChoice()`. This special function, however, needs inputs from the destination patch the turtle arrived on after travelling. You will need to input the information on the surroundings of the **destination patch**, the patch where the turtle arrived after it determined a direction and the number of steps (i.e., patches).

A list of all habitats available and their respective codes can be seen using the following function:

```
availableHabitatsTurtles()
```

To input the information on the surroundings of the **destination patch**, you will need to provide which habitat is available in each direction of the **destination patch** as (for example!):

```
simulatePatchChoice(N = "F",
                    NE = "F",
                    E = "C",
                    SE = "A",
                    S = "A",
                    SW = "H",
                    W = "C",
                    NW = "B")
```

Please note that this code above is just an example! You will need to fill the arguments yourself based on where **your** turtle is landing.

Note that if a direction is not available (end of board), you just have to repeat the direction where the turtle came from as below:

- If N is not available, use the same as S
- If NE is not available, use the same as SW
- If E is not available, use the same as W
- If SE is not available, use the same as NW
- If S is not available, use the same as N
- If SW is not available, use the same as NE
- If W is not available, use the same as E
- If NW is not available, use the same as SE

If the turtle lands in a *corner patch*, just simulate a new direction and distance.

Taking notes on each year's habitat chosen At last, once the turtle has reached its ~~vacation~~ wintering destination you will need to take notes of the habitat type it landed on. To make it easier on you, use the code (capital letters from A to I) provided by the function `availableHabitatsTurtles()`.

You will need to do at least 5 rounds of 10 movements (i.e., for each of the 5 rounds, simulate the patch choice 10 times). Once you are done with each round, you will need to summarize the total number of times the turtle chose each one of the different types of habitat, and fill in a table (column `totalNumberOfArrivals`), which has already been prepared for you. This table can be found in the `data` folder of the `predictiveEcologyInAction` directory. The table to fill in is called `turtleArrivals.csv`.

> It is *VERY IMPORTANT* that you do not change the file name or type, as the code was built to read `.csv` files, not normal Excel `.xlsx` files.

