

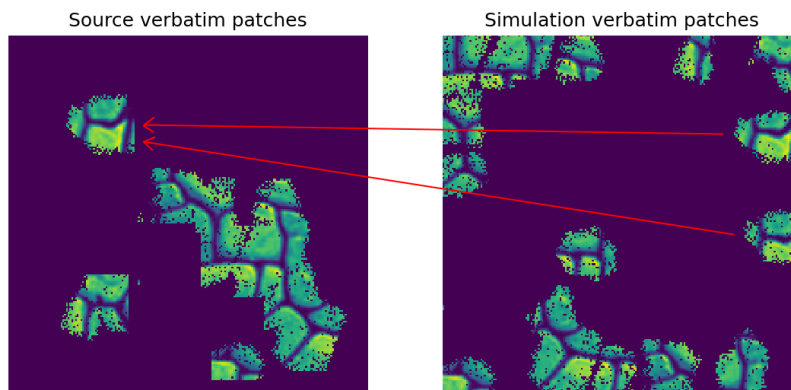
UTRECHT UNIVERSITY

MSc. APPLIED DATA SCIENCE THESIS

---

# The analysis of verbatim copy in pixel-based Multiple-Point statistics

---



**Author:** Matthias Meester  
**First supervisor:** Mathieu Gravey  
**Second supervisor:** Derek Karssenberg

July 1, 2022



**Utrecht  
University**

# Contents

<b>1</b>	<b>Abstract</b>	<b>2</b>
<b>2</b>	<b>Introduction</b>	<b>3</b>
<b>3</b>	<b>Data</b>	<b>4</b>
3.1	Simulation data . . . . .	4
3.2	Dummy data . . . . .	5
3.2.1	Full verbatim map . . . . .	5
3.2.2	Full random map . . . . .	5
3.2.3	Checkerboard verbatim map . . . . .	6
3.2.4	Random patch verbatim map . . . . .	6
3.2.5	Long range verbatim map . . . . .	6
<b>4</b>	<b>Methods</b>	<b>8</b>
4.1	Kernel functions . . . . .	8
4.1.1	Filter-based verbatim . . . . .	8
4.1.2	Filter-based neighbour verbatim . . . . .	9
4.2	Statistics . . . . .	10
4.2.1	Mean heat value . . . . .	10
4.2.2	Threshold neighbour analysis . . . . .	10
4.2.3	Spatial dependency analysis . . . . .	11
4.2.4	Patch size analysis . . . . .	11
4.3	Method application . . . . .	11
4.3.1	Filter-based verbatim / mean heat value . . . . .	12
4.3.2	Filter-based verbatim / threshold neighbour analysis . . . . .	12
4.3.3	Patch size analysis . . . . .	13
4.4	Method evaluation . . . . .	13
4.4.1	Evaluation on dummy data . . . . .	13
4.4.2	Evaluation on simulation data . . . . .	14
<b>5</b>	<b>Results</b>	<b>15</b>
5.1	Dummy data results . . . . .	15
5.2	Simulation data results . . . . .	16
5.2.1	Visual validation . . . . .	16
5.2.2	Relation of QuickSampling parameters and statistics . . . . .	16
5.2.3	Spatial dependency analysis . . . . .	19
<b>6</b>	<b>Discussion</b>	<b>21</b>
<b>7</b>	<b>Conclusion</b>	<b>22</b>
<b>A</b>	<b>Full dummy data results</b>	<b>23</b>
<b>B</b>	<b>Additional simulation data results</b>	<b>24</b>
	<b>Bibliography</b>	<b>27</b>

# 1. Abstract

Multiple-Point statistics are a class of geostatistical simulation techniques used to reproduce complex spatial structures present in a training image and can be used for conditional and unconditional simulations. A weakness in multiple-point statistics is the presence of verbatim copy in realizations. Verbatim copy occurs when a continuous part of the input is 'copied' to the simulation. Verbatim copy can result in a large part of the simulation containing identical patches of the input image, which is undesirable. This thesis has focused on the research of developing new metrics that can summarize the presence of verbatim copy in pixel-based simulations. New methods are presented which can summarize the proportion of verbatim copy. Artificial verbatim copy can be detected with these techniques, and promising results are shown on real simulation data. Both long- and short-range verbatim copy can be detected using the new methods.

## 2. Introduction

Multiple-Point statistics (MPS) are a class of geostatistical simulation techniques used to reproduce complex spatial structures present in a training image (TI) and can be used for conditional and unconditional simulations. One of the main advantages over classic interpolation techniques is that MPS can create realistic spatial patterns. The existing techniques can be divided into two main groups; pixel-based methods, such as snesim (Strebelle, 2002), Impala (Straubhaar et al., 2011), direct sampling (Mariethoz et al., 2010) and QuickSampling (Gravey & Mariethoz, 2020) and patch-based methods such as SIMPAT (Arpat & Caers, 2005), CCSIM (Tahmasebi et al., 2012) and CIQ (Mahmud et al., 2014). Pixel-based methods generate the simulation pixel by pixel, whereas patch-based methods can transfer multiple points from the training image to the simulation at the same time.

A weakness in multiple-point statistics is the presence of verbatim copy in realizations (Mariethoz & Caers, 2014). Verbatim copy occurs when a continuous part of the input is 'copied' to the simulation. This can result in a large part of the simulation containing identical patches of the input image, which is undesirable.

This thesis focuses on developing metrics that can summarize the presence of verbatim copy in simulations. The simulations that will be used for developing this statistic have been generated using QuickSampling, which is a pixel-based simulation method. This method works by transferring one pixel from the source image to the simulation by finding the  $k$  best matches and picking one of them. QuickSampling has the advantage that it provides an index map. This map shows for each pixel in the simulation the index of the source pixel in the training image where it is sampled from.

Several methods have been previously proposed to identify verbatim copy. For example, (Li et al., 2016) have proposed an MPS method with an index that allows for identifying verbatim copy, the merging index. The problem with this method is that it only works with the specific used patch-based algorithm.

Methods such as the consistency and innovation factor proposed by (Abdollahifard et al., 2019), have the disadvantage that they do not work well on pixel-based simulations such as QuickSampling. Pixel-based simulations can give short-range inconsistencies, which the mentioned methods can see as low consistency and high innovation, meaning low verbatim copy. The corresponding area could actually be one where much verbatim copy is present.

The lack of knowledge on how to summarize the proportion of present verbatim copy in pixel-based simulation techniques such as QuickSampling creates the need for additional research to find methods to do this. This thesis will look to find the answer to the following research question: "How can the proportion of present verbatim copy be summarized in pixel-based simulation methods?".

The thesis documents the tried methods to find verbatim copy and presents the ones which can yield useful information on the proportion of present verbatim copy. Chapter 3 elaborates on the data that will be used. Chapter 4 explains which methods will be used to analyze verbatim copy. Chapter 5 lays out the results and comparison of the used methods. Discussion points and recommendations for future research are given in chapter 6. The conclusion can be found in chapter 7.



## 3. Data

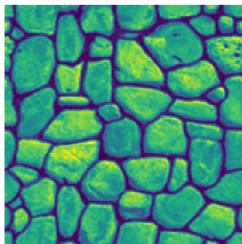
The different metrics that will be created to analyse verbatim copy will be evaluated using both real Quick-Sampling (QS) sample input and output data, and created dummy data. The dummy data is not generated by the QuickSampling algorithm. It is instead created, to get control over the input data to be able to evaluate the different metrics more effectively. The dummy data consists of different forms of index maps. The index map is the one that will be used in the different metrics because it can yield information about whether and where verbatim copy is present (Lefebvre & Hoppe, 2005). Section 3.1 elaborates on the simulation data. Section 3.2 elaborates on the different forms of dummy data.

### 3.1 Simulation data

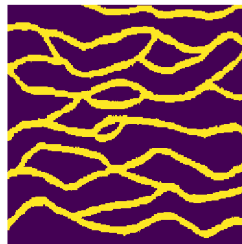
An open database<sup>1</sup> with sample input and output data of the QuickSampling algorithm is available to use for creating the different metrics. The sample input data consists of a training image and the  $k$ ,  $n$  parameters as described in the QuickSampling algorithm. The sample output data consists of the simulation and the simulation index map.

The database used has simulations for two different types of training images, 'stone' (Mariethoz & Caers, 2014) with size  $200 \times 200$ , and 'strebelle' (Strebelle, 2002) with size  $250 \times 250$ . 'stone' has a continuous training image for the done simulations and 'strebelle' has a binary training image for the done simulations, see figure 3.1. For both image types, there are simulations for  $n$  ranging from 1 – 199, and  $k$  drawn from [1, 1.01, 1.02, 1.05, 1.15, 1.2, 1.3, 1.5, 1.7, 2, 2.5, 3, 5, 10], the simulations have the same size as the training image. The simulation width and height are defined by  $w_{sim}$  and  $h_{sim}$  respectively. A sample 'stone' input can be seen in figure 3.2.

'strebelle' simulations have less verbatim copy present on average. There are many locations to choose from for each pixel during the 'strebelle' simulation, as many patterns fit because of the binary nature of the training image. 'stone' simulations have more verbatim copy present on average, as continuous patterns are more complex. As the patterns are more complex, fewer options are available for each pixel during the simulation, and thus more verbatim copy will be present.



