Computers & Geosciences 54 (2013) 293–308

Contents lists available at SciVerse ScienceDirect

Computers & Geosciences

journal homepage: www.elsevier.com/locate/cageo

Multiple-point geostatistical simulation using the bunch-pasting direct sampling method

Hassan Rezaee [a,*,1], Gregoire Mariethoz [b], Mohammad Koneshloo [c], Omid Asghari [d]

[a] Department of Mining Engineering, University College of Engineering, University of Tehran, Tehran, Iran
[b] School of Civil and Environmental Engineering, University of New South Wales, Sydney, NSW, Australia
[c] Department of Mining, Petroleum & Geophysics Engineering, Shahrood University of Technology, Shahrood, Iran
[d] Department of Mining Engineering, University College of Engineering, University of Tehran, Tehran, Iran

ARTICLE INFO

Article history:
Received 13 October 2012
Received in revised form
27 January 2013
Accepted 29 January 2013
Available online 13 February 2013

Keywords:
Patch-based simulation
Unilateral simulation path
Conditioning
Computational benefits
Matlab implementation

ABSTRACT

Multiple-point geostatistics has opened a new field of methodologies by which complex geological phenomena have been modeled efficiently. In this study, a modified form of direct sampling (DS) method is introduced which not only keeps the strength of DS simulation technique but also speeds it up by one or two orders of magnitude. While previous methods are based on pasting only one point at a time, here the simulation is done by pasting a bunch of nodes at a time, effectively combining the flexibility of DS with the computational advantages of patch-based methods. This bears the potential of significantly speeding up the DS method. The proposed simulation method can be used with unilateral or random simulation paths. No overlap occurs in the simulation procedure because the bunch takes the shape of the empty space around the simulated nodes. Systematic tests are carried on different training images including both categorical and continuous variables, showing that the realizations preserve the patterns existent in the training image. To illustrate the method, a Matlab implementation of the method is attached to the paper.

1. Introduction

Complex geological scenarios pose many difficulties in the process of modeling and simulation. Meander channels in hydrocarbon reservoirs, post-mineralization phenomena in mineral deposits, complex underground water conduits and labyrinth secondary permeability in fractured reservoirs are some examples of such complexity. Two-point geostatistics (Chiles and Delfiner, 1999; Goovaerts, 1997; Isaaks, 1990; Journel, 1983) has brought insights in many problems geosciences and in particular of mining and petroleum industry by accounting for the spatial relationships of data. However, it fails at reproducing complicated and non-linear natural phenomena for example curvilinear shapes (Zinn and Harvey, 2003; Bianchi et al., 2011), or variables that present complex non-linear or even non-unique dependencies (Wackernagel, 2003; Mariethoz et al., 2009). Such continuities are difficult to define using variogram-based geostatistical methods; they require more than two points or a pattern of nodes to be fully modeled. The abundance of such complicated geological phenomena calls for methods that account for more than two points at a time. To solve this type of problem, various methods have been proposed and further developed by many authors.

One way to deal with the above mentioned complexities is to simulate the objects rather than points. Such methods are so called object-based methods (Deutsch and Wang, 1996; Haldorsen and Lake, 1984; Holden et al., 1998; Stoyan et al., 1987), which encompass marked point processes (Kleingeld et al., 1997) and Boolean methods (Lantuejoul, 2001). Geobody's features are drawn from a pre-defined distribution and scattered over the simulation grid. The main problem of these methods appears in the conditional simulation because it is not straightforward to honor large amounts of conditioning data. Conditional simulation is a fundamental task in the practice of modeling since often we aim at producing precise realizations which honor data at some known points. The advent of multiple-point geostatistics filled this gap by providing realistic geology while honoring conditioning data.

Multiple-point geostatistical simulation was firstly introduced by Guardiano and Srivastava (1993), which laid the basic concept for almost all further algorithms introduced in this field till now. They based their idea on using a geological conceptual model named "*training image*" from which high order spatial statistics are borrowed and used for the simulation. Several algorithms to

* Corresponding author. Tel.: +98 9159860773; fax: +98 2188008838.
E-mail addresses: h.rezaee@ut.ac.ir, hassan_rezaee65@yahoo.com (H. Rezaee), gregoire.mariethoz@minds.ch (G. Mariethoz), web2-koneshloo@shahroodut.ac.ir (M. Koneshloo), o.asghari@ut.ac.ir (O. Asghari).
[1] This is the present address and exposed to change in future.

0098-3004/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.cageo.2013.01.020

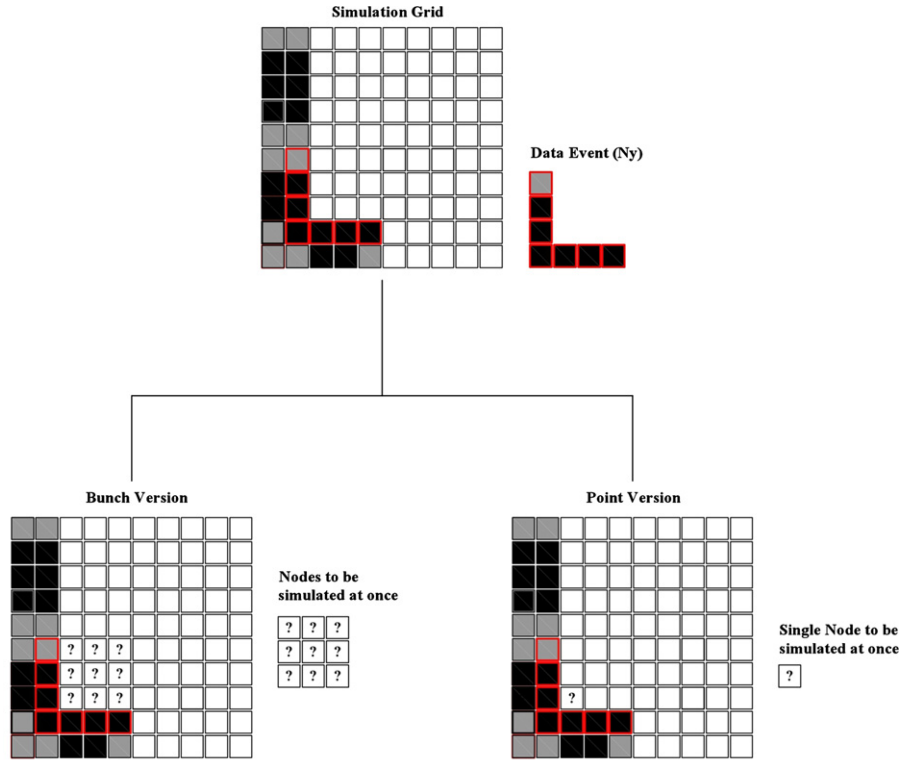

Fig. 1. Pixel-based and bunch-based states in a simulation process. The data event is shown on top. At the bottom, the left sub figure shows the bunch version where nine nodes are pasted to the simulation grid at once and the right sub figure shows the case where just one node is simulated at once.

perform multiple-point simulations are currently available. Broadly, multiple-point methods can be divided in two main categories: pixel-based methods that simulate one node at a time, and patch-based methods that simulate groups of nodes at a time. Among the pixel-based ones, Strebelle (2002) proposed the SNESIM algorithm, in which the conditional probability of a data events occurrences are stored in a tree-like structure, or search tree, in which all patterns existent in the training image are saved once in advance and are then used for extracting similar data events. SNESIM suffers from two main shortcomings: first, it applies only to categorical training images and secondly it lacks computational efficiency when performing simulation with a large number of facies. As an alternative, Straubhaar et al. (2011) proposed the IMPALA code that is equivalent to SNESIM but uses lists instead of trees, providing lighter memory needs at the price of higher computational cost, which can be alleviated through parallelization. Some of the limitations of these methods were partly addressed by the patch-based FILTERSIM algorithm (Zhang et al., 2006; Wu et al., 2008). This method not only solves the lack of computational efficiency of SNESIM but also accommodates both categorical and continuous variables. Arpat and Caers (2007) introduced the SIMPAT algorithm, which is patch-based and uses distance functions to calculate the similarity between training image and conditioning data events. Lately, Honarkhah and Caers (2010) used the multidimensional scaling (MDS) to cluster the non-linear systems which, in FILTERSIM, were done assuming that clusters are separable by linear functions. With the patch-based CCSIM method (Tahmasebi et al., 2012a,b), a cross-correlation is used to compute the similarity between patterns. The method is computationally efficient and delivers realizations which reproduce TI's patterns well. Direct sampling (DS) is a pixel-based multiple-point geostatistical simulation method which has been introduced by Mariethoz and Renard (2010) and Mariethoz et al. (2010). With this method, once the conditioning data event is found in the simulation grid, the algorithm looks for a similar pattern in the training image. The most distinguishing feature when compared with previously-introduced methods is that at the first occurrence of a similar pattern in the training image, the central node is pasted in the simulation.

Some unique features of the DS are its ability to work with multiple variables simultaneously, to use specific distances such as the transform-invariant distances (Mariethoz and Kelly, 2011), and the very light memory requirement. However, for large training images and large number of simulated nodes, the computational effort can be large. In this study, a new method is proposed which can potentially solve this problem. We extend the DS algorithm to use it with patches, therefore accelerating by a factor proportional to the patch size. The basic idea consists in not only pasting the central node of the data event in the simulation grid, but also a bunch of other nodes close to the central node (see Fig. 1).

In the next sections the method is described and evaluated with systematic tests. The performance of the proposed method in conditioning is tested with satisfying results. Patterns reproduction is examined for several training images, and validation is carried on using quantitative tools. Compared to original DS algorithm, this computational gain is found to scale linearly with the bunch size, without significant loss of quality in the results except for very large bunch sizes.

2. Notations

- $\boldsymbol{Z(x)}$ is the random variable which is defined in R in 2D or 3D.
- **SG** denotes the simulation grid.
- **TI** stands for training image; accordingly TI($\boldsymbol{u}$) is the value of TI at point $\boldsymbol{u}$.
- $\boldsymbol{N}_y$ and $\boldsymbol{N}_x$ are data events extracted from training image and conditioning data (from **SG**), respectively.

- $d\{\boldsymbol{N}_x, \boldsymbol{N}_y\}$ is a distance function through which two data events are compared with each other. In the simplest form, it can be Manhattan distance. The distance function is determined by the type of variable simulated: whether categorical or continuous.
- $\boldsymbol{E}$ is the number of nodes in the neighborhood of the simulated node in $\boldsymbol{SG}$ whose value is not determined either by hard or previously simulated data.
- $\boldsymbol{T}$ is the window size within which $\boldsymbol{N}_x$ is extracted.
- $\boldsymbol{B}$ is the bunch size: maximum number of nodes in a bunch. For example, a bunch of size nine which allows pasting eight surrounding nodes which with the addition of the central node sums to nine nodes.
- $\boldsymbol{F_r}$ is the fraction of TI allowed to be scanned in every simulation node.

3. Methodology

The DS method was originally developed by Mariethoz et al. (2010). Below is a short description of this first DS simulation algorithm which is then followed with the new algorithm proposed in this study. At the end the new conditioning strategy applied is described.

