Math Geosci (2011) 43:611–624
DOI 10.1007/s11004-011-9350-9

Facies Modeling Using a Markov Mesh Model Specification

Marita Stien · Odd Kolbjørnsen

Received: 28 April 2010 / Accepted: 13 February 2011 / Published online: 27 July 2011

Abstract The spatial continuity of facies is one of the key factors controlling flow in reservoir models. Traditional pixel-based methods such as truncated Gaussian random fields and indicator simulation are based on only two-point statistics, which is insufficient to capture complex facies structures. Current methods for multi-point statistics either lack a consistent statistical model specification or are too computer intensive to be applicable. We propose a Markov mesh model based on generalized linear models for geological facies modeling. The approach defines a consistent statistical model that is facilitated by efficient estimation of model parameters and generation of realizations. Our presentation includes a formulation of the general framework, model specifications in two and three dimensions, and details on how the parameters can be estimated from a training image. We illustrate the method using multiple training images, including binary and trinary images and simulations in two and three dimensions. We also do a thorough comparison to the *snesim* approach. We find that the current model formulation is applicable for multiple training images and compares favorably to the *snesim* approach in our test examples. The method is highly memory efficient.

Keywords Sequential simulation · Unilateral scan · Generalized linear models

1 Introduction

Reservoir models are commonly described by a two-step approach by first defining the geometry of the facies and then populating the model with petrophysical properties (Damsleth et al. 1992). Simulation studies show that the facies model is often one of the main sources of variability in flow (Skorstad et al. 2005). The

M. Stien (✉) · O. Kolbjørnsen
Norwegian Computing Center, Gaustadalleèn 23, 0314 Blindern, Norway
e-mail: marita.stien@nr.no

spatial distribution of facies is therefore a crucial part of any reservoir model. The use of multi-point statistics for geological facies modeling was proposed nearly two decades ago (Guardiano and Srivastava 1993). Since then, several methods have been developed and tested, along two main paths of development: the statistical model-driven-approach and the algorithmic approach. Markov random fields (Tjelmeland and Besag 1998) have been the preferred statistical model. The problem with these models is that they are highly time-consuming, both in terms of model estimation and simulation. The development of algorithmically driven methods aims to formulate a simulation procedure that reproduces patterns for a limited template. This approach experienced a break-through with the introduction of search trees (Strebelle 2000). All these methods have been criticized for their lack of consistency, since the statistical model depends on the simulation path. A more serious concern with the algorithmic approach is, however, a strong dependence on pattern frequencies in the training image. The problem is not with the patterns seen in the training image but, rather, with how the method treats patterns that are not present in the training image. Current practice involves reducing the size of the pattern, but there is obviously room for more advanced approaches. In this respect, methods using statistical models have an advantage over algorithmic methods, since they can interpolate between observed patterns to compute the probability of patterns that are not present in the training image.

We propose to model facies dependencies through a Markov mesh model (Abend et al. 1965). Markov mesh models are a sub-class of Markov random fields that is defined through a unilateral path (Daly 2005). The probability model is defined using framework of generalized linear models, hereafter GLMs (McCullagh and Nelder 1989). This type of model is also discussed in Cressie and Davidson (1998), but we are explicit in the formulation of the model and extend it to three dimensions. Our formulation enables a fast estimation of the model's parameters through iterated weighted least squares, and fast simulation by the sequential definition. This parameterization is suited to model phenomena with a high degree of spatial continuity, as in facies structures. It captures the consistency of the modeling approach and the speed of the algorithmic approach during simulation and is also memory efficient.

This paper is organized as follows: First, we describe the Markov mesh model and how we parameterize it, starting with the two-dimensional parametrization that we extend to three dimensions. Then we explain how model parameters are adjusted to reach a target volume fraction. We show examples in two and three dimensions with both binary and trinary training images, and compare results to the *snesim* approach. The estimation of the parameters are outlined in the appendix. This work focuses on how the statistical model can be formulated and how to simulate unconditional realizations. The challenge of conditional realizations is discussed in Kjønsberg and Kolbjørnsen (2008).

2 Markov Mesh Models

Markov mesh models are defined by a unilateral path and a conditional probability for each cell value given the cell values in a sequential neighborhood. The sequential

