Optimization of the Super Block Search for Scattered Data

S. Lyster and C.V. Deutsch

Searching for nearby data to condition local estimates and models of uncertainty is required by many geostatistical algorithms. Fast searching is important for manageable CPU time. A spiral search is used for data on a regular grid, but we are often required to find data that are not regularly sampled. The super block search is well established for scattered data. It is a type of search tree. A tuned *super block search is considered to be the fastest approach to locate nearby scattered data. This paper addresses the issue of optimizing the parameters in the super block search. The use of optimized parameters leads to an improvement over default parameters taken in most geostatistical software.*

Introduction

Kriging and simulation are widely accepted and applied techniques in spatial prediction. At times, there is a need to create large 3-D models and computer time becomes an issue. The computer intensive operations are (1) searching for relevant data, (2) constructing the kriging system of equations, and (3) solving the kriging equations. Disk access and other overhead may also be concerns. The focus of this paper is in the first operation – searching for relevant data.

Most kriging and simulation algorithms consider a limited number of nearby conditioning data. The main reason for this is to improve speed. The CPU time required to solve a kriging system increases as the number of data cubed, e.g., doubling the number of data leads to an eightfold increase in CPU time. Furthermore, adopting a global search neighborhood would require knowledge of the covariance for the largest separation distance between the data. The covariance is typically poorly known for distances beyond one-half or one-third of the field size. A local search neighborhood does not call for covariance values outside the search ellipsoid.

Another reason for a limited search neighborhood is to reduce the consequence of choosing a global search. A local search allows local rescaling of the mean when using ordinary kriging and other techniques. All of the data may have been pooled together to establish a reliable histogram and variogram; however, at the time of estimation it is often better to relax the decision of stationarity locally and use only nearby data.

A number of constraints are used to establish the nearby data that should be considered: (1) only those data falling within a search ellipsoid centered at the location being estimated are considered, (2) the allowable data may be further restricted by a specified maximum, and (3) a maximum per drillhole/well or maximum number per octant may also be imposed to ensure reasonable data availability surrounding the location being considered. The closest data are retained. Closeness is measured by the anisotropic Euclidean distance. In certain cases, the variogram distance is used for searching. At times the number of data available within a particular search specification is used to assign a measure of confidence in the results.

The super block search used by the GSLIB set of programs automatically sets the number of super blocks as the number of grid cells in a dimension divided by two, to a maximum of 50 super blocks in any one dimension. This is a fast and easy method for setting the search parameters but is probably not optimal for CPU efficiency. Guidelines for better selection of the number of super blocks are proposed here.

Super Block Search

The super block search is arguably the most efficient search for scattered data. A number of search trees and other algorithms have been implemented for spatial searching, but it is difficult to improve on the speed of the super block search. The super block requires some preprocessing of the data, but that improves the overall speed when considering many locations in the domain being estimated/simulated. The central ideal of the super block search is that data are partitioned into a grid network *super*imposed on the field being considered. When estimating any one location, it is then possible to limit the search to those

data falling in nearby *super blocks*. This search has been adopted for non-gridded data in most kriging and simulation programs.

The data are classified and ordered according to a regular network of grid blocks; see Journel and Huijbregts, 1978, page 361. This grid network is not the same as the grid network of points/blocks being estimated or simulated. Typically, the size of the search network is much larger than the final estimation or simulation grid node spacing. When estimating any one point, only those data within nearby super blocks are checked. A large number of data are thus quickly eliminated because they have been classified in super blocks beyond the search limits. This is illustrated in 2D on Figure 1, where a super block grid network has been established over an area containing scattered data.

The key parameters in establishing the super block search is the number of blocks in X, Y, and Z: nsb_X, nsb_Y and nsb_Z. The user specifies these parameters or they accept the default settings. The values are specified in the GSLIB code, but few people modify or optimize these parameters. The aim of this paper is to come up with a response surface that gives the optimum numbers based on the number of data and search distance.