3.1. Direct sampling simulation

The idea of DS simulation is derived from Shannon's (1948) paper on producing Markovian sequences of random English by drawing letters from a book conditionally to previous occurrences. In this method no pattern database like what exists in the form of search tree in SNESIM (Strebelle, 2002) and pattern data base (patdb) in SIMPAT (Arpat and Caers, 2007), is used. This lowers the load of algorithm that can result in lower PC configuration requirements.

To avoid repeating the description of DS simulation, the readers are referred to Mariethoz et al. (2010) for the detail of the method. Here just the procedure of DS simulation is summarized in the form of the algorithm below:

1. Define a random or unilateral path through which nodes are visited
   For each node under simulation do:
2. Extract $\boldsymbol{N}_x$ within a window of specific size or number of nodes.
3. Using a random or unilateral path through which $\boldsymbol{F_r}$ is scanned, do for each TI node:
   Based on the lag vectors by which $\boldsymbol{N_x}$ is defined extract $\boldsymbol{N}_y$. Calculate $d\{\boldsymbol{N}_x, \boldsymbol{N}_y\}$ and check the following:
   a) If the dissimilarity is less than the minimum of distances which are calculated so far, store the central node's value of $\boldsymbol{N_y}$.
   b) If the distance is less than the pre-defined distance threshold, assign the value of the central node of $\boldsymbol{N_y}$ to the simulated node in **SG** and go on to simulate the next node.
   c) If the number of iterations is more than a pre-defined maximum, assign the training image's data event central node corresponding to the minimum distance.
4. End of the simulation.

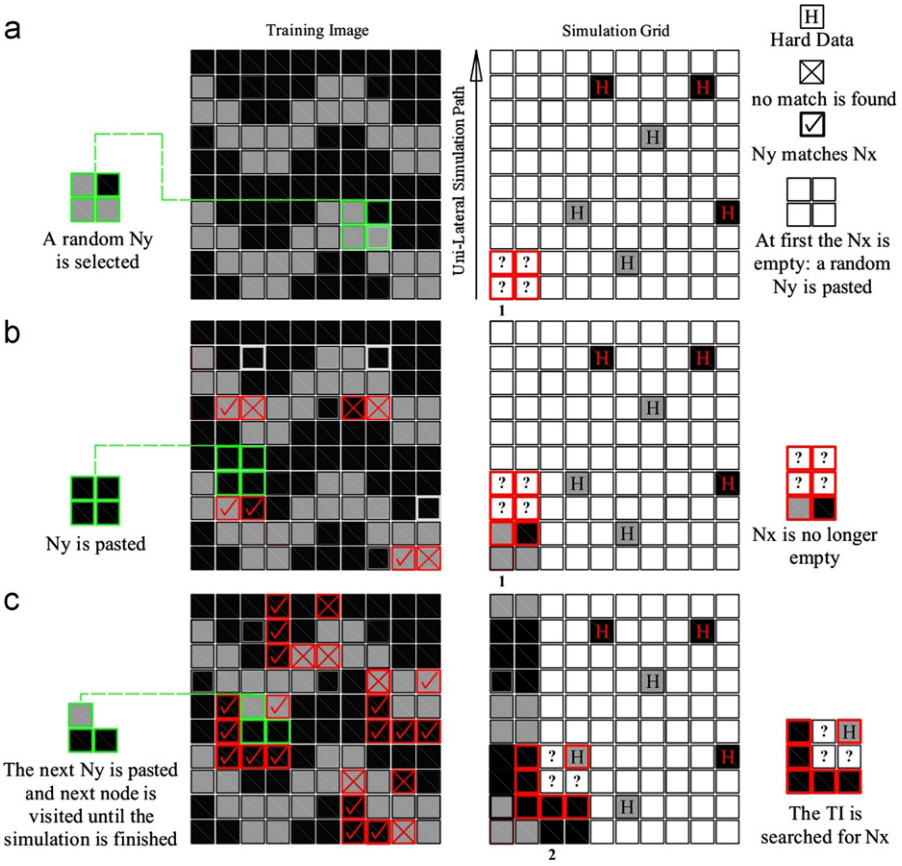

Fig. 2. Bunch-pasting DS algorithm flowchart. Unilateral simulation path is used to illustrate the algorithm. Random path can be used identically.

In the original implementation of the DS algorithm, once a location is sampled in the TI, only one point is pasted at a time, which takes a lot of CPU time. In the proposed methodology, a "*bunch*" of nodes is pasted at a time.

3.2. Bunch-pasting direct sampling simulation

In the DS, the scan of the TI can be seen as a search for appropriate nodes locations in the TI. When a suitable *location* is found in the TI (i.e., with a distance under the threshold), the corresponding value is assigned by a simple look-up operation. The DS method can therefore be seen as a simulation of locations in the TI rather than actual values. In this process, one often observes that the values of nodes close to each in the training image tend to be also close to each other in the simulation. With this concept in mind, it appears that pasting a bunch of pixels rather than a single one is a shortcut that may lead to a very similar outcome. This bears the potential of accelerating the speed of the algorithm. Since with DS no conditional probability is calculated, applying the bunch pasting idea is straightforward. In this section the idea of bunch pasting DS is introduced.

At first, a simulation path, unilateral or random, is defined through the SG. The methodology is illustrated in Fig. 2. In this figure a unilateral simulation path is defined in the vertical direction. It also shows that the TI is scanned through a random path. As with other methods, the path can be defined in either direction, but the direction may affect the results, therefore a sensitivity analysis should be carried on. After that, the informed nodes within T are extracted to form N_x. For the first simulated nodes (Fig. 2a) and in a conditional simulation mode, the first $\boldsymbol{N}_x$ is empty. In this case, the usual approach in sequential simulation is to take a value from the marginal probability (histogram) when there are no conditioning data. Here a random $\boldsymbol{N}_y$ is selected from the **TI** and pasted at the location of the simulated node. For the subsequent bunch (Fig. 2b), the conditioning data event consists of two nodes because the maximum size of the window is set to 3×3 nodes. When the simulation is further in the path (Fig. 2c), the data event can have a different shape because in this case there are conditioning data below and on the left of the central node. Since the DS does not store data events associated to a particular shape, the changes in data event can be accommodated in a straightforward manner. In Fig. 2c, a hard conditioning data is found at the upper-right corner of the search window as well as the previously-simulated conditioning point on the left and bottom side. This hard data is captured in the N_x and the TI is scanned for that specific configuration.

The $\boldsymbol{N}_y$ bunch shape is directly determined by that of $\boldsymbol{N}_x$ (Fig. 3) such that it fills the gaps between the known nodes. Fig. 3 illustrates this in the case of a random simulation path where known and unknown nodes are typically disposed in a disordered manner. A window including $\boldsymbol{B}$ nodes is placed on the central node. The bunch can be of squared, circular or disc shape. Only the values at the empty nodes are pasted (i.e., nodes which have not been simulated nor hard conditioning data).

The bunch shape is defined based on lag vectors (in the case of a bunch of size nine; h_i; $i=1,\ldots,9$) which are indicators of the relative position of each node from the simulated node. Hence the shape of the bunch is flexible and changes every time a node is

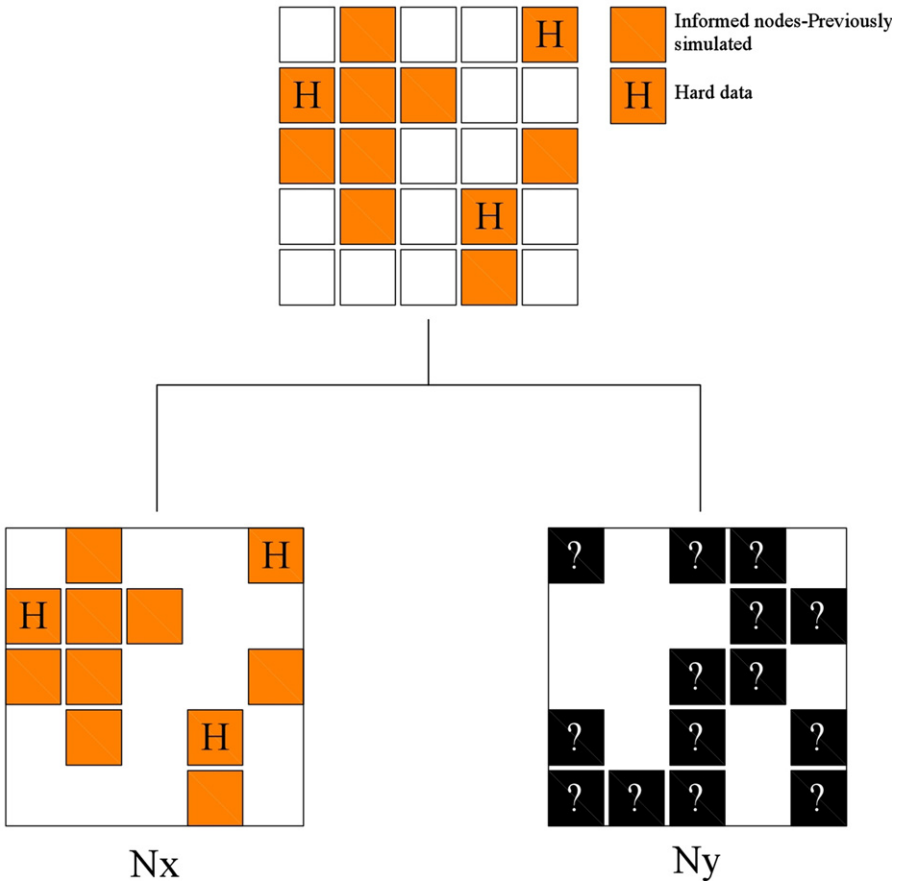

Fig. 3. Above: hard conditioning and previously-simulated data within a square search window in SG, below-left: conditioning data event ($\boldsymbol{N}_x$), below-right: TI data event ($\boldsymbol{N}_y$).

