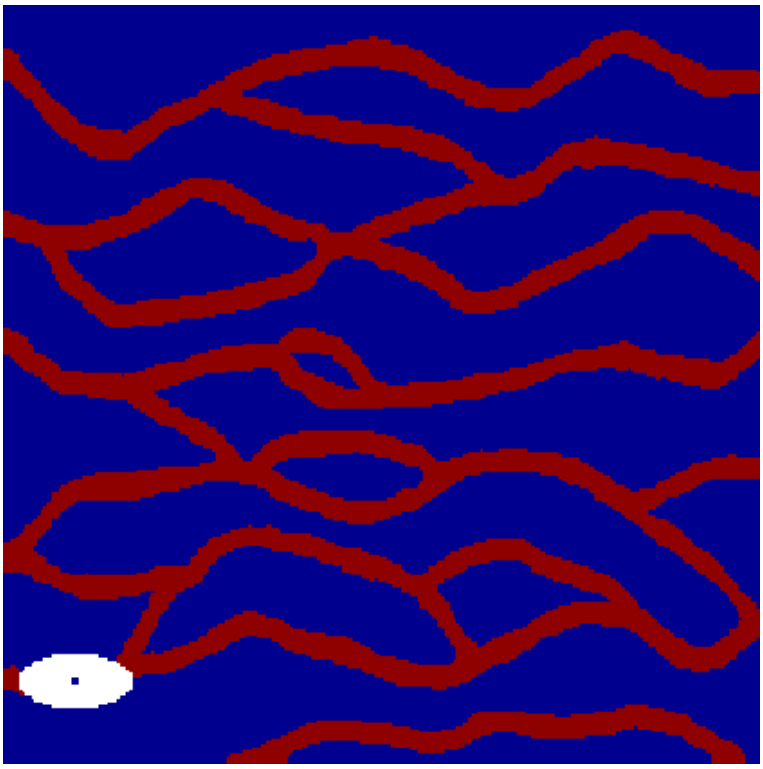
Norsk Regnesentral
NORWEGIAN COMPUTING CENTER

Note

Local update of S-GeMS realizations: The `local` utility



Note no SAND/10/07
Author Harald H. Soleng
Date 31st October 2007

Norsk Regnesentral
NORWEGIAN COMPUTING CENTER

Norwegian Computing Center

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.

Norsk Regnesentral
Norwegian Computing Center
Postboks 114, Blindern
NO-0314 Oslo, Norway

Besøksadresse
Office address
Gaustadalléen 23
NO-0373 Oslo, Norway

Telefon · telephone
(+47) 22 85 25 00
Telefaks · telefax
(+47) 22 69 76 60

Internett · internet
www.nr.no
E-post · e-mail
nr@nr.no

Title	**Local update of S-GeMS realizations: The `local` utility**
Author	**Harald H. Soleng**
Date	31st October 2007
Publication number	SAND/10/07

Abstract

This document is a how-to for doing local updates in S-GeMS. It also describes the use of the `local` program for generating local update input to S-GeMS.

Keywords	Facies modelling, local updates, multipoint
Target group	S-GeMS users and developers
Availability	Open
Project	Multipoint
Project number	808002
Research field	Facies modelling
Number of pages	9

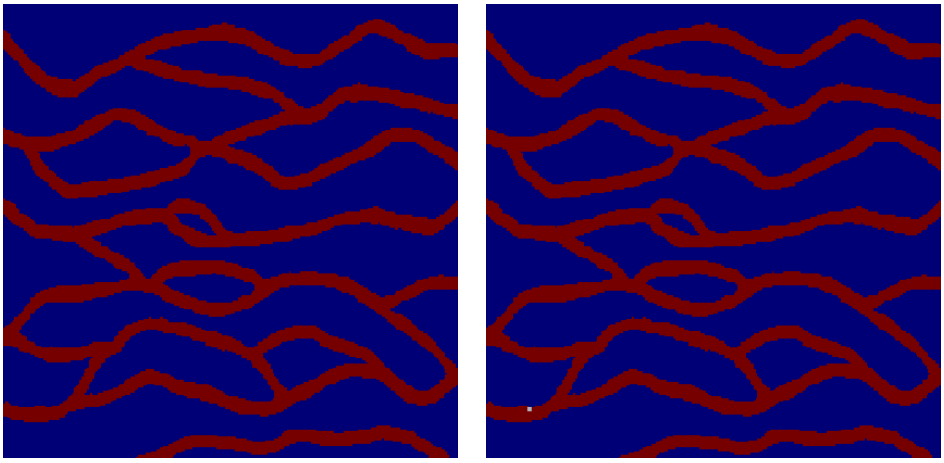
Contents

List of Figures

1 Introduction

There is often need to be able to modify a realization only slightly to accomodate new data. In S-GeMS this can be done by treating part of the old realization as well data and then updating only parts of the realization.

Consider the realization in Figure 1(a). A new well is drilled and comes in with conflicting new data, c.f. Figure 1(b). In this situation one could resimulate the whole real-



(a) Input realization (b) Conflicting well data

Figure 1. Original realization and conflict location. On the left we have the original realization and on the right the position of conflicting new well data is indicated.

ization, or one could take out only a small part around the conflict and resimulate.

