Computers & Geosciences 156 (2021) 104894

Contents lists available at ScienceDirect

Computers and Geosciences

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

Research paper

Estimation using multiple-point statistics

Óli D. Jóhannsson *, Thomas Mejer Hansen

Aarhus University, Høegh-Guldbergs Gade 2, 8000 Aarhus C, Denmark

ARTICLE INFO

Keywords:
Statistics
Estimation
Higher-order statistics
Multiple-point statistics

ABSTRACT

In the last two decades, several geostatistical simulation techniques have appeared that allow taking into account complex structural information, by quantifying a multiple-point statistical (MPS) model. The higher-order statistics are typically informed from a training image, sample model, or geological exposure. Such MPS models can represent the subsurface and are then used to simulate subsurface models with high(er) order of realism than is possible using, for example, the widely used 2-point statistical models (such as sequential Gaussian simulation).

All available methods dealing with MPS models are simulation methods, that can be used to generate multiple realizations of the underlying statistical model conditioned to hard and, to some extent, soft data. Such realizations can be very useful but are also computationally expensive to obtain. Here we consider the case when the end-user may not be interested in the set of produced realizations themselves, but rather in parameter-wise marginal statistical properties of a single model parameter derived from the simulated realizations. To obtain these, one would typically generate a larger number of independent realizations, and then compute some marginal statistics of these.

Here we propose an MPS estimation algorithm, a variant of the widely used sequential simulation algorithm, that can be used to directly compute and store parameter-wise conditional statistics. This allows for a potentially faster and more accurate estimation than using sequential simulation. The method is demonstrated on both ENESIM- and SNESIM-type MPS algorithms and results compared using both sequential simulation and estimation. As an example, the method is applied for estimating the existence of near-surface buried valley systems in Kasted, Denmark.

1. Introduction

Geostatistical modelling can be seen as a way to quantify information about structural relationships between one or more physical parameters, $\mathbf{m}$, through a probability distribution $f(\mathbf{m})$. Once inferred, or selected, such a geostatistical model can be combined with for example geophysical data or well-log data to provide a probabilistic model of the physical parameters of interest. Information can then be inferred about sets of model parameters at any location in space. Most geostatistical models can be grouped in one of two different types of models. '2-point' statistical models are based on statistics between pairs of points. A typical example is the multivariate Gaussian model defined by a mean and the covariance between pairs of model parameters. 2-point statistical models can be both parametric and non-parametric (Deutsch and Journel, 1992; Goovaerts, 1997). 'Multiple-point statistical' (MPS) models utilize statistics between multiple model parameters at once. Multiple-point statistics are often inferred from a training image (TI), that represents expected spatial variability. MPS models allow representing more complex, and geologically realistic, structures than 2-point statistical models (Mariethoz and Caers, 2014; Tahmasebi, 2018).

There are two fundamental different uses of geostatistical models: estimation and simulation. The goal of using geostatistical 'estimation' methods is to estimate the expected value of a single physical parameter m_i, given a geostatistical model, and information available about some of the model parameters, $\mathbf{m}_{obs}$, in the form of the probability distribution $f(m_i|\mathbf{m}_{obs})$. A typical example can be that of estimating properties of the value of a well-log (e.g. porosity) away from well-log locations where the porosity has been directly measured. One can then use estimation methods to compute for example the most probable porosity value at any position in space, say $\mathbf{m}_{ML} = [m_{1\,ML}, m_{2\,ML}, \ldots]$. Geostatistics as a research field originated in the development of such estimation methods with application to mining statistics (Krige, 1951; Journel and Huijbregts, 1978). Estimation methods provide parameter-wise statistics, that should be analysed parameter by parameter. An image of an estimated property, such as $\mathbf{m}_{ML}$, is a statistical property of $f(\mathbf{m})$, but it does not represent a realization of $f(\mathbf{m})$. In the multivariate

* Corresponding author.
E-mail addresses: oli.d.johannsson@geo.au.dk (Ó.D. Jóhannsson), tmeha@geo.au.dk (T.M. Hansen).

https://doi.org/10.1016/j.cageo.2021.104894
Received 11 December 2020; Received in revised form 10 June 2021; Accepted 21 July 2021
Available online 4 August 2021

Gaussian case, $f(m_i|\mathbf{m}_{obs})$ is a 1-D normal distribution where the mean is exactly $\mathbf{m}_{ML}$. The mean and standard deviation can in this case be computed analytically using for example simple kriging (Journel and Huijbregts, 1978).

The goal of geostatistical 'simulation' is on the other hand to generate multiple realizations of a specific geostatistical model $f(\mathbf{m})$. A realization, $\mathbf{m}^*$, represents an outcome of $f(\mathbf{m})$. Any statistical property of $f(\mathbf{m})$ can be estimated from a sample, a set of independent realizations, of $f(\mathbf{m})$. If $f(\mathbf{m})$ is based on a TI with meandering channel structures, then a realization $\mathbf{m}^*$ should also show meandering channel structures. The ability to generate such realizations of $f(\mathbf{m})$ is essential to analyse for example flow response of hydrological models. From a 3D geostatistical model representing hydrological parameters, one can generate a set of realizations of hydrological parameters (with realistic spatial variability), based on which flow modelling can be performed to realistically quantify uncertainty related to the hydrological model (Barfod et al., 2018; Vilhelmsen et al., 2019). This is not possible using the models computed using estimation methods. Goovaerts (1997) and Deutsch and Journel (1992) provide an excellent overview of estimation and simulation methods based on 2-point statistics and the multivariate Gaussian model, while Mariethoz and Caers (2014) provide a thorough introduction to MPS based simulation methods.

While 2-point geostatistical models were first suggested for estimation (Journel and Huijbregts, 1978), and later developed for simulation (Goovaerts, 1997; Deutsch and Journel, 1992), MPS models have solely been developed for simulation (Mariethoz and Caers, 2014). parameter-wise statistics from an MPS model can be found by generating multiple realizations, from which any parameter-wise statistics can be computed. This can be time-consuming, as even a single realization of a 3D model based on MPS can be computationally expensive. Further, in some cases, one may not be interested in the realizations themselves, but only the parameter-wise statistics. In such cases, it would be useful to be able to estimate the parameter-wise 1-D statistics directly without the need for expensive simulation.

In the following, we present an efficient and simple to implement MPS estimation algorithm and demonstrate it on a small synthetic example, as well as a real-world case. This MPS estimation algorithm will provide estimation for MPS models, as simple kriging offers estimation for multivariate Gaussian models.

2. Theory

Consider a model space $\mathcal{M}$ parameterized into M model parameters $\mathbf{m} = [m_1, m_2, \dots, m_M]$. A specific value of $\mathbf{m}$ represents a point in $\mathcal{M}$. $\mathbf{m}$ typically represents a physical property in a 3D geo-model consisting of M voxels. Let $f(\mathbf{m})$ represent a joint discrete probability mass function (pmf) or continuous probability density (pdf) function. In the following, it will be referred to as a probability density (pd).

Using the chain rule, the joint pd $f(\mathbf{m})$ can be written as the product of a set of conditional probability densities:

$$\begin{aligned} f(\mathbf{m}) &= f(m_1, m_2, \dots, m_M) \\ &= f(m_1)\ f(m_2|m_1)\ \dots\ f(m_M|m_1, \dots, m_{M-1}). \end{aligned} \tag{1}$$

2.1. Sequential simulation

In geostatistics, the chain rule has been utilized through the widely used sequential simulation algorithm (Journel and Alabert, 1989), also known as the conditional distribution method (Devroye, 1986), in which a realization of $f(\mathbf{m})$ can be generated by sampling from a series of univariate conditional distributions.

In sequential simulation, the M model parameters $\mathbf{m} = [m_1, m_2, m_3, \dots, m_M]$ are visited, one by one (*i.e.* sequentially, but in any order). At each step, a realization of the currently visited model parameter, m_i, is generated as m_i^* from an univariate conditional distribution, through the following algorithm:

$$\begin{aligned} &\text{Visit } m_1, \text{ sample } m_1^* \text{ from } f(m_1) \\ &\text{Visit } m_2, \text{ sample } m_2^* \text{ from } f(m_2|m_1^*) \\ &\text{Visit } m_2, \text{ sample } m_3^* \text{ from } f(m_3|m_2^*, m_1^*) \\ &\qquad\vdots \\ &\text{Visit } m_M, \text{ sample } m_M^* \text{ from } f(m_M|m_{M-1}^*, m_{M-2}^*, \dots, m_1^*) \end{aligned} \tag{2}$$

$\mathbf{m}^* = [m_1^*, m_2^*, \dots, m_M^*]$ then represents an unconditional realization of $f(\mathbf{m})$.

Most often some information about some of the model parameters are known either in the form of known values of some model parameters (hard data) and/or as uncertain information (soft data) about some model parameters. Let $\mathbf{m}_c$ refer to such known information about the model parameters. In order to simulate from $f(\mathbf{m})$ conditional to $\mathbf{m}_c$, i.e. from $f(\mathbf{m}|\mathbf{m}_c)$, one must simply condition to $\mathbf{m}_c$ at each step of the sequential simulation, such that at iteration number i one samples from

$$f(m_i|m_1^*, \dots, m_{i-1}^*, \mathbf{m_c}). \tag{3}$$

The ability to generate a sample of this univariate conditional distribution is at the core of any sequential simulation-based algorithm.