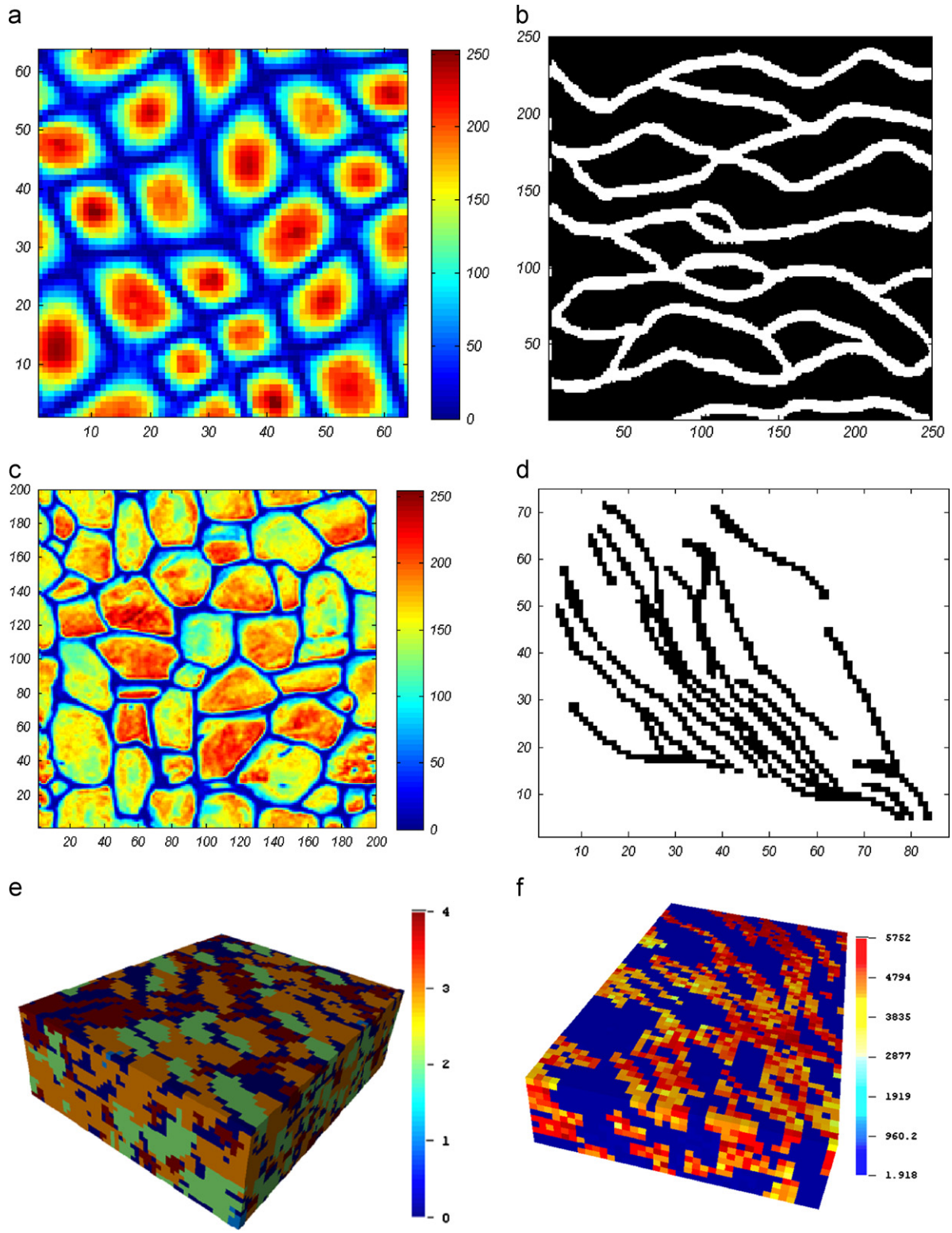

Fig. 4. Training images used in this study.

simulated, but the shape of the space within which the bunch of nodes is extracted is fixed during the simulation. In Fig. 3 this space is squared in shape.

Having pasted the first bunch of nodes, the N_x is no longer empty and it contains informed nodes which belong to the previously simulated ones (for non-conditional simulation or can be the case of conditional simulation with sparse hard data). Like the original DS algorithm, the **TI** is searched for the similar N_y using the distance function. A new random path in the **TI** each time would be very inefficient computationally. Instead a unilateral path with starting point is adopted (when it arrives at the end it restarts at the beginning). This starting point of the path for each simulated node is randomly selected. Like in the DS algorithm, once N_y with distance below the threshold is found, the

search process is stopped and the bunch of nodes, whose shape is determined by the empty space around the node in **SG**, is pasted on to the **SG**.

Since searching the whole **TI** for each node is computationally expensive, $\boldsymbol{F_r}$ is set. If the iteration number exceeds $\boldsymbol{F_r}$, the $\mathbf{N_y}$ with minimum distance is pasted on the **SG**. $\boldsymbol{F_r}$ is usually 0.1–0.5 of the size of **TI**. However, using a higher $\boldsymbol{F_r}$**s** can improve the realizations.

The used distance function depends on the type of variable under study. Although several types of distance have been proposed (Mariethoz and Kelly, 2011; Mariethoz et al., 2010), in the case of a categorical variable, the following relation is used to quantify the dissimilarity between $\boldsymbol{N_x}$ and $\boldsymbol{N_y}$:

$$d\{N_x,N_y\} = \sum_{i=1}^{E} Z_i,\ Z_i = \begin{cases} 1 & \text{if} \quad N_x(hi) = N_y(h_i) \\ 0 & \text{if not} \end{cases} \tag{1}$$

In the case of dealing with continuous variable the function changes into the following:

$$d\{N_x,N_y\} = \frac{1}{n}\sum_{i=1}^{E} N_x(h_i) - N_y(h_i)/d_{\max}, \qquad \in [0,\ 1] \tag{2}$$

$$d_{\max} = \max(Z(y)) - \min(Z(y)), \qquad y \in \mathrm{TI}$$

3.3. The new conditioning strategy

Pasting bunch of nodes instead of one can deteriorate the conditioning capabilities of the DS algorithm. For this problem, one idea is that when computing the distance, one can use a different weight for each pixel in the data event. A larger weight for the pixels corresponding to conditioning data can be used. For example, if one of the pixels is a conditioning point, it counts 10 times more in the distance than the other pixels. This is also what is implemented in FILTERSIM (Zhang et al., 2006) and recalled in DS (Mariethoz et al., 2010). This means that if it is difficult to find a pattern in the training image, the algorithm will choose one that at least matches the data well, even if it results in an artifact in different portions of the domain. The weight of the conditioning data (10 in this paper) tells how strong the conditioning should be.

4. Simulation results

In this study systematic tests have been carried out to assess the potential of bunch-pasting DS algorithm in simulating complex phenomena. This section briefly introduces the training images used, and then the corresponding realizations and the results are shown and discussed further.

4.1. TIs used in this study

Six training images are used which are introduced and illustrated in Fig. 4. For convenience, these training images are named ***A***, ***B***, ***C*** and ***D***.

- *Training image* ***A*** (Wei and Levoy, 2000): TI ***A*** illustrates a pattern of textures containing repeating areas of high and low values, with high connectivity of the low values. The training image is illustrated in Fig. 4a.
- *Training image* ***B***: Strebelle (2002) used the channel training image for the assessment of SNESIM algorithm. It displays meander channels which are of paramount importance in reservoir modeling and simulation (Fig. 4b).

Table 1
Training images specifics.

TI	Size	Cat./cont.	Proportion/ mean	No. of facies	Range of variable
A	64×64	Continuous	98.37	–	0–250
B	250×250	Categorical	0.26	2	–
C	200×200	Continuous	127.87	–	0–250
D	75×88	Categorical	0.1476	2	–
E	$51 \times 62 \times 20$	Categorical	See Table 3	4	–
F	$60 \times 37 \times 11$	Continuous	2102.8	–	1.8–5752

Table 2
Input parameters and their notations used in simulation.

Parameter	**Index**
Bunch size	B
Search radii	R_x and R_y
Fraction	$\boldsymbol{F_r}$
Distance threshold	δ
Maximum number of nodes in dataevent	MaxNb
Window type	W
Simulation path mode	Path

- *Training image* ***C***: Wall training image (Zhang et al., 2006) is another example of a continuous TI which is used to check the proposed algorithm's performance in dealing with more complicated structures (Fig. 4c). This TI shows a complex structure of bricks with a high degree of connectivity of the lowest values. Further tests using this training image are carried on in Section 6.1.
- *Training image* ***D***: This image (Rezaee, 2012) shows the dyke-like structures occurring in a porphyry copper deposit that is intruded in secondary phase of intrusion (Fig. 4d).
- Training image ***E***: 3D example corresponding to a facies representation of a carbonate reservoir (Fig. 4e).
- *Training image* ***F***: This TI is from Stanford *V* database accessed online[2] in Stanford Center for Reservoir Forecasting Center. It shows spatially varying porosity of facies within the channels. A part of this TI is selected and shown in Fig. 4f.

The major parameters of the TIs are summarized in Table 1.

4.2. Tests for conditional simulation

This section shows and discusses the results of the bunch-pasting DS algorithm. Before going any further, and since many of the tests are done in a conditional simulation mode, the procedure to generate the conditioning data is described. One possibility is to keep the values but to randomize their location so that the samples are taken with the correct histogram. This procedure preserves the histogram, but not the variogram (or higher order measures) since the coordinates of samples are randomized. In order to keep both histogram and variogram, we first produce a non-conditional realization from the training image (using MPS methods), from which samples are taken. In this study, the MPS simulation method which is applied is DS in its original mode (Mariethoz et al., 2009).

The input parameters of each realization may differ than the other one. In this regard the initial parameters used for each one is provided at the bottom of each figure that illustrates a realization using bunch DS. Table 2 shows the index used to notate each of the parameters. A comprehensive sensitivity

[2] http://scrf.stanford.edu/resources.software.dataset.php.

analysis of the various parameters governing the DS algorithm bas been carried on by Meerschman et al. (2013). In this paper we only test the sensitivity of the additional parameters related to the bunch-pasting method.

4.2.1. Simulation using TI *A*

20 realizations were produced conditioned to 100 conditioning data. One conditional realization is shown in Fig. 5a with the conditioning data displayed as circles. Although some of the conditioning data can cause artifacts in the conditional realization, the majority of the data points are surrounded with the nodes of identical value, showing satisfying conditioning. Moreover, the E-type map is provided in Fig. 5b, where pattern can be observed. The smoothing effect is typical of the E-type maps since it is the average of all realizations.

The variability is expected to be reduced in the vicinity of conditioning data. To verify this, hard data are taken throughout the training image except for a gap zone in the upper-right corner (highlighted by a red rectangle in Fig. 6a). As displayed in Fig. 6b, this area shows more uncertainty than the rest of the domain, with uncertainty increasing away from the data. Even with no hard data at some parts, the proposed algorithm can obtain realistic patterns. The uncertainty behavior is also shown on the interquartile range (the difference between the upper and lower quartiles) map of the realizations, which is shown in Fig. 6c. In the gap zone, the interquartile range is significantly more than on other areas, while at the conditional points it is reduced to zero (shown with blue pixels in Fig. 6c). This shows that, as expected, the variability is highest away from the data. It should be noted that due to the structures present in the training image, the zero variability areas are not limited to the conditioning locations, but also propagate around the conditioning points.

4.2.2. Simulation using TI *B*

Bunch pasting DS algorithm is applied on TI ***B*** and one corresponding realization is illustrated in Fig. 7a where the connected channels are reproduced. In this case, a bias is observed in the structures with the channels being exceedingly parallel, a tendency that has been observed with other methods, and that is typically related to the use of a unilateral path (e.g., Parra and Ortiz, 2011).

4.2.3. Simulation using TI *C*

Fig. 7b shows one realization simulated by bunch-pasting DS. The input parameters of the simulation are provided at the bottom of Fig. 7. The simulation looks like the input TI ***C***.

4.2.4. Simulation using TI *D*

The performance of bunch DS for a categorical TI is checked in this section. Fig. 8 displays a realization produced using TI ***D***, which is not stationary. However, the non-stationary of the TI is reproduced in the simulation, thanks to the large number of neighbors used (hence large data events that span over non-stationary areas) and to the presence conditioning data that guide the simulation.

4.2.5. Simulation conditional to an object

An important issue in the field of Geophysical modeling and underground fracture detection is to produce realizations which honor the presence of known geobodies. For example, geophysical attributes are used to visualize channels, mounds, clinoforms, etc., in order to improve reservoir modeling (Tahmasebi et al., 2012a,b; Arpat and Caers, 2007). A realization is shown in Fig. 9 and the E-type map is also provided. The object is well reproduced in the final average map. Not only the object, but also its areas are informed with much decreased uncertainty.