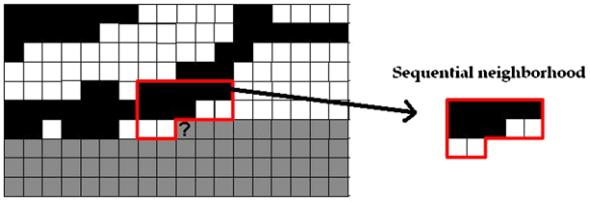

Fig. 1 This illustration displays a snapshot of an unfinished simulation, where the *gray cells* have not yet been simulated. The sequential neighborhood is represented by the cells within the *red* border

neighborhood is a subset of previously simulated cells, as illustrated in Fig. 1 on a two-dimensional grid. Consider a finite, regular grid in two or more dimensions, and let the one-dimensional index i label the cells of the grid. The set of all cells is $\{1, 2, \ldots, N\}$, where the cell value x_i can take K different facies values, $x_i \in (1, K)$. Assuming that the conditional probability for facies at cell i depends only on a subset Γ_i of all cells $i < j$, we can write

$$\pi(x_i|x_{j<i}) = \pi(x_i|x_{\Gamma_i}). \tag{1}$$

Markov mesh models are fully specified through the conditional probabilities in (1), such that the joint probability is

$$\pi(x_1, x_2, \ldots, x_N) = \prod_{i=1}^{N} \pi(x_i|x_{\Gamma_i}). \tag{2}$$

Simulation from the model is carried out by following the unilateral path, $i = 1, 2, \ldots, N$, throughout the grid. For each cell the facies value is drawn according to the conditional probability $\pi(x_i|x_{\Gamma_i})$. Each cell is visited once, and the resulting grid configuration follows the joint probability distribution in (2).

2.1 Unilateral Path

Using a unilateral path in the model definition ensures a consistent, well defined model for which the estimation procedure is straight-forward. The unilateral path may introduce skewness in the model when a final extent of the sequential neighborhood is used. The skewness is, however, partially countered in the estimation, and the effect decreases as the size of the sequential neighborhood increases. An additional challenge introduced by the unilateral path, is that of conditional simulation. Data located ahead in the path is not accounted for in the sequential simulation. This issue is discussed in Kjønsberg and Kolbjørnsen (2008) which also propose a method to solve it. They use indicator Kriging to approximate the likelihood of points ahead in the path, and update the probabilities computed by the Markov mesh model with this likelihood.

3 Model Specification

The statistical model is defined by parameterizing the conditional probabilities in (1). Our model is based on GLMs (McCullagh and Nelder 1989). The formulation is

chosen such that the parameters are efficient to estimate and simple to interpret. We do not claim that our model is unique in this respect; there exist many alternative model formulations with similar characteristics.

The idea in GLM is that the distribution of a response variable depends on a linear combination of explanatory variables through a non-linear link function. In our application, we let the facies x_i be the response variable and the explanatory variables be functions of the sequential neighborhood. The facies x_i is one of K categories and is encoded with binary variables x_i^k such that $x_i^k = 1$ if $x_i = k$, and zero otherwise. Further we let $\mathbf{z}$ be a $(P+1) \times 1$ vector of explanatory variables with elements that are functions of cells from the sequential neighborhood. We propose particular functions below, but for now we write $z_{ij} = f_j(x_{\Gamma_i})$ for some $j \in (1, P)$. The conditional probability in (1) is then

$$\pi\left(x_i|\mathbf{z}_i, \boldsymbol{\theta}^1, \ldots, \boldsymbol{\theta}^K\right) = \frac{\prod_{k=1}^K \exp\{\mathbf{z}_i^T \boldsymbol{\theta}^k x_i^k\}}{\sum_{k=1}^K \exp\{\mathbf{z}^T \boldsymbol{\theta}^k\}}.$$

The joint probability in (2) is, furthermore,

$$\pi(x_1, \ldots, x_N) = \prod_{i=1}^N \frac{\prod_{k=1}^K \exp\{\mathbf{z}_i^T \boldsymbol{\theta}^k x_i^k\}}{\sum_{k=1}^K \exp\{\mathbf{z}_i^T \boldsymbol{\theta}^k\}}. \tag{3}$$

Interpreted as a likelihood for the model parameters, this expression is a GLM. The maximum likelihood estimation of the parameters in the Markov mesh formulation can therefore be solved with the iterative weighted least squares scheme. The Appendix provides details of our implementation.

Although (3) is identical to the likelihood of a GLM, the assumptions that lead to them are very different. In GLMs the assumption is independence, whereas a Markov mesh model uses a sequential formulation. Thus, even though the maximum likelihood estimate is identical, other properties of the estimators in the GLM do not generally hold. In our application to facies modeling, this has an unfortunate effect on the volume fraction. We account for this by post-processing the parameter estimates explained in Sect. 4. The estimated model depends on the choice of the unilateral path. If we rotate the training image, we obtain other parameter values. We achieve the best results when we simulate in the direction of the longest correlation range.

3.1 Specification of the Neighborhood Functions

We specify the neighborhood functions $f_j(x_{\Gamma_i})$ for $j = 1, \ldots, P$, for a two-dimensional model initially and subsequently extend this model to three dimensions. Facies continuity and transitions to other facies are the most important features of geological structures, and our model description therefore focuses on these features. A multi-point interaction of order l is that between the reference cell and a function of the values of $l-1$ cells in its sequential neighborhood. A two-point interaction thereby refers to the interaction between the reference cell and the value of one cell in its sequential neighborhood.

