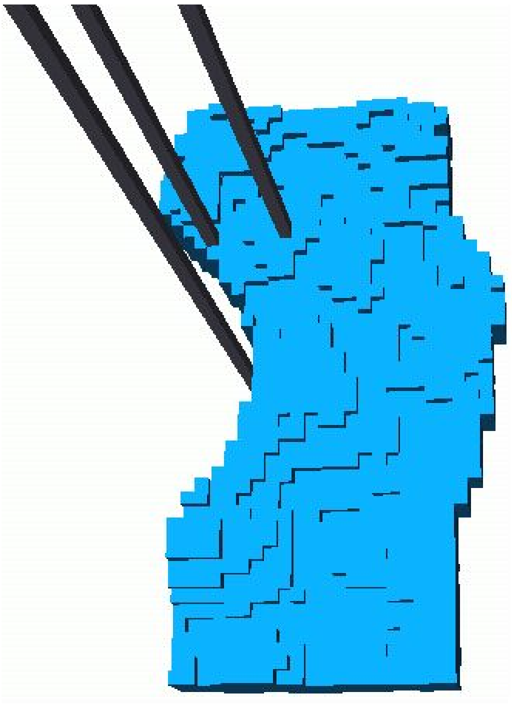
Norsk Regnesentral
NORWEGIAN COMPUTING CENTER

Note

Local update of object models in RMS



Note no SAND/11/07

Authors Anne Randi Syversveen, Frode Georgsen and Ragnar Hauge

Date October 2007

Norsk Regnesentral

Norsk Regnesentral (Norwegian Computing Center, NR) is a private, independent, non-profit foundation established in 1952. NR carries out contract research and development projects in the areas of information and communication technology and applied statistical modeling. The clients are a broad range of industrial, commercial and public service organizations in the national as well as the international market. Our scientific and technical capabilities are further developed in co-operation with The Research Council of Norway and key customers. The results of our projects may take the form of reports, software, prototypes, and short courses. A proof of the confidence and appreciation our clients have for us is given by the fact that most of our new contracts are signed with previous customers.

Roxar

Roxar is a leading international provider of products and associated services for reservoir management and production optimization.

Roxar has signed a three-year agreement with Norsk Regnesentral to further develop stochastic reservoir modelling, reduce uncertainty and maximize reservoir performance.
Roxar is a leading technology solutions provider to the upstream oil and gas industry.
With this agreement, NR will use Roxar as an exclusive commercialization partner for reservoir modelling software and, in return, Roxar will commit to a certain volume of purchased research projects.
Roxar and NR have cooperated on developing the modelling software Irap RMS for many years. NR has been responsible for development of all methods and algorithms currently used for 3D property modelling in Irap RMS.

Roxar and NR are also partners in the research program Fieldwatch which is financed by Norsk Forskningsråd, Statoil and Roxar. The presented project is a part of this program.

Title	**Local update of object models in RMS**
Authors	**Anne Randi Syversveen, Frode Georgsen and Ragnar Hauge**
Date	October
Year	2007
Publication number	SAND/11/07

Abstract

The note describes an algorithm for local updating of an existing facies realizations the RMS modules Facies:Composite and Facies:Channels. The update will correct an existing realization for new wells, and, for Facies:Composite also gives the possibility to change the realization in a specified area.

Keywords	Object models, local update, well conditioning
Target group	Users of object models in RMS
Availability	Open
Project number	642024, 643029
Research field	Stochastic reservoir description
Number of pages	21
© Copyright	Norsk Regnesentral

Contents

1 Introduction

In this note, we describe an algorithm for local updating of an existing Facies:Composite or Facies:Channels realization in RMS. For both modules, the idea is to start with an existing facies realization, and correct it for new well data. This new well data can be new wells that are drilled, changes in the location or logs in existing wells or new specification of coupling between wells. In addition for Facies:Composite it is possible to add and remove objects in a predefined area of the reservoir. This option is not invoked for Facies:Channels since the channel objects typically runs through the reservoir and therefore are not limited to a specific area.

The simulation algorithm can be summarized as follows: First, read the realization which is going to be updated. Then, do local changes to fulfill well conditioning. Finally, for Facies:Composite, do simulation as before starting from the current realization, but restrict changes to a given area.

Examples of using the algorithm to update realizations conditioned on new well information are shown for both modules.

Finally, some suggestions to further developments that will possibly improve the algorithms are given.

2 Changes in model file

A new key word 'RESTART' in the model file indicates that we will do local updating of an existing facies realization. The parameter that must be given in this command is the file name for the parametric file of the realization which is going to be updated. In addition three floats can be given (optionally), representing tolerances in well data. When old well data are compared to new data, they can differ by these given numbers in x-, y- and z direction and still be considered to be the same observations. Finally, for Facies:Composite, a trend parameter can be given (optionally). This gives the specification of the area which is going to be changed and is a 2D or 3D parameter of zeros and ones, where the ones give the area to be changed. When no area parameter is given, only local changes to fulfill well conditioning are done.

The format for the model file differs between Facies:Channels and Facies:Composite. A detailed desription on how the format must be is given in Appendix A.

It is possible to change some model parameters (e.g. size and shape of objects) in the updating model. This might be most relevant in Facies:Composite when an area for resimulation is specified. In any case this should be done with care in order to get a smooth behaviour where the changed and unchaged parts of the object are joined together.

3 Writing and reading parametric file format

All information necessary to recreate the objects in the realization must the stored in the parametric file format. This includes geometry of each object and well observations. All well observations in the realization are stored, and for each object, the conditioning observations are identified. Parameters used during simulation are not stored. They will be given in the model file as before. This is for instance the mean and standard deviations in the distributions for width, thickness or center point position of objects. It is therefore possible for a realization to 'wake up' in a different model than the realization originally was generated from. But object types in Facies:Composite can not be changed.

