APPLICATION OF LEAST SQUARES AND KRIGING IN MULTIVARIATE OPTIMIZATIONS OF FIELD DEVELOPMENT SCHEDULING AND WELL PLACEMENT DESIGN

A REPORT
SUBMITTED TO THE DEPARTMENT OF PETROLEUM ENGINEERING
OF STANFORD UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
MASTER OF SCIENCE

By
Yan Pan
July 1995

I certify that I have read this report and that in my opinion it is fully adequate, in scope and in quality, as partial fulfillment of the degree of Master of Science in Petroleum Engineering.

Dr. Roland Horne
(Principal advisor)

Acknowledgements

I would like to thank my research advisor Professor Roland N. Horne for his constant support, encouragement and guidance during the course of this study. Sincere thanks are also due to my friends, Antonio Bittencourt, Jorge Landa and Xianfa Deng for generously providing good ideas and techniques for this research.

Financial support from Stanford University Petroleum Research Institute (SUPRI B) and the Department of Petroleum Engineering are gratefully acknowledged.

Finally I like to thank my family for their love and support. My gratitude is endless to my parents, Zhengpu Pan and Ruyuan Zhang and to my brother Lei Pan.

Abstract

This study applied two multivariate interpolation algorithms, Least Squares and Kriging, to interpolate a limited number of data obtained from simulation in order to predict the optimal strategies in a field development scheduling project and a waterflood project with a significant reduction in the simulation efforts required.

The two projects were application examples with known answers. The objective of this study was to examine the feasibility and efficiency of multivariate interpolation in solving optimization problems by reducing the simulation effort required. The net present value was used as the objective function in both projects. The field development scheduling simulation was achieved by an economic model taking acount of all the costs and profits during the time period being studied. The waterflooding simulation was achieved by solving the Laplace equation and generating streamlines within the specified pattern. The movements of the waterfront were tracked and then the oil and water production at each producer was calculated.

The results obtained by interpolation showed that the multivariate algorithms are able to provide a global sketch of the objective function surface, to avoid possible failure at local optima, and to reach the absolute optimum by refining the grids near the intermediate optimum. The approaches were shown to be practical techniques in optimization, and are likely to be useful in full scale problems in which simulation costs would be prohibitive for iterative nonlinear optimization methods.

Contents

List of Tables

List of Figures

Section 1

Introduction

Optimization of oil and gas reservoir development requires integration of quantitative geological and geophysical analysis with appropriate flow models to assess alternative development and completion schemes and their relative economic values. Development plans are made early in the life of a reservoir, and may be difficult to reverse. It is critical to make optimal development decisions in order to obtain the maximum profit from the future oil and gas production.

In reservoir engineering, there are many parameters that may affect the oil production history, such as the reservoir properties, well positions and production scheduling parameters. In order to optimize the oil production, many evaluations of the possible combinations of all the decision variables are required. Running a simulator only to obtain all the data required is time consuming and expensive. For an example with 5 decision variables and 10 possible values for each parameter, the total combinations are 10^5. If it took only one minute for each simulation run, it would take 69.4 days to evaluate the 10^5 cases. This seems infeasible. A cheaper substitute would be to apply multivariate interpolation to the data obtained from a reduced number of simulations and then to search for the optimal strategies among the new realizations. The sample data need to be selected properly to represent the basic structure of the parameter space because the interpolation results depend on the reliability and distribution of

the samples. The optimal solution obtained from new realizations needs to be validated by simulation or from other sources.

There are several possible algorithms that can be used for multivariate interpolation: Lagrange, Spline, Least Squares and Kriging. Lagrange is direct and simple, but it is not stable because it tends to use high order polynomials in order to traverse each of the prior data points. Splines can generate smooth curves or surfaces, but for a multidimensional problem are very complicated because of the need to calculate all the derivatives of high order for all the variables. This study found that Least Squares and Kriging are two practical methods.

The Least Squares algorithm has been applied widely in fitting experimental data. It is usually used to solve overdetermined inverse problems. In recent years, some robust modified Least Squares algorithms have been developed and applied in reservoir engineering. Watson, Lane and Gatens [1990] developed an appropriate weighted-least-squares performance index and formulated a measurement-error model for history matching with cumulative production data. Olarewaju [1990] applied an automatic weighted constrained least-squares parameter estimation technique and the analytical model for pressure transient analysis to solve the permeability alteration problem around the wellbore. Chung and Kravaris [1991] proposed a novel history-matching algorithm on the basis of regularization, which is capable of incorporating a priori information about unknown reservoir parameters like porosity or permeability. This algorithm proceeds as optimization of an objective function which is formulated by combining three indices of different nature: (i) the ordinary least-squares term which measures the deviation of the model output from the pressure observation data, (ii) the stabilizing term which measures the non-smoothness of the parameter, and (iii) another penalty term which measures the deviation of the estimates from the a priori point data. Mohammed and Wattenbarger [1991] developed a PVT simulator to model routine PVT laboratory tests. This simulator generates the K-value correlation for the mixture and employs a least squares-linear programming optimization routine to fine tune the correlation such that the simulated PVT behavior matches

the actual one. Chardaire-Riviere, et al [1992] applied least-squares technique to determine relative permeability and capillary pressure simultaneously from the results of a single two-phase flow experiment. A high-order numerical scheme was used to reduce the numerical diffusion and the optimal control theory was used to solve the minimization problem. Bonalde and Romones [1994] presented a robust and efficient least-squares algorithm for parameter estimation in well test analysis. The algorithm is a Levenberg-Marquardt method with a 'trust region' approach for global convergence along with restriction in the unknown parameters.

Since Matheron [1] built the theory of regionalized variables and proposed a method of estimation which was called 'Kriging' in the 1960s, the applications of this method have been extended from mining to hydrosciences and petroleum engineering. Delhomme [1978] was one of the first to apply Kriging to water resources problems. He presented a series of case-studies in automatic contouring, data input for numerical models, estimation of average precipitation over a given catchment area, and measurement network design. Doctor [1979] evaluated the Kriging techniques for high level radioactive waste repository site characterization. Dunlap [1984] applied Kriging techniques to interpolate water-table altitudes in west-central Kansas. Wackernagel [1989] described a program for the factor analysis of multivariate data from samples taken in a physical environment. The spatial correlation of the samples is represented by a model of nested spatial structures. A series of applications to estimate heterogeneous reservoir properties have been reported in recent years. Tavares Ribeiro and da Costa e Silva [1986] presented a geomathematical model (Kriging) describing the spatial behavior and distribution of the principal reservoir properties. Brummert, et al [1989] compared the accuracy of four-point estimation methods: simple averaging, fifth-degree bicubic spline, inverse weighted distance squared and Kriging in five reservoir layers using data sets consisting of five different reservoir properties (horizontal permeability, vertical-horizontal permeability ratio, thickness, porosity and top of structure). Kriging was found to be the optimum method in petroleum-related problems. Al Rumhy, Archer and Daltaban [1991] used the geostatistical method of Kriging conditioned with geological information to indicate potential methods for

characterizing permeability of reservoir domains. Egeland, Hatlebakk, Holden and Larsen [1992] applied experimental design methods and Kriging techniques to predict the response variable from the sample input variables based on a relation between them estimated from the output of simulator ECLIPSE. The idea is to vary several input variables simultaneously and intelligently, compared to the traditional sensitivity study approach where one input variable is varied at a time. Watkins [1993] described how the framework developed for universal Kriging can accommodate a general formulation of the reservoir history matching problem. Poquioma and Kelkar [1994] presented the results of an investigation on using ordinary Kriging and sequential indicator simulation to generate distributions of porosity, permeability, initial water saturation, layer thickness and top of structure of an oil field in Venezuela.

Other techniques, such as linear programming, dynamic programming, simulated annealing, genetic algorithm, neural network, polytope and tabu search have also been used in optimization problems in petroleum engineering and water resources. Reznicek and Cheng [1991] reviewed various approaches to mathematical programming which deal with the uncertainties of the nature of hydrologic parameters in reservoir management. Hansen, et al [1992] applied mixed integer programming and tabu search to optimize the locations and capacities of offshore platforms for oil exploration. Alcobia, et al [1994] described an asset management approach which intergrates both a reservoir simulation model and an economic model to optimize the development of large multiplatform fields which have a wide variation in static and dynamic reservoir properties. Tauzin and Horne [1994] proposed an algorithm that uses simulated annealing to constrain the permeability distribution of a given reservoir model to the well test data at several wells. Fujii and Horne [1994] applied Newton-type methods, polytope method and genetic algorithm to optimize networked oil and gas production system. Rogers, Dowla and Johnson [1995] presented an approach to nonlinear optimization that uses the artificial neural networks and the genetic algorithm to search for more cost-effective remediation strategies within practical time frames in an environmental cleanup problem. Lo, Starley and Holden [1995] developed a production forecasting model that uses linear programming concepts to generate

composite forecasts from multiple streams produced through a common facility. The model treats petroleum processing as a linear optimization problem to maximize the oil production under a set of independent constraints.

This particular study concentrates on the optimization of multiple decision variables by combining simulation with multivariate interpolation (Least Squares and Kriging algorithms).

Section 2

Optimization Procedure

The main objective of this study is to optimize the field development scheduling and oil production by maximizing the net profit. Actually this is always the main concern of oil producing companies. As shown in Figure 2.1, the mathematical approach to the optimization developed in this study can be achieved by the following procedure:

1. Problem Analysis

In petroleum engineering, the objective function is normally the total oil production or the net present value over a certain time period. Based on the analysis of a given problem, the parameters that may have significant influence on oil production history and the potential profit are chosen as the decision variables to be optimized. They could be the properties of the reservoir, the well locations, well shut-in time or other production scheduling parameters. Depending on the constraints provided by the problem or by practical analysis, the range of each decision variable can be determined, and then the domain of the parameter space is specified. After the objective function and all the decision variables are determined, the optimization problem can be formulated as a maximization problem subject to certain constraints.

2. Sample Point Distribution Design

The parameters in the domain under study need to be discretized into certain number of points, and the objective function value at each grid point will be investigated in the following optimization process. Due to limited computer time or limited funding, exhaustive tests are not possible, therefore, only a reasonable number of tests can be performed. A combination of different levels of all the decision variables is designed to be used to run the simulators or perform field tests. Uniform design, an application of number theory in test design, which was developed by Fang [1980], is a method that can generate a test point distribution uniformly and reduce the number of combinations significantly. This method was applied in this study to develop the sample data point distribution. If the decision variables (reservoir properties or well locations) are correlated with reservoir geometry, then a nonuniform distribution based on the structure map obtained from geologists may be a good alternative choice.

3. Simulation

According to the test design, the objective function values at all the sample data point are evaluated by running the simulator. Different simulation model may be applied to solve different problems. Normally appropriate flow models for alternative development and completion schemes with relative economic values have to be considered. In this step, the prior data are obtained as a set of objective function values. Since the following interpolation results depend on the reliability of the prior data, it is important to select appropriate model and to perform the simulation accurately.

4. Multivariate Interpolation

Starting from the prior data, multivariate interpolation is performed to obtain new realizations at values of the decision variables that lie between the values used during simulation, normally at the exhaustive discretized grid points. The number of new realizations can be much larger than the number of prior data since the interpolation is much cheaper to compute than the full simulation. The distribution of the new

realizations is usually uniform in the domain investigated. In this study, two interpolation algorithms were investigated:

(a) Least Squares

(b) Kriging

During this process, the control parameters such as the degree of representative polynomials, have to be selected carefully to keep the balance between the interpolation accuracy and smoothness.

5. Optimization

The objective function surface can be generated from the new realizations. If it represents the basic structure of the parameter space under study, an optimal solution for all the decision variables can be obtained by searching among the new realizations for the maximum objective function value. This objective value may not be the real value. The main idea is to identify the vicinity of the optimal value in the domain. Then a more accurate objective value can be calculated by running the simulator again provided the optimal values of the decision parameters.

6. Validation

The recommended realization obtained from the previous step needs to be validated either by performing the simulation again near the intermediate optimal region or from other sources. If it is necessary, the new simulation data may be added to the previous sample data set, and interpolation, optimization and validation (step 4 to step 6) can be repeated until a satisfactory optimal solution is obtained.

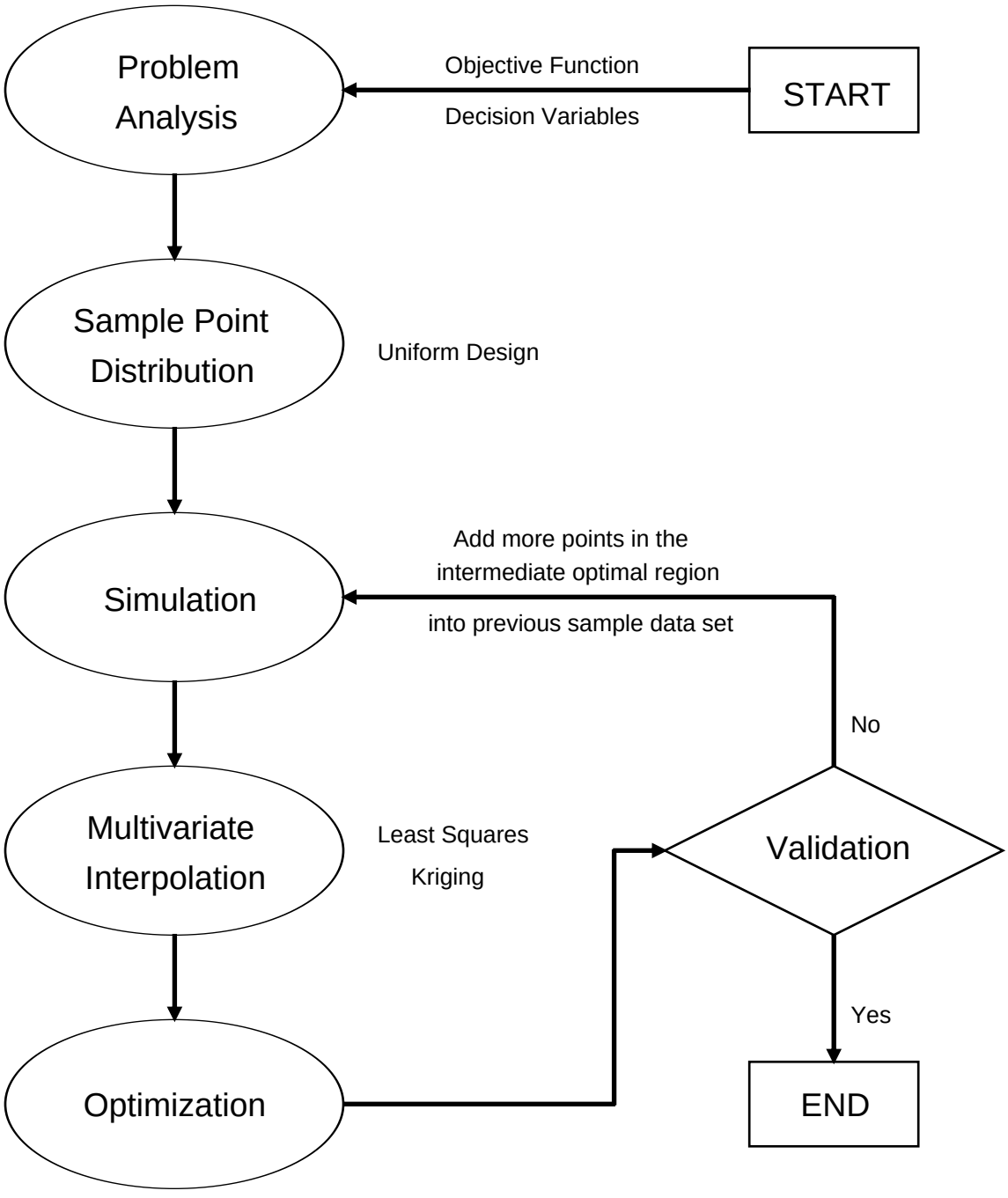

Figure 2.1: Optimization Procedure

Section 3

Uniform Design

3.1 Introduction

Instead of using traditional sensitivity studies where one input variable is varied at a time, a more efficient test design method for multiple factor point distribution problem was applied in this study. Uniform design (Fang, 1980) is based on number theory. At present, only tests with equal levels of each factor are considered. Assume there are s factors (decision variables) and each factor has q levels (discretized possible values of parameters), the number of total possible combinations is q^s. Orthogonal test design (Fang, 1977) can reduce the test number to q^2, uniform design can reduce it to only q tests.

The key idea of uniform design is to generate a test point distribution in the domain as uniformly as possible.

3.2 Principles

Uniform design follows some general principles:

1. q is an odd number q_o or $q_o - 1$.

Select positive integers $a_1, a_2, \ldots, a_s$ satisfying $(a_i, q) = 1,\ i = 1, \ldots, s.$ where (a, b) is the maximum common factor of a and b. Then the distributed points will be:

$$P_q(k) = (ka_1, ka_2, \ldots, ka_s)(mod\, q_o),\ k = 1, \ldots, q_o. \tag{3.1}$$

2. q is a prime number p or $p - 1$.

Select a positive integer $a(0 < a < p)$, then:

$$P_q(k) = (k, ka, ka^2, \ldots, ka^{s-1})(mod\, p),\ k = 1, \ldots, p. \tag{3.2}$$

Notice that 2 is a special case of 1, so one theorem to decide $a_1, a_2, \ldots, a_s$ is sufficient.

3.3 Uniform Theorem

The vector $\vec{a} = (a_1, a_2, \ldots, a_s)$ or $(1, a, a^2, \ldots, a^{s-1})$ is the one which minimizes the following function:

$$\xi(q, \vec{a}) = \frac{1}{q} \sum_{k=1}^{q} \prod_{i=1}^{s} (1 - \frac{2}{\pi} ln(2 \sin \pi \frac{a_{ik}}{q+1})) \tag{3.3}$$

where $a_{ik} \equiv ka_i (mod\, q)$ and $a_{iq} \equiv q (mod\, q)$.

3.4 Examples

Example 1

Consider a simple two dimensional problem. There are two factors, and each factor has 11 levels: $s = 2$, $q = 11$. Select $a_1 = 1$, $1 < a_2 < 11$, then:

$$\begin{aligned}
\xi(11;\ 1,2) &= \xi(11;\ 1,6) &&= 0.761644690,\\
\xi(11;\ 1,3) &= \xi(11;\ 1,4) &&= 0.749591730,\\
\xi(11;\ 1,5) &= \xi(11;\ 1,9) &&= 0.753757910,\\
\xi(11;\ 1,7) &= \xi(11;\ 1,8) &&= 0.744916550,\\
&\xi(11;\ 1,10) &&= 0.802664600.
\end{aligned}$$

The minimum is $\xi(11;\ 1,7)$ or $\xi(11;\ 1,8)$, therefore, the best choice of $\vec{a}$ is $\{1,7\}$ or $\{1,8\}$, and the uniform distribution is:

$$\begin{aligned} P_q(k) &= (k, 7k)(mod\ 11),\ k = 1, 2, ..., 11, \text{ or} \\ P_q(k) &= (k, 8k)(mod\ 11),\ k = 1, 2, ..., 11. \end{aligned}$$

Example 2

In the field development scheduling project [section 5], two decision variables with 35 levels: starting time and development time were selected as the prior data input to the simulator. The point distribution was designed uniformly applying the above method. The number of test points was reduced from $35^2 = 1225$ to 35 only. Figure 3.1 shows the test point distribution of the two parameters.

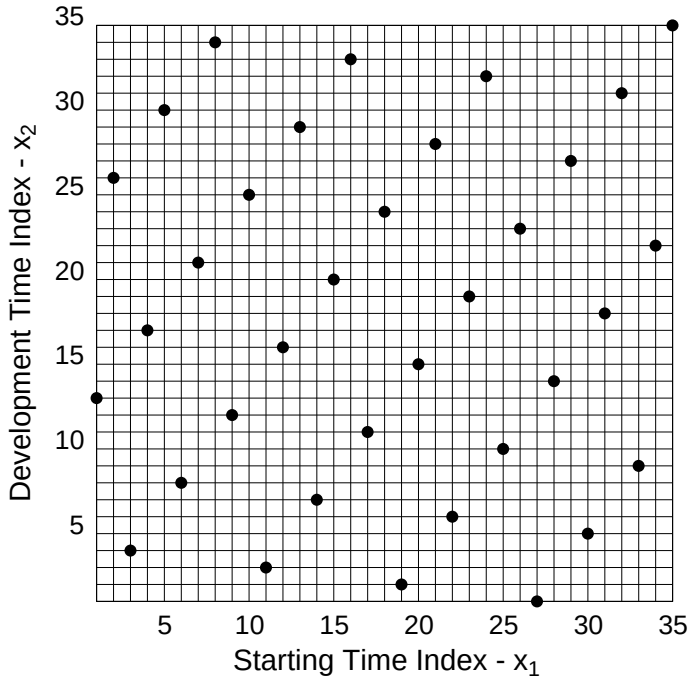

Figure 3.1: Test Point Distribution for Scheduling Simulation

Section 4

Multivariate Interpolation

In this study, the Least Squares and Kriging algorithms were applied to perform the multivariate interpolation. The basic idea of the Least Squares algorithm is to minimize the squared residuals between the data values and the representative function values. Only one linear system needs to be solved to generate all the new realizations. The surface generated by Least Squares interpolation may not traverse all the sample data points. The basic idea of the Kriging algorithm is to make the prediction of the random function representing the regionalized variable unbiased and optimal. For each new realization, one linear system needs to be solved. Therefore, Kriging interpolation takes more computation time than Least Squares, but is an exact interpolator and takes account of more information among the prior data which the Least Squares method ignores. These two algorithms are described in this section, and the corresponding interpolation results are presented in the following sections.

4.1 Least Squares Algorithm

The purpose of Least Squares is to construct a representative function F composed of simple known functions, such as polynomials, which minimizes the sum of the squared residuals between the data values and the function values. If the number of prior data equals the number of unknown functional coefficients, the representative surface generated by Least Squares interpolation may traverse all the prior data points, but

in general, to generate a smooth surface, the number of coefficients is chosen to be smaller than the number of data points. In other words, the Least Squares algorithm is normally applied to solve overdetermined problems.

Problem Description

For an n-dimensional problem, there are n decision variables to be optimized and a set of m prior data is to be interpolated. The prior data can be given as:

$$\mathbf{X} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ x_{m1} & \cdots & \cdots & x_{mn} \end{bmatrix} \quad \mathbf{y} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{pmatrix} \quad \mathbf{W} = \begin{bmatrix} w_1 & 0 & \cdots & 0 \\ 0 & w_2 & \cdots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & \cdots & w_m \end{bmatrix} \tag{4.1}$$

where $\mathbf{X}$ represents a set of m points in the n-dimensional data space; $\mathbf{y}$ is a vector composed of the corresponding objective values; $\mathbf{W}$ is a diagonal matrix of the weighting factors of the data points. In this study, equal weighting factors were used, so $\mathbf{W} = \mathbf{I}$.