3.1.1 Two-Dimensional Specification

The challenge with multi-point statistics is that there are too many possibilities. It is not possible to extract all properties, since this would create a problem of missing patterns, as in the traditional *snesim* approach. We extract a subset of properties we believe are important in order to reproduce geological structures. These properties are not necessarily suited for all possible training images. It is possible to achieve other characteristics by adding or removing neighborhood functions.

The two-point interactions are the simplest, and our model includes all two-point interactions in a subset of the sequential neighborhood. Figure 2 shows how this can be described by the two lengths l_x and l_y to give interactions with $2 \cdot l_x \cdot l_y + l_x + l_y$ cells. For each cell j among these, we include for every facies k an indicator function $f^k(x_j)$ that equals one if cell j has facies k, and zero otherwise. This yields one parameter for each facies value of each cell in the sequential neighborhood, resulting in $K(2 \cdot l_x \cdot l_y + l_x + l_y)$ parameters. The impact of the four nearest cells γ_i^4, illustrated in Fig. 3, is very important; therefore all combinations of these are considered, resulting in K^5 parameters. For multi-point interactions at longer range, we focus on the continuity and transitions of facies and therefore include multi-point patterns where all cells have identical facies. The sets of cells in these patterns are chosen to capture the shape and extension of the facies object. We limit our selections to a set of directions (Fig. 4). In each of these directions, we first include the interactions with the two nearest cells, which are three-point interactions. Then we increase one cell at the time, until we reach a limit L. Figure 5 illustrates these interaction terms. Let $x_{\gamma_i}^{l-1}$ be a set of $l-1$ neighbors in an l-point interaction term. Then the indicator functions are expressed as

$$f^k\left(x_{\gamma_i}^{l-1}\right) = \begin{cases} 1 & \text{if all } x_j \in x_{\gamma_i}^{l-1} = k, \\ 0 & \text{otherwise.} \end{cases}$$

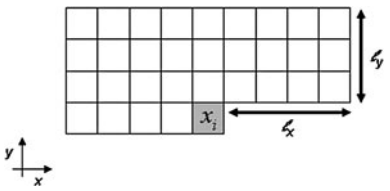

Fig. 2 Illustration of the two-point interaction neighborhood: l_x and l_y yield the span of the neighborhood

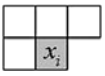

Fig. 3 The model includes all possible combinations of these five cells

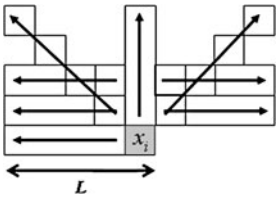

Fig. 4 Illustration of the strips of cells where higher-point interactions are considered. The *arrows* indicate the directions and in which order the number of interaction terms increases

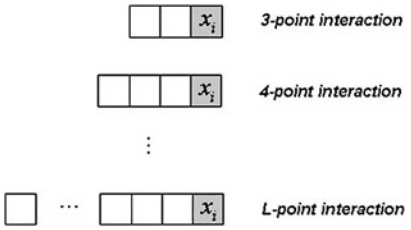

Fig. 5 Example of higher-point interactions that are included in one set of directions

For each facies and direction, one such indicator function is included in our model up to the highest interaction term of length L, resulting in $8 \cdot (L-2)K$ terms.

Our explicit parametrization is easy to interpret: Two-point statistics measure direct dependencies between cells; nearest cells indicate preferences for particular structures at the minimum scale; and indicators of facies continuity promote continuity if the parameter corresponding to the same facies is large, and promotes transition if the parameter corresponding to an other facies is large. Some of the functions described are redundant; for instance, it is not necessary to include two-point interactions with the nearest four cells, since these are covered when all configurations are considered. Another situation that may occur is that a multi-point pattern described in one of the explanatory variables is not present in the training image; it may happen that one particular facies does not have four connected cells, or all facies need not be continuous for all lengths included. These issues are resolved in the parameter reduction prior to estimation, which is presented in Appendix.

3.1.2 Three-Dimensional Specification

In three dimensions, the number of cells in the dependency structure increases significantly. It is far more challenging to capture the main features of the facies structures while keeping a low number of interaction terms. For sequential simulation, all cells within the sequential neighborhood are previously simulated. We take advantage of the large redundancy of information in the sequential neighborhood and ignore some cells. By systematically selecting the cells, we still have good information about the neighborhood and are able to keep the number of parameters from exploding. This is different for *snesim*, which has unestablished patterns in its template. In order to account for all possible combination, the template needs to be fully informed.

