User's Guide

January 2016

Julien Straubhaar

Software under patent protection

Contents

List of Figures

1 Introduction

DeeSse is a parallel software for multiple-point statistics simulations. The method is based on the direct sampling of the training image (conceptual model) [Mariethoz et al., 2010] and is patented [Mariethoz et al., 2014]. The code is written in C, and the parallel version used OpenMP and is devoted for shared memory machines.

DeeSse is a program to generate random fields reproducing the structures given by a conceptual model: the training image. Uni- and multi-variate simulations are allowed. In this latter case, spatial statistics between and within the variables of the training image are reproduced in the simulated images. In particular, multi-variate simulations can be used to deal with non-stationary training images.

DeeSse's main features.

- Categorical and continuous variables.
- Simulation for single variable or multiple joint variables.
- Conditioning to (punctual) data.
- Conditioning to block data (mean target values on subsets of pixels) [Straubhaar et al., 2015].
- Dealing with non-stationarity given by homothetics and rotations.
- Dealing with global and local probability constraints.
- Dealing with incomplete training images and training data set.
- Using multiple training images.

Meerschman et al. [2013] proposed a practical guide dealing with basic simulation cases.

Remark. *All grids (for simulation grids, training images, etc.) are defined as 3D regular grids. This allows to perform spatial simulations in 1D, 2D and 3D.*

Part I
Using DeeSse as console application

2 Grid, image, point set and block data set objects

Grid, image, point set and block data set are objects used in DeeSse. They are defined in the following. The file format used in DeeSse for image and point set are also given.

2.1 Grid object

A grid is a 3D regular box containing nodes (pixels). A grid is defined by the following parameters:

- Nx, Ny, Nz, the number of nodes in each direction;
- Sx, Sy, Sz, the spacing (unit for one node) in each direction;
- Ox, Oy, Oz, the origin of the grid: coordinates of the bottom low left corner of the node at the bottom low left of the grid (bottom, low and left are relative to z, y and x axis direction here).

The grid defined by the parameters above contains $Ng = Nx \cdot Ny \cdot Nz$ nodes; they are numbered from 0 to $Ng-1$, with x index varying first, then y index and finally z index. Thus, the node located at position of indices (ix, iy, iz) (in nodes) in the grid, has the (global) index $ix + Nx \cdot (iy + Ny \cdot iz)$, for $0 \leqslant ix \leqslant Nx-1$, $0 \leqslant iy \leqslant Ny-1$, $0 \leqslant iz \leqslant Nz-1$. The origin localizes in 3D space the bottom low left corner of the 0-th node of the grid. The top up right corner of the $(Ng-1)$-th node of the grid has the spatial coordinates $(Ox + Nx \cdot Sx, Oy + Ny \cdot Sy, Oz + Nz \cdot Sz)$.

Remark. *A grid is always defined in 3D space; however, setting Nx and/or Ny and/or Nz to one allows to consider a grid in 1D or 2D space.*

2.2 Image object

An image is a grid with one or more variable(s) attached to each node. An image is given by:

- a grid;
- a number $nvar$ of variable(s);
- the name of each variable;
- $nvar$ vectors $Z1, \ldots, Znvar$ of size Ng (number of nodes in the grid) containing the variable values at each node of the grid: $Zi(j)$ is the value of the i-th variable at the j-th node of the grid, $i = 1, \ldots, nvar$, $j = 0, \ldots, Ng-1$.

An image is given in a file in the following format (based on GSLIB, Geostatistical Software LIBrary, see http://www.gslib.com/).

Image file format

```
Nx Ny Nz [Sx Sy Sz [Ox Oy Oz]]
nvar
name_of_variable_1
...
name_of_variable_nvar
Z1(0)    ... Znvar(0)
...
Z1(Ng-1) ... Znvar(Ng-1)
```

Remark. *The parameters Sx, Sy, Sz and Ox, Oy, Oz are optional, the default value are $Sx = Sy = Sz = 1.0$ and $Ox = Oy = Oz = 0.0$.*

2.3 Point set object

A point set is a list of points spatially localized in 3D space with three coordinates, with one or more variable(s) attached to each of these points. A point set is givent by:

- a number *npoint* of point(s);
- a number *nvar* of variable(s);
- the coordinates x, y, z of each point;
- the name of each variable;
- *nvar* vectors $Z1, \ldots, Znvar$ of size *npoint* containing the variable values at each point: $Zi(j)$ is the value of the i-th variable at the j-th point, $i = 1, \ldots, nvar$, $j = 1, \ldots, npoint$.

A point set is given in a file in the following format (based on GSLIB, Geostatistical Software LIBrary, see http://www.gslib.com/).

Point set file format

```
npoint
nvar+3
name_for_x_coordinate
name_for_y_coordinate
name_for_z_coordinate
name_of_variable_1
...
name_of_variable_nvar
x(1)      y(1)      z(1)      Z1(1)      ... Znvar(1)
...
x(npoint) y(npoint) z(npoint) Z1(npoint) ... Znvar(npoint)
```

Remark. *One number ($nvar + 3$) is written on the second line; the name for x, y and z coordinates must begin by x or X, y or Y and z or Z respectively; $x(j)$, $y(j)$ and $z(j)$ are the values of the x, y and z coordinates of the j-point.*

2.4 Block data set object

A block data set is a list of block data where each block data is given by a set of nodes spatially localized in an underlying grid (indeed the simulation grid), a mean value attached to the set of nodes, a tolerance value (around the mean), and a minimal and maximal "activation proportion"

indicating when the block data constraint is activated (during the simulation, the proportion of informed nodes plus the node currently simulated in the block is compared to these bounds and the block data constraint for this block is activated or not). A block data set is given by:

- a number $nblockData$ of block data;
- for each block data $1 \leqslant i \leqslant nblockData$:
  - a number $nnode(i)$ of nodes;
  - a mean value $value(i)$;
  - a tolerance value $tolerance(i)$;
  - a minimal "activation proportion" value $propMin(i)$;
  - a maximal "activation proportion" value $propMax(i)$;
  - three vectors $ix(i)$, $iy(i)$, $iz(i)$ of size $nnode(i)$ containing the indices in the underlying grid of each node constituting the i-th block (set of nodes).

A block data set is given in a file in the following format.

Block data set file format

```
nblockData
nnode(1) value(1) tolerance(1) propMin(1) propMax(1)
ix(1)(1)        iy(1)(1)        iz(1)(1)
...
ix(1)(nnode(1)) iy(1)(nnode(1)) iz(1)(nnode(1))
...
nnode(nblockData) value(nblockData) tolerance(nblockData) propMin(nblockData) propMax(nblockData)
ix(nblockData)(1)                 iy(nblockData)(1)                 iz(nblockData)(1)
...
ix(nblockData)(nnode(nblockData)) iy(nblockData)(nnode(nblockData)) iz(nblockData)(nnode(nblockData))
```

Remark. *The entries $ix(i)(j)$, $iy(i)(j)$, $iz(i)(j)$ are node indices in a grid along x, y, z direction respectively. Node indices in a grid start from* 0 *(Sect. 2.1).*

2.5 Missing value

A reserved constant, `MPDS_MISSING_VALUE`, are used in DeeSse for encoding a missing (or undefined) value. `MPDS_MISSING_VALUE` is set to $-9'999'999$. This allows to define a variable on a part of an image or a point set.

