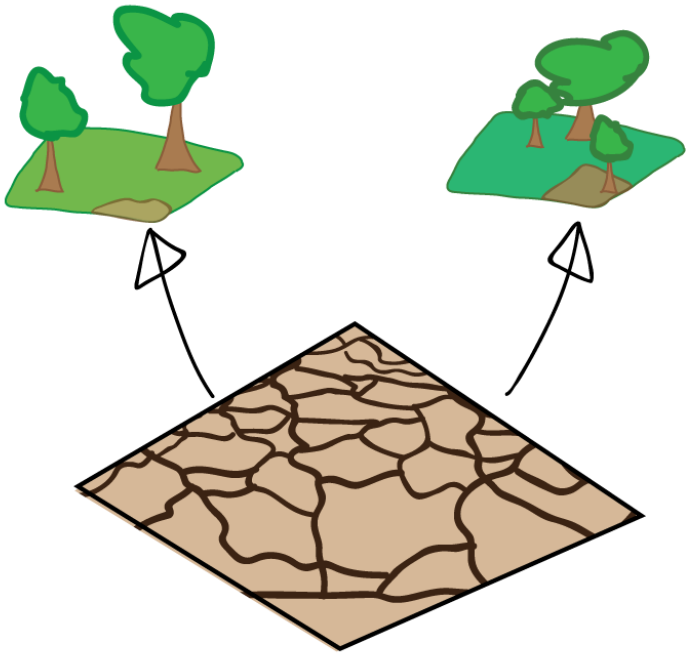
Cellular automata modelling of desertification using artificial neural networks

in south eastern Spain



Thomas Hallundbæk Petersen
Aalborg University Copenhagen
2014

Titile:
Cellular automata modelling of desertification using artificial neural networks
– in south eastern Spain

Project period:
February 3rd – June 12th 2014

Author:
Thomas Hallundbæk Petersen

Supervisor:
Henning Sten Hansen

Copies:
3

Pages:
86

Appendices:
2

Abstract:
Desertification is to an increasing extent being identified as a problem for the region of south eastern Spain. For politicians and planners to fight the desertification processes, there is a need for information regarding the severity and spatial extent of the desertification problem.
In this thesis a desertification simulation model is built, with the aim of providing knowledge about desertification.
The model is constructed as a continuous cellular automata model with an artificial neural network as transition rule. In this way satellite imagery, climatic forecasts and topographic data are used to simulate desertification as a geographical phenomenon.
To model desertification a desertification index is constructed from NDVI satellite images and information on soil moisture and to include information about erosion, RUSLE is studied.
As a result the model uses data for 2001-2012 to predict the level of desertification in 2013. The prediction shows an average inaccuracy of 9.84 %, and it is concluded that such a model is capable of modelling highly complex spatial processes such as desertification. The measured inaccuracy is not sufficiently low for the model to aid politicians and planners to make decisions on how to fight desertification and more research is needed to improve the model.

Preface

This project is the resulting work of a master thesis in the Master of Science programme in Geoinformatics at Aalborg University Copenhagen. It has been written in the spring semester 2014, from February 3rd to June 12th. As a master thesis of one semester this project represents 30 ECTS.
The project has been created under the supervision of Professor Henning Sten Hansen.

In the following project references in text is structured like this:
([Author] [year of publication]). If the reference is to a specific page like this:
([Author] [year of publication],[pagenumber]), so (Petersen 2014, 45) could be a reference to this report at page 45.
A complete list of references can be found on page 84.
Internal references also exist in the following. References to other sections in the report will take this form: section [section number] [section name], example: see section 2.1.3 Drivers of desertification.

Table of Contents

1 Introduction

The quality of soil is of major importance to local people, and through trade and migration patterns, to an increasing extent neighbouring peoples, countries and regions. Spain is the climatic extension of Africa in Europe and with nearly two thirds of its land being either arid, semi arid or dry sub-humid (Ministerio de Medio Ambiente 2008, 12) it is predicted that more than 37 % of the total landmass is in high or medium risk of irreversibly losing all productive capacities to desertification (Ministerio de Medio Ambiente 2008, 138). Also the DISMED (Desertification Information System for the Mediterranean) project under the European Environment Agency, displays alarming figures. The map in Figure 1 is computed by combining aridity, soil and vegetation quality (EEA 2008).

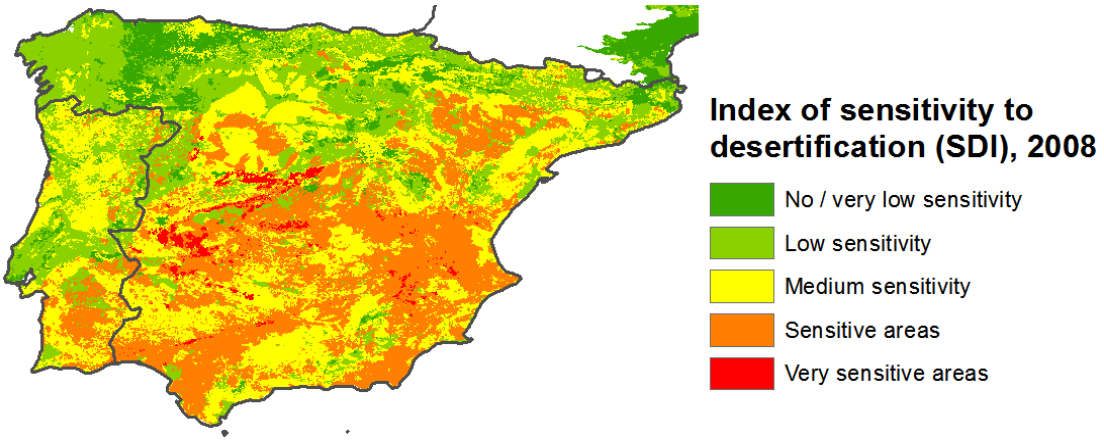


Figure 1 - Desertification sensitivity (EEA 2008)

In this estimate 35.8 % of the Spanish territory on the Iberian Peninsula is either sensitive or very sensitive to desertification (in red and orange).

To keep desertification at bay it is necessary to sustain water availability and to keep top soil from eroding. In south eastern Spain this is complicated by frequent and long dry seasons combined with non-native monocultural land uses, which cause land degradation and lead to desertification. South eastern Spain is the most arid part of Europe (Gómez, López-Bermúdez and Rubio 2011, 936), the rain pattern has high variability and high intensity (ibid.) which means that some years are characterised by droughts and others by erosion. Water resources are being overexploited to accommodate agriculture, tourism and increasing urban activities

(Oñate and Peco 2005, 104). Also the geology of southern Spain makes the region prone to erosion with vulnerable soil in hilly dryland terrain (ibid.).
To add insult to injury the climate is gradually becoming warmer and drier. According to multi-model averages provided by the IPCC, Spain will become 1-1.5 degrees warmer in the most conservative climate scenario (RCP 2.6) and somewhere between 2 and 3 degrees warmer in the direr scenario RCP 8.5, in average in the years 2046-2065 (Intergovernmental Panel on Climate Change 2013, 1063) as compared to the 1986-2005 average. The Iberian Peninsula is also projected to have less precipitation especially the summer months (June, July, August) which will be 10 to 30 % less rainy according to the RCP 8.5 (Intergovernmental Panel on Climate Change 2013, 1078).

The quality of soil and continuous availability of water is vital to keep social stability in the region which depends highly on tourism and agriculture. This is why desertification is such an important issue in south eastern Spain and why there is a need to understand and measure desertification processes. The Food and Agriculture Organization of UN (FAO) in their position on sustainable development and combating desertification concluded that:

"Despite modern observation techniques using satellite imagery and software to analyse recorded data, there is still a great deal of uncertainty at the local, national and world level as to the origin, extent and gravity of the desertification processes. This uncertainty handicaps natural resource managers in planning and decisionmaking." (FAO 1993).

Here we are 20 years later and not much has changed. The emerging threats of climate change and continued failure to correctly counter land degradation in the worlds drylands, makes desertification an ever growing ongoing danger. To aid planners by knowing the speed, extent and severity of desertification, it is necessary to model this process. Spatial models have long been a tool used in many fields of planning, and it is important to keep researching and developing these models to make better predictions for better understanding and conserving our environments.

It seems that south eastern Spain is a good test site for studying desertification. As displayed, many sources point to south eastern Spain as an area where changes will occur. But can the desertification process be predicted? And how?

Much success in spatial modelling using cellular automata (CA) has been displayed, especially in modelling urban development. CA is also showing its applications in other areas of geographical modelling. CA opens up for combinations with other computer models. Artificial neural networks (ANN) is a subfield of computer science inspired by structures in the brain. They are computational models capable of recognising patterns in data and thereby process new unknown data based on the recognised patterns. The cellular automata and artificial neural networks can, if combined by replicating complex spatiotemporal patterns of desertification simulate future development of this geographic phenomenon.

1.1 Problem statement

South eastern Spain is chosen as the location to study the potential of desertification simulation. The prospect of applying cellular automata modelling in combination with artificial neural networks to predict spatiotemporal development of desertification in south eastern Spain will be investigated. The main question that will be attempted answered in the following is:

How can desertification be modelled with cellular automata and artificial neural networks?

To investigate this problem statement, it will be divided into smaller more operational research questions, which will be addressed individually to then provide a basis for answering the problem statement:

- **Which variables should be included in the model?**
- **How should the cellular automaton be set up?**
- **How should the artificial neural network be set up?**
- **With what accuracy can desertification be predicted with such a model?**

To address the first research question, desertification as a geographical phenomenon must be investigated. Links to climate, and topology geology and so on, must be examined, so spatial data can form the basis for a sound model.

To address the second research question cellular automata modelling must be studied, and it must be concluded which properties lead to the desired behaviour.

To address the third research question the capabilities of artificial neural networks to regress complex relationships must be researched.

And finally to address the fourth research question the complete model must be put together and the model outputs must be compared with actual measurements, to quantify the success of the model.

1.2 Delimitation

To predict desertification a model will be devised. A model is an operational simplification of reality. When building a model, one must consider which elements of reality to include in the model. Not everything can be included because of time, data availability and other constraints. This model must be able to function with only the data it produces itself and available forecasted data, in order to make predictions about the future. Climate data for the future has been computed and can be put into a model in this manner. For this reason it will not be possible to include the impacts of human interventions. Management of water resources, spatial planning and changes in agricultural planning are topics too difficult to predict to meaningfully insert into this model, in the scope of this project.

1.3 Report structure

This report consists of the following chapters:

Theory
In the very next chapter, the principal theoretical background for each component is presented. First the concept of desertification is discussed. Focus is here, on what causes it, and how this can be put into a spatial computer model. Next, cellular automata modelling is introduced, and it is discussed how this type of spatial model can be applied to simulate desertification. Finally artificial neural networks, their properties and capabilities are discussed.

Methodology
In the methodology chapter, it is described how, on the basis of the theory, the research questions and the problem statement will be answered. In this chapter the applied analysis design is presented, and the most important methodological considerations are described.

Data processing
The model designed in this project relies mainly modelling in terms of real world data, and in terms of computer model design. The first element is presented in this chapter. Every model input is analysed, and the data used to compute each input is described along with the processing steps taken to reach a ready model input.

Modelling
In this chapter the design process of the model is described. First an overview of the model is given; next the individual components are addressed. These are; the cellular automata part of the modelling, the artificial neural network part of the modelling, the implementation and finally model experimentation.

Results
Here, the resulting model is applied for simulation, and its output is displayed along with relevant performance measures.

Discussion
In this chapter the models simulation capacities are evaluated and the results of this evaluation and the underlying reasons and consequences are discussed. An important focus of the discussion is on how to improve the model in possible future research.

Conclusion
In this chapter the answer to the research questions are summarised, and the emerging answer to the problem statement is outlined.

Perspectives
The conclusion does not put an end to all discussions. In this chapter perspectives of future research based on this project are drawn.

2 Theory

The following chapter consists of three main topics each relating to one of the three first research questions, as defined in the previous chapter, section 1.1 Problem statement. In each of these topics the theoretical background will be laid, to analytically approach each question individually. First the geographic topic of desertification will be discussed. The aim here is to provide operational definitions of involved processes, so they can be included in a model.
Next cellular automata modelling will be described, and it will be discussed how a model based on cellular automata can be designed to simulate the spatial behaviour of desertification over time.
Finally artificial neural networks will be discussed. This section is focussed on how artificial neural networks can be built and trained to regress to a function like the one used in cellular automata for desertification modelling.

2.1 Desertification

Desertification has been defined by the UNCCD as:

"...land degradation in arid, semi-arid and dry sub-humid areas resulting from various factors, including climatic variations and human activities" (United Nations 1994, 16).

So desertification is not just, as one could think, an area becoming a desert. It is simply land in arid, semi-arid or dry sub-humid areas losing its potential to support vegetation through land degradation processes. As formulated in the Millennium Ecosystem Assessment (Niemeijer, et al. 2005) desertification can be viewed as:

"...persistent decline in the ability of a dryland ecosystem to provide goods and services associated with primary productivity." (Niemeijer, et al. 2005, 636 - 637)

A range of processes are involved in desertification at a range of scales. To model desertification it is important to be aware of the scale, and let the scale determine which processes should be involved in a model. In this case desertification is investigated as a geographical event, impacting the south eastern part of Spain, so focus will be on the local geographical and geophysical attributes which determines the rate of land degradation.

2.1.1 The geophysical environment

The potential of soil to support vegetation and thus be of use to people, either through natural services or through cultivation, is controlled mainly by two components; the quality of soil and the availability of water. The soil must have nutrition and must not be saline. The water availability must correspond to the needs of the vegetation; some vegetation is very sensitive to changes in the water availability while other types of vegetation are adapted to irregular weather conditions. In drylands (arid, semi-arid, dry sub-humid and hyper arid areas) water is the main constraint on vegetation growth and health (Niemeijer, et al. 2005, 628).

The soil quality can deteriorate through the accumulation of salts, known as salinisation. It can deteriorate through lack of accumulation of nutrients. And most importantly the top soil can be transported away either by water or wind, this is known as erosion. The erosion is highly dependent on the weather, but also the soil type, terrain slope and vegetation cover.

The water availability is dependent on the weather of course. The amount and temporal pattern of precipitation as well as the evaporation are controlled by meteorology and climate. The water availability is also controlled by the terrain; some soils will hold better on to water than others and the slope controls how fast the water runs of. Hence the soils ability to store water, decides how long after rainfall, water will be available (Niemeijer, et al. 2005, 628). Finally the vegetation cover can both keep an area moist through provision of shade, transpiration and prevention of runoff.

It becomes apparent that the vegetation and soil play vital roles in this system, which is also why desertification is often referred to as irreversible or very difficult to reverse; soil quality and vegetation are mutually dependant, and necessary for sustained ecosystem health.

2.1.2 Measuring desertification

To operationalise the concept of desertification it must be defined in measureable terms, so it can be determined, either if an area is desertified or not or to witch degree an area is desertified. Desertification is of course a scientific concept rather than indisputable property. This provides a bit of freedom in determining desertification, it can be done both in situ by soil samples or expert visual assessment. But it can also be conveniently monitored from space by addressing desertification indicators. For example, when desertification is defined by the ability of an area to support vegetation, the primary production per unit area can be an indicator of desertification. But then another problem arises; a satellite image represents the captured reality at one instance of time. Any indicator might give an inaccurate impression due to probabilistic nature of weather patterns. Therefore as an example, instead of using primary production as an indicator, what should be the indicator is the primary production potential.

2.1.2.1 Fuzzy logic

Fuzzy logic follows the realization that not everything is exact. Some things are modelled well binary, like male or female, some things are modelled well in classes, like types of vegetation. But sometimes the truths are not that simple, and there can be a benefit from having less clear defining borders between classes (Liu 2009, 66). Desertification can be modelled binary;

an area is desertified or not. This will yield a sharp spatial boundary between the two classes, but it might make more sense to estimate to which degree an area is desertified. This works well with indicators as they will always yield a scaling, which will hold more information than whether an area can be classified as being desertified or not.

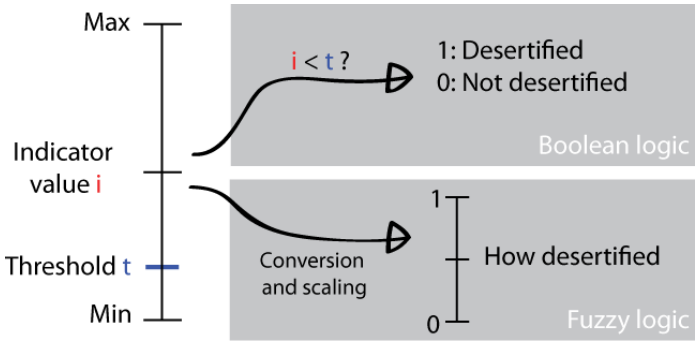


Figure 2 - Conventional logic versus fuzzy logic for indicator values of desertification.

In Figure 2 the indicator value is being converted to a binary classing in the Boolean logic, by an if-then- else-statement. The information will be compressed to a minimum, and there is a need for a threshold value. Figure 2 also displays the alternative fuzzy logic where the indicator value is being converted to a scalar between one and zero. This will ensure that a minimum of information from the indicator scale is lost, so if desired, an extreme indicator value will be mapped to an extreme fuzzy value, and vice versa.

2.1.2.2 A desertification index

When desertification is defined by an areas lack of ability to sustain primary production, the primary production can be a good indicator for desertification. There are a few problems however with using the primary production as an indicator value. These will be discussed in the following sections.

2.1.2.2.1 Measuring primary production

The primary production cannot be directly inferred from satellite imagery. But some sources suggest a near linear relationship between the above ground net primary production (ANPP) and the normalized difference vegetation index (NDVI) summed over time (Prince, Brown de Colstoun and Kravitz 1998) (Tucker, et al. 1983).

What is NDVI?

NDVI is an index derived from remotely sensed imagery by relating the near infrared radiation to the red radiation. It is used in phonological studies because of the information it provides about plant life.

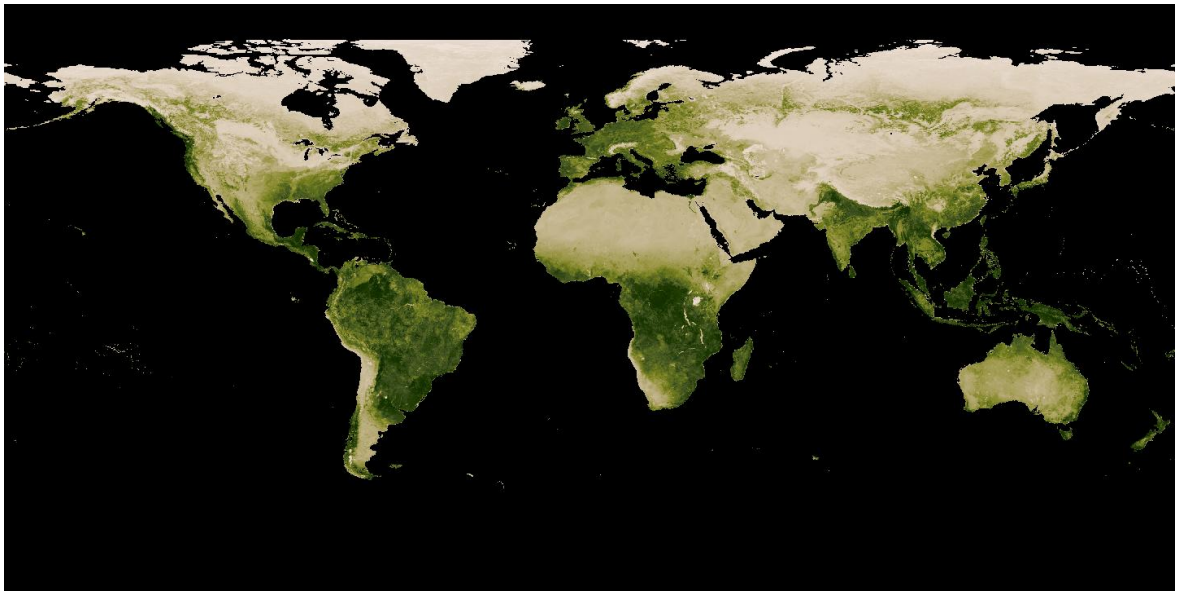


Figure 3 - NDVI values for February 2014 Dark green: NDVI 1, Grey: NDVI 0. Image from TERRA/MODIS (NASA 2014)

Figure 3 shows a map composite of satellite data, which shows average NDVI values in February 2014. The image allows us to quickly make out where on our planet, plant life is dominating, and where this is not the case.

The idea is to relate this concept of remotely sensed NDVI values to the rate of primary production. The connection between primary production and NDVI values follows the logic of understanding NDVI as an indicator of plant vigour or plant health and thereby as an indicator for the rate of primary production.