(a) 'stone' image



(b) 'strebelle' image

Figure 3.1: Used training images

---

<sup>1</sup><https://surfdrive.surf.nl/files/index.php/s/7pQ9hfU36aU0jZW>

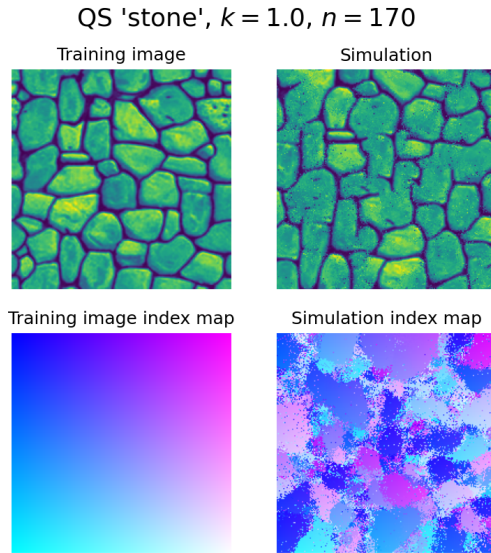


Figure 3.2: Sample input consisting of the 'stone' training image, simulation and the resulting index map represented by a color space.

## 3.2 Dummy data

The dummy index maps consist of different variations and patterns of verbatim patches intertwined with full random patches. This makes it so that it is easy to for example induce 50% of the image to be 'verbatim'. The metrics can then be evaluated based on this. The dummy maps will have a size of  $200 \times 200$  pixels, as this is the same as the 'stone' simulation data set that is used, making the results comparable between the two.

### 3.2.1 Full verbatim map

The full verbatim map is the most basic form of the dummy data. It assumes that the full training image has been copied directly to the simulation. The way this can be simulated is by setting the simulated index map to the training image index map. This means that the simulation index map looks the same as the 'Training image index map' from figure 3.2. In this scenario, the metrics should indicate full verbatim copy; attaining their maximal attainable value.

### 3.2.2 Full random map

The full random map is the map where it is assumed that the simulation is as random as possible. The way this map is created is that for every index in the simulation index map a random index from the training image index map is sampled. In this scenario, the metrics should indicate little to no verbatim copy, as the chance of verbatim occurring is very small as the indices are randomly aligned. It should be noted that inevitably some verbatim can occur by random chance; there can be adjacent pixels in the simulation that are adjacent to each other in the training image index map as well. An example instance of this map can be seen in figure 3.3a.

### 3.2.3 Checkerboard verbatim map

In the checkerboard verbatim map, full verbatim blocks sampled from the full verbatim map are alternated by patches from the full random map in a checkerboard pattern. The sizes of the blocks are determined with the block size parameter  $b$ . The reason for this map is to see how the size of the blocks affects the different statistics; ideally, they would indicate 50% verbatim copy, regardless of the block size. The checkerboard can be used to see how the different metrics react to changes in this block size. This is useful to know, as in the real simulations the patches can also vary in size, and how the metrics react to this should be known. Additionally, this checkerboard is a tool to test whether metrics can detect the spatial structure of verbatim copy. An example instance of this map can be seen in figure 3.3b.

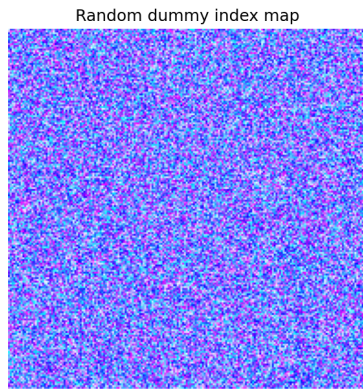
### 3.2.4 Random patch verbatim map

The base for the random patch verbatim is the full random map, but a number of circular patches of the full verbatim map are overlayed. The number of patches is defined by parameter  $c$ , and the patch size is defined by parameter  $s$ . There is also some noise overlayed, set to a proportion  $N_p$  of the patch, to make it harder to detect the patch. The reason for using this map to test the different metrics is the same as for the checkerboard map, to see how they react to different types of verbatim patches. The statistics should ideally indicate the same amount of verbatim copy as the proportion of pixels contained in verbatim patches. An example instance of this map can be seen in figure 3.3c.

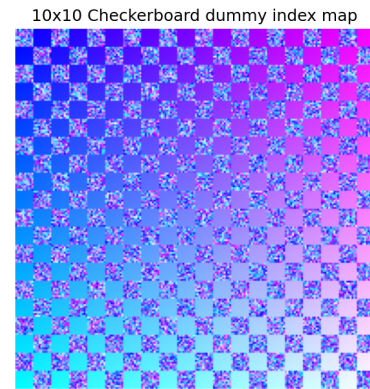
### 3.2.5 Long range verbatim map

The base for the long-range verbatim map is the full random map, but a proportion  $p$  of pixels is assigned to random groups of pixels of 20-100 pixels, which are set to be verbatim to each other (and only to each other). In this way, random pixels are in verbatim position to each other; but they are so far apart and randomly assigned that they are almost impossible to see by the eye.

This map can be used to analyse whether these kinds of verbatim occurrences are also detectable. Ideally, the statistics would indicate the same amount of verbatim copy as the proportion of pixels randomly drawn from the full random map. An example instance of this map can be seen in figure 3.3d.

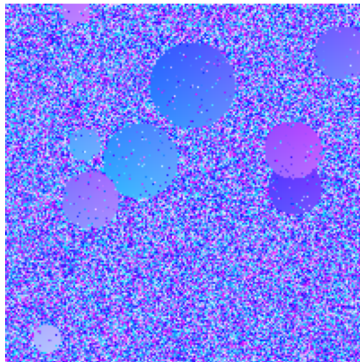


(a) Map consisting of a random index map.



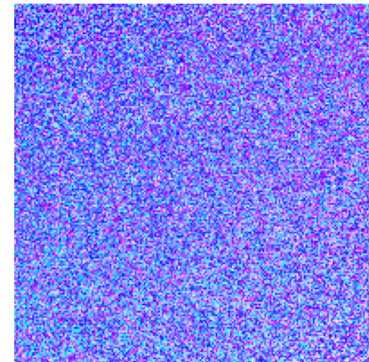
(b) Map consisting of a 10x10 checkerboard.

Patch dummy index map, proportion verbatim = 0.157



(c) Map containing verbatim patches, resulting in 15.7% of the image consisting of pixels included in the verbatim patches.

Long range index map, proportion verbatim = 0.1



(d) Map containing random verbatim paired pixels resulting in 10% of the image consisting of these pixels.

Figure 3.3: Illustration of various types of dummy index maps.

# 4. Methods

The approach I have come up with to determine the presence of verbatim copy consists of two steps: The first step is to apply kernel-based functions with varying parameters to identify different types of verbatim copy to the index map. The second step is to use the output of these kernel-based functions to generate statistics on verbatim copy.

Section 4.1 elaborates on the kernel-based functions. Section 4.2 elaborates on the different statistics that are applied to the output of these kernel-based functions. Section 4.3 explains how the different statistics will be applied. Section 4.4 explains the way the statistics will be evaluated.

## 4.1 Kernel functions

A kernel or filter is a matrix used to get more information from an image. This kernel applies functions to each of the cells included in the kernel, also known as convolution. In this case, the kernel will have a radius of  $r$ , resulting in a kernel with size  $(2r + 1) \times (2r + 1)$ . This kernel is then run over the index map  $I$ . After applying the kernel on each pixel, the output is a so-called verbatim heat map ( $H$ ), where every cell contains a heat value between 0 and 1. This heat value indicates the amount of verbatim copy present at that specific location. Additionally, a number of  $w_{sim} \cdot h_{sim}$  kernels are output, as a kernel has been applied to each pixel in the simulation map. These kernels can be analysed for the spatial dependency of verbatim heat. Section 4.1.1 and section 4.3.2 both elaborate on a different kernel-based function.

### 4.1.1 Filter-based verbatim

Filter-based verbatim will indicate how much a point is surrounded by 'verbatim neighbours'; points that lie in the same respective position as in the source map. The maximum achievable value of each pixel is 1, meaning that verbatim neighbours entirely surround the point.

For every point  $I_{(x,y)}$  in the index map  $I$ , the verbatim heat value of the heat map  $H$  will be determined with a kernel  $K_{(x,y)}$ . This kernel checks every point around the pixel if the pixel index is equal to the expected verbatim neighbour index. It then puts a '1' if they are equal and a '0' if not. This kernel is then summed, so the more pixels around the center pixel are 'verbatim' pixels, the larger the heat value becomes.

The expected verbatim neighbour index is defined by  $E_{(x,y)}$ , which returns the index that would contain verbatim at that position. This function depends on how the source index map of the  $TI$  looks and is trivial to implement. In the case of the dataset mentioned in section 5.2, where the index in the  $TI$  is incrementing with every pixel, first row-wise and then column-wise, and where  $TI$  and  $I$  have the same shape, this function becomes  $E_{(x,y)} = y \cdot w_{sim} + x$ .

The heat value is also weighted by calculating the Hadamard product with a weight matrix  $W$ . Weighting is done so that the farther a verbatim pixel occurs, the less impact it makes. This is done to see whether verbatim estimation improves when closer verbatim pixels are weighted more heavily than farther verbatim pixels. The weights in  $W$  are determined using the inverse distance weighting function  $DI_{(dx,dy)}$ . How much distance impacts this weighting is determined by  $d$ .  $H$  is normalised by dividing by the sum of  $W$ , so the heat values stay between 0 and 1.