4.2.6. 3D simulation

In this section 3D simulation results for two TIs are provided.

*4.2.6.1. TI **E**.* In this TI all facies are interconnected. Fig. 10a shows the realization produced using the 3D simulation. The simulation size is $60 \times 37 \times 11$. The proportions of each facies existent in this TI and that of the realization are calculated and shown in Table 3. The values of proportions are normalized by the number of nodes in each TI and realization to ease the comparison. The facies proportion reproduction is good not perfect; however without using a correction factor like servosystem we cannot expect perfection.

*4.2.6.2. TI **F**.* The realization produced using this TI is shown in Fig. 10b. Three sections of the same realization are shown. The size of simulation grid is: $50 \times 50 \times 20$.

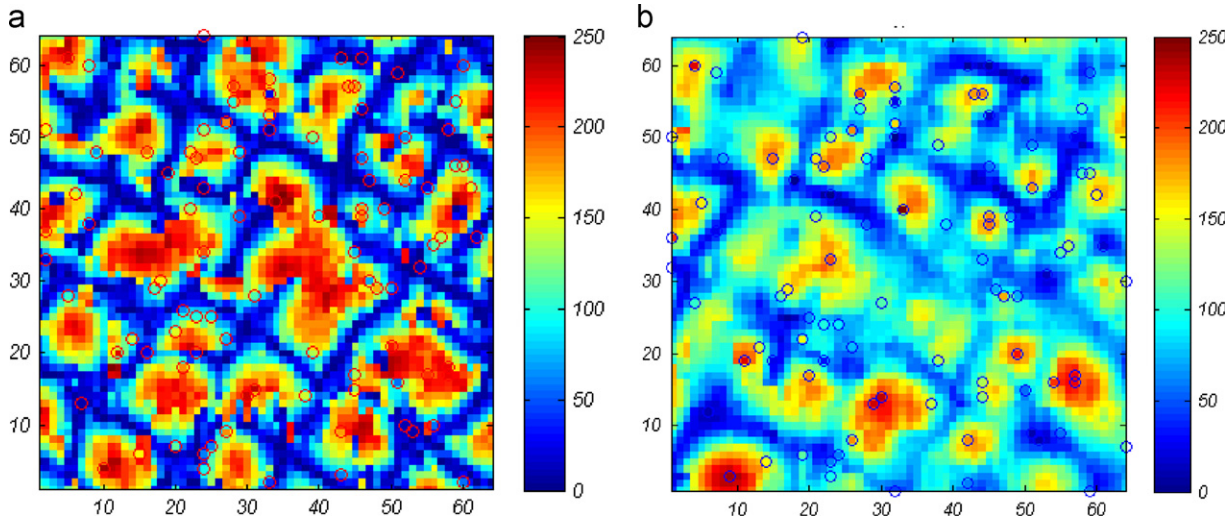

Fig. 5. (a) One conditional realization with TI ***A***. The position of conditioning data is shown on the realization with small circles. (b) The E-type of conditional simulations of the same TI with bunch-pasting DS. (B=25; R_x and R_y=12, 12; $\boldsymbol{F_r}$=0.8; δ=0.01; MaxNb=70; W=circular; path=unilateral).

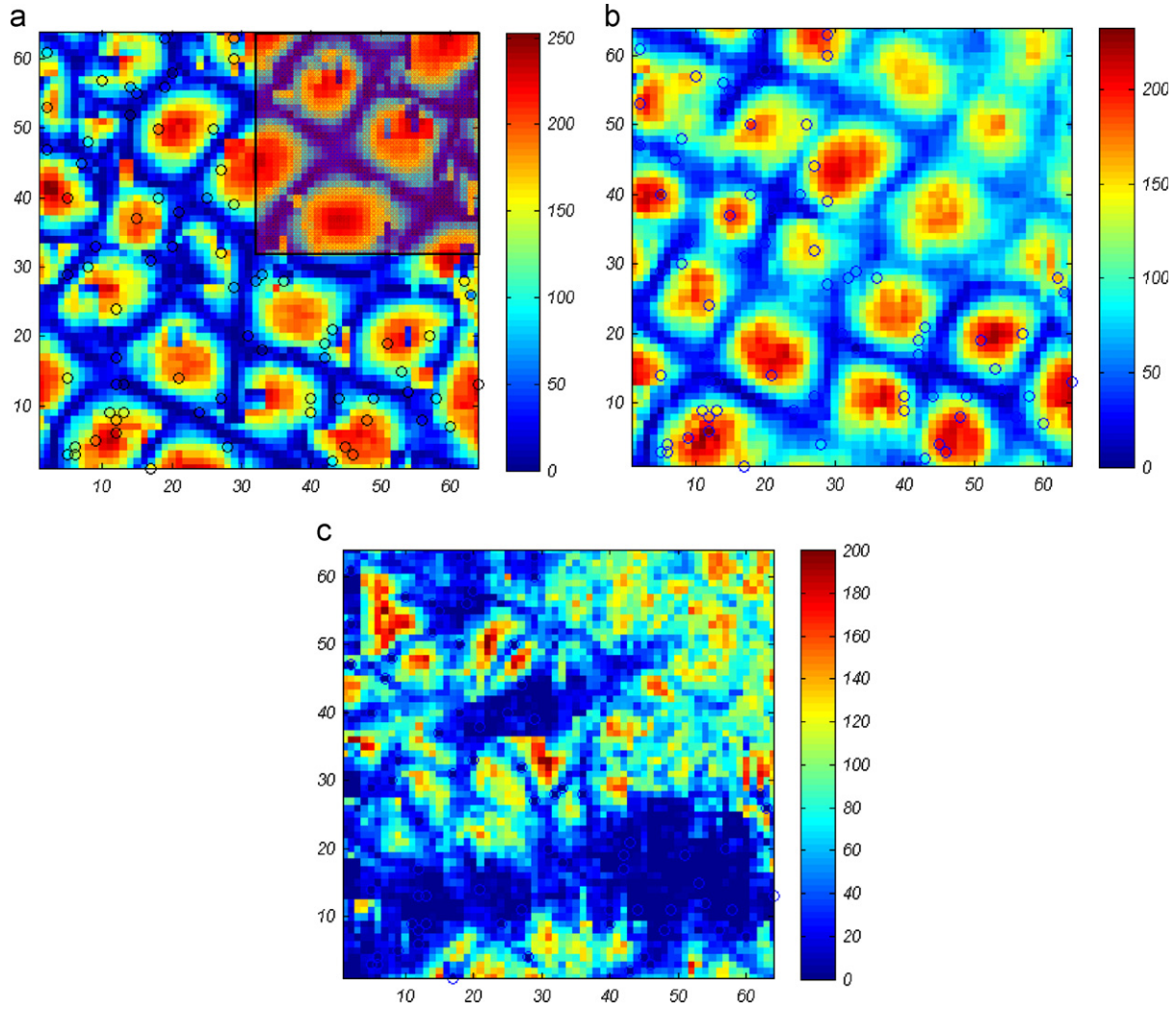

Fig. 6. Sparse data conditional simulation using TI ***A***: (a) Realization, (b) E-type map and (c) interquartile range map. The red square on sub-figure a shows the gap zone (B=25; R_x and R_y=15, 15; $\boldsymbol{F_r}$=0.8; δ=0.01; MaxNb=50; W=circular; path=unilateral). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

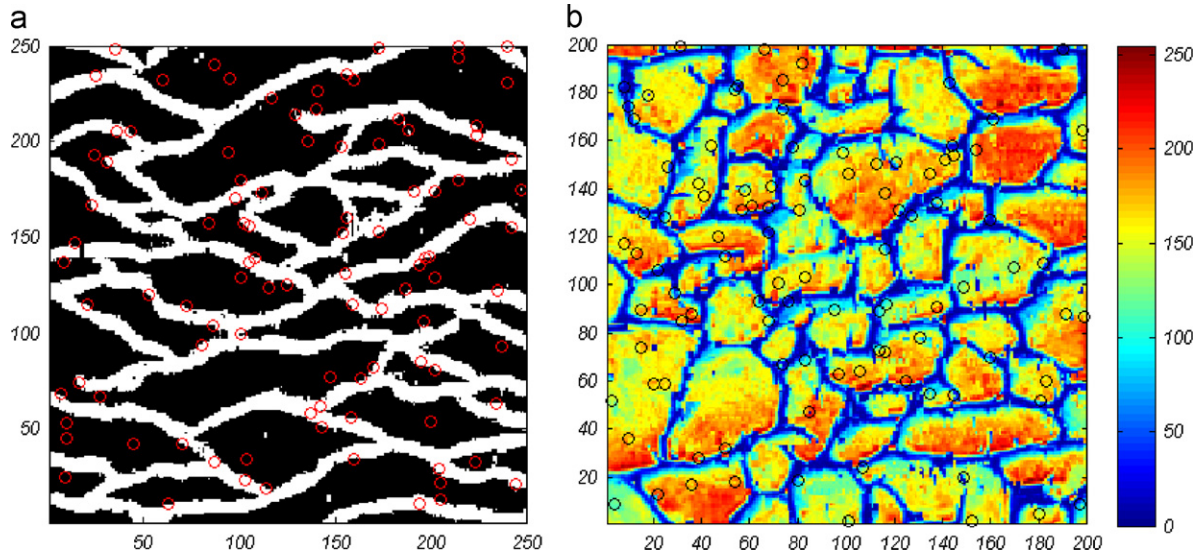

Fig. 7. (a) Realization using TI ***B*** (B=25; R_x and R_y=15, 15; $\boldsymbol{F_r}$=0.8; δ=0.01; MaxNb=150; W=Circular; Path=Unilateral). (b) Bunch pasting DS simulation on TI ***C***. (B=25; R_x and R_y=15, 15; $\boldsymbol{F_r}$=0.8; δ=0.01; MaxNb=150; W=Circular; Path=Unilateral).

5. Computational gains

The main advantage of using bunch-pasting is the CPU gain. Pixel-based and bunch-based DS simulations are compared in terms of CPU time, using the TI ***A***. In Fig. 11, the speed-up factor $sp(n)$ is computed as the ratio between the CPU time for $\boldsymbol{B}=1$ the CPU time for $\boldsymbol{B}=n$, $\mathrm{SP}(n)=\mathrm{CPU\_Time}\,(B=n)/\mathrm{CPU\_Time}(B=1)$ Fig. 11 shows that the computational gains scale close to linearly with the bunch size.

6. Validation of the results

For validation purposes, a series of criteria are evaluated to compare the statistical properties of the simulations and those of the TI. These include the reproduction of histogram, variogram and connectivity, considering the TI as reference. Since bunch size is one of the most important factors in the bunch-DS algorithm, the reproduction of statistics of different orders should be analyzed when the method takes different bunch sizes for simulation. In all cases four bunch sizes are considered: 1, 9, 25 and 49. These validation tests necessitate a large number of realizations. Since we use a Matlab non-optimized implementation, in order to keep computing times reasonable the size of all training images and simulation grids are reduced to 150×150. All other settings are kept identical.