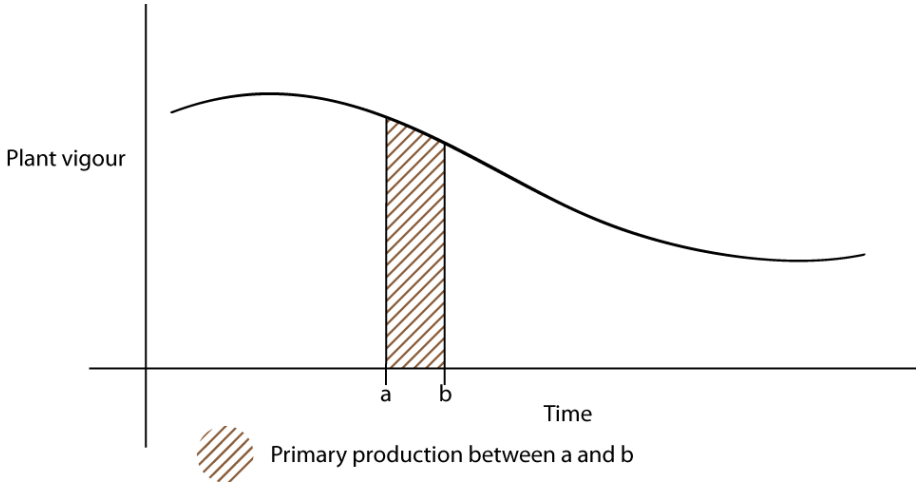


Figure 4 - Relationship between plant vigour and primary production

As Figure 4 displays, if NDVI is assumed to be a good indicator of the rate of primary production, investigating how the NDVI varies in time, can help us calculate primary production from space.

And in this manner ANPP (Above ground Net Primary Production) can be formulated as a function of the integral of calculated NDVI values:

$$ANPP = f(\int_{1.Jan}^{31.Dec} NDVI\ dt\ )$$

Prince, Brown de Colstoun and Kravits (1998, 363) takes it further and assumes f is a linear function, thus;

$$ANPP = b + a \int_{1.Jan}^{31.Dec} NDVI\ dt,$$

and regress a and b from field harvest measurements (ibid.), this function yields an amount of biomass in $\frac{mg}{ha \cdot year}$.

Whether f is well approximated by a linear function might change from location to location, it might change with the level of detail of the survey. It is worth a scientific investigation of its own, which lies outside the scope of this project. The linear model is very simple and effective and it will be assumed to be suitable in this project.

2.1.2.2.2 The variability of weather

Weather is a stochastic variable. When measuring primary production in drylands the main constraint, the availability of water, may differ hugely from year to year. If one year provides favourable rainfall, and the following year features a catastrophic rainfall deficiency, there will be a decline in primary production. We are interested in persistent decline in primary production, and not decline which is explained by the variability of the weather, as it will not be possible to predict future rainfall with a high temporal resolution anytime soon. Therefore the aim is to find a more stable indicator, which behaves more like how desertification is perceived; slow and steady, rather than rapid change from year to year. The net primary production can be stabilized by dividing by the available water; this is called the Rain-use efficiency (RUE) (Houérou 1983) and can be used as an improved indicator for desertification compared to the ANPP (Niemeijer, et al. 2005, 629). The simplest rain-use efficiency is simply $\frac{ANPP}{P_{year}}$ where P_{year} is the total rain in mm throughout the year. P_{year} might not be the best way, to factor in the rain. A high amount of rain could come at unsuitable times or low amounts of rain might be spread out to perfectly provide for primary production needs. Therefore higher temporal resolution of rain data could facilitate a more meaningful rain-use efficiency value.

Another measure of water availability is the soil moisture. The wetness of the soil is more resistant to extreme rain events, as the soil will quickly be saturated, and a maximum value will be reached. The desertification index can therefore be formulated as

$$di = \frac{b + a \int NDVI\ dt}{W_a}.$$

where NDVI are remotely sensed values, W_a is water availability measured either by rainfall or soil moisture, and b and a are regression constants.

2.1.2.3 Conclusions

An applicable index of desertification is adopted like described; as a calculation based on remotely sensed NDVI values and water availability. To finalize the index the net primary production must be regressed from NDVI integrals and it must be tested whether total rainfall or soil moisture is better water availability indicator. This index follows the fuzzy logic, in the sense it will never determine whether an area is desertified or not, it will provide a value, which can be related across time and space.

2.1.3 Drivers of desertification

To predict desertification it is not enough to measure the phenomenon itself, also the processes that drives desertification must be modelled.
A range of factors has an impact on the desertification process. Still there are a lot of unknowns both in climate research and specifically in desertification research. Which factors to include in such a model, depends on data availability and assumed importance. It is assumed that human impacts are real and severe. But they are difficult to predict. In order to successfully predict how irrigation patterns and water usage patterns evolve a broad understanding of socio economic relationships must also be included. Modelling these specifically factors is outside the scope of this project. This will mean that if large areas are changed from one land use to another due to political decisions or similar, this is regarded as noise in the model.

Instead what can be modelled are the natural drivers. Described in the following is water erosion and climate change. Even though both are, at least to some extent, anthropogenic, they can be modelled as natural drivers on a short timescale. Wind erosion as well as soil salinisation are not considered, as Middleton & Thomas (1997, 32-33) consider them less important for the particular geography of south eastern Spain. In a more elaborate model, these two could be included to model desertification more accurately, but due to time constraints, they will not be considered.

2.1.3.1 Water erosion

The soil quality is paramount for dryland ecosystems; therefore failure in soil conservation (i.e. prevention of erosion and soil formation) is a major driver of desertification (Niemeijer, et al. 2005, 628). Water erosion happens when the energy and mass from a rain drop cannot be absorbed by the soil (Middleton and Thomas 1997, 27). Soil particles will be loosened and run off along with the water. Many factors collectively determine the rate of water erosion, Middleton and Thomas (1997) lists the factors shown in Figure 5:

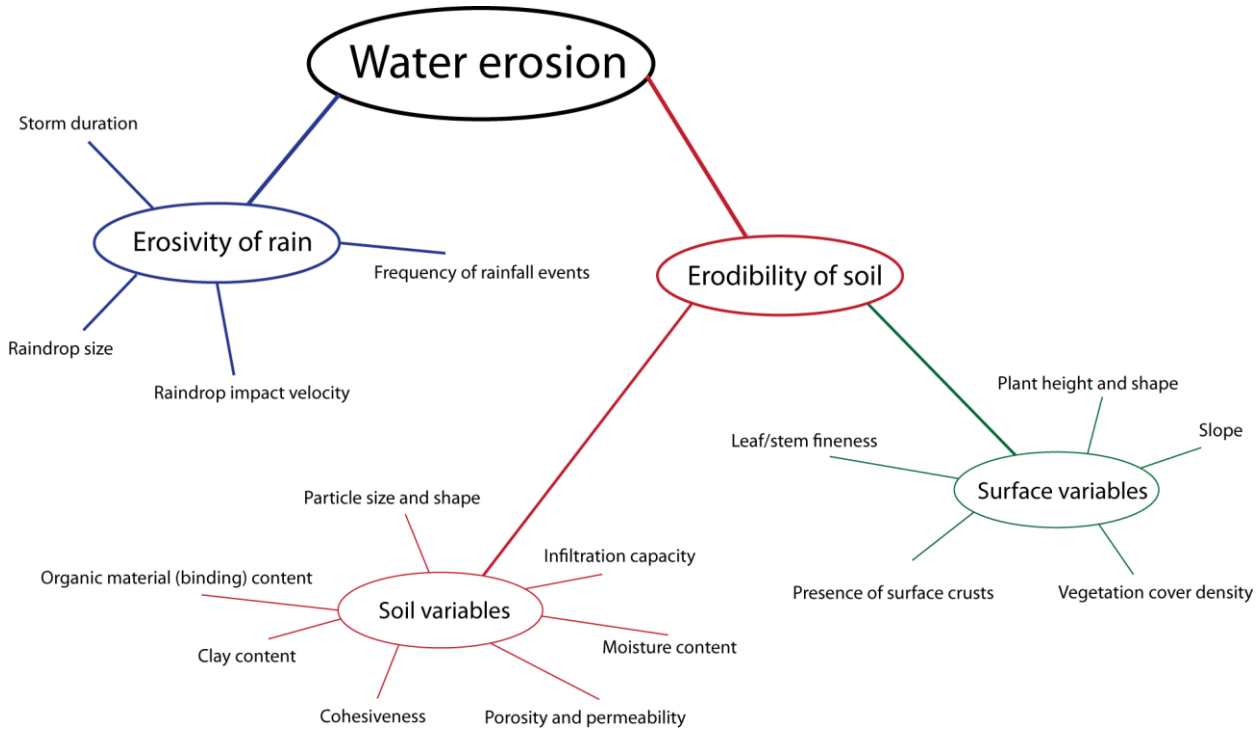


Figure 5 - Factors determining erosion. From Middleton & Thomas (1997)

Not all of these can be easily modelled because of difficulty in data collection. Perhaps the most successful attempt at establishing a mathematical relationship between the most important variables and the water erosion, is the Universal Soil loss Equation (USLE) and its successor; the Revised USLE.

2.1.3.1.1 Measuring water erosion

Water erosion can be computed by the Revised Universal Soil Loss Equation (RUSLE) (Renard, et al. 1997). In RUSLE, soil loss is calculated by the following equation:

$$A = R \cdot K \cdot L \cdot S \cdot C \cdot P, \text{ where:}$$

A is the resulting erosion, in volume of soil lost per unit time

R is the **Erosivity factor** of the rain. The intensity and number of rainfalls a given time period is used to compute this.

K is the **Erodibility factor** of the soil. The soil texture and compaction determines how susceptible to water erosion the soil is.

L is the **Slope length factor** of the terrain. The amount of run-off water affecting a plot of soil, is not only determined by the rainfall but also by the terrain. Water may fall somewhere else and run down slope and affect this plot as well.

S is the **Slope steepness factor** of the terrain. Steepness determines the runoff speed and hence the force of the water.

C is the **Cover-Management factor**. Vegetation cover prevents erosion, as previously mentioned, due to roots and rain cover.

P is the **Support-practice factor** which covers human intervention practices such as terracing and stripcropping (Renard, et al. 1997, 15). No available information on this subject was found and the factor is set to 1.

2.1.3.1.2 The Erosivity factor

The Erosivity factor R, is considered the most important factor when measuring water erosion (Maria, Soupios og Vallianatos 2009, 486).

The Erosivity factor as defined in RUSLE (Renard, et al. 1997) is given by $\sum E \cdot I_{30}$, where E is the total storm energy and I_{30} is the maximum 30 minutes intensity of the storm, these values are multiplied and summed for each storm in the investigated time span.
To calculate this value, high temporal resolution of rainfall data is needed, and this is not always available, therefore shortcuts are taken in order to estimate the Erosivity factor. For precipitation data of daily resolution, EI_{30} can be estimated by assuming that no rainfall lasts longer than a day, and at no day are there more than one rainfall. Then each day with measured precipitation counts as one rainfall with a Erosivity of $EI_{30} = \alpha \cdot P^{\beta}$, where P is the precipitation while α and ß are regression constants (Yu 2008, 259). ß varies mostly within the range of 1.5 and 1.8 and the higher values are primarily found at higher latitudes (ibid.). In Italy ß has found to be 1.53 with a standard deviation of ±0.19 (ibid.). No information about ß in studies of Spain has been found, and the best estimate is that ß similar to that of Italy applies to south eastern Spain. So the Erosivity factor with daily rainfall data is estimated as:

$$R = \alpha \sum_{i=day_1}^{day_n} P_i^{1.53}$$

α here is not important, as it will be described in section 4.5.2 Normalisation, because all factors will be normalised to lie the range of 0 and 1.

2.1.3.1.3 The Erodibility factor

The Erodibility factor K, describes how easily the soil is eroded by rain. This parameter is easier to incorporate because it is more stable. The soil variables that make up the K factor change through slow processes, and can be assumed to be constant in the entire study.

The Erodibility factor is given by

$$K = \frac{2.1 \cdot 10^{-4}(12-OM)M^{1.14} + 3.25(S-2) + 2.5(p-3)}{100},$$

where OM is the organic matter content, M relates to the texture (clay, silt, sand-percentages), s is a structure class and p a permeability class (Renard, et al. 1997, 74). These values can be found by analysis of soil samples and subsequent interpolation (Panagos, Meusburger, et al., Soil erodibility estimation using LUCAS point survey data of Europe 2011).

2.1.3.1.4 The Slope Length factor

The Slope Length factor L, is designed to account for the run off energy, which also impacts the soil. Water running along the terrain from higher altitudes will not only affect soil where the rain drop hits the ground, but also every other place where the drop runs down until it is absorbed, accumulated or evaporated. In RUSLE, the Slope Length factor is given by the following formula:

$$L = (\frac{\lambda}{22.1m})^{M}, where$$

$$M = \frac{\beta}{1+\beta}, where$$

$$\beta = \frac{11.16\ \sin(\theta)}{3\ \sin\ (\theta)^{0.8} + 0.56}$$

Where θ is the terrain slope and λ is the slope length (Renard, et al. 1997, 105). The slope length as defined in RUSLE is measured as the horizontal length of the slope (see Figure 6)

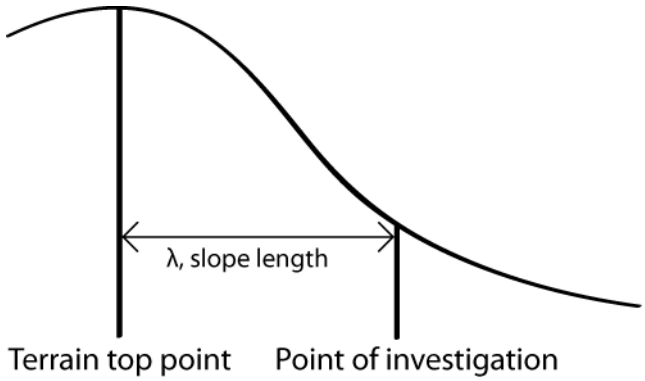


Figure 6 - Slope length, λ as defined for RUSLE (Renard, et al. 1997, 18)

However, this type of slope length was meant for in situ manual assessment and not for calculation based on digital elevation maps. Hickey, R (2000) proposed a way to estimate a cumulative slope lengths based on a digital elevation model, which not only accounts for the maximum slope length of a plot but for each point calculates all slope lengths leading to the point. This is performed by first calculating the flow direction of each point by finding steepest angle to a neighbour cell. Then calculate for each cell the lengths leading from this cell, this is called the Non cumulative slope length, NCSL:

- For a high point this is set $NCSL = 0.5\ C$
- For a point with a flow through in a cardinal direction (N, S, E, W), $NCSL = C$
- For a point with a flow through in a non cardinal direction (NE, NW, SE, SW), $NCSL = \sqrt{2}\ C$,
  where C is the cell size, the DEM raster resolution.

Then these are then added up in the flow direction to make the accumulative slope length which is used as λ.

2.1.3.1.5 The Slope Steepness factor

The Slope Steepness factor S, is incorporated in order to account for the influence on the steepness on the erosion. Just like the K factor and the L factor, S is assumed to be constant in time but not in space. In RUSLE the Slope Steepness factor is calculated by three different formulae, depending on local terrain properties. Just like it was the case with the Slope Length factor this is not a convenient way of calculating the Slope Steepness factor based on a DEM. Therefore Nearings (1997) single equation moderation is adopted:

$$S = -1.5 + \frac{17}{1 + e^{2.3 - 6.1\sin(\theta)}}, \text{where } \theta \text{ is the slope angle}$$

(Nearing 1997, 918).

2.1.3.1.6 The Cover Management factor

The cover Management factor C, is included in order to account for the favourable effect of plant cover. This factor is meant to be the one, which shows the effect of a conservation management plan (Renard, et al. 1997, 146), if the C factor is 0.5, it means there is a plant cover which halves the rate of erosion, compared to no plant cover. Van Der Knijff, Jones and Montanarella (1999) suggest the following relationship between remotely sensed NDVI values and the C factor:

$$C = e^{\left(\frac{-2 \cdot NDVI}{1 - NDVI}\right)}$$

first in an Italian context (van der Knijff, Jones and Montanarella 1999, 26) and later for most of Europe (van der Knijff, Jones og Montanarella 2000, 18). This is convenient as the NDVI values are also included in the desertification index see section 2.1.2.2 A desertification index. The only problem however, is that NDVI values for the future will not be available, as NDVI is information drawn from satellite imagery and these are only available up until present, only the desertification index is available as a model output:

$$di = \frac{b + a \int NDVI \, dt}{W_a}.$$

A function average from u to v is given by:

$$avg\big(f(x)\big)_{u \to v} = \frac{1}{v-u} \int_u^v f(x) dx.$$

This means that if we multiply our desertification index with W_a and divide by the time range we will get an average NDVI value.

$$avg(NDVI) = \frac{di \cdot W_a - b}{a \cdot T},$$

where di is the desertification index, W_a is the water availability either in total rainfall or soil moisture, and T is the total time of the desertification index and a, b are the regression coefficients used to relate the NDVI integral to primary production. This means that:

$$C = e^{\left(\frac{2(di \cdot W_a - b)}{-a \cdot T + di \cdot W_a - b}\right)}$$

2.1.3.1.7 Conclusions

As displayed in the following, almost every part of RUSLE can be estimated (R, K, S, L, C), only the P-factor is left out.

Apart from the K factor, every factor will have to be estimated based on other data, and in every factor a small error will be introduced. The RUSLE model in itself also is not perfect, as it is a relatively simple simulation of a highly complex process. However including RUSLE like described provides the best possible basis for including soil erosion.

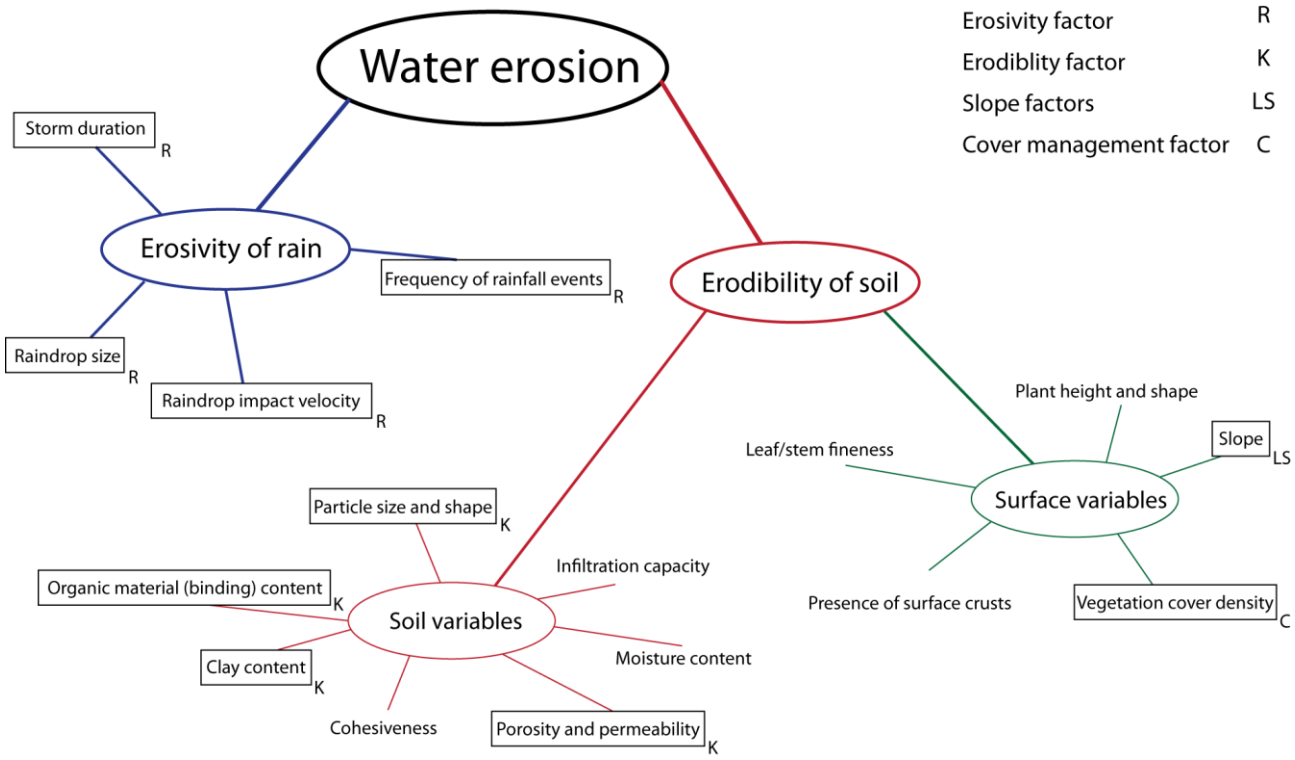


Figure 7 - Factors covered by RUSLE

Figure 7 displays how the factors from Middtleton & Thomas (1997) are covered by the chosen adaptation of RUSLE. Clearly it is less specific in the vegetation part of the surface variables and in the soil variables and this might be significant.

2.1.3.2 Climate change

Modelling and understanding the processes that are drivers of and driven by climate change continues to be one of the major scientific struggles. Desertification is one of those processes which both drives climate change and is driven by it (UNCCD 2012, 29). Soil is capable of large scale carbon sequestration and so is the vegetation that it supports, and therefore can soil degradation be viewed as a driver of climate change. On the other hand increases in aridity, droughts and erosion are consequences of climate change.

Dryland ecosystems are thought to be the ones that will suffer most from the climatic changes (Nicholson 2011). It follows, that if water is the principal constraint, changing precipitation

patterns might have a direct impact on plant life. Changing precipitation patterns will have an impact on water erosion.
Changes in temperature will mean changes in aridity as water vaporisation rates, therefore three measures of climate change will be investigated:

- Rainfall deficiency
- Erosion, which is already covered in a previous section.
- Temperature change

This is by no means a complete list of impacts on desertification by changes in the climate. On the other hand these are the impacts which harmonise both with data availability and the scope of this project.