The heat map  $H$  will thus be determined by applying function  $H_{(x,y)}$  to each index  $I_{(x,y)}$  as follows:

$$H_{(x,y)} = \frac{\text{sum}(K_{(x,y)} \circ W)}{\text{sum}(W)}$$

With:

$$K_{(x,y)} = \begin{bmatrix} eq(I_{(x-r,y-r)}, E_{(x-r,y-r)}) & \cdots & eq(I_{(x+r,y-r)}, E_{(x+r,y-r)}) \\ \vdots & \ddots & \vdots \\ eq(I_{(x-r,y+r)}, E_{(x-r,y+r)}) & \cdots & eq(I_{(x+r,y+r)}, E_{(x+r,y+r)}) \end{bmatrix}$$

$$eq(i_a, i_b) = \begin{cases} 1 & \text{if } i_a = i_b \\ 0 & \text{otherwise} \end{cases}$$

$$W = \begin{bmatrix} DI_{(-r,-r)} & \cdots & DI_{(r,r)} \\ \vdots & \ddots & \vdots \\ DI_{(r,r)} & \cdots & DI_{(r,r)} \end{bmatrix}$$

$$DI_{(dx,dy)} = \begin{cases} (\sqrt{dx^2 + dy^2})^{-d} & \text{if } 0 < \sqrt{dx^2 + dy^2} \leq r \\ 0 & \text{otherwise} \end{cases}$$

Values indexed outside of the image bounds are not taken into account. This is done by cropping the kernel to be contained within the matrix. The weight matrix  $W$  is cropped in the same way, so it becomes the same shape as the kernel. For example, if the top part of the kernel is out of bounds, only the bottom part of the kernel is taken into account, and thus only the corresponding bottom part of the weight matrix is taken into account. Another way of implementing out-of-bounds indexing could be to pad the index map with zeros, but then the normalisation method:  $\text{sum}(W)$  should be changed accordingly as well, so the heat value stays between 0 and 1.

An example result of filter-based verbatim can be found in figure 4.1. The two values that are to be determined by the user are the kernel radius  $r$  and the inverse distance scalar  $d$ . Filter-based verbatim considers that verbatim copy only occurs if the neighbour is exactly the same respective index as in the source map. This is an assumption and subjective.

#### 4.1.2 Filter-based neighbour verbatim

Filter-based neighbour verbatim works similarly to filter-based verbatim. However, instead of looking at how many neighbours are the same respective neighbours as in the source map, the method considers that if a point is sampled from a point close to the expected neighbour, this point is also a bit like a verbatim copy. This introduces two extra parameters  $r_{cl}$  and  $d_{cl}$ .  $r_{cl}$  determines the distance at which points are still considered as 'verbatim copy' when sampled from a pixel in the neighbourhood.  $d_{cl}$  determines the inverse distance scalar; points farther from the pixel generate less heat.

This changes the  $eq$  function from section 4.1.1 to the  $cl$  function, which checks if a pixel is sampled from a close distance;

$$cl(i_a, i_b) = \begin{cases} 1 & \text{if } i_a = i_b \\ dst_{cl}(i_a, i_b)^{-d_{cl}} & \text{if } dst_{cl}(i_a, i_b) \leq r_{cl} \\ 0 & \text{otherwise} \end{cases}$$

With  $dst_{cl}(i_a, i_b)$  returning the distance between indices  $i_a$  and  $i_b$ . Using this over the normal short-range verbatim is a difference in the assumption of whether a closely sampled verbatim pixel is also considered verbatim copy. An example result of filter-based neighbour verbatim can be found in figure 4.1.



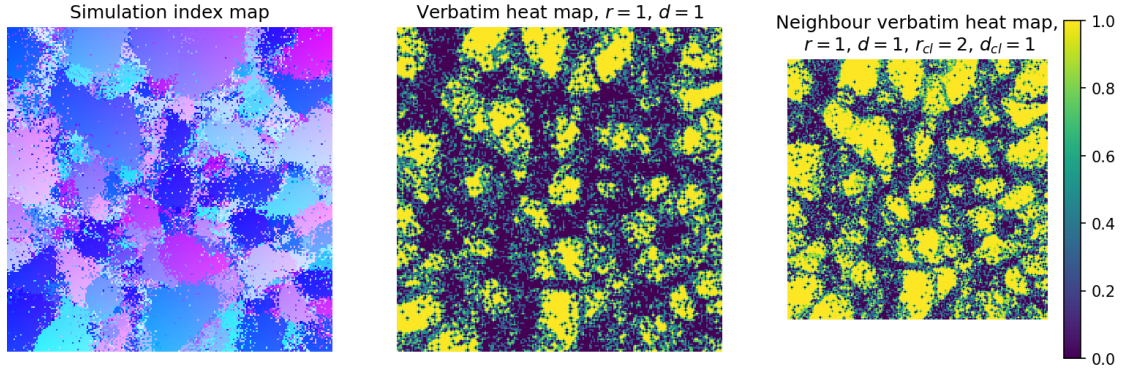


Figure 4.1: Example of verbatim filters applied to a simulation index map, resulting in heat maps.

## 4.2 Statistics

The statistics are built such that they can be applied to the heat maps generated by the functions as defined in section 4.1. Different combinations of statistics and kernel functions and varying parameters can give different perspectives of how verbatim copy is present. More on this is explained in section 4.3.

### 4.2.1 Mean heat value

The mean heat value (*MHV*) simply takes the mean of all the heat values in the generated heat map, thus giving a value between 0 and 1. This value then determines the average heat value of the map. This statistic can be useful to get a quick insight into how on average each pixel is determined by verbatim copy of the neighbouring pixels.

### 4.2.2 Threshold neighbour analysis

Threshold neighbour analysis (*TNA*) looks at a non-weighted heat map;  $d = 0$ . This means that every pixel in the resulting heat map indicates the threshold  $T$  of neighbours in the kernel that are verbatim to this respective pixel. We can then put different thresholds on this threshold to determine pixels which for example have 1% of their neighbours in the kernel being verbatim, or to determine pixels that are fully verbatim; 100% of the neighbours are verbatim. The output of this analysis is a binary map with pixels with a larger number of verbatim neighbours than the threshold marked as '1' and pixels with a smaller number of verbatim neighbours than the threshold marked as '0'. Consequently, the total proportion of 1s can be calculated, yielding a single proportion on how much verbatim is present.

$T$  can be set in two ways; with a fixed value, or a value can be calculated by doing dummy simulations. To do the latter, 10 non-weighted heat maps are generated on the full random noise map as mentioned in section 3.2.2. The parameters for generating these non-weighted maps are the same as used in the threshold neighbour analysis. After generating the heat maps, the maximal 1% of heat values is selected. This is done to get the maximal occurring 'heat', denoted by  $m_{nh}$ . This  $m_{nh}$  is then multiplied by some scalar  $S$  and can be used as the threshold, such that  $T = S \cdot m_{nh}$ . Note that the  $m_{nh}$  can possibly be calculated without the need for simulations, using statistics, but doing this is out of the scope of this thesis. The  $S \cdot m_{nh}$  information can be used to say something about other pixels which pass this threshold during the threshold neighbour analysis; apparently, they contain  $S$  times more heat than would occur maximally naturally.  $S$  should be calibrated, depending on how much verbatim copy is tolerated to occur in comparison to random noise. Either  $S$  or  $T$  should be calibrated by the user, depending on their preference and simulation parameters.

Note that  $S$  depends on the input and simulation sizes, as this changes the amount of verbatim copy that occurs by random chance. This is why fixing  $S$  also leads to problems. In this thesis,  $S$  will be fixed, as the simulation size and input size are fixed, but future research could look into how to determine what a good threshold is depending on simulation parameters.

### 4.2.3 Spatial dependency analysis

Spatial dependency analysis (*SDA*) is done to see how verbatim is distributed spatially. This can already be seen in the heat maps, as can be seen in figure 4.1, but it can also be analysed how verbatim is distributed distance-wise. This is done by analysing the  $w_{sim} \cdot h_{sim}$  kernels output by the algorithm. These can be analysed to see how local (within-kernel) verbatim patterns occur and to see how distance influences the presence of verbatim copy.

Kernel verbatim patterns are analysed by summing all kernels and normalising them. This summed kernel can be plotted and is a quick way to see the mean local verbatim pattern. Advanced analysis can also be done to see which kernel patterns contribute to the overall mean local verbatim pattern, but this was out of the scope of this thesis.

To see how distance influences the presence of verbatim copy, for each possible verbatim distance within range  $r$  a value is set to 0. Then for each pixel in each kernel, the distance to the kernel center pixel is taken, and the value for this distance is then incremented with the verbatim value. The value is then divided by the number of pixels that have that same distance. A bar plot can then be made to show the relation of distance to verbatim presence.

### 4.2.4 Patch size analysis

Patch size analysis aims to find the sizes of patches that are present in the verbatim simulation. This has the advantage that one can, for example, know the mean patch size or the largest patch that is present in the image. If this patch is too large, the simulation could be considered to contain an unwanted form of verbatim. This could be unknown when only considering total mean heat value or threshold neighbour analysis proportion if there is little verbatim globally.

The map from threshold neighbour analysis will be used to determine where patches are. This map only contains 1s and 0s, which is ideal to combine with the method `connectedComponentswithStats` from OpenCV. This method works on a grayscale image and finds the 8-way connected components using Spaghetti Labeling (Bolelli et al., 2019), connected verbatim patches in this case. It then yields information on them, such as location and size. This information can be utilised to find, e.g. the maximum/mean patch size and the location of large undesired verbatim patches in the simulation.

## 4.3 Method application

Different combinations of kernel functions, parameters and statistics will be used to find the several types of verbatim that can occur. In this way, different combinations of methods can potentially show the different aspects of verbatim occurring. A focus on two main types of verbatim is taken into account; short-range verbatim and long-range verbatim. Both can be useful to know, and there should be a way to determine if any or both of them are happening. The methods focus on filter-based verbatim, as mentioned in section 4.1.1, as filter-based neighbour verbatim is very similar and will likely give comparable results. Assigning neighbouring source indices as verbatim is a choice. I have chosen to not further analyse this scenario to increase the research time invested into true verbatim copy. The following combinations of the filter-based verbatim kernel function and the statistics will be used to find verbatim:



### 4.3.1 Filter-based verbatim / mean heat value

Combining filter-based verbatim with the mean heat value will aim to find short-ranged verbatim copies. The reason for this is that combining the two will aim to indicate the 'heat' of each pixel, indicating how much influence the direct neighbours are in determining this pixel. This will not work well with long-range verbatim copies, as it is statistically unlikely that a large filter is fully composed of verbatim copies. This means that the heat value quickly diminishes as the filter radius increases, see figure 4.2. This combination will thus be tested on smaller filter radii. The parameters that will be tried are:  $(d = 1, r = 1)$ ,  $(d = 2, r = 1)$   $(d = 1, r = 2)$  and  $(d = 2, r = 2)$  for filter-based verbatim, the mean heat value does not have parameters.