2 Conflict finder

A small utility program called `local` has been written to generate the resimulation region on a format understood by S-GeMS. The utility reads in the input realization and all well data. It identifies all conflicts. Then an ellipsoid is digged out around each conflict point. Finally the result it converted to a point data file and exported as well data. The result is shown in Figure 2.

2.1 Required input

The program takes the following input:

Input realization file name: on geoeas grid format (number of grid cells are assummed to be present in the first line of the header on the format $N_x \times N_y \times N_z$.

Grid size information: grid size (in meters) in all directions.

Well data file name: on geoeas point data format.

Restart radii: One radius in each direction (x, y, and z).

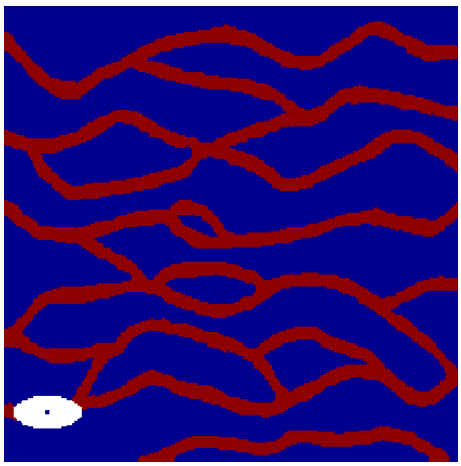


Figure 2. Update ellipsoid.

Output file name: on geoeas point data format.

2.2 Natural extensions

Possible additional input could include

Number of grid cells/grid resolution information: Since this is not part of the official geoeas file format header, it could be given by the user as separate xml attributes.

Azimuth and dip: It would be natural to allow the user to orient the local update ellipsoides.

2.3 Location of source files

The program uses NRlib and Boost. The source files of the program are located at the NR file system at
`/nr/project/sand/Multipoint/software/local_update`.

2.4 Input file format

The program takes input on ascii xml format.

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<?xml-stylesheet type="text/css"
href="http://intern.nr.no/sand/css/evaluate.css"?>
<local>
  <title>Example</title>
  <name>Local update</name>

  <logfile>test.log</logfile>

  <action type="updateregion">

    <inputfile format="GeoEAS">
```

```
      data/channelTI.dat
    </inputfile>

    <size_x> 250 </size_x>

    <size_y> 250 </size_y>

    <size_z> 1 </size_z>

    <well format="GeoEAS">
      data/well.dat
    </well>

    <radius_x> 20 </radius_x>

    <radius_y> 10 </radius_y>

    <radius_z> 0 </radius_z>

    <outputfile format="GeoEAS">
      restartinput.dat
    </outputfile>

  </action>
</local>
```

3 Simulation results

After loading the training image and the point data file generated by the `local` program into the snesim algorithm in S-GeMS, we got the updated realizations of Figure 3. In nine out of ten realizations, the algorithm was able to connect the channel divided by the local update ellipsoid. This opens up the possibility of using local updates as an automatic repair algorithm to fix problematic regions provided these problematic regions can be identified automatically.

4 Conclusion

Local updates are easily done in S-GeMS software provided the input realization is converted to a point data set with a suitable region digged out around conflicts. Then, using the result as well data, the simulation proceeds as usual. A utility for file conversion has been implemented to help with this task.

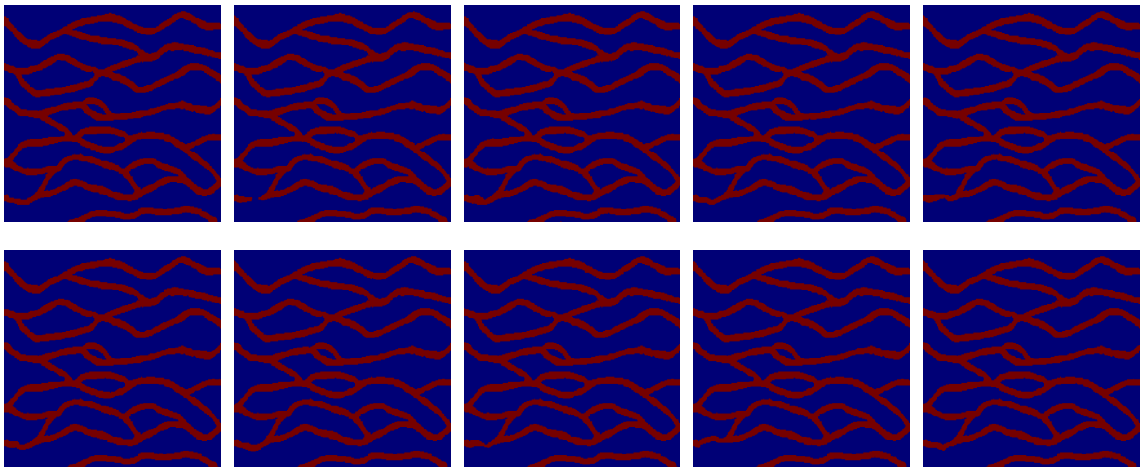


Figure 3. Ten local update snesim realizations.

The effectiveness of local updates in snesim opens the possibility of using this as a postprocessor to fix and repair problematic regions in generated realizations. This could be automated provided these regions could be identified automatically.