Computers & Geosciences 52 (2013) 470–480

Contents lists available at SciVerse ScienceDirect

Computers & Geosciences

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

SIPPI: A Matlab toolbox for sampling the solution to inverse problems with complex prior information Part 1—Methodology

Thomas Mejer Hansen [a,*], Knud Skou Cordua [a], Majken Caroline Looms [b], Klaus Mosegaard [a]

[a] Technical University of Denmark, Center for Energy Resources Engineering, DTU Informatics, Asmussens Alle, Building 305, DK-2800 Lyngby, Denmark
[b] University of Copenhagen, Department of Geography og Geology, Øster Voldgade 10, DK-1350 København K, Denmark

ARTICLE INFO

Article history:
Received 22 June 2012
Received in revised form
7 September 2012
Accepted 10 September 2012
Available online 13 October 2012

Keywords:
Inversion
Nonlinear
Sampling
A priori
A posteriori

ABSTRACT

From a probabilistic point-of-view, the solution to an inverse problem can be seen as a combination of independent states of information quantified by probability density functions. Typically, these states of information are provided by a set of observed data and some a priori information on the solution. The combined states of information (i.e. the solution to the inverse problem) is a probability density function typically referred to as the a posteriori probability density function. We present a generic toolbox for Matlab and Gnu Octave called SIPPI that implements a number of methods for solving such probabilistically formulated inverse problems by sampling the a posteriori probability density function. In order to describe the a priori probability density function, we consider both simple Gaussian models and more complex (and realistic) a priori models based on higher order statistics. These a priori models can be used with both linear and non-linear inverse problems. For linear inverse Gaussian problems we make use of least-squares and kriging-based methods to describe the a posteriori probability density function directly. For general non-linear (i.e. non-Gaussian) inverse problems, we make use of the extended Metropolis algorithm to sample the a posteriori probability density function. Together with the extended Metropolis algorithm, we use sequential Gibbs sampling that allow computationally efficient sampling of complex a priori models. The toolbox can be applied to any inverse problem as long as a way of solving the forward problem is provided. Here we demonstrate the methods and algorithms available in SIPPI. An application of SIPPI, to a tomographic cross borehole inverse problems, is presented in a second part of this paper.

1. Introduction

Inverse problems are abundant in almost any type of scientific research field. An inverse problem occurs when a set of unknown parameters, that describe a physical system, pixel values of an image or some mathematical expression, have to be inferred based on indirect observations of these parameters. Examples of inverse problems are image deblurring, tomographic reconstruction, solutions to certain differential equations, or reconstructing the earth's interior based on surface observations. There are several ways to solve an inverse problem. In a probabilistic formulation the inverse problem can be seen as a way of combining information: Given knowledge about the system (differential equation, physical law, or blurring mechanisms), and a set of observations (signal intensities, pixel values, gravity field), and some prior expectations about the parameters, the goal is to quantify how probable a number of possible scenarios are of explaining the observations and the prior information. A successful probabilistic inversion will, in principle, locate all solutions to the problem and assign a probability to each scenario given the information at hand.

In this paper we present a Matlab[1] toolbox (SIPPI), compatible with Gnu Octave,[2] that can be used to solve inverse problems in a probabilistic formulation. In this formulation the solution to the inverse problem is a probability density function (pdf) referred to as the *a posteriori* pdf that describe all information available about a system. While the toolbox is generally applicable to inverse problems, it has been designed specifically for geophysical inverse problems, where the model parameters typically describe a 1D–3D space, such as for example the subsurface of the earth.

Initially we lay out the theory of probabilistically formulated inverse problems. Then we show how so-called *a priori* information

* Corresponding author. Tel.: +45 45253086; fax: +45 45882673.
E-mail addresses: tmeha@imm.dtu.dk (T. Mejer Hansen), kcor@imm.dtu.dk (K. Skou Cordua), mcl@geol.ku.dk (M. Caroline Looms), kmos@imm.dtu.dk (K. Mosegaard).

[1] http://mathworks.com/
[2] http://www.gnu.org/software/octave

0098-3004/$ - see front matter © 2012 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.cageo.2012.09.004

about the model parameters, and uncertainty of data observations can be specified. Finally we show how realizations of the a posteriori pdf can be generated using least squares based methods, and sampling techniques such as rejection sampling and Metropolis sampling.

In the second part of this paper we demonstrate the application of SIPPI to a cross borehole traveltime tomographic inverse problem (Hansen et al., this issue).

2. Probabilistic inverse problem theory

Consider some data, **d**, which are indirect measurements of some model parameters, **m**, describing a system, such as for example the subsurface of the Earth. Let **d** and **m** be related through the function g:

$$\mathbf{d} = g(\mathbf{m}). \tag{1}$$

The above equation, referred to as the forward problem, can be solved with various degrees of accuracy for a number of physical problems.

Inversion of geophysical data amounts to infer information about the model parameters, **m**, given some data, **d**, the forward relation between model parameters and data, g, and a priori existing knowledge about the model parameters. Such an inverse problem can be solved in a variety of ways. In this paper we will deal with the general probabilistic formulation of inverse problems. Note that many types of deterministic inversion methods can be formulated as special cases of the probabilistic inverse theory as we consider here.

Tarantola and Valette (1982b) formulate a probabilistic approach for solving inverse problems where all available states of information are described by pdfs. The solution to the inverse problem is the pdf that combines known states of information. In a typical inverse problem the states of information can be described by the *a priori* pdf and the *likelihood* function. The a priori pdf, $\rho M(\mathbf{m})$, describes prior knowledge about the model parameters. The likelihood function, $L(\mathbf{m})$, is a probabilistic measure of how well a given model **m** explains the observed data.

The general solution to such a probabilistically formulated inverse problem is the *a posteriori* pdf, which is proportional to the product of the a priori pdf and the likelihood function:

$$\sigma_{\mathrm{M}}(\mathbf{m}) = k\, \rho_{\mathrm{M}}(\mathbf{m}) L(\mathbf{m}) \tag{2}$$

where k is a normalization constant and the likelihood is given by

$$L(\mathbf{m}) = \int_{\mathcal{D}} d\mathbf{d} \frac{\rho_{\mathrm{D}}(g(\mathbf{m}))\theta(\mathbf{d}|\mathbf{m})}{\mu_{\mathrm{D}}(\mathbf{d})} \tag{3}$$

$\rho_{\mathrm{D}}(\mathbf{d})$ describes *measurement uncertainties*, typically related to uncertainties in the instrument that records the data. $\theta(\mathbf{d}|\mathbf{m})$ describes the *modelization error*, i.e. the error caused by using an imperfect forward model g or an imperfect parameterization. $\mu_{\mathrm{D}}(\mathbf{d})$ describes the homogeneous state of information that ensures that the parameterization is invariant to changes in the coordinate system. For the reminder of the text we shall assume that $\mu_{\mathrm{D}}(\mathbf{d})$ can be approximated by a constant. For more details on the homogeneous pdf, see e.g. Mosegaard and Tarantola (2002).

The a posteriori pdf describes the distribution of models consistent with the combined states of information given by the a priori model and the data.

The probabilistic formulation of inverse problems allows utilization of the movie strategy advocated by Tarantola (2005), who suggest to visualize and compare a sample from the a priori pdf and the a posteriori pdf, respectively, as movies. The 'rior movie' will make it apparent what prior choices have been made. The difference between the prior and the posterior movie will emphasize the effect of using data.