The parametric file format used is the same as is used by the gridder (paramFaciesRealToGrid). Some information not influencing the gridder is also written to the realization file. This is module specific global data and object related data that is needed when the local update is performed. In the parametric file format these blocks of information are framed by the keywords MODULE_START and MODULE_END.

4 Conditioning algorithm

In Facies:Composite, only three object types can be locally updated in order to fulfill well conditioning. That is, Axial, Angular and Backbone objects. These object types have flexible edges represented by Gaussian random fields which can be changed locally according to new well data. If well data are in conflict with other object types, the objects are removed, and new objects are created to fulfill conditioning.

In the following we denote objects present in the original realization as existing objects.

The conditioning is divided in five steps. (i) Identify which observations that are conditioned to existing objects and which are not and also which observations that are in conflict with existing objects. (ii) Deal with couplings involving existing objects. (iii) Assign all unconditioned observations to existing objects, if possible. Local changes should be as small as possible, and we try to include unconditioned observations in existing objects if possible. (iv) Simulate local changes in existing objects such that well conditionings are fulfilled. (v) Simulate new objects to condition observations that are still unconditioned.

4.1 Identifying unconditioned observations and conflicts

When reading the parametric realization, the well observations must be compared to the wells given in the model file in order to identify unconditioned observations. Observations from the old realization which are not present in the current wells in the model file are not taken into account.

At the initial state observations that are in conflict with the original realization must be identified. This is done while reading the objects from the parametric realization file. The objects might have 2D Gaussian fields defining top and bottom. Storing these fields for all objects is too memory demanding. Due to that, one object at the time is read from file and checked against the observations in order to find if there are conflicts. The type of conflicts that can occur is background facies penetrating an object, wrong facies type or discrepancy between the size of the object and the size of the well observation. In case a conflict is found, the conflict is marked, and the 2D-fields for the object are deleted from memory before the next object is read. If an object goes through the update procedure without being changed, the original 2D-fields are used, by reading them from the realization file again. If the object is changed, new 2D-fields are simulated for the whole object, even if just a part of it is changed.

4.2 Coupling of observations

Before we assign the unconditioned observations to objects, we must check if observations of existing objects are defined to be in contact or not contact with other observations in the update model.

Since the objects in Facies:Channels generally do not have limited extension within the reservoir, adding or removing objects might have more dramatic consequences with respect to reservoir property and volume fraction than in Facies:Composite. This means that in the conditioning, Facies:Channels tends to find solutions that keep old objects to a greater extent than Facies:Composite, and accordingly tends to be more restrictive in creating new objects.

In order to handle couplings in a consistent and robust way, the following situations and solutions have been identified:

1. Two or more observations in an existing object are specified to not be in contact. In Facies:Composite the object is removed. In Facies:Channels the object is kept, but only one of the observations in the group consisting of illegal contacts are kept in the object and the other are identified as conflict observations. Two observations can be specified to not be in contact even if this is not stated explicitly if they are both specified to not be in contact with a third observation.

2. An observation in an existing object is specified to be in contact with observations in one or more other existing objects. All the involved objects are removed. Two observations can be specified to be in contact even if this is not stated explicitly if they are both specified to be in contact with a third observation.

3. An unconditioned observation is specified to be in contact with one or more observations of an existing object. In Facies:Composite, the unconditioned observation is assigned to the existing object as long as it is not also specified to not be in contact with other observations in the existing object. In that case rule 1 applies, and the object is removed. In Facies:Channels, a more refined approach is applied. The number of observations in contact with the unconditioned observation (positive couplings) and the number of observations not in contact with the unconditioned observation (negative couplings) are found. If the number of positive couplings is greater than or equal to the number of negative couplings, the unconditioned observation is assigned to the object and the negative coupled observations are removed from the object. Otherwise, the unconditioned observation is not assigned to the object and the positive coupled observations are removed from the channel.

If whole objects are removed, or single observations are removed from objects, the ‘homeless’ observations are classified as unconditioned observations. In Facies:Composite these will be used to generate new objects, while they in Facies:Channels might be assigned to other objects in an iterative process (see below).

4.3 Assigning observations to objects

All unconditioned observations must be assigned to an existing changeable object, if possible. To decide which object the observation should be assigned to, we measure the distance from the observation to the object. In Facies:Composite, the distance to axial and backbone objects are only defined if the observation can be projected to the center line of the object, because the length of the objects are not changed in order to fulfill well conditioning. The distance is given as the sum of weighted squared distance from the center line of the object in y and z direction. The weight factor is one over the variance of the edge and height of the object, respectively. For angular objects, the distance is always well defined, as the edge can change locally around the whole object. For all other object types in Facies:Composite, the distance is not defined, because the objects can not be changed locally.

In Facies:Channels the distance is defined as a weighted distance from the *expected* (not the actual) position of the channel. For channel observations the distance is measured to the expected edge of the channel, and for crevasse observations the distance is to the expected edge of a crevasse. The weigths are defined in the same way as above.

An observation is assigned to the closest object that allows this observation in its couplings. For the unconditioned observations not assigned during the coupling step 3 above, we calculate the distance to all objects that the observation can possible be assigned to, that is objects of same facies. All distances are stored in a table indexed by observations and objects. Observations in contact with each other have a common distance, given as the distance from the observation furthest away. If the distance is not defined, or the observations are specified to not be in contact with an observation of the object, the distance is set to be missing.

When assigning observations to objects, we start by finding the smallest distance in the table, and the corresponding observation and object. Then the observation is assigned to the object, and the distances from this observation to all objects are set to missing. We go on by finding the smallest distance, and from now we must check that the observation can be in contact with the observations already assigned to the actual object. If not, the distance is set to missing, and we go on until all distances in the table are missing. Then all observations which can possibly belong to an existing object, are assigned to an object.

In Facies:Channels one restriction is made regarding the assigning of crevasse observations to channels. The number of crevasses belonging to a channel from the original realization is not changed during the update. This means that a new crevasse observation will not be included in a channel unless it is possible to assign it to already existing crevasses for this channel.