3 Input parameters file for DeeSse (simple example)

```
/* SIMULATION GRID (SG) */
200    200     1    // size    in each direction
  1.0    1.0   1.0  // spacing in each direction
  0.0    0.0   0.0  // origin

/* SIMULATION VARIABLES */
/* Number of simulation variable(s), and for each variable:
      variable name, output flag (0 / 1), and if output flag is 1: format string (as passed to fprintf).
   Write DEFAULT_FORMAT for format string to use default format.
   Example (with 3 variables):
      3
      varName1  1  %10.5E
```

```
      varName2  0
      varName3  1  DEFAULT_FORMAT
*/
1
facies      1  DEFAULT_FORMAT

/* OUTPUT SETTINGS FOR SIMULATION */
/* Key word and required name(s) or prefix, for output of the realizations:
      - OUTPUT_SIM_NO_FILE:
           no file in output,
      - OUTPUT_SIM_ALL_IN_ONE_FILE:
           one file in output,
           requires one file name
      - OUTPUT_SIM_ONE_FILE_PER_VARIABLE:
           one file per variable in output (flagged as 1 above),
           requires as many file name(s) as variable(s) flagged as 1 above
      - OUTPUT_SIM_ONE_FILE_PER_REALIZATION:
           one file per realization,
           requires one prefix (for file name) */
OUTPUT_SIM_ALL_IN_ONE_FILE
test_simul.gslib

/* OUTPUT REPORT */
/* Flag (0 / 1), and if 1, output report file. */
1
test_report.txt

/* TRAINING IMAGE */
/* Number of training image(s) (nTI >= 1), followed by nTI file(s)
   (a file can be replaced by the string "_DATA_" which means that the
   simulation grid itself is taken as training image), and
   if nTI > 1, one pdf image file (for training images, nTI variables). */
1
ti.gslib

/* DATA IMAGE FILE FOR SG */
/* Number of image file(s) (n >= 0), followed by n file(s). */
0

/* DATA POINT SET FILE FOR SG */
/* Number of point set file(s) (n >= 0), followed by n file(s). */
0

/* MASK IMAGE */
/* Flag (0: mask not used / 1: mask used) and if 1, mask image file
   (this image contains one variable on the simulation grid: flag (0 / 1)
   for each node of the simulation grid that indicates if the variable(s)
   will be simulated at the corresponding node (flag 1) or not (flag 0). */
0

/* HOMOTHETY */
/* 1. Homothety usage, integer (homothetyUsage):
        - 0: no homothety
        - 1: homothety without tolerance
        - 2: homothety with tolerance
   2a. If homothetyUsage == 1,
          then for homothety ratio in each direction,
          first for x, then for y, and then for z-axis direction:
             - Flag (0 / 1) indicating if given in an image file,
               followed by
                  - one value (real) if flag is 0
                  - name of the image file (one variable) if flag is 1
   2b. If homothetyUsage == 2,
          then for homothety ratio in each direction,
          first for x, then for y, and then for z-axis direction:
             - Flag (0 / 1) indicating if given in an image file,
               followed by
                  - two values (lower and upper bounds) (real) if flag is 0
```

```
                - name of the image file (two variables) if flag is 1
*/
0

/* ROTATION */
/* 1. Rotation usage, integer (rotationUsage):
        - 0: no rotation
        - 1: rotation without tolerance
        - 2: rotation with tolerance
   2a. If rotationUsage == 1,
          then for each angle,
          first for azimuth, then for dip, and then for plunge:
             - Flag (0 / 1) indicating if given in an image file,
               followed by
                - one value (real) if flag is 0
                - name of the image file (one variable) if flag is 1
   2b. If rotationUsage == 2,
          then for each angle,
          first for azimuth, then for dip, and then for plunge:
             - Flag (0 / 1) indicating if given in an image file,
               followed by
                - two values (lower and upper bounds) (real) if flag is 0
                - name of the image file (two variables) if flag is 1
*/
0

/* CONSISTENCY OF CONDITIONING DATA (TOLERANCE RELATIVELY TO THE RANGE OF TRAINING VALUES) */
/* Maximal accepted expansion in both extremities of the range of values in training images
   for covering the conditioning data values; e.g. if this number is set to 0.05,
   the conditioning data values can be beyond the range of the values in the training images
   (in both extremities) of at most 5%; this separately applies to all variables for which
   the distance is absolute (not relative, see "relative distance flag" below) and the
   distance type is not 0 (see "distance type" below). */
0.05

/* NORMALIZATION TYPE (FOR VARIABLES FOR WHICH DISTANCE TYPE IS NOT 0 AND DISTANCE IS ABSOLUTE) */
/* Available types:
      - NORMALIZING_LINEAR
      - NORMALIZING_UNIFORM
      - NORMALIZING_NORMAL */
NORMALIZING_LINEAR

/* SEARCH NEIGHBORHOOD PARAMETERS */
/* A search neighborhood is a 3D ellipsoid, defined by:
      - search radii (in number of nodes), for each direction
        (-1.0 for a default value automatically computed)
      - anisotropy ratios, for each direction, i.e. numbers of nodes corresponding
        to a distance of one, in each direction; for example (1.0, 1.0, 2.0) means
        that the distance to the central node is the Euclidean distance where
        the unit (distance=1) corresponds to 1, 1 and 2 nodes for the 1st, 2nd and
        3rd direction respectively.
      - angles (azimuth, dip and plunge) defining the rotation that sends the coordinates
        system xyz onto the coordinates system x'y'z' in which the search radii
        and the anisotropy ratios are given
      - power at which the distance is elevated for computing the weight of each
        node in the search neighborhood
   Note that
      - the search neighborhood is delimited by the search radii and the angles
      - the anisotropy ratios are used only for computing the distance to the central
        node, from each node in the search neighborhood
      - the nodes inside the search neighborhood are sorted according to their
        distance to the central node, from the closest one to the furthest one */
/* SEARCH NEIGHBORHOOD PARAMETERS FOR VARIABLE #0 */
 -1.0   -1.0    0.0   // search radius in each direction
  1.0    1.0    1.0   // anisotropy ratio in each direction
  0.0    0.0    0.0   // angles (azimuth, dip, plunge in degrees) for rotation
  0.0                 // power for computing weight according to distance
```

```
/* MAXIMAL NUMBER OF NEIGHBORING NODES FOR EACH VARIABLE (as many number(s) as number of variable(s)) */
20

/* MAXIMAL DENSITY OF NEIGHBORING NODES IN SEARCH NEIGHBORHOOD FOR EACH VARIABLE (as many number(s)
   as number of variable(s)) */
1.0

/* RELATIVE DISTANCE FLAG FOR EACH VARIABLE (as many flag(s) (0 / 1) as number of variable(s)) */
0

/* DISTANCE TYPE FOR EACH VARIABLE (as many number(s) as number of variable(s)) */
/* Available distance (between data events):
      - 0: non-matching nodes (typically for categorical variable)
      - 1: L-1 distance
      - 2: L-2 distance
      - 3: L-p distance, requires the real positive parameter p
      - 4: L-infinity distance */
0

/* WEIGHT FACTOR FOR CONDITIONING DATA, FOR EACH VARIABLE (as many number(s) as number of variable(s)) */
/* For the computation of distance between data events, if a value is a conditioning
   data, its corresponding contribution is multiplied by the factor given here. */
1.0

/* SIMULATION AND PATH PARAMETERS */
/* Key word for simulation type:
      - SIM_ONE_BY_ONE:
           successive simulation of one variable at one node in the simulation grid (4D path)
      - SIM_VARIABLE_VECTOR:
           successive simulation of all variable(s) at one node in the simulation grid (3D path) */
SIM_ONE_BY_ONE

/* Key word for path type:
      - PATH_RANDOM:
           random path, for simulation type:
              - SIM_ONE_BY_ONE     : path visiting all nodes and variables in a random order
              - SIM_VARIABLE_VECTOR: path visiting all nodes in a random order
      - PATH_UNILATERAL:
           unilateral path, for simulation type:
              - SIM_ONE_BY_ONE: requires a vector of size 4.
                   Example: u = (0, -2, 1, 0) means that the path will visit all nodes:
                   randomly in xv-sections, with increasing z-coordinate, and then decreasing y-coordinate.
              - SIM_VARIABLE_VECTOR: requires a vector of size 3.
                   Example: u = (-1, 0, 2) means that the path will visit all nodes:
                   randomly in y-sections, with decreasing x-coordinate, and then increasing z-coordinate.
           This vector must be given after the key word PATH_UNILATERAL. */
PATH_RANDOM

/* DISTANCE THRESHOLD FOR EACH VARIABLE (as many number(s) as number of variable(s)) */
0.01

/* PROBABILITY CONSTRAINTS */
/* FOR EACH VARIABLE:
   1. Probability constraint usage, integer (probabilityConstraintUsage):
        - 0: no probability constraint
        - 1: global probability constraint
        - 2: local probability constraint

   2. If probabilityConstraintUsage > 0, then the classes of values (for which the
        probability constraints will be given) have to be defined; a class of values
        is given by a union of interval(s): [inf_1,sup_1[ U ... U [inf_n,sup_n[;
        Here are given:
           - nclass: number of classes of values
           - for i in 1,..., nclass: definition of the i-th class of values:
                - ninterval: number of interval(s)
                - inf_1 sup_1 ... inf_ninterval sup_ninterval: inf and sup for each interval
                     these values should satisfy inf_i < sup_i
```

```
   3a. If probabilityConstraintUsage == 1, then
          - global probability for each class (defined in 2. above), i.e.
            nclass numbers in [0,1] of sum 1
   3b. If probabilityConstraintUsage == 2, then
          - one pdf image file (for every class, nclass variables)
            (image of same dimensions as the simulation grid)
          - support radius for probability maps (i.e. distance according to
            the unit defined in the search neighborhood parameters for the
            considered variable)
          - method for computing the current pdf (in the simulation grid),
             integer (LocalCurrentPdfComputation):
               - 0: "COMPLETE" mode: all the informed node in the search neighborhood
                    for the considered variable, and within the support are taken into account
               - 1: "APPROXIMATE" mode: only the neighboring nodes (used for the
                    search in the TI) within the support are taken into account

   4. If probabilityConstraintUsage > 0, then
         method for comparing pdf's, integer (comparingPdfMethod):
            - 0: MAE (Mean Absolute Error)
            - 1: RMSE (Root Mean Squared Error)
            - 2: KLD (Kullback Leibler Divergence)
            - 3: JSD (JSD (Jensen-Shannon Divergence)
            - 4: MLikR (Mean Likelihood Ratio (over each class indicator, sym. target interval))
            - 5: MLikR2 (Mean Likelihood Ratio 2 (over each class indicator, custom target interval))

   5. If probabilityConstraintUsage > 0, then
         - deactivation distance, i.e. one positive number
           (the probability constraint is deactivated if the distance between
           the current simulated node and the last node in its neighbors (used
           for the search in the TI) (distance computed according to the corresponding
           search neighborhood parameters) is below the given deactivation distance)

   6. If probabilityConstraintUsage > 0, then
         - threshold type for pdf's comparison, integer (probabilityConstraintThresholdType)
              - 0: constant threshold
              - 1: dynamic threshold
           note: if comparingPdfMethod is set to 4 or 5, probabilityConstraintThresholdType must be set to 0
         6.1a If probabilityConstraintThresholdType == 0, then
                 - threshold value
         6.1b If probabilityConstraintThresholdType == 1, then the 7 parameters:
                 - M1 M2 M3
                 - T1 T2 T3
                 - W
              These parameters should satisfy:
                 0 <= M1 <= M2 < M3,
                 T1 >= T2 >= T3,
                 w != 0.
              The threshold value t is defined as a function of the number M
              of nodes used for computing the current pdf (in the simulation grid)
              including the candidate (i.e. current simulated) node by:
                 t(M) = T1, if M < M1
                 t(M) = T2, if M1 <= M < M2
                 t(M) = (T3 - T2) / (M3^W - M2^W) * (M^W - M2^W) + T2, if M2 <= M < M3
                 t(M) = T3, if M3 <= M
*/
/* PROBABILITY CONSTRAINTS FOR VARIABLE #0 */
0

/* BLOCK DATA */
/* FOR EACH VARIABLE:
   1. Block data usage, integer (blockDataUsage):
         - 0: no block data
         - 1: block data: block mean value

   2. If blockDataUsage == 1, then
         - block data file name
*/
/* BLOCK DATA FOR VARIABLE #0 */
```

```
0

/* MAXIMAL SCAN FRACTION FOR EACH TI (as many number(s) as number of training image(s)) */
0.3

/* TOLERANCE */
/* Tolerance t on the threshold value for flagging nodes:
   let d(i) be the distance between the data event in the simulation grid and in the training
   image for the i-th variable and s(i) be the distance threshold for the i-th variable, and let
   e(i) = max(0, (d(i)-s(i))/s(i)) be the relative error for the i-th variable, i.e. the relative
   part of the distance d(i) beyond the threshold s(i); during the scan of the training image, a node
   that minimizes e = sum (e(i)) is retained (the scan is stopped if e = 0); if e is greater than the
   tolerance t (given here), then the current simulated node and all non-conditioning nodes of the
   data events (one per variable) in the simulation grid are flagged for resimulation (post-processing).
   Note that if probability constraints is used, a similar error as e(i) is computed from the comparison
   of the pdf's, and will contributed in the sum defining the error e. */
0.0

/* POST-PROCESSING */
/* 1. Maximal number of path(s) (npostProcessingPathMax)
   2. If npostProcessingPathMax > 0:
      key word for post-processing parameters (i. e. number of neighboring nodes, distance threshold,
      maximal scan fraction, and tolerance):
         - POST_PROCESSING_PARAMETERS_DEFAULT: for default parameters
         - POST_PROCESSING_PARAMETERS_SAME   : for same parameters as given above
         - POST_PROCESSING_PARAMETERS_MANUAL : for manual settings
   3. If npostProcessingPathMax > 0 and POST_PROCESSING_PARAMETERS_MANUAL:
         MAXIMAL NUMBER OF NEIGHBORING NODES FOR EACH VARIABLE (as many number(s) as number of variable(s))
         MAXIMAL DENSITY OF NEIGHBORING NODES IN SEARCH NEIGHBORHOOD FOR EACH VARIABLE (as many number(s)
            as number of variable(s))
         DISTANCE THRESHOLD FOR EACH VARIABLE (as many number(s) as number of variable(s))
         MAXIMAL SCAN FRACTION FOR EACH TI (as many number(s) as number of training image(s))
         TOLERANCE
*/
1
POST_PROCESSING_PARAMETERS_DEFAULT

/* SEED NUMBER AND SEED INCREMENT */
444
1

/* NUMBER OF REALIZATIONS */
1

END
```