2.1. The linear inverse Gaussian problem

Consider a linear forward problem, where the data **d** is linearly related to the model parameters **m** using the linear operator **G**, such that $\mathbf{d} = \mathbf{Gm}$. Let $\mathcal{N}(\mathbf{a},\mathbf{A})$ refer to a Gaussian distribution with mean **a** and covariance **A**. If in addition both the a priori model $\mathcal{N}(\mathbf{m}_0,\mathbf{C}_{\mathrm{M}})$, the noise model $\mathcal{N}(0,\mathbf{C}_d)$ and the modelization error $\mathcal{N}(0,\mathbf{C}_T)$ can be described by a Gaussian pdf, then the a posteriori pdf (Eq. (2)) can be described analytically by a Gaussian pdf, $\mathcal{N}(\tilde{\mathbf{m}},\tilde{\mathbf{C}}_{\mathrm{M}})$ (Tarantola and Valette, 1982a):

$$\tilde{\mathbf{m}} = \mathbf{m}_0 + \mathbf{C}_{\mathrm{M}}\mathbf{G}^t(\mathbf{G}\mathbf{C}_{\mathrm{M}}\mathbf{G}' + \mathbf{C}_{\mathrm{D}})^{-1}(\mathbf{d}_0 - \mathbf{G}\mathbf{m}_0) \tag{4}$$

$$\tilde{\mathbf{C}}_{\mathrm{M}} = \mathbf{C}_{\mathrm{M}} - \mathbf{C}_{\mathrm{M}}\mathbf{G}^t(\mathbf{G}\mathbf{C}_{\mathrm{M}}\mathbf{G}' + \mathbf{C}_{\mathrm{D}})^{-1}\mathbf{G}\mathbf{C}_{\mathrm{M}} \tag{5}$$

Note that Gaussian measurement errors and modelization errors combine through addition of the covariance operators, such that the combined covariance model is given by $\mathbf{C}_{\mathrm{D}} = \mathbf{C}_d + \mathbf{C}_T$. This allows accounting of Gaussian modelization errors directly as given in Eqs. (4) and (5) (Tarantola, 2005).

If $\tilde{\mathbf{m}}$ and $\tilde{\mathbf{C}}_{\mathrm{M}}$ are available from Eqs. (4) and (5), then samples from the a posteriori pdf can be generated using e.g. Cholesky decomposition of the a posteriori covariance model, Eq. (5) in Le Ravalec et al. (2000).

Sampling the a posteriori pdf of a linear inverse Gaussian problem can also be performed using sequential Gaussian simulation without the need for explicitly computing $\tilde{\mathbf{m}}$ and $\tilde{\mathbf{C}}_{\mathrm{M}}$ (Hansen et al., 2006). Hansen and Mosegaard (2008) extend this approach to work with direct sequential simulation. This allows a non-Gaussian a priori distribution of model parameters.

An alternative approach is to use kriging through error simulation (Journel and Huijbregts, 1978, p. 495), in a co-kriging formulation as proposed by Gloaguen et al. (2005a, 2005b) This approach may be faster than the methods based on sequential simulation, but is only valid for strictly Gaussian a priori models.

The above-mentioned methods rely on the fact that in a linear formulation, data can be seen as weighed averages of the model parameters. While not specifically making the link to inverse problems, such ideas has also been explored by Journel (1999) and Gómez-Hernández et al. (2005).

2.2. The non-linear Inverse problem

The linear and Gaussian assumptions considered above are convenient as they lead to computationally efficient algorithms. However, in reality the inverse problem is typically non-linear and the Gaussian assumption not valid. This may lead to severe artifacts in the inversion if the least-squares based approaches, as described above, are used. Instead one can use sampling techniques to sample the a posteriori pdf.

Rejection sampling: Perhaps the simplest method to sample the a posteriori pdf is the rejection sampler that can be implemented as follows:

1. Propose a model candidate from the a priori pdf, $\mathbf{m}_{pro}$.
2. Compute $L(\mathbf{m}_{pro})$.
3. Accept the proposed model as a realization of the a posteriori pdf with probability
$$P_{acc} = L(\mathbf{m}_{pro})/L_{max} \tag{6}$$

where L_{max} is the maximum value the likelihood function can obtain. Typically the value of L_{max} is not known and must be set to 1.

The only requirement for using the method is that one must be able to generate independent realizations of the a priori pdf and compute the corresponding likelihood. The collection of models accepted by the rejection sampling algorithm will be a sample of the a posteriori pdf. The main problem with the rejection sampler is that it is computationally very inefficient for anything but very low dimensional problems.

The extended Metropolis sampler: Mosegaard and Tarantola (1995) propose an extended version of the Metropolis algorithm (Metropolis et al., 1953; Hastings, 1970) that allows sampling the a posteriori pdf of an inverse problem with, in principle, arbitrary complex a priori information as given by Eq. (2). Using the classical Metropolis algorithm one must be able to evaluate the a posteriori probability $\sigma_M(\mathbf{m})$ and, hence, typically also the a priori probability, in order to evaluate Eq. (2).

The extended Metropolis algorithm differs from the classical Metropolis algorithm in that one neither need to evaluate the a posteriori probability $\sigma_M(\mathbf{m})$, nor the a priori probability $\rho_M(\mathbf{m})$ of a given model $\mathbf{m}$. If only an algorithm is present that can sample the a priori pdf and a method exist for evaluating the likelihood, $\rho_D(g(\mathbf{m}))$, then the extended Metropolis algorithm will sample the a posteriori pdf.

The extended Metropolis algorithm is a Markov Chain Monte Carlo method and can be implemented as a random walk in the space of a priori acceptable models as follows. If initially a realization of the a priori pdf is generated as $\mathbf{m}_{cur}$, and the associated likelihood $L(\mathbf{m}_{cur})$ is evaluated using Eq. (3), then the following algorithm will sample the a posteriori pdf:

1. In the vicinity of $\mathbf{m}_{cur}$, propose a new model candidate, $\mathbf{m}_{pro}$, consistent with the a priori model.
2. Compute $L(\mathbf{m}_{pro})$.
3. Accept the proposed model with probability $P_{acc} = \min([1, L(\mathbf{m}_{pro})/L(\mathbf{m}_{cur})])$.
4. If the proposed model is accepted, then the transition from $\mathbf{m}_{cur}$ to $\mathbf{m}_{pro}$ is accepted, and the proposed model becomes the current model, $\mathbf{m}_{cur} = \mathbf{m}_{pro}$. Otherwise, the random walker stays a location $\mathbf{m}_{cur}$ and $\mathbf{m}_{cur}$ counts again.

There are only two requirements for running the extended Metropolis algorithm: (1) One must be able to evaluate the likelihood function, Eq. (3). This is most often trivial, even if it may be computationally demanding, as it requires one to solve the forward problem and evaluate the corresponding data fit given the noise model. (2) One must be able to sample the a priori pdf such that aperiodicity and irreducibility is ensured (Mosegaard and Sambridge, 2002). In addition, it is preferable to be able to control the exploratory nature (often referred to as the step length) of the sampling algorithm, i.e. step 1 in the above algorithm, which is closely linked to the computational efficiency. See Mosegaard and Tarantola (1995) for details on the extended Metropolis algorithm.

The sequential Gibbs sampling algorithm provides such a general way to sample complex a priori models, with arbitrary step length ensuring aperiodicity and irreducibility (Hansen et al., 2012). Sequential Gibbs sampling can be used with any pdf that can be sampled using sequential simulation, which is the case for most of the statistical models developed in the geostatistical community over the last decades. The resampling strategy inherent in the sequential Gibbs sampler was initially proposed by Hansen et al. (2008), and subsequently Irving and Singha (2010) and Mariethoz et al. (2010) proposed similar methods. Hansen et al. (2012) demonstrate how the method is similar to an application of the Gibbs sampler and show that the method leads to a way of sampling the a priori pdf where aperiodicity and irreducibility is ensured.

3. SIPPI

SIPPI is a Matlab toolbox (SIPPI), compatible with Gnu Octave, that can be used to solve inverse problems in the formulation given by Eqs. (2) and (3) by allowing *S*ampling the solution to *I*nverse *P*roblems with complex A *P*riori *I*nformation.