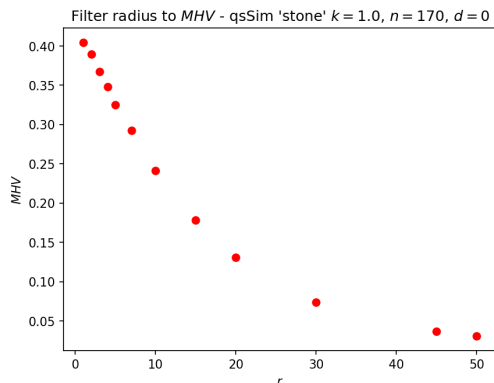


Figure 4.2: Relation of the filter radius to the  $MHV$ , illustrating that  $MHV$  is more useful for analysing short-range verbatim, as the  $MHV$  decreases when  $r$  increases.

### 4.3.2 Filter-based verbatim / threshold neighbour analysis

Combining filter-based verbatim with threshold neighbour analysis will aim to find both short and long-ranged verbatim copies. The reason for combining these two is the way verbatim copy happens. Verbatim copy is normally statistically unlikely to happen if a simulation is truly random; there is very little chance for a pixel to have a neighbouring pixel that is respectively exactly in the same position as in the source index map. For example; for a training image sized  $200 \times 200$  each pixel with 8 neighbours in the simulation only has a chance of containing a verbatim neighbour by random chance of  $P = 1 - \left(\frac{39999}{40000}\right)^8 = 0.0002$ . We can take this information to our advantage; if a pixel contains any neighbours that are 'verbatim' in respective to it, this is remarkable. Thus we can threshold the filter-based verbatim map and say that if a pixel contains proportionally more than  $T$  verbatim neighbours, the pixel itself is verbatim, as it has unlikely neighbours. This  $T$  can be really small because of the statistical unlikeliness of this happening. This holds for both short-ranged and long-ranged neighbouring pixels; they should not be likely to occur in the same place as they would in the source index map, respectively.

The following parameters, which can possibly indicate short range verbatim copy will be tested;  $(d = 0, r = 1, T = 0.001)$ ,  $(d = 0, r = 2, T = 0.001)$ ,  $(d = 0, r = 3, T = 0.001)$ .

The following parameters, which can possibly indicate long range verbatim copy will be tested;  $(d = 0, r = r_{max})$  for filter-based verbatim and  $T = 0.1, T = 0.01, T = 0.001, T = 10m_{nh}, T = 100m_{nh}$  for the proportional neighbour analysis, where  $r_{max}$  is the maximal distance that a pixel can be from another pixel ( $r_{max} = \sqrt{w_{sim}^2 + h_{sim}^2}$ ).

Note that for the statistics,  $T$  and  $S$  are fixed, but this is due to the training image and simulation being set to a fixed size. This parameter is dependent on the input and simulation sizes, as mentioned in 4.2.2. The most important variable to look at is the  $r$ , which determines if we are looking at short- or long-range verbatim copy.

### 4.3.3 Patch size analysis

Patch size analysis will be done with the focus on short-range as if long-range verbatim occurs, it can not be detected with the `connectedComponentswithStats` method, as the pixels are not directly adjacent to each other, but there is a (large) gap between them. The threshold map that will be used for this is the best map found when doing *TNA* (as described in section 4.3.2).

## 4.4 Method evaluation

The methods will be evaluated based on the two different data sources mentioned in chapter 3. This section explains how the different data sources will be evaluated. The methods will first be tested on dummy data to see if they work correctly on controlled data sets, see section 4.4.1. They will then be tested to see if they can be used to analyse simulations, see section 4.4.2.

### 4.4.1 Evaluation on dummy data

The proposed statistics as mentioned in section 4.3.1 and section 4.3.2 will be tested on the dummy data as described in section 3.2. Each dummy map has a induced verbatim value  $v$ , e.g: the checkerboard pattern has  $v = 0.5$ ; 50% of the map consists of verbatim copy. Each statistic will be run on different dummy simulations. They will be evaluated on how close they come with their predicted verbatim value  $\hat{v}$  to the induced verbatim value  $v$ . Note that the induced verbatim value  $v$  could be slightly off; for example, in the checkerboard pattern pixels in the noisy blocks could be in verbatim position to each other; increasing the  $v$  to be a little bit above 0.5. We do not know the exact truth about whether this is the case; this is what we are trying to solve in this thesis.

For each category of dummy map mentioned (full random, full verbatim, checkerboard, patches, long-range), simulations will be done. For the maps with a parameter (checkerboard, patches and long-range), simulations will be done for three different levels of the parameter:

- **Checkerboard:**  $b = 1, b = 3, b = 5$ ; This is to check whether bigger scale checkerboards impact the statistics.
- **Patches:**  $c = 5, c = 10, c = 20$ ; with every patch having a random size between  $s = 5$  and  $s = 25$  and with noise proportion  $N_p = 0.05$ ; This is to see how varying number/size of verbatim patches can influence the statistics.
- **Long range:**  $p = 0.05, p = 0.10, p = 0.15$ ; This is to test how varying amounts of long range verbatim can influence the statistics.

This means that in total, there are  $2 + 3 + 3 + 3 = 11$  different dummy simulation maps that the statistics will be evaluated on. For each type of simulation map, 10 instances will be generated to get different variations of dummy simulations (for example, maps with different patch variations). This means that each statistic will be run on 110 instances of dummy maps.

For each instance category, the predicted and induced verbatim values are used to find an average error for how well the statistic scores in this category of simulation. This error is found using the Mean Relative Error (MRE).

The MRE should be minimized, with the best value attainable being 0, meaning that the error deviates on average by a fraction of 0. The statistics that score near 0 in a simulation category are very well at estimating verbatim copy in that category. For the full random map where the verbatim values are set to an induced value of  $v = 0$ , not the MRE is used, but the mean absolute error (MAE), as the MRE is not defined when the denominator is equal to 0. The MAE is equal to the mean of the predicted values in this case, see below:

$$MAE(v = 0, \hat{v} > 0) = \frac{1}{n} \sum_{i=1}^n |\hat{v}_i - v_i| = \frac{1}{n} \sum_{i=1}^n |\hat{v}_i - 0| = \bar{\hat{v}}$$

$$MRE = \frac{1}{n} \sum_{i=1}^n \frac{|\hat{v}_i - v_i|}{v_i}$$

#### 4.4.2 Evaluation on simulation data

The QuickSampling simulation data mentioned in section 3.1 will be hard to evaluate on whether the verbatim copy methods are valid, as there is no objective truth known on how and where verbatim copy is present. To evaluate the simulation data, we should thus confirm in other ways whether the metrics yield useful information on the verbatim copy presence in the simulation. The statistics used to analyse real simulation data will depend on which statistics perform well on the dummy data mentioned in the previous section.

Firstly, the statistics will be visually verified; several simulations with varying  $n$  and  $k$  will be done, and the statistics will be run on them. We can then see whether the statistics match up with what is seen in the pictures; if the statistics indicate a lot of verbatim, we can check if this is also visually reflected in the index map/simulation.

Secondly, the relation of parameters  $k$  and  $n$  of the QuickSampling algorithm and the statistics can be evaluated. The relation of  $k$  with the statistics should be inverse. The higher the  $k$ , the lower the statistics should indicate verbatim copy, as with a higher  $k$ , each point is less likely to be sampled from the point it aligns mostly with. The relation of  $n$  with the statistics should be positive; the higher the number of points that we use to align, the higher the chance that we find verbatim copies.

Thirdly, the spatial dependence of verbatim copy will be evaluated using the methods mentioned in section 4.2.3. This method is applied only to the simulation data, as in the dummy data, the spatial dependency is already known; it is induced with the construction of the index map. For example, the checkerboard pattern will have a checkerboard-like local spatial dependence.

## 5. Results

As mentioned in section 4.4 the results are generated and analyzed in two ways: using dummy data and using simulations. Section 5.1 elaborates on the results of the methods applied to dummy data. Section 5.2 elaborates on the results of the methods applied to simulation data.

### 5.1 Dummy data results

In table 5.1 the MAE/MRE scores for each statistic on each dummy map category is shown. In table A the full results for each subcategory can be seen. The lower the error value is in each cell, the better the performing statistic is.

For categories 'Full Random' and 'Full Verbatim', every statistic performs very well. There is no clear best one. For 'Checkerboard' almost all the *TNA*-based statistics work well, except for *TNA* with  $r = 1$ , the range is too small to detect the changes in the checkerboard. 'Patches' can both be shown with *MHV* and *TNA*, with the best performing *TNA* statistics having  $T = 0.001/T = 10m_{nh}$  and the best *MHV* statistic having  $r = 1$ . For 'Long Range', the best performing statistics are the ones with  $r = r_{max}$ . This makes sense, as they are the only ones to be able to detect the long-range verbatim occurring, as it is within their filter radius.

To select the best statistics to apply to simulation data, a combination of statistics and parameters is chosen such that each verbatim scenario is covered, and the statistics are easy to interpret by the user. I have chosen the following combination of statistics, as I believe they together cover several aspects of occurring verbatim, both on short-range and long-range and give a good understandable reflection of the present verbatim.