The illustration to the right on Figure 1 shows the area of interest when estimating a point anywhere within the dark gray super block, only those data within the dark black line need be considered. Note that all search resolution less than the size of a super block has been lost. Also note that the light gray region is defined by the search ellipse (circle in this case) with its center translated to every node to be estimated within the dark gray super block. All super blocks intersected by the light gray region must be considered to ensure that all nearby data are considered for estimation of any node within the central dark gray super block.

The first task is to build a template of super blocks, centered at the super block that contains the node being estimated. For example, the template is the relative locations of all 21 blocks enclosed by the dark line on Figure 1 (right). With this template, the nearby super blocks are easily established when considering any new location.

A second important computational *trick* is to sort all of the data by super block index number. Each super block is indexed from 1 to nsb_X x nsb_Y x nsb_Z. using a 1-D index (see GSLIB book). An array (of size equal to the total number of blocks in the super block network) is constructed that stores the cumulative number of data for each super block and all super blocks with lesser block indices, i.e., $c(i) = \sum_{j=1}^{i} nisb(j)$

where *nisb(j)* is the number of data in super block j, and *c(0)=0*. Then the number falling within any super block i is *c(i)-c(i-1)* and their index location starts at *c(i-1)+1*. Therefore, this one array contains information on the number of data in each super block and their location in memory.

Simulation Parameters

There are four parameters that affect the time required for simulation and will be considered here:

1. Number of cells in the grid (nx, ny, and nz)
2. Number of data to search for (ndat or maxdat)
3. Search radius parameterized by a_x, a_y and a_z
4. Number of super blocks (nsb_x, nsb_y, and nsb_z)

The number of super blocks used in GSLIB programs (KT3D, SGSIM, etc) is currently tied to the number of cells in the model. This is a good rule of thumb as the cell size, and therefore number of cells, should be selected to account for the data spacing. However, in terms of measuring CPU efficiency the time required for simulation (or estimation) increases with the number of cells in the grid regardless of the number of super blocks used. Similar statements may be made about the number of data retained and the search radius; retaining more data at further distances is a decision made based on the data configuration and range of correlation. These decisions will impact both the quality of the simulation and the time required to generate realizations; the number of super blocks is a parameter that does not change the results.

2D Data Set

A 2D data set with 140 data points was used to determine the effect of super blocks on the CPU time for simulation. Figure 2 shows location maps of the data and the grid domain, with 10 (left) and 50 (right) super blocks superimposed. The data extents are 50m x 50m. The smaller super block size has one datum per block. A number of cases were simulated with a modified version of SGSIM (Deutsch and Journel, 1998) that allows the user to define the number of super blocks. The parameters were varied as follows:

1. The number of cells were varied from nx = ny = 100, 200, and 400
2. The number of data to retain from the search was ndat = 8, 16, 24, and 32
3. The search radii were set as $a_x = a_y = 10$, 20, and 40
4. The number of super blocks in the grid were $nsb_x = nsb_y = 1$, 2, 5, 10, 25, 50, 100, and 200

Of these parameter sets, 24 different cases with varying numbers of super blocks were simulated with 100 realizations in each. Figure 3 shows a graph of the number of super blocks on a side of the grid vs. the simulation time required, with all 24 cases represented by lines. In Figure 3 it may be seen that the optimal number of super blocks for this data set is a low number, with the best for individual cases ranging from 1 to 25; in many cases the simulation times using 1, 2, 5, and 10 super blocks are within a second of one another. Using longer search radii, fewer super blocks is a slightly more efficient choice.

The simulation times are very similar for low numbers of super blocks. This suggests that for models with few data the number of super blocks is largely inconsequential and it is nearly as efficient to just search all of the data (corresponding to one super block). When too many super blocks are used the search over all blocks within range becomes more cumbersome than simply considering the individual data points.

3D Data Set

Data are not random or two-dimensional. They are organized along drillholes and wells. An example data set was used that contains 3303 data points in 62 vertical wells. The data extents are approximately 10,500m x 10,500m x 400m. Aerial views of this data with 5 and 20 super blocks are shown in Figure 4 and Figure 6 shows cross-sectional views with 20 super blocks horizontally and 5 and 20 vertically. The data are spaced approximately every 1000m horizontally and 1m vertically.