In order to solve a probabilistic framed inverse problem as presented previously, one needs (at least) three ingredients: (1) a choice of an a priori model, (2) a choice of how to solve the forward problem, and (3) a choice of a noise model that describes the uncertainty of the observed data and the modelization error. Once these choices have been made one can solve the inverse problem using any of the applicable inversion methods.

SIPPI provides a generic approach to define the a priori model and the noise model in the form of the two data structures `prior` and `data`.

3.1. *The a priori model*

All information about the a priori model are defined in the Matlab structure called `prior`, which can specify any number of a priori type of models. For example an a priori choice of a 2D Gaussian velocity field can be specified in `prior{1}` and a 1D parameter describing a bias correction can be specified in `prior{2}`. Once the `prior` has been defined, a realization of the corresponding a priori pdf can be generated by calling

```
m=sippi_prior(prior);
```

`m` is a Matlab structure of the same size as `prior`. If three types of a priori models have been defined in `prior{1}`, `prior{2}`, and `prior{3}`, then the corresponding realizations will be stored in `m{1}`, `m{2}`, and `m{3}`. Considering the example above, `m{1}` will hold a realization of a 2D a priori model, while `m{2}` will hold a realization of a 1D a priori model. For the remainder of the text the index `im` will point to a specific number of a priori model, `prior{im}`.

A number of different types of a priori models can be selected using a `type` field to the `prior` data structure. The following four types of a priori models are available as part of SIPPI:

```
im=1;
prior{im}.type='GAUSSIAN';
prior{im}.type='FFTMA';
prior{im}.type='VISIM';
prior{im}.type='SNESIM';
```

Generalized Gaussian: `prior{im}.type=GAUSSIAN'` defines a 1D generalized Gaussian distribution

$$f_{gg}(m_0,\sigma,p) = \frac{p^{1-1/p}}{2\sigma\Gamma(1/p)} \exp\left(-\frac{1}{p}\frac{|m-m_0|^p}{\sigma^p}\right) \quad (7)$$

where p is the norm and σ is the variance. f_{gg} is symmetric around m_0, the a priori mean value. In the limit of $p \to \infty$ f_{gg} will define a uniform distribution. The following code defines a 1D Gaussian distribution with mean 10 and standard deviation 2:

```
im=1;
prior{im}.type='GAUSSIAN';
prior{im}.m0=10;
prior{im}.std=2;
```

If not set, the `norm` is by default set to 2. The following code defines a 1D close to uniform distribution in the interval [8,12]:

```
im=1;
prior{im}.type='GAUSSIAN';
prior{im}.m0=10;
```

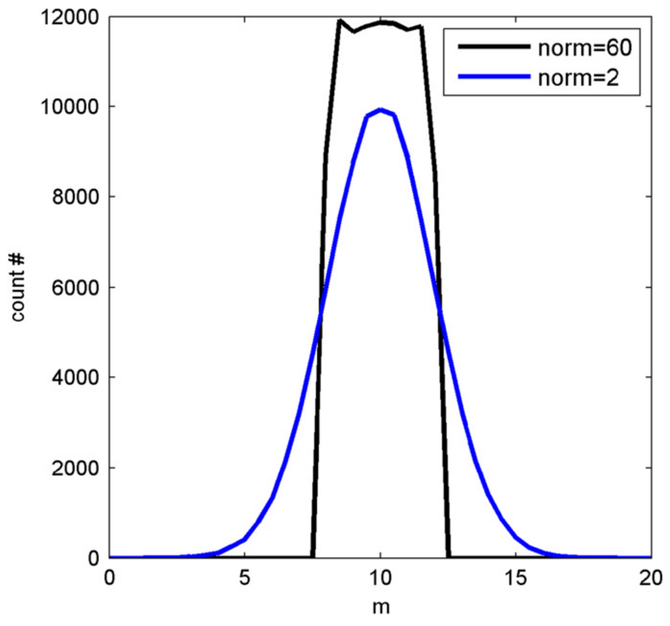

Fig. 1. Histogram of 100,000 unconditional realizations from a generalized Gaussian, GAUSSIAN type prior model with norm 60 and 2.

```
prior{im}.std=2;
prior{im}.norm=60;
```

A histogram of a sample of size 100,000 of these two 1D prior models is shown in Fig. 1.

The FFTMA, VISIM and SNESIM type priors all describe a 1D to 3D a priori model defined on a Cartesian grid, which is defined as (for a 3D case)

```
im=1;
prior{im}.prior.x=[0:1:10]; % X array
prior{im}.prior.y=[0:1:20]; % Y array
prior{im}.prior.z=[0:1:30] ; % Z array
```

For a 1D prior only `prior{im}.prior.x` needs to be defined, and for a 2D prior `prior{im}.prior.x` and `prior{im}.prior.y` need to be defined.

Both the FFTMA and VISIM type a priori models describe a multivariate Gaussian a priori pdf, which requires the specification of an a priori mean and covariance model. The a priori mean m0 can be either a scalar, indicating a constant a priori mean model, or a matrix of the size of the a priori model, allowing for a varying a priori mean model. The model of spatial variability is defined by a, possibly anisotropic, covariance model (equivalent to a semivariogram model) given by the Cm (or equivalent the Va) field. The specification of the covariance model uses the same notation as used in Pebesma and Wesseling (1998). For example a multivariate Gaussian model defined by a 2D Spherical type covariance model with sill (or variance) 1, a maximum correlation length of 10 in the direction west to east (i.e. horizontal), and a perpendicular range (i.e. vertical) of 2.5 (hence an anisotropy factor of 0.25) and a mean of 10, is given by

```
prior{im}.m0=10;
prior{im}.Cm='1 Sph(10,90,0.25)';
```

FFT moving average: `prior{im}.type='FFTMA'` defines a spatially correlated multivariate Gaussian a priori model where a priori realizations are generated using the FFT Moving Average generator (FFTMA) (Le Ravalec et al., 2000). The FFTMA algorithm is very efficient for generating unconditional realizations from a multivariate Gaussian model. In addition it also allows separation of the random component field and the structural parameters that define spatial correlation. We will discuss the use of this feature in more details later.

A 2D FFTMA type a priori model, on a 200×100 grid, can for example be given by

```
im=1;
prior{im}.type='FFTMA';
prior{im}.prior.x=[0:.1:10]; % X array
prior{im}.prior.y=[0:.1:20]; % Y array
prior{im}.m0=10;
prior{im}.Va='1 Sph(10,90,.25)';
```

Fig. 2a shows a set of five realizations from this choice of a priori model.

VISIM : `prior{im}.type='VISIM'` defines a spatially correlated multivariate Gaussian a priori model where a priori realizations are generated using the VISIM algorithm (Hansen and Mosegaard, 2008). VISIM can run using sequential Gaussian simulation, in which case the model parameters are assumed to be normally distributed. It can also run using direct sequential simulation, which allows a (non-Gaussian) target distribution to be set that describes the a priori distribution of the model parameters, while at the same time ensuring that the a priori chosen mean and covariance will be honored.

An a priori model similar to the one described above for the FFTMA type prior, but with an a priori assumption of a bimodal distribution of model parameters can be given as

```
im=1;
prior{im}.type='VISIM';
prior{im}.prior.x=[0:1:10]; % X array
prior{im}.prior.y=[0:1:20]; % Y array
prior{im}.m0=10;
prior{im}.Va='1 Sph(10,90,.25)';
% target distribution
N=10,000;
prob_chan=0.5;
d1=randn(1,ceil(N*(1-prob_chan)))*.5+8.5;
d2=randn(1,ceil(N*(prob_chan)))*.5+11.5;
d_target=[d1(:);d2(:)];
prior{im}.target=d_target;
```

Fig. 3 shows a set of five realizations from this VISIM type of a priori model (a) without a specification of a target distribution and (b) using a target distribution. Once `[m,prior]=sippi_prior(prior)` has been called once, a data structure will be available as `prior{im}.V`, which allows access to all options available for running the VISIM algorithm. See Hansen and Mosegaard (2008) for more details on VISIM.

