INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

ProQuest Information and Learning
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA
800-521-0600

University of Alberta

Calculation of Time Variations in Mineral Grades from High Resolution Geostatistical Models

by

Xuesi Wang 

A thesis submitted to the Faculty of Graduate Studies and Research in partial fulfillment of the requirements for the degree of Master of Science

In

Mining Engineering

Department of Civil & Environmental Engineering

Edmonton, Alberta

Fall 2000

Abstract

Geostatistics provides us an effective and powerful tool to estimate ore grades. High resolution geostatistical models characterize the spatial variability of the orebody. It is often useful to predict the time variation of mineral grades prior to mining since the processing plant efficiency depends on the statistical behavior of the time variation of mining production before decision-making. A methodology is presented for this purpose.

Time variation depends on the mining sequence and rate. A Spatial-Temporal Modeling(STMOD) methodology, is developed to calculate the grade variation in a series of time intervals. It transforms variability from a spatial distribution to a time domain. Several examples are presented to demonstrate the algorithm and the associated program.

STMOD program has been developed to predict the time variation of studied mining process. The results for intermingled waste and ore must be evaluated on the basis of total mining rate. Furthermore, advanced applications, such as multiple mining faces have been explored with the proposed methodology.

A detailed case study using oil sands mining has been presented as an application of the STMOD methodology and program.

Table of Contents

Acknowledgements

I would like to thank Clayton V. Deutsch, my supervisor, for his many suggestions and constant support during this research. I am also thankful to Dr. Yulong Xie for his guidance through the early years of chaos and confusion.

Edmonton, Alberta
July 1, 2000

Xuesi Wang

Chapter 1

Introduction

Quality control in mine extraction is a very important problem for the geologist and mining engineer [10]. They must consider the quality of the ore in terms of ore grade and processing characteristics. They must also consider different time frames. The problem to be addressed here is the characterization of the temporal variability of the raw materials used by the processing plants, i.e., time variation of ore grade. This is not a new problem, however, it is becoming more serious.

The practice of ore reserves is difficult given the geological variations, mining methods, and information effect. As a result, mine development often carries uncertainties and risks related to reserve quality and quantity, capacity to develop the mine within budget and schedule, ability to meet the mine production targets, startup date, length of break-in period, costs, quantity and quality, and projected cash-flow [12].

Time variations of ore grades are based on space-time relations. Time variation will be generated using an orebody model. High resolution geostatistical simulations can be designed to answer a variety of practical questions about the interaction between the mining procedure and the spatial distribution of ore grades within a mineral deposit. A common problem is how to select a mining procedure that will

somehow minimize the variability of ore grades delivered to a processing plant, while satisfying constraints on product quality [13]. If the variability of the delivered ore can be cheaply reduced by a change in mining scheme, it may be possible to simplify or reduce the capacity of the plant's homogenization systems. The mining methods that affect variability include: the dimensions and orientations of mining blocks; the number of blocks being mined and blended together; the geologic information used to select the blocks to be blended; and the possibility of splitting some heterogeneous blocks into ore and waste sub-blocks.

Extraction plants commonly rely on homogenization and proportioning facilities to reduce the natural variability of incoming run-of-mine materials and to blend different materials to satisfy chemical specifications. To determine appropriate design and capacity for such facilities, engineers must know the time variation of the run of mine materials as they arrive at the plant.

As a common means for ore reserve estimation, geostatistical simulation [1, 2, 3, 4, 6] provides numerical tools to quantify spatial variability, make optimal predictions, and assess spatial and temporal uncertainty. These simulated data should reproduce all statistical characteristics of the deposit, including the spatial variability observed in the sample data from the real deposit. The result of a mining simulation applied to a numerical orebody is a time-series simulation of mining products that mimics the statistical behavior of real materials to be delivered to the plant in the future. It is especially useful for processing plants to predict variations of ore grade in successive time periods of mining and processing.

Application of geostatistical modelling generates conditional realizations, which consists of a large set of regionalized variables distributed among nodes within a two

or three dimensional regular grid covering the region of the deposit to be mined. The predicted mining production is determined directly from the specified mining scheme and mining rate within this modelled space.

For the mining industry there is little or no resources available concerning the characterization of time variation of mine production based on geostatistical simulation. Much research work is focused on the optimal ore reserve estimates and predictions. Examples of this work include, "Conditional Simulation and Kriging as an Aid to Oil Sands Development", which was presented in 1985 by Dimiitrakopoulos et. al. Also, a recent CIM paper "Optimal Drill-hole Spacing and Uncertainty in Prediction of Oil sands Bitumen and Fines Content" by C. V. Deutsch and A. Beardow dealt with this subject. The latter paper also demonstrates the application of geostatistical tools to quantify the "Goodness" of bitumen/fines predictions for different drill-hole spacing. These reports provide the practical experience necessary for the application of geostatistical tools to the time variation problem.

The STMOD algorithm, which is presented in this thesis, allows for flexibility in the applied mining scheme, variable mining rate and spatial models. Simulations of alternative mining and homogenization procedures applied to an orebody illustrate the method of using conditional simulation to select the optimized mining and homogenization procedures before mining of deposit. By directly applying this algorithm, a time-series simulation of material compositions that mimics real materials to be delivered to the processing plant is produced. Furthermore, multiple mining faces and different mining schemes can be used simultaneously in this algorithm.

Chapter 2

Geostatistical Modeling for Ore Reserve Estimation

Before a mine is planned in detail, a preliminary analysis should be conducted first. This preplanning permits the mining engineer to make a rapid assessment as to whether or not a particular mining property warrants further consideration [12]. The estimation of the various characteristics of a reserve, such as quantity, grade, and thickness, is an ongoing process throughout the life of a mining venture. Because mining investments are generally large, the economic consequences of making investment decisions are very important. Therefore, it is crucial that we evaluate the mineralization and its potential very carefully. In particular, it is critical that we consider all the existing information at each decision step.

Resource evaluation is therefore a process, not an event. Resources will always be estimated sequentially and with more information as the project progresses.

Geostatistics provides a consistent conceptual framework for resource evaluation that makes maximal use of available information.

In this Chapter we give a summary of geostatistical application for ore reserve estimate. Geostatitics is a powerful tool for mining engineering. A simple description

of this concept follows.

2.1 Introduction

The term "geostatistics" naturally should mean "statistics applied to earth sciences" [11], and originally that is exactly what it did mean. However, after the development in the past several years, the definition of geostatistics has gradually been changed into "application of the theory of regionalized variables" [1], regardless of whether those applications happen to the applied earth sciences. Geostatistics (as the term is now applied) is used to infer and interpret phenomena that appear to vary continuously in space and time according to a spatial probability law, such as ore grades, porosities, pollutant concentrations, chemical analysis of rocks, and so forth. Such values are implicitly assumed to be correlated with each other, and the study of such a correlation is usually called a "structural analysis" [1] or "variogram modeling". After structural analysis, estimations at unsampled locations are made using "kriging" or they can be simulated using "conditional simulations", e.g., Gaussian Sequential Simulation.

In mining applications, useful results that can be obtained by geostatistical methods include optimal estimates of total reserves, local ore grades, and recoverable reserves in mineral deposits, estimates of variability of different-sized blocks of ore, selection of best drilling locations for the improvement of quality of ore estimation, and simulations of the spatial distribution of ore grades for use in mine and process planning.

Briefly, the steps in a geostatistical study [4] (lec 02-17) include:

(a) Define a population / area (A) for averaging;

(b) Collect all information relevant to special variability within A;

(c) Synthesize all the information into a stationary RF model representative of A, e.g., variogram $\gamma_A(h)$ (calculation and modeling of variograms).

(d) Use this model, e.g, $\gamma_A(h)$, for further calculations.

A clear and concise introduction to basic geostatistical concepts can be found in Chapter I of Journel and Huijbregts(1978) [1]. The terminology and most of the notation used in this thesis conform to their usage.

2.2 Random Functions and Regionalized Variables

Random Function theory (RF) is a common mathematical terminology. Now it is applied to many disciplines, such as geostatistics. That is, the samples v_j are viewed as outcomes of a random variable that is a function of the spatial coordinates $V(x_j)$. A regionalized variable is any variable that is distributed in space (or time). The theory says any measurement can be viewed as a realization of a random function (or random process, or random field, or stochastic process). This theory forms the underpinnings of geostatistics. Though the true values of the variable are strictly deterministic and uniquely exist, the opportunity for measurement and other errors allows one to take this probabilistic approach. The linear estimator $v(0)$ is then viewed as an outcome of the RF $V(x_0)$ which approximately represent the true value at spatial location x_0. The actual error due to measurement and other factors can then be viewed as an outcome of the RF $R(x0) = V(x0) - V0$ which is called the estimation error in statistics.

To facilitate the understanding and explanation, let's consider the following scenario. You collect some samples at a number of locations. Applying the linear

estimator with some fixed choice of weights, you can produce and estimate by linear estimator at some unsampled location. Suppose now that you return to the same sample locations and resample. Due to measurement error, the sample values may be slightly different or largely distinct. Applying the previous linear estimator you produce a second estimate for the same unsampled location. And so on. Each time, you are in fact producing a realization of the estimation error. The estimation error is a kind of RF process.

What would be the desirable properties of this estimation error RF? In fact we can not exactly determine true value of estimation error. However we can describe it by statistical analysis. Ideally, it should be identically zero. Failing this one should focus on two things. Firstly, the estimation error should be unbiased. That is, its expected value is zero: $E[R(x0)] = 0$. This loosely means that on average the error incurred is zero. The second property is that the estimation error should have minimum spread. That is, its variance $Var[R(x0)]$ is minimal. Linear estimators which produce estimation error RF's having the properties of unbiasedness and minimum variance are called BLUE (Best Linear Unbiased Estimators) [5].

2.3 Estimation

First let us begin with a brief look at the more common estimation techniques. A group of techniques known as Inverse Distance propose the following formula for the weights: $wi = D(x0, xi)^{-p}/W$. The non-negative parameter p is chosen to reflect the assumed measure of spatial continuity of the variable. For example, with $p = 5$ much more weight is given to the nearest sample, than if using $p = 1$. The factor W is a normalization factor chosen so that the weights sum to unity. This gives the estimate

the desirable property of unbiasedness. Four variants of Inverse Distance estimation are common:

Local Average: p=0

Inverse Distance: p=1

Inverse Distance Squared: p=2

Polygonal: p $\rightarrow$: $infinity$

The Local Average (LA) technique assumes no spatial continuity of the variable, and simply uses the arithmetic mean of the sample values. The Inverse Distance (ID) and Inverse Distance Squared (IDS) techniques assume a loss in spatial continuity over greater distances, with the loss assumed greater for the latter. Finally, the Polygonal (P) technique is perhaps the simplest and uses the value of only the nearest sample as the estimate.

Which of these techniques is best? We can not make any conclusion without an exhaustive data set. Only examples can be found, which rank each of the techniques as superior. Here what we also need to mention is that samples should be representative for all studied areas. If the samples collected do not have representativity we need to perform declustering [4, 5]. Even more we need to use the secondary information, e.g., calibration approach with conditional distributions [4, 5] due to data scarcity. Therefore, in practice, the criterion used to rank the techniques becomes vague and less subjective and tend to depend heavily on the practitioner's experience and judgement.

From last section we know there exists certain estimation error for linear estimator. The geostatistical estimation problem is to obtain a BLUE estimator by choosing the weights wi, i=1, ... , n . Kriging is a typical BLUE. The unbiasedness property is easily

achieved if one is willing to assume that the expected values of the sample values are equal, and identical to the expected value of the true value at the unsampled location: $E[V(xj)] = v,\ j = 0, 1, ..., n$. This is the assumption of stationarity and is probably realistic, if the sample locations and unsampled location are not too distant from one another. The constraint on the weights can easily be deduced as: $w1 + ... + wn = 1$. Note that this constraint is explicit in the formula for the Inverse Distance estimators, i.e., they are unbiased linear estimators.

Geostatistics was developed to address this problem of subjectivity and to give a firmer mathematical foundation for the estimation of spatially varying quantities. No attempt will be made here to give a complete or necessarily accurate description of the theory. Sufficient references exist for this purpose [1, 4, 5, 6].

2.4 Variogram

Conceptually, variogram is defined to describe the spatial variability of variable in study area. The variogram (or its counterpart, the covariance) underpins all of geostatistics. You can use the variogram to model the conway two values which in space or time are correlated. Most people intuitively know that two values in space that are close together tend to be more similar than two values farther apart. Univariate statistics cannot take this into account. Two distributions might have the same mean and variance, but differ in the way they are correlated with each other. Geostatistics allows one to quantify the correlation between any two values separated by a distance h (usually called the lag distance) and uses this information to make predictions at unsampled locations.

Consider two numerical values z(x) and z(x+h) separated by the vector h. The variability between these two quantities is characterized by the variogram function $2\gamma(x,h)$, which is defined as the expectation of the random variables $[z(x)-z(x+h)]^2$:

$$2\gamma(x,h) = E[z(x)-z(x+h)]^2 \tag{2.4.1}$$

In all generality, this variogram $2\gamma(x,h)$ is a function of both the point x and vector h. In practice, at least in mining applications, there is an intrinsic hypothesis—stationary. The hypothesis is that the variogram function $2\gamma(x,h)$ depends on the separation vector h and not on the specific location x. The physical meaning is there is strong spatial continuity in certain direction.

It is then possible to calculate the variogram $2\gamma(x,h)$ from the available data. An estimator $2\gamma(x,h)$ is the arithmetic mean of the squared differences between two experimental measures $[z(x_i), z(x_i+h)]$ at any two pair points separated by the vector h:

$$2\gamma(x,h) = \frac{1}{N(h)}\sum_{i=1}^{N(h)}[z(x_i)-z(x_i+h)]^2 \tag{2.4.2}$$

Where N(h) is the number of the experimental pairs $[z(x_i), z(x_i+h)]$ of data separated by the vector h. This is also called the experimental variogram calculation. In the definition of the variogram $2\gamma(x,h)$, h represents a vector of modulus $\|h\|$ and direction α. Consider a particular direction α, beginning at the origin, $\gamma(0)=0$, the variogram increases in general with the modulus $\|h\|$. This is simply an expression of the fact, on average, the difference between two grades taken at two different points increases as the distance $\|h\|$ between them increases. The manner in which this variogram increases for small values of $\|h\|$ characterizes the degree of spatial continuity of the variable studied.

Variogram models are usually described mainly by six parameters. We here concentrate on three main parameters. The range is the lag distance at which all successive values are independent of each other. The sill is the variogram value corresponding to the range. When we want to model the variogram, we have to specify a type of model as well. Nugget effect is the value of variogram for h=0. Though nugget effect should be strictly equal to zero, several factors, such as sampling error and short scale variability, may cause the sample values to be quite different. This causes a discontinuity of variogram at origin. Variogram modeling is a prerequisite for kriging and Gaussian sequential simulation. Variogram modeling is a modeling of such a spatial correlation structure.

2.5 Kriging

In practice sample data for many physical parameters often and invariably occur on irregular grids. Frequently, the values of the variable are desired on a regular and significantly finer grid than that is practical to sample. This presents the problem of estimation, or of using sample data to predict values in areas which are not sampled. The simplest estimation techniques use weighted linear combinations of nearby sample values to estimate the value at a point: $V(0) = w1 * v1 + ... + wn * vn$. Here, $V(0)$ is an estimate of the value of the variable at x0, the unsampled location. The estimation problem is how to choose the weights $wi, i = 1, ..., n$ to minimize the error in estimation. However, this goal is impossible to achieve in practice, since it would require prior knowledge of the true values at the unsampled locations. As a result, most estimation techniques make substantial simplifying assumptions regarding the behaviour of the variable in order to predict unsampled values.

Most methods give greater weight to samples which are closer to the point being estimated. That is, wi is inversely proportional to D(x0, xi) where D(A, B) is the distance between samples A and B. This intuitive notion is an implicit statement of continuity of the variable, in that values nearby are more likely to be similar than values far apart. This is indeed true for most variables occurring in physical and earth sciences. However, these methods do not take into account the observable spatial correlation between sample points. That is, the continuity between vi and vj, or called redundancy between the data [4, 5]. This can be used to improve the weighting given to vi based initially on $D(x0, xi)$ by the information contained in the correlation between vi and the other nearby samples based on $D(xi, xj)$. Kriging, which is the core of Geostatistics, is the technique which utilizes this additional information to produce more reliable estimates. Perhaps, this technique also produces an estimate of the error incurred.

However, only Inverse Distance estimators are not BLUE. Kriging estimators are. Kriging is a procedure for constructing a minimum error variance linear estimate at a location where the true value is unknown [4, 5]. Mathematically, we have a constrained minimization problem so that the method of Lagrange Multipliers is used. The mathematics is somewhat messy but not too difficult. The end result is a set of linear equations, the ordinary kriging system [6], that the weights must satisfy: $C_{ij} * w + mu = di$ and the unbiasedness equation, where the elements C_{ij} are the covariances between samples vi and vj, and the elements di are the covariances between the sample vi and the unsampled value $v0$. The parameter mu is the Lagrange multiplier, and is useful for calculating the resulting minimized error variance.

All of the covariances in this linear system are actually unknown. Instead, we

approximate these by the sample covariance of samples separated by the same distance and direction. Due to data scarcity, we often ignore the direction and incorporate adjustable tolerances to produce sufficient data pairs to give more reliable estimates for the covariances. In practice, to facilitate the calculation and interpretation we often calculate the sample variogram or semi-variogram $\gamma(h)$, which is defined as "one half the average squared difference of data values separated by h." The variogram plus the covariance must equal the sample variance [4, 5]:

$$\gamma(h) = c(h) + c(0)$$

.

To guarantee a unique solution to the kriging equations, the variogram model must be positive definite (a legitimate measure of distance) [4]. We usually model the sample variogram with a smooth curve having the following properties:

$$\gamma(0) = c0 \geq 0$$

$$\gamma(a) = c \geq c0, \quad a > 0$$

$$\gamma(h) = c, \quad h \geq a.$$

The constant $c0$ is called the nugget and represents any small-scale data variability or possible sampling errors. The constant c is called the sill and should roughly equal the sample variance. The smallest distance at which the variogram reaches the sill, h=a, is called the range and represents the largest separation the correlated samples could have. The increase of the variogram from the nugget to the sill can be modelled in several ways. There are a number of known legitimate models that are commonly used, such as spherical, exponential, Gaussian, linear and so on. The most common

is the spherical model:

$$\gamma(h) = c0 + (c - c0) * [1.5 * h/a - 0.5 * (h/a)^3], \quad h \leq a$$

Returning to the kriging equations, one can now see how the spatial correlation between samples can influence the weight given to a single sample. If the samples were uncorrelated, then the off-diagonal entries of C would be zero, and the weights would be $wi = di/Cii$. Since the covariance decreases with distance from the unsampled location, this formula for the weights is akin to the Inverse Distance estimation weights. We know that covariance or variogram is a measure of geostatistical distance, not physically geometric distance. However, if the samples are spatially correlated, then the off-diagonal entries are non-zero, and the weights are given by $w = Cinv * d$. This premultiplication by $Cinv$ adjusts the raw statistical distance weights in d by accounting for possible redundancies due to sample clustering.

For example, two samples close together will generally contribute less information to the estimation than samples farther apart and will be reflected by larger values in the C matrix off-diagonal entries. The raw weight assigned to these samples will generally be redistributed to other samples that are farther away but less redundant. This possible redundancy between samples does not just depend on the geometric distance between them, but also depends on the spatial continuity.

Thus the kriging system accounts for two important aspects of estimation at unsampled locations. As with the Inverse Distance estimation techniques, the distance to the samples that reflects the general weight distribution, or called data closeness. Secondly, the clustering between samples, or called data redundancy, is recorded by a statistical distance measure, which can be used to redistribute the weights according to any redundancies present.

Kriging is a geostatistical estimation technique. It uses a linear combination of surrounding sampled values to make such predictions. To make such predictions, we need to know the weights applied to each surrounding sampled data. Kriging allows you to derive weights that result in optimal and unbiased estimates. Within a probabilistic framework, kriging attempts to: (a) minimize the error variance; and (b) systematically set the mean of the prediction errors to zero, so that there are no over- or under-estimates.

There are some cases when other, usually more abundantly sampled data, can be used to help in the predictions. Such data are called secondary data (as opposed to primary data) and are assumed to be correlated with the primary data. For example, we can predict porosities based not only on the well measured porosities but also on seismically derived porosities. In this situation, we can try a cokriging. To perform cokriging, we need to model not only the variograms of the secondary and primary data, but also the cross- variograms between the primary and secondary data.

Kriging is expressed in a so-called "kriging system", which relates the covariance between the samples, the covariance between each sample to the location to be estimated, and the unknown weights. The covariance matrix is inverted to solve for the weights. However, Kriging is a smooth estimator. The shorter scale variability there exists in data, the smoother for kriging estimation; the wider, the data spacing, the smoother [4, 5]. But the smoothing effect can be quantified, namely we can add the missing kriging variance, this will lead us to develop the simulation method which we will discuss in the next section.

2.6 Simulation

All kriging estimates and other estimates provide a smooth image of underlying reality: the variogram of these estimates would not reproduce the data variogram. The concept of conditional simulation allows generating alternative, equiprobable, images which honor data values at their locations and reflect a series of spatial continuity functions [8]. Unlike kriging or other estimators, simulation provides alternative global representations of the variables studied, not just the local and unique estimation. On the other hand, simulation provides complete local accuracy, namely, both local and several locations.

Kriged estimates are too smooth and therefore inappropriate for most engineering applications. Simulation corrects for this smoothness and ensures that the variogram / covariance is honored [4]. Although the covariance between the kriged estimat and data is correct, the variance is too small:

$$Var\{Y^*(u)\} = c(0) - \sigma^2_{SK}(u)$$

the missing variance is the kriging variance $\sigma^2_{SK}(u)$. We need to add back in the missing variance without changing the covariance reproduction.

There are many different simulation algorithms, sequential Gaussian is simple and most widely used. Gaussian distribution has a distinct feature that the global N(0,1) distribution will be preserved if we always use Gaussian distributions. Therefore, if we transform the samples to normal score, next we perform a serials calculations, then we transform the results back to original distribution domain. A key idea of sequential simulation is to use previously kriged/simulated values as data so that we reproduced the covariance between all of the simulated values. To perform this we

need to select the random path during the sequential simulation. In brief, the steps in Sequential Gaussian Simulation [4] are as follows:

1. transform data to "normal space"

2. establish grid network and coordinate system

3. assign data to the nearest grid node (take the closest of multiple data assigned to the same node)

4. determine a random path through all of the grid nodes (a)find nearby data and previously grid nodes (b)construct the conditional distribution by kriging (c)draw simulated value from conditional distribution

5. check results (a) honor data (b) honor histogram: N(0,1) - standard normal with a mean of zero and a variance of one? (c) honor variogram? (d) honor concept of geology?

6. back transform to original data space

Chapter 3

Time Variations in Mineral Grades from High Resolution Geostatistical Models

In this chapter, an algorithm for transferring spatial variation of ore grades to time domain is presented. Geostatistics provides an effective and powerful tool to build an orebody model. High resolution geostatistical models characterize the spatial variability in an ore body. This spatial variability must be transferred to time variation of mining production to help with decision-making. Examples for sensitivity analysis are presented as well.

3.1 Introduction

Extraction plants commonly rely on homogenization and proportioning facilities to reduce the natural variability of incoming raw materials and to blend different materials to satisfy chemical specifications. To determine appropriate design and capacity for such facilities, engineers must know the statistical behavior of each raw materials arriving at the plant. Conventional ore estimations reproduce the spatial variability

of the grades. They do not represent the variation of ore grades in successive time periods of mining and processing.

High resolution geostatistical simulations could be designed to answer a variety of practical questions about the interaction between mining procedures and the spatial distribution of ore grades within a mineral deposit. A common problem is the selection of a mining procedure that will somehow minimize the variability of ore grades delivered to a processing plant, while satisfying constraints on product quality. If the variability of the delivered ore can be cheaply reduced by a change in mining scheme, it may be possible to simplify or reduce the capacity of the plant's homogenization systems. The methods that affect the variability include: the dimensions and orientations of mining blocks; the number of blocks being mined and blended together; geologic information to select the blocks to be blended; and the possibility of splitting heterogeneous blocks into ore and waste sub-blocks [10].

Application of geostatistical modeling generates conditional simulations, which consist of a large set of simulated raw material analysis distributed among the nodes of a fine two- or three-dimensional regular grid covering the region of the deposit to be mined. Mining production results directly from the specified mining scheme and mining rate.

An algorithm (STMOD), which allow an flexible mining scheme, variable mining rate and spatial models, is presented. By applying this algorithm, a time-series simulation of material compositions that mimics real materials to be delivered to the processing plant, is reproduced. Furthermore, two mining work faces and different mining schemes can be used by using this algorithm.

3.2 Algorithm for Time-variation calculation

To obtain time variations, the mining scheme must be specified. Next, by incorporating the mining scheme and mining rate, time variation can be generated based on the relationship between the spatial distribution of the grades and the mining scheme. Figure 3.1 shows a classical 3-D block model. Here we assume that the size of blocks in the 3D model is dx by dy by dz. Orebody of simulated site consists of nx by ny by nz blocks, the number of blocks in x, y, z directions, respectively. Then the total volume (m^3) can be calculated as follows:

$$Volume = nx \cdot dx \cdot ny \cdot dy \cdot nz \cdot dz, \qquad [m^3] \tag{3.2.1}$$

The specific gravity, SG, could be locally variable. The tonnage of ore on the bench is given by:

$$Tonnage = Volume \cdot \overline{SG}\ , \qquad [t] \tag{3.2.2}$$

Where $\overline{SG}$ is the average specific gravity (tonnage per cubic meters, t/m^3) of the ore. The total time (in days, d) needed to mine the ore can also be obtained as:

$$T = \frac{Tonnage\ , \quad [t]}{rate\ , \quad [t/d]}\ , \qquad [d] \tag{3.2.3}$$

Where $rate$ is the mine production rate (Tonnage per day, t/d) in the mining. Note that here we assume that the entire bench is ore. A mixture of ore and waste could be considered; however, this assumption simplifies presentation of the time variation.

The mining operation follows the sequence illustrated by numbers in Figure 3.3, 3.4. Here fa, fw and bh define mining scheme and the means to transfer spatial variability to temporal variability. fa is the length of face advance in blocks(x direction), fw is the width of face advance in blocks(y direction), and bh is the height of

"benches"(vertical direction). In general, all benches have same height, but could be different one from another if different mining process is considered by using high resolution geostatistical simulation. The mining volume in direction of one face advance consists of fw by fa by bh blocks, which are dx by dy by dz. Therefore, time, t_f, to mining one face advance is as follows:

$$t_f = \frac{fa \cdot fw \cdot bh \cdot dx \cdot dy \cdot dz \cdot SG}{rate}, \quad [d] \tag{3.2.4}$$

Accordingly, the average grade in the time t_f can be as follows:

$$\overline{G_{t_f}} = \frac{1}{fa \cdot fw \cdot bh} \cdot \sum_{i=1}^{fa} \sum_{j=1}^{fw} \sum_{k=1}^{bh} g_{(i,j,k)} \tag{3.2.5}$$

where $g_{(i,j,k)}$ is the ore grade in the block (i, j, k) which is dx by dy by dz. In the same way, the time variations in the next time interval t_f can be calculated for next successive fa by fw by bh blocks and so on.

To facilitate calculation, we transform 3-D cell grades to one dimension cell grades of time series based on mining scheme, namely from the $g_{(i,j,k)}, (i = 1, 2, 3, ..., nx; j = 1, 2, 3, ..., ny; k = 1, 2, 3, ..., nz)$ to $g_{(l)}, (l = 1, 2, 3, ..., L; L = nx \cdot ny \cdot nz)$. The process of analysis can be explained as Figure 3.5 in which we assume that $fa = 3dx$, $fw = 3dy$ and $bh = 3dz$. Then we can get the time variation of grade in time interval t_f by averaging grades in the successive time series. Here the time interval directly depends on mining rate. The final time variation can be expressed as following form:

$$\overline{G_{t_f}(l)} = \frac{1}{t_f} \cdot \sum_{i=1+(l-1)\cdot t_f}^{t_f \cdot l} g_{(i)} \tag{3.2.6}$$

l =1, 2, 3, ..., T/t_f; $t_f = fa \cdot fw \cdot bh$; T is total mining time.

The mining scenario illustrated in Figure 3.2 translates spatial variability to a particular pattern of time variation. Once we obtain average grade in the time interval, by the same way time variation of grades can be calculated in subsequent time

intervals. The dash lines in Figure 3.6 show grade variation in the t_f time intervals and solid line $6 \cdot t_f$ time intervals.

