Math Geosci
https://doi.org/10.1007/s11004-018-9766-6

Fast and Interactive Editing Tools for Spatial Models

Julien Straubhaar[1] · **Philippe Renard**[1] · **Grégoire Mariethoz**[2] · **Tatiana Chugunova**[3] · **Pierre Biver**[3]

Received: 19 December 2017 / Accepted: 31 August 2018

Abstract Multiple point statistics (MPS) algorithms allow generation of random fields reproducing the spatial features of a training image (TI). Although many MPS techniques offer options to prescribe characteristics deviating from those of the TI (e.g., facies proportions), providing a TI representing the target features as well as possible is important. In this paper, methods for editing stationary images by applying a transformation—painting or warping—to the regions, similar to a representative pattern selected by the user in the image itself, are proposed. Painting simply consists in replacing image values, whereas warping consists in deforming the image grid (compression or expansion of similar regions). These tools require few parameters and are interactive: the user defines locally how the image should be modified, then the changes are propagated automatically to the entire image. Examples show the ability of the proposed methods to keep spatial features consistent within the entire edited image.

Keywords Self-similarity · Image transformation · Grid deformation · Multiple point statistics

✉ Julien Straubhaar
julien.straubhaar@unine.ch

[1] The Centre for Hydrogeology and Geothermics (CHYN), University of Neuchâtel, Rue Emile-Argand 11, 2000 Neuchâtel, Switzerland

[2] Institute of Earth Surface Dynamics (IDYST), University of Lausanne, UNIL-Mouline, Geopolis, 1015 Lausanne, Switzerland

[3] Geostatistics and Uncertainties, TOTAL SA, 64000 Pau, France

Published online: 12 September 2018

1 Introduction

In Earth Sciences, geostatistical simulation algorithms based on a conceptual model, such as multiple point statistics (MPS) techniques, are widely used for modeling the underground, landscapes or natural processes and for assessing the associated uncertainty. These methods allow the user to generate random fields reproducing the spatial structures present in a training image (TI), which must be given by the user. The TI can come from analog sites, satellite photographs, or simply be drawn by hand. MPS requires that the TI contains repeated patterns, possibly with some non-stationarity. Most MPS algorithms provide options to generate realizations presenting features that are not in the TI (e.g., orientation/dilation of the patterns, local/global facies proportions, etc.). Most often, this implies a compromise between the reproduction of such specific features and the "quality" of the realizations. When target properties are far away from what is found in the TI, one cannot predict what the MPS simulations will look like. Indeed, for example, considering a binary TI with channelized structures, imposing a target proportion of channels less than the proportion in the TI does not offer any control on how this target proportion will be honoured: decreasing the proportion of channels can be achieved by making the channels thinner or by reducing their number. Hence, it is preferable to modify the TI before performing simulations, such that it depicts the structures to be simulated as well as possible.

Motivated by the above concerns, the aim of this paper is to develop tools for adapting the TI and then allowing the user to better control further MPS simulations. Indeed, geological concepts and interpretations are inherently subjective. For this reason, translating geological concepts in the creation of TIs is a real practical problem and a longstanding limitation to the applicability of MPS. Thus, such editing tools bring practical solutions to this question. Moreover, as MPS methods are available for the simulation of categorical and continuous attributes, for example the direct sampling technique (Mariethoz et al. 2010), these two types of variables are considered.

In computer graphics, Brooks and Dodgson (2002) proposed methods for editing "textures" consisting of stationary images encoded with the three color channels, RGB. In this paper, we propose interactive tools for editing images which preserve their stationarity (i.e., the repeated structures), adapted for continuous images as well as for categorical ones (i.e., images defined by real (continuous) variables or discrete variables expressed by facies code representing, for example, the lithology in geological applications). The methods consist in applying the same transformation everywhere the image presents a similar structure to the one displayed in a pattern selected in the image itself by the user. Two main transformation operations are proposed: painting and warping. Painting consists in locally changing the values of the variables, with smooth transitions for continuous images, and allowing a modification of the facies code by existing code or new code for categorical images. The warping operation consists of local geometrical transformations—expansion or shrinkage of image regions. It is done by first applying a spatial deformation, and then by re-interpolating the variable onto the initial regular grid of the image. The technique is quite different and more sophisticated than mathematical morphology tools such as dilation and erosion operations, which do not act on the support of the image. It is also quite different from available methods for locally modifying properties of MPS simulations, such as block

constraints (Straubhaar et al. 2016), proportion trends (Mariethoz et al. 2015), or local affinities and orientations (Strebelle and Zhang 2005). For example, warping allows the user to make channels thinner without breaking them, which is not necessarily the case when performing successive erosions or by constraining MPS to a low proportion of channels.

Tahmasebi (2017) proposes a method to update existing geological models also based on underlying grid deformations, but with the purpose of correcting stochastic realizations not honoring conditioning data, which can happen, in particular, if patch-based algorithms are used. In this situation, problematic conditioning points are moved to target points where the variable is equal to the data values and a grid deformation is computed by using the method of thin-plate splines (Bookstein 1989). Contrary to the warping process we propose, the values at the new locations are brought back to the initial ones.

While the methods proposed in this paper can also be used to adapt a geological model, their main purpose is to provide flexible tools for editing conceptual models (TIs) in a consistent manner. Both operations—painting and warping—allow adaptation of the features of displayed structures, and modification of the distribution of the variable in the entire image. These editing processes are interactive: the user defines locally how to change the image, then the entire image is automatically updated by propagating these changes. Based on the self-similarity of the input image, the user can also control the strength of the transformation. A dissimilarity map defined at each pixel as a distance between the selected pattern and the pattern centered at that pixel is computed. Then, the transformation is applied over each region where the dissimilarity measure falls below a threshold. A small threshold implies a transformation of small intensity, and its relaxation results in increasing the effects of the transformation.

The paper is organized as follows. The principle of the proposed methods and the notion of dissimilarity, common to both types of transformation, are introduced in Sect. 2. The algorithms for painting and warping processes are presented in Sects. 3 and 4 respectively. In Sect. 5, additional examples for both techniques applied on two- and three-dimensional images are given as illustrations and the computational performance is discussed. Finally, a conclusion and perspectives are given in Sect. 6.

2 Defining Editing Regions in Self-Similar Images

The principle of the proposed methods consists in a semi-automated editing process, based on the self-similarity of an image defined by a variable $Z(x)$ on a regular grid. Considering a stationary image (i.e. depicting repeated structures), a specific location is selected and similar changes are automatically applied everywhere the structures are similar. Two types of transformation are considered: **painting**, which consists in directly modifying the values of the variable, and **warping**, which consists in a space deformation.

