J Petrol Explor Prod Technol (2014) 4:153–174
DOI 10.1007/s13202-014-0107-0

ORIGINAL PAPER - EXPLORATION ENGINEERING

A sensitivity study of FILTERSIM algorithm when applied to DFN modeling

Rouhollah Ahmadi · Mohsen Masihi · Mohammad Reza Rasaei · Kiomars Eskandaridalvand · Reza Shahalipour

Received: 20 January 2013 / Accepted: 24 January 2014 / Published online: 6 March 2014

Abstract Realistic description of fractured reservoirs demands primarily for a comprehensive understanding of fracture networks and their geometry including various individual fracture parameters as well as network connectivities. Newly developed multiple-point geostatistical simulation methods like SIMPAT and FILTERSIM are able to model connectivity and complexity of fracture networks more effectively than traditional variogram-based methods. This approach is therefore adopted to be used in this paper. Among the multiple-point statistics algorithms, FILTERSIM has the priority of less computational effort than does SIMPAT by applying filters and modern dimensionality reduction techniques to the patterns extracted from the training image. Clustering is also performed to group identical patterns in separate partitions prior to simulation phase. Various practices including principal component analysis, discrete cosine transform and different data summarizers are used in this paper to investigate a suitable way of reducing pattern dimensions using outcrop maps as training image. Because of non-linear nature of patterns present in fracture networks, linear clustering algorithms fail in determining the borders between the actual partitions; non-linear approaches like Spectral methods, however, can act more efficiently in diagnosing the right clusters. A complete sensitivity analysis is performed on FILTERSIM algorithm regarding search template dimension, type of filtering technique and the number of clusters for each clustering approach described above. Interesting results are obtained for each parameter that is changed during analysis.

Keywords Discrete fracture network · Geometrical modeling · Multiple-point geostatistics · Dimensionality reduction technique · Clustering

R. Ahmadi (✉) · M. Masihi
Department of Chemical and Petroleum Engineering, Sharif University of Technology, Tehran, Iran
e-mail: rouhollahahmadi@yahoo.com

M. Masihi
e-mail: masihi@sharif.edu

M. R. Rasaei
Institute of Petroleum Engineering, Tehran University, Tehran, Iran
e-mail: mrasaei@ut.ac.ir

K. Eskandaridalvand
DeGolyer and MacNaughton Petroleum Consulting Company, Dallas, TX, USA
e-mail: kiomarseskandari@yahoo.com

R. Shahalipour
Petroiran Development Company, Tehran, Iran
e-mail: shahalipour@gmail.com

Introduction

Naturally fractured reservoirs are one of the most complex structures in petroleum geology. They represent great heterogeneities in all facets from the geological structure to the flow responses so that characterization and modeling of these reservoirs have been a real challenge for petroleum engineers. Presence of fracture networks plays an important role in flow of hydrocarbon fluids toward the wellbores. They may act as barriers, hindering the production and thus causing different scenarios for economical production. Although the fracture system can increase the porosity and hence the initial oil in place, this causes the drilling fluid lost circulation. Mud invasion through the fracture system gives rise to the cost and time of drilling operation; therefore modeling of fracture system is crucial for assessing the connectivity of the reservoir. In general, familiarity of the fracture system model is important for the development of oil fields, economical issues of geothermal reservoirs and production from groundwater sources. This knowledge can benefit petroleum engineers for better estimates of reserves, production forecasts and optimization of oil recovery factor (Tran et al. 2002; Tran 2004).

Modeling of naturally fractured formations starts from the geological model, ending to the flow simulation; therefore modeling of fracture network requires two steps: fracture geometry characterization and simulation of its dynamic behavior. Geometrical fracture modeling is thus the first step in describing the fractured reservoir.

Characterization of highly fractured regions is one of main purposes of reservoir management, and it is after that one can design the well locations and their trajectories to cut through the fractures. Therefore, construction of the facture intensity distribution models sounds important. In addition, other properties of the individual fractures as well as of the network are necessary to be modeled, since fractures with different properties (like size, dip and connectivity) show different effects on the fluid flow behavior; more fluids flow through the longer and wider fractures. Fracture orientation controls the anisotropy and permeability of the reservoir at different directions. For example subparallel fractures cannot be connected to each other to form a continuous and connected network for fluid flow; in this case, the well trajectory must be deviated to pass through the fractures to increase the production recovery. Using discrete fracture networks facilitates the numerical simulation of fluid flow in the reservoir. For example, where geometric fracture properties (location, orientation, size and width) are available, finite element or boundary element methods can be used to obtain the permeability tensor of the discrete network analytically (Oda et al. 1987; Lee et al. 1999; Teimoori et al. 2004). Cacas et al. (1990) used geometrical fracture statistics instead of permeability tensor to calculate the large-scale hydraulic conductivity of the fractured rock.

How to use DFN models in the flow simulation of fractured reservoirs and the estimation of hydraulic properties are discussed by Bourbiaux et al. (1997), Baker et al. (2001) and Dershowitz et al. (2002).

This paper aims at geometrical modeling of natural fracture networks observed on the outcrops. Since the MPS approaches are capable of reproducing the complex and curvilinear geological features such as natural fractures (Strebelle 2002; Caers and Arpat 2005; Journel and Zhang 2006), this technique is adopted to be used in this work. Utilizing reservoir outcrop maps taken at different locations and structures as training images, fracture networks are modeled under geological and tectonically conditions. We have focused on reproducing the multiple-point statistics incident to an initial training image through the FILTERSIM algorithm. Flow simulation results are finally used to validate the generated realizations by comparing their production recoveries with that of the prior model. FILTERSIM has the priority of fast computations over previously defined algorithms, like SIMPAT, through the classification of extracted patterns in separate clusters (Journel and Zhang 2006; Wu 2007). To reduce dimensionality of pattern space, FILTERSIM has been designed to classify structural patterns using selected filter statistics as specific linear combinations of pattern pixel values that represent different properties. Instead, patterns could be abbreviated in a small set of statistical measures called summaries. Data summaries are also used in this paper to do filtering in FILTERSIM. A full description of clustering algorithms and their applications in FILTERSIM algorithm is provided in the following sections. Furthermore, a new classification of dimensionality reduction practices that could be effectively used in FILTERSIM is provided. Several solutions would also be developed to improve different parts of the available FILTERSIM algorithm in line with the simulation of naturally fracture networks. A complete sensitivity analysis would be finally performed on FILTERSIM parameters including search template size, kind of filtering practice, type of clustering approach and the number of clusters used in partitioning the pattern space. Useful results are induced after this analysis that would help us with improving the FILTERSIM algorithm in further applications.

Discrete fracture network models

DFN models simulate the individual fracture properties, predicting the fluid flow behavior using the fracture geometry and conductivity. These models can be obtained either by deterministic or stochastic approaches. Due to deficiencies inherent to deterministic methods when applying to modeling of complex fractured media, most discrete fracture models make use of probabilistic and stochastic simulation concepts to determine the distribution and orientation of fractures having access to the structural growth stress and tectonic history information which can be obtained with difficulty.

In two-dimensional models (Long et al. 1982; Smith and Schwartz 1984; Mukhopadhyay and Sahimi 1992), fractures are represented as lines distributed randomly in a plane. The center of lines of fractures follows a Poisson distribution while the length and orientation of a fracture can be selected from any given distribution. Hestir and

Long (1990) analyzed various two-dimensional models to determine fracture connectivity and relate the parameters of fracture networks to standard percolation network parameters.

More complicated three-dimensional models were developed (Long and Witherspoon 1985; Charlaix et al. 1987; Pollard et al. 1976) to allow for realistic representation of fracture networks. Fractures are represented as flat planes of finite dimensions and in most of the cases as circular discs.

Fracture data (such as orientation, width and density) are primarily available at well locations. These data are analyzed statistically to estimate the distribution of fracture properties at inter-well locations.

DFN stochastic simulation has many advantages over the geomechanical and mathematical deterministic methods (either continuum or discrete). Dealing with these models, the approximations incident to the continuum models are eliminated by employing the probability distribution functions. The fracture information from different data sources can be used to construct the systems with individual fracture detailed properties in this approach. Stochastic simulation can be conditioned to identified fractures as well as dynamic data by taking the variable uncertainties into consideration. This improves the validity of fracture network models with respect to the reservoir condition.

In general, there are two stochastic approaches for modeling of fracture networks (1) object-based approach in which the fractures are depicted by the closed form objects with clean edges and (2) pixel-based approach, which simulates the nodal values one by one and then by combining these values, final fracture network is constructed. There are other algorithms such as tessellation and Voronoi grid methods which can be attributed to this class. However, due to deficiencies for applying these approaches, they could not be used in practice for modeling natural fracture networks.

In object-based models, fractures are defined as entities with specific center, shape, size and orientation, randomly positioned in space. Random disk models represent the fractures as two-dimensional convex circular disks. The radii of disks can be obtained by a log-normal distribution whose parameters are inferred from trace length distribution observed on the outcrops. Distribution and location of the disks, radii and their orientations are assumed uncorrelated from one fracture to another. Modeling of fracture clusters is difficult with the assumption of random fracture locations. Clustering is performed by parent and daughter model in which a spatial density function is used for cluster modeling. Seed locations of the fractures are determined by this density function and the fracture clusters are simulated over a predefined volume near the initial seed locations (Long et al. 1987).

Fractures are not circular disks; they have elliptical shapes in reality. Based on Long et al.'s (1987) and Hestir et al.'s (1987) findings, if there is not enough data to determine the real shape of the fractures, it is a logical assumption to imagine the fractures as disks. This assumption has many disadvantages and outcrop data do not confirm that.

Many drawbacks are inherent to these models; independence assumption between the parameters, difficulties in inferring the statistical distribution parameters, problems in soft data conditioning, and convex and planar assumption of individual fractures are limitations of this method.

While object-based techniques construct the features with clean edges, however, pixel-based algorithms simulate the nodal values on a discrete network. Useful algorithms such as SISIM, SGSIM or PFSIM are developed for modeling continuous (petrophysical data) and discrete (facies) parameters traditionally. One-point or two-point statistics are not able to reproduce the objects like fractures.

Traditional pixel-based algorithms use the primary data values at their locations as well as univariate statistical distributions and bivariate moments as the required information in the constructed models. This causes the curvilinear and continuous structure of geological entities to be reproduced in a poor manner through the variogram-based geostatistical algorithms.

Two-point statistical moments like variogram and covariance between simulation variables take only the two-point spatial relationships into consideration at one time; the reproduction of such complex geological images requires a relationship between more than two points at the same time and therefore the development of multiple-point simulation approaches seems to be necessary.

MPS techniques are not only capable of handing the prior maximum entropy models and performing multivariate Gaussian algorithms, but they can act very effectively in using the prior structural models of low entropy having more compatibility with hard conditioning data or prior geological models.

Guardiano and Srivastava (1993) developed the first MPS algorithm based on the Journel's extended normal equations. In the following section, a short review of available MPS algorithms with an emphasis on FILTERSIM is introduced.

A review of multiple-point geostatistical simulation algorithms

Various algorithms have been developed for multiple-point simulation of complex geological phenomena. To overcome the problems inherent to the two-point statistics, the issue of multiple-point geostatistics was introduced by Deutsch (1992) and Guardiano and Srivastava (1993). Later, development of a practical MPS simulation algorithm, SNESIM (Strebelle 2002), has led to the wide adoption of MPS simulation for reservoir facies or element modeling. SNESIM draws the patterns of a training image according to the well-defined probabilities and conditioned to the available hard and soft data. This algorithm utilizes a sequential simulation approach in its modeling scheme as in traditional pixel-based simulations: (1) the unknown nodal values of the simulation grid are visited along a predefined random path separately in time; (2) in each unknown cell, the conditional probability is estimated regarding the neighborhood data and the previously simulated values, and (3) a value is randomly selected from the constructed probability distribution and assigned to the current simulation cell. In SNESIM, the conditional probability is generated from the training image, considering the frequency of replicas of the present data event (Strebelle 2000, 2002).