Determination of Polynomials

At first, the polynomials representing the relation between the objective value and the decision variables have to be determined. Since this is an overdetermined problem, the highest degree k of the polynomials is chosen to satisfy $l = \begin{pmatrix} n+k \\ n \end{pmatrix} \leq m$. The number of unknown coefficients l is not bigger than the number of data m. The exhaustive basis of the polynomials with highest degree k in n-dimensional space are:

$$\begin{aligned} \vec{f} = \{f_j\} \quad = \quad & \{1, x_1, x_2, \ldots, x_n, x_1^2, x_1x_2, \ldots, x_n^2, x_1^3, x_1^2x_2, \ldots, x_1x_2x_3, \ldots, x_n^3, \\ & \ldots, x_1^k, \ldots, x_1x_2\cdots x_k, \ldots, x_n^k\} \end{aligned} \tag{4.2}$$

In this study, to simplify the problem, instead of using the entire basis of the polynomials in the n-dimensional space, only the terms composed of one or two variables

were used, in other words, only the correlation between each pair of two variables was taken into account. The simplied basis can be expressed as:

$$\begin{aligned}\vec{f} = \{f_j\} &= \{1, x_1, x_2, \ldots, x_n, x_1^2, x_1x_2, \ldots, x_n^2, \ldots, x_1^k, x_1^{k-1}x_2, \ldots, x_1x_n^{k-1}, \\ &\quad \ldots, x_{n-1}x_n^{k-1}, x_n^k\} \end{aligned} \tag{4.3}$$

Then the representative function F can be constructed as a linear combination of the simplified polynomial basis.

$$F = \sum_{j=0}^{l} \{c_j \cdot f_j\} \tag{4.4}$$

Determination of Functional Coefficients

The second step is to calculate the coefficient of each term c_j in Equation 4.4. This is achieved by minimizing the norm of the residual vector.

$$\begin{aligned} \text{Given: } y_i &= F_i(\vec{x_i}) + r_i \qquad i = 1, \ldots, m \\ \text{where } F_i &= \sum_{j=0}^{l} \{c_j \cdot f_j(\vec{x_i})\} \text{ the function value} \\ \vec{x_i} &= \{X_{i1}, X_{i2}, \ldots, X_{in}\} \text{ the position of the data point} \\ y_i &\quad \text{the corresponding objective value} \\ r_i &\quad \text{the residual} \end{aligned} \tag{4.5}$$

Equation 4.5 may be written in matrix form:

$$\begin{aligned} \mathbf{y} &= \mathbf{Ac} + \mathbf{r} \\ \text{where } \mathbf{c} &= \{c_0, c_1, \ldots, c_l\} \text{ the coefficient vector} \\ \mathbf{A} &= \{f_j(\vec{x_i})\} \quad i = 1, \ldots, m;\ j = 1, \ldots, l \end{aligned} \tag{4.6}$$

Weighting Equation 4.6 by $\mathbf{W}$,

$$\begin{aligned} \text{then,} \quad \mathbf{Wy} &= \mathbf{WAc} + \mathbf{Wr} \\ \text{or} \quad \mathbf{y}' &= \mathbf{A}'\mathbf{c} + \mathbf{r}' \\ \text{where} \quad \mathbf{y}' &= \mathbf{Wy} \\ \mathbf{A}' &= \mathbf{WA} \\ \mathbf{r}' &= \mathbf{Wr} \end{aligned} \tag{4.7}$$

To minimize $\Psi(\mathbf{c}) = \mathbf{r^T W r} = \mathbf{r'^T r}$, set $\dfrac{\partial \Psi(\mathbf{c})}{\partial \mathbf{c}} = \mathbf{0}$.

$$\Psi(\mathbf{c}) = (\mathbf{y'} - \mathbf{A'c})^{\mathbf{T}}(\mathbf{y} - \mathbf{Ac}) = \mathbf{y'^T y'} - 2\mathbf{c^T A'^T y} + \mathbf{c^T A'^T Ac}$$

$$\text{Set} \qquad \frac{\partial \Psi(\mathbf{c})}{\partial \mathbf{c}} = 2\mathbf{A'^T Ac} - 2\mathbf{A'^T y} = \mathbf{0}$$

$$\text{Then} \qquad \mathbf{A'^T Ac} = \mathbf{A'^T y}$$

$$\text{i.e.} \qquad \mathbf{A^T WAc} = \mathbf{A^T Wy} \tag{4.8}$$

The coefficient vector $\vec{c}$ can be obtained by solving linear system, Equation 4.8.

$$\mathbf{c} = (\mathbf{A^T WA})^{-1}\mathbf{A^T Wy} \tag{4.9}$$

Then, any new realizations can be evaluated by substituting the new parameters into the representative function directly.

$$y_{new} = F(\vec{x}_{new}) = \sum_{j=0}^{l} \{c_j \cdot f_j(\vec{x}_{new})\} \tag{4.10}$$

In summary, the Least Squares interpolation is performed following the steps in Equations 4.1, 4.3, 4.4, 4.9 and 4.10.

Properties of Least Squares

The accuracy of Least Squares interpolation depends on the highest degree of the polynomials used. Within the allowable range, the higher the degree of polynomials chosen, the smaller the residuals will be. However, the degree has the opposite effect on the smoothness of the representative function. Therefore, the balance between the interpolation accuracy and smoothness has to be considered.

4.2 Kriging Algorithm

The Kriging algorithm was first introduced by Matheron [1] in the beginning of the sixties. It is based on the theory of regionalized variables. The term 'regionalized' describes a phenomenon which is spread out in space (and/or in time) and which shows a certain structure. A variable which characterizes such a phenomenon is called a 'regionalized variable'. In fact, not only variables describing the underground or the atmosphere can be treated as regionalized variables. Any characteristic variable of natural phenomena can be spread out in the related parameter space. There exist apparent or hidden correlations among the parameters. Therefore, the application of the theory of regionalized variables can be extended to general interpolation problems. From a mathematical standpoint, a regionalized variable is simply a function $z(\vec{x})$ which gives the value at point $\vec{x}$ in an n-dimensional space of a characteristic z of the natural phenomenon being studied. In this study, z refers to the objective function value and $\vec{x}$ refers to the decision variable vector.

The Intrinsic Hypothesis

The probabilistic theory of random functions is used to deal with regionalized variables. A regionalized variable denoted by $z(\vec{x})$ is interpreted as a realization of a random function denoted by $Z(\vec{x})$. The intrinsic hypothesis is applied to reconstitute the distribution law of this random function from the prior data. The hypothesis is that: for any vector $\vec{h}$, the increment of the random function $Z(\vec{x}+\vec{h}) - Z(\vec{x})$ has zero expectation and a variance which is independent of the point $\vec{x}$. This can be expressed as:

$$\begin{cases} E[Z(\vec{x}+\vec{h}) - Z(\vec{x})] = 0 \\ var[Z(\vec{x}+\vec{h}) - Z(\vec{x})] = 2\gamma(\vec{h}) \end{cases} \tag{4.11}$$

A random function which satisfies this hypothesis is called an intrinsic random function. The function $\gamma(\vec{h})$ is called the semivariogram.

The Semivariogram

As in Equation 4.11, the semivariogram is defined as:

$$\gamma(\vec{h}) = \frac{1}{2} var[Z(\vec{x} + \vec{h}) - Z(\vec{x})] \tag{4.12}$$

It may also be written as:

$$\gamma(\vec{h}) = \frac{1}{2} E[(Z(\vec{x} + \vec{h}) - Z(\vec{x}))^2] \tag{4.13}$$

which means that $2\gamma(\vec{h})$ is the mean squared difference for two points separated by a distance $\vec{h}$. The distance may not be the actual geological distance, it is the distance in the parameter space.

The semivariogram can be estimated from the sample data points:

$$\gamma(\vec{h}) = \frac{1}{2N(h)} \sum_{i=1}^{N(h)} (z(\vec{x_i} + \vec{h}) - z(\vec{x_i}))^2 \tag{4.14}$$

where $z(\vec{x_i})$ are the experimental values at points $\vec{x_i}$ such that data are available both at $\vec{x_i}$ and $\vec{x_i} + \vec{h}$, and $N(h)$ is the number of pairs of data points separated by a gap equal to $\vec{h}$. In general, the experimental points are irregularly spaced, they need to be regrouped by classes of distance. The separation vector $\vec{h}$ is specified with certain direction and distance tolerance.

The semivariogram is a function of vector $\vec{h}$ which is composed of the modulus $|h|$ and the direction. A comparison of the semivariograms obtained for different directions will reveal the anisotropies of the phenomenon. In the isotropic case $\gamma(\vec{h})$ is a function of the modulus $|h|$ only.

Due to the limited number of prior data and the sampling fluctuations, it is necessary to fit experimental semivariograms to simple theoretical models. A theoretical model for $\gamma(\vec{h})$ is used to solve a Kriging system which is described in the next section. The model has to be positive definite to ensure the existence and uniqueness of the solutions of the Kriging equations. The most commonly used semivariogram models

are described in Table 4.1 and their graphs are shown in Figure 4.1. In this study, based on the analysis of the sample data obtained by simulations, and to simplify the problem, the isotropic monomial model $\gamma(h) = h^a$ was used in the Kriging system.

Table 4.1: Models of Semivariograms in Widest Use

Type	Formula
Monomial	$\gamma(h) = c \cdot h^a \qquad 0 < a < 2, \qquad c > 0$
Spherical	$\gamma(h) = c \cdot Sph(\frac{h}{a}) = \begin{cases} c \cdot [1.5\frac{h}{a} - 0.5(\frac{h}{a})^3] & \text{for } h \leq a \\ c & \text{for } h \geq a \end{cases}$
Exponential	$\gamma(h) = c \cdot Exp(\frac{h}{a}) = c \cdot [1 - exp(-\frac{h}{a})]$
Gaussian	$\gamma(h) = c \cdot [1 - exp(-\frac{h^2}{a^2})]$

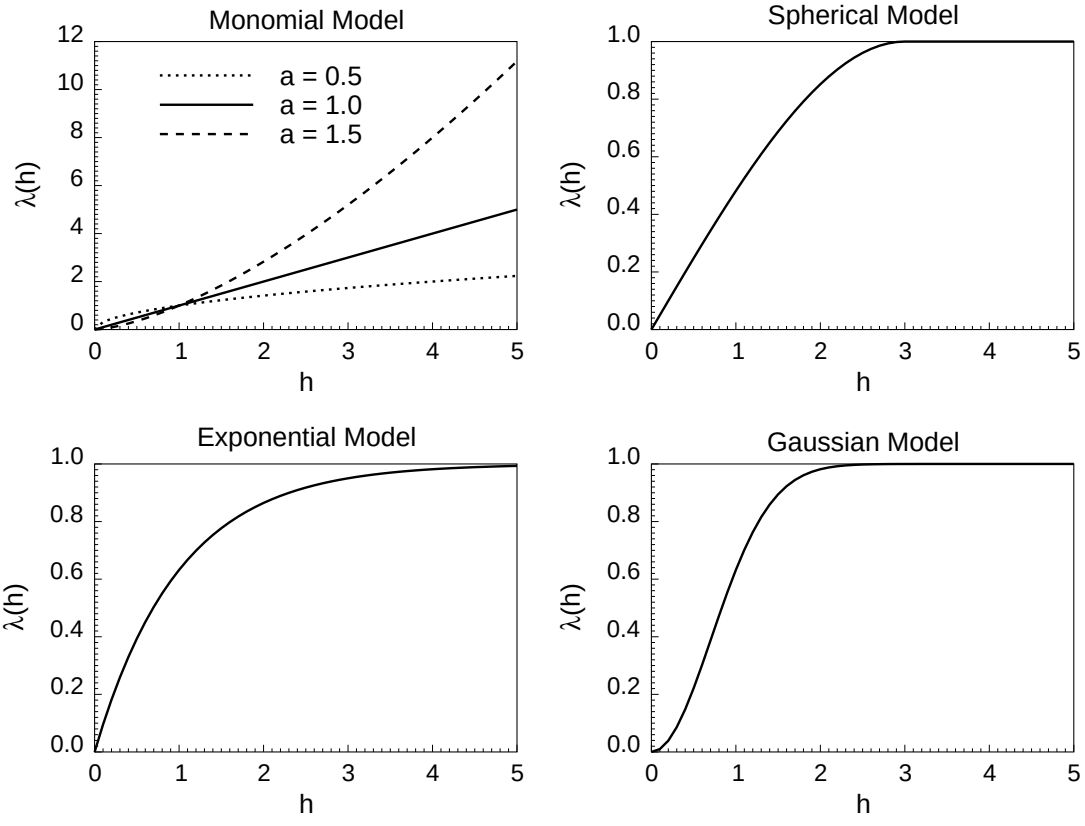

Figure 4.1: Graphs of Semivariogram Models in Widest Use

The Kriging System

In some cases, the hypothesis of constant mean is not reasonable, the intrinsic hypothesis (4.11) must be extended. It is assumed that the expectation of the random function $Z(\vec{x})$ is a function $m(\vec{x}) = E[Z(\vec{x})]$ that varies slowly relative to the working scale. This function $m(\vec{x})$ is called the drift, it can be expressed as:

$$m(\vec{x}) = \sum_{l=1}^{k} a_l \cdot f^l(\vec{x}) \tag{4.15}$$

where the f^l are given basic functions such as the polynomials shown in Equation 4.2 or 4.3 and the a_l are unknown coefficients. This model also includes the case without drift (i.e. with a constant expectation) where the only basic function is $f^0 = 1$. The second part of the intrinsic hypothesis is maintained.

The values of the variable $z(\vec{x})$ under study are known at n experimental points $\vec{x}_1, \vec{x}_2, \ldots, \vec{x}_n$, and the problem is to estimate the value of z_0 at a new point $\vec{x}_0$. The estimate of $z_0 = z(\vec{x}_0)$ can be expressed as a linear functional of the known variable values:

$$z_0^* = \sum_{i=1}^{n} \lambda_i \cdot z(\vec{x}_i) \tag{4.16}$$

The problem is to find the set of weights λ_i which give the best possible estimation.

From the probabilistic interpretation of the regionalized variable $z(\vec{x})$, z_0^* can be considered as a realization of a regionalized variable:

$$Z_0^* = \sum_{i=1}^{n} \lambda_i \cdot Z(\vec{x}_i) \tag{4.17}$$

and the weights λ_i must be chosen to satisfy that Z_0^* is

(1) unbiased (no systematic over or under estimation) and

(2) optimal (with minimum mean squared error).

These conditions can be expressed as:

$$\begin{cases} E[Z_0^* - Z_0] = 0 \\ var[Z_0^* - Z_0] \text{ minimum} \end{cases} \tag{4.18}$$

Taking account of the drift, the unbiasedness condition in Equation 4.18 becomes:

$$\sum_{i=1}^{n} \lambda_i (\sum_{l=1}^{k} a_l f_i^l) - \sum_{l=1}^{k} a_l f_0^l = 0 \tag{4.19}$$

where

$$f_i^l = f^l(\vec{x}_i) \quad \text{and} \quad f_0^l = f^l(\vec{x}_0)$$

Since Equation 4.19 must be fulfilled for arbitrary values of a_l, then:

$$\sum_{i=1}^{n} \lambda_i f_i^l - f_0^l = 0 \quad l = 1, \ldots, k \tag{4.20}$$

Taking $f^1 \equiv 1$, the first constraint is:

$$\sum_{i=1}^{n} \lambda_i - 1 = 0 \tag{4.21}$$

From Equation 4.21, the error $(Z_0^* - Z_0)$ can be expressed as a linear combination of the increments of the random function Z:

$$Z_0^* - Z_0 = \sum_{i=1}^{n} \lambda_i Z(\vec{x}_i) - Z_0 = \sum_{i=1}^{n} \lambda (Z(\vec{x}_i) - Z_0) \tag{4.22}$$

Thus, after a few calculations, the variance of this error can be expressed in terms of the semivariogram:

$$var[Z_0^* - Z_0] = -\sum_{i=1}^{n} \sum_{j=1}^{n} \lambda_i \lambda_j \gamma_{ij} + 2 \sum_{i=1}^{n} \lambda_i \gamma_{i0} \tag{4.23}$$

where

$$\gamma_{ij} = \gamma(\vec{x}_i - \vec{x}_j) \quad \text{and} \quad \gamma_{i0} = \gamma(\vec{x}_i - \vec{x}_0)$$

Then, by minimizing the variance in λ_i (Equation 4.23) under the k constraints (Equation 4.20), the following linear system, called the 'Kriging system' is obtained:

$$\begin{cases} \sum_{j=1}^{n} \lambda_j \gamma_{ij} + \sum_{l=1}^{k} \mu_l f_i^l = \gamma_{i0} & i = 1, \ldots, n \\ \sum_{j=1}^{n} \lambda_j f_j^l = f_0^l & l = 1, \ldots, k \end{cases} \tag{4.24}$$

The solution of this system of $n + k$ equations gives the n weights λ_i and the k Lagrange multipliers μ_l. The Kriging estimate can be obtained by substituting the weights λ_i into Equation 4.16:

$$z_0^* = \sum_{i=1}^{n} \lambda_i \cdot z(\vec{x}_i)$$

The Kriging variance can also be obtained as:

$$\sigma^2 = var[Z_0^* - Z_0] = \sum_{i=1}^{n} \lambda_i \gamma_{i0} + \sum_{l=1}^{k} \mu_l f_0^l \tag{4.25}$$

For the generation of new realizations, there are as many Kriging systems as there are points to be estimated. However, if all the available sample data points are taken into account every time, it appears that only the right-hand side changes and the matrix of the Kriging system needs to be inverted only once. In some cases, the number of experimental points is too large to allow such a procedure, then only those 10 to 20 data points situated in the immediate neighborhood of the point to be estimated will be utilized. In this study, the complete neighborhood was taken into account.

The Properties of Kriging

The Kriging system actually takes into account:

(1) the distances between the estimated point and the sample data points.

(2) the distances between prior data points themselves.

(3) the structure of the variable through the semivariogram γ.

When $\vec{x}_0$ coincides with a data point $\vec{x}_i$, the solution of the system is $\lambda_i = 1, \lambda_j = 0$ for $j \neq i$. It is verified that $Z_0^* = Z(\vec{x}_i)$ and $\sigma^2 = 0$. Therefore, Kriging is an exact interpolator.

The Kriging system and the Kriging variance only apply the structure and the geometric configuration of the sample data points and the point to be estimated. They do not depend upon the real values of the measurements $z(\vec{x}_i)$.

Section 5

Application in Field Development Scheduling

5.1 Problem Description

This project concerns the optimal scheduling of the development of an offshore oil field. When the production of the active reservoir declines, the maximum capacity of the installed facilities is no longer satisfied and the operating company wishes to supply additional production from another zone/reservoir in order to achieve full capacity. A simple model was developed to investigate the sensitivity of the response to the scheduling parameters. Taking the economic factors into account, the total net present value during the time period under study was selected as the objective function. A simplified problem was considered with two decision parameters t_{i2} and dt, namely: the starting time of the development of the second field and the time required to produce oil in the new field at full facility capacity Q_{max}. The model is shown in Figure 5.1 where Q_1 represents the total production in the first active field and Q_2 represents the total production in the second field. The objective of this project is to search for the optimal strategies maximizing the net present value subject to the constraint of facility capacity.

In this study, the optimal strategies were searched among 1225 cases corresponding

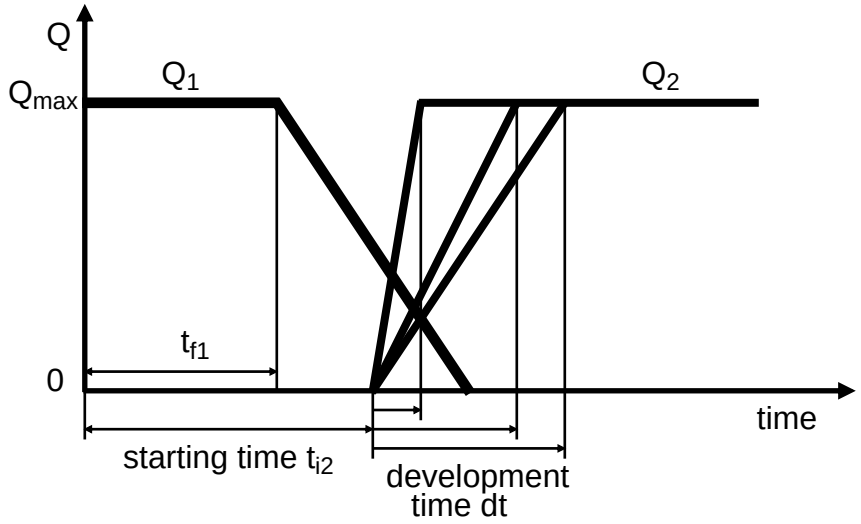

Figure 5.1: A Simple Model for Field Development Scheduling

to different values of the two decision variables. Two methods were applied to solve this problem. The first one was to run the simulator directly for each case to obtain the exhaustive data set. The second method was to perform only 35 simulations to obtain a prior data set and then to apply multivariate interpolation techniques to generate the 1225 realizations. The results were compared and the feasibility and efficiency of the second method were investigated.

5.2 Simulation Model

One important step in the optimization procedure described in section 2 is the oil production simulation. For this project, the simulation was achieved by an economic model. It was assumed that the development system consists of an existing production platform with 10 wells already paid. At a specific time t_{f1} the oil production in the first reservoir begins to decline at a constant rate. At some later time t_{i2} the second reservoir starts to produce oil at such a rate that the maximum capacity of the platform is not exceeded. Based on this constraint, the number of wells required to produce at this flow rate can be evaluated. Two rigs on a permanent platform

are applied to drill the wells before they begin producing oil at certain time. Each rig takes about 4 months to drill one well. Therefore, the rigs have to be scheduled properly in order to drill all the producing wells in time.

The permanent platform is contracted two years before the first well to be drilled. The payment is 30% down and 70% over 23 months. The facilities are also contracted one year before the start of production and are paid in the same way. The time period under investigation is 360 months(3 years). The other parameters applied in the cost function evaluation are shown in Table 5.1.

Table 5.1: The Parameters of Cost Function

Names	**Values**	**Units**
Number of platform	1	
Number of rigs	2	
Number of wells	10	
Rig capacity	4	months/well
Maximum total flow rate(facility capacity)	15000	bbl/day
Maximum oil production rate per well	1500	bbl/day
Production decline starting time t_{f1}	144	months
Declining time down to $Q_1 = 0$	60	months
Oil price	30	US$/bbl
Platform cost	100,000,000	US$/15 wells
Rig cost	600,000	US$/month
Mobilization cost	1,000,000	US$/month
Offshore well drilling cost	1,400,000	US$/month
Facility initial cost	400	US$/month
Insurance cost	12,500	US$/month
Operational cost	5,000	US$/well/month
Fluids handling cost	1	US$/bbl
Discount rate	10%	per year
Overhead factor	0.15	
Tax and Transportation factor(T factor)	0.0435	

The cost function is defined as:

$$cop \;=\; (operational\ cost\ per\ well) * (number\ of\ wells) * (time\ interval)\ \ (US\$)$$

$$ch = (fluids\ handling\ cost) * (oil\ rate\ of\ the\ period) * (time\ interval)\ \ (US\$)$$
$$isl = (Tfactor) * (oil\ price) * (oil\ rate\ of\ the\ period) * (time\ interval)\ \ (US\$)$$

The production cost and income may be expressed as:

$$\begin{aligned} Production\_Cost &= (1 + overhead\_factor) * (cop + ch) \\ &+ insurance\_cost + isl \quad (US\$) \end{aligned} \tag{5.1}$$
$$\begin{aligned} Production\_Income &= (oil\_rate\ of\ the\ period) * time\_interval \\ &* oil\_price \quad (US\$) \end{aligned} \tag{5.2}$$

Then the objective function net present value can be obtained.

$$\begin{aligned} Net\ Present\ Value &= \sum_{\triangle t} \{(Production\_Income - Production\_Cost) \\ &* (1 + discount\_rate)^{(reference\_time - time)}\} \quad (US\$) \end{aligned} \tag{5.3}$$

The simulation results for two particular cases are shown here. The drilling and production schedules from the time when the first field production begins to decline to the time when the first field stops producing any oil are shown in Table 5.2 and 5.3. The corresponding curves of production income, production cost and net present value versus time are shown in Figure 5.2 and 5.3.

Discretizing the Parameter Domain Investigated

The following $35 \times 35 = 1225$ combinations of the two decision variables were investigated to predict the optimal strategy.