- **MHV( $d = 1, r = 1$ )** : Whereas the mean heat value does not perform well in all categories, especially failing in the 'Checkerboard' category, it is easy to understand to the user and gives a quick summary of the present short-range verbatim. The parameters  $d = 1, r = 1$  perform best overall from the tried *MHV* parameters, excluding the 'Checkerboard' category. This statistic will be referred to from now on as *MHV*.
- **TNA( $r = 3, T = 100m_{nh}$ )** : This statistic performs well in almost all short-range categories. Although is tied with the *TNA*:  $r = 3, T = 0.001$  statistic I believe this one provides better value as it is less 'hard-coded', although the  $S$  parameter is still fixed to 100. This statistic will be referred to from now on as *TNA<sub>s</sub>*.
- **TNA( $r_{max}, T = 100m_{nh}$ )** : This statistic performs well in almost all categories, but especially in the long-range category. Although it performs less overall than *TNA*( $r = r_{max}, T = 10m_{nh}$ ), the advantage that it has is that it reports less short-range verbatim. This may seem counter-intuitive and disadvantageous, but this makes it useful, as it reacts only when long-range verbatim occurs, but not when small short-range patches occur, which is what *MHV* and *TNA<sub>s</sub>* are good at in detecting. Another advantage of having a statistic with a larger  $r$  is that this statistic will only start increasing once verbatim patches are larger. For small patches, the threshold will not be exceeded, as there is not enough verbatim. This can come in handy when we only want to look for larger verbatim patches, ignoring noise, although this can also be achieved by doing patch size analysis, as mentioned in 4.2.4. This statistic will be referred to from now on as *TNA<sub>l</sub>*.

Name	MAE Full Random	MRE Full Verbatim	MRE Checkerboard	MRE Patches	MRE Long Range
MHV( $d = 1, r = 1$ )	0.0	0.0	0.555	0.093	0.9
MHV( $d = 2, r = 2$ )	0.0	0.0	0.464	0.104	0.899
MHV( $d = 1, r = 2$ )	0.0	0.0	0.446	0.107	0.899
MHV( $d = 2, r = 1$ )	0.0	0.0	0.555	0.093	0.9
TNA( $r = 1, T = 0.001$ )	0.0	0.0	0.333	0.005	0.665
TNA( $r = 2, T = 0.001$ )	0.0	0.0	0.0	0.003	0.32
TNA( $r = 3, T = 0.001$ )	0.001	0.0	0.0	0.004	0.099
TNA( $r = 3, T = 100m_{nh}$ )	0.001	0.0	0.0	0.004	0.099
TNA( $r = r_{max}, T = 0.1$ )	0.0	0.0	0.0	1.0	0.557
TNA( $r = r_{max}, T = 0.01$ )	0.0	0.0	0.0	0.137	0.0
TNA( $r = r_{max}, T = 0.001$ )	0.0	0.0	0.0	0.001	0.0
TNA( $r = r_{max}, T = 10m_{nh}$ )	0.0	0.0	0.0	0.001	0.0
TNA( $r = r_{max}, T = 100m_{nh}$ )	0.0	0.0	0.0	0.137	0.0

Table 5.1: Resulting MAE/MRE scores for each ran statistic on each dummy map category.

## 5.2 Simulation data results

In this section, the earlier found statistics from 5.1 will be analysed further to check whether they can give the user a well-informed view on how verbatim copy is present in simulation data.

### 5.2.1 Visual validation

For visual validation, the found statistics have been applied to a different number of simulations with varying  $n$  and  $k$  for both 'stone' and 'strebelle'. I have only visually analysed 'stone', the statistical results of the 'strebelle' simulations can be found in appendix B, but are not further elaborated on. Table 5.2 shows the relation between what the index map looks like and what the statistics show. On the index map, it is easy to see where verbatim patches are and how much verbatim would be present if visually estimating it. Table B1 shows the same numbers, but with the simulation map shown instead of the index map, for another reference point.

It can be seen that as  $n$  increases the image gets more patches and  $MHV$ ,  $TNA_l$  and  $TNA_s$  increase. As  $k$  increases the image gets noisier and  $MHV$ ,  $TNA_l$  and  $TNA_s$  decrease. The statistics that are given with the different combinations of  $n$  and  $k$  make sense and give a good reflection of the present verbatim copy in my opinion. Looking at the index map given by  $k = 1.0, n = 199$  for example, the index map shows a lot of directly copied patches of the source index map. The statistics  $MHV = 0.412$ ,  $TNA_l = 0.248$  and  $TNA_s = 0.513$  seem to fit this map well. Looking at the opposite end;  $k = 10.0, n = 1$ , we see that all the statistics indicate no verbatim copy, which is also reflected by visually looking at the index map; it looks very noisy. However, the presence of long-range verbatim copy is hard to visually validate, as this is not reflected in a single patch, as is the case with short-range verbatim copy, but in separate pixels that are in verbatim position to each other. This validation can be better explained by the relations of the long-range statistic  $TNA_l$  with  $k$  and  $n$  explained in section 5.2.2.

Another visual validation technique can be seen in figure 5.1, where it can be seen that verbatim patches can be cleanly detected using the statistics; for example by thresholding  $TNA_l = 1$  and  $TNA_s = 1$ .

### 5.2.2 Relation of QuickSampling parameters and statistics

The parameters that have been analysed from the QuickSampling algorithm are  $n$  and  $k$ . Every simulation and thus every combination of  $n$  and  $k$  from the available 'stones' simulation dataset as mentioned in 3.1 has been taken into account. For each combination, the simulation was analysed on both the  $MHV, TNA_s, TNA_l$  statistics as the patch statistics; mean patch size and max patch size. This means

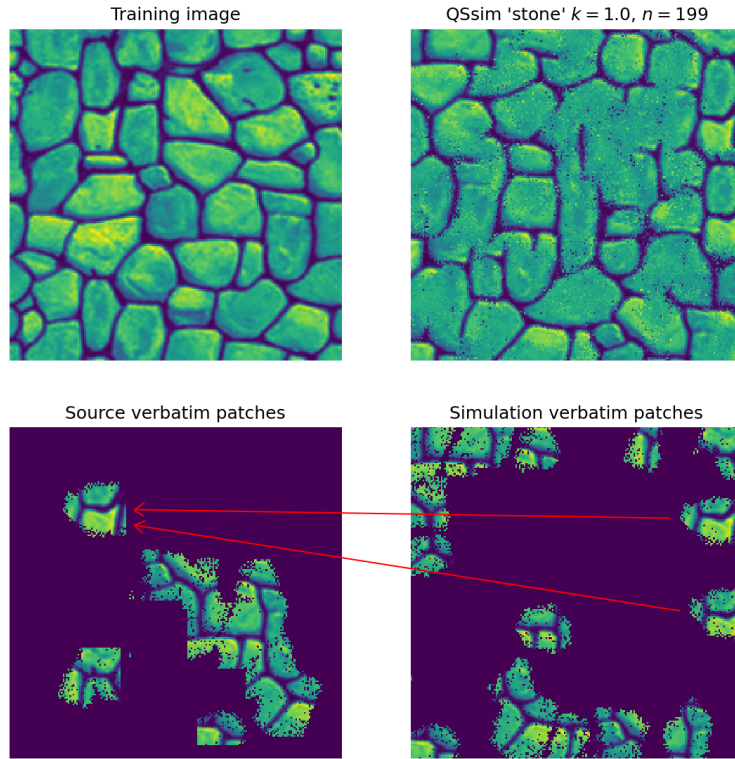
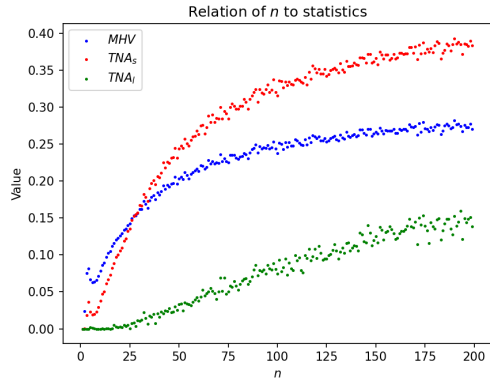


Figure 5.1: A combination of two statistics detecting problematic verbatim patches in a simulation, with the source patches shown. The red arrows illustrate where two simulation verbatim patches are sampled from.

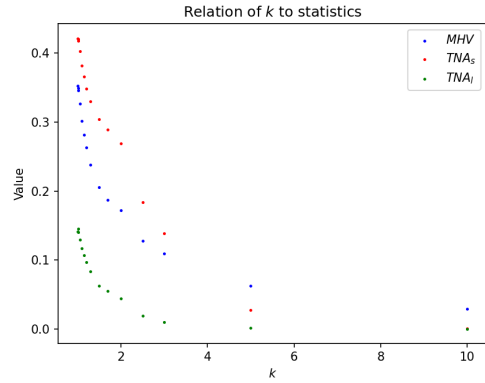
that for the figures where the effect of  $n$  was analyzed, each point represents an average value, with  $k$  ranging in  $[1, 1.01, 1.02, 1.05, 1.15, 1.2, 1.3, 1.5, 1.7, 2, 2.5, 3, 5, 10]$ . For the figures where the effect of  $k$  was analysed, each point represents an average value, with  $n$  ranging from 1 – 199. The results of this can be seen in figure 5.2 and figure 5.3.

As can be seen in figure 5.2 all the verbatim proportion statistics behave as expected; as  $n$  increases, the verbatim copy should increase, and as  $k$  increases, the verbatim copy should decrease, this is what the statistics indicate. As can be seen in figure 5.3 the patch statistics also behave as expected. As  $n$  increases, there is more verbatim; the mean and max patch sizes get bigger. As  $k$  increases, there is less verbatim; the mean and max patch sizes get smaller.

The difference between  $TNA_s$  and  $TNA_l$  can also clearly be seen;  $TNA_s$  starts increasing immediately as  $n$  increases from 1, but  $TNA_l$  stays near a value of 0. Only when verbatim starts occurring on a longer range the value starts increasing, but almost linearly, whilst the growth  $TNA_s$  stagnates. This is desired behaviour;  $TNA_l$  can be used to determine whether there is long-range verbatim, whilst  $TNA_s$  can be used to determine whether there is short-range verbatim.