Further, we can consider the utility of multiple mining work faces. By choosing multiple mining work faces, the high and low grade ore averages out and reduces the time fluctuation of ore grade. Figure 3.3 is the mining scheme of two working faces.

However, the orebody is a mixture of ore and waste. This requires us to separate ore and waste in the mining process since the processing plant only receives ore. To realize this objective, firstly we need to select the proper block size to effectively separate waste and ore. Then, we can calculate the responding time variation of ore and waste. We will leave this issue to next chapter.

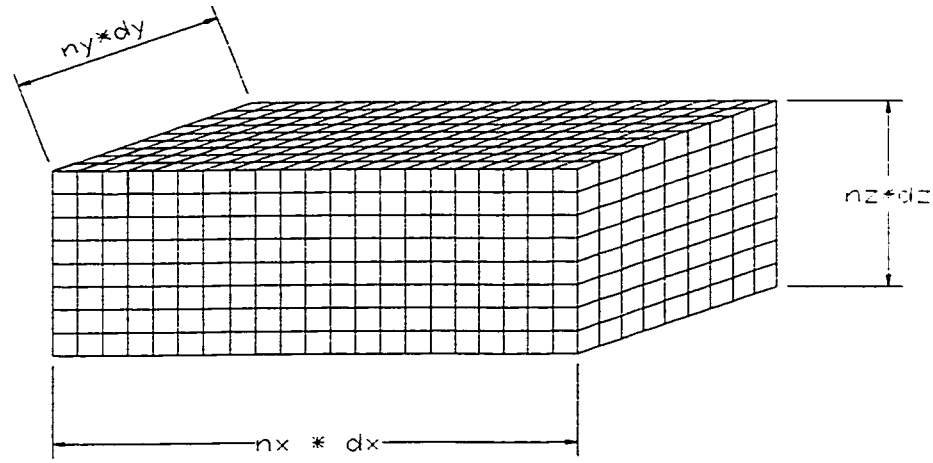


Figure 3.1: A 3D block model, which could come from geostatistical simulation.

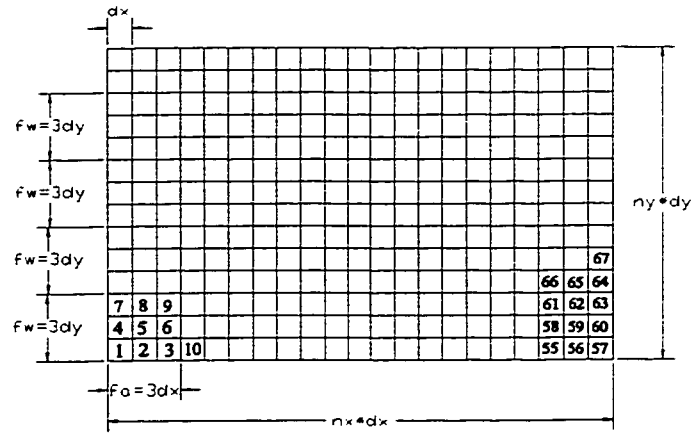


Figure 3.2: Plan view of mining scheme.

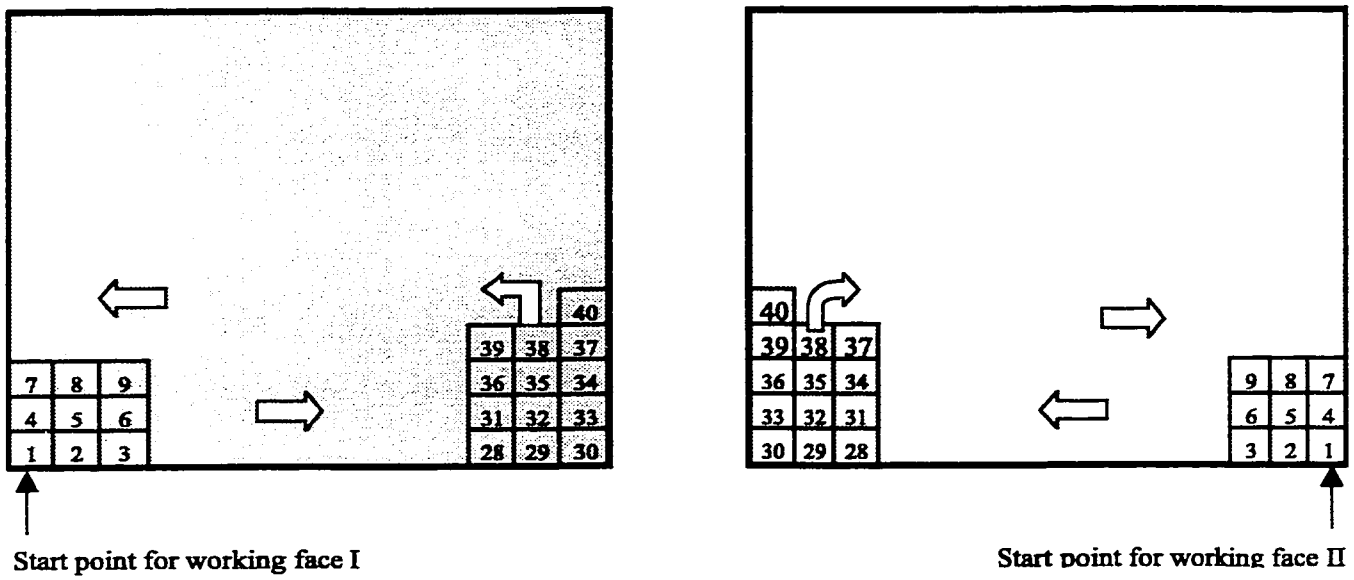


Figure 3.3: Mining scheme of two working faces.

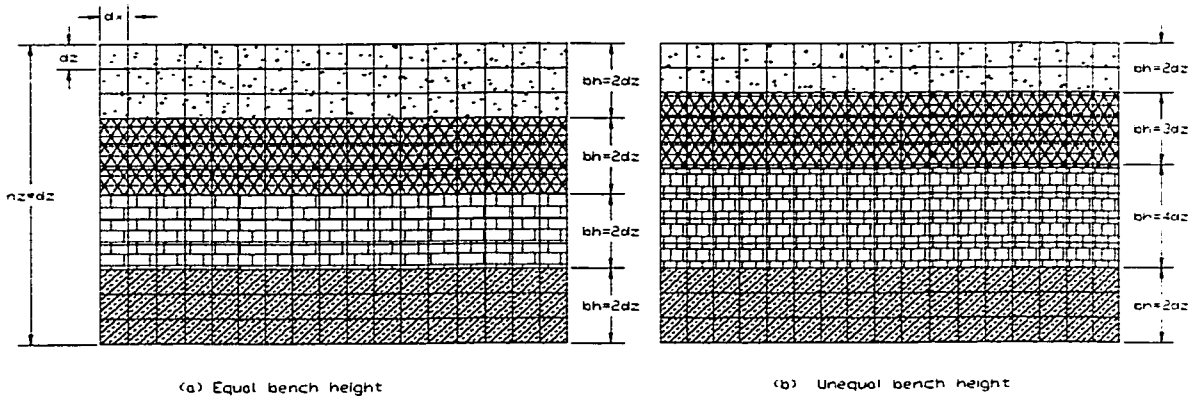


Figure 3.4: Two different bench height selection.

Mining scheme			Time variation
Cell Sequence *l*	Cell (i,j,k)	Cell grade	Average Grade G_l in time interval t_f=9
1	(1,1,1)	g_1	$G_1=(g1+g2+g3+g4+g5+g6+g7+g8+g9)/t_f$
2	(1,1,2)	g_2	
3	(1,1,3)	g_3	
4	(2,1,1)	g_4	
5	(2,1,2)	g_5	
6	(2,1,3)	g_6	
7	(3,1,1)	g_7	
8	(3,1,2)	g_8	
9	(3,1,3)	g_9	
10	(1,2,1)	g_{10}	$G_2=(g10+g11+g12+g13+g14+g15+g16+g17+g18)/t_f$
11	(1,2,2)	g_{11}	
12	(1,2,3)	g_{12}	
13	(2,2,1)	g_{13}	
14	(2,2,2)	g_{14}	
15	(2,2,3)	g_{15}	
16	(3,2,1)	g_{16}	
17	(3,2,2)	g_{17}	
18	(3,2,3)	g_{18}	
19	(1,3,1)	g_{19}	$G_3=(g19+g20+g21+g22+g23+g24+g25+g26+g27+g28)/t_f$
20	(1,3,2)	g_{20}	
21	(1,3,3)	g_{21}	
22	(2,3,1)	g_{22}	
23	(2,3,2)	g_{23}	
24	(2,3,3)	g_{24}	
25	(3,3,1)	g_{25}	
26	(3,3,2)	g_{26}	
27	(3,3,3)	g_{27}	
28	(4,1,1)	g_{28}	$G_4=(g28+g29+g30+g31+g32+g33+g34+g35+g36)/t_f$
29	(4,1,2)	g_{29}	
30	(4,1,3)	g_{30}	
31	(5,1,1)	g_{31}	
32	(5,1,2)	g_{32}	
33	(5,1,3)	g_{33}	
34	(6,1,1)	g_{34}	
.	.	.	.
.	.	.	.
nx*ny*nz	(nx,ny,nz)	$g_{(nx*ny*nz)}$	.

Figure 3.5: Grade transform from spatial variability to time domain

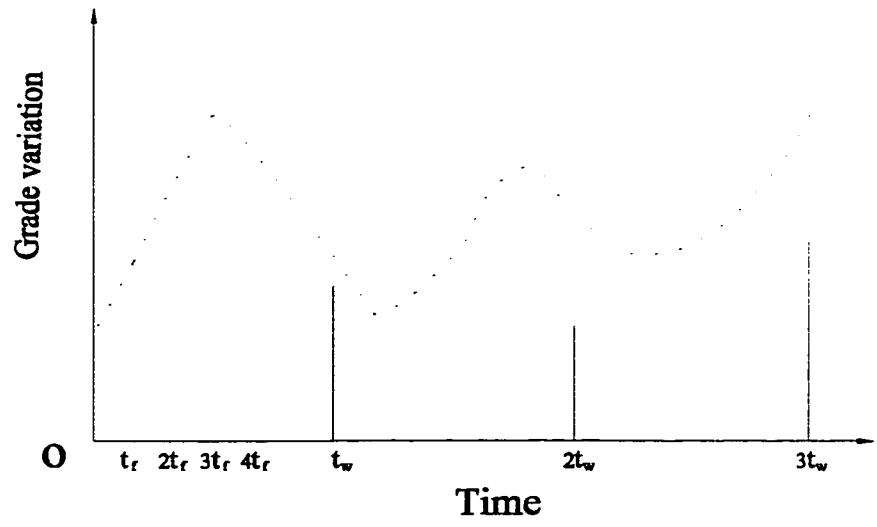


Figure 3.6: Time variation of grade for given mining scheme.

Parameters for STMOD
****

START OF PARAMETERS:

```
1                                          - 2: two working faces, 1: one working face
5.5                                        - cutoff grade
5600                                       - total number of time units
\matlab\bin\STMOD                          - output file for time variation output
back_1.out                                 - input file with simulated 3D data I
2                                          - column for variable
328       288       15                     - nx,ny,nz
1         1         1                      - Cell size
-1.E+21   1.E+21                           - trimming limits(variable limits)
1         1                                - 1: regular(1) or irregular bench(0), number of benches
15        13        15        13       15  - bench height for working face I
32        48                               - advance and width of working face I
960                                        - mining rate I (m^3/unit_time)
back_2.out                                 - file with simulated 3D data II
2                                          - column for variable
328       288       15                     - nx,ny,nz
1         1         1                      - Cell size
-1.E+21   1.E+21                           - trimming limits(variable limits)
1         1                                - 1: regular(1) or irregular bench(0), number of benches
15        13        15        13       15  - bench height for working face I
32        48                               - advance and width of working face II
960                                        - mining rate II (m^3/unit  time)
```

Figure 3.7: Example Parameter file for STmod.

3.3 Implementation of the algorithm

A C implementation of this algorithm is coded in program STMOD. The program is designed for general use by changing the parameters in the input file. Considering the variable amount of 3D data, the dynamic memory allocation has been adopted in the program. The parameters required by STMOD are listed below and are shown in Figure 3.7:

- wkface: 2, two work faces; 1, one work face.
- cutoff: cutoff grade setting.
- totalT: total number of time units.
- OutputFile: file for time variation output.
- datafile: a input data file in simplified Geo-EAS format containing the simulated data variable.
- ivr: the column number for the variable.
- nx, ny, nz: definition of the grid system of data, number of blocks in horizontal x, y and vertical z directions.
- dx, dy, dz: size of each block is dx by dy by dz.
- tmin, tmax: all value beyond the range will be ignored;
- ire-bench, nbenhts: if ire-bench is zero, all benches have same height, namely equal bench height. if ire-bench is 1, different bench height is considered;

- height1, height2..., : if the ire-bench is 1, then different bench height are selected, otherwise there exists only one bench, bench height is height1;
- fa, fw: are the length and width of mining working face in one advance of selected the mining scheme;
- rate: mining rate (t/d) in the unit time;

3.4 Examples for Sensitivity Analysis

To illustrate the effects of different parameters on time variation, we use the same data with different mining schemes and mining rates, respectively. Here we select the 3D model, 328 by 288 by 15, the corresponding size in three directions x, y and z. Some results are shown in Figures 3.8, 3.9 and 3.10, which correspond to different mining schemes by changing parameters fa, fw, *mining rate* and number of mining working faces.

3.4.1 Changing parameter fw

The $fa = 8$, $bh = 15$, $mining\ rate = 960m^3/unit\ time$ are held constant and fw is varied (8, 24, 48, 96, 144, 288). The time variation appears cyclic in Figure 3.8. This can be directly related to geological model trends and cyclic mining method. With the increase of fw, the cyclic changes in time variation is not seen. When fw reaches length of $dx \cdot nx$, 328 m, the new periodicity appears again as shown in Figure 3.8. The new periodicity occurs due to mining back and forth along Y direction. In general, this conclusion is based on observing the outcomes above: by setting other parameters in mining scheme as constants and only changing fa, the

time variation of ore grade will depend on the variability of x direction. The more variable in x direction, the greater the variability in time variation.

3.4.2 Changing parameter fa

Next, $fw = 48$, $bh = 15$, $mining\ rate = 960m^3/unit\ time$ are constrained and fa is varied (4, 8, 16, 32, 64, 328). For this setting no apparently cyclic changes in time variation can be observed, as shown in Figure 3.9. This means that ore distribution in that direction appear more anisotropic. However, when fa reaches length $nx \cdot dx = 328$ m, periodicity in time variation of grade can be observed as shown in Figure 3.9(f). This is due to the trend that mining proceeds back and forth along X direction. Similarly, we can obtain the same conclusion as in section 3.4.1.

3.4.3 Changing mining rate

Next, $fa = 32$, $fw = 48$, $bh = 15$ are constrained and *mining rate* is varied ($60, 240, 960$ and $3840\ m^3$ / *unit time*). The outcomes from these different parameter settings are shown in Figure 3.10. From these figures it can be observed that time variations become smoother with increasing of mining rate. When we increase the mining rate, we extract more blocks in same amount of time. This results in the averaging of more single block grades over a unit time, which results in less variance in the averages of each unit time. This reduces the fluctuation and it results in smoother time variation.

3.4.4 Two Mining Faces vs. Single Mining Face

Furthermore, we can observe different trends of high or low ore grade distributions for different mine sites. To reduce the variability in the plant feed, blending could be utilized between two mining faces or between two mine sites. This will smooth out the time variation. We can reduce the fluctuation of time variation by selecting the proper parameters by trial and error.

To illustrate the difference of time variation between single mining site and two mining sites, an example was constructed. For this example the prerequisite conditions is that overall mining production rate is the same, specially the mining rate for a single mining working face is equal to the combined mining rate of two working faces. All other parameters were held constant over both cases. Then, we compare the outcomes from four mining schemes as shown in Figure 3.11 and Figure 3.12. It can be observed that a smoother time variation is realized by utilizing two mining sites. The production of waste with respect to time should show similar reduction in variance as well.

Discussion

In general, it can be observed that the mining scheme has a great impact on the time variation. The extent of the impact of mining scheme depends on spatial structure of the geologic features. The mining rate does not change the general trend of time variation. In addition, an increase in the number in mining faces decreases the fluctuation of time variations due to blending.

The accurate prediction of time variation will greatly increase the recovery realized

at the process plant. Optimized low variability feed depends on the correct setting of the mining and blending parameters. The desired mining parameters depend on pit grade, reasonable mining scheme and fine blending. To meet this requirement, we need to evaluate the time variation in detail and perform the corresponding sensitivity tests. This chapter has demonstrated a potential method for accomplishing this time variance modelling and analysis.

Time variation is modelled in the time interval size assigned to ore, since ore may be extracted in a smaller volume unit than waste. Note, it is possible to scale up to a larger support size, but it is not possible to scale down. The time variation of ore grade in this small scale does not provide appropriate information for the processing plant to make decisions. The small time intervals must be scaled up through linear averaging to the support size handled by the plant. This specific plant support size is dependent on the blending and stockpile facilities at the plant.

Due to the varying market factors, such as interest rates, price fluctuations, the cost of labor and time value of money, processing plants need to adjust the cutoff grade or production output rate. The ability to dynamically change the operation plan is essential for a plant to optimize profit. By combining all significant factors mentioned above, plants can dynamically alter the operation strategy provided that the pertinent grade variation information is available. In the future, further post processing tools should be developed to aid in the design of homogenization facilities and stockpile size.

In actual operations, the processing plants deal with the larger time interval than that we used in STMOD. Therefore, there are two common problems required to be solved for the processing plant. The first is the distribution of proportion of ore to

waste during equal time intervals. Another is the distribution of waste time variation assumed that amount of ore production are constant. In the next chapter we need to build a new model to deal with these problems.

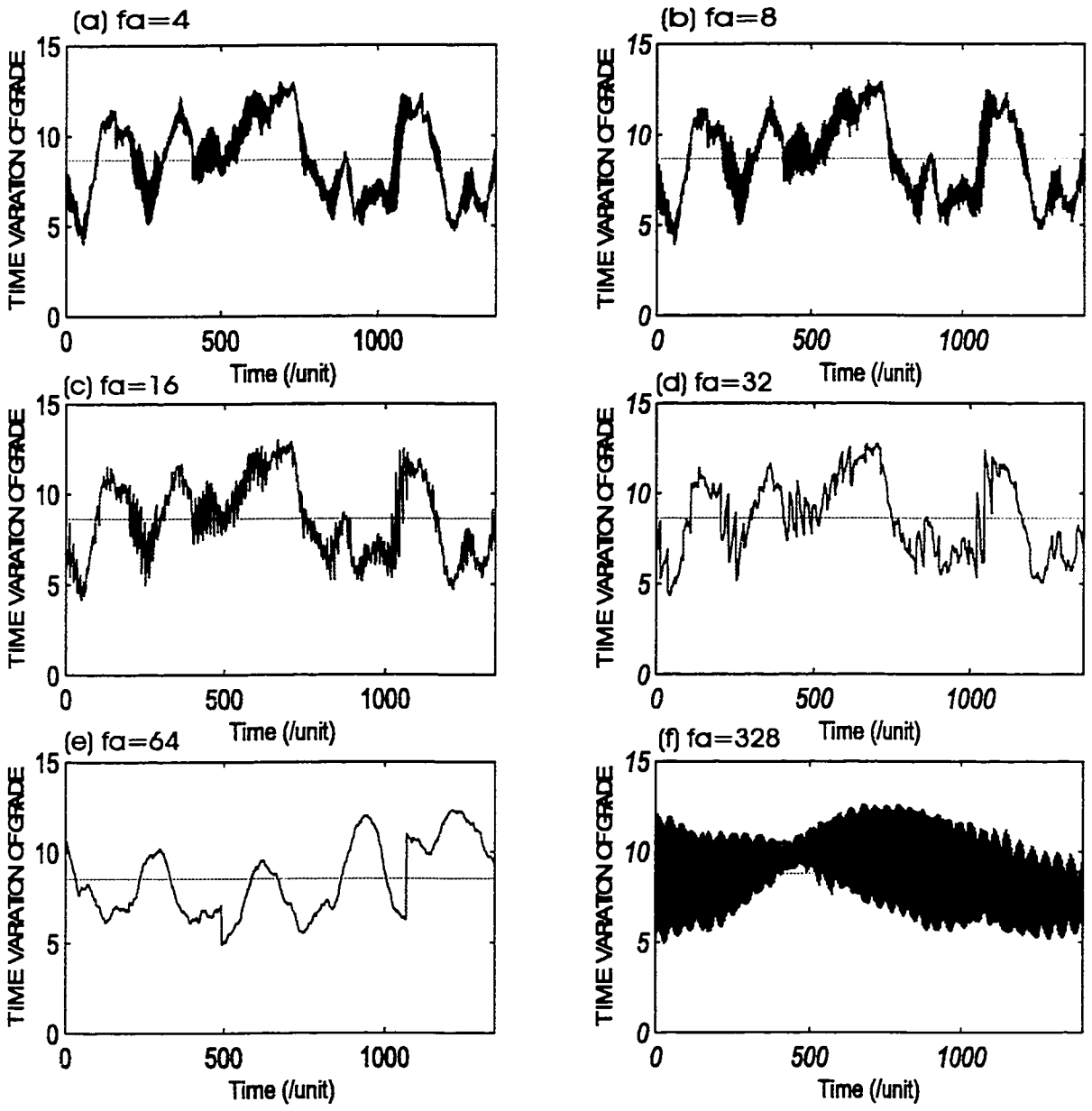


Figure 3.8: Time variation with $fa = 8$ and different fw in the mining scheme.
(a) fw=8; (b) fw=24;
(c) fw=48; (d) fw=96;
(e) fw=144; (f) fw=288;

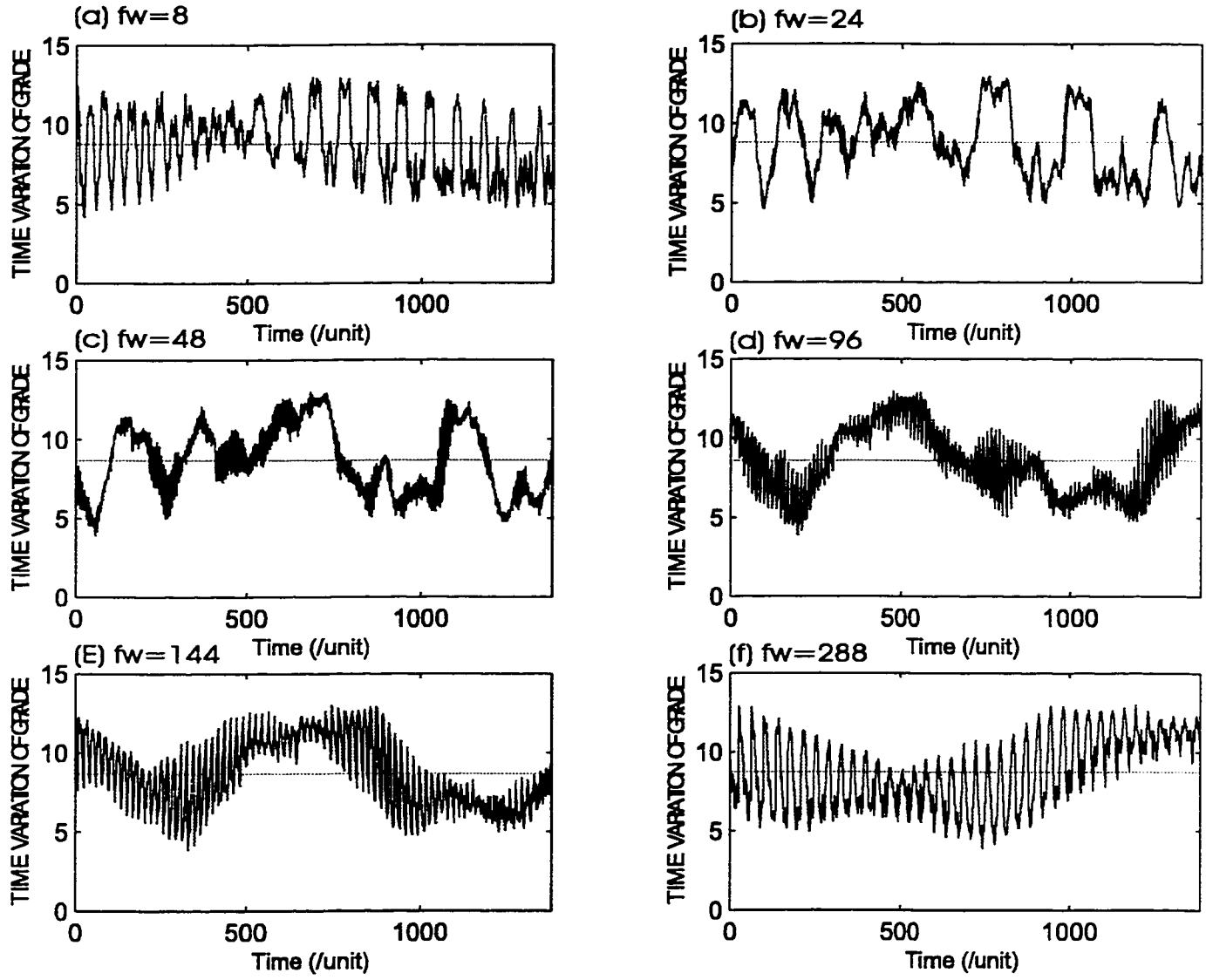


Figure 3.9: Time variation with $fw = 48$ and different fa in the mining scheme.
(a) fa=4; (b) fa=8;
(c) fa=16; (d) fa=32;
(e) fa=64; (f) fa=328;

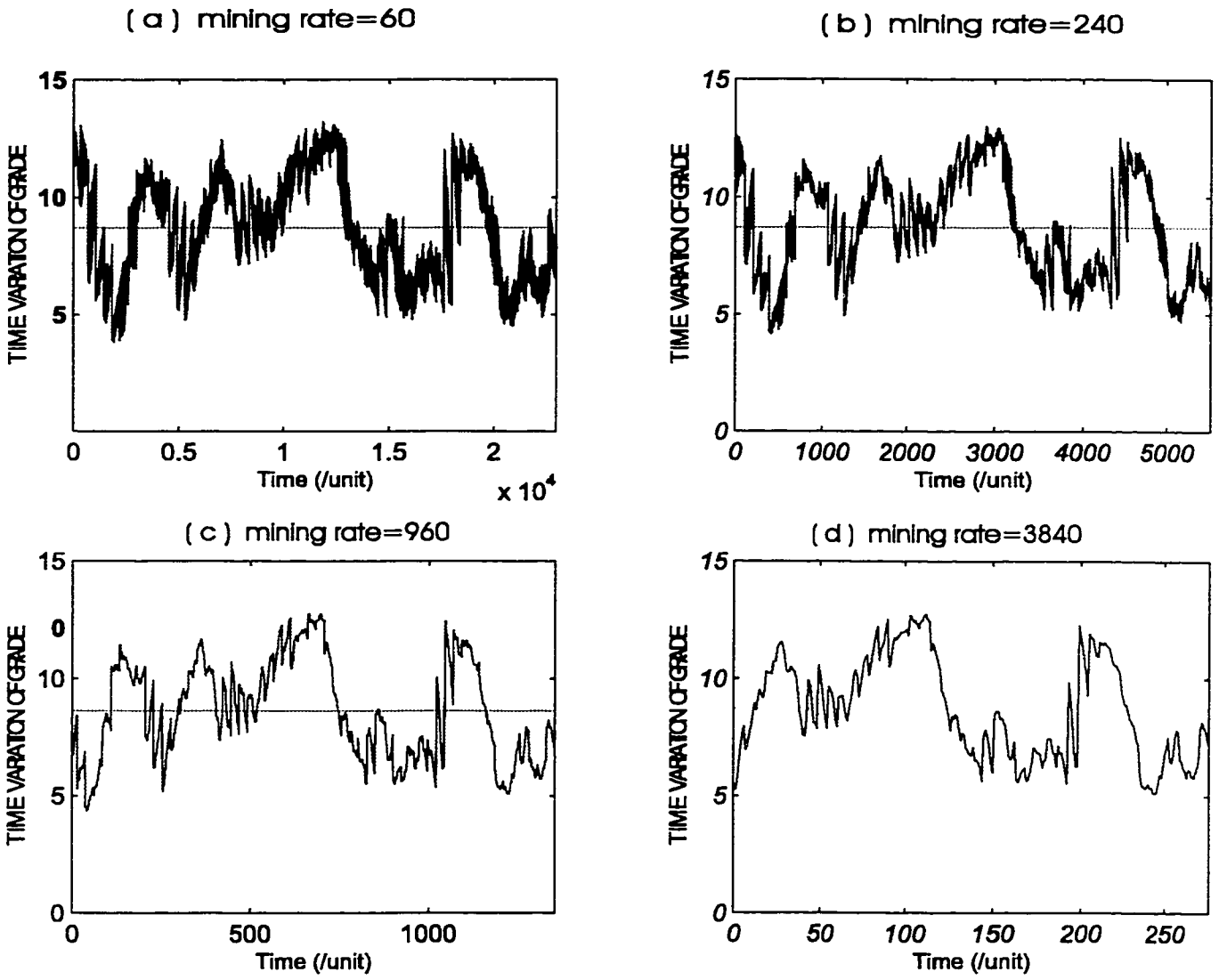


Figure 3.10: Time variation with $fa = 32$, $fw = 48$ and different mining rates.
(a) Mining rate=60; (b) Mining rate=240;
(c) Mining rate=960; (d) Mining rate=3840;