4.4 Simulate local changes

When all unconditioned observations are assigned to an object, we start perturbating the objects one by one. The observations assigned to the object, and those in conflict with the object, together decide which parts of the object that must be changed. We choose to change within one range to each side of these observations affecting the object. If the ranges for the different parameters defining the shape and size of the object differs, the longest range is chosen. This gives one or more intervals where new Gaussian random fields for the object position are drawn. The nodes in the 1D grids are drawn sequentially. The object position outside the change intervals define conditioning points for the drawing of the new position together with the conditioning points defined by the new constraining information. Legal intervals for the edge are found according to the observations. With high flexibility (large variance) on the edges, the conditioning will be easier to fulfill. Low flexibility on edges can make it impossible to condition the observations.

If it is impossible to create an object conditioned on the observations assigned to it, the observation furthest away from the original object position is removed, and the object is then redrawn. If observations are coupled, all observations in the coupling must be removed at the same time. In Facies:Composite the removed observations define new objects. In Facies:Channels they are inserted back to the list of unassigned observations. It is marked that these observations should not be assigned to the current channel in future iterations. If, after a certain number of trials, it is impossible to fulfill the conditioning for a channel, the whole family is removed from the realization, and all belonging observations are inserted to the list of unassigned observations.

Facies:Channels performs an iterative procedure in order to assign as many unconditioned observations as possible to existing objects. This means that if the list of unassigned observations change during the redrawing of a channel (for instance because an observation is removed from the channel) or the family is removed, the procedure for assigning observations and simulating local changes starts all over. This is because the assigning of observations will change, and thereby also potentially which channels and which parts of the channels that are influenced. This iterative process goes on until the list of unassigned observations is empty, or until a predescribed number of iterations are performed.

In both Facies:Composite and Facies:Channels objects with no unconditioned observations assigned to it, and with no conflicts with wells, are kept unchanged after local update. The only exeption from this is in Facies:Composite when the reference point lies inside the restart area.

4.5 Creating new objects

At the end of the algorithm, new objects are created to condition observations that are not captured by existing objects.

5 Facies:Composite

5.1 Area parameter

When an area parameter is given, all objects with reference point inside the specified area are changed. They are removed before simulation. All objects outside the area are kept unchanged, except necessary changes according to well conditioning. Well conditioning must always be fulfilled, also outside the restart area. If no area parameter is given, the realization is only corrected for well data, and no further simulation is run.

5.2 Examples

In the following, we show some simulation examples. In the first example, we look at an axial object, which initially is conditioned to only one well (Figure 1). Then we do two different updates. First, two wells with observations of the object facies penetrating the object are added.

After updating (middle of the figure), only Gaussian fields on top and bottom of the object are changed, and conditioning of all three wells is fulfilled. In the second update, one of the wells contains only background facies, and must be avoided. The result is shown to the right in the.

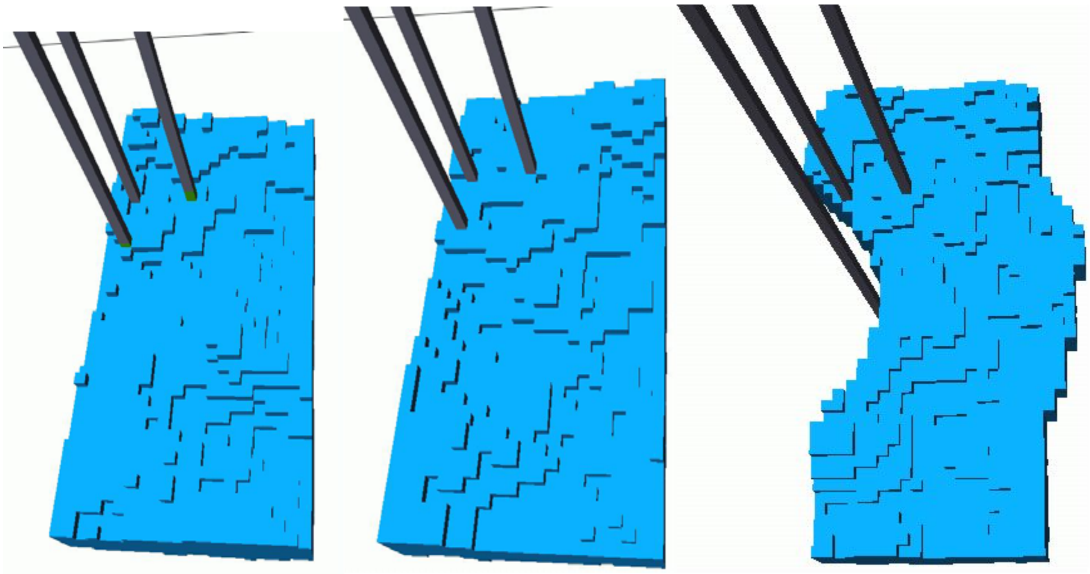


Figure 1: Axial object before (left) and after (middle and right) local update. Before update, the object is only conditioned to one well, see the mismatch with two wells to the left. After update (middle), the object conditions correctly all three wells. In the right update, one of the wells has only background facies, and must be avoided.

In the second example, we look at a backbone object. In Figure 2, the well paths before and after local update are shown. We see that we have got a new facies observation in well 5, and want to update the object to condition to this observation. We see the result in Figure 3. The right edge is extended to cover well number 5.