In SNESIM, the training image is scanned only once; for each particular search template, all the conditional proportions present in the training image are stored in a smart search tree data structure to be retrieved as fast as possible in the next steps. SNESIM is composed of two main parts: (1) construction of search tree to save all probability proportions, and (2) simulation section where these proportions are read from the tree and used for simulation of unknown values.

SNESIM can handle only categorical variables such as facies distributions or discrete fracture networks, and is not able to use continuous data. This algorithm takes a large amount of memory especially in the case of large training images containing many various classes of different patterns; thus SNESIM can make only use of a limited number of classes (<5). This algorithm appears suitable for modeling discrete fracture networks. However, using the pattern-based algorithms instead can result in more accurate final realizations. In addition, using quite large scanning templates to capture the large-scale geological features increases the levels of constructed search tree and hence more memory will be required.

Another limitation of SNESIM code is the cost of retrieving the conditional probability when the conditional data event is not fully informed (having missing data locations). At each such missing data location, the SNESIM algorithm will traverse and trace all its children branches in the search tree for all possible data values. This cumbersome tree search is extremely memory and CPU consuming. Indeed, any SNESIM simulation is very slow at the beginning, and gets faster when more and more previously simulated nodes become conditioning data (Hu and Chugunova 2008; Straubhaar 2008; Mariethoz 2010). Non-stationarity problems are the other major drawbacks of the SNESIM algorithm (Chugunova and Hu 2008; DeVries 2009).

To solve the shortcomings of SNESIM algorithm, Caers and Arpat (2005) presented a new approach based on the statistical pattern recognition concepts, named as SIMPAT, in which the simulation is performed using the extracted patterns from the training image (Caers and Arpat 2005, 2007).

Firstly, the training image *TI* is scanned through the search template T to capture all the available patterns present. A filter may be applied to avoid keeping undesired or very similar extracted patterns. The filtered patterns are then stored in a database to be retrieved in the next simulation steps.

During the sequential simulation employed by SIMPAT, each unknown node u is randomly visited and the local conditioning data event $\mathrm{dev}_T(u)$ is recorded simultaneously. Afterward, $\mathrm{dev}_T(u)$ is compared to all available patterns in the database according to a predefined similarity criterion; the most common of which are distance functions. The purpose is to find out the most similar training pattern pat^*_T to the local data event $\mathrm{dev}_T(u)$. In other words, the algorithm evaluates the distance d for all existing patterns and picks up the pattern with the smallest distance as pat^*_T. Having selected the most similar pattern, $\mathrm{dev}_T(u)$ is replaced by pat^*_T such that part of the values inside pat^*_T is pasted onto the simulation grid centered at the current location u (Caers and Arpat 2005).

Due to global search performed on the training patterns database to find the most similar pattern to the data event, SIMPAT suffers from a high computational load (Honarkhah and Caers 2010). This is because of the large number of stored patterns and their high-dimensional feature space in the case of using large scanning templates. A new algorithm known as FILTERSIM was presented by Zhang in 2006 to alleviate the standard SIMPAT's computational intense (Journel and Zhang 2006). The default FILTERSIM starts to decompose each training pattern to a small number of linear filter scores (6 in 2D and 9 in 3D) each of which is a weighted average of the pattern features. These transformed training patterns will then be classified in the new filter space; the aim of clustering is to group the identical patterns into separate classes so that the similarity between them is defined as their distances to each other. The clustering is performed only once for each TI prior to the simulation stage. The sequential simulation approach is adopted later on to see the unknown nodes in a random manner. For each unknown node, the local conditioning data event is compared to the training cluster prototypes in order to select the closest one based on the calculated distance measure. A training pattern is picked up from the selected class in a random manner or conditioned to the available soft data; an inner part of the pattern (called "patch") is then fixed at the simulation grid centered at the current node; their location addresses are then removed from the predetermined random path and the remaining nodes will be replaced by the future evaluated values. Like all the geostatistical algorithms, hard data conditioning is achieved through freezing the corresponding nodal values at their locations not to be changed ever during the simulation step; these primary values are preferentially used in estimating the dissimilarity distances and therefore ensure us that no pattern involving the structural discontinuities will be selected (Wu 2007).

Contrary to the first multiple-point simulation algorithm, SNESIM, in which the *single nodal values* are modeled using the pattern data at each simulation step, SIMPAT and FILTERSIM evaluate *patterns* from the available pattern data. Compared to SNESIM, FILTERSIM is not limited to categorical variables and can be run with continuous values as well. FILTERSIM looks like the SIMPAT algorithm in many aspects, except that the latter does not perform any clustering on the extracted patterns, but searches for the pattern with the most similarity with the multiple-point data event.

Clustering in FILTERSIM

Clustering methods haves gained a great application in many areas such as data mining, document retrieval, image segmentation and pattern classifications. The purpose is to group the patterns according to a similarity (or dissimilarity) criterion so that each cluster represents a set of similar patterns. Each pattern is denoted by a set of system characteristics. There are several books published on classical data clustering (Anderberg 1973; Jain and Dubes 1988; Duda et al. 2001). Clustering algorithms have also been extensively studied in data mining and machine learning books (Han and Kamber 2000; Tan et al. 2005; Bishop 2006). A good survey of different clustering algorithms could be found in Xu and Wunsch (2005), Jain (2010).

After implementing the preprocessing step of FILTERSIM algorithm, it is required to utilize the appropriate similarity criteria between the patterns; the most common similarity measure of standard representation is "distance" (Perner 2007). The distance function used in this paper is of Euclidean type. Other functions could be also applied in this manner as fully described in Ahmadi (2011).

Clustering algorithms try to reorganize the set of training patterns in the database in the form of several separate classes based on a predefined similarity measure. Dimensionality of patterns may have been reduced prior to the clustering. Having accomplished the clustering stage, the patterns in a particular partition are much similar to each other than the patterns in other groups. K-means with a rich and diverse history as it was independently discovered in different scientific fields (Steinhaus 1956; Lioyd, proposed in 1957 but published in 1982; Ball and Hall 1965; MacQueen 1967) is one of the most commonly used clustering algorithms that was applied by Journel and Zhang (2006) in the multiple-point simulation approaches to partition the obtained normal filter scores. Standard partitional clustering methods (such as K-means, fuzzy C-means, Self Organized Maps (SOM) and neural gas) consider the linear borders between the formed clusters and are not able to separate the complex and non-linear data. Other methodologies have been presented by the ability of constructing the separable hypersurfaces. These algorithms are classified into two main categories: kernel-based and Spectral clustering methods.

In the case of kernel-based algorithms, several clustering techniques such as K-means, C-means and SOM have been widely used with the aid of kernel functions. Using kernels, data will be transformed into a new feature space of higher dimension where data could be linearly separated. Separability is performed easier in the higher-dimensional space. This change of coordinate system maintains the structural principles inside data; but it alters the way in which they are represented, so that the principles will be discriminated more straightforwardly. In other words, the non-linear relationship between the information inside data is able to be captured via non-linear mapping of data to a new higher-dimensional space. Correspondingly, the classical clustering algorithms such as K-means acts better in the new space and a more accurate and powerful clustering is achieved henceforth (Shawe and Cristianini 2004; Filippone et al. 2008).

Using the kernel-based clustering methods, however, requires the prior determination of the kernel function and its defining parameters. As an instance, in the case of choosing the Gaussian kernel as the base function, we should try in finding the optimal value of the width; since this parameter which is equivalent to the Gaussian standard deviation indicates the openness of the Gaussian function and the neighborhood area affected by the defined kernel that will influence the final clustering results significantly. However, due to unsupervised nature of clustering algorithms, optimality of kernel parameters does not appear so easily; with the manifold assumption of data, the spectral algorithms could therefore be applied in order for the non-linear pattern clustering.

Spectral clustering algorithms take advantage of graph theory concepts. The main idea is to construct a weighted graph from the initial data set; each node represents a pattern while the weighted edges are simply the similarity degrees between the two patterns on the heads. In this framework, the clustering could be considered like a graph cut problem which may be solved through the spectral graph theory. The fundamental idea is the eigenvalue decomposition of the Laplacian matrix of the weighted graph obtained from data; in reality, there is a close relationship between the second smallest eigenvalue of the Laplacian and the graph cut problem (Chung 1997; Cristianini et al. 2001).

Dimensionality reduction in FILTERSIM

Scanning a training image using a large template size brings about a high dimensionality of the extracted patterns; a preprocessing step would therefore be required to reduce data dimension as much as possible and select part of the features optimally; running clustering algorithms on such a high-dimensional dataset demands for a large computational burden and time.

Dimensionality reduction may be achieved in FILTERSIM through the following classifications:

1. Using image statistics and application of predefined filters to the training patterns,
2. Extraction of unique features from the training patterns or using feature transformations such as features generated from discrete Fourier power spectrum of each pattern,
3. Using common dimensionality reduction techniques such as principal component analysis (PCA), linear discriminant analysis (LDA) and discrete cosine transform (DCT).

The main contribution of this paper is to make use of (1) K-Nearest Neighbor Index (a simple but very common filter, used in the pattern recognition and spatial data mining tasks (Cover and Hart 1967; Toussaint 2005); introduced from the first classification above), (2) PCA (a very common technique in data analysis practices (Jackson 1991; Jolliffe 2002); proposed from the third classification above), and finally (3) DCT (a common transform used in digital image applications (Jain 1989; Pennebaker and Mitchell 1993); also brought from the third classification above). Other methods that are not tried in this paper may be the subject of future works.

Filters

A filter is composed of a set of weights relevant to the search template nodes T_J. The filter value or the weight of node u_j of the template is denoted by f_j; therefore, a J-node filter is defined as $\{f(h_j); j = 1, 2, 3, \ldots, J\}$, where h_j represents the coordinates of the search template T_J with respect to the central node u. The filter is applied to each local training pattern, *pat*, centered at node u in the training image. Application of F filters simultaneously would result in constructing a vector of F filter scores that summarizes any of available patterns in F new dimensions:

$$S_T(u) = \sum_{j=1}^{n} f_i(h_j).\mathrm{pat}(u + h_j); i = 1, 2, \ldots., F \tag{1}$$

where, $S_T(u)$ denotes the computed filter score and h_j is the offset of each template node from the central node u.

Representation of a pattern with its relevant filter score vector causes a severe dimensionality reduction from the initial pattern size $(n_x \times n_y)$ to F dimensions. Figure 1 shows the process of filtering a 2D training pattern and construction of the corresponding filter score value (Wu 2007).

Instead of averaging the values inside the training patterns through a specific filter, we may summarize each pattern by performing mathematical and statistical operations on their contents. This would be done through the application of *data summaries*.