1. The number of cells were varied from nx = ny = 41 and 61, and nz = 40, 80, and 125
2. The number of data to retain from the search was ndat = 8, 16, and 24
3. The search radii were frozen as $a_x = a_y = 3000$ and $a_z = 100$
4. The number of super blocks in the grid were $nsb_x = nsb_y = 1$, 10, 20, 30, and 40 with $nsb_z = 10$, and $nsb_z = 1$, 5, 10, 20, and 40 with $nsb_x = nsb_y = 20$

The simulation times for 10 realizations in each case are shown in Figure 5 for varying X and Y super blocks; and Figure 7 for varying Z super blocks. The most efficient number of super blocks in the X and Y directions appears to be in the 10-20 range with little to no difference within this range in all cases. This corresponds to super blocks with sizes of 50m – 100m.

For the cases that vary the number of super blocks in the Z direction, the optimal number of superblocks is about 10 in all cases. This means the optimal super block size is about 40m, or 40 data intervals. There are on the order of several hundred super blocks containing data for 20 x 20 x 10 super blocks in the grid; searching these blocks is more efficient than searching through thousands of data locations or super blocks.

Conclusions

Ever increasing computer speed has removed focus from improving the speed of geostatistical algorithms; however, constructing large 3-D geostatistical models remains computer intensive and taking all reasonable steps to improve speed is important. For data sets that have scattered points the best super block size is of the same order of magnitude as the data spacing; for strings of data the optimal super block size is that

which reduces the number of distance calculations from thousands (the number of data) to hundreds (the number of informed super blocks).

To set up an optimal search the size of the super blocks could be varied, keeping the anisotropy constant, until the number of super blocks containing data is in the hundreds rather than thousands. If there are only a few hundred data points than optimization of the super block search parameters will most likely not be worth the computational effort required.

References

Deutsch, C. V., and Journel, A.G. (1998) *GSLIB: Geostatistical Software Library*, Oxford University Press, New York.

Deutsch, C. V. (2002) *Geostatistical Reservoir Modeling*, Oxford University Press, New York.

Journel, A.G., and Huijbregts, Ch. (1978) *Mining Geostatistics*, Academic Press.

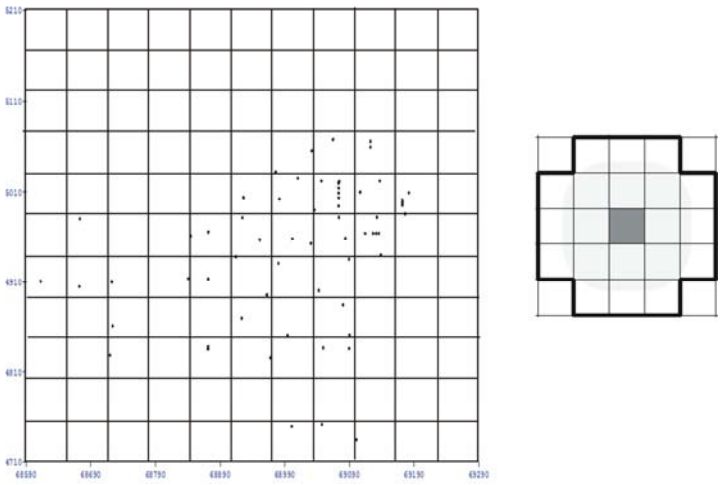

Figure 1: Illustration of a super block grid network over some data (left) and the relevant super blocks to consider searching (right).