2.1.3.2.1 Rain deficiency

In the 80's huge parts of dryland Africa were struck by severe droughts. Desertification was immediately linked to droughts and even though this gave birth to a lot of discussions about desertification it has since been questioned (Niemeijer, et al. 2005, 646) how strong this connection really is, and examples were given where the soils were able to bounce back after being hit by droughts (ibid.). However when climate change alters the frequency and severity of droughts, it might be a different story.

There is no universal definition of droughts or drought severity (Nicholson 2011, 407). But several indices do exist with different merits and demerits. One of the simplest indices of drought is the standardized precipitation index SPI by McKee et al. (1993), which is a measure of how likely the observed precipitation is. More elaborate is the popular Palmer drought severity index (PDSI) which relates available moisture to the moisture needs of plants (Palmer 1965).

The fact that the SPI can be computed using only rainfall data and climate normals, makes it applicable for this use, because it allows us not only to measure historical droughts but in addition one can combine it with climate projections to gain information about severity of future drought patterns. The computation though applicable, is rather extensive as it relies on the computation of a gamma distribution function space. Instead a more easily computable mean rainfall is used.

2.1.3.2.2 Temperature rise

Temperature has implications for plant life both on a short time scale, as it impacts the biological processes of plants and raises the moisture evaporation rates. And on a longer scale, as the aridity will slowly change the spatial extent of habitats. An example of the latter is forest migration. As a simple indicator the yearly temperature mean is adapted.

2.1.3.2.3 Conclusions

Although climate change is much more than just change in mean temperature and mean rainfall, these two factors will help enhancing the model by introducing differences and interannual trends in the weather.

2.1.4 Desertification conclusions

A theoretical understanding of desertification has now been established, and operational definitions of desertification as geographical phenomenon as well as two its main drivers, water erosion and climate change, has been established. This will serve as the basis for spatial modelling of desertification, as it now is known which data is important for a desertification simulation.

2.2 The cellular automaton

Cellular automata or CA, are a category of computer models which simulates a certain behaviour of a n-dimensional grid. CA is commonly used to simulate natural growth like systems such as urban development, cancer cells, animal populations and more.

In the following cellular automata will be defined, a small display of its capabilities will discribed and it will be discussed how it can be applied in desertification modelling.

2.2.1 Defining characteristics

Wolfram (1984, 419) uses the following characteristics to describe cellular automata in one dimension:

- It consists of a line of 'sites'
- Each site has a value of 0 or 1
- Each site is updated in discrete time steps
- Each update is determined by one identical rule
- This rule is a function of the site neighbourhood

In this way for each time step, the system develops according to the rule, and very simple rules can result in complex patterns.

If the above is seen as a family member of the CA family of models, it can be generalised a bit to include more family relatives. The CA properties as described by (Liu 2009, 28) are defined as the following:

The grid
The one dimensional CA has a line of 'sites' or a 1-dimensional array. More generally; in an n-dimensional space, the line of sites becomes an n-dimensional grid of sites or cells. This is referred to as the grid.

The states
Each cell has a value associated with it. This is called the state. Basically there are two types of states discrete or continuous. The binary model with 0 and 1 is an example of a discrete state system and this can be interpreted as dead or alive (0 or 1), but also more complex class systems exists, where cells can have more than two different states. The other type is the continuous model where the cell state is defined by a scalar.

Time
The cells states evolve in time. Time in these models is discrete, which means that for a certain time step every cell will have a certain state.

The transition rule
How each cell evolve is governed by the transition rule. This rule is a function of the states of

the previous time step. Only cells within a defined proximity of each cell are included in this function.
For a more sophisticated model, the transition rule function can also include **external factors**, to create site and/or time dependant transition rule variation.

The neighbourhood
Whether a cell should be included in the function of the transition rule is determined by the size and shape of the neighbourhood. The neighbourhood can be of any shape but is typically spherical or quadratic. In two dimensional CA two small typical neighbourhoods are the von Neuman and the Moore neighbourhoods, exemplifying the smallest possible spherical (von Neuman) and quadratic (Moore) neighbourhoods.

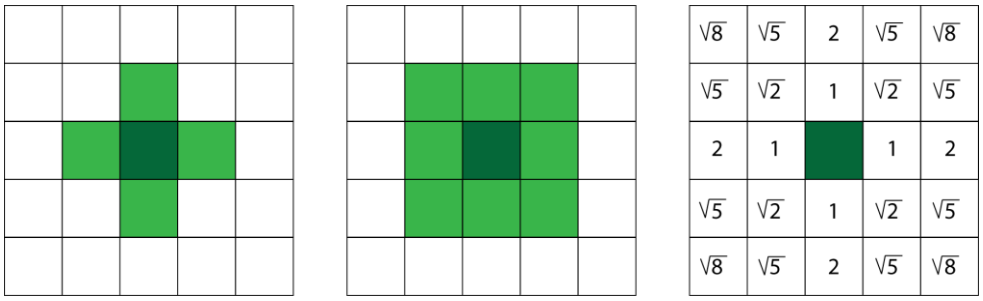


Figure 8 - The von Neumann, and the Moore neighbourhoods, and a distance matrix

The fact that the CA models depend on neighbourhoods makes it topological in the sense that locality matters. There is an obvious analogy to Toblers first law of geography "*Everything is related to everything else, but near things are more related than distant things.*" (Tobler 1970, 236), and as a furthering of this analogy, the neighbourhood can be accompanied by a decay function which weights the neighbourhood so, that close neighbours are more significant than distant ones. This is displayed in Figure 8 where the distance to the centre of each cell is calculated for a unit size grid. The weight associated with each cell will then be a function of this distance such as $w(d) = \frac{1}{d}$ or $w(d) = \frac{1}{d^2}$.

2.2.2 The simple complexity of cellular automata

Although a very simple system, a cellular automaton is capable of complex behaviour. In one dimension much research has been conducted by Stephen Wolfram. Wolfram describes what he calls class 4 behaviour as
"...class 4 involves a mixture of order and randomness: Localized structures are produced which on their own are fairly simple, but these structures move around and interact with each other in very complicated ways." (S. A. Wolfram 2002, 235).

This is inherently false, the cellular automata are not capable of random behaviour as they follow strict deterministic rules, but it is still an accurate description as the patterns produced do indeed appear randomised. As examples of this class four behaviour three cellular automata have been constructed to store three numbers in each cell. These are shown in Figure 9.

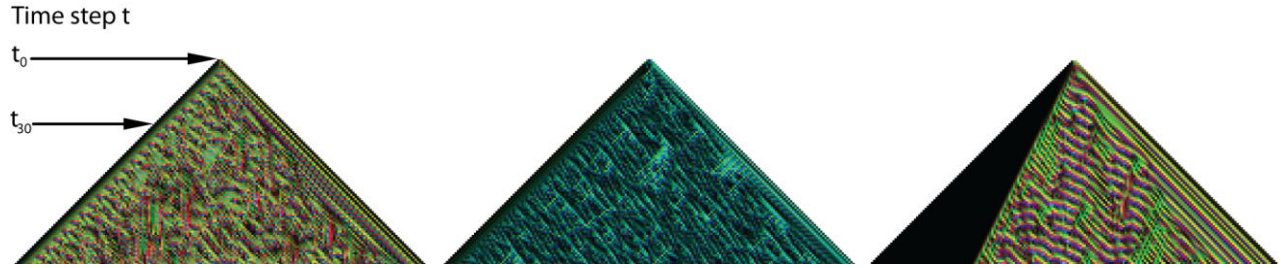


Figure 9 - 1-dimensional CA with colour mapping

The top pixel represents the first time step, the next line of pixels the second time step and so on. The three values of each cell, is calculated based on values, of the previous time step from the same cell and the cells next to it, using trigonometric functions. The three values are then mapped in RGB to produce a colour, so each pixel represents one cell at one point in time. In the leftmost pyramid a very organic pattern emerges with round shapes. In the middle pyramid a very different pattern is produced, consisting of edges and lines and in the rightmost pyramid wavelike structures appear. This demonstrates how simple rules can produce complex patterns resembling randomness.

In two dimensions, perhaps the most well known example of a cellular automaton has been designed by John Conway. "Game of life" was presented as a pastime activity (Gardner 1970), but it proved more important and has since become canonical in cellular automata. Game of life consists of a two dimensional grid with cell states dead or alive. Cells evolve based on a Moore neighbourhood with transition rules inspired by a biological reality. Cells can die from loneliness (fewer than two live neighbours) or over crowdedness (more than three live neighbours) and live cells can spread in the right environment (exactly three live neighbours). Conway's game of life is ultimately the inspiration for most work with cellular automata for modelling spatial systems.

2.2.3 Desertification cellular automata

With correctly chosen properties a CA model can be constructed to simulate geographical systems. This has been extensively researched in urban modelling, where simulations predict growth and structural changes of urban areas (Liu 2009).

Cellular Automata can also be used to simulate other geographical phenomena such as desertification. Here desertification can be treated as either a class property as in 0 or 1 (not desertified/desertified) or a fuzzy classification, as cell states. Ding, Chen & Wang (2009) simulated desertification in the Heibei province surrounding Beijing. Here desertification was simulated with binary cell states, and they concluded that further research should be carried out to combine the desertification CA with fuzzy set theory and to include erosion modelling (Ding, Chen and Wang 2009, 327).
To design a CA for desertification, every property of CA must be modelled with this purpose in mind. In the following the properties are discussed on this background.

The grid

In a desertification CA the grid is 2-dimensional, it could perceivably be in three dimensions to

include something like top soil depth or similar, but it makes sense to model desertification as a 2-dimensional geographical phenomenon. It should be a square grid to ensure that even spatial relationships exist between all neighbouring cells. This would not be true in say, a rectangular grid. It could however be true for a triangular or a hexagonal grid, these do not align well with most raster processing or conventional cellular automata. That is of course not the same as saying it cannot be done.

The states
Desertification in terms of cell states, can be modelled in a number of ways. Desertification can be classified or modelled continuously as a fuzzy cell state describing how desertified a cell is, like it was suggested by Ding, Chen & Wang (2009).

Time
The time property of a desertification CA is straight forward. The desertification level at a location evolves in time just as a CA system cell. The length of the time steps must be determined to fit the temporal scale of the modelled phenomenon.

The transition rule
How the level of desertification changes, appears to be governed by a series of factors as discussed in section 2.1.3 Drivers of desertification, but it is clear that external factors such as climatic attributes must be included in order to model the phenomenon precisely. Rain may be influenced by the level of desertification, but also other factors affect rain patterns, for instance we know that the climate changes both in space and in time.

The presence of surface freshwater can also be expected to have an impact on desertification processes, as surface water alters the availability of water which, as previously stated is the primary constraint on dryland systems (see section 2.1.1 The geophysical environment).

The neighbourhood
The neighbourhood is relevant in desertification as a geographical phenomenon, because resource cycles (water, nutrition, flora, fauna etc.) are part of the ecological properties of a locality. One specific place is more likely to fall into desertification if its neighbouring areas are desertified, because less resource transactions will occur, resource transactions meaning ecosystem resources transferring from one area to another, such as migrating wildlife or moisture spread through transpiration etc.
As most ecosystem transactions happen slowly in short distances a small neighbourhood such as the Moore neighbourhood seems appropriate. With the Moore neighbourhood only neighbouring cells with shared boundaries are included.

2.2.4 Conclusions
Cellular automata systems are capable of serving as the basis for modelling geographical phenomena like desertification. A lot of unknowns persist in how the system should be set up. These will be discussed further in 5.2 Cellular automaton modelling.

2.3 Artificial Neural Networks

The artificial neural network (ANN) is a type of computer model, successful in the field of pattern recognition (Bishop 2006, 226). In this section the feed forward neural network will be described and discussed with the purpose of applying the model in recognising the spatiotemporal pattern of desertification. Artificial neural networks have one of two distinct objectives; classification or regression. The cellular automaton as described in the previous section finds an output value, by evaluating a function of values of the neighbourhood – the transition rule. This function could be in any form conceivable; so by applying artificial neural network regression to zone in on the best possible function the ANN could serve as a transition rule.

An artificial neural network is an attempt to create a mathematical representation of how information is processed in biological systems (Bishop 2006, 226) such as the human brain. Information is stored in Neurons and passed through the network, for the network to collectively achieve its goals. In the feed forward type neural network, information is only passed forward in the network and only a relevant subgroup of these types of ANNs will be described. As this is not meant to be a study of the neural network, which aims to improve algorithms or general knowledge of the ANNs, the focus will be on the basic functionality and not the mathematical theoretical background. This will serve as help in the modelling of the network for the specific uses it has.

This way of using an artificial neural network as a transition rule for a cellular automaton for spatial modelling, has previously been carried out in the field of urban modelling by Li and Yeh (2001). In this study the authors highlight the robustness and versatility of the networks' abilities in spatial cellular automata (Li and Yeh 2001, 1446).

2.3.1 Feed forward ANN

Feed forward networks are the most basic type of neural networks. They are limited in the way that information is only processed in a forward direction.

A feed forward neural network has an input layer any number of hidden layers and an output layer. For this purpose only neural networks with one hidden layer will be considered. Information from the input layer is passed on to the hidden layer, processed and then again passed on to the output layer, processed again to yield the model output.

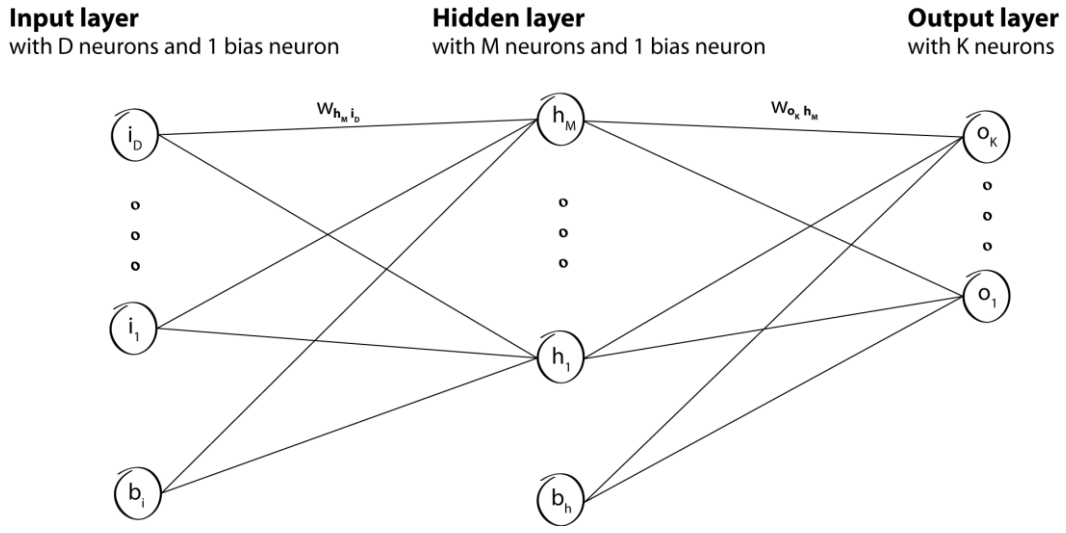


Figure 10 - Basic feed forward ANN

Figure 10 displays a basic feed forward neural network structure, where all neurons of the input layer are connected to all neurons of the hidden layer, except for the bias neuron, just like all neurons of the hidden layer are connected to all neurons of the output layer. All links have a weight associated with it; this weight is a scalar and it could be zero simulating no connection between the two neurons. The neural network function of the model depicted in Figure 10 is evaluated in three steps. First, linear combinations of the input layer values are sent to the hidden layer:

$$v_y = w_{h_y\, b_i} + \sum_{x=1}^{D} w_{h_y\, i_x} \cdot i_x$$

where $w_{b\, a}$is the weight of the connection from a to b, i_x is the value of the neuron x of the input layer and y iterates from 1 to M (Bishop 2006, 227).
These values are then processed by an activation function A to form the values of the hidden layer:

$$h_y = A(v_y)$$

The activation function A should be a continuous function; this is often an s-shaped function such as the logistic sigmoid function which work in the codomain]0,1[or the tanh which works in the codomain]-1,1[, depending on structure of input and output values. These two functions are illustrated in Figure 11.

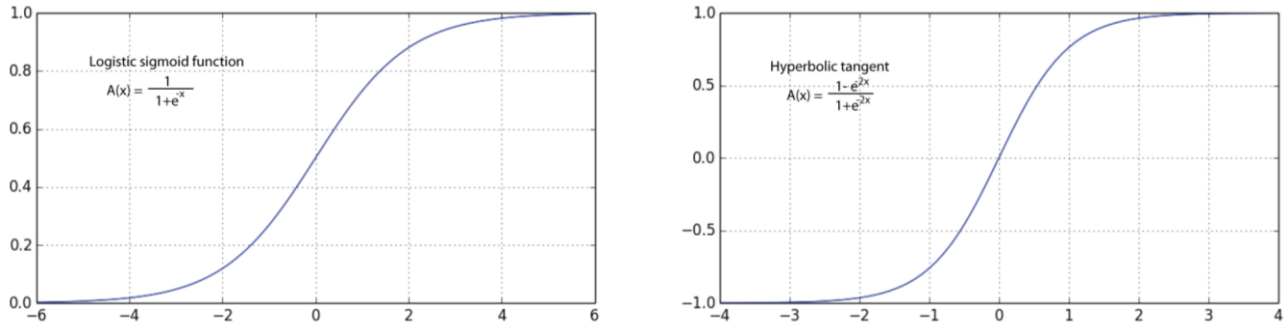


Figure 11 - Most common activation functions

Now that the values of the hidden layer are known, these are again combined:

$$v_z = w_{o_z\, b_h} + \sum_{y=1}^{M} w_{o_z\, h_y} \cdot h_y$$

(Bishop 2006, 227). And finally these values are run through the activation function, to form the final outputs of the network.

$$o_z = A(v_z)$$

The complexity of the neural network depends on the architecture, the number of neurons in each layer, which has implications for the output quality, amount of needed, training data and calculation time.

2.3.2 Network training and backpropagation

When the network architecture is setup, the network can be trained. Network training refers to adjusting the weights, so the output approximates an expected output more and more accurately; this is known as supervised training as the model is supplied with an expected output.

Training of artificial neural networks happens through a procedure known as backpropagation. Backpropagation works by subdividing the error between the incoming connections to a neuron. And then these errors are send backwards even further through the network, to again subdivide between incoming connections from the previous layer. That way, each weight is associated with a part of the model error. For each weight one can imagine, that the error will change as the weight changes.

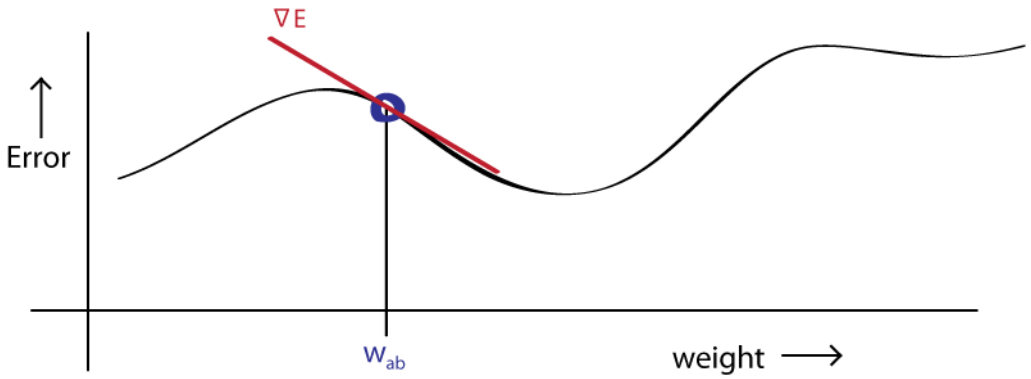


Figure 12 - Error - weight space, displaying gradient for specific weight

Calculating the global minimum for each weight for each step in the training process, is simply impossible computationally, so instead the main working tool is the gradient, ∇E – how the error changes with the weight. This gradient can be used to iteratively improve the weights of the network; if the gradient is positive the error will decrease if the weight is decreased and vice versa. Depending on numerical circumstances and training algorithm, using the gradient to search for better weights, can yield three results if enough iterations are applied; a global minimum, a local minimum, or the weights may simply not converge.

The gradients can be calculated using the following formulae:

$$\nabla E = \delta \cdot v,$$

where v is the incoming value, and δ (called the node delta) is calculated differently for the output layer neurons and the hidden layer neurons. For the neurons in the output layer:

$$\delta_z = (o_z - t_z) \cdot A'(v_z),$$

where o_z is the output value of neuron z and t is the expected value of neuron z. For neurons in the hidden layer the node deltas are calculated a little bit differently:

$$\delta_y = A'(v_y) \cdot \sum_{z=1}^{K} w_{o_z h_y} \cdot \delta_z$$

by summing weights and node deltas from the output layer, and multiplying by the derivative of the activation function evaluated in the incoming value (Bishop 2006, 243 - 244).

2.3.2.1 Training algorithms

A range of training algorithms have been developed, each with its advantages and disadvantages. In the following, two of these will be described and their differences will be demonstrated.