- t_{i2} varying from 0 to 204 months by 6 months each grid
- dt varying from 5 to 175 months by 5 months each grid

Since the simulation model used in this project is not complicated and takes only seconds to perform one simulation run, the exhaustive 1225 cases were investigated by the simulator to provide the reference data. The net present value surface and the contour map over the two decision variable domain are shown in Figure 5.4.

It can be noticed that the starting time t_{i2} is the most sensitive parameter to the object function value, and the optimal solution is starting time $t_{i2} = 144$ months and development time $dt = 60$ months with the maximum net present value 1391 million US dollars.

To apply the multivariate interpolation algorithms, a limited number of sample data was chosen to perform the simulation. The distribution of the 35 prior data points obtained by uniform design technique was shown earlier in Figure 3.1. The grids show the distribution of the exhaustive data points and the black dots represent the sample data points. Since there was no information showing any priorities of any particular points, the sample data points were distributed uniformly in the parameter domain being investigated and the weighting factor was chosen to be unit for each sample point when the Least Squares interpolation was performed.

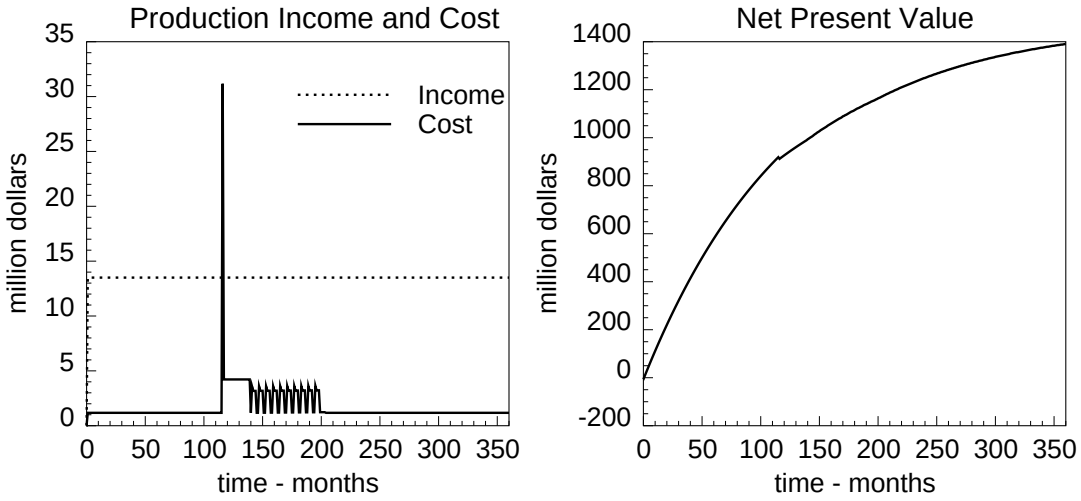

Figure 5.2: Simulation Results for the Optimal Case

$t_{i2} = 144$ months, $dt = 60$ months, $NPV = \$1391 * 10^6$

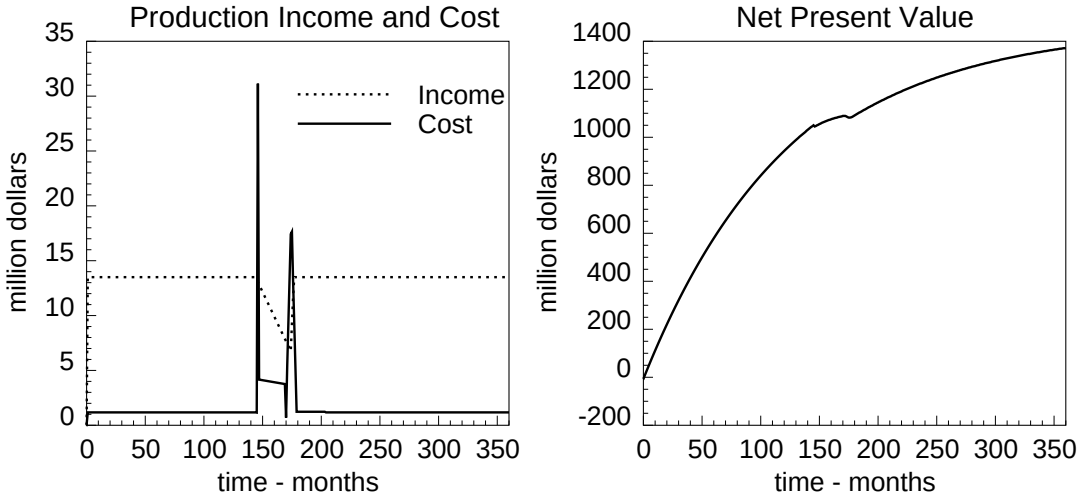

Figure 5.3: Simulation Results for One Case

$t_{i2} = 174$ months, $dt = 5$ months, $NPV = \$1372 * 10^6$

Table 5.2: The Optimal Drilling and Production Schedules

$t_{i2} = 144$ months, $dt = 60$ months

time	nw1	nw2	rig1	rig2	q1	q2	qt	income	cost
144	10	0	0	0	1.500E+04	0.000E+00	1.500E+04	1.350E+01	3.175E+00
145	10	1	0	1	1.475E+04	2.500E+02	1.500E+04	1.350E+01	1.181E+00
146	10	1	0	0	1.450E+04	5.000E+02	1.500E+04	1.350E+01	1.181E+00
147	10	1	0	0	1.425E+04	7.500E+02	1.500E+04	1.350E+01	3.581E+00
149	10	1	0	0	1.375E+04	1.250E+03	1.500E+04	1.350E+01	3.181E+00
150	10	1	0	0	1.350E+04	1.500E+03	1.500E+04	1.350E+01	3.181E+00
151	10	2	0	1	1.325E+04	1.750E+03	1.500E+04	1.350E+01	1.186E+00
152	10	2	0	0	1.300E+04	2.000E+03	1.500E+04	1.350E+01	1.186E+00
153	10	2	0	0	1.275E+04	2.250E+03	1.500E+04	1.350E+01	3.586E+00
155	10	2	0	0	1.225E+04	2.750E+03	1.500E+04	1.350E+01	3.186E+00
156	10	2	0	0	1.200E+04	3.000E+03	1.500E+04	1.350E+01	3.186E+00
157	10	3	0	1	1.175E+04	3.250E+03	1.500E+04	1.350E+01	1.192E+00
158	10	3	0	0	1.150E+04	3.500E+03	1.500E+04	1.350E+01	1.192E+00
159	10	3	0	0	1.125E+04	3.750E+03	1.500E+04	1.350E+01	3.592E+00
161	10	3	0	0	1.075E+04	4.250E+03	1.500E+04	1.350E+01	3.192E+00
162	10	3	0	0	1.050E+04	4.500E+03	1.500E+04	1.350E+01	3.192E+00
163	10	4	0	1	1.025E+04	4.750E+03	1.500E+04	1.350E+01	1.198E+00
164	10	4	0	0	1.000E+04	5.000E+03	1.500E+04	1.350E+01	1.198E+00
165	10	4	0	0	9.750E+03	5.250E+03	1.500E+04	1.350E+01	3.598E+00
167	10	4	0	0	9.250E+03	5.750E+03	1.500E+04	1.350E+01	3.198E+00
168	10	4	0	0	9.000E+03	6.000E+03	1.500E+04	1.350E+01	3.198E+00
169	10	5	0	1	8.750E+03	6.250E+03	1.500E+04	1.350E+01	1.204E+00
170	10	5	0	0	8.500E+03	6.500E+03	1.500E+04	1.350E+01	1.204E+00
171	10	5	0	0	8.250E+03	6.750E+03	1.500E+04	1.350E+01	3.604E+00
173	10	5	0	0	7.750E+03	7.250E+03	1.500E+04	1.350E+01	3.204E+00
174	10	5	0	0	7.500E+03	7.500E+03	1.500E+04	1.350E+01	3.204E+00
175	10	6	0	1	7.250E+03	7.750E+03	1.500E+04	1.350E+01	1.209E+00
176	10	6	0	0	7.000E+03	8.000E+03	1.500E+04	1.350E+01	1.209E+00
177	10	6	0	0	6.750E+03	8.250E+03	1.500E+04	1.350E+01	3.609E+00
179	10	6	0	0	6.250E+03	8.750E+03	1.500E+04	1.350E+01	3.209E+00
180	10	6	0	0	6.000E+03	9.000E+03	1.500E+04	1.350E+01	3.209E+00
181	10	7	0	1	5.750E+03	9.250E+03	1.500E+04	1.350E+01	1.215E+00
182	10	7	0	0	5.500E+03	9.500E+03	1.500E+04	1.350E+01	1.215E+00
183	10	7	0	0	5.250E+03	9.750E+03	1.500E+04	1.350E+01	3.615E+00
185	10	7	0	0	4.750E+03	1.025E+04	1.500E+04	1.350E+01	3.215E+00
186	10	7	0	0	4.500E+03	1.050E+04	1.500E+04	1.350E+01	3.215E+00
187	10	8	0	1	4.250E+03	1.075E+04	1.500E+04	1.350E+01	1.221E+00
188	10	8	0	0	4.000E+03	1.100E+04	1.500E+04	1.350E+01	1.221E+00
189	10	8	0	0	3.750E+03	1.125E+04	1.500E+04	1.350E+01	3.621E+00
191	10	8	0	0	3.250E+03	1.175E+04	1.500E+04	1.350E+01	3.221E+00
192	10	8	0	0	3.000E+03	1.200E+04	1.500E+04	1.350E+01	3.221E+00
193	10	9	0	1	2.750E+03	1.225E+04	1.500E+04	1.350E+01	1.227E+00
194	10	9	0	0	2.500E+03	1.250E+04	1.500E+04	1.350E+01	1.227E+00
195	10	9	0	0	2.250E+03	1.275E+04	1.500E+04	1.350E+01	3.627E+00
197	10	9	0	0	1.750E+03	1.325E+04	1.500E+04	1.350E+01	3.227E+00
198	10	9	0	0	1.500E+03	1.350E+04	1.500E+04	1.350E+01	3.227E+00
199	10	10	0	1	1.250E+03	1.375E+04	1.500E+04	1.350E+01	1.232E+00
200	10	10	0	0	1.000E+03	1.400E+04	1.500E+04	1.350E+01	1.232E+00
201	10	10	0	0	7.500E+02	1.425E+04	1.500E+04	1.350E+01	1.232E+00
203	10	10	0	0	2.500E+02	1.475E+04	1.500E+04	1.350E+01	1.232E+00
204	0	10	0	0	0.000E+00	1.500E+04	1.500E+04	1.350E+01	1.175E+00

nw_1, nw_2 - number of active producing wells in the two fields;
rig_1, rig_2 - number of extra wells which have been drilled by the two rigs and are ready to produce oil at that time;
q_1, q_2, q_t (bbl/day) - the two fields production rates and the total amount;
time - months; Production Income, Cost - million dollars.

Table 5.3: The Drilling and Production Schedules for One Case

$t_{i2} = 174$ months, $dt = 5$ months

time	nw1	nw2	rig1	rig2	q1	q2	qt	income	cost
144	10	0	0	0	1.500E+04	0.000E+00	1.500E+04	1.350E+01	1.175E+00
146	10	0	0	0	1.450E+04	0.000E+00	1.450E+04	1.305E+01	3.114E+01
147	10	0	0	0	1.425E+04	0.000E+00	1.425E+04	1.283E+01	4.163E+00
148	10	0	0	0	1.400E+04	0.000E+00	1.400E+04	1.260E+01	4.145E+00
149	10	0	0	0	1.375E+04	0.000E+00	1.375E+04	1.238E+01	4.126E+00
151	10	0	0	0	1.325E+04	0.000E+00	1.325E+04	1.193E+01	4.089E+00
152	10	0	0	0	1.300E+04	0.000E+00	1.300E+04	1.170E+01	4.071E+00
153	10	0	0	0	1.275E+04	0.000E+00	1.275E+04	1.148E+01	4.053E+00
154	10	0	0	0	1.250E+04	0.000E+00	1.250E+04	1.125E+01	4.034E+00
156	10	0	0	0	1.200E+04	0.000E+00	1.200E+04	1.080E+01	3.997E+00
157	10	0	0	0	1.175E+04	0.000E+00	1.175E+04	1.058E+01	3.979E+00
158	10	0	0	0	1.150E+04	0.000E+00	1.150E+04	1.035E+01	3.960E+00
159	10	0	0	0	1.125E+04	0.000E+00	1.125E+04	1.013E+01	3.942E+00
161	10	0	0	0	1.075E+04	0.000E+00	1.075E+04	9.675E+00	3.905E+00
162	10	0	0	0	1.050E+04	0.000E+00	1.050E+04	9.450E+00	3.887E+00
163	10	0	0	0	1.025E+04	0.000E+00	1.025E+04	9.225E+00	3.868E+00
164	10	0	0	0	1.000E+04	0.000E+00	1.000E+04	9.000E+00	3.850E+00
166	10	0	0	0	9.500E+03	0.000E+00	9.500E+03	8.550E+00	3.813E+00
167	10	0	0	0	9.250E+03	0.000E+00	9.250E+03	8.325E+00	3.795E+00
168	10	0	0	0	9.000E+03	0.000E+00	9.000E+03	8.100E+00	3.776E+00
169	10	0	0	0	8.750E+03	0.000E+00	8.750E+03	7.875E+00	3.758E+00
171	10	0	0	0	8.250E+03	0.000E+00	8.250E+03	7.425E+00	5.478E+00
172	10	0	0	0	8.000E+03	0.000E+00	8.000E+03	7.200E+00	9.459E+00
173	10	0	0	0	7.750E+03	0.000E+00	7.750E+03	6.975E+00	1.344E+01
174	10	0	0	0	7.500E+03	0.000E+00	7.500E+03	6.750E+00	1.742E+01
175	10	2	1	1	7.250E+03	3.000E+03	1.025E+04	9.225E+00	1.764E+01
176	10	4	1	1	7.000E+03	6.000E+03	1.300E+04	1.170E+01	1.305E+01
177	10	6	1	1	6.750E+03	9.000E+03	1.500E+04	1.350E+01	9.209E+00
178	10	8	1	1	6.500E+03	1.200E+04	1.500E+04	1.350E+01	5.221E+00
179	10	10	1	1	6.250E+03	1.500E+04	1.500E+04	1.350E+01	1.232E+00
180	10	10	0	0	6.000E+03	1.500E+04	1.500E+04	1.350E+01	1.232E+00
181	10	10	0	0	5.750E+03	1.500E+04	1.500E+04	1.350E+01	1.232E+00
182	10	10	0	0	5.500E+03	1.500E+04	1.500E+04	1.350E+01	1.232E+00
183	10	10	0	0	5.250E+03	1.500E+04	1.500E+04	1.350E+01	1.232E+00
184	10	10	0	0	5.000E+03	1.500E+04	1.500E+04	1.350E+01	1.232E+00
186	10	10	0	0	4.500E+03	1.500E+04	1.500E+04	1.350E+01	1.232E+00
187	10	10	0	0	4.250E+03	1.500E+04	1.500E+04	1.350E+01	1.232E+00
188	10	10	0	0	4.000E+03	1.500E+04	1.500E+04	1.350E+01	1.232E+00
189	10	10	0	0	3.750E+03	1.500E+04	1.500E+04	1.350E+01	1.232E+00
191	10	10	0	0	3.250E+03	1.500E+04	1.500E+04	1.350E+01	1.232E+00
192	10	10	0	0	3.000E+03	1.500E+04	1.500E+04	1.350E+01	1.232E+00
193	10	10	0	0	2.750E+03	1.500E+04	1.500E+04	1.350E+01	1.232E+00
194	10	10	0	0	2.500E+03	1.500E+04	1.500E+04	1.350E+01	1.232E+00
196	10	10	0	0	2.000E+03	1.500E+04	1.500E+04	1.350E+01	1.232E+00
197	10	10	0	0	1.750E+03	1.500E+04	1.500E+04	1.350E+01	1.232E+00
198	10	10	0	0	1.500E+03	1.500E+04	1.500E+04	1.350E+01	1.232E+00
199	10	10	0	0	1.250E+03	1.500E+04	1.500E+04	1.350E+01	1.232E+00
201	10	10	0	0	7.500E+02	1.500E+04	1.500E+04	1.350E+01	1.232E+00
202	10	10	0	0	5.000E+02	1.500E+04	1.500E+04	1.350E+01	1.232E+00
203	10	10	0	0	2.500E+02	1.500E+04	1.500E+04	1.350E+01	1.232E+00
204	0	10	0	0	0.000E+00	1.500E+04	1.500E+04	1.350E+01	1.175E+00

nw_1, nw_2 - number of active producing wells in the two fields;
rig_1, rig_2 - number of extra wells which have been drilled by the two rigs and are ready to produce oil at that time;
q_1, q_2, q_t (bbl/day) - the two fields production rates and the total amount;
time - months; Production Income, Cost - million dollars.

5.3 Interpolation Results

Applying multivariate interpolation algorithms, the prior set with 35 sample data points was interpolated to evaluate the objective function values at the 1225 grid points in the decision variable domain. When the Least Squares algorithm was applied, the highest polynomial degree was chosen as 4 to obtain a smooth interpolation surface with reasonably small residuals. When the Kriging algorithm was applied, the highest polynomial degree for the drift (variable mean) was chosen as 2. The net present value surfaces and contour maps of the new realizations obtained by the Least Squares and the Kriging algorithms are shown in Figure 5.5 and 5.6. The optimal solutions for the reference data, sample data, Least Squares realizations and Kriging realizations are compared in Table 5.4.

Table 5.4: Optimal Solutions for Different Data Set

Data Set	t_{i2} (months)	dt (months)	NPV (million US$)
35 Sample Simulation Data	144	50	1390
1225 Least Squares realizations	144	05	1388
1225 Kriging realizations	138	45	1390
1225 Reference Simulation Data	144	60	1391

In general, the sample data may not include the optimal point as in this example. Whether the multivariate interpolation can generate the objective function surface over all the decision variable domain under study from the general limited sample data is a good test of the feasibility and accuracy of this approach. Comparing Figure 5.5 and 5.6 with Figure 5.4, both of the Least Squares and Kriging algorithms capture the basic shape of the reference objective function surface. This is important in a global optimum searching process. The net present value contour maps also show that the optimal region of the realizations is very close to that of the reference data. Even though the exact optimal point has not been reached, the intermediate optimal region has been found with only 35 simulations. The objective values calculated by interpolation may not represent the real values, this is avoidable by performing the simulation to obtain the accurate objective value after reaching the optimal point.

The next step is to refine the grids in the proposed optimal region and run the simulator again for the additional data set. Then the improved optimum can be obtained from these data. If it is necessary, these data points can be added to the previous sample data set and the interpolation process is repeated. Any optimal solutions obtained by interpolation techniques need to be validated by running the simulator at those points again to obtain the accurate objective function values.