The proposed tools require in input: (i) to select a pattern in the image, (ii) to specify the transformation to be applied, and (iii) to give threshold value(s) for the dissimilarity to control the intensity of the transformation. The main computation steps are then to

map the dissimilarity to the selected pattern, and to apply the transformation (painting or warping) to the regions where the dissimilarity is below the given threshold.

Consider a pattern

$$d(x_0, \tau) = \{Z(x_0 + h_1), \ldots, Z(x_0 + h_N)\}, \tag{1}$$

selected in the image, where Z denotes the variable defined on each cell/pixel, x_0 the central cell of the pattern and $\tau = \{h_1, \ldots, h_N\}$ the set of (lag) vectors defining the geometry of the pattern. For each location x in the image grid G, a distance $D(d(x, \tau), d(x_0, \tau))$ between the pattern $d(x, \tau)$ centered at x and the reference pattern $d(x_0, \tau)$ is computed. Then, the dissimilarity map is defined at each cell $x \in G$ as

$$\mathrm{dis}(x) = \frac{D(d(x, \tau), d(x_0, \tau))}{\max_{y \in G} (D(d(y, \tau), d(x_0, \tau))}. \tag{2}$$

It consists in the relative dissimilarity with values in the range [0, 1], so $\mathrm{dis}(x) = 0$ means that the pattern centered at x is identical to the selected pattern, and $\mathrm{dis}(x) = 1$ that it is the pattern in the image having the highest distance from the selected pattern.

The distance D is defined as follows. If Z is a categorical variable, D is the proportion of mismatching nodes in the pattern, and if Z is continuous, the Root Mean Squared Error (RMSE) is used. (The Mean Absolute Error (MAE) could also be employed.) Note that extrapolations are applied at the borders of the images to obtain a smooth map. Illustrations are presented in Figs. 1a, b and 2a, b, for a categorical image (Fig. 1a from Allard et al. 2011) and a continuous image (Fig. 2a from Zhang et al. 2006), respectively.

3 Self-Similarity-Based Painting

Self-similarity-based painting consists in modifying the value of the variable in the regions similar to the selected pattern (reference pattern). Those regions are determined by using the dissimilarity map: the pixels where the dissimilarity is below a given threshold are modified. The ensemble of those pixels is called the matching region or matching pixels. The method requires only two input parameters: a threshold value t, and a target value z_{new}.

For categorical images, the new value z_{new} is simply assigned to the matching pixels: the variable $Z^\star$ in the output image is defined as

$$Z^\star(x) = \begin{cases} z_{\mathrm{new}} & \text{if } \mathrm{dis}(x) < t \\ Z(x) & \text{otherwise.} \end{cases} \tag{3}$$

With continuous images, the new value cannot be simply assigned to all the matching pixels, because this would result in very sharp interfaces at the borders of the matching region, where "jumps" of the variable would be observed. Hence, to ensure a smooth result in the case of a continuous variable, the output value at a pixel x is defined as

$$Z^\star(x) = Z(x) + \max\left(0, 1 - \frac{\mathrm{dis}(x)}{t}\right) \cdot (z_{\mathrm{new}} - Z(x)). \tag{4}$$

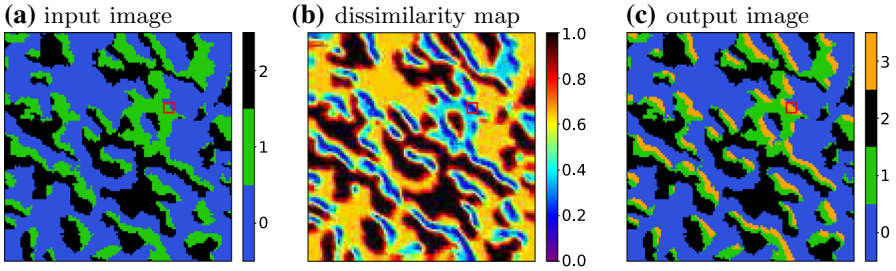

Fig. 1 Self-similarity-based painting on a categorical image (114 × 114): **a** input image; **b** dissimilarity map; **c** output image for $t = 0.3$ and $z_{\text{new}} = 3$. In **a**–**c**, the red square is the 5 × 5 reference pattern

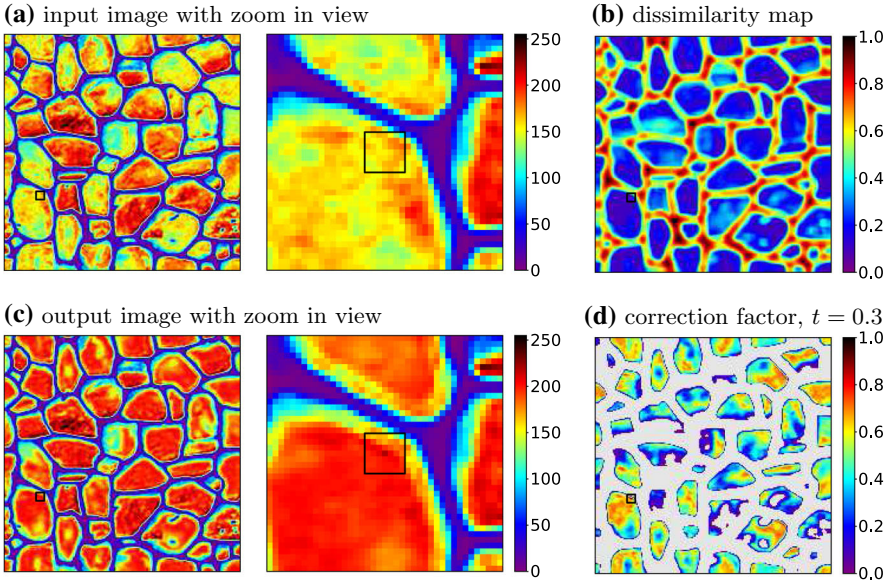

Fig. 2 Self-similarity-based painting on a continuous image (200 × 200): **a** input image ($Z(x_0) = 155$); **b** dissimilarity map; **c** output image for $t = 0.3$ and $z_{\text{new}} = 220$; **d** correction factor map (0 in the gray area). In **a**–**d**, the black square is the 7 × 7 reference pattern

Thus, for a matching pixel ($\text{dis}(x) < t$), the correction $z_{\text{new}} - Z(x)$ is multiplied by the factor $1 - \text{dis}(x)/t$, which has a linear dependence on the dissimilarity.

