Pergamon

Computers & Geosciences Vol. 24, No. 8, pp. 765–769, 1998
© 1998 Elsevier Science Ltd. All rights reserved
Printed in Great Britain
0098-3004/98 $ - see front matter

PII: S0098-3004(98)00066-1

GENERATING SPATIALLY CORRELATED FIELDS WITH A GENETIC ALGORITHM

YAKOV PACHEPSKY[1,2]* and DENNIS TIMLIN[1]

[1]USDA-ARS Remote Sensing and Modeling Laboratory, BARC-WEST, Beltsville, MD 20705, USA, and [2]Duke University Phytotron, Duke University, Durham, NC 27708, USA

(Received 14 August 1997; revised 29 March 1998)

Abstract—Generated realizations of random fields are used to quantify the natural variability of geological properties. When the realizations are used as inputs for simulations with a deterministic model, it may be desirable to minimize differences between statistics of sequential realizations and make the statistics close to ones specified at generating the realizations. We describe the use of a genetic algorithm (GA) for this purpose. In unconditioned simulations, statistics of the GA-generated realizations were significantly closer to the input ones than those from sequential Gaussian simulations. Distributions of generated values at a particular node over sequential realizations were close to the normal distribution. The GA is computationally intensive and may not be suitable for fine grids. The sequential Gaussian algorithm conditioned with GA-generated values on a coarse grid can produce a set of realizations with similar statistics for the fine grids embedding the coarse one. © 1998 Elsevier Science Ltd. All rights reserved

Code available at http://www.iamg.org/CGEditor/index.htm

Key Words: Geostatistical simulations, Random fields, Genetic algorithm.

INTRODUCTION

Spatial correlation is an important feature of geological data. To use a knowledge of spatial correlations, geostatistical stochastic simulations are used that employ some input probability distributions and semivariograms of geophysical variables and provide random fields of these variables with probability distributions and semivariograms similar to the input ones (Deutsch and Journel, 1992). Several geostatistical simulation techniques are developed (Gotway and Rutherford, 1994).

Realizations of random fields generated with the existing techniques may have probability distributions and semivariograms that are substantially different from the input distributions and semivariograms. This is considered as an advantage when the 'original statistics are inferred from sparse samples and cannot be deemed exactly representative of the population statistics' (Deutsch and Journel, 1992). However, if the input statistics are considered to be exact, it is correct to look for an algorithm that would reproduce accurately the input statistics in each realization (Deutsch and Journel, 1992). The latter situation is commonplace, for example, in subsurface transport simulation studies when the goal is to estimate mass transport applying a deterministic transport model in a porous media with preselected statistical properties (Mackay and others, 1996; Tsang and others, 1996). In this instance, all realizations of the random field should have probability distributions and semivariograms that are close to the input ones. Grindrod and Impey (1993) generated fields having power-law semivariograms and used a constrained optimization technique to obtain fields that have given mean and variance within a specified tolerance. Their technique involved the use of a random-number generator and so the generated fields were nonunique. Each realization in an ensemble was a plausible representation of the 'real' field.

Genetic algorithms (GA) became an efficient tool for search of optimal solutions in multiparametric spaces. The technique originated from the idea that mimicking evolution can be used to solve engineering problems (Mitchell, 1996). GA is a method for moving from one population of chromosomes (e.g. strings of ones and zeroes, or bits) to a new population by using a kind of 'natural selection' together with the genetics inspired operators of crossover and mutation (Mitchell, 1996). Each chromosome consists of 'genes' (e.g. bits), each gene being an instance of a particular allele (e.g. 0 or 1). Arrayed binary representations of model parameters are used as the chromosomes in GA when the parameters have to be optimized to maximize an optimization cri-

*Corresponding author. Fax: +1-301-504-5823; E-mail: ypachepsky@asrr.arsusda.gov.

terion. The optimization criteria calculated with parameters coded in a chromosome defines the ‘fitness’ of this chromosome. The selection operator chooses those chromosomes in the population that will be allowed to reproduce and on average the ‘fitter’ chromosomes produce more offspring than the ‘less fit’ ones. Crossover exchanges parts of two chromosomes, roughly mimicking biological recombination between two single-chromosome organisms. Mutation randomly changes the allele values in some locations in the chromosome. Each iteration of selection, crossover, mutation and discarding of less-fit chromosomes is called a generation. Initial chromosomes are selected randomly. Usually several hundreds of generations produce a chromosome with good fitness.

