Contents lists available at ScienceDirect

Computers & Geosciences

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

Research paper

PCTO-SIM: Multiple-point geostatistical modeling using parallel conditional texture optimization

Mohammadreza Pourfard[a,*], Mohammad J. Abdollahifard[b], Karim Faez[a], Sayed Ahmad Motamedi[a], Tahmineh Hosseinian[c]

[a] *Electrical Engineering Department, Amirkabir University of Technology, 424 Hafez Ave, Tehran, Iran*
[b] *Electrical Engineering Department, Tafresh University,Tafresh, Iran*
[c] *Basic Science Department, Shariaty Faculty Technical and Vocational University, Tehran, Iran*

ARTICLE INFO

Keywords:
Big data
Conditional multiple-point simulations
Constrained optimization
Constrained texture synthesis
Energy minimization
GPU
Parallel processing
Self-enrichment

ABSTRACT

Multiple-point Geostatistics is a well-known general statistical framework by which complex geological phenomena have been modeled efficiently. Pixel-based and patch-based are two major categories of these methods. In this paper, the optimization-based category is used which has a dual concept in texture synthesis as texture optimization. Our extended version of texture optimization uses the energy concept to model geological phenomena. While honoring the hard point, the minimization of our proposed cost function forces simulation grid pixels to be as similar as possible to training images. Our algorithm has a self-enrichment capability and creates a richer training database from a sparser one through mixing the information of all surrounding patches of the simulation nodes. Therefore, it preserves pattern continuity in both continuous and categorical variables very well. It also shows a fuzzy result in its every realization similar to the expected result of multi realizations of other statistical models. While the main core of most previous Multiple-point Geostatistics methods is sequential, the parallel main core of our algorithm enabled it to use GPU efficiently to reduce the CPU time. One new validation method for MPS has also been proposed in this paper.

1. Introduction

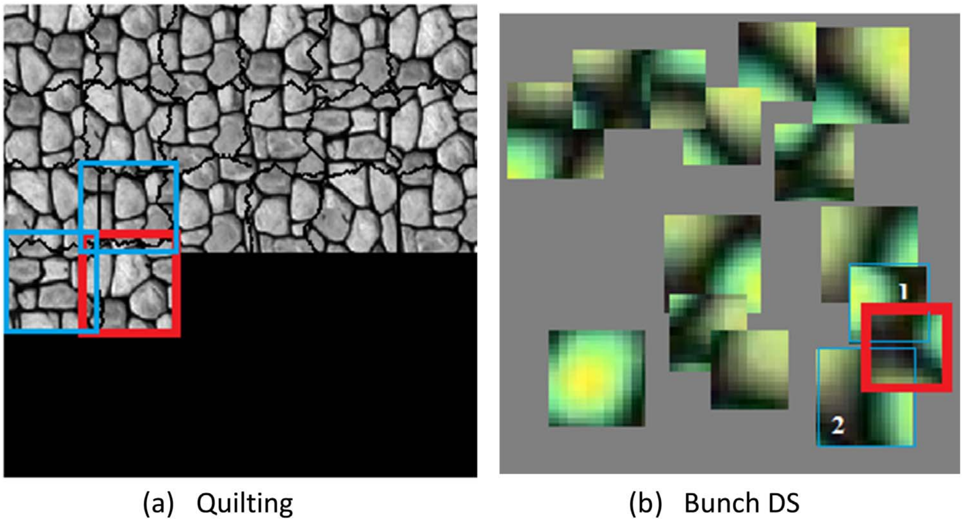
Multiple-point statistics (MPS) simulation methods have recently attracted significant attention (Pyrcz and Deutsch, 2014; Comunian et al., 2011; He et al., 2013; Cressie and Wikle, 2011; Caers, 2011; Sahimi, 2011; Hu and Chugunova, 2008). MPS methods provide a direct way for incorporating the geologists' knowledge into the simulation process through a training image (TI) (Yoram, 2003; Zhang, 2001; Wackernagel, 2003; Goovaerts, 1997; Ringrose and Bentley, 2015). MPS methods are capable of handling complicated structures more appropriately as compared to two-point statistics methods (Guardiano and Srivastava, 1993; Remy et al., 2009; Chiles and Delfiner, 2012; Hashemi et al., 2014). Object-based methods are also capable of handling complex structures through defining basic shapes representing geobodies and placing them in the model domain based on a probability model (Keogh et al., 2007; Michael et al., 2010; Khodabakhshi and Jafarpour, 2013). However, it is difficult to incorporate hard conditioning data in object-based methods.

Although MPS was initially developed for reservoir modeling, it finds broad applications in different areas including climate prediction (Jha et al., 2013a), physics of porous media (Tahmasebi and Sahimi, 2013), remote sensing (Boucher et al., 2008; Ge and Bai, 2011), seismic inversion (González et al., 2007), geomorphology (Vannametee et al., 2014), and some other hydrology (Goovaerts, 2000; Park et al., 2013; Kessler et al., 2013; Huysmans and Dassargues, 2009; Renard, 2007) and hydrogeological applications (Feyen and Caers, 2006; dell'Arciprete et al., 2012; Jha et al., 2013b; Le Coz et al., 2011). MPS have also been used in other fields like the oil industry (Aitokhuehi and Durlofsky, 2005; Caers et al., 2003; Strebelle et al., 2003; Hoffman and Caers, 2007; Tamayo-Mas et al., 2016) and medical imaging (Pham, 2012).

MPS simulation usually proceeds by scanning the simulation grid (SG) sequentially in a specific order. A data-event is extracted around each grid point. Next, the TI patterns are searched to find a match. Then, the matching pattern from TI is transferred to the SG and the process continues until the SG is completed. In pixel-based MPS methods, one pixel is pasted into the SG in each step (Mariethoz et al., 2010; Strebelle, 2002; Straubhaar et al., 2011; Strebelle and

[*] Corresponding author.
E-mail addresses: pourfardm@aut.ac.ir (M. Pourfard), mj.abdollahi@tafreshu.ac.ir (M.J. Abdollahifard), kfaez@aut.ac.ir (K. Faez), motamedi@aut.ac.ir (S.A. Motamedi), hosseiniant@gmail.com (T. Hosseinian).

http://dx.doi.org/10.1016/j.cageo.2016.12.012
Received 12 December 2015; Received in revised form 19 December 2016; Accepted 21 December 2016
Available online 23 December 2016



(a) Quilting (b) Bunch DS

Fig. 1. Image quilting (Mahmud et al., 2014) and MS_CCSIM (Tahmasebi et al., 2014) method have a serial structure. Every new patch is simulated based on its previous top and left patches. The Bunch DS (Rezaee et al., 2013) method also has a serial structure. Judgment about a new patch depends on the places and numbers of previously synthesized patches (E.g. patches 1, 2 in (b)). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Cavelius, 2014). Patch-based methods work much faster by pasting a patch of data at a time (Abdollahifard and Faez, 2012, 2013; Arpat and Caers, 2007; Mahmud et al., 2014; Abdollahifard et al., 2016; Tahmasebi et al., 2012a; Rezaee et al., 2013). The SG is usually scanned in a random or raster order (Zhang et al., 2006; Wu et al., 2008; Honarkhah and Caers, 2010; Meerschman et al., 2013).

Error accumulation and propagation is one of the most fundamental problems in sequential simulation. Due to the limited extent of data-events, data could be synthesized inconsistent with far values. Such inconsistencies propagate in the SG and result in undesirable discontinuities and artifacts. In these methods, simulating a new patch (or pixel) is dependent on previous simulations. The place and number of previously synthesized patches have an influence on the newly synthesized patch (Fig. 1). Although such remedies (Mahmud et al., 2014; Abdollahifard, 2016) have been proposed to decrease the inconsistencies in the overlap region of subsequent data-events, they still suffer from a fundamental limitation of sequential simulation namely preventing previously synthesized values from being refined.

MPS methods have dual (twin) methods (Wei et al., 2009; Mariethoz and Lefebvre, 2014) in image processing and computer graphics (Efros and Leung, 1999; Wei and Levoy, 2000; Efros and Freeman, 2001; Kwatra et al., 2003; Ramanarayanan and Bala, 2007; Kopf et al., 2007; Xu et al., 2009). This is while another major category of the texture synthesis algorithms (Shen et al., 2010) exists which has not yet been widely used in the MPS context (e.g (Peredo and Ortiz, 2011; Abdollahifard and Ahmadi, 2016).). Optimization-based texture synthesis methods developed in the computer graphics community allow iterative refinement of previously synthesized values in order to achieve as less inconsistencies as possible (Wei et al., 2009). Starting from a random image, such methods proceed by finding a match from the TI for every patch in the SG and attempting to update the patch in a way consistent with the training pattern.

Concurrent to the submission of this manuscript, Yang et al., 2016 have used the optimization-based techniques based on Kwatra et al. (2005). The original version of Kwatra et al. (2005) has not addressed handling 3D and hard data. Kopf et al. (2007) have also extended it for 3D unconditional simulation but it is limited to a 2D exemplar for its TI. One of the main problems of the optimization-based methods is their computational cost due to their widespread search in TI. Yang et al. (2016) have used the patch-match idea of Barnes et al. (2009) to limit the search space of Kwatra et al. (2005). However, this acceleration is obtained by the cost of accuracy reduction. It is possible to trap in the local minimum in the search space due to the approximate nearest neighborhood (ANN) strategy. Both Yang et al. (2016) and Barnes et al. (2009) have also misunderstood the concept of EM (Expectation Maximization) and defined Expectation instead of Maximization and vice versa.

On the other hand, parallel techniques have been used in MPS simulation (Straubhaar et al., 2011; Ingram and Cornford, 2010; Mariethoz, 2010) to increase the speed. Tahmasebi et al. (2012b) have also proposed the usage of GPU as an advanced parallel hardware. Huang et al. (2016) have also used GPU for parallel implementation of SNESIM. However, parallel techniques could be used efficiently if the main core (the portion of the algorithm which has the most computational cost) has a parallel structure. This is while the main core of the pixel-based and patch-based MPS simulation methods are serial (sequential). Therefore, as the main core of our algorithm has a totally parallel structure, it has increased the speed of Kwatra et al. (2005) without decreasing its accuracy like the patch-match strategy of Yang et al. (2016). Concurrent to the submission of this manuscript, Kalantari and Abdollahifard (2016) have used sparse optimization-based techniques (Ashikhmin, 2001). Their method has a different simulation strategy than ours and has the same speed problem of Kwatra et al. (2005).

In this paper, an optimization-based simulation method based on Kwatra et al. (2005) is proposed which updates the image patches in parallel. Avoiding the sequential nature of traditional algorithms, the newly developed method reduces the inconsistencies and artifacts in produced realizations. Our strategy for honoring hard data is different from Yang et al. (2016). In fact, Yang et al. (2016) consider hard data only once in every cell of realization locally, while our method considers them twice both locally and globally. Our categorical simulation strategy is also based on quantization which is different from k-means++ clustering of Yang et al. (2016). We have also used a pseudo inverse of a sparse matrix of (Peyre, 2009) to accelerate the calculation of averaging in the expectation phase of Kwatra et al. (2005) just in one operation. This feature is important due to some advances in calculating the inverse of the sparse matrix in numerical calculations (peyre, 2007). This feature allows our algorithm to be used in any high dimension (including 3D and time-varying 3D grids).

Our proposed method also has a few innovations and distinguishing features. Our algorithm has the fastest exhaustive search in the training database in comparison with other MPS due to its totally parallel structure. Our algorithm is one of the pioneering works that has a selfenriching capability (Gardet et al.,; Rezaee et al., 2015). It is able to create a new pattern (patch texture) that is not seen in the TI even in dense grid simulation. These patches are visually similar to the TI and have preserved its main structure. Mariethoz et al. Mariethoz and Kelly (2011) have done it to some-extent through transform-invariant distances and Mahmud et al. (2014) through minimum border cut of the TI patch, but they are not able to create totally new patches out of the TI scope. We have also done extensive experiments to compare the ability of our methods with others both in conditional and unconditional simulation. Our comparison is based on visual judgment, ensemble average, variance, variogram reproduction, connectivity reproduction and analysis of distance (ANODI). We have also introduced a new criterion for validation of MPS methods which is the interpolation (or self-enrichment) capability to simulate the SG through different TI while preserving the hard data and continuity of patterns.

The paper is organized as follows. In Section 2, Kwatra et al. (2005) method and its extensions in the optimization-based category will be described. In Section 3, the conceptual and mathematical formulation of our proposed method will be described. Then in Section 4, our simulation results for 2D and 3D in both conditional and unconditional simulation will be shown. They are also compared with some state-of-the-art MPS methods. Computational cost and parallel implementation of our algorithm through GPU and the multi-core of CPU will be described in Section 5. Section 6 is concentrated on the validation of our results. Section 7 evaluates the sensitivity of our algorithm to its different parameters. Conclusions and further work will be put forth in Section 8.

2. Kwatra's method and its extensions

Kwatra et al. (2005) have introduced an energy term for simulating textures and attempt to minimize this energy term through optimization techniques. Its pioneering work acts like an Expectation Maximization (EM) in the pattern recognition field (Wei et al., 2009). The algorithm starts with a random texture as an initial point. Then it extracts all the patches (normally with overlap) of the TI and SG. Then in the M-step, for every patch of the SG, a corresponding best match patch, which honors hard data with a high priority, would be found in the TI. Then in the E-step, the average of all overlapping patches for every pixel would be replaced. For the next iteration, the hard data will be copied to the SG. These two phases would be repeated iteratively while honoring hard data to merge to a constant final SG.

Let T denote the TI with size $A \times B$ and S denote the SG with size $C \times D$. Let t be the column vectorized version of T with size $AB \times 1$ and s be the column vectorized version of S with size $CD \times 1$. Denote N_p as the neighborhood in T centered on pixel p with the size of $w_1 \times w_2$. Then, the sub-vector of t that corresponds to the pixels in N_p is denoted by t_p. Further, let s_p be the vectorized pixel neighborhood in S whose appearance is most similar to t_p under the Euclidean norm (or any other distance function). The sparse grid $S^\dagger$ is a subset of S ($S^\dagger \subset S$) in which the energy is computed (E.g. a patch). The energy over T is defined as:

$$E(s,\{t_p\}) = \sum_{p \in S^\dagger} \|t_p - s_p\|_2^2 \tag{1}$$

Texture-optimization aims at minimizing the above-mentioned cost function in terms of s.

$$s^{*}=\underset{s}{\operatorname{argmin}} E(t,\ s) \tag{2}$$

An EM-like optimization method could be employed to solve this problem (Kwatra et al., 2005). In the expectation step, Eq. (1) is minimized w.r.t. s by assuming that the matches for every data-event in the SG are given. This is done by setting the derivative of Eq. (1) w.r.t. s to zero which leads to a linear system of equations. In the maximization step, the TI is searched to find the best match for every patch of the SG. In the following subsections, these steps are presented in more detail.

2.1. First phase, match finding, maximization step

In the maximization phase, the image is split into patches ($s_{p=1:H}$). This can be done by considering either overlapping or non-overlapping patches, but the patches must be considered so that they include all pixels of the image. The patches can be obtained by sliding a window of size $n = w_1 \times w_2$ in horizontal and vertical directions with step-sizes α and β, respectively. Maximum accuracy can be obtained by considering the maximum possible overlap ($\alpha = \beta = 1$) and undergoing maximum computational burden. On the other hand, by setting $\alpha{=}w_1$ and $\beta{=}w_2$ (no overlap) the minimum computational burden can be achieved at the expense of minimum accuracy. Then for every patch, the best match (t_p) will be found in the TI. This search can either be exhaustive or it can be done only in a fraction, r, of the whole TI. In the following, r is supposed to be one.

2.2. Second phase, averaging, expectation step

In the previous phase, for every patch in the SG the corresponding match in the TI has been found. However, for the SG it is necessary to define each pixel's value. Thus, every pixel of the SG would be placed into a few overlapping patches of the SG and consequently it has a few corresponding best match (patches) in the TI (except for the case on non-overlapping patch extraction). Thus, normally more than one value for every pixel is proposed in the first phase of their algorithm (Fig. 2). Therefore, the final pixel's value will be defined based on the average value of all the proposed pixels' value (E-step). Both Yang et al., (2016) and Barnes et al. (2009) have misunderstood the EM and they have considered the approximate nearest neighbor search of M-step as E-step and vice versa. This mistake has been repeated in many places in their paper including section title, block diagram, and description of the algorithm.

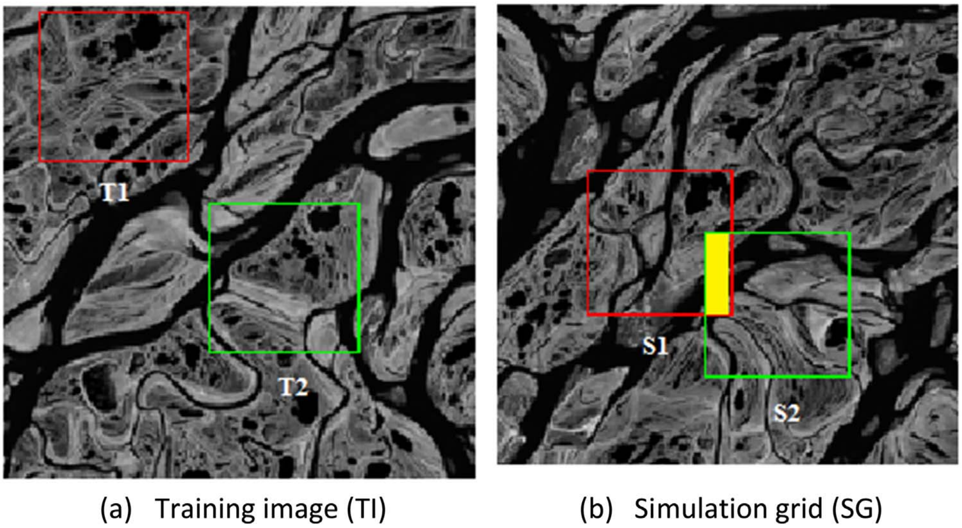
These two steps E and M will be repeated iteratively until the algorithm converges to the final simulating result (Table 1). The SG is initialized through random patch selection from the original image instead of the random pixels' value to have better histogram matching. Then, their algorithm iterates between two expectations and the maximization steps. The results of different iterations of their algorithm are shown in Fig. 3. Fig. 3f is a smoothed version of Fig. 3a due to averaging in E-step. The quadratic structure of their optimization cost function is a well-known cost function in the optimization field. The convergence of their convex cost function is proven in the optimization text (Boyd and Vandenberghe, 2004). It also has a high rate of convergence. Empirically, it will be converged between 2 and 5 iterations.