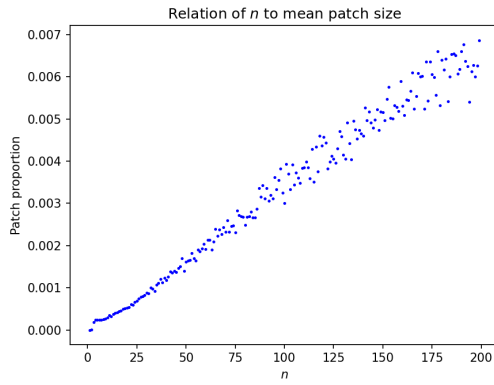


(a) Relation of  $n$  to the statistics

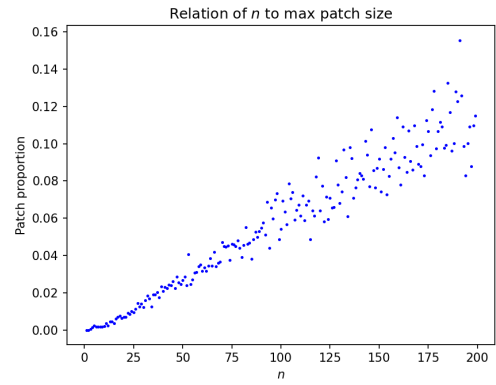


(b) Relation of  $k$  to the statistics

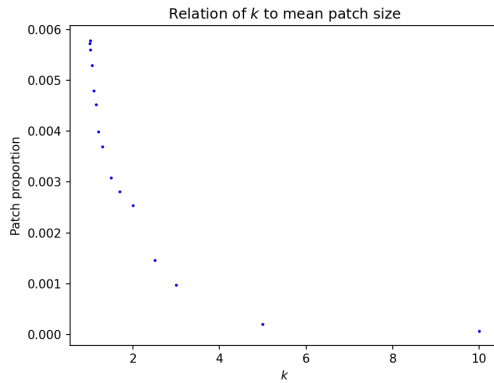
Figure 5.2: Relation of  $k$  and  $n$  to  $MHV$ ,  $TNA_s$  and  $TNA_l$  for the 'stone' simulations.



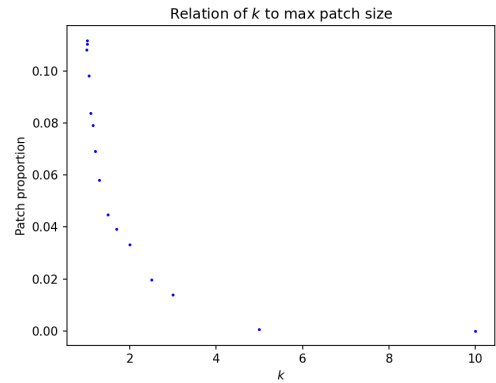
(a) Relation of  $n$  to mean patch size



(b) Relation of  $n$  to max patch size



(c) Relation of  $k$  to mean patch size



(d) Relation of  $k$  to max patch size

Figure 5.3: Relation of  $k$  and  $n$  to the patch statistics for the 'stone' simulations.

### 5.2.3 Spatial dependency analysis

The results of the spatial dependency analysis can be seen in figure 5.4. The left bar plots show that short-range verbatim is more occurring than large-range verbatim. Additionally, we can see in the right graphs that in this simulation, there is no interesting global occurring verbatim pattern. The value just keeps declining the farther away we are from the center pixel. We can also see that going from  $n = 5$  to  $n = 199$  both increases the average verbatim chance and makes verbatim occurring at larger distances more likely. Both are to be expected. Increasing the  $n$  decreases the number of possibilities that we can sample from increasing verbatim chance, and increasing the  $n$  makes each pixel in the simulation look at a bigger neighbourhood, increasing occurring verbatim at larger distances.

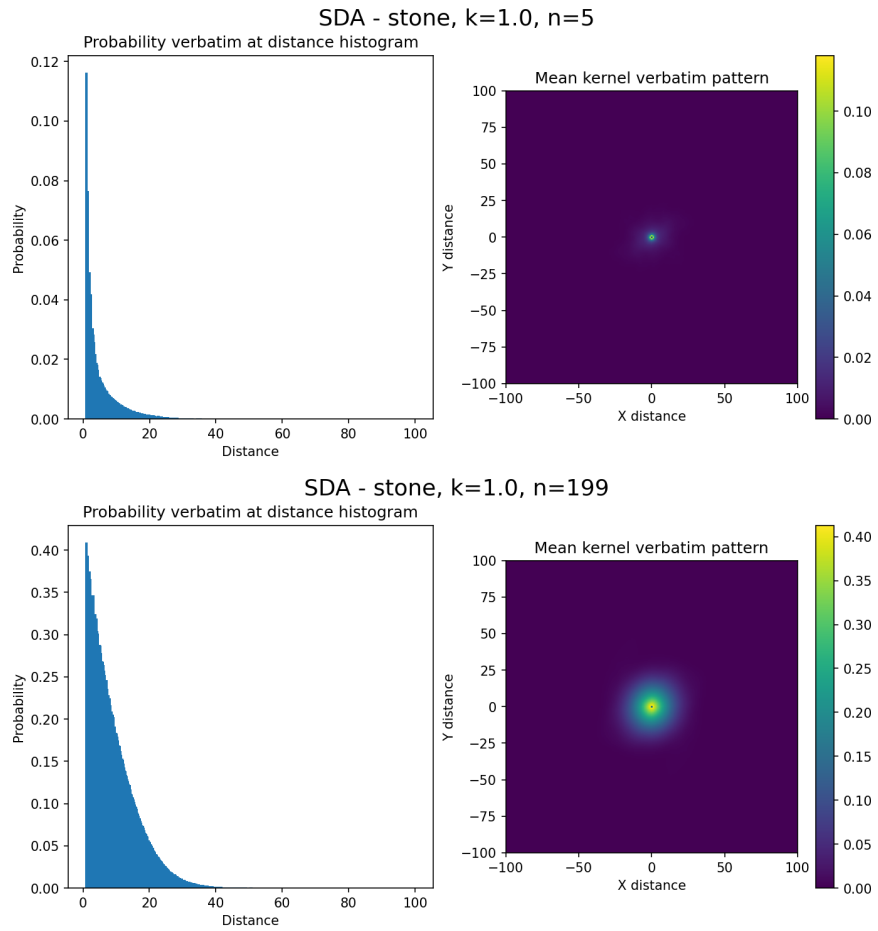


Figure 5.4: Verbatim spatial dependence analysed on a 'stone' simulation, for two different values of  $n$ .



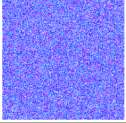
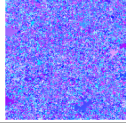
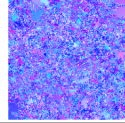
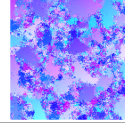
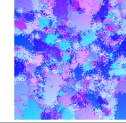
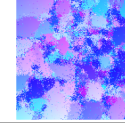
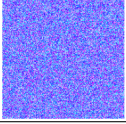
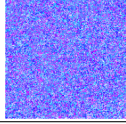
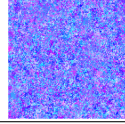
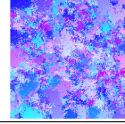
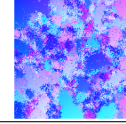
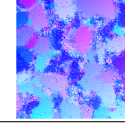
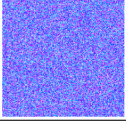
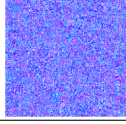
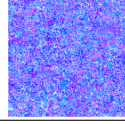
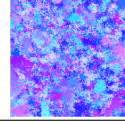
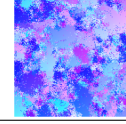
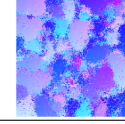
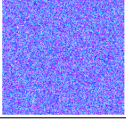
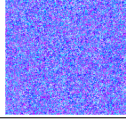
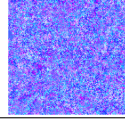
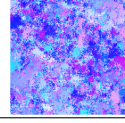
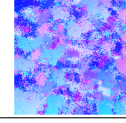
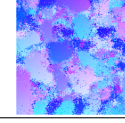
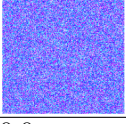
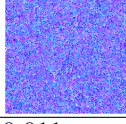
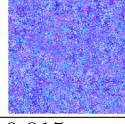
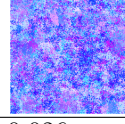
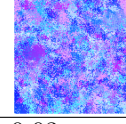
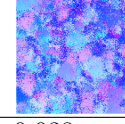
	$n = 1$	$n = 5$	$n = 10$	$n = 50$	$n = 100$	$n = 199$
$k = 1.0$						
$MHV$	0.0	0.117	0.139	0.351	0.362	0.412
$TNA_l$	0.0	0.0	0.0	0.115	0.118	0.248
$TNA_s$	0.0	0.066	0.103	0.407	0.448	0.513
$k = 1.5$						
$MHV$	0.0	0.056	0.07	0.186	0.221	0.241
$TNA_l$	0.0	0.0	0.0	0.0	0.101	0.087
$TNA_s$	0.0	0.005	0.025	0.251	0.335	0.397
$k = 2.5$						
$MHV$	0.0	0.036	0.046	0.105	0.148	0.177
$TNA_l$	0.0	0.0	0.0	0.0	0.015	0.101
$TNA_s$	0.0	0.001	0.004	0.11	0.228	0.322
$k = 3.0$						
$MHV$	0.0	0.031	0.039	0.096	0.121	0.139
$TNA_l$	0.0	0.0	0.0	0.0	0.022	0.023
$TNA_s$	0.0	0.0	0.003	0.093	0.166	0.22
$k = 10.0$						
$MHV$	0.0	0.011	0.015	0.026	0.03	0.038
$TNA_l$	0.0	0.0	0.0	0.0	0.0	0.0
$TNA_s$	0.0	0.0	0.0	0.0	0.0	0.002

Table 5.2: Statistics on various types of 'stone' simulations, with varying  $n$  and  $k$ . Showing the  $MHV$ ,  $TNA_l$  and  $TNA_s$  and the simulation index map.