The FFTMA and VISIM type prior models only allow reproducing the first two moments of the distribution describing the spatial variability, the mean and the covariance (i.e. Gaussian variability between sets of two data points). Maximum entropy is implicitly assumed in higher order moments (Journel and Zhang, 2006). This is the reason why geological structures such as for example meandering channels cannot be reproduced by Gaussian statistics. To achieve this one can make use of statistical models based on higher order moments.

SNESIM : `prior{im}.type='SNESIM'` defines an a priori model based on a higher order statistical moments (a multiple point statistical model) describing spatial variability as inferred from a training image.

There are several methods that allow sampling from an a priori model defined by multiple point statistics. Here, we use the

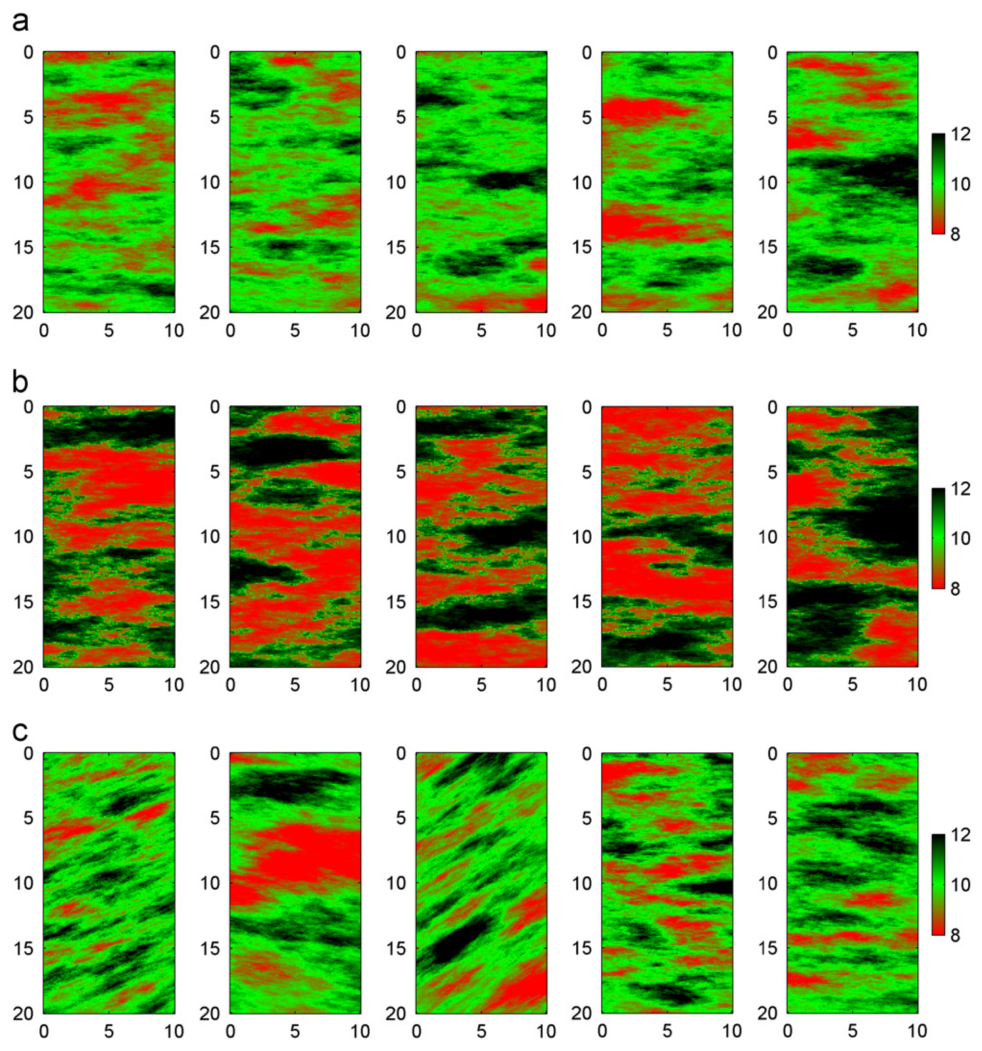

Fig. 2. Unconditional realizations from a FFTMA type priori model with (a) Gaussian distribution, (b) target distribution, and (c) random structural parameters (range and rotation).

SNESIM algorithm, originally developed by Strebelle (2000, 2002), and we make use of the implementation available in the SGeMS software package (Remy et al., 2008). It works by initially extracting a multiple point based statistical model from a training image. Then sequential simulation is used to generate realizations of this statistical model.

Optionally the `scaling` and `rotation` field can be specified. `prior{im}.scaling=2` scales the axis of the training image such that spatial structures appears twice as large. `prior{im}.rotation=45` rotates the training image 45° clockwise.

A 2D SNESIM type prior with the training image 'channels.ti' (Fig. 4) rotated 30° and scaled by a factor of 0.75, with two categories ('0' and '1'), and where the first category '0' reflect a model parameter value of 8, and the second category '1' reflect a value of 12, is given by

```
im=1;
prior{im}.type='SNESIM';
prior{im}.x=[0:.1:10];
prior{im}.y=[0:.1:20];
prior{im}.ti='channels.ti';
prior{im}.index_values=[0 1]; % optional
prior{im}.m_values=[8 12]; % optional
prior{im}.scaling=.75; % optional
prior{im}.rotation=30; % optional
```

Fig. 5 shows a set of five realizations from this choice of a priori model. Once `[m,prior]=sippi_prior(prior)` has been called, a data structure will be available as `prior{im}.S` which allow access to all options available for running the SNESIM algorithm as implemented in SGeMS. See Remy et al. (2008) for more details on setting up the SNESIM algorithm.

Distribution transform: A normal score transform can be defined for any of the Gaussian based a priori models, that allow the transformation of the normally distributed model parameters to any desired distribution, see e.g. Goovaerts (1997). It requires only that the user defines the 'target' distribution, in the form of a sample of the target distribution in the `d_target` field. For example a bimodal distribution with increased probability of values around 8.5 and 11.5, can be given by

```
N=10,000;
prob_chan=0.5;
d1=randn(1,ceil(N*(1-prob_chan)))*.5+8.5;
d2=randn(1,ceil(N*(prob_chan)))*.5+11.5;
d_target=[d1(:);d2(:)];
prior{im}.d_target=d_target;
```

Note that the number N here reflects the size of the sample generated and used to describe the target distribution in the `d_target` field, and can be chosen arbitrarily large. The larger the sample, the better the accuracy of reflecting a specific