In general, each pattern extracted from a fracture network training image could be interpreted in three ways: (1) a digital image, (2) a spatial phenomenon, or (3) a fracture phenomenon; regarding each perspective, the appropriate data summary might be designed for the FILTERSIM algorithm:

1. As a digital image, patterns could be defined by their statistical moments of different orders; mean as the first moment and variance with respect to the mean as the second one are common statistics of a random phenomenon. Summarizing each pattern in a small set of its statistical measures might have the potential of reducing dimensionality of the existing patterns in an appropriate and unique manner so that patterns of different occurrences of available categories at their different nodal locations are easily distinguished from each other in the new coordinate system of the selected summaries. No overlapping is expected for the originally separated patterns in the new dimensionality space. Where required, using a more number of efficient measures may benefit to the accuracy of summarizing practice.
2. From the spatial data mining views, the spatial correlation indices such as nearest neighbors index (NNI) (Cover and Hart 1967; Toussaint 2005) and Moran-I parameter (Moran 1950) are suggested to summarize the patterns as the spatial phenomena. To achieve a better and more accurate summarization of training patterns in this mode, it may be practical to

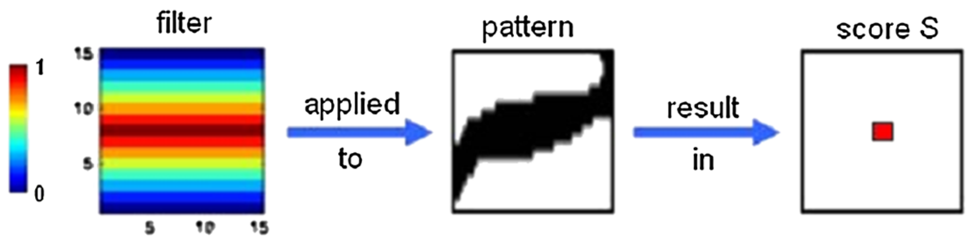

Fig. 1 Filtering process of a training pattern (Wu 2007)

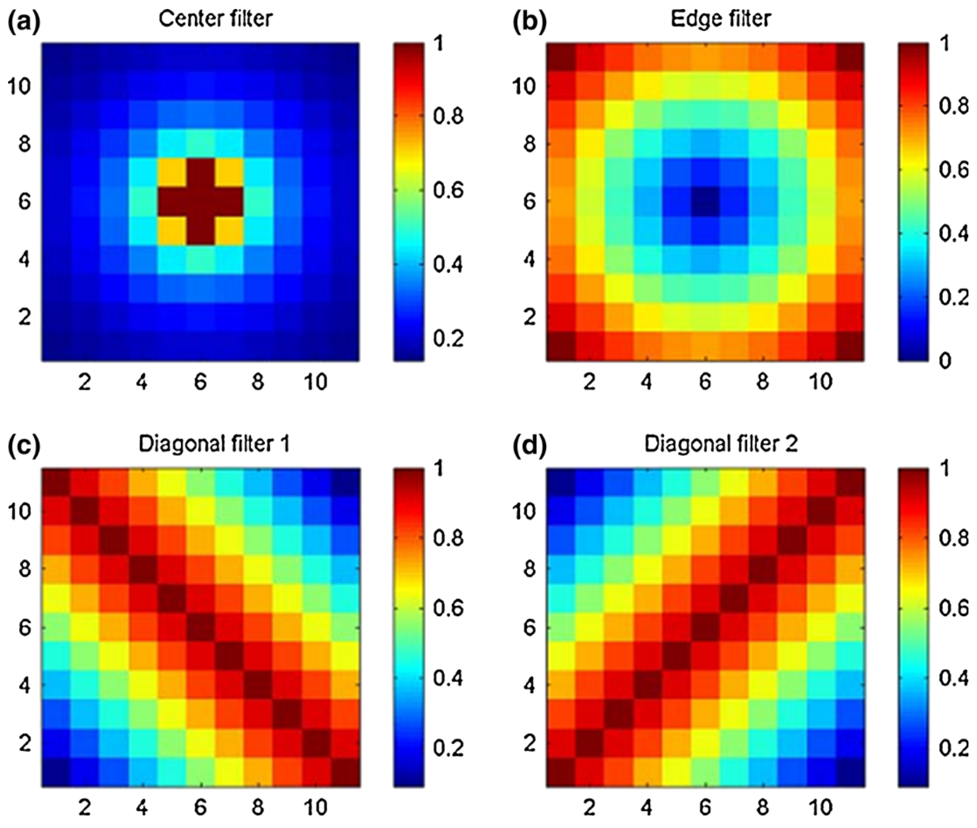

Fig. 2 Four user-defined 2D filters of size 11 × 11 (Wu 2007)

use farther points in the neighborhood of fracture cells inside each pattern.

3. As a fracture network system, it is reasonable to define some new summaries based on the fractures connectivity and the orientation of the fractures along the straight lines.

Default FILTERSIM algorithm makes use of three filters (average, gradient and curvature) in each coordinate direction (Journel and Zhang 2006). The user could instead design its own filters depending on the features of interest in the training image that should be reproduced in the simulation. For instance, the isotropic or oriented filters can have great applications in the case of particular geological structures involving different orientations, such as fracture networks. Examples are central, edge, diagonal I and diagonal II filters shown in Fig. 2 (Wu 2007).

In this paper, we have designed two specific data summaries exclusive to fracture networks including connectivity and direction of individual fractures to obtain a more accurate geological system involving more realistic and connected fractures.

New data summaries for fracture networks modeling

One of the most important and challenging issues in modeling fracture networks and other geological structures like channel reservoirs is the connectivity of final realization so that disconnectivity of simulated nodes belonging to a particular structure is minimal.

Two schematic fracture networks of different connectivities are shown in Fig. 3. Based on a new definition, connectivity of any specific fracture node in the template is defined as the number of fracture nodes to which it is connected immediately. Degree of connectivity of the whole fracture network would then be equal to the sum of connectivities of all fracture nodes existing in the network.

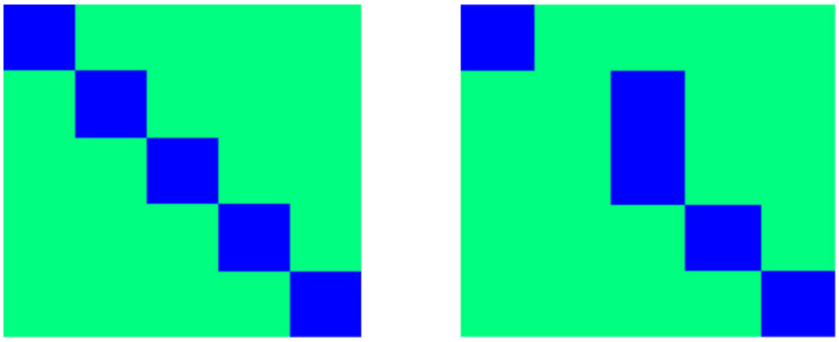
Fig. 3 Two 5 × 5 fracture pattern networks with different connectivities; the blue nodes are in fracture and the *green color* denotes the matrix

The application of connectivity criterion is ranking and selection of the best patterns. Amongst the extracted patterns, the one with the larger degree of connectivity will be won to be selected. How to calculate the connectivity of each pattern network and use the estimated scores consequently as a selection criterion is demonstrated below for the two typical patterns illustrated in Fig. 3. The left-hand-side calculations denote the connectivity of each fracture node in the left pattern, while the right-hand-side computations evaluate the connectivity of fracture nodes for the right pattern:

$$N\{TI(1,1)\} = 1 \quad N\{TI(1,1)\} = 0$$

$$N\{TI(2,2)\} = 2 \quad N\{TI(2,3)\} = 1$$

$$N\{TI(3,3)\} = 2 \quad N\{TI(3,3)\} = 2$$

$$N\{TI(4,4)\} = 2 \quad N\{TI(4,4)\} = 2$$

$$N\{TI(5,5)\} = 1 \quad N\{TI(5,5)\} = 1$$

$$N_{G1} = \sum_{i,j \in F} N\{TI(i,j)\} = 8 \; N_{G2} = \sum_{i,j \in F} N\{TI(i,j)\} = 6$$

where F denotes the set of nodes belonging to the fracture class; TI is the value of current node in the training image TI, and N equals the number of neighborhood nodes in the fracture. For example, the fracture node with the coordination (2,2) in the left pattern in Fig. 3 connects immediately to fracture nodes of coordinations (1,1) and (3,3) in the same pattern. So connectivity of the mentioned fracture node is evaluated as 2. In Fig. 3, since connectivity measure (N_G) of the fracture pattern in the left is greater than that of the right network, the former is selected as the preferable pattern; as it is also clear from Fig. 3, connectivity of the left pattern is visually more than the right one.

Now consider the third fracture pattern in Fig. 4; although it seems visually to be less proper than the pattern in the left part of Fig. 3, however, they have the same degree of connectivity measures mathematically.

$$N_{G3} = N_{G1} = 8$$

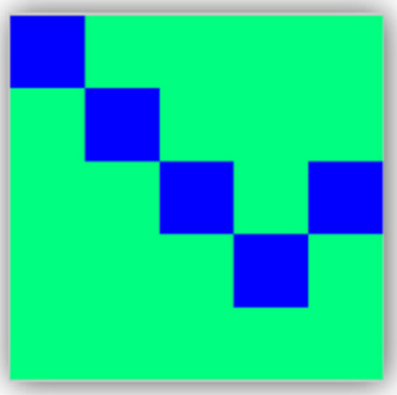
Fig. 4 **a** 5 × 5 pattern of different disconnectivity than the pattern in the left side of Fig. 3; the *blue nodes* are in fracture and the *green color* denotes the matrix

To solve this problem, a new measure could be designed by estimating deviation of the fracture nodes from a straight line. This really investigates orientation of fracture pixels along a specific extension (including all diagonals of the pattern matrix). Regarding the disconnectivity quantity, the left pattern in Fig. 3 will be selected rather than the network in Fig. 4, because the fracture nodes in the first pattern are located only along one diagonal (the main diagonal), while in the second pattern fractures are placed alongside the two diagonals and hence have a more disconnectivity than the first one does. To do a right comparison in the case of all possible fracture patterns, it is better to consider the total number of network diagonals with at least a fracture node on those.

Simulation and results

General FILTERSIM algorithm could be broken into four steps including (1) training image scanning, (2) extracted patterns filtering, (3) pattern space clustering, and (4) simulation stage. Simulation is performed through the sequential simulation algorithm as described previously. A sensitivity analysis is, however, performed on the first three steps of FILTERSIM algorithm as explained below:

1. The training image is scanned four times using four square templates of different sizes including 17, 21, 25 and 29 grids on each edge.
2. Clustering algorithms require a relatively large computational time when applied to the pattern database of a very high dimension (depending on the size of search template). It is therefore strongly suggested to perform a preprocessing step on the pattern database and reduce the patterns dimensionality as much as possible. Dimensionality reduction of the extracted patterns, prior to the clustering stage, has been achieved through the application of three different techniques including PCA, KNN and DCT. DCT is applied to each training pattern, while the other two techniques are executed once to the whole set of database. It should be noted that in the case of using 2D DCT as a dimensionality reduction technique, those coefficients will be selected as the transformed features that contain 99 % of the total energy of the whole pattern image. To see the effect of filtering on both running times and accuracy of the simulation algorithm, the models obtained through the above filtering techniques will be compared with the cases where no filtering has been done on the patterns (labeled as FILTERLESS case).
3. It is a common practice in FILTERSIM to group the identical patterns in several clusters prior to the distance minimization step. Using two common clustering methods, K-means and Spectral techniques, partitioning the hard training patterns database and then the modeling tasks are elaborated. Three different numbers of clusters including 1/200, 1/400 and 1/800 of the total number of training patterns extracted from the training image have been used in the clustering algorithm.

Therefore, a complete sensitivity analysis will be performed on the FILTERSIM algorithm, regarding size of the scanning template (17, 21, 25 and 29), type of clustering technique (K-means and Spectral), the number of clusters (1/200, 1/400 and 1/800 of the total number of training patterns) and type of filtering method (PCA, KNN, DCT and Without-Filter). As a result, 48 cases of FILTERSIM runs will be implemented for each clustering method.

Figure 5 shows an outcrop map of a naturally fractured reservoir in south west region of Iran. This site is located in the flank of an anticline structure, next to its nose. The major feature of this site is a large transverse fault observed at the right side of the map. In addition, there are several normal faults as well as many conjugate fractures more or less parallel to the local fold axis. The site is affected by two main joint sets perpendicular to the bedding, oriented at NNE and NS directions. The sensitivity analysis on FILTERSIM algorithm is performed using this training image.