## 6. Discussion

The parametrisation of both  $TNA$  and  $MHV$  have been optimised to fit the given circumstances that they have been analysed on in this thesis. In practice, the parameters used, such as  $r$ ,  $d$  and  $T$  should be calibrated to fit the users' needs, as they are case-dependent. Additionally,  $T$  can possibly be determined automatically using a dynamic threshold. Future research could look into how to statistically determine this dynamic threshold by using the fact that it is dependent on the simulation type and simulation / TI size. This would be a massive improvement on the currently used value of  $S \cdot m_{nh}$ , where  $S$  is hard-coded and  $m_{nh}$  is calculated by doing noise simulations.

Additionally, research can be done into neighbour verbatim, for example, with the approach as described in 4.1.2. How this influences certain parameters of statistics such as  $TNA$  and  $MHV$  was not explored due to time constraints.

Further improvements can also be yielded by looking into patch sizes of the training image, for example, analysing 'source verbatim patches' from figure 5.1. This thesis mainly looked at the patch proportions of the simulation image. The simulation sample rate per pixel of the training image can also be considered for this, meaning the number of times that a source pixel from the training image got sampled in the simulation. Then it can be seen whether some areas/patches are sampled too frequently from the source image. This could indicate over-representation of those areas, potentially indicating verbatim copy.

## 7. Conclusion

The thesis aimed to answer the following research question: “How can the proportion of present verbatim copy be summarized in pixel-based simulation methods?”

With the results found in chapter 5, we can conclude that the present verbatim copy can be summarized in pixel-based simulation methods. Both the Mean Heat Value and the Threshold Neighbour Analysis techniques proposed in chapter 4 are viable options to summarize the proportion of verbatim copy. Artificial verbatim copy can be detected with the methods, and promising results are shown on simulation data generated by QuickSampling. Both long- and short-range verbatim copy can be detected, on binary images such as 'strebelle' and grey-scale images such as 'stone'.

Although several parameters are case-dependent such as the threshold from Threshold Neighbour Analysis and the kernel radius depending on what is considered 'long-range' and 'short-range', they can be adjusted by a user to fit their specific use case. The amount of verbatim copy present can then be evaluated according to their preferences, with clearly defined parameters.

### *Acknowledgements*

I would like to thank my supervisor Mathieu Gravey for the encouraging supervision. I am grateful for the weekly guidance and valuable feedback that he provided.

### *Code availability*

The source code for the methods, results and the generated dummy maps can openly be found on Github<sup>1</sup>. The code is written in Python3, with readability and efficiency goals in mind.

### *Competing interests*

I declare that I have no conflicts of interest.

### *Ethical and legal considerations*

The data and source code that has been used does not contain any personal data and is openly available. The developed statistics thus do not present any ethical or legal considerations to be considered.

---

<sup>1</sup>[https://github.com/matthiasmeester/verbatim\\_copy\\_analysis](https://github.com/matthiasmeester/verbatim_copy_analysis)

# A. Full dummy data results

name	MAE Full random	MRE Full verbatim	MRE Checkerboard 3x3	MRE Checkerboard 1x1	MRE Checkerboard 5x5
MHV( $d = 1, r = 1$ )	0.0	0.0	0.332	1.0	0.332
MHV( $d = 2, r = 2$ )	0.0	0.0	0.41	0.573	0.41
MHV( $d = 1, r = 2$ )	0.0	0.0	0.442	0.454	0.442
MHV( $d = 2, r = 1$ )	0.0	0.0	0.332	1.0	0.332
TNA( $r = 1, T = 0.001$ )	0.0	0.0	0.0	1.0	0.0
TNA( $r = 2, T = 0.001$ )	0.0	0.0	0.0	0.0	0.0
TNA( $r = 3, T = 0.001$ )	0.001	0.0	0.0	0.0	0.0
TNA( $r = 3, T = 100m_{nh}$ )	0.001	0.0	0.0	0.0	0.0
TNA( $r = r_{max}, T = 0.1$ )	0.0	0.0	0.0	0.0	0.0
TNA( $r = r_{max}, T = 0.01$ )	0.0	0.0	0.0	0.0	0.0
TNA( $r = r_{max}, T = 0.001$ )	0.0	0.0	0.0	0.0	0.0
TNA( $r = r_{max}, T = 10m_{nh}$ )	0.0	0.0	0.0	0.0	0.0
TNA( $r = r_{max}, T = 100m_{nh}$ )	0.0	0.0	0.0	0.0	0.0

Table A1: Resulting MAE/MRE scores after 10 replications for full random, full verbatim and checkerboard dummy maps.

name	MRE Patches 5	MRE Patches 10	MRE Patches 20	MRE Long range 0.05	MRE Long range 0.10	MRE Long range 0.15
MHV( $d = 1, r = 1$ )	0.09	0.091	0.099	0.95	0.901	0.848
MHV( $d = 2, r = 2$ )	0.103	0.099	0.109	0.95	0.9	0.848
MHV( $d = 1, r = 2$ )	0.106	0.102	0.113	0.95	0.9	0.848
MHV( $d = 2, r = 1$ )	0.09	0.091	0.099	0.95	0.901	0.848
TNA( $r = 1, T = 0.001$ )	0.005	0.002	0.007	0.815	0.656	0.523
TNA( $r = 2, T = 0.001$ )	0.005	0.001	0.004	0.538	0.281	0.142
TNA( $r = 3, T = 0.001$ )	0.008	0.003	0.001	0.241	0.046	0.009
TNA( $r = 3, T = 100m_{nh}$ )	0.008	0.003	0.001	0.241	0.046	0.009
TNA( $r = r_{max}, T = 0.1$ )	1.0	1.0	1.0	1.0	0.671	0.0
TNA( $r = r_{max}, T = 0.01$ )	0.164	0.11	0.137	0.0	0.0	0.0
TNA( $r = r_{max}, T = 0.001$ )	0.001	0.001	0.002	0.0	0.0	0.0
TNA( $r = r_{max}, T = 10m_{nh}$ )	0.001	0.001	0.002	0.0	0.0	0.0
TNA( $r = r_{max}, T = 100m_{nh}$ )	0.164	0.11	0.135	0.0	0.0	0.0

Table A2: Resulting MAE/MRE scores after 10 replications for patches and long range dummy maps.

## B. Additional simulation data results

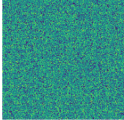
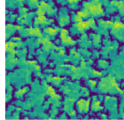
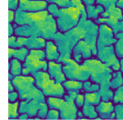
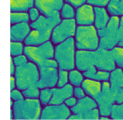
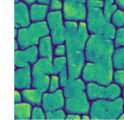
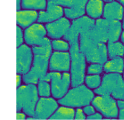
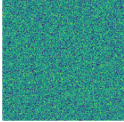
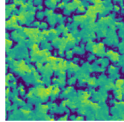
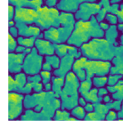
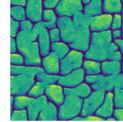
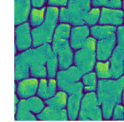
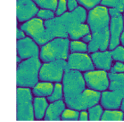
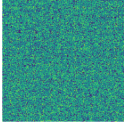
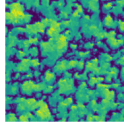
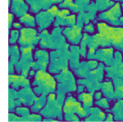
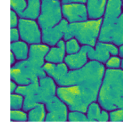
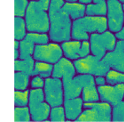
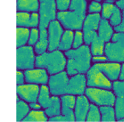
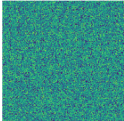
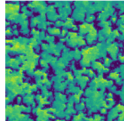
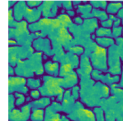
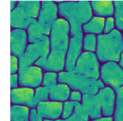
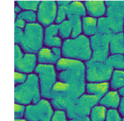
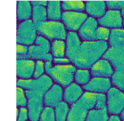
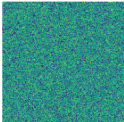
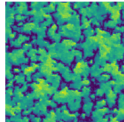
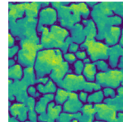
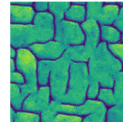
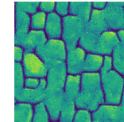
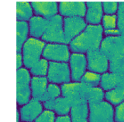
	$n = 1$	$n = 5$	$n = 10$	$n = 50$	$n = 100$	$n = 199$
$k = 1.0$						
$MHV$	0.0	0.117	0.139	0.351	0.362	0.412
$TNA_l$	0.0	0.0	0.0	0.115	0.118	0.248
$TNA_s$	0.0	0.066	0.103	0.407	0.448	0.513
$k = 1.5$						
$MHV$	0.0	0.056	0.07	0.186	0.221	0.241
$TNA_l$	0.0	0.0	0.0	0.0	0.101	0.087
$TNA_s$	0.0	0.005	0.025	0.251	0.335	0.397
$k = 2.5$						
$MHV$	0.0	0.036	0.046	0.105	0.148	0.177
$TNA_l$	0.0	0.0	0.0	0.0	0.015	0.101
$TNA_s$	0.0	0.001	0.004	0.11	0.228	0.322
$k = 3.0$						
$MHV$	0.0	0.031	0.039	0.096	0.121	0.139
$TNA_l$	0.0	0.0	0.0	0.0	0.022	0.023
$TNA_s$	0.0	0.0	0.003	0.093	0.166	0.22
$k = 10.0$						
$MHV$	0.0	0.011	0.015	0.026	0.03	0.038
$TNA_l$	0.0	0.0	0.0	0.0	0.0	0.0
$TNA_s$	0.0	0.0	0.0	0.0	0.0	0.002

Table B1: Statistics on various types of 'stone' simulations, with varying  $n$  and  $k$ . Showing the  $MHV$ ,  $TNA_l$  and  $TNA_s$  and the simulation.