3. Our proposed method

In this part, we have emphasized the exact contribution of our method to develop the original version of Kwatra et al. (2005). The difference of Yang et al. (2016) as another extension of Kwatra et al. (2005) will be described here.

3.1. Sparse matrix

In this section, we have proposed using pseudo-inverse instead of averaging in E-step based on Peyré (2007) work. This is due to some advances in numerical calculations to calculate the inverse of the sparse matrix very fast (Golub and Loan, 2012). This pseudo-inverse strategy also allows us to do averaging just by one operation. The pseudo-inverse strategy has another promising feature, which allows it to be used simply in high dimensions. The extension of MPS methods to high dimensions is a great challenge and is complicated in a magnitude of



(a) Training image (TI)

(b) Simulation grid (SG)

Fig. 2. T1 and T2 are two matches for S1 and S2 patches in the SG. The pixels in the yellow region will be computed based on every patch, which contains it (here only two patches are shown for simplicity). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Table 1
The conceptual algorithm of iterative unconditional simulation of Kwatra et al. (2005).

Conceptual algorithm of unconditional texture optimization

```
//Initialization:
$T_p^1 \leftarrow$ Random neighborhood in T, $\forall\, p \in S^{\dagger}$
//Iteration loop:
For iteration $n = 1{:}N$ do
// M-step via search
$t_p^{n+1} \leftarrow nearstpatchfors_p^{n+1} inT \quad \forall\ p \in S^{\dagger} = \quad argmin_t \|s_p - t_p\|_2^2$
// E-step via least square (result of each iteration)
$s^{n+1} \leftarrow argmin_s E(s, \{T_p^n\}) = \quad argmin_s \sum_{p \in S^{\dagger}} \|s_p - t_p\|_2^2$
If $t_p^{n+1} = t_p^n \forall\ p \in S^{\dagger}$ then
$s \leftarrow \quad s^{n+1}$
break
end if
end for
```

orders. E.g. Liu and Caselles (2013) are not able to extend its algorithm to 3 dimensions and the Quilting method (Mahmud et al., 2014) is very complicated in three rather than two dimensions. The extension of this algorithm to more dimensions is not mentioned in their paper and is much more complex than 3 dimensions.

Here a sparse matrix is introduced, the pseudo inverse of which can solve our optimization problem more efficiently. Lets extract all the patches (G patches) of TI and all the patches (H patches) of SG with $n = w_1 \times w_2$ pixels. In fact, every pixel of the SG can be included by $n = w_1 \times w_2$ patches except for the pixels in the border. In the M-step of Kwatra et al. (2005), a corresponding patch in the TI is found for every patch of SG. Thus, $U = V \times C \times D \geq V$ numbers of equations are obtained in the maximization step (M-step) to be used later in E-step to obtain the entire pixel's value of the SG. In the highest precise case (maximum overlap or dense sampling of SG) α=1, β=1 so the maximum number of equations is at hand (V=[$A - w_1$] × [$B - w_2$],U=[$A - w_1$] × [$B - w_2$]×C × D) and the matrix is over-determined. In the lowest precise case, patches are extracted without overlap (non-overlapping case) so there is only one equation for every unknown ($V = 1$, $U = C \times D$). In the mathematical formulation, our result will be as follows:

$$\varphi_{n \times U} \mathrm{F}_{U \times 1} = \rho_{n \times 1} \tag{3}$$

$$\Phi_{nU \times U} \mathrm{F}_{U \times 1} = \mathrm{P}_{nU \times 1} \tag{4}$$

$$\Phi^T_{U \times nU} \Phi_{nU \times U} \mathrm{F}_{U \times 1} = \Phi^T_{U \times nU} \mathrm{P}_{nU \times 1} \tag{5}$$

$$F_{U \times 1} = (\Phi^T \Phi)^{-1}_{U \times U} \Phi^T_{U \times nU} \mathrm{P}_{nU \times 1} = \Phi^{\dagger}_{U \times nU} \mathrm{P}_{nU \times 1} \tag{6}$$

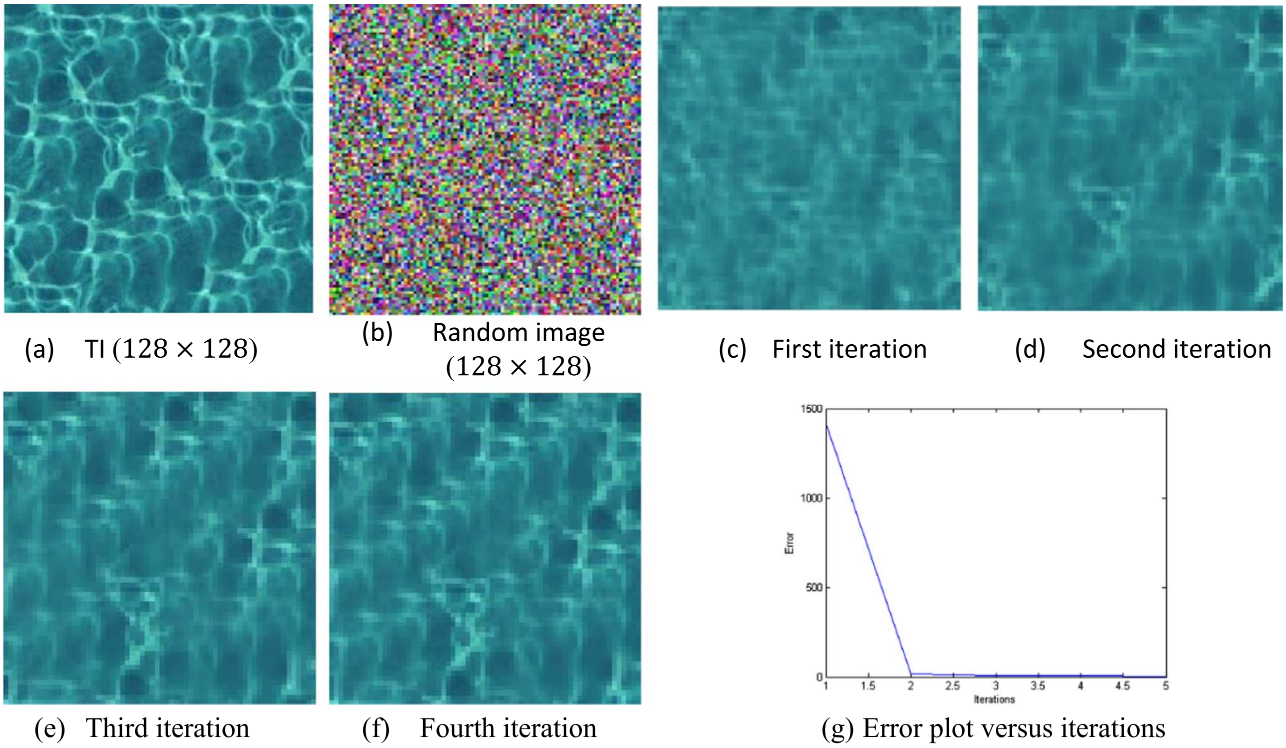
In Eq. (3), $F_{U \times 1}$ is a vectorized version of the pixels of an image. $\rho_{n \times 1}$ includes extracted pixels of one patch of the image. The extended version of Eq. (3) is shown in Eq. (4) which extracts all the patches of the image simultaneously (highest precise case α=1, β=1 is considered for more simplicity). It is worthy to express that $\Phi_{nU \times U}$ concatenates different patches of $\varphi_{n \times U}$ in row dimension. A sample of this event is shown in Fig. 4. In this figure, four overlapping patches are extracted.

The final value of $F_{U \times 1}$ is obtained through pseudo-inverse of $\Phi_{nU \times U}$ (Eqs. (5) and (6)). $\Phi_{nU \times U}$ is a sparse matrix in which there are a few non-zero elements in each of its columns and rows. Every row has $w_1 \times w_2$ non-zero elements which are equal to the pixels of each patch. Every column of this matrix will have at most $w_1 \times w_2$ non-zero elements. The column's value shows which patches contain each pixel. This characteristic is valuable in computing the inverse of the matrix in Eq. (6). Some software such as MATLAB can easily handle the inverse of the large matrix if they are able to represent it in sparse format. For very large matrices, a direct solution of linear equations will be used.

Now in this part the implementation of our algorithm via the sparse matrix will be discussed (Table 1). The sparse patch extraction matrix Φ and pseudo inverse matrix $\Phi^{\dagger}$are also computed in advance. To prevent an unwanted singular case, a very small value (ϵ) can be added to the matrix value of the SG.

3.2. 3D simulation and higher dimensions

The original texture optimization simulation (Kwatra et al., 2005) does not address the 3D image. In this paper, an extended version of texture optimization is introduced which could handle 3D image simulation or their dual concept video texture synthesis in texture terminology. The logic of our algorithm for 3D image simulation is the same as 2D. Therefore, only small modification in the algorithm is needed. Let L, $\hat{L}$ be the number of planes (frames) of 3D simulating and TIs. Let patch size be $n = w_1 \times w_2 \times w_3$. Let the amount of displacements between two consecutive patches for patch extraction of the SG be α, β, γ. Let the TI size be $A \times B \times L$. Let the SG size be $C \times D \times \hat{L}$. Then all the voxels of the TI, T ($G = \hat{U} voxels$) with $\hat{U} = [\frac{A - w_1}{\alpha}] \times \left[\frac{B - w_2}{\beta}\right] \times \left[\frac{L - w_3}{\gamma}\right]$ will be extracted. All the voxels of the



(a) TI (128 × 128)

(b) Random image (128 × 128)

(c) First iteration

(d) Second iteration

(e) Third iteration

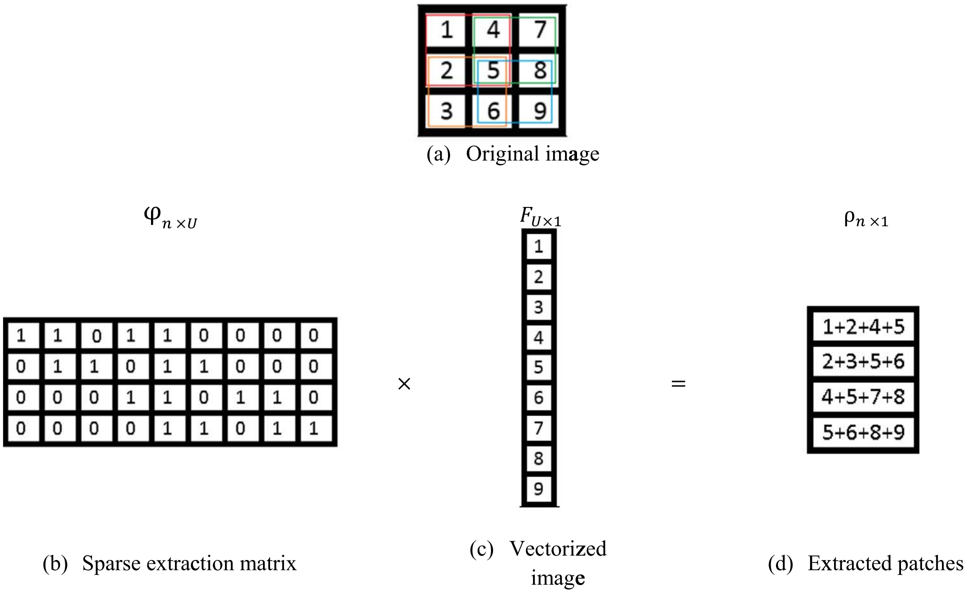
(f) Fourth iteration

(g) Error plot versus iterations

Fig. 3. Figure (a) is a color-TI (128 × 128) of fluid which our model is based on. Figures (b-f) are the different iterations (result of E-step) of our algorithm with patch size 32 × 32 and with patch displacement ($\alpha = 1, \beta = 1$). Figure (g) shows the differences between two subsequent iterations. This difference can be used as a stopping criterion for our algorithm. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

SG $S^{1}_{U\times 1}$ with $w_1-\alpha$, $w_2-\beta$,$w_3-\gamma$ overlap ($H = nU$ voxels), $U=[\frac{C-w_1}{\alpha}]\times\left[\frac{D-w_2}{\beta}\right]\times\left[\frac{\hat{L}-w_3}{\gamma}\right]$ will be extracted too. Then the pseudo inverse $\Phi^{\dagger}$will also be computed (Eqs. (3)–(6)).

The matrix representation (Table 2) of our algorithm allows it to be used in any high dimension. In fact, our algorithm extracts any n-dimensional patch of TI and SG with overlap and puts them in the 1D column. Then all these patches of TI are concatenated in the second dimension of the Φ matrix. Then the inverse of this sparse matrix will be calculated ($\Phi^{\dagger}$). Therefore, this matrix always transforms any n-dimensional data into a two dimensional sparse format. This fact has made our algorithm much easier to work with than the other pixel-based and patch-based methods of MPS like Quilting (Mahmud et al., 2014) and Graphcut (Zahner et al., 2016; Li et al.,) in high dimensional



(a) Original image

$$\underset{\varphi_{n\times U}}{\begin{bmatrix} 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 \end{bmatrix}} \times \underset{F_{U\times 1}}{\begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \\ 9 \end{bmatrix}} = \underset{\rho_{n\times 1}}{\begin{bmatrix} 1+2+4+5 \\ 2+3+5+6 \\ 4+5+7+8 \\ 5+6+8+9 \end{bmatrix}}$$

(b) Sparse extraction matrix

(c) Vectorized image

(d) Extracted patches

Fig. 4. Figure (a) is the pixel's number of original image with 9 pixels. Surrounding patches are shown via color squares. Figure (b) is the sparse extraction matrix with a patch size of 2 × 2. Figure (c) is the column-vectorized version of Figure (a). Figure (d) is the extracted patches of Figure (a) with patch displacement ($\alpha = 1, \beta$=1). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Table 2
Matrix implementation of iterative unconditional and conditional 3D simulation.

Matrix implementation of unconditional and conditional 3D texture optimization

```
//Initialization:
  1)  Define the patch size:$n$=$w_1 \times w_2 \times w_3$.
  2)  Define the amount of patch displacements for patch extraction of SG
      ($\alpha$, $\beta$, $\gamma$).
  3)  Choose the TI with size $Y = A \times B \times L$
  4)  Randomly initialize the SG with size $C \times D \times \hat{L}$.
  5)  Consider $\Phi$ (Eq.(4)) then compute pseudo inverse matrix $\Phi^{\dagger}$ from it (Eq.(6)).
  6)  $TT_{G\times1}$ =    Extract all the patches of TI, (G patches).
  7)  $SS^{1}_{nU\times1}$= Extract all the patches of SG $S^{1}_{U\times1}$ with $w_1-\alpha$,   $w_2-\beta$,   $w_3-\gamma$
overlap ($H$patches).
//Iteration loop:
  8)  For iteration n = 1:N do
//M-step via search
  9)  $Z^{n}_{nU\times1}$ =    Find the best match for each patch of SG $S^{n}_{U\times1}$ in TI (TT) through
search in $r$ portion of it.
  10)  $SS^{n}_{nU\times1} = Z^{n}_{nU\times1}$.
//E-step via least square
  11)  $S^{n+1}_{U\times1}$ =$\Phi^{\dagger}_{U\times nU} \times SS^{n}_{nU\times1}$ //New SG corresponding to line 4 ofTable 1.
  a   Change the hard data points to their initial values //only for conditional
      simulation (Global strategy)
  12)  If $S^{n+1}_{U\times1} - S^{n}_{U\times1} <$ threshold or n=$N_{max}$
  13)  $S^{Final} \leftarrow$    $S^{n+1}_{U\times1}$
  14)  break
  15)  end if
  16)  end for
```

data. However, our method has an important limitation. It is encountered with computation of large pseudo inverse for large TI due to its over-determined structure. In that case, multi-resolution starategy will be used. For coarse resolution, low patch displacement (PD) will be used which uses the accuracy of our algorithm for main structure. For fine resolution, large patch displacement will be used which makes our algorithm similar to patch-based algorithm in speed parameter.

3.3. Conditional simulation with extended cost function

In the previous section, our algorithm is described in unconditional simulation. However, our algorithm is also able to handle the conditional simulation. Conditional simulation is being done through some modification and extension of our cost function. In fact, the specific structure of the algorithm based on cost function makes it able to handle any situation, which can be represented through a newly added term in the cost function. E.g. in Eq. (7), a gradient term (edge of the image) can be added (Ds_p and Dt_p) (Kwatra et al., 2005).

$$E(t, s_p) = \sum_{p\in S^{\dagger}} \|t_p - s_p\|^2 + \mu \sum_{p\in S^{\dagger}} \|Ds_p - Dt_p\|^2 \tag{7}$$

Any other term can be added with its corresponding weighting factor. The information of any secondary data can also be added to our cost function provided that they are able to formulate as a specific mathematical term in convex format or convergable non-convex format. The hard data of conditional simulation can also be added as a term to our cost function. For this goal, a term is added to line 4 of Table 1. In other words, the cost function also considers the hard data point (s_p^{hard} and t_p^{hard}) with a weighting coefficient (λ in Eq. (8)). This forces the search part of our algorithm (M-step) to find only the matches that maximally honor hard points.

$$T_p^{n+1} = argmin_t(\|s_p - t_p\|^2 + \lambda\|s_p^{hard} - t_p^{hard}\|^2) \tag{8}$$

Table 3
Best match part of 2D and 3D conditional simulation.