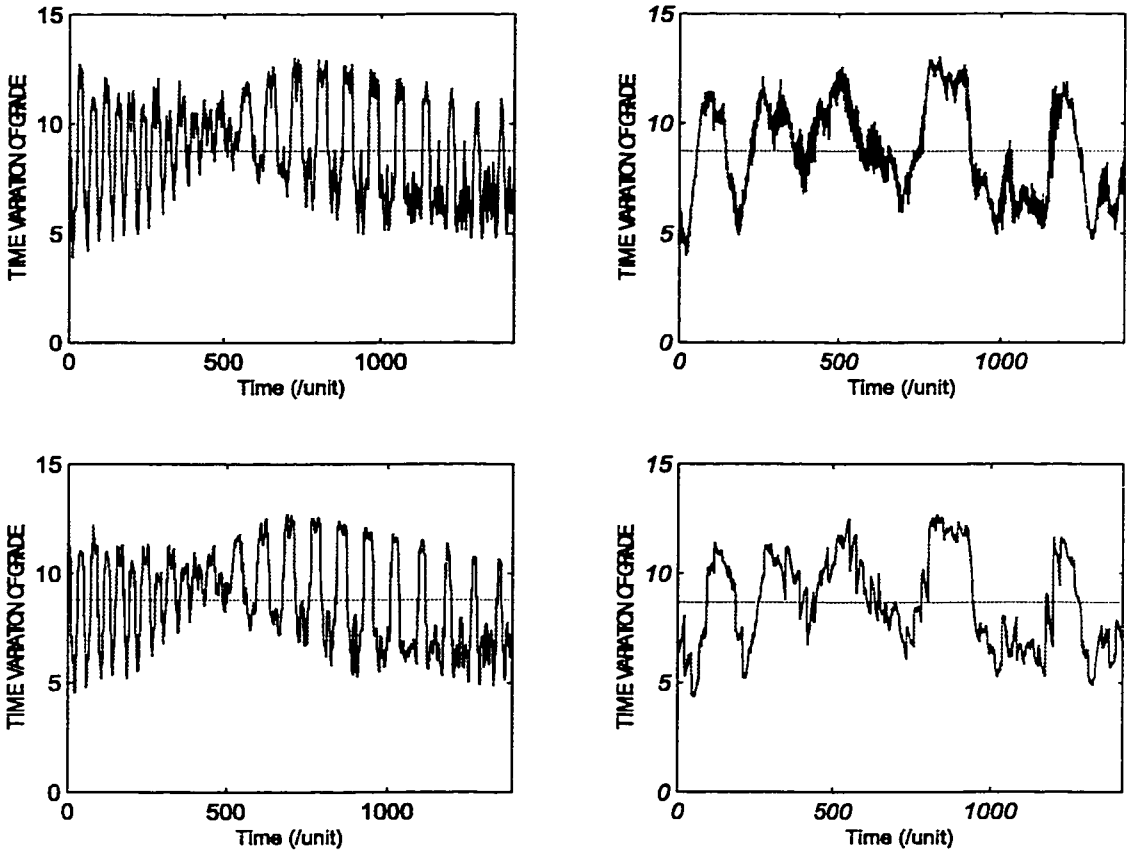


Figure 3.11: Time variation of ore grade from four different mining schemes for single working face with *mining rate* = 960 / *unit time*. Left top: $fa = 8, fw = 8$, Right top: $fa = 8, fw = 36$, Left bottom: $fa = 41, fw = 8$, Right top: $fa = 41, fw = 36$

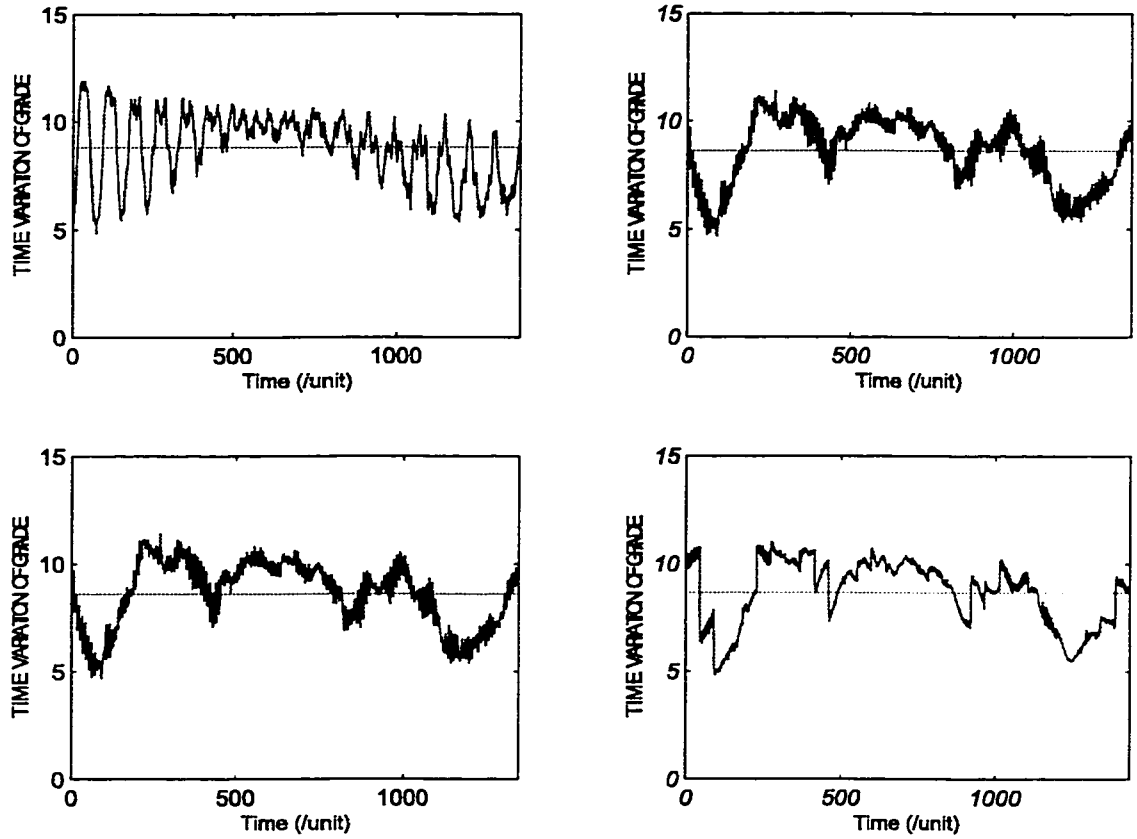


Figure 3.12: Time variation of ore grade from four different mining schemes for two working faces with *mining rate* = 960 / *unit time*. Left top: $fa = 8, fw = 8$, Right top: $fa = 8, fw = 36$, Left bottom: $fa = 41, fw = 8$, Right top: $fa = 41, fw = 36$

Chapter 4

Post Processing for Time Variation

Spatial-Temporal Modeling (STMOD) has been used to predict the time variation of mining process. However, the processing plant can not completely depend on the outcomes from time variations for decision-making. They also need to know the variation between ore and waste in certain larger time periods, says production statistics for one-week time interval. Post-Processing analysis is designed to conduct this process simulation in detail.

4.1 Introduction

In mining operation, the two factors, mining rate and production rate (ore mining rate, not including waste), are of direct interest to mining engineers. Before evaluating the mining process, the mining rate must be known. The cutoff grade is key factor to determine the production rate. The ore reserve estimate can be modeled by the widely applied geostatistical tools [1, 2, 3, 4, 6]. First, the time variations of the ore grade distributions will be produced by STMOD program. Next, we can determine the production rate by the cutoff grade, mining rate and time variations of grade. Here it is assumed that the mining proceeds with the same mining scheme as used in

the calculation of time variation. In this way, we can evaluate the statistical behavior of the mining processes. This statistical behavior includes the distribution of waste to the amount of ore constant and the proportion distributions of ore and waste during sequential time intervals.

Due to market factors, such as interest rate, price fluctuation, and the cost of labor, time value of money, the processing plant need to tune the mining schedule for obtaining the maximum profit. This is a dynamic process. On one hand, plants need the proportional distributions of ore and waste during equal-time intervals. This includes the forecasting of the maximum production rate and maximum waste rate, the average waste and ore production during the mining process. The selection of equipment would be required to handle with the maximum amount of ore. The stockpile and dump are designed to consider the maximum amount of ore and waste during certain larger time intervals. On the other hand, if the plant needs to keep the unchanged production rate (ore rate), they must take waste production into account. Due to the fluctuation of time variation of ore grade, the time spent on obtaining the equal amount of ore is always varying no matter what mining rate plants run at. Correspondingly, the amount of waste in the equal-time intervals is also subject to changing. The stockpile size are directly related to the amount of waste produced in this time period if we need to keep amount of ore production constant. Combining all factors mentioned above, plants can dynamically alter the operation strategy provided that plants could obtain necessary information.

4.2 Application of STMOD in Mineral-Processing

4.2.1 Design of homogenization Facilities and Stockpile Size

An important and difficult feature to specify is the size of stockpile required to achieve the required reduction in material variability between successive stockpiles. If the design is insufficiently precise then the stockpiles must be made extra large for safety, causing excessive cost[Parnaby et., 1973, p. 325].

The output of a mine is a time series of ore grades and ore tonnages. Stockpile homogenization systems are designed to filter out some of the variability in this time series. To a close approximation, the effect of a stockpile prehomogenizer can be concisely expressed, in geostatistical terms, as a regularization of time series. If the mine-output time series were stationary and observable over a long period of time, the effectiveness of any homogenizer could be calculated straightforwardly from the variogram of time series [11].

If both tonnage and grade time series are available, the time series might be converted to a "tonnage series", in which grade is expressed as a function of the cumulative tonnage of ore delivered to the stockpile. The tonnage-series representation is the correct choice if the stacker is designed to stack a constant tonnage per unit length along the axis of the stockpile. This is the only approach available in a simulation and grades delivered to the stockpile. A detailed simulated time series of tonnage would involve some modeling of the hauling and crushing systems, or reliance on historical tonnage data. This level of detail is not justified in most simulation studies of homogenization systems, where the main parameter of the interest is the size of the pile. The rate at which the pile is built is of little or no relevance in determining pile sizes.

The principal objective of a homogenization system is to reduce the variability

$D^2(Q, L)$ of a raw material to be used in the plant. The support Q is a critical sampling quantity that is actually determined by the sampling system and quality-control objectives of the plant, but in this discussion it is more convenient to identify Q with the amount of material contained in one "slice" recovered from the face of the stockpile by a reclaiming device [15, 11]. Some reclaimers, such as bucket wheels, move cyclically back and forth across the face of the pile, so that a slice can be well defined as the material reclaimed from the pile during a single cycle. Some more elaborate reclaimers attempt to recover material from the whole face simultaneously and continuously, so that the definition of a slice may become arbitrary. The support "L" may be taken to be "∞" if the plant operates continuously for a long period of time, or it may be some finite tonnage of material if the plant works on a " batch" basis. Each variance can be decomposed into "slice" and "pile" terms, i.e.,

$$D^2(Q, L) = D^2(Q, P) + D^2(P, L) \tag{4.2.1}$$

where "P" is the amount of material contained in a full stockplie. A stockpile homogenizer of a given capacity is able to reduce only $D^2(Q, P)$, the slice-to-slice output variability from an individual pile. $D^2(P, L)$, the pile-to-pile output variability, remains equal to $D^2(P, L)$. However, $D^2(P, L)$ can be decreased by increasing the size of pile or by somehow decreasing the large-scale variability of the input time series, e.g., by selective mining, or by blending mined materials from several faces into each pile.

The effect of a stockpile homogenizer can be represented exactly as a regularization if $D^2(Q, P)$ can be reduced to zero, e.g., if the material within the stockpile has been completely homogenized. In theory this could be accomplished by building up the pile from a practically infinite number of very thin layers. In practice this variance never

vanishes and does not usually decrease noticeably with the number of layers increase. However, if the mill has a large enough capacity to mix several slices of material together, or if the plant has a powder homogenization silo, the output from the total stockpile-mill-silo system may have very nearly a zero $D^2(Q,P)$. In a simulation study, we can normally assume that the stockpile will be built a sufficient number of layers to make $D^2(Q,P)$ small in comparison to $D^2(P,L)$.

4.2.2 Design of Proportioning Facilities

If the time-series realizations (real or simulated) are available for all (or at least all "variable") raw materials to be used by a plant, valuable information can be gained on the degree of flexibility that should be built into the plant's proportioning facilities [11]. The solutions to deterministic long-term proportioning problems provide only an impression of the "typical" amount of each raw material that will be required by the plant. If the chemical quality specifications are tight and some of the raw materials are variable in composition, the actual day-to-day or even year-to-year raw-material proportions that the plant will require may depart severely from the deterministic solution. In this application the time-series data must first be expressed as "tonnage" because the rate at which material is consumed will vary according to the instructions provided by the specified proportioning algorithm. The maximum proportions suggest how much storage and feeder capacity should be installed for each material. The minimum proportion suggests whether facilities should be available to measure out very small quantities of some materials, and whether some sources of raw materials may not be used at all from time to time.

4.2.3 Dilution of ore Grades During Mining

"Dilution" is defined as the material which is below the marginal grade but is extracted along with the ore. Dilution occurs at the following steps [10]: (a). at the deposit interpretation/inventory step(internal dilution); (b). at the mine design and planning step(planned waste); (c). at the mine exploitation step(mined waste): material included because of deviations from the mining plan, and material that fall from the roof or wall. There are, of course, some compensatory phenomena from one step to the other. For instance, dilution at the exploitation stage could actually consist of mineralization above the marginal grade that was unduly omitted from the deposit interpretation or was beyond the outline of the stope contours. Typically dilution occurs when mining activities accidentally wander across an ore-waste contact and a small amount of waste material is incorporated into the recovered ore, diluting its grade. The effects of this kind of dilution on the time (or tonage) series of ore grades delivered to a processing facility can be studied by simulation, provided that the ore grades in question have been simulated in both ore and waste population and provided that the locations of ore and waste within the simulation domain have also been simulated. In surface mines, dilution by incompletely stripped overburden may be an important source of contamination of the ore [11].

The effects of this kind of dilution are more difficult to reproduce in a simulation, as the amount of overburden recovered depends not only on the nature of the orebody surface but also on the effectiveness of stripping operation, which is difficult to model. In some cases, a comparison of historical delivered ore grades with simulated grades of clean ore mined from the same area may permit estimation of the statistical properties of overburden contamination experienced in the past. For a new mine, one would

simply have to make "reasonable assumption" or reply on experience from similar operations.

Assessment of the extent of this kind of contamination cannot be made directly using only core-drilling records, as the unconsolidated clay and other materials filling the cavities are washed out of the rock during the drilling. The local departure of individual data from the simulation can be attributed to this results. For practical mining, to reduce this problem, we can adopt the geostatistical tools to predict the range of this ore fluctuation mined according to the specified proportioning requirement.

4.3 Method and Application for Post-processing

Assume that we know the time variations of ore grade g_i, $i = 1, 2, 3, ...n$, cutoff grade g_{cutoff}, and the mining rate $MiningRate$. The time variations of waste based on obtaining the constant amount of ore can be calculated as follow:

$$\sum_{i=1}^{i=K} t_i \cdot MiningRate = SizeOre \tag{4.3.1}$$

$$\sum_{i=1}^{i=K} (1 - t_i) \cdot MiningRate = SizeWaste \tag{4.3.2}$$

$$\begin{cases} 0 < t_i \leq 1, & g_k \geq g_{cutoff}, \\ 0, & \text{otherwise.} \end{cases}$$

where i=1, 2, 3, ... , K, K is the number of time interval needed to obtain equal amount of ore production $SizeOre$. Similarly, the waste distributions $SizeWaste$ can be obtained according to the equation (4.3.2).

On the other hand, in a series of larger equal-time intervals T, the proportional distribution of ore versus waste could be described as follows:

$$OreOfT = \sum_{k=1}^{k=K} t_k \cdot MiningRate \tag{4.3.3}$$

$$WasteOfT = \sum_{k=1}^{k=K} (1 - t_k) \cdot MiningRate \tag{4.3.4}$$

$$\begin{cases} 0 < t_k \leq 1, & g_k \geq g_{cutoff}, \quad k = 1, 2, 3, ...T \\ 0, & \text{otherwise.} \end{cases}$$

where k=1, 2, 3, ..., K, K is the number of equal-time intervals. OreOfT and WasteOfT are the amount of ore and waste during the equal time interval T, correspondingly.

4.4 Post Processing Program

For analysis and calculation, a C/C++ program has been built based on the above algorithm. The input data come from the result of time variation – STMOD. This program can provide the plant more information given the cutoff and mining rate. The time of maximum waste and the amount of waste can be predicted provided that we keep the ore production rate constant. Then, the requirement of maximum handling capacity of equipment for ore and waste can be determined. Totally, by this program we could realize the following estimations: time variation of waste for the whole mining process, the distributions of ore and waste in bigger equal-time interval T, waste variation by keeping the same amount of ore, the maximum value of waste for getting same amount of ore, and time to reach maximum waste. The parameters shown in Figure 4.1 for the program are listed as follow:

- datafile: a data file in simplified Geo-EAS format containing the time variation of the mine site;
- ivr: the column number for the variable;
- nx : definition of the size of data variable;
- tmin, tmax: all value beyond the range will be ignored;
- cutoff: cutoff grade for the mine site;
- m-rate: mining rate in the unit time;
- T: the bigger time interval, namely, number of time intervals during which the distributions of ore and waste are related to mining process. For instance, if we assume that the basic time unit is hour, and T may be chosen as 24, in this case we say that the bigger time interval is daily time interval, 24-hour interval. Then we need to determine the distribution of ore and waste during the 24-hour time interval;
- SizeOre: the amount of ore to be sent to plants for processing in a specific time interval; we also say that this item mean the processing rate of plant. However, here we are more concern about the amount of ore to be treated. In order to keep this processing rate, plant need to adjust the mining rate by the different time variation of ore grade.
- outdata: the three output data files for the post processing. First one is used to store the data of time variation of waste for reaching equal amount of ore. The second is used to store the data of ore and waste during equal time intervals.

The last one is used to store the data of time variation of waste during the whole mining process.

4.5 Case study

An example has been presented to illustrate the application of this program. Input data come from the modeling of the geostatistical application, the time variation of grade for the mining process could be produced by the STMOD program. However, STMOD program does not consider such factors as the cutoff grade, equipment and site selection requirement for mining and processing plant. The program of POST-STMOD conducts the sensitivity analysis for different potential factors. There are several evaluation indexes, such as the maximum waste mining for keeping constant ore production rate, the average value of waste for equal time intervals in the mining process, the proportional distributions of ore and waste during equal time intervals. These predictions will provide us the detailed outline for planning the production process.

First, we focus on proportional distributions of ore and waste on the equal time interval T. Time variations of ore grade can be produced by using STMOD. And we estimate an economic cutoff grade which is determined by the classification of ore and waste. Then we can use post processing program, POSTSTMOD, to predict the results we need. Figure 4.2 show the time variation of waste, the amount of waste to be produced by keeping equal amount of ore production, proportional distribution of ore versus waste during the equal time intervals. In order to illustrate the extent of the impact of cutoff grade to production process, three different cutoff grades have been selected and get the different results. Figure 4.2, 4.3 and 4.4 correspond to

cutoff 4.5, 6.5, 8.5. As the cutoff increases, more wastes is encountered, see Figure 4.2, 4.3.

Second, considering a bigger time interval T by keeping same ore production rate, we can obtain the distributions of waste during the consecutively equal time interval T. Figure 4.5 shows another result of PostStmod, in which cutoff grade = 6.5, number of time interval =500, $MiningRate$=20. In Figure 4.3 and 4.5, we kept the same cutoff grade, but different mining rate and number of time intervals. Now let's take a look at the corresponding outcomes. On one hand, due to same cutoff grade, the upper parts of Figure 4.3 and 4.5 should look same. On the other hand, because of different mining rate and different number of time intervals, the times of arriving local maximum waste have been changed, and the time interval of waste variation has been changed as well.

4.6 Future work

We also need to estimate the amount of waste and the amount of ore during equal time intervals assuming that we have known specific gravity of ore. Furthermore, we assumed that the production price and cost of operation are provided, we can directly evaluate operation cost and project profit.

```
                    Parameters for PostTime
                    **********
START OF PARAMETERS:
stmod.out                    - input file of time variation
1                            - column for variable
6000                         - size of data
6.5                          - cutoff grade
-1.0e21   1.0e21             - trimming limits(variable ore limits)
20                           - mining rate in unit time interval
500                          - number of time interval
7500                         - amount of ore to plant in time interval T
PostStmod.out                - Output file for keeping the time variation of waste
PostStmod.rat                - file for ratio of ore to waste
PostStmod.rak                - file for ratio of ore to waste
```

Figure 4.1: An example of parameter file for PostStmod program.

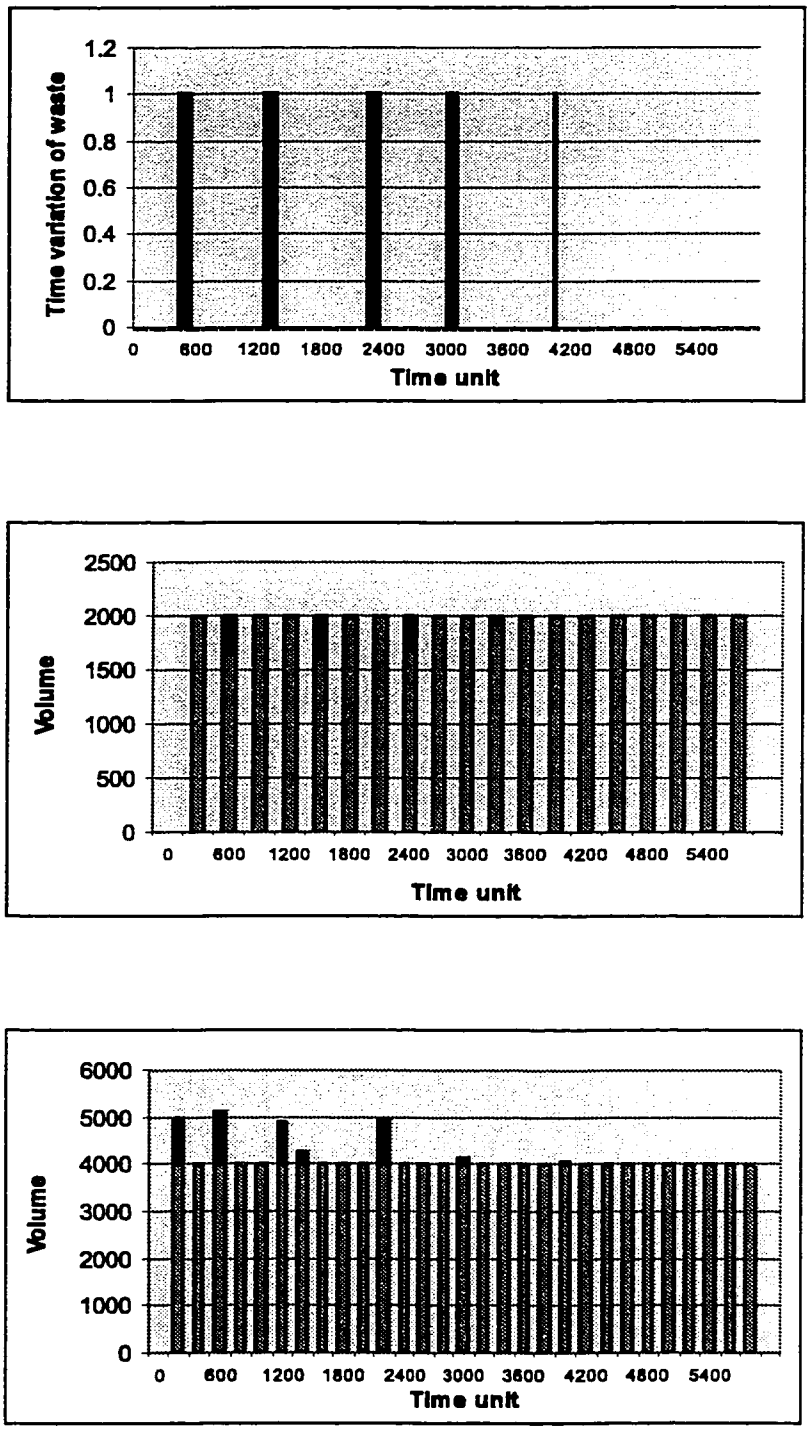
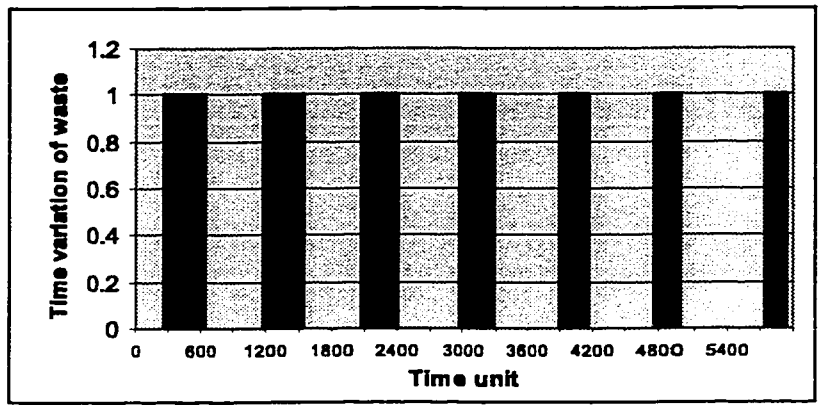
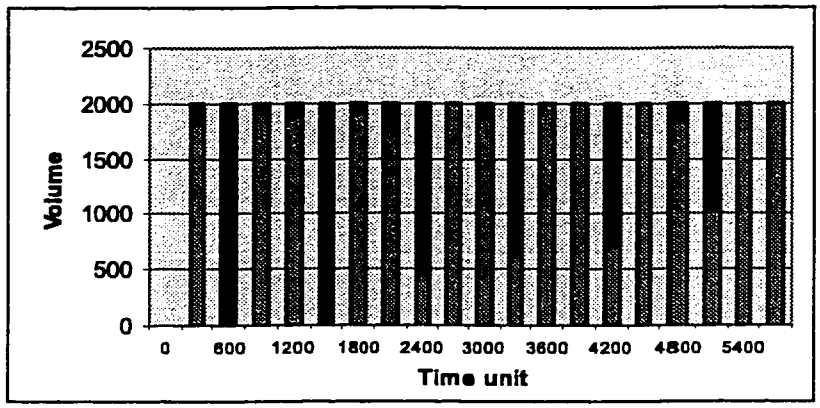


Figure 4.2: Results from PostStmod program, cutoff grade is 4.5, MiningRate 20, SizeOre 40000, number of time interval 300.
Upper: time variation of waste for the whole mining process;
Middle: Ore versus waste during the equal time intervals;
Lower: Time variation of waste during the time period for getting equal ore production.