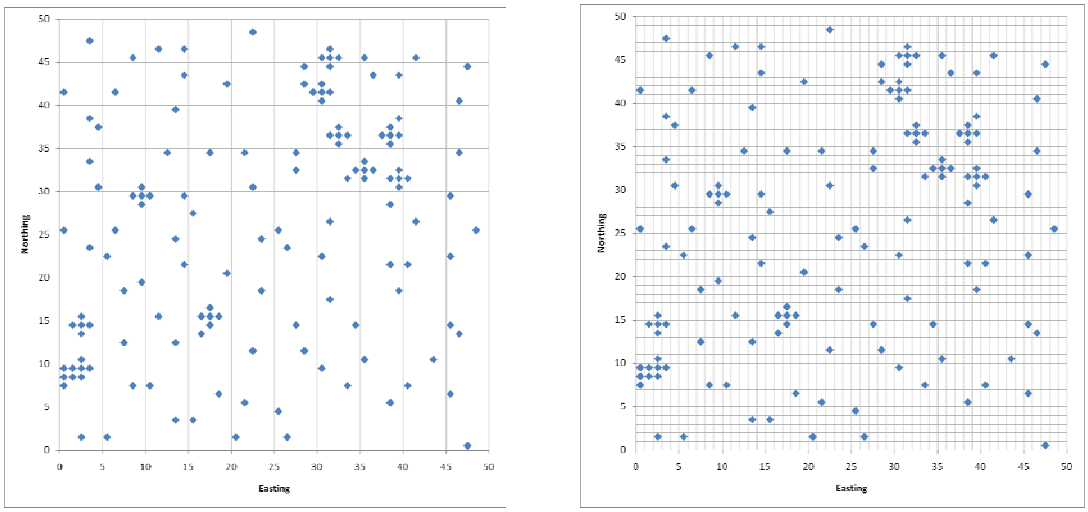

Figure 2: Location map of the 140 2D data. Left: 10 super blocks in the X and Y directions; right: 50 super blocks in the X and Y directions.

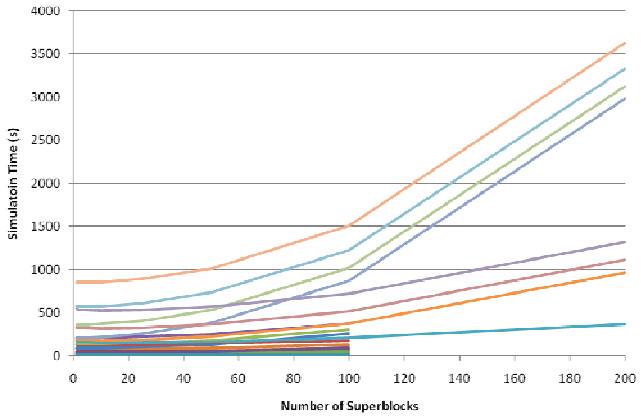

Figure 3: Simulation times for 100 SGSIM realizations using the 2D data. The horizontal axis is the number of super blocks in the X and Y directions. Each data series has all other parameters fixed.

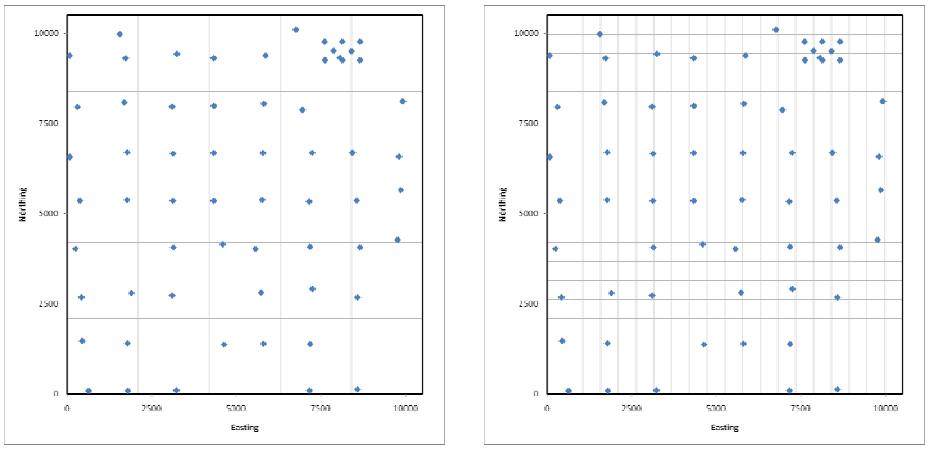

Figure 4: Areal location map of the 3D well data. Left: 5 super blocks in the X and Y directions; right: 20 super blocks in the X and Y directions.

