Probabilistic simulations of livestock resilience

Florian D. Schneider

14.01.2016

In this document I am exploring how to use the caspr package can be used for a probabilistic assessment of future events in a landscape. This adds three elements to the previous version of the cellular automata model:

1. random noise on parameters to simulate climatic variation or habitat heterogeneity
2. setting a timeseries of parameters (e.g. for climate change or rotational management)
3. a function that assesses the frequency of potential outcomes of the lattice over a particular period in time

I will discuss possibilities to achieve that in the following paragraphs.

Parameter manipulations

Climatic variability

To add an element of climatic variation, the model includes a parameter `sigma` which sets the standard deviation of a random number around the environmental quality b. This defaults to value 0, which means that environmental quality equals b at each timestep. If `sigma` takes any value larger than 0, at each update, a gaussian random number is generated with mean b and standard devation `sigma`, with a lower limit of 0. This number subsitutes environmental quality during the given year.

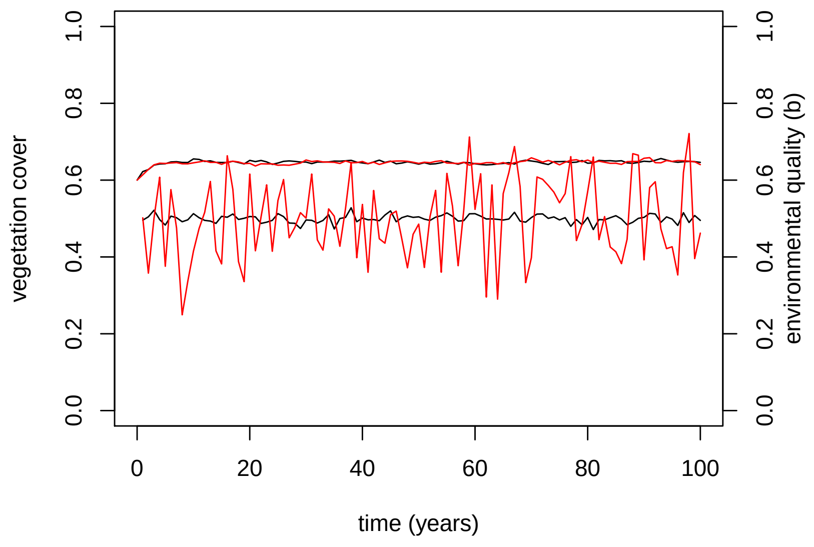
```
climate <- rnorm(1, b, sigma)
if (climate < 0) climate = 0


...


growth <- with(parms, (r * (climate + (1 - climate) *
    f * q_one_one) * rho_one^(1 + alpha) * (1 -
    (rho_one/(K * (1 - c * q_one_one))))/(1 -
    rho_one)) * 1/subs)  # recolonisation rates of all cells
```

The environmental quality thus varies from year to year.

```
## [1] 0
```



```
## [1] 0
```

It seems, though, that high climatic variability (red scenario) has only little effect on the simulation outcome. Rare events might cause a phase transition in close-to-critical systems.

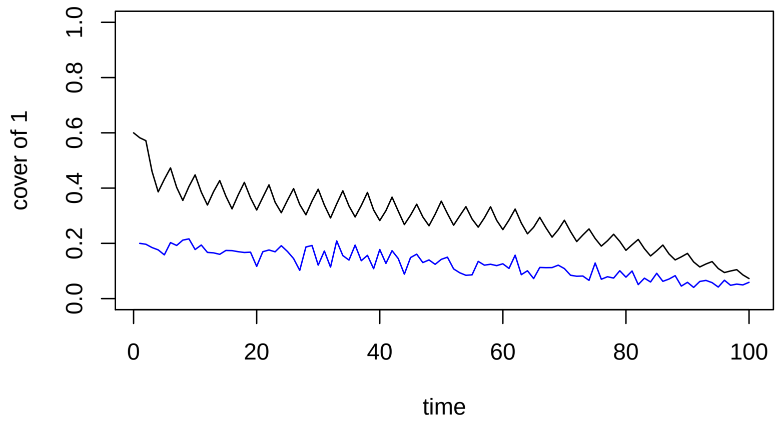
Other parameters could be subject to noise as well, e.g. the carrying capacity or water runoff of the landscape might vary strongly due to slope and exponation. This could be implemented easily by adding further sigma variables that turn the parameter from a constant to a probabilistic bell shaped distribution. This would allow to introduce uncertainty in the model.