The most basic training algorithm is known as gradient descent, where a learning rate η is set, and each weight is updated by: $w_{t+1} = w_t - \eta \cdot \nabla E$ (Bishop 2006, 240). So the learning rate and the size of the gradient determines the size of the weight change. This will ensure that we find a local minimum, if the learning rate is not too large for the weight to converge. One way

to improve this is by adding a momentum term. This will make the search speed up if the gradient is small and there is a long way to the minimum, furthermore it can help the search, to 'jump' out of a local minimum if the momentum is great enough: $w_{t+1} = w_t - \eta \cdot \nabla E + m \cdot \Delta w_{t-1}$. Both the momentum and learning rate, should be between 0 and 1. Setting these values correctly is not simple, and to overcome this difficulty another algorithm can be applied, this is called Rprop (Igel og Huesken 2000, 115). Resilient Backpropagation, or Rprop, was proposed by by Riedmiller and Braun (1993). Since Rprop has been proposed further research has been conducted and improvements has been added (see for instance (Igel og Huesken 2000)), here only the basic Rprop with weight back-tracking (Rprop$^+$) will be discussed. In Rprop the step size is governed independently of the size of the gradient. The step size is found by comparing the gradient, with the previous gradient:

1. If the gradients have the same sign, indicating that the weight should be moved further in the same direction, the step size is increased: $\Delta_t = \eta^+ \cdot \Delta_{t-1}$, so as long as the weight is continuing down the same slope, the step size will increase exponentially independently of the gradient size.
   the weight will then be updated by: $\Delta w_t = -sign(\nabla E) \cdot \Delta_t$
2. If the gradients have different sign, indicating the weight has crossed past a local minimum, the step size is decreased: $\Delta_t = \eta^- \cdot \Delta_{t-1}$
   the previous weight change will be cancelled (called weight backtracking) so:
   $\Delta w_t = -\Delta w_{t-1}$
   and the gradient is stored as zero, so for the next iteration, the gradients will neither have the same or different signs, and we enter 3.
3. Else the step size will be maintained: $\Delta_t = \Delta_{t-1}$, this is when on the previous iteration the step was too large, so it has already been decreased, and a more careful step is taken in the same direction.

In Rprop, like gradient descent, some variables need to be set; the learning rates:
$0 < \eta^- < 1 < \eta^+$. Also there needs to be an initial step size, as well as maximum and minimum step sizes.
But for Rprop these values are not as crucial, as it is the case in gradient descent (Riedmiller og Braun 1993, 588). And the Riedmiller and Braun suggest the following settings, which is empirically tested to give good results regardless of the problem:

$$\eta^- = 0.5$$
$$\eta^+ = 1.2$$
$$\Delta_0 = 0.1$$
$$\Delta_{min} = 1e^{-6}$$
$$\Delta_{max} = 50$$

(Riedmiller og Braun 1993, 588).

These 'standardized' values help save time in the modelling process, as one does not have to search for good values and check for convergence.

As a simple demonstration a regression problem is constructed. A neural network with 1 input 3 hidden neurons and one output, with the activation function $A(x) = \tanh(x)$, is trained with random weights from -0.1 to 0.1. The goal is to approximate the function $f(x) = abs(x)$, as represented by 20 random points along the function.

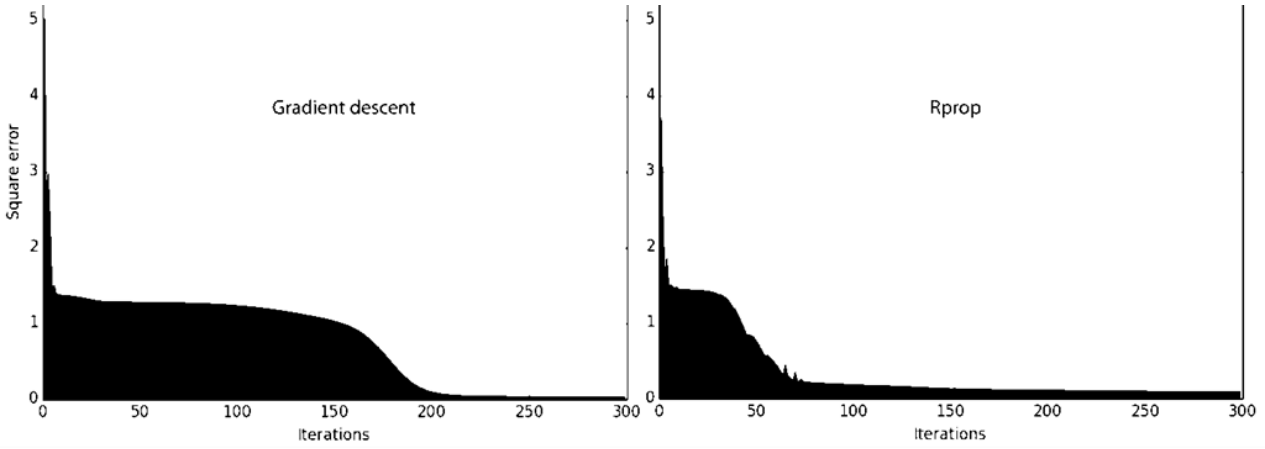


Figure 13 - Training error after each completed iteration of training.

As displayed in Figure 13, the error of the network behaves very differently for the two training algorithms. With gradient descent, the training was highly dependent on the chosen variables learning rate, and momentum. Also when training the network with the same initial conditions, it showed a high dependency on which 20 random points, which were used. There was a substantial variation on how fast it converged, or if it converged at all. The graph displayed was one of the best results.
The Rprop behaved more consistently, it converged every time, and the graph looked very similar each time. One thing to note is that the final error is smaller in the Gradient descent example. And this might be attributed to the momentum term. Based on this, the Rprop is selected as the training algorithm, as it shows more consistent results, and it depends less on the variables. This will allow more focus on experimenting with the network architecture.

2.3.3 Conclusions

Artificial neural networks can be designed to solve a regression problem to find functions that mimic the behaviour of inputted data. With the Rprop training algorithm, training data can be entered into the model to make it adjust without much experimentation or prior knowledge about the modelled data.

2.4 Collective conclusion

In the desertification section it was concluded how a fuzzy index of desertification could be computed from satellite data. Also discovered were operational measures of erosion, and climate change. These measures will serve as the data background for the model. Still lacking are the actual acquisition, processing and calculation of these datasets.

A cellular automaton can be the basis for a desertification simulation model, it is capable of being set up according to modelling goals, and of producing rich patterns which must be

controlled through the transition rule. The properties of the cellular automaton are not trivial and a discussion of these properties will follow.

Artificial neural networks can flexibly solve regression problems to find a function without prior knowledge about this function. This is exactly what is needed as a transition rule in the cellular automaton to control how desertification develops in time. An artificial neural network can be designed in many different ways, so a discussion of how it should look will follow.

3 Methodology

In this chapter it will be described how the question posed in the problem statement will be answered. First general methodological considerations will be discussed, then applied tools are described and finally the analytical design structure will be outlined.

3.1 Methodological considerations

In the following some key methodological considerations of the modelling will be discussed. The following topics will be touched in the section:

- The black box nature of using artificial neural networks
- Spatial and temporal independence of the model
- Data acquisition

3.1.1 Black box

The aim of this study is to investigate, how well suited cellular automata models with an artificial neural network as transition rule, are for modelling the geographical phenomenon of desertification. Often when building a simulation model, information about the nature of the studied phenomenon will be gained. This is not the case when using artificial neural networks. The derived transition function is not generally applicable to modelling desertification. If a new model is build or this model is improved, and perhaps other factors are included, the training process will have to be performed again. This is the trade off when using artificial neural networks, and for this reason ANNs are referred to as a black box (Liu 2009, 50).

3.1.2 Spatial and temporal independence

The modelling is performed using spatially and temporally independent data. By spatially independent is meant, that no inputted data is specific to the geographic region of south eastern Spain, and the same analysis could have just as easily be performed somewhere else, if the same data was available. In this sense, the model treats all pixels the same regardless of coordinates. If the model was not spatially independent, every pixel could be modelled separately, and it would be exposed to over fitting.
The same is true for the time dimension. Time is not a model variable, and the model does not use information about which year it is for its predictions.

By keeping the model spatially and temporally independent, the modelling process strides for generality; the modelling procedure could be moved to any location, to any point in time if the corresponding data could be acquired.

3.1.3 Data acquisition strategy

Data acquisition is an important subject in a study like this. Such an overwhelming amount of spatial and non spatial data is freely available online. Similarly for other kinds of information, not all sources are equally reliable, and so it is important to be aware of where the data is coming from, and on what background it has been published. In the subsequent chapter, all data sources are listed, here the nature of the data is described for each data source.

Overall data has been acquired from very few sources. This makes it easy to keep track of the data, and the problems that may be related to these sources.
Every applied dataset is well documented, there exist metadata describing the data motives and processing information.

3.2 Geographical research area

As formulated in the problem statement, the case area for testing the model capabilities is south eastern Spain. In this study south eastern Spain is selected, because it is believed that desertification may be imminent in this area. The geographical delimitation is defined by the common area of the two major datasets; the eMODIS NDVI imagery and the Eurostat DEM (described in Chapter 4 Data processing).

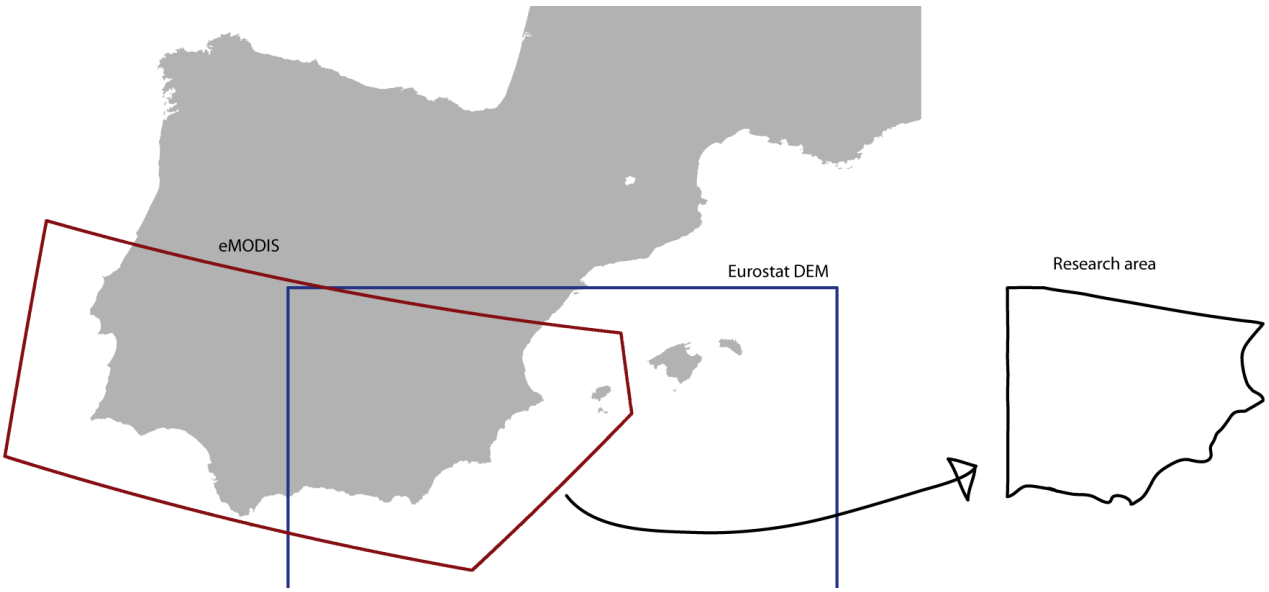


Figure 14 - Research area - common area of the the DEM and the eMODIS datasets.

Figure 14 displays the research area in geographical relation to the rest of the Iberian Peninsula.

For some tasks only a subset of the research area was used. This is area will be known as the “training area”. The training area which can be viewed in Figure 15, is used for model training and other specific tasks, where the entire area was unnecessary or otherwise inappropriate. When the training area is used instead of the research area, it will be stated in the text.

Figure 15 - Geographic definition of the training area.

3.3 Applied tools

ArcPy is a python library created by ESRI to allow python programming with ArcGIS tools. The research work presented in the following chapters, was performed mainly with ArcPy. ArcPy allows integration between the programming elements and the spatial elements, such as raster calculations, so a model can be programmed to include spatial data processing, directly in python.

Python with ArcPy has been the main tool both for modelling and data processing. ArcGIS 10.2 is a Desktop GIS, which allows manipulation of spatial data via a set of tools, and a visual map interface. For some tasks, when the visual output needed inspection, the ArcGIS 10.2 environment was applied.

CDO – Climate Data Operators is a collection of tools for manipulating multi dimensional climate data. CDO has been used for data operations regarding the climate data, where tools in the ArcGIS environment has not been sufficiently equipped.

3.4 Analysis design

The problem statement was operationalised using four research questions, three of which relates to the development of the model and one relating to the quality of the achieved model. Hence, the first three research questions will be answered, then the fourth which will lead to final conclusions on the problem statement.
In the theory chapter the framework for the model was laid out. But still questions remained on how get the best results out of available tools. The following three chapters will be laid out as displayed in analysis design structure, Figure 16. The very next chapter will concern the processing of the raw data, which leads to suitable inputs for

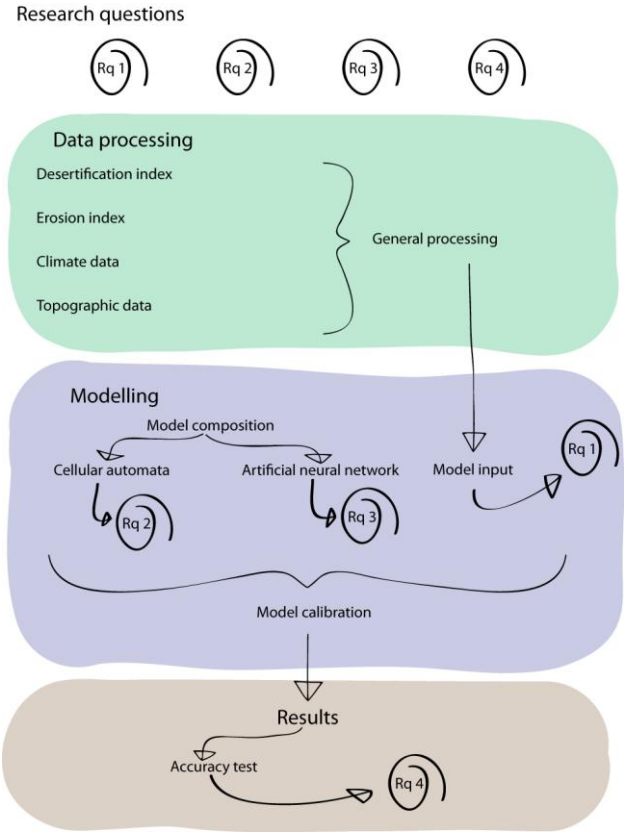


Figure 16 - Analysis design.

the model. In figure 16 the key data processing tasks are displayed.
In the Modelling chapter the structure and development of the model will be discussed. This will in turn lead to answering the first three research questions. Finally the resulting model will run predictions. These predictions will result in a test of how accurately the model can predict desertification of south eastern Spain. This will assist in answering the fourth and final research question.

4. Data processing

In this chapter all applied data will be presented and discussed. To prepare the data to be included in the modelling, it needs to be processed individually according the diverse nature of the data from different sources. So the aim of the following chapter is twofold; to present data sources, and to describe the processing it has undergone.

First the data and processing related to the desertification index will be discussed, and then the data and processing related to the erosion index. Then all climatic data will be discussed, the datasets in the section are applied in the model on its own and as part of the two indices, desertification and erosion. The last presented datasets are the topographic datasets. Finally the general processing, not specific to any dataset are described. Each subsection is structured by starting with an introduction to the topic, listing its data components, then describing the data sources and finally describing the processing applied.

4.1 Desertification index

In section 2.1.2.2 A desertification index it was derived that a desertification index can meaningfully be formulated as $\frac{ANPP}{W_a}$, where the NPP can be found from regression: $ANPP = a + b \cdot \int NDVI$. Both NPP yearly and NDVI weekly are products provided by NASA's MODIS satellites. Yearly NPP are available from 2000-2010 while weekly NDVI's are available from March 2000 up until present. Both W_a candidates, rain and soil moisture are datasets available as part of the Ensembles, which is separately discussed in section 4.3.1 Ensembles – Climate data.

Apart from working as the model output, the desertification index, also provides average NDVI values used in the erosion index, described in section 4.2 Erosion index.

4.1.1 eMODIS – NDVI

The EROS Moderate Resolution Imaging Spectroradiometer by NASA provides free 250 m NDVI products 4 times a month for Africa among other continental datasets. Africa is defined as everything within the bounding box of N: 40.0, S: -39.0, E: 52.0, W: -20.0, this means that data for southern Spain is also included in these products, see Figure 17. The data appear in 10-day composites, so 10 images have been combined to remove clouds from the products images (Jenkerson, Maiersperger and Schmidt 2010). In spite of this effort, the quality of the imagery is still varying, and there is need for further processing, this is addressed in section 4.1.3.1 NDVI Cloud removal.

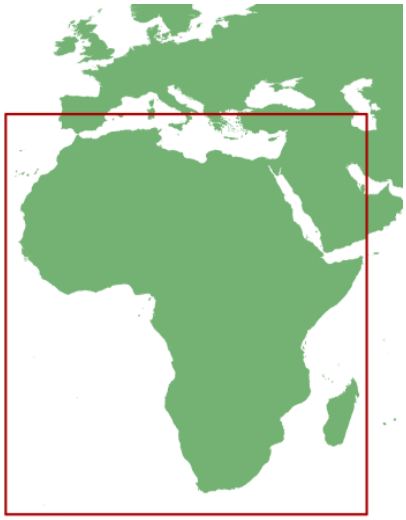


Figure 17 - eModis Africa area. World map provided by Bjørn Sandvik, Thematicmapping.com

As the datasets are available four times each month they provide a good basis for estimating an integral. The datasets are each approximately 1.5 gb large, so only downloading one dataset per month was feasible, due to hard disk space, time and bandwidth limitations.

4.1.2 MODIS – NPP

NASA also provides global primary production estimates. These are independent of the NDVI and are calculated from corrected Leaf Area index, LAI and Fraction of Photosynthetically Active Radiation, FPAR (Running and Zhao 2011). This product is in 1 km spatial resolution and has global cover. The dataset unit is $\frac{kg\ C}{m^2}$.

Several places NPP (**N**et **P**rimary **P**roduction) and ANPP (**A**bove ground **N**et **P**rimary **P**roduction) are used interchangeably. The difference between these two terms, in this matter is small. The desertification index is related to NPP and not ANPP, even though it was initially based on an index that was related to ANPP (see section 2.1.2.2 A desertification index). Desertification will also have an impact on the growth of roots and other below ground primary production sources, so it is not viewed as a problem to switch ANPP for NPP.

4.1.3 Processing

Before a useful desertification index can be inserted into the model, these four tasks needs to be performed, on the above data:

- NDVI cloud removal
- NDVI integration
- NPP regression
- Normalisation of the final output

The processing methodology in each of these is described in the following. Except for the normalisation processing which is described separately for all input data in section 4.5 Non data specific processing.

4.1.3.1 NDVI cloud removal

Even though each eMODIS NDVI product is based on a composite of several images, total cloud free imagery has not been possible. Clouds disturb measurement of NDVI values, as they allow less light through than the remaining atmosphere.

Removal of clouds from satellite imagery is usually a difficult task. Missing information may be recreated from imagery of the same place at different times, but this poses two problems: The information that was not recorded might have changed in the passing time, and the images might be radiometrically incompatible, in the sense that the pixel values in one image, might not represent the same amount of light, as the same pixel value in the next. Lin et al. (2013) described a method for cloning information in Landsat 7 imagery, addressing the problem of radiometrical incompatibility. However this problem is not considered in the following, because the eMODIS NDVI images were designed for multi temporal analysis, radiometrical incompatibility is assumed to be a small problem, and data can be recreated by simpler means. The method described in the following, is not the best way to deal with cloud contamination in general, but should be seen as a "quick and dirty" fix, this will be discussed further in Chapter 7 Discussion.

The disturbance that the clouds cause is removed by interpolation. The missing information can be recreated by taking advantage of temporal and spatial information relationships. Generally there can be two types of trends in the NDVI data, an intraannual cyclic trend and a climatic interannual trend. How the image with the missing data differs from these trends, depends on incidental weather, impacting plants before the image was captured. This information can to some extent be included by comparing mean NDVI for the non clouded parts of the images.

But first, the clouds must be identified in the images.