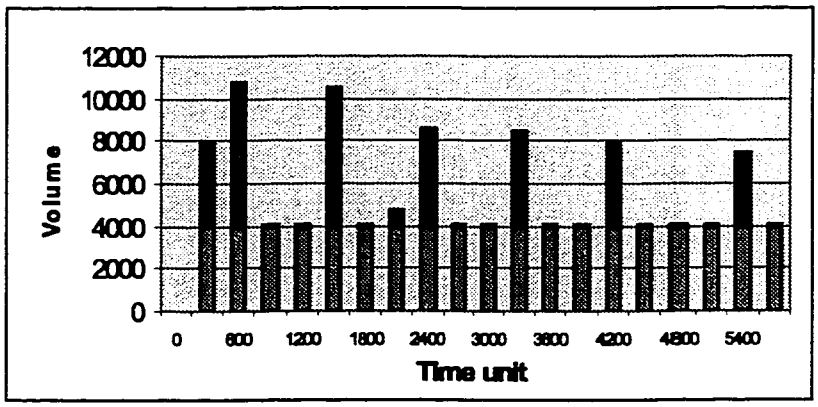
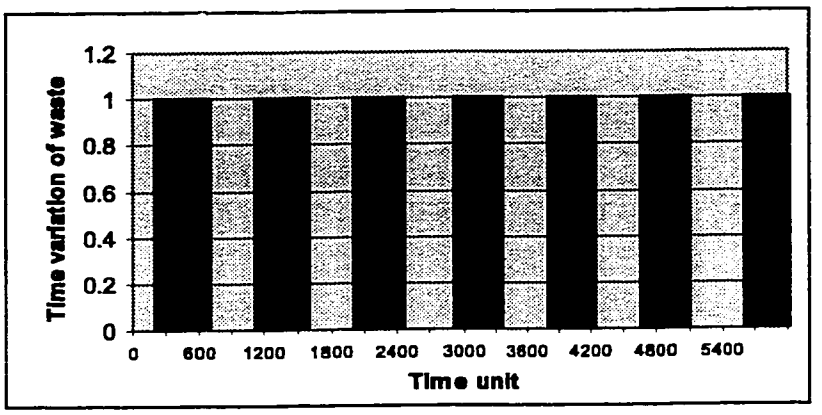
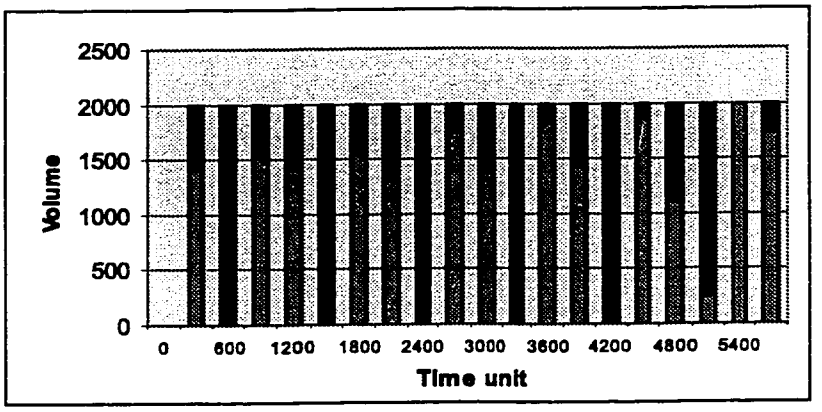


Figure 4.3: Results from PostStmod program, cutoff grade is 6.5, MiningRate 20, SizeOre 400000, number of time interval 300.
Upper: time variation of waste for the whole mining process;
Middle: Ore versus waste during the equal time intervals;
Lower: Time variation of waste during the time period for getting equal ore production.





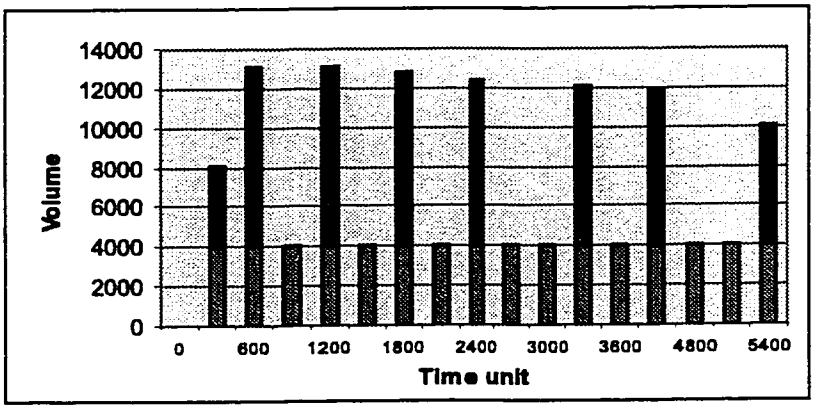
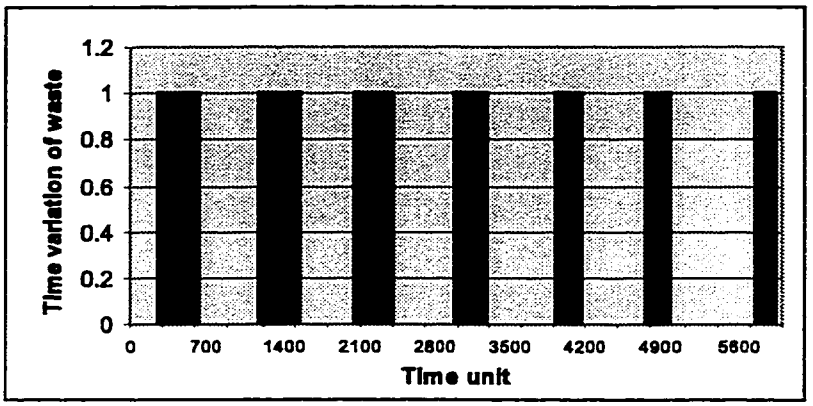
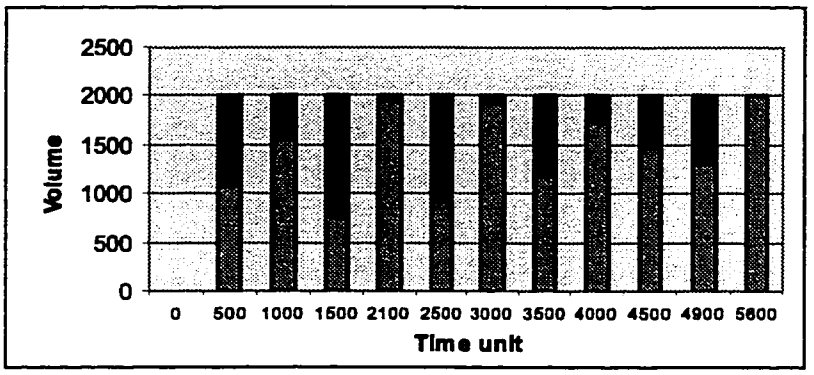


Figure 4.4: Results from PostStmod program, cutoff grade is 8.5, MiningRate 20, SizeOre 40000, number of time interval 300.
Upper: time variation of waste for the whole mining process;
Middle: Ore versus waste during the equal time intervals;
Lower: Time variation of waste during the time period for getting equal ore production.





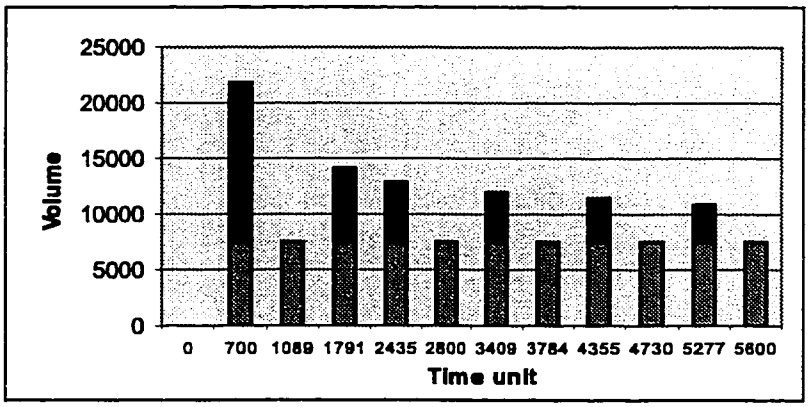


Figure 4.5: Results from PostStmod program, cutoff grade is 6.5, MiningRate 20, SizeOre 7500, number of time interval 500.
Upper: time variation of waste for the whole mining process;
Middle: Ore versus waste during the equal time intervals;
Lower: Time variation of waste during the time period for getting equal ore production.

Chapter 5

Case studies

Bitumen and fines content are important factors affecting hydrocarbon recovery in oil sands extraction. Estimates of bitumen and fines prior to the production process has potential to greatly increase recovery. A detailed case study for oil sands is presented in this chapter.

5.1 Introduction

Canada's oil sands are the emerging giant of the petroleum industry. Currently there are numerous oil sands projects and expansions in the construction, design or conceptual stage representing billions dollar investments over the next 9 years. Production will increase by 470% and represent about 50% of Canada's crude oil production[10].

The practices in mineral field are elusive targets, given the variation of geological conditions, mining methods, the great variation of grade, and information effect. As a result, the investment required for mine development often carries undue risks regarding reserve quality and quantity, capacity to develop the mine within budget and schedule, a capacity to meet the mine production targets: startup date, length

of break-in period, costs, quantity and quality, and projected cash-flow. All these properties always play a vital role in decision-making in mine operations.

The processing plant needs to understand time variations of ore grade to reduce the natural variability of incoming raw materials and to blend different materials into plant feeds. Processing parameters, like the amount of solvent, amount of energy consumed, can be adjusted accordingly. Therefore, it is useful to know the *bitumen* and *fine* content prior to processing.

For this purpose, the time variations of bitumen and fines content in different volumes transported from oil sands field to the operation plant are estimated. Based on the spatial variability of oil sands, the transform from spatial continuity to time variation is realized through a mining scheme. The algorithm for time variation was presented in previous chapters, a detailed case study is shown here.

The analytical variogram models have been established by applying geostatistical theory. Due to the different volume support from core sample data to simulation cell, up-scaled variograms and histograms have been presented in this study. We know that bitumen and fines are related. Bitumen sample data are often collected and data are rarely collected for fines. Then, bitumen content is first simulated using Gaussian sequential simulation. Cosimulation is adopted for fines. The average changes of time variation and variograms of time variation are also analyzed.

5.2 Data

The data comes from an oil sands deposit in the Fort McMurray area. There are 875 vertical drillholes over a 40 km area. *Bitumen* content are measured for 47331 samples and *fines* content was measured for 16153. Figure 5.1 and Figure 5.2 show

the histograms for *bitumen* and *fines*. The *bitumen* content appears to have a bimodal distribution related to either porosity facies or water saturation. The *fines* do not show this behavior. Figure 5.3 shows the cross plot between *bitumen* and *fines* for 15030 of collocated sample pairs. A fairly strong negative correlation with correlation coefficient of -0.71 can be seen from Figure 5.3, which is expected since the existence of *fines* reduces the available space for *bitumen* in the pores.

We assume that about 25,000 cubic meters of oil sands are taken to the plant for processing every day. The thickness of the oil sand layer being mined is about 15 meters, which means there is about an 8 by 8 by 15 meter volume mined each hour. Since the *bitumen* and *fines* have strong spatial variability, the *bitumen* and *fines* have related time variation.

5.3 Procedures of analysis and calculation

To assess the temporal variation of *bitumen* and *fines* it is necessary to establish the relation between the time domain and spatial domain. We first need to estimate the spatial variations in *bitumen* and *fines* in the oil sand volume to be processed. This will be done through geostatistical simulation. Then, the average content of *bitumen* and *fines* in the volume for a unit time (say an hour) processing can thus be estimated.

For the spatial estimation of *bitumen* and *fines* content, we need variogram models that quantify the spatial variability of the *bitumen* and *fines*. Variograms are calculated from the sample data and then modeled. However, in geostatistical modeling, volume support of core sample is generally smaller than that used for the simulation. In this study, the core volume could be considered as quasi-point, e.g. 0.05 by 0.05 by 0.61 meter (where 0.05 meter is the radius of the drillhole and 0.61 meter is the

median length of the core samples), but the simulation cell will be realtively large, say 1 by 1 by 1 meter. Since the variance is reduced as the volume increases, the variogram model from the core samples should not be used directly in subsequent simulation. It is necessary to scale up (or down in other cases) the variogram from the core to the simulation cell. This can be accomplished analytically by considering known volume-variance relations.

Bitumen and *fines* should not be modeled independently. A model of coregionization is required for cosimulation. For this purpose, we model the experimental variograms with a linear model of coregionalization.

Most geostatistical algorithms are devised to work with the normal distribution. Therefore, the *bitumen* and *fines* variables are transformed to standard normal distributions prior to variogram calculation and simulation. The simulated values are then back transformed from the normal distribution to the units of the original data.

Overall, our procedure consists of the following steps:

- normal score transform of *bitumen* and *fines*,
- calculate experimental variograms in principal directions of continuities of *bitumen* and *fines*, and cross variogram of *bitumen* and *fines* with `gamv`,
- model direct variogram of *bitumen* and *fines* with `vmodel` for the purpose of simulation,
- construct a linear model of coregionalization (LMC) for *bitumen* and *fines* for the purpose of cosimulation,
- define a volume which is about one day's production (say 8 by 8 by 15 meter) and divided into small cells e.g., 1 by 1 by 1 meter,

- scale-up the direct variogram models and LMC from core volume support to simulation volume support. Analytical functions are based on volume-variance relations,
- estimate the content of *bitumen* for each small cell in the defined daily processing volume with sequential Gaussian simulation (`sgsim` in `gslib`),
- estimate the content of *fines* for each cell of the volume through cosimulation of `sgsim` by considering the content of *bitumen* already assigned,
- assess the temporal variation of the content of *bitumen* and *fines* by observing the correlation between variable values in successive unit time (say successive hours); average the variation in volumes of one hour processing and plot the expected value versus time,
- Quantify time variation and plot the variation.

5.3.1 Experimental Variograms, Modeling and LMC

For Gaussian simulation, data should obey a Gaussian histogram. However, in practice, data are rarely Gaussian. Therefore, data must be transformed into a Gaussian or normal distribution prior to simulation. The transformation is done by program `nscore`. Figure 5.4 shows the cross plot between normal scores of *bitumen* and *fines* of the 15030 collocated samples. The correlation does not change much with a new correlation coefficient of -0.704 compared to the raw correlation shown in Figure 5.3.

Figures 5.5 show the directional variograms and cross variogram of *bitumen* and *fines* in the two principal horizontal directions together with the omnidirectional variograms. There is little difference in the spatial continuity in the two principal

horizontal directions. Therefore, it is assumed that the spatial variabilities of *bitumen* and *fines* are isotropic in horizontal plane.

The red dots in Figure 5.6 show the vertical and isotropic horizontal experimental variograms for *bitumen* and *fines*, and the cross variogram of *bitumen* and *fines*.

In order to have legitimate covariance everywhere in the kriging, it is necessary to model the experimental variogram with legimate analytical functions:

$$\gamma(h) = C^0 + \sum_{i=1}^{nst} C^i \Gamma^i(h)$$

where $\gamma(h)$ is the variogram model, C^0 is the nugget effect, nst is the number of nested variogram structure used to fit the variogram, C^i is the variance contribution of each nested structure, $i = 1, \ldots, nst$, and $\Gamma^i(h)$ is the nested structure consisting of an analytical function such as *spherical, exponential, Gaussian etc.* The experimental variograms shown in Figure 5.6 are modeled as the follows:

$$\gamma(h)^{bitumen} = 0.35 + 0.35\, Exp_{200,13}(h) + 0.13\, Gau_{700,25}(h) + 0.17\, Gau_{5000,25}(h)$$

$$\gamma(h)^{fines} = 0.35 + 0.45\, Exp_{200,35}(h) + 0.10\, Exp_{600,40}(h) + 0.10\, Sph_{3000,60}(h)$$

The fitted variograms are also shown as red solid lines in Figure 5.6. As shown in Figure 5.6 the models above that the nugget effect is 0.35 for both *bitumen* and *fines*. That means there is 35% of total variance contributed by randomness and short scale variability. The horizontal correlation of *fines* decreases faster than that of *bitumen* as the distance increases, but the vertical correlation lengths of *fines* is larger than

that of *bitumen*. All the variance contributions and ranges of each nested structure are associated with the current sampling volume. The variance will go down and the range will be longer when the volume increases.

For the purpose of cosimulation, the variograms of *bitumen* and *fines*, and the cross variogram of *bitumen* and *fines* must be modeled together to form a licit linear model of coregionization (LMC). That means all three variograms must be modeled with the same nested structures with the same ranges and different variance contribution coefficients. The coefficients in the nested structures must satisfy constraints defined as followed:

$$C^i_{Y,Y} \geq 0, \qquad C^i_{Z,Z} \geq 0, \qquad C^i_{Y,Y} \times C^i_{Z,Z} \geq {C^i_{Y,Z}}^2$$

The LMC for the experimental variograms shown in Figure 5.6 are as follows and shown in Figure 5.7:

$$\gamma(h)^{Bitumen} = 0.35 + 0.35\, Exp_{200,15}(h) + 0.13\, Sph_{1000,30}(h) + 0.17\, Sph_{5000,50}(h)$$

$$\gamma(h)^{Fines} = 0.35 + 0.45\, Exp_{200,15}(h) + 0.10\, Sph_{1000,30}(h) + 0.10\, Sph_{5000,50}(h)$$

$$\gamma(h)^{Cross} = -0.20 - 0.30\, Exp_{200,15}(h) - 0.10\, Sph_{1000,30}(h) - 0.11\, Sph_{5000,50}(h)$$

5.3.2 Scale up variogram models and LMC from core to simulation cell

There are always volume support issues in geostatistical modeling. The variogram model, calculated from data associated with a certain volume support (data scaling), *only* specifies the spatial variability of the variables at that scale. In geostatistical simulation, smaller or larger volumes than data volume support may be preferred. It is then necessary to scale the variogram calculated from a data volume to simulation cell.

Consider a variogram model at arbitrary scale v, where v often represents the small core scale:

$$\gamma_v(h) = C_v^0 + \sum_{i=1}^{nst} C_v^i \Gamma_v^i(h)$$

where $\gamma_v(h)$ is the variogram model at the v scale, C_v^0 is the nugget effect, nst is the number of nested variogram structure used to fit the variogram, C_v^i is the variance contribution of each nested structure, $i = 1, \ldots, nst$, and $\Gamma_v^i(h)$ is the nested structure consisting of an analytical function. Here the "sill" of each nested structure $\Gamma_v^i(h)$ is unity, the C describes the variance contributions of each nested structure. The sum of the variance contribution is the variance at the v-scale and is called the dispersion variance:

$$D^2(v, A) = C_v^0 + \sum_{i=1}^{nst} C_v^i$$

where $D^2(v, A)$ is the variance of volumes of size v in the entire area of interest A. The variance decreases as the volume increases. That is because high or low values are averaged out as the volume of investigation increases. The ranges increase as the averaging volume increases. Moreover, experience has also shown that the variogram

shape does not change much. Therefore, the following scaling relations are established in the geostatistics:

- The range at a large volume V increases in volume size $(|V| - |v|)$ in each particular direction:

$$a_V = a_v + (|V| - |v|)$$

where $||$ means the size of the volume in a specific direction.

- Assuming the variogram shape does not change, we must quantify how the variance contributions $C_v^i, i = 1, \ldots, nst$, change.

For the nugget effect, it is due to random variance, and it decreases with an inverse relationship of the volume, i.e.,

$$C_V^0 = C_v^0 \frac{|v|}{|V|}$$

where $|v|$, $|V|$ represent the volume of each scale, respectively.

The variance contribution of each nested structure changes as following:

$$C_V^i = C_v^i \frac{1 - \bar{\Gamma}(v, v)}{1 - \bar{\Gamma}(V, V)}$$

where $\bar{\Gamma}(V, V)$ and $\bar{\Gamma}(v, v)$ are the average variogram or "gamma-bar" values. Notice here the change in the variance contribution is calculated separately for each nested structure. The "gamma-bar" values are thus calculated based on the unit variogram of each nested structure.

- When all variance contributions have been corrected for each nested structure separately, the coefficients $C_V^i, i = 0, 1, \ldots, nst$ of a variogram model will be normalized by the corresponding dispersion variance for normal score simulation.

The "gamma-bar" value represents the mean value of $\gamma(h)$ when one extremity of the vector h describes the domain $v(u)$ and the other extremity independently describes domain $V(u')$. In mathematical notation the "gamma-bar" value is expressed as:

$$\bar{\gamma}(v(u), V(u')) = \frac{1}{v.V} \int_{v(u)} \int_{V(u')} \gamma(y - y') dy dy'$$

Although there exist certain analytical solutions to $\bar{\gamma}(v(u), V(u'))$, the value of "gamma-bar" is usually estimated numerically by discretizing the volume $v(u)$ and $V(u')$ into a number of points and simply averaging the variogram values:

$$\bar{\gamma}(v(u), V(u')) \approx \frac{1}{n.n'} \sum_{i=1}^{n} \sum_{j=1}^{n'} \gamma(u_i - u'_j)$$

where n is the number of regular spacing point discretized for volume $v(u)$ and n' is the number of regular spacing point discretized for volume $V(u')$. Each point normally represents the same fractional volume of $v(u)$ or $V(u')$.

For the scaling up of *direct* variogram models, Table 5.1 and Table 5.2 list the $\bar{\Gamma}$ values, original and corrected range, original, corrected and normalized variance contributions after correction of each nested structure of variograms for *bitumen* and *fines*, respectively. It can be seen from Table 5.1 and Table 5.2 that *range* does not change much since the change of volume is relatively small. From row 7 of Table 5.1 and Table 5.2 it can be seen that nugget effect decreases much faster than the variance contributions of the nested structures as the volume increases. This is expected because nugget effect is due to randomness and small scale variability, the randomness will be reduced quickly with volume averaging. From row 7, one will see slight influence of the volume change on the variance contributions. However, the significant

change of the nugget effect and the normalization of all variance contribution, the variance contributions in the final corrected model, which are shown in the last row of Table 5.1 and Table 5.2, have big changes to the original variogram models. The solid line in Figure 5.8 show the variogram models after scaling.

For the LMC model, since the nested structures are the same for three variograms, the "gamma-bar" values of each nested structure are the same for the correction of variograms of *bitumen* and *fines*, and of the cross variogram of *bitumen* and *fines*. Also, the range changes in each nested structure are also the same for three variograms. Table 5.3 list the $\bar{\gamma}$ values, original and corrected range for LMC scaling up of the three nested structures. The corrected and normalized variance contributions after correction are tabulated in Table 5.4 for each nested structure together with the variance contributions in the original LMC. The LMC after scaling up are shown as green solid lines in Figure 5.9. Note that the constraints for variance contributions of a LMC still hold.

Compare the variogram models after scaling up to the simulation volume (solid lines in Figure 5.8 and Figure 5.9) with the original variogram models in the core volume (dash lines in Figure 5.8 and Figure 5.9), there is higher connectivity in the shorter scales.

5.3.3 Correct Histogram Using Variance-Volume Relation

The variogram models have been scaled up from the drillhole vloume to the simulation volume. It should be noted that the histogram will also be changed after such volume change. For the correction of histogram associated with the support volume effect, we need to have the mean, variance value and shape of the new histogram. The

	nugget effect	Γ_h^1	Γ_h^2	Γ_h^3
$\bar{\gamma}(v,v)$		0.0449	0.000891	0.000891
$\bar{\gamma}(V,V)$		0.0733	0.002386	0.002380
a^{orig}		200.0,200.0,13.0	700.0,700.0,25.0	5000.0,5000.0,25.0
a^{corr}		200.9,200.9,13.4	701.0,701.0,25.4	5000.9,5000.9,25.4
C^{orig}	0.35	0.35	0.13	0.17
C^{corr}	0.0005	0.3396	0.1298	0.1697
C^{norm}	0.0008	0.5309	0.2029	0.2654

Table 5.1: $\bar{\gamma}$ values, original and corrected range, original, corrected and normalized variance contributions of variogram model of *bitumen*

	nugget effect	Γ_h^1	Γ_h^2	Γ_h^3
$\bar{\gamma}(v,v)$		0.0171	0.01495	0.000891
$\bar{\gamma}(V,V)$		0.0299	0.02477	0.008290
a^{orig}		200.0,200.0,35.0	600.0,600.0,40.0	3000.0,3000.0,60.0
a^{corr}		200.9,200.9,35.4	601.0,601.0,40.4	3001.0,3001.0,60.4
C^{orig}	0.35	0.45	0.10	0.10
C^{corr}	0.0005	0.4441	0.0990	0.0.0993
C^{norm}	0.0008	0.6908	0.1540	0.1544

Table 5.2: $\bar{\gamma}$ values, original and corrected range, original, corrected and normalized variance contributions of variogram model of *fines*

	Γ_h^1	Γ_h^2	Γ_h^3
$\bar{\gamma}(v,v)$	0.03885	0.01008	0.006085
$\bar{\gamma}(V,V)$	0.06415	0.01658	0.009881
a^{orig}	200.0,200.0,15.0	1000.0,1000.0,30.0	5000.0,5000.0,50.0
a^{corr}	200.9,200.9,15.4	1001.0,1001.0,30.4	5000.9,5000.9,50.4

Table 5.3: $\bar{\gamma}$ values, original and corrected range, of LMC of *bitumen* and *fines*

	nugget effect	Γ_h^1	Γ_h^2	Γ_h^3
	Bitumen			
C^{orig}	0.35	0.35	0.13	0.17
C^{corr}	0.0005	0.3408	0.1291	0.1694
C^{norm}	0.0008	0.5326	0.2019	0.2647
	Fine			
C^{orig}	0.35	0.45	0.10	0.10
C^{corr}	0.0005	0.4381	0.0993	0.0996
C^{norm}	0.0008	0.6871	0.1558	0.1562
	Cross			
C^{orig}	-0.20	-0.30	-0.10	-0.11
C^{corr}	-0.0003	-0.2921	-0.0993	-0.1096
C^{norm}	-0.0004	-0.4137	-0.1407	-0.1552

Table 5.4: original, corrected and normalized variance contributions of LMC of *bitumen* and *fines*

global mean remains unchanged when the support volume changes. Variance will be reduced when the support volume increases. The magnitude of the variance change is predictable as documented above.

There is an additivity relation (*Krige* relation) in the dispersion variance:

$$D^2(.,A) = D^2(.,v) + D^2(v,A)$$

The quasi-point dispersion variance, i.e., total variance $D^2(.,A)$ in the entire area A is the *within* blocks variance, $D^2(.,v)$ plus the *between* block variance, i.e., the block dispersion variance $D^2(v,A)$ in the entire area A.

We can calculate average covariance values(or C-bar) in a similar manner to gammar-bar values. We simply replace $\bar{\gamma}$ by $\bar{C}$, which is equal to $C(0) - \bar{\gamma}$ after the decision of stationarity. Thus,

$$D^2(v,A) = \bar{\gamma}(A,A) - \bar{\gamma}(v,v)$$

	$\bar{\gamma}(v,v)$	f
bitumen	0.3796	0.6214
fines	0.3671	0.6329

Table 5.5: $\bar{\gamma}$ values and variance reduction factor f of *bitumen* and *fines*

When A is sufficiently large, $\bar{\gamma}(A,A) \longrightarrow \sigma^2$ and the dispersion variance may be written:

$$D^2(.,A) = \sigma^2 - \bar{\gamma}(.,.) = \sigma^2$$

When we consider quasi-point volumes within any arbitrary volume v the dispersion variance is written simply as:

$$D^2(.,v) = \bar{\gamma}(v,v) - \bar{\gamma}(.,.) = \bar{\gamma}(v,v)$$

Then a variance reduction factor f which is the ratio of the block dispersion variance $D^2(v,A)$ and the quasi-point dispersion variance $D^2(.,A)$ is defined as:

$$f = \frac{D^2(v,A)}{D^2(.,A)} = \frac{D^2(.,A) - D^2(.,v)}{D^2(.,A)} = 1 - \frac{D^2(.,v)}{D^2(.,A)} = 1 - \frac{\bar{\gamma}(v,v)}{\sigma^2}$$

Table 5.5 shows the "gammabar value" and variance reduction factor f of *bitumen* and *fines*.

The last issue about the histogram correction is the "change of shape". The affine correction and indirect correction based on permanence of a lognormal distribution model are two easily applied analytical models for change of histogram shape. For our case study the sample histogram before variance reduction is not lognormal, then we choose the affine correction.

The basic idea behind *affine correction* is that the variance of a distribution can be reduced without changing its mean simply by moving all the values closer to the mean.

The *affine correction* transforms q, a quantile (or value) of one distribution, to q', a quantile (or value) of the corrected distribution using the following linear formula:

$$q' = \sqrt{f} \times (q - m) + m$$

where f is the variance reduction factor and m is the mean value.

Figure 5.10 and Figure 5.11 are the histogram of *bitumen* and *fines* after *affine correction*. It is noted that the ratio of variances of the distributions of *bitumen* and *fines* before and after the transformation are 0.6246 ($\frac{4.07^2}{5.15^2}$) and 0.6324 ($\frac{19.03^2}{23.93^2}$), respectively, which are consistent with the variance reduction factors f, 0.6214 and 0.6319 as shown in Table 5.5. The shape change in the histograms and averaging effect can be noticed from the *minimum* and *maximum* values of the histograms *before* and *after* the correction. *Maximum* values become smaller and *minimum* values become larger after the correction.