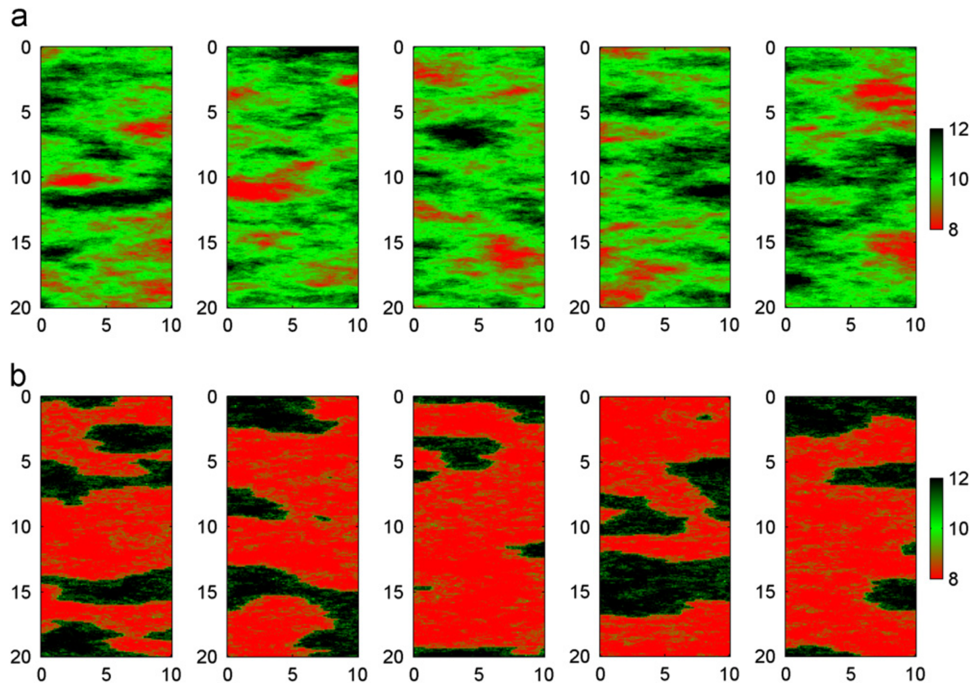

Fig. 3. Unconditional realizations from a VISIM type a priori model with (a) Gaussian distribution and (b) target distribution.

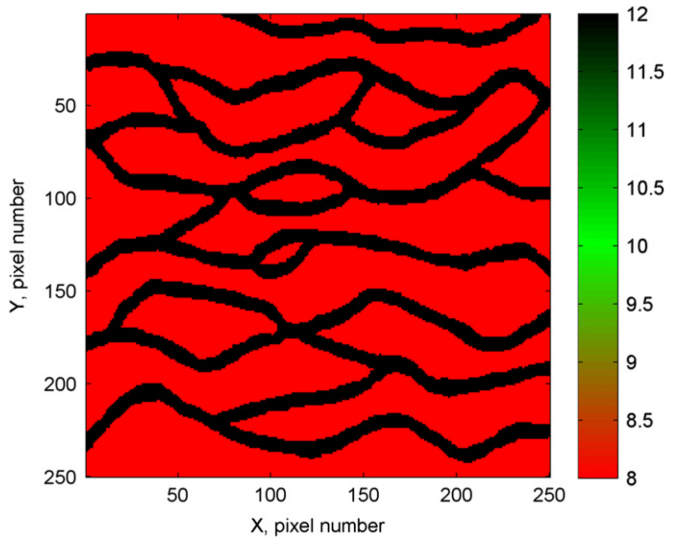

Fig. 4. Example of a training image for use with the SNESIM type a priori model.

distribution. An example of combining this distribution transform with the FFTMA type prior used to generate Fig. 2a is shown in Fig. 2b.

Note that when using the VISIM type prior one can use a target distribution directly, while ensuring that the chosen a priori covariance model is still honored. Using the distribution transform with the FFTMA prior will not preserve the properties of the a priori chosen covariance model.

Randomizing the model of spatial variability: As mentioned for the 'FFTMA' prior type model, the structural parameters that describe the a priori model covariance can be separated from the random number series that defines the random component. Therefore, all properties of the covariance model can be treated as model parameters, such as scaling and rotation. The properties of the model covariance can be perturbed independently of the random number series defining the random component (Le Ravalec et al., 2000).

In order to randomize a specific component of the covariance model, a GAUSSIAN type prior model needs to be defined for this component. The name of the specific prior model must be either `range_1`, `range_2`, or `range_3` to define the range, or one of `ang_1`, `ang_2`, or `ang_3` to define the rotation, and `m0` to define the a priori mean, and `sill` to define the sill. In addition, one must set the `prior_master` field to point the prior model that define the prior for the corresponding FFTMA a priori model.

As an example, consider the FFTMA example used to generate Fig. 2a. To randomize the maximum correlation length to be close to uniform between 6 and 14, and randomize the primary rotation angle to be close to uniform between 40 and 130 degrees (from north) use

```
im=1;
prior{im}.type='gaussian';
prior{im}.name='range_1';
prior{im}.m0=10;
prior{im}.std=4;
prior{im}.norm=80;
prior{im}.prior_master=3;
im=2;
prior{im}.type='gaussian';
prior{im}.name='ang_1';
prior{im}.m0=90;
prior{im}.std=50;
prior{im}.norm=80;
prior{im}.prior_master=3;

im=3;
prior{im}.type='FFTMA';
prior{im}.prior.x=[0:1:10]; % X array
prior{im}.prior.y=[0:1:20]; % Y array
prior{im}.m0=10;
prior{im}.Va='1 Sph(10,90,.25)';
```

Fig. 2c shows an example of five realizations from such an a priori model.

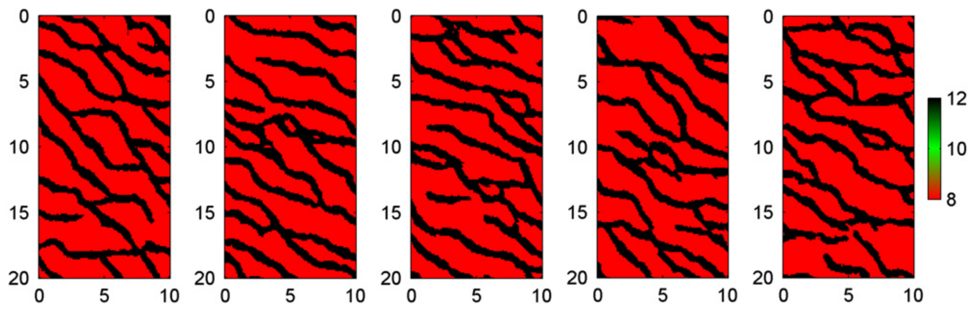

Fig. 5. Unconditional realizations from a SNESIM type a priori model.

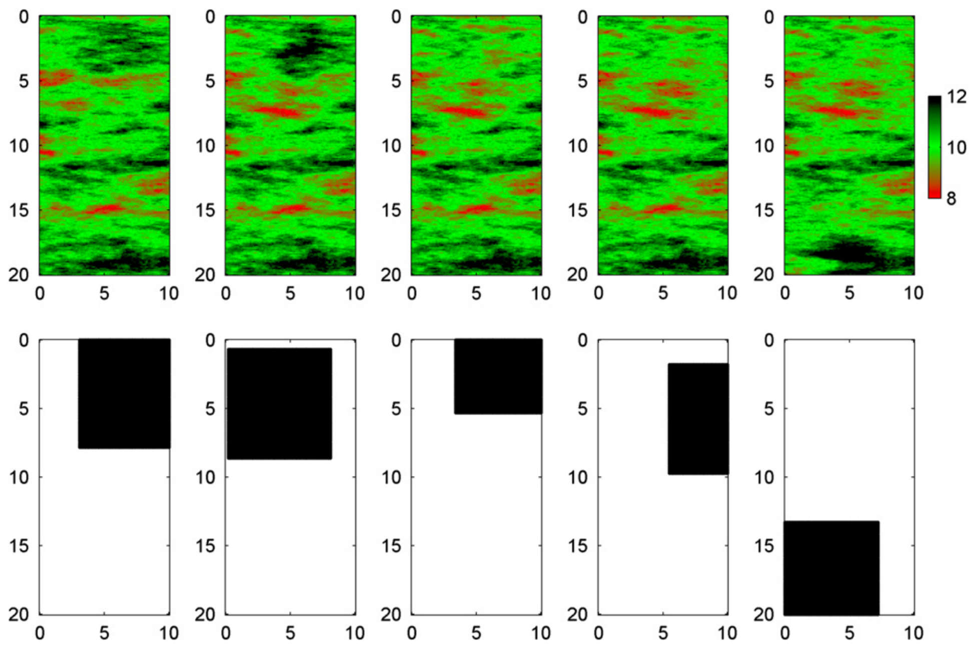

Fig. 6. (Top) Random walk using sequential Gibbs sampling with box type re-simulation, and the VISIM type a priori model. (Bottom) Black pixels indicate the model parameters that are simulated conditional to the value of the model parameters indicated by pixels.