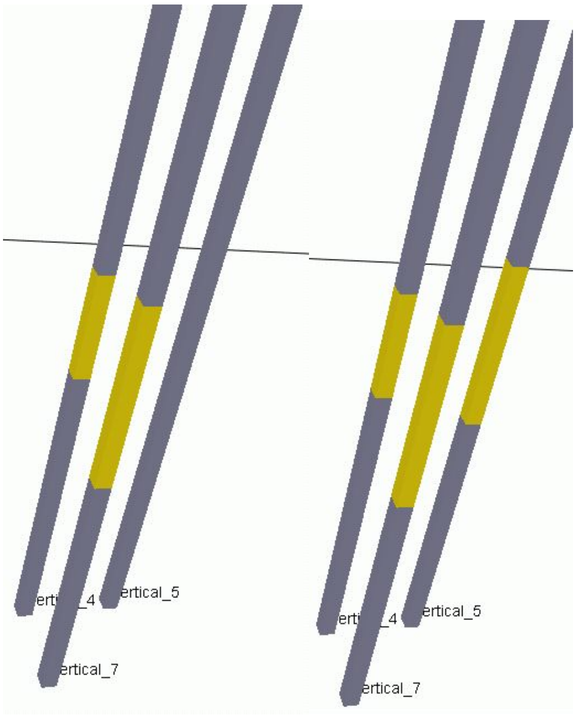


Figure 2: Well paths before (left) and after (right) local update in Figure 3.

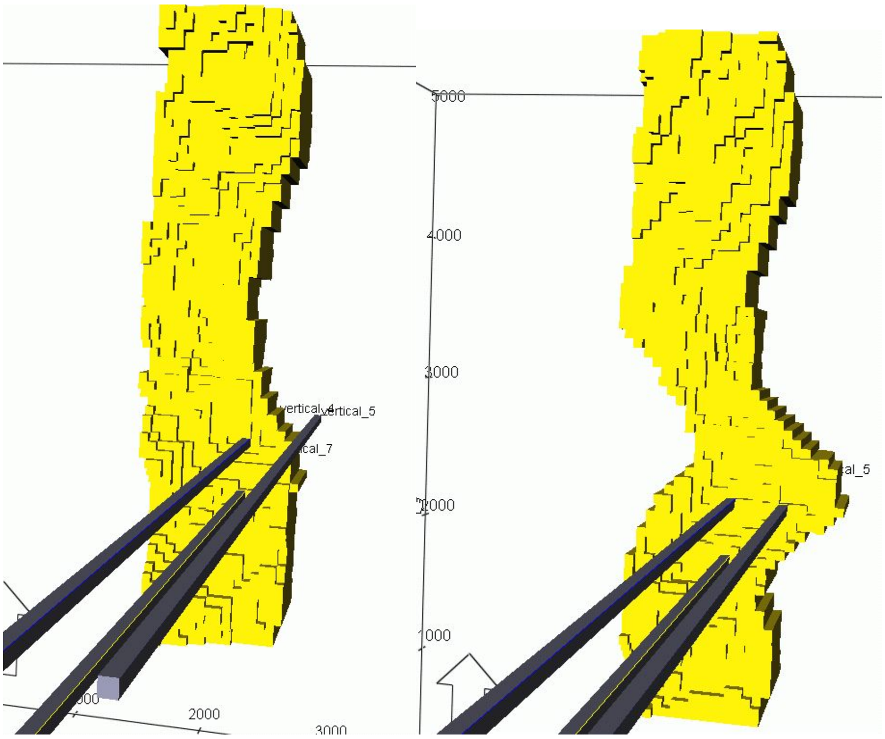


Figure 3: Backbone object before (left) and after (right) local update. The well paths before and after update are shown in Figure 2.

In Figure 4, we show an example with a radial object. Before update, the object is conditioned only to well number 4. After update, we see that the object is extended to cover well number 7, and the Gaussian random fields on top and bottom are changed to condition the observation in well number 5.

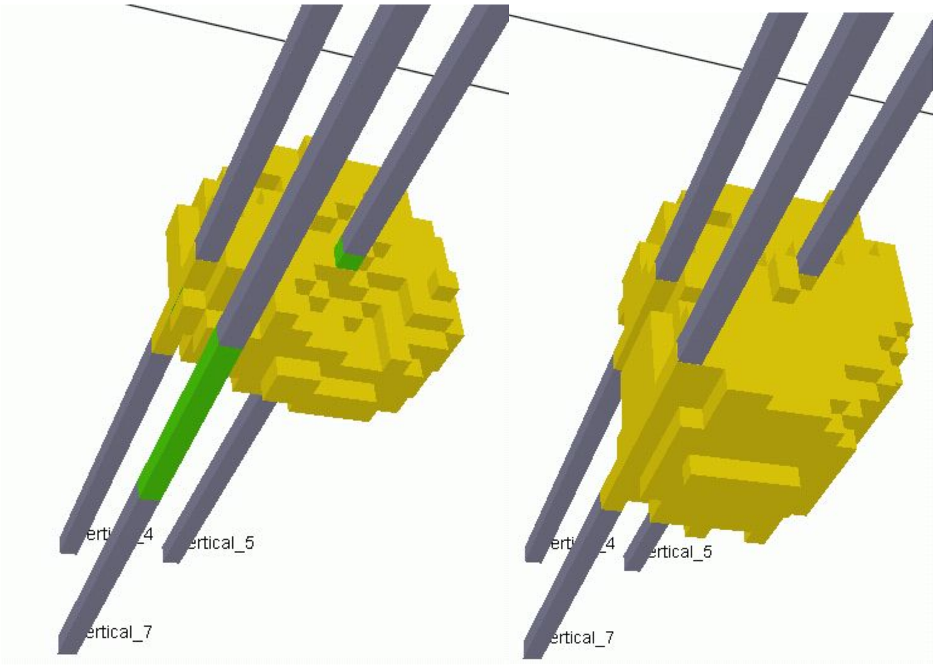


Figure 4: Radial object before (left) and after (right) update. Before update, the object is conditioned to only well 4.