About the input parameters file

- The file can contain C-style comments.
- The file ends with the character string `END`.
- The output flag for the simulation variable(s) indicates if the corresponding variable will be considered for output or not.
- The output settings for simulation are determined by one of the following key words:
  - `OUTPUT_SIM_NO_FILE`: no file will be produced in output.
  - `OUTPUT_SIM_ALL_IN_ONE_FILE`: one file in image file format will be produced in output, all variables for all realizations will be written in columns; for N+1 variables and M+1 realizations:

```
var0_real0 ... varN_real0 ... var0_realM ... varN_realM
```

  - `OUTPUT_SIM_ONE_FILE_PER_VARIABLE`: one file in image file format per simulation variable will be produced in output, in each file all realizations will be written in columns; for variable I and M+1 realizations:

    `varI_real0 ... varI_realM`

  - `OUTPUT_SIM_ONE_FILE_PER_REALIZATION`: one file in image file format per realization will be produced in output, in each file all variables will be written in columns; for N+1 variables and realization J:

    `var0_realJ ... varN_realJ`

- A report file can be written, it contains some informations about the simulation.

- A TI is given in a file in image file format. The number of variable(s) must be equal to the number of simulation variable(s), and the i-th variable corresponds to the i-th simulation variable, whatever its name. If more than one TI is used, one file for PDF is given in image file format; its grid must have the same parameters as those of the SG and it must contain as many variables as number of TIs, the i-th variable is the probability to choose the i-th TI.

- A TI can be incomplete, *i.e.* some variable(s) at some nodes can be uninformed (undefined), using the constant `MPDS_MISSING_VALUE` ($-9'999'999$). Moreover, a TI can be replaced by the SG itself, which means that the SG is scanned instead the TI during the simulation; for this, the TI file name is replaced by the string `_DATA_` in the input parameters file.

- Conditioning data can be given in image(s) or in point set(s). The grid of an image of conditioning data must have the same parameters as those of the SG. The number of variable(s) given in image(s) or point set(s) of conditioning data can be smaller than the number of simulation variable(s); however the name(s) of variable(s) describing the conditioning data must correspond to the name(s) of the wanted simulation variable(s). Note that the conditioning data given in image(s) are first integrated in the SG and then those given in point set(s) (in the given order), by eventually erasing the previous values.

- The grid of the eventual mask image file must have the same parameters as those of the SG.

- The grid of the eventual image file(s) for homothety and/or rotation must have the same parameters as those of the SG.

- If local probability constraints are used for a simulated variable, one file for local PDF is given in image file format; its grid must have the same parameters as those of the SG and it must contain as many variables as number of specified classes of values, the i-th variable is the local probability for the i-th class. Moreover, some nodes can be uninformed; for such a node, the value for all the variables is set to `MPDS_MISSING_VALUE` ($-9'999'999$).

Remark. *Constant values for* DeeSse *are defined in the header file MPDSConst.h.*

4 Launching DeeSse

For launching the serial version of DeeSse as console application, you have to type in the terminal

```
DeeSse <params.in>
```

where `DeeSse` is the executable for the serial version of DeeSse and `<params.in>` the name of the input file.

For using the parallel version of DeeSse (based on OpenMP), you have to type

```
DeeSseOMP <nthreads> <params.in>
```

where `DeeSseOMP` is the executable for the parallel version of DeeSse, `<ntreads>` the number of threads to be used and `<params.in>` the name of the input file.

Details for the input file are given in the previous section.

5 Recommendations to users

Here some useful recommendations about input parameters are given.

- Use a small simulation grid in a first step, to see if there is already some trouble.
- The maximal number(s) of neighbors, the distance threshod(s), and the maximal scanned fraction(s) of the TI(s) are key parameters; they can have a great impact on the CPU time required by the simulation. Hence, start with parameters not too demanding, and then adapt them. Note that the maximal scanned fraction of a TI should be chosen by taking into account the size of the TI.
- Search neighborhoods should be large enough (in number of nodes) for catching the structures within the TI(s).

Part II
Examples

6 Example 1: simple simulation

This example consists in an unconditional simulation of one categorical variable. The main input parameters are given below, the TI is shown in Fig. 1a and the simulation in Fig. 1b. Note that the input parameters file given in Sect. 3 is used for this example.

Main input parameters for example 1 (Fig. 1).

- SG: $Nx = 200$, $Ny = 200$, $Nz = 1$, ($Sx = Sy = Sz = 1.0$, $Ox = Oy = Oz = 0.0$);
- one simulation variable, categorical taking two values;
- unconditional;
- no homothety, no rotation;
- distance type 0 (mismatching nodes);
- one TI of size $Nx = 250$, $Ny = 250$, $Nz = 1$ (Fig. 1a);
- search neighborhood: $r_x = -1.0$, $r_y = -1.0$, $r_z = 0.0$, $a_x = a_y = a_z = 1.0$, $\alpha = \beta = \gamma = 0.0$, $p = 0.0$ ($r_x = -1.0$ and $r_y = -1.0$ results in this example in (automatically computed) radii of $r = 124$ in number of nodes);
- random simulation path;
- maximal number of neighbors: $N = 20$;
- distance threshold: $t = 0.01$;
- no probability constraint;
- no block data;
- maximal scanned fraction of the TI: $f = 0.3$;
- tolerance (on relative error): $tol = 0.0$;
- one post-processing path with the default parameters.

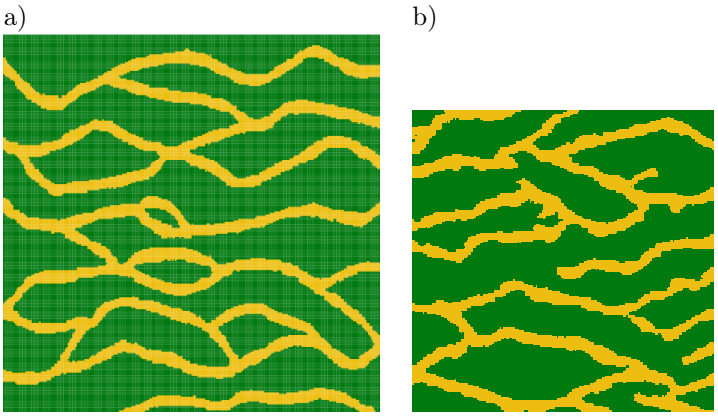

Fig. 1 DeeSse example 1: simple simulation. a) TI, $Nx = Ny = 250$, $Nz = 1$, one categorical variable (two values); b) unconditional simulation, $Nx = Ny = 200$, $Nz = 1$, obtained by using the input parameters given in section 6. (The input parameters file is given in section 3)

7 Example 2: conditional simulations of one continuous variable

For this example, the TI shown in Fig. 2a is used; it has one continuous variable taking its values in the interval $[50.0, 100.0]$. A simulation conditioned to 10 data and using the L_1 distance in absolute mode (standard) is shown in Fig. 2b. A simulation conditioned to 24 data and using the L_1 distance in relative mode is shown in Fig. 2c; this second simulation shows that the relative mode allows to adapt the range of the simulated value to the conditioning data.

Main input parameters for example 2 (Fig. 2).

- SG:

  2.1) $Nx = 250$, $Ny = 250$, $Nz = 1$ for simulation in Fig. 2b;

  2.2) $Nx = 600$, $Ny = 250$, $Nz = 1$ for simulation in Fig. 2c;

  (and $Sx = Sy = Sz = 1.0$, $Ox = Oy = Oz = 0.0$ for both cases);

- one simulation variable, continuous;

- conditional, with:

  2.1) 10 data for simulation in Fig. 2b;

  2.2) 24 data for simulation in Fig. 2c;

- no homothety, no rotation;

- distance type 1 (L_1), in

  2.1) absolute mode (and linear normalization) for simulation in Fig. 2b;

  2.2) relative mode for simulation in Fig. 2c;

- one TI of size $Nx = 400$, $Ny = 500$, $Nz = 1$ (Fig. 2a);

- search neighborhood: $r_x = 180.0$, $r_y = 180.0$, $r_z = 0.0$, $a_x = a_y = a_z = 1.0$, $\alpha = \beta = \gamma = 0.0$, $p = 0.0$;

- random simulation path;

- maximal number of neighbors: $N = 16$;

- distance threshold:

  2.1) $t = 0.01$ for simulation in Fig. 2b;

  2.2) $t = 0.5$ for simulation in Fig. 2c;

- no probability constraint;

- no block data;

- maximal scanned fraction of the TI: $f = 0.1$;

- tolerance (on relative error): $tol = 0.0$;
- one post-processing path with the default parameters.

Note that the variable is not normalized in the second case because relative distance is used; this explains the choice of a larger distance threshold.