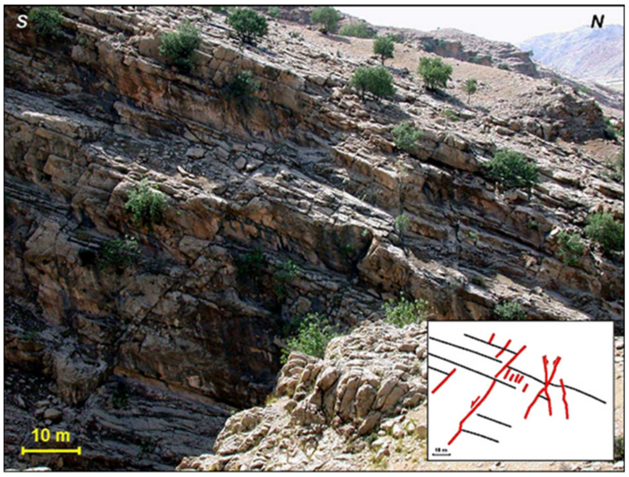

Fig. 5 An outcrop map of a naturally fractured reservoir in south west region of Iran

As an important notation, K-means clustering algorithm may be stopped at the middle iterations due to convergence issues posed by improper initial random assignment of data points to the clusters. Two methods have been adapted to choose the initial cluster centroid positions, sometimes known as seeds. The first approach selects k observations from the dataset X at random. Selecting k points uniformly at random from the range of X is the second method used in the present K-means clustering. Another approach is to perform a preliminary clustering phase on a random 10 % subsample of X. This phase is itself initialized using the first technique. This method is, however, not used in the present study.

In addition, due to dimensionality reduction incident to the Spectral clustering algorithms themselves (where a limited number of eigenvectors of the affinity matrix is to be selected and partitioned), they are expected to have a higher computational speed with respect to the K-means algorithm and thus appear to be more suitable in modeling large fracture networks in this view. This will be tracked during the sensitivity analysis too.

Results comparison

Simulation results could be evaluated through a comparison between the spatial patterns based on several quantitative ways such as MPS, flow simulations or even semi-variograms. There are many different types of MPS measures such as connectivity function, lacunarity, the multiple-point density function (MPDF), transition probabilities and distribution of runs (Boisvert et al. 2008; Pyrcz et al. 2008). Flow simulation curves of oil recovery factor have been adapted in this work to be used in verification of results.

A five-spot pattern water flooding with 4 injectors at the corners and 1 producer at the center of model is simulated to obtain the oil recovery factor curve as model performance. Such a flow simulation is designed for each realization as well as the original training image. The simulation is done using a commercial ECLIPSE100 simulator, while it is linked to the MATLAB programming environment to read the images, do the required preprocessing, pass them to the simulator and analyze and post-process the output results. An E-type is generated using 10 FILTERSIM realizations for each case of simulations. Each continuous E-type is converted to a binary image by a threshold of 0.5. The final image to be passed to the flow simulator is therefore a two-class fracture network. The nodes in the fractures are devoted 1,000-millidarcy permeability and full porosity of 1, while the background facies (i.e., non-fracture locations) are assumed to have 10-millidarcy permeability and porosity of 0.1. The training image has a size of 69 × 104 cells in XY coordinates (Fig. 5). The four injection wells are drilled in (1, 1), (69, 1), (1, 104) and (69, 104) coordinates, while the production well is located at (32, 68) coordinate. The wells are allowed to inject or produce for 7,500 days from the beginning. The mean squared error (MSE) values are evaluated between the recovery factor of the original training image and that of each FILTERSIM case.

Description of figures and tables

The final results of sensitivity analysis include the computational times as well as the MSE values (as the measures of accuracy) for the implemented simulations. These data are provided in Tables 1, 2, 3 for the cases that have used K-means clustering method and in Tables 4, 5, 6 for the simulations performed through the Spectral clustering techniques. Simulation times of different parts of FILTERSIM algorithm (including scanning, filtering, clustering and simulation procedures) and MSE of recovery data are shown in these Tables. The errors are displayed in the last column of Table 1 for the realizations obtained through application of K-means clustering technique and in the last column of Table 4 for those realizations modeled using Spectral clustering method. It should be noted that the simulation parameters as in the second columns of the mentioned Tables indicate the order of "filtering method, search template size, ratio of cluster numbers", respectively; for example, the term "PCA, 25, 400" means a FILTERSIM case where filtering is done by PCA, search template size is 25 in each direction on a squared grid and the number of clusters used is equal to 1/400 of the total number of training patterns.

Ranking of different simulation cases based on running times of different sections (including scanning, filtering, clustering and simulation procedures) for realizations obtained through application of K-means clustering technique is displayed in Table 2. The same results, but in the case of applying Spectral technique, are provided in Table 5. Each column of Tables 2 and 5 indicates the index of a specified running time in an ascending order. The numbers indicate the row index of each case as specified in Tables 1 and 4. The last column displays label of simulation case corresponding to ranking results of total time of simulation.

In addition, Tables 3 and 6 show ranking results of different FILTERSIM cases based on MSE values of production recoveries in the case of K-means and Spectral clustering, respectively.

Figure 6 illustrates the most and the least accurate results of sensitivity analysis for the cases generated by applying K-means and Spectral clustering techniques. Figure 6a shows the binary training image that has been used in the FILTERSIM simulations for the analysis. It is actually the digital form of the analog outcrop map displayed in Fig. 5. Red nodes indicate fracture locations whereas blue nodes are in background facies (i.e., non-fracture locations). It has a size of 69 × 104 grids on a two-dimensional coordinate. The modeling area has been built through selection of 300 data points from the training image at random. Figure 6b displays the least accurate E_type obtained through application of K-means clustering algorithm during sensitivity analysis. As it is clear in Table 3, the least accurate result of K-means cases is obtained using KNN as filtering method, a search template of size 17 × 17, and a number of clusters equal to 1/800 of the total number of training patterns (i.e., a total number of 7,176 scanned patterns). Figure 6c illustrates the most accurate E_type when using K-means clustering. This is achieved without any filtering, using a search template of size 21 × 21, and a number of clusters equal to 1/200 of the total number of training patterns. The least accurate E_type of FILTERSIM using Spectral clustering method is demonstrated in Fig. 6d where it comes from KNN as filtering method, a search template of size 29 × 29, and a number of clusters equal to 1/200 of the total number of training patterns. Finally, the most accurate E_type of Spectral cases is shown in Fig. 6e; using PCA as filtering method, a search template of size 17 × 17, and a number of clusters equal to 1/200 of the total number of

Table 1 Simulation times of different sections (including scanning, filtering, clustering and simulation procedures) and mean squared errors of recovery data for realizations obtained through application of K-means clustering technique

Row	Simulation parameters (label)	Scanning time (s)	Filtering time (s)	Clustering time (s)	Simulation time, (s)	Total time (s)	Recovery MSE
1	pca 17,200	114.6	1.4	148.7	119.3	384.1	5.00E−04
2	pca 17,400	111.5	1.1	104.3	93.6	310.5	5.00E−03
3	pca 17,800	110.7	1.7	98.8	71.8	282.9	9.00E−04
4	pca 21,200	134.7	3.2	185.5	140.4	463.9	4.70E−03
5	pca 21,400	128.5	3.1	164.9	103.1	399.6	2.00E−04
6	pca 21,800	126.8	3.0	96.5	93.0	319.3	4.00E−04
7	pca 25200	168.5	6.8	238.5	185.8	599.6	1.00E−04
8	pca 25,400	168.5	6.8	160.1	180.9	516.2	1.00E−04
9	pca 25,800	170.9	6.9	112.0	115.5	405.4	3.00E−04
10	pca 29,200	219.0	14.5	331.1	302.6	867.1	2.90E−03
11	pca 29,400	217.9	14.4	235.6	212.7	680.7	5.90E−03
12	pca 29,800	218.7	15.0	263.8	222.9	720.3	3.00E−04
13	knn 17,200	104.3	13.5	60.8	61.1	239.7	1.40E−03
14	knn 17,400	113.0	13.4	69.2	65.7	261.4	4.90E−03
15	knn 17,800	111.7	12.9	71.3	68.0	264.0	6.40E−03
16	knn 21,200	130.4	21.8	65.4	64.9	282.5	8.00E−04
17	knn 21,400	134.4	22.2	69.1	63.6	289.3	5.10E−03
18	knn 21,800	130.2	22.5	67.6	62.3	282.6	1.00E−03
19	knn 25,200	172.7	29.2	84.4	76.9	363.1	1.00E−04
20	knn 25,400	169.3	29.1	82.6	75.3	356.2	5.00E−04
21	knn 25,800	168.6	29.0	83.5	76.3	357.5	2.00E−04
22	knn 29,200	224.4	40.1	107.5	97.7	469.7	5.90E−03
23	knn 29,400	226.4	40.0	108.0	98.1	472.5	2.80E−03
24	knn 29,800	222.9	40.0	107.0	97.3	467.2	7.00E−04
25	dct 17,200	112.2	9.5	69.1	64.0	254.7	3.50E−03
26	dct 17,400	106.8	9.5	67.0	63.0	246.3	7.00E−04
27	dct 17,800	112.1	9.4	67.7	64.4	253.6	5.00E−04
28	dct 21,200	132.9	15.7	67.5	62.3	278.4	5.00E−04
29	dct 21,400	138.3	15.9	67.8	62.9	284.9	4.00E−04
30	dct 21,800	138.1	15.8	68.0	62.5	284.4	5.00E−04
31	dct 25,200	172.9	23.3	83.7	75.9	355.8	6.00E−04
32	dct 25,400	178.5	23.2	85.8	78.4	365.9	2.00E−04
33	dct 25,800	172.2	25.2	85.8	78.2	361.4	3.00E−04
34	dct 29,200	259.3	34.0	120.5	108.2	522.0	1.00E−04
35	dct 29,400	226.8	33.6	106.1	96.3	462.7	5.10E−03
36	dct 29,800	249.6	34.9	117.7	105.4	507.6	2.00E−04
37	filterless 17,200	105.8	0.0	180.4	214.9	501.1	4.00E−04
38	filterless 17,400	112.0	0.0	185.3	146.5	443.7	9.00E−04
39	filterless 17,800	127.6	0.0	111.1	115.1	353.8	9.00E−04
40	filterless 21,200	143.7	0.0	304.5	255.2	703.5	0.00E+00
41	filterless 21,400	128.4	0.0	176.0	172.7	477.2	2.00E−04
42	filterless 21,800	126.4	0.0	117.2	98.3	341.9	4.50E−03
43	filterless 25,200	173.1	0.0	359.5	361.4	894.0	1.00E−04
44	filterless 25,400	172.6	0.0	291.6	314.3	778.6	2.90E−03
45	filterless 25,800	171.8	0.0	172.3	187.7	531.7	8.00E−04
46	filterless 29,200	220.9	0.0	542.4	484.9	1,248.2	2.90E−03
47	filterless 29,400	217.4	0.0	470.6	235.8	923.9	2.50E−03
48	filterless 29,800	215.5	0.0	351.5	177.7	744.7	2.90E−03

Table 2 Ranking of different simulation cases based on running times of different sections (including scanning, filtering, clustering and simulation procedures) for realizations obtained through application of K-means clustering technique