3.1.1. *A random walk in the a priori model space*

To perform a random walk in the prior probability space, as needed by the extended Metropolis sampler, we make use of sequential Gibbs sampling (Hansen et al., 2012). An application of the sequential Gibbs sampler essentially amounts to selecting a subset, which can be any subset of model parameters, and simulate these conditional to the rest of the model parameters. The number of chosen model parameters in the subset controls the exploratory nature (i.e. step-length) of the sequential Gibbs sampler (which controls the degree of correlation between successive realizations), and hence the efficiency of the extended Metropolis sampler. All properties of the sequential Gibbs sampler are controlled by the seq_gibbs structure, which is a field in the prior data structure. Two different methods for selecting the subset of model parameters for conditional re-simulation have been implemented.

Box type subset: If prior{im}.seq_gibbs.type=1, then a line/rectangle/cube of model parameters (for the 1D, 2D and 3D case, respectively) is selected as the subset used for conditional re-simulation. The width of the box is defined by prior{im}.seq_gibbs.step. For example a box with dimension $2\times3\times4$ (in the units of the prior model considered - typically meters) is given by prior{im}.seq_gibbs.step=[2 3 4]. The center of the 'box' is chosen randomly

Randomly selected subset: If prior{im}.seq_gibbs.type=2, then a randomly selected number of the total number of model parameters is selected as the subset used for conditional resimulation. The number of data used for conditional re-simulation is given by prior{im}.seq_gibbs.step. If prior{im}.seq_gibbs.step is smaller than 1, it is interpreted as a percentage of the total number of model parameters.

As an example, five iterations of sequential Gibbs sampling can in SIPPI be performed using iterative calls to sippi_prior as

```
[m_current,prior]=sippi_prior(prior);
for i=1:5
[m_proposed,prior]=sippi_prior(prior,
m_current);
end
```

Figs. 6 and 7 show examples of using sequential Gibbs sampling with a box type selection and random type selection of model parameters for conditional re-simulation, respectively. The a priori model is in both cases the same as the one used to generate the unconditional realizations of Fig. 3. The options for the box type re-simulation are

```
prior{im}.seq_gibbs.type=1;
prior{im}.seq_gibbs.step=[4 4];
```

while the options for the random type re-simulation, with only 0.5% of the total number of model parameter used as conditional data for re-simulation, are

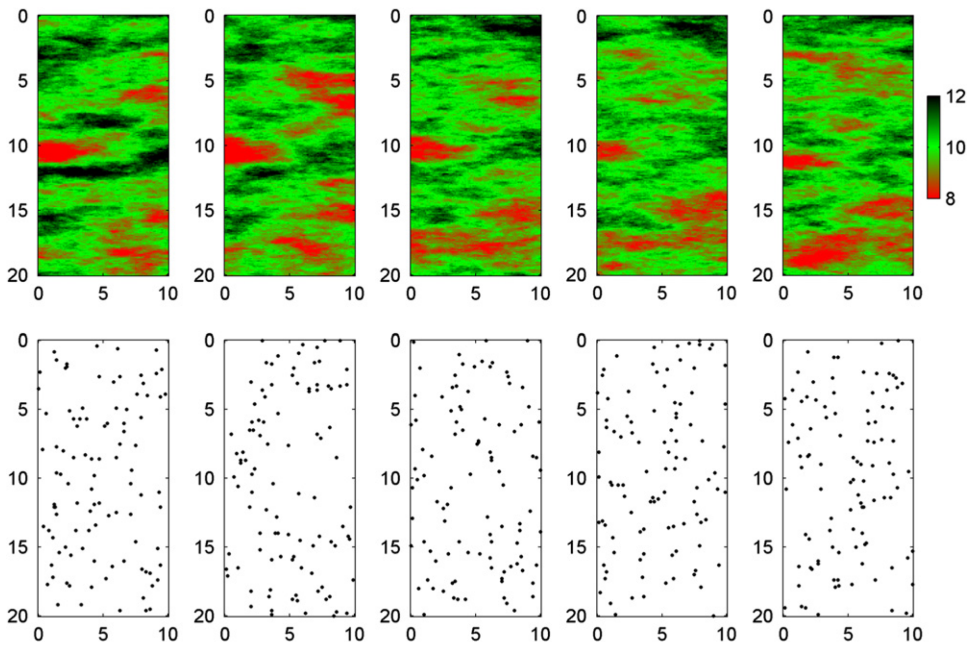

Fig. 7. (Top) Random walk using sequential Gibbs simulation with random choice of model parameters for resimulation, and the VISIM type a priori model. (Bottom) Black pixels indicate the model parameters that are simulated conditional to the value of the model parameters indicated by white pixels.

```
prior{im}.seq_gibbs.type=2;
prior{im}.seq_gibbs.step=0.995;
```

The sequential Gibbs sampler can be used with the FFTMA, VISIM, and SNESIM types a priori models. For the 1D GAUSSIAN type a priori model we use an alternate method. Given a current realization of the a priori model, a step length between 0 and 1 will generate a new realization of the prior, in the vicinity if the current realization. A step length of '0' indicates no change, while a step length of '1' will generate a new unconditional realization of the a priori model.

Fig. 8 shows the first 300 iterations when sampling the same a priori model as sampled in Fig. 1 using a step length of 0.25, `prior{im}.seq_gibbs.step=0.25`. After 100,000 iterations the histogram of the sampled model parameters resemble that of Fig. 1, and is therefore not shown here.

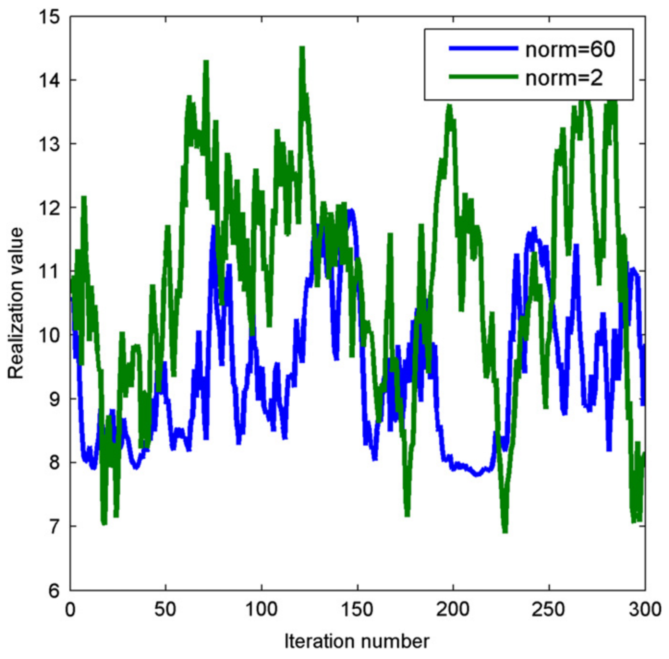

Fig. 8. The first 300 realizations from the GAUSSIAN type a priori model with a mean of 10, and a norm 60 and 2, respectively, using a step length of 0.25.

3.2. *Data, data uncertainties, modelization errors and the likelihood function*

Observed data must be given in the `data` data structure along with a description of the noise model. As for the `prior` structure, the `data` structure may consist of many types of data, where each data type number `id` is defined in the `data{id}` structure. Observed data are stored in the `d_obs` field. Uncorrelated uncertainty can be given either in the form of standard deviation, `d_std`, or variance, `d_var`. A simple data structure with such uncorrelated uncertainties can be given by

```
id=1;
data{id}.d_obs=[0 3 4]';
data{id}.d_std=[2 2 2]';
```

If the data uncertainties are uncorrelated, the noise model can be described by a generalized Gaussian model as defined in Eq. (7), if the norm of the generalized Gaussian is set by `data{id}.norm`. If not specified a Gaussian noise model (using a norm of 2) is chosen by default.