Our selection considers those cells located in two-dimensional orthogonal slices intersecting in the reference cell i. These are cells from layers above the current layer and cells from earlier in the path in the current layer. For each of the three two-dimensional slices, we adopt the two-dimensional model above, that is, we include a similar set of interaction terms for each slice. In addition, we extend the neighborhood beyond these two-dimensional slices by considering cells extending diagonally out in all three directions. We include four such diagonal lines of cells, from which we add the same set of interaction terms as illustrated in Fig. 5. Figure 6 displays a three-dimensional illustration of the neighborhood. In geological structures, there is much similarity between consecutive layers in all directions. Therefore, we believe our choice of neighborhood to be sufficient to capture the most important correlations. An advantage of the two-dimensional cross-section of neighboring cells is

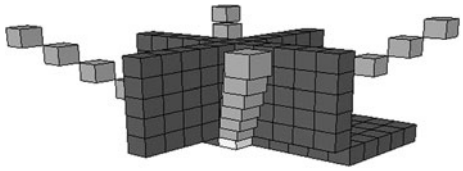

Fig. 6 An illustration of the cells included in the three-dimensional model specification, the cells from three two-dimensional slices and from four diagonal lines

that it yields a simpler interpretation of the parameters, since we can relate to two-dimensional patterns.

4 Volume Fraction Steering

Reproduction of the correct facies fraction from the training image is of great importance for our application. This is not a problem in standard GLMs, as long as the explanatory variables are generated from the same distribution as was used in the estimation. However, since our model is defined sequentially, the response variable will be an explanatory variable when we move along the unilateral path. The explanatory variables in our model are therefore not from the distribution used for estimation, unless we are able to fully reproduce all the statistics of the training image.

To reproduce the volume fraction, we adjust the estimated parameters to meet our requirement. Advantageously, the model formulation enables us to select parameters that correspond to the continuity of facies. If a facies is under-represented, we increase the parameters that represent continuity of that facies. For instance, assume that the fraction of facies k is too low, we adjust parameters within the vector $\boldsymbol{\theta}^k = (\theta_1^k, \theta_2^k, \ldots, \theta_{P+1}^k)$ by slightly increasing those parameters that represent continuity of facies k. For our parametrization, we adjust those parameters that represent strips with facies k. We propose an iterative method where the volume fraction of a realization is measured and, if the value deviates from that of the training image, the selected parameters are increased by a small value. A new realization is generated based on the adjusted parameter values, the volume fraction computed, and the parameters readjusted if necessary. This process is repeated until the volume fraction of the realization is sufficiently close to the target.

5 Examples

First we consider two two-dimensional binary training images, where one of them is used for a more thorough comparison to the *snesim* approach. This is done in terms of visual inspection, a statistical analysis of properties within realizations, and the effect of volume fraction steering. Next we consider a two-dimensional trinary and a three-dimensional binary training image, and we visually compare these to the result of the *snesim* approach. To execute *snesim*, we have used an open-source computer package from Stanford Geostatistical Modeling Software or SGeMS (Remy et al. 2009) that can be downloaded from http://sgems.sourceforge.net/.

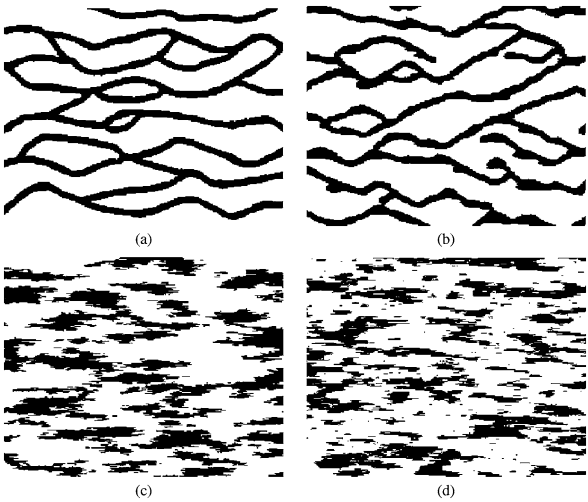

Fig. 7 Training images in (**a**) and (**c**) and corresponding realizations from our two-dimensional Markov mesh model in (**b**) and (**d**)

5.1 Two-Dimensional Binary Models

Figure 7 displays two training images and the corresponding realization from the two-dimensional Markov mesh model. We use a two-point interaction neighborhood of extension $l_x = l_y = 5$ and higher-point interactions of maximum size $L = 8$ in all directions. From a visual inspection of the results, it is clear that the model reproduces similar features as in the training images, although they are generally more rugged. The simulation direction is revealed by the slight skewness of the pattern in the realizations.

5.2 Comparison with the *Snesim* Approach

In the *snesim* algorithm we define the template size to be 60 and set the servo system factor to fully reproduce the volume fraction (Liu 2006). The Markov mesh model is defined on the original grid. It is therefore natural to compare it to the *snesim* algorithm defined on the same grid. It is, however, common to use a multigrid approach with the *snesim* algorithm, (Strebelle 2002), so we include results using three multigrids.