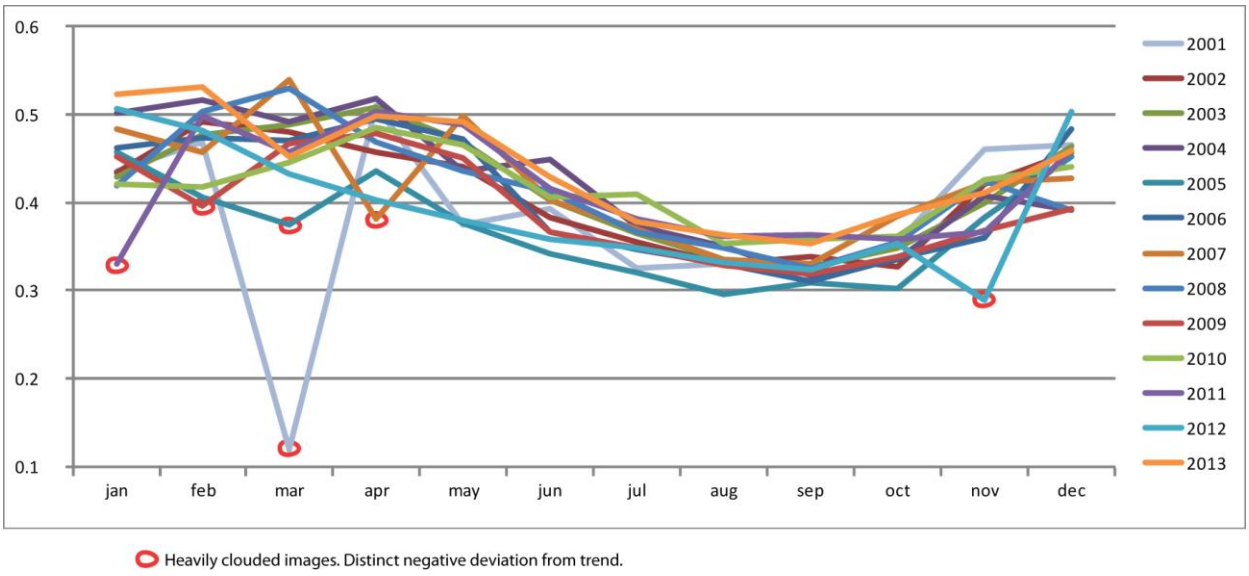


Figure 18 - Monthly NDVI means

In Figure 18 the mean values of the NDVI values are plotted to show the intraannual cycle. The months with heavy cloud cover can easily be spotted; they are identified by unexpected drops from the trend. Clouds can naively be distinguished from undistorted information by applying a threshold. In Figure 19 a threshold of 0.07 is applied to group pixels as either 'good values' or 'bad values'; we see that this works fairly well, identifying water and clouds which have low NDVI values (b) in Figure 19).

Figure 19 - a) Original image, clipped to the southern part of the Iberian Peninsula b) Pixels with NDVI < 0.07 (in white). c) Image histogram

Now that the missing information is identified, it can be recreated by combining available data from the same locality in other images:

- The **intraannual** trend is included by using information from the next and previous month
- The **interannual** trend is included by using information from the same month the next and previous year.
- The information used is adjusted by multiplying with the non cloud mean ratio, using the parts that are without clouds in both images.

This means that four pieces of information about the missing data is computed, and the average value of these four distinct approximations is used as an estimate.

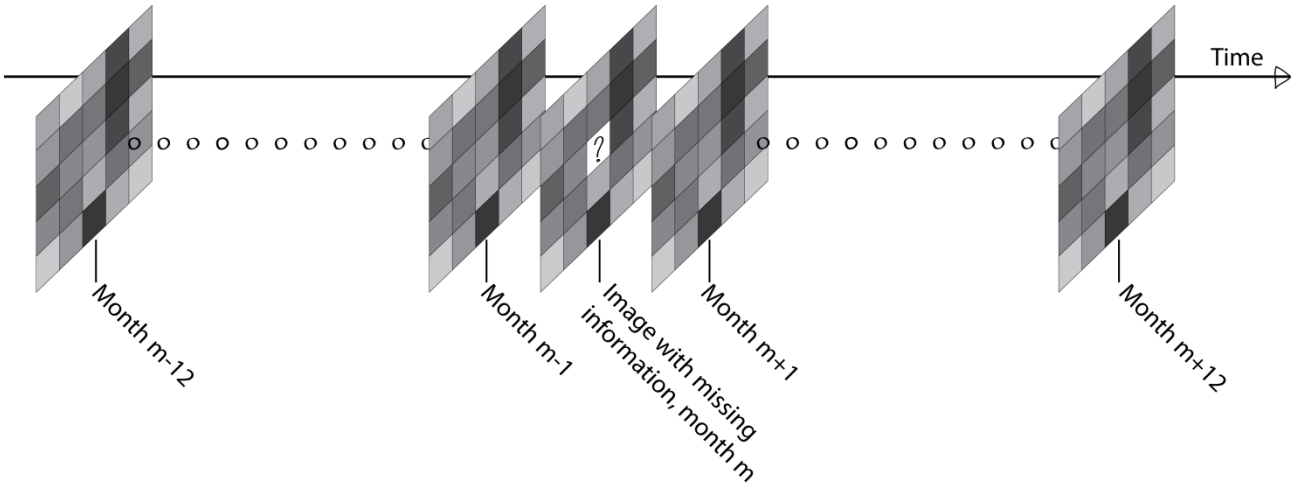


Figure 20 - Information used for reconstructing missing information.

So the pixel value p, at a given month m, is approximated by:

$$p(m) = \frac{1}{4}\left(\frac{\mu(m)}{\mu(m-12)}p(m-12) + \frac{\mu(m)}{\mu(m-1)}p(m-1) + \frac{\mu(m)}{\mu(m+1)}p(m+1) + \frac{\mu(m)}{\mu(m+12)}p(m+12)\right)$$

Where $\mu(m)$ is the non cloud NDVI mean for the month m.

Some images do not have all four components for this equation. For the first image for instance, neither the m-1 nor the m-12 exist, and it can only be estimated based on m+1 and m+12.

4.1.3.2 NDVI integration

In order to find the integral of ('area under') the NDVI curve a numerical estimation known as the trapezoidal rule is applied:

$$\int_a^b f(x)\,dx \approx (b-a)\frac{f(a)+f(b)}{2}$$

This can be applied to the NDVI data, as both a timestamp and the NDVI value are available for each dataset.

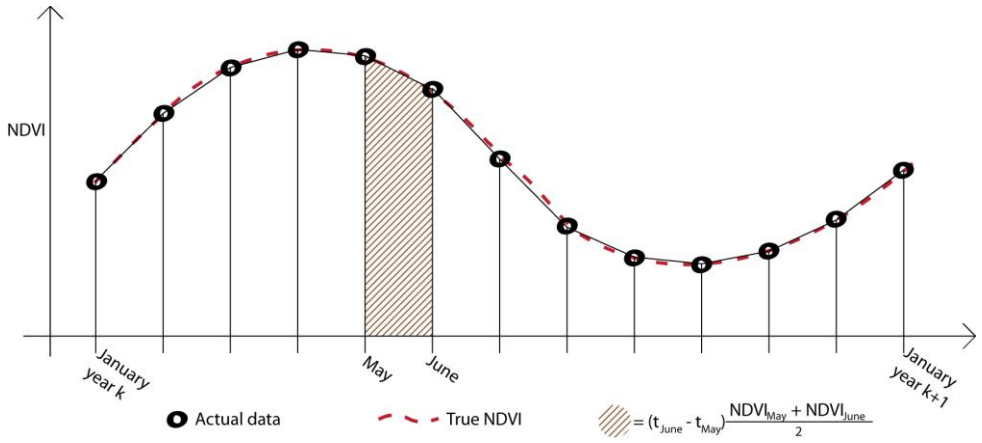


Figure 21 - How the trapezoidal rule is applied to approximate the NDVI integral

In Figure 21 it is displayed how the NDVI integral is estimated for the area between two known points on the curve, this is then summed for a year to approximate the integral of the whole year:

$$\int^{year_k} NDVI\ dt \approx \sum_{i=Jan_k}^{jan_{k+1}} (t_{i+i} - t_i) \cdot \frac{NDVI_i + NDVI_{i+1}}{2}$$

4.1.3.3 NPP regression

The yearly net primary production (NPP) is available for the years 2000-2010 and the NDVI integrals have been calculated for the years 2001-2013 which means, that there is an overlap of the years 2001-2010. The correlation between the NPP and the NDVI integral can be found from a simple least squares regression, as the relationship is thought to be linear. As both datasets represents continuous raster data, it is possible to compare points individually. After resampling the data to grid as explained in section 4.5.1 Gridding, corresponding pixels from the two datasets can be extracted in pairs. The NDVI and the NPP behave very differently on water, therefore the regression is only applied to the cells near the centre of the data extent, because this excludes proximity to the ocean.

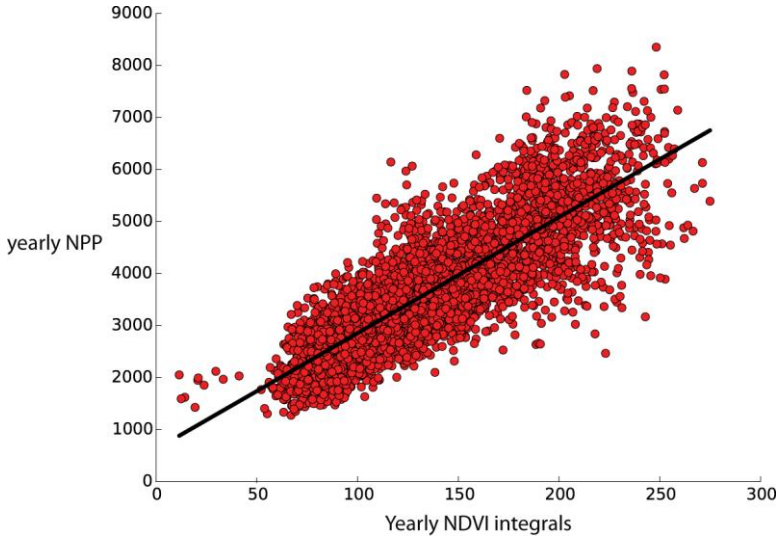


Figure 22 - Result of the regression

On Figure 22 all data used in the regression is plotted. There is a clear linear trend in the data, but also a considerable error. The ordinary least square fit yields: $NPP \approx 626.9 + 22.29 \int^{year} NDVI\ dt$ with an average error of 468 $\frac{kgC}{m^2}$.

4.2 Erosion index

The erosion index is based on RUSLE as described in section 2.1.3.1 Water erosion; $Erosion = R \cdot K \cdot S \cdot L \cdot C$, excluding the P-factor and any constants. The calculation of each factor in the erosion index is also explained there. Only the slope length factor remains a topic, as it is less simple than the others. The erosion index is based on the following data:

- Rain data. Used in the R-factor
- The K-factor is an external dataset in it self
- EU-DEM. Used in the L and S factors
- Desertification index. Used in the C factor, described in section 4.1 Desertification index.

The K-factor and the EU-DEM are explained in the following. The rain data along with other climatic datasets are explained in a later section, the desertification index, has already been discussed in the previous section.

4.2.1 ESDAC - K-factor

The K-factor is provided in 500m resolution by the European Commission – European Soil Portal of the European soil data Centre (ESDAC). It is based on 22000 soil samples collected across Europe in 2009 within the project LUCAS (Panagos et al. 2014, 143). The data is designed to be included in USLE and RUSLE calculations of soil erosion.

4.2.2 The EU-DEM

The EU- Digital Elevation model is likewise provided by the European commission for free download online. It is in 25 meter resolution (EEA 2013). The DEM has been computed by combining the SRTM near global DEM from 2000 and the ASTER GDEM from 2009.

4.2.3 Processing

Every element of the erosion is created from presented data using simple raster calculations. The exception is the L-factor, for which the processing will be described in the following. The following steps constitute the processing of the erosion index data:

- L-factor calculation
- Raster calculations as described in section 2.1.3.1 Water erosion
- Normalisation

4.2.3.1 L-factor calculation

In section 2.1.3.1 Water erosion the slope length factor (L-factor) was described to be given by the formula

$$L = (\frac{\lambda}{22.1m})^M$$

where M is derived from Raster calculation based on the DEM and λ is the horizontal length of the slope. This is adopted from the original RUSLE (Renard, et al. 1997, 105). The maximum slope length at a given point on a DEM, can be calculated with ArcGIS Hydrology toolset, by

first Running the tool Flow Direction and then the tool Flow Length. The accuracy of this however is very dependent on the DEM resolution. Interruptions in the flow can occur without it being recorded in the DEM because it is on smaller scale than the DEM. Due to the relative coarseness of the DEM, the slope lengths may be grossly overestimated. It has not been possible to find any best practices; therefore an experimental approach with a number of maximum lengths (λ_{max}) is adopted. The model will be run with several maximum slope lengths to see what difference it makes for the final result. To illustrate the difference a maximum slope length makes, three histograms has been plotted on a logarithmic scale:

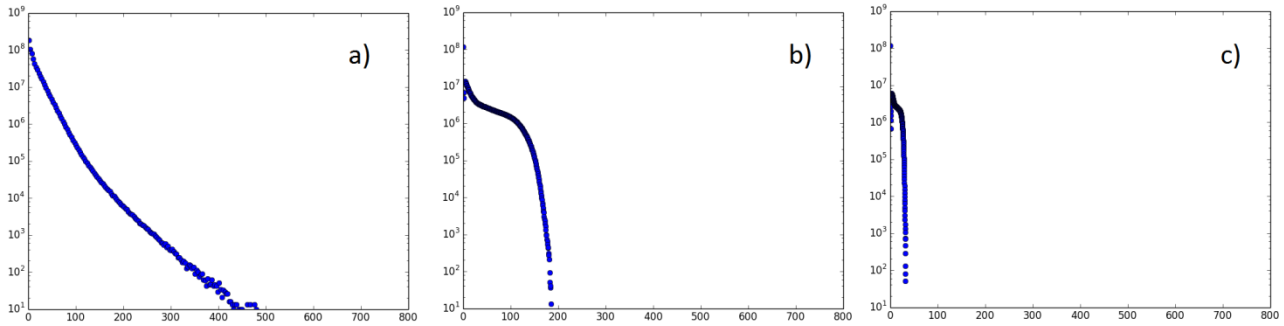


Figure 23 - L-factor histogram, a: without λ_{max}, b: with λ_{max} =1000 m, c: with λ_{max} = 100 m.

Figure 23 displays how the L-factor distribution is altered with a maximum slope length (λ_{max}). These are calculated from the original DEM in 25m resolution before resampling for modelling.
Without a maximum, the L-factor goes beyond 700 for a few pixels indicating that the erosion happens more than 700 times faster in these pixels.

4.3 Climatic data

The climatic data plays a role in several different factors; the desertification index depends on rainfall or soil moisture, and the erosion index depends on the erosivity of the rain. But the changing climate is also included as factors separately. The climatological data is the only source of data used, which is needed to extend into the future. Every other dataset is either assumed constant or estimated as a model output. This is possible due to the availability of publicly free climate modelling outputs in the Ensembles project.

4.3.1 Ensembles – climate data

The Ensembles project was initiated by the European Commission, in order to inform researchers and decision makers about climate change (Linden and Mitchell 2009, 3). Ensembles is a series of research themes of which the third (RT3) was *"Formulation of very-high-resolution Regional Climate Model ensembles for Europe."* (Linden and Mitchell 2009, 4). This has resulted in a range of climate model outputs with uninterrupted data from 1951 to 2100, for different climate scenarios with different models. Within the Ensembles project the EHTZ-CLM climate model in 25 km spatial and 1 day temporal resolution by Böhm et al. (2006) has been used. It is a local realization of the general circulation model (GCM) HadCM3 by the Hadley Centre. This particular model was selected as it was run with the needed output variables and it is in 25 km resolution (rather than 50 km). It was run for the A1B climate scenario, which is from the A1 family of scenarios that describe a world of rapid economic and population growth until 2050. A1FI is in the same family, but here energy demands are met by a Fossil Intensive energy production. The A1T is where the energy demands are met by more by non fossil energy production, and the A1B lies somewhere in between, with B for balanced energy production (IPCC WG3 2000, 4). Due to time constraints is has not been possible to test different climate models or scenarios, and the significance of differences and uncertainties within these models has not been tested.

4.3.2 Processing

Three climatic factors are included in the model:

- Rainfall data is included both in the Erosivity index of the Erosion index, and as a separate factor
- Soil moisture
- Temperature

The processing of each dataset consists of simple arithmetic as discussed in 2.1.3.2 Climate change, and the processing will not be discussed further.

4.4 Topographic data

Lastly the topographic data included will be described. Applied in the model are surface water; lakes and rivers as well as urban areas. Even though all of these topographic features are dynamic both in terms of size and probably also in impact on desertification, they are assumed static. Modelling the temporal development of these topographic features is beyond the scope of this project.
Including water bodies in this model is important as water is the primary constraint on primary production. Including urban areas is important for two reasons. Firstly urban areas physically obstruct natural development. Secondly urban areas will not be more and more desertified, as they are a special type of land use, which responds very differently to changes in climate and erosion.

4.4.1 ECRINS – Surface water data

The European Catchments and Rivers Network System (ECRINS) is the name a European hydrographical model containing both river and lake data in Europe. The data is being used for hydrographical purposes in by European Union institutions such as EEA and WISE (EEA 2012, 8).

ECRINS river data contains an information attribute named "SurfC", which holds information on the cumulated catchment area upstream (EEA 2012, 89). This field can be used as a size factor to make large rivers (rivers with large catchments) count more than small rivers.

4.4.2 Corine land cover – Urban areas

Corine Land Cover (CLC) is a European project to create a land cover map of Europe derived from satellite imagery. CLC 2006 provides data in 100 m spatial resolution with land cover types defined in 44 different classes (EEA 2006, 6-7).

CLC classes 1-6 are defined like this:

1. Continuous urban fabric
2. Discontinuous urban fabric
3. Industrial or commercial units
4. Road and rail networks and associated land
5. Port areas
6. Airports

These 6 classes represent the two first super classes; urban fabric and Industrial, commercial and transport units. These are the ones relating to urbanity and the ones that are extracted to be included in the model. The remaining classes 7-44 represents other land uses which are not included.

To include the rivers, lakes and urban areas in the model, they must be converted to a suitable raster format the processes are very similar for the three datasets.
Even though the Corine data is already in a raster format this is not suitable for raster calculations because it is a categorical dataset. There is no mathematical relationship between

the classes; class 4 is not twice the amount of class 2. To account for this the Corine data is treated as a discrete vector dataset.

4.4.3 Gridding of topographic vector data

This process has the goal of quantifying the discrete topographical data, so they can be represented as a continuous raster datasets which can be included in the model.

The impact on desertification of either of the topographic datasets is assumed to be proportional to the amount of each feature in a cell times a weight factor.
Rivers are weighted by their cumulated catchment area (SurfC), so if cell A contains 200m river with catchment of 10 km², and cell B contains 400m river with catchment of 20 km², cell B has four times "as much river" as cell A.

Urban areas are assigned a weight of 2.0 for Continuous urban fabric, and a weight of 1.0 for the other classes. So if Cell A contains 400 m² industrial and commercial units and cell B contains 200 m² continuous urban fabric, cell A and B are equally urban.

Lakes are not weighted; they are measured only by their size.

The gridding follows these steps, illustrated in Figure 24:

1. Spatial intersection between the grid and the vector features, dividing the features at each grid border.
2. Calculation of measure of each divided feature.
    a. Rivers: Length times SurfC
    b. Urban: Area times 2 for continuous urban fabric, times 1 otherwise
    c. Lakes: Area
3. Spatial join, matching each divided feature with a cell.
4. Add up values.

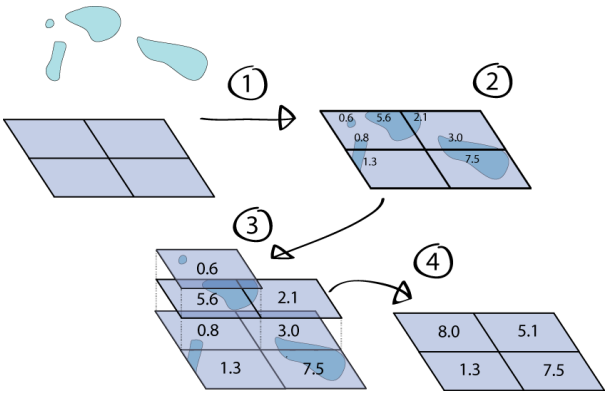


Figure 24 - Gridding process for topographical vector data

4.5 Non data specific processing

Each model variable has now been processed to a state where it is ready to be part of desertification modelling. Two final processing steps remain before the data can incorporated into this particular model. These processing steps are common for all data sets.
For the cellular automaton to function properly, data must be put in to a regular square grid, as explained in section 2.1.3 Desertification cellular automata. This process is known as gridding.
Secondly for the data to be modelled properly by the artificial neural network, the data values must fall in the range of [0,1], to achieve this data must be normalised.

4.5.1 Gridding

As presented in the previous parts of this chapter, data from different sources is not in the same format. Data does not have the same spatial resolution or the same projection. Classical cellular automata requires all data sources to be in the same grid, therefore all data must be fitted into a grid. This process is known as gridding. Most data sources have a finer resolution than the grid, and the gridding process will lead to a loss of information, but for the climate data, which is in 25 km resolution, the information density will have to be increased. Both the downscaling and upscaling of information density, happens with interpolation. The gridding was performed in ArcPy with the ProjectRaster function. This function reprojects an input raster to a chosen projection with a chosen cell size and an anchor point. This way all data can be projected to the same grid. The change of information density called resampling, is handled by a chosen interpolation method. ArcGIS offers four choices of resampling technique (ESRI 2014):

- **Nearest neighbour** simply assigns the nearest value from the input dataset to each cell.
- **Majority** assigns the most popular value from a filter window.
- **Bilinear interpolation**, which interpolates using the four nearest cell values.
- **Cubic convolution**, which interpolates using the 16 nearest cell values.

Nearest neighbour and Majority can be instantly ruled out as they are primarily targeted for categorical data, such as land cover types or soil types. Both the Bilinear interpolation and the cubic convolution are designed for continuous data, but based on different algorithms.