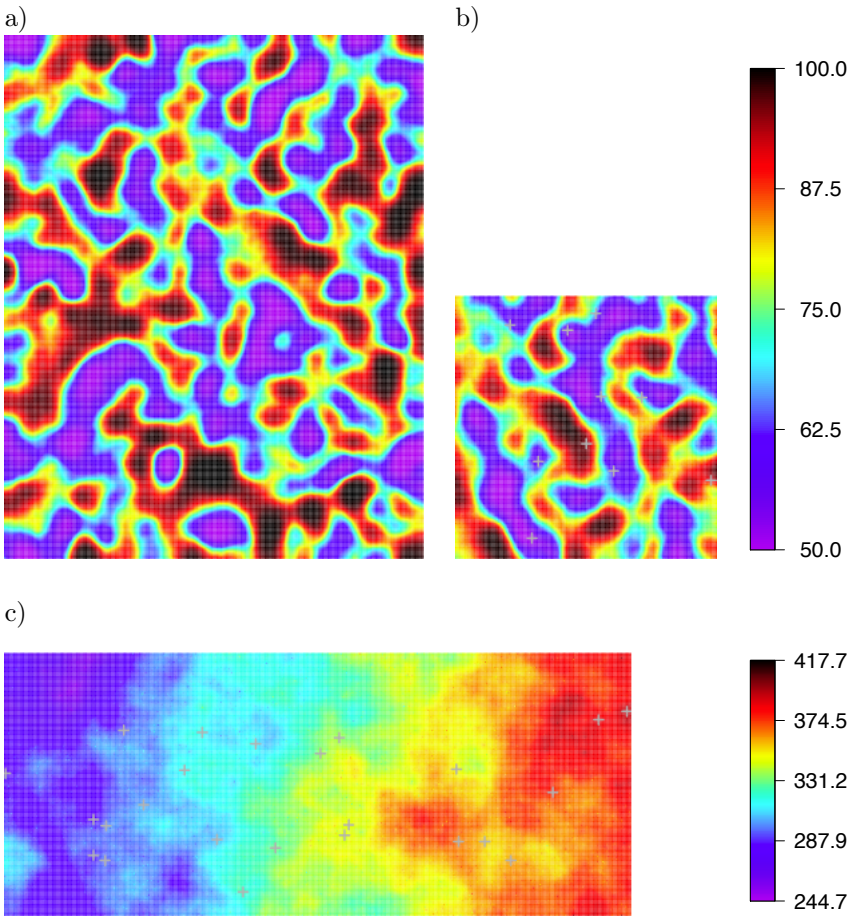

Fig. 2 DeeSse example 2: conditional simulations of a continous variable. a) TI, $Nx = 400$, $Ny = 500$, $Nz = 1$, one continuous variable; b) conditional simulation, $Nx = Ny = 250$, $Nz = 1$, using distance L_1 (in absolute mode), 10 conditioning data; c) conditional simulation, $Nx = 600$, $Ny = 250$, $Nz = 1$, using distance L_1 in relative mode, 24 conditioning data. The gray crosses show the location of the conditioning data. The input parameters for both cases are given in section 7

8 Examples 3 and 4: simulations using homothety and rotation

The usage of homothety and rotation is illustrated by two examples: the example 3 (Fig. 3) where fixed mode is used, and the example 4 (Fig. 4) where flexible mode is used. In each example, two simulations are performed: one simulation where the homothety and rotation parameters are globally defined on the whole SG, and one simulation where they are defined on the SG with images.

Main input parameters for example 3 and 4 (Figs 3 and 4).

- SG: $Nx = 400$, $Ny = 300$, $Nz = 1$, ($Sx = Sy = Sz = 1.0$, $Ox = Oy = Oz = 0.0$);
- one simulation variable, categorical taking two values;
- unconditional;
- homothety and rotation:

  3.1) fixed mode, the same parameters for all nodes in the SG:
  - for homothety: $t_x = 1.33$, $t_y = 0.75$, $t_z = 1$;
  - for rotation: $\alpha = 20.0$, $\beta = 0.0$, $\gamma = 0.0$;

  3.2) fixed mode, the parameters for the nodes of the SG given by images:
  - for homothety: image in Fig. 3c for t_x and t_y ($t_z = 1.0$ everywhere);
  - for rotation: image in Fig. 3d for α ($\beta = \gamma = 0.0$ everywhere);

  4.1) flexible mode, the same parameters for all nodes in the SG:
  - for homothety: $t_x, t_y \in [0.5, 2.0]$ ($t_z = 1.0$ everywhere);
  - for rotation: $\alpha \in [0.0, 180.0]$ ($\beta = \gamma = 0.0$ everywhere);

  4.2) flexible mode, the parameters for the nodes of the SG given by images:
  - for homothety: lower an upper bounds for t_x and t_y given by the images in Figs 4c and 4d respectively ($t_z = 1.0$ everywhere); these images are obtained from the image in Fig. 3c by adding and subtracting 0.3 resp.;
  - for rotation: lower an upper bounds for α given by the images in Figs 4e and 4f respectively ($\beta = \gamma = 0.0$ everywhere); these images are obtained from the image in Fig. 3d by adding and subtracting 45.0 resp.;

- distance type 0 (mismatching nodes);
- one TI of size $Nx = 250$, $Ny = 250$, $Nz = 1$ (Figs 3a and 4a);
- search neighborhood: $r_x = 80.0$, $r_y = 80.0$, $r_z = 0.0$, $a_x = a_y = a_z = 1.0$, $\alpha = \beta = \gamma = 0.0$, $p = 0.0$;
- random simulation path;
- maximal number of neighbors: $N = 20$;
- distance threshold: $t = 0.01$;

- no probability constraint;
- no block data;
- maximal scanned fraction of the TI: $f = 0.5$;
- tolerance (on relative error): $tol = 0.0$;
- one post-processing path with the default parameters.

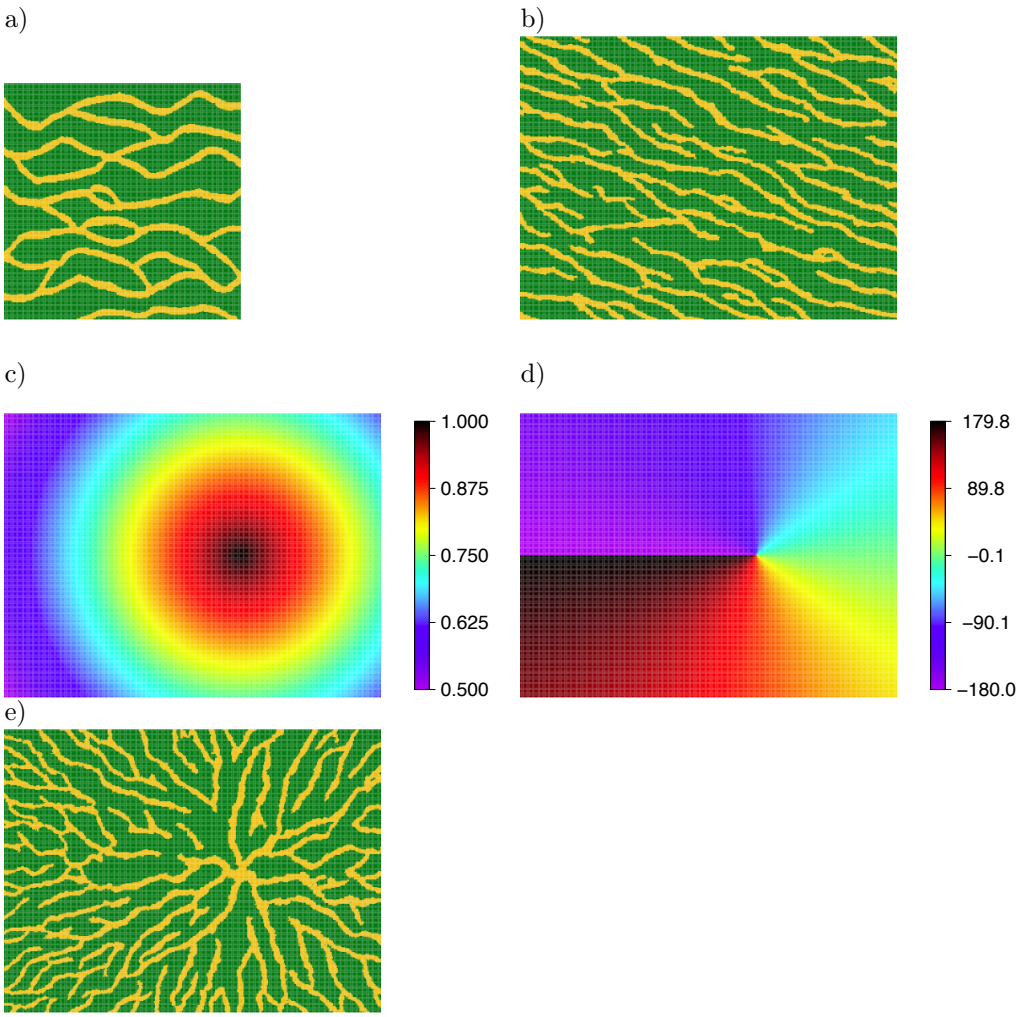

Fig. 3 DeeSse example 3: simulations using fixed homothety and rotation. a) TI, $Nx = 250$, $Ny = 250$, $Nz = 1$, one categorical variables (two values); b) simulation, $Nx = 400$, $Ny = 300$, $Nz = 1$, with fixed homothety given by $t_x = 1.33$, $t_y = 0.75$, $t_z = 1$ and fixed rotation given by $\alpha = 20.0$, $\beta = \gamma = 0.0$, for all nodes in the SG; c) homothety factors t_x and t_y in the SG for the simulation e); d) azimuth α in the SG for the simulation e); e) simulation, $Nx = 400$, $Ny = 300$, $Nz = 1$, with fixed homothety and rotation given by c) and d) respectively. The input parameters for both simulations are given in section 8