Figure 8 shows the results (i) the training image, (ii) a realization from our model, (iii) a *snesim* realization on one grid and (iv) a *snesim* realization using three multigrids. We clearly see the need for multigrids in the *snesim* approach. Figure 8 also suggests that the use of multigrids in the Markov mesh formulation may improve results even further. To check how the models reproduce the correct statistics, we focus on features in the realizations. Important features of geological structures that should be reproduced are the range of the dependency, the size and shape of objects, the number of objects, and the volume fraction. Soleng et al. (2006) describes a facies properties program that computes these statistics from realizations. The program detects the various facies objects and computes their volumes, surface areas, and extensions in each direction. We apply the facies properties pro-

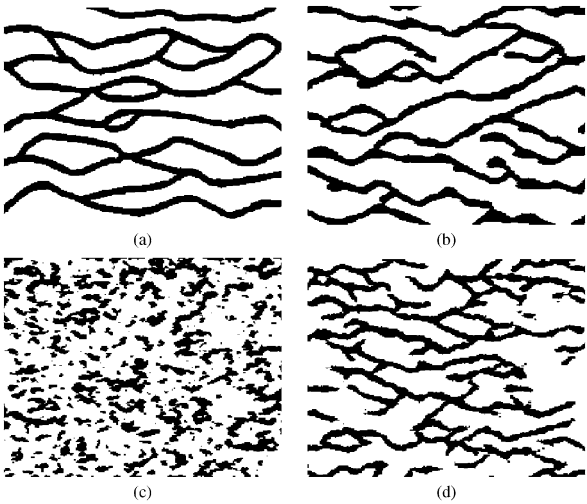

Fig. 8 Two-dimensional binary training image in (**a**) with realizations from the Markov mesh model in (**b**) and the *snesim* algorithm ((**c**) without multigrid and (**d**) with multigrid)

gram to compare our model with the multigrid *snesim* approach, since it does not make sense to compare it to *snesim* on one grid. We run the program on the training image and on 100 realizations from both methods. Statistics are displayed in the form of the box-plots in Fig. 9, where the straight vertical line represents the training image and the boxes span the data from the realizations. The leftmost box-plots represent the background facies and the rightmost represents the channel facies.

The number of channel objects is higher for *snesim* realizations because of all the loose ends. Consequently, the number of background objects increases and the extensions, average area, and average object volumes decrease. The latter measure in the box-plot is the average volume divided by the area of each object. This gives an indication of the smoothness of the edges of the objects. Channel objects from the Markov mesh model are slightly smoother than from *snesim*. The difference between the models for the background facies is much smaller than it appears in Fig. 9 because of the difference in scale. The general impression left by the comparison is that the distributions for the Markov mesh model encapsulate the training image more often than the *snesim* realizations.

5.2.1 Effect of Volume Fraction Steering

We run both our model and the *snesim* algorithm with and without volume fraction steering. We use the training image with channels from Fig. 7, which has a 0.28 fraction of channels. Figure 10 displays the results. The volume fraction before steering is 0.37 for the Markov mesh model and 0.33 for *snesim*. After steering, it is 0.29 for both methods. Note that for this particular training image, *snesim* fits the volume fraction quite well without steering.

When the Markov mesh model reproduces the facies fraction, we adjust the parameters based on their interpretation and fit a model that yields more background facies without losing the continuity of the channels. For *snesim*, volume fraction steering is