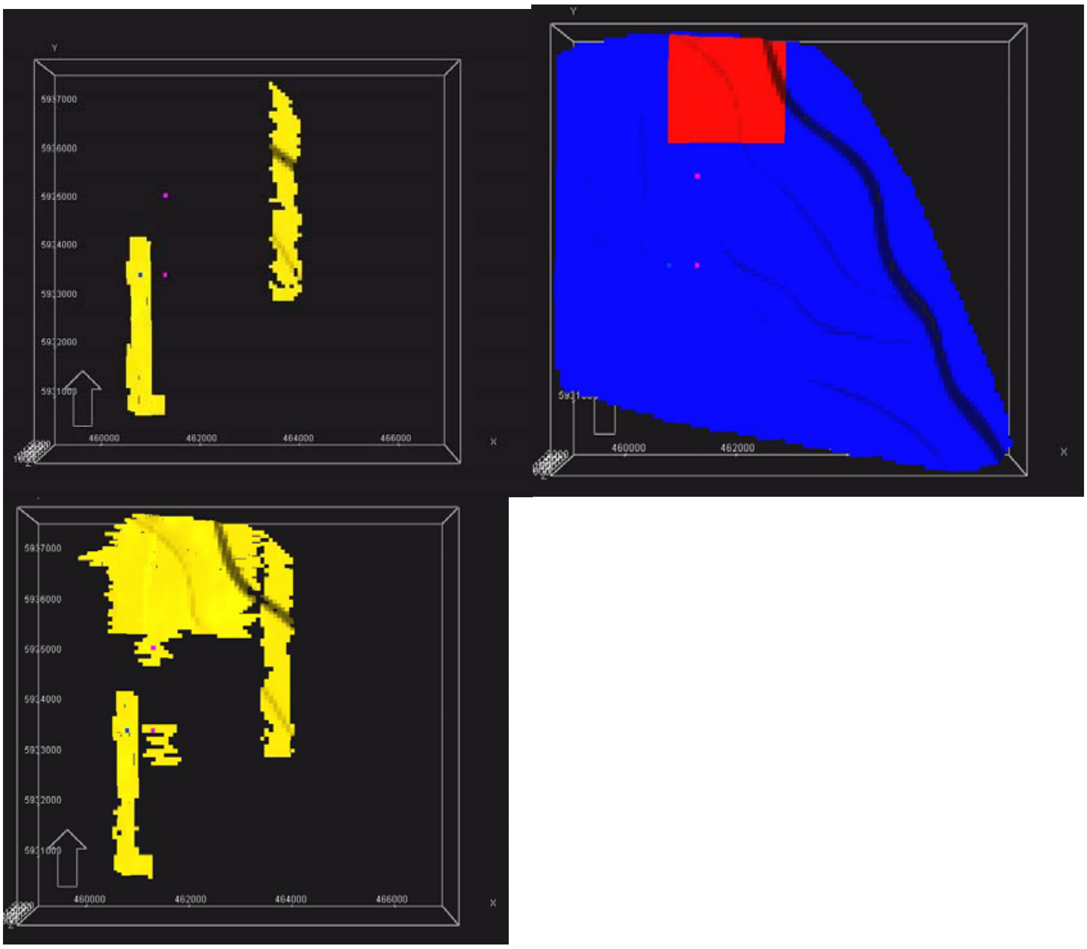


Figure 5: Example on use of restart area. Top left, we see the objects to be updated. To the right, we see the restart area in red. At bottom, the updated realization is shown, where new objects are added in the restart area and on well observations.

In Figure 5, we see an example on use of restart area. Initially, we have only three objects, shown top left in the figure. New wells are shown in pink. When doing local update, we want to change the red area in the picture top right in the figure. The bottom picture shows the updated realization. All old objects are kept, since their reference points are outside the red area. The red area is filled with objects, according to the given intensity function. The new wells are conditioned on, and two objects outside the red area have been necessary here. None of the existing objects have been able to condition on these observations, possible because of low flexibility of edges. Note that the new objects are shorter than the old ones. Old and new objects do not necessarily have the same size distribution, but they must have the same shape.

6 Facies:Channels

Examples of updating two different original realizations will the shown. The first, shown in Figures 6-9, has two facies, channels and background, while the other, shown in Figures 10-12 in addition has crevasses.

6.1 Example of updating a realization with channels.

The following figures show in birds view an example of an original realization that has been updated three times sequentially. The original realization (Figure 6) is conditioned on two wells (original_1 and original_2). The three wells that will be used to update the realization are also shown in the figure. In Figure 7 the realization has been updated after the inclusion of well update_1 which has a channel observation. The figure shows that the red channel has included this observation. The next update is performed on the realization from Figure 7 including conditioning on well update_2 which contains only background observation. As Figure 8 shows, the red channel has changed location in order to avoid this well. In the last example a new well (update_3) containing channel observation is included. In addition the constraint that the observation in update_1 and update_3 belongs to the same channel is applied. This realization is shown in Figure 9 showing that the green channel is left untouched, while the original red channel is removed (since the conditioning was not possible to fulfill) and two new channels are added, one (the red) conditioning wells update_1 and update_3 and another (yellow) conditioning well original_1.