2.2. Estimation

The goal of estimation is, on the other hand, to evaluate, for any m_i, the marginal conditional distribution

$$f(m_i|\mathbf{m_c}). \tag{4}$$

As discussed, this can be achieved by generating multiple, say N, realizations $[\mathbf{m}^{1*}, \mathbf{m}^{2*}, \dots, \mathbf{m}^{N*}]$ from $f(\mathbf{m}|\mathbf{m_c})$ using the sequential simulation algorithm. From these realizations any parameter-wise statistic of m_i can be calculated from $[m_i^{1*}, m_i^{2*}, \dots, m_i^{N*}]$.

Most simulation algorithms based on sequential simulation compute the conditional distribution, Eq. (3), as part of running the algorithm. A notable exception is the direct sampling algorithm that samples from the conditional pd directly, without explicitly computing it (Mariethoz et al., 2010). A simple variation of the sequential simulation algorithm can, therefore, be considered that allow direct estimation of the conditional statistics. Instead of simulating a value m_i^* from the conditional distribution at each iteration, it is suggested to simply store the conditional distribution. This leads to the following estimation algorithm:

$$\begin{aligned} &\text{Visit } m_1, \text{ compute and store } f(m_1|\mathbf{m}_c) \\ &\text{Visit } m_2, \text{ compute and store } f(m_2|\mathbf{m}_c) \\ &\text{Visit } m_3, \text{ compute and store } f(m_3|\mathbf{m}_c) \\ &\qquad\vdots \\ &\text{Visit } m_M, \text{ compute and store } f(m_M|\mathbf{m}_c) \end{aligned} \tag{5}$$

In principle any algorithm that is based on sequential simulation can be used for estimation using the above method. For example, simple kriging can be applied using Eqs. (5) when one wishes to perform parameter-wise conditional estimation using multivariate Gaussian model, as for example implemented in Hansen and Mosegaard (2008). Here we consider using Eqs. (5) for estimation based on MPS models. The memory overhead using estimation rather than simulation is at most the memory needed to store the full conditional distribution for each parameter. This amounts to $N_o - 1$ times the size of the simulation grid, for discrete models with N_o categories. For continuous models it becomes more complex, and either a discretization or parameterization of the conditional probability distribution can be performed. For very large and complex models, memory requirements

can be reduced by only storing the statistical properties of interest, instead of the full conditional distribution.

While the model parameters must be visited in a sequential manner using sequential simulation, as each iteration depends on the outcome of previous iterations, Eqs. (1)–(3), estimation, Eqs. (5), can be performed in any order.

3. Estimation for multiple-point statistical models

Guardiano and Srivastava (1993) propose a general algorithm (the ENESIM algorithm) for borrowing conditional statistics from a TI, such that it can be used with the sequential simulation algorithm to simulate from an MPS model. The ENESIM algorithm has proved to be the basis of many other MPS simulation methods (Strebelle, 2002; Mariethoz et al., 2010; Straubhaar et al., 2011; Hansen et al., 2016). These methods differ mostly in how the conditional statistics are retrieved from the TI, i.e. in how $f(m_i|\mathbf{m}_c)$ in Eq. (3) is established and sampled from. In one group of algorithms, the TI is scanned during runtime to find and sample from the conditional distribution (ENESIM Guardiano and Srivastava, 1993, GENESIM Hansen et al., 2016, Direct Sampling Mariethoz et al., 2010). In another group of algorithms, the TI is scanned before running the sequential simulation algorithm, and the conditional statistics are stored in memory for easy later retrieval (SNESIM-tree Strebelle, 2002, SNESIM-list Straubhaar et al., 2011).

The first example will be based on a small test case for reference, designed not to be realistic, but to provide some insights. Fig. 1a shows a binary TI (from Strebelle (2002)) that is believed to represent an underlying discrete pd, $f(\mathbf{m})$. Fig. 1b shows a case with a small amount of conditioning data, where a channel is known to exist at 3 data points. Below we compare the parameter-wise probability of locating a channel, using sequential simulation and MPS estimation (using both ENESIM and SNESIM type algorithms) respectively, given the TI and observed data in Fig. 1.

3.1. ENESIM/GENESIM

In each iteration of the ENESIM algorithm, the conditional pd, (Eq. (3)) is obtained by scanning a TI for *all* occurrences of the conditioning event (relative to each pixel m_i) (Guardiano and Srivastava, 1993). The GENESIM algorithm is a simple variant of the ENESIM algorithm, in which the TI is scanned for a maximum of n_{max} occurrences of the conditional event (Hansen et al., 2016), which leads to faster simulation time. In one extreme, $n_{max} = \infty$, GENESIM will behave exactly as the ENESIM algorithm. In another extreme $n_{max} = 1$, GENESIM will behave like the Direct Sampling algorithm, in which at each iteration the first matching event, along a random search path, is selected as a realization of the $f(m_i|\mathbf{m}_c)$ (Mariethoz et al., 2010).

In case using ENESIM and GENESIM the conditional pd in Eq. (4) is computed and available using any implementation, and hence it is, in principle, trivial to adapt ENESIM and GENESIM simulation algorithms for MPS estimation, simply by storing the conditional pd at each iteration, and by ignoring the step of simulating an outcome from the conditional pd and adding it to the realization to the list of conditional data.

The main control parameter for ENESIM type algorithms is the number of conditioning data, n_c, used to construct the conditional event. Also, one may select a limited search neighbourhood within which to search for conditioning data. Here an exhaustive neighbourhood is used. The results shown below are generated using the GENESIM algorithm with $n_{max} = 1000$ (i.e. where the conditional pd is based on a maximum of 1000 occurrences of the conditional event).

Fig. 2a shows the parameter-wise probability of locating a channel, obtained from 1000 realizations obtained using sequential simulation, in case using $n_c = [0, 1, 2, 4, 6, 8, 16, 32, 64]$. Fig. 2b shows the corresponding results obtained using MPS estimation. Fig. 3 shows the CPU running time both in case using estimation and simulation, as well as the average of the entropy of the conditional distribution, H_{av}, defined as

$$H_{av} = \frac{1}{N}\sum_{i=1}^{N} H(f(m_i|\mathbf{m}_c)) \tag{6}$$

$$= \frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{N_o} f(m_i = m^j|\mathbf{m}_c)\ log_{N_o}(f(m_i = m^j|\mathbf{m}_c)), \tag{7}$$

where N is the number of parameters being simulated, N_o is the number of categorical outcomes, and m^j denotes the jth possible outcome.

The entropy is a measure of lack of information, or the uncertainty, of a pd (Shannon, 2001). The average conditional entropy H_{av}, as defined in Eq. (7), is a number between 0 and 1. $H_{av} = 0$ suggests that each model parameter is completely informed (with no uncertainty), and $H_{av} = 1$ suggests that all model parameters are completely uninformed (which is the case if all conditional distributions are uniformly distributed). In the following, we argue that, in general, a lower value of H_{av} reflects a better result as it suggests that more information has been extracted from the available data.

Comparing Fig. 2a and b it seems evident that as the number of conditional data increases, using both sequential simulation and MPS estimation leads to similar results. It also suggests that one can use fewer conditional data using MPS estimation as compared to sequential simulation, to achieve the same level of information.

This trend is confirmed and quantified by the average conditional entropy in Fig. 3 where the dashed lines show the average conditional entropy H_{av} as function of n_c, using both sequential simulation and estimation. In case $n_c = 0$ the average conditional entropy is naturally similar and high, and in case $n_c = 64$ the average entropy is similar but, smaller. However, with estimation, the lowest average conditional entropy is found using $n_c = 4$, while the lowest level, using simulation, is found at $n_c \geq 32$. The reason for this is that using MPS estimation more of the conditional information is taken into account than using simulation. During sequential simulation, the conditional distribution is typically based on the n_c closest nodes (model parameters), regardless of whether they belong to the original conditioning data, or are simulated data; the algorithm does not distinguish. This reduces the weight of the conditional data relative to using MPS estimation.

In theory, the full entropy of a pd cannot increase when adding additional conditional information. Yet, the red curve, reflecting H_{av} increases from $n_c = 0$ to $n_c = 2$. The reason this happens is that the full entropy is not considered, but only the average marginal entropy. See e.g. Hansen (2020) for details on entropy of MPS models.

Fig. 3 shows that each single realization can be simulated faster using sequential simulation than by performing MPS estimation for $n_c \leq 32$. Recall that only at $n_c = 32$ does an average of 1000 realizations of sequential simulation provide similar results in terms of information content as the MPS estimation. For this particular example, the MPS estimation can be applied at the same time it would take to create a single realization using sequential simulation with the same kind of information content (using $n_c = 4$ for MPS estimation, and $n_c = 32$ for sequential simulation). In order to compute the marginal statistics, such as the probability of locating channel, this (and any other 1-D marginal statistical property) one would need to run the MPS estimation only once, but the sequential simulation would need to be run many times, depending on how reliable the marginal statistics need to be. Here we used 1000 realizations, which then leads to a computational speedup of 1000 comparing MPS estimation to simulation.