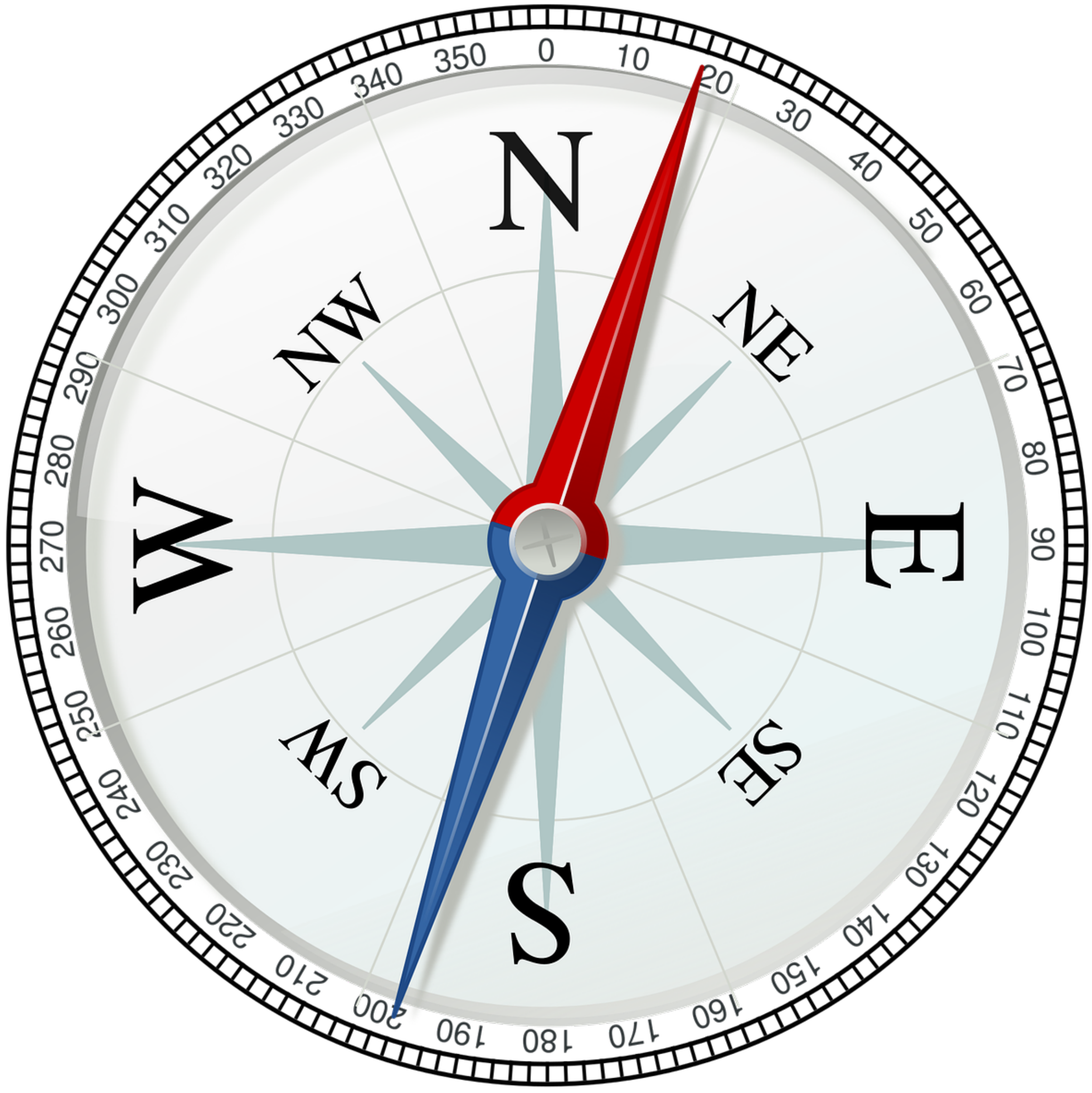

Figure 1: Here is a nice compass to help with the directions.

Remember, the more information you collect, the better your model will be – as in most cases when it comes to science!

5. Fitting the Habitat Selection Model

Once you have finished your five rounds of movements (or more, if you are really efficient!), you will move to building a model to be able to forecast where the turtle is now and will be in the future landscapes. This is a complicated statistical procedure under the hood of a very simple code, as shown below. The first step will be to import the table you created in Excel:

```r
turtleDataset <- fread("data/turtleArrivals.csv")
```

After importing the data, we will then fit a simple linear model, which says that the total number of arrivals depends on habitat type (or in other words, that the turtle chooses patches based on their habitat type):

```r
turtleModel <- lm(formula = totalNumberOfArrivals ~ habitatType,
                  data = turtleDataset)
```

Once we have our turtle model, we will build a table that will generically predict the habitat selection for the turtle based on the information we collected:

```r
forecasts <- round(predict(turtleModel,
                           newdata = data.frame(habitatType = c(LETTERS[1:9])),
                           type = "response"), 0)
```

And we will put it in a better format so we can better see the values:

```r
turtleResults <- data.table(habitatType = c(LETTERS[1:9]),
                            habitatPreference = forecasts)
print(turtleResults)
```

Once we have our final table, we will save it and bring the saved object to me, so we can explore it further with the others.

```r
saveTurtleResults(turtleResults)
```

If you see the message: `Results saved!`, you just have to bring me the file `data/turtleResults.rds`, or upload it to the exercise google folder (`https://drive.google.com/drive/folders/1X5vCgxLnoRwXDRWEsyHZ4ip-CO18KjBx`) and you have finished the exercise!

Congratulations! Now you know how likely your turtle will chose different habitats. We need, however, to know how the landscape looks like now and in the future. And then see what the birds' group has to do with it!

Once all groups are done, we will integrate all projects and answer our research question. :)

7. Playing further on

If you enjoyed the simulations and would like to play further on, just restart the game by running the function:

```r
startOverTurtle()
```

You can then place the turtle randomly on the board (avoiding the edges) and re-run the present script from the beginning.

I hope you had some fun, but most of all, could grasp the general mechanisms behind how we model animal habitat suitability.