Bilinear interpolation and cubic convolution will be tested in two different examples: For low resolution data, that needs to be fitted in to a cell size smaller than the original data, and high resolution data that needs to be fitted into a grid of smaller size. In Figure 25 gridding into 10 km cells of 25 km data is illustrated. There is very little noticeable difference between the results of the two algorithms. Both reproduce the spatial pattern very similarly. One important thing to note, is that the bilinear interpolation decreased the value range of the data from [50558, 0.0005] to [48895, 0.0052], while the cubic convolution creates values outside of the original value range. Cubic convolution even comes up with negative values. Negative values for this specific dataset are problematic as there cannot be a negative Erosivity, because that would imply a negative amount of rain. Not only are the negative values problematic in a contextual sense, but it will also have implications for the normalisation, if negative values occur where they should not.

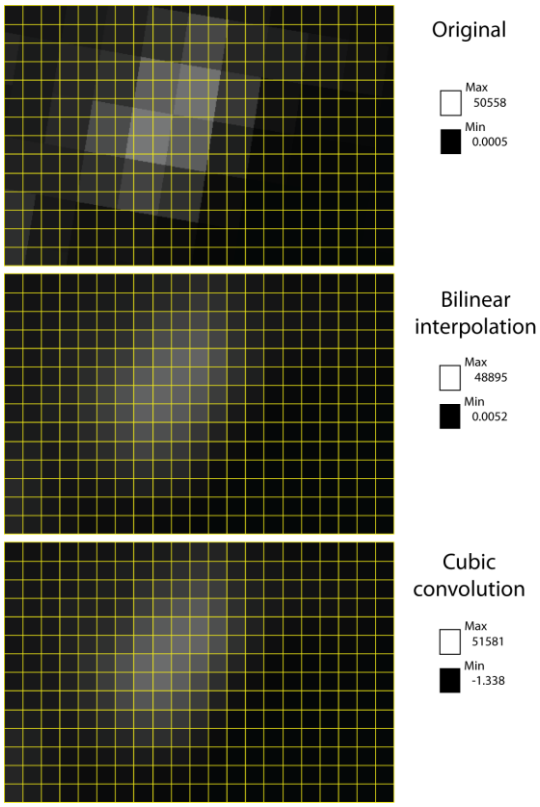


Figure 25 - Bilinear interpolation and Cubic convolution of Erosivity data from Ensemble. 25 km. to 10 km. resampling.

Figure 26 displays the result of resampling "the other way around", going from fine grained raster size of 25 m to the same 10 km cells as before. Here the two results are also very similar. In blue two mountain ridges are marked on the original data, these features can also be identified in the gridded results. In red there are three cells which seem to be miscalculated. The top one seems like it should be darker, and the two next ones should be lighter. This problem is caused by the fact that the interpolation methods use the 4 and 16 "closest cells" respectively. Closest here, refers to closest to the centre of the output cell. In the three example cells values in the middle does not correspond well with the averages, and the algorithm fails. The problem will be proportional to the difference between cell sizes. The problem is tolerated as it is expected to be small and because no other resampling methods are available within the ArcGIS environment. One might also note, that the max values in both cases lies way below the max for the original, this is caused by the way the

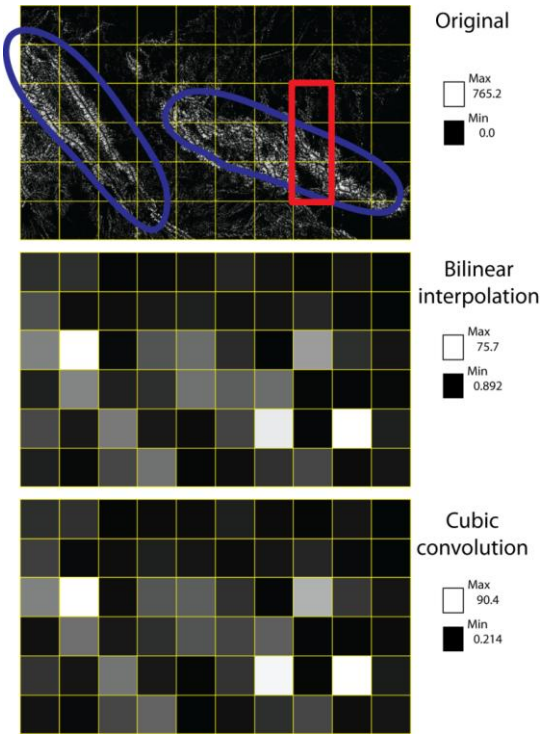


Figure 26 - Bilinear interpolation and Cubic convolution of L factor data based on the DEM. 25 km to 10 km resampling.

algorithms are calculated by fitting smooth curves on the data. This is not considered a problem because it means that data will be less sensitive to a few extreme pixel values. Very few pixels in this specific dataset have a value of above 100, so these few pixels will have less of an impact on the dataset.
Overall the two methods yield similar results. To avoid values outside the original range (especially negative values) bilinear interpolation is the chosen method, which is applied for all datasets.

4.5.2 Normalisation

The aim of the normalisation process is, that all individual pixel values remain inside the range of [0,1]. This can be achieved, by searching for the maximum value throughout all years, and then simply dividing each raster dataset by this value.

The maxima are found using the ArcPy function GetRasterProperties.

It is not necessary to search for minima, as no model inputs deal with negative values, and simply dividing the raster data by the maximum value will put all values inside the range of 0 and 1, and it will keep the spatial and temporal ratios of values between pixels.

4.6 Conclusions

In this chapter it was described which data is applied for the desertification modelling. It was described how the net primary production values were derived from monthly NDVI data, as a part of the desertification index. It now remains to be investigated whether the yearly rainfall data or the yearly soil moisture averages should be combined with the NDVI integrals, to form the desertification index.

It was described how the erosion index has three possible rasters based on different slope length maximum limit (λ_{max}). It now remains to be explored which limit, or if any limit at all will yield the best predictive erosion index.

It was also described how the topographic data could be included in the model, which only accepts continuous raster datasets.

Finally it was described how the datasets could be conformably arranged, to meet the requirements of the cellular automaton and artificial neural network modelling.

All data processing not described in this chapter, is left out because it is performed using simple raster calculations based on equations stated in chapter 2 Theory.

5 Modelling

In the following chapter the modelling of the desertification simulation will be described and discussed. This will be laid out in six subchapters. The first three subchapters describe the model structure, and modelling choices; first an overview of the model will be drawn and then in the two subsequent subchapters the two main components of the model, the cellular automaton and the artificial neural network will be discussed individually.
Next the two candidates for desertification index will be compared, to see which one is better fit for its purpose. The next subchapter deals with the practical implementation of the model, which includes an explanation of how the implementation code works.
Finally the sensitivity of the implemented model will be tested.

5.1 Model overview

The model is a fuzzy cellular automaton with transition rules defined by the regression capacities of an artificial neural network. This means that the desertification, that is being simulated, is given by a scalar between 0 and 1 corresponding to the degree of desertification in each cell. It is modelled in a discrete grid in discrete time steps. The degree of desertification can be found in two different ways:

- For model training, meaning for any point in time until now. The desertification index is calculated by remotely sensed data and historical meteorological data like described in the previous chapter.
- For model simulation, meaning for any point in time after present, where remotely sensed data and historical meteorological data will not be available, the desertification index is the output of the model.

For the model to be able to run iteratively into the future, each input of the model has to have two sources of data, just like the desertification index as described above; one for the model training and one for model simulation.

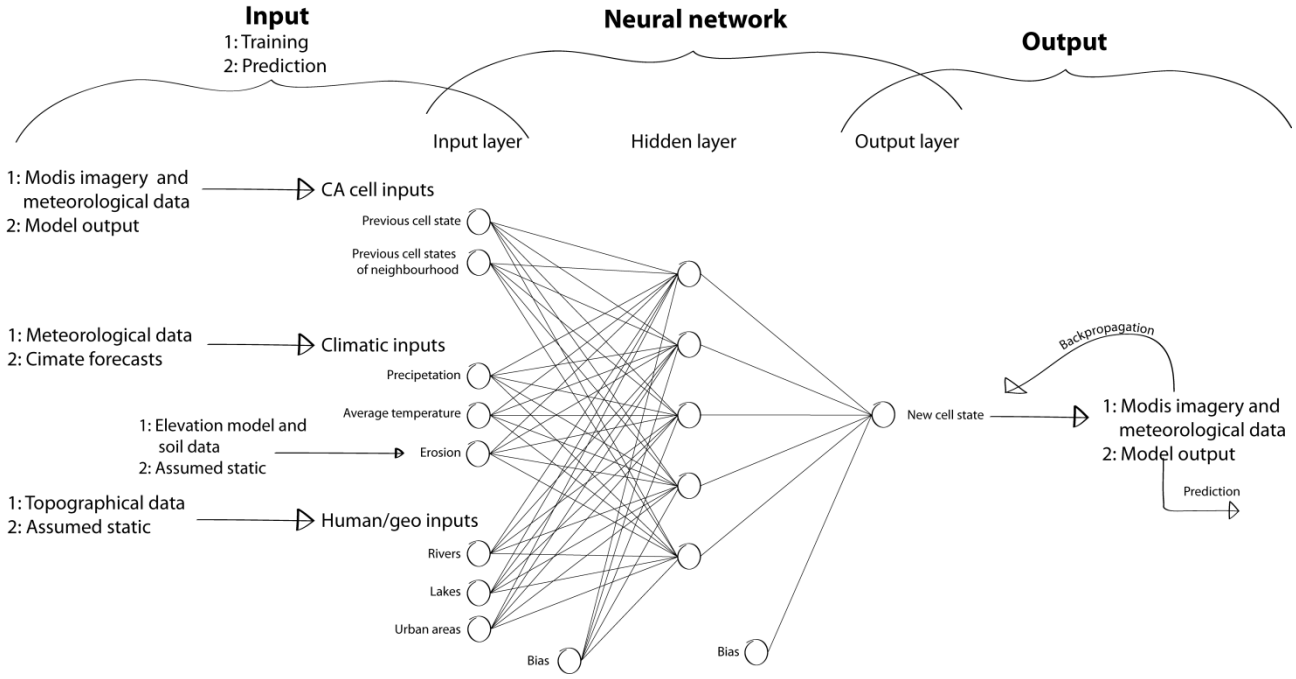


Figure 27- Model overview

On the left, the model inputs are displayed in three main categories: The cellular automata related inputs, the climatic inputs and the topographical inputs. These inputs are then processed through an artificial neural network to compute an output value for each cell in a grid this output is used to compare with observed data for training the neural network and to simulate the desertification process.

5.2 Cellular automaton modelling

As described in section 2.2 The cellular automaton, cellular automata are a family of models which holds a range of properties which determines which kind of CA model it is. Setting these properties appropriately to match the aim of the model is what is called CA modelling. So in the following these parameters will be discussed and selections will be explained.

5.2.1 The properties

The following discussion is based on the properties described in section 2.2.1 Defining characteristics.

5.2.1.1 The grid

The grid is a two dimensional square grid. It is in two dimensions because the phenomenon of desertification is happening primarily in two dimensions. This is realised in the Lambert Azimuthal Equal Area projection, which ensures that the cells are in fact equal area across the studied area. The implication of having this projection is that the angles will not be equal across the studied area, so the cells will in fact not be perfectly square, but slightly skewed one way or the other. The consequences of this are assumed to be almost nonexistent due to small geographical scale, and the distance from the projection origin.

The grid also needs a size. Unless all used data is recorded in a way that favours a certain cell size, for example if all data was already in the same grid, selecting a grid size is something that is difficult to substantiate. There should be enough cells to see spatial patterns; there should be enough cells for meaningful model training. Too many cells will mean longer calculation time and problems with resampling data with lower resolution to this small cell grid.

The cell size is tested for three values; 5 km, 10 km and 20 km to see how sensitive the model is to this parameter. This test is described in 5.6 Model sensitivity.

5.2.1.2 The cell states

The cell states, which describes the level of desertification in each cell, is selected to be a fuzzy variable from 0 to 1. As explained in 2.1.2.2 A desertification index, this value is based on the primary production per available water unit. This means that higher values correspond to more primary production per available water unit, which means desertificated cells will have low values. The desertification index is given by yearly net primary production divided by the water availability, and as explained in 4.5.2 Normalisation this is normalised to always remain in the range of 0 and 1.

5.2.1.3 Time

In cellular automata, time is kept in discrete time steps of a given length. As much of the climatic data this model is based on, has a specific intraannual trend the time steps should not be less than one year. As an example, the primary production happens very differently throughout the year, and the year cannot be subdivided in a meaningful way. How many years one time step should consist of, is a balance of the amount of available data. On the one hand, more years in a time step would ensure more stable trends and more accurate results and the

phenomenon of desertification might not be necessary to study on high temporal resolution. On the other hand longer time steps will decrease the number of available time steps for training. The MODIS NDVI product is only available for the complete years 2001-2013 which is 13 years. This leaves the following options:

Years in a Timestep	Example	Number of Training datasets	Years lost
1	in: 2001, out: 2002	11	0
2	in: 2001-2002, out: 2003-2004	4	1
3	in: 2001-2003, out: 2004-2006	2	1
4	in: 2001-2004, out: 2005-2008	1	1

Figure 28 - Four different time step options

All these options will yield different results in terms of accuracy and temporal resolution. The model can be adjusted to use any of these settings.
The 1 year resolution is used for all results, this implies that no data is lost and there are 11 available training dataset pairs.

5.2.1.4 The transition rule

How cells transition in time, should be a continuous function of desertification level and external factors. As displayed in the model overview, Figure 27 in section 5.1 Model overview, this will be an eight dimensional function as there are a total of eight inputs. To find a fitting function could prove difficult without much knowledge about how the function is supposed to behave, also studying the function in different domains across all the dimensions, with all the unknowns of the modelling seems like a task for a computer. While extremely fast at complex calculations computers are poor at seeing patterns and making decisions. Therefore the transition rule function is defined in a flexible way to support whatever configurations the model is tested with, by always seeking the smallest possible error. This is achieved through an artificial neural network regression, the modelling of which is the focus of the subsequent subchapter.

5.2.1.5 The neighbourhood

As discussed in section 2.2.3 Desertification cellular automata, a small neighbourhood seems more appropriate. Therefore the following two neighbourhood sizes will be tested: the Moore neighbourhood, consisting of the 8 surrounding cells, and the double Moore neighbourhood, consisting of the Moore neighbourhood and the 16 cells on the outside of the Moore neighbourhood.
The neighbourhood values are included by the ArcPy function Focal statistics. This allows the summing of all cells in a 3x3 or 5x5 neighbourhood. The middle cell of the neighbourhood has already been included so this value is subtracted, and finally the value is divided by the number of cells included 8 (3^2 - 1) for the Moore neighbourhood and 24 (5^2 - 1) for the double Moore neighbourhood). The division is performed to ensure that the value stays between 0 and

1. On the edge of the dataset, it will not be possible to collect data for the entire neighbourhood. This will cause the neighbourhood values to be lower than in other parts of the grid, and make the calculations for these cells go sour. This problem is handled by only dividing by the number of included cells.

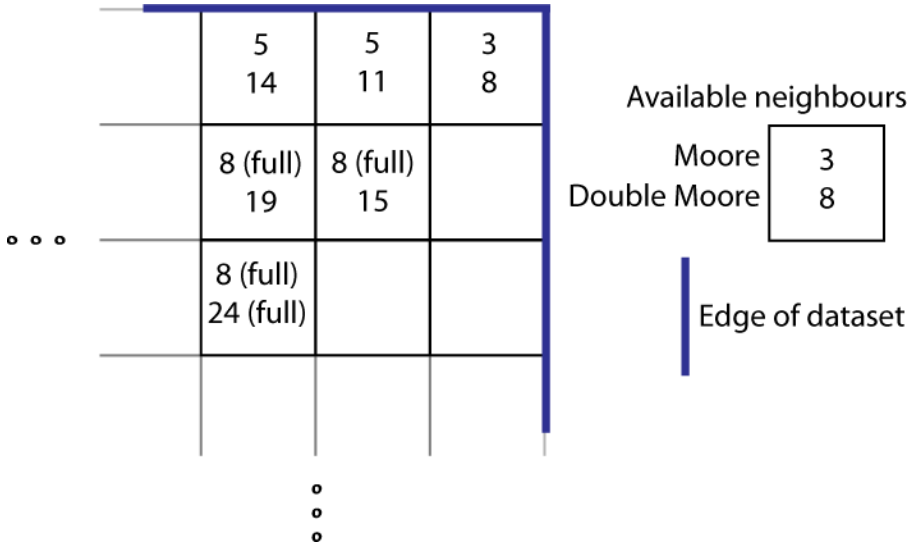


Figure 29 - Number of neighbours of edge cells

Figure 29 shows an example of how the number of available neighbours is counted for the Moore and the double Moore neighbourhoods, near the edges of the dataset. This can be computed by using the same ArcPy function, Focal statistics, on a raster grid where all cell values are 1.

When the above is performed as explained, all neighbours are equally weighted, and this may not be correct, as discussed in 2.2.1 Defining characteristics. To introduce a decay function, the neighbourhood values must be weighted by a function of how far away they are. The functions tested will be $\frac{1}{dist}$ and $\frac{1}{dist^2}$. This is also achieved with ArcPy Focal statistics. In this case the Focal statistics sums using a weight matrix. A script for creating these matrices can be found in Appendix 2.

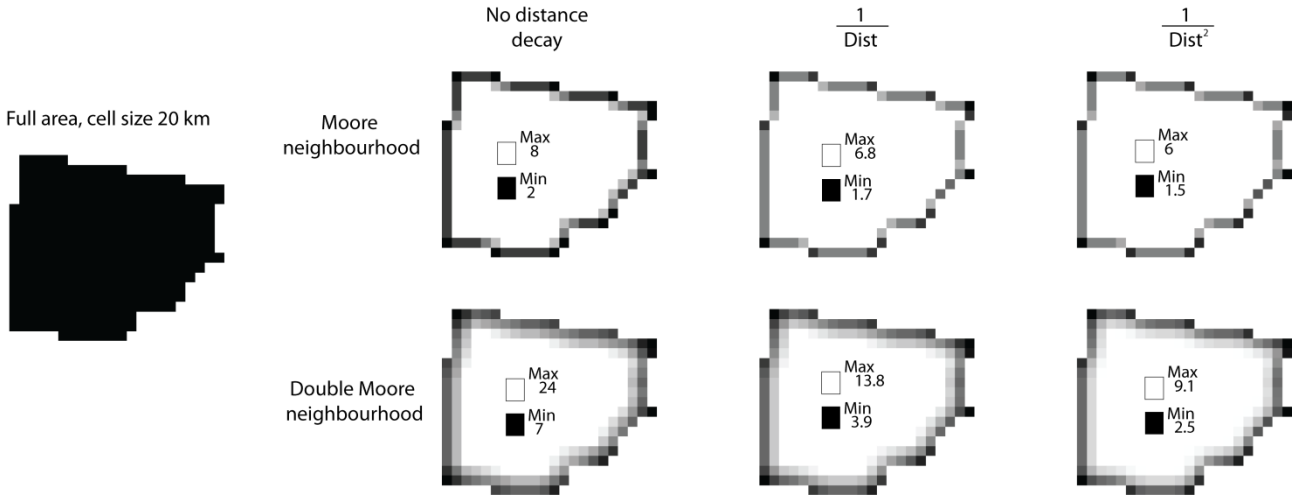


Figure 30 - Focal statistics using a raster of ones. White cells in the centres represent the maximum values.

Figure 30 displays the different distance decay functions and neighbourhood sizes applied to a grid of ones, with cell size 20 km The rasters displayed in Figure 30 show how many weighted cells each cell will include in the calculation and therefore they can be used to normalise the neighbourhood calculation.

5.3 Artificial neural network modelling

Just as the cellular automata part of the modelling has a range of properties to be determined, so does the ANN. These will be discussed in the following.

5.3.1 The activation function

All values used in the modelling are measures of something; primary production, erosion, nearby surface water etc. They are measuring the amount of something, and are therefore never negative. There cannot be a negative soil erosion, primary production or surface water. As all values lie in the positive range, they can be normalised to lie between 0 and 1, just by dividing with the maximum value (see section 4.5.2 Normalisation). This is the range where the logistic sigmoid function operates, therefore this activation function is chosen for the artificial neural network, and not the hyperbolic tangent.

$$A(x) = \frac{1}{1 + e^{-x}}$$

5.3.2 Network architecture

The network architecture describes the number of layers, neurons and connections of the ANN. Here only considered are the feed forward ANNs with one hidden layer. For both the input layer and the hidden layer there will be one bias neuron transmitting ones. The output layer will have one neuron, which is the output desertification index for the next time step. The input layer will have one neuron for each input value plus one for the bias. The input values are:

- Desertification index
- Neighbourhood value, calculated from the desertification index of surrounding cells
- Erosion index
- Average temperature
- Total rainfall
- Nearby rivers
- Nearby Lakes
- Nearby urban area
- Bias

So there will be a total of 9 neurons in the input layer.

The number of neurons in the hidden layer is less simple; there are no easy rules to determine which number is the best (Li and Yeh 2001, 1460). The optimal numbers of neurons usually lie between the number of input neurons (9) and the number of output neurons (1) so a good guess would be around 5. Therefore the network will be tested with both 4, 5 and 6 neurons in the hidden layer, and one of these neurons will be a bias neuron.

5.4 Modelling with the desertification index

As was concluded in section 2.1.2.2 A desertification index, there are two candidate desertification indices:

- Above ground net primary production per mm rainfall
- Above ground net primary production per kg water of soil moisture

The following section is a discussion of which one to use in the model. The desertification index is the most important factor in the model. It is the internal factor which the modelling is based on, and the one which the output of the model should resemble. The index of desertification should therefore ideally match the initial ideas about how desertification evolves in time. As discussed in section 2.1.2.2.2 The variability of weather, desertification is a slow process so the index should be stable, cells should not change dramatically from one year to the next (unless of course the land use is changed).