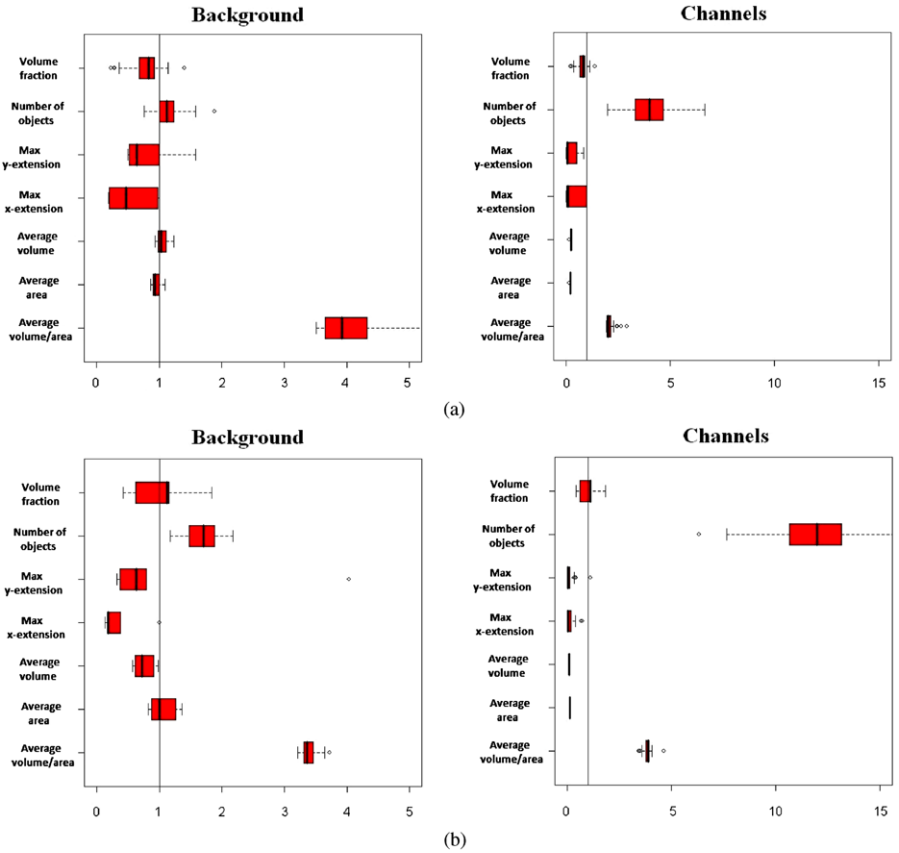

Fig. 9 Statistical analysis of Markov mesh realizations in (**a**) and *snesim* realizations in (**b**)

not concerned with the geological properties of the training image, only the global and local facies fractions during the simulation. This results in the many loose end channels that appear in the realizations.

5.3 Two-Dimensional Trinary Models

Here we give an example of a training image with three facies, background, channels, and crevasses. Again we display the results of *snesim* with and without multigrids (Fig. 11). We use a template of size 60 and three multigrids for *snesim*. For the Markov mesh model, we use a two-point interaction neighborhood with $l_x = 7$ and $l_y = 4$. The higher-point interaction neighborhood is set to length $L = 7$ for all directions. The Markov mesh model produces realizations with properties that appear to be smoother than for the *snesim* algorithm with multigrids. Note that we display only parts of the whole training image and the realizations to highlight the structure. The original training image is four times the size displayed in Fig. 11.

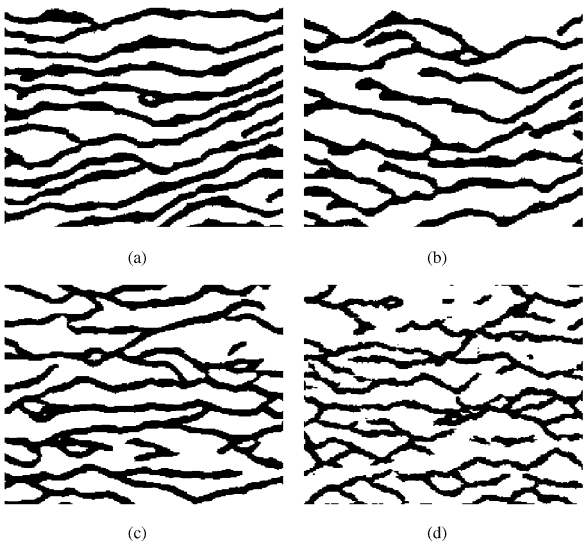

Fig. 10 In (**a**) and (**c**) Markov mesh and *snesim* realizations, respectively, without volume fraction steering. Similarly for (**b**) and (**d**) Markov mesh and *snesim* realizations, respectively, with volume fraction steering

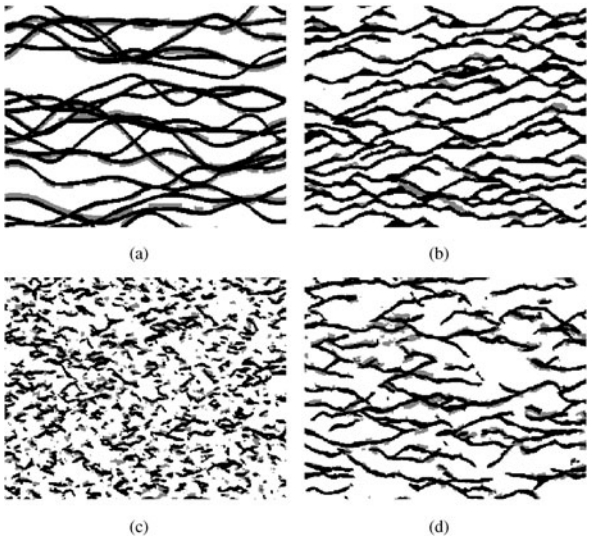

Fig. 11 Two-dimensional trinary training image in (**a**) with realizations from the Markov mesh model in (**b**) and the *snesim* algorithm without multigrid in (**c**) and with multigrid in (**d**)

5.4 Three-Dimensional Model

In three dimensions we use a training image with channels displayed by three cross-sections (Fig. 12). This training image is generated by an object model. The parameter choice for the model is set to $l_x = l_y = 4$ for the two-point interaction neighborhood and a length $L = 6$ for all directions of the higher-point interaction neighborhood. For *snesim*, we use a template of size 80 of equal radius in all directions, and three multigrids. The simulation results are promising (Fig. 12), but a more thorough analysis indicates that the model has problems preserving the continuity of channels.