5.3.4 Mining scheme

We assume there are about 1000 m^3 of oil sands treated per hour in the plant. we are trying to estimate the time variation in different time scales, say hourly, six-hour, 12-hour, 24-hour and up to monthly variations. Seven weeks of operation is needed to establish this time variation, it is then needed to simulate volume for $7 \times 7 \times 24 = 1176$ hours operation. It is assumed that the oil sand bench is about 15 meters and the width of the mine face is 8 meters, we define one hour's digging volume to be $8 \times 8 \times 15 = 960m^3$ (approximate 1000 m^3). To facilitate the average calculation, field size of 328 by 288 by 15 meter is designed for the simulation, namely, parameters $L = fw = 288, W = 328, H = 15, a = b = 8$ (unit: meter) in Figure 5.12 and 5.13. There are total 1,416,960 unit cubic meter simulation cells. The hourly

mining rate is 960 cells and each cell is 1 cubic meter; then, we take six-hour blocks, which are 5760 cubic meters. The detail of this process is shown in Figure 5.14, in which each dashed block expresses hourly operation and the arrows indicate the mining sequence.

5.3.5 Simulation

The simulation is conducted in a two-step procedure. First, the content of *bitumen* is estimated by sequential Gaussian simulation. Then, by using the simulated values of *bitumen*, the content of *fines* is cosimulated with the option of co-located cokriging. Unconditional simulations are adopted. From the previous calculation of up-scaling model, the corresponding parameters have been used in the sequential Gaussian simulation.

To back-transform the simulated results, one needs to have the distribution of *bitumen* and *fines* for conducting the transformation. The histograms of *bitumen* and *fines* after *affine correction* have been used. The *direct* variogram model tabulated in Table 5.1 and shown in Figure 5.6 is used for the simulation.

The variograms between model and simulation are compared in Figure 5.18, in short scale the variograms from simulation are produced well, the deviations of vertical direction from the input model could be explained by the smaller simulation size so there exists fewer pairs in that direction.

A better correlation coefficient -0.75 between *bitumen* and *fines* has been gotten by simulation. Cross plots in Figure 5.15 show the relation before and after the simulation.

Figure 5.16 and Figure 5.17 show a 3-dimensional view of one realization of *bitumen* and a joint simulation of *fines*.

5.3.6 Time Variation

After simulation we assess the temporal variation of *bitumen* and *fines* content in successive volumes of hourly processing based on the mining scheme. In order to observe trends of time variation for different time intervals, six-hour, twelve-hour and twenty four-hour time intervals are also adopted. Figures 5.19 and 5.20 show the time variations in *bitumen* and *fines* based on different time intervals, respectively. In order to compare their corresponding change trends in the real same time we draw time variations of two variables together in the same plot shown in Figure 5.21. From this plot it is relatively easy to detect the related variation trend existing between them, namely, the *bitumen* content increases while the *fines* content decreases in general. Histograms of both variables are shown in Figures 5.22 and 5.23, from which one can notice that histograms of time variation for different time intervals are generally similar. Figure 5.24 shows the variances at different time intervals, respectively. From the variance we can find the ranges of alteration of each time variation, namely, for both of the variables with the the lengthening of time interval, the variance has a downward trend which means that the change trends become weak. It is reasonable that the variance decreases with the increase of volume. The relations of dependence for two variables at different time intervals can be depicted in the cross plot of Figure 5.25. There are stable correlation coefficients at different time interval, we think it is appropriate due to the average effects which smooth out some extreme values, so the coefficient factors may enhance a little with the time interval

increase. Like spatial continuity of the attributes the variograms of time variations for *bitumen* and *fines* have similar variabilities as shown in Figures 5.26 and 5.27. And the time variation variograms can tell us the continuity of variation for *bitumen* and *fines* content in time domain. Further, we can observe the histograms of average time variation in successive time intervals in Figures 5.28 and 5.29, namely, the change from last time interval to next time interval. Distribution of this kind of change is very useful for predicting the next time interval content of the variables. And the plant can use this value to control the related operation processes.

5.4 Conclusion

The objective of this case study is to obtain the time variations for content of *bitumen* and *fines* in the different volumes transported to processing plant. we assume the *bitumen* and *fines* content must have related time variation since they reappears strongly spatial variability. Based on this relations time variation was generated and analyzed in detail.

To estimate the time variation, we need to simulate the spatial variability of *bitumen* and *fines* using the geostatistical simulation. First variogram inference and analytical models have been established for *bitumen* and *fines* based on the data spatial continuity. Due to the different volume support from core samples, e.g. 0.05 by 0.05 by 0.61 meter to simulation cell, say 1 by 1 by 1 meter the up-scaling variograms and histograms have been done according to the volume-variance relations. From the given sample data we also can notice that *bitumen* and *fines* content have strong negative correlation, data for *bitumen* are collected preferentially and taken sparsely for *fines*, therefore *bitumen* content needs to be simulated first by using sequential

Gaussian simulation, then, cosimulation is adopted for *fines* content with the option of collocated cokriging. In order to meet the on-site mining requirement we need produce the final time variations by selecting proper mining scheme. The curves of time variations and related histograms have been presented in this study.

Once we obtain the result of time variations for *bitumen* and *fines*, variograms of time variations and the average changes in the next time interval for *bitumen* and *fines* are also generated. The distribution of average changes of time variations are useful for the plant to design the production plan and adjust corresponding operation parameters. This is also the purpose of this paper. However, we also know that the outcome just came from limited realizations, the production rate is an constant or linear variables, and a single bench is considered. Therefore, in the future we still need include multiple benches in the modeling, perform the simulation of mining rate and process dynamic optimization simulation, and consider the optimal use of multiple realizations in presence of uncertainty in prediction of oilsands bitumen and fines content.

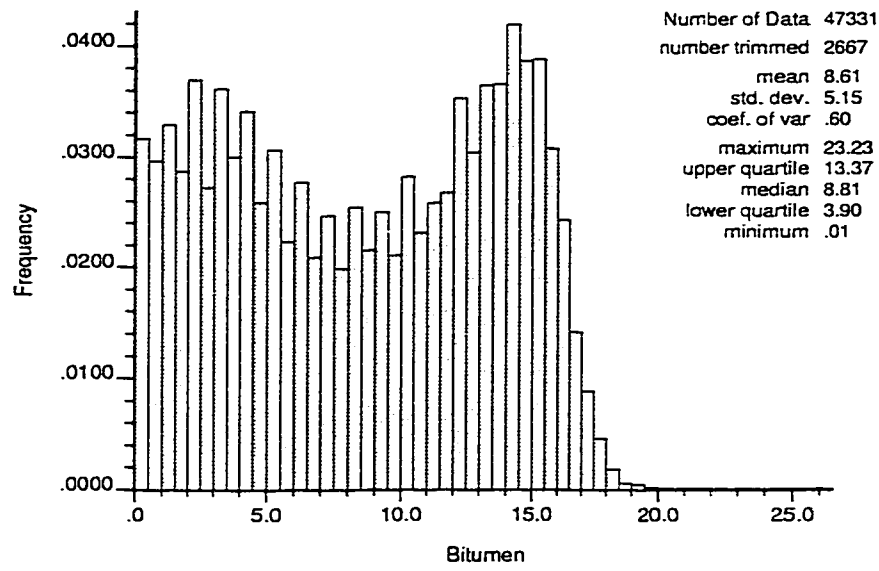


Figure 5.1: Histogram of *bitumen* content %

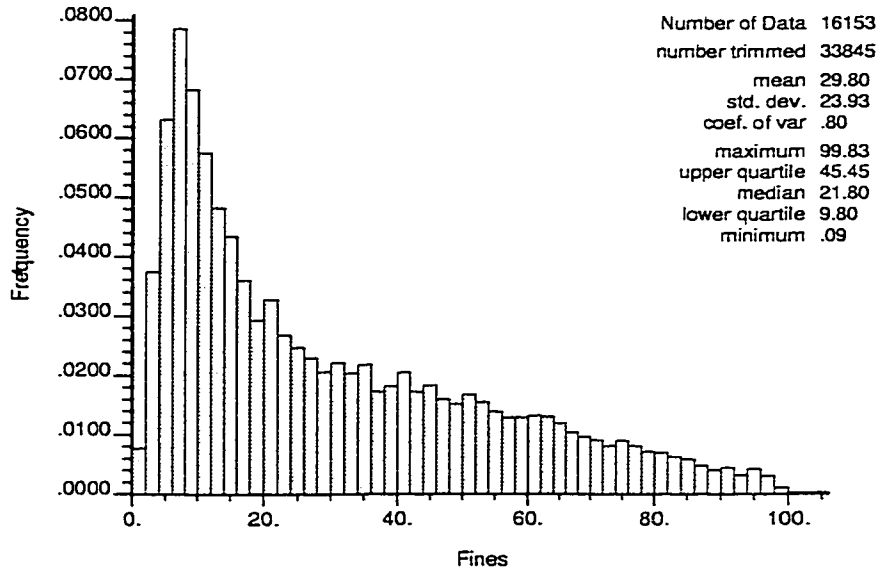


Figure 5.2: Histogram of *fine* content %

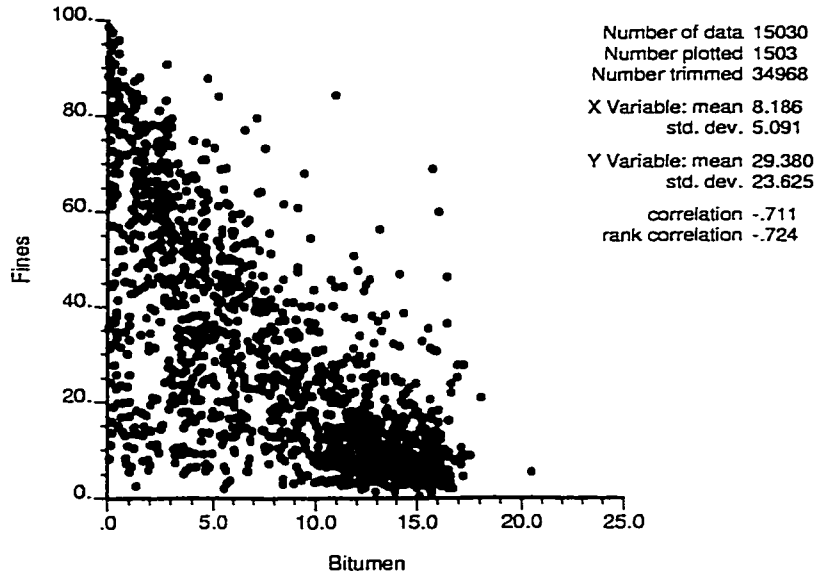


Figure 5.3: Cross plot of *fines* content % versus *bitumen* content %

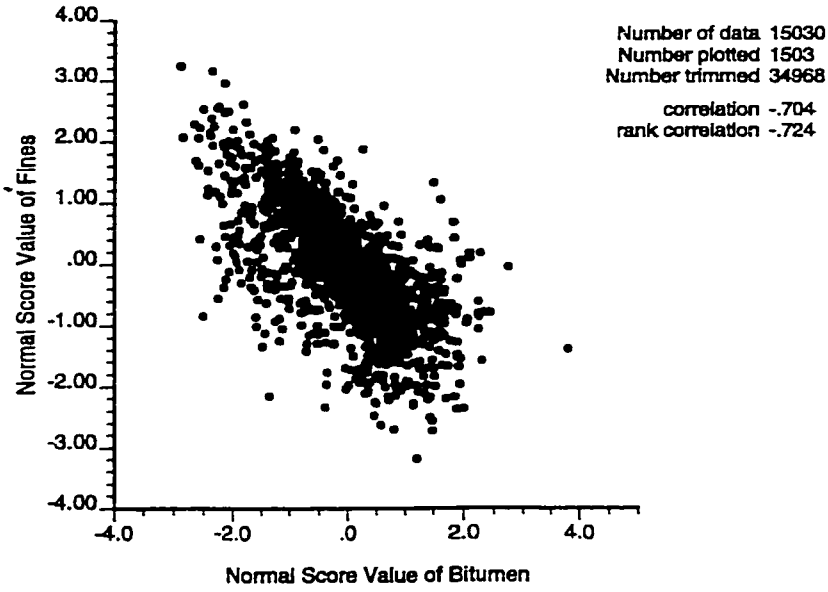


Figure 5.4: Cross plot of normal scores of *fines* content versus normal scores of *bitumen* content

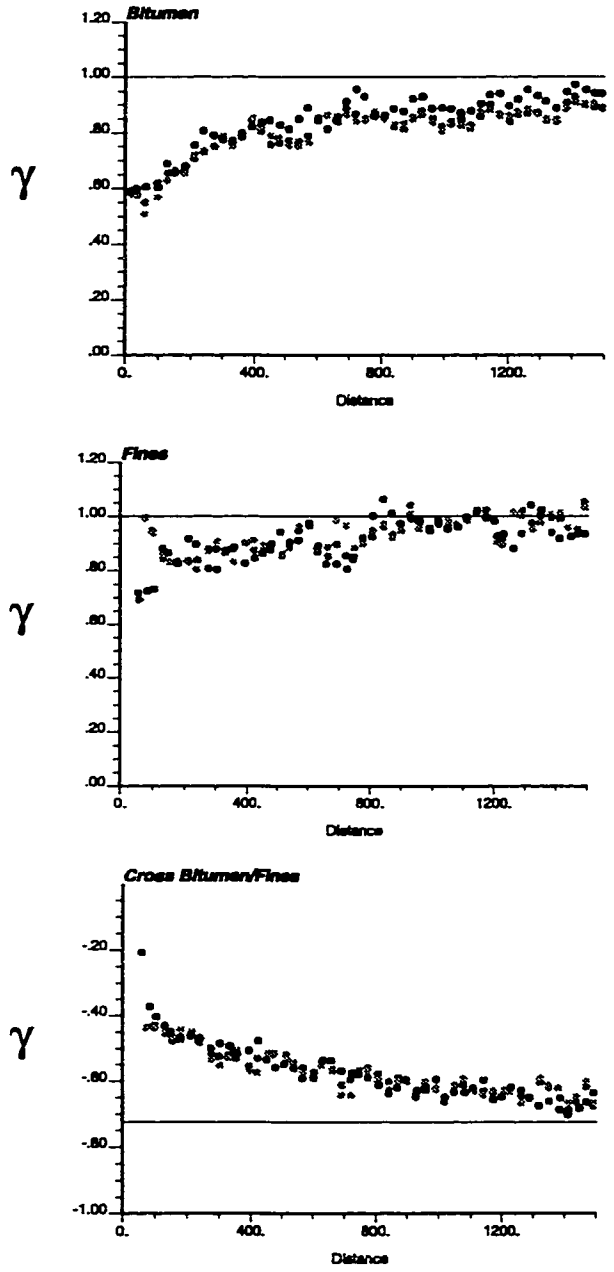


Figure 5.5: Directional ($N25W$ (*green dots*) and $N65E$ (*blue dots*)) and omnidirectional (*red dots*) horizontal variogram for *bitumen* (top), *fines* (middle) and cross variogram of *bitumen* and *fines* (bottom) *nlag =50 and dlag = 30*

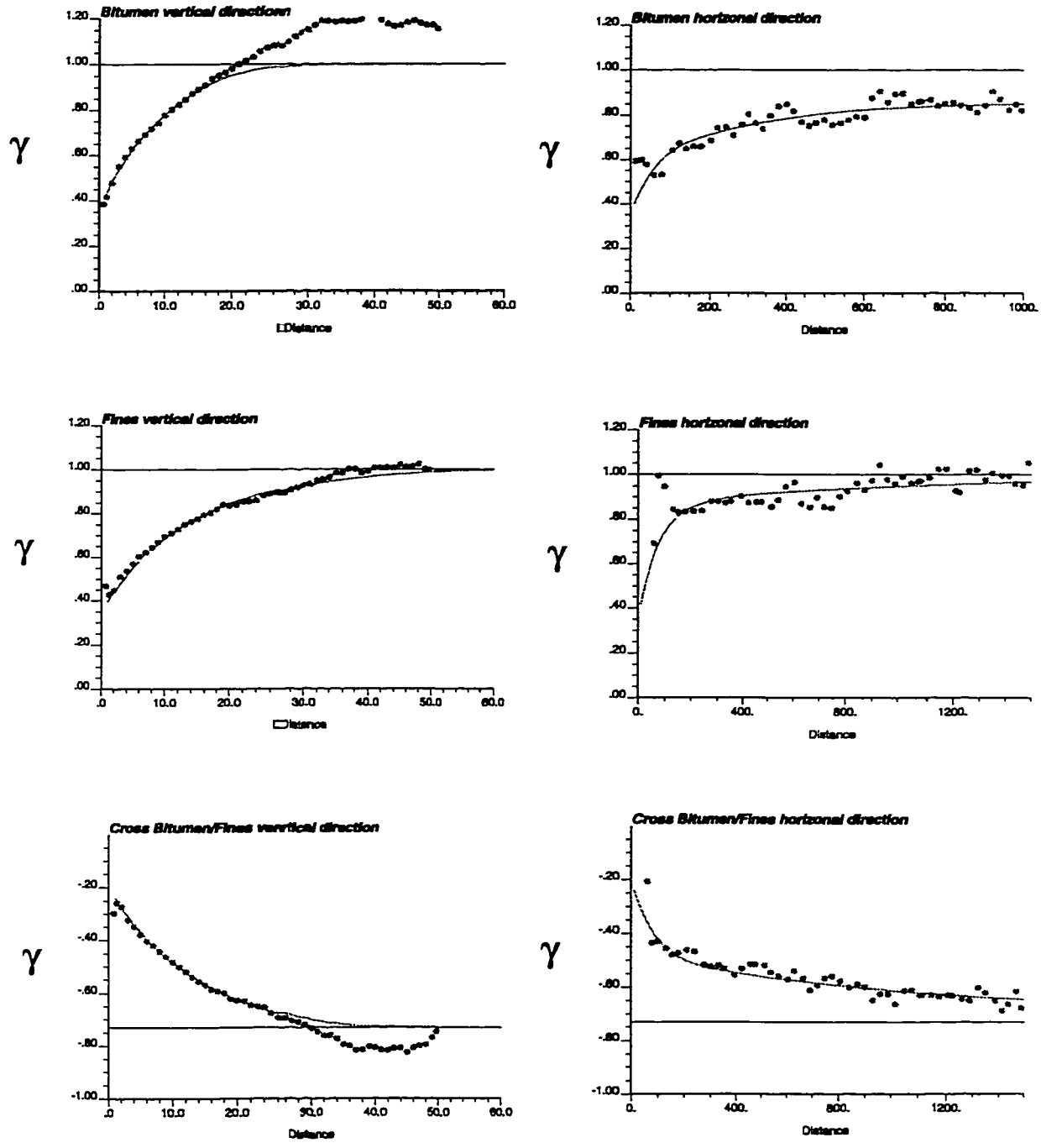


Figure 5.6: Experimental variogram and cross variogram (red dotted lines), and variogram models in core volume scale (red solid lines). Left: vertical, Right: horizontal, Top: variogram of *bitumen*, Middle: variogram of *fines*, Bottom: cross variogram of *bitumen* and *fines*

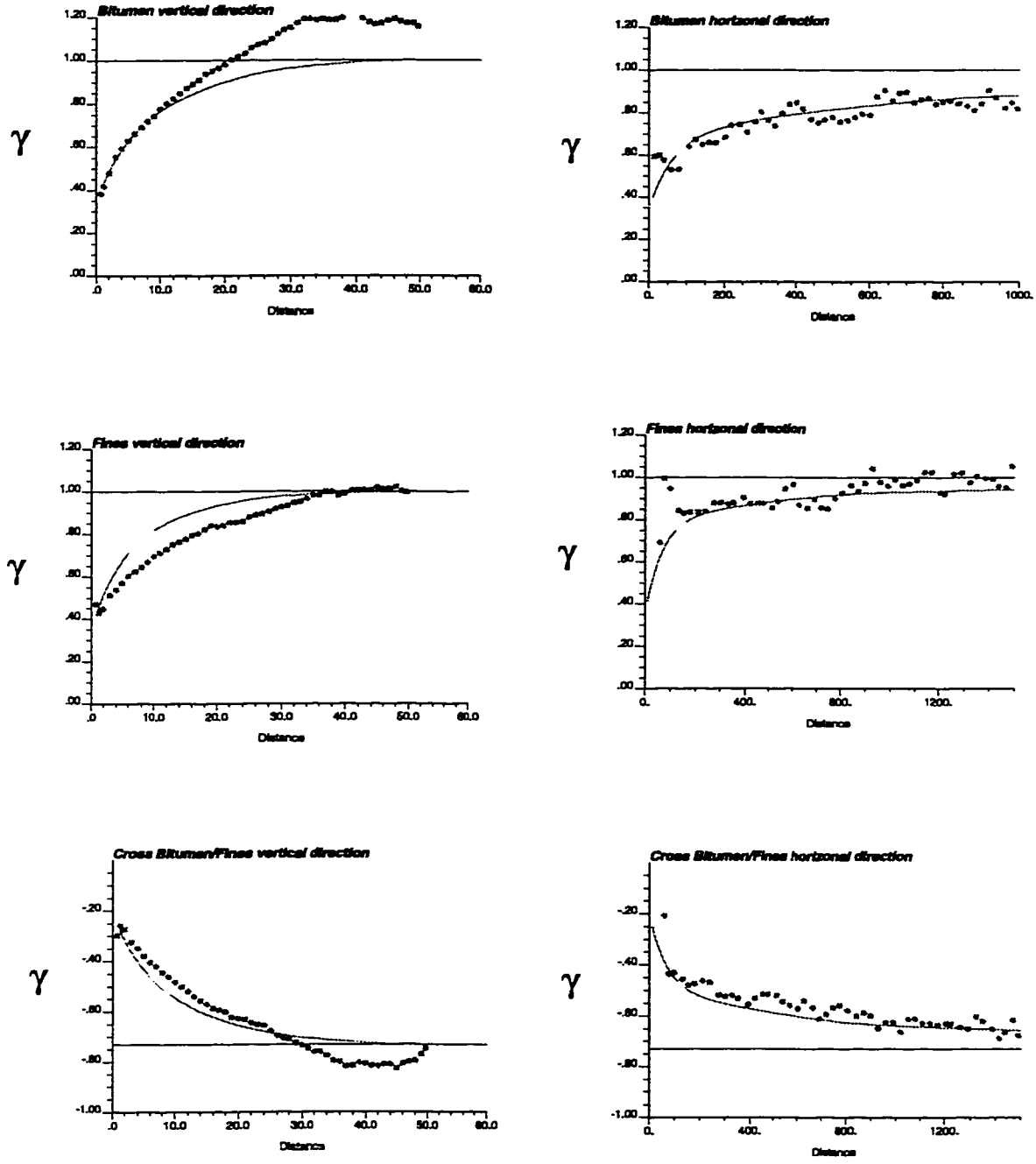


Figure 5.7: Experimental variogram and cross variogram (red dotted lines), and LMC in core volume scale (red solid lines). Left: vertical, Right: horizontal Top: variogram of *bitumen*, Middle: variogram of *fines*, Bottom: cross variogram of *bitumen* and *fines*.

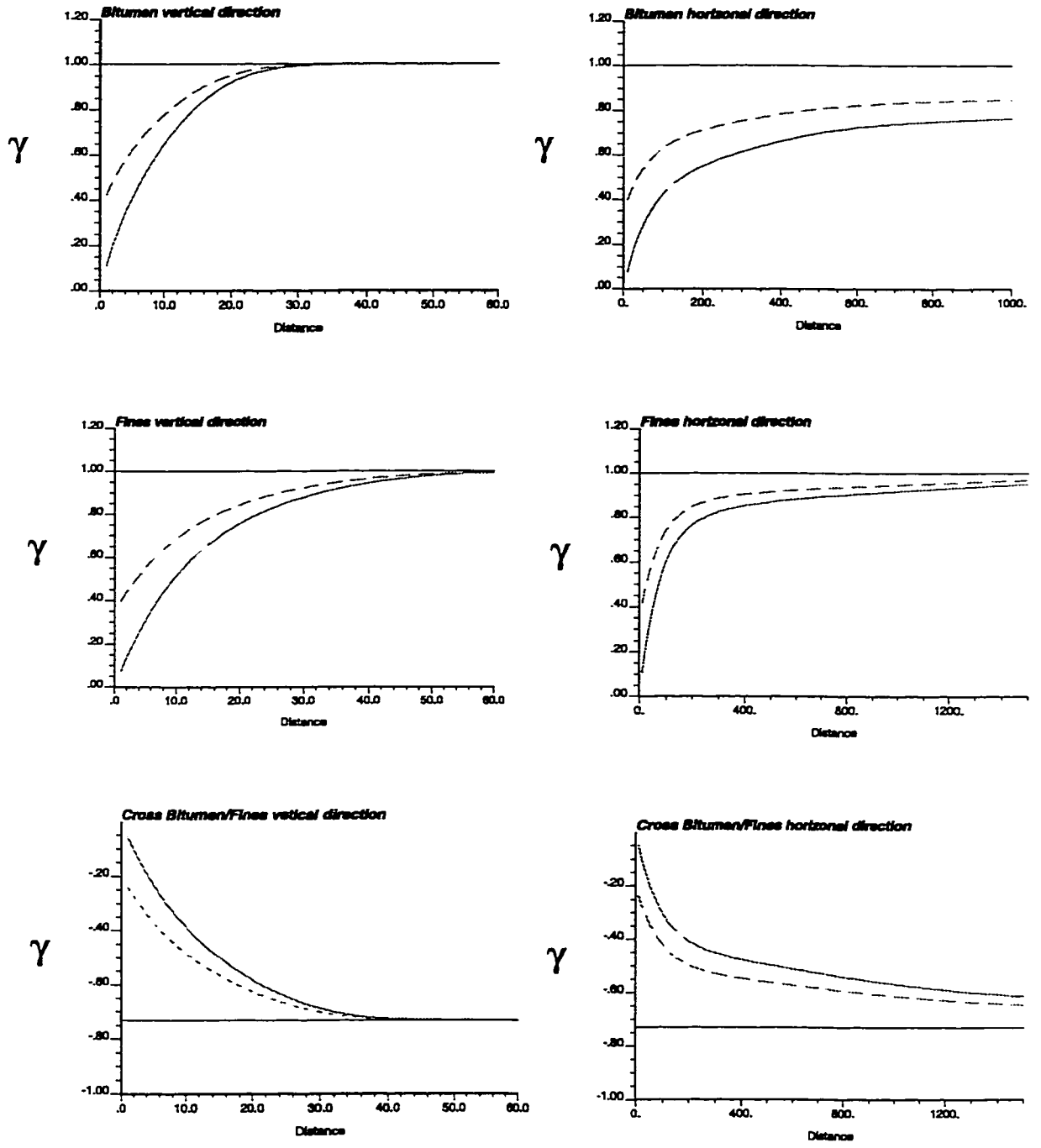


Figure 5.8: Variogram models in core volume scale(dash lines) and in simulation cell scale (solid lines).