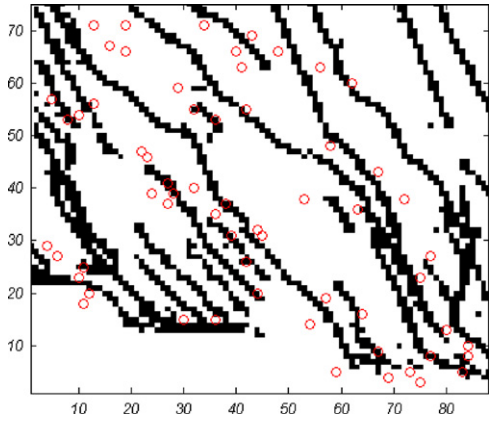

Fig. 8. Conditional simulation using TI ***D*** ($B=1$; R_x and $R_y=15$, 15; $\boldsymbol{F_r}=0.7$; $\delta=0.01$; MaxNb=50; W=circular; path=unilateral).

6.1. Histogram reproduction

Evaluating the reproduction of the histogram is done using continuous variable TIs ***A*** and ***C***. For each TI, 20 realizations are produced using bunch-pasting DS and their histograms are calculated and compared to the histogram of the TI in Fig. 12. In both cases the histogram of the TIs are reasonably well reproduced in the realizations and no difference is seen when bunch size varies from 1 to 49.

In the case of TI ***C***, a high variability is observed in the middle part of the histogram, corresponding to the mode. These values mainly correspond to the areas on the surface of the bricks, hence we attribute these fluctuations to the variable number of bricks in each realization, especially since we use small realizations (150×150 pixels).

6.2. Variogram reproduction

In this section the reproduction of variogram is checked out and its sensitivity to bunch size is analyzed and for TIs ***A*** and ***C***.

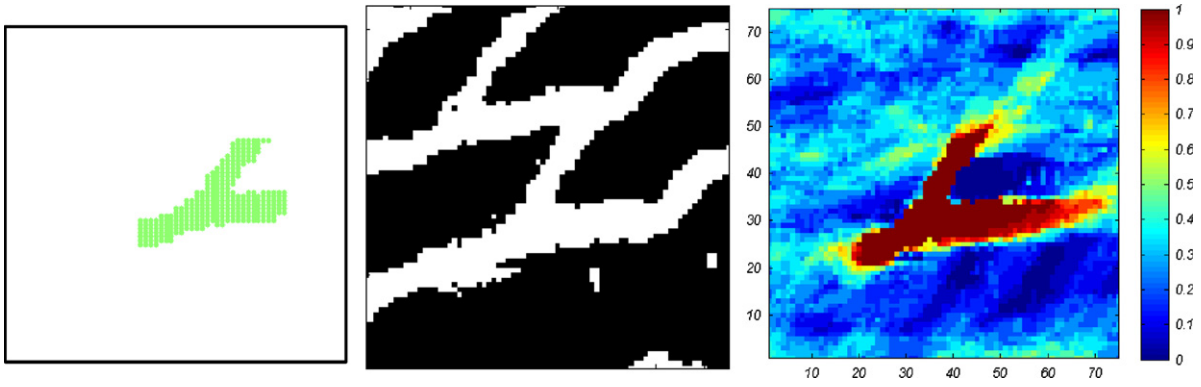

Fig. 9. Simulation conditional to an object. Left: Geobody, middle: realization, right: E-type map ($B=25$; R_x & $R_y=15$, 15; $\boldsymbol{F_r}=0.8$; $\delta=0.01$; MaxNb=150; W=circular; path=unilateral).

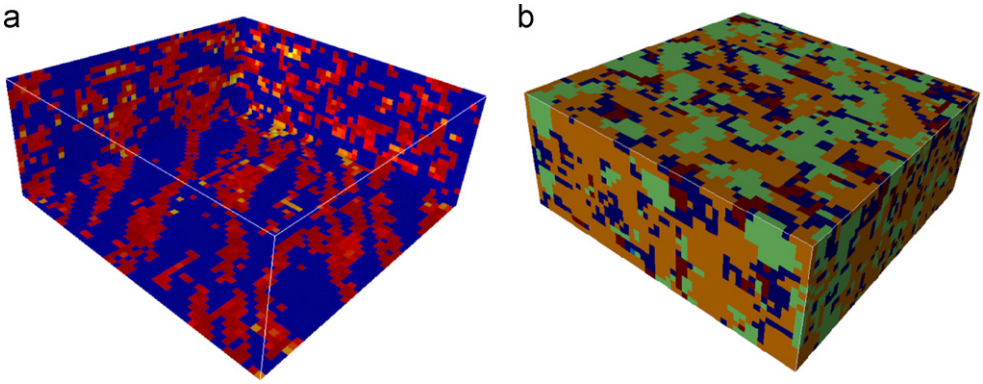

Fig. 10. (a) 3D simulation using TI ***E*** ($B=25$; R_x and $R_y=15$, 15; $\boldsymbol{F_r}=0.8$; $\delta=0.01$; MaxNb=150; W=circular; path=unilateral), (b) TI ***F*** realization using 3D bunch DS ($B=25$; R_x and $R_y=15$, 15; $\boldsymbol{F_r}=0.7$; $\delta=0.01$; MaxNb=150; W=Circular; Path=Unilateral).

Table 3
The reproduction of facies in the simulation made for TI ***E***.

Facies	Proportion ($\times 10^6$)	Facies	Proportion ($\times 10^6$)
Facies-0-TI	3.07	Facies-0-Sim	3.49
Facies-1-TI	0.13	Facies-1-Sim	0.01
Facies-2-TI	4.75	Facies-2-Sim	5.71
Facies-3-TI	6.25	Facies-3-Sim	9.18
Facies-4-TI	1.61	Facies-4-Sim	1.61

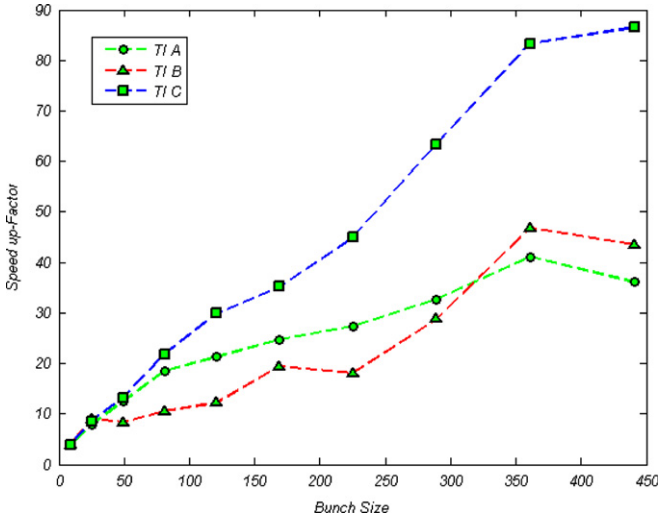

Fig. 11. Speed-up factor of bunch-pasting DS compared to traditional DS.

The omnidirectional variograms are calculated considering different values of bunch size. Fig. 13 shows the results of these tests. No profound changes in variogram reproduction occur when the factor B varies from 1 (pixel-based DS mode) to 49. The same can be found for TI ***C*** (Fig. 13) when the reproduction of omnidirectional variogram does not change so much when bunch takes different sizes, although the overall variogram reproduction is less good in this case because more complex structures are considered. This illustrates a positive feature of the proposed methodology since by increasing bunch size the algorithm speeds up more and more while preserving its capability at reasonably reproducing two-point statistics of the reference training image. Similarly as for the histogram tests, all training images and simulation grids are reduced to 150×150 for tests on variogram reproduction.

6.3. Connectivity reproduction

Reproduction of connectivity and the distribution of geobodies area are evaluated using TIs ***B*** and ***D***. For the selected TIs, the connectivity of facies is of paramount importance: in the case of TI ***B*** the channels are conduits for fluid flow (hydrocarbon) and the connectivity of dykes is important since they can act as barriers to hydrothermal systems (mineral deposits). Connectivity functions used in this study are defined as the probability of 2 nodes separated by a lag along a certain direction to be connected (Renard and Allard, 2011). Here TI ***B*** and the corresponding simulation grid are reduced to a size of 100×100. Fig. 14 illustrates the connectivity functions for both realizations and training image. A relatively good match can be found between the connectivity factors of TI and realizations of TI ***B*** for different bunch sizes, especially for larger bunch sizes.

TI ***D*** (Fig. 4d) used in this study is considered for comparison based on geobodies area (De Iaco and Maggio, 2010). It should be noted that in this TI geobodies are considered as dykes. The comparison criterion is based on the cumulative area of geobodies in the realizations, which are normalized to that of the training image. Fig. 15 shows the histogram of normalized areas for different bunch sizes. The closer this value is to unity, the more similar the simulated maps are to the training image as it is done in De Iaco and Maggio, 2010. This figure suggests a close similarity between realizations and the TI which slightly falls for larger bunch sizes (25 and 49).

7. Sensitivity tests

The sensitivity of the algorithm to some of the input parameters is tested in this section. The four most important parameters are checked which are simulation path, bunch shape and bunch size. The sensitivity of these different parameters is evaluated using the different training images presented above, each time choosing the training image(s) that are most suited to illustrate the discussion.

7.1. Sensitivity to simulation path

In general the unilateral path produces better patterns which are due to the Markovian character of the simulation Daly (2004) and Pickard (1980). This fact is shown in this study and illustrated in (Fig. 16). The drawbacks are difficulties for conditioning, especially when large objects are present, and the arbitrary decision of a path direction. Most methods can be used with unilateral or random paths. One way to solve the problem of conditioning in unilateral simulation is to use lag vectors and therefore consider neighbors that can be far away (Parra and Ortiz, 2011). This includes conditioning data that are still far, and that the algorithm sees coming in advance. By using flexible configurations for data event which is defined by lag vectors, the algorithm "looks ahead" for conditioning data (both hard and previously simulated data), making it possible to condition the simulation early enough.

7.2. Sensitivity to bunch shape

Three bunch shapes are considered: square, circle and disc, and for each of which one realization is produced. In this case we have used TI ***D***. In Fig. 17, three realizations are shown based on the bunch shape shown in the upper part of the figure. There is not a big difference between the circle and rectangle shapes. The disc shape is associated with small-scale noise because it tends to ignore short-range patterns, therefore breaking the continuity of the dykes. The conclusion of this example is that it is better to use simple bunch shapes to avoid intractable algorithm behavior causing artifacts. It should be noted that except for the tests done in the current section, all realizations are produced using a square shape for the bunch.

Similar as in the original DS method, the number of conditioning nodes considered n as well as the threshold t should be adjusted such that an appropriate trade-off is found between simulation time and patterns reproduction. For a complete sensitivity analysis of the DS parameters, we refer the reader to Meerschman et al. (2013).