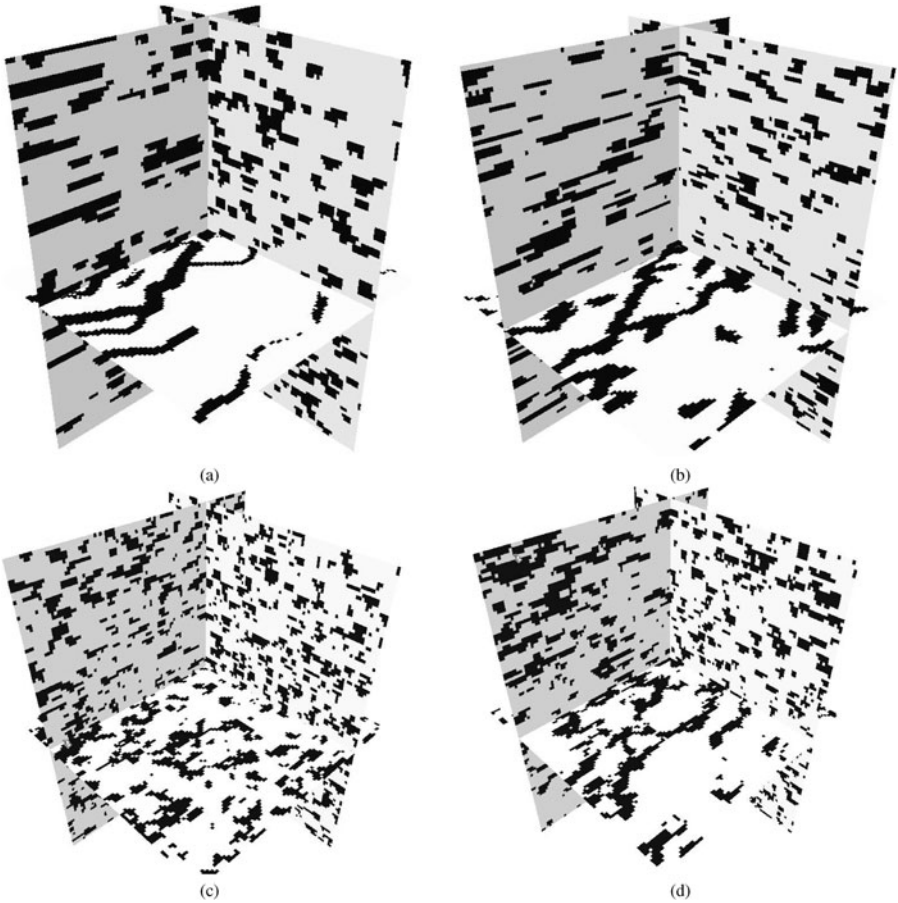

Fig. 12 Three-dimensional binary training image in (**a**) with realizations from the Markov mesh model in (**b**) and the *snesim* algorithm without multigrid in (**c**) and with multigrid in (**d**)

However, compared to *snesim*, the shape and continuity of the channel objects seem slightly improved.

6 Conclusions

We develop a Markov mesh model by using a GLM framework. This approach yields a consistent model that enables efficient parameter estimation and fast simulation. The parameterization of the model is chosen to capture the continuity of geological structures. We further adapt the approach such that it preserves the volume fractions of facies by increasing the probability of facies continuity. Test examples show good pattern reproduction. In two dimensions the model captures the curvilinearity and continuity of channel objects and at the same time reproduces the correct volume fraction. In one test case we observe skewness in the simulation pattern that may be

caused by the sequential simulation path. In three dimensions, the model performs reasonably, but tends to make objects too small. When compared to the *snesim* approach, our model yields substantially better results when defined in the same grid. The results obtained are comparable with those obtained by *snesim* using multigrids. In a test case, our approach better reproduces the statistics of the training image than *snesim* does with multigrids. Our choice of neighborhood functions is satisfactory on the examples we use.

Appendix: Generalized Linear Models

Standard generalized linear models assume that one has N independent observations of the model and one obtains the likelihood function

$$L\big(\boldsymbol{\theta}^1,\ldots,\boldsymbol{\theta}^K;\mathbf{Z},\mathbf{X}\big)=\prod_{i=1}^{N}\frac{\prod_k \exp\{\mathbf{z_i}^T\boldsymbol{\theta}^k x_i^k\}}{\sum_{j=1}^{K}\exp\{\mathbf{z}_i^T\boldsymbol{\theta}^j\}}, \tag{4}$$

where $\mathbf{X}$ is an $N\times K$ matrix of observations where the ith row contains 0-1 coding for the ith observation and $\mathbf{Z}$ is a design matrix of dimensions $N\times(P+1)$ where the ith row is the set of explanatory variables corresponding to the ith observation.