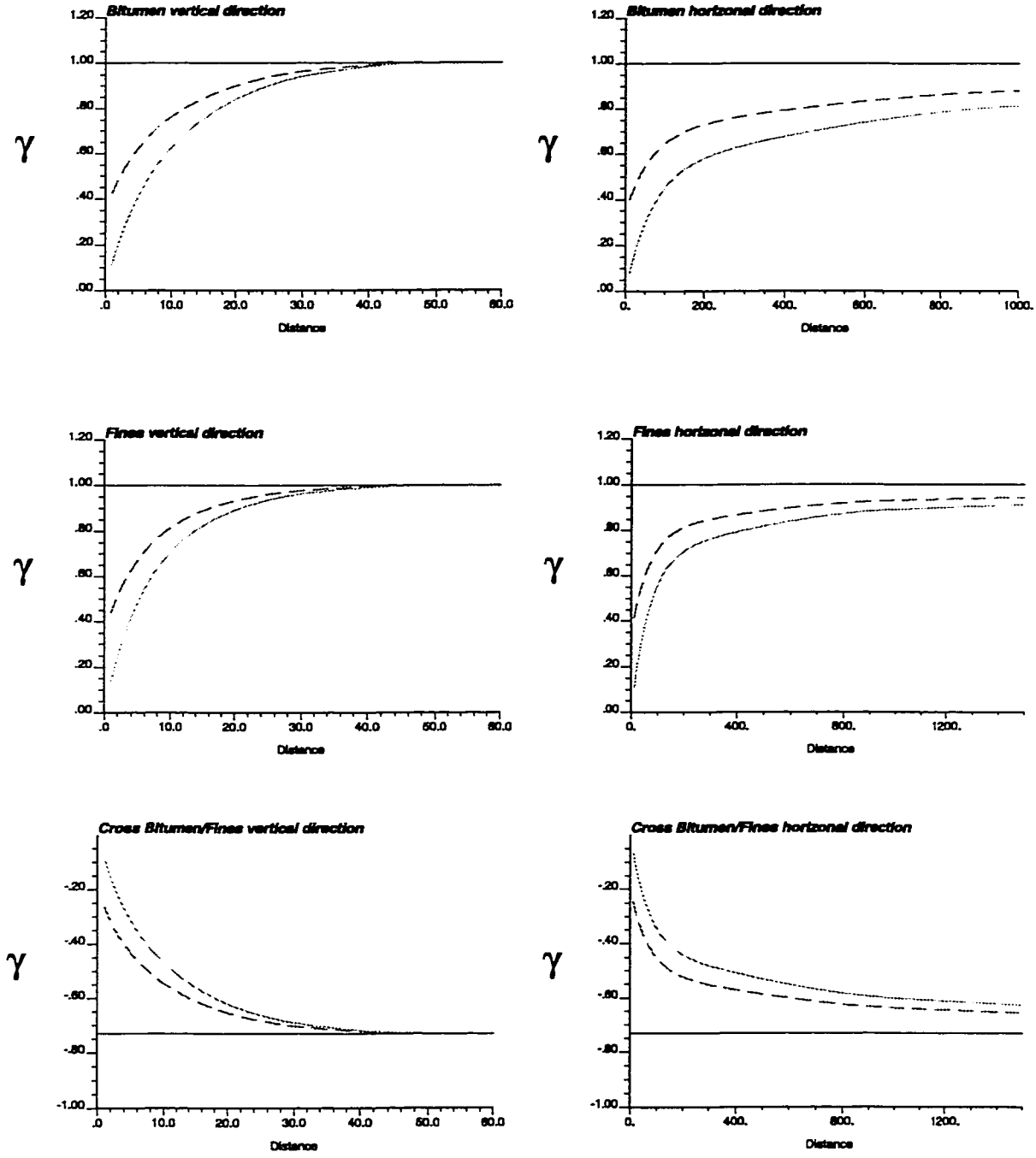


Figure 5.9: LMC in core volume scale (dash lines) and in simulation cell scale (solid lines).

Figure 5.10: Histogram of *bitumen* content % after *affine correction.*

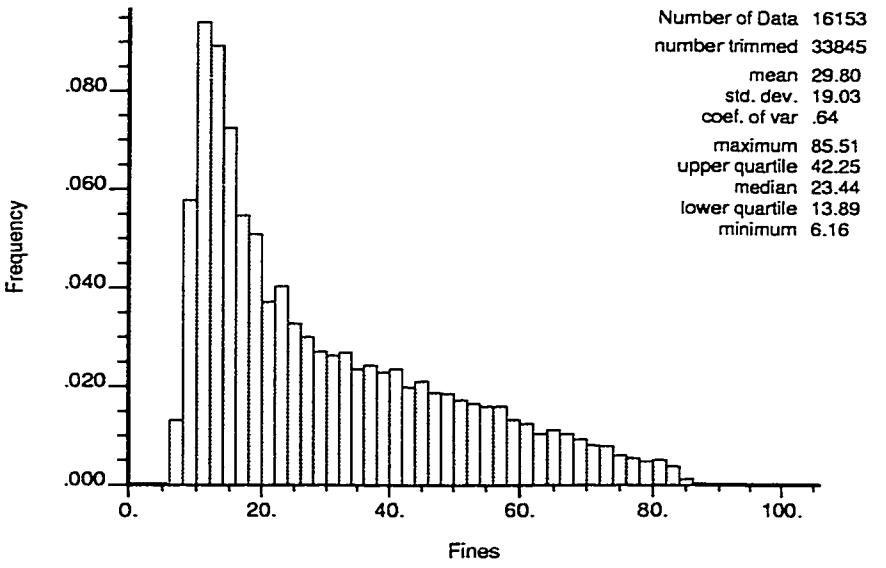


Figure 5.11: Histogram of *fines* content % after *affine correction*

Figure 5.12: A 3D block model, which could come from geostatistical simulation.

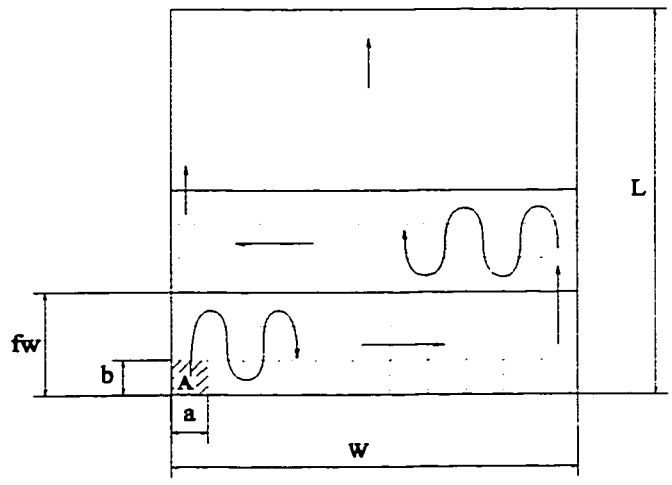


Figure 5.13: Plan view of mining scheme.

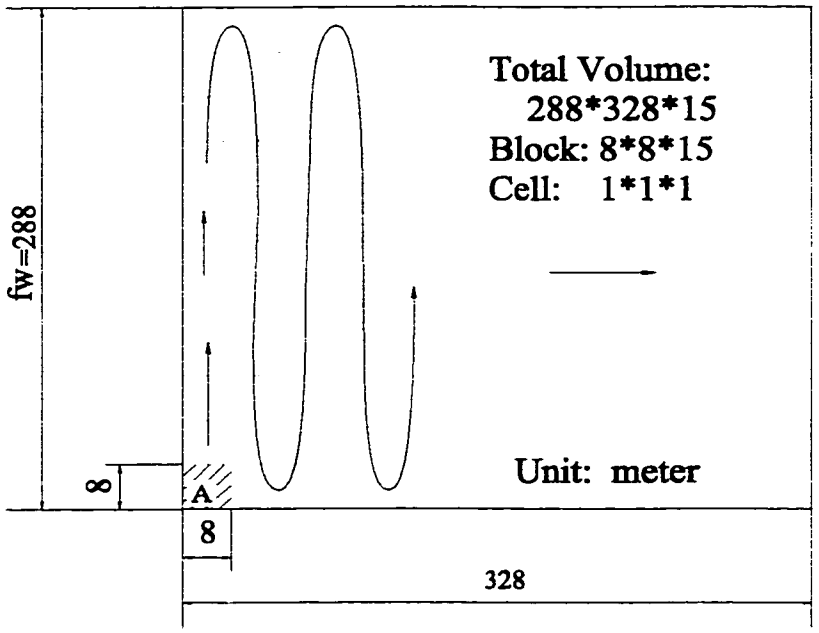


Figure 5.14: Mining scheme.

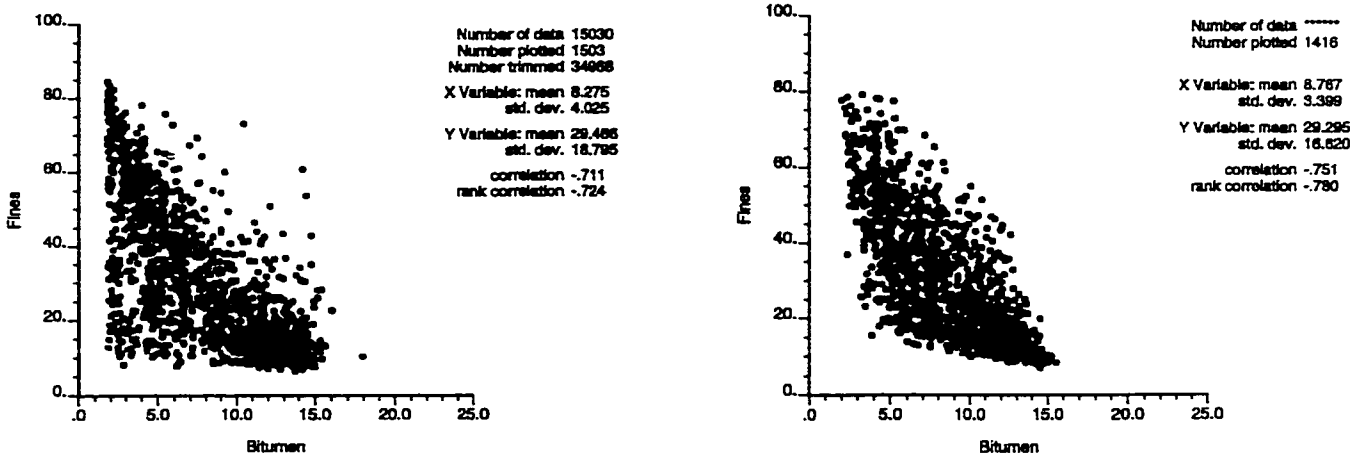


Figure 5.15: Cross plots of *bitumen* content versus *fines* content, Left is the affine correction result, right is the simulated one.

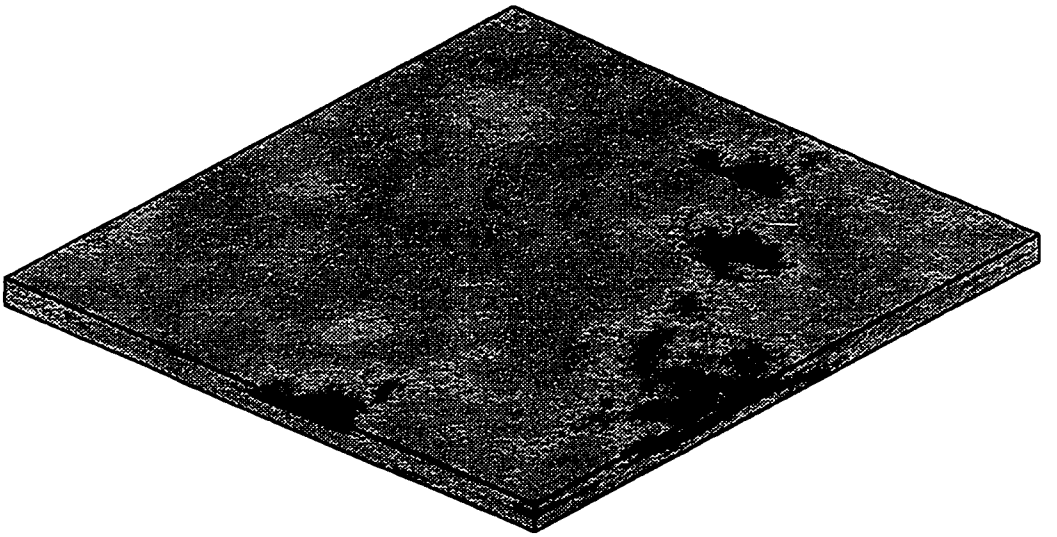


Figure 5.16: 3D view of simulated *bitumen* content

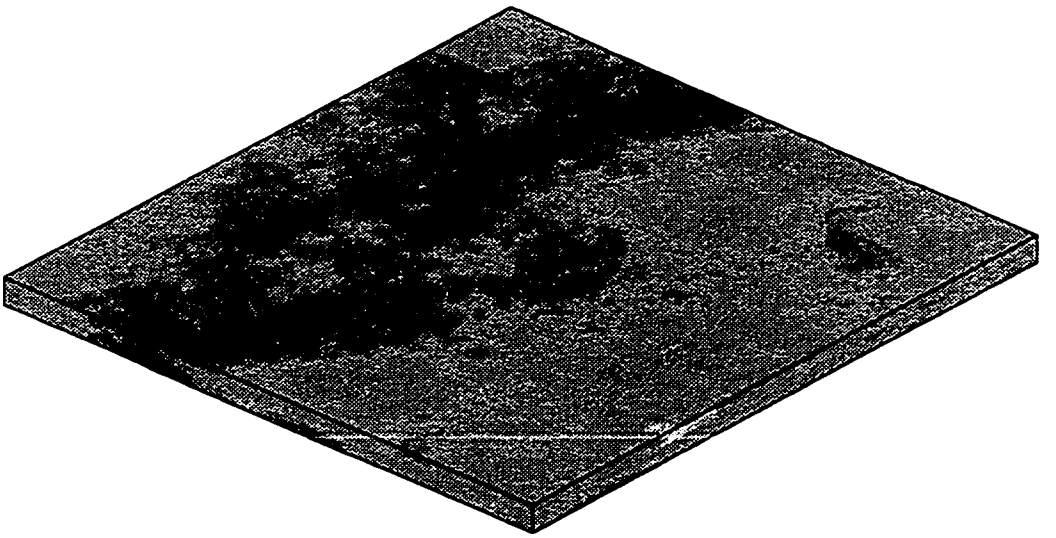


Figure 5.17: 3D view of cosimulated *fines* content

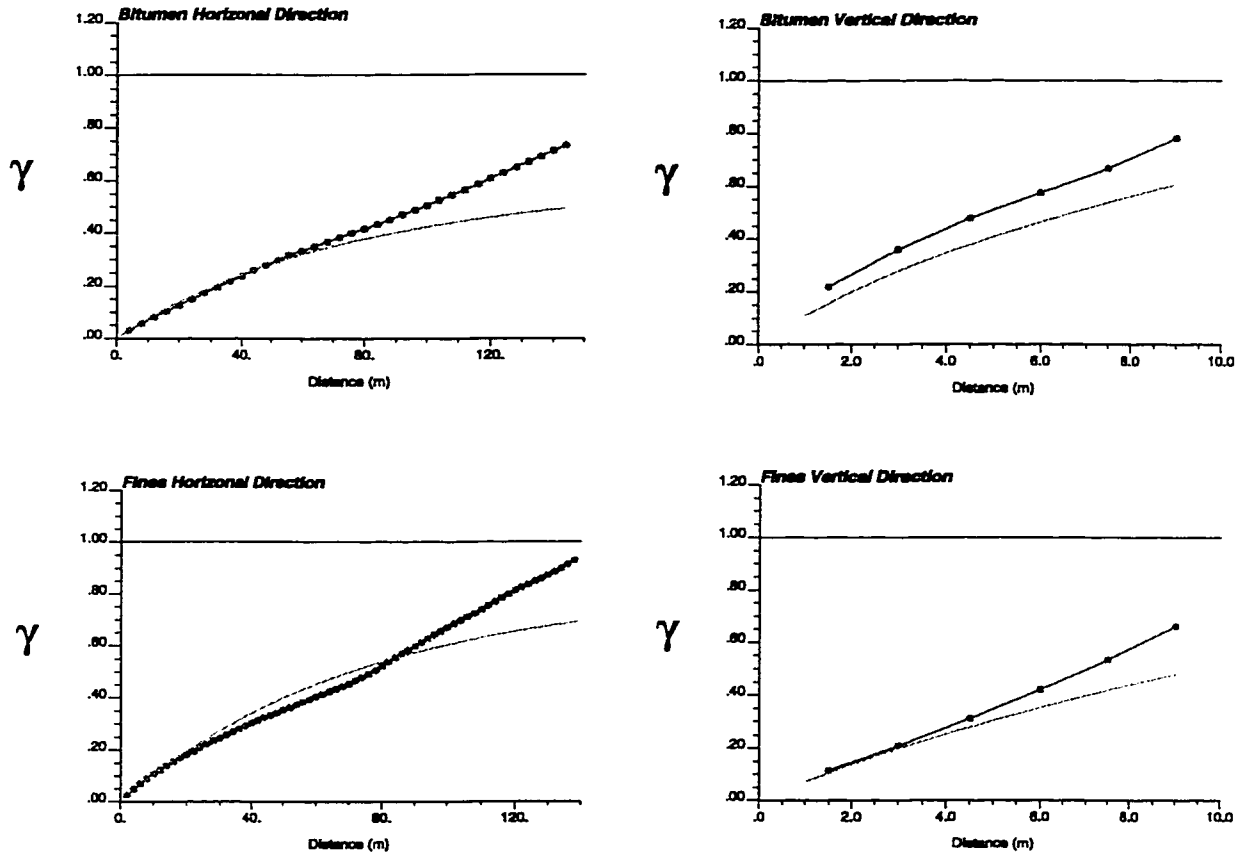


Figure 5.18: Variograms of *bitumen* content and *fines* content. The solid line is the variogram model, the dotted line is from simulation, upper left is the horizonal direction variogram for *bitumen*, upper right is vertical direction variogram for *bitumen*. Lower left is the horizonal direction variogram for *fines*, lower right is vertical direction variogram for *fines*.

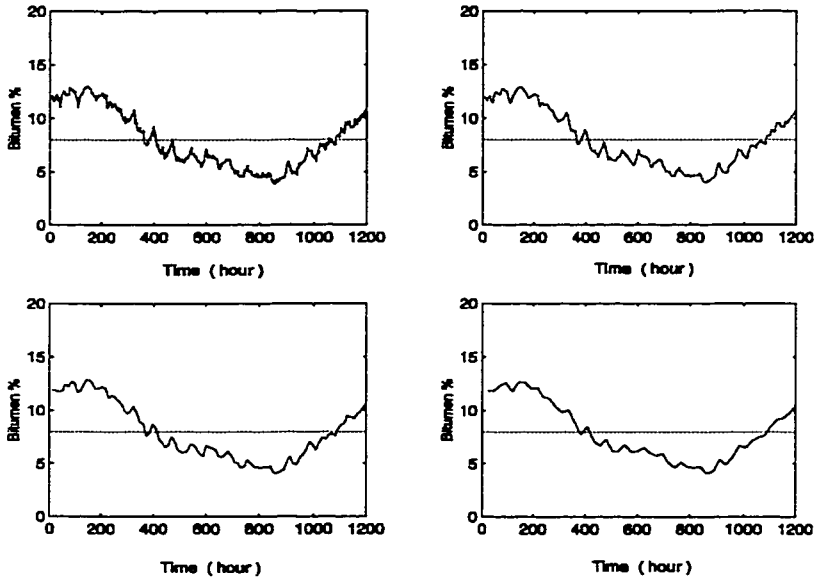


Figure 5.19: Curves of time variation for *bitumen* content, left of Line 1 is hourly interval for *bitumen*, right of Line 1 is 6-hour interval for *bitumen*, left of Line 2 is 12-hour interval for *bitumen*, right of Line 2 is 24-hour interval for *bitumen*

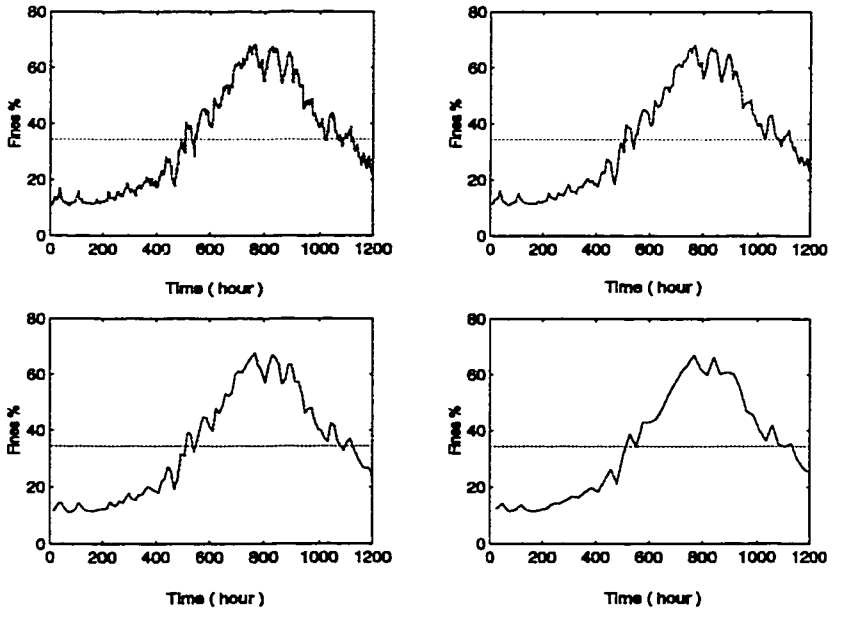


Figure 5.20: Curves of time variation for *fines* content, left of Line 1 is hourly interval for *fines*, right of Line 1 is 6-hour interval for *fines*, left of Line 2 is 12-hour interval for *fines*, right of Line 2 is 24-hour interval for *fines*.

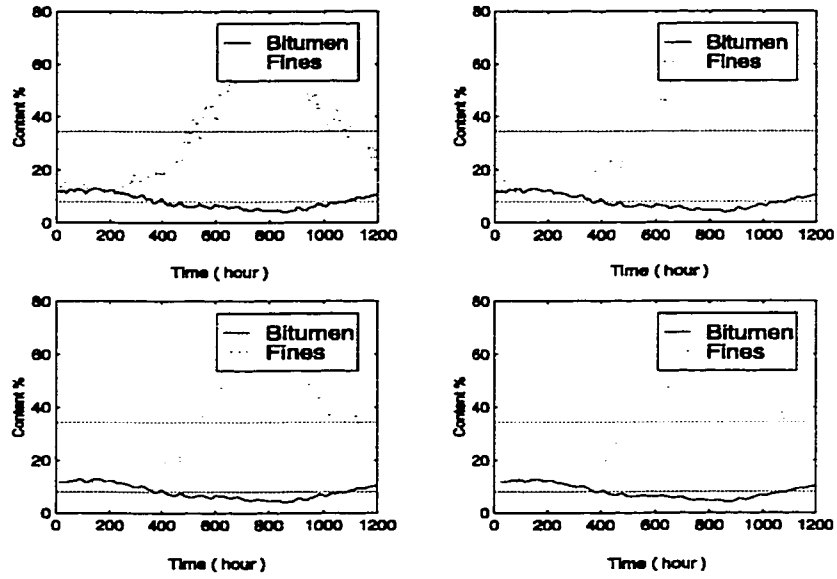


Figure 5.21: Comparison of time variations for *bitumen* and *fines* in different time intervals.

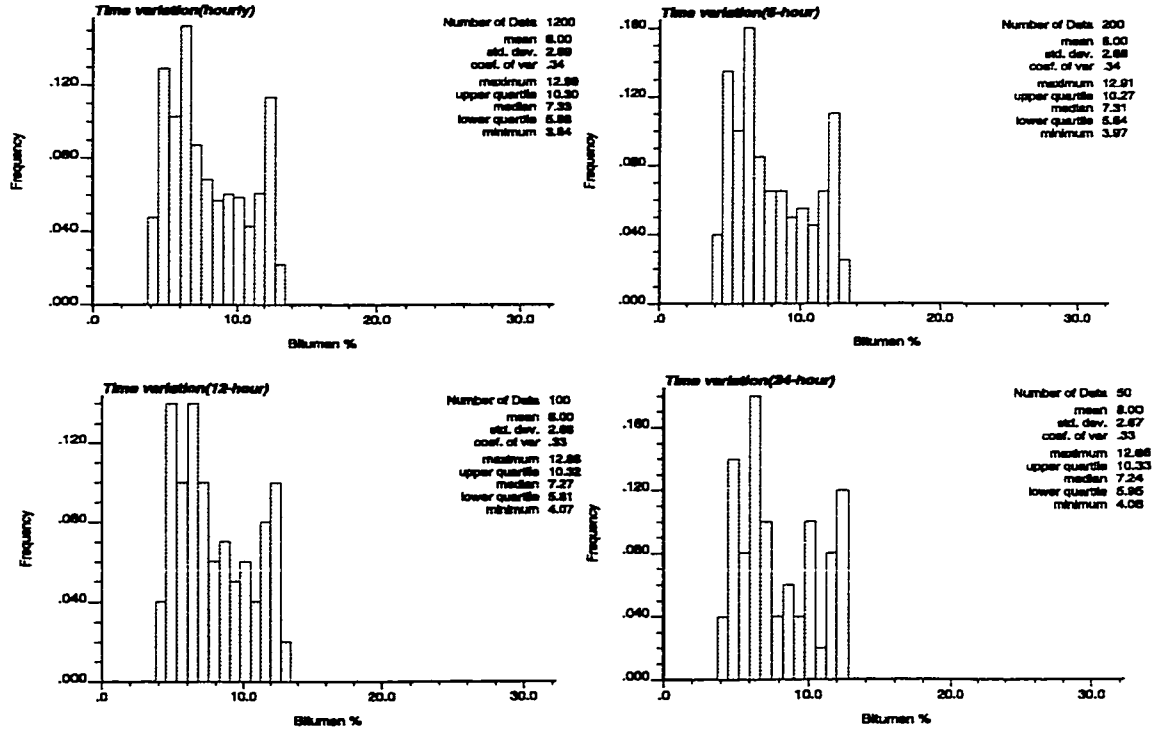


Figure 5.22: Histograms of simulated *bitumen* content at four time intervals

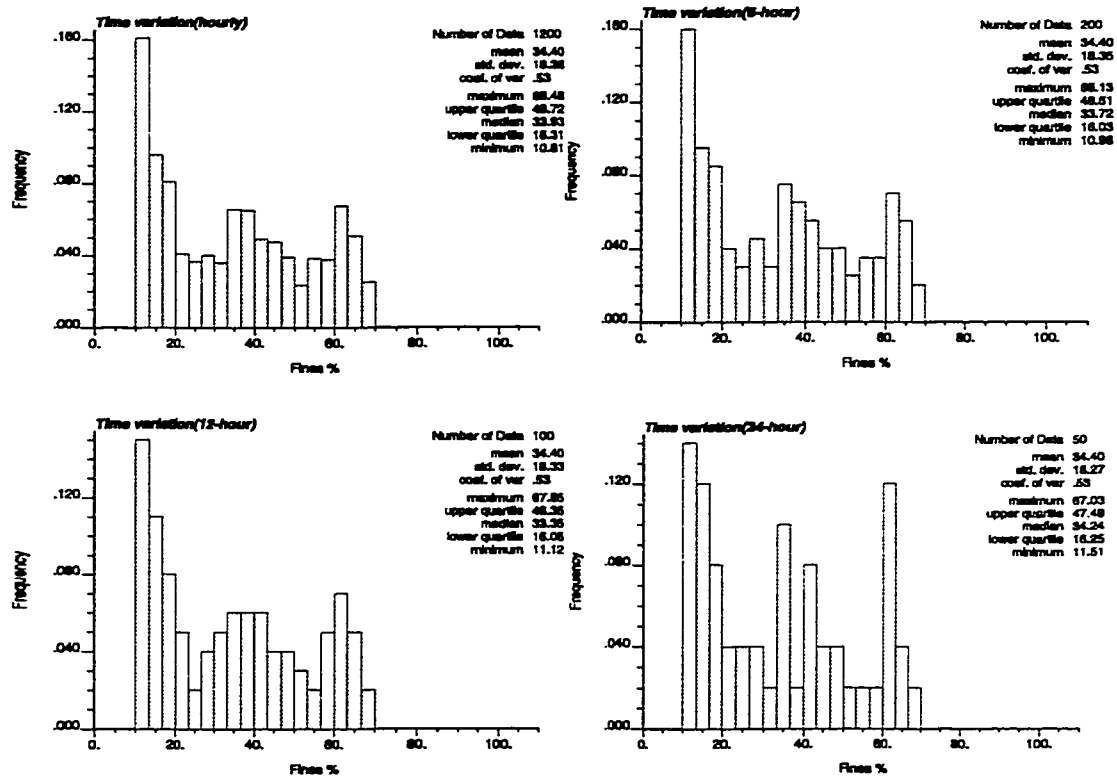


Figure 5.23: Histograms of simulated *fines* content at four time intervals

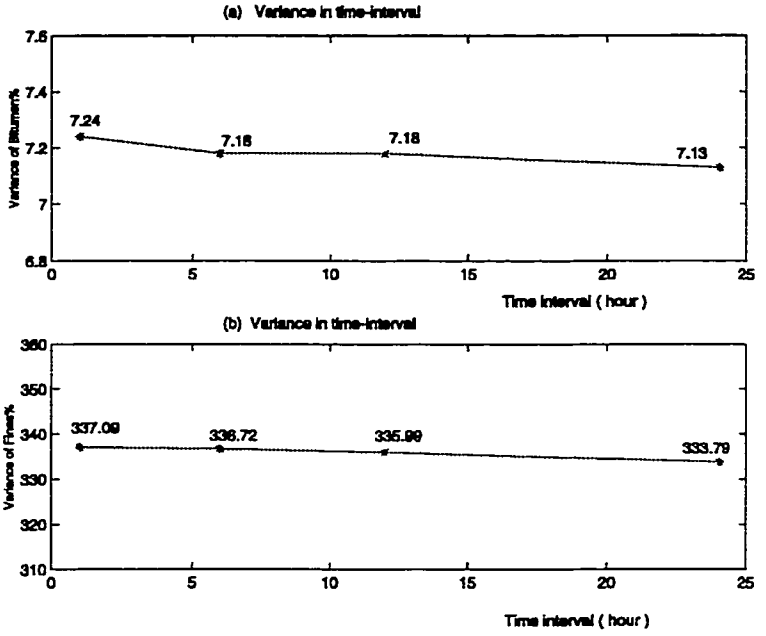


Figure 5.24: Variances of *bitumen* and *fines* at different time intervals. Upper shows bitumen time variation, lower is for fines.