The painting process is illustrated in Figs. 1 and 2 for a categorical image and a continuous image, respectively. For the categorical case, the dissimilarity is computed based on the proportion of mismatching nodes in the pattern, and a new facies (in orange) is introduced. For the continuous case, the dissimilarity ($\text{dis}(x)$) is computed based on the RMSE over the pattern and the correction factor, $\max(0, 1 - \text{dis}(x)/t)$ from Eq. (4), is displayed.

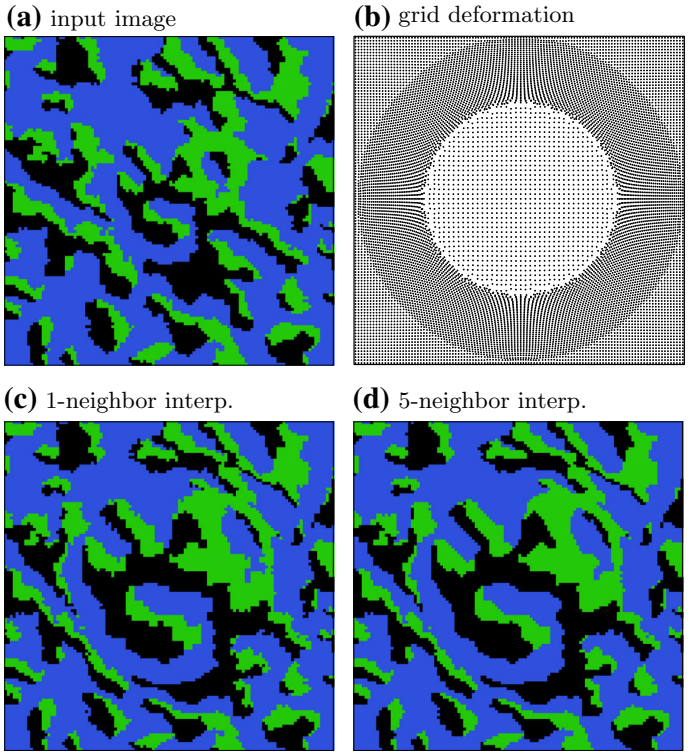

Fig. 3 Space deformation (not self-similarity-based): **a** input image (114 × 114); **b** transformation; **c**–**d** output image using the nearest neighbor method (**c**) and the inverse distance method on facies indicators with 5 neighbors (**d**)

4 Self-Similarity-Based Warping

Self-similarity-based warping consists in applying a space deformation (expansion or compression) onto the zones similar to the reference pattern selected by the user. This is done by following two main steps: (1) moving the center of the cells of the original grid according to the desired deformation, and (2) interpolating the values known at the new locations onto the original grid.

In the following, let us first describe the interpolation step (Sect. 4.1) and then how to set the space deformation based on self-similarity (Sect. 4.2).

4.1 Interpolation onto the Original Grid

Figure 3 illustrates the warping of a categorical image, which is not based on self-similarity but corresponds to a zoom in the central part. In Fig. 3b, the black dots correspond to the new locations of the center of each grid cell.

Let x_j be the new location of the center of each pixel of the grid after applying a given space deformation. The interpolation step consists in defining the new value

$Z^\star(x)$ in a pixel centered at x in the original grid from the values $Z(x_j)$ known on the migrated pixels.

For continuous images, an inverse distance method can be used. One can define

$$Z^\star(x) = \sum_{i=1}^{K} \omega_i Z(x_i), \tag{5}$$

where x_i, $i = 1, \ldots, K$, are the K nearest neighbors of x among the migrated pixels, and where the weights ω_i, $i = 1, \ldots, K$, are set inversely proportional to the distance between x and x_i

$$\omega_i = \frac{||x - x_i||^{-1}}{\sum_{j=1}^{K} ||x - x_j||^{-1}}. \tag{6}$$

For categorical images, direct interpolation does not make sense, because the new values have to correspond to a facies code of the input image. However, the inverse distance method can be used on the facies indicators: if $z_1, \ldots, z_M$ are the facies code, one computes

$$P_m(x) = \sum_{i=1}^{K} \omega_i \cdot \delta\left(Z(x_i), z_m\right), \tag{7}$$

for $m = 1, \ldots, M$, where $\delta(Z(x_i), z_m) = 1$ if $Z(x_i) = z_m$ and 0 otherwise. These values $P_m(x)$, $m = 1 \ldots, M$, sum to 1, and can be interpreted as probabilities of having the facies z_m at the node x. Then, a facies maximizing this probability is simply assigned to the location x

$$Z^\star(x) = z_{m^\star}, \text{ with } m^\star = \text{argmax}_{m=1,\ldots,M} P_m(x). \tag{8}$$

Note that if $K = 1$, the inverse distance method is equivalent to the nearest neighbor method, which consists in assigning the value attached to the nearest neighbor. In the previous illustration (where the transformation is not based on self-similarity), the results of the interpolation with $K = 1$ and $K = 5$ are displayed in Fig. 3c and d, respectively. One can observe that using more than 1 neighbor prevents noise in some areas of the resulting image. Nevertheless, using too many neighbors could lead to overly "smooth" images. In the following, the number of neighbors is set to 5.

4.2 Setting the Space Deformation

The two main ingredients used for self-similarity-based warping are transformations and magnification fields (Keahey and Robertson 1997). In this section, these notions are introduced for the bi-dimensional case. The generalization in three dimensions is straightforward.

A space deformation is described by a transformation T, a function from $\mathbb{R}^2$ to $\mathbb{R}^2$ defining the displacement of the pixels in the grid

$$T : (x, y) \longmapsto T(x, y) = (T_x(x, y), T_y(x, y)). \tag{9}$$

The determinant of its Jacobian matrix

$$M_T = \det(DT) = \frac{\partial T_x}{\partial x} \cdot \frac{\partial T_y}{\partial y} - \frac{\partial T_x}{\partial y} \cdot \frac{\partial T_y}{\partial x}, \tag{10}$$

gives the associated magnification field (i.e. the magnification rate in every point in $\mathbb{R}^2$) associated to T.

In the context of self-similarity-based warping, the user selects a pixel x_0 in an image and specifies a magnification rate to be applied to similar regions. The aim at this point is to: (i) construct a target magnification field M (defined on the grid nodes) based on the dissimilarity map and the desired magnification rate f (expansion or compression), and (ii) find a transformation T for which the associated magnification field M_T is close to M.