Pseudo code of best match finding in conditional simulation

```
1.  For i=1: H //This For is parallel
2.  Er=inf; //very large value for initial error
3.  Choose a random path in TI
4.  For $j = 1{:}rG$// Search in$r$portion of it
5.  Error(j)=$\|t_i - s_j\|_2^2 + \lambda\|t_i^{hard} - s_j^{hard}\|_2^2$//Local strategy
6.  If (Error(j) < Er)
7.  Er= Error(j);
8.  J_best (i)=j;
9.  End if
10.  End for
11.  End for
```

To further clarify the impact of hard data points in conditional simulation, the subsequent change in matrix implementation will be discussed. Table 3 has shown the expansion of the best match found in 2D and 3D simulation (line 9 Table 2). In Table 2, every t_i,s_j has n elements. But normally only a portion of a patch may contain hard data points ($n^{hard}<n$). Therefore, a comparison of hard data points of the TI patch t_i^{hard} and SG patch s_j^{hard} will be done on n^{hard} data points if they exist. This strategy deals with hard data locally in every patch simulation similar to Yang et al. (2016). However, we have also proposed a global strategy. The pixels value of the hard data points can be changed in pseudo inversion ($S_{U\times1}^{n+1}$ in line 11 in Table 2). We will bring back these values to their initial values (line 11.a in Table 2). This helps the algorithm to honor hard data much better. Then, in the next iteration, these pixels will be weighted (λ) for the best match found in the TI. Therefore, our algorithm tries to find the patches that are mostly adapted to the current hard data points. A mixture of this local and global strategy does not change the convexity of our constrained optimization formulation (Boyd and Vandenberghe, 2004) and its convergence is still guaranteed.

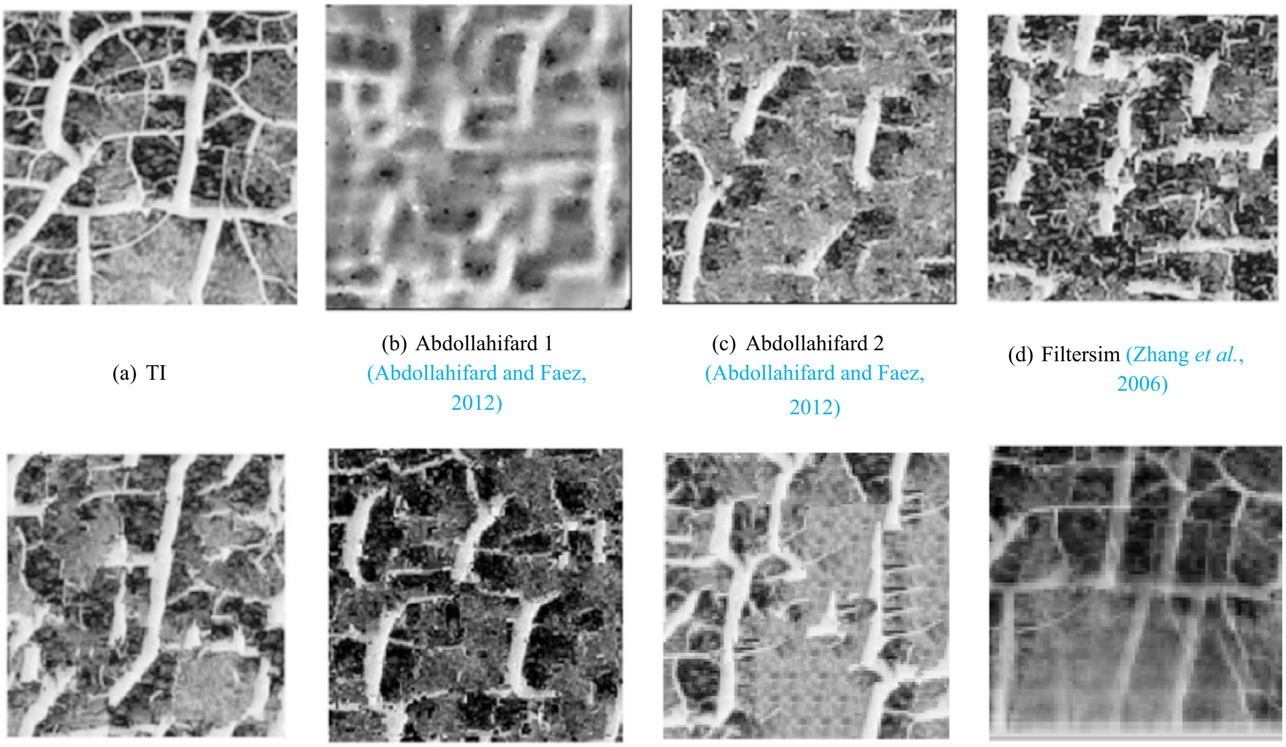
3.4. Parallel implementation using GPU

In fact, every node of our SG (in M-step) can be simulated independently. Consider Fig. 2. Each of the S_1 and S_2 patches find their corresponding matches in the TI (T_1 and T_2) independently although they have common pixels in the current iteration. The central pixel of the best matches of each S_1 and S_2 will be replaced into the SG in the next iteration of our algorithm. This means that each pixel of the SG can be devoted to a core of CPU (E.g. through *parfor* command in MATLAB) or GPU (E.g. through *GPU kernel* in MATLAB) and the result will be saved. Finally, after the parallel simulations are obtained, all pixels (nodes of simulation) will be pasted to the SG in the next iteration. In fact, the second *For* of Table 3 (line 1–11) can be executed parallel and every core of GPU (or CPU) can handle one patch s_j independently.

The main parallel structure of our algorithm allows it to use the multi-core of CPU or the great ability of cores of the GPU to search the entire training database as fast as possible. This is while the sequential structure of most MPS methods either random scan (Mariethoz et al., 2010; Abdollahifard and Faez, 2013) or raster scan (Mahmud et al., 2014; Abdollahifard, 2016; Tahmasebi et al., 2012b) has deprived them from using fully parallel fast techniques. Yang et al. (2016) have also not emphasized this great ability of the optimization-based method. They have attempted to use the non-exhaustive patch-match search strategy of Barnes et al. (2009) instead of the fully parallel exhaustive search. Therefore, our algorithm has the fastest search strategy in TI among all exhaustive search methods of MPS methods.

4. Simulation results

In this part, the experimental result of our algorithm will be shown.



(a) TI

(b) Abdollahifard 1 (Abdollahifard and Faez, 2012)

(c) Abdollahifard 2 (Abdollahifard and Faez, 2012)

(d) Filtersim (Zhang *et al.*, 2006)

(e) DISPAT (Honarkhah and Caers, 2010)

(f) DS (Mariethoz *et al.*, 2010)

(g) Quilting (Mahmud *et al.*, 2014)

(h) PCTO_SIM

Fig. 5. Unconditional simulations of differen t MPS methods with complex TI.

All the experiments were carried out on an Intel core i7-2600k, 3.4 GHz with 16 GB RAM. Our operating system is 64 bit Windows 7. Our graphics card, which supports GPU, is GeForce GTX 590 with 512 CUDA cores. It has 4 GB RAM GDDR5 with a graphics clock rate of 607 MHz, and processor clock rate of 1215 MHz.

4.1. TIs used in this study

This section briefly introduces the 2D and 3D TIs used. In the following, the corresponding realizations and the results will be shown. For convenience, these TIs are named AA, BB, CC, DD, EE, FF and GG. Training image AA (370 ×370) (Fig. 2a) consists of gray scale values of a satellite image of the Lena Delta in (Russia) (Mahmud et al., 2014). Training image BB (250 × 250) is a color image of water (Fig. 3a) (Kwatra et al., 2005). Training image CC (159 × 159) (Fig. 5a) is a continuous TI (Honarkhah and Caers, 2010) consisting of thick and narrow cracks as its extreme features. Training image DD (180 × 180 × 180) (Fig. 6a) is generated with the FFT-MA (Fast Fourier Transform-Moving Average) method (Goovaerts, 1997; Le Ravalec et al., 2000), using a Gaussian variogram model that results in very smoothly varying structures. Training image EE (150 × 150 × 150) (Fig. 6i) consists of 3-D meandering channels[1] (Mahmud et al., 2014). These channels are connected in all X, Y and Z directions and it is a good benchmark for the 3D connectivity test. Training image FF (250 × 250) is a binary image (Fig. 7a) of a reservoir consisting of long connected sinuous channels. This image is adapted extensively as a proper image for evaluating MPS methods like SNESIM (Strebelle, 2002), FILTERSIM (Zhang et al., 2006), DISPAT (Honarkhah and Caers, 2010), CCSIM (Tahmasebi et al., 2012a), DS (Mariethoz et al., 2010), and Quilting (Mahmud et al., 2014) in the recent decade. It displays meandering channels, which are highly important in reservoir modeling and simulation. Training image GG (100 × 100) (Fig. 10a) is extracted from DISPAT (Honarkhah and Caers, 2010). It has four facies that can be used for multi-variant simulations.

4.2. Tests for unconditional simulation

In this part, the result of our algorithm for unconditional simulation for both 2D and 3D states will be evaluated.

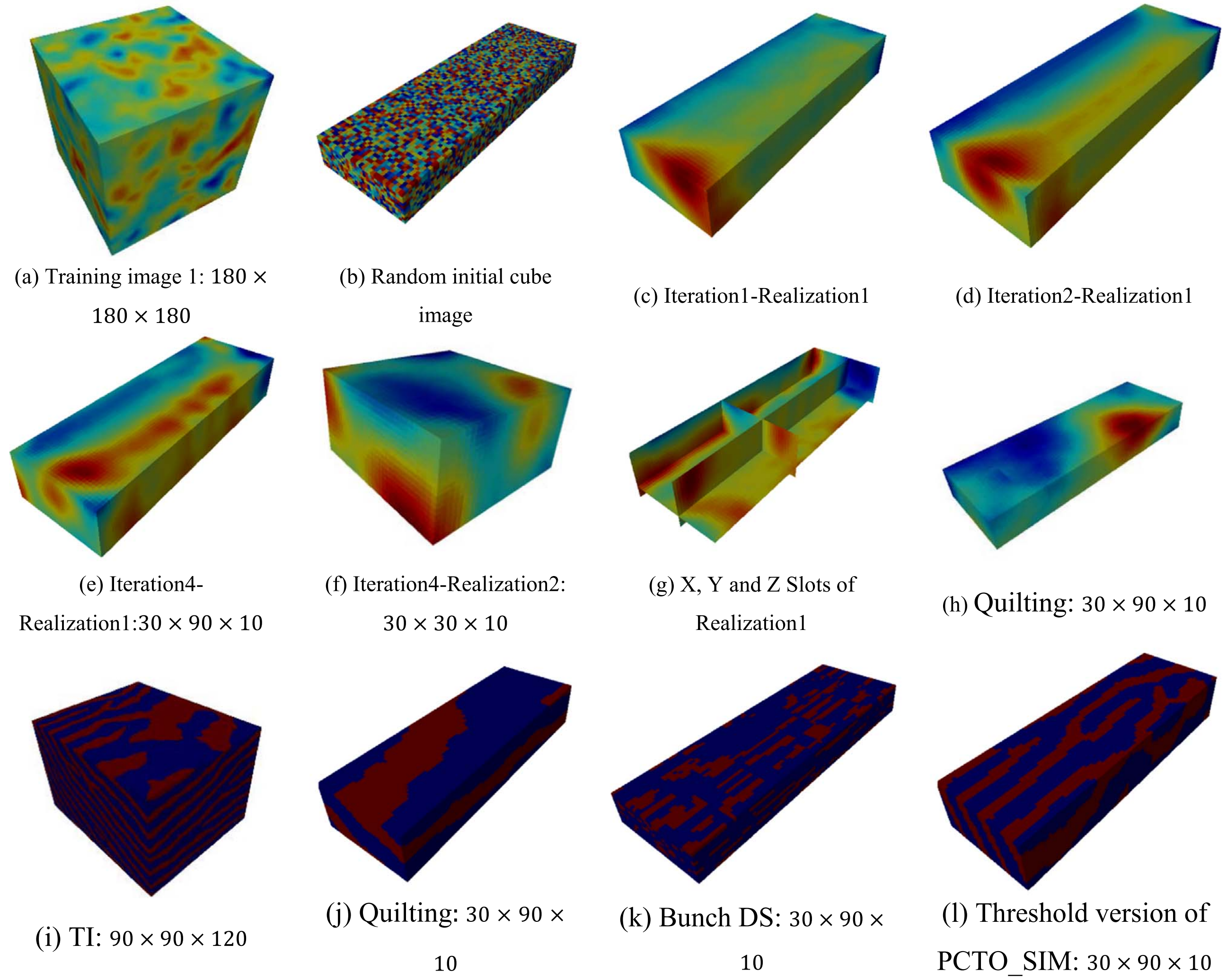
4.2.1. 2D unconditional simulation

In this part, the result of our 2D unconditional simulation will be evaluated on complex and highly structured images (Honarkhah and Caers, 2010) (Fig. 5a). We have compared our method with six well-known MPS methods. Most of the images are extracted from their corresponding paper. Fig. 5b, c are from Abdollahifard and Faez (2012). Fig. 5d, e are from Honarkhah and Caers (2010). Fig. 5f is from Abdollahifard and Faez (2013). Fig. 5g is created through the execution of its author's MATLAB code in GitHub.[2]

It is observable that for this image, all of the methods including ours are not able to create a good similar image. For most of the methods, there is discontinuity in the image lines. This is while our result has preserved the continuity of the image much better visually. Our simulation also simulated the niceties (like small and thin lines) of the TI. The important problem of our method is its smoothed structure. This is created because of the averaging operation of our algorithm. This problem can be solved to some extent by deblurring techniques in image processing (Szeliski, 2010; Abdollahifard and Ahmadi, in preparation) but it is out of the scope of this paper.

[1] https://github.com/SCRFpublic/Stanford-VI-E

[2] https://github.com/pejmant/MS_CCSIM.


(a) Training image 1: 180 × 180 × 180

(b) Random initial cube image

(c) Iteration1-Realization1

(d) Iteration2-Realization1

(e) Iteration4-Realization1:30 × 90 × 10

(f) Iteration4-Realization2: 30 × 30 × 10

(g) X, Y and Z Slots of Realization1

(h) Quilting: 30 × 90 × 10

(i) TI: 90 × 90 × 120

(j) Quilting: 30 × 90 × 10

(k) Bunch DS: 30 × 90 × 10

(l) Threshold version of PCTO_SIM: 30 × 90 × 10

Fig. 6. Comparison of our 3D unconditional simulation with Quilting (Mahmud et al., 2014). Different realizations of our 3D unconditional simulation are shown. Patch size is 16 × 16 × 5. Threshold value is 0.5. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

4.2.2. 3D unconditional simulation

In this part we will examine our method for simulating an unconditional 3D application. We have used SGeMS software for 3D image rendering (Remy et al., 2009). Fig. 6 shows different iterations of our algorithm. It also shows two different realizations of our algorithm. It is visually observable that our algorithm has preserved the continuity of the structure well. Fig. 6g shows the interior areas of one of our realizations. It is observable that the continuity of the patterns has been preserved in sample X, Y and Z slots.

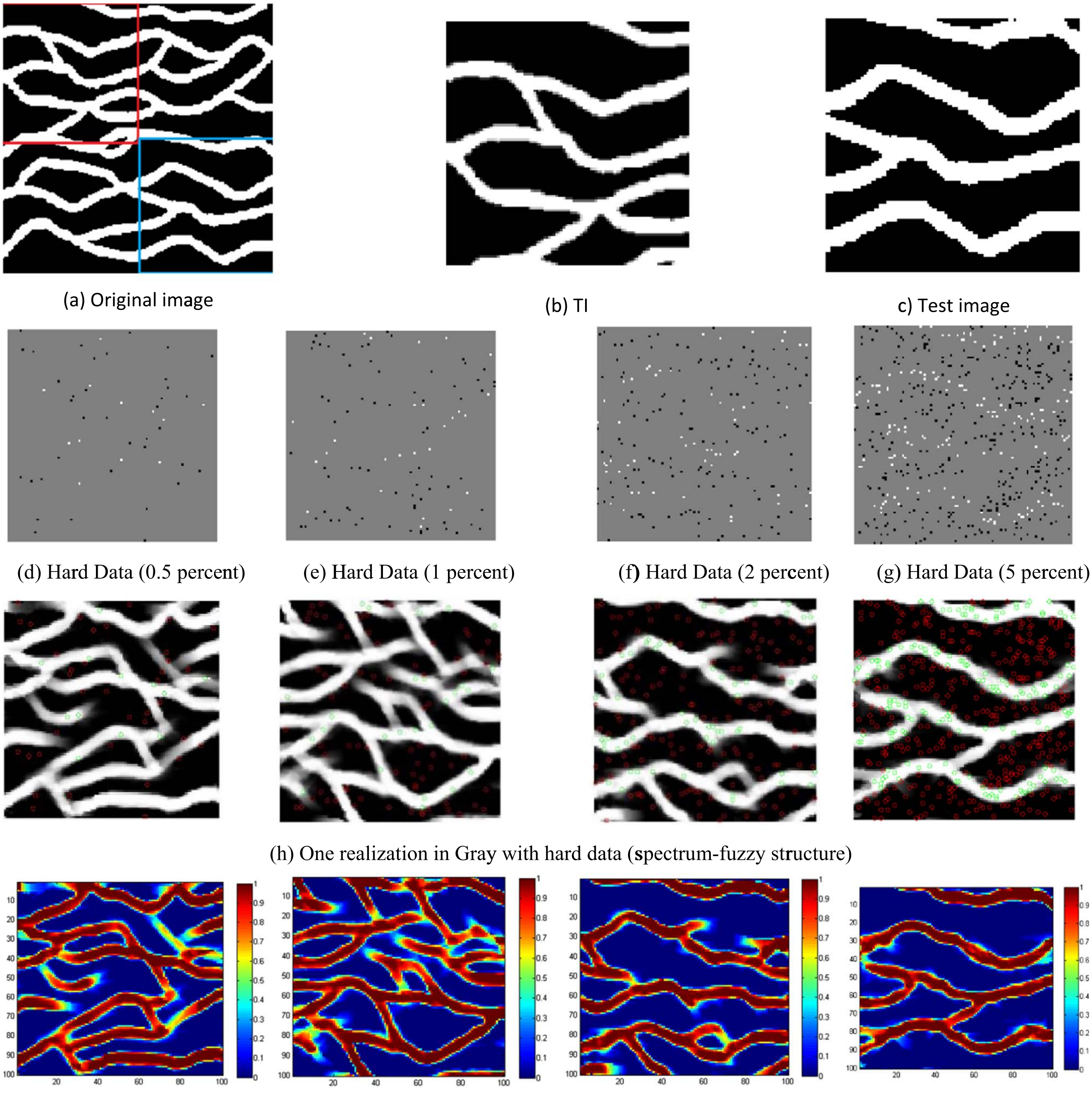
Fig. 6 has also compared our algorithm with Bunch DS (Rezaee et al., 2013) and Quilting (Mahmud et al., 2014) as one of the best MPS results up to now, both of these methods have good results in continuity preservation. They also have the same structure of their TIs. The result of Bunch DS (Rezaee et al., 2013) has not preserved the continuity of the channels well. It is also worthy to note that for continuous data, the original result of our algorithm will be shown, but for categorical data the threshold version of our result will be shown.