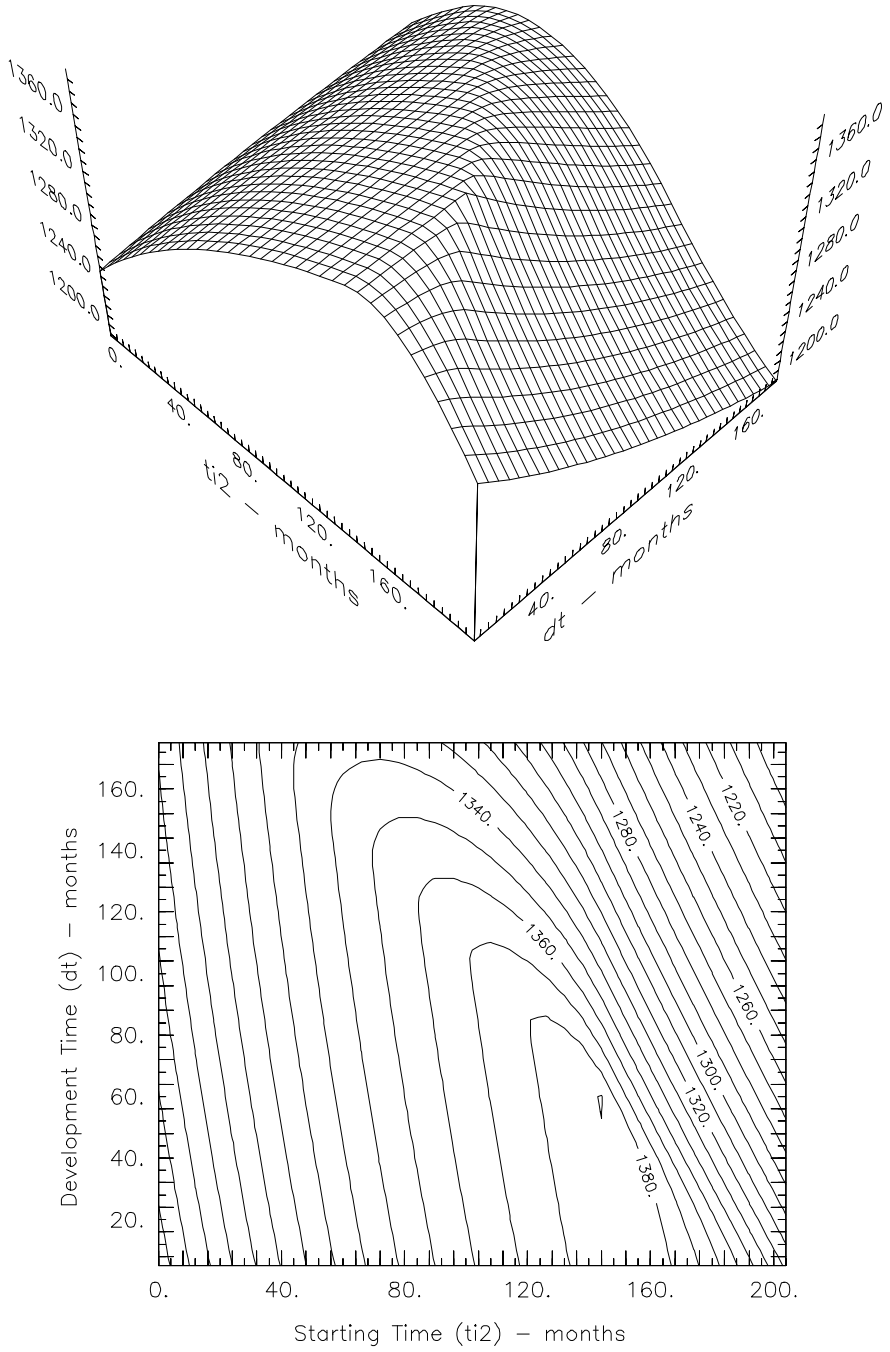

Figure 5.4: Reference NPV Surface and Contour Map

Optimal Solution: $t_{i2} = 144$ months, $dt = 60$ months, $NPV = \$1391 * 10^6$

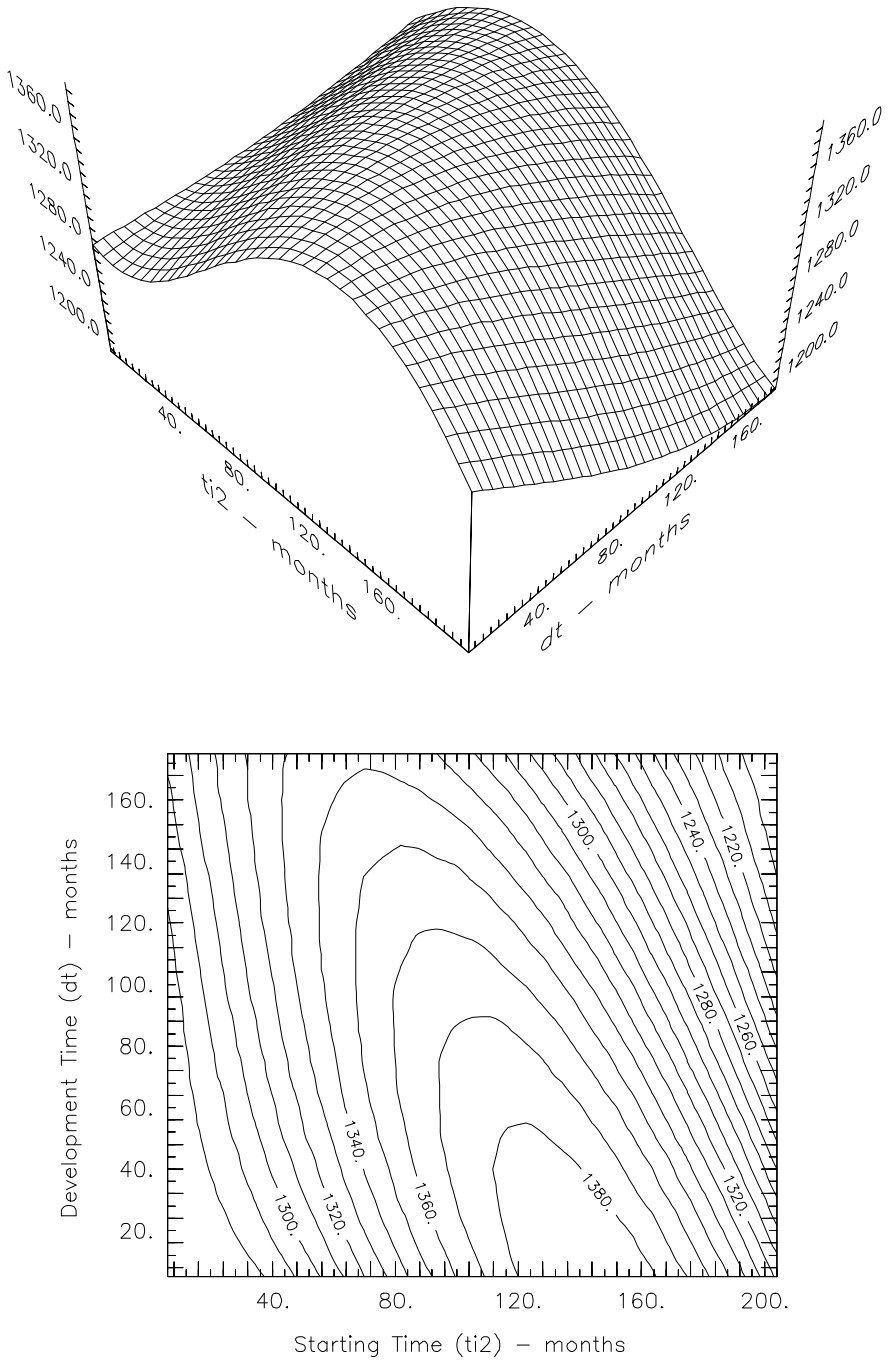

Figure 5.5: NPV Surface and Contour Map Obtained by Least Squares Interpolation

Optimal Solution: $t_{i2} = 144$ months, $dt = 5$ months, $NPV = \$1388 * 10^6$

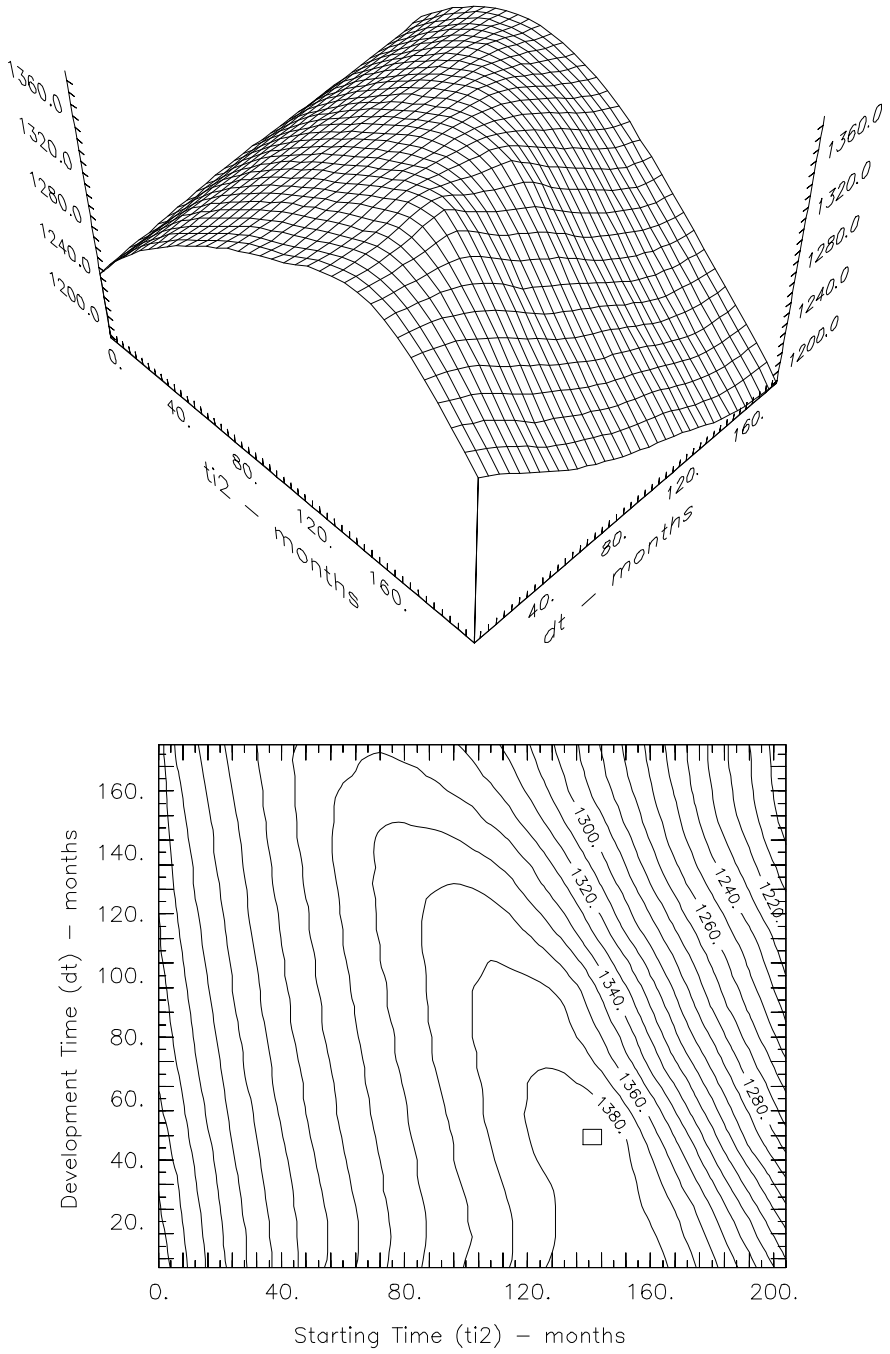

Figure 5.6: NPV Surface and Contour Map Obtained by Kriging Interpolation

Optimal Solution: $t_{i2} = 138$ months, $dt = 45$ months, $NPV = \$1390 * 10^6$

5.4 Validation of Recommended Realizations

To reach the final global optimum and to validate the realizations provided by the multivariate interpolation, the simulations were performed at another $7 \times 13 = 91$ data points located in the intermediate optimal region. The grids were refined as following:

- t_{i2} varying from 128 to 152 months by 4 months each grid
- dt varying from 5 to 65 months by 5 months each grid

Then the optimal point among these data was obtained as:

$$t_{i2} = 144 \text{ months}, \quad dt = 60 \text{ months}, \quad NPV_{max} = \$1391 * 10^6$$

This is the same optimal strategy as the one obtained from the 1225 reference data.

By analyzing the problem (Figure 5.1) directly, it can also be demonstrated that the above optimal solution is reasonable. If the starting development time of the second field is earlier than the 144th month when the production of the first field begins to decline, the wells in the second field have been drilled but can not produce oil because of the constraint of the maximum facility capacity. This means investment has been made to develop the second field but there is no income from it for a certain period. If the starting time is later than the 144th month, then there is certain period that the two fields are not producing at the full capacity. The influence of the development time is similar. If it is too short, both rigs have to be applied to drill all the wells required at the same time and their cost has to be paid in short time, but some wells in the second field may not produce any oil because of the capacity constraint. If it is longer than 60 months, the second field has not been completely developed to produce at full capacity when the first field is drained. All these cases are not desirable. The only optimal strategy to obtain the maximum net present value or highest net profit is to start developing the second field at the 144th month and to spend 60 months to fully develop it.

5.5 Summary

In this field development scheduling optimization project, applying the multivariate interpolation techniques, the total number of simulations required to obtain the global optimum was reduced from 1225 to $35+91 = 126$, and the optimal solution is reliable as represented in Table 5.5. This demonstrates the efficiency and feasibility of this approach.

Table 5.5: Comparison of Two Optimization Approaches

Method	Number of Simulations	Optimal Solution		
		t_{i2} (months)	dt (months)	NPV (million US$)
Simulation	1225	144	60	1391
Simulation + Interpolation	126	144	60	1391

Section 6

Application in Waterflooding Project

6.1 Problem Description

This project concerns the optimal well placement and optimal injection time control in a waterflooded field. These factors are very important for performing the waterflooding efficiently in order to obtain maximum oil production while reducing the water treatment cost after breakthrough. It was assumed that the reservoir is homogeneous, and the mobility ratio of water and oil is unit. As shown in Figure 6.1, a pattern is composed of four producers and one injector. Both repeated patterns and isolated patterns were investigated in this study. For a given reservoir with certain size, the area of each pattern can be estimated, therefore, it was provided as a constant area. In this project, the total production rate was considered equal to the injection rate, but each producer could have different flow rate. Taking account of the economic factors associated with oil and water flow rates: the sale income of oil, the operation costs of water injection and oil production and the treatment cost of water after breakthrough, the net present value during the production period was selected as the objective function. Figure 6.1 shows the four decision variables chosen to be optimized: the injector location (c, d), the pattern shape factor b/a and the total injection time t_{opt}. The parameter t_{opt} represents the optimal total injection time with

maximum net present value for each particular pattern, which means the injector location and pattern shape are fixed. The investigating range of of each parameter can be determined from practical analysis and experiences.

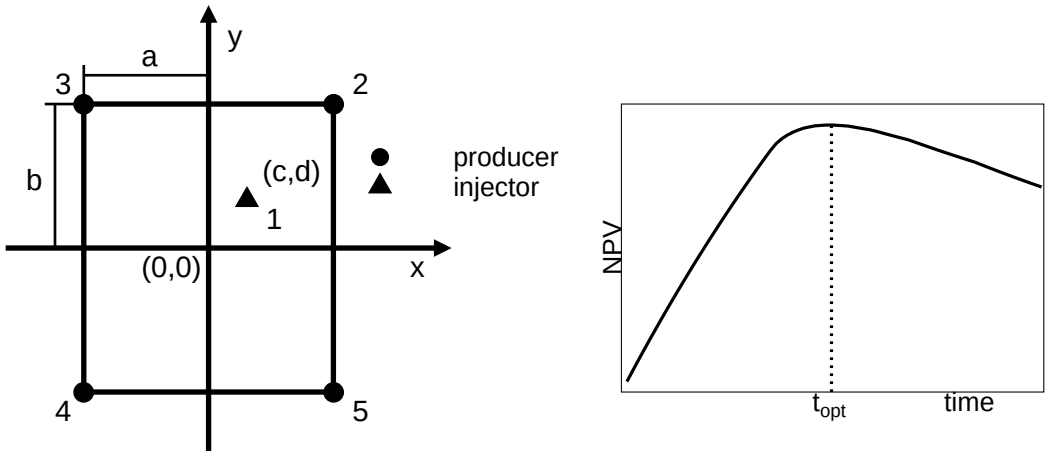

Figure 6.1: Waterflooding Optimization Problem

In summary, the problem can be described as:

- Objective Function:

$$NPV = \sum_{time} [(profit - cost) \times (1 + discount\ rate)^{(reference\ time - time)}] \quad (6.1)$$

- Decision Variables:
  - injector location: (c, d) $\quad (-a < c < a;\ -b < d < b)$
  - pattern shape: b/a $\quad (1 \leq b/a \leq 4)$
  - injection time: t_{opt} $\quad (0 \leq t_{opt} \leq 2$ PV$)$
- Reservoir Conditions:
  - homogeneous
  - mobility ratio $= 1$

The optimal strategies were searched within the investigating range of each decision parameter. After discretizing, the number of total combinations of the four parameters is a big number. Since it requires lots of computer time to perform the exhaustive evaluations, only the alternative mathematical approach was applied in this study. Even though the reference data are not provided, the optimization results can be compared with the analytical solution and checked by other sources.

6.2 Simulation Model

Based on the previous definition of the objective function, all the water injection, oil and water production rates at each producer at each time step need to be calculated to obtain the curve of net present value versus time for different well placement. In order to know the details of the fluid movements within the pattern, instead of using a commercial simulator, the simulation was achieved by solving the Laplace equation and generating streamlines within the pattern. The movements of the water front were tracked and then the oil and water production rates at each producer were calculated.

The governing Laplace equation can be expressed as:

$$\nabla^2 \Phi = 0 \tag{6.2}$$

The velocity potential and stream function at location (x, y) are defined as:

$$\Phi(x,y) = \frac{1}{2\pi h} \sum_i q_i ln r_i \quad \text{velocity potential} \tag{6.3}$$

$$\Psi(x,y) = \frac{1}{2\pi h} \sum_i q_i \theta_i \quad \text{stream function} \tag{6.4}$$

where q_i is the flow rate at well i, defined as $q_i > 0$ for production and $q_i < 0$ for injection; r_i is the distance from well i to the point (x, y); θ_i is the polar angle centered at well i.

Then the particle velocity at any location (x, y) can be obtained as:

$$\begin{cases} v_x(x,y) &= -\ \frac{1}{\phi}\frac{\partial \Psi(x,y)}{\partial y} = -\frac{1}{\phi}\frac{\partial \Phi(x,y)}{\partial x} \\ v_y(x,y) &= \frac{1}{\phi}\frac{\partial \Psi(x,y)}{\partial x} = -\frac{1}{\phi}\frac{\partial \Phi(x,y)}{\partial y} \end{cases} \tag{6.5}$$

where ϕ is the porosity of the reservoir.

Along each streamline starting from the injector to the producer, the particle can be tracked by estimating the next location at each time step, and this predicted location can be corrected by applying Newton's method subject to the same stream function value.

$$\begin{aligned} x' = x + v_x \cdot dt; \quad y' = y + v_y \cdot dt \\ \Psi_j(x', y') = \Psi_j(x, y) = constant \end{aligned} \tag{6.6}$$

where j is the index of the streamline. Then the water breakthrough time t_b at each producer can be recorded, and the dimensionless breakthrough time can be calculated as:

$$W = \frac{t_b \cdot q_1}{PV}; \qquad PV = 4ab \cdot h \cdot \phi \tag{6.7}$$

where q_1 is the injection rate, h is the depth of the reservoir, and PV is the pore volume of the pattern.

The water cut at each producer can be obtained as:

$$(f_w)_l = \frac{\triangle(\Psi_{BT})_l}{q_l/h} \qquad l = 2, 3, 4, 5 \tag{6.8}$$

where $\triangle(\Psi_{BT})_l$ is the largest stream function value difference among those breakthrough streamlines, and q_l is the flow rate of the producer l.

Then the total oil and water production at each producer can be obtained, and the net present value at any time period can be calculated from Equation 6.1. The flow chart of the simulation is shown in Figure 6.2.

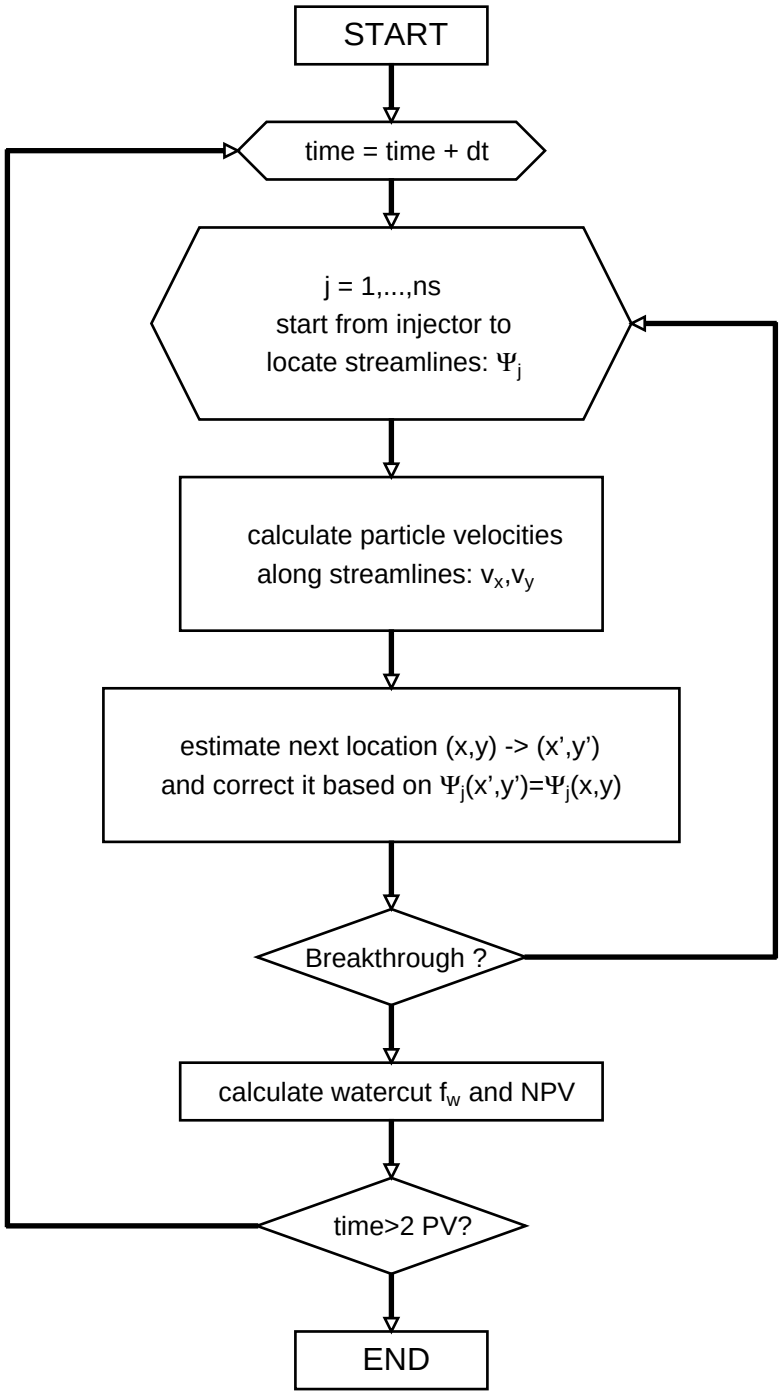

Figure 6.2: Waterflooding Simulation Flow Chart

Reliability of Simulation

To check the reliability of the simulation, the calculated breakthrough time of four different repeated patterns in a homogeneous reservoir were compared with the analytical solutions as shown in Table 6.1. The accuracy depends on the number of streamlines and number of well images applied in the simulation, the more streamlines and images used, the more accurate the results will be. Applying 120 streamlines and 40 images for each well, the results obtained are very close to the analytical solutions.

Table 6.1: Breakthrough Time of Repeated Pattern in Homogeneous Reservoir
(40 images for each well, 120 streamlines)

Pattern Shape b/a	Analytical Solutions (PV)	Simulation Results (PV)	Error (%)
1	0.71770	0.7147	0.4
2	0.78884	0.7906	0.2
3	0.85339	0.8687	1.8
4	0.88971	0.9304	4.6

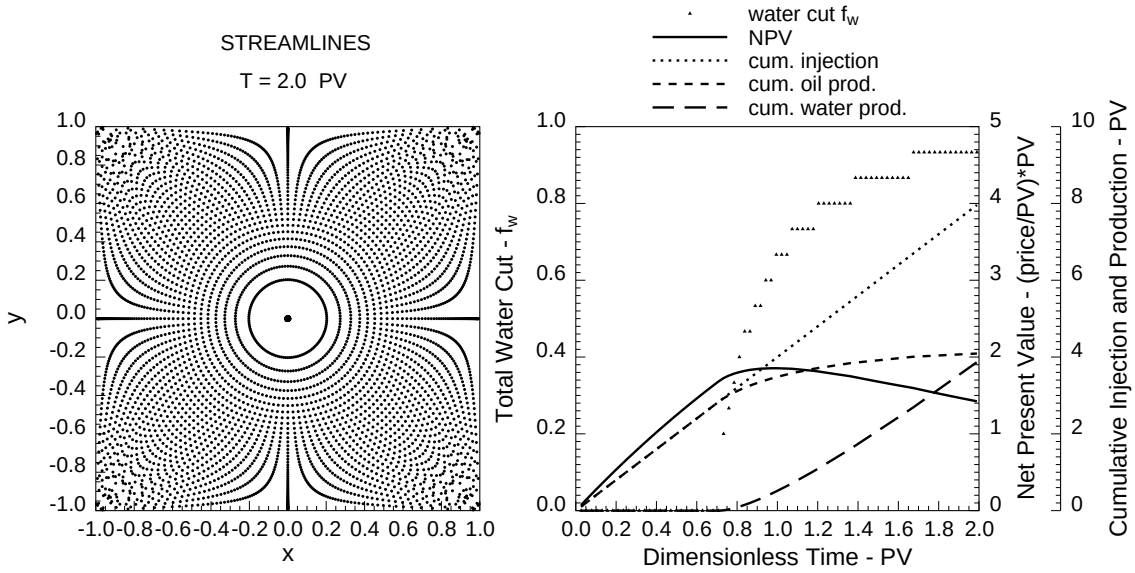

Figure 6.3: Simulation Results: b/a = 1; (c,d)=(0,0)
Repeated Pattern in Homogeneous Reservoir

Since this project concerns searching optimal strategies, only relative values and dimensionless parameters were used. If required, the absolute values may be substituted. Some simulation results are shown in Figure 6.3 and 6.4.