A GA should be used in optimization problems when the parameter space is large, the response is not perfectly smooth or is not well understood and the task does not require a global optimum to be found (Mitchell, 1996). Genetic algorithms were successfully used in designing a multiobjective groundwater monitoring problems (Cieniawski and others, 1995), in earthquake source parameter estimation (Zhou, Tajima and Stoffa, 1995), in 2D migration velocity estimation in heterogeneous media (Jervis, Stoffa and Sen, 1993) and in many other instances. Genetic algorithms are under continuous development (Homaifar, Qi and Lai, 1994; Mitchell, 1996).

Let us consider parameters to be random field values at the nodes of a grid and define a fitness in terms of the closeness of a realization probability distribution and a semivariogram to the input ones. Then the problem of generating random fields with accurate reproduction of input statistics becomes suitable for the application of a GA.

In many studies, a model of the spatial variability of some parameters of the media is chosen and random spatial distributions of the parameters are generated at nodes of a grid. Sometimes it is desirable to minimize differences among the probability distribution functions and among the semivariograms of consecutive generated random fields. At the same time, values of the parameter at the same grid point need to be allowed to vary randomly among the realizations. In such cases, a control of the accuracy of statistics of the generated fields is needed, so that the small differences among statistics of the random fields can be achieved. Traditional geostatistical simulation techniques do not have a built-in mechanism to provide such accuracy control. The purpose of this note is to demonstrate that GA can be used to minimize differences in statistics among generated random fields.

*E-mail: carroll@uxh.cso.uiuc.edu

METHODS

Although the general idea of the survival of the fittest is common to all genetic algorithms, there are many versions of GA differing in the techniques used to encode parameters, to select parent chromosomes, to perform crossovers and mutations and to select the fittest (Davis, 1991).

We used a FORTRAN version of a genetic algorithm GAFORTRAN version 1.6 written by David L. Carroll from the University of Illinois*. Nodal random field values were encoded as N-bit binary strings, where 2^N is the number of intervals between minimum $y_{\min}$ and maximum $y_{\max}$ boundaries. Thus the precision was $(y_{\max}-y_{\min})/2^N$. The values $y_{\min} = -2.5$, $y_{\max} = 2.5$ and $N = 6$ were used.

The selection of a parent was done with a tournament technique. With this technique, all chromosomes have equal chances to compete for becoming a parent and the fitter of any two becomes a parent. Elitism was allowed, that is the best parent reproduced in each generation. One child per pair was allowed and 5 members were retained in the population after each generation. To preserve the diversity in the population, the individuals similar to many other individuals were punished before the tournament by decreasing their fitness value, and individuals different from other were rewarded by the fitness increase. The reward and punishment technique was the multidimensional phenotypic sharing scheme with a triangular sharing function (Goldberg, 1989). Number of generations was 10000.

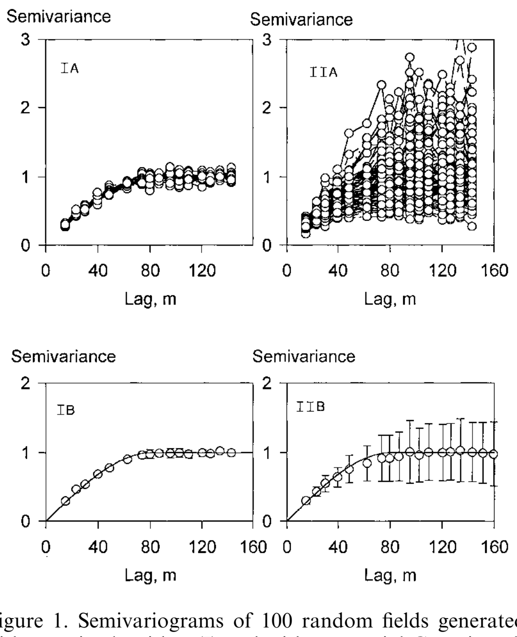

Figure 1. Semivariograms of 100 random fields generated with genetic algorithm (I) and with sequential Gaussian algorithm (II); (A) all semivariograms, (B) average semivariograms with standard deviations as error bars. Line shows expected semivariogram.