Self-similarity-based warping is illustrated in Fig. 4. Given the input image (Fig. 4a) and the reference pattern (in red), the dissimilarity map (Fig. 4b) is computed, and a target magnification field (Fig. 4c) is determined from the dissimilarity map and according to the user inputs (see Sect. 4.2.1 for details). Then, a space deformation (Fig. 4d) is computed from the target magnification field. The grid dots in Fig. 4d correspond to the new locations of the center of each grid cell. The output image (Fig. 4e) results from the interpolation of the original values at these new locations onto the original grid (see Sect.4.1). The computation of the space deformation (shown in Fig. 4d) is done by an iterative process (see Sect. 4.2.2): the evolution of the error is shown in Fig. 4f.

4.2.1 Target Magnification Field

If Ω denotes the domain in $\mathbb{R}^2$ on which the image is defined, the target magnification field M on Ω should verify the following properties. All magnification rates have to be positive: $M(x, y) > 0$ for all $(x, y) \in \Omega$. A value greater than 1 corresponds to an expansion and a value less than 1 to a compression. The target magnification field M should be associated to a transformation mapping the image grid Ω onto itself

$$\int_\Omega M(x, y)\mathrm{d}x\mathrm{d}y = \int_\Omega \mathrm{d}x\mathrm{d}y (= \mathrm{Area}(\Omega)), \tag{11}$$

(i.e., the expansion and compression regions should be balanced). Let

$$G = \{(i, j), i = 1, \ldots, N_x, j = 1, \ldots, N_y\}, \tag{12}$$

be the set of grid nodes (pixel) of the initial image of size $N_x \times N_y$. Each pixel represents an area of 1, and the previous properties are expressed as follows

$$M(i, j) > 0, \ i = 1, \ldots, N_x, \ j = 1, \ldots, N_y, \tag{13}$$

and

$$\sum_{i=1}^{N_x} \sum_{j=1}^{N_y} M(i, j) = N_x \cdot N_y. \tag{14}$$

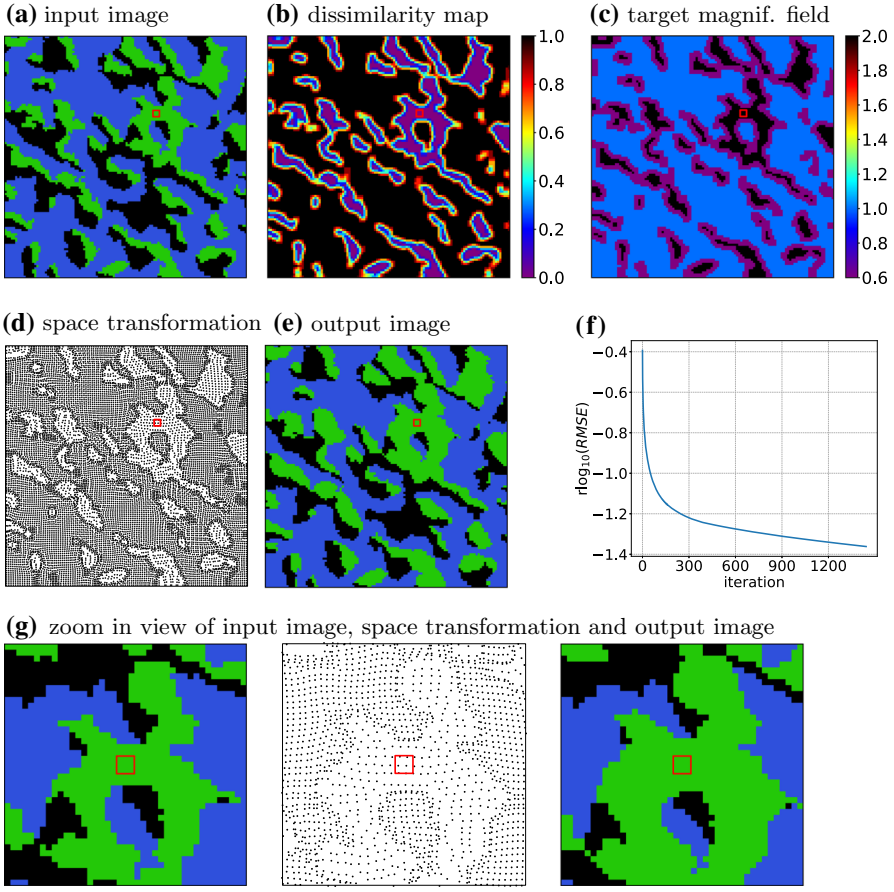

Fig. 4 Self-similarity-based warping: **a** input image (114×114); **b** dissimilarity map; **c** target magnification field M for $f = 2.0$, and $t_1 = 0.2$, $t_2 = 0.9$; **d** space transformation; **e** output image; **f** evolution of the error [Eq. (16)]; **g** zoom in views around the 3×3 reference pattern (in red)

To construct the target magnification field M from the dissimilarity map dis (Sect. 2), three input parameters are required: two threshold values $0 < t_1 < t_2 \leqslant 1$, and a magnification rate $f > 0$. The thresholds are used to divide the grid G into three regions: $G_{\mathrm{dis}<t_1} = \{(i, j) : \mathrm{dis}(i, j) < t_1\}$, $G_{t_1 \leqslant \mathrm{dis} \leqslant t_2}$ and $G_{\mathrm{dis}>t_2}$ (defined similarly). Then, the magnification rate is set to the specified rate f on $G_{\mathrm{dis}<t_1}$, the region considered as similar to the selected pattern, to 1 on $G_{\mathrm{dis}>t_2}$, region not modified, and to $\tilde{f}$ on $G_{t_1 \leqslant \mathrm{dis} \leqslant t_2}$, $\tilde{f}$ being computed such that the property (14) is verified. It follows that

$$\tilde{f} = 1 + \frac{\left|G_{\mathrm{dis}<t_1}\right|}{\left|G_{t_1 \leqslant \mathrm{dis} \leqslant t_2}\right|} \cdot (1 - f), \tag{15}$$

where $|\cdot|$ denotes the cardinality (number of elements). The idea is to use the region $G_{t_1 \leqslant \mathrm{dis} \leqslant t_2}$, around $G_{\mathrm{dis}<t_1}$, to compensate the specified magnification. Note that

there is a constraint on the choice of the input parameters t_1, t_2 and f, because the magnification field must be positive on the whole grid (13). Hence, if the value of M computed on the area $G_{t_1 \leqslant \text{dis} \leqslant t_2}$ is negative or vanishes, the input parameters are rejected, and the user should decrease f or t_1 or increase t_2. Note that if one defines $t_2 = 1$, the set $G_{\text{dis}>t_2}$ is empty, and then the complementary set of $G_{\text{dis}<t_1}$ is used for balancing the magnification field.