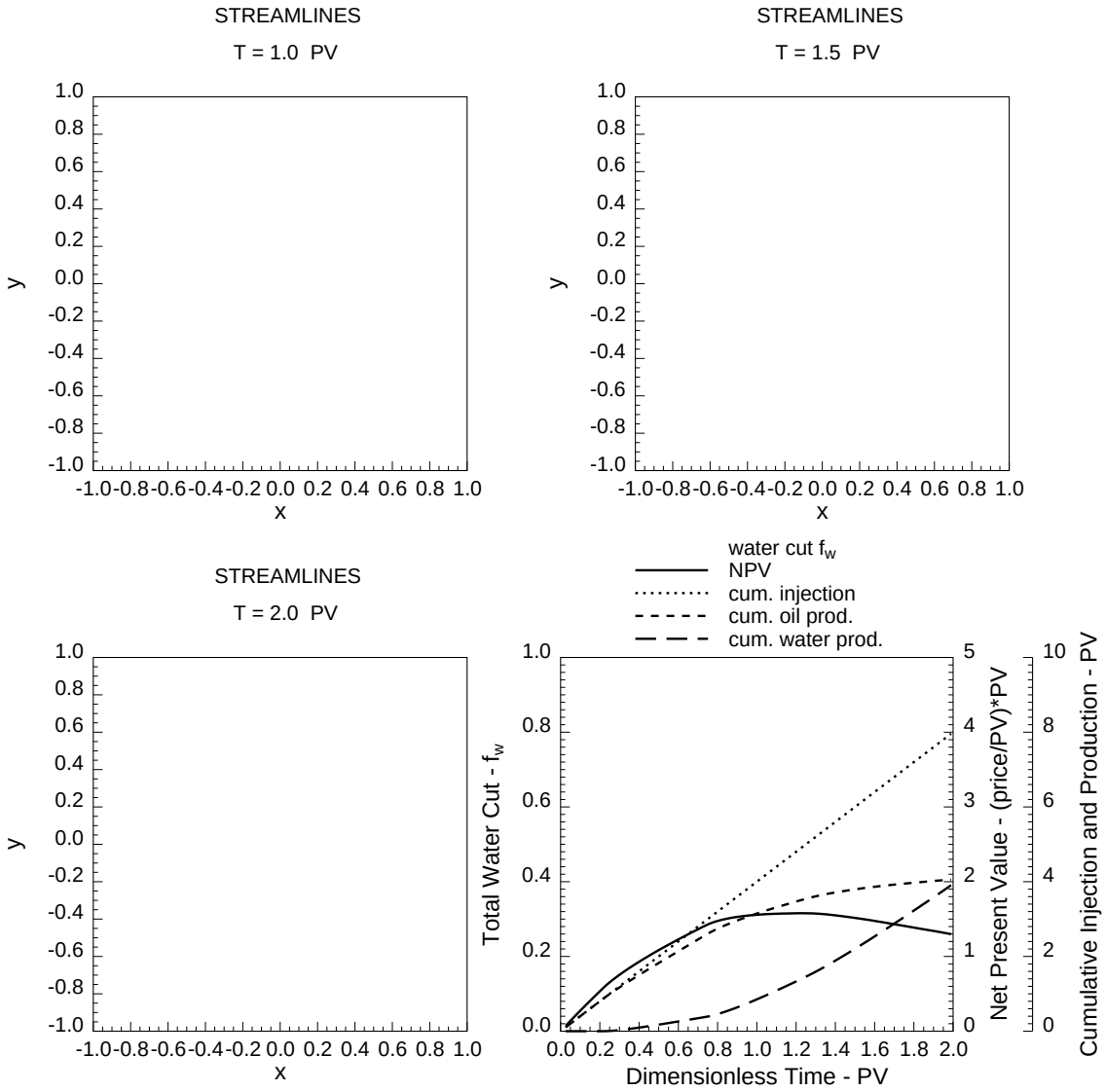

Figure 6.4: Simulation Results: b/a = 1; (c,d)=(0.45,0.45)
Repeated Pattern in Homogeneous Reservoir

Figure 6.3 shows the simulation results of a repeated 5-spot pattern with the injector

located in the center. The water broke through to the four producers simultaneously when 0.7147 PV of water were injected and the total water cut increased rapidly after breakthrough. The variations of the total water cut, the cumulative water injection, the cumulative oil and water production and the net present value versus dimensionless time are shown by the curves. When the dimensionless time was 0.9948 PV, the NPV reached the maximum value 1.8542, after that, the NPV began to decline. This shows the best time to cease operation for the biggest profit. Figure 6.4 shows the movements of the waterfront at different time steps and the relative curves for an asymmetric pattern. The breakthrough happened much earlier than the previous case and the total water cut increased gradually following the breakthrough sequence. The water broke through to the producer closest to the injector at first and broke through to the producer farthest at last. When the dimensionless time was 1.2305 PV, the NPV reached the maximum value 1.5782, after that, the NPV began to decline. In these two cases, the production rate of each well is the same. Comparing the results, the symmetric pattern predicted the biggest NPV while operating during the optimal time period.

Effects of Decision Variables

The simulation results are sensitive to all the decision variables, and different relative production rates of the wells affect the movements of the water front noticeably. Figure 6.5 shows the effect of different flow rates. The water always broke through to the producer with highest flow rate first. This shows the asymmetric behavior of the symmetric pattern. Figure 6.6 shows the effect of injector location. The curve shape of net present value versus time for repeated pattern is different from that of an isolated pattern, but in both cases, the injector location influences the NPV significantly. The maximum NPV can be obtained if the injector is located a little away from the high flow rate region. Figure 6.7 shows the effect of the pattern shape. Among the four shapes compared, the staggered line drive with the ratio $b/a = 4$ predicted the maximum NPV over all the time period under study. Overall, this demonstrates that the decision variables selected are all essential to obtain the maximum net present value.

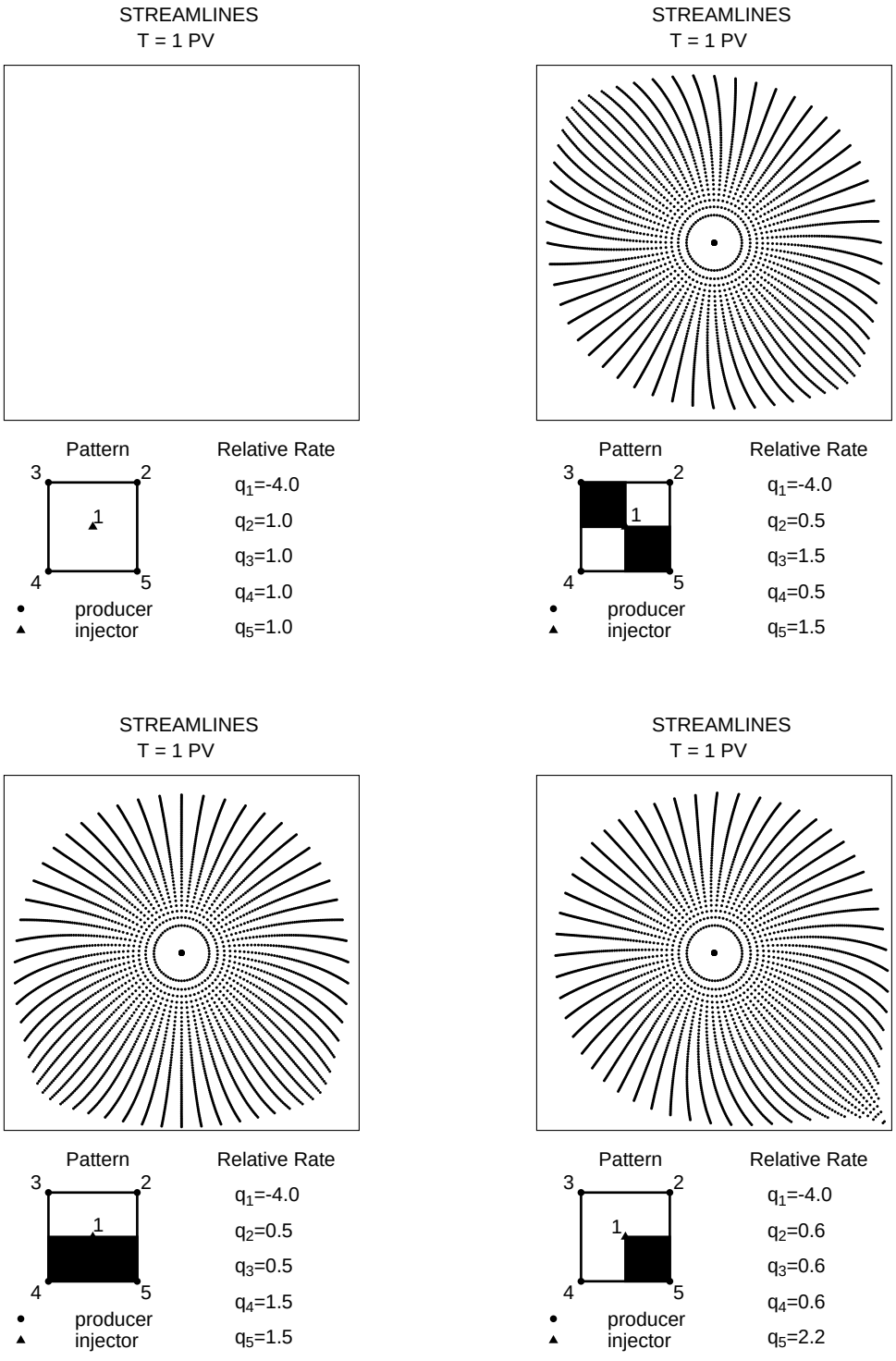

Figure 6.5: Effect of Different Flow Rates: b/a=1; (c,d)=(0,0)
Isolated Pattern in Open Homogeneous Reservoir

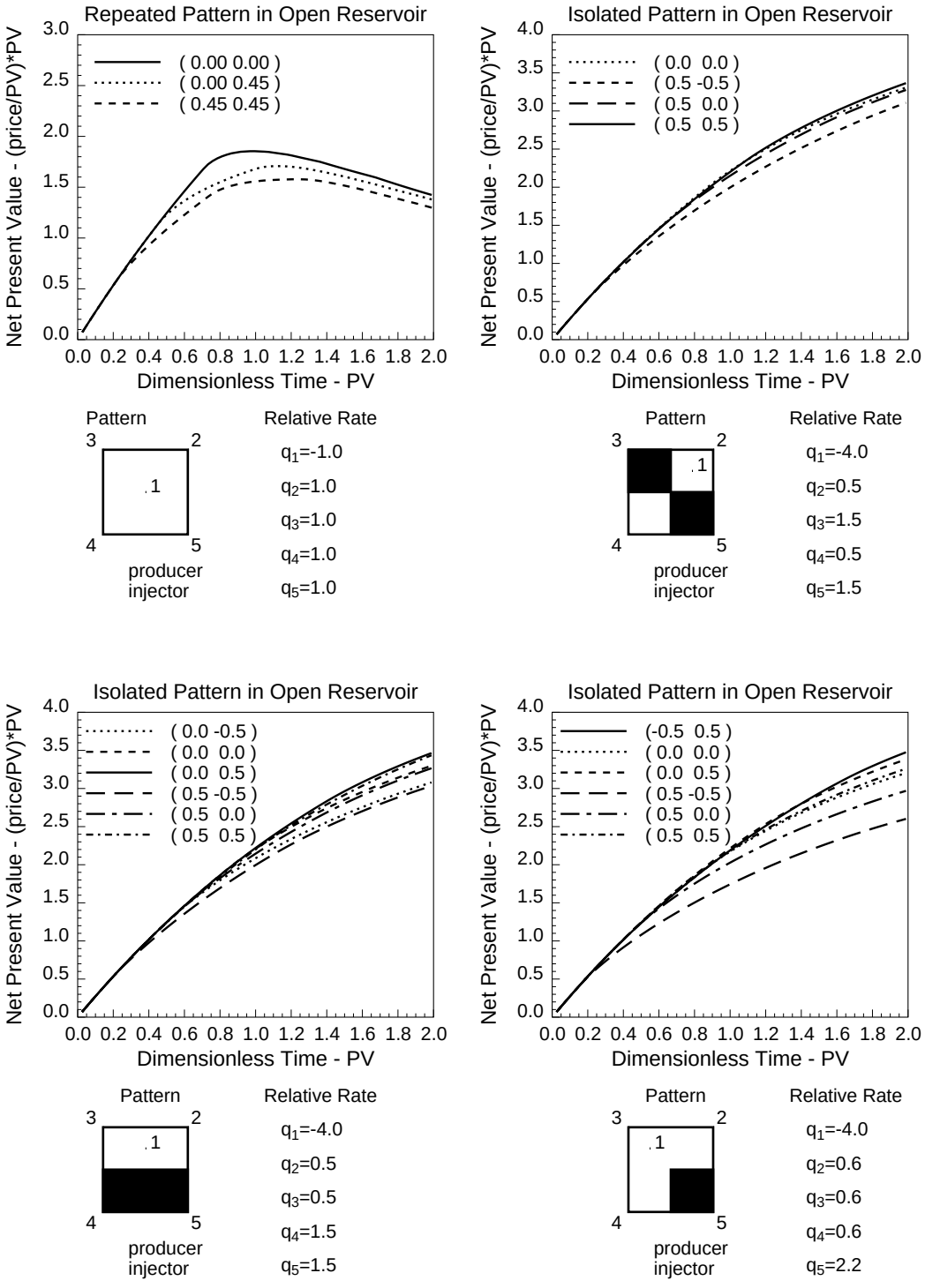

Figure 6.6: Effect of Injector Location: b/a=1

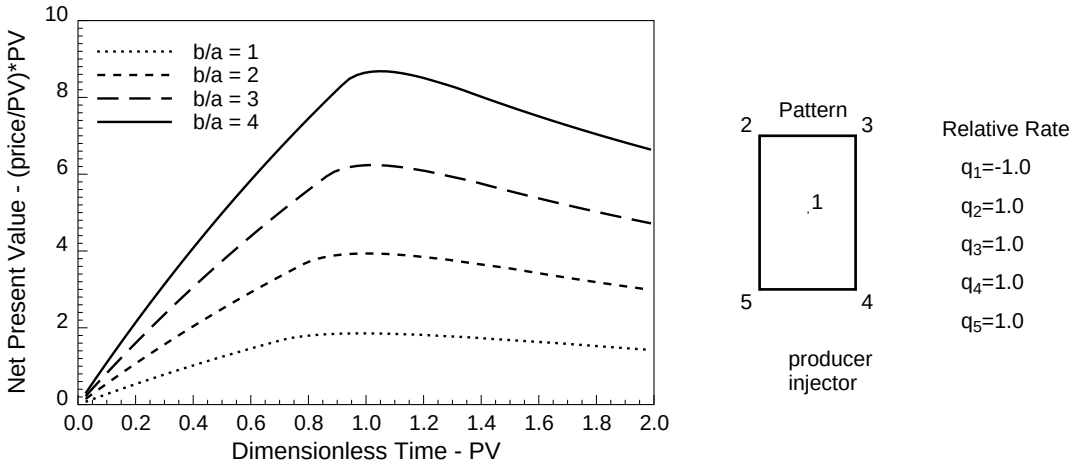

Figure 6.7: Effect of Pattern Shape: (c,d)=(0,0)
Repeated Pattern in Homogeneous Reservoir

Discretizing the Parameter Domain Investigated

The values of the decision variables were discretized uniformly in the four-dimensional parameter space investigated. The optimal strategy would be searched among these discretized points. In this project, the number of exhaustive combinations was $4 \times 26 \times 26 \times 11 = 29744$. This would require $4 \times 26 \times 26 = 2704$ simulation runs. Using the interpolation approach, only $4 \times 3 \times 3 = 36$ simulations were performed, and the net present values at 4 time steps were recorded in each simulation run, so that altogether $36 \times 4 = 144$ combinations were investigated by simulation to provide the sample data for interpolation as shown in Table 6.2.

Table 6.2: Numbers of Discretized Parameter Levels

Data Set	pattern shape	injector location		optimal injection time	total number of combinations
	b/a	c	d	t_{opt}	
Sample Simulation Data	4	3	3	4	144
Interpolation Realizations	4	26	26	11	29744

6.3 Interpolation Results

Applying the multivariate interpolation algorithms, the 144 prior data obtained by simulation were interpolated to evaluate the objective function values at the 29744 points in the parameter space. When the Least Squares algorithm was applied, the highest polynomial degree was chosen as 2 to maintain the smoothness with reasonable small residuals. When the Kriging algorithm was applied, the highest polynomial degree for the variable mean was chosen as 2. The optimal solution was searched for the maximum net present value among the new realizations. The optimal solutions of the repeated pattern and isolated pattern for different relative production rates are shown in Tables 6.3 and 6.4.

In all cases, the optimal pattern shape is a rectangle with aspect ratio $b/a = 4$. In a homogeneous reservoir with constant flow rate at each well, this is a reasonable answer, because when the producers are located far away from the injector, then the breakthrough of water happens later and more oil will be produced within a certain time period. The same results were obtained from the previous simulation analysis as shown in Figure 6.7. For an isolated pattern in an open homogeneous reservoir, the optimal total injection time is always 2 PV within the time period under study even though each producer may produce oil at a different flow rate. This is due to the open pattern boundaries. The water is injected into the reservoir and pushes the oil to the production wells, certain amounts of water may pass by the producers and flow outside of the pattern. This reduces the water production and increases the oil production. The repeated pattern has closed pattern boundaries, so the water is forced to flow to the producers, this causes earlier breakthrough so that the net present value begins to decline earlier as shown in the first plot of Figure 6.6. For the four flow rate assignments investigated, the optimal total operation time is around 1.0 PV for repeated patterns. The optimal location of the injector is very sensitive to the relative production rate of each producer. Comparing the two coordinates of the injector, the x coordinate is less sensitive than the y coordinate for the optimal pattern shape.

Comparing the Least Squares and Kriging algorithms, both methods could capture the basic structure of the parameter space, but it seemed that the Kriging interpolation predicted the optimal solution more accurately. The Least Squares method tended to estimate the objective value with greater deviation, this could be corrected by performing the simulation again in the optimal region obtained in this step. Some interpolation results are shown in Figure 6.8, 6.9 and 6.10. The color maps of net present values are presented in the domain of two parameters while fixing the other two parameters.

Table 6.3: Optimal Solutions of Isolated Pattern

flow rates: $q_1 = -4.0, q_2 = 1.0, q_3 = 1.0, q_4 = 1.0, q_5 = 1.0$

Data Set	Number of Points	pattern shape	injector location		optimal inj. time	objective function
		b/a	c	d	t_{opt}	NPV
Sample Simulation Data set	144	4	0.0	0.0	2.000	14.44
Least Squares Realizations	29744	4	1.0	0.0	2.000	14.55
Kriging Realizations	29744	4	1.0	0.0	2.000	14.44

flow rates: $q_1 = -4.0, q_2 = 0.5, q_3 = 1.5, q_4 = 0.5, q_5 = 1.5$

Data Set	Number of Points	pattern shape	injector location		optimal inj. time	objective function
		b/a	c	d	t_{opt}	NPV
Sample Simulation Data set	144	4	0.00	0.00	2.000	14.39
Least Squares Realizations	29744	4	-1.0	-.48	2.000	14.23
Kriging Realizations	29744	4	-.36	-.16	2.000	14.39

flow rates: $q_1 = -4.0, q_2 = 0.5, q_3 = 0.5, q_4 = 1.5, q_5 = 1.5$

Data Set	Number of Points	pattern shape	injector location		optimal inj. time	objective function
		b/a	c	d	t_{opt}	NPV
Sample Simulation Data set	144	4	0.50	0.00	2.000	14.31
Least Squares Realizations	29744	4	0.04	1.44	2.000	14.50
Kriging Realizations	29744	4	0.04	0.80	2.000	14.47

flow rates: $q_1 = -4.0, q_2 = 0.6, q_3 = 0.6, q_4 = 0.6, q_5 = 2.2$

Data Set	Number of Points	pattern shape	injector location		optimal inj. time	objective function
		b/a	c	d	t_{opt}	NPV
Sample Simulation Data set	144	4	-0.5	0.00	2.000	14.27
Least Squares Realizations	29744	4	-1.0	1.12	2.000	14.53
Kriging Realizations	29744	4	-1.0	0.80	2.000	14.47

Table 6.4: Optimal Solutions of Repeated Pattern

flow rates: $q_1 = -1.0, q_2 = 1.0, q_3 = 1.0, q_4 = 1.0, q_5 = 1.0$

Data Set	Number of Points	pattern shape	injector location		optimal inj. time	objective function
		b/a	c	d	t_{opt}	NPV
Sample Simulation Data set	144	4	0.0	0.0	1.059	8.682
Least Squares Realizations	29744	4	0.0	0.0	1.800	8.349
Kriging Realizations	29744	4	0.8	0.0	1.200	8.727

flow rates: $q_1 = -1.0, q_2 = 0.5, q_3 = 1.5, q_4 = 0.5, q_5 = 1.5$

Data Set	Number of Points	pattern shape	injector location		optimal inj. time	objective function
		b/a	c	d	t_{opt}	NPV
Sample Simulation Data set	144	4	0.00	0.00	0.928	7.883
Least Squares Realizations	29744	4	-.12	-.16	1.200	6.259
Kriging Realizations	29744	4	0.04	0.16	1.000	7.740

flow rates: $q_1 = -1.0, q_2 = 0.5, q_3 = 0.5, q_4 = 1.5, q_5 = 1.5$

Data Set	Number of Points	pattern shape	injector location		optimal inj. time	objective function
		b/a	c	d	t_{opt}	NPV
Sample Simulation Data set	144	4	0.00	2.00	1.117	9.035
Least Squares Realizations	29744	4	0.04	4.00	2.000	10.920
Kriging Realizations	29744	4	0.04	2.08	1.200	9.106

flow rates: $q_1 = -1.0, q_2 = 0.6, q_3 = 0.6, q_4 = 0.6, q_5 = 2.2$

Data Set	Number of Points	pattern shape	injector location		optimal inj. time	objective function
		b/a	c	d	t_{opt}	NPV
Sample Simulation Data set	144	4	-0.50	2.00	1.089	8.453
Least Squares Realizations	29744	4	1.00	4.00	1.200	9.554
Kriging Realizations	29744	4	-0.28	2.08	1.000	8.366