7.3. Sensitivity to bunch size

The main parameter governing the method is the bunch size. Although this parameter has already been investigated in Section

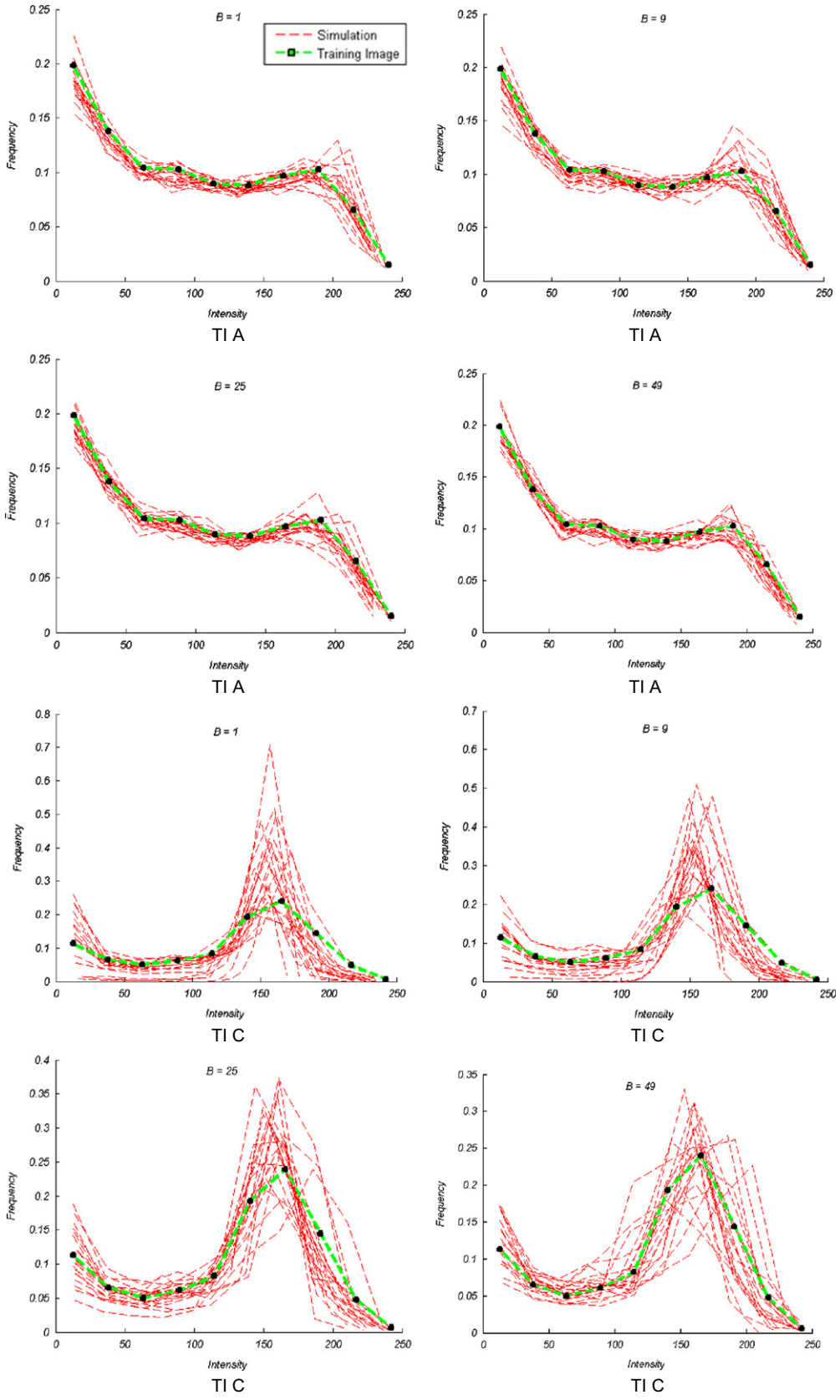

Fig. 12. Histogram reproduction and its sensitivity to bunch size for TIs ***A*** and ***C***. The green line corresponds to the pdf of the training image and the red lines represent the pdfs of the 20 realizations. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

6, here we further investigate its sensitivity in order to understand its effect on the simulated spatial patterns. The TIs ***A***, ***B*** and ***C*** are used for simulations considering bunch sizes, of n^2, with $n=1,\ldots,21$. The resulting sizes are 1 (which corresponds with the pixel-based simulation mode), 9, 25, etc. 12 realizations are obtained for each TI and each bunch size.

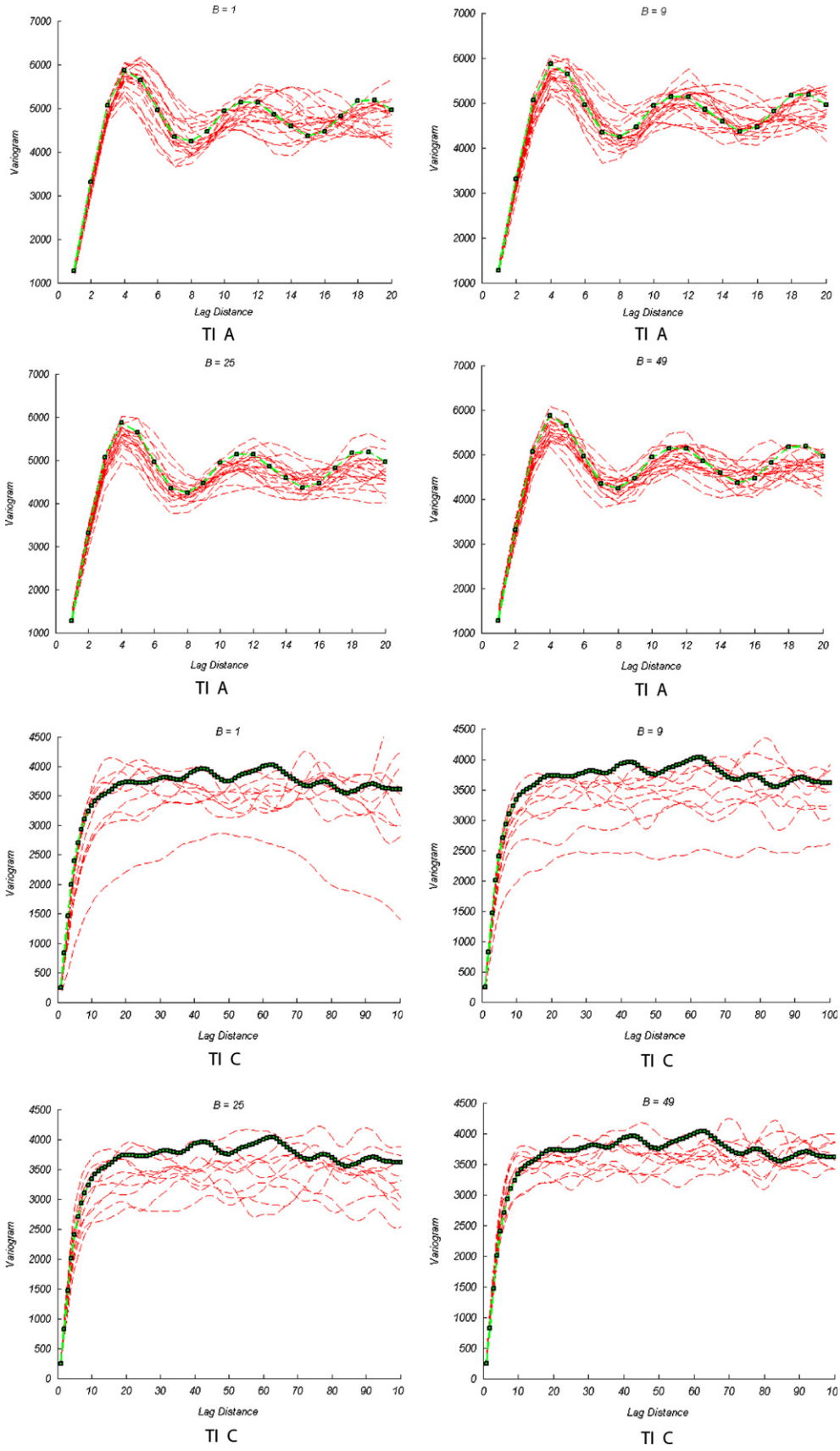

Fig. 13. Variogram reproduction and its sensitivity to bunch size for TIs ***A*** and ***C***.

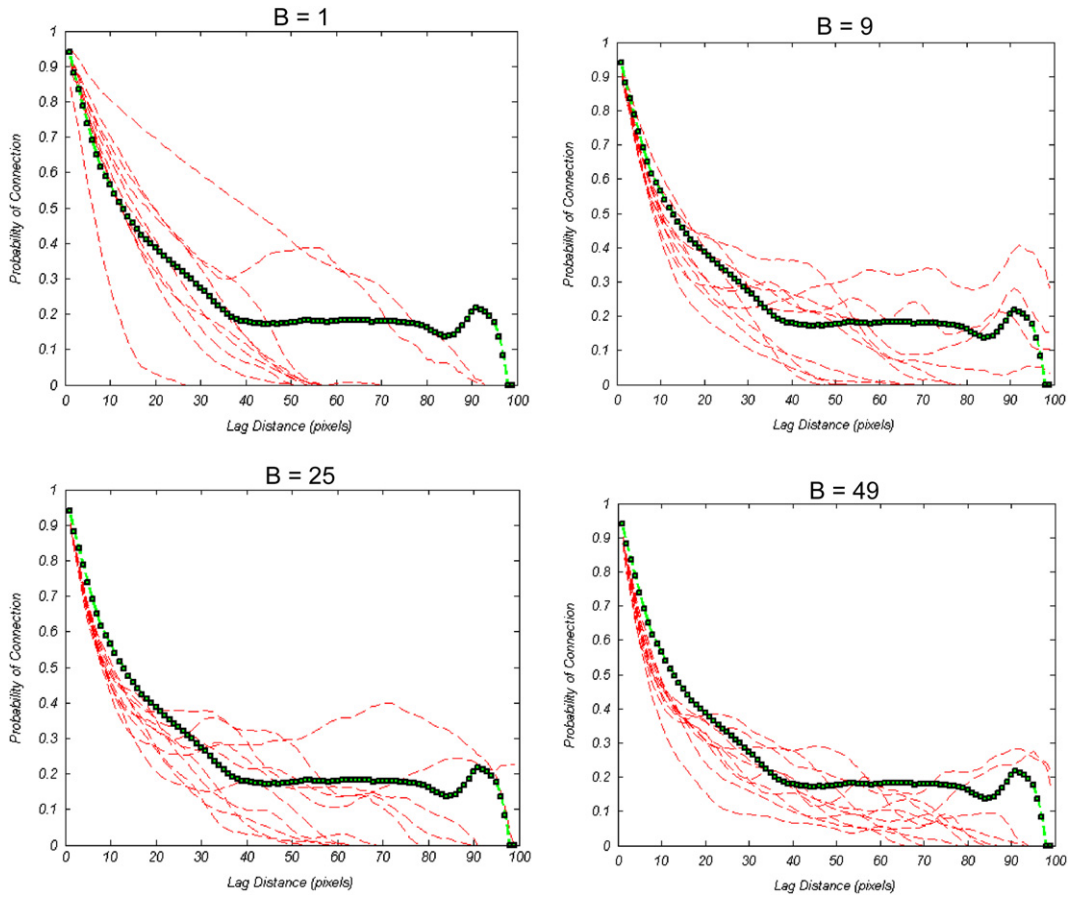

Fig. 14. Connectivity reproduction in direction X and its sensitivity to B (bunch size) for TI ***B***.