3.2. SNESIM

SNESIM-tree (Strebelle, 2002) and SNESIM-list (Straubhaar et al., 2011) differ from ENESIM type algorithms in that the TI is scanned only once, to obtain conditional statistics, which is then stored in a search-tree (SNESIM-tree) or lookup list (SNESIM-list) in memory. Both

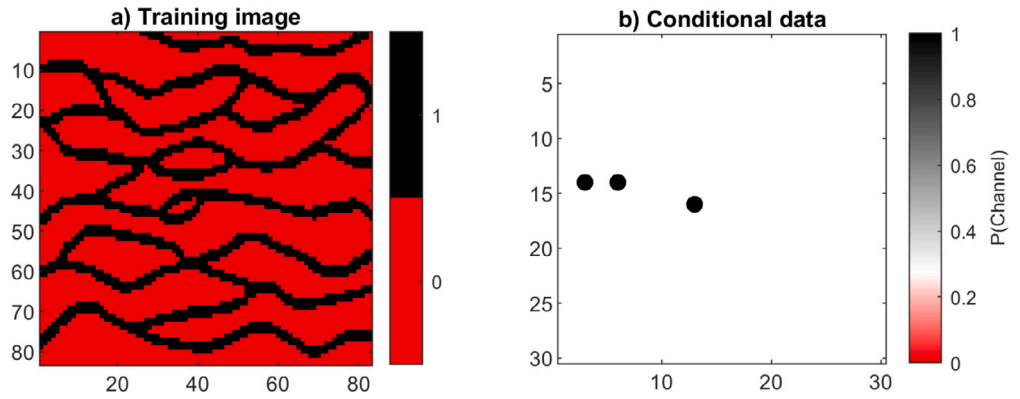

Fig. 1. (a) Training image (Strebelle, 2002), (b) conditional data set. Black: Channel, Red: No channel. White: marginal distribution of (a). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

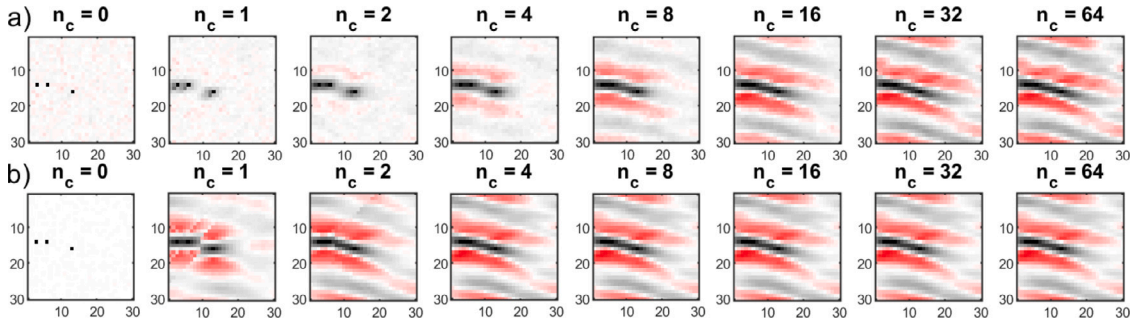

Fig. 2. Pointwise mean probability of channel, obtained using ENESIM from (a) 1000 realizations obtained using sequential simulation and (b) MPS estimation. Colourscale as in Fig. 1b. White indicates the marginal probability of a channel, $f(m = 1)$, as computed from the training image.

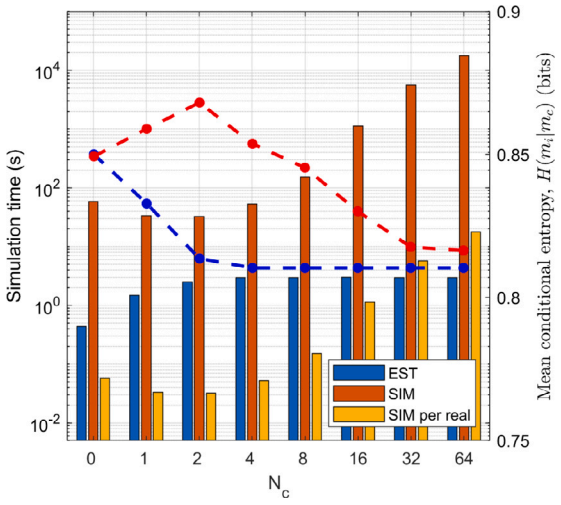

Fig. 3. CPU time for 1 estimation, 1000 realizations, and 1 realization using ENESIM. Mean conditional entropy shown as dotted lines, see right vertical axis. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

algorithms can in principle easily be adapted for MPS estimation as described above.

SNESIM-tree and SNESIM-list should provide the same results (for both simulation and estimation), as they differ mostly in how the conditional statistics are stored. SNESIM-tree typically requires more memory and is computationally very efficient when conditional data are close to the simulation point, but less efficient when conditional data a sparse and located further away from the simulation point (Strebelle, 2002). SNESIM-list requires less memory and is less sensitive to the distribution of the conditioning event (Straubhaar et al., 2011).

Both SNESIM-tree and SNESIM-list makes use of a search neighbourhood and multiple grids, in order to reduce the size of the stored conditional statistics and to increase simulation efficiency. For simplicity, we use SNESIM algorithms without multiple grids, for easier comparison with MPS estimation.

In the following, a search neighbourhood of 20 × 20 pixels and no multigrid is used.

Fig. 4a shows the parameter-wise probability of locating a channel, obtained from 1000 realizations obtained using sequential simulation and SNESIM-list, in case using $n_c = [0, 1, 2, 4, 6, 8, 16, 32, 64]$. Fig. 4b shows the corresponding results obtained using MPS estimation. The results obtained using SNESIM-tree are similar, and not shown.

The use of a specific template is apparent as little (in case using simulation) to no (in case using estimation) information is noticeable in the rightmost part of the model, farthest away from the conditional data, outside the template distance of ± 10 pixels from the conditional data. Besides this, the difference in information content between simulation and estimation is the same as for the ENESIM cases: More information is retrieved, for smaller conditional event size, n_c, using estimation and simulation. This is also clear from Fig. 5a, that show that the lowest value H_{av} (indicating max level of information) is reached at $n_c = 4$. Using sequential simulation the same level of entropy can be reached only using $n_c > 16$.

While the simulation results will be similar using SNESIM with a list or a tree structure to store conditional statistics, the simulation time differs, as shown in Fig. 5a–b.

For estimation, the use of list structures, SNESIM-list, provides a much faster estimation, Fig. 5a, than using tree structures, SNESIM-tree, Fig. 5b. The reason is that it is computationally expensive to traverse a

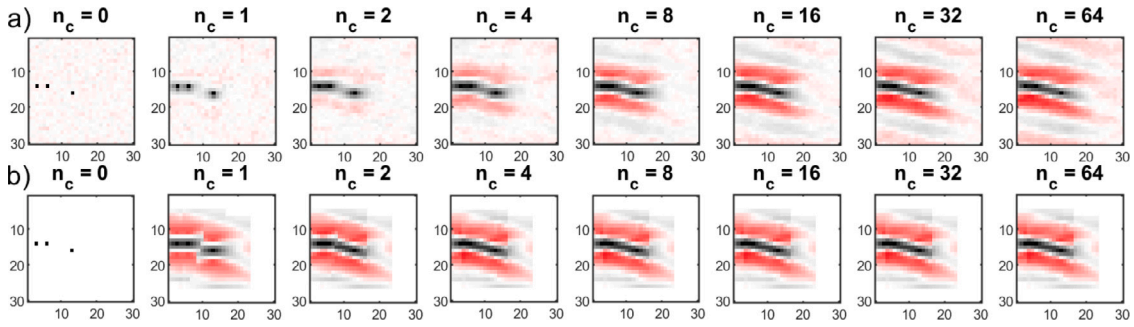

Fig. 4. Parameter wise mean probability of channel, obtained using SNESIM-list from (a) 1000 realizations obtained using sequential simulation and (b) MPS estimation. Colourscale as in Fig. 1b. White indicates the marginal probability of a channel, $f(m = 1)$, as computed from the training image.