Scanning time	Filtering time	Clustering time	Simulation time	Total time	Simulation method
40	26	22	22	1	pca 17,200
39	1	21	20	2	pca 17,400
14	46	17	16	3	pca 17,800
9	10	36	34	4	pca 21,200
7	15	23	23	5	pca 21,400
41	3	31	31	6	pca 21,800
11	33	11	45	7	pca 25,200
32	18	37	48	8	pca 25,400
8	14	39	42	9	pca 25,800
23	36	48	10	10	pca 29,200
12	21	7	11	11	pca 29,400
46	19	44	12	12	pca 29,800
13	37	13	13	13	knn 17,200
2	48	18	29	14	knn 17,400
15	41	27	26	15	knn 17,800
27	40	30	25	16	knn 21,200
6	2	20	3	17	knn 21,400
25	43	25	27	18	knn 21,800
30	27	35	35	19	knn 25,200
28	7	6	32	20	knn 25,400
17	8	3	6	21	knn 25,800
33	28	1	9	22	knn 29,200
44	30	8	1	23	knn 29,400
45	12	34	39	24	knn 29,800
3	47	28	30	25	dct 17,200
37	38	16	28	26	dct 17,400
26	39	26	18	27	dct 17,800
38	42	29	17	28	dct 21,200
42	45	15	15	29	dct 21,400
1	44	14	14	30	dct 21,800
16	4	33	33	31	dct 25,200
29	25	24	24	32	dct 25,400
4	9	2	2	33	dct 25,800
48	32	38	8	34	dct 29,200
20	11	42	36	35	dct 29,400
43	17	41	41	36	dct 29,800
31	16	45	38	37	filterless 17,200
21	13	9	5	38	filterless 17,400
18	5	32	19	39	filterless 17,800
10	20	12	37	40	filterless 21,200
19	29	5	4	41	filterless 21,400
5	6	19	21	42	filterless 21,800
35	24	43	44	43	filterless 25,200
22	34	10	40	44	filterless 25,400
47	31	4	7	45	filterless 25,800
34	22	46	46	46	filterless 29,200
36	23	47	43	47	filterless 29,400
24	35	40	47	48	filterless 29,800

Table 3 Ranking of different simulation cases based on flow simulation recoveries for realizations obtained through application of K-means clustering technique

Simulation method	MSE
filterless 21,200	0.00E+00
dct 29,200	1.00E−04
pca 25,400	1.00E−04
filterless 25,200	1.00E−04
pca 25,200	1.00E−04
knn 25,200	1.00E−04
pca 21,400	2.00E−04
dct 25,400	2.00E−04
knn 25,800	2.00E−04
filterless 21,400	2.00E−04
dct 29,800	2.00E−04
pca 29,800	3.00E−04
pca 25,800	3.00E−04
dct 25,800	3.00E−04
filterless 17,200	4.00E−04
pca 21,800	4.00E−04
dct 21,400	4.00E−04
knn 25,400	5.00E−04
dct 21,200	5.00E−04
pca 17,200	5.00E−04
dct 21,800	5.00E−04
dct 17,800	5.00E−04
dct 25,200	6.00E−04
knn 29,800	7.00E−04
dct 17,400	7.00E−04
filterless 25,800	8.00E−04
knn 21,200	8.00E−04
filterless 17,400	9.00E−04
pca 17,800	9.00E−04
filterless 17,800	9.00E−04
knn 21,800	1.00E−03
knn 17,200	1.40E−03
filterless 29,400	2.50E−03
knn 29,400	2.80E−03
filterless 29,800	2.90E−03
filterless 29,200	2.90E−03
pca 29,200	2.90E−03
filterless 25,400	2.90E−03
dct 17,200	3.50E−03
filterless 21,800	4.50E−03
pca 21,200	4.70E−03
knn 17,400	4.90E−03
pca 17,400	5.00E−03
knn 21,400	5.10E−03
dct 29,400	5.10E−03
knn 29,200	5.90E−03
pca 29,400	5.90E−03
knn 17,800	6.40E−03

training patterns has led to the most accurate result obtained through application of Spectral clustering. Figure 6 provides only a visual inspection into sensitivity analysis performed. Other figures, however, exhibit the complete results including running times and MSE values evaluated.

Running times of different simulation stages and mean squared errors of production recoveries of different realizations are plotted in Figs. 7 and 8, respectively, in the case of K-means clustering technique and in Figs. 9 and 10 for models generated by Spectral method. The vertical axes in Figs. 7 and 9 show time of simulation in seconds, for each quarterly stages; while the horizontal axes denote numeric index of each FILTERSIM case as labeled in Tables 1 and 4.

The vertical axes on Figs. 8 and 10, on the other hand, show MSE values for each simulation stage; numeric index of each FILTERSIM case as labeled in Tables 1 and 4 is displayed on the horizontal axes, however. These two graphs are simultaneously plotted on Fig.11 with the same axes description as before for a better comparison between K-means and Spectral clustering methods.

Results and discussions

As observed from tables of data (Tables 1, 4) and Figs. 7 and 9, scanning process takes the most running time in a simulation procedure. Clustering and simulation sections stand on the second and third stages, respectively, in this view point. Filtering process, however, takes a small time compared to other sections and does not have a substantial impact on total running time. Yet, the filter type utilized may have a considerable effect on clustering and simulation running times as well as on the accuracy of final realizations. This is a valid statement in all cases including use of different filtering and clustering methods through application of different-sized scanning templates.

All running times of scanning, filtering, clustering and simulation procedures increase as search template size is growing. This could be easily followed up by looking into the Figs. 7 and 9.

The number of clusters used in Spectral clustering scheme does not have a considerable effect on any parts of running time, even on clustering section; on the contrary, an increase in the number of clusters decreases computational load of K-means clustering either when the training patterns stored in the database have been filtered through PCA method or they have not been filtered anyway. Accordingly, total time of simulation decreases as a lower number of partitions is used to group the training patterns. Running times of simulations utilizing DCT or KNN techniques in filtering section are

Table 4 Simulation running times of different sections (including scanning, filtering, clustering and simulation) and mean squared errors of recovery data for realization obtained through application of Spectral clustering technique

Row	Simulation parameters	Scanning time (s)	Filtering time (s)	Clustering time (s)	Simulation time (s)	Total time (s)	Recovery MSE
1	pca 17,200	88.9	1.0	89.0	80.7	259.6	4.60E−05
2	pca 17,400	91.4	1.0	90.8	82.1	265.4	1.12E−04
3	pca 17,800	91.1	1.1	90.0	81.4	263.7	3.42E−03
4	pca 21,200	122.8	2.4	112.0	100.7	337.9	2.11E−04
5	pca 21,400	126.2	2.6	114.3	103.5	346.6	1.18E−04
6	pca 21,800	126.1	2.5	113.9	102.8	345.3	1.68E−04
7	pca 25,200	168.4	6.2	145.2	130.5	450.3	3.18E−04
8	pca 25,400	166.2	6.2	143.4	129.3	445.1	2.24E−04
9	pca 25,800	168.2	6.1	143.6	129.1	447.1	1.09E−04
10	pca 29,200	212.0	13.1	177.6	159.5	562.2	4.76E−05
11	pca 29,400	215.9	13.1	180.8	161.4	571.2	1.53E−03
12	pca 29,800	223.4	14.8	184.3	165.2	587.7	1.37E−03
13	knn 17,200	92.3	13.8	71.0	65.0	242.1	2.31E−04
14	knn 17,400	90.5	12.9	71.3	65.6	240.3	5.77E−05
15	knn 17,800	91.3	13.2	71.6	65.6	241.7	5.68E−05
16	knn 21,200	127.3	18.9	85.8	78.1	310.0	1.28E−04
17	knn 21,400	128.6	18.1	86.0	78.5	311.1	2.28E−03
18	knn 21,800	131.2	18.6	86.6	79.0	315.3	2.26E−04
19	knn 25,200	171.1	24.2	103.2	93.4	391.9	2.29E−04
20	knn 25,400	167.1	24.0	101.1	91.3	383.4	3.57E−04
21	knn 25,800	167.3	24.2	101.1	91.5	384.0	4.74E−04
22	knn 29,200	214.7	33.4	121.4	110.1	479.7	3.75E−03
23	knn 29,400	212.9	33.4	121.7	109.6	477.6	2.55E−03
24	knn 29,800	213.4	33.4	121.6	109.9	478.3	2.41E−04
25	dct 17,200	91.2	8.9	72.5	66.0	238.5	1.32E−04
26	dct 17,400	90.4	8.9	72.8	66.6	238.8	8.69E−05
27	dct 17,800	90.1	9.0	71.6	65.5	236.1	2.34E−03
28	dct 21,200	124.9	13.8	87.3	79.1	305.1	7.36E−05
29	dct 21,400	124.6	13.7	87.0	79.1	304.4	1.49E−04
30	dct 21,800	124.3	13.8	86.6	78.6	303.3	1.38E−04
31	dct 25,200	166.2	20.5	105.4	95.0	387.1	1.19E−04
32	dct 25,400	165.7	20.5	105.5	95.5	387.2	4.40E−04
33	dct 25,800	166.6	20.5	105.1	95.4	387.6	8.14E−05
34	dct 29,200	215.0	29.6	126.8	114.3	485.7	2.17E−03
35	dct 29,400	214.9	29.4	127.2	114.9	486.4	5.96E−05
36	dct 29,800	214.2	29.4	126.6	114.3	484.4	5.77E−05
37	filterless 17,200	89.7	0.0	101.0	91.5	282.2	1.34E−04
38	filterless 17,400	90.1	0.0	100.9	91.5	282.5	2.33E−03
39	filterless 17,800	90.6	0.0	100.8	91.5	282.9	2.50E−03
40	filterless 21,200	124.6	0.0	131.8	118.6	375.0	4.65E−05
41	filterless 21,400	124.1	0.0	131.8	118.4	374.3	1.69E−03
42	filterless 21,800	124.4	0.0	131.3	118.5	374.2	6.66E−05
43	filterless 25,200	167.2	0.0	170.0	152.7	489.8	8.33E−05
44	filterless 25,400	166.3	0.0	170.3	152.8	489.4	1.04E−03
45	filterless 25,800	169.6	0.0	174.4	156.1	500.1	1.16E−03
46	filterless 29,200	224.0	0.0	222.2	197.1	643.3	2.41E−04
47	filterless 29,400	192.8	0.0	207.1	182.9	582.8	3.59E−04
48	filterless 29,800	218.7	0.0	213.5	190.7	622.9	1.13E−04

Table 5 Ranking of different simulation cases based on running times of different sections (including scanning, filtering, clustering and simulation procedures) for realizations obtained through application of Spectral clustering technique

Scanning time	Filtering time	Clustering time	Simulation time	Total time	Simulation method
39	47	16	16	1	pca 17,200
25	46	18	30	2	pca 17,400
3	41	17	17	3	pca 17,800
28	9	21	38	4	pca 21,200
5	7	19	19	5	pca 21,400
6	8	20	37	6	pca 21,800
9	12	35	35	7	pca 25,200
43	28	36	36	8	pca 25,400
21	13	34	34	9	pca 25,800
35	35	10	10	10	pca 29,200
34	36	11	11	11	pca 29,400
48	23	47	47	12	pca 29,800
14	40	26	26	13	knn 17,200
38	42	15	15	14	knn 17,400
26	45	25	25	15	knn 17,800
42	4	39	20	16	knn 21,200
40	6	38	21	17	knn 21,400
29	5	37	39	18	knn 21,800
20	30	23	22	19	knn 25,200
32	14	4	4	20	knn 25,400
31	10	6	6	21	knn 25,800
19	16	40	40	22	knn 29,200
7	17	42	41	23	knn 29,400
45	18	41	42	24	knn 29,800
37	39	14	27	25	dct 17,200
27	48	27	14	26	dct 17,400
1	38	13	13	27	dct 17,800
30	3	2	2	28	dct 21,200
41	2	3	3	29	dct 21,400
4	1	1	1	30	dct 21,800
8	11	5	5	31	dct 25,200
44	15	22	23	32	dct 25,400
33	29	24	24	33	dct 25,800
10	33	9	8	34	dct 29,200
23	31	7	7	35	dct 29,400
47	32	8	9	36	dct 29,800
15	43	30	18	37	filterless 17,200
2	44	29	29	38	filterless 17,400
13	37	28	28	39	filterless 17,800
18	27	32	32	40	filterless 21,200
17	26	31	33	41	filterless 21,400
16	25	33	31	42	filterless 21,800
36	21	44	44	43	filterless 25,200
24	20	43	43	44	filterless 25,400
22	19	45	45	45	filterless 25,800
46	22	46	46	46	filterless 29,200
11	34	12	12	47	filterless 29,400
12	24	48	48	48	filterless 29,800