4.3. Tests for conditional simulation

In this part, the result of our algorithm for conditional simulation for both 2D and 3D state will be evaluated. Conditional simulation is a dual concept of constrained optimization in optimization terminology (Boyd and Vandenberghe, 2004).

4.3.1. 2D conditional simulation

In MPS methods, unconditional simulation will be used to define the similarity of one realization of the stochastic model with real data. However, the main challenge is in a conditional simulation especially with sparse hard data. This is exactly the main strength of our proposed algorithm. In other words, while our algorithm has a good result in unconditional simulation, it may not be the best one. But in a conditional simulation, our algorithm has the superior result in comparison to the state-of-the-art algorithm, especially for sparse hard data. In Fig. 7 we have chosen the training (Fig. 7b) and test image (Fig. 7c) from different parts of a well-known shale and sand image (Fig. 7a). This non-overlapping strategy is done deliberately to evaluate the interpolation ability of our algorithm. The results of our algorithm with a different percentage of hard data with sparse sampling (Fig. 7d) to dense sampling (Fig. 7g) from Fig. 7c are shown in Fig. 7h. It is observable that even the sparse data has honored the hard data well. In fact, every realization of our algorithm has passed from the white color channel data and has not passed from the black color non-channel data. These are shown through green and red circles in the SG of Fig. 7h. Long connected paths have also been simulated even with very sparse data. It is observable that sparse data results have some imprecise estimates about the other unknown parts of the image. This event is expected because the hard data is very low for accurate judgment. It is also worthy to note that Fig. 7h for sparse hard data



(a) Original image

(b) TI

c) Test image

(d) Hard Data (0.5 percent)

(e) Hard Data (1 percent)

(f) Hard Data (2 percent)

(g) Hard Data (5 percent)

(h) One realization in Gray with hard data (spectrum-fuzzy structure)

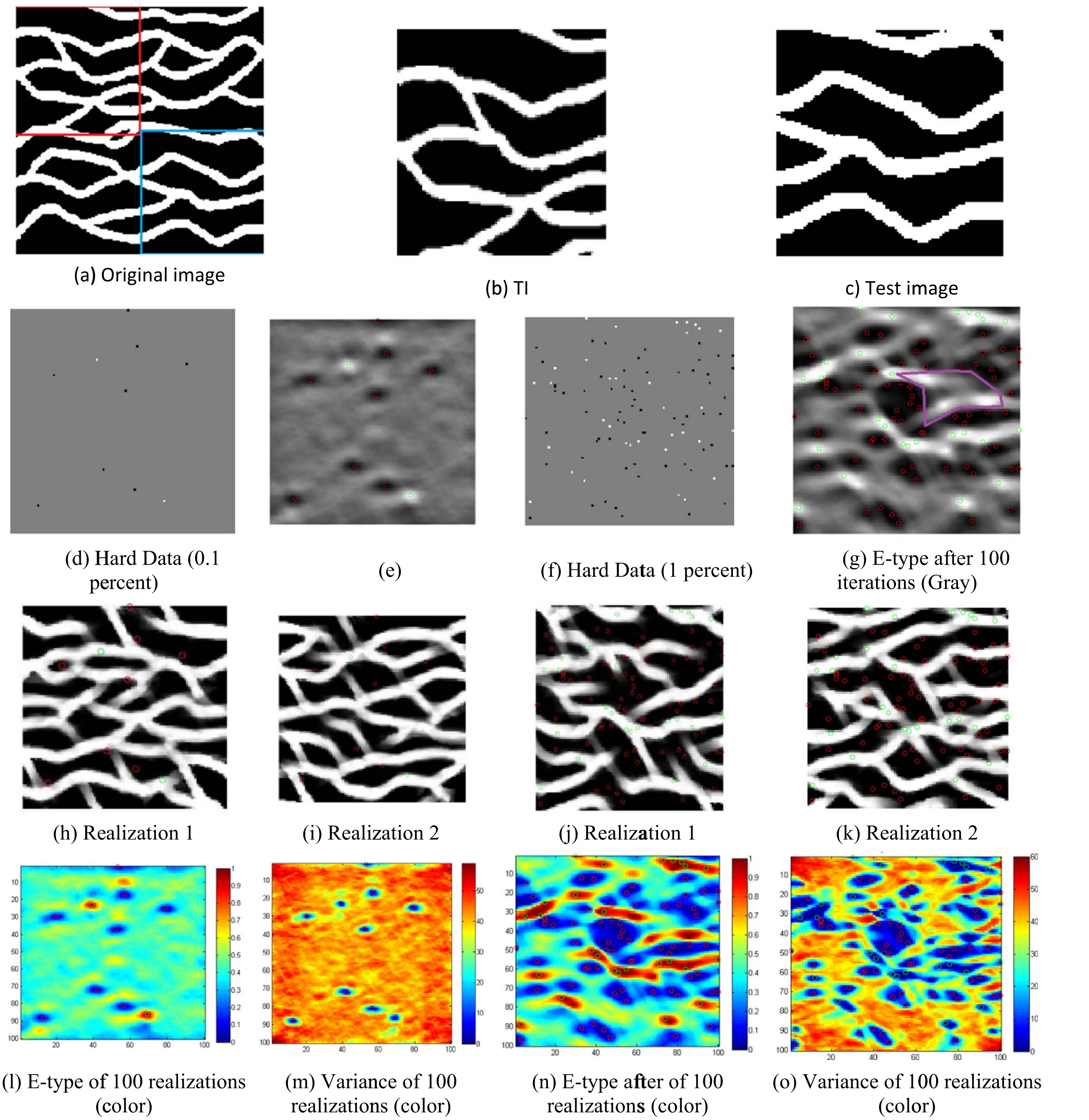
(i) One realization in color (spectrum-fuzzy structure)

Fig. 7. The result of our 2D Conditional Simulation with different percentages of hard data. Training and test images are chosen from two different squares of Figure a. The SG is 100 × 100 with a patch size of 16 × 16 and $\lambda = 20$. All the results are shown after 5 iterations with patch displacement ($\alpha = 1, \beta = 1$). The green circle is channel hard data and the red circle is non-channel hard data. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

(0.5% and 1%) has more white channels than the TI. This is due to the fact that the hard data is too low and the large weighting factor of hard data forces the algorithm to choose channels which encompass the hard data.

It is also worthy to note that one realization of our algorithm has a spectrum (fuzzy) structure (Fig. 7i). This event takes place because of the fact that in every realization we have an average of some overlapping patches. Therefore, one realization of our algorithm has a semi e-type structure. It is not a real E-type due to its local averaging in comparison with global averaging of e-type. This feature has caused our algorithm to preserve the hard data well. In fact, it uses the judgment of a few realizations just in one realization. As it is expected, uncertainty increases when we take distance from hard data (Fig. 7h). This behavior is only observable in expected results of other algorithms while it can be seen in one realization of ours.

It is also notable that if we choose the threshold value as strictly very low, highly connected simulating channels will be created and if we choose it very high we will experience some discontinuities in the simulating channels. We have chosen a threshold value of 0.5 in all our experiments for more simplicity. Other thresholding strategies like Otsu's method can also be used here (Sezgin, 2004). We have also repeated our methods 100 times to create an E-type result. In Fig. 8, it is observable that each realization has honored the hard data point very well while our training and test are chosen from different parts. This good hard data preservation has rarely been found in any other MPS method including CCSIM (Tahmasebi et al., 2014), Quilting (Mahmud et al., 2014), DS (Mariethoz et al., 2010) and Bunch DS (Rezaee et al., 2013) and it shows the superiority of our algorithm to most MPS


(a) Original image

(b) TI

c) Test image

(d) Hard Data (0.1 percent)

(e)

(f) Hard Data (1 percent)

(g) E-type after 100 iterations (Gray)

(h) Realization 1

(i) Realization 2

(j) Realization 1

(k) Realization 2

(l) E-type of 100 realizations (color)

(m) Variance of 100 realizations (color)

(n) E-type after of 100 realizations (color)

(o) Variance of 100 realizations (color)

Fig. 8. E-type results of our 2D conditional simulation. Training and test images are chosen from two different squares of Figure a. The SG is 100 × 100 with a patch size of 16 × 16 and $\lambda = 20$. All the results are shown after 5 iterations with patch displacement ($\alpha = 1, \beta = 1$). The green circle is channel hard data and the red circle is non-channel hard data. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

methods. This feature is due to our global hard conditioning strategy in conditional simulation which is not in Yang et al. (2016). Our E-type result also has a precise result. Expectation results have a low value near the non-channel hard data point. Conversely, it has a high value near the channel hard data point. All the hard data points are met. For simplicity, channel pixels are shown via a green circle and non-channel pixels via a red circle. Connectivity of the pattern is also preserved well visually.

One important feature of our result is its interpolation capability. Fig. 8g shows that our algorithm has created channels in places where no hard data exists. This high probability channel is created in the violet region to preserve the connectivity of the channels of the whole image. Yellow colored channels in Fig. 8l, n also show that our algorithm has attempted to preserve the continuity even in places where there are no hard data. Some of these channels coincide with real channels of the test image Fig. 8c.

4.3.2. 3D conditional simulation

In this part, the ability of our algorithm for handling 3D conditional simulation will be evaluated. In Fig. 9, we have chosen a 3D image that has connectivity in all directions. Training and test images are different. One realization of our algorithm is shown in Fig. 9d. Our threshold image (Threshold=0.5 for image with gray-level=[0 1]) is shown in Fig. 9.g. We have also shown the slots (cuts) of our

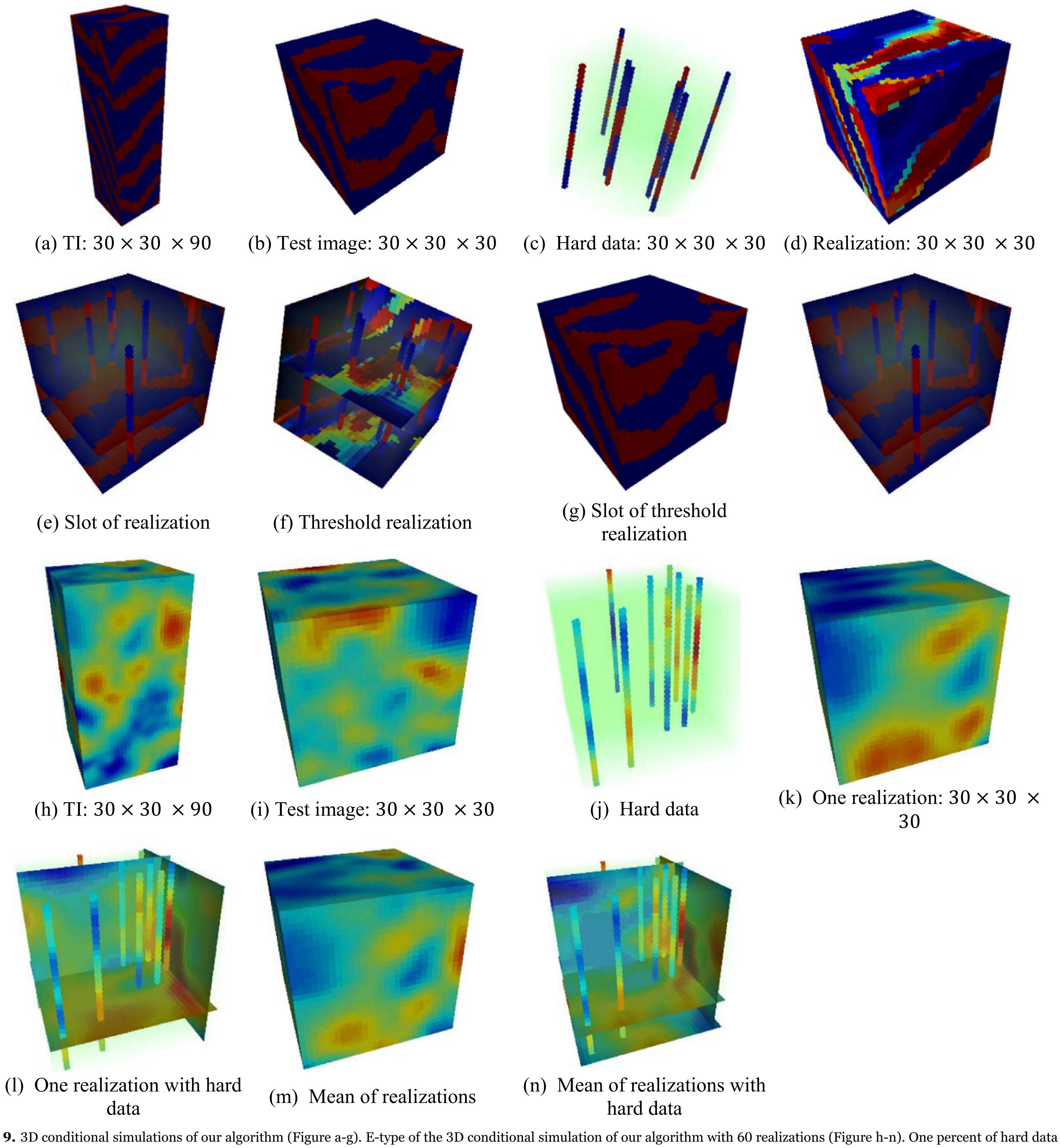

(a) TI: 30 × 30 × 90
(b) Test image: 30 × 30 × 30
(c) Hard data: 30 × 30 × 30
(d) Realization: 30 × 30 × 30
(e) Slot of realization
(f) Threshold realization
(g) Slot of threshold realization
(h) TI: 30 × 30 × 90
(i) Test image: 30 × 30 × 30
(j) Hard data
(k) One realization: 30 × 30 × 30
(l) One realization with hard data
(m) Mean of realizations
(n) Mean of realizations with hard data

Fig. 9. 3D conditional simulations of our algorithm (Figure a-g). E-type of the 3D conditional simulation of our algorithm with 60 realizations (Figure h-n). One percent of hard data is honored well in all x, y, and z slots of our result. Patch size is 16×16×16. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

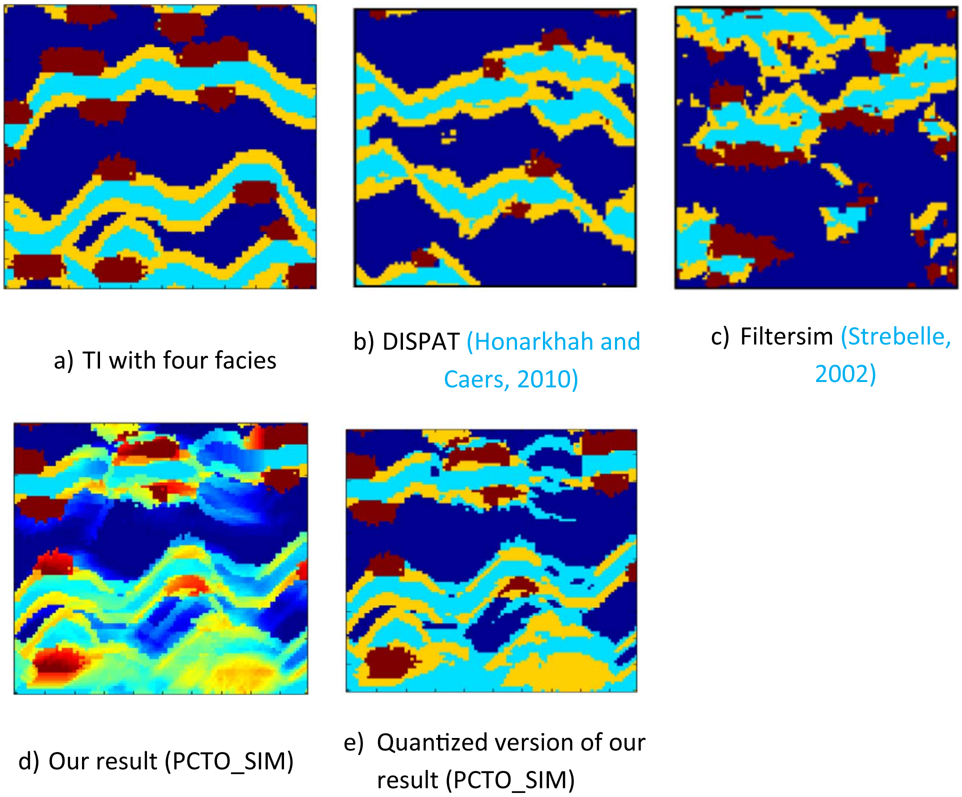
realizations with hard data. It is observable that it has honored the data well in x, y, and z directions.