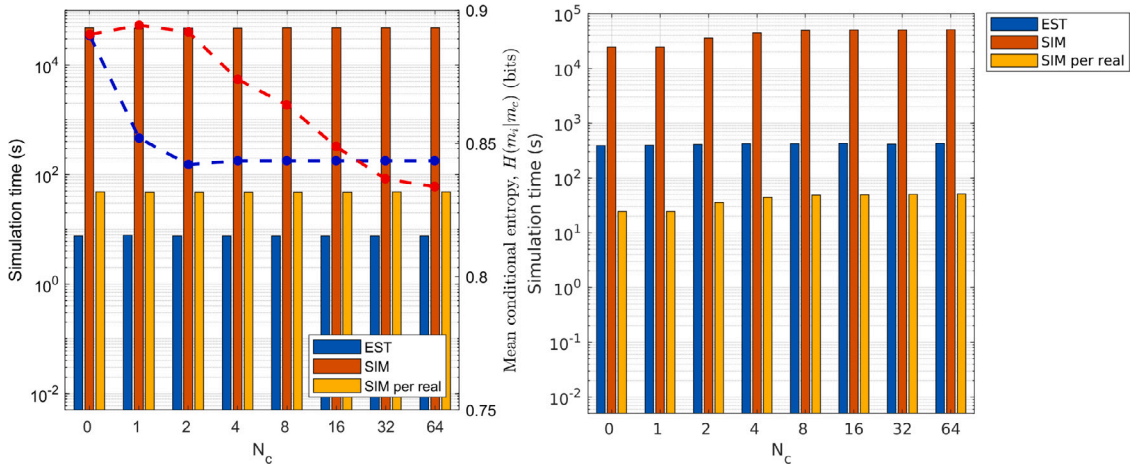

Fig. 5. CPU time for 1 estimation, 1000 realizations, and 1 realization using **(left)** SNESIM-list (with mean conditional entropy shown as dotted lines, see right vertical axis), **(right)** SNESIM-tree.

search tree for a sparse conditional template. In general, using SNESIM-tree sequential simulation (generating 1 realization) is about 10 times faster than MPS estimation.

Using list structures, on the other hand, there is little computational difference in searching for a data event consisting of conditional data that are either close or far away from the parameter being simulated. In addition, using a list structure, there is only a little difference in CPU time scanning the list for either a small or larger conditional event, as can be seen in Fig. 5, where the time to perform MPS estimation is not very sensitive to the number of conditional data. In general, using SNESIM-list, MPS estimation is about 10 times faster than using sequential simulation (to generate 1 realization).

From these results, it is evident that SNESIM-list using list structures is much better suited for MPS estimation than using SNESIM-tree with a search tree, as it leads to an estimation of 1-D conditional statistics several orders of magnitude faster than using SNESIM-tree.

3.3. ENESIM/SNESIM (DIRECT SAMPLING/GENESIM)

Based on the results above we suggest to make use of the GENESIM algorithm to perform MPS estimation, as (1) it is the fastest of the considered methods (for the specific case considered), (2) does not rely on a search template, and (3) it does not rely on multiple grids and is hence not prone to any issue related to data relocation on coarse grids.

SNESIM-list may be an option, especially if it can be used with very large templates, in order to avoid using multiple grids. Finally, we suggest avoiding using MPS estimation with the SNESIM-tree algorithm as is, because it will be computationally inefficient, due to how the search tree is traversed using sparse conditional data.

In the following example, we will demonstrate the use of GENESIM for a practical application of MPS estimation.

4. Locating buried valleys in Kasted Denmark

Buried valleys (Sandersen and Jørgensen, 2003; Jørgensen and Sandersen, 2006; Piotrowski et al., 2009) are common in parts of northern Europe, that had ice-cover during the Quaternary. The features were likely eroded, by meltwater released in *jökulhlaups*, and have since been filled by younger sediments, that often act as a good aquifer; hence buried valleys can be of great importance to groundwater, and therefore mapping them is of both scientific interest and of importance to groundwater resource management.

Since these networks of buried valleys can be quite complex (e.g. intersections of multiple long, meandering features), mapping them using the relatively sparse data from boreholes can be difficult, especially when limited to using Kriging, Gaussian simulation, or other 2-point statistical methods. Airborne electromagnetic surveys have been very useful as a base for geologists to create detailed 3D maps of buried valleys (Høyer et al., 2015). The process is though expensive, time-consuming, and requires several steps of manual interpretation.

Here we consider the problem of identifying the existence of buried valleys in a 2D grid, based on information from boreholes and a TI representing the expected spatial variation of buried valley structures.[1]

Fig. 6 shows the location of 112, out of in total 1137, boreholes situated near Kasted, Denmark. At these 112 locations, the information from the boreholes allows confident identification of either "buried valley", or "no buried valley". We use this data set as hard data. Information from the other boreholes is either inconclusive, e.g. the

[1] Code and data used for estimation based on Kasted data is available at https://github.com/ergosimulation/mpslib/tree/master/data/kasted.

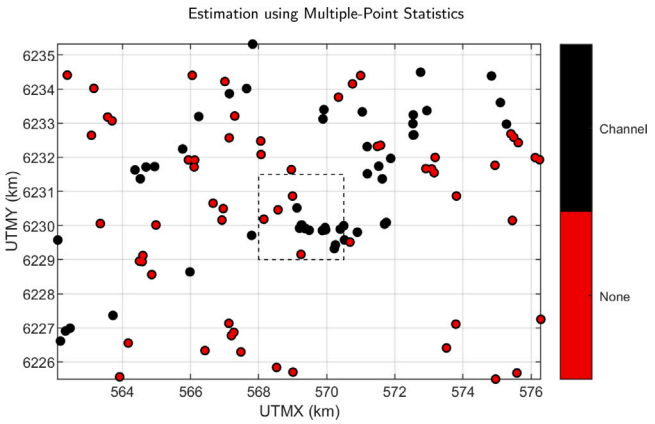

Fig. 6. Location of well logs, and interpreted probability of locating a buried valley, around Kasted, Denmark. Dotted rectangle: subset used in analysis of estimation algorithm parameters.

boreholes may be too shallow, and therefore these sites have been ignored.

Fig. 7a (see the 'original' layer) shows a representation of expected spatial distribution of buried valleys in Kasted. It is a manually crafted image, combined of sections of buried valley structures from other locations in Denmark (Exizidou, 2014), believed to represent the same type of underground as present in Kasted. All the data in Figs. 6–7a is based on Exizidou (2014).

4.1. Practical application of multiple-point statistical estimation and considerations

The specific configuration of the hard data in Fig. 6 does not exist in the original TI, Fig. 7a. So, in practice, there is inconsistency between the hard data and information in the TI. This is often an issue when applying MPS methods in practice. In this case-study the amount of conditioning data is much higher than for the simple test-cases considered in Fig. 1, and, at the same time, the TI is relatively small compared to the size of the area in which it is used for estimation. In traditional MPS based sequential simulation, this will lead to some simulation artefacts related to the lack of conditional statistics, caused by inconsistency between conditional data and the patterns in the TI. Similarly, in MPS estimation as we consider here, such inconsistencies lead to some issues with locating enough conditional events to obtain a well-resolved conditional distribution, Eq. (3).

In order to get a well resolved conditional probability distribution for each model parameter, there are some strategies, each with their benefits and disadvantages. These strategies refer to, for example:

- Increasing the size and variability of the training image
- Decreasing the number of conditioning data used
- Allowing for a mismatch between the data event, and the matches in the training image.

4.1.1. Increasing training image size

The TI in Fig. 6b, represents one specific example of how the spatial distribution of the buried valleys could be. In reality, the buried valleys could have slightly different thickness and could be rotated, and still be consistent with the geological knowledge. A simple way to describe such expected variability of the TI is by constructing several TIs that represent slight variations of the original TI.

Morphing operations, such as scaling, rotating, or growing/eroding features provides a way to add variability to the TI. This can also be implemented at an algorithm level (Mariethoz and Kelly, 2011). We add some variability to the TI, by morphing the thicknesses of the buried valley structures, without changing the connectivity or anisotropy of the conceptual buried valley network that the TI represents. This can increase the number of matches that the algorithm finds for given data-event, while still being representative of the kind of structures we expect to find. Any manipulations of the training image have to be assuredly within the conceptual model that the expert knowledge dictates.

Four consecutive morphological erosion operations were performed, corresponding to four increments of 100-metre thinner buried valleys (See Fig. 7). No grow operations were performed, as the marginal distribution of the TI, is already skewed towards the existence of buried valley compared to the conditioning data. The training image used is comprised of all of the images.

4.1.2. Decreasing the number of conditioning data used

The simplest way to increase the number of conditional data-events to represent Eq. (3), is to limit the number of conditioning data used, n_c, used to estimate each model parameter. This has the immediate drawback of reducing the effective neighbourhood size (distance from model parameter being estimated to the most distant conditioning data being used). This leads to an increase in entropy (disorder), and thus may introduce unwanted artefacts such as disconnected valleys. Therefore, one should not use a too-small number of conditioning data.

4.1.3. Allowing for mismatch between the data event, and the matches in the training image

An alternative to reducing the number of conditioning data is to introduce a weight that favours matching of conditional events close to the parameter being simulated, as opposed to events further away.

A distance measure, d, between two conditioning events, $\mathbf{m}_c^1, \mathbf{m}_c^2$ can, following Mariethoz et al. (2010), be introduced as:

$$d(\mathbf{m}_c^1, \mathbf{m}_c^2) = \frac{\sum_{i=1}^{n_c} a_i \|\mathbf{h}_i\|^{-p}}{\sum_{i=1}^{n_c} \|\mathbf{h}_i\|^{-p}}, \tag{8}$$

where $\mathbf{h}_i$ is the lag vector between the parameter being estimated (or simulated) and elements of the conditioning data. Additionally let $a_i = 0$ when $m_{c\,i}^1 = m_{c\,i}^2$ and $a_i = 1$ when $m_{c\,i}^1 \neq m_{c\,i}^2$ for categorical models. For models with continuous variables Euclidean distance can be used to calculate an appropriate a_i.