Parameters that are associated to plant traits, like the attractant decoy, facilitation or associational resistance effects, are unlikely to vary between landscapes and should be considered as constants.

providing a climatic or management scenario

Manipulations to the parameters over time will be required for the intended research questions. Either because the questions address the impacts of ongoing climate change on arid landscape stability without intervention, or because the consequences of intervention by management methods are to be assessed. In both cases, a predefined time sequence of a parameter needs to be fed into the updating procedure of the cellular automata. This will be achieved by providing the model with vectors instead of single value parameters. Each vector has to have the length of the requested timeseries or it will be recycled, e.g. for rotational management measures.

Thus, as required by the caspr package, the parameter set is given as a list of parameters, but each list entry can be either a single value or a vector representing the sequence of parameter values over time (in years). This is inflated by the requested time to evaluate using the function `parms_timeseries()` of caspr.



Interestingly, some parameters do have the power to dominate the stochastic variation of the cellular automata spatial dynamics. In this example, after an initial decline of cover to steady state, the decline in environmental quality is well compensated by the ability to cope with aridity through local facilitation (y 20-50). The rotational grazing pressure has little permanent effect until a critical threshold of environmental quality is hit (y70). Then vegetation collapses within few decades.

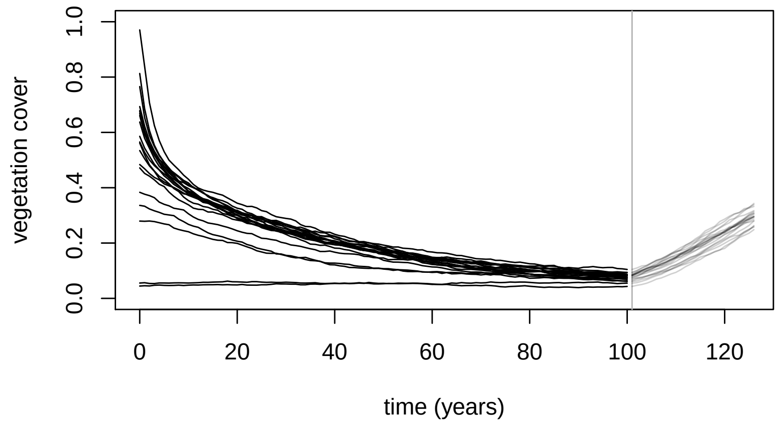
Probability distribution of outcomes

In a next step, we want to explore the space of possible outcomes under a given set of parameters. This is achieved by running the model iteratively, starting from steady state situations, and by mapping the trajectories and analysing the final state of the model after a couple of years.

timeseries simulations and steady state

All simulations have to start out from a pre-formed landscape with realistic patch structure, since the spatial structure is what determines the future development of the landscape. That is particularly important if looking at only short time spans such as 5 or 50 years.

For now, we assume that steady state is achieved after a period of 100 years, as seems realistic and robust looking at the following 20 timeseries that start from random initial cover.



functions for updating

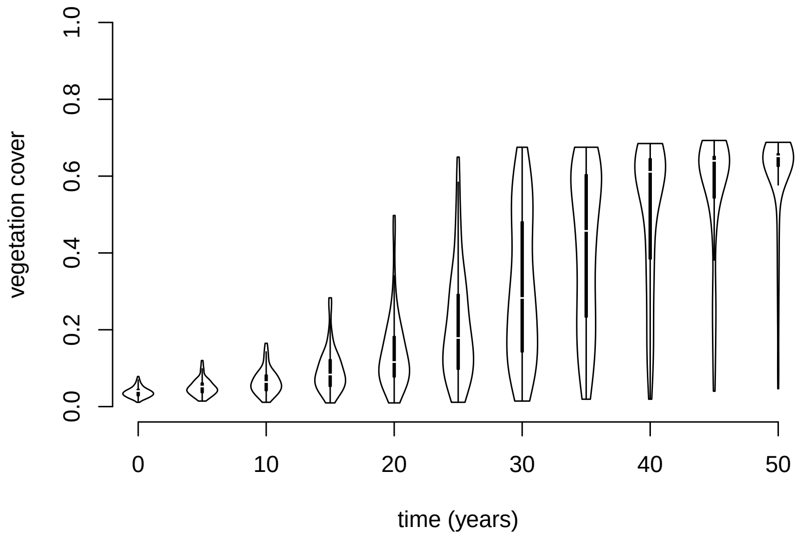
The goal is to run iterative simulations on a given parameter set to evaluate the distribution of outcomes after certain time periods, e.g. 5 years , 10 years, 20 years.