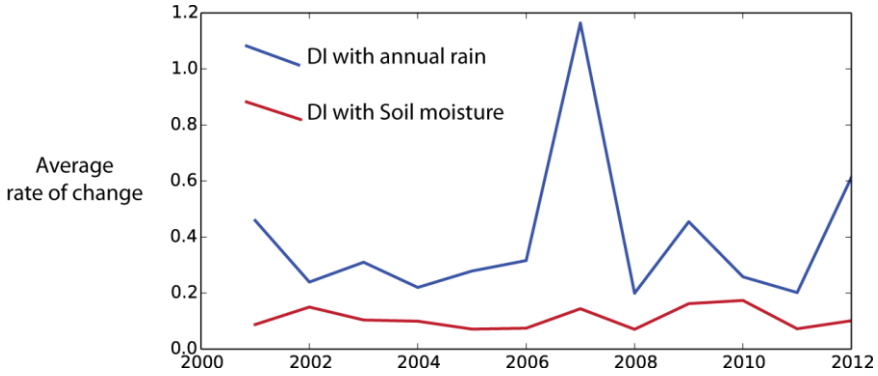


Figure 31 - The rate of change of the two candidate indices

Figure 31 clearly depicts the difference in stability of the two candidate indices. The desertification index with soil moisture content as a normaliser is not stable. All training pixels show on average almost 10% change from year to year, but it is much more stable than its rival candidate. In 2007 large parts of the training area experienced major rainfall, which could not be converted into primary production. The soil moisture was less sensitive to this. On this basis, the more stable and less noisy desertification index, based on soil moisture content is chosen.

The structure of the model and all of its components are now drawn. The cellular automaton, with the artificial neural network as transition rule, is modelled to simulate desertification.

5.5 Model implementation

In the following the practical implementation procedure of the model is described. The implementation of the model was performed in Python using the ArcPy module which allow ArcGIS raster calculations in the python environment. In the following the implementation of the model will be explained. Only the key lines of code will be described, the model code in its entirety can be found in Appendix 1.

The following subchapter consists of two sections; one concerning the implementation of the artificial neural network, and one concerning the CA part of the model. The reader might notice that this is the reversed order as compared to earlier in this chapter and in previous chapters where; first the cellular automaton was explained, then the artificial neural network. This is because the artificial neural network can be viewed as a component of the CA model – the transition rule. For implementation however, the ANN comes first. First the all components must be defined, and then they are applied. So the description in the following matches the order of the actual code.

5.5.1 Artificial neural network implementation

In the following the artificial neural network part of the model implementation is described. The ANN part of the model consists mainly of three methods; one for running the network, one for arranging raster data as trainable sets of pixel values and one for training the network. These three methods are described in the following.

5.5.1.1 General implementation strategy

The artificial neural network was implemented as three lists, one for each layer (input, hidden, output) and each neuron was implemented as class object. This is not an efficient way of implementing the neurons, as it blocks for the use of matrix multiplication. The focus in the implementation is to give the reader an overview of what happens and how it happens, not computational performance.

Weights are initiated in the range of 0.1 and -0.1, in all experiments the same initial weights have been used.

5.5.1.2 The Run method

The run method generates the output from the input, this is the method which is meant to be run when the network is trained. Basic arithmetic operations like +,-,* and / in python both handles numbers and Rasters, when the ArcPy module is initialised. The activation function is implemented using ArcPy raster calculation interface so it also both handles rasters and single numbers:

```
#sigmoid
def activation(a):
    return 1/(1+Exp(-a))
```

where the exponential function `Exp(-a)` is an ArcPy raster calculator function that can handle both individual numbers and full rasters cell-wise.

In this way the run method can both be used to process single numbers, which will be needed when the network is trained, and for a whole raster simultaneously, when the network is fully trained.

The run method takes a list of input values, equivalent to the number of neurons in the input layer.

- First an iteration through input values, to assign them the neurons in the input layer, named "inputs".

```
for i in range(numinputs):
    inputs[i].val = inputnumbers[i]
```

- Now all values from the input layer neurons are linearly combined to the hidden layer and processed in the activation function. This is performed for every non bias neuron in the hidden layer.

```
for neuron in hidden1:
    if not neuron.bias:
        sumval = 0
        for inpval in neuron.getConsTo():
            sumval += inpval.val*inpval.getWeight(neuron)[0]
        neuron.vsum = sumval
        neuron.val = activation(sumval)
```

- Then an almost identical process is performed to find the incoming and output values of the output layer neurons.

```
for neuron in outputs:
    sumval = 0
    for inpval in neuron.getConsTo():
        sumval += inpval.val*inpval.getWeight(neuron)[0]
    neuron.val = activation(sumval)
    neuron.vsum = sumval
```

- Finally the values of the output neurons are collected into a list and returned as a function result.

```
outputvals = []
for out in outputs:
    outputvals.append(out.val)
return outputvals
```

5.5.1.3 The training set generator

This method generates a training set, based on inputted rasters and an expected output raster. A training area has been defined as a rectangular area in the centre of the research area, see section 3.2 Geographical research area.

The method takes the input and output rasters.

- First all rasters are converted into numpy matrices, for the training area.

```
trainingcorner = Point(3082508.745,1723162.51)
nprasters = []
for ras in inputrasters:
    nparr = arcpy.RasterToNumPyArray(ras,trainingcorner,27,20)
    nprasters.append(nparr)

npout = arcpy.RasterToNumPyArray(expectedoutput[0],trainingcorner,27,20)
```

- Then each point on the raster is treated as a training set. Lists of spatially corresponding points are grouped together as training sets, so the k'th element of 'ins' are the input pixel values of the same area as the k'th element of 'outs'.

```
ins = []
outs = []
for i in xra:
    for j in yra:
        runvals = []
        for ras in nprasters:
            runvals.append(ras[j,i])

        ins.append(runvals)
        outs.append([npout[j,i]])
```

- Then ins and outs are returned.

5.5.1.4 The Rprop algorithm

The network is trained with the Rprop algorithm as described in 2.3.2.1 Training algorithms. Rprop is implemented as a function that takes a list of input values for each training set, the corresponding output values (the output of the training set generator function) and an integer determining how many iterations of the Rprop algorithm that should be carried out.

- For each training iteration run the following:
  - For each training set of corresponding inputs and expected output:
    - First the run method is executed to find out how different the output is from the expected output – the error
    - Then the node delta for each node in the output layer are calculated.

      $\delta_{output} = -E \cdot A'(sum)$, where the sum is the incoming value not the outputvalue

```
for neuron in outputs:
    neuron.nodeDelta = -error*actder(neuron.vsum)
```

    - Then the error is backpropagated to the hidden layer, and the node deltas for the neurons in this layer are calculated by summing weights and node deltas for all outgoing connections:

$$\delta_{hidden} = A'(sum) * \sum^{cons} \delta_{con} \cdot w_{con}$$

```
for neuron in hidden1:
     if not neuron.bias:
         ndelta = 0
         for con in neuron.getConsFrom():
             ndelta += neuron.getWeight(con)[0]*con.nodeDelta

         neuron.nodeDelta = ndelta * actder(neuron.vsum)
```

- Then the gradient for each weight is calculated or recalculated. For each weight, five numbers are stored; the weight, the gradient, the previous weight change, the previous gradient and the learning rate.
  The gradient for the current data point is simply the negative product of the neurons outgoing value and the nodeDelta of the neuron it is going to. This gradient is then added to the recorded gradient. This will result in the sum of all gradient calculations for the same weight, and thereby in the end be negative if the gradients are mostly negative and vice versa. Only the gradient is updated in this step (`info[1]`) the other elements are left untouched.

```
    for layer in layers[:-1]:
        for neuron in layer:
            for toNeuron in neuron.weights.keys():
                #[weight,gradient,dweight,previous gradient,lr]
                this_grad = -toNeuron.nodeDelta*neuron.val
                info = neuron.weights[toNeuron]
                info[1] += this_grad
                neuron.setWeight(toNeuron,info)
```

- The looping of each training set is now exited. So the following is only executed once per iteration, and not for each training set.
  At this point we know if the gradients are overall positive or negative in the training set so based on the Rprop algorithm the weights are updated. See 2.3.2.1 Training algorithms.

```
for layer in layers[:-1]:
    for neuron in layer:
        for toNeuron in neuron.weights.keys():
            #[weight,gradient,dweight,previous gradient,lr]
            this_grad = neuron.weights[toNeuron][1]
            prev_grad = neuron.weights[toNeuron][3]
            prev_lr = neuron.weights[toNeuron][4]
            prev_dw = neuron.weights[toNeuron][2]

            #same direction
            if this_grad*prev_grad >0:
                lr = min(prev_lr*Eta_plus,lr_max)
                dw = -np.sign(this_grad)*lr

            #opposite direction
            elif this_grad*prev_grad<0:
                lr = max(prev_lr*Eta_minus,lr_min)
                this_grad = 0
                dw = -prev_dw"

            #new direction (prev_grad = 0)
```

```
        else:
            lr = prev_lr
            dw = -np.sign(this_grad)*prev_lr

        newW = neuron.weights[toNeuron][0] + dw
        neuron.setWeight(toNeuron,[newW,0,dw,this_grad,lr])
```

This concludes the artificial neural network part of the model implementation. The usage of the ANN follows this structure:

- The modules are imported and the network is initialised.
- Training sets are generated from the rasters
- The network is trained using rprop algorithm
- The Run method can now serve as the transition rule for the cellular automaton

5.5.2 Cellular automata implementation

In the following the CA model implementation is described. In this part of the model implementation no methods are defined. Only the ones defined in ArcPy are used. The code displays the simplicity of cellular automata, as only a few lines of code is needed.

- First the neighbourhood weight matrices are defined. Four possible settings are used, these are defined by a txt document, as described in section 5.2.1.5 The neighbourhood, `_nbhsize==1` refers to the single Moore neighbourhood, and `_distdec == 'sqr'` refers to the distance decay using square distances.

```
if _nbhsize==1:
    if _distdec == 'sqr':
        nbhweights = os.path.join(kernalfolder,'mooresqr.txt')
    else:
        nbhweights = os.path.join(kernalfolder,'moore.txt')
else:
    if _distdec == 'sqr':
        nbhweights = os.path.join(kernalfolder,'2mooresqr.txt')
    else:
        nbhweights = os.path.join(kernalfolder,'2moore.txt')
```

- Then a normalisation raster is prepared. This aims to find out how many neighbours are included for each cell, and what are their weights. This is performed using `FocalStatistics` on a raster grid with the value 1 in each cell (called 'grid'). This raster uses the weight matrix chosen in the previous step.

```
nbh_norm = FocalStatistics(grid,NbrWeight(nbhweights), "SUM","")
```

- Now the actual neighbourhood value raster is calculated. This is performed in the exact same way, but now the desertification index is the input (called 'di'), instead of a raster of only ones. This raster is then normalised by division with the normalisation raster defined in the previous step.

```
nbh = FocalStatistics(di, NbrWeight(nbhweights), "SUM","")/nbh_norm
```

This raster 'nbh' along with 'di' and the other input rasters, can now be used in the already defined training and run methods. And this concludes the model implementation.

5.6 Model sensitivity

Now only a few questions remain open in the modelling process. These last uncertainties will be determined with experiments that aim to test which are the best options and how sensitive the model is to minor changes. Ideally all setup combinations would be tested, but due to time constraints this has not been possible.

Data from 2013 are withdrawn from the training data pool. The model is trained using all remaining data, and then the model is run for 2012 data to see how well the model predicts the 2013 data.

The variables tested are:

- Erosion index L-factor λ_{max}
  - No λ_{max}
  - λ_{max} of 1000 m
  - λ_{max} of 100 m
- Grid size
  - 5 km
  - 10 km
  - 20 km
- Neighbourhood size
  - Moore neighbourhood
  - Double Moore neighbourhood
- Decay function
  - $\frac{1}{dist}$
  - $\frac{1}{dist^2}$
- Neurons in the hidden layer
  - 4
  - 5
  - 6

In all experiments the neural network converged, each training process was stopped after 50 Rprop iterations.

5.6.1 L-factor limit

Whether there should be a limit to the λ in the Rusle L-factor or not, is tested in the following. The following three experiments are performed, with these settings in common:

Grid size: 10 km, neighbourhood size: double Moore, decay function: $\frac{1}{dist^2}$, hidden neurons: 4

- L-factor with λ_{max}
- L-factor with a λ_{max} of 1000 m and
- L-factor with a λ_{max} of 100 m

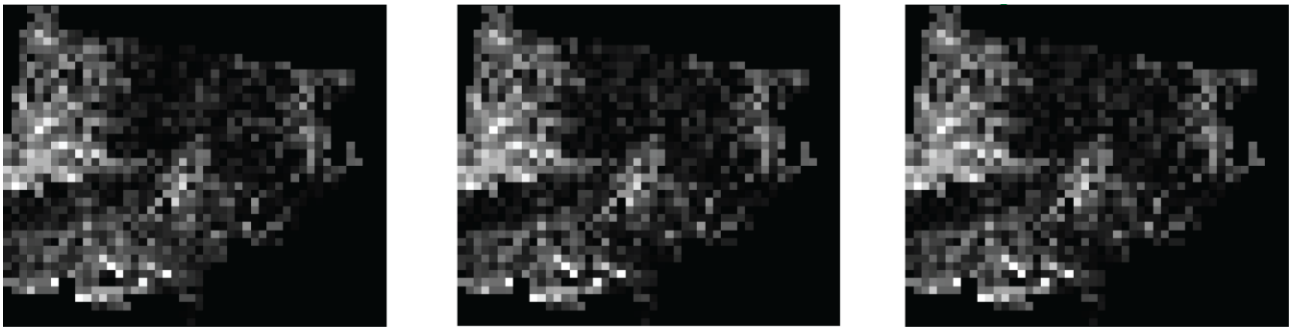


Figure 32 - Error rasters L-factor experiments. No limit, 1000 m limit and 100 m limit

Figure 32 shows the results of the three experiments in terms of absolute error, bright colours signify a high error(up to around 0.3 (White), and dark colours signify a low error, black is zero). The three results are very similar, they do well in the same geographic locations, and all of them does not do well in the north western corner. The average errors are measured to be:

- No λ_{max}: 0.052
- 1000 m λ_{max}: 0.054
- 100 m λ_{max}: 0.055

So almost no difference, but the experiment without a limit seems just a tad better. There is no guarantee that the no limit λ, is better with different settings for the other parameters. But at least the difference will most likely be very small.

5.6.2 Grid size

The following three experiments is meant to test the significance of the grid size. Just like in the prior set of experiments all parameters, except the one that is being tested, are held steady, while the grid size is varied.

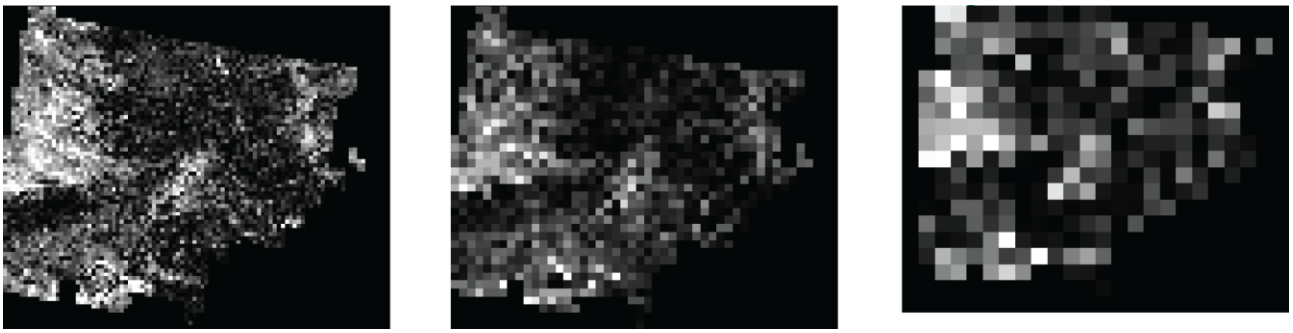


Figure 33 - Error raster from the grid size experiments. 5 km, 10 km and 20 km cells

Here the settings kept constant were: no λ_{max}, neighbourhood size: double Moore, decay function: $\frac{1}{dist^2}$, hidden neurons: 4. Again the results are quite similar, in terms of errors. The average errors are:

- 5 km: 0.051
- 10 km: 0.052
- 20 km: 0.055

It is unexpected that the finer grain size show higher accuracy. One could expect that bigger cells would be easier to predict, as the values averages of larger areas. But the differences are very small, and the model is also not very sensitive to cell size. Also noteworthy, is that we see the exact same spatial error pattern in all three experiments.

5.6.3 Neurons in the hidden layer

To test the result of having different number of neurons in the hidden layer, again three experiments are performed, this time to see the effect of altering the number of neurons in the hidden layer. The three following experiments have these settings in common: Grid size: 5 km, no λ_{max}, neighbourhood size: double Moore, decay function: $\frac{1}{dist^2}$.

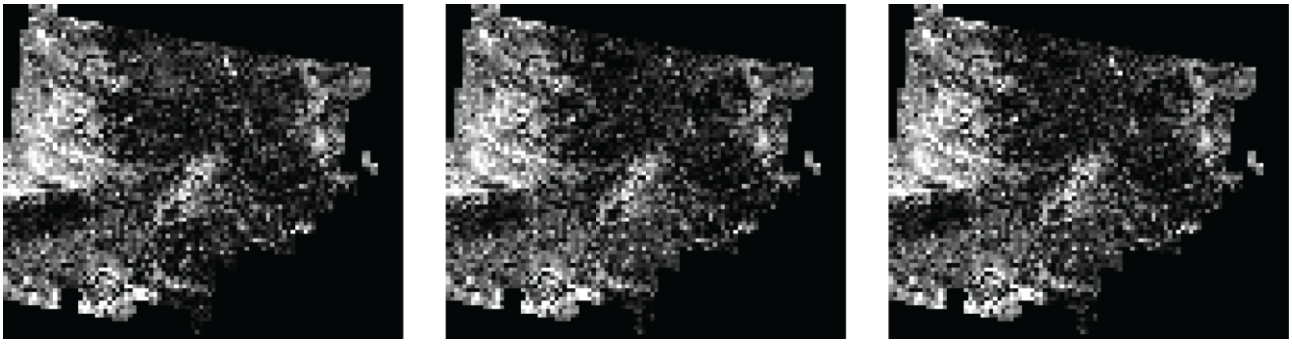


Figure 34 – Error rasters from the experiments regarding neurons in the hidden layer. 4, 5 and 6

From Figure 34, again we see a very familiar result; similar spatial patterns and similar values.

- 4 Neurons: 0.051
- 5 Neurons: 0.049
- 6 Neurons: 0.050

The best result appears when there are 5 neurons in the hidden layer, this could be a sign that, when 6 neurons are applied the model overfits the training data, and the model is too specific to the training data, to be generally applicable, and so it fails in predicting new data. But the difference is very small and making any absolute conclusions from just this one example is not possible.

5.6.4 Neighbourhood

In the following experiments, two related factors are tested, the neighbourhood size and the decay function. This yields combinations which are all conducted with these settings: Grid size: 10 km, no λ_{max}, 5 neurons in the hidden layer.

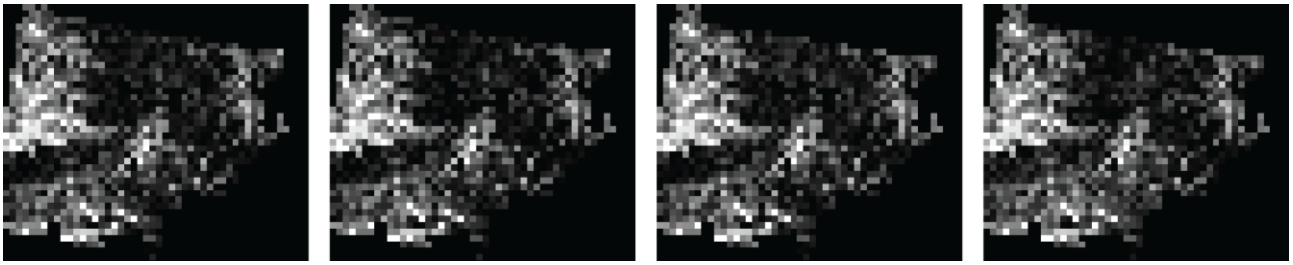


Figure 35 - Error rasters of the neighbourhood experiments. Single Moore neighbourhood and $\frac{1}{dist}$ as decay function, single Moore neighbourhood and $\frac{1}{dist^2}$ as decay function, double Moore neighbourhood$\frac{1}{dist}$ as decay function, double Moore neighbourhood and $\frac{1}{dist^2}$ for decay function.

The four experiments once again, show very similar outputs. The main values of the error in the same order as presented in Figure 35:

- Single Moore neighbourhood, $\frac{1}{dist}$ as distance decay: 0.050
- Single Moore neighbourhood, $\frac{1}{dist^2}$ as distance decay: 0.052
- Double Moore neighbourhood, $\frac{1}{dist}$ as distance decay: 0.052
- Double Moore neighbourhood, $\frac{1}{dist^2}$ as distance decay: 0.053

Very slight differences, but the experiment yielded a better result using the smaller neighbourhood, and $\frac{1}{dist}$ as decay function.