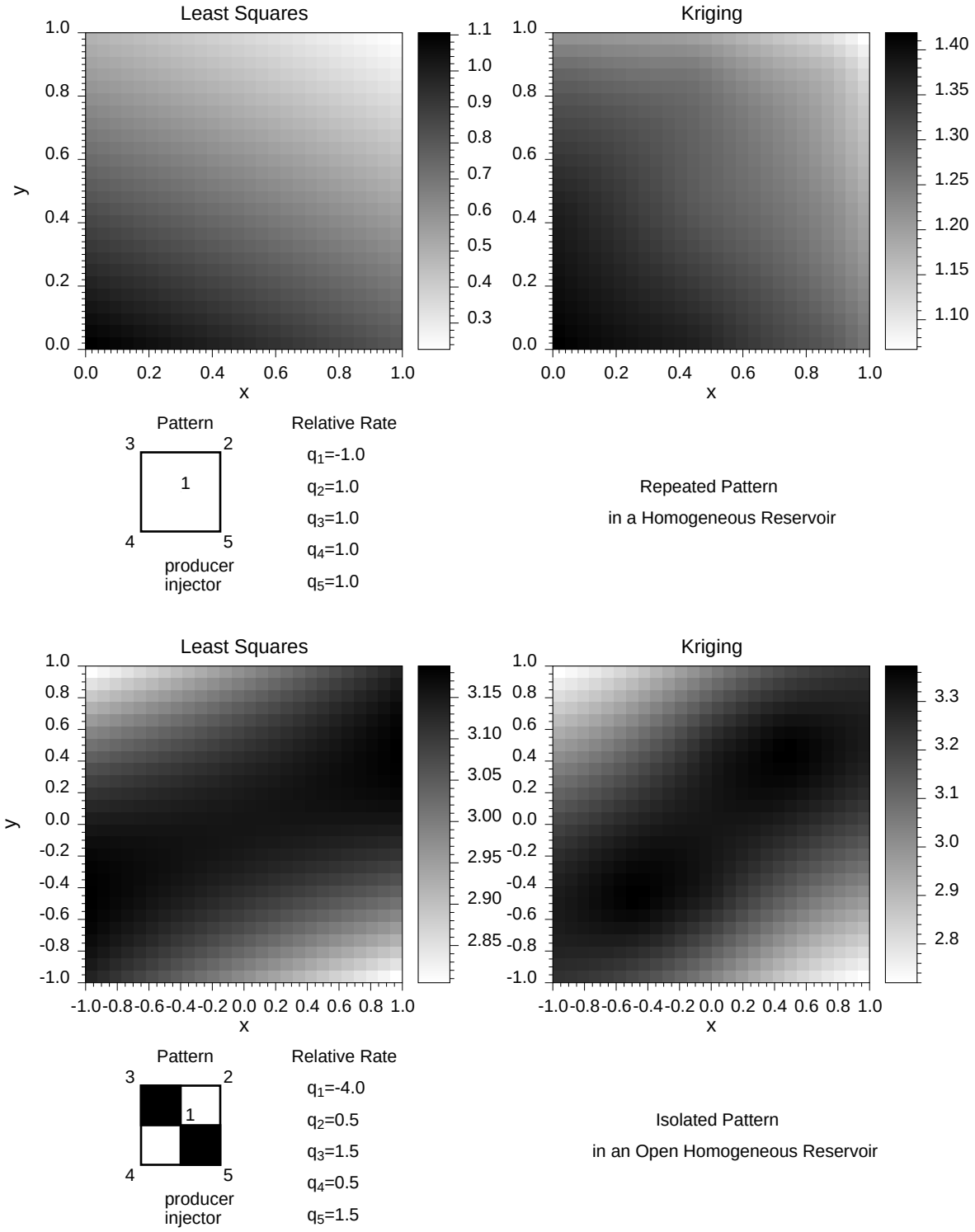

Figure 6.8: NPV Color Maps for Optimal Injector Location: b/a=1, T=2.0 PV

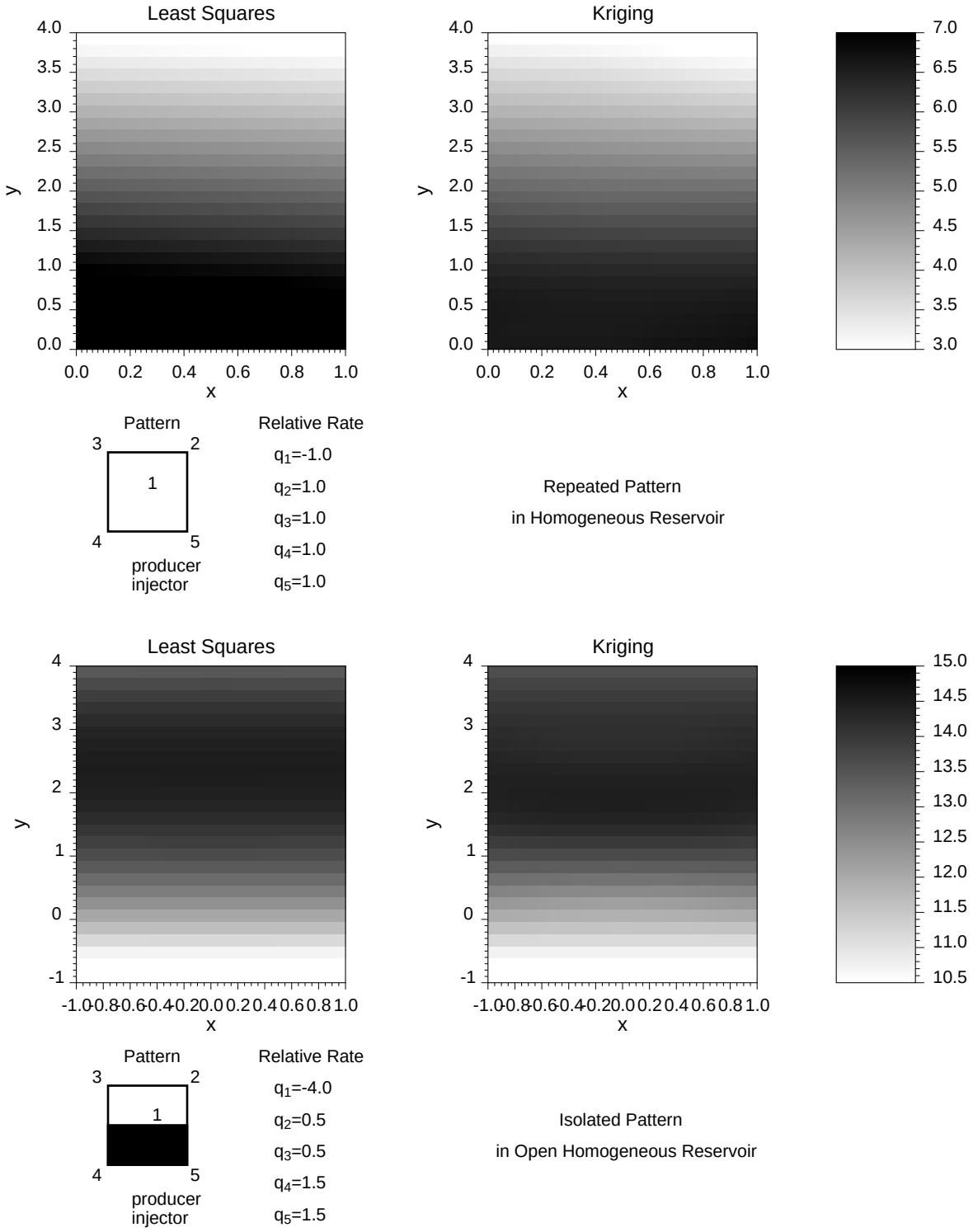

Figure 6.9: NPV Color Maps for Optimal Injector Location: b/a=4, T=2.0 PV

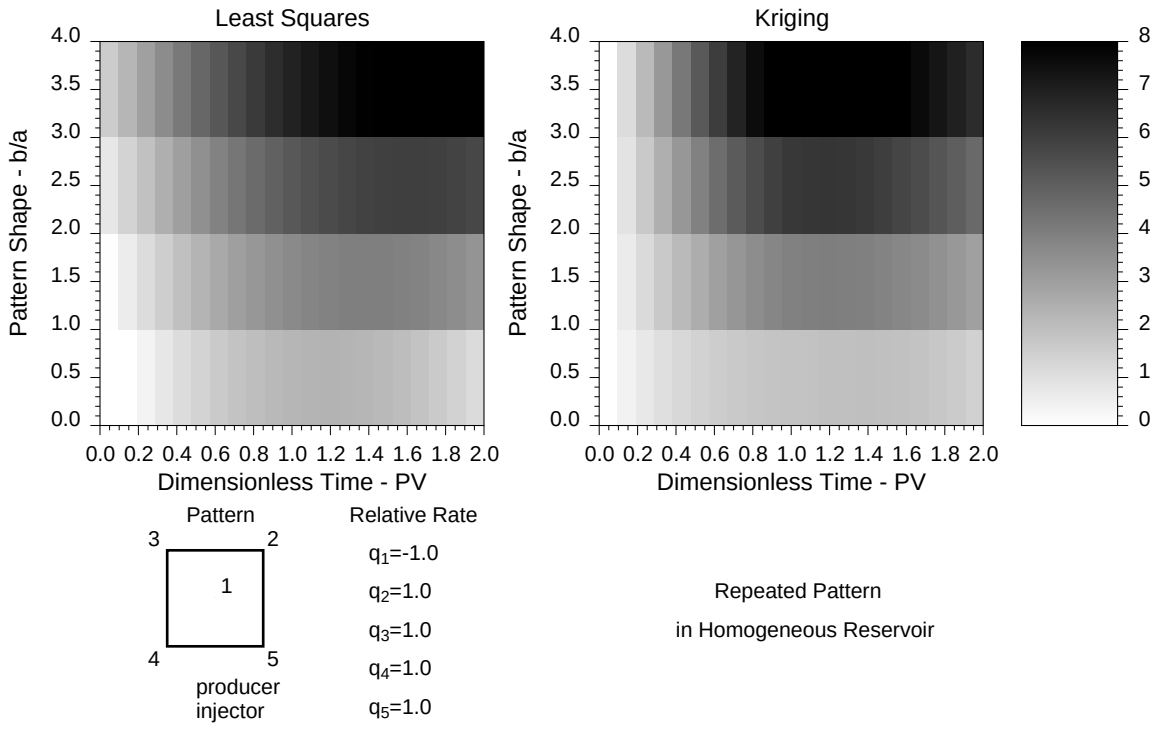

Figure 6.10: NPV Color Maps for Optimal b/a and t_{opt}: (c,d)=(0,0)

6.4 Validation of Recommended Realizations

The optimal solutions obtained from interpolations need to be validated. This requires the execution of the simulations in the intermediate optimal region to search for the final global optimum and calculate the real objective function values.

From the previous interpolation analysis, the pattern shape factor $b/a = 4$ was proven to be the best among the 4 investigated shapes, therefore, more simulations were performed by changing the injector location only in the optimal region. After the curve of NPV versus time was obtained, the optimal operation time and the maximum NPV could be found.

For the repeated pattern with equal relative production rates (first case shown in Table 6.4), another $6 \times 5 = 30$ injector locations in the optimal region predicted by interpolations were investigated by simulation. Then the final optimal solution was obtained as:

$$b/a = 4, \qquad (c, d) = (0.0, 0.0), \qquad t_{opt} = 1.059, \qquad NPV = 8.682$$

This is the same as the analytical solution and is well known as the best choice symmetric staggered line drive.

The other cases could be validated similarly, but the processes are omitted here.

6.5 Summary

In this well placement design, applying the multivariate interpolation techniques, the total number of simulations required to obtain the global optimum was reduced to $36 + 30 = 66$, and the optimal solution was shown to be reliable as represented in Table 6.5 for the case of repeated patterns with equal relative production rates. This demonstrates the efficiency and feasibility of this mathematical approach.

Table 6.5: Comparison of Two Optimization Approaches

Method	Number of Simulations	Optimal Solution				
		pattern shape	injector location		optimal inj. time	objective function
		b/a	c	d	t_{opt}	NPV
Simulation	2704	4	0.0	0.0	1.059	8.682
Simulation+Interpolation	66	4	0.0	0.0	1.059	8.682

Section 7

Conclusions and Directions of Future Projects

7.1 Conclusions

In this study, a mathematical approach was developed to solve multivariate optimization problems. This method applies the Least Squares or Kriging interpolation techniques to generate new realizations from a limited number of prior data obtained from simulation. The recommended solutions can be obtained by searching for the optimal objective function values among the interpolation realizations. Additional simulation runs may be performed to search for the final optimal solution in the vicinity of the intermediate optimal region and to validate the results. The feasibility and efficiency of this algorithm were investigated in two optimization projects: field development scheduling and well placement design. From the application examples, the following conclusions can be made:

1. The algorithm is practical in obtaining reliable optimal strategies. It can reduce the number of simulation required substantially, especially for multivariate optimization problems with more than three parameters.

2. The Least Squares and Kriging interpolations can provide a global sketch of the objective function surface and maintain the basic structure of the parameter space under study to avoid possible failure at local optima. The absolute optimum may be reached by refining the grids near the intermediate optimum.

3. The interpolation results depend on the number and distribution of the prior data. Therefore, it is important to obtain reliable sample data. The smoothness of the objective function surface can be controlled by modifying the highest degree of the representative polynomials.

4. Because the objective surface generated by the Kriging algorithm traverses all the sample data points, the objective function values estimated are more accurate than those estimated by Least Squares interpolation, but both algorithms require validation by simulations after optimal solutions are obtained.

7.2 Future Extensions

There are many practical direct searching algorithms being used in optimization problems, such as the polytope, genetic algorithm and tabu search methods introduced in section 1. It may be more efficient if the optimization is performed by combining multivariate interpolation with the direct searching algorithms. The interpolation can provide the sketch of objective function surface, and the direct searching may be started from the intermediate optimal region without searching for many extra steps. The searching direction can be modified by analyzing the representative surface, at the same time the interpolation can be updated by adding more points obtained from direct algorithm into the sample data set. This approach may save lots of time and money without reducing the accuracy of the optimal solution.

Bibliography

[1] Matheron, G.: *Les variables Régionalisées et leur Estimation, une Application de la Theorie de Fonctions Aleatoires aux Sciences de la Nature*, Paris, Masson et Cie (1965).

[2] Watson, A. Ted; Lane, H. Scott and Gatens, J. Michael III: "History Matching with Cumulative Production Data," *JPT* (January 1990) 96–100.

[3] Olarewaju, Joseph S.: "Mathematical Model of Permeability Alteration around Wells," *Int J Numer Anal Methods Geomech* (April 1990) 191–207.

[4] Chung, Chang Bock and Kravaris, Costas: "Incorporation of a Priori Information in Reservoir History Matching by Regularization," *Soc Pet Eng AIME Pap SPE* (February 1991) 45.

[5] Mohammed, S. A. and Wattenbarger, R. A.: "Simplified Method for Computing Phase Behavior of Carbon Dioxide/Hydrocarbon Mixtures in Compositional Simulation," paper SPE 15720 presented at the 1991 Proceedings of the 7th Middle East Oil Show, Manama, Bahrain, November 16-19.

[6] Chardaire-Riviere, Catherine; Chavent, Guy; Jaffre, Jerome; Liu, Jun and Bourbiaux, Bernard J: "Simultaneous Estimation of Relative Permeabilities and Capillary Pressure," *SPE Form Eval* (December 1992) 283–289.

[7] Bonalde, I. and Ramones, M.: "Robust Algorithm for Parameter Estimation in Well Tests," *SPE Advanced Technology Series* (March 1994) 119–125.

[8] Delhomme, J. P.: "Kriging in the Hydrosciences," *Advances in Water Resources* (May 1978) 251–266.

[9] Doctor, P. G.: *An Evaluation of Kriging Techniques for High Level Radioactive Waste Repository Site Characterization*, Dept. of Energy, Pacific Northwest Laboratory (1979).

[10] Dunlap, L. E.: *Interpolating Water-table Altitudes in West-central Kansas Using Kriging Techniques*, Dept. of Interior, U. S. Geological Survey (1984).

[11] Wackernagel, Hans: "Description of a Computer Program for Analyzing Multivariate Spatially Distributed Data," *Computers and Geosciences* (1989) 593–598.

[12] Tavares Ribeiro, Luis and da Costa e Silva, A.: "Geomathematical Model Tested on an Oil Reservoir," paper SPE 10481 presented at the 1986 19th Application of Computers and Operations Research in the Mineral Industry, University Park, PA, April 14-16.

[13] Brummert, A. C.; Pool, S. E.; Portman, M. E.; Hancock, J. S.; Ammer, J. R.: "Determining Optimum Estimation Methods for Interpolation and Extrapolation of Reservoir Properties: A Case Study," paper SPE 12641 presented at the 1989 SPE Annual Technical Conference and Exhibition, San Antonio, TX, October 8-11.

[14] Al Rumhy, M. H.; Archer, J. S.; Daltaban, S. T.: "Synergistic Approach to Characterization of Reservoir Permeability: A Conditional Kriging Method," paper SPE 15720 presented at the 1991 Proceedings of the 7th Middle East Oil Show, Manama, Bahrain, November 16-19.

[15] Egeland, Thore; Hatlebakk, Einar; Holden, Lars; Larsen, E. A.: "Designing Better Decisions," paper SPE 16702 presented at the 1992 Proceedings of the European Petroleum Computer Conference, Stavanger, Norway, May 24-27.

[16] Watkins, A. J.: "Linking Universal Kriging and Reservoir History Matching," paper SPE 19464 presented at the 1993 Proceedings of the 1993 SPE Annual Technical Conference and Exhibition, Houston, TX, October 3-6.

[17] Poquioma, W. and Kelkar, Mohan: "Application of Geostatistics to Forecast Performance for Waterflooding an Oil Field," *SPE Advanced Technology Series* (March 1994).

[18] Reznicek, K. and Cheng, T. C. F.: "Stochastic Modelling of Reservoir Operations," *Eur J Oper Res* (1991).

[19] Hansen, Pierre; de Luna Pedrosa Filho, Eugenio; Ribeiro, Celso Carneiro: "Location and Sizing of Offshore Platforms for Oil Exploration," *Eur J Oper Res* (1992).

[20] Alcobia, V. M.; Goodearl, Anthony; Brindle, Dvid; Wood, Michael: "Field Development Optimization through Economic Post Processing of Reservoir Simulation Results," paper SPE 21514 presented at the 1994 Proceedings of the European Petroleum Conference, London, UK, October 25-27.

[21] Tauzin, Eric and Horne, R. N.: "Influence Functions for the Analysis of Well Test Data from Heterogeneous Permeability Distributions," paper SPE 21443 presented at the 1994 Proceedings of the SPE Annual Technical Conference and Exhibition, New Orleans, LA, USA, September 25-28.

[22] Fujii, Hikari and Horne, R. N.: "Multivariate Optimization of Networked Production Systems," paper SPE 20222 presented at the 1994 Proceedings of the European Production Operations Conference and Exhibition, Aberdeen, UK, March 15-17.

[23] Rogers, Leah L.; Dowla, Farid U.; Johnson, Virginia M.: "Optimal Field-Scale Groundwater Remediation Using Neural Networks and the Genetic Algorithm," *Environ. Sci. Technol* (1995).

[24] Lo, K. K.; Starley, G. P.; Holden, C. W.: "Application of Linear Programming to Reservoir Development Evaluations," *SPE Reservoir Engineering* (1995).

[25] Fang, Kaitai: "Uniform Design: Application of Number Theory in Test Design," *ACTA Mathematicae Applicatae Sinica* (1980).

[26] Fang, Kaitai, Statistics Dept., Inst. of Mathematics, Chinese Academy of Sciences: *The Analysis of Deviation*, Beijing, Science Publishing House (1977).

[27] Ravindran, Niranjan: "Multivariate Optimization of Production Systems - the Time Dimension," Master's thesis, Stanford University (June 1992).

Appendix A

Computer Programs

The fortran codes used for uniform design, multivariate interpolation and simulation for those two projects mentioned in section 5 and section 6 are presented here.

A.1 Uniform Design Program

```
c************************************************************************
c  File: udis.f Apr. 10, 1995 *
c------------------------------------------------------------------------*
c Uniform Distribution Design of Multi-factor & Multi-level Test *
c written by: Yan Pan *
c *
c Reference: Uniform Design -- Fang, Kaitai (ACTA 1980) *
c s - # of factors/dimensions q - # of levels/grids *
c is() - the distributed point index *
c Principles: *
c 1) one test for each level of each factor, totally q tests *
c 2) q = odd # kq or kq -1, select integer a(1)..a(s) satisfying: *
c    max common factor [a(i),kq] = 1, then: *
c    is() = [ka(1),ka(2),..,ka(s)][mod kq], k = 1,..,kq *
c 3) q = prime # p or p - 1 then: *
c    is() = [k,ka,ka**2,..,ka**(s-1)][mod p], k = 1,..,q *
c *
c prim() -- prime #'s q_pf() -- prime factors of q *
```

```
c ia() -- #'s < q & [ia(),q_pf()] = 1 *
c a(s) -- possible combination of ia ns() -- #'s left in ia *
c ar(s), amin -- best choice of a in 2), 3) *
c**************************************************************************
program udis

parameter(np = 25)
integer s,q,prim(np),q_pf(np),is(np),ia(np),ns(np)
real a(np),ar(np),amin

data prim/2,3,5,7,11,13,17,19,23,29,31,37,41,43,47,
     & 53,59,61,67,71,73,79,83,89,97/

c open(1,file='dat',status='old')
5 continue
c read(1,*,end = 999)s,q
write(*,*)'factor: s = ?;  level: q = ?'
read(*,*)s,q

open(2,file='dis',status='unknown')
11 format(1x,a6,15(1x,i3))

c------ prime number less than q -----
c call prime(q,mprim,prim)

c --- check if q = odd # ---
if(real(q)/2.eq.q/2) then
  kq = q + 1
else
  kq = q
endif
c ---- check if q = prime number p or p-1 ---
kdis = 0
do 20 j = 1, max0(q,np)

if(kq.eq.prim(j)) then
  goto 25
```

```
else if(kq.ge.25) then
  do l = 2, 4
   if(dble(kq).eq.dble(prim(j))**dble(l)) goto 25
  end do
else if(kq.lt.prim(j)) then
  kdis = 1
  mprim = j - 1
  goto 25
endif

20 continue

25 continue
write(*,*)
write(*,11)'s q  =',s,q

a(1)=1
cmin = 1.e5
if(kdis.eq.0) then

c------ distribution when q = p or p-1 -----
amin = 1
do 200 n = 2, kq - 1

do 30 l = 2, s
a(l) = real(n)**(l-1.)
30 continue

call best(s,kq,a,cita)

if(cita.lt.cmin) then
 cmin = cita
 amin = a(2)
endif

200 continue
```

```
write(*,11)'amin =',nint(amin)

c --- calculation of distribution ---
do 500 k = 1, q
is(1) = fmod( dble(k), kq )
do 400 i = 2, s
z = amin
do 300 j = 2, i - 1
300 z = fmod( z*amin, kq )
is(i) = fmod( z*dble(k), kq )
400 continue
write(2,88) (is(l), l = 1, s)
500 continue

c------ distribution when q.ne.p -----
else

c --- find the prime factors of q: q_pf ---
m = 0
do 1000 i = 1, mprim
if(mod(kq,prim(i)).eq.0) then
 m = m + 1
 q_pf(m) = prim(i)
endif
1000 continue
mpf=m
write(*,11)'q_pf =',(q_pf(l),l=1,mpf)

c --- find ia satisfying mod(ia,q_pf)=0 i.e. common(ia,q)=1 ---
m=0
do 1040 i = 2, kq-1
do 1020 j = 1, mpf
if(mod(i,q_pf(j)).eq.0) goto 1040
1020 continue
m = m + 1
ia(m) = i
1040 continue
```

```
ma = m
write(*,11)'ia   =',(ia(l),l=1,ma)

c --- calculation of best a(s) ---
do l = 1, s
ar(l) = 1
end do

s1 = s - 1

if(s1.eq.1) then
  do i = ma, 1, -1
    a(s) = ia(i)
    call best(s,kq,a,cita)
    if(cita.lt.cmin) then
      cmin = cita
      ar(s) = a(s)
    endif
  end do
  goto 1300
endif

do i = 1, s1 -1
j = ma - i + 1
a(i+1) = ia(j)
ns(i) = j - 1
end do

1100 continue
do i = ns(s1-1), 1, -1
  a(s1+1) = ia(i)
  call best(s,kq,a,cita)
  if(cita.lt.cmin) then
    cmin = cita
    do j = 1, s
     ar(j) = a(j)
    end do
```

```
  endif
end do

do 1200 j = s1-1, 1, -1
if(ns(j).gt.(s1-j+1)) then
  a(j+1) = ia(ns(j))
  ns(j) = ns(j) - 1
  do i = j+1, s1-1
    a(i+1) = ia(ns(i-1))
    ns(i) = ns(i-1) - 1
  end do
  goto 1100
else
  do i = j, s1
  a(i+1) = ia(s1-i+1)
  end do
  call best(s,kq,a,cita)
  if(cita.lt.cmin) then
    cmin = cita
    do i = 1, s
     ar(i) = a(i)
    end do
  endif
endif
1200 continue

1300 write(*,11)'a(s) =',(nint(ar(l)), l = 1, s)

c --- calculation of distribution ---
do 2000 k = 1, q
do 1500 i = 1, s
is(i) = fmod( ar(i), kq )
is(i) = fmod( is(i)*dble(k), kq )
1500 continue
write(2,88)(is(l), l = 1, s)
2000 continue
88 format(30(1x,i2))
```

```
endif

close(2)
c goto 5

999 stop
end

c------ subroutine of searching for best factors ------
subroutine best(s,lev,a,cita)

parameter(np = 25)
integer s,lev
real a(np)
data pie/3.1415926535/

cita = 0.
do 100 k = 1, lev
pm = 1.
do 50 i = 1, s
aik = fmod( a(i), lev )
aik = fmod( aik*dble(k), lev )
pm = pm * (1.- 2/pie * log(2*sin(pie*aik/(lev+1))) )
50 continue
cita = cita + pm
100 continue
cita = cita / lev

return
end

c------ function of calculating residual ------
function fmod(x,n)

fmod = dmod( dble(x), dble(n))
if(fmod.ge.n) then
```

```
  write(*,*)'wrong calculation: x,n,fmod'
  write(*,*)x,n,fmod
  stop
endif
if(fmod.eq.0) fmod = n

return
end
```