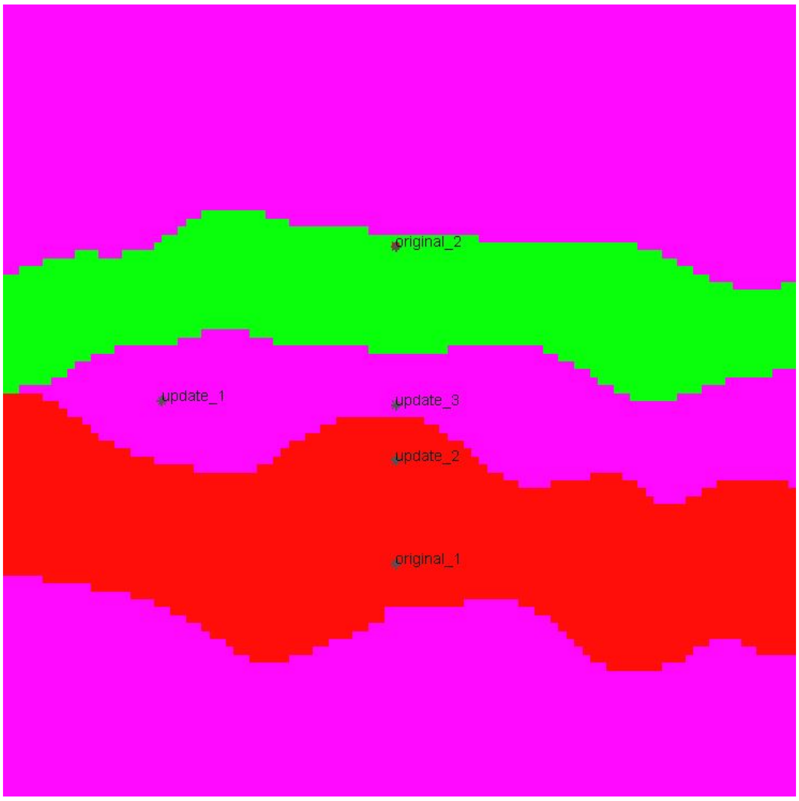


Figure 6: Original realization with two channels conditioned on wells original_1 and original_2 respectivally

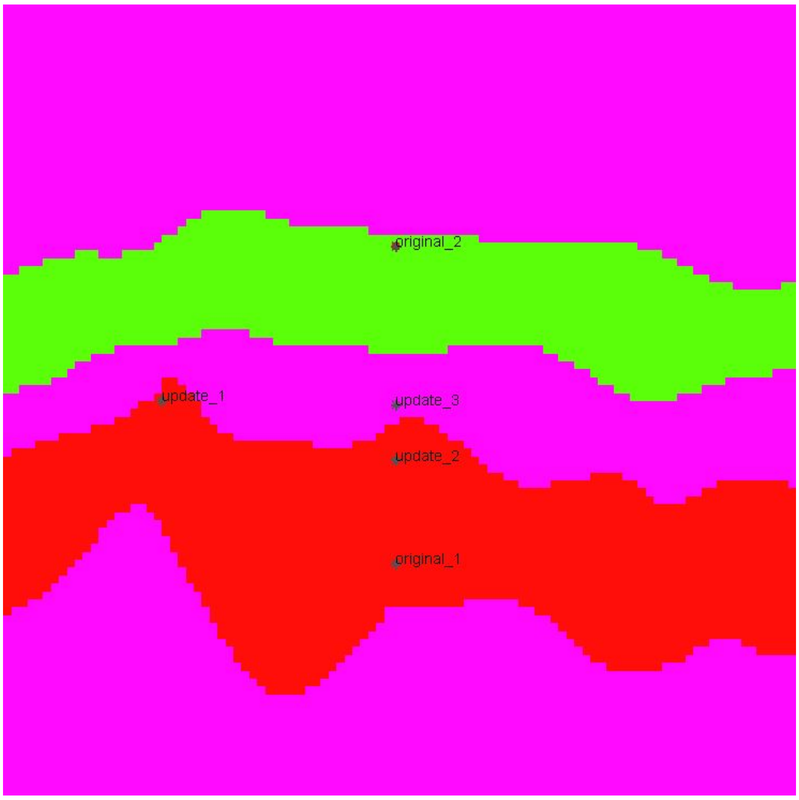


Figure 7: One well (update_1) with channel observation is added. After update the red channel has included this observation and has changed position near the well and remains unchanged further away (to the right).

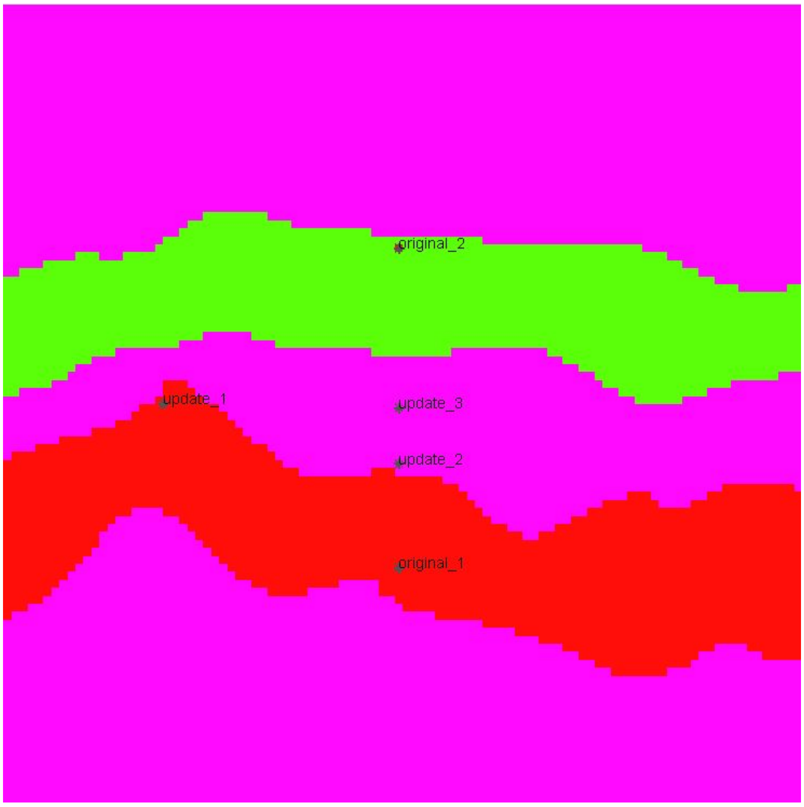


Figure 8: The realization from figure 7 updated after well update_2 with only background is included.