Conditioning data closer to the point being simulated will carry a relatively higher weight, which can be controlled by the exponential decay parameter p. In the following we use $p = 1$.

Mariethoz et al. (2010) suggest to accept conditional events with a maximum distance of d_{max} in the Direct Sampling, and we make use of it here to accept events when establishing the conditional distribution, Eq. (3). Selecting $d_{max} = 0$ implies that only a perfect match with the conditional event will be accepted, just as in the original ENESIM algorithm. $d_{max} > 0$ will lead to accepting increasingly more conditional events (with a preference for matching events close to the parameter being simulated, up until $d_{max} = 1$ which implies that all conditioning events will be accepted, which will imply that the conditional distributions will be exactly the 1-D marginal distribution of the TI (i.e. conditional distributions will be unconditional). Thus the distance measure allows, through the definition of one parameter, d_{max}, an alternative approach to increase the number of conditional statistics.

4.1.4. Parameter sensitivity analysis

In this section, we investigate the effect of varying the allowed distance $d_{\max}$ and the number of conditioning data used, n_c, using both sequential simulation and estimation. Figs. 8 and 9 show the probability of locating a buried valley in a small subset (see dotted rectangle in Fig. 6a) of the full 2D simulation grid, obtained using simulation (from 1000 realizations) and MPS estimation respectively.

Using sequential simulation, Fig. 8 demonstrates that increasing n_c and decreasing $d_{\max}$, leads to more well pronounced features. Qualitatively there is no incentive to use $d_{\max} > 0$ in simulation, apart from

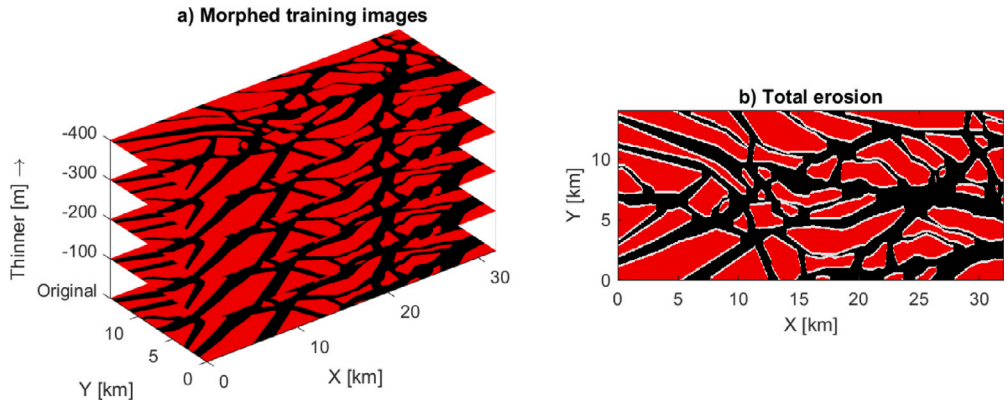

Fig. 7. (a) Training images used. Each individual training image is based on Fig. 6b, and has had the buried valley features eroded using morphological operations resulting in narrower buried valleys. The spatial resolution is 50 by 50 metres per pixel. Adapted from Exizidou (2014) (b) Illustration of the maximal eroded area shown in white.

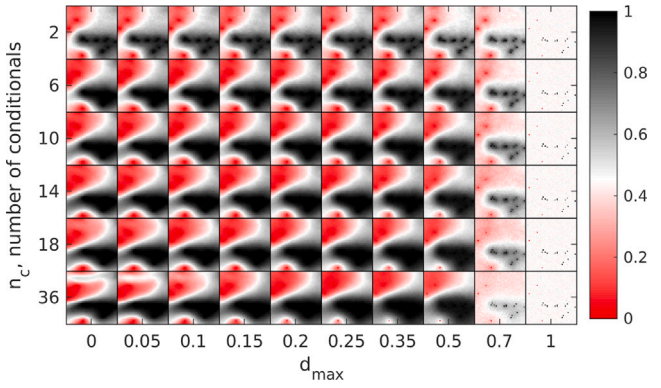

Fig. 8. Probability of buried valley 1-D marginal of 1000 realizations, achieved through MPS simulation, for different values of the two parameters, number of conditional data used n_c and allowed mis-match distance, d_{max}. White is set to the marginal distribution of the training image, and hence corresponds to the least informed state.

the massively reduced computation time (more than a factor 10 in reduction in simulation time is observed going from d_{max}=0 to d_{max} = 0.25 for example).

Using MPS estimation, the results are different (Fig. 9), as low values of $d_{max}(\leq 0.2)$ introduce apparent artefacts when the number of conditioning data is increased. These can be recognized as discontinuities and isolated pixels (see for example the very lower left corner of Fig. 9), and are due to boundary effects and lack of consistent conditional data.

The discontinuities are related to boundary effects using a limited neighbourhood and are apparent in the top row of Fig. 9. Apparent distinct discontinuities, indicate the boundary between different local neighbourhoods for a specific sets of conditional data. This effect is inherent in MPS algorithms, but becomes negligible as the number of conditioning data increases, e.g., as the neighbourhood size becomes large enough (Mariethoz and Caers, 2014). What is seen in the left column in Fig. 9, however, is that the artefacts produced by limited neighbourhood are present until artefacts produced by a low count of matches in the TI arise. While boundary effects are present in MPS simulation, parameter wise statistics tends to average out these effects, as seen in Fig. 8.

A simple approach to alleviate issued related to boundary effects, is to use increase the size of the conditioning event, i.e. to increase n_c. As the number of conditioning data used, n_c, increases, the neighbourhood size increases as well, and stays large throughout. While in simulation, the neighbourhood size quickly becomes smaller as model parameters are simulated and are then used as new conditioning data. The consequence is that the estimation algorithm is scanning the TI for very large data-events, that by their nature have a smaller probability of existing in a finite-sized TI, than corresponding smaller data-events.

To quantify when artefacts due to low counts of conditioning events appear, recording the number of counts is helpful. Fig. 10 shows the number of conditioning events (the number of samples of the TI, that matched the data event: i.e. the conditioning data used) used to

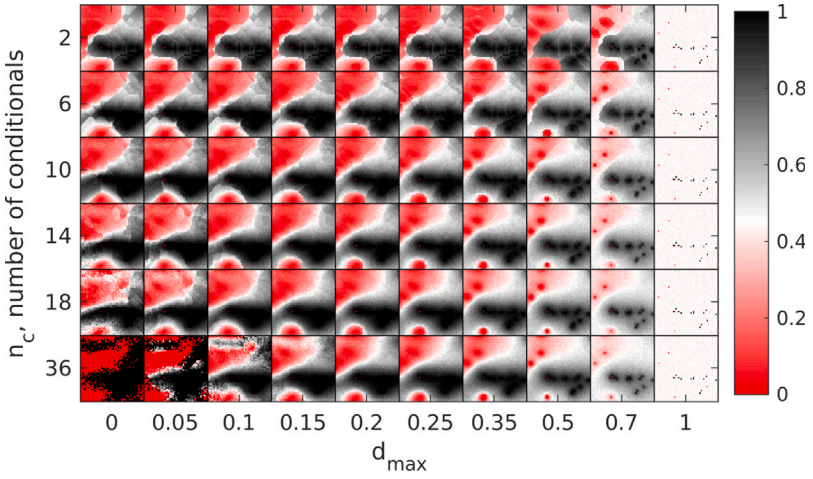

Fig. 9. Probability of buried valley 1-D marginal achieved through MPS estimation, for different values of the two parameters, number of conditional data used n_c and allowed mis-match distance, d_{max}. White is set to the marginal distribution of the training image, and hence corresponds to the least informed state.