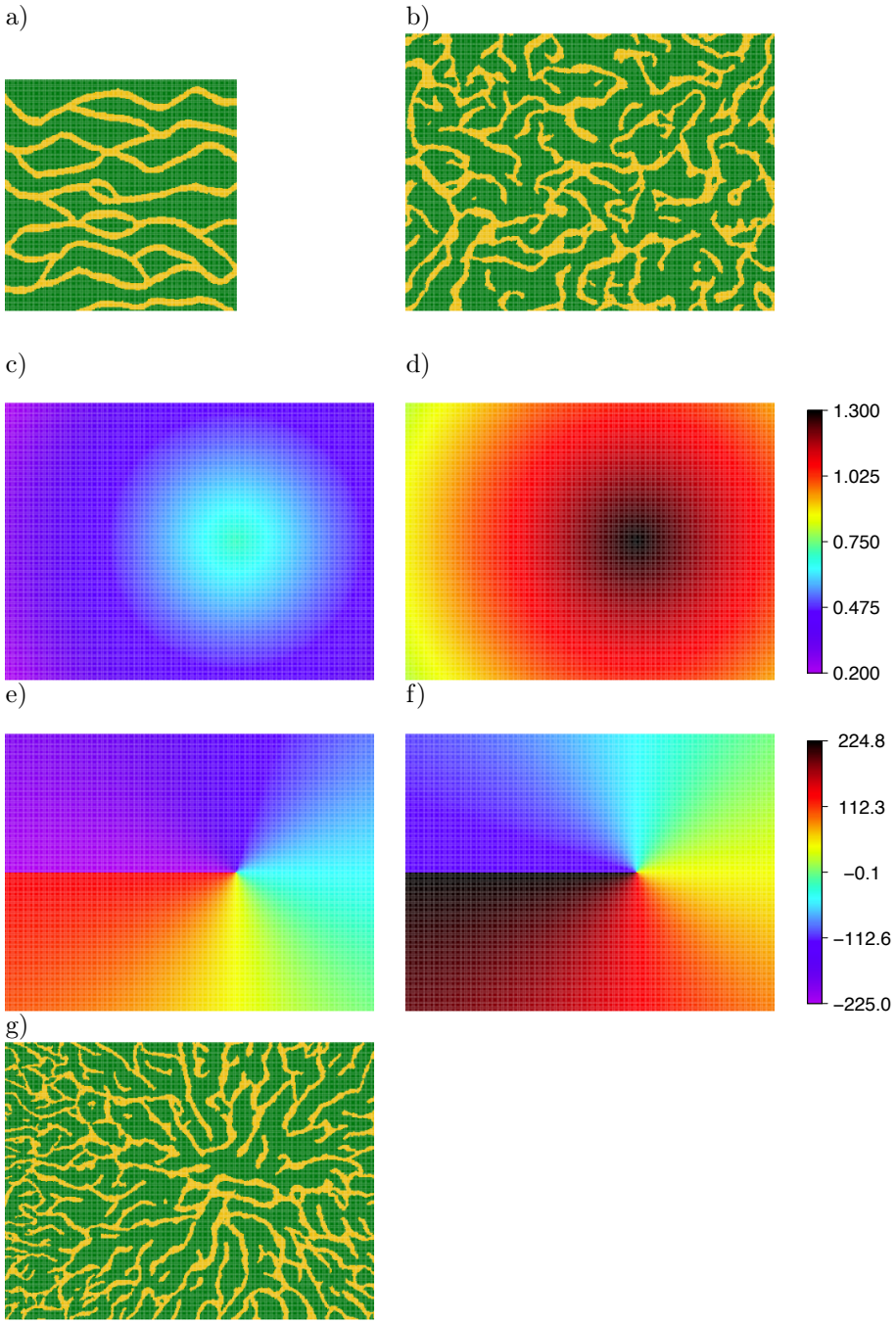

Fig. 4 DeeSse example 3: simulations using flexible homothety and rotation. a) TI, $Nx = 250$, $Ny = 250$, $Nz = 1$, one categorical variables (two values); b) simulation, $Nx = 400$, $Ny = 300$, $Nz = 1$, with flexible homothety given by $t_y, t_x \in [0.5, 2.0]$, $t_z = 1$ and flexible rotation given by $\alpha \in [0.0, 180.0]$, $\beta = \gamma = 0.0$, for all nodes in the SG; c) lower and d) upper bounds for homothety factors t_x and t_y in the SG for the simulation g); e) lower and f) upper bounds for azimuth α in the SG for the simulation g); g) simulation, $Nx = 400$, $Ny = 300$, $Nz = 1$, with flexible homothety and rotation whose lower and upper bounds are given by c), d), e) and f). The input parameters for both simulations are given in section 8

9 Example 5: bivariate simulations

This example consists in unconditional simulations of two variables (categorical and continuous). Two cases are considered with a simulation path defined by using the "full 4D" and "vector" simulation modes respectively. The TI is shown in Figs 5a: it gives the structures and the correlation of the two variables. The simulations provided by the main input parameters below are shown in Figs 5b and 5c.

Main input parameters for example 5 (Fig. 5).

- SG: $Nx = 500$, $Ny = 200$, $Nz = 1$, ($Sx = 1.95312$, $Sy = 1.5625$, $Sz = 1.0$ as for TI, $Ox = Oy = Oz = 0.0$);
- two simulation variables, first variable: categorical taking two values, second variable: continuous;
- unconditional;
- no homothety, no rotation;
- distance type 0 (mismatching nodes) for the first variable, distance type 1 (L_1, in absolute mode) for the second variable;
- one TI of size $Nx = 764$, $Ny = 239$, $Nz = 1$, with $Sx = 1.95312$, $Sy = 1.5625$, $Sz = 1.0$ (Fig. 5a);
- search neighborhood for the two variables: $r_x = 200.0$, $r_y = 115.0$, $r_z = 0.0$, $a_x = a_y = a_z = 1.0$, $\alpha = \beta = \gamma = 0.0$, $p = 0.0$;
- random simulation path,

  5.1) in "full 4D" simulation mode for simulation in Fig. 5b;

  5.2) in "vector" simulation mode for simulation in Fig. 5c;
- maximal number of neighbors: $N_1 = 24$ and $N_2 = 20$ for the first and the second variable respectively;
- maximal density of neighbors: $MD_1 = 1.0$ and $MD_2 = 1.0$ for the first and the second variable respectively;
- distance threshold: $t_1 = 0.01$ and $t_2 = 0.02$ for the first and second variables respectively;
- no probability constraint;
- no block data;
- maximal scanned fraction of the TI: $f = 0.25$;
- tolerance (on global relative error): $tol = 0.0$;
- one post-processing path with the default parameters.

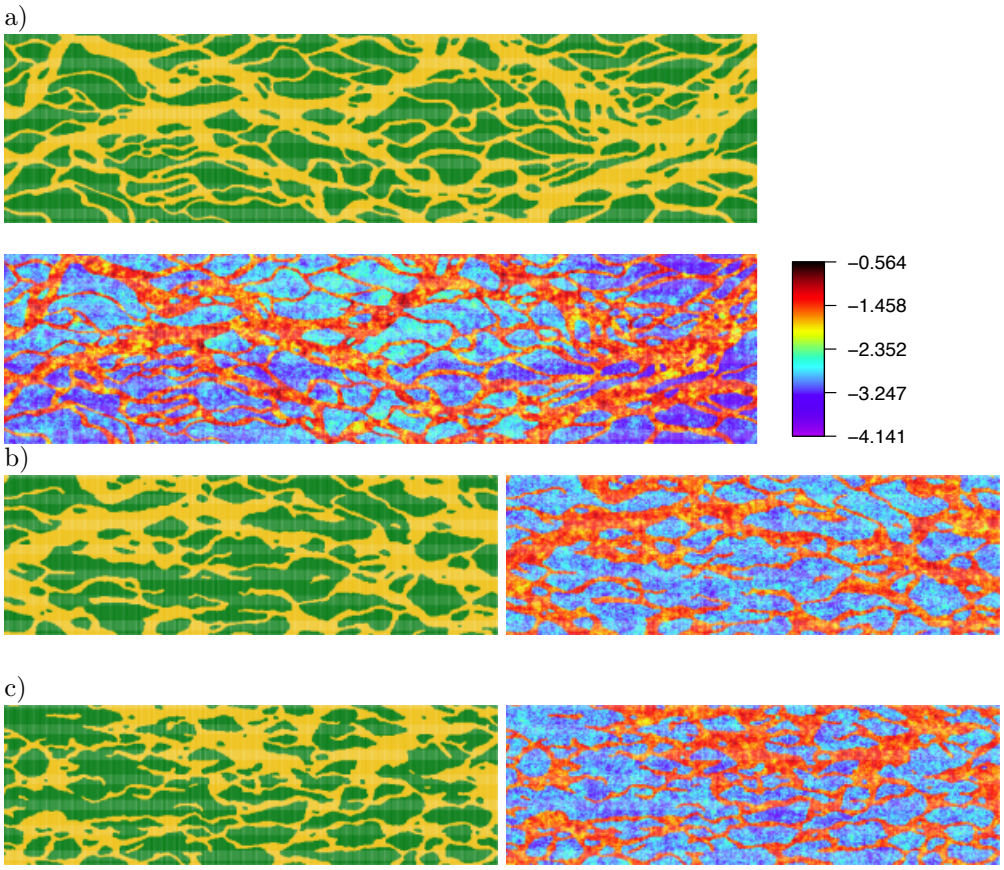

Fig. 5 DeeSse example 5: bivariate simulations; a) first and second variables of the TI, $Nx = 764$, $Ny = 239$, $Nz = 1$; b) first and second variables for a simulation using "full 4D" simulation mode for path; c) first and second variables for a simulation using "vector" simulation mode for path. Th size of the SG is $Nx = 500$, $Ny = 200$, $Nz = 1$, and for the TI and the SG: $Sx = 1.95312$, $Sy = 1.5625$, $Sz = 1.0$, and $Ox = Oy = Oz = 0.0$. The color bar is used for all images of the second variable. The input parameters for both simulations are given in section 9

10 Example 6: simulations using multiple TIs

For this example, the TIs shown in Figs 6a and 6b are used; they have one categorical variable taking two values. The Figs 6c and 6d show the probability at each node of the SG for choosing these TIs. The resulting simulation is shown in Fig. 6e.

Main input parameters for example 6 (Fig. 6).

- SG: $Nx = 300$, $Ny = 300$, $Nz = 1$, ($Sx = Sy = Sz = 1.0$, $Ox = Oy = Oz = 0.0$);
- one simulation variable, categorical taking two values;
- unconditional;
- no homothety, no rotation;
- distance type 0 (mismatching nodes);
- two TIs, first TI of size $Nx = 250$, $Ny = 250$, $Nz = 1$ (Fig. 6a), second TI of size $Nx = 400$, $Ny = 400$, $Nz = 1$ (Fig. 6b); a PDF for choosing these TIs are given on the SG (Fig. 6c for the first TI and Fig. 6d for the second TI);
- search neighborhood: $r_x = 120.0$, $r_y = 120.0$, $r_z = 0.0$, $a_x = a_y = a_z = 1.0$, $\alpha = \beta = \gamma = 0.0$, $p = 0.0$;
- random simulation path;
- maximal number of neighbors: $N = 20$;
- distance threshold: $t = 0.01$;
- no probability constraint;
- no block data;
- maximal scanned fractions of the TIs: $f_1 = 0.5$ and $f_2 = 0.3$ for the first and the second TI respectively;
- tolerance (on relative error): $tol = 0.0$;
- one post-processing path with the default parameters.