Table 6 Ranking of different simulation cases based on flow simulation recoveries for realizations obtained through application of Spectral clustering technique

Simulation method	MSE
pca 17,200	4.60E−05
filterless 21,200	4.65E−05
pca 29,200	4.76E−05
knn 17,800	5.68E−05
dct 29,800	5.77E−05
knn 17,400	5.77E−05
dct 29,400	5.96E−05
filterless 21,800	6.66E−05
dct 21,200	7.36E−05
dct 25,800	8.14E−05
filterless 25,200	8.33E−05
dct 17,400	8.69E−05
pca 25,800	1.09E−04
pca 17,400	1.12E−04
filterless 29,800	1.13E−04
pca 21,400	1.18E−04
dct 25,200	1.19E−04
knn 21,200	1.28E−04
dct 17,200	1.32E−04
filterless 17,200	1.34E−04
dct 21,800	1.38E−04
dct 21,400	1.49E−04
pca 21,800	1.68E−04
pca 21,200	2.11E−04
pca 25,400	2.24E−04
knn 21,800	2.26E−04
knn 25,200	2.29E−04
knn 17,200	2.31E−04
filterless 29,200	2.41E−04
knn 29,800	2.41E−04
pca 25,200	3.18E−04
knn 25,400	3.57E−04
filterless 29,400	3.59E−04
dct 25,400	4.40E−04
knn 25,800	4.74E−04
filterless 25,400	1.04E−03
filterless 25,800	1.16E−03
pca 29,800	1.37E−03
pca 29,400	1.53E−03
filterless 21,400	1.69E−03
dct 29,200	2.17E−03
knn 21,400	2.28E−03
filterless 17,400	2.33E−03
dct 17,800	2.34E−03
filterless 17,800	2.50E−03
knn 29,400	2.55E−03
pca 17,800	3.42E−03
knn 29,200	3.75E−03

not sensibly affected by cluster numbers while using K-means clustering method.

Amongst the filtering approaches used in this study, PCA takes the smallest time to do a reduction in problem dimensionality than KNN and DCT techniques. KNN has slightly brought more running time compared to DCT practice. Considering total simulation time, however, Figs. 7 and 9 verify that DCT, KNN, PCA and FILTERLESS practices do the filtering in the lowest time, respectively. This trend is more regular in the case of using Spectral technique rather than K-means method.

Obviously, time of clustering depends, to some extent, on the outputs of filtering process. This could be easily incepted in Tables 1 and 4. Usage of KNN and DCT techniques in filtering process leads to smaller (and nearly the same) running times of clustering. Utilizing PCA method or skipping filtering (FILTERLESS case) increases running time subsequently. The same conclusion could be induced about running time of simulation process itself. Total time of FILTERSIM, from scanning process through simulation stage, obeys the same trend with the filter type.

The graphs shown in Figs. 8 and 10 are displayed on a single graph in Fig. 11 for a better comparison of recovery MSE values for both K-means and the Spectral clustering techniques. As it is clear from these figures, the realizations obtained through application of the Spectral clustering method have shown a better accuracy than those acquired by applying K-means clustering. This is related to the non-linear behavior of the pattern database and the fact that Spectral methods are able to partition the pattern space through the non-linear boundary, whereas K-means is only capable of regarding linear borders between the clusters. The two graphs exhibit nearly the same trend, although some deviations exist from what we expected.

Another important conclusion is that skipping the filtering section during the simulation procedure traces a better accuracy within the simulated realizations obtained through application of either Spectral or K-means clustering method in comparison with the cases where dimensionality reduction has been implemented prior to clustering section. Leaving extracted patterns without passing them to filtering section, however, brings about more running times in clustering stage and hence in total simulation procedure, as observed in Tables 1 and 4. Moreover, when patterns are filtered through PCA or DCT method, generated realizations have quite a reliable agreement with original training image from the reservoir characteristic point of view, with application of both clustering methods. The simple KNN method, however, does fail in reproducing the connected patterns carefully to capture the flow behavior of the system in a reasonable way when patterns are clustered through K-means method. However, when Spectral method is used in clustering scheme, KNN approach does the filtering more satisfactorily.

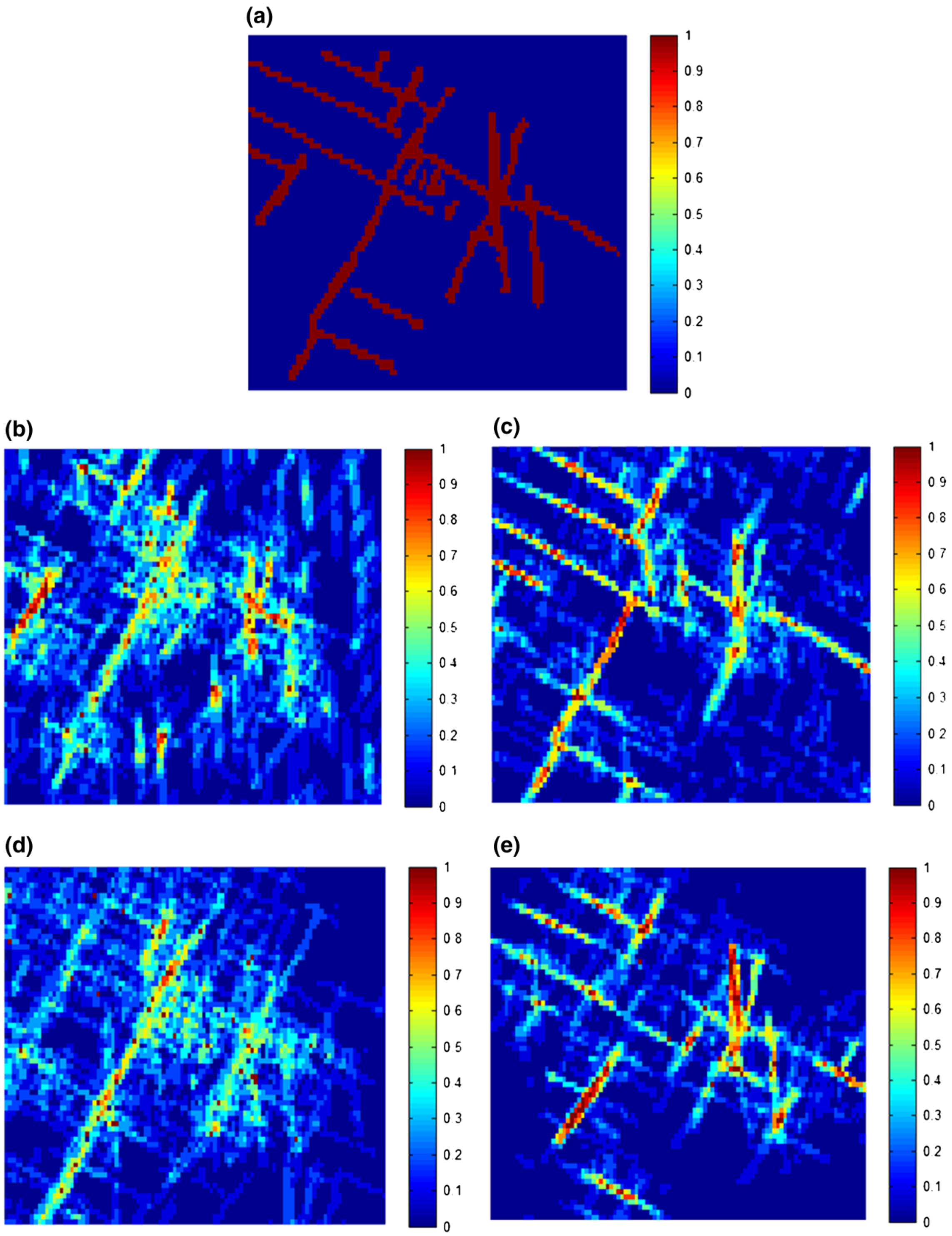

Fig. 6 Some results of FILTERSIM sensitivity analysis. **a** Training image of size 69 × 104 grids used in sensitivity analysis; **b** The least accurate E_type of FILTERSIM cases obtained through application of K-means clustering algorithm; **c** The most accurate E_type of FILTERSIM cases obtained through application of K-means clustering algorithm; **d** The least accurate E_type of FILTERSIM cases obtained through application of Spectral clustering algorithm; and **e** The most accurate E_type of FILTERSIM cases obtained through application of Spectral clustering algorithm. *NB* The colorbar next to each figure represents degree of fracture evidence in it. The more-*red* the nodes, the more the probability that the specified node is located in fracture; on the other hand, more-*blue nodes* denote less-fractured areas of the system

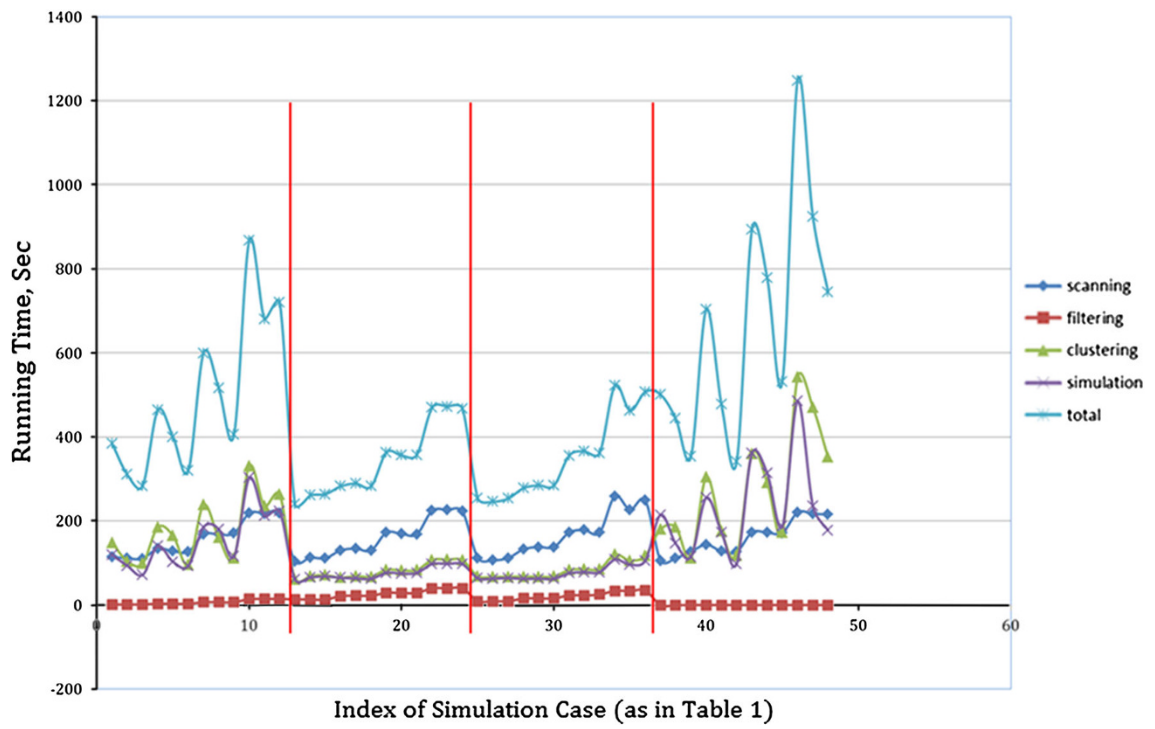

Fig. 7 Running times of different simulation stages (including scanning, filtering, clustering and simulation procedures) for realizations obtained through application of K-means clustering technique. The *vertical axis* is in seconds, showing running times of each simulation stage; while the *horizontal axis* denotes number of each FILTERSIM case as labeled in Table 1. *Vertical solid lines* in *orange* indicate the boundary of each filtering method; from *left* to *right*, the regions are exclusive to PCA, KNN, DCT and FILTERLESS techniques