We have also repeated our experiment for the E-type image of our realizations in Fig. 9h–n. It is observable that both one realization and the E-type of our realizations have honored hard data well. The connectivity of the pattern is also preserved while the structure of the simulated grid is similar to the TI.

4.4. Categorical simulation

Our algorithm is also able to simulate Categorical (multi-variant) cases (Fig. 10). If different colors are devoted to each of the z variants, then the simulated grid will have some new medium colors, which do not belong to any of the variants. This is due to the averaging feature of our method. To avoid this problem, we can quantize the value of the SG to the z number of facies (Fig. 10g). We have also compared our method with the first strategy with DISPAT (Honarkhah and Caers, 2010) and Filtersim methods (Strebelle, 2002). The simulation of these two methods are extracted from Honarkhah and Caers (2010).

All of the methods still have some problems to visually mimic the behavior of TI. DISPAT (Honarkhah and Caers, 2010) has some isolated blobs and the size of red blobs are smaller than TI. It also


a) TI with four facies

b) DISPAT (Honarkhah and Caers, 2010)

c) Filtersim (Strebelle, 2002)

d) Our result (PCTO_SIM)

e) Quantized version of our result (PCTO_SIM)

Fig. 10. Comparison of our unconditional multi-variant simulation with the DISPAT and Filtersim method. Patch size is 64 × 64. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

has thinned the blue channels in some parts. Filtersim (Strebelle, 2002) has both the previous problems in addition to unconnected channels and the contact of light blue regions with dark blue regions. Our method has the same contact problem of Filtersim (Strebelle, 2002) but it has preserved the connectivity of the channels better than them. Our method is also better than DISPAT (Honarkhah and Caers, 2010) in creating the isolated point and the size of red blobs. It seems that the of rank of results are visually DISPAT (Honarkhah and Caers, 2010), PCTO_SIM and Filtersim (Strebelle, 2002). It is also worthy to express that if the contact region of light blue parts with deep blue parts is important, we can add a secondary data to our cost function to simply force our model to consider it. Our method is sensitive to the natural orders between facies. Therefore, its result differs with different order of facies and in the cases where there is no clear order; it can offer some different results.

In section 6, it will be shown that our algorithm in unconditional simulation is not necessarily the best one. However, our method has a superior result with the others in conditional simulation.

5. Computational cost

One of the main challenges in the MPS method is the speed of the simulation. The computational cost of these algorithms consists of two major elements. The first factor is the number of nodes in the SG (n_1). These nodes are a great challenge especially in 3D images. A few methods have focused on decreasing the simulation nodes. The patch-based method tries to simulate one group of pixels at a time instead of one pixel to alleviate this problem (Mahmud et al., 2014; Rezaee et al., 2013).

Another factor is the computational cost of each node. This factor normally depends on the computational cost of the proposed algorithm (n_3)and the searching time (n_4) to find the best match in the TI. The computational cost of the algorithms is normally constant and some minor modifications can be exerted on it. However, the most important and time-consuming factor is the searching time in the training database. This has vital importance in a 3D large TI database. In fact, richer databases are larger ones that cause all the algorithms to have a more precise judgment. Therefore, lots of methods have been proposed to decrease the amount of search in the training database while having an efficiently rich database. Dimension reduction (Honarkhah and Caers, 2010), gradient descent search (Abdollahifard and Faez, 2013), and list approach (Renard et al., 2011) are a few techniques proposed for this goal. However, these techniques often decrease the accuracy of the algorithm and may probably trap in local minimum. Another way to speed up this computation step is to compute the distances by convolution (Kwatra et al., 2003; Tahmasebi et al., 2014).

The parallel main structure of our algorithm allows it to search exhaustively in a richer database (or in a larger portion of a database) by consuming shorter time in comparison with other MPS algorithms.

In Table 4, the computational cost of our algorithm is compared with other well-known MPS algorithms. Our GPU has 1024 threads. Our algorithm is superior to the other algorithms in searching the TI database. It is also in the current trend of the computer hardware, which is in the growth of the CPU and GPU cores for better parallel computation. Quilting (Mahmud et al., 2014) and Bunch DS (Rezaee et al., 2013) (patch DS) are superior to the other algorithms in the simulation of larger SGs (decrease of n_1).

In contrast, the main strength of our algorithm is a decrease of n_3 due to its parallel structure. Our algorithm is weaker than them in simulating nodes due to its over-determined matrix in M-step. In fact, while patch-based algorithms (Mahmud et al., 2014) find only a match for a bunch of pixels (a patch) in the simulation, our algorithm finds more than one match for every pixel of the simulation image. Thus, it is superior to the other methods in speed parameter only in large TI. This feature is shown in the second and third rows of Table 4. Small TI has 100 × 100 and the large TI has 1000 × 1000 size. Here all the algorithms are implemented through 3 levels of the Gaussian pyramid.

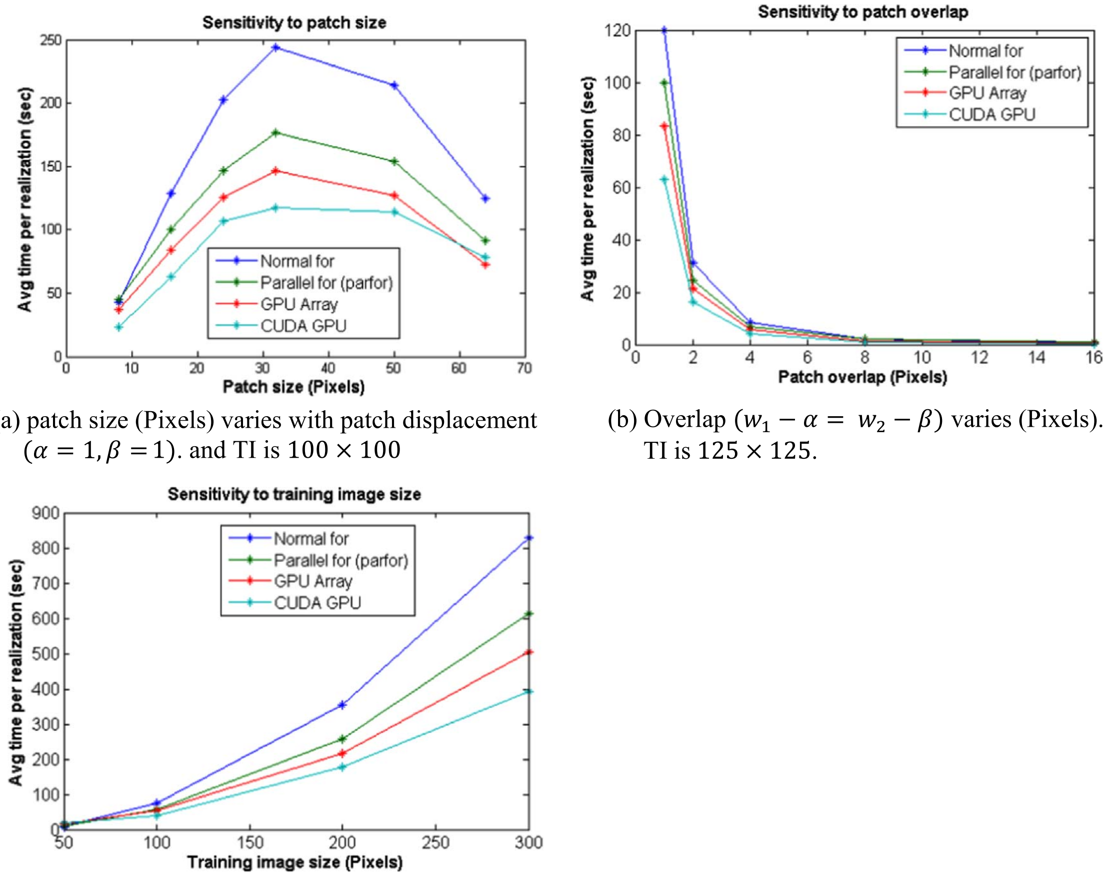
We have also tested three different parallel strategies. First of all,

Table 4
Average computational time and Mean Square Error (MSE) of different MPS methods in 100 realizations.

Case	Method	Percentage of hard data	Bunch DS (Rezaee et al., 2013)	DS (Mariethoz et al., 2010)	Quilting (Mahmud et al., 2014)	PCTO_SIM (CUDA GPU)
	Time (second)	2D small TI	18.9324	223.0004	5.5847	180.5609
		2D large TI	424.1230	3505.3596	410.75	389.3795
Case 1 *TI* = *Test*	Normalized MSE of hard data	0.1	0.3519	0.1259	0.0000	0.0015
		2	0.1617	0.1072	0.0000	0.0010
	Normalized MSE of whole image	0.1	0.4098	0.4161	0.1264	0.2853
		2	0.3111	0.3256	0.0264	0.0853
Case 2 *TI* ≠ *Test*	Normalized MSE of hard data	0.1	0.2481	0.1692	0.0101	0.0046
		2	0.3770	0.2972	0.0187	0.0060
	Normalized MSE of whole image	0.1	0.3182	0.3150	0.2820	0.2567
		2	0.4133	0.3956	0.0374	0.0108

the parallel for (*parfor command*) in MATLAB is used which uses the different cores of the CPU, e.g. in 100 iterations of command in a Quad-core CPU it divides 100 iterations into four 25 iterations and devotes each of them to each core simultaneously. Our second strategy uses the GPU array of MATLAB. Some functions in MATLAB support the GPU array. They execute faster with the GPU input array than normal input array. Our third strategy uses the CUDA kernel in C language to execute the main parallel core of our algorithm very fast. It devotes different iterations to different cores of GPU. Our experimental result shows that the CUDA kernel strategy has the best result (Fig. 11c).

We have also computed the execution time of our algorithm with a different patch size and overlap size (Fig. 11) per realization per iteration. Increasing the patch size increases our computational cost because larger patches have larger pixels to be compared with patches of the TI. This increase continues until it reaches a maximum (Fig. 11a) and then we have a decrease. This is because of the fact that a very large patch will decrease the number of patches in the TI, which is supposed to be compared. Increasing the patch displacement decreases our computational cost (Fig. 11b). This is due to the fact that a larger patch displacement will create less equations. These equations are needed to solve the optimization cost function for obtaining the fixed simulation nodes of our algorithm. Therefore, controllability of the patch displacement (or its reverse concept patch overlap) allows our proposed algorithm to decrease n_3 besides n_4.



(a) patch size (Pixels) varies with patch displacement ($\alpha = 1, \beta = 1$). and TI is 100 × 100

(b) Overlap ($w_1 - \alpha = w_2 - \beta$) varies (Pixels). TI is 125 × 125.

(c) TI size varies with our different parallel strategies with patch size of 16 × 16 and with patch displacement ($\alpha = 1, \beta = 1$).

Fig. 11. Execution time for three different parallel strategies, unconditional simulation on grid 100 × 100. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

6. Validation of the results

Evaluation of the MPS method is a challenging task of geostatistic simulation. Evaluation has also remained unsolved in texture synthesis (as a dual concept of MPS in computer graphics) while some efforts, especially for near-regular texture, have been made to introduce some criteria for proper texture synthesis and modeling (Lin et al., 2004, 2006; Liu et al., 2004). Visual judgment is still the best criterion for good texture modeling (Mirmehdi et al., 2008). Therefore, we have used visual judgment as one criterion to evaluate our results similar to other MPS methods (Mariethoz et al., 2010; Honarkhah and Caers, 2010). In multiple point simulation, a few criteria like variogram, connectivity reproduction (Rezaee et al., 2013), and hard data preservation (Rezaee et al., 2013) have been used in the literature. Tan et al. (2014) have proposed an analysis of distance (ANODI) for this purpose. However, their method is very good for more simple images like the sand and shale image of Fig. 7a. Although none of these criteria are perfect, they can be used as some intelligent machine-based approaches. We have also used these criteria to show the ability of our proposed method. We have also introduced a new criterion for a comparison of different MPS methods. This is the minimum square error of the whole test image.

6.1. *Interpolation capability (self-enrichment capability)*

One of the main features of our algorithm is its self-enrichment capability. Our algorithm is able to synthesize new patches, which it has not seen in the training database. It creates these patches through an average of the available patches of the TI. This capability makes our algorithm able to enrich a sparse training database to a richer one. This is the reason that our algorithm has better results in handling large missing data in comparison to other methods. In other words, our algorithm is able to work with the same precision as other methods with a less dense database so it can search in a smaller portion of the available database. This feature increases the speed of our algorithm.

In MPS, conditional simulation researchers want to learn a model from the TI and then reconstruct a test image with as few sparse data as possible. We also want diversity in our different simulations. However, in most previous MPS simulations, the test image in conditional simulations is not explicitly shown and it has been overlooked (Mariethoz et al., 2010; Abdollahifard and Faez, 2012, 2013; Mahmud et al., 2014; Rezaee et al., 2013; Zhang et al., 2006; Honarkhah and Caers, 2010). Therefore, the ability of their algorithm for image reconstruction cannot be exactly evaluated. In this paper, we have shown both our training and test images.

We have also shown our results both with the same training and test images (Fig. 12) and different training and test images (Fig. 13) from the same category of texture. In fact, a good MPS method must be able to reconstruct a test image perfectly with as few sparse hard data as possible. Different training and test images are one of the major challenges of MPS methods and all the algorithms try to have good results in this case. The strength of different MPS algorithms has been compared with ours for different amounts of hard data.

Patch size is the same (16×16) for all algorithms and its best size is selected. For all the experiments, training and test image size are 100×100. The implementation code is selected from the author's site (DS[3] (Mariethoz et al., 2010), Bunch DS[4] (Rezaee et al., 2013), MS-CCSIM (Tahmasebi et al., 2014) and Quilting[5] (Mahmud et al., 2014)). For Quilting (Mahmud et al., 2014) and MS-CCSIM (Tahmasebi et al., 2014) patch overlap is 5. A search is also done in the whole image. The number of level of multi-resolution reconstruction is 1. The number of candidate patterns from which the final pattern is selected is 5 and the co-template size is (Comunian et al., 2011) as default values. The $n = 40$, $f = 1$, $t = 0$ and distance criterion is euclidean for Advanded DS (A-DS).

6.1.1. *Similar simulating and training images*

Fig. 12 shows the result of the first experiment. We have done this experiment to test our algorithm in an ideal situation. A black border (Fig. 12w to ab) is added to the image due to the padaaray strategy of our algorithm implementation in MATLAB. This border does not change the image size and simply can be ignored. It is observable that DS (Mariethoz et al., 2010) and Bunch DS (Rezaee et al., 2013) have a good result at 10%. They do not honor the hard data even in the lower percentage of hard data. This is while Quilting (Mahmud et al., 2014) has a precise result with only 1% of hard data. Our method has good results with 2% of hard data. Both Quilting and our method have honored the hard data even at 0.1% of hard data. The superiority of the Quilting method to our method is due to the fact that it chooses the simulation patches exactly from the TI and in the case of similar training and test images it would be the best strategy.

DS (Mariethoz et al., 2010) and Bunch Ds (Rezaee et al., 2013) act much better in dense hard data than sparse hard data. This issue will be described now. In dense sampling, there are always a few hard samples in every large enough patch of SG. Therefore, only one accurate patch will be found as a match in TI. But if we sparsely sample the TI, there are some patches which do not encompass any hard data. Therefore, DS chooses a random patch from TI and pastes it to the SG. The place of this patch in SG can be different from its original place in TI. Then, based on the serial structure of DS (Mariethoz et al., 2010) and Bunch Ds (Rezaee et al., 2013), all the other consecutive patches and pixels will be simulated based on these previously synthesized soft data and the error will be propagated. Advanced DS (Mariethoz et al., 2010) has solved this problem to some extetnt due to its variable window size strategy however; its result is not still perfect.

6.1.2. *Different simulating and training images*

Fig. 13 has shown the result of the second experiment. It is observable that DS and bunch DS (with random simulation path) do not have a good result even at 10% of the data. They do not honor the hard data even at 10% of the data. This is due to their random patch selection strategy from the TI. In fact, the error of wrong choice of a patch will be propagated in the entire image through simulation. The Quilting method does not have a good result. At its best result (10% of hard data) it has simulated the main structure of the test image while there are lots of noises and scattered blobs. This is while our method has a perfect result at 2% of the hard data. It has created a connected channel while honoring the hard value even at 0.1% of the hard data.

Our superior result to the well-known method of MPS is due to the interpolation capability of our algorithm. The reason that DS **(**Mariethoz et al., 2010**)** and Bunch DS **(**Rezaee et al., 2013**)** have large error rates in honoring hard data is that these methods want to match a patch from the TI to directly simulate the image. Sometimes the TI is not so rich to have a good match patch to honor all the hard data of the SG. Therefore, some errors will be created. However, the self-enrichment capability of our algorithm has allowed it to create some new patches that honor the hard data well. The Quilting method (Mahmud et al., 2014) has done it through cutting the TI patch in minimum cut which can honor hard data. Although our method is visually good at reconstruction of the test image, it also depends on the TI. Therefore, if the training and test image belong to a different category of textures (Mirmehdi et al., 2008), it does not work.

For DS (Mariethoz et al., 2010), Bunch DS (Rezaee et al., 2013) and Quilting if the coefficient of hard data is considered low ($\lambda \approx 1$), the continuity of pattern and soft data coherence becomes better but the hard data are not met well. On the other hand, if the coefficient of hard data is considered high ($\lambda > 5$), the hard data are honored much better but the continiuity of patterns and soft data coherence is bad (Fig. 13h–v).