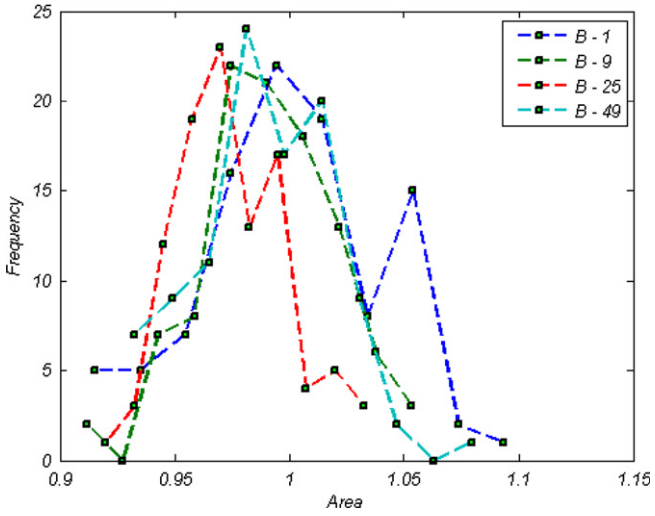

Fig. 15. Geobodies area distribution for the realization produced using TI ***D*** and different values of B (bunch size).

The first lag of the variogram (assimilated to the nugget effect) is used to represent small-scale noise. Fig. 18 shows the nugget effect computed on the realizations and the training image (nugget effect of the TI shown as a horizontal red line). Around bunch sizes 9 and 25, the nugget effect increases suddenly (the trend slightly differs for two other TIs), indicating noise which is due to pasting bunches of nodes which are not compatible with the previously pasted bunches, and resulting in discontinuities. This effect can be seen in Fig. 19. Based on this observation and for the tested training images (both categorical and continuous cases), we found that in general the optimal bunch size is between 9 and 25. By increasing the bunch size to a point where it is larger than the dominant size of objects and patterns existent in the training image, it is likely that complete objects will be pasted into the simulation grid. This can be observed in Fig. 19 (the figures corresponding to $B=121$). This may result in some incompatibilities in the realizations. In this regard, the method used by Honarkhah and Caers (2010) to determine the most optimum size of search window (to scan the training image), can be used but for determining the bunch size.

Moreover by increasing the bunch size, the size of N_x is increased and accordingly it is less likely to find the N_y that meets the distance threshold. This is why the quality of the realizations decreases for larger bunch sizes.

8. Summary and conclusions

Multiple-point geostatistical simulation has been a topic of interest in the last two decades. The existing algorithms are either pixel-based or patch-based. A new hybrid methodology is introduced that consists in pasting a bunch of nodes rather than one at a time. Our method is intermediate between pixel-based and bunch-based since the bunch size is flexible.

The idea proposed in this paper is implemented in the DS method which allows such flexibility due to the absence of

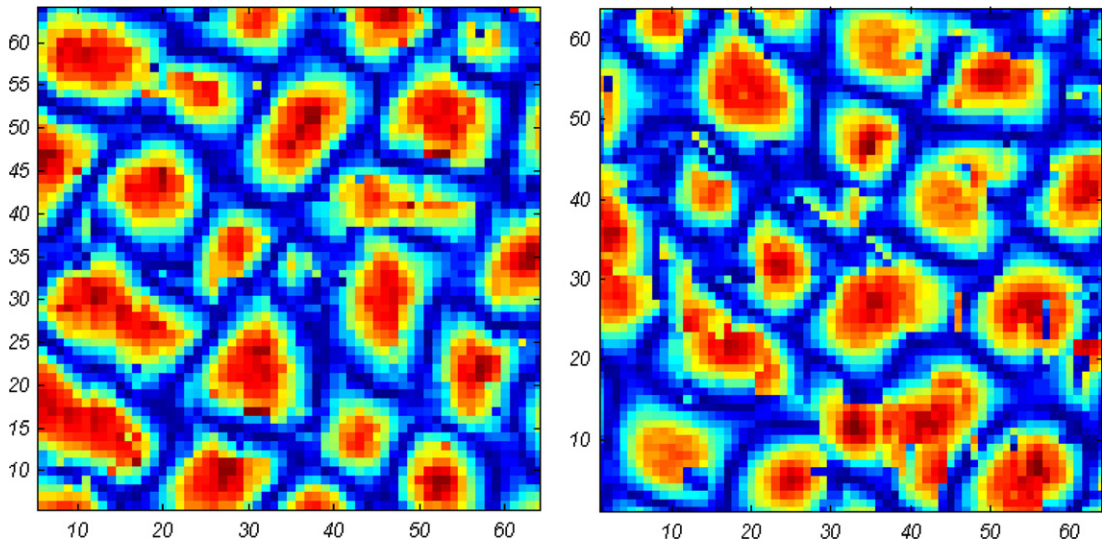

Fig. 16. Realizations produced by Left: Unilateral simulation path and right: random simulation path ($B=25$; R_x and $R_y=8, 8$; $\boldsymbol{F_r}=0.3$; $\delta=0.05$; MaxNb=30; W=square).

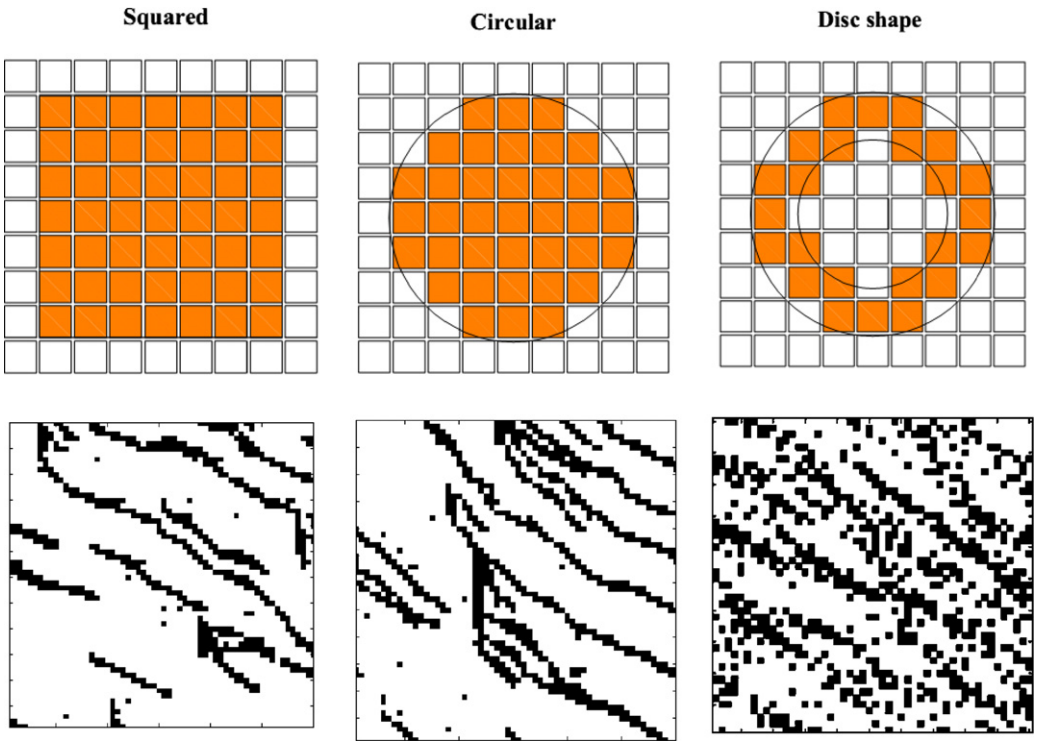

Fig. 17. Sensitivity to the window shape of the bunch: Above: bunch shapes. Below: corresponding realizations ($B=25$; R_x & $R_y=8, 8$; $\boldsymbol{F_r}=0.3$; $\delta=0.05$; MaxNb=30; path=unilateral).

storage. The performance of the proposed method is checked with a series of TIs of both categorical and continuous variables. In all cases it can reasonably well reproduce the patterns that exist in the TI, although there is a general trend of decreased quality with increased bunch size.

The most important contribution of bunch-pasting simulation is the computational gain it can offer with reasonable compromise regarding patterns reproduction or honoring conditioning data. Our tests have shown that the CPU time generally scales down in a quasi-linear fashion when the bunch size is increased. The significant reduction in CPU cost opens avenues for applications involving very large simulations.

However, one may be cautious since bigger bunches just copy and paste the patterns existent in the TI. For this reason, very large bunches may result in artifacts in the simulations and lack variability. Based on the sensitivity analysis carried out in the paper, a bunch size between 9 and 25 is recommended which speeds up the DS by at least a factor of 10. Implementation of the bunch DS in C or C++, including parallel computing (Tahmasebi et al., 2012a,b; Mariethoz, 2010), should yield much better absolute computational performance.

Most pixel-based methods are based on computing the probability of the value at the central node conditional to the neighbors. With bunch-pasting, this probability cannot be

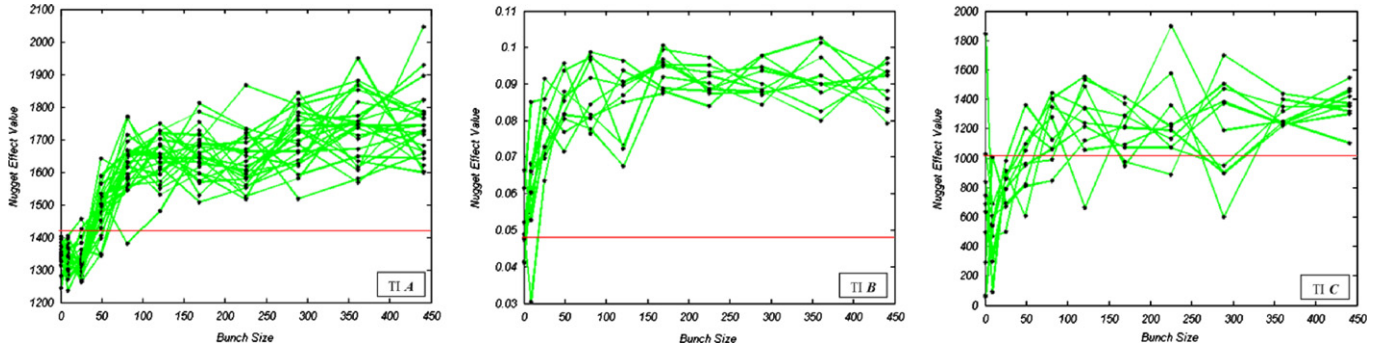

Fig. 18. Nugget effect structure value of realizations which are produced by different bunch sizes TIs ***A***, ***B*** and ***C***. Number of realizations for TI ***A***, ***B*** and ***C*** is 25, 12 and 12, respectively. Red line represents the nugget effect of each TI. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