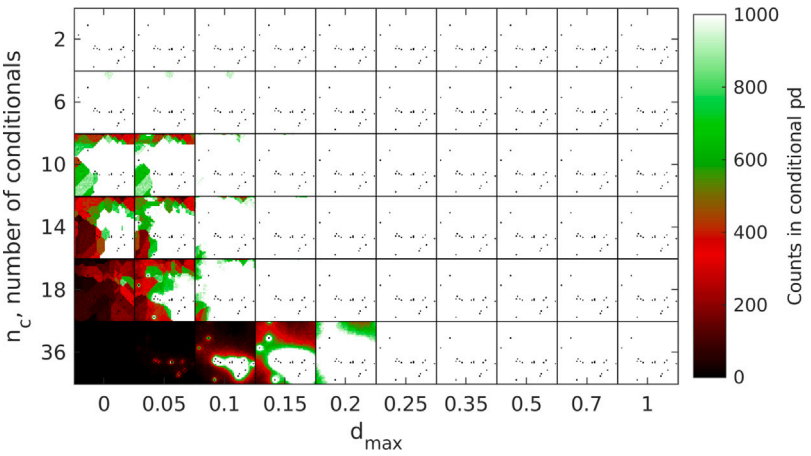

Fig. 10. Sensitivity analysis of MPS estimation, showing the number of conditional events, for the two parameters, number of conditional data used n_c and allowed mis-match distance, $d_{\max}$. Colours on colourbar indicate number of conditional event matches found in the training image, for the data event at the given model parameter being estimated. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

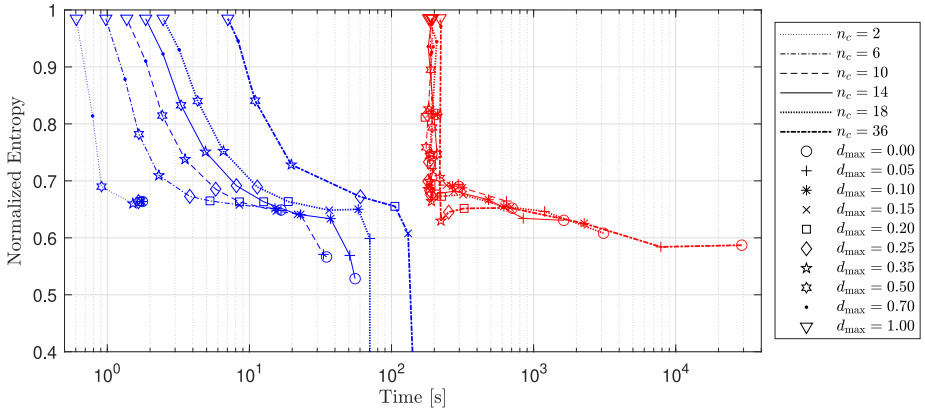

Fig. 11. Averaged pixel-wise entropies resulting from the simulations (red, 1000 realizations) and estimations (blue) in Figs. 8 and 9 respectively, as function of runtime. Lines indicate results obtained from running the algorithms with equal number of conditioning data. Symbols indicate value of $d_{\max}$. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

create the conditional probability density for each model parameter for sequential simulation. Areas with a low number of conditioning events correlate very well with the appearance of some artefacts (compare Fig. 10 with Fig. 9).

These artefacts appear because of inconsistencies between the dataset and the TI. Artefacts seen for $n_c \leq 6$, however, are not due to data-TI inconsistencies, but can instead be attributed to the limited neighbourhood size, that follows from using fewer conditioning data as mentioned above.

So while one can increase n_c and decrease $d_{\max}$ using sequential simulation to maximize information, it seems this can only be done up to a certain threshold level using MPS estimation, as artefacts will appear.

In order to quantify the information-content of the output of the estimation algorithm, the mean parameter-wise entropy is used (Eq. (7)). Fig. 11 shows mean parameter-wise entropy calculated from the estimation, and the mean of realizations, respectively, for the different values of the parameters n_c and $d_{\max}$ used in Figs. 8–10.

It can be seen, that increasing number of conditional data used n_c, and decreasing allowed distance $d_{\max}$ generally decreases entropy until a plateau is found (Fig. 11, red line). Using MPS estimation the same plateau is reached (Fig. 11, blue line), until the problems with obtaining an informed conditional event occurs , leading to the artefacts described above. These artefacts are to be avoided, as they are not representative of the higher-order statistics of the TI. Therefore a method is proposed to obtain the optimal value of $d_{\max}$, minimizing entropy and artefacts.

4.2. Strategy for finding $d_{\max}$

Ideally $d_{\max}$ and n_c should be chosen such that information content is maximized (entropy minimized), but without artefacts due to lack of conditional events. To obtain $d_{\max}$, for a specific choice of n_c, in practice we propose a simple semiautomatic approach in which estimation is performed on a sparse grid, using an decreasing value of $d_{\max}$. This is done until artefacts due to lack of conditional data appears, at which point the optimal value of $d_{\max}$ is located.

A strategy to get the most informed estimation result, that does not exhibit artefacts, could be to start with using a number of conditioning data used (n_c), high enough to capture the longer scale structures. Thus, using a specific n_c, we propose to use an initial relaxed allowed distance ($d_{\max} > 0.5$), that is then successively reduced until artefacts start to appear. Numerically this can be detected by counting the number of conditioning events for each model parameter (see Fig. 10). This is less computationally expensive than starting with $d_{\max} = 0$ and increasing it until artefacts disappear, as proportionally less of the TI needs to be scanned for large allowed distances to achieve the desired amount of conditional events. It also appears that the artefacts are strongly spatially correlated, which would suggest that the exploration of a suitable minimal $d_{\max}$, can be performed on a coarser grid, i.e. on a sparse subset of the model parameters, without losing much accuracy, as the conditional distributions will be based on similar number of counts in the training image. This will be revisited in the following section. Since only a small subset of the model parameters have to

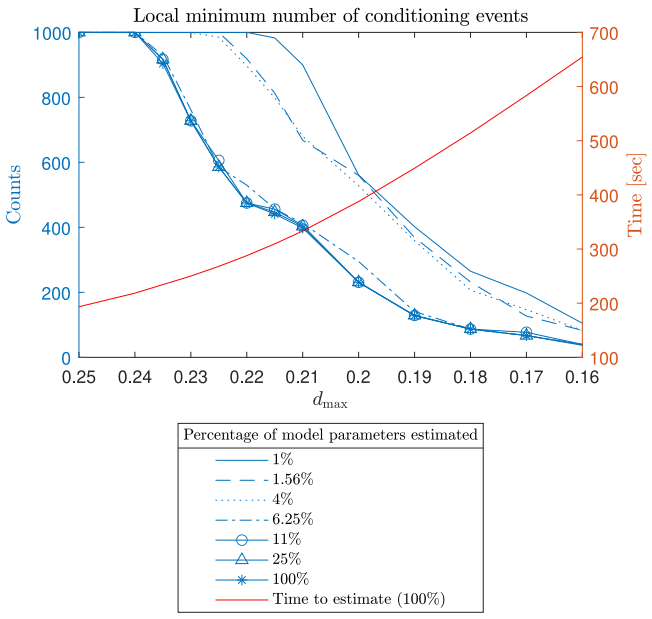

Fig. 12. Finding an appropriate d_{max} for the data-set (given the parameters, $n_c = 18$, and specific TI). When the number of conditioning events found in the TI falls, the conditional pd will be less accurately resolved, eventually leading to visually identifiable artefacts. The spatial correlation of the number of conditioning events for the model parameters, allow this analysis to be done with a smaller, uniformly distributed, subset of the model parameters. In this specific example evaluating more than ≈ 10% of the model parameters does not lead to different conclusions regarding d_{max}. The time taken to estimate rises with a lowering of d_{max}, since the MPS algorithm has to, on average, scan more of the TI to reach the desired amount of conditioning events.

be estimated in order to evaluate if a specific value of d_{max} provides enough conditional events, the added computational cost of finding a suitable d_{max} is only on the order of what the full estimation cost would be. This analysis can be performed for multiple choices of n_c, where the choice of n_c is akin to the choice of n_c for the initial few iterations of any MPS simulation algorithm.

4.3. Estimation in the whole Kasted model

The above strategy will now be used to find an appropriate global d_{max} for the entire region of interest (ROI) for the Kasted data-set. To speed up the process of finding the appropriate d_{max}, a subset of the model parameters can be used in the strategy introduced above, and detailed below:

1. Estimate a sparse subset of the model parameters using a high value of d_{max}.
2. If the minimum number of conditional events for all of the estimated parameters is satisfactory (above some threshold), then decrease d_{max}. Else increase d_{max}.
3. Repeat until a good approximation of the highest achievable d_{max} is found.

Fig. 12 shows the minimum- and averaged, parameter-wise, number of conditional events, found in the TI, based on how the allowed mismatch, d_{max} is decreased. Several levels of sparsity of model parameters are included, showing that a very good approximation of d_{max} can be found, analysing only a very small subset of the model parameters. This accelerates the process of finding an appropriate d_{max} proportionally to the level of sparsity.

Fig. 13 shows the results of MPS estimations performed for the Kasted data-set, using d_{max} of 0.15, 0.2, 0.25 and 0.35 respectively. According to the analysis above, Fig. 13a ($d_{max} = 0.15$) should lead to some artefacts due to non-informed nodes. On the other hand, Fig. 13d ($d_{max} = 0.35$) should represent a case in which too little conditional information is retrieved from the TI which is indeed seen as the probability map is more smooth than, when using $d_{max} < 0.35$.

The above analysis suggested that conditional estimation can provide both low entropy (high information), and little to no artefacts, when $d_{max} = 0.2$–0.25. This is supported by Figs. 13b–c that both show low entropy, but at the same time little to none artefacts.

Fig. 14 shows the corresponding results obtained using sequential simulation. Here it is clear that the conditional statistics obtained for any of the considered values of d_{max} are rather smooth. This is expected from the sensitivity analysis above where it was demonstrated that sequential simulation leads to less information than sequential simulation, for the same considered number of conditional data. The conditional statistics obtained using MPS estimation, Fig. 13, reveal the boundaries of potential subsurface valley structures significantly sharper than those obtained using sequential simulation.