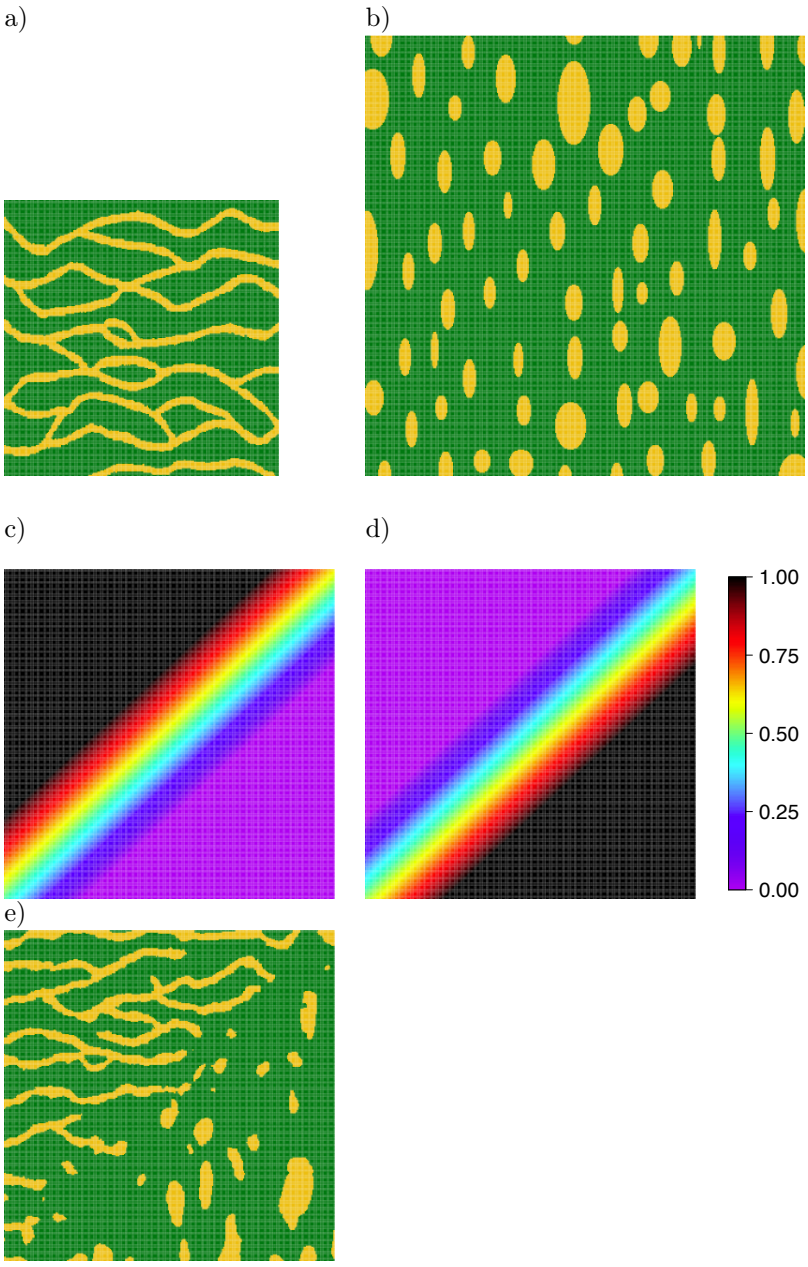

Fig. 6 DeeSse example 6: simulations using two TIs. a) first TI, $Nx = Ny = 250$, $Nz = 1$; b) second TI, $Nx = Ny = 400$, $Nz = 1$; c) probability (defined on the SG) for choosing the first TI; d) probability (defined on the SG) for choosing the second TI; e) simulation, $Nx = Ny = 300$, $Nz = 1$, using the TIs a) and b) according to the probability c) and d). The input parameters for both cases are given in section 10

11 Example 7: simulations using incomplete TI / training data set

This example illustrates the usage of an incomplete TI, *i.e.* TI containing uninformed nodes, and a training data set, that is the SG itself is used as TI starting with the conditioning data. The image in Fig. 7a is used as incomplete TI for the unconditional simulation in Fig. 7b, as incomplete TI and conditioning data for the simulation in Fig. 7c, and as training data set for the simulation in Fig. 7d.

Main input parameters for example 7 (Fig. 7).

- SG: $Nx = 250$, $Ny = 250$, $Nz = 1$, ($Sx = Sy = Sz = 1.0$, $Ox = Oy = Oz = 0.0$);
- one simulation variable, categorical taking two values;
- conditioning data:
  - 7.1) none (unconditional simulation) for simulation in Fig. 7b;
  - 7.2) image in Fig. 7a for simulation in Fig. 7c;
  - 7.3) image in Fig. 7a for simulation in Fig. 7d;
- no homothety, no rotation;
- distance type 0 (mismatching nodes);
- TI:
  - 7.1) image in Fig. 7a for simulation in Fig. 7b;
  - 7.2) image in Fig. 7a for simulation in Fig. 7c;
  - 7.3) SG (training data set);
- search neighborhood: $r_x = 120.0$, $r_y = 120.0$, $r_z = 0.0$, $a_x = a_y = a_z = 1.0$, $\alpha = \beta = \gamma = 0.0$, $p = 0.0$;
- random simulation path;
- maximal number of neighbors: $N = 50$;
- distance threshold: $t = 0.01$;
- no probability constraint;
- no block data;
- maximal scanned fraction of the TI: $f = 0.5$;
- tolerance (on relative error): $tol = 0.0$;
- one post-processing path with the default parameters.

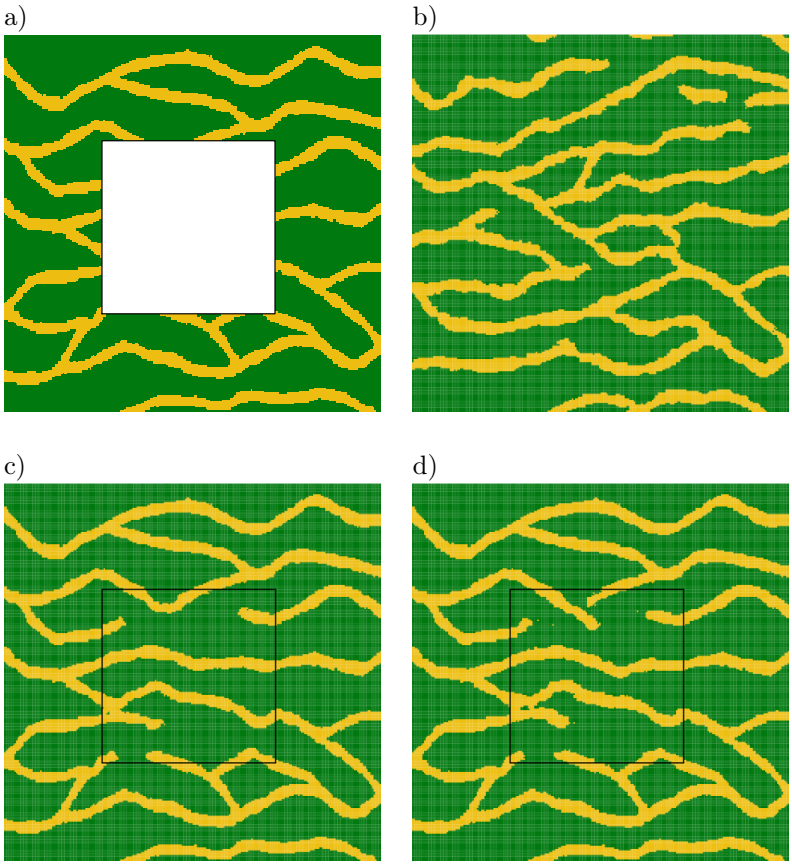

Fig. 7 DeeSse example 7: incomplete TI and training data set. a) incomplete image, $Nx = Ny = 250$, $Nz = 1$, one categorical variable (two values); b) unconditional simulation, $Nx = Ny = 250$, $Nz = 1$, obtained by using the image a) as TI; c) simulation, $Nx = Ny = 250$, $Nz = 1$, obtained by using the image a) as conditioning data and as TI; d) simulation, $Nx = Ny = 250$, $Nz = 1$, obtained by using the image a) as training data set, *i.e.* as conditioning data, and the SG itself as TI. The area corresponding to the uninformed nodes in a) is surrounded with a black line in b), c) and d). The input parameters for these three simulations are given in section 11

12 Example 8: 3D simulation

An unconditional 3D simulation is presented in this section. The TI and the simulation are shown in Fig. 8; The main input parameters are given below.

Main input parameters for example 8 (Fig. 8).

- SG: $Nx = 100$, $Ny = 94$, $Nz = 60$, ($Sx = Sy = Sz = 1.0$, $Ox = Oy = Oz = 0.0$);
- one simulation variable, categorical taking four values;
- unconditional;
- no homothety, no rotation;
- distance type 0 (mismatching nodes);
- one TI of size $Nx = 100$, $Ny = 94$, $Nz = 60$ (Fig. 1a);
- search neighborhood: $r_x = 45.0$, $r_y = 45.0$, $r_z = 25.0$, $a_x = a_y = a_z = 1.0$, $\alpha = \beta = \gamma = 0.0$, $p = 0.0$;
- random simulation path;
- maximal number of neighbors: $N = 32$;
- distance threshold: $t = 0.02$;
- no probability constraint;
- no block data;
- maximal scanned fraction of the TI: $f = 0.5$;
- tolerance (on relative error): $tol = 0.0$;
- one post-processing path with the default parameters.

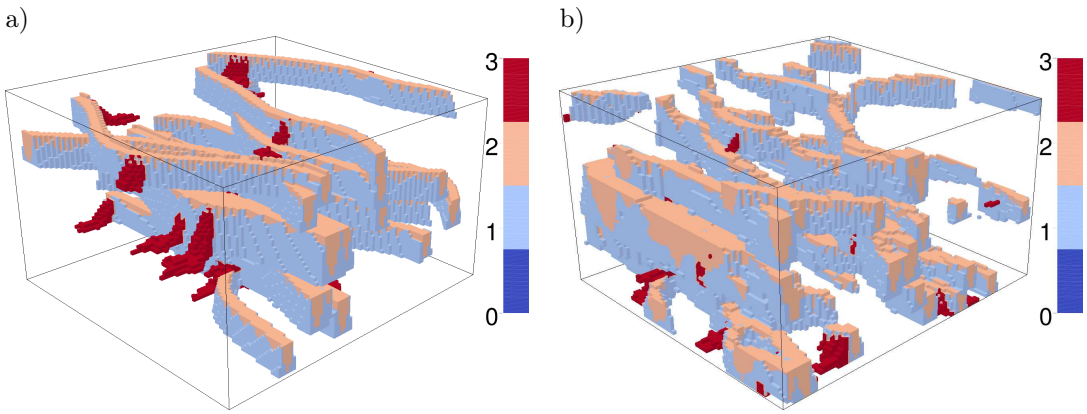

Fig. 8 DeeSse example 8: 3D simulation. a) TI, $Nx = 100$, $Ny = 94$, $Nz = 60$, one categorical variable (four values); b) unconditional simulation, $Nx = 100$, $Ny = 94$, $Nz = 60$, obtained by using the input parameters given in section 12

13 Examples 9 and 10: simulations with probability constraints on a categorical variable

The usage of global and local probability constraints for a categorical variable is illustrated in this section. The TI used for the examples is shown in Fig. 9a and has one categorical variable taking the values 0, 1, 2 with proportions 0.52, 0.23, 0.25. For both examples, each value correspond to one class for PDF contraints. In example 9, global probability constraints are considered, the target proportions are 0.20, 0.50, 0.30 and the proportions in the simulation displayed in Fig. 9b are 0.21, 0.50, 0.29. In example 10, local probability constraints are considered, the target local probabilities for value 0, 1, 2 are shown in Figs 10a, 10b, 10c respectively, and the support radius (distance) is set to 12.0 (nodes). The simulation is displayed in Fig. 10d. The main input parameters are given below for both examples.

Main input parameters for examples 9 and 10 (Figs 9 and 10).