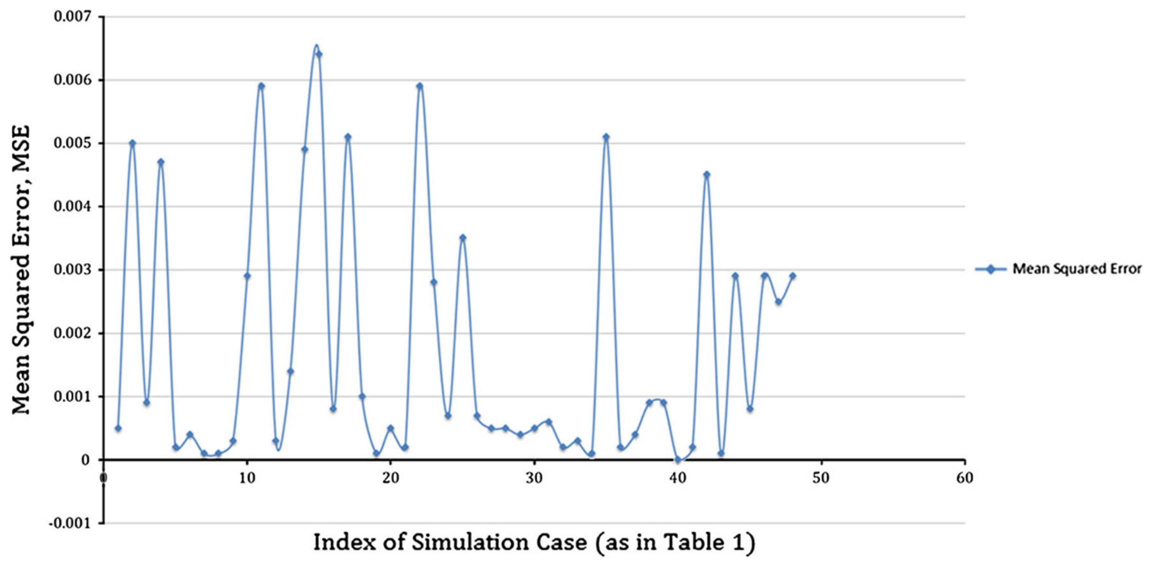

Fig. 8 Mean squared errors of production recoveries of different realizations obtained through application of K-means clustering technique. The *vertical axis* shows MSE value for each simulation stage; while the *horizontal axis* denotes number of each FILTERSIM case as labeled in Table 1

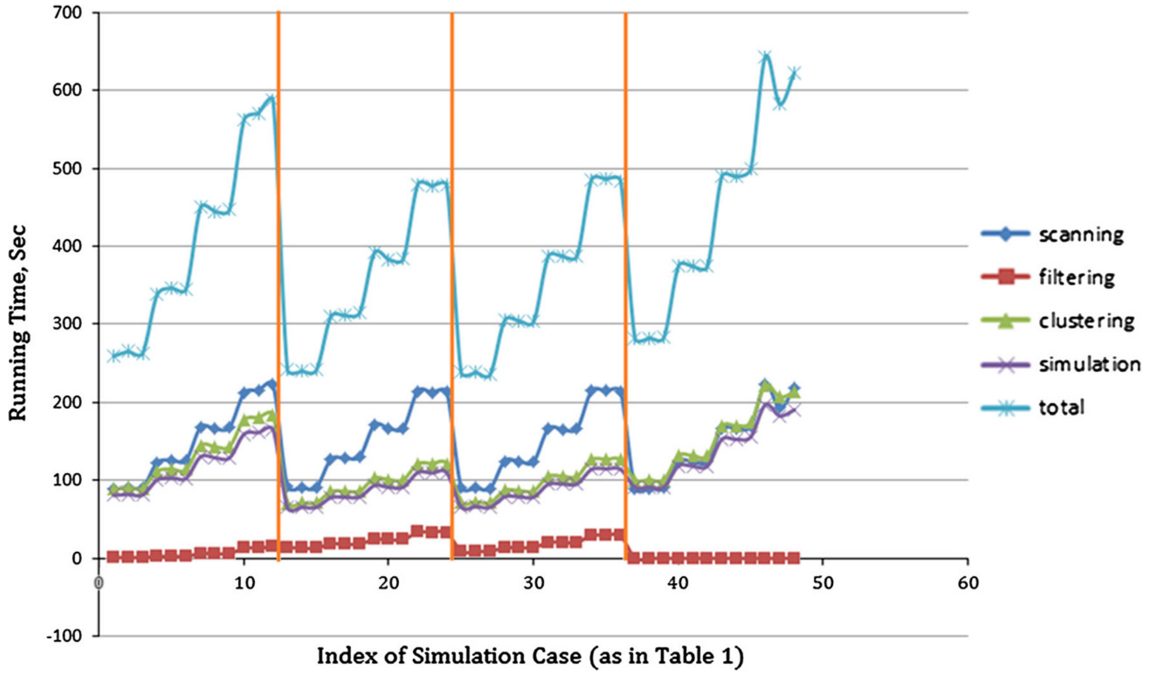

Fig. 9 Running times of different simulation stages (including scanning, filtering, clustering and simulation procedures) for realizations obtained through application of Spectral clustering technique. The vertical axis is in seconds, showing running times of each simulation stage; while the horizontal axis denotes number of each FILTERSIM case as labeled in Table 4. *Vertical solid lines* in *orange indicate* the boundary of each filtering method; from *left* to *right*, the regions are exclusive to PCA, KNN, DCT and FILTERLESS techniques

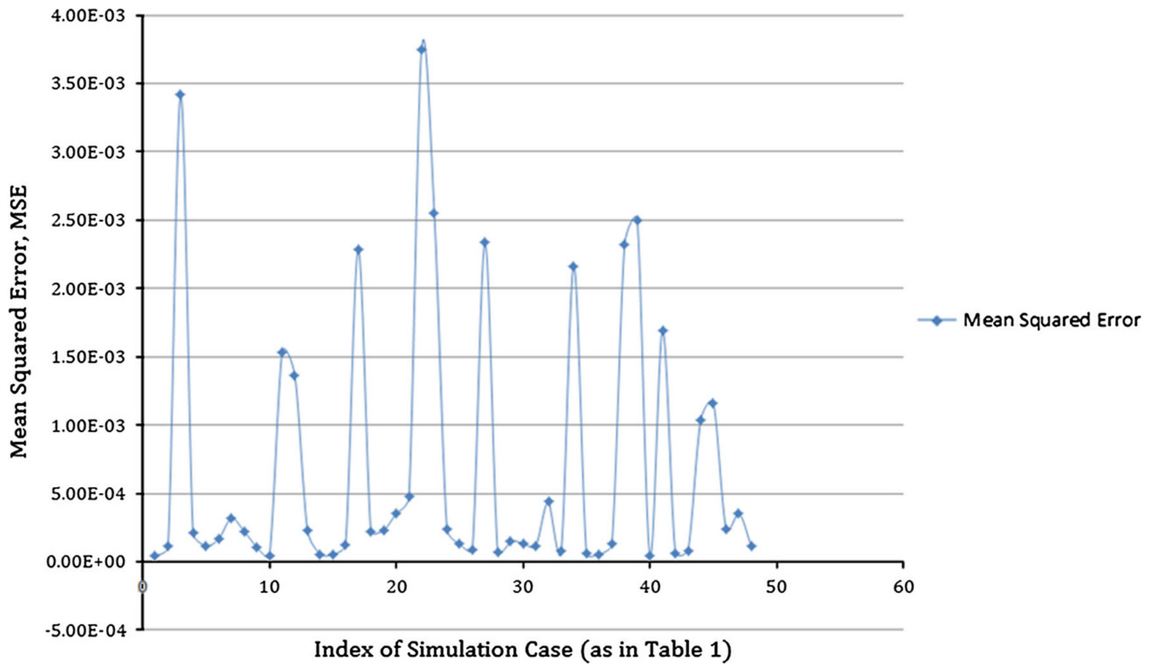

Fig. 10 Mean squared errors of production recoveries of different realizations obtained through application of Spectral clustering technique. The *vertical axis* shows MSE value for each simulation stage; while the *horizontal axis* denotes number of each FILTERSIM case as labeled in Table 4

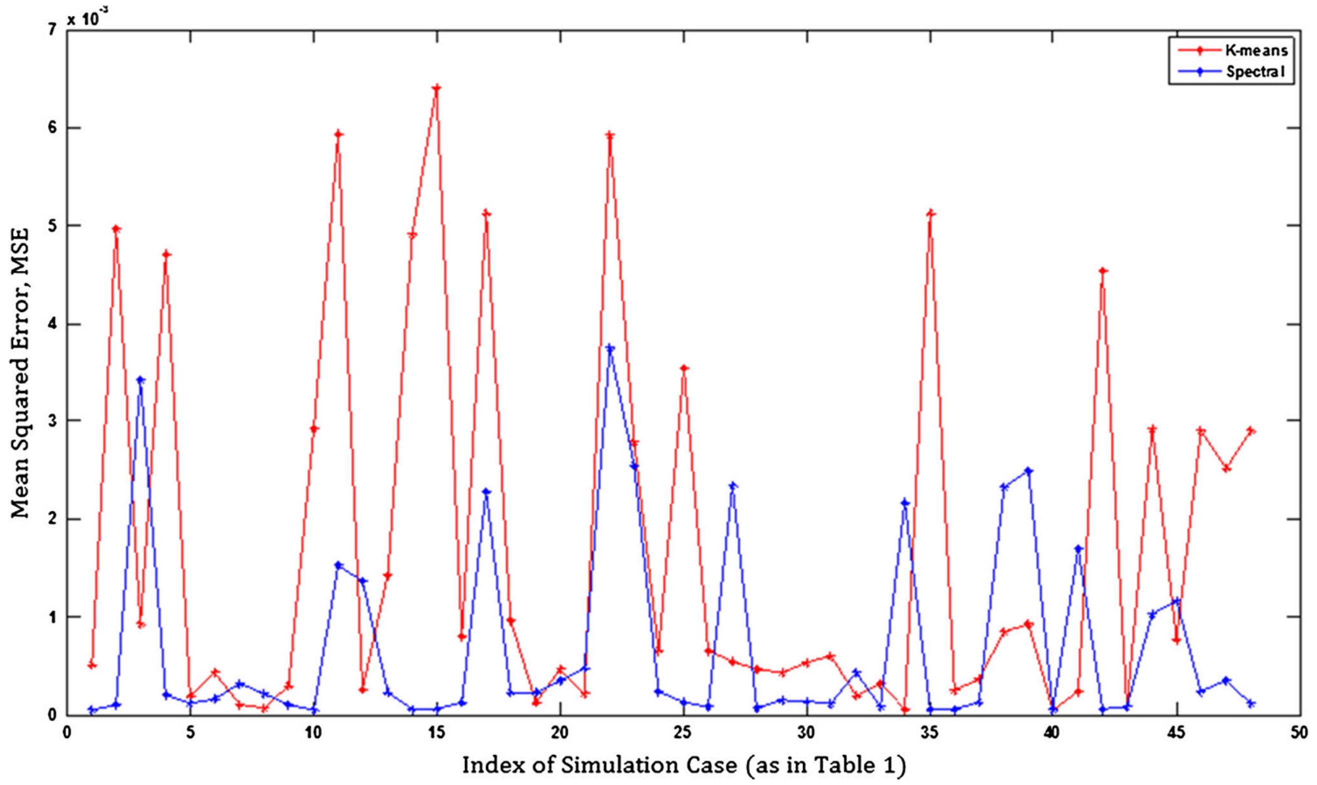

Fig. 11 Mean squared errors of production recoveries of different realizations obtained through application of **a** K-means clustering, and **b** Spectral clustering technique. As shown on legends, *red lines* denote K-means results and *blue lines* indicate Spectral results. The *vertical axis* shows MSE value for each simulation stage; while the *horizontal axis* denotes number of each FILTERSIM case as labeled in Table 1