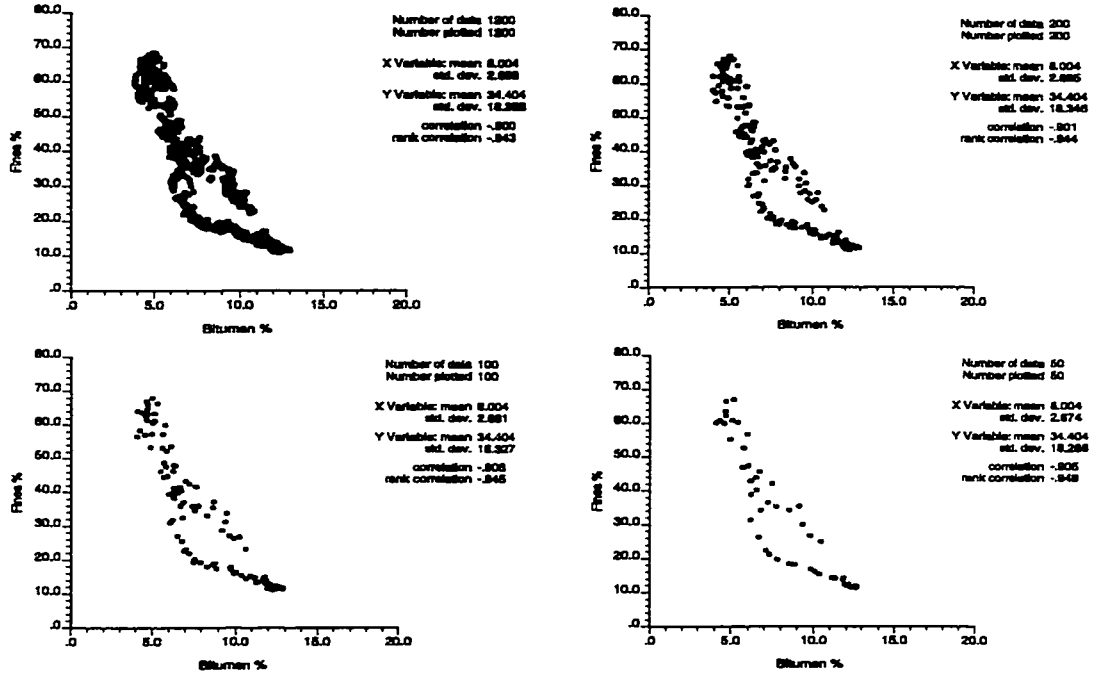


Figure 5.25: Cross plots for *bitumen* content versus *fines* content in different time intervals, upper left is hourly interval, upper right is 6-hour interval, lower left is 12-hour interval, lower right is 24-hour interval.

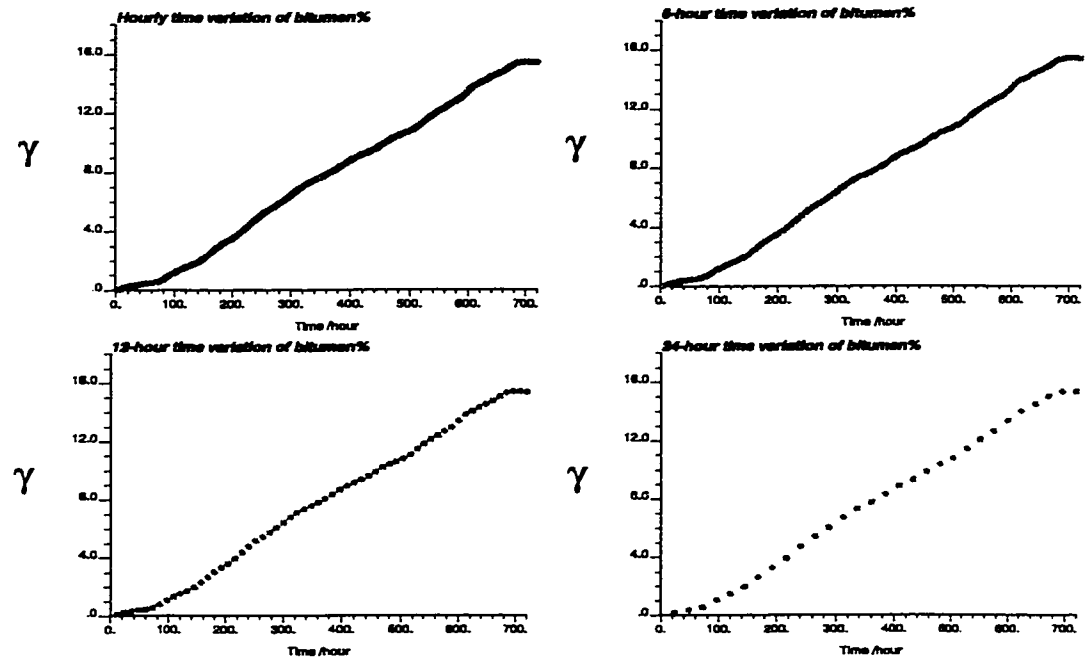


Figure 5.26: Variograms of time variation of *bitumen* at different time intervals. Line 1: Left is for hourly interval for *bitumen* content, right is 6-hour interval for *bitumen* content. Line 2: Left is for 12-hour interval for *bitumen* content.

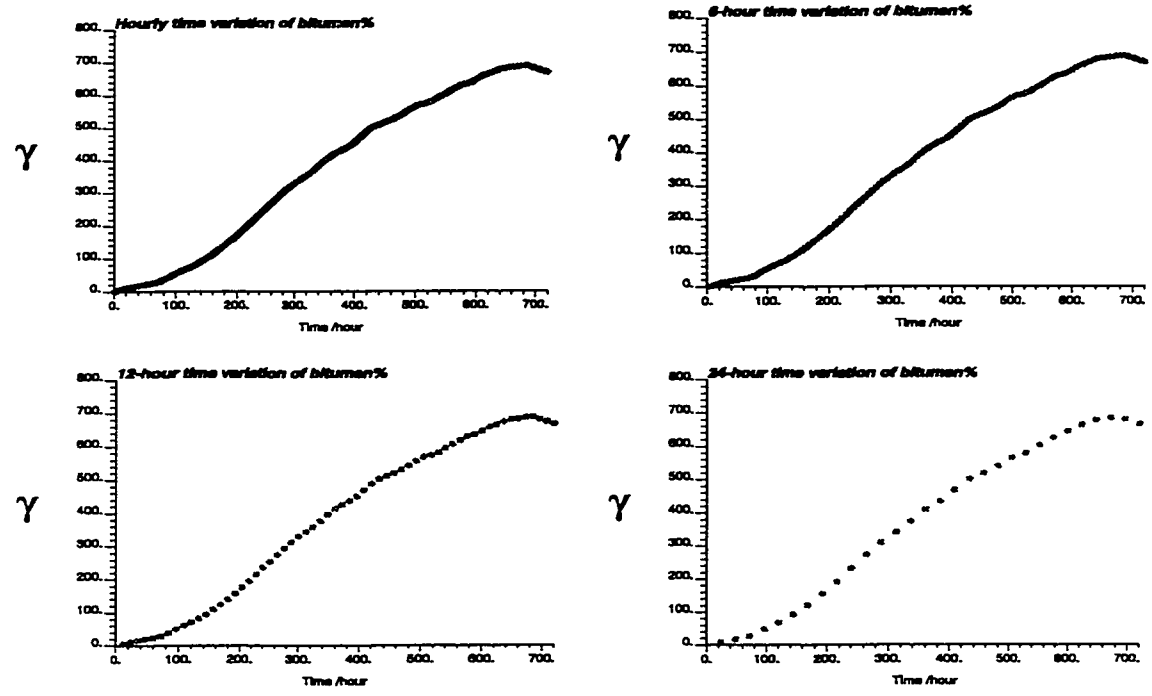


Figure 5.27: Variograms of time variation of *fines* at different time intervals. Line 1: Left is for hourly interval for *fines* content, right is 6-hour interval for *fines* content. Line 2: Left is for 12-hour interval for *fines* content, right is 24-hour interval for *fines* content.

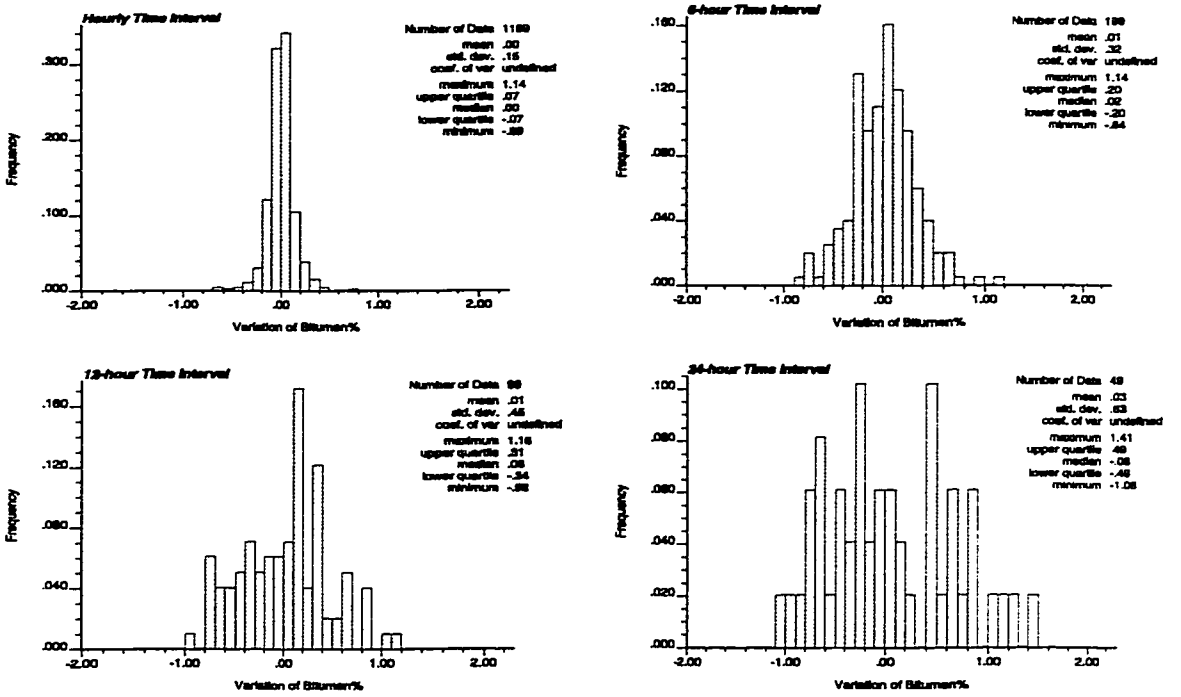


Figure 5.28: Time variations for *bitumen* content in successive two time intervals. Line 1: left is hourly time interval for *bitumen*, right is 6-hour time interval for *bitumen*. Line 2: left is 12-hour time interval for for *bitumen*, right is 24-hour time interval for *bitumen*.

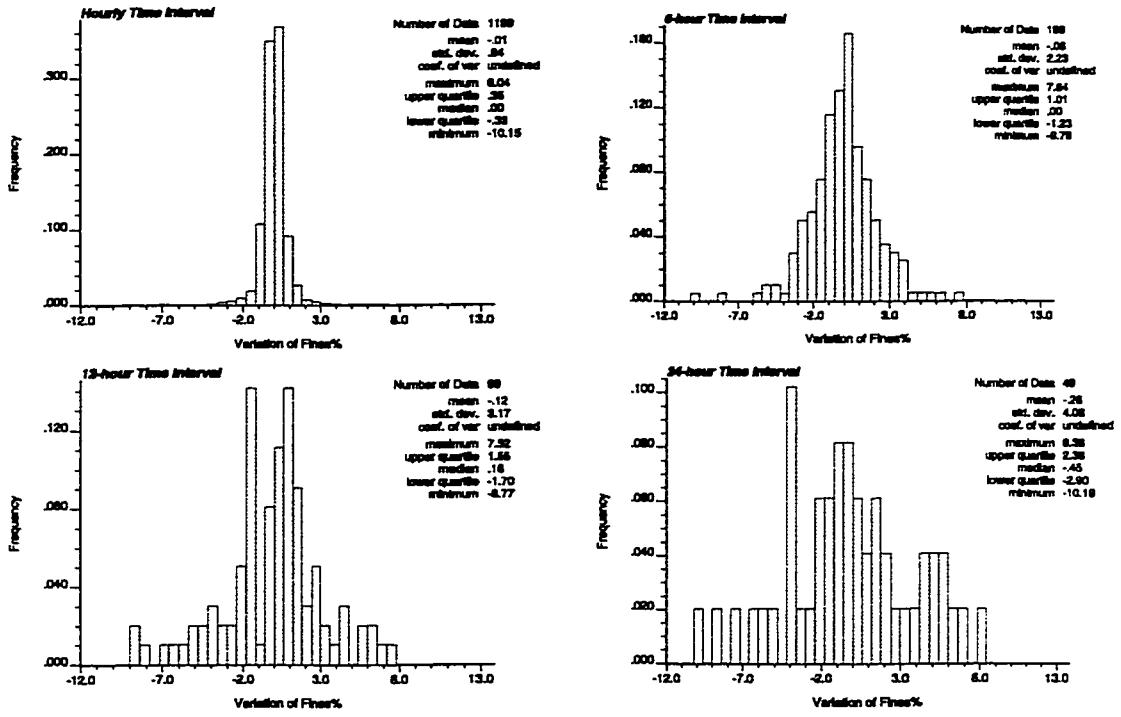


Figure 5.29: Time variations for *fines* content in successive two time intervals. Line 1: left is hourly time interval for *fines*, right is 6-hour time interval for *fines*. Line 2: left is 12-hour time interval for *fines*, right is 24-hour time interval for *fines*.

Bibliography

[1] A. G. Journel and C. J. Huijbregts, *Mining Geostatistics*, Academic Press, New York, NY, 1978.

[2] E. H. Isaaks and R. M. Srivastava, *An Introduction to Applied Geostatistics*, Oxford University Press, New York, NY, 1978.

[3] C. V. Deutsch and A. G. Journel, *GSLIB: Geostatistical Software Library and User's Guide*, Oxford University Press, NY, 1997

[4] C. V. Deutsch, *MIN 612: Geostatistical Methods for modelling Earth Sciences Data*, Class Notes, 1999.

[5] C. V. Deutsch and Karl P. Norrena, *MIN 310: Ore Reserve Estimation*, Class Notes, 1999.

[6] Goovaerts, P., *Geostatistics for Natural Resource Evaluation*, Oxford University Press, New York, NY, 1997.

[7] Michel David, *Handbook of Applied Geostatistical Ore Reserve Estimation*, Elsevier, Amsterdam, 1988.

[8] A. G. Journel, *Fundamental of Geostatistics in Five Lessons*, Volume 8 Short Course in Geology, Mathematical Geology, 12(4):285-303, 1980.

[9] Dimiitrakopoulos Rousssos-Georgios, *Conditional Simulation and Kriging as an aid to oil sands development: an application in part of Athabasca deposit*, University of Alberta, Edmonton, Alta, 1985.

[10] Marcel Vallee, Michel David, *Guide To The Evaluation Of Gold Deposits*, Special Volume 45, Geological Society of CIM, 1992.

[11] Gordon Ray Luster, *Application Of Conditional Simulation Of Coregionalization*, PhD thesis, Stanford University, Stanford , CA, September, 1985.

[12] Sidney G.Seiffert, *Mine Feasibility Study*, Project: Surface Coal Mines.

[13] R.G. Goebel and Mike Lipsett, *Constraint-based Scheduling for Surface Mining Operations*, CIM, 1999.

[14] C. V. Deutsch and A. P. Beardow, *Optimal Drillhole Spacing and Uncertainty in Prediction of Oilsands Bitumen and Fines Content*, CIM, 1999.

[15] Schofield, C. G., *Homogenisation/Blending Systems Design and Control for Minerals Processing*, Trans Tech, Clausthal, germany, 315 pp 1980.

[16] Juan Camus, *The Opportunity Cost in Mining Planning* , CIM, 1999.

[17] Parnably, J., P. G. Battye, and G. S. Waite(1973), *Optimal Design of Homogenising Systems Incorporating Layered Stockpiles and Fluidised Silos for the Control of Raw Materials Quality*, Transactions, Institution of Chemical Engineers, v. 51, pp. 323-330.

[18] Gy, P. M. , *A new Theory of Bed-Blending Derived From the Theory of Sampling – Development and Full-Scale Experimental Check* , International Journal of Mineral Processing, 1981, v. 8, pp. 201-238.

[19] Gy, P. M. , *Sampling of Particulate Materials: Theory and Practice*, Theory and Practice, 2nd ed., Elsevier, Amsterdam 1982, 431 pp.

[20] William W.Almdal, *From Potential to Prosperity, Progress in Canada's Oil Sands* , CIM, 1999.

[21] Randy John Mikula, *Oil Sands Conditioning, Bitumen Release Mechanisms and New Process Development* , CIM, 1999.

[22] J. Rand Ridges and Baker Process , *Intelligent Enterprise-wide Adaptive Optimizing Control* , CIM, 1999.