A.2 Interpolation Programs

A.2.1 Generator of Input Data File for Interpolation

```
c**************************************************************************
c  File: fit.f Apr. 26, 1995 *
c-------------------------------------------------------------------------*
c To Generate Datafile 'in' for LS/Kriging Interpolation     *
c written by: Yan Pan *
c**************************************************************************
program fit

parameter(nm = 50)
character*15 fn
dimension n(nm),m(nm)
real a(nm),x(nm),xmin(nm),xmax(nm),dx(nm)

write(*,*)'input filename = ?'
read(*,'(a)')fn
open(1,file=fn,status='old')

c write(*,*)'dimension nd=?; # of data points: np=?'
c read(*,*)nd,np
c write(*,*)'# of col. of x() & y: n(),nv = ?'
c read(*,*)(n(i), i=1,nd),nv

data nd,np/4,144/
data n(1),n(2),n(3),n(4),nv/1,2,3,4,5/

c do i = 1, nd

c write(*,*)i,'th dimension:'
c write(*,*)'interpolation range: xmin(),xmax() = ?'
c read(*,*)xmin(i),xmax(i)
c write(*,*)'# of points to be generated: m() = ?'
c read(*,*)m(i)

c end do
```

```
data xmin(1),xmax(1),m(1)/1,4,4/
data xmin(4),xmax(4),m(4)/0,2,21/

data xmin(2),xmax(2),m(2)/-1,1,26/
data xmin(3),xmax(3),m(3)/-1,1,26/

open(2,file='in',status='unknown')
write(2,11)nd,np
11 format(5(1x,i6))
write(2,*)

c --- sample data ---
do 50 i = 1, np
read(1,*)(a(j), j = 1, nv)
do j = 1, nd
x(j) = a(n(j))
end do
y = a(nv)
write(2,22)(x(l), l = 1, nd), y
50 continue

c --- interpolation points ---
mt = 1
do i = 1, nd
mt = mt * m(i)
end do

write(2,*)
write(2,11)mt
write(2,*)

do i = 1, nd
if(m(i).eq.1) then
  dx(i) = 0
else
  dx(i) = (xmax(i) - xmin(i)) / (m(i) - 1.)
```

```
endif
end do

do 500 i2 = 1, m(2)
x(2) = xmin(2) + dx(2) * (i2 - 1.)

do 500 i1 = 1, m(1)
x(1) = xmin(1) + dx(1) * (i1 - 1.)

do 500 i3 = 1, m(3)
x(3) = xmin(3) + dx(3) * (i3 - 1.)
c -- y coordinate of injector is dependent on pattern size b
x(3) = x(3) * x(1)

do 500 i4 = m(4), 1, -1
x(4) = xmin(4) + dx(4) * (i4 - 1.)

c do 500 i5 = 1, m(5)
c x(5) = xmin(5) + dx(5) * (i5 - 1.)
c do 500 i6 = 1, m(6)
c x(6) = xmin(6) + dx(6) * (i6 - 1.)

write(2,22)(x(l), l=1,nd)
500 continue
22 format(10(1x,f9.4))

end
```

A.2.2 Main Program of Least Squares Interpolation

```
c**********************************************************************
c  File: lsq.f Dec. 30, 1994 *
c  Usage: link with files:  slu.f inf.f mtp.f fit.cm *
c----------------------------------------------------------------------*
c Program of Multivariate Least Squares Interpolation *
c written by: Yan  Pan *
c *
c polynomial fitting function = sum{Ck*(Xi**Ki)*(Xj**Kj)} *
c i,j = 1, ..., n  k = 1, ..., m1 *
c n  -- # of dimensions m  -- # of data pts known *
c kk -- degree of polynomial m1 -- # of coefficients of poly *
c ifm(m,4) -- information carrier ifm(i,j,ki,kj) *
c mm -- # of new data pts to be evaluated *
c *
c linear system needs to be solved: A'WA*C = A'W*Y *
c {Xmn} -- data set {Ym} -- function value *
c {Wm}  -- vector of Weight factors *
c {Amm1} - matrix of basis {Cm1} - vector of coefficients *
c {EL} = A'WA {ER} = A'WY
c**********************************************************************
        program lsq

include '/oseberg/home/pete/pan/fit/fit.cm'

        real x(np,np),y(np),w(np)
        real a(np,np),c(np),el(np,np),er(np),b(np,np)
        dimension ifm(np, 4)

        open(1,file='in',status='old')
        open(2,file='chk.lsq',status='unknown')
        open(3,file='out.lsq',status='unknown')

c       ----- input known data -----
        read(1,*)n,m
        IF (n.le.0.or.m.le.0) THEN
          write(*,*)'Error=1: N or M is less than zero.'
```

```
          STOP
        ELSE IF (m.eq.1) THEN
          write(*,*)'M=1: The evaluation value is constant'
          STOP
        END IF

do 5 i = 1, m
w(i) = 1.
5 continue

        read(1,*)
        do 10 i=1,m
c read(1,*)(x(i,l),l=1,n),y(i),w(i)
        read(1,*)(x(i,l),l=1,n),y(i)
10      continue

ymax=y(1)
imax=1
do 12 i=2,m
if(y(i).ge.ymax) then
  ymax=y(i)
  imax=i
endif
12 continue

c---------- choose the best degree of polynomials -----
        kk = 1
i1 = factorial(1, n)
15      continue
i2 = factorial(kk+1, kk+n)
        m1 = i2 / i1
        IF (m1.lt.m) THEN
          kk = kk + 1
          GOTO 15
        ELSE
          write(*,*)m,'=m',kk,'=kk',m1,'=m1'
          write(*,*)'Input k=?'
```

```
          read(*,*)kk
  i2 = factorial(kk+1, kk+n)
          m1 = i2 / i1
        END IF
        write(2,11)n,kk,m1
11      format(/,1x,10(1h-),' LSQ ',10(1h-),/,
     &          ' Dimension: N =',i2,/,
     &         ' Degree of Polynomial:   K  = ',i2,/,
     &          ' Number of Coefficients: M1 = ',i3)

c---------- fitting -----
        IF (kk.eq.0) THEN
            write(*,*)'KK=0: The evaluation value is constant'
            STOP
        END IF

c------ create the information carrier IFM for KK degree
write(*,*)'----- fitting -----'
call inf(n,kk,ifm,m1)
write(2,22)m1
22 format(' # of coefficients used:  M1 = ',i3)
write(2,23)m
23 format(' Original Data  ( ',i5,' pts ):')
write(2,37)ymax,(x(imax,l),l=1,n)

c------ generate matrix of basis A
do 2000 j = 1, m1
j1 = ifm(j, 1)
j2 = ifm(j, 2)
k1 = ifm(j, 3)
k2 = ifm(j, 4)
        do 2000 i = 1, m
if(x(i,j1).ne.0.and.x(i,j2).ne.0) then
  a(i, j) = x(i,j1)**k1 * x(i,j2)**k2
else
  a(i,j) = 0
endif
```

```
2000 continue

c---------- solution of coefficients -----
c       -- B = A'W --
        do 2010 j=1,m1
        do 2010 i=1,m
        b(j, i) = a(i, j) * w(i)
2010    continue
c       -- EL = BA = A'WA --
call mtp(m1, m, m1, b, a, el)
c       -- ER = BY = A'WY --
call mtp(m1, m, 1, b, y, er)
        write(*,*)'----- calling lu -----'
c       -- solve ER*C = ER  i.e.  A'WAc=A'Wy --
call lu(m1, m1, el, er, c)

c---------- evaluation -----
        write(*,*)'----- evaluation -----'
c       -- check the residual --
c       write(2,*)'----- Residuals -----'
        rr=0
        do 2800 i=1,m
        yy = 0
        do 2300 j=1,m1
        j1 = ifm(j, 1)
        j2 = ifm(j, 2)
        k1 = ifm(j, 3)
        k2 = ifm(j, 4)
if(x(i,j1).ne.0.and.x(i,j2).ne.0) then
  yy = yy + c(j) * x(i,j1)**k1 * x(i,j2)**k2
endif
2300    continue
        if(y(i).ne.0) then
  err = (yy - y(i))/y(i)
        else
          err = 1.
        endif
```

```
c write(2,33)(x(i,l), l=1,n), y(i), yy, err
        rr = rr + (yy - y(i))**2
2800    continue

c       -- evaluation --
        read(1,*)
        read(1,*)mm
        read(1,*)
        do 3000 i = 1, mm
        yy = 0

c       -- input data -- (Wn - new data pts)
        read(1,*)(w(l), l = 1, n)
        do 2500 j=1,m1

c       -- calculation -- (YY - new function value)
        j1 = ifm(j, 1)
        j2 = ifm(j, 2)
        k1 = ifm(j, 3)
        k2 = ifm(j, 4)
if(w(j1).ne.0.and.w(j2).ne.0) then
  yy = yy + c(j) * w(j1)**k1 * w(j2)**k2
endif
2500    continue

c -- search for maximum value & position --
if(i.eq.1) then
  ymax = yy
  do 2511 l = 1, n
  x(l,1) = w(l)
2511   continue
else if(yy.ge.ymax) then
  ymax = yy
  do 2515 l = 1, n
  x(l,1) =w (l)
2515   continue
endif
```

```
c       -- output --
        write(3,33)(w(l), l = 1, n), yy
3000    continue
33      format(7(1x,1pe10.3))
write(2,36)mm
36 format(' Generated Data ( ',i5,' pts ):')
write(2,37)ymax,(x(l,1),l=1,n)
37 format(' Zmax =',1pe10.3,'  Position:',10(1x,e10.3))
        write(2,38) sqrt(rr)/m
38      format(' Norm(ERR) = ',1pe10.4)
        CLOSE (1)
        CLOSE (3)
        STOP
        END

function factorial(n1, n2)
j = n1
do 50 i = n1 + 1, n2
j = j * i
50 continue
factorial = j
return
end

c**************************************************************************
c  File: slu.f Feb. 20, 1995 *
c-------------------------------------------------------------------------*
c Linear System Solver by LU Factorization *
c written by: Yan Pan *
c *
c A0[n,n1]  B0[n]  B[n1]    -----   A0 * B = B0 *
c**************************************************************************
  subroutine LU(n, n1, a0, b0, b)

include '/ekofisk/suprid/pan/fit/fit.cm'
  real a0(np,np),b0(np),b(np),a(np,np)
```

```
  dimension iperm(np)

c       --- initial value ---
  beta = 1
  do 30 i=1,n
  iperm(i) = i
  b(i) = b0(i)
  do 30 j=1,n1
  a(i, j) = a0(i, j)
30      continue
  do 35 i=1,n
  do 35 j=1,n
35      continue

c       --- change A to LU ---
  kr = 0
  do 200 k=1,n

  ipiv = 0
  xmax = 0
  do 100 i=k,n
  x = ABS(a(i, k))
  IF (x.gt.xmax) THEN
    xmax = x
    ipiv = i
  END IF
100     continue

c       ----- check the basis of coefficient matrix -----
  IF (xmax.lt.1e - 14) THEN
    write(*,11)n,n1,kr
11        format(1x,'N=',i3,'  N1=',i3,'  Rank(An)=',i3)
    IF (kr.lt.n1) THEN
      write(*,*)'Error: The basis of given data < # of data'
      n1 = kr
      GOTO 105
    ELSE
```

```
      IF (kr.gt.n1) THEN
        write(*,*)'Error: Rank(A) > N1, Calculation Error'
        STOP
      END IF
      GOTO 105
    END IF
  END IF
  kr = kr + 1

c       ----- pivoting -----
  IF (ipiv.ne.k) THEN
    if(abs(a(k,k)).gt.beta*xmax) then
      ipiv = k
    ELSE
      x = b(k)
      b(k) = b(ipiv)
      b(ipiv) = x
      ll = iperm(k)
      iperm(k) = iperm(ipiv)
      iperm(ipiv) = ll
      do 120 j=1,n
      x = a(k, j)
      a(k, j) = a(ipiv, j)
    a(ipiv, j) = x
120         continue
    END IF
  END IF

c       ----- LU factorization -----
  IF (k.ge.n) GOTO 105
  do 150 i=k+1,n
  x = a(i, k) / a(k, k)
  b(i) = b(i) - x * b(k)
  do 150 j=k,n
  a(i, j) = a(i, j) - x * a(k, j)
150     continue
```

```
200     continue

105     continue
c       --- solution of x ---
  b(n1) = b(n1) / a(n1, n1)
  do 300 i=n1-1,1,-1
  x = b(i)
  do 250 j=i+1,n1
250     x = x - a(i, j) * b(j)
300     b(i) = x / a(i, i)

  RETURN
  END

c*************************************************************************
c  File: mtp.f Dec. 27, 1994 *
c-------------------------------------------------------------------------*
c Matrix Multiplication written by: Yan Pan *
c *
c Q1[n1,n]  Q2[n,n2]  Q[n1,n2]     -----     Q = Q1 * Q2 *
c*************************************************************************
SUBROUTINE MTP(n1,n,n2,Q1,Q2,Q)

include '/ekofisk/suprid/pan/fit/fit.cm'
real q1(np,np),q2(np,np),q(np,np)

DO 100 I=1,n1
DO 100 J=1,n2
X=0
DO 50 K=1,n
50 X=X+Q1(I,K)*Q2(K,J)
100 Q(I,J)=X
RETURN
END
```

A.2.3 Main Program of Kriging Interpolation

```
c*************************************************************************
c  File:        krig.f                                    Dec. 30, 1994   *
c  Usage: link with files:  klu.f gama.f inf.f fit.cm *
c-------------------------------------------------------------------------*
c        Program of Multivariate Kriging Interpolation           *
c               written by:     Yan     Pan                             *
c                                                                       *
c reference : Kriging in Hydrosciences -- J.P. Delhomme  *
c Z(Xi) -- func value @ given pt Xi   Z0 -- value @ new pt X0 *
c Z0 = sum{ (lamda)i * Z(Xi) } i = 1, m *
c A * C = D i,j = 1, m l = 1, m1 s -> X0 *
c A = | {gamma}ij  {f}il |  D = | {gamma}is |  C = | {lamda}i | *
c     | {f}lj      {0}ll |      | {f}ls     |      | {miu}l   | *
c *
c       polynomial basic function FL = sum{ (Xi**Ki) * (Xj**Kj) } *
c               i,j = 1, ..., n         k = 1, ..., kk                  *
c       n  -- # of dimensions           m  -- # of data pts known       *
c       kk -- degree of polynomial      m1 -- # of basis in FL *
c       ifm(m,4) -- information carrier ifm(i,j,ki,kj)                  *
c       mm -- # of new data pts to be evaluated                         *
c*************************************************************************

program krig

include '/ekofisk/suprid/pan/fit/fit.cm'
real x(np,nd),z(np),x0(nd),xi(nd),xj(nd),xmax(nd)
real a(np,np),d(np),ai(np,np),di(np),xp(np,np)
dimension ifm(np,4),iperm(np)

open(1,file='in',status='old')
open(2,file='chk.krig',status='unknown')
open(3,file='out.krig',status='unknown')

c --- m: # of pts;  n: # of dimension
read(1,*)n,m
read(1,*)
```

```
c --- kk: degree of polynomial FL
write(*,*)'Input the degree of polynomial FL: kk=?'
read(*,*)kk
c --- m1: # of coefficients of FL
call inf(n,kk,ifm,m1)

c --- input data ---
read(1,*)(x(1,j), j = 1, n), z(1)
zmax = z(1)
imax = 1
li = 1

do 20 i = 2, m
read(1,*)(x0(j), j = 1, n), zz

c --- check duplicate points
do 10 l = 1, i-1
do 13 j = 1, n
if(x0(j).ne.x(l,j)) goto 10
13 continue
goto 20
10 continue
li = li + 1
do 15 j = 1, n
x(li,j) = x0(j)
15 continue
z(li) = zz
if(zz.ge.zmax) then
  zmax = zz
  imax = li
endif

20 continue

m = li
        write(2,11)n,kk,m1
11      format(/,1x,10(1h-),' Kriging ',10(1h-),/,
```

```
     &         ' Dimension: N =',i2,/,
     &         ' Degree of Polynomial:   K  = ',i2,/,
     &         ' Number of Coefficients: M1 = ',i3)

write(2,12)m
12 format(' Original Data  ( ',i5,' pts ):')
write(2,33) zmax, (x(imax,j), j = 1, n)

c --- form solution matrix A of Eq. 26 in Delhomme ---
do 50 i = 1, m

do 30 j = 1, m
do 25 l = 1, n
xi(l) = x(i,l)
xj(l) = x(j,l)
25 continue
a(i,j) = gamma(xi, xj, n)
30 continue

a(i,m+1) = 1
do 40 l = 2, m1
j1 = ifm(l,1)
j2 = ifm(l,2)
k1 = ifm(l,3)
k2 = ifm(l,4)
if(xi(j1).ne.0.and.xi(j2).ne.0) then
  a(i,m+l) = xi(j1)**k1 * xi(j2)**k2
else
  a(i,m+l) = 0
endif
40 continue

50 continue

do 60 j = 1, m
a(m+1,j) = 1
60 continue
```

```
do 70 l = 2, m1
do 70 j = 1, m

do 65 i = 1, n
65 xj(i) = x(j,i)
j1 = ifm(l,1)
j2 = ifm(l,2)
k1 = ifm(l,3)
k2 = ifm(l,4)
if(xj(j1).ne.0.and.xj(j2).ne.0) then
  a(m+l,j) = xj(j1)**k1 * xj(j2)**k2
else
  a(m+l,j) = 0
endif

70 continue

do 80 l = 1, m1
do 80 j = 1, m1
a(m+l, m+j)=0
80 continue

c --- inverse of matrix A ---
c write(*,*)'--- inverse of A ---'
c call inv(m+m1,a,ai)

c --- LU factorization of A ---
write(*,*)'--- LU of A ---'
call lu_a(m+m1,a,ai,iperm,xp)

c --- generate new points ---

read(1,*)
read(1,*)mm
read(1,*)
```

```
do 200 ii = 1, mm

c write(*,*)'new point: ', ii
read(1,*) (x0(jj), jj = 1, n)

c --- form solution vector D of Eq. 26 ---
iz = 0
do 150 i = 1, m

do 155 j = 1, n
xi(j) = x(i,j)
155 continue

c --- check if new pt is a duplicate of given pt
do 160 jj = 1, n
if(x0(jj).ne.xi(jj)) goto 165
160 continue
iz = i
z0 = z(i)
c goto 185

165 continue
d(i) = gamma(xi,x0,n)

150 continue

d(m+1) = 1
do 170 l = 2, m1
j1 = ifm(l,1)
j2 = ifm(l,2)
k1 = ifm(l,3)
k2 = ifm(l,4)
if(x0(j1).ne.0.and.x0(j2).ne.0) then
  d(m+l) = x0(j1)**k1 * x0(j2)**k2
else
  d(m+l) = 0
endif
```

```
170 continue

c --- LU factorization of D ---
call lu_b(m+m1,d,di,iperm,xp)

c --- solve the Eqs.
c call mtp(m+m1,m+m1,1,ai,d,di)
call slv(m+m1,ai,di)

c --- interpolation
z0 = 0
do 180 i = 1, m
z0 = z0 + di(i) * z(i)
180 continue

185 continue
c if(iz.eq.0) then
  write(3,22) (x0(jj), jj = 1, n), z0
c else
c   write(3,22) (x0(jj), jj = 1, n), z0, z(iz)
c endif

if(ii.eq.1) then
  zmax = z0
  do 190 i = 1, n
190   xmax(i) = x0(i)
else if(z0.ge.zmax) then
  zmax = z0
  do 195 i = 1, n
195   xmax(i) = x0(i)
endif

200 continue

22 format(7(1x,1pe10.3))
write(2,31)mm
31 format(' Generated Data ( ',i5,' pts ):')
```

```
write(2,33) zmax, (xmax(i), i = 1, n)
33 format(' Zmax =',1pe10.3,'  Position:',10(1x,e10.3))

close(1)
close(2)
close(3)

stop
end

c**************************************************************************
c  File:        klu.f                                      Dec. 27, 1994   *
c-------------------------------------------------------------------------*
c        Linear System Solver by LU Factorization  *
c                        written by:     Yan     Pan                     *
c                                                                        *
c lu_a: lu factorization of matrix A0 => A *
c lu_b: lu factorization of vector B0 => B *
c       slv: linear solver  --  A * Y = B  (Y - new B)              *
c**************************************************************************
subroutine lu_a(n,a0,a,iperm,xp)

include '/ekofisk/suprid/pan/fit/fit.cm'
dimension a0(np,np),a(np,np),b(np)
dimension iperm(np),xp(np,np)

do 15 i=1,n
iperm(i)=i
do 15 j=1,n
a(i,j)=a0(i,j)
15 continue
c       --- change A to LU ---
kr = 0
do 200 k=1,n
ipiv = k
xmax = abs(a(k,k))
do 100 i=k+1,n
```

```
x = ABS(a(i, k))
IF (x.gt.xmax) THEN
  xmax = x
  ipiv = i
END IF
100     continue
c       ----- check the rank of coefficient matrix -----
IF (xmax.lt.1e - 14) THEN
  write(*,11)n,kr
11        format(1x,'N=',i3,'  Rank(An)=',i3)
stop
END IF
kr = kr + 1
c       ----- pivoting -----
IF (ipiv.ne.k) THEN
  iperm(k)=ipiv
  do 120 j=1,n
  x = a(k, j)
  a(k, j) = a(ipiv, j)
  a(ipiv, j) = x
120       continue
END IF
c       ----- factorization -----
do 150 i=k+1,n
xp(i,k)=a(i, k)/a(k, k)
do 150 j=k,n
a(i,j) = a(i,j)-xp(i,k)*a(k,j)
150     continue
200     continue
return
end

subroutine lu_b(n,b0,b,iperm,xp)

include '/ekofisk/suprid/pan/fit/fit.cm'
dimension b0(np),b(np),iperm(np),xp(np,np)
```

```
do 10 i=1,n
b(i)=b0(i)
10 continue
do 100 k=1,n-1
c --- pivoting ---
ipiv=iperm(k)
x=b(k)
b(k)=b(ipiv)
b(ipiv)=x
c --- factorization ---
do 50 i=k+1,n
b(i)=b(i)-xp(i,k)*b(k)
50 continue
100 continue
return
end

subroutine slv(n,a,b)

include '/ekofisk/suprid/pan/fit/fit.cm'
dimension a(np,np),b(np)

b(n) = b(n) / a(n, n)
do 300 i=n-1,1,-1
x = b(i)
do 250 j=i+1,n
250     x = x - a(i, j) * b(j)
300     b(i) = x / a(i, i)
RETURN
END

c**************************************************************************
c  File:        gama.f                                    Dec. 27, 1994  *
c-------------------------------------------------------------------------*
c       Calculation of Variogram for Kriging Interpolation                *
c                       written by:     Yan     Pan                       *
c                                                                         *
```

```
c relation parameter: h = |xi - xj| *
c       Monomial Model: gama = omega * h**(flamada)                    *
c Gaussian Model: gamma=omega*(1-exp(-h/a)) *
c**********************************************************************
real function gamma(xi,xj,n)

        include '/ekofisk/suprid/pan/fit/fit.cm'
        dimension xi(nd),xj(nd)

c       --- variogram ---
        h=0
        do 50 k=1,n
        h=h+(xi(k)-xj(k))**2
50      continue

c       --- Monomial Model ---
        flamda=1.5
        omega=1
        gamma=omega*h**(flamda/2)

c       --- Gaussian Model ---
c       omega=10000
c       a1=35
c       a2=27
c       cita=3.1415926535*65/180
c       if(xi(1).ne.xj(1)) then
c        alfa=cita+atan((xi(2)-xj(2))/(xi(1)-xj(1)))
c       else
c        alfa=cita+3.1415926535/2
c       endif
c       a=(a1*cos(alfa))**2+(a2*sin(alfa))**2
c       gamma=omega*(1-exp(-h/a))

        return
        end
```