Therefore, I wrote a set of functions which take a list of landscape objects (created by function `init_list()`) and updates it with the given parameter set for the requested time period (by applying function `update_list()`). If a parallel backend is provided, the single runs are processesed in parallel. The output is an updated list of landscape objects which can be summarized into a report of the distribution of outcomes.

visualisation and analysis

Violin plots represent the distribution of the iterations current cover and can be interpreted as probabilities. The distribution pattern varies with the number of replicates, the amount of environmental noise and size of the landscape.

```
## Loading required package: sm
## Package 'sm', version 2.2-5.4: type help(sm) for summary information
```



Thus, the likely development of the landscape unfolds over time as the distribution of potential outcomes. In this case example, degraded landscapes are exposed to even higher aridity, but grazing is drastically reduced as a management method. Only few landscapes recover within a comparatatively short period of 20-30 years. Most need longer, but all landscapes will recover eventually.

Only in situations very close to the tipping point a stochastic event would express in a split of the violin plots into persistent, bimodal distributions. It seems, however, that under the given model specification catastrophic shifts are a rare event. We will need to explore parameter space to find cases where this happens or add more stochasticity by adding further noise.

Note that the resulting vegetation state is closer to the overal average in greater landscapes (central limit theorem). However, large landscapes come with high computational costs and will limit simulations to smaller number of replicates, wich would reduce the reliability of the distribution kernel. For production use, i.e. for publication quality data, a tradeoff between small landscape size and high number of replicates must be found. An educated guess would be to use a small landscape size of 50 x 50 cells (i.e. 625m^2 = 1/16 hectare) and run 1000 replicates to obtain reliable kernels.

analysis of distributions

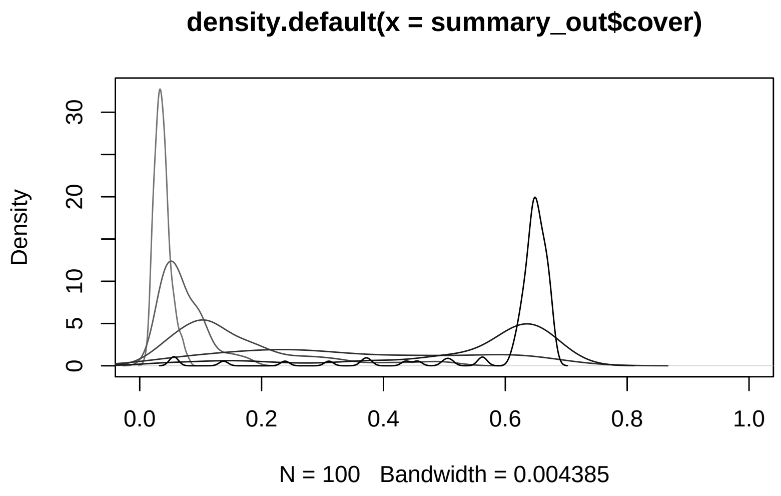
Each model step represents a set of replicated landscapes that were exposed to a parameter set. I defined a method for the `summary()` function that extracts the relevant information describing ecosystem state, that is, the total cover and the average local cover. In the terminal view, this function reports only the mean and standard deviation. But the full vectors can be extracted using the squared brackets or `summary(L)$cover` or `summary(L)$local`. The function also computes a binned Kernel density Estimate (using function `density()`) for total cover, which can be extracted by `summary(L)$kernel`.

A plot of the Kernels visualises the likelihood of the future states of the landscape. If vegetation cover is translated into some value or revenue, the kernel can be used to project the expected payoff of a management method in say 5, 10, 20 years.

```
summary(L100)
```

```
## Assessing n = 100 landscapes:
##   mean total cover:   0.0371 (± 0.0129 )
##   mean local cover:   0.0712 (± 0.0441 )
##   clustering coefficient:     1.95
```

```
plot(summary(L100)$kernel, type = "l", xlim = c(0,
    1))
```


density.default(x = summary_out$cover)
Density
N = 100   Bandwidth = 0.004385