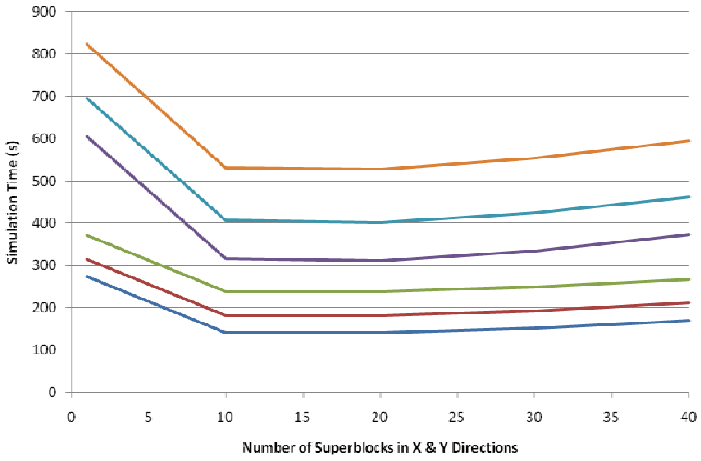

Figure 5: Simulation times for 10 SGSIM realizations using the 3D data. The horizontal axis is the number of super blocks in the X and Y directions. Each data series has all other parameters fixed.

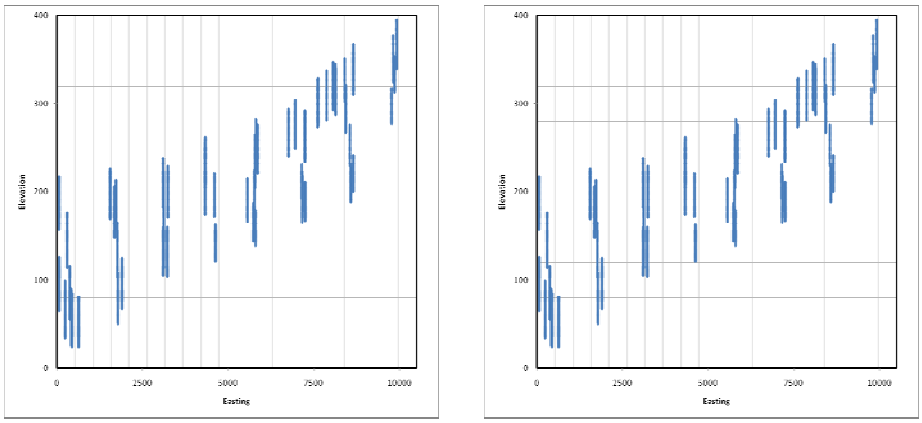

Figure 6: Vertical location map of the 3D well data. Left: 20 super blocks in the X and Y directions and 5 in the Z direction; right: 20 super blocks in the X and Y directions and 10 in the Z direction.

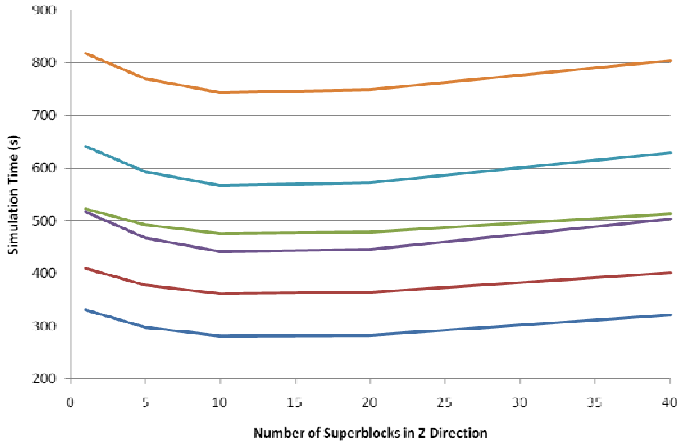

Figure 7: Simulation times for 10 SGSIM realizations in all of the cases using the 3D data. The horizontal axis is the number of super blocks in the Z direction. Each data series has all other parameters fixed.