4.2.2 Computing a Space Transformation

Once the target magnification field M is constructed, one must find a transformation T for which the corresponding magnification field $M_T = \det(DT)$ is close to M. While the magnification field of a given transformation is straightforwardly computed by calculating the determinant of the Jacobian matrix, the reverse is not obvious. This is done iteratively, by the following steps (Keahey and Robertson 1997)

(1) Initialize $T =$ identity.
(2) Compute field M_T.
(3) Compute the error field $M_E = M - M_T$.
(4) Compute the Root Mean Squared Error (RMSE)

$$||M_E|| = \left(\frac{1}{N_x \cdot N_y} \sum_{i=1}^{N_x} \sum_{j=1}^{N_y} M_E(i, j)^2 \right)^{\frac{1}{2}}. \tag{16}$$

(5) If the error $||M_E||$ is below a given tolerance, accept T, otherwise update T and go to step 2.

A maximal number of iterations is set, and if the error M_E is stabilized over the last iterations, one also exits the loop. Moreover, one can accept an error tol_{cell} on each grid cell (pixel) by replacing the error $M_E(i, j) = M(i, j) - M_T(i, j)$ in step 3 by

$$M_E(i, j) = \text{sign}(M(i, j) - M_T(i, j)) \max\left(|M(i, j) - M_T(i, j)| - \text{tol}_{\text{cell}}, 0\right). \tag{17}$$

In step 2, the derivatives of T must be computed to retrieve M_T [see Eq. (10)] at each pixel (i, j). The derivatives according to x are estimated by

$$\begin{aligned}
\frac{\partial T}{\partial x}(i, j) &\approx (T(i+1, j) - T(i-1, j))/2 && \text{for } 1 < i < N_x,\ 1 \leqslant j \leqslant N_y, \\
\frac{\partial T}{\partial x}(1, j) &\approx T(2, j) - T(1, j) && \text{for } 1 \leqslant j \leqslant N_y, \\
\frac{\partial T}{\partial x}(N_x, j) &\approx T(N_x, j) - T(N_x - 1, j) && \text{for } 1 \leqslant j \leqslant N_y,
\end{aligned}$$

and the derivatives according to y by

$$\begin{aligned}
\frac{\partial T}{\partial y}(i, j) &\approx T(i, j+1) - T(i, j-1)/2 && \text{for } 1 \leqslant i \leqslant N_x,\ 1 < j < N_y, \\
\frac{\partial T}{\partial y}(i, 1) &\approx T(i, 2) - T(i, 1) && \text{for } 1 \leqslant i \leqslant N_x, \\
\frac{\partial T}{\partial y}(i, N_y) &\approx T(i, N_y) - T(i, N_y - 1) && \text{for } 1 \leqslant i \leqslant N_x.
\end{aligned}$$

The most delicate task is to update T in step (v) when the target magnification is not reached. To decrease the error, the transformation T can be modified by moving the (at maximum) 4 neighbors $T(i-1, j)$, $T(i, j-1)$, $T(i+1, j)$, $T(i, j+1)$ of $T(i, j)$ a little bit away from (resp. closer to) $T(i, j)$ if the error $M_E(i, j)$ is positive (resp. negative). One proceeds as follows:

1. Initialize the displacement vector $V(i, j) = (V_x(i, j), V_y(i, j)) = (0, 0)$ for all pixels (i, j) in G.
2. Visiting all pixels (i, j) in G, update the displacement vector at each neighbor pixel as follows. If $M_E(i, j) > 0$

$$\begin{aligned}
V(i+1, j) &\longleftarrow V(i+1, j) + \alpha \cdot M_E(i, j) \cdot (T(i+2, j) - T(i+1, j)),\\
V(i-1, j) &\longleftarrow V(i-1, j) + \alpha \cdot M_E(i, j) \cdot (T(i-2, j) - T(i-1, j)),\\
V(i, j+1) &\longleftarrow V(i, j+1) + \alpha \cdot M_E(i, j) \cdot (T(i, j+2) - T(i, j+1)),\\
V(i, j-1) &\longleftarrow V(i, j-1) + \alpha \cdot M_E(i, j) \cdot (T(i, j-2) - T(i, j-1)),
\end{aligned}$$

   and if $M_E(i, j) < 0$

$$\begin{aligned}
V(i+1, j) &\longleftarrow V(i+1, j) + \alpha \cdot M_E(i, j) \cdot (T(i+1, j) - T(i, j)),\\
V(i-1, j) &\longleftarrow V(i-1, j) + \alpha \cdot M_E(i, j) \cdot (T(i-1, j) - T(i, j)),\\
V(i, j+1) &\longleftarrow V(i, j+1) + \alpha \cdot M_E(i, j) \cdot (T(i, j+1) - T(i, j)),\\
V(i, j-1) &\longleftarrow V(i, j-1) + \alpha \cdot M_E(i, j) \cdot (T(i, j-1) - T(i, j)),
\end{aligned}$$

   with a positive factor α. Note that the expressions in which one pixel index falls out of the grid G are not applied.
3. Update T, ensuring that the nodes on the borders of the grid remain on the borders

$$\begin{aligned}
T_x(i, j) &\longleftarrow T_x(i, j) + V_x(i, j), \text{ for } 1 < i < N_x,\ 1 \leqslant j \leqslant N_y,\\
T_y(i, j) &\longleftarrow T_y(i, j) + V_y(i, j), \text{ for } 1 \leqslant i \leqslant N_x,\ 1 < j < N_y.
\end{aligned}$$

   Note that the displacement vectors $V(i, j)$ are reduced if needed, to ensure that the spatial order of the pixels is maintained.

Self-similarity-based warping is illustrated in Fig. 4. A magnification rate of $f = 2$ and the threshold values $t_1 = 0.2$ and $t_2 = 0.9$ are used.

5 Further Examples

In this section, some examples showing the practical application of the proposed techniques are presented.