[3] http://www.minds.ch/gm/downloads.htm

[4] https://github.com/HassanRezaee/MPS-Bunch-DS

[5] https://github.com/juliohm/ImageQuilting.jl and http://www.minds.ch/gm/IQ.htm

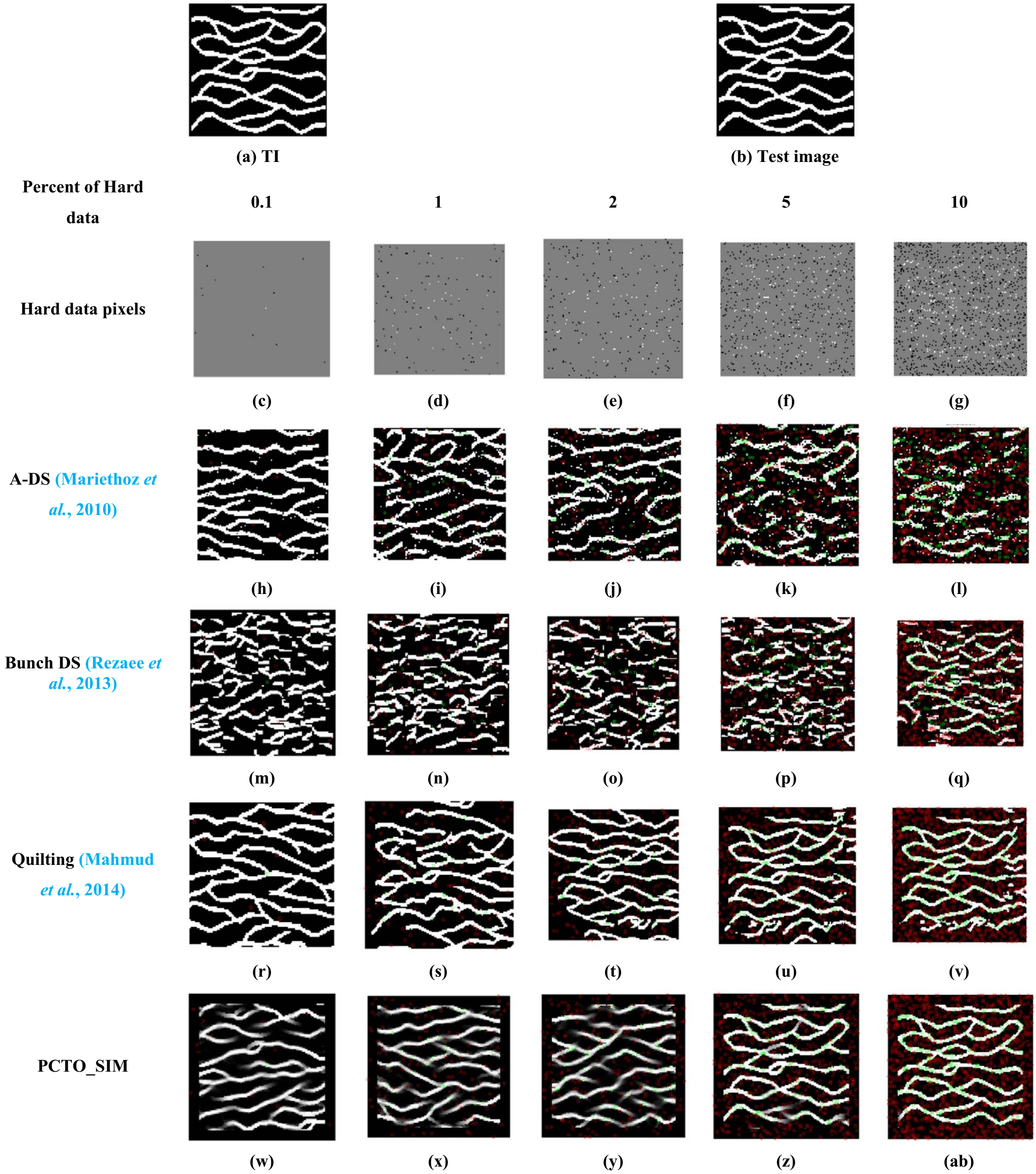


Fig. 12. Result of our conditional simulation in comparison with some other MPS methods with different hard data ($\lambda = 20$). The training and test image are the same. Our training and test image size are 100×100. Patch size is 16×16. Red and green circles are 0 and 1 hard data. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

We have also compared the E-type and variance of four MPS methods (Fig. 14). Our method has preserved the continuity of channels very well while honoring the hard data. Quilting has an E-type with zero variance. Therefore, it has a major problem. This method is biased to a single result, so it has lack of uncertainty. In fact, this method tries to find the best match exactly in the TI. Therefore, for some hard data, the same match will always be found for every realization. This is while our algorithm starts from a different random initial image every time so it simulates different results in its every realization while honoring hard data. Therefore, our algorithm does not have the low diversity problem of Quilting. The square patch in Fig. 14b has zero variance due to the same reason. In fact, the same match has always been found for this patch.

6.2. Variogram reproduction

One of the two point statistics that are used in the literature for evaluating the MPS result is variogram (Rezaee et al., 2013). Our non-threshold experimental result for unconditional simulation in Fig. 15a, b shows the similarity between the variogram of different realizations

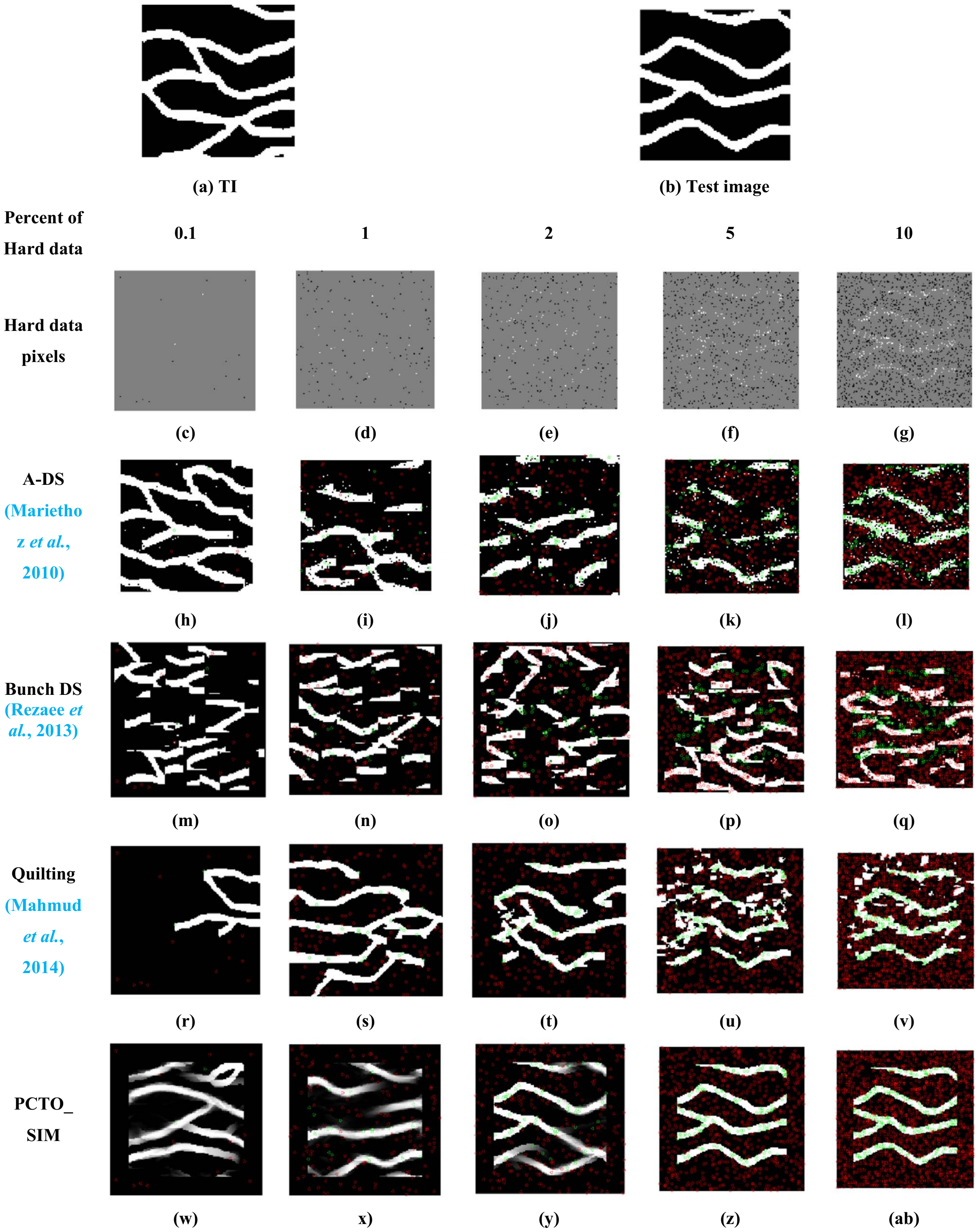


Fig. 13. Results of our conditional simulation in comparison with some other MPS methods with different hard data ($\lambda = 20$). Training and test images are chosen from different parts of Fig. 7a. Red and green circles are 0 and 1 hard data. Patch size is 32×32. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

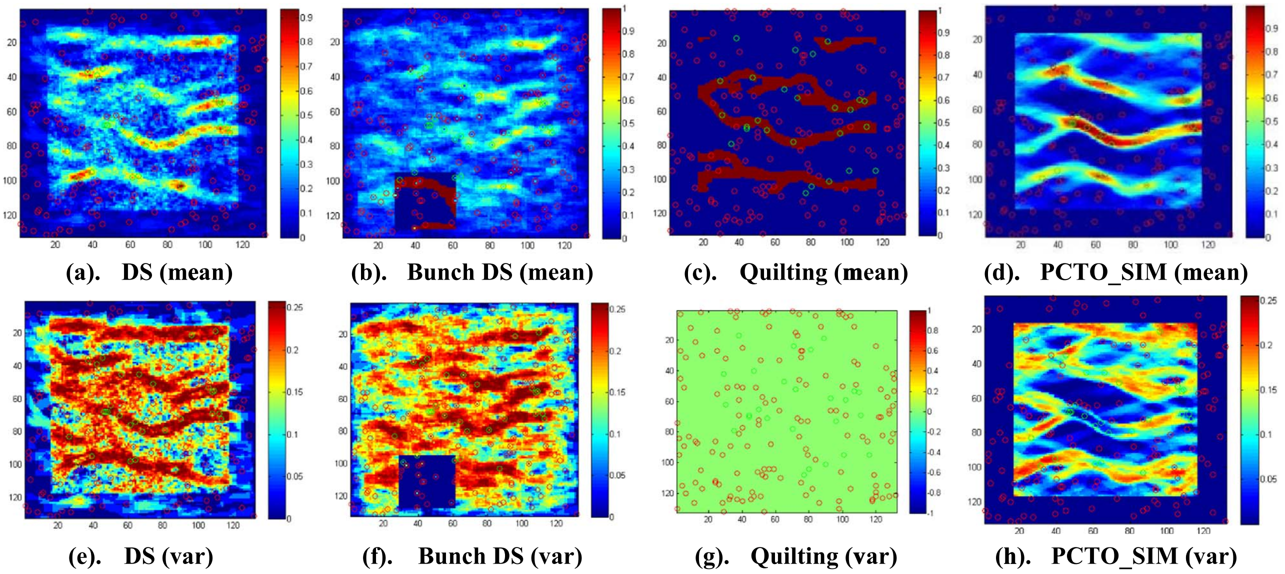
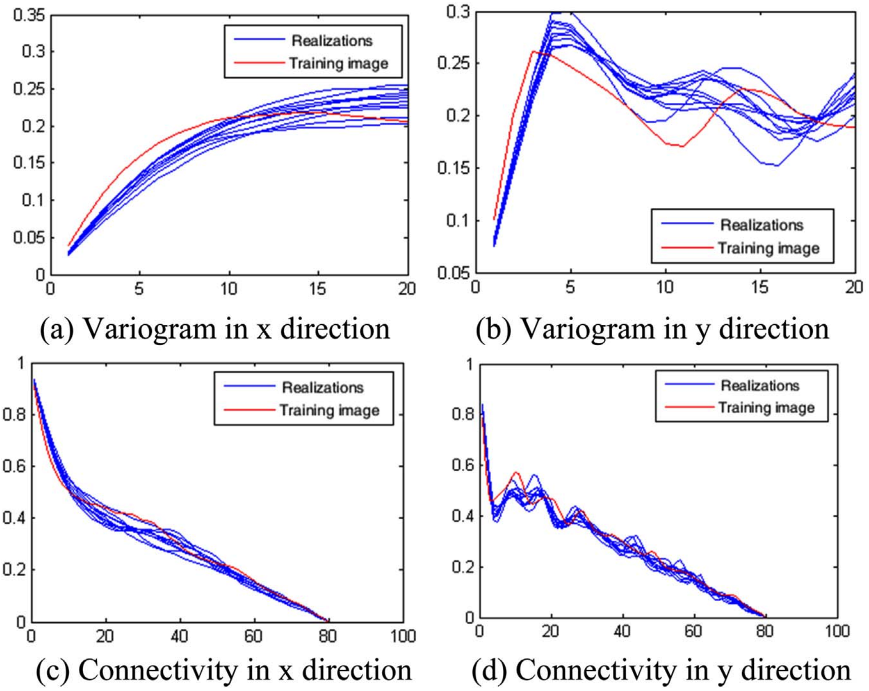

(a). DS (mean)
(b). Bunch DS (mean)
(c). Quilting (mean)
(d). PCTO_SIM (mean)
(e). DS (var)
(f). Bunch DS (var)
(g). Quilting (var)
(h). PCTO_SIM (var)

Fig. 14. E-type and variance of four MPS methods in 33 iterations with 1% hard data of Fig. 13b. Our algorithm has a good e-type shape for channel continuity preservation while it has a good diversity simultaneously. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

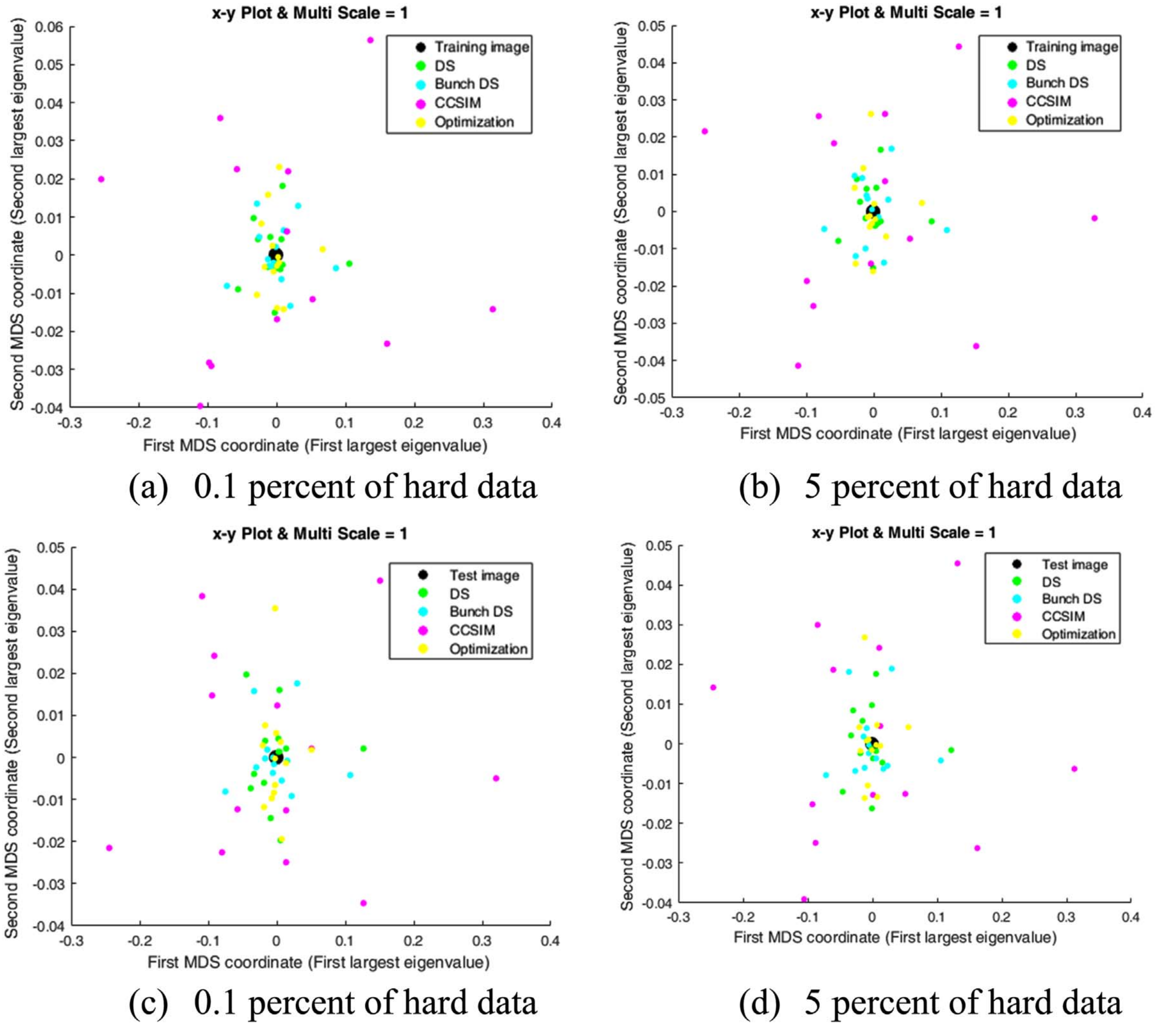


(a) Variogram in x direction
(b) Variogram in y direction
(c) Connectivity in x direction
(d) Connectivity in y direction

Fig. 15. In this figure, the variogram and connectivity of the TI with different realizations of our algorithm are compared. The TI is Fig. 7a. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Table 5
ANODI scores of DS (Mariethoz et al., 2010), Bunch DS (Rezaee et al., 2013), Quilting (Mahmud et al., 2014) and thresholded PCTO-SIM in presense of hard data.