5.6.5 Best options

Apart from the partial results from the previous experiments, a few other setting combinations were tested. The experiment which overall yielded the best result had the following settings:

Grid size: 10 km, no λ_{max}, 6 neurons in the hidden layer, double Moore neighbourhood and $\frac{1}{dist}$ as distance decay. This option yielded an average error of 0.038. As previously stated, not every combination of settings has been tested, and there might be a better setup.

5.6.6 Conclusions

Drawing conclusions from these results is not easy, because every result is nearly identical. The most noteworthy experience is how little a difference it makes, when these parameters are adjusted slightly. When one setting shows a better result in terms of average error it is not a proof that this setting is better for two reasons:

- It could be coincidental. The test sample is small, and the way data is split into a grid could favour one thing or the other based on where the grid is.
- Even if the variable setting is better in one setting for the other variables, it is not necessarily better with different settings for the other variables.

The best guess based on these experiments is these experiments are that the settings should be:

- L-factor with no λ_{max}
- Grid size 10 km
- 6 Neurons in the hidden layer
- Double Moore neighbourhood
- $\frac{1}{dist}$ as distance decay function

5.7 Collective conclusion

In this chapter it has been showed how a model for predicting desertification can be built from cellular automata with an artificial neural network as transition rule. The answer to the first three research questions has been found.

In reference to the first research question regarding the model inputs, the following inputs have been shown to be usable in the model:

- A desertification index calculated by dividing net primary production by average soil moisture. The net primary production can be derived from NDVI images.
- Neighbourhood values consisting of the average of the values in the double Moore neighbourhood with the decay function $\frac{1}{dist}$, calculated from the desertification index.
- An erosion index based on RUSLE where the L-factor can be calculated using ArcGIS' hydrology toolsets without a λ_{max}.
- The average temperature
- The total rainfall
- The amount of nearby river surface water
- The amount of nearby lake surface water
- The amount of nearby urban area

In reference to the second research question regarding cellular automata modelling, it has been concluded that all the above listed data input should be put in to a square grid using a 10 km grid size in an equal area projection, though other grid sizes can work almost equally well. The cellular automaton is based on fuzzy indices rather than classes, and the transition rule is a continuous function derived by applying an artificial neural network.

In reference to the third research question regarding the modelling of the artificial neural network, the following setup has been found successful:

- Feed forward network with one hidden layer
- Training based on the Rprop algorithm, training for a range of single cell values.
- 6 neurons in the hidden layer.

6 Results

In this chapter the model developed in the previous chapters will be evaluated. The best settings found in section 5.6.5 Best options, are used again, this time with more training iterations.

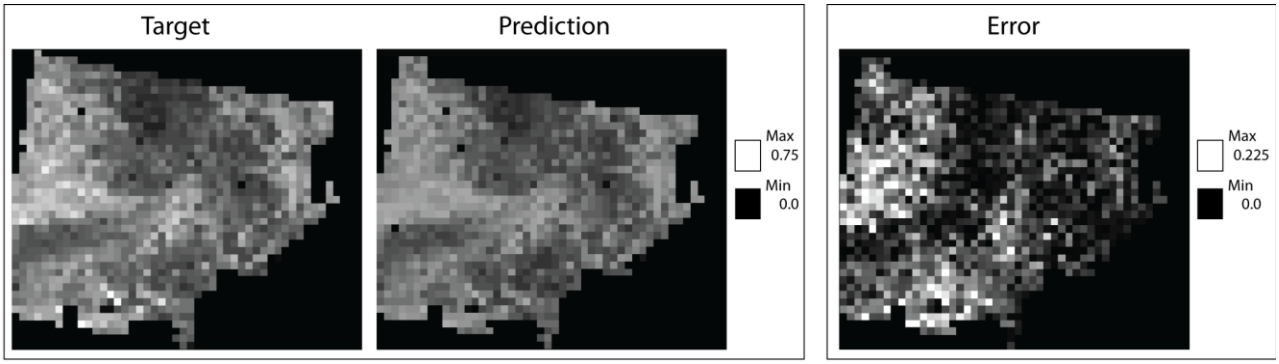


Figure 36 - Evaluation of the model output for 2013. Target raster, predicted raster and the absolute difference.

Figure 36 displays the target raster and the predicted raster, for the desertification index of 2013. It shows many similarities in terms of spatial variation. The model correctly predicts where high DI values and where low DI values are found, based on past data.
The model however, does not reproduce the individual values very precisely. Figure 36 also displays the absolute error. On average the prediction is 0.0372 off. This provides an answer for the fourth and final research question which relates to the accuracy of the model; the best prediction for 2013 has a mean absolute error of 0.0372. The 2013 target raster has a mean desertification index value of 0.378, so the inaccuracy is 9.84 %.

For this forecast, unlike the sensitivity experiments in the previous chapter, the Rprop algorithm was allowed to run for more iterations.

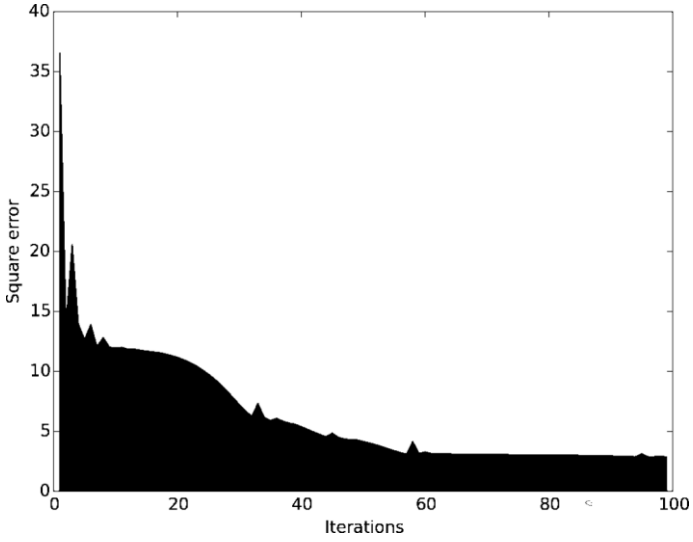


Figure 37 - Training error (sum of squares)

Figure 37 shows the training performance progress of the Rprop algorithm in the artificial neural network. The training error is measured in summed squared errors for 1430 training samples across the 2001-2011 data sets. As Figure 37 displays, the training process converges, and after iteration 60 the error decreases very slowly, and not much improvement is found.

7 Discussion

In chapter 6 Results the best model prediction for the desertification of 2013 was displayed. On one hand, the results demonstrate some of the possibilities of the combination of cellular automata and artificial neural networks, there was a low level of sensitivity to small alterations, demonstrated in section 5.6 Model sensitivity, which showcases some of the ability of the artificial neural network to react to data noise. On the other hand the results also show an inaccuracy of 9.84 %. In the following the strengths and weaknesses of the models will be discussed with the goal of drawing conclusions from the modelling process.

When evaluating a result like the one presented in the previous chapter, the context of the evaluation is important. The model was inspired by the threads posed by desertification, and the lack of knowledge relating to the advancement in terms of speed and extent. If politicians and planners are to act on the basis of modelling results, the accuracy of the model is paramount. 9.84 % average error is a substantial amount, and as such it is not possible for politicians and planners to make decisions on the basis on a model with amount of much error. The result presented in the previous chapter has one CA iteration. It predicts 2013 from 2012 data, in an ideal scenario, it should be able to iterate further and predict 2014 based on the predicted 2013 output, and then again 2015 and so on. In an iterative process like this, the error can potentially increase for each iteration and it may effectively change the conclusion from “the area is in recovery” to “the threat is increasing”.
But that does not mean that the attempt was in vain. The modelling process was constrained by time and data availability. Many possible improvements to the model do exist.

7.1 The desertification index

To improve the model a good place to start would be the desertification index. The desertification index needs to be stabilised to a higher degree, this can be done by working with the processes behind the net primary production. In section 5.4 Modelling with the desertification index, we saw that the net primary production varies from year to year, and the average soil moisture level does not stabilise the primary production sufficiently. The index must be modelled more carefully, by including biochemical knowledge about primary production. The index, as described in section 2.1.2.2 A desertification index, is based on the work of le Houérou (1983) and Prince et al. (1998) both sources used the index for longer

periods, and maybe the index is not fine tuned enough to work with, in as high a temporal resolution as only one year.

The modelling described in the previous chapters is based on data from 2001 to 2013, this may be too short a time frame. The data availability was the limiting factor for this choice. Perhaps the applied NDVI data could be combined with other older datasets to extent the analysis to include more years. Satellites with the AVHRR sensor have previously been applied in desertification studies.

By stabilising the desertification index through a more accurate measure and a longer study period the result will change drastically, and if it helps to predict desertification, it is worth investigating.

7.2 Model design

The model design needs further development. Several elements can be improved either through experimentation or deeper study of how the model components react in different setups.

The architecture of an artificial neural network is paramount for its functionality. For this reason much research has been conducted, trying to uncover the secrets of architecture in artificial neural networks. In section 5.6 Model sensitivity only a few architectures were tested, by having a different number of neurons in the hidden layer, but many more elaborate experiments could be interesting. But also the network could be build with more hidden layers or with recurrence, where connections in the network are allowed to loop. One of the dangers of having too advanced a network is over fitting. The function might incorrectly fit the noise in the training data, and then expect the same noise in its predictions. To fight this, the amount of training data can be increased by widening the training area, or as previously suggested, having data available from more years.
Also experimentation with more than one hidden layers, could possibly enhance the network capabilities.

Another one of the shortcoming, which has already been discussed, is the lack of modelling of human interventions. Perhaps most importantly is the artificial irrigation. An artificially high level of primary production can be observed, in spite of terrible rain deficits. Information about the spatial patterns of irrigation could have a big impact on the model accuracy. But then another problem arises; how could these irrigation patterns be predicted? It seems that the desertification of a highly developed agricultural region is

7.3 Methodological improvements

The following deal with possible improvements which could be included in the modelling and data processing. These were performed differently due to time constraints.

7.3.1 Cross validation

The model settings have been tested, by seeing how well all data up until 2012 predicts 2013 data. This is a fragile validation method, as 2013 may not be a very good validation example.

If more time was available the validation and experimentation process would have used every time step as validation by excluding it from the training data pool, and trying to predict it. This is called cross validation.

The model is meant to make predictions more than one time step into the future, so having that in mind, the validation process should include iterative model predictions to see how the model behaves when the model is less controlled by the known data.

7.3.2 Cloud removal

Perhaps the most important data set in the modelling is the NDVI data, from which the desertification index originates. The NDVI was to some degree afflicted by cloud contamination. Cloud removal is a large field of research in remote sensing, and several methods exist for removing clouds. Methods exists both for cloud removal based on data cloning (Lin, et al. 2013), and based on time series analysis. Some combination of the two fields would be appropriate in this case, as the process should both take account for spatial relationships and for the trends in the NDVI data. Development of a methodology specifically for this purpose could be a future research topic.

7.3.3 Resample techniques

In section 4.5.1 Gridding it was noted, that none of ArcGIS' available resampling techniques were good for resampling from very fine grained raster to a very coarse one. Other resampling techniques could be included from other software. Or perhaps a new technique or techniques could be developed from scratch; it seems that gridding for cellular automata, is unnecessarily difficult using available tools.

8 Conclusion

The aim of this thesis has been to investigate how to combine cellular automata modelling with an artificial neural network to model the geographic process of desertification.

It is concluded that a combination of cellular automata and artificial neural networks as transition rule can, at least to some extent, work as a basis for modelling desertification.

It is important to have a good indicator for desertification. The index developed in the model presented here is based on the RUE, net primary production and available water, as it was suggested by le Houérou (1983) and Prince et al. (1998).

Also included in the model as inputs, are erosion based on RUSLE, temperature, rain, nearby surface water and nearby urban area.

To model the spatiotemporal changes in desertification with cellular automata, all factors were presented as fuzzy variables from zero to one. This means that the transition rule should be a continuous function, and the degree of desertification is presented as a number between zero and one. To model the desertification using a cellular automaton, the data should be presented in a square grid using an equal area projection. 5 km, 10 km and 20 km has been tested and they all yield similar results. The length of the time step has only been tested as one year. Other time steps may be more suiting, as using longer time steps will even out some of the noise of the data. Different neighbourhood settings have been tested, all with not too different results, but the best one found was the double Moore neighbourhood with $\frac{1}{dist}$ as decay function.

An artificial neural network can be designed to find the transition rule function. This has been successfully done with the Rprop learning algorithm. The network was designed as a feed forward network with one hidden layer. The hidden layer was tested with 4, 5 and 6 neurons. These test did not result in a clear cut conclusion, as in one example 5 was the better choice and 6 in another. The conclusion is, that the number of neurons in the hidden layer should be fine tuned as the last step in the modelling process.

Desertification proved not to be an easy modelling subject, at the selected temporal and spatial scale. When subtracting 2013 from the training data pool, and predicting the 2013 data, the best prediction was off by 0.037 on average which is an accuracy of 9.84 %.

To improve the modelling accuracy it is necessary to investigate and address the elements that hamper the modelling. Continued work to improve a desertification index, and to investigate the factors that impact the process would be a good place to start. Also considering ways to model the process with lower temporal resolution, on a longer time scale is believed to potentially improve the results.

Desertification will have more and more focus in Europe in the future and as an increasing amount of satellite data among other data accumulates and is improved in quality, the possibilities for better predictions will also improve.

9 Perspectives

It is clear from the discussion and conclusion that the modelling of desertification is in no way "solved", and that science now can move on to other things. The work presented in the previous chapters can lead to further research on the topics, and the model can be improved. In the following chapter a few topics on further work and development possibilities will be discussed.

9.1 Further research

Two things noted as possible explanations for the shortcomings of the designed models are the data not included, and chosen time scale.

Desertification happens for two reasons, firstly 'naturally' due to climatic and geomorphological reasons, secondly due to spatial and water mismanagement. The second portion has not been included in this model, and that is a serious shortcoming. Land use change can be modelled, and hence its impacts on desertification can also be included in this kind of model. A cellular automaton could have two output values for each cell, which are calculated simultaneously. In this manner land use and desertification could be modelled side by side in an integrated model.

9.2 Other applications

The cellular automaton with artificial neural network as transition rule, has been tested for desertification modelling. But it has many other possible applications in spatial modelling and simulation. The artificial neural network also solves classification problems, so a cellular automaton with a non continuous output is also possible.

References

Bishop (2006), Christopher M. *Pattern Recognition and Machine Learning.* Springer.

Böhm, U., et al.(2006). "ETHZ-CLM climate model output." CLM.

Ding, Huo-ping, Jian-ping Chen, and Gong-wen Wang. (2009). "A Model for Desertification Evolution Employing GIS with Cellular Automata." *International Conference on Computer Modeling and Simulation*: 324 - 328.

EEA (2006). "CLC2006 technical guidelines.".

EEA (2008). "DISMED Sensitivity to desertification index (SDI)." EEA.

EEA (2012). "EEA Catchments and Rivers Network System.".

EEA (2013). *EU-DEM Metadata.* http://www.eea.europa.eu/data-and-maps/data/eu-dem#tab-metadata (accessed May 21, 2014).

ESRI (2014). *ArcGIS Help 10.2, 10.2.1, and 10.2.2.* 5 June 2014. http://resources.arcgis.com/en/help/main/10.2/index.html#//00170000009t000000 (accessed June 5, 2014).

FAO (1993). *Sustainable development of drylands and combating desertification.* FAO position paper, Rome: Food and Agriculture Organization of the United Nations.

Gardner, Martin (1970). "The fantastic combinations of John Conway's new solitaire game "life"." *Scientific American*: 120-123.

Gómez, Jorge García, Francisco López-Bermúdez, and Juan Manuel Quiñonero Rubio (2011). "Land-use Changes, Desertification, and Climate Change Impacts in South-eastern Spain." *Coping with Global Environmental Change, Disasters and Security*: 935-945.

Houérou, Henri N (1983). "Rain use efficiency: a unifying concept in arid-land ecology." *Journal of arid Environments*: 213-247.

Igel, Christian, and Michael Huesken (2000). "Improving the Rprop Learning Algorithm." *Proceedings of the Second International Symposium on Neural Computation*: 115-121.

Intergovernmental Panel on Climate Change (2013). *Fifth Assessment Report of the Intergovernmental Panel on Climate Change.* Working Group I Contribution, Cambridge University Press.

IPCC WG3 (2000). "Emissions Scenarios." Summary for Policy Makers.

Jenkerson, Calli, Thomas Maiersperger, and Gail Schmidt (2010). *eMODIS: A user-friendly data source: U.S. Geological Survey Open-File Report.* USGS.

Li, Xia, and Gar-On Yeh (2001). “Calibration of cellular automata by using neural networks for the simulation of complex urban systems.” *Environment and Planning*: 1445 - 1462.

Lin, Chao-Hung, Po-Hung Tsai, Kang-Hua Lai, and Jyun-Yuan Chen (2013). “Cloud Removal from Multitemporal Satellite Images Using Information Cloning.” *Transactions on Geoscience and Remote Sensing*: 232-241.

Linden, P., and J. F. B. Mitchell (2009). *ENSEMBLES: Climate Change and its Impacts: Summary of research and results from the ENSEMBLES project.* Exeter: Met Office Hadley Centre.

Liu, Yan (2009). *Modelling Urban Development with Geographic Information systems and Cellular Automata.* Taylor & Francis.

Maria, Kouli, Pantelis Soupios, and Filippos Vallianatos (2009). “Soil erosion prediction using the Revised Universal Soil Equation (RUSLE) in a GIS framework, Chania, Northwestern Crete, Greece.” *Environmental Geology*: 483-497.

McKee, Thomas B., Nolan J. Doesken, and John Kleist (1993). “The Relationship of Drought Frequency and Duration to Time Scales.” *Eighth Conference on Applied Climatology*: 179-184.

Middleton, Nick, and David Thomas (1997). *World atlas of Desertification.* UNEP.

Ministerio de Medio Ambiente (2008). “Programa de Acción Nacional Contra la Desertificación.” Action programme.

NASA (2014). “Vegetation Index [NDVI].” *NASA Earth Observations.* Februar 2014. http://neo.sci.gsfc.nasa.gov/view.php?datasetId=MOD13A2_M_NDVI (accessed May 29, 2014).

Nearing, M. A. (1997). “A Single, Continous Function for Slope Steepness Influence on Soil Loss.” *Soil Science society of America*: 917-919.

Nicholson, Sharon E. (2011). *Dryland Climatology.* University Press, Cambridge.

Niemeijer, David, Puigdefabregas, Juan, White, Robin, Lal, Rattan, Winslow, Mark, Ziedler, Juliane, Prince, Stephen, Archer, Emma and King, Caroline (2005) “Dryland systems.” In *Millinium Ecosystem Assessment*, 623 - 662.

Oñate, Juan J., and Begoña Peco (2005). “Policy impact on desertification: stakeholders’ perceptions in southeast Spain.” *Land Use Policy*: 103-114.

Palmer, Wayne C. (1965). *Meteorological Drought.* Research Paper, U. S. Weather Bureau.

Panagos, P., K. Meusburger, C. Alewell, and L. Montanarella (2011). “Soil erodibility estimation using LUCAS point survey data of Europe.” *Environmental Modelling & Software*: 143-145.

Panagos, P., K. Meusburger, C. Alewell, and L. Montanarella (2014). "Soil Erodibility Estimation Using LUCAS Point Survey Data of Europe." *Environmental Modelling & Software*: 143-145.

Prince, S. D., E. Brown de Colstoun, and L. L. Kravitz (1998). "Evidence from rain-use efficiencies does not indicate extensive sahelian desertification." *Global Change Biology*: 359-374.

Renard, K. G., G. R. Foster, G. A. Weesies, D. K. McCool, and D. C. Yoder (1997). *Predicting soil erosion by water: a guide to conservation planning with the Revised Universal Soil Loss Equation.* Handbook, USDA.

Riedmiller, Martin, and Heinrich Braun (1993). "A Direct Adaptive Method for Faster Backpropagation Learning: The RPROP Algorithm." *Procedings of the IEEE international conference on Neural Networks*: 586-591.

Running, Steven, and Maosheng Zhao (2011). "Note to users on use of ODIS GPP/NPP (MOD17) datasets." Numerical Terradynamic Simulation Group.

Tobler, Waldo (1970). "A Computer Movie Simulating Urban Growth in the Detroit Region." *Economic Geography*: 234-240.

Tucker, C. J., C. Vanpraet, E. Boerwinkel, and A. Gaston (1983). "Satellite remote-sensing of total dry-matter production in the Senegalese Sahel." *Remote Sensing of Environment*: 461-474.

UNCCD (2012). *Desertification: A Visual Synthesis*.

United Nations (1994). *Report of the Intergovernmental Negotiating Committee.* Paris: United Nations General Assembly.

van der Knijff, J. M., R. J. A. Jones, and L. Montanarella (1999). *Soil Assessment Risk in Italy.* Joint Research Centre - European Commission.

van der Knijff, J. M., R. J. A. Jones, and L. Montanarella (2000). *Soil Risk Assessment in Europe.* Joint Research Centre - European Commission.

Wolfram, Stephen A (2002). *A New Kind of Science.* Wolfram Media.

Wolfram, Stephen A (1984). "Cellular Automata as a Model of Complexity." *Nature*: 419-424.

Yu, Bofu (2008). "Erosion and Precipitation." In *Encyclopedia of Water Science*, by Stanley W. Trimble, 258-261. Francis & Taylor.