The noise model can also be given in the form of a correlated Gaussian model, for both the data noise, $\mathbf{C}_d$, and the modelization error, $\mathbf{C}_T$. The following will for example specify a correlated Gaussian noise model:

```
id=1;
data{id}.d_obs=[0 3 4]';
data{id}.Cd=[4 0 .1 ; 0 4 0 ; .1 0 4];
```

If a Gaussian model for the modelization error, $\mathcal{N}(\mathbf{d}_T,\mathbf{C}_T)$, is available it can be specified as

```
data{id}.dt=[0 -1 0]';
data{id}.Ct=[4 .1 .1 ; .1 4 .1 ; .1 .1 4];
```

where $\mathbf{d}_T$ is a bias correction.

One can choose to consider only a subset of the available data using the `i_use` field. To use for example only data number 1 and 3 use

```
id=1;
data{id}.d_obs=[0 3 4]';
data{id}.i_use=[1 3];
```

Once the data structure has been setup in `data`, the log-likelihood and the likelihood of a given data response d can be computed using

```
[log L,L,data]=sippi_likelihood(d,data);
```

3.3. The forward problem

The forward problem is naturally problem dependent, and to use SIPPI, the user needs to supply the solution to the forward problem, wrapped in the m-file `sippi_forward.m`.

The input to `sippi_forward.m` is the `forward`, `data` and `prior` Matlab structures. The `forward` structure can contain information on how to solve the forward problem. The output must be the data obtained by solving the forward problem, in the form of the data structure `d` which must be of the same length as the `data` structure, and each entry of `d{id}` must have the same size as `data{id}.d_obs`, or the size of `data{id}.i_use` if a data subset is specified.

As an alternative for providing `sippi_forward`, one can provide a generic name for the m-file solving the forward problem by setting `forward.forward_function`. Part 2 of this paper will provide an example of setting up `sippi_forward.m` (Hansen et al., this issue).

When the forward model has been setup, the process of generating an unconditional realization of the a priori model, `m`, followed by solving the forward problem and computing the likelihood of `m` can be done using

```
m=sippi_prior(prior);
d=sippi_forward(m,forward,prior,data);
log L=sippi_likelihood(d,data);
```

In the specific case where the forward relation is linear, the linear forward operator must be specified as the matrix G

```
forward.G
```

such that the forward problem can be solved using `d{1}=forward.G * m{1}`.

3.4. Sampling the a posteriori pdf

When the forward problem, `sippi_forward`, and the `prior`, `data`, and `forward` data structures have been defined, the a posteriori pdf can be sampled using the rejection sampler or the extended Metropolis sampler in the general non-linear case. In the linear Gaussian case, least-squares based inversion can be utilized.

3.5. Rejection sampling

Simple rejection sampling, using 30,000 iterations, of the a posteriori pdf can be performed using

```
options.mcmc.nite=30,000;
sippi_rejection(data,prior,forward,options);
```

By default the $L_{max}=1$, see Eq. (6). This can be manually changed by providing the `options.mcmc.Lmax`.

3.5.1. Metropolis sampling

All available a priori model types and noise models in SIPPI - work seamlessly as part of the extended Metropolis algorithm. The extended Metropolis sampling algorithm can be applied using

```
options=sippi_metropolis(data,prior,forward,options);
```

The `options` structure define some properties of how the Metropolis algorithm will run.

`options.mcmc.nite` determines the number of iterations of the extended Metropolis algorithm. `options.mcmc.i_sample` sets how often the current model is saved to disc, measured in the number of iterations. `options.mcmc.i_plot` sets the number of iterations between updating figures showing the progress of the algorithm. If any of these parameters are not set, then the following values will be chosen by default:

```
options.mcmc.nite= 30,000;
options.mcmc.i_sample= 500;
options.mcmc.i_plot: 50
```

Perturbation strategy: The choice of the number of model parameters to be perturbed in each iteration of the extended Metropolis algorithm can have large impact on its computational performance. By default a random type of model parameter is perturbed in each iteration. Thus if three types of a priori models have been specified in `prior{1}`, `prior{2}`, and `prior{3}`, the probability of perturbing each individual type of prior model in each iteration is 1/3. This default behavior can be changed by choosing a perturbation strategy. `options.mcmc.pert_strategy.i_pert` selects the number of a prior model types to perturb, and `options.mcmc.pert_strategy.i_pert_freq` set the relative frequency of each selected type of prior model. Thus, to perturb prior model 1 and 3 (but never model 2), such that prior model 3 is perturbed nine times as often as prior type 1, one could use

```
options.mcmc.pert_strategy.i_pert=[1 3];
options.mcmc.pert_strategy.i_pert_freq=[1 9];
```

Automatic adjustment of the exploration rate (step length): The exploratory nature of the Metropolis sampling algorithm, controlled by the 'step length', has large impact on its computational demands. A small step-length provides a dense local sampling, but the algorithm will use many iterations to move away from the initial point, i.e. a less exploratory algorithm. A large step length will lead to a very exploratory sampling algorithm that will not get trapped in local minima, but many models that are proposed will be rejected. Gelman et al. (1996) argue that a step-length leading to an acceptance rate in the Metropolis sampler of about 20–40% will lead to a good compromise between exploration and rejection rate. SIPPI allows automatic detection of the step length leading to an acceptance rate specified by `prior{im}.seq_gibbs.P_target`, using the method given by Cordua et al. (2012). Note that the Metropolis sampler will not sample the a posteriori pdf correct until the step-length is fixed, and unchanged. Therefore, one can set the number of initial iterations in which adjustment of the step length is allowed using `prior{im}.seq_gibbs.i_update_step_max`. After this, actual sampling of the a posteriori pdf will start, if the algorithm has reached burn-in. `prior{im}.seq_gibbs.i_update_step` sets the number of iterations between updating the step length. `prior{im}.seq_gibbs.step_min` and `prior{im}.seq_gibbs.step_max` determine the minimum and maximum allowed step length.

The default choice of the step length is to use infinitely long step-length, resulting in a prior sampler generating statistically independent realization of the prior in each iteration.

As an example, a preferred acceptance ratio of 0.3, adjusted in the first 1000 iterations, allowing step lengths in the interval 1–100 (using type 1 data subset), can be specified using:

```
prior{im}.seq_gibbs.type=1;
prior{im}.seq_gibbs.step_min=1;
prior{im}.seq_gibbs.step_max=100;
prior{im}.seq_gibbs.step=100;
prior{im}.seq_gibbs.i_update_step_max=1000;
prior{im}.seq_gibbs.P_target=0.3;
```

3.5.2. *Linear Gaussian inverse problems*

In the specific case where the forward problem is linear, and the a priori model Gaussian, as defined by the `VISIM` of `FFTMA` type a priori model, the a posteriori pdf can be sampled directly without the need for the Metropolis algorithm using

```
[m_reals,m_est,Cm_est]
  =sippi_least_squares(data,prior,forward,
n_reals,lsq_type);
```

`n_reals` sets how many a posteriori realizations, as output in `m_reals`, that are generated. `lsq_type` determines the method used to solve sample the a posteriori pdf. `m_est` and `Cm_est` are the a posteriori mean and covariance as given by Eq. (5), and are only available if least squares types of inversion is performed.

Three methods described previously are available to generate samples of the a posteriori pdf, and can be selected by setting the `lsq_type` argument when calling `sippi_least_squares`.