Ranking	DS		Bunch DS		Quilting		Thresholded PCTO-SIM	
Hard data	0.1%	5%	0.1%	5%	0.1%	5%	0.1%	5%
Uncertainty space (between)	0.7778	0.7397	1.1806	1.1527	3.9263	4.3733	1	1
Pattern reproduction (within)	2.5981	2.3189	2.6560	2.4395	6.1680	6.0934	1	1
Total (between/within)	0.2994	0.3190	0.4445	0.4725	0.6366	0.7177	1	1



(a) 0.1 percent of hard data

(b) 5 percent of hard data

(c) 0.1 percent of hard data

(d) 5 percent of hard data

Fig. 16. Comparison of four MPS methods through MDS of Tan et al. Tan et al. (2014). Only 12 samples for every method are shown for better visual presentation. First and second rows are compared with TI and test image consequently. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

of our algorithm with its TI in both x and y direction. The small difference of variogram in simulating results with the TI is due to the blurring feature of our algorithm, but it cannot be numerically evaluated. The selected image is selected from an unconditional simulation with threshold=0.5. As it was mentioned in section 5.1.2, the similarity of the SG with TI will be improved in conditional simulation. Fig. 15a shows stronger short-range continuities in the realizations as compared to the TI. This is probably due to averaging features of our algorithm. This averaging decreases noise and makes the patterns more similar to each other. In fact, it is more probable to have similarity between n times average of m random variables than m random variables alone.

6.3. Connectivity reproduction

A good SG must have a good connectivity reproduction. To evaluate the connectivity of the SG, a connectivity function will be considered as in (Rezaee et al., 2013). Our connectivity functions are defined as the probability of 2 nodes separated by a lag along a certain direction to be connected (Renard and Allard, 2013). Fig. 15c, d illustrates the connectivity functions for both realizations and the TI. This figure shows a close similarity between realizations and TI.

6.4. Hard data preservation

The ability of our algorithm to preserve hard data is evaluated in Table 4. Two criteria are considered. First is the Normalized Minimum Square Error (MSE) of hard data between the SG and sampled image. This criterion shows how much an algorithm is able to honor the hard data.

Second is the Normalized MSE of the whole image's pixel (Normalization is done based on the whole image pixels' number and maximum gray-level). This criterion computes how much the SG is similar to the test image. This new criterion is proposed for the first time in this paper to evaluate the performance of the MPS methods in simulating the entire image with only a sparse portion of data. It is observable that in a case of different training and test images, our algorithm outperforms the other MPS methods. The training and test images size is 100 × 100, while patch size is 16 × 16. Green and red circles in Figs. 12 and 13 show the ability of our algorithm to preserve hard data. It is observable that our algorithm has honored hard data even at 0.1%.

6.5. Analysis of distance (ANODI)

Histogram and variogram are popular traditional criteria for evaluating MPS methods. These methods show good result for some pixel-based and patch-based methods due to their semi-verbatim copying strategy (DS (Mariethoz et al., 2010), Bunch DS (Rezaee et al., 2013), MS-CCSIM (Tahmasebi et al., 2014) and Quilting (Mahmud

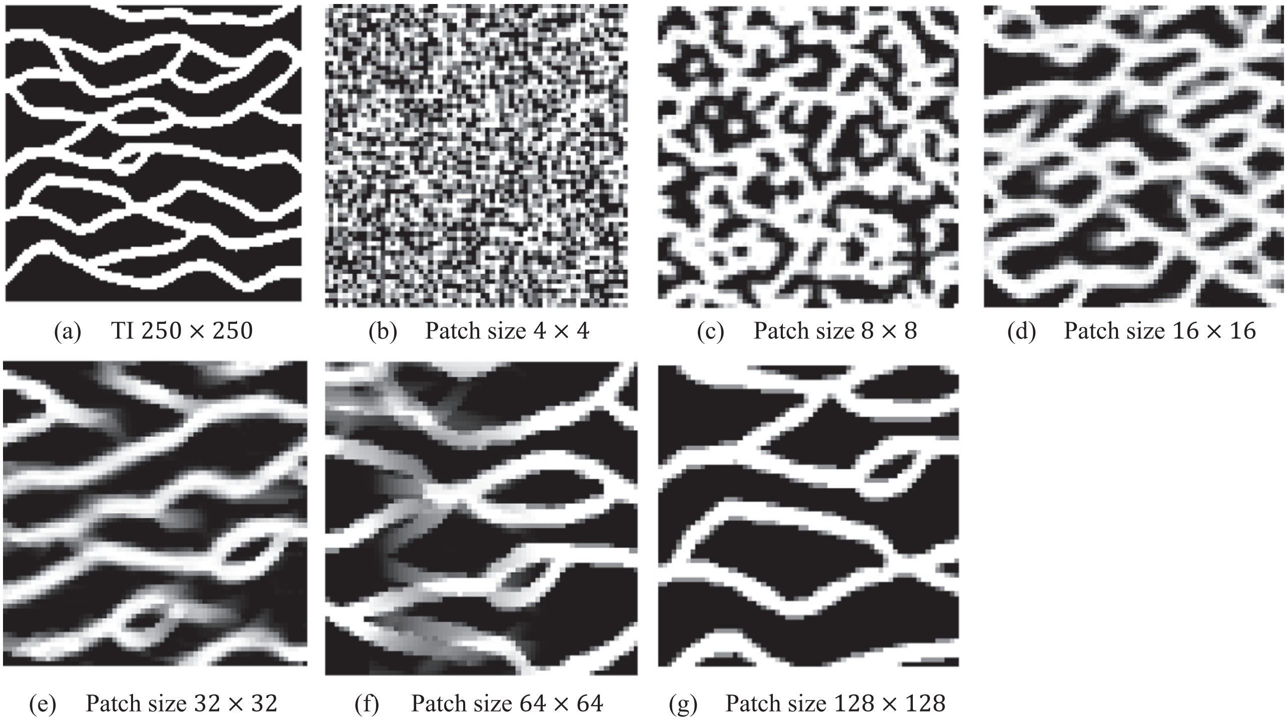

(a) TI 250 × 250
(b) Patch size 4 × 4
(c) Patch size 8 × 8
(d) Patch size 16 × 16
(e) Patch size 32 × 32
(f) Patch size 64 × 64
(g) Patch size 128 × 128

Fig. 17. Effect of patch size on unconditional simulation. The SG is 160 × 160 and all the results are shown after 5 iterations with patch displacement ($\alpha = 1, \beta = 1$).

et al., 2014)). However, our PCTO-SIM as a new category of MPS methods (optimization-based method) needs non-biased general criterion. Therefore, here a dimension reduction technique has used.

Tan et al. (2014) have claimed that a good MPS method must be able to create realizations that are as similar to the TI as possible (within-realization variability). At the same time, they must also be as different as possible from each other (between-realization variability). Therefore, the relative ratio of these two variances is their final criterion. They have also used Multidimensional scaling (MDS) as one of the dimension reduction algorithms for their second criterion.

In fact, two variances can be defined in evaluating MPS methods. The variance around the hard data must be as low as possible to honor the hard data (first variance). On the other hand, total variance of the image (second variance) must be as high as possible without losing the continuity of the image structure (here connectivity of the channels).

We have compared our method with DS **(**Mariethoz et al., 2010**)**, Bunch DS **(**Rezaee et al., 2013**)**, and Quilting (Mahmud et al., 2014) with 50 realizations. The result of ANODI score[6] with both sparse (0.1%) and dense (5%) conditional simulation have also been shown in Table 5. TI is Fig. 13a and patch size is 32 × 32. Our method has lower pattern reproduction than Bunch DS **(**Rezaee et al., 2013**)**, and Quilting (Mahmud et al., 2014) but it has the best (largest) total variance. In fact, these methods have obtained better uncertainty space and pattern reproduction by the cost of bad pattern continuity (Fig. 13). This is while Yang et al. (2016) have obtained their good total variance only when they have used the TI database enlargement strategy. This is probably due to the limitations of their local search strategy. Clearly, enrichment of TI can make our algorithm much better than now.

The results of the MDS dimension reduction strategy (Fig. 16a and b) show the goodness of our method. The SG samples of our algorithm are very close to TI and it also has the lowest variance in the first MDS coordinates (largest eigenvalue).

Tan et al. (2014) have not considered the test image in their ANODI criteria. We have also introduced it as a new criterion (MDS with test image) for evaluating the MPS method. We have compared our results with the other MPS methods. The results show the goodness of our method (Fig. 16c and d). Our method has honored hard data well and it has low first variance and high second variance, simultaneously.

7. Sensitivity tests

In this part, the sensitivity of our algorithm to its different parameters will be evaluated. For simplicity only 2D unconditional simulation is considered.

7.1. Sensitivity to patch size

In this part a few experiments on patch size are carried out. It is observable that the small patch size cannot contain the main structure of the TI (Fig. 17b). On the other hand, large patch size does verbatim copying of some parts of TI (Fig. 17g). Smaller patch size also creates more abundant white facies in SG than TI (Fig. 8h–k).

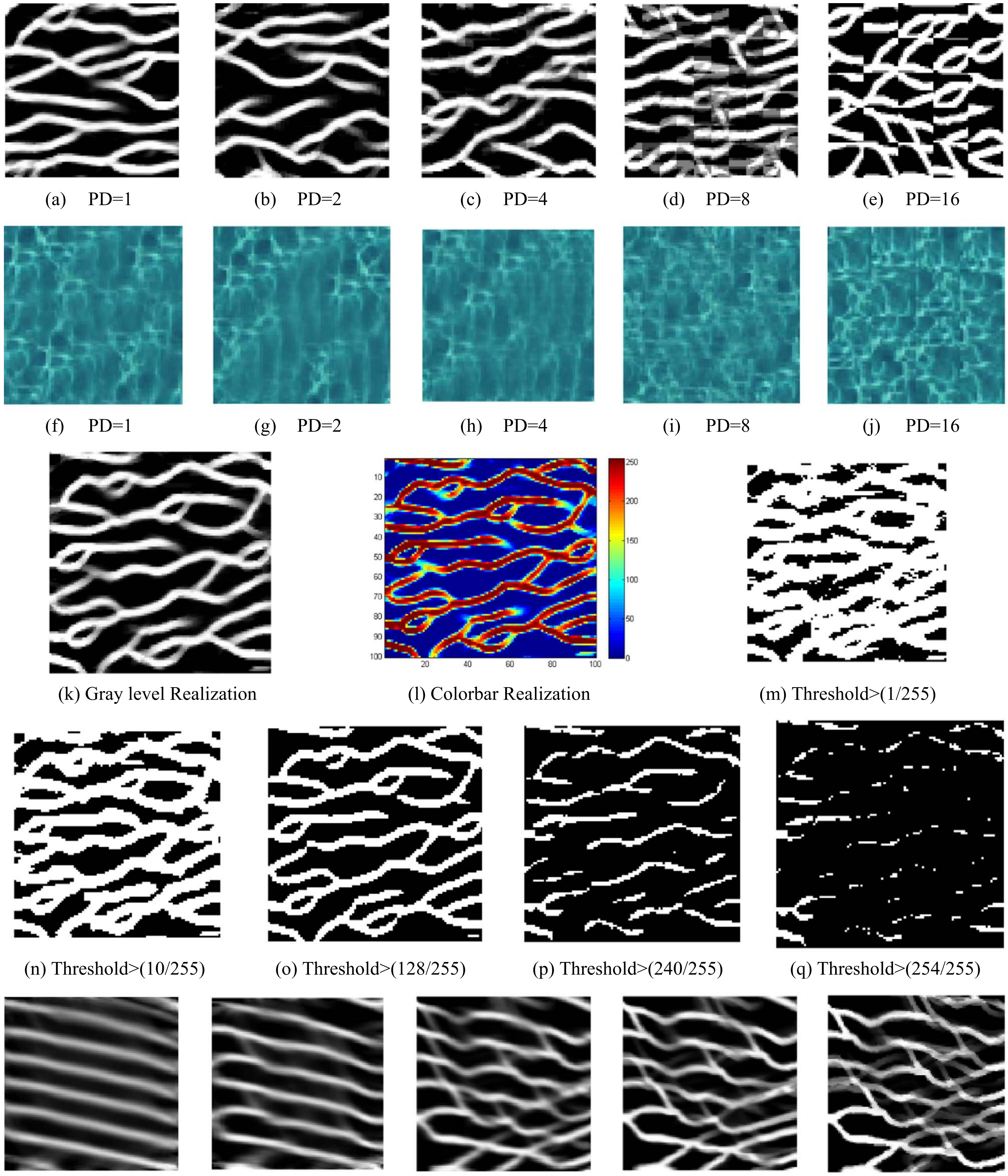
7.2. Sensitivity to patch overlap

In this part a few experiments on patch displacement (PD) (α, β) are carried out. The high patch overlap ($w_1 - 1,\ w_2 - 1$) or low patch displacement ($\alpha = 1, \beta = 1$) smoothens the image (Fig. 18a and f). It also has the highest computational cost because it has the largest equation for a fixed number of equations. The low patch overlap (α=patch size_x,β=patch size_y) will create a blocky image (Fig. 18e and j) and it has the lowest computational cost. Therefore, there is always an optimum value for patch overlap. Empirically, we have observed that one-tenth of the patch size has a good quality with proper speed.

7.3. Sensitivity to number of iterations

Different iterations of our algorithm on a gray-scale and color images are shown in Fig. 2. Our algorithm starts from an initial random image, and then after only one iteration our image gets the global structure of the TI. After two iterations it obtains its main structure. Other subsequent iterations increase the accuracy of our algorithm.

[6] https://github.com/SCRFpublic/ANODI



(a) PD=1 (b) PD=2 (c) PD=4 (d) PD=8 (e) PD=16

(f) PD=1 (g) PD=2 (h) PD=4 (i) PD=8 (j) PD=16

(k) Gray level Realization (l) Colorbar Realization (m) Threshold>(1/255)

(n) Threshold>(10/255) (o) Threshold>(128/255) (p) Threshold>(240/255) (q) Threshold>(254/255)

(r) $r = 1\%$ (s) $r = 5\%$ (t) $r = 10\%$ (u) $r = 30\%$ (v) $r = 100\%$

Fig. 18. Effect of patch displacement (PD) on unconditional simulation is shown in Figure a-j. SG is 128×128 and patch size is 32×32. Effect of threshold on our spectrum-like (fuzzy-like) realization (Figure k) is shown in Figure l-q. The gray level is in the range of 0–1. The result of the search in different r random percentages of the TI is shown in Figure r-v. Patch size is 64×64 and the simulated image is 250×250. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Adding some other terms like the gradient of the image can increase the accuracy of our algorithm. These terms can be chosen based on the specific structure of our TI heuristically. Error plots (Fig. 2g, n) also show that our algorithm will have low variability after its second iteration.

7.4. Sensitivity to threshold

Our algorithm has a spectrum-like (Fig. 18k and l) result. In other words, its result differs from 0 to 255. Some channels are strong while others are weak. This allows the user to choose between dense and highly connected channels (Fig. 18m) or sparse and lowly connected channels (Fig. 18q). Normally, choosing a threshold with $T = 0.5$ (half of the maximum gray-level) yields good results. Some morphological actions in image processing can give a proper highly connected channel with the same width as the TI.

7.5. Sensitivity to replicate r percent

In this part we will evaluate the sensitivity of our algorithm on the richness of the TI. Fig. 18r–v has shown the result of our algorithm through a search on different r percent of the TI. It is observable that searching at 10% of the TI will have the same result as 100%. This means that our algorithm has a good interpolation capability. In fact, our algorithm is able to create patches that have never been seen in the TI by averaging the observed patches.

8. Conclusion and future work

In this paper, the optimization paradigm is used to model and synthesize the Multiple-point Geostatistics. Our energy term tries to make the SG as similar as possible to the TI while honoring hard data and preserving the continuity of the image structure. The self-enrichment capability of our algorithm makes it possible to work with a highly sparse database and have a better result in reconstructing the structure of the SG in comparison to DS, Bunch DS, and Quilting methods. It is also able to create new patches, which have not been seen in the TI even in dense grids. The result of our algorithm has a fuzzy (spectrum) judgment. Thus, our result is similar to the expectation results of a few number of realizations of other algorithms only with one realization. The fuzzy result of our algorithm shows the uncertainty values in every region of the SG. Thus, it can be interpreted as a fuzzy map (uncertainty map).

The vectorization structure of our algorithm makes it possible to simulate the hydrological and Geostatistical phenomena in any dimension even higher than 3 dimensions. Our algorithm is also able to handle categorical simulation. Our algorithm defines an over-determined system of equations to solve its cost function. It makes the computational cost of our algorithm even higher than the pixel-based MPS methods. So our algorithm is slower than the other MPS methods in the simulating phase. However, the totally parallel main core of our proposed algorithm enables it to be implemented via GPU. Therefore, our algorithm is faster than the other MPS methods in searching exhaustively in training the image database. The interpolation capability of our algorithm enabled it to create a richer database from our accessible database by mixing the existing patches of the TI.

Our primary cost function can open a new field in MPS simulation based on the energy concept. This cost function is susceptible to be extended later for a general TI or for a specific type of TI, adaptively. These terms can increase the accuracy of our texture modeling techniques. Some other convex cost functions can also be proposed later for MPS simulation. Lots of regularization methods (Bishop, 2006) exist which have not been used in hydrological and Geostatistical fields. They can be added to our cost function for better accuracy. Different types of optimization techniques can also be used later to solve these cost functions. Different definitions of cost functions like sparse theory (Elad, 2010) and sparse modeling of textures (Peyre, 2009, 2007; Tartavel et al., 2013) can also be used in MPS simulation through our newly proposed optimization-based paradigm.

Our method uses a simple averaging in its E-step, but different weighing approaches can be used for better accuracy in simulation. In the overlapping regions of the patches, our algorithm considers the average values of these patches in common pixels. It is also possible to consider any other transformation like the non-linear median filter for these values.