Standard derivations yield the following system of equations to be solved

$$\mathbf{Z}^T\mathbf{x}^k=\mathbf{Z}^T\mu^k\big(\boldsymbol{\theta}^1,\ldots,\boldsymbol{\theta}^K\big),\quad \text{for } k=1,\ldots,K,$$

where the vector $\mu^k(\boldsymbol{\theta}^1,\ldots,\boldsymbol{\theta}^K)$ is the $N\times 1$ vector of $\mu_i^k(\boldsymbol{\theta}^1,\ldots,\boldsymbol{\theta}^K)$, while $\mathbf{x}^k=[x_1^k,x_2^k,\ldots,x_N^k]^T$. The above expressions are solved using a gradient search, which yields the iterated weighted least squares where each iteration is given by

$$\boldsymbol{\theta}_{m+1}^k=\boldsymbol{\theta}_m^k+\big(\mathbf{Z}^T\mathbf{W}_m^k\mathbf{Z}\big)^{-1}\mathbf{Z}^T\big(\mathbf{x}^k-\mu^k\big(\boldsymbol{\theta}^1,\ldots,\boldsymbol{\theta}^K\big)\big)$$

for $k=1,\ldots,K$. Here $\mathbf{W}_m^k$ is an $N\times N$ matrix of weights given by

$$\mathbf{W}_m^k=\operatorname{diag}\big\{\big(1-\mu_1^k(\boldsymbol{\theta}_m)\big)\mu_1^k(\boldsymbol{\theta}_m),\ldots,\big(1-\mu_N^k(\boldsymbol{\theta}_m)\big)\mu_N^k(\boldsymbol{\theta}_m)\big\}.$$

This is the exact same equation used in our approach.

We reduce the number of parameters by extracting the principal components of the explanatory variables. This also solves the problems that arise when the design matrix is singular. In principal component analysis, we find the eigenvalues and eigenvectors of the empirical covariance matrix through the equality

$$\mathbf{VDV}^T=\mathbf{Z}^T\mathbf{Z},$$

where $\mathbf{V}$ is a $P\times P$ matrix of eigenvectors and $\mathbf{D}$ is a $P\times P$ diagonal matrix of eigenvalues. We select the $\tilde{P}<P$ eigenvectors $\tilde{\mathbf{V}}$ corresponding to the $\tilde{P}$ largest

eigenvalues and use these to generate the reduced $N \times \tilde{P}$ matrix $\tilde{\mathbf{Z}}$ of principal components

$$\tilde{\mathbf{Z}} = \mathbf{Z}\mathbf{V}.$$

From this set of reduced variables, we estimate the corresponding parameters $\{\tilde{\boldsymbol{\theta}}^1, \ldots, \tilde{\boldsymbol{\theta}}^K\}$. If we want a set of parameters that applies to the original variables, we find this by defining

$$\boldsymbol{\theta}^k = \mathbf{V}\tilde{\boldsymbol{\theta}}^k.$$

This is equivalent to computing $\tilde{\mathbf{Z}}\tilde{\boldsymbol{\theta}}^k$ or $\mathbf{Z}\boldsymbol{\theta}^k$, since we have the relation

$$\tilde{\mathbf{Z}}\tilde{\boldsymbol{\theta}}^k = \mathbf{Z}\mathbf{V}\tilde{\theta}^k = \mathbf{Z}\boldsymbol{\theta}^k.$$

References

Abend K, Harley T, Kanal L (1965) Classification of binary random patterns. IEEE Trans Inf Theory IT-11:538–544

Cressie N, Davidson J (1998) Image analysis with partially ordered Markov models. Comput Stat Data Anal 29(1):1–26

Daly C (2005) Higher order models using entropy, Markov random fields and sequential simulation, vol 1. Springer, Berlin, pp 215–224

Damsleth E, Tjølsen C, Omre H, Haldorsen H (1992) A two-stage stochastic model applied to a north sea reservoir. SPE JPT 44(4):402–408

Guardiano, Srivastava (1993) Multivariate geostatistics: beyond bivariate moments, vol 1. Kluwer, Dordrecht, pp 133–144

Kjønsberg H, Kolbjørnsen O (2008) Markov mesh simulations with data conditioning through indicator kriging, vol 1. Kluwer, Dordrecht, pp 257–266

Liu Y (2006) Using the Snesim program for multiple-point statistical simulation. Comput Geosci 32:1544–1563

McCullagh P, Nelder J (1989) Generalized linear models, 2 edn. Chapman and Hall, London

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

Skorstad A, Kolbjørnsen O, Fjellvoll B, Howell J (2005) Sensitivity of oil production to petrophysical heterogeneities. Springer, Dordrecht

Soleng H, Syversveen A, Kolbjørnsen O (2006) Comparing facies realizations: Defining metrices on realization space. In: Proceedings of the 10th European Conference in Mathematics of Oil Recovery, p A014

Strebelle S (2000) In: Multivariate Geostatistics: Beyond Bivariate Moments, vol 1

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

Tjelmeland H, Besag J (1998) Markov random fields with higher-order interactions. Scand J Stat 25:415–433