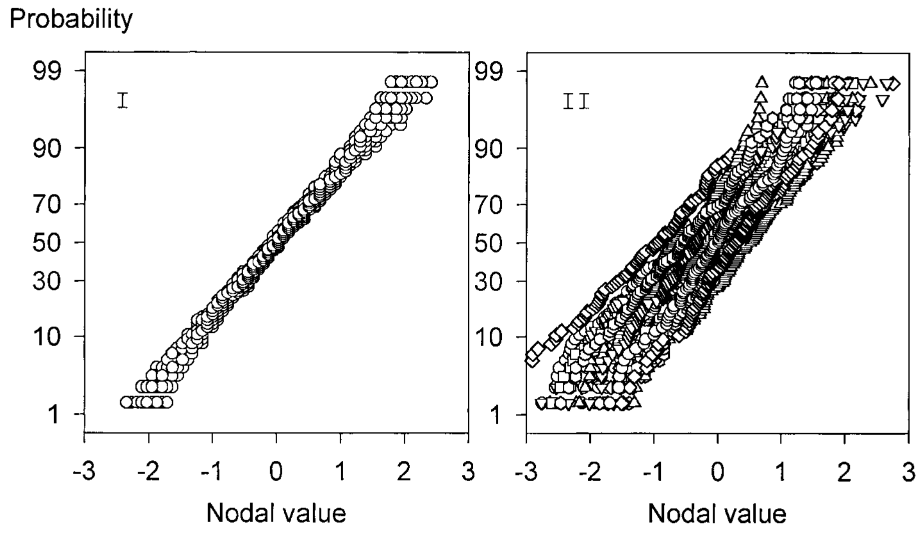

Figure 2. Probability distribution functions of nodal values in random fields simulated with genetic algorithm (A) and with sequential Gaussian algorithm (B).

The crossover was uniform and the allele exchange between parents happened in each bit position with the probability equal to 0.9. Mutations occurred with probability 0.04.

The fitness criterion was the inverse value of the sum of squared differences between calculated and expected variances at selected number of lags (20) and between calculated and expected probabilities for each value to appear. A normal probability distribution was assumed for the generated values. The probability function was calculated using the subroutine *erf* and related subroutines from (Press and others, 1994). The subroutine *gamv2* of Deutsch and Journel (1992) was used to calculate semivariograms

Initial seed numbers for the random number generator to generate initial chromosomes were chosen randomly from the interval between −32000 and −1000.

The subroutine *sgsim* of Deutsch and Journel (1992) was used to perform sequential Gaussian simulations for comparison purposes.

RESULTS AND DISCUSSION

We present a typical example that demonstrates the performance of the GA in comparison with the sequential Gaussian simulation technique. Random fields were simulated on a two-dimensional 10×10 grid with 10 and 15 m increments in x and y directions, respectively. The random values were expected to have normal distribution with zero mean and unit variance. The random fields were expected to have spherical semivariogram with zero nugget, unit sill and 80 m radius. Ordinary kriging was used and 200 m search radius was allowed in sequential Gaussian simulations.

Semivariograms obtained from one hundred unconditioned simulations are shown in Figure 1 and 2. The genetic algorithm provided reasonable correspondence between calculated and expected semivariograms (Fig. 1IA and IB) The initial seed number for the random number generator did not have much effect on the performance of the genetic algorithm. Distributions of the semivariance values at the same lag value obtained in the hundred runs were symmetrical and variation in generated values grew slightly as the lag value increased (not shown).

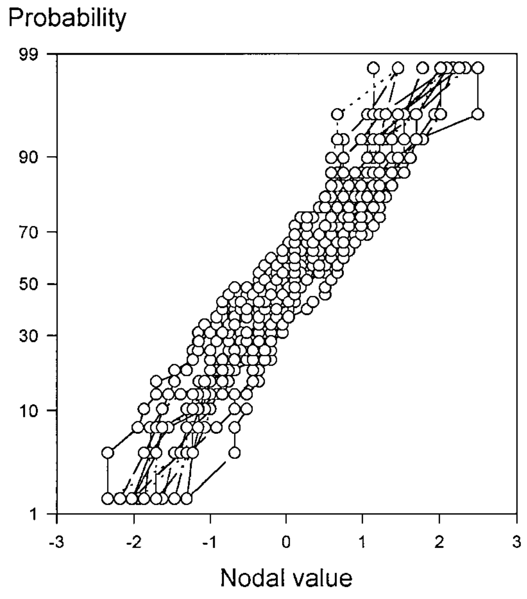

Figure 3. Probability distributions of nodal values in 100 random fields generated with genetic algorithm. Each curve shows distribution of values generated in particular node.