First, the painting process can be used to adapt the proportion of each category for discrete images or to adapt the distribution of values for continuous images, while controlling which spatial features of the input image need to be kept or modified. For example, in Fig. 5, the proportion of the blue and green facies are modified by

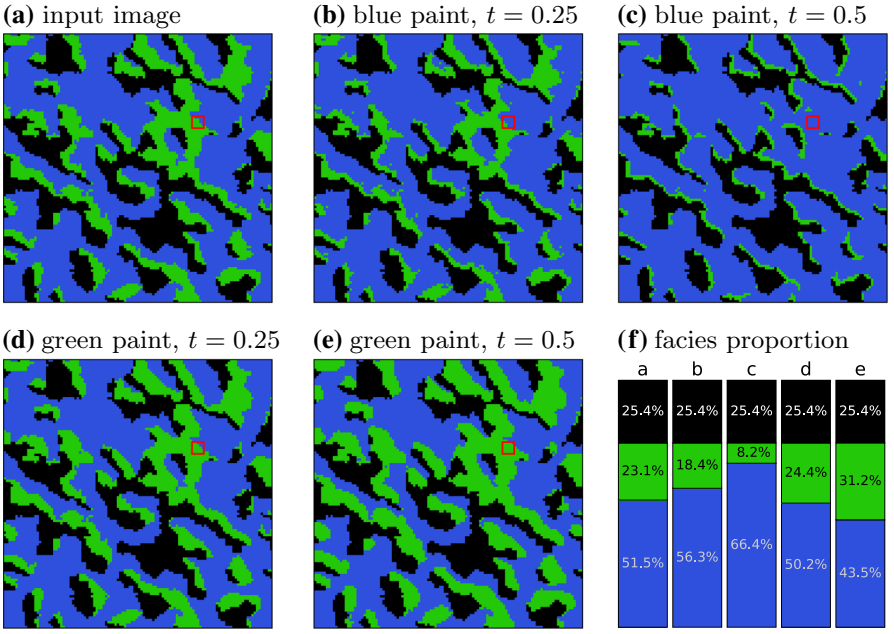

Fig. 5 Painting examples (114 × 114 categorical image, 5 × 5 reference pattern in red): **a** input image; **b–e** output image for different settings ($z_{\text{new}} = 0$ for blue paint, $z_{\text{new}} = 1$ for green paint); **f** facies proportion for each case

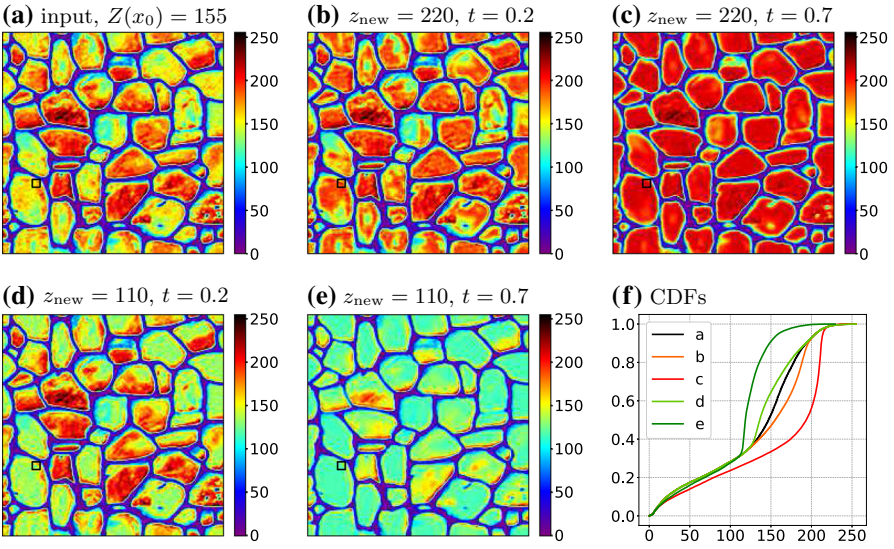

Fig. 6 Painting examples (200 × 200 continuous image, 7 × 7 reference pattern in black): **a** input image ($Z(x_0) = 155$); **b–d** output image for different settings; **f** cumulative distribution function for each case

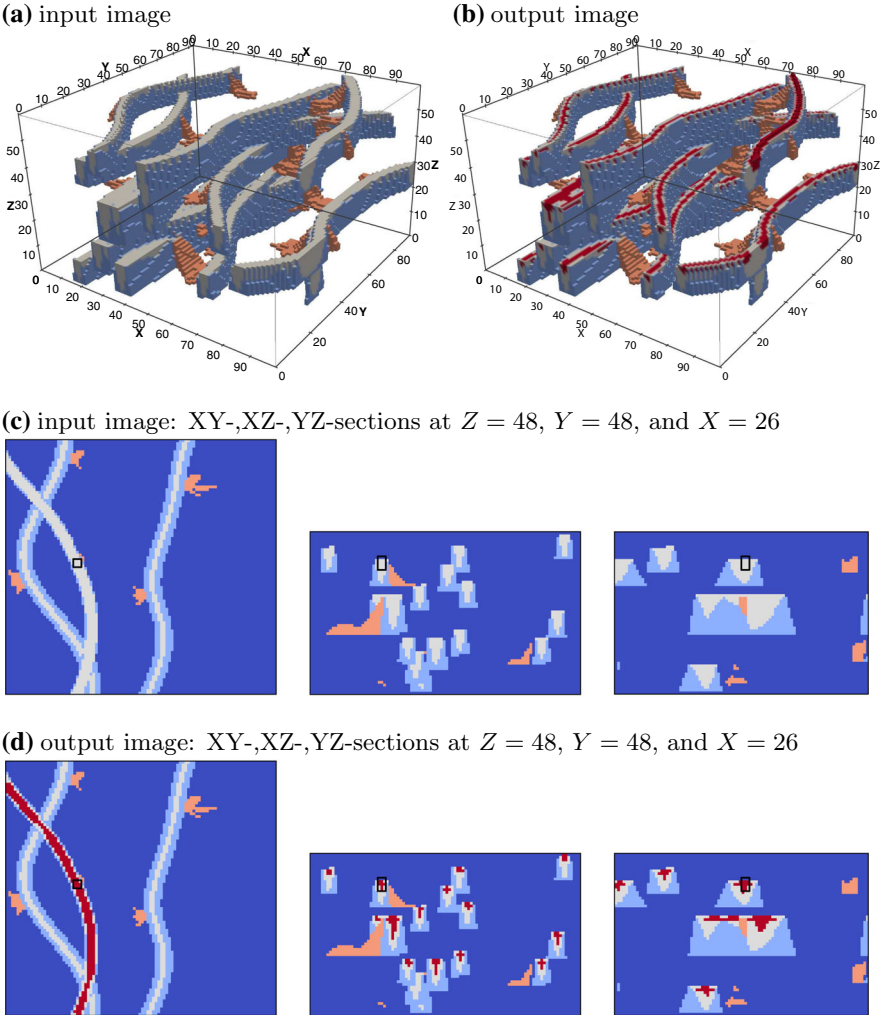

Fig. 7 Painting example for a three-dimensional categorical image ($100 \times 94 \times 60$, courtesy of TOTAL): **a** input image; **b** output image (new facies paint (red), threshold $t = 0.27$); **c–d** section views going through the center of the $3 \times 3 \times 5$ reference pattern (in black) for the input and output images

