Landscape Simulation Exercise

Tati Micheletti

23 May 2023

Simulating Landscape Changes

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 Tree group, which we will rename to ***Landscape Simulation*** group. Fancier, right? :)

You will be responsible for simulating changes in the landscape and, ultimately, influencing how the turtles and birds populations will change.

In the folder you downloaded, you received a spreadsheet named `landscape_simulation.xlsx`. This sheet brings the aerial view of your landscape and you can see how it changes through time.

1. Getting started

Before you start, you will need to install three 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. The third is a library to deal with spatial analyses.

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

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 changes in the landscape. These functions are described in the file `landscapeSimulation.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 `landscapeSimulation.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 `landscapeSimulation.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.

```
source("landscapeSimulation.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. Possible fates for different landscape types

For each 10 year time step, we will simulate the fate of 1. Forests 2. Human disturbance

To make it fun and understandable, we will do a very (VERY!) simple exercise on simulation, where water bodies are kept constant through time, climate change is ignored (although it should NOT in real life), and the processes for all animals and landscape a very simplified. After the exercise we will discuss a study that brings in considerably more complexity. But one thing at a time.

First, we will discuss how each element can be changed. Then, we will start changing them.

3.1. Forests

There are two important things about forests in our game that you need to remember:
1. Forests have one very important property for animals: **age**.
2. Forests can have only three behaviors: **grow**, **burn** or be **replaced by human development**.
In our game, older forests are less resilient to fire, followed by middle-aged
forests, and then by young forests. If forests do not burn nor are replaced by human development, they will grow. A forest that is 10 years old today will be 20 years old in 10 years, the time step we are using for our landscape simulations.

3.2. Human disturbance

There is only one level of human disturbance, which can be grey or black. The size of new developments will always be 1. Human disturbance can only **appear** in a *forested area* or in a *burned area*, being more likely in a burned area, followed by a *young forest*, *middle-aged forest* and only then by *older forest* (in our game, everyone understand and respect the importance of forests!).

4. Simulating landscape changes

First of all, we will create a landscape where the changes will happen:

```
landscape <- getInitialLandscape()
```

Here we can see the legend of our landscape: 1. Old forest (dark green) 2. Middle-aged forest (middle green) 3. Young forest (light green) 4. Recently burned patch (light yellow) 5. Human disturbed patch (grey or black) 6. Water (light blue)

Once we have our landscape, we will have changes happening in the following order: 1. Forest growth 2. Human development 3. Fire

4.1. Forest growth

To know how forest grew during the decade, you can use the function:

```
simulateForestGrowth()
```

4.2. Human development

To know which cells became disturbed by human actions during the decade, you can use the function:

```
simulateHumanDisturbance()
```

4.3. Fire

To know which cells burned during the decade, you can use the function:

```
simulateFire()
```

5. Checking changes to landscape

We will then check which cells changed and follow the changes in landscape.

```
changes <- checkChangesInLandscape()
changes
```

6. Implementing changes to landscape

At last, we will add our changes to the original landscape created here and observe how the landscape changed. In other words, we will update our landscape with the observed changes.

```
landscape <- updateLandscape(landscape, changes)
```

Congratulations! Now you reached the end of the second time step of the simulation (the first step is considered the initial conditions). You can now keep simulating forest growth, human disturbance and fire (i.e., steps 4.1 to 4.3, 5 and 6) until you reach 10 time steps.

Once you finished all simulations, our last step is to save the landscape map and the send it to me. We will do that with the code below:

```
saveLandscapeResults(landscape)
```

If you see the message: `Results saved!`, you just have to upload the files `data/landscapeResults.tif` and `data/landscapeResults.tif.aux.xml` to the exercise google folder (`https://drive.google.com/drive/folders/15QOytBr`
and you have finished the exercise!

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 see how playing with the probabilities of human disturbance and fire would change your landscape, you can restart the game by running the following function:

```
startOverLandscape()
```

You can now re-run the code contained in this `.Rmd` file while modifying the functions for simulation of human disturbance and fire disturbance as such: `simulateHumanDisturbance(intensity = newLevel)` `simulateFire(intensity = newLevel)` where `newLevel` is a multiplier value to the probabilities of these two changes. For example, for a scenario where human disturbance is 3 times more likely than the default set up, and the fire is only half likely to happen, you can pass the following:
`simulateHumanDisturbance(intensity = 3)`
`simulateFire(intensity = 0.5)`

I hope you had some fun, but most of all, could grasp the general mechanisms behind how we simulate landscape changes.