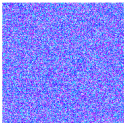
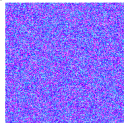
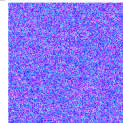
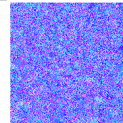
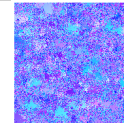
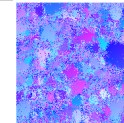
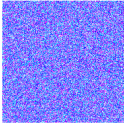
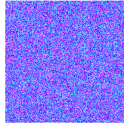
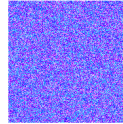
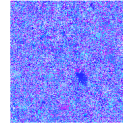
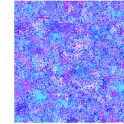
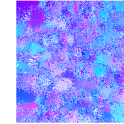
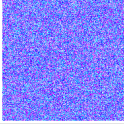
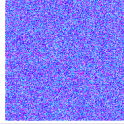
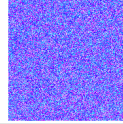
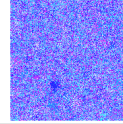
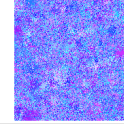
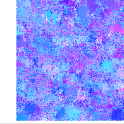
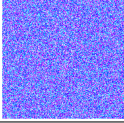
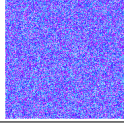
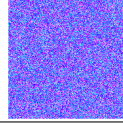
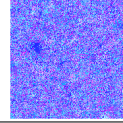
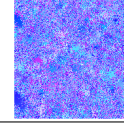
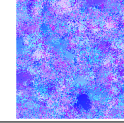
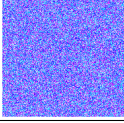
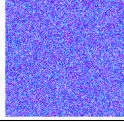
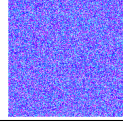
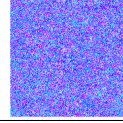
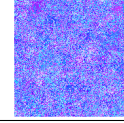
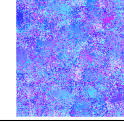
	$n = 1$	$n = 5$	$n = 10$	$n = 50$	$n = 100$	$n = 199$
$k = 1.0$						
$MHV$	0.0	0.001	0.002	0.04	0.122	0.273
$TNA_l$	0.0	0.0	0.0	0.0	0.02	0.127
$TNA_s$	0.0	0.0	0.0	0.05	0.194	0.413
$k = 1.5$						
$MHV$	0.0	0.001	0.002	0.024	0.074	0.151
$TNA_l$	0.0	0.0	0.0	0.0	0.0	0.06
$TNA_s$	0.0	0.0	0.0	0.025	0.131	0.307
$k = 2.5$						
$MHV$	0.0	0.001	0.002	0.02	0.044	0.084
$TNA_l$	0.0	0.0	0.0	0.0	0.0	0.0
$TNA_s$	0.0	0.0	0.0	0.016	0.06	0.173
$k = 3.0$						
$MHV$	0.0	0.001	0.002	0.018	0.041	0.064
$TNA_l$	0.0	0.0	0.0	0.0	0.0	0.0
$TNA_s$	0.0	0.0	0.0	0.011	0.055	0.115
$k = 10.0$						
$MHV$	0.0	0.001	0.001	0.008	0.014	0.025
$TNA_l$	0.0	0.0	0.0	0.0	0.0	0.0
$TNA_s$	0.0	0.0	0.0	0.001	0.002	0.006

Table B2: Statistics on various types of 'strebelle' simulations, with varying  $n$  and  $k$ . Showing the  $MHV$ ,  $TNA_l$  and  $TNA_s$  and the simulation index map.



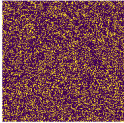
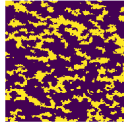
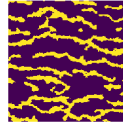
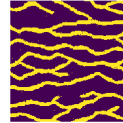
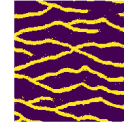
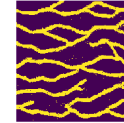
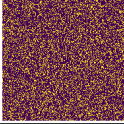
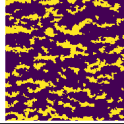
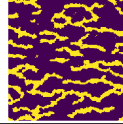
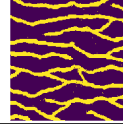
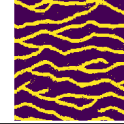
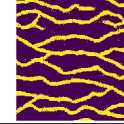
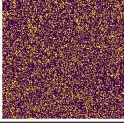
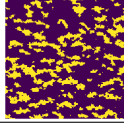
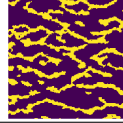
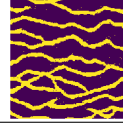
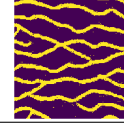
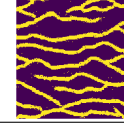
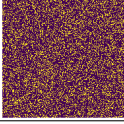
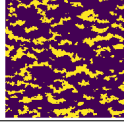
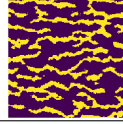
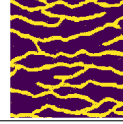
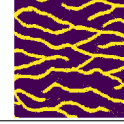
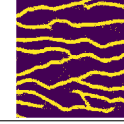
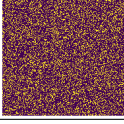
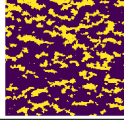
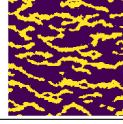
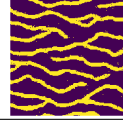
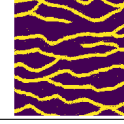
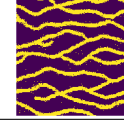
	$n = 1$	$n = 5$	$n = 10$	$n = 50$	$n = 100$	$n = 199$
$k = 1.0$						
$MHV$	0.0	0.001	0.002	0.04	0.122	0.273
$TNA_l$	0.0	0.0	0.0	0.0	0.02	0.127
$TNA_s$	0.0	0.0	0.0	0.05	0.194	0.413
$k = 1.5$						
$MHV$	0.0	0.001	0.002	0.024	0.074	0.151
$TNA_l$	0.0	0.0	0.0	0.0	0.0	0.06
$TNA_s$	0.0	0.0	0.0	0.025	0.131	0.307
$k = 2.5$						
$MHV$	0.0	0.001	0.002	0.02	0.044	0.084
$TNA_l$	0.0	0.0	0.0	0.0	0.0	0.0
$TNA_s$	0.0	0.0	0.0	0.016	0.06	0.173
$k = 3.0$						
$MHV$	0.0	0.001	0.002	0.018	0.041	0.064
$TNA_l$	0.0	0.0	0.0	0.0	0.0	0.0
$TNA_s$	0.0	0.0	0.0	0.011	0.055	0.115
$k = 10.0$						
$MHV$	0.0	0.001	0.001	0.008	0.014	0.025
$TNA_l$	0.0	0.0	0.0	0.0	0.0	0.0
$TNA_s$	0.0	0.0	0.0	0.001	0.002	0.006

Table B3: Statistics on various types of 'strebelle' simulations, with varying  $n$  and  $k$ . Showing the  $MHV$ ,  $TNA_l$  and  $TNA_s$  and the simulation.

# Bibliography

- Abdollahifard, M. J., Mariétoz, G., & Ghavim, M. (2019). Quantitative evaluation of multiple-point simulations using image segmentation and texture descriptors. *Computational Geosciences*, 23(6), 1349–1368.
- Arpat, G. B., & Caers, J. (2005). A multiple-scale, pattern-based approach to sequential simulation. In *Geostatistics banff 2004* (pp. 255–264). Springer.
- Bolelli, F., Allegretti, S., Baraldi, L., & Grana, C. (2019). Spaghetti labeling: Directed acyclic graphs for block-based connected components labeling. *IEEE Transactions on Image Processing*, 29, 1999–2012.
- Gravey, M., & Mariétoz, G. (2020). Quicksampling v1. 0: a robust and simplified pixel-based multiple-point simulation approach. *Geoscientific Model Development*, 13(6), 2611–2630.
- Lefebvre, S., & Hoppe, H. (2005). Parallel controllable texture synthesis. In *Acm siggraph 2005 papers* (pp. 777–786).
- Li, X., Mariétoz, G., Lu, D., & Linde, N. (2016). Patch-based iterative conditional geostatistical simulation using graph cuts. *Water Resources Research*, 52(8), 6297–6320.
- Mahmud, K., Mariétoz, G., Caers, J., Tahmasebi, P., & Baker, A. (2014). Simulation of earth textures by conditional image quilting. *Water Resources Research*, 50(4), 3088–3107.
- Mariétoz, G., & Caers, J. (2014). *Multiple-point geostatistics: stochastic modeling with training images*. John Wiley & Sons.
- Mariétoz, G., Renard, P., & Straubhaar, J. (2010). The direct sampling method to perform multiple-point geostatistical simulations. *Water Resources Research*, 46(11).
- Straubhaar, J., Renard, P., Mariétoz, G., Froidevaux, R., & Besson, O. (2011). An improved parallel multiple-point algorithm using a list approach. *Mathematical Geosciences*, 43(3), 305–328.
- Strebelle, S. (2002). Conditional simulation of complex geological structures using multiple-point statistics. *Mathematical geology*, 34(1), 1–21.
- Tahmasebi, P., Hezarkhani, A., & Sahimi, M. (2012). Multiple-point geostatistical modeling based on the cross-correlation functions. *Computational Geosciences*, 16(3), 779–797.