changing the color of the paint (z_{new}) and the threshold value (t). Similarly for the continuous case, the distribution of values is altered (Fig. 6).

The painting process can also be used to add a new facies to a categorical image to distinguish spatial structures. A three-dimensional example is displayed in Fig. 7, where a new facies is assigned at the top of the inner part of the channels.

Alternatively, the warping process allows a modification of the sizes of spatial features. For example, it is used to modify the thickness of channels in example of Fig. 8. Selecting a pattern in a channel, a magnification rate greater (resp. smaller) than 1 allows one to make the channels thicker (resp. thinner). Notice that although

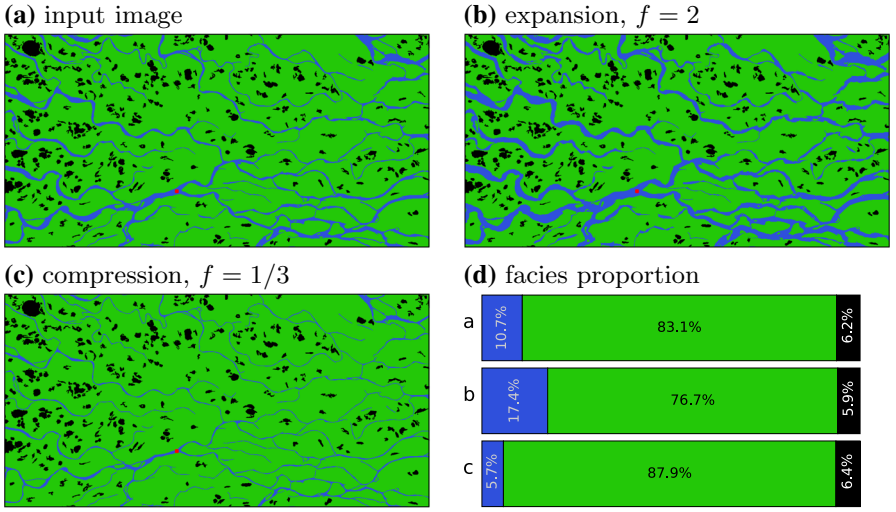

Fig. 8 Warping example (1101×551 categorical image, 5×5 reference pattern in red): **a** input image (Lena River, from *Landsat 7 image, USGS/EROS and NASA Landsat Project*); **b–c** output image after expansion and compression ($t_1 = 0.5$, $t_2 = 1.0$); **d** facies proportion in each case

the thickness of the channels is variable within the input image (Fig. 8a), the proposed method is able to reduce it without breaking the channels (Fig. 8c).

The last example (Fig. 9) illustrates the warping process in three dimensions with a continuous image. Applying an expansion (magnification rate greater than 1) on the regions with low values allows a "swelling" of the blue area.

5.1 Computational Performance

All the examples and illustrations in this paper are produced running parallel code with 4 threads on a quad-core machine (*Intel(R) Core(TM) i7-4810MQ CPU @ 2.80 GHz*). All the results have been obtained in less than 0.1 s (real time), except for the three last examples: about 1 s and 2 s for the three-dimensional examples (Figs. 7, 9 resp.), and about 5–6 s for each output image of Fig. 8.

Whereas editing images by painting is straightforward, editing by warping is more CPU-demanding because of the iterative procedure for computing the space transformation (Sect. 4.2.2). However, as shown in Fig. 4f, the error is strongly decreasing in the beginning of this iterative procedure, and then diminishes slowly. Hence, one can reduce the maximal number of iterations, and re-edit the output image if needed. Nevertheless, satisfactory results can often be obtained after only a few iterations.

6 Conclusions

The proposed methods allow the user to edit bi- or three-dimensional, categorical or continuous, stationary images, while keeping spatial features consistent. The methods

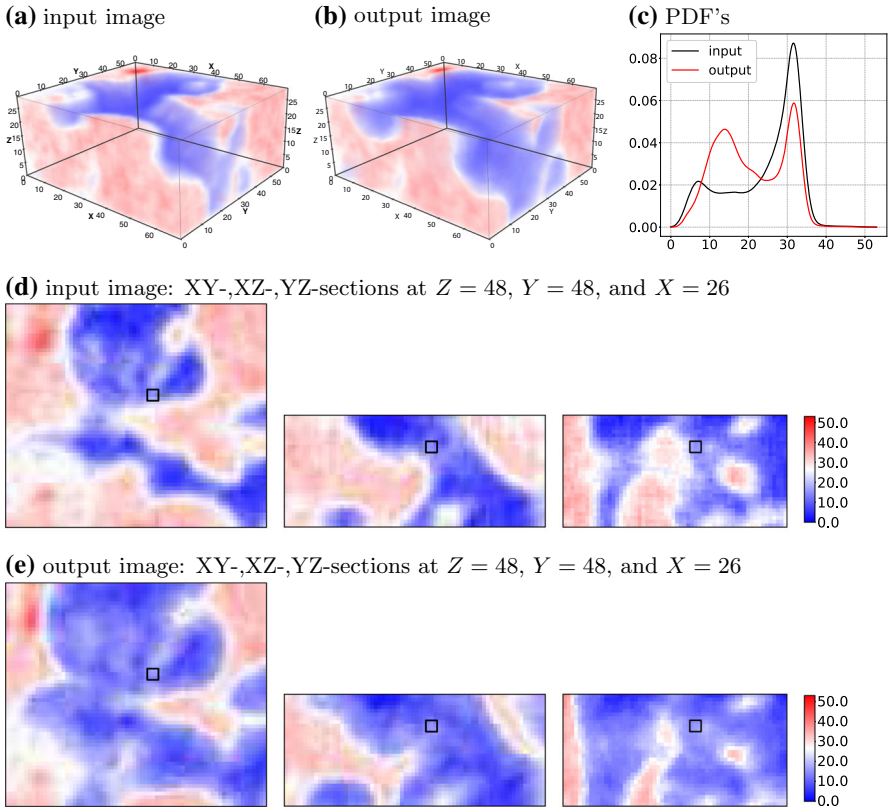

Fig. 9 Warping example for a three-dimensional continuous image ($70 \times 60 \times 30$): **a** input image; **b** output image ($f = 3$, $t_1 = 0.2$, $t_2 = 1.0$); **c** distribution of values for the input and output images; **d–e** section views going through the center of the $3 \times 3 \times 3$ reference pattern (in black)