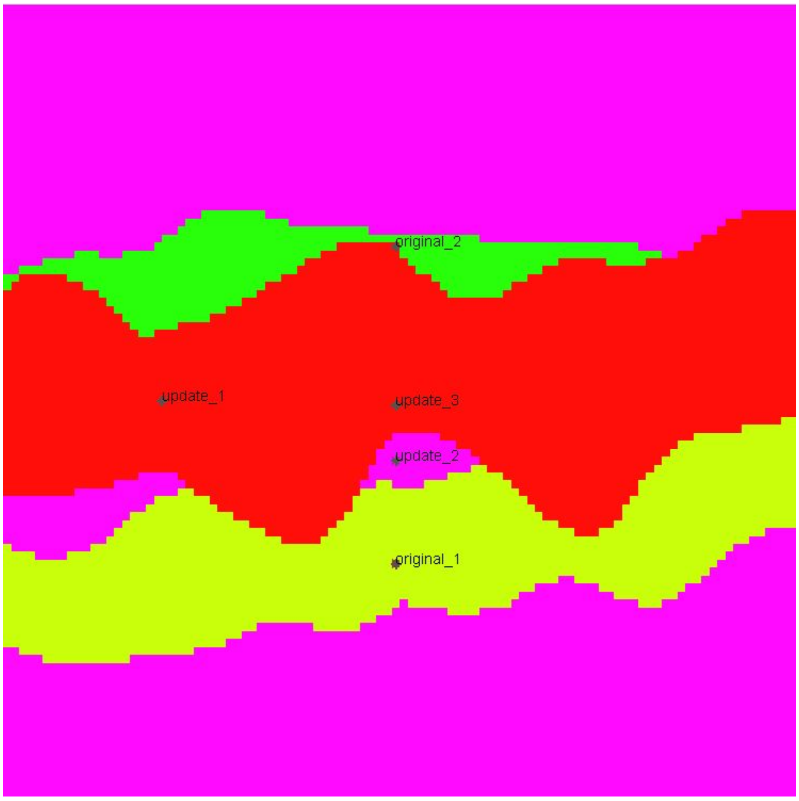


Figure 9: Another well (update_3) with channel observation is included and surely coupled to well update_1. One channel from the original realization is removed and substituted by two new channels fulfilling the conditioning constraints.

6.2 Example of updating a realization with channels and crevasses

An original realization (Figure 10) is conditioned on one well (original_1) containing a crevasse observation. The channel is green, with one crevasse (red) conditioning the well. The two other wells, used to update, are also shown. In Figure 11 the original realization is conditioned on a well (update_2) that contains only background. It penetrates the crevasse in the original realization, but avoids it after the update. Note that both the channel and crevasse position is changed around the well since they are closely dependent on each other. In Figure 11, the realization from the previous figure is conditioned on another well (update_4). This well contains crevasse observation, but penetrates the channel in the input realization. In the updated realization the channel, now yellow, avoids this well, and the crevasse conditions now also the new well.

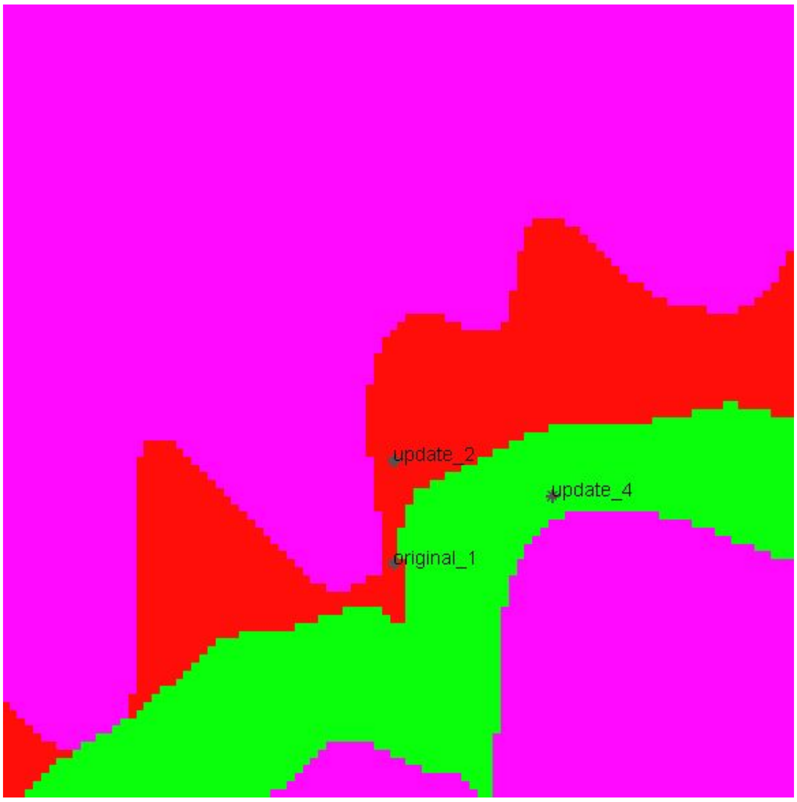


Figure 10: Channel (green) with crevasse (red) conditioned on one crevasse observation in well original_1.