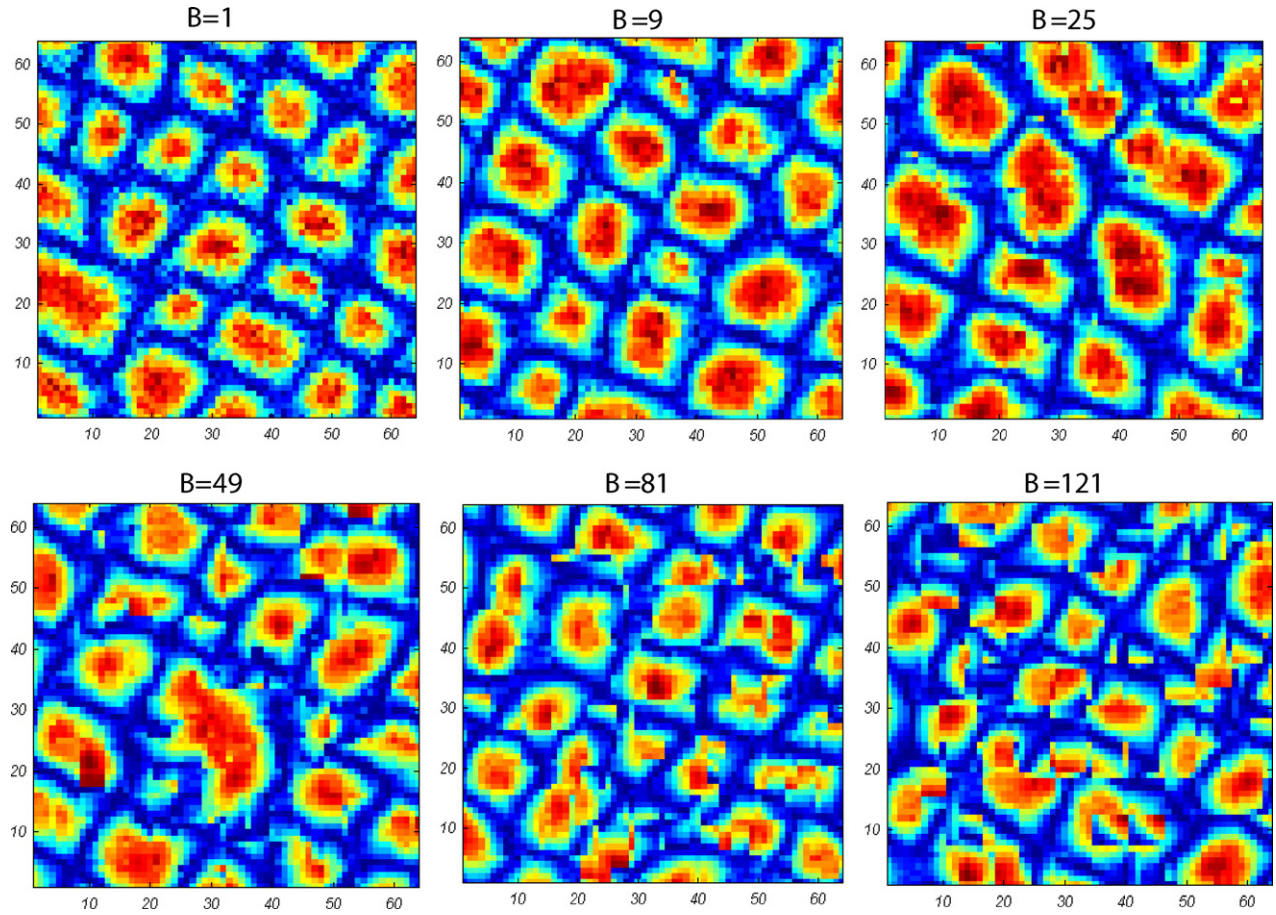

Fig. 19. Sensitivity of simulation to bunch size (R_x and R_y=8, 8; $\mathbf{F_r}$=0.3; δ=0.05; MaxNb=30; W=square; path=unilateral).

computed since it would be the joint probability of all pixels in the bunch. Therefore the bunch-pasting principle is specifically applicable with DS because no probability is computed. Instead a location is sampled in the TI, making it possible to extract a bunch around this location. This principle could not be used with methods storing probabilities, such as SNESIM or IMPALA.

A Matlab implementation of the method presented in this paper is available for the readers.

Acknowledgment

The authors would like to express their thanks to Sébastien Strebelle from Chevron Texaco for providing with TI ***E*** to be used in 3D simulation tests. We also thank anonymous reviewers whose comments and suggestions improved the paper.

References

Arpat, B., Caers, J., 2007. Conditional simulations with patterns. Mathematical Geosciences 39 (2), 177–203.

Bianchi, M., Zheng, C., Wilson, C., Tick, G.R., Liu, G., Gorelick, S.M., 2011. Spatial connectivity in a highly heterogeneous aquifer: from cores to preferential flow paths. Water Resources Research 47 (5), W05524, http://dx.doi.org/10.1029/2009WR008966.

Chiles, J.P., Delfiner, P., 1999. Geostatistics: Modeling Spatial Uncertainty. Wiley, New York, NY, 695 pp.

Daly, C., 2004. Higher Order Models Using Entropy, Markov Random Fields and Sequential Simulation. Kluwer Academic Publisher, Banff, Alberta, Canada 1139 pp.

De Iaco, S., Maggio, S., 2010. Validation techniques for geological patterns simulations based on variogram and multiple-point statistics. Mathematical Geosciences 43 (4), 483–500, http://dx.doi.org/10.1007/s11004-011-9326-9.

Deutsch, C.V., Wang, L., 1996. Hierarchical object-based geostatistical modeling of fluvial reservoirs. In: 1996 SPE Annual Technical Conference and Exhibition, Denver, USA, http://dx.doi.org/10.2118/36514-MS.

Goovaerts, P., 1997. Geostatistics for Natural Resources Evaluation. Oxford University Press, New York, 496 pp.

Guardiano, F., Srivastava, M., 1993. Multivariate geostatistics: beyond bivariate moments. In: Soares, A. (Ed.), Geostatistics-Troia. Kluwer Academic Publications, Dordrecht, pp. 133–144, http://dx.doi.org/10.1007/978-94-011-1739-5_12.

Haldorsen, H.H., Lake, L.W., 1984. A new approach to shale management in field-scale models. Society of Petroleum Engineers Journal 24 (8), 447–452.

Holden, L., Hauge, R., Skare, O., Skorstad, A., 1998. Modeling of fluvial reservoirs with object models. Mathematical Geology 30 (5), 473–496.

Honarkhah, M., Caers, J., 2010. Stochastic simulation of patterns using distance-based pattern modeling. Mathematical Geosciences 42 (5), 487–517.

Isaaks, E., 1990. The Application of Monte Carlo Methods to the Analysis of Spatially Correlated Data. Ph.D. Dissertation. Stanford University, California, USA, 213 pp.

Journel, A.G., 1983. Nonparametric estimation of spatial distributions. Mathematical Geology 15 (3), 445–468.

Kleingeld, W.J., Lantuejoul, C., Prins, C.F., Thurston, M.L., 1997. The conditional simulation of a Cox process with application to deposits with discrete particles. In: Baafi, E.Y., Schofield, N.A. (Eds.), Geostatistics Wollongong, 96. Kluwer Academic, Dordrecht, pp. 683–694.

Lantuejoul, C., 2001. Geostatistical Simulation: Models and Algorithms. Springer, 256 pp.

Mariethoz, G., 2010. A general parallelization strategy for random path based geostatistical simulation methods. Computers and Geosciences 36 (7), 953–958, http://dx.doi.org/10.1016/j.cageo.2009.11.001.

Mariethoz, G., Renard, P., Froidevaux, R., 2009. Integrating collocated auxiliary parameters in geostatistical simulations using joint probability distributions and probability aggregation. Water Resources Research 45 (8), http://dx.doi.org/10.1029/2008WR007408.

Mariethoz, G., Renard, P., 2010. Reconstruction of incomplete data sets or images using direct sampling. Mathematical Geosciences 42 (3), 245–268, http://dx.doi.org/10.1007/s11004-010-9270-0.

Mariethoz, G., Renard, P., Straubhaar, J., 2010. Direct sampling method to perform multiple point geostatistical simulations. Water Resources Research 46 (W11536), http://dx.doi.org/10.1029/2008WR007621.

Mariethoz, G., Kelly, B.F.J., 2011. Modeling complex geological structures with elementary training images and transform-invariant distances. Water Resources Research 47 (W07527), http://dx.doi.org/10.1029/2011WR010412.

Meerschman, E., Pirot, G., Mariethoz, G., Straubhaar, J., Van Merivenne, M., Renard, P., 2013. A practical guide to performing multiple-point geostatistical simulations with the direct sampling algorithm. Computers and Geosciences, http://dx.doi.org/10.1016/j.cageo.2012.09.019.

Parra, A., Ortiz, J.M., 2011. Adapting a texture synthesis algorithm for conditional multiple point geostatistical simulation. Stochastic Environmental Research and Risk Assessment 25 (8), 1101–1111.

Pickard, D., 1980. Unilateral Markov fields. Advances in Applied Probability 12, 655–671.

Renard, P., Allard, D., 2011. Connectivity metrics for subsurface flow and transport. Advances in Water Resources, http://dx.doi.org/10.1016/j.advwatres.2011.12.001.

Rezaee, H., 2012. Multiple-Point Geostatistics: Theory and Application in Dyke Modeling; a Case Study of Sungun Porphyry Copper Deposit, Unpublished M.Sc. Thesis. University of Tehran, Tehran, Iran, 113 pp.

Shannon, C.E., 1948. A mathematical theory of communication. The Bell System Technical Journal 27, 379–423.

Stoyan, D., Kendall, W.S., Mecke, J., 1987. Stochastic Geometry and Its Applications. Wiley, New York 456 pp.

Straubhaar, J., Renard, P., Mariethoz, G., Froidevaux, R., Besson, O., 2011. An improved parallel multiple-point algorithm using a list approach. Mathematical Geosciences 43 (3), 305–328, http://dx.doi.org/10.1007/s11004-011-9328-7.

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., 2012a. Multiple-point geostatistical modeling based on the cross-correlation functions. Computational Geosciences 16 (3), 779–797, http://dx.doi.org/10.1007/s10596-012-9287-1.

Tahmasebi, P., Sahimi, M., Mariethoz, G., Hezarkhani, A., 2012b. Accelerating geostatistical simulations using graphical processing units. Computers and Geosciences 46, 51–59, http://dx.doi.org/10.1016/j.cageo.2012.03.028.

Wackernagel, H., 2003. Multivariate Geostatistics: an Introduction with Applications, third ed. Springer-Verlag, ed, 256 pp.

Wu, J., Zhang, T., Journel, A.G., 2008. A fast FILTERSIM simulation with score-based distance. Mathematical Geosciences 40 (7), 773–788.

Wei, L., Levoy, M., 2000. Fast Texture Synthesis Using Tree-Structured Vector Quantization, In: SIGGRAPH '00: 27th Annual Conference on Computer Graphics and Interactive Techniques. ACM Press/Addison-Wesley, New Orleans.

Zhang, T., Switzer, P., Journel, A.G., 2006. Filter-based classification of training image patterns for spatial simulation. Mathematical Geology 38 (1), 63–80.

Zinn, B., Harvey, C., 2003. When good statistical models of aquifer heterogeneity go bad: a comparison of flow, dispersion, and mass transfer in connected and multivariate Gaussian hydraulic conductivity fields. Water Resources Research 39 (3), WR001146, http://dx.doi.org/10.1029/2001WR001146.