- SG: $Nx = 300$, $Ny = 300$, $Nz = 1$, ($Sx = Sy = Sz = 1.0$, $Ox = Oy = Oz = 0.0$);
- one simulation variable, categorical taking three values (0, 1, 2);
- unconditional;
- no homothety, no rotation;
- distance type 0 (mismatching nodes);
- one TI of size $Nx = 114$, $Ny = 114$, $Nz = 1$ (Fig. 9a);
- search neighborhood: $r_x = 55.0$, $r_y = 55.0$, $r_z = 0.0$, $a_x = a_y = a_z = 1.0$, $\alpha = \beta = \gamma = 0.0$, $p = 0.0$;
- random simulation path;
- maximal number of neighbors: $N = 24$;
- distance threshold: $t = 0.1$;
- probability constraints: 3 classes of values: {0}, {1}, {2}

  9) global constraints, target PDF: $(0.2, 0.5, 0.3)$, comparing PDFs with MLikR2, deactivation distance: 4.0, constant threshold: 0.001

  10) local constraints, target PDF: maps in Figs 10a, 10b, 10c, support radius (distance): 12.0, comparing PDFs with MLikR2, deactivation distance: 4.0, constant threshold: 0.01

- no block data;
- maximal scanned fraction of the TI: $f = 0.5$;
- tolerance (on relative error): $tol = 0.0$;
- one post-processing path with the default parameters.

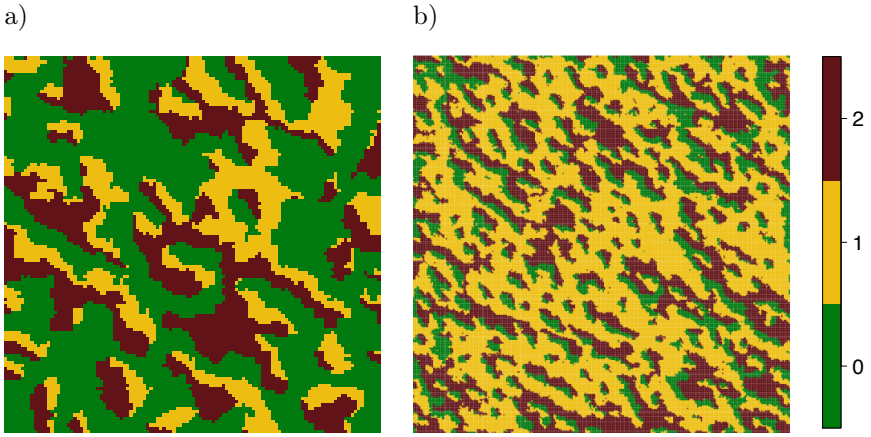

Fig. 9 DeeSse example 9: global probability constraints on a categorical variable. a) TI, $Nx = Ny = 114$, $Nz = 1$, one categorical variable (values 0, 1, 2 with proportions 0.515, 0.231, 0.254); b) simulation with global probability constraints, $Nx = Ny = 300$, $Nz = 1$, obtained by using the input parameters (9) given in section 13 (target proportions: 0.20, 0.50, 0.30 / proportions in simulation: 0.191, 0.513, 0.296)

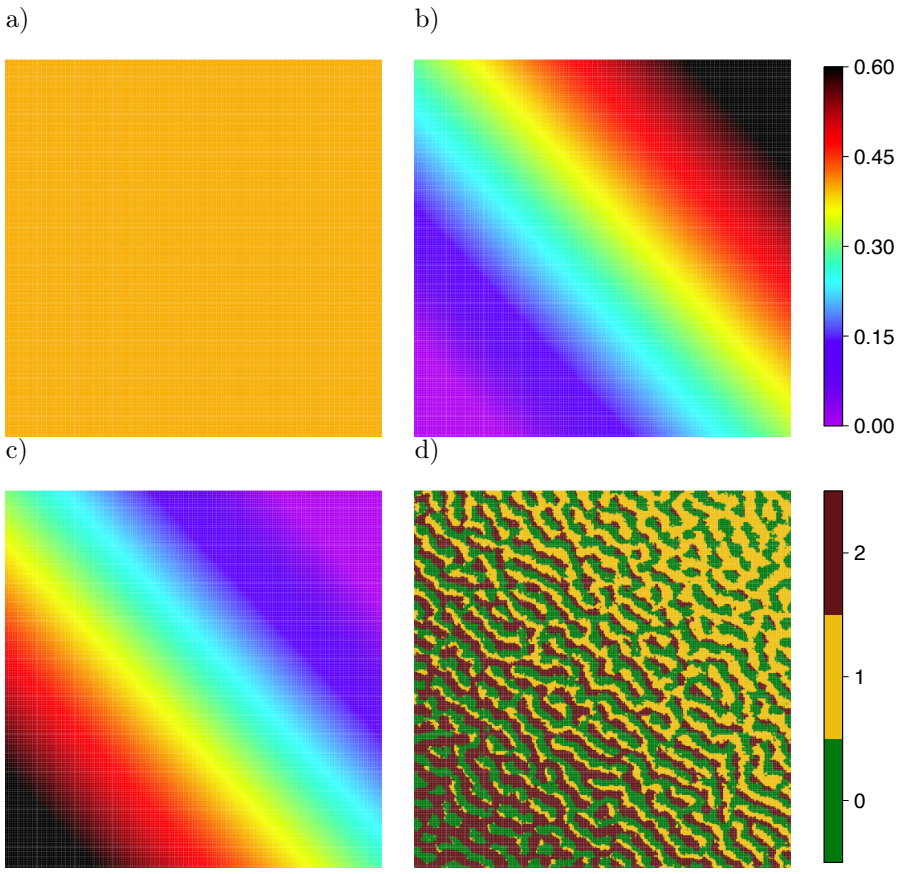

Fig. 10 DeeSse example 10: local probability constraints on a categorical variable. The TI used is shown in figure 9a. a) probability (defined on the SG) for value 0; b) probability (defined on the SG) for value 1; c) probability (defined on the SG) for value 2; d) simulation with local probability constraints shown in a), b) and c) and a support radius of 12.0 (nodes), $Nx = Ny = 300$, $Nz = 1$. (Same color scale is used for a), b) and c). The input parameters (10) for the simulation are given in section 13)

14 Examples 11 and 12: simulations with probability constraints on a continuous variable

The usage of global and local probability constraints for a continuous variable is illustrated in this section. The TI used for the examples is shown in Fig. 11a and has one continuous variable taken values in $[0, 256[$. The histogram of values in the TI is displayed in Fig. 11b. In example 11, global probability constraints are considered, the target proportions are the uniform distribution on the 10 classes of values obtained by dividing the interval $[0, 256[$ in regular sub-intervals. The simulation and its histogram of values (on the specified classes) are shown in Figs 11c and 11d. In example 10, local probability constraints are considered, with the two following classes of values: $[0, 50[$ and $[50, 256[$. The target local probabilities for the class of small values are shown in Fig. 12a, and the support radius (distance) is set to 12.0 (nodes). The simulation is displayed in Fig. 12b. The main input parameters are given below for both examples.

Main input parameters for examples 11 and 12 (Figs 11 and 12).

- SG: $Nx = 200$, $Ny = 200$, $Nz = 1$, ($Sx = Sy = Sz = 1.0$, $Ox = Oy = Oz = 0.0$);
- one simulation variable, continuous taking values in $[0, 256[$;
- unconditional;
- no homothety, no rotation;
- distance type 1 (L_1);
- one TI of size $Nx = 200$, $Ny = 200$, $Nz = 1$ (Fig. 9a);
- search neighborhood: $r_x = 90.0$, $r_y = 90.0$, $r_z = 0.0$, $a_x = a_y = a_z = 1.0$, $\alpha = \beta = \gamma = 0.0$, $p = 0.0$;
- random simulation path;
- maximal number of neighbors: $N = 24$;
- distance threshold: $t = 0.05$;
- probability constraints:

  11) global constraints, 10 classes of values: regular subdivision of interval $[0, 256[$, target PDF: uniform, comparing PDFs with MLikR2, deactivation distance: 2.0, constant threshold: 0.0

  12) local constraints, 2 classes of values: $[0, 50[$ and $[50, 256[$, target PDF: map for probabilities for the class of small values in Fig. 12a, support radius (distance): 12.0, comparing PDFs with MLikR2, deactivation distance: 4.0, constant threshold: 0.01

- no block data;
- maximal scanned fraction of the TI: $f = 0.5$;
- tolerance (on relative error): $tol = 0.0$;

- post-processing:

  11) no post-processing;

  12) one post-processing path with the default parameters.

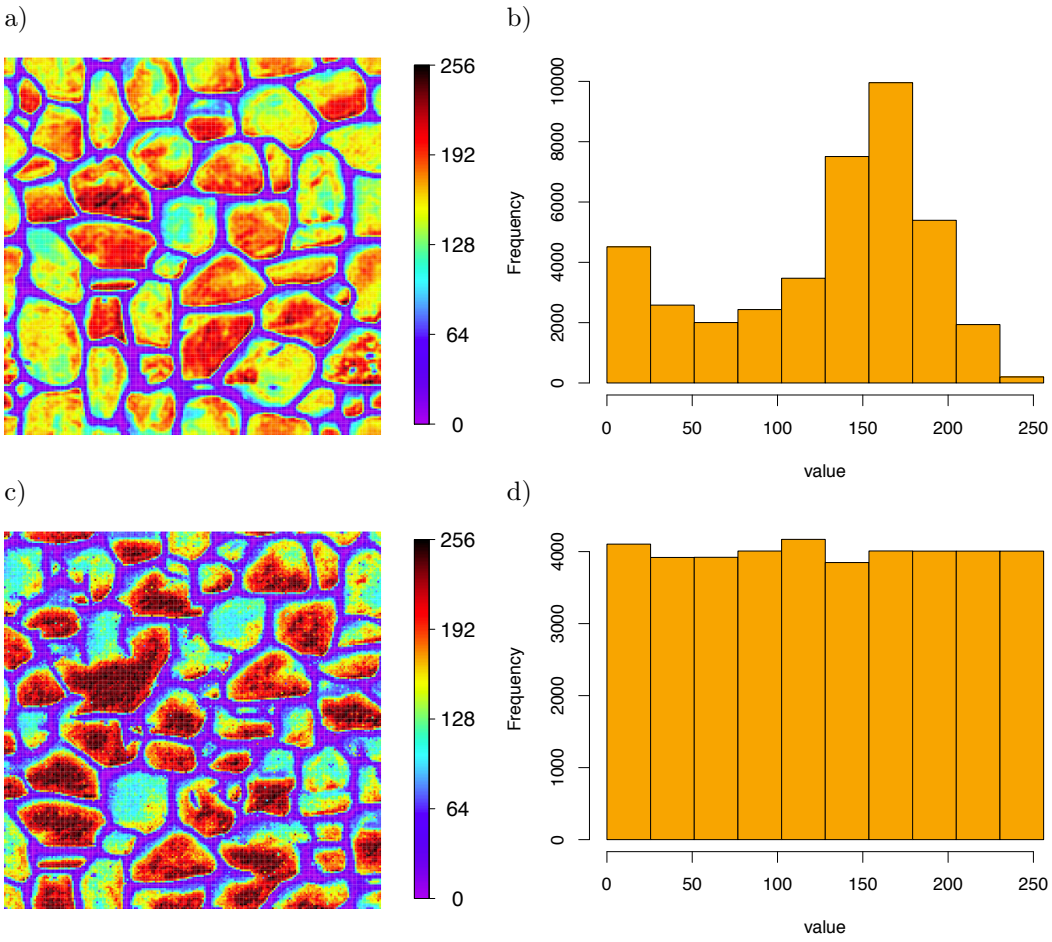

Fig. 11 DeeSse example 11: global probability constraints on a continuous variable. a) TI, $Nx = Ny = 200$, $Nz = 1$, one continuous variable (values in $[0, 256[$); b) histogram of values in TI, with a subdivision in 10 regular classes of values; c) simulation with global probability constraints, $Nx = Ny = 200$, $Nz = 1$, obtained by using the input parameters (11) given in section 14; d) histogram of values in simulation, with a subdivision in 10 regular classes of values; (target proportions: uniform distribution on the 10 regular classes)