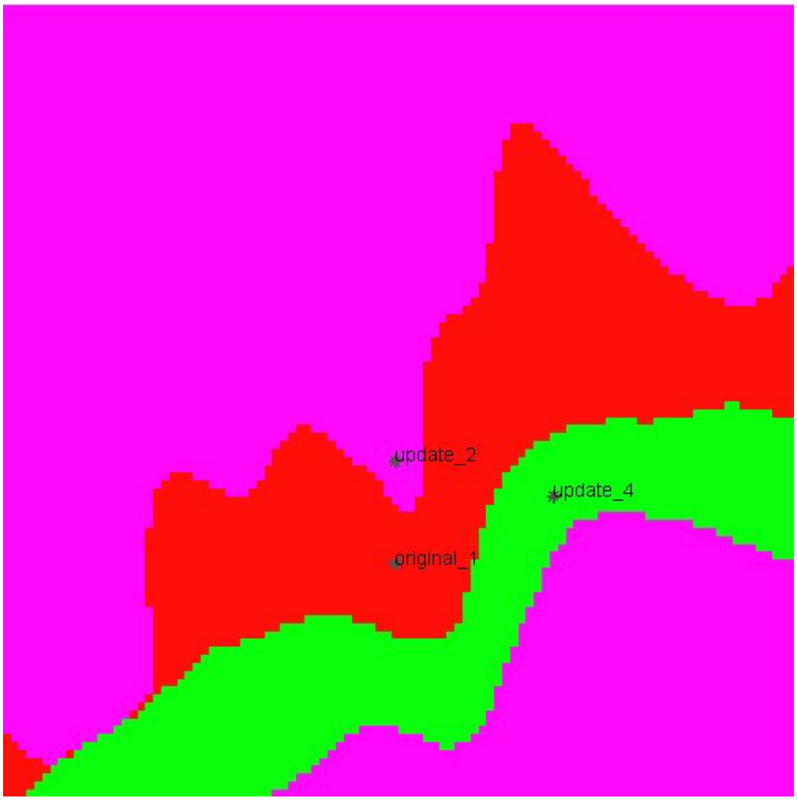


Figure 11: One well, update_2, containing only background is added. The crevasse and channel avoids this well.

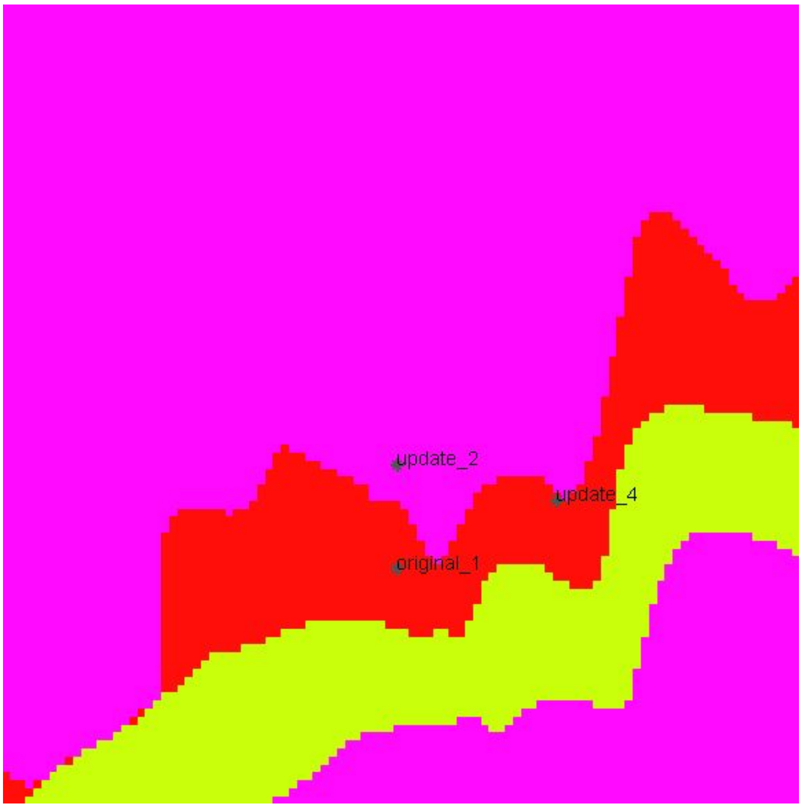


Figure 12: Another well, update_4 , is added containing crevasse observation which the updated crevasse includes.

7 Possible future developments

Experience when using the method presented here on synthetic and real studies will certainly reveal a need for improvements and developments. An obvious matter that might be crucial is that the algorithm does not account for changes in volume fraction in the updated realization. When a realization has converged to a desired volume fraction it is reasonable to suppose that one will not have an updated realization that differs too much in volume fraction from the initial target. On the other hand, new well information might indicate that the target volume fraction should be adjusted. Within the scope of this project it has not been possible to account for these considerations.

For Facies:Channels it is obvious that the way new crevasse observations are treated is not optimal. Such observations should influence the number of crevasses attached to the channel and eventually update this number in order to condition the observation. In a reservoir with a high proportion of crevasses the presented method might generate too many channels.

Experience in using the method will probably reveal that the rules for dealing with coupling between observations need to be changed or refined, in order to give a more optimal updated realization.

A. Appendix – Format of model file

Below follows the format used to initiate a local update.

A.1. Facies:Composite

RESTART

Description: Optional command. Do local update of an existing realization. Five input parameters, the four last are optional.

1. Parametric file.

   Description: Full path of parametric file of the realization to be restarted.

2. Well observation tolerance parameters

   Description: 3 decimal numbers describing the tolerance in well observation in x- y- and z- direction. When an old and new well observation differs less than the tolerance in all three directions, they are regarded as the same observation. Optional. If not given, default values 0.1, 0.1 and 0.01 are used.

3. Update area

   Description: A basic trend function of type Grid-1D, Grid-2D or Grid-3D with no transformations, consisting of zeros in locations where the realization should be unchanged and ones in locations where the realization should be updated. Optional.

A.2. Facies:Channels

RESTART

1. <restart-file(string)>

   Description: The name of the parametric realisation file from the realisation that is to be updated.

   Legal values: A string without any blanks or commas.

2. [x-tol-limit(real)]

   Description: The tolerance (in absolute value) of the displacement of well positions in x-direction in the update realisation compared to the original. Displacement bigger than this value will define a new well.

   Legal values: Real number > 0.0.

Default values: 0.1

3. [y-tol-limit(real)]

Description: The tolerance (in absolute value) of the displacement of well positions in y-direction in the update realisation compared to the original. Displacement bigger than this value will define a new well.

Legal values: Real number > 0.0.

Default values: 0.1

4. [z-tol-limit(real)]

Description: The tolerance (in absolute value) of the displacement of well positions in z-direction in the update realisation compared to the original. Displacement bigger than this value will define a new well.

Legal values: Real number > 0.0.

Default values: 0.1

Example of file format:

```
RESTART
<restart-file(string)> old_param_real.fett
[x-tol-limit(real)] 1.0
[y-tol-limit(real)] 1.0
[z-tol-limit(real)] 0.05
;
```