[23] Sean Dessureault and Malcolm Scoble, *Management Tools to Evaluate the Justification for Advanced Technologies in Mining* , CIM, 1999.

```
//*****************************************************
//* Time variation recently modified date : 1/16/00  *
//*****************************************************
#include "math.h"
#include "stdio.h"
#include "stdlib.h"
#include "malloc.h"
#include "string.h"
// External functions
void    readpara();
void    readdata();
void    writepara();
void    disres();
void    timevari();
void    fintime();
// Variables read parameter file
int     nx,ny,nz,var_colum,
        nx2,ny2,nz2,var_colum2;
int     unitx, unity, unitz,unitx2,
        unity2, unitz2;//cell size
char    *inputdata[60], *inputdata2[60],
        *outdata[60],*var_name[80];
float   tmin,tmax,tmin2,tmax2;
int     ire_ben,nben,mbenht[20];
int     ire_ben2,nben2,mbenht2[20];
int     nux,nuy,nuz;//(fa,fw);benht int
```

```
        nux2,nuy2,nuz2;//(fa,fw);benht
float   mining_rate,mining_rate2;
float   total_mining, cutoff;
int     mwface;
// Variables read data file
float   *var,*iorder,*time, *time2;

int main() {
    char   mfname[60]={"stmod.par"},ft; // name of parameter file
    printf("The parameter file is 'STmod.par', Y/N? [Y]:");
    ft=getchar();
    if((ft=='n')||(ft=='N'))
    {   printf("\nPlease input new parameter file:");
        scanf("%s",&mfname);
        printf("\nThe parameter file is %s\n",mfname);
    }
    readpara(mfname);
    disres();
    timevari();
    fintime();
    return 1;
}

//Read data file
void readdata(fname,colum_v) int colum_v; char *fname[40]; {
    FILE *frp;
```

```
char ch;
int i,l,colum,txyz,size;
float cx[10];
txyz=nx*ny*nz;
size=4;
var=calloc(txyz,size);
if (var==0){printf("Allocate memory error!\n"); exit;}
if((frp=fopen(fname,"rb"))==NULL)
    {printf("\nCan not open the data file or not exist!");
     exit(0);}
else{
             //fw=fopen("read.txt", "wb");
             ch=fgetc(frp);
             while(ch!=10){ ch=fgetc(frp);}
             fscanf(frp,"%d\n",&colum);

             for(i=0;i<colum;i++)
             {
                // if(i==(colum_v-1))
                fscanf(frp,"%80s\n",&var_name);
                ch=fgetc(frp);
                while(ch!=10){ ch=fgetc(frp);}

             }
            colum_v--;
```

```
                for(i=0; i<txyz;i++)
                        {
                            for(l=0; l<colum; l++)
                            {  size=
                                fscanf(frp,"%f",&cx[l]);
                               if(size<1)
                               {    printf("Error!
                                  No enough data in
                               the data file!\n");
                                    exit(0);
                               }
                            }
                            var[i]=cx[colum_v];
                            //fprintf(fw,"%f \n",var[i]);
                        }
                 //printf("size= %d",size);

                 fclose(frp);
                //fclose(fw);
        }
}

void readpara(fname1) char *fname1[60]; {
    FILE *frp;
    char ch;
    int i,tem;
```

```
    if(!(frp=fopen(fname1,"rb")))
        {printf("\nCan not open the parameter file,
                    Or %s  does not exist!!
           \nA new formatted parameter file has been
             produced!!\n",fname1);
         writepara();
         exit(0);
        }
    else{
            for(i=0; i<4; i++)
                {
                        ch=fgetc(frp);
                        while(ch!=10){ ch=fgetc(frp);}
                }
                 //printf("Read line 4 \n");

// Mutiple work faces? 2(two w faces) or 1 (one w face)
                    fscanf(frp,"%d",&mwface);
                    ch=fgetc(frp);
                    if(ch=='.') {printf("'-number of work
                         faces ' error1
                       in parameter file\n"); exit(0);}
                    if(mwface<0){printf("'-work face '
                           error in parameter
                        file\n"); exit(0);}
                    while(ch!=10){ ch=fgetc(frp);}
```

```
// cutoff
                    fscanf(frp,"%f",&cutoff);
                    ch=fgetc(frp);
                    if (cutoff<=0) {printf("' -Cutoff
                        grade' error in
                        parameter file\n"); exit(0);}
                    while(ch!=10){ ch=fgetc(frp);}

// total mining time units

                    fscanf(frp,"%f",&total_mining);
                    // printf("Total mining time = %f,
                        \n",total_mining);
                    ch=fgetc(frp);
                    if (total_mining<=0)
                      {printf("' -total number of time
                        units ' error in parameter file\n");
                      exit(0);}
                    while(ch!=10){ ch=fgetc(frp);}

// file outputing data
                    fscanf(frp,"%s",&outdata);
                    // printf("output data = %s, \n",outdata);
                    ch=fgetc(frp);
                    while(ch!=10){ ch=fgetc(frp);}
                    //ch=fgetc(frp);
```

```
// file for inputing data of site one, work face one
                    fscanf(frp,"%s",&inputdata);
                    // printf("Read file name = %s
                       \n",inputdata);
                    ch=fgetc(frp);
                    while(ch!=10){ ch=fgetc(frp);}
// column for variable
                    fscanf(frp,"%d",&var_colum);
                    // printf("var colum = %d  \n",var_colum);
                    ch=fgetc(frp);
                    if(ch=='.') {printf("'-column for
                         variable I' error\n");
                         exit(0);}
                    if(var_colum<=0){printf("'-column
                          for variable I'
                       error in parameter file\n");
                       exit(0);}
                    while(ch!=10){ ch=fgetc(frp);}
// nx, ny,nz
                    fscanf(frp,"%d  %d  %d ",&nx,&ny,&nz);
                    //printf("Nx = %d, ny= %d, nz= %d \n",
                         nx,ny,nz);
                    ch=fgetc(frp);
                    if(ch=='.') {printf("'-nx,ny,nz'
                        error in parameter
                       file\n"); exit(0);}
```

```
                    if((nx<=0) ||(ny<=0)||(nz<=0))
                      {printf("'-nx,ny,nz'
                       error in parameter file\n");
                       exit(0);}
                    while(ch!=10){ ch=fgetc(frp);}
// unit size of x, y, z
                    fscanf(frp,"%d  %d  %d",&unitx,&unity,&unitz);
                    //printf("Cell size x= %d, y= %d , z= %d\n",
                      unitx, unity,unitz);
                    ch=fgetc(frp);
                    if(ch=='.') {printf("'-cell size' error in
                        parameter file\n"); exit(0);}
                    if((unitx<=0) ||(unity<=0)||(unitz<=0))
                       {printf("
                       '-cell size' error in parameter file\n");
                        exit(0);}
                    while(ch!=10){ ch=fgetc(frp);}

// limit for Min and Max
                    fscanf(frp,"%f  %f ",&tmin,&tmax);
                    //printf("tmin = %5.2f, tmax= %5.2f \n",
                         tmin,tmax);
                    ch=fgetc(frp);
                    if((tmin>tmax)||(tmax<=0))
                     {printf("'-trimming limits'
                      error in parameter file\n");
```

```
                exit(0);}
              while(ch!=10){ ch=fgetc(frp);}

// I iregular bench?
              fscanf(frp,"%d %d ",&ire_ben,&nben);
              //printf("Regular ben = %d, num ben= %d
                  \n",ire_ben,nben);
              ch=fgetc(frp);
              if(ch=='.') {printf("'-number of benchs
                ' error1 in
                parameter file\n"); exit(0);}
              if(nben<=0){printf("'-number of
              benchs ' error2 in parameter file\n");
              exit(0);}
              while(ch!=10){ ch=fgetc(frp);}

              if(ire_ben>=1)
              {
                  fscanf(frp,"%d",&nuz);
                  //printf("benht = %d \n",nuz);
                  ch=fgetc(frp);
                  if(nuz<=0){printf("'-bench height
                    ' error in
                    parameter file\n");
                    exit(0);}
                  while(ch!=10){ ch=fgetc(frp);}
```

```
                    }

                    else{
                    if(nben<=1){printf("'-number of benchs '
                     error in parameter
                       file\n"); exit(0);}
                    tem=0;
                    for(i=0; i<nben; i++)
                    { fscanf(frp,"%d",&mbenht[i]);
                       if(mbenht[i]<=0){printf("'-bench height
                        for irregular benchs
                         ' error in parameter file\n");
                          exit(0);}
                       // printf("Multi-ben %d \n",mbenht[i]);
                       tem=tem+mbenht[i];
                    }
                    if ((tem<=0) ||(tem>nz)) {printf("'-bench
                    height for irregular
                       benchs  ' error of sum of multi_bench:
                        in parameter file\n");
                       exit(0);}

                    ch=fgetc(frp);
                    while(ch!=10){ ch=fgetc(frp);}
                }
//Advance and width of mining scheme
```

```
                    fscanf(frp,"%d  %d ",&nux,&nuy);
                    //printf("Advance length, width = %d,
                    %d \n",nux,nuy);
                    ch=fgetc(frp);
                    if(ch=='.') {printf("'-advance and
                    width of working
                       face(scheme) ' error in parameter file\n");
                        exit(0);}
                    if ((nux<=0) ||(nuy<=0)){printf("'-advance and
                     width of working face(scheme) ' error in
                     parameter file\n");
                     exit(0);}
                    while(ch!=10){ ch=fgetc(frp);}

// Mining rate
                    fscanf(frp,"%f",&mining_rate);
                    //printf("Mining rate = %f, \n",mining_rate);
                    ch=fgetc(frp);
                    if (mining_rate<=0) {printf("' -mining rate
                    II (m^3/unit time)
                    WF I' error in parameter file\n"); exit(0);}
                    while(ch!=10){ ch=fgetc(frp);}

//////////////////////////////////////////////////////
////   Second II work faces parameters setting   ////
//////////////////////////////////////////////////////
```

```
        if (mwface>1)
        {

// file for inputing data of site one, work face one
                    fscanf(frp,"%s",&inputdata2);
                    ch=fgetc(frp);
                    while(ch!=10){ ch=fgetc(frp);}
                    // printf("\n file is  %s \n ",inputdata2);
// column for variable
                    fscanf(frp,"%d",&var_colum2);
                    ch=fgetc(frp);
                    if(ch=='.') {printf("'-column for variable
                     II' error\n");
                     exit(0);}
                    if(var_colum2<=0){printf("'-column II
                      for variable'error in parameter file\n");
                      exit(0);}
                    while(ch!=10){ ch=fgetc(frp);}
// nx, ny,nz
                    fscanf(frp,"%d  %d  %d ",&nx2,&ny2,&nz2);
                    //  printf("\n unit 2 %d  %d  %d \n ",
                    nx2,ny2,nz2);
                    ch=fgetc(frp);
                    if(ch=='.') {printf("'-nx,ny,nz II' error in
                        parameter file\n"); exit(0);}
                    if((nx2<=0) ||(ny2<=0)||(nz2<=0))
```

```
                    {printf("'-nx,ny,nz
                        II' error in parameter file\n");
                         exit(0);}
                    while(ch!=10){ ch=fgetc(frp);}
// unit size of x, y, z
                    fscanf(frp,"%d  %d  %d",&unitx2,&unity2,&unitz2);
                    //
                    ch=fgetc(frp);
                    if(ch=='.') {printf("'-cell size II' error
                     in parameter file=%d\n",ch);
                     exit(0);}
                    if((unitx2<=0) ||(unity2<=0)||(unitz2<=0))
                    {printf("'-cell
                       size' error in parameter file\n");
                       exit(0);}
                    while(ch!=10){ ch=fgetc(frp);}

// limit for Min and Max
                    fscanf(frp,"%f  %f ",&tmin2,&tmax2);
                    ch=fgetc(frp);
                    if((tmin2>tmax2)||(tmax2<=0))
                    {printf("'-trimming limits
                     II' error in parameter file\n");
                      exit(0);}
                    while(ch!=10){ ch=fgetc(frp);}
```

```
// I iregular bench?
                    fscanf(frp,"%d %d ",&ire_ben2,&nben2);
                    ch=fgetc(frp);
                    if(ch=='.') {printf("'-number of benchs II'
                     error1 in parameter file\n");
                     exit(0);}
                    if(nben2<=0){printf("'-number of benchs
                     II' error2 in parameter file\n");
                     exit(0);}
                    while(ch!=10){ ch=fgetc(frp);}

                  if(ire_ben2>=1)
                      {
                          fscanf(frp,"%d",&nuz2);
                          ch=fgetc(frp);
                          if(nuz2<=0){printf("'-bench height II' error in
                             parameter file\n"); exit(0);}
                          while(ch!=10){ ch=fgetc(frp);}
                      }

                  else{
                      if(nben2<=1){printf("'-number of benchs
                      II' error in parameter file\n");
                      exit(0);}
                      tem=0;
                      for(i=0; i<nben2; i++)
```

```
                    {   fscanf(frp,"%d",&mbenht2[i]);
                        if(mbenht2[i]<=0)
                        {printf("'-bench height for
                        irregular benchs II ' error in
                        parameter file\n");
                        exit(0);}
                        tem=tem+mbenht2[i];
                    }
                if ((tem<=0) ||(tem>nz2))
                {printf("'-bench height
                   for irregular benchs II  ' error of s
                   um of multi_bench:
                      in parameter file\n");
                      exit(0);}
                ch=fgetc(frp);
                while(ch!=10){ ch=fgetc(frp);}
                }
//Advance and width of mining scheme
                fscanf(frp,"%d  %d ",&nux2,&nuy2);
                ch=fgetc(frp);
                if(ch=='.') {printf("'-advance and width of
                 working face(scheme) II ' error in
                  parameter file\n");
                     exit(0);}
                if ((nux2<=0) ||(nuy2<=0))
                {printf("'-advance
```

```
                    and width of working face(scheme)
                    ' error in parameter file\n");
                    exit(0);}
                  while(ch!=10){ ch=fgetc(frp);}
// Mining rate
                  fscanf(frp,"%f",&mining_rate2);
                  ch=fgetc(frp);
                  if (mining_rate2<=0)
                  {printf("' -mining rate II
                   (m^3/unit time) WFII'error in
                   parameter file\n");
                   exit(0);}
                  // while(ch!=10){ ch=fgetc(frp);}
                  // pay attention to
                  // printf("Mining rate = %f,
                  \n",mining_rate2);
        }
////////////////////////////////////////////////////////////
    }
        fclose(frp);
}

//////////////////////////
// write parameter file //
//////////////////////////
void writepara() {
```

```
FILE *fwr;
fwr=fopen("STMOD.par","wb");
fprintf(fwr,"                    Parameters for STMOD\n
**********\n\n");
fprintf(fwr,"START OF PARAMETERS:\n");
fprintf(fwr,"2
-2: two work faces, 1: one work face\n");
fprintf(fwr,"5.5
-cutoff grade\n");
fprintf(fwr,"60
-total number of time units\n");
fprintf(fwr,"STMOD.out
-file for time variation output\n");
fprintf(fwr,"data.dat
-file with site I data input\n");
fprintf(fwr,"2
-column for variable\n");
fprintf(fwr,"328    288  15
-nx,ny,nz\n");
fprintf(fwr,"1  1   1
-Cell size\n");
fprintf(fwr,"-1.0e21   1.0e21
-trimming limits(variable limits)\n");
fprintf(fwr,"1  1
-1: regular(1) or irregular bench(0), number of benches\n");
fprintf(fwr,"15     3   5   3   3
```

```
    -bench height for irregular benchs\n");
    fprintf(fwr,"8  8
    -advance and width of working face(scheme)\n");
    fprintf(fwr,"44.2
    -mining rate(m^3/unit time)\n");
    fprintf(fwr,"data2.dat
    -file with site I data input\n");
    fprintf(fwr,"1
    -column for variable\n");
    fprintf(fwr,"300    200  50
    -nx,ny,nz\n");
    fprintf(fwr,"1  1   1
    -Cell size\n");
    fprintf(fwr,"-1.0e21   1.0e21
    -trimming limits(variable limits)\n");
    fprintf(fwr,"1  1
    -1: regular(1) or irregular bench(0), number of benches\n");
    fprintf(fwr,"15     3   5   3   3
    -bench height for irregular benchs\n");
    fprintf(fwr,"5  6
    -advance and width of working face(scheme)\n");
    fprintf(fwr,"50.5
    -mining rate(m^3/unit time)\n");
    fclose(fwr);
}
```

```
////////////////////////////////////////////////////////////
// Check and display the result from reading the parameter  //
// & data files                                             //
////////////////////////////////////////////////////////////
void disres() {
    int i;
    if(mwface>1)
         printf("TWO mining work faces for calculating time
          variation!\n");
    else printf("ONE mining work face for calculating time
     variation!\n");
    printf("Cutoff grade:            %f\n",cutoff);
    printf("Total mining time units: %f\n",total_mining);
    printf("Output data file:    %s \n",outdata);

// Site one
    printf("Data file for site I:    %s\n",inputdata);
    printf("Variable colum is:   %d\n",var_colum);
    printf("nx, ny, nz:      %d, %d, %d \n",nx,ny,nz);
    printf("Cell size:       %d, %d, %d\n",unitx, unity,unitz);
    printf("tmin=%6.3e,        tmax=%6.3e \n",tmin,tmax);
    printf("Regular 1:  %d bench number:  %d\n",ire_ben,nben);
    if(ire_ben>=1)
                    {
    printf("bench height:            %d \n",nuz);
              if(nuz>nz){printf("Bench height is too big!\n");
```

```
                    exit(0);}
                }
            else{   printf("Multi-bench height:");
                    for(i=0; i<nben; i++)
                    {
                      printf("   %d,  ",mbenht[i]);
                      if(mbenht[i]>nz){printf("Mutiple
                      Bench height is too big!\n");
                      exit(0);}
                    }
                    printf("\n");
                }
    printf("Advance length, width:   %d, %d \n",nux,nuy);
    if(nux>nx){printf("Advance length is too big!\n");
        exit(0);}
    if(nuy>ny){printf("Advance width is too big!\n");
        exit(0);}
    printf("Mining rate:          %f \n",mining_rate);

//  Site II
if(mwface>=2) {
    printf("Data File for Site II:   %s\n",inputdata2);
    printf("Variable colum is:   %d\n",var_colum2);
    printf("nx, ny, nz:      %d, %d, %d \n",nx2,ny2,nz2);
    printf("Cell size:       %d, %d, %d\n",unitx2, unity2,
    unitz2);
```

```
printf("tmin=%6.3e,        tmax=%6.3e \n",tmin2,tmax2);
printf("Regular 1:     %d          bench number:  %d\n",
ire_ben2,nben2);
if(ire_ben2>=1)
                {
printf("bench height:              %d \n",nuz2);
                    if(nuz2>nz2){printf("Bench height is
                    too big!\n");
                    exit(0);}
                }

            else{   printf("Multi-bench height:");
                    for(i=0; i<nben2; i++)
                    {
                      printf("   %d,  ",mbenht2[i]);
                      if(mbenht2[i]>nz2){printf("Mutiple
                      Bench height is too big!\n");
                      exit(0);}
                    }
                    printf("\n");
                }
printf("Advance length, width:   %d, %d \n",nux2,nuy2);
if(nux>nx){printf("Advance length is too big!\n"); exit(0);}
if(nuy>ny){printf("Advance width is too big!\n"); exit(0);}
printf("Mining rate:               %f",mining_rate2);
}
```

```
        // readdata(inputdata,var_colum);
}

////////////////////////////////////////////////////
/// Time Variation Calculation          //////////////
////////////////////////////////////////////////////
void timevari() {
    long   i,j,k,num,ii;                // loop count
    int    nxyz,nxyz2,enxyz,enxyz2; // nxyz =nx *ny*nz;
    int    ix,iy,iz;
    // index of cell in the mining scheme
    int    numsx,numsy,numsz,numsx2,numsy2,numsz2;
    // number of segments in x,y,z direction
    long   index,indexx,indexy;
    int    mining_unit, mining_unit2,mining_count;
    int    int_1,int_2;
    float  float_1, float_2, tem1, tem2;

    readdata(inputdata,var_colum);
    mining_unit=unitx*unity*unitz;
    nxyz=nx*ny*nz;
    numsx=nx/nux;
    numsx=(int)numsx;
    numsy=ny/nuy;
    numsy=(int)numsy;
```

```
if(ire_ben>=1) // determine Z value
{
        numsz=nz/nuz;
        numsz=(int)numsz;
        for (i=0;i<numsz; i++)
        { mbenht[i]=nuz;}
        nuz=nuz*numsz;
        // printf("ire_ben=%d, nuz= %d, numsz=%d
        height= %d \n",ire_ben,nuz, numsz,mbenht[0]);
}
else
{       numsz=nben;
        nuz=0;
        for(i=0; i<nben; i++)
        nuz=nuz+mbenht[i];
}

enxyz =(numsx*nux)*(numsy*nuy)*nuz;
tem1= enxyz*mining_unit/mining_rate;
mining_rate = mining_rate/mining_unit;
if (tem1< total_mining)  total_mining=tem1;

iorder=calloc(nxyz,4);
if (iorder==0) {printf("Allocate memory for
'iorder' I error!\n"); exit;}
num=0;
```

```
for(i=1; i<=numsz; i++)
{
    nuz=0;
    for(ii=0;ii<(i-1); ii++)
       nuz=nuz+mbenht[ii];

    for(j=1; j<=numsy; j++)
    if(fmod(j,2)==1)
    {
        for(k=1; k<=numsx; k++)
        {
            for(iy=0; iy<nuy; iy++)
            //{{
            {   indexy=(iy+(j-1)*nuy)*nx;

                for(ix=0; ix<nux; ix++)
                //{{{
                {
                    indexx=ix+(k-1)*nux;
                    for(iz=0; iz<mbenht[i-1]; iz++)

                    {
                    index=indexx+indexy+(iz+nuz)*nx*ny;
                    iorder[num]=var[index];
                    num=num+1;
                    }
```

```
            } // {{{
        } // {{
    }
}
else
{
    //printf("This  is  even  =  %d\n",j);
    for(k=numsx; k>0;  k--)
    {
        for(iy=0; iy<nuy; iy++)
        //{{
        {   indexy=(iy+(j-1)*nuy)*nx;

            for(ix=(nux-1); ix>=0; ix--)
            //{{{
            {
                indexx=ix+(k-1)*nux;
                for(iz=0; iz<mbenht[i-1]; iz++)

                {index=indexx+indexy+(iz+nuz)*nx*ny;
                 iorder[num]=var[index];
                 num=num+1;
                }
            }//{{{
        }//{{
    }
```

```
    }
}
    free(var);
    int_1=0;
    float_1=0;
    mining_count=0;
    time=calloc((int)total_mining,4);
    if (time==0){printf("Allocate memory
    for 'time' I error!\n"); exit;}
    while(mining_count<(int)total_mining)
    {
            // determine the range of cells for calculation;
            tem1=(mining_count+1)*mining_rate;
            int_2=(int) tem1;
            float_2=tem1-int_2;
            // average time variation;
            tem2=iorder[int_1]*(1-float_1);
            for(i=(int_1+1); i<int_2; i++)
                tem2=tem2+iorder[i];
            tem2=tem2+iorder[int_2]*float_2;
            time[mining_count]=tem2;
            // Loop again;
            int_1=int_2;
            float_1=float_2;
            mining_count=mining_count+1;
    }
```

```
      free(iorder);
if(mwface>=2)
{     // if {
      mining_unit2=unitx2*unity2*unitz2;
      nxyz2=nx2*ny2*nz2;
      numsx2=nx2/nux2;
      numsx2=(int)numsx2;
      numsy2=ny2/nuy2;
      numsy2=(int)numsy2;
      if(ire_ben2>=1)
           {
              numsz2=nz2/nuz2;
              numsz2=(int)numsz2;
              for (i=0;i<numsz2; i++)
                  { mbenht2[i]=nuz2;}
                  nuz2=nuz2*numsz2;
          }
      else{   numsz2=nben2;
              nuz2=0;
              for(i=0; i<nben2; i++)
              nuz2=nuz2+mbenht2[i];
          }
      enxyz2=(numsx2*nux2)*(numsy2*nuy2)*nuz2;
      tem1=enxyz2*mining_unit2/mining_rate2;
      mining_rate2=mining_rate2/mining_unit2;
      if(tem1<total_mining) total_mining=tem1;
```

```
    printf("\nReal Mining Time Units is %d!\n",
    (int) total_mining);

    readdata(inputdata2,var_colum2);
    iorder=calloc(nxyz2,4);
    if (iorder==0) {printf("Allocate memory for
    'iorder' II error!\n"); exit;}
num=0;
for(i=1; i<=numsz2; i++)
{
    nuz2=0;
    for(ii=0;ii<(i-1); ii++)
       nuz2=nuz2+mbenht2[ii];

    for(j=1; j<=numsy2; j++)
    if(fmod(j,2)==1)
    {
        for(k=1; k<=numsx2; k++)
        {
            for(iy=0; iy<nuy2; iy++)
            //{{
            {   indexy=(iy+(j-1)*nuy2)*nx2;

                for(ix=0; ix<nux2; ix++)
                //{{{
                {
```

```
                    indexx=ix+(k-1)*nux2;
                    for(iz=0; iz<mbenht2[i-1]; iz++)

                    {
                        index=indexx+indexy+(iz+nuz2)*nx2*ny2;
                        iorder[num]=var[index];
                        num=num+1;
                    }
                } // {{{
            } // {{
        }
    }
    else
    {
        //printf("This  is  even  =  %d\n",j);
        for(k=numsx2; k>0;  k--)
        {
            for(iy=0; iy<nuy2; iy++)
            //{{
            {   indexy=(iy+(j-1)*nuy2)*nx2;

                for(ix=(nux2-1); ix>=0; ix--)
                //{{{
                {
                    indexx=ix+(k-1)*nux2;
                    for(iz=0; iz<mbenht2[i-1]; iz++)
```

```
                        {   index=indexx+indexy+(iz+nuz2)*nx2*ny2;
                            iorder[num]=var[index];
                            num=num+1;
                        }
                    }//{{{
                }//{{
            }
        }
    }
        free(var);
        int_1=0;
        float_1=0;
        mining_count=0;
        time2=calloc((int)total_mining,4);
//  printf("\nTime 2 is %f!\n",total_mining);
        if(time2==0){printf("Allocate memory for
        'time' II error!\n"); exit;}
        while(mining_count<(int)total_mining)
        {
                // determine the range of
                // cells for calculation;
                tem1=(mining_count+1)*mining_rate2;
                int_2=(int) tem1;
                float_2=tem1-int_2;
```

```
                // average time variation;
                tem2=iorder[int_1]*(1-float_1);
                for(i=(int_1+1); i<int_2; i++)
                    tem2=tem2+iorder[i];
                tem2=tem2+iorder[int_2]*float_2;
                time2[mining_count]=tem2;
                // Loop again;
                int_1=int_2;
                float_1=float_2;
                mining_count=mining_count+1;
        }
        free(iorder);
    } // if }
}

////////////////////////////////////////////////
// Final time variation calculation        //
////////////////////////////////////////////////
void fintime() {          int i;
            FILE   *out;
            float tem1,tem2;

        //  strcpy(outdata1,outdata);
        //  strcat(outdata,".out");
        //  strcat(outdata1,".was");
```

```
            out=fopen(outdata, "wb");
//  printf("\n String out= %s,   %s \n",outdata,outdata1);
        //  out1=fopen(outdata1, "wb");
            i=1;
            // if (to
            fprintf(out,"STMOD--Time variation calculation
            \n%d\nTime Variation of %s\n",i,var_name);
        //  fprintf(out1,"STMOD--Time variation of Waste
        \n%d\nTime Variation of waste\n",i);
            if(mwface>=2)
            {
                for (i=0; i<(int) total_mining; i++)
                {
                    tem1=(time[i]+time2[i])/(mining_rate+mining_rate2);
                    if(tem1<cutoff) tem2=1.0;else tem2=0.0;
                    fprintf(out,"%f \n",tem1);
                //  fprintf(out1,"%f \n",tem2);
                }
                free(time2);
                free(time);
            }
            else
            {
                for (i=0; i<(int) total_mining; i++)
                {
                    tem1=time[i]/mining_rate;
```

```
                    if(tem1<cutoff) tem2=1.0;else tem2=0.0;
                    fprintf(out,"%f \n",tem1);
                //  fprintf(out1,"%f \n",tem2);
                    }
                    free(time);
                }
                fclose(out);
                //fclose(out1);
}

//*********************************************************************
// Post Processing for Time variation based on the output file from *
// Time1.c, in which there are two columns, one for grade variation,*
// another for waste using binary expression(0 is the ore, 1 for    *
// waste). First created on Jan. 3, 2000.                           *
//*********************************************************************
#include "math.h"
#include "stdio.h"
#include "stdlib.h"
#include "malloc.h"
#include "string.h"
// External functions
void  readpara();
void  readdata();
void  writepara();
void  disres();
```

```
void   postp();

// Variables read parameter file
int      var_colum, total_mining,dsize;
char     *inputdata[60],
         *outdata[60],*outdata2[60],
         *outdata3[60],*var_name[80];
float    tmin,tmax;
float    Ore_total_mining, mining_rate;
float    cutoff;// total_mining is the time interval.
         // Variables read data file
float    *var;

int main() {
    char   mfname[60]={"PostTime.par"},ft;
    // name of parameter file
    printf("The parameter file is 'PostTime.par', Y/N? [Y]:");
    ft=getchar();
    if((ft=='n')||(ft=='N'))
    {   printf("\nPlease input new parameter file:");
        scanf("%s",&mfname);
        printf("\nThe parameter file is %s\n",mfname);
    }
    readpara(mfname);
    disres();
    readdata(inputdata,var_colum,dsize);
```

```
    postp();
    return 0;
}

//Read data file
void readdata(fname,colum_v,txyz) int colum_v; char *fname[40]; {
    FILE    *frp;
    char    ch;
    char    *temvar[60];
    int     i,l,colum,size;
    float   cx[10];
    size=4;
    var=calloc(txyz,size);
    if (var==0){printf("Allocate memory error!\n"); exit;}
    if((frp=fopen(fname,"rb"))==NULL)
        {printf("\nCan not open the data file or not exist!");
         exit(0);}
    else{
                //fw=fopen("read.txt", "wb");
                ch=fgetc(frp);
                while(ch!=10){ ch=fgetc(frp);}
                fscanf(frp,"%d\n",&colum);

                for(i=0;i<colum;i++)
                {
                   do  { ch=fgetc(frp);    //putchar(ch);
```

```
            }while(ch!=10);
          // putchar(ch);
        }
        colum_v--;
        for(i=0; i<txyz;i++)
                  {
                      for(l=0; l<colum; l++)
                      {
                         size=fscanf(frp,"%f", &cx[l]);
                         if(size<1)
                         {    printf("\n column=%d ,
                         dataN=%d \n",colum, i);
                              printf("Error!
                              No enough data in
                              the data file!\n");
                              exit(0);
                         }
                      }
                      var[i]=cx[colum_v];
                      //printf("\ncolumn, i=%f", var[i]);
                      //fprintf(fw,"%f \n",var[i]);
                  }
        //printf("size= %d",size);
        fclose(frp);
       //fclose(fw);
}
```

```
}

void readpara(fname1) char *fname1[40]; {
    FILE *frp;
    char ch;
    int i;
    if((frp=fopen(fname1,"rb"))==NULL)
        {printf("\nCan not open the parameter file, Or %s
        does not exist!!\nA new formatted parameter file has
        been produced!!\n",fname1);
         writepara();
         exit(0);}
    else{
            for(i=0; i<4; i++)
                {
                        ch=fgetc(frp);
                        while(ch!=10){ ch=fgetc(frp);}
                }
                //printf("Read line 4 \n");
// file for inputing data of site one, work face one
                    fscanf(frp,"%s",&inputdata);
                    //printf("Read file name = %s
                    \n",inputdata);
                    ch=fgetc(frp);
                    while(ch!=10){ ch=fgetc(frp);}
// column for variable
```

```
                    fscanf(frp,"%d",&var_colum);
                    //printf("var colum = %d  \n",
                    var_colum);
                    ch=fgetc(frp);
                    if(ch=='.') {printf("'-column for
                    variable' error\n");
                    exit(0);}
                    if(var_colum<=0){printf("'-column
                    for variable' error in parameter file\n");
                    exit(0);}
                    while(ch!=10){ ch=fgetc(frp);}
// size of data
                    fscanf(frp,"%d",&dsize);
                    //printf("Total mining time = %f,
                    \n",total_mining);
                    ch=fgetc(frp);
                    if (dsize<=0) {printf("' -total number
                    of time units ' error in parameter file\n");
                    exit(0);}
                    while(ch!=10){ ch=fgetc(frp);}
// cutoff
                    fscanf(frp,"%f",&cutoff);
                    ch=fgetc(frp);
                    if (cutoff<=0) {printf("' -Cutoff grade'
                    error in parameter file\n");
                    exit(0);}
```

```
                    while(ch!=10){ ch=fgetc(frp);}
// limit for Min and Max
                    fscanf(frp,"%f  %f ",&tmin,&tmax);
                    //printf("tmin = %5.2f, tmax= %5.2f \n",
                    tmin,tmax);
                    ch=fgetc(frp);
                    if((tmin>tmax)||(tmax<=0))
                    {printf("'-trimming limits' error
                    in parameter file\n");
                    exit(0);}
                    while(ch!=10){ ch=fgetc(frp);}
// Mining rate in given interval
                    fscanf(frp,"%f",&mining_rate);
                    //printf("Mining rate = %f, \n",
                    mining_rate);
                    ch=fgetc(frp);
                    if (mining_rate<=0)
                    {printf("'
                    -mining rate II (m^3/unit time)
                    WF I' error in parameter file\n");
                    exit(0);}
                    while(ch!=10){ ch=fgetc(frp);}
// Ttime _intervals  for mining in given intervals
                    fscanf(frp,"%d",&total_mining);
                    //printf("Mining rate = %f, \n",
                    mining_rate);
```

```
                    ch=fgetc(frp);
                    if (total_mining<=0) {printf("'
                    -Total mining II (m^3/unit time) WF I'
                    error in parameter file\n");
                    exit(0);}
                    while(ch!=10){ ch=fgetc(frp);}
// Ore Mining rate
                    fscanf(frp,"%f",&Ore_total_mining);
                    //printf("Mining rate = %f, \n",
                    mining_rate);
                    ch=fgetc(frp);
                    if (Ore_total_mining<=0)
                    {printf("' -mining rate II (m^3/unit time)
                    WF I' error in parameter file\n");
                    exit(0);}
                    while(ch!=10){ ch=fgetc(frp);}
// file outputing data 1
                    fscanf(frp,"%s\n",&outdata);
                    //printf("output data = %s, \n",outdata);
                    ch=fgetc(frp);
                    while(ch!=10){ ch=fgetc(frp);}
// file outputing data 2
                    fscanf(frp,"%s\n",&outdata2);
                    //printf("output data = %s, \n",outdata);
                    ch=fgetc(frp);
                    while(ch!=10){ ch=fgetc(frp);}
```

```
// file 3
                    fscanf(frp,"%s\n",&outdata3);
                    //ch=fgetc(frp);

//////////////////////////////////////////////////////////
    }
        fclose(frp);
}

//////////////////////////
// write parameter file //
//////////////////////////
void writepara() {
    FILE *fwr;
    fwr=fopen("PostTime.par","wb");
    fprintf(fwr,"

    Parameters for PostTime\n
    **********************\n\n");
    fprintf(fwr,"START OF PARAMETERS:\n");
    fprintf(fwr,"stmod.out
    -input file of time variation\n");
    fprintf(fwr,"1
    -column for variable\n");
    fprintf(fwr,"60
    -size of data\n");
```

```
    fprintf(fwr,"11.5
    -cutoff grade\n");
    fprintf(fwr,"-1.0e21   1.0e21
    -trimming limits(variable ore limits)\n");
    fprintf(fwr,"3.1
    -mining rate in unit time interval t\n");
    fprintf(fwr,"2
    -number of time interval t\n");
    fprintf(fwr,"4.0
    -amount of ore to plant in time interval T\n");
    fprintf(fwr,"PostStmod.ouT
    -Output file 1\n");
    fprintf(fwr,"PostStmod.raT
    -Output file 2\n");
    fprintf(fwr,"PostStmod.raK
    -Output file 3\n");
    fclose(fwr);
}

////////////////////////////////////
// Check and display the result from//
reading the parameter & data files/ //
////////////////////////////////////
void disres() {
    printf("Data file for site I:    %s\n",inputdata);
    printf("Variable colum is:   %d\n",var_colum);
```

```
    printf("Size of data :       %d\n",dsize);
    printf("Cutoff grade:            %f\n",cutoff);
    printf("Tmin=%6.3e,        Tmax=%6.3e \n",tmin,tmax);
    printf("Mining rate:         %f\n",mining_rate);
    printf("Total mining time units: %d\n",total_mining);
    printf("Ore mining rate:         %f\n",Ore_total_mining);
    printf("Output data file 1:  %s \n",outdata);
    printf("Output data file 2:  %s \n",outdata2);
    printf("Output data file 3:  %s \n",outdata3);
}

void postp() {
    FILE *fw;
    long int i,j;
    double rem_test,tem;
    long int nloop;
    double  n_ore, rem, temfw;
    fw=fopen(outdata3, "wb");
    if(fw==0){printf("\nThe output file can not be opened\n");
    exit(0);}
    fprintf(fw, "Ore ranking by cutoff grade\n2\nEqual time
    intervals\nTime Variation of Waste, 1:waste, 0: ore\n");
    for (i=0; i<dsize; i++)
    {
        if (var[i]<cutoff)
        {
```

```
            var[i]=1;
        }
        else
        {   var[i]=0.0;
        }
        fprintf(fw, "%-6d   %-5d\n",(i+1),(int)var[i]);
    }
    fclose(fw);

////////////////////////////////////////////////////////
//       use to calculate the waste distribution
//         number in obtaibing equal ore volume
////////////////////////////////////////////////////////
    n_ore=Ore_total_mining/mining_rate;
    rem=n_ore;
    rem_test=10.0;
    i=0;
    tem=0;
    fw=fopen(outdata,"wb");
    if(fw==0){printf("\nThe output file can not be opened\n");
    exit(0);}
    fprintf(fw, "Waste variation with time interval
    by obtaining equal ore\n3\nTime intervals\nOre
    distribution\nWaste distribution\n");
    do
    {
```

```
        while (rem_test>1.0)
        {   if(var[i]>=1.0){ tem=tem+1 ;}
            else
            {   if      (rem>=1.0) {rem=rem-1;}
                else
                {
                    rem_test=0.0;
                    rem=n_ore-(1-rem);
                    temfw=tem+rem;
                    tem=1-rem;
                }
            }
            i=i+1;
        }
        rem_test=10.0;
        fprintf(fw, "%-6d   %-9.3f   %-9.3f\n",i,
        n_ore*mining_rate,temfw*mining_rate);
    }while(i<dsize);
    fclose(fw);
////////////////////////////////////////////////
// use to calculate ratio of ore to waste volume
in equal time interval//
////////////////////////////////////////////////
    nloop=(int) dsize/total_mining;
    fw=fopen(outdata2,"wb");
    if(fw==0){printf("\nThe output file can not be opened\n");
```

```
    exit(0);}
    rem=(float) total_mining;
    fprintf(fw,"Ratio of Ore to total mining amount in equal
    time intervals\n3\nTime intervals \nOre ratio \nWaste ratio\n");
    for (i=0; i<nloop; i++)
    {
        tem=0;
        for(j=0;j<total_mining; j++)
        {
            if(var[i*total_mining+j]<1) { tem=tem+1;}
        }
        n_ore=100*(tem/rem);
        fprintf(fw, "%-6d  %-9.3f  %-9.3f \n",(i+1)*total_mining,
        n_ore*mining_rate,(100.0-n_ore)*mining_rate);
    }
    fclose(fw);
free(var);
}
```