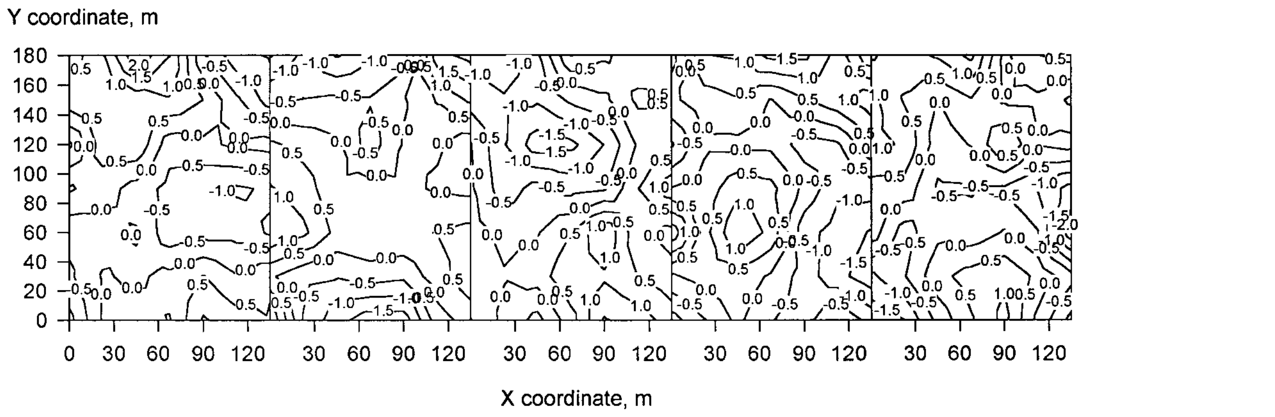

Figure 4. Contour graphs of five consecutive random fields generated with genetic algorithm.

The sequential Gaussian simulations followed the expected semivariogram (Fig. 1IIA and IIB), but the scatter was significantly larger than that for the genetic algorithm. For the sequential Gaussian algorithm, deviations from the expected semivariogram grew as the lag increased and distributions of the variances at the same lag were skewed with large values appearing more seldom than small ones. The skewness increased as the lag increased (not shown). Average semivariograms were close to the expected one for both algorithms (Fig. 1IA and IIB).

Probability distribution functions calculated from 100 nodal values are shown in Figure 2. The genetic algorithm (Fig. 2A) provided the distributions close to the expected normal $N(0, 1)$. Some discrepancy is observed at the tails of the distributions. This is related to the assumed range {−2.5, 2.5} of the generated values. Sequential Gaussian simulations (Fig. 2B) yielded a wide range of distribution functions that were scattered around the expected $N(0, 1)$ and the range of deviations was larger than in the random fields simulated with the genetic algorithm.

The important requirement for the genetic algorithm was the ability to produce spatial patterns of generated values that would differ from one realization to another. We had randomly chosen fifteen nodes of the grid and plotted distribution functions of random values that were obtained from the hundred runs in each of the nodes. The results are shown in Figure 3. They demonstrate that nodal values randomly change from one realization to another with the general pattern of distributions close to $N(0, 1)$. Contour plots of five consecutive fields generated with the genetic algorithm are shown in Figure 4. The plots show how different the generated fields can be as the initial random seed changes.

The genetic algorithm required a large number of iterations to produce a random field with statistical properties close to the expected ones. The rate of improvement in the performance is illustrated in Figure 5. Improvement slows down as the number of iterations grows and in some cases no significant improvements were achieved during last 7,000 iterations. A significant computer time was required to

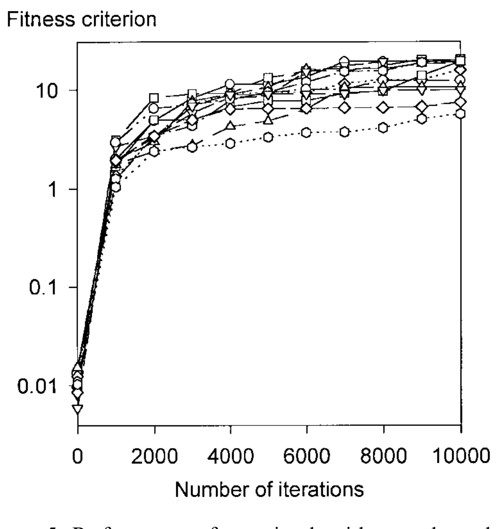

Figure 5. Performance of genetic algorithm as dependent on number of generations.