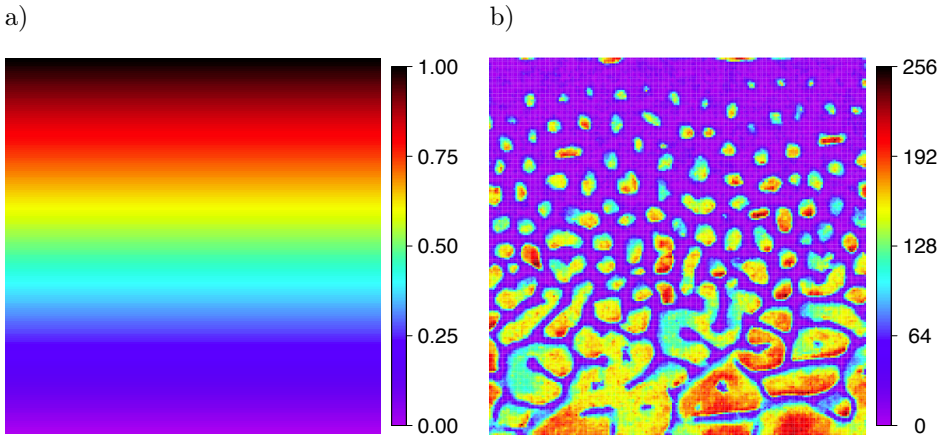

Fig. 12 DeeSse example 12: local probability constraints on a continuous variable. The TI used is shown in figure 11a. Two classes of values are considered: $[0, 50[$ and $[50, 256[$. a) probability (defined on the SG) for the class of small values ($[0, 50[$); b) simulation with local probability constraints shown in a) and a support radius of 12.0 (nodes), $Nx = Ny = 200$, $Nz = 1$. The input parameters (12) for the simulation are given in section 14

15 Example 13: simulations conditioned to block data

The usage of block data is illustrated in this section. The TI used for the examples is shown in Fig. 13a. Two cases are considered. In the first case (Fig. 13b), input block data (target mean value) are given on blocks of 10×10 pixels covering the whole SG without overlapping, and a threshold for block data of 0.5. In the second case (Fig. 13c), six overlapping block data of different sizes (support) are given in input and a threshold for block data of 0.02. Comparison of block data (target mean value on blocks) and mean on blocks computed on the simulation for both cases is shown in Fig. 14. Both plots shows that the block constraints are well respected according to the threshold used for block data. The main input parameters for both cases are given below.

Main input parameters for example 13 (Fig. 13).

- SG: $Nx = 240$, $Ny = 120$, $Nz = 1$, ($Sx = 1.95312$, $Sy = 1.5625$, $Sz = 1.0$ as for TI, $Ox = Oy = Oz = 0.0$);
- one simulation variable, continuous;
- unconditional (no punctual data);
- no homothety, no rotation;
- distance type 1 (L_1);
- one TI of size $Nx = 764$, $Ny = 239$, $Nz = 1$, with $Sx = 1.95312$, $Sy = 1.5625$, $Sz = 1.0$ (Fig. 13a);
- search neighborhood: $r_x = 200.0$, $r_y = 115.0$, $r_z = 0.0$, $a_x = a_y = a_z = 1.0$, $\alpha = \beta = \gamma = 0.0$, $p = 0.0$;
- random simulation path;
- maximal number of neighbors: $N = 32$;
- maximal density of neighbors: $MD = 1.0$;
- distance threshold: $t = 0.02$;
- no probability constraint;
- block data (mean target value):

  13.1) 288 blocks of 10×10 pixels, not overlapping (Fig. 13b); block data threshold: 0.5 (always activated);

  13.2) 6 blocks of different sizes, overlapping (Fig. 13c); block data threshold: 0.02 (always activated);

- maximal scanned fraction of the TI: $f = 0.25$;
- no post-processing.

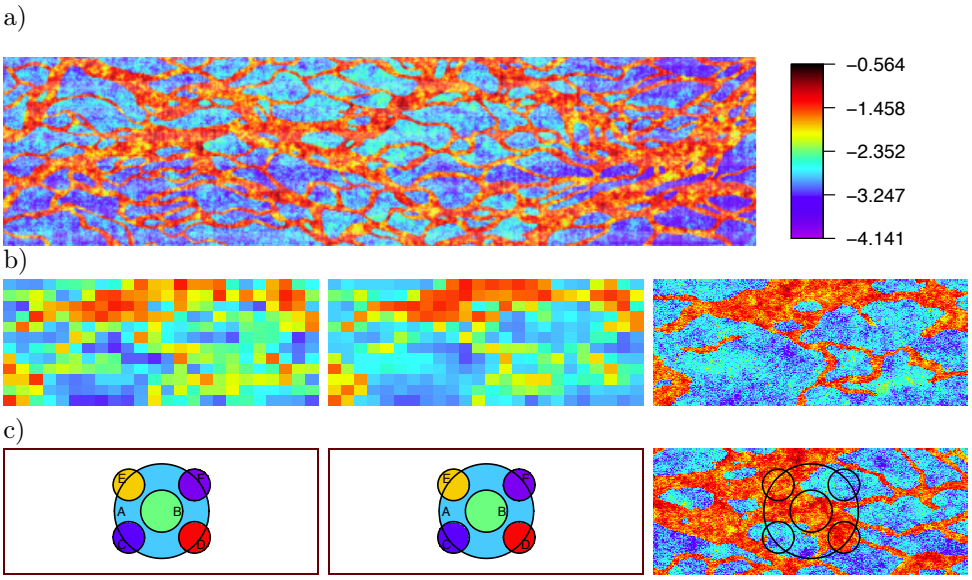

Fig. 13 DeeSse example 13: simulations conditioned to block data; a) TI, $Nx = 764$, $Ny = 239$, $Nz = 1$; b) example 13.1, from left to right: input block data, mean over the blocks for the simulated image, simulated image; c) example 13.2, from left to right: input block data, mean over the blocks for the simulated image, simulated image. The size of the SG is $Nx = 240$, $Ny = 120$, $Nz = 1$, and for the TI and the SG: $Sx = 1.95312$, $Sy = 1.5625$, $Sz = 1.0$, and $Ox = Oy = Oz = 0.0$. An exaggeration of 4/3 is used for the simulations (b and c). The color bar is used for all images. The input parameters for both cases are given in section 15

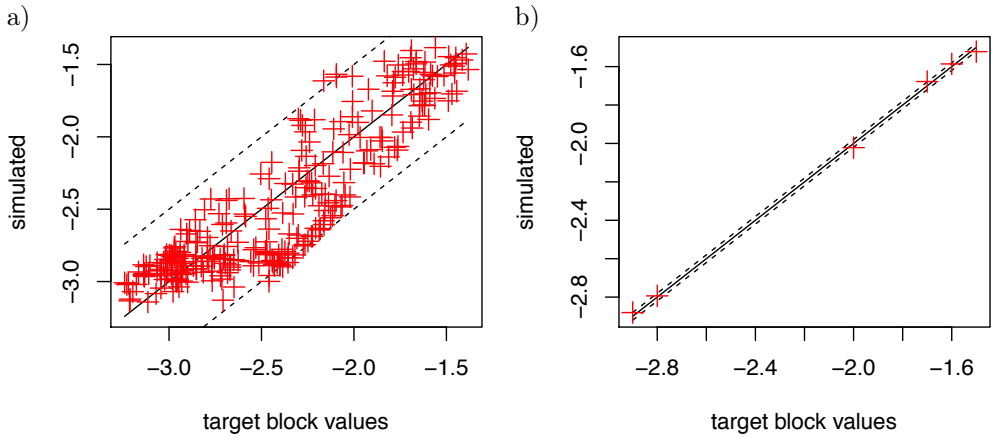

Fig. 14 DeeSse example 13: comparison of block data (target mean value on blocks) and mean on blocks computed on the simulation for a) example 13.1 shown in figure 13b; b) example 13.2 shown in figure 13c. The solid line is the line of equation $y = x$ and the dashed lines are the lines of equation $y = x \pm t$, where t is the block data tolerance used (same tolerance on each block)

References

G. Mariethoz, P. Renard, and J. Straubhaar. *A deterministic version of the multiple point geostatistics simulation / reconstruction method with the simulated / reconstructed values are directly taken from the training images without prior estimation of the conditional*, Publication number: US8682624 B2, Publication date: Mar 25, 2014, Application number: US 13/108,393, Applicant: University of Neuchâtel (CH), 2014.

G. Mariethoz, P. Renard, and J. Straubhaar. The Direct Sampling method to perform multiple-point geostatistical simulations. *WATER RESOURCES RESEARCH*, 46, NOV 20 2010. DOI: 10.1029/2008WR007621.

E. Meerschman, G. Pirot, G. Mariethoz, J. Straubhaar, M. Van Meirvenne, and P. Renard. A practical guide to performing multiple-point statistical simulations with the Direct Sampling algorithm. *COMPUTERS & GEOSCIENCES*, 52:307–324, MAR 2013. DOI: 10.1016/j.cageo.2012.09.019.

J. Straubhaar, P. Renard, and G. Mariethoz. Conditioning multiple-point statistics simulations to block data. *SPATIAL STATISTICS*, 2015. Submitted.

To table of contents