are based on self-similarity following two different modes, painting and warping, and provide interactive tools. The results are intuitive and few parameters, easy to understand, are required in input. In both modes, the user selects in the image a pattern representing the type of structures she or he wants to modify. The rate of dissimilarity to this pattern is computed everywhere in the image. In painting mode, all the grid nodes having a dissimilarity rate below a specified threshold t are directly modified by using a painting value (z_{new}). In warping mode, a space deformation is applied to the image grid, and the values are then re-interpolated onto the initial grid. The space transformation is iteratively computed such that the associated magnification field, which is given by the determinant of the Jacobian matrix, matches a target field defined by a specified magnification rate f and two thresholds $t_1 < t_2$. The target rate is set at each grid node to f if the dissimilarity rate is below t_1 (region similar to the picked pattern), to 1 if it is above t_2 (frozen zone), and otherwise to a value automatically computed to have a consistent field.

Based on self-similarity, these methods are designed for stationary images. Nevertheless, as the transformations are applied everywhere the image presents a structure

similar to that of the selected pattern, these techniques are still consistent for images displaying some non-stationarities. Then, as multiple point statistics techniques can handle non-stationary training images, see for example Chugunova and Hu (2008), Mariethoz et al. (2010), Straubhaar et al. (2011), the proposed tools can also be useful in this case.

The computational time is rather low and these tools can be used interactively. A graphical user interface (GUI) could be developed for these tools, in particular to enable direct pattern picking on the view of the input image. In addition to viewing of the output image, histograms and other statistical measures could be integrated to offer better control of the editing processes.

The proposed tools give the user enhanced flexibility to set up the inputs required by multiple point statistics algorithms. They allow a pre-processing step consisting in adapting the training image to better fit some features desired in the output realizations, such as the facies proportions or the thicknesses of some spatial structures. For geologist users of multiple point statistics, assessing whether the resulting training image is satisfying largely relies on subjective criteria. In this context, the methods described in this paper, combined with statistical measures and used in an iterative manner, can be helpful tools.

Alternative uses of the proposed tools can be envisioned. Self-similarity-based editing processes may be applied to an interpreted geological model (i.e., the output of a modeling procedure). For example in mining applications, several models can be generated from an existing model by moving the geological contacts whose positions are uncertain (Boucher et al. 2014). In the same vein, such tools may also be integrated into procedures aimed at solving inverse problems in hydrogeology (Li et al. 2015; Jäggli et al. 2017): whatever the stochastic algorithm used to generate the parameter field, at each step, a realization can be slightly modified to increase its likelihood, given some observation data. The way to perturb a realization (i.e., the set-up for the editing process) should then be automated.

Acknowledgements We are grateful to TOTAL S.A. for co-funding this work.

References

Allard D, D'Or D, Froidevaux R (2011) An efficient maximum entropy approach for categorical variable prediction. Eur J Soil Sci 62(3):381–393. https://doi.org/10.1111/j.1365-2389.2011.01362.x

Bookstein FL (1989) Principal warps: thin-plate splines and the decomposition of deformations. IEEE Trans Pattern Anal Mach Intell 11(6):567–585. https://doi.org/10.1109/34.24792

Boucher A, Costa JF, Rasera LG, Motta E (2014) Simulation of geological contacts from interpreted geological model using multiple-point statistics. Math Geosci 46(5, SI):561–572. https://doi.org/10.1007/s11004-013-9510-1

Brooks S, Dodgson N (2002) Self-similarity based texture editing. ACM Trans Graph 21(3):653–656

Chugunova T, Hu L (2008) Multiple-point simulations constrained by continuous auxiliary data. Math Geosci 40(2):133–146. https://doi.org/10.1007/s11004-007-9142-4

Jäggli C, Straubhaar J, Renard P (2017) Posterior population expansion for solving inverse problems. Water Resour Res 53(4):2902–2916. https://doi.org/10.1002/2016WR019550

Keahey TA, Robertson EL (1997) Nonlinear magnification fields. In: Proceedings of the 1997 IEEE symposium on information visualization (INFOVIS '97). IEEE Computer Society, Washington, DC, USA, pp 51–58

Li L, Srinivasan S, Zhou H, Jaime Gomez-Hernandez J (2015) A local-global pattern matching method for subsurface stochastic inverse modeling. Environ Modell Softw 70:55–64. https://doi.org/10.1016/j.envsoft.2015.04.008

Mariethoz G, Renard P, Straubhaar J (2010) The direct sampling method to perform multiple-point geostatistical simulations. Water Resour Res. https://doi.org/10.1029/2008WR007621

Mariethoz G, Straubhaar J, Renard P, Chugunova T, Biver P (2015) Constraining distance-based multipoint simulations to proportions and trends. Environ Modell Softw 72:184–197. https://doi.org/10.1016/j.envsoft.2015.07.007

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

Straubhaar J, Renard P, Mariethoz G (2016) Conditioning multiple-point statistics simulations to block data. Spat Stat 16:53–71. https://doi.org/10.1016/j.spasta.2016.02.005

Strebelle S, Zhang T (2005) Non-stationary multiple-point geostatistical models. Springer, Dordrecht, pp 235–244. https://doi.org/10.1007/978-1-4020-3610-1_24

Tahmasebi P (2017) Structural adjustment for accurate conditioning in large-scale subsurface systems. Adv Water Resour 101:60–74. https://doi.org/10.1016/j.advwatres.2017.01.009

Zhang T, Switzer P, Journel A (2006) Filter-based classification of training image patterns for spatial simulation. Math Geol 38(1):63–80. https://doi.org/10.1007/s11004-005-9004-x