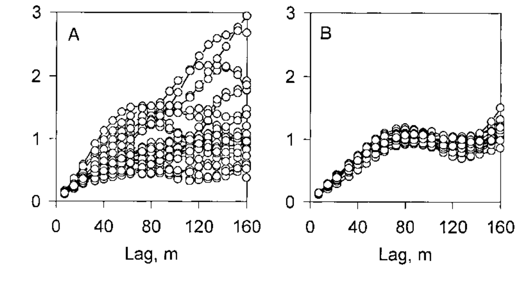

Figure 6. Semivariograms of random fields obtained from sequential Gaussian simulations on 37×37 grid; (A) unconditioned simulations, (B) simulations conditioned with data from simulations generated by genetic algorithm on 10×10 grid embedded in 37×37 grid.

run the GA program: ten thousand generations took almost 2 h in a 100 MHz PC. Since it is frequently necessary generate random fields on much more dense grids, the computer resources may present an impediment in the use of the genetic algorithm to generate random fields because there is no a priori knowledge about the convergence rate.

It is known that the performance of the sequential Gaussian algorithm can be improved with the use of data conditioning (Deutsch and Journel, 1992). We experimented with combining the two techniques. The genetic algorithm was used to generate a random field on a 10×10 grid mentioned previously and a sequential Gaussian algorithm was used to generate the field in the grid 37×37 where the distances between grid points were 4 times less than in the old grid and the new grid was embedded in the old grid. Results are shown in Figure 6. The conditioning improved the performance of the sequential simulator as was expected. The semivariograms of the generated fields were much closer to the expected ones than without the conditioning.

Both the described numerical experiments and other experiments (not shown here) led us to the conclusion that genetic algorithms can be useful tools in generating random fields with preselected spatial dependencies. When computer time becomes a limitation, a hybrid field generator can be used that will generate the field on a coarse grid with the genetic algorithm and then generate the field on a refined grid using conditioned simulations with sequential Gaussian algorithm.

Acknowledgments—We are grateful to Dr. C.G. Crawford for a stimulating discussion and for suggestions for improvements.

REFERENCES

Cieniawski, S. E., Eheart, J. W. and Ranjithan, S. (1995) Using genetic algorithms to solve a multiobjective groundwater monitoring problem. *Water Resources Res.* **31**(2), 399–409.

Davis, L. (Ed.) (1991) *Handbook on Genetic Algorithms.* Van Nostrand Reinhold, New York, 385 pp.

Deutsch, C.V. and Journel, A.G. (1992) *GSLIB: Geostatistical Software Library and User's Guide.* Oxford University Press, Oxford, New York, 340 pp.

Goldberg, D.E. (1989) *Genetic Algorithms in Search, Optimization and Machine Learning.* Addison-Wesley, Reading, Massachusetts, 412 pp.

Gotway, C.A. and Rutherford, B.M. (1994) Stochastic simulation for imaging spatial uncertainty: comparison and evaluation of available algorithms. In *Geostatistical Simulations*, eds. M. Armstrong, P.A. Dowd. Kluwer Academic Publishers, Dordrecht, pp. 1–22.

Grindrod, P. and Impey, M. D. (1993) Channeling and Fickian dispersion in fractal simulated porous media. *Water Resources Res.* **29**(12), 4077–4089.

Homaifar, A., Qi, C. X. and Lai, S. H. (1994) Constrained optimization via genetic algorithms. *Simulation* **62**(4), 242–254.

Jervis, M., Stoffa, P. L. and Sen, M. K. (1993) 2D Migration velocity estimation using a genetic algorithm. *Geophys. Res. Lett.* **20**(14), 1495–1498.

Mackay, R., Cooper, T. A., Metcalfe, A. V. and O'Connell, P. E. (1996) Contaminant transport in heterogeneous porous media: a case study. 2. Stochastic modeling. *J. Hydrol.* **175**(1–4), 429–452.

Mitchell, M. (1996) *An Introduction to Genetic Algorithms.* MIT Press, Cambridge, Massachusetts, 205 pp.

Press, W.H., Teukolsky, S.A., Vetterling, W.T. and Flannery, B.P. (1994) *Numerical Recipes in FORTRAN. The Art of Scientific Computing*, 2nd ed. Cambridge University Press, Cambridge, 963 pp.

Tsang, Y. W., Tsang, C. F., Hale, F. V. and Dverstorp, B. (1996) Tracer transport in a stochastic continuum model of fractured media. *Water Resources Res.* **32**(10), 3077–3098.

Zhou, R., Tajima, F. and Stoffa, P. L. (1995) Earthquake source parameter determination using genetic algorithms. *Geophys. Res. Lett.* **22**(4), 517–520.