5. Discussion

5.1. Differences in marginal statistics from Estimation vs. Simulation

The parameter-wise marginal statistics from estimation, $f(m_i|\mathbf{m}_c)$, should be exactly the same as those that can be derived from a sufficiently large sample of $f(\mathbf{m}, \mathbf{m}_c)$ obtained using sequential simulation, provided that the neighbourhood is infinite, and there is perfect consistency between the data and the statistics of the training image. This, however, is often not the case, as using infinite neighbourhoods is computationally costly and finding TI's that are perfectly consistent with all data is difficult. Many algorithms use the number of conditioning data used, as the controlling parameter for deciding the neighbourhood size, which during the course of a simulation shrinks the neighbourhood as visited parameters are added to the conditioning data for that realization. In estimation, no such growth in available conditioning data occurs, and hence the neighbourhood size, stays large throughout. This leads to estimation using more of the original conditioning data to derive the conditional pd, than does simulation. I.e. $f(m_i|\mathbf{m}_c)$, (4), can be obtained directly using estimation, or by computing the marginal from a large some sample $f(\mathbf{m}, \mathbf{m}_c)$ obtained using sequential simulation.

The results demonstrate a fundamental difference in how inconsistencies between hard data and the TI affect the estimated posterior statistics. Using MPS estimation areas where the hard data is inconsistent with the TI will lead to apparent extreme low entropy, due to the fact that only few (if any) matches to the conditional data can be found in the TI. Using MPS simulation, the area of inconsistency will instead represent high (maximum) entropy; the opposite result of using MPS estimation. This can have serious implications to the interpretation of the results. An area with low entropy suggest no lack of information. Thus a place where one would potentially collect more data, But if such an area of low entropy is due to inconsistent data, then the reality is not a lack of information, but instead inconsistency. Using MPS estimation such inconsistencies are conveyed clearly to the user in form of low counts of conditional statistics, and noisy statistics. Using MPS simulation, such artefact are hidden from the user, and an area with a consistency problem, is presented as an area with lack of information (high entropy). Therefore we argue that in any case, if the goal is to make use of point-wise posterior statistics, to make use of MPS estimation rather than MPS simulation followed by statistical analysis, in order to correctly identify areas of inconsistency, so appropriate actions can taken.

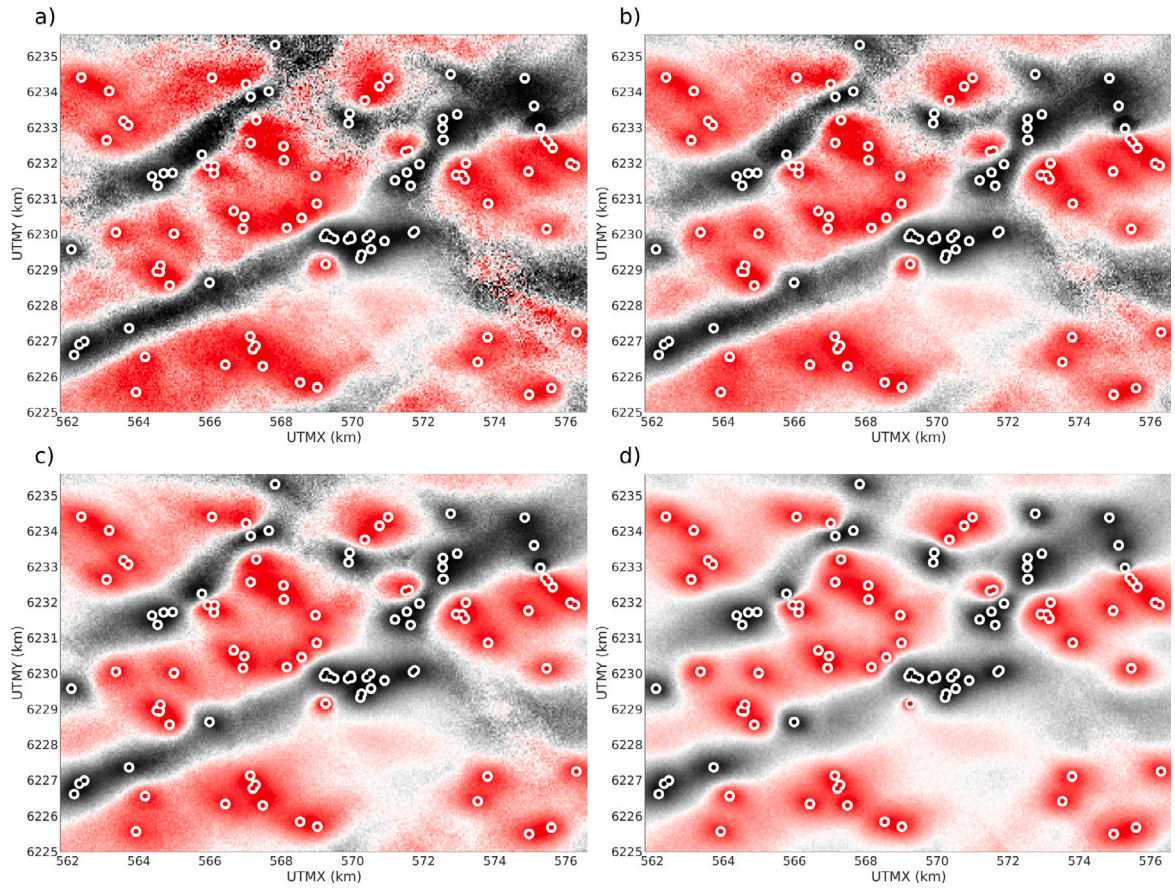

Fig. 13. MPS estimation in Kasted using (a) $d_{\max} = 0.15$, (b) $d_{\max} = 0.20$, (c) $d_{\max} = 0.25$, (d) $d_{\max} = 0.35$, to obtain the parameter-wise probability of a buried valley. Colour-scale same as previous figures, with black indicating buried valley, red indicating no buried valley, and white corresponding to the marginal distribution of the TI. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

5.2. Performance

MPS estimation and MPS simulation (of one realization) can in theory reach speed parity, if they rely on the same underlying MPS algorithm, and there exists conditioning data close to all model parameters. As soon as data is sparse, however, MPS simulation (per single realization) becomes quicker, as the introduction of new conditioning data limits the neighbourhood size needed, thus greatly reducing time spent searching for the data-event in the TI. To get a well resolved 1-D marginal using simulation, however, one would need many realizations, and MPS estimation will, in general, be much faster than sequential simulation. Additionally, the algorithm can be very easily parallelized, as the conditioning data does not change during run-time, as is the case with sequential simulation.

5.3. Applications/Use cases

The obvious application of MPS estimation is in cases where one is mainly interested in 1-D marginal statistics (such as the mean, variance, entropy, mode, quantile, etc.) of an MPS model and not the realizations themselves. In these cases, MPS estimation can estimate the conditional distribution directly, which can be both faster and account for more of the available information, than an analysis of realizations generated through simulation.

As an example use-case of the above mentioned marginal statistics, entropy could be used to effectively determine where there is little to no information in an area of interest; i.e. where in a given model current data is insufficient, or where should the next survey focus. Gulbrandsen et al. (2019) performs just such an analysis using MPS simulation.

Further examples of applications of MPS where one target is 1-D conditional stats can be found in e.g. Madsen et al. (2021) and Knight et al. (2020).

Figs. 10–12 suggest that MPS estimation, and the summary statistics achieved, can be used to quantify consistency between conditional data and TI, i.e. a data set and training image, which require a high $d_{\max}$ for a low value of n_c, to get an acceptable number of samples to construct the conditional pd, will suggest high inconsistency between them. On the other hand, a training image and data set which have plenty of counts in the TI using a $d_{\max}$ of zero for a $n_c = \infty$, would indicate perfect consistency.

6. Conclusion

MPS estimation is an alternative approach to achieving parameter-wise statistics, that would normally require the simulation of many realizations to achieve. MPS estimation is generally significantly faster than using MPS simulation followed by marginalization. Additionally our examples suggest that MPS estimation requires fewer conditioning data to obtain the same degree of conditional information as in MPS simulation. MPS estimation can potentially be used as a fast method to identify inconsistencies between a training image and conditioning data before running costly MPS simulation. Finally, MPS estimation is straight forward to implement in existing sequential simulation based algorithms based on GENESIM and SNESIM.