Acknowledgments

We thank the editor and anonymous reviewers for their critical review of this challenging paper. The esteemed reviewers have encouraged us to resubmit our paper after its first submission in Feb 18, 2015 (before the submission of similar paper GOSIM in 2016). This paper is scrutinized fourth times by four reviewers in one year of revision and many new points and experiments have been added to it based on their constructive comments.

Appendix A. Supplementary material

Supplementary data associated with this article can be found in the online version at doi:10.1016/j.cageo.2016.12.012.

References

Abdollahifard, M.J., 2016. Fast multiple-point simulation using a data-driven path and an efficient gradient-based search. Comput. Geosci. 86, 64–74.
Abdollahifard, M.J., Faez, K., 2012. Stochastic simulation of patterns using Bayesian pattern modeling. Springer, Computat. Geosci. 17, 99–116.
Abdollahifard, M.J., Faez, K., 2013. Fast Direct Sampling for Multiple-point Stochastic Simulation. Springer, Arabian Journal of Geoscience.
Abdollahifard, M.J., Ahmadi, S., 2016. Reconstruction of binary geological images using analytical edge and object models. Comput. Geosci. 89, 293-251.
Abdollahifard, M.J., Mariethoz, G., Pourfard, M., 2016. Improving in situ data acquisition using training images and a Bayesian mixture model. Comput. Geosci. 91, 49–63.
Aitokhuehi, I., Durlofsky, L.J., 2005. Optimizing the performance of smart wells in complex reservoirs using continuously updated geological models. J. Petrol. Sci. Eng. 48, 254–264.
Arpat, G.B., Caers, J., 2007. Conditional simulation with patterns. Math. Geol. 39, 177–203.
Ashikhmin, M., 2001. Synthesizing natural textures. In: Proceedings of the 2001 symposium on Interactive 3D Graphics, Presented at New York, NY, USA.
Barnes, C., Shechtman, E., Finkelstein, A., Goldman, D., 2009. PatchMatch: a randomized correspondence algorithm for structural image editing. ACM Trans. Gr.-TOG 28, 24.
Bishop, C.M., 2006. Pattern Recognition and Machine Learning. Springer.
Boucher, A., Kyriakidis, P.C., Cronkite-Ratcliff, C., 2008. Geostatistical solutions for super-resolution land cover mapping. Geosci. Remote Sens. IEEE Trans. 46, 272–283.
Boyd, S., Vandenberghe, L., 2004. Convex Optimization. Cambridge University Press.
Caers, J., Strebelle, S., Payrazyan, K., 2003. Stochastic integration of seismic data and geologic scenarios: a WestAfrica submarine channel saga. Lead. Edge 22, 192–196.
Caers, J., 2011. Modeling Uncertainty in the Earth Sciences: Wiley-Blackwell.
Chiles, J.-P., Delfiner, P., 2012. second edGeostatistics: Modeling Spatial Uncertainty 497. John Wiley & Sons.
Comunian, A., Renard, P., Straubhaar, J., Bayer, P., 2011. Three-dimensional high resolution fluvio-glacial aquifer analog–Part 2: Geostatistical modeling. J. hydrol. 405, 10–23.
Cressie, N., Wikle, C.K., 2011. Statistics for Spatio-temporal Data. John Wiley & Sons.
dell'Arciprete, D., Bersezio, R., Felletti, F., Giudici, M., Comunian, A., Renard, P., 2012. Comparison of three geostatistical methods for hydrofacies simulation: a test on alluvial sediments. Hydrogeol. J. vol. 20, 299–311.
Efros, A., Leung, T., 1999. Texture synthesis by non-parametric sampling. In: Proceedings of the Presented at the International Conference on Computer Vision.
Efros, A.A., Freeman, W.T., 2001. Image quilting for texture synthesis and transfer, presented at. the Proceedings of the 28th Annual Conference on Computer Graphics and Interactive techniques, Tec.
Elad, M., 2010. Sparse and Redundant Representations From Theory to Applications in Signal and Image Processing. Springer.
Feyen, L., Caers, J., 2006. Quantifying geological uncertainty for flow and transport modeling in multi-modal heterogeneous formations. Adv. Water Resour. 29, 912–929.
Gardet, C., Le Ravalec, M., Gloaguen, E. Pattern-based conditional simulation with a raster path: a few techniques to make it more efficient, Stochastic Environmental

Research and Risk Assessment, pp.1-18.

Ge, Y., Bai, H., 2011. Multiple-point simulation-based method for extraction of objects with spatial structure from remotely sensed imagery. Int. J. Remote Sens. 32, 2311–2335.

Golub, G.H., Loan, C.F. Van, 2012. Matrix Computations 3. JHU Press.

González, E.F., Mukerji, T., Mavko, G., 2007. Seismic inversion combining rock physics and multiple-point geostatistics. Geophysics vol. 73, R11–R21.

Goovaerts, P., 1997. Geostatistics for Natural Resources Evaluation. Oxford university press.

Goovaerts, P., 2000. Geostatistical approaches for incorporating elevation into the spatial interpolation of rainfall. J. Hydrol. vol. 228, 113–129.

Guardiano, F.B., Srivastava, R.M., 1993. Multivariate Geostatistics: Beyond Bivariate MomentsGeostatistics Troia'92. Springer, 133–144.

Hashemi, S., Javaherian, A., Ataee-pour, M., Khoshdel, H., 2014. Two-point versus multiple-point geostatistics: the ability of geostatistical methods to capture complex geobodies and their facies associations—an application to a channelized carbonate reservoir, southwest Iran. J. Geophys. Eng. 11, 065002.

He, X., Sonnenborg, T., Jørgensen, F., Jensen, K., 2013. The effect of training image and secondary data integration with multiple-point geostatistics in groundwater modeling. Hydrol. Earth Syst. Sci. Discuss. vol. 10, 11829–11860.

Hoffman, B.T., Caers, J., 2007. History matching by jointly perturbing local facies proportions and their spatial distribution: Application to a North Sea reservoir. J. Petrol. Sci. Eng. 57, 257–272.

Honarkhah, M., Caers, J., 2010. Stochastic simulation of patterns using Distance-Baseddistance-based pattern Modeling. Math. Geosci. 42, 487–517.

Hu, L., Chugunova, T., 2008. Multiple-point geostatistics for modeling subsurface heterogeneity: A comprehensive review. Water Resour. Res. vol. 44.

Huang, T., Lu, D.-T., Li, X., Wang, L., 2013. GPU-based SNESIM implementation for multiple-point statistical simulation. Comput. Geosci. 54, 75–87.

Huysmans, M., Dassargues, A., 2009. Application of multiple-point geostatistics on modelling groundwater flow and transport in a cross-bedded aquifer (Belgium). Hydrogeol. J. 17, 1901–1911.

Ingram, B., Cornford, D., 2010. Parallel Geostatistics for Sparse and Dense Datasetsin geoENV VII–Geostatistics for Environmental Applications ed. Springer, 371–381.

Jha, S.K., Mariethoz, G., Kelly, B.F., 2013. Bathymetry fusion using multiple-point geostatistics: Novelty and challenges in representing non-stationary bedforms. Environ. Modelli. Softw 50, 66–76.

Jha, S.K., Mariethoz, G., Evans, J.P., McCabe, M.F., 2013. Demonstration of a geostatistical approach to physically consistent downscaling of climate modeling simulations. Water Resour. Res. 49, 245–259.

Kalantari, S., Abdollahifard, M.J., 2016. Optimization-based multiple-point geostatistics: A sparse way. Comput. Geosci. 95, 85–98.

Keogh, K.J., Martinius, A.W., Osland, R., 2007. The development of fluvial stochastic modelling in the Norwegian oil industry: a historical review, subsurface implementation and future directions. Sediment. Geol. 202, 249–268.

Kessler, T.C., Comunian, A., Oriani, F., Renard, P., Nilsson, B., Klint, K.E., et al., 2013. Modeling fine-scale geological heterogeneity—examples of sand lenses in tills. Groundwater 51, 692–705.

Khodabakhshi, M., Jafarpour, B., 2013. A Bayesian mixture-modeling approach for flow-conditioned multiple-point statistical facies simulation from uncertain training images. Water Resour. Res. 49, 328–342.

Kopf, J., Fu, C.-W., Cohen-Or, D., Deussen, O., Lischinski, D., Wong, T.-T., 2007. Solid texture synthesis from 2d exemplars. in ACM Trans. Gr. (TOG), 2.

Kwatra, V., Essa, I., Bobick, A., Kwatra, N., 2005. Texture optimization for example-based synthesis. ACM Trans. Gr. 24, 795–802.

Kwatra, V., Schodl, A., Essa, I., Turk, G., Bobick, A., 2003. Graphcut textures: image and video synthesis using graph cuts. ACM Trans. Graph. 22, 277–286.

Le Coz, M., Genthon, P., Adler, P.M., 2011. Multiple-point statistics for modeling facies heterogeneities in a porous medium: The Komadugu-Yobe alluvium, Lake Chad basin. Math. Geosci. 43, 861–878.

Le Ravalec, M., Noetinger, B., Hu, L.Y., 2000. The FFT moving average (FFT-MA) generator: an efficient numerical method for generating and conditioning Gaussian simulations. Math. Geol. 32, 701–723.

Li, X., Mariethoz, G., Lu, D., Linde, N. Patch-based iterative conditional geostatistical simulation using graph cuts, Water Resources Research.

Lin, W.-C., Hays, J., Wu, C., Kwatra, V., Liu, Y., 2006. Quantitative evaluation of near regular texture synthesis Algorithms. presented at the Comput. Vis. Pattern Recognit. (CVPR).

Lin, W.-C., Hays, J., Wu, C., Kwatra, V., Liu, Y., 2004. A Comparison Study of Four Texture Synthesis Algorithms on Near-regular Textures, presented at the ACM SIGGRAPH 2004 Posters, Los Angeles, California.

Liu, Y., Caselles, V., 2013. Exemplar-based image inpainting using multiscale graph cuts. Image Process. IEEE Trans. 22, 1699–1711.

Liu, Y., Lin, W.-C., Hays, J., 2004. Near-regular texture analysis and manipulation. ACM Trans. Graph. 23, 368–376.

Mahmud, K., Mariethoz, G., Caers, J., Tahmasebi, P., Baker, A., 2014. Simulation of Earth textures by conditional image quilting. Water Resour. Res. 50, 3088–3107.

Mariethoz, G., Kelly, B.F., 2011. Modeling complex geological structures with elementary training images and transform-invariant distances. Water Resour. Res. 47.

Mariethoz, G., Lefebvre, S., 2014. Bridges between multiple-point geostatistics and texture synthesis: Review and guidelines for future research. Comput. Geosci. 66, 66–80.

Mariethoz, G., Renard, P., Straubhaar, J., 2010. The direct Sampling method to perform multiple point geostatistical simulations. Water Resour. Res. 46, 1–14.

Mariethoz, G. g, 2010. A general parallelization strategy for random path based geostatistical simulation methods. Comput. Geosci. 36.

Meerschman, E., Pirot, G., Mariethoz, G., Straubhaar, J., Van Meirvenne, M., Renard, P., 2013. A practical guide to performing multiple-point statistical simulations with the direct sampling algorithm. Comput. Geosci. 52, 307–324.

Michael, H., Li, H., Boucher, A., Sun, T., Caers, J., Gorelick, S., 2010. Combining geologic-process models and geostatistics for conditional simulation of 3-D subsurface heterogeneity. Water Resour. Res. 46.

Mirmehdi, M., Xie, X., Suri, J., 2008. Handbook of Texture Analysis. Imperial College Press.

Park, H., Scheidt, C., Fenwick, D., Boucher, A., Caers, J., 2013. History matching and uncertainty quantification of facies models with multiple geological interpretations. Comput. Geosci. 17, 609–621.

Peredo, O., Ortiz, J.M., 2011. Parallel implementation of simulated annealing to reproduce multiple-point statistics. Comput. Geosci. 37, 1110–1121.

peyre, G., 2007. Sparse Represent. Signals: represent. signal.: theory and Appl. Adapt. Sparse Represent..

Peyre, G., 2009. Sparse modeling of textures. J. Math. Imaging Vis. 34, 17–31.

Peyré, G., 2007. Sparse Modeli. Textures.

Peyre G., 2007. Non-negative sparse modeling of textures, presented at the Proceedings of the SSVM.

Pham, T.D., 2012. Supervised restoration of degraded medical images using multiple-point geostatistics. Comput. Methods Progr. Biomed. 106, 201–209.

Pyrcz, M.J., Deutsch, C.V., 2014. Geostatistical Reservoir Modeling. Oxford university press.

Ramanarayanan, G., Bala, K., 2007. Constrained texture synthesis via energy minimization. Vis. Comput. Gr. IEEE Trans. 13, 167–178.

Remy, N., Boucher, A., Wu, J., 2009. Applied Geostatistics with SGeMS: A user's guide. Cambridge University Press.

Renard, P., 2007. Stochastic hydrogeology: what professionals really need? Groundwater vol. 45, 531–541.

Renard, P., Allard, D., 2013. Connectivity metrics for subsurface flow and transport. Adv. Water Resour. 51, 168–196.

Renard, P., Straubhaar, J., Caers, J., Mariethoz, G., 2011. Conditioning facies simulations with connectivity data. Math. Geosci. 43, 879–903.

Rezaee, H., Mariethoz, G., Koneshloo, M., Asghari, O., 2013. Multiple-point geostatistical simulation using the bunch-pasting direct sampling method. Comput. Geosci. 54, 293–308.

Rezaee, H., Marcotte, D., Tahmasebi, P., Saucier, A., 2015. Multiple-point geostatistical simulation using enriched pattern databases. Stoch. Environ. Res. Risk Assess., 1–21.

Ringrose, P., Bentley, M., 2015. Reservoir Model Design. Springer.

Sahimi, M., 2011. Flow and Transport in Porous Media and Fractured Rock From Classical Methods to Modern Approaches. Wiley-VCH.

Sezgin, M., 2004. Survey over image thresholding techniques and quantitative performance evaluation. J. Electron. Imag. 13, 146–168.

Shen, J., Sun, H., Jia, J., Zhao, H., Jin, X., Fang, S., 2010. A unified framework for designing textures using energy optimization. Pattern Recognit. 43, 457–469.

Straubhaar, J., Renard, P., Mariethoz, G., Froidevaux, R., Besson, O., 2011. An improved parallel multiple-point algorithm using a list approach. Math. Geosci. 43, 305–328.

Strebelle, S., 2002. Conditional simulation of complex geological structures using multiple-point statistics. Math. Geol. 34.

Strebelle, S., Cavelius, C., 2014. Solving speed and memory issues in Multiple-Pointmultiple-point statistics simulation program SNESIM. Math. Geosci. 46, 171–186.

Strebelle, S., Payrazyan, K., Caers, J., 2003. Modeling of a deepwater turbidite reservoir conditional to seismic data using principal component analysis and multiple-point geostatistics. SPE J. 8, 227–235.

Szeliski, R., 2010. Computer Vision: Algorithms and Applications. Springer.

Tahmasebi, P., Sahimi, M., 2013. Cross-correlation function for accurate reconstruction of heterogeneous media. Phys. Rev. Lett. 110, 078002.

Tahmasebi, P., Hezarkhani, A., Sahimi, M., 2012. Multiple-point geostatistical modeling based on the cross-correlation functions. Comput. Geosci. 16, 779–797.

Tahmasebi, P., Sahimi, M., Caers, J., 2014. MS-CCSIM: accelerating pattern-based geostatistical simulation of categorical variables using a multi-scale search in Fourier space. Comput. Geosci. 67, 75–88.

Tahmasebi, P., Sahimi, M., Mariethoz, G., Hezarkhani, A., 2012. Accelerating geostatistical simulations using graphics processing units (GPU). Comput. Geosci. 46, 51–59.

Tamayo-Mas, E., Mustapha, H., Dimitrakopoulos, R., 2016. Testing geological heterogeneity representations for enhanced oil recovery techniques. J. Petrol. Sci. Eng. 146, 222–240.

Tan, X., Tahmasebi, P., Caers, J., 2014. Comparing training-image based algorithms using an analysis of distance. Math. Geosci. 46, 149–169.

Tartavel, G., Gousseau, Y., Peyré, G., 2013. Constrained Sparse Texture SynthesisScale Space Variational Methods Comput. Vision 7893. Springer, 186–197.

Vannametee, E., Babel, L., Hendriks, M., Schuur, J., de Jong, S., Bierkens, M., et al., 2014. Semi-automated mapping of landforms using multiple point geostatistics. Geomorphology 221, 298–319.

Wackernagel, H., 2003. Multivariate Geostatistics. Springer.

Wei, L.-Y., Lefebvre, S., Kwatra, V., Turk, G., 2009. State of the Art in Example-based TextureSynthesis. The Eurogr. Association.

Wei, L.-Y., Levoy, M., 2000. Fast texture synthesis using tree-structured vector quantization, presented at. the 27th International Conference on Computer Graphics and Interactive Techniques.

Wu, J., Zhang, T., Journel, A., 2008. Fast FILTERSIM simulation with score-based distance. Math. Geosci. 40, 773–788.

Xu, K., Cohen-Or, D., Ju, T., Liu, L., Zhang, H., Zhou, S., et al., 2009. Feature-aligned shape texturing. ACM Trans. Gr. (TOG) vol. 28, 108.

Yang, L., Hou, W., Cui, C., Cui, J., 2016. GOSIM: A multi-scale iterative multiple-point statistics algorithm with global optimization. Comput. Geosci..

Yoram, R., 2003. Applied Stochastic Hydrogeology. Oxford University Press.

Zahner, T., Lochbühler, T., Mariethoz, G., Linde, N., 2016. Image synthesis with graph cuts: a fast model proposal mechanism in probabilistic inversion. Geophys. J. Int. 204, 1179–1190.

Zhang, D., 2001. Stochastic Methods for Flow in Porous Media: Coping with Uncertainties. Academic Press.

Zhang, T., Switzer, P., Journel, A., 2006. Filter-Basedfilter-based classification of training image patterns for spatial Simulation. Math. Geol. 38, 63–80.