`lsq_type='lsq'` uses classical least-squares inversion where the complete Gaussian a posteriori pdf can be analytically derived in the form of a posteriori mean and covariance of Eqs. (4) and (5). Then Cholesky decomposition of the a posterior covariance is used to generated realizations of the a posteriori pdf.

`lsq_type='error_sim'` makes use of kriging simulation through error simulation to generate a sample of the a posteriori pdf (Journel and Huijbregts, 1978; Gloaguen et al., 2005a, 2005b; Hansen and Mosegaard, 2008).

`lsq_type='visim'` makes use of the VISIM algorithm for sampling the a posteriori pdf (Hansen and Mosegaard, 2008). The type of prior model must be chosen as a VISIM type prior model. If the target distribution is set as `prior{im}.target`, then VISIM runs as a direct sequential simulation algorithm. If it is not set, VISIM will run as a sequential Gaussian simulation algorithm.

4. Conclusions

A generic Matlab and Gnu Octave toolbox for sampling the a posteriori pdf of linear and non-linear inverse problems has been presented. Prior information about the model parameters can be described by any number of the following types of a priori models: (1) 1D arbitrarily distributed pdf, (2) 1D–3D multivariate Gaussian pdf as sampled using the FFTMA method, (3) 1D–3D multivariate Gaussian model as sampled using the VISIM algorithm (utilizing both sequential Gaussian simulation and direct sequential simulation), or (4) 1D–3D multiple-point based statistical models as sampled using the SNESIM algorithm.

For linear Gaussian inverse problems the a posteriori pdf can be sampled using (1) traditional least squares inversion combined with Cholesky decomposition of the a posteriori covariance, (2) sequential Gaussian simulation, (3) direct sequential simulation and (4) Gaussian simulation through error simulation.

For non-linear and non-Gaussian inverse problems the a posteriori pdf can be sampled using the rejection sampler or the extended Metropolis sampler. The computational efficiency of the extended Metropolis sampler can be controlled by using a flexible perturbation mechanism, based on sequential Gibbs sampling, allowing arbitrary long or short step length. The choice of the step length can optionally be automatized.

The combination of the FFTMA method with the extended Metropolis algorithm allows treating the properties describing the Gaussian a priori model, to be treated as model parameters, and thus inferred as part of the inversion.

Acknowledgment

We thank DONG for financial support. Interfacing geostatistical algorithms have been done using the mGstat toolbox, http://mgstat.sourceforge.net/. SIPPI source code can be downloaded from http://sippi.sourceforge.net/.

References

Cordua, K.S., Hansen, T.M., Mosegaard, K., 2012. Monte Carlo full waveform inversion of crosshole GPR data using multiple-point geostatistical a priori information. Geophysics 77, H19–H31.
Gelman, A., Roberts, G., Gilks, W., 1996. Efficient metropolis jumping rules. In: Bernardo, J., Berger, K., Dawid, A., Smith, A. (Eds.), Bayesian Statistics, vol. 5. Clarendon Press, Oxford, pp. 599–608.
Gloaguen, E., Marcotte, D., Chouteau, M., 2005a. A non-linear tomographic inversion algorithm based on iterated cokriging and conditional simulations. In: Leuangthong, O., Deutsch, C. (Eds.), Geostatistics Banff 2004, vol. 1. Springer, pp. 409–418.
Gloaguen, E., Marcotte, D., Chouteau, M., Perroud, H., 2005b. Borehole radar velocity inversion using cokriging and cosimulation. Journal of Applied Geophysics 57 (4), 242–259.
Gómez-Hernández, J., Froidevaux, R., Biver, P., 2005. Exact conditioning to linear constraints in kriging and simulation. In: Leuangthong, O., Deutsch, C. (Eds.), Geostatistics Banff 2004, vol. 2. Springer, pp. 999–1005.
Goovaerts, P., 1997. Geostatistics for natural resources evaluation. Applied Geostatistics Series. Oxford University Press.
Hansen, T., Cordua, K., Looms, M., Mosegaard, K. SIPPI: a Matlab toolbox for Sampling the solution to Inverse Problems with complex Prior Information: Part 2—Application to cross hole GPR tomography. Computers & Geosciences, http://dx.doi.org/10.1016/j.cageo.2012.10.001, this issue.
Hansen, T.M., Cordua, K.C., Mosegaard, K., 2012. Inverse problems with non-trivial priors—efficient solution through sequential Gibbs sampling. Computational Geosciences 16 (3), 593–611.
Hansen, T.M., Journel, A.G., Tarantola, A., Mosegaard, K., 2006. Linear inverse Gaussian theory and geostatistics. Geophysics 71 (6), R101–R111.
Hansen, T.M., Mosegaard, K., 2008. VISIM: sequential simulation for linear inverse problems. Computers and Geosciences 34 (1), 53–76.
Hansen, T.M., Mosegaard, K., Cordua, K.C., 2008. Using geostatistics to describe complex a priori information for inverse problems. In: Ortiz, J.M., Emery, X. (Eds.), VIII International Geostatistics Congress, vol. 1. Mining Engineering Department, University of Chile, pp. 329–338.
Hastings, W., 1970. Monte Carlo sampling methods using Markov chains and their applications. Biometrika 57 (1), 97.
Irving, J., Singha, K., 2010. Stochastic inversion of tracer test and electrical geophysical data to estimate hydraulic conductivities. Water Resource Research 46.
Journel, A., Zhang, T., 2006. The necessity of a multiple-point prior model. Mathematical Geology 38 (5), 591–610.
Journel, A.G., 1999. Conditioning geostatistical operations to nonlinear volume averages. Mathematical Geology 31, 931–953.
Journel, A.G., Huijbregts, C.J., 1978. Mining Geostatistics. Academic Press.
Le Ravalec, M., Noetinger, B., Hu, L.Y., 2000. The FFT moving average (FFT-MA) generator: an efficient numerical method for generating and conditioning Gaussian simulations. Mathematical Geology 32 (6), 701–723.
Mariethoz, G., Renard, P., Caers, J., 2010. Bayesian inverse problem and optimization with iterative spatial resampling. Water Resources Research 46 (11), W11530.
Metropolis, N., Rosenbluth, M., Rosenbluth, A., Teller, A., Teller, E., 1953. Equation of state calculations by fast computing machines. Journal of Chemical Physics 21, 1087–1092.
Mosegaard, K., Sambridge, M., 2002. Monte Carlo analysis of inverse problems. Inverse Problems 18 (3), 29–54.

Mosegaard, K., Tarantola, A., 1995. Monte Carlo sampling of solutions to inverse problems. Journal of Geophysical Research 100 (B7), 12431–12447.

Mosegaard, K., Tarantola, A., 2002. Probabilistic approach to inverse problems. In: Lee, W., Kanamori, H., Jennings, P., Kisslinger, C. (Eds.), International Handbook of Earthquake and Engineering Seismology, vol. 81A, pp. 237–265 (Chapter 16).

Pebesma, E.J., Wesseling, C.G., 1998. Gstat: a program for geostatistical modelling, prediction and simulation. Computers & Geosciences 24 (1), 17–31.

Remy, N., Boucher, A., Wu, J., 2008. Applied Geostatistics with SGeMS: A User's Guide. Cambridge University Press.

Strebelle, S., 2000. Sequential Simulation Drawing Structures from Training Images. Ph.D. Thesis, Stanford University.

Strebelle, S., 2002. Conditional simulation of complex geological structures using multiple-point statistics. Mathematical Geology 34 (1), 1–20.

Tarantola, A., 2005. Inverse Problem Theory and Methods for Model Parameter Estimation. SIAM.

Tarantola, A., Valette, B., 1982a. Generalized nonlinear inverse problems solved using the least squares criterion. Reviews of Geophysics and Space Physics 20 (2), 219–232.

Tarantola, A., Valette, B., 1982b. Inverse problems = quest for information. Journal of Geophysics 50 (3), 150–170.