Conclusions

1. MPS simulation algorithms are effectively able to model complex, heterogeneous and curvilinear geological structures of discrete fracture networks. This is achieved through capturing the multiple-point patterns of training images and considering the spatial relationship between more than two points in the model simultaneously.
2. Amongst the four steps of FILTERSIM algorithm, scanning process, clustering section and simulation procedure take the most part of total simulation time, respectively. Filtering process, however, takes a small time compared to other sections and does not have a significant effect on total running time.
3. Growing search template dimensions will lead to a corresponding increase in all running times during scanning, filtering, clustering and simulation procedures.
4. Changing the number of partitions used in Spectral clustering algorithm does not affect any parts of total running time considerably, even including the clustering section. Using a less number of clusters, however, raises the computational speed of K-means clustering technique either when the training patterns stored in the database have been filtered using PCA method or they have not been filtered anyway (FILTERLESS case).
5. PCA takes the smallest time to do a reduction in problem dimensionality than KNN and DCT techniques which bring nearly the same filtering times. Regarding FILTERSIM computational time, however, DCT, KNN, PCA and FILTERLESS practices do the filtering in the lowest time, respectively.
6. Application of KNN and DCT as filtering techniques leads to a smaller clustering time. Utilizing PCA method or skipping filtering (FILTERLESS case) increases running time subsequently. The same conclusion could be induced about running time of simulation process itself as well as total time of FILTERSIM.
7. Realizations obtained through application of Spectral clustering method show a better accuracy than those generated by K-means clustering.
8. Filtering patterns prior to clustering them may decrease the accuracy of final realizations, though they a have a great effect on the reduction of simulation times. Nevertheless, PCA and DCT

methods may lead to quite acceptable results with both clustering approaches, while at the same time total simulation time decreases substantially.

References

Ahmadi R (2011) Geometrical fracture modeling within multiple-point statistics framework. MS Dissertation, Sharif University of Technology, Department of Chemical and Petroleum Engineering, Iran

Anderberg MR (1973) Cluster analysis for applications. Academic Press, New York

Baker RO, Schechter DS, McDonald P, Knight WH, Leonard P, Rounding C (2001) Development of a fracture model for spraberry field, Texas USA. SPE 71635. In: Annual technical conference and exhibition, Louisiana

Ball G, Hall D (1965) ISODATA, a novel method of data analysis and pattern classification. Technical report NTIS AD 699616. Stanford Research Institute, Stanford, CA

Bishop CM (2006) Pattern recognition and machine learning. Springer, Berlin

Boisvert JB, Pyrcz MJ, Deutsch CV (2008) Multiple-point statistics for training image selection. Nat Resour Res 16(4):313–321

Bourbiaux BJ, Cacas MC, Sarda S, Sabathier JC (1997) A fast and efficient methodology to convert fractured reservoir images into a dual-porosity model. SPE 38907. In: Annual technical conference and exhibition, Texas, USA

Cacas MC, Ledoux E, de Marsily G, Tillie B (1990) Modeling fracture flow with a stochastic discrete fracture network: calibration and validation. Water Resour Res 26:479–489

Caers J, Arpat GB (2005) A multiple-scale, pattern-based approach to sequential simulation. PhD dissertation, Stanford University, Department of Petroleum Engineering, Stanford, USA

Caers J, Arpat GB (2007) Conditional simulation with patterns. Math Geol 39(2):177–203

Charlaix E, Guyon E, Roux S (1987) Permeability of an array of fractures with randomly varying apertures. Transp Porous Med. 2:31–43

Chugunova T, Hu L (2008) Multiple-point simulations constrained by continuous auxiliary data. Math Geosci 40(2):133–146

Chung FRK (1997) Spectral graph theory. In: CBMS regional conference series in mathematics 92, American Mathematical Society, Providence, RI

Cover TM, Hart PE (1967) Nearest neighbor pattern classification. IEEE Trans Inf Theory 13(1):21–27

Cristianini N, Taylor JS, Kandola JS (2001) Spectral kernel methods for clustering. In: NIPS, pp 649–655

Dershowitz W, Shuttle D, Parney R (2002) Improved oil sweep through discrete fracture network modelling of gel injections in the South Oregon Basin Field, Wyoming. In: SPE 75162, energy thirteenth improved oil recovery symposium, Oklahoma, USA

Deutsch CV (1992) Annealing techniques applied to reservoir modeling and the integration of geological and engineering (well test) data. PhD Dissertation, Stanford University, Stanford

DeVries L (2009) Application of multiple point geostatistics to non-stationary images. Math Geosci 41(1):29–42

Duda R, Hart P, Stork D (2001) Pattern Classification, 2nd edn. Wiley, New York

Filippone M, Camastra F, Masulli F, Rovetta S (2008) A survey of kernel and spectral methods for clustering. Pattern Recognit. 41:176–190

Guardiano F, Srivastava R (1993) Multivariate geostatistics: beyond bivariate moments. Geostatistics-troia. Kluwer Academic Publications, Dordrecht, pp 133–144

Han J, Kamber M (2000) Data mining: concepts and techniques. Morgan Kaufmann, San Francisco

Hestir K, Long JCS (1990) Analytical expressions for the permeability of random two-dimensional Poisson fracture networks based on regular lattice percolation and equivalent media theories. J Geophys Res 95(B13):565–581

Hestir K, Long JC, Chilks JP, Billaux (1987) Some techniques for stochastic modeling of three-dimensional fracture networks. In: Proceedings of the conference on geostatistical, sensitivity, and uncertainty methods for ground -water flow and radionuclide transport modeling, San Fransisco, CA, pp 495–519

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

Hu L, Chugunova T (2008) Multiple-point geostatistics for modeling subsurface heterogeneity: a comprehensive review. Water Resour Res 44(W11413). doi:10.1029/2008WR006993

Jackson JE (1991) A user's guide to principal components. Wiley, NewYork, p 592

Jain AK (1989) Fundamentals of digital image processing. Prentice-Hall, Englewood Cliffs

Jain AK (2010) Data clustering: 50 years beyond K-means. Pattern Recognit Lett 31:651–666

Jain AK, Dubes RC (1988) Algorithms for clustering data. Prentice Hall, Upper Saddle River

Jolliffe IT (2002) Principal component analysis, 2nd edn. Springer, Berlin

Journel A, Zhang T (Dec.2006) The necessity of a multiple-point prior model. Math Geol 38(5)

Lee SH, Jensen CL, Lough MF (1999) An efficient finite difference model for flow in a reservoir with multiple length-scale fractures. In: SPE 56752, annual technical conference and exhibition, Texas

Lioyd S (1982) Least squares quantization in PCM. IEEE Trans Inf Theory 28:129–137 (Originally as an unpublished Bell laboratories Technical Note (1957))

Long JCS, Witherspoon PA (1985) The relationship of the degree of interconnections to permeability in fracture networks. J Geophys Res 90(B4):3087–3098

Long JCS, Remer JS, Wilson CR, Witherspoon PA (1982) Porous media equivalents for networks of discontinuous fractures. Water Resour Res 18(3):645–658

Long JC, Billaux D, Hestir K, Chilks JP (1987) Some geostatistical tools for incorporating spatial structure in fracture network modeling. Proc Congr Int Soc Rock Mech 6(1):171–176

MacQueen JB (1967) Some methods for classification and analysis of multivariate observations, vol 1. In: Proceedings of 5th Berkeley symposium on mathematical statistics and probability, Berkeley, University of California Press, pp 281–297

Mariethoz G (2010) A general parallelization strategy for random path based geostatistical simulation methods. Comput Geosci 37(7):953–958

Moran PAP (1950) Notes on continuous stochastic phenomena. Biometrika 37(1/2):17–23. (Biometrika Trust)

Mukhopadhyay S, Sahimi M (1996) Scaling properties of a percolation model with long-range correlations. Phys Rev E 54(4):3870–3880

Oda M, Hatsuyama Y, Ohnishi Y (1987) Numerical experiments on permeability tensor and its application to jointed granite at stripa mine, Sweden. J. Geophys Res 92:8037–8048

Pennebaker WB, Mitchell JL (1993) JPEG Still image data compression standard, Chap 4. Van Nostrand Reinhold, New York, NY

Perner P (2007) Machine learning and data mining in pattern recognition. In: Proceedings of the 5th international conference, MLDM 2007, Leipzig, Germany

Pollard DD (1976) On the form and stability of open hydraulic fractures in the Earth's crust. Geophys Res Lett 3(9):513–516

Pyrcz MJ, Boisvert JB, Deutsch CV (2008) A library of training images for fluvial and deepwater reservoirs and associated code. Comput Geosci 34(5):542–560

Shawe TJ, Cristianini N (2004) Kernel methods for pattern analysis. Cambridge University Press, Cambridge

Smith L, Schwartz FW (1984) An analysis of the influence of fracture geometry on mass transport in fractured media. Water Resour Res 20(9):1241–1252

Steinhaus H (1956) Sur la division des corp materiels en parties. Bull Acad Polon Sci IV (C1.III):801–804

Straubhaar J (2008) Optimization issues in 3D multipoint statistics simulation. In: Geostats, Santiago, Chile

Strebelle S (2000) Sequential simulation drawing structures from training images. PhD Dissertation, Stanford University, Stanford, CA

Strebelle S (2002) Conditional simulation of complex geological structures using multiple-point statistics. Math Geol 34(1):1–21

Tan PN, Steinbach M, Kumar V (2005) Introduction to data mining. Addison-Wesley Longman Publishing Co., Inc., Boston

Teimoori A, Tran NH, Chen Z, Rahman SS (2004) Simulation of fluid flow in naturally fractured reservoirs with the use of effective permeability tensor. In: Annual proceeding of geothermal resources council (GRC), Australia

Toussaint GT (2005) Geometric proximity graphs for improving nearest neighbor methods in instance-based learning and data mining. Int J Comput Geom Appl 15(2):101–150

Tran NH (July 2004) Characterization and modeling of naturally fractured reservoirs. PhD Dissertation, The University of New South Wales, School of Petroleum Engineering, Sydney, Australia

Tran NH, Rahman MK, Rahman SS (2002) A nested neuro-fractal-stochastic technique for modeling naturally fractured reservoirs. In: SPE 77877, Asia Pacific oil and gas conference and exhibition, Melbourne, Australia

Wu J (2007) 4D seismic and multiple-point pattern data integration using geostatistics. PhD Dissertation, Stanford University, Department of Energy Resources Engineering

Xu R, Wunsch D (2005) Survey of clustering algorithms. IEEE Trans Neural Netw. 16(3):645–678