Computer code availability

Name: Estimation branch of MPSlib
Developer: T.M. Hansen (2nd author)

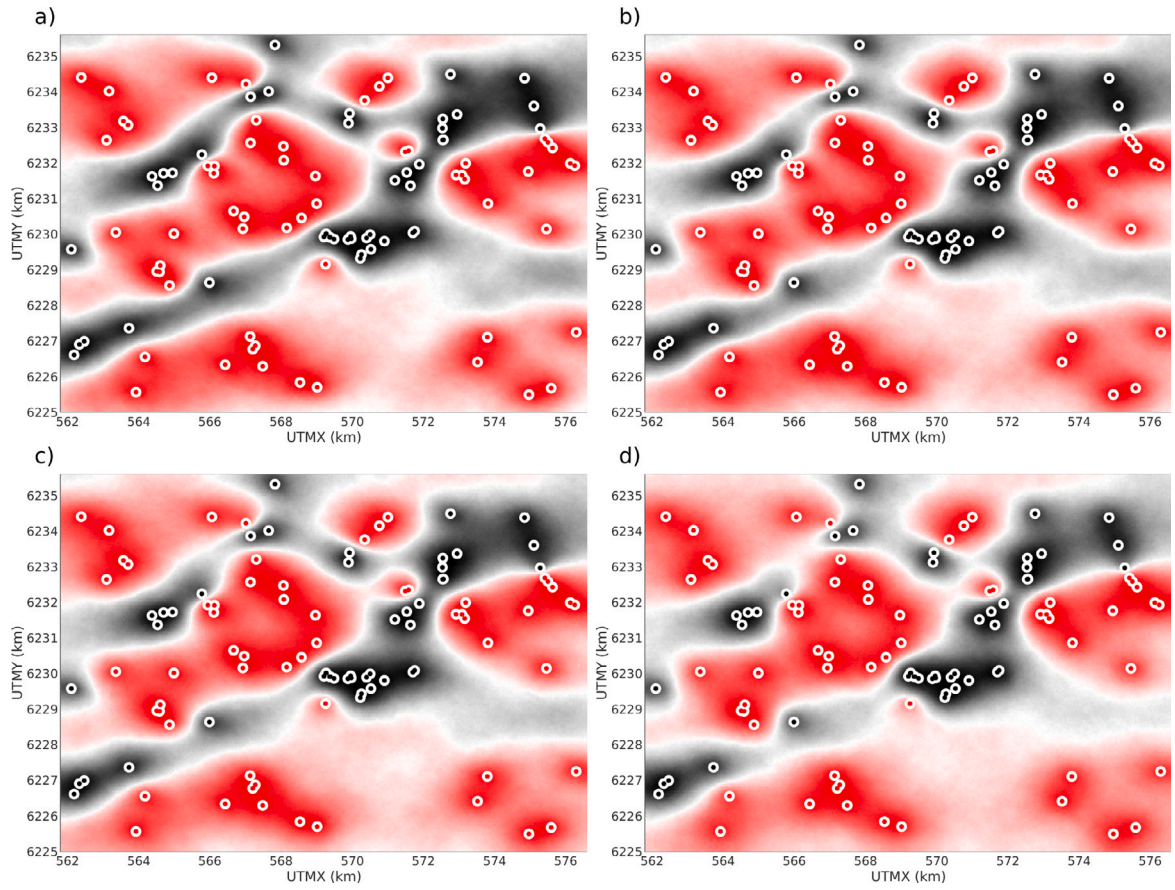

Fig. 14. Sequential simulation in Kasted using (a) $d_{max} = 0.15$, (b) $d_{max} = 0.20$, (c) $d_{max} = 0.25$, (d) $d_{max} = 0.35$, to obtain the parameter-wise probability of a buried valley. Colour-scale same as previous figures, with black indicating buried valley, red indicating no buried valley, and white corresponding to the marginal distribution of the TI. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

First available: 2020

No hardware requirement.

Software required: GNU C++ compiler, mingw-w64 (Win64)

Program language: C++

Source-code available at: https://github.com/ergosimulation/mpslib/.

CRediT authorship contribution statement

Óli D. Jóhannsson: Conceptualization, Methodology, Software, Analysis, Investigation, Writing – review & editing. **Thomas Mejer Hansen:** Conceptualization, Methodology, Software, Analysis, Investigation, Writing – review & editing, Supervision, Project administration, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This work is related to grant 7017-00160B from the Independent Research Fund of Denmark.

References

Barfod, A.A., Vilhelmsen, T.N., Jørgensen, F., Christiansen, A.V., Høyer, A.-S., Straubhaar, J., Møller, I., 2018. Contributions to uncertainty related to hydrostratigraphic modeling using multiple-point statistics. Hydrol. Earth Syst. Sci. 22 (10), 5485–5508.

Deutsch, C.V., Journel, A.G., 1992. GSLIB: Geostatistical Software Library and User's Guide. Oxford University Press.

Devroye, L., Sample-based non-uniform random variate generation, In: Proceedings of the 18th Conference on Winter Simulation, 1986, pp. 260–265.

Exizidou, D., 2014. Geostatistical Modeling of Buried Valleys (Master's thesis). Technical University of Denmark, Kongens Lyngby, Denmark.

Goovaerts, P., 1997. Geostatistics for Natural Resources Evaluation. In: Applied Geostatistics, Oxford University Press, p. 496.

Guardiano, F.B., Srivastava, R.M., 1993. Multivariate geostatistics: beyond bivariate moments. pp. 133–144.

Gulbrandsen, M.L., Hansen, T.M., Jensen, N.P., 2019. Where do we lack information? MPS realizations can tell you where to drill!. In: 32nd Symposium on the Application of Geophysics to Engineering and Environmental Problems, SAGEEP 2019. Environmental and Engineering Geophysical Society (EEGS).

Hansen, T.M., 2020. Entropy and information content of geostatistical models. Math. Geosci. 1–22.

Hansen, T.M., Mosegaard, K., 2008. VISIM: Sequential simulation for linear inverse problems. Comput. Geosci. 34 (1), 53–76.

Hansen, T.M., Vu, L.T., Bach, T., 2016. MPSLIB: A C++ class for sequential simulation of multiple-point statistical models. Softw. X.

Høyer, A.-S., Jørgensen, F., Sandersen, P., Viezzoli, A., Møller, I., 2015. 3D geological modelling of a complex buried-valley network delineated from borehole and AEM data. J. Appl. Geophys. 122, 94–102.

Jørgensen, F., Sandersen, P.B., 2006. Buried and open tunnel valleys in Denmark—erosion beneath multiple ice sheets. Quat. Sci. Rev. 25 (11–12), 1339–1363.

Journel, A., Alabert, F., 1989. Non-Gaussian data expansion in the earth sciences. Terra Nova 1 (2), 123–134.

Journel, A.G., Huijbregts, C.J., 1978. Mining Geostatistics, Vol. 600. Academic press London.

Knight, R., Auken, E., Buck, C., Cannia, J., Dewar, N., Gosselin, P., Halkjær, M., Jensen, N., Kang, S., Martin, C., Bjergsted Pedersen, J., Zdeba, D., 2020. The Stanford Groundwater Architecture Project: Utilizing Advanced Geophysical and Computational Methods for the Development of Hydrogeologic Conceptual Models. Groundwater Resources Association.

Krige, D.G., 1951. A statistical approach to some basic mine valuation problems on the witwatersrand. J. Southern Afr. Instit. Mining Metall. 52 (6), 119–139.

Madsen, R.B., Kim, H., Kallesøe, A.J., Sandersen, P.B.E., Vilhelmsen, T.N., Hansen, T.M., Christiansen, A.V., Møller, I., Hansen, B., 2021. 3d multiple-point geostatistical simulation of joint subsurface redox and geological architectures. Hydrology and Earth System Sciences (ISSN: 1607-7938) 25 (5), 2759–2787. http://dx.doi.org/10.5194/hess-25-2759-2021, https://hess.copernicus.org/articles/25/2759/2021/.

Mariethoz, P., Caers, P., 2014. Multiple-Point Geostatistics: Stochastic Modeling with Training Images. Wiley.

Mariethoz, G., Kelly, B.F., 2011. Modeling complex geological structures with elementary training images and transform-invariant distances. Water Resour. Res. 47 (7).

Mariethoz, G., Renard, P., Straubhaar, J., 2010. The direct sampling method to perform multiple-point geostatistical simulations. Water Resour. Res. 46 (11), 1–14.

Piotrowski, J.A., Hermanowski, P., Piechota, A.M., 2009. Meltwater discharge through the subglacial bed and its land-forming consequences from numerical experiments in the polish lowland during the last glaciation. Earth Surface Proces. Landforms 34 (4), 481–492.

Sandersen, P.B., Jørgensen, F., 2003. Buried quaternary valleys in western Denmark—occurrence and inferred implications for groundwater resources and vulnerability. J. Appl. Geophys. 53 (4), 229–248.

Shannon, C.E., 2001. A mathematical theory of communication. ACM SIGMOBILE Mob. Comput. Commun. Rev. 5 (1), 3–55.

Straubhaar, J., Renard, P., Mariethoz, G., Froidevaux, R., Besson, O., 2011. An improved parallel multiple-point algorithm using a list approach. Math. Geosci. 43 (3), 305–328.

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

Tahmasebi, P., 2018. Multiple point statistics: a review. In: Handbook of Mathematical Geosciences. Springer, Cham, pp. 613–643.

Vilhelmsen, T.N., Auken, E., Christiansen, A.V., Barfod, A.S., Marker, P.A., Bauer-Gottwein, P., 2019. Combining clustering methods with MPS to estimate structural uncertainty for hydrological models. Front. Earth Sci. 7, 181.