A.2.4 Common Files and Subroutines of the Least Squares and Kriging Algorithms

```
c**************************************************************************
c  File: fit.cm Dec. 27, 1994 *
c-------------------------------------------------------------------------*
c Variable Declaration of Multivariate  Interpolation Program *
c written by: Yan  Pan *
c**************************************************************************
parameter(np = 1500, nd = 10)

c**************************************************************************
c  File: inf.f Dec. 27, 1994 *
c-------------------------------------------------------------------------*
c Generator of the Basis of the KK degree Polynomials  *
c written by: Yan  Pan *
c *
c polynomial fitting function = sum{Ck*(Xi**Ki)*(Xj**Kj)} *
c i,j = 1, .., n;   k = 1, .., m1;   0 <= Ki + Kj <= KK; *
c n -- # of dimensions m1 -- # of coefficients *
c kk -- highest degree of polynomials *
c ifm(m,4) -- information carrier ifm(i,j,ki,kj) *
c**************************************************************************
        subroutine inf(n,kk,ifm,m1)

include '/ekofisk/suprid/pan/fit/fit.cm'
        dimension ifm(np, 4)

c write(2,*)'  j    j1  j2  k1  k2'
c       -- k=0 --
        j = 1
        ifm(j, 1) = 1
        ifm(j, 2) = 1
        ifm(j, 3) = 0
        ifm(j, 4) = 0
c write(2,22)j,(ifm(j,l),l=1,4)
```

```
c       -- k=1 --
        do 100 jj= 1, n
        j = j + 1
        ifm(j, 1) = jj
        ifm(j, 2) = 1
        ifm(j, 3) = 1
        ifm(j, 4) = 0
c write(2,22)j,(ifm(j,l),l=1,4)
100     continue
        IF (kk.eq.1) GOTO 1000

c       -- k>=2 --
        do 900 k = 2, kk, 2
c       --k = even
do 300 j1 = 1, n
j = j + 1
ifm(j, 1) = j1
ifm(j, 2) = 1
ifm(j, 3) = k
ifm(j, 4) = 0
c write(2,22)j,(ifm(j,l),l=1,4)
300 continue
        do 400 k1 = 1, k/2 - 1
        k2 = k - k1
        do 400 j1 = 1, n
        do 400 j2 = 1, n
if(j1.eq.j2) goto 400
        j = j + 1
        ifm(j, 1) = j1
        ifm(j, 2) = j2
        ifm(j, 3) = k1
        ifm(j, 4) = k2
c write(2,22)j,(ifm(j,l),l=1,4)
400     continue
        k3 = k / 2
        do 500 j1 = 1, n - 1
        do 500 j2 = j1 + 1, n
```

```
        j = j + 1
        ifm(j, 1) = j1
        ifm(j, 2) = j2
        ifm(j, 3) = k3
        ifm(j, 4) = k3
c write(2,22)j,(ifm(j,l),l=1,4)
500     continue
        IF (k.eq.kk) GOTO 1000
c       --k = odd
do 700 j1 = 1, n
j = j + 1
ifm(j, 1) = j1
ifm(j, 2) = 1
ifm(j, 3) = k + 1
ifm(j, 4) = 0
c write(2,22)j,(ifm(j,l),l=1,4)
700 continue
        do 800 k1 = 1, k3
        k2 = k + 1 - k1
        do 800 j1 = 1, n
        do 800 j2 = 1, n
if(j1.eq.j2) goto 800
        j = j + 1
        ifm(j, 1) = j1
        ifm(j, 2) = j2
        ifm(j, 3) = k1
        ifm(j, 4) = k2
c write(2,22)j,(ifm(j,l),l=1,4)
800     continue
900     continue
1000    continue
22      format(1x,i4,4(1x,i3))
m1 = j

return
end
```

A.3 Field Development Scheduling Simulation Program

A.3.1 Generator of Input Data

```
c**************************************************************************
c  File:        dis.f                                    Jun.  6, 1995   *
c--------------------------------------------------------------------------*
c               Program of Generating Input Data for mod.f *
c picking up the sample data points obtained by Uniform Design *
c from the exhaustive reference data set *
c *
c                       written by:     Yan Pan                          *
c**************************************************************************
        program dis

parameter(np = 1000, ns = 100, nd = 10)
        real x(np,np),y(np),w(np),xmin(nd),xmax(nd)
dimension is(ns),js(ns),k(nd)
character fn*15

        open(2,file='mod.in',status='unknown')

write(*,*)'Input sample data index filename?'
read(*,'(a)')fn
        open(11,file=fn,status='old')

c -- reference data filename --
read(11,'(a)')fn
open(1,file=fn,status='old')
c -- # of reference data points --
read(11,*)ni,nj
c -- # of sample data points --
read(11,*)m

c -- # of dimensions --
n = 2
```

```
        write(2,5)n,m
5       format(1x,2(i5,1x))
        write(2,*)

c --- Input sample points index: is(mi), js(mj) ---
do l=1,m
read(11,*)js(l),is(l)
enddo

li=0
do 35 j=1,nj

do 10 l=1,m
if(j.eq.js(l)) goto 17
10 continue
do 15 ii=1,ni
15 read(1,*)
goto 35
17 do 34 i=1,ni
if(i.eq.is(l)) then
 li=li+1
 read(1,*)x(li,1),x(li,2)
else
 read(1,*)
endif
34 continue

35 continue

c --- output sample data ---
        do 50 i=1,m
        write(2,11)(x(i,j),j=1,n)
50      continue
11      format(7(1x,1pe10.3))

        close(2)
        end
```

A.3.2 Main Program

```
c************************************************************************
c  File:        mod.f                                    Jun.  7, 1995   *
c  Usage: regular grids: input from keyboard *
c irregular grids: use dis.f to generate mod.in *
c-----------------------------------------------------------------------*
c               Program of Development Scheduling Simulation             *
c                       written by:     Yan Pan                          *
c************************************************************************
program model

parameter(np  =  500)
dimension nw1(np),nw2(np),nrig1(np),nrig2(np)
real q1(np),q2(np),qt(np)
real income(np),cost(np)
real intres,insur,inflat,ti2,deve,prof
integer ifa,ii,i0

open(2,file = 'mod.gc',status = 'unknown')
open(3,file = 'chk',status = 'unknown')
open(5,file = 'rec1',status = 'unknown')
open(11,file = 'rec2',status = 'unknown')

c ----- initial parameters for cash flow -----
nwell = 10
intres = 0.1
inflat = intres
iref = 0
qmax = 15000
qmax = qmax*30
tf1 = 144
decl = 60
qeach = 1500
qeach = qeach*30
price = 30
pltfm = 1.e8
rig = 6.e5
```

```
mobil = 1.e6
drill = 1.4e6
facil = 400
insur = 12500
opera = 5000
handl = 1.
headf = .15
f = .13

c ----- initial data (for irregular grids) -----
c open(1,file = 'mod.in',status = 'old')
c read(1,*)nd,nn
c read(1,*)

c ----- initial data (for regular grids) -----
c write(*,*)'Input ti2_min,ti2_max,n1'
c read(*,*)ti2_min,ti2_max,n1
c write(*,*)'Input deve_min,deve_max,n2'
c read(*,*)deve_min,deve_max,n2
ti2_max = 152
ti2_min = 128
deve_max = 70
deve_min = 5
n1 = 7
n2 = 14
dtk1  =  (ti2_max - ti2_min) / (n1 -1.)
dtk2  =  (deve_max - deve_min) / (n2 - 1.)

c - total time intervals & time step in months -
nt = 360
dt = 1

c   --- cash flow for differect ti2 and deve ---
ymax = 0
x1 = ti2_min
x2 = deve_min
```

```
c --- irregular grids ---
c do 2000 k = 1, nn
c read(1,*)ti2,deve

c --- regular grids ---
do 2000 k1 = 1, n1
ti2 = ti2_min + dtk1 * (k1-1.)
do 2000 k2 = 1, n2
deve = deve_min + dtk2 * (k2-1.)

do 80 i = -50, nt
income(i) = 0
cost(i) = 0
80 continue

c ----- flow rate and income -----
do 100 i = 1, nt
t = dt * real(i)
if(t.gt.tf1 + decl) then
 q1(i) = 0
else
 if(t.lt.tf1) then
  q1(i) = qmax
 else
  q1(i) = qmax * (1. - (t-tf1)/decl)
 endif
endif
if(t.lt.ti2) then
 q2(i) = 0
else
 if(t.gt.ti2 + deve) then
  q2(i) = qmax
 else
  q2(i) = qmax * (t-ti2) / deve
 endif
endif
qt(i) = q1(i) + q2(i)
```

```
if(qt(i).ge.qmax) qt(i) = qmax
income(i) = qt(i) * dt * price
100 continue

c ----- cost -----
do 500 i = 0, nt
t = dt * real(i)

c --- drill cost
if(t.lt.tf1+decl) then
 nw1(i) = nwell
else
 nw1(i) = 0
endif
qe = q2(i)/qeach
kqe = int(qe)
if(qe.gt.kqe) then
 nw2(i) = kqe + 1
else
 nw2(i) = kqe
endif
ndr = nw2(i) - nw2(i-1)
nrig1(i) = ndr/2
nrig2(i) = nrig1(i) + mod(ndr,2)

c --- first rig cost
if(nrig1(i).eq.0) goto 250
imob = i - nrig1(i) * 4
cost(imob) = cost(imob) + mobil + drill
do 200 l1 = i-1, imob+1, -1
cost(l1) = cost(l1) + rig + drill
200 continue

c --- second rig cost
250 continue
if(nrig2(i).eq.0)goto 350
imob = i - nrig2(i) * 4
```

```
cost(imob) = cost(imob) + mobil + drill
do 300 l2 = i-1, imob+1, -1
cost(l2) = cost(l2) + rig + drill
300 continue

c --- insurance isl and operation cost
350 continue
prod = qt(i) * dt
cost(i) = insur*dt + .05 * price * prod * (1-f)
ch = handl * prod
cop = opera * (nw1(i) + nw2(i)) * dt
cost(i) = cost(i) + (1.+headf) * (cop+ch)
500 continue

c --- cost for platform
ii = int((ti2-28)/dt)
cost(ii) = cost(ii) + .3 * pltfm
dpltfm = .7/23 * pltfm
do 600 i = int((ti2-5)/dt), ii+1, -1
cost(i) = cost(i) + dpltfm
600 continue

c --- facilities initial cost
ifa = -int(12/dt)
cost(ifa) = cost(ifa) + facil * qmax/30

c ----- Net Profit over the whole period -----
if((ti2.eq.144.and.deve.eq.60).or.
     & (ti2.eq.174.and.deve.eq.5)) then
  write(3,77)
endif
77 format(/,'time& nw1& nw2&rig1&rig2&',4x,'q1&',8x,'q2&',8x,'qt&',
     &  6x,'income&',6x,'cost\\ \hline')

prof = 0.
i0 = int(min(ifa,ii))
```

```
do 800 i = i0, nt
dprof = (income(i)-cost(i))*(1.+inflat)**(real(iref-i)/12.)
prof = prof + dprof

if(ti2.eq.144.and.deve.eq.60) then
  write(5,81) i,income(i)/1.e6,cost(i)/1.e6,dprof/1.e6,prof/1.e6
endif
if(ti2.eq.174.and.deve.eq.5) then
  write(11,81) i,income(i)/1.e6,cost(i)/1.e6,dprof/1.e6,prof/1.e6
endif
81 format(1x,i4,5(1x,1pe10.3))

if(((ti2.eq.144.and.deve.eq.60).or.(ti2.eq.174.and.deve.eq.5))
     & .and.(i.ge.144.and.i.le.204)) then
        write(3,88)i,nw1(i),nw2(i),nrig1(i),nrig2(i),
     &   q1(i)/30,q2(i)/30,qt(i)/30,income(i)/1.e6,cost(i)/1.e6
88 format(i4,'&',2(1x,i2,'&'),2(2x,i2,'&'),4(1x,1pe9.3,'&'),
     & 1x,e9.3,'\\')
endif

800 continue

if((ti2.eq.144.and.deve.eq.60).or.(ti2.eq.174.and.deve.eq.5)) then
  write(3,66)ti2,deve,prof/1.e6
endif

c if(mod(int(ti2),12).eq.0) then
write(2,99)ti2,deve,prof/1.e6
99 format(3(2x,1pe12.5))
c endif

c --- search for the maximum profit ---
if(prof.gt.ymax) then
  ymax = prof
  x1 = ti2
  x2 = deve
endif
```

```
2000 continue
c ----- end of loop -----

write(3,66)x1,x2,ymax/1.e6
66 format(/,' ti2 = ',f4.0,' months   deve = ',f4.0,
     & ' months   profit = ',1pe10.3,' million dollars')

close(2)
close(3)
stop
end
```

A.4 Waterflooding Simulation Program

A.4.1 Input Data File

```
******************
 file name: in
******************
.01 120  600  600  1  1          -- rw, ns, np, nt, itermax, kw, kpat
9999  1.e-4  80  0.3  1.2        -- itermax, err, maxcvg, factor, flim
1.0  0.1  0.2  0.3               -- price, cost1, cost2, cost3
.10  2.0  0.2  1                 -- rate, time_ref, period, scale
1.0  1.0  1.0                    -- (q1),q2,q3,q4,(q5)
2                                -- # of patterns
1.00 0.00 0.00                   -- (a), b, c, d
stm01 rfw01 tfw01                -- fn1,fn2,fn3
1.00 0.00 0.45
stm02 rfw02 tfw02
```

A.4.2 Main Program

```
c*************************************************************************
c  File: cut.f Apr. 24, 1995 *
c  Usage: link with files: well.f strm.f cita.f pote.f delr.f *
c newx.f newy.f decl.cm *
c-------------------------------------------------------------------------*
c Main Program of Waterflooding Project *
c written by: Yan Pan *
c *
c Calculation : *
c (1)the time of breakthrough *
c (2)water cut (3) net present value (NPV) at given time *
c Condition : *
c (1)isolated/repeated well pattern *
c (2)open/closed homogeneous reservoir (mobility=1) *
c *
c using particle tracking + correction: XO->V->X'->X *
c *
c vx = - d(pote)/dx  vy = - d(pote)/dy  vs = sqrt(vx**2+vy**2) *
```

```
c vx = - d(strm)/dy  vy = + d(strm)/dx *
c BreakThrough Time: tb = integ(phi/vs*ds) {injector=>producer} *
c Injected PV @ BT:   W = tb * qw(1)/(4ab*phi) *
c *
c WaterCut: fw = sum{ d_strm(j) / (qw(j)/h) } j = 1, nw *
c d_strm(j) -- the maximum difference of stream values between *
c the streamlines which reach producer j *
c *
c Cash Flow (NPV): *
c profit = price*q_oil - cost1*q_inj - cost2*q_oil - cost3*q_wat *
c profit = profit * (1+rate) ** ((time_ref - time)/period) *
c price -- price of oil rate, period -- inflation rate,period *
c cost1,2,3 -- cost of injection & production of oil, water *
c**********************************************************************
program cut

include '/ekofisk/suprid/pan/waterfld/decl.cm'
common err,factor,maxcvg
parameter(nj = 720)
real palfa(nj),xp(nj),yp(nj),sc(nj),drp(nj)
real tb(nj),sbt(nw,nj),sq(nj),fw(nw)
dimension nb(nj),nbs(nw)
character*6 fn1,fn2,fn3

data pie/3.1415926535/
data a, qw(1)/1.0, -1.0/

halfpie = pie/2
deg = 180/pie

open(1,file='in',status='old')
c open(2, file='chk', status ='unknown')
open(11,file='opt',status='unknown')

c write(*,*)'iter'
c read(*,*)ikk
```

```
c------ input control parameters
read(1,*)rw,ns,np,nt,kw,kpat
read(1,*)itermax,err,maxcvg,factor,flim
read(1,*)price,cost1,cost2,cost3
read(1,*)rate,tt_ref,period,scale
read(1,*)(qw(l),l = 2, nw-1)

c --- discretization of grids
dalfa = 2.*pie/ns
dx = 1./ np
dt = pie/q/nt

c --- imags for repeated pattern/closed reservoir
img = nw * (2*kw + 1)**2
c write(2,7)img
7 format(' img =',i4)

c --- keep balance of flow rate
if(kpat.eq.0) then
c -- isolated pattern: sum[q_prd] = q_ing
  nprd = 1
  do l = 2, nw - 1
  qw(l) = qw(l) / (nw - 1.)
  end do
else
c -- repeated pattern: sum[q_prd] = q_ing * nprd
  nprd = nw - 1
endif
qw(nw) = - qw(1) * nprd
do l = 2, nw - 1
qw(nw) = qw(nw) - qw(l)
end do

read(1,*)nk
c------ mail loop for each pattern
do 1000 kk = 1, nk
```

```
read(1,*)b,c,d
read(1,17)fn1,fn2,fn3
17 format(3(a5,1x))

c open(3, file=fn1, status ='unknown')
c open(5, file=fn2, status ='unknown')
c open(7, file=fn3, status ='unknown')

c --- set up wells locations and flow rates
call well

c write(5,19)
19 format(/,'  iter   time',4(4x,'#    fw'))

c --- constants --
kp = b / a * 20
tpv = abs(qw(1)) / a / b / 4.
tt1 = tt_ref/2 - .5 * dt * tpv
tt2 = tt_ref/2 + .5 * dt * tpv

c    --- criterions of crossing multivalue streamline
cf = 1./err
tiny = .5 / deg
rr = rw/2.
c write(2,23)
23 format(/,'jumps of stream values at each well:',
     & /,6x,'xw',8x,'yw',8x,'qw',8x,'90',8x,'270')
do l = 1, nw
xc = xw(l) + rr * cos(tiny)
y1 = yw(l) - rr * sin(tiny)
y2 = yw(l) + rr * sin(tiny)
cri = strm(xc,y2) - strm(xc,y1)
x1 = xw(l) + rr * sin(tiny)
x2 = xw(l) - rr * sin(tiny)
yc = yw(l) + rr * cos(tiny)
dstr(l,1) = anint((strm(x2,yc) - strm(x1,yc)-cri)*cf)/cf
dstr(l,2) = abs(- dstr(l,1) - qw(l)/h)
```

```
dstr(l,1) = abs(dstr(l,1))
c write(2,222)xw(l),yw(l),qw(l),dstr(l,1),dstr(l,2)
end do

c --- start to locate streamlines around the injector
alfa0 = 0./deg
do 50 j = 1, ns
alfa = alfa0 +  + dalfa*(j-1.)
palfa(j) = alfa * deg
xp(j) = xw(1) + rw*cos(alfa)
yp(j) = yw(1) + rw*sin(alfa)
sc(j) = strm(xp(j),yp(j))
call delr(xp(j),yp(j),drp(j),nb(j))
tb(j) = 0.
c write(3,222)xp(j),yp(j),sc(j)
50 continue

c --- set initial values
itb = 0
tbm = 99
fwa = 0
q_inj = 0
q_oil = 0
q_wat = 0
profit = 0
c -- optimal point --
tp = tt_ref
fwp = 0
profit_max = -1.e5
c -- starting time --
tt = 0.
write(11,333)b,c,d,tt,profit/scale,tbm,fwa

c------ main loop for each time step
do 800 iter = 1, itermax
tt = tt + dt*tpv
```

```
c------ loop of locating each streamline
do 500 j = 1, ns
if(tb(j).ne.0) goto 500

c --- calculate the local velocity along the streamline
vy = ( pote(xp(j),yp(j)-dx) - pote(xp(j),yp(j)+dx) )/2/dx
vx = ( pote(xp(j)-dx,yp(j)) - pote(xp(j)+dx,yp(j)) )/2/dx

c --- locate the next point on the same streamline
ds = sqrt(vx**2 + vy**2) * dt
x1 = xp(j) + vx*dt
y1 = yp(j) + vy*dt
dr = sqrt((xw(nb(j))-x1)**2 + (yw(nb(j))-y1)**2)
if(ds.ge.drp(j).or.dr.le.rw) then
  xp(j) = xw(nb(j))
  yp(j) = yw(nb(j))
  goto 200
endif

ss = strm(x1,y1)
s_jump = ss - sc(j)
sj = abs(s_jump)

c --- discontinuity of stream value ---
do 100 i1 = 1, nb(j), nb(j) - 1
do 100 i2 = 1, 2

dstrm = dstr(i1,i2)
xx = x1
yy = y1

if(sj.lt.dstrm*factor) then
  cc = sc(j)
else
  cc = sc(j) + sign(dstrm, s_jump)
endif
if(abs(ss-cc).lt.err) goto 150
```

```
if(abs(vx).gt.abs(vy)) then
  call newy(cc,xx,yy,yp(j),dstrm,flim*ds)
else
  call newx(cc,xx,yy,xp(j),dstrm,flim*ds)
endif
if(xx.ne.99.and.yy.ne.99) goto 150

100 continue

do 120 i1 = 1, 2
do 120 i2 = 1, 2

dstrm = abs(dstr(nb(j),i1) - dstr(1,i2))
xx = x1
yy = y1

if(sj.lt.dstrm*factor) then
  cc = sc(j)
else
  cc = sc(j) + sign(dstrm, s_jump)
endif
if(abs(ss-cc).lt.err) goto 150

if(abs(vx).gt.abs(vy)) then
  call newy(cc,xx,yy,yp(j),dstrm,flim*ds)
else
  call newx(cc,xx,yy,xp(j),dstrm,flim*ds)
endif
if(xx.ne.99.and.yy.ne.99) goto 150

120 continue

c --- not converge ---
tb(j) = -1
c write(2,444)iter,palfa(j),vx,vy,x1,y1,sc(j),ss,dstrm,ds
444 format(/,' iter alfa',5x,'vx',6x,'vy',6x,'xx',6x,'yy',6x,'sc',6x
```

```
     & ,'ss',6x,'ds',6x,'dx',/,1x,i4,1x,f4.0,8(1x,f7.3))
goto 500

c --- converge ---
150 xp(j) = xx
yp(j) = yy
c if(iter/kp.eq.real(iter)/real(kp)) write(3,222)xx,yy,sc(j)

c------ calculate the breakthrough time of each streamline
call delr(xp(j),yp(j),drp(j),nb(j))
if(drp(j).le.rw) then
200   continue
  tb(j) = tt

c --- check if it's the 1st BT streamline
  if(itb.eq.0) then
    tbm = tb(j)
    alfam = palfa(j)
    nbm = nb(j)
    itb = 1
  endif

endif

c------ end of locating all streamlines
500 continue

fwa = 0.
do 520 l = 2, nw
520 fw(l) = 0.
if(itb.eq.0) goto 720

c------ calculation of watercut
do l = 2, nw
nbs(l) = 0
end do
```

```
c --- accounting the # of streamlines at each producer
do 600 j = 1, ns

if(tb(j).le.0.) goto 600
nbs(nb(j)) = nbs(nb(j)) + 1

if(nbs(nb(j)).eq.1) then
  sbt(nb(j),nbs(nb(j))) = sc(j)
else
  s_jump1 = sc(j) - sbt(nb(j),1)
  s_jump2 = sc(j) - sbt(nb(j),nbs(nb(j))-1)
  sj1 = abs(s_jump1)
  sj2 = abs(s_jump2)
  if(sj1.lt.sj2) then
s_jump = s_jump1
sj = sj1
  else
s_jump = s_jump2
sj = sj2
  endif
  if(dstr(1,1).ne.0.and.
     &     abs(sj-dstr(1,1)).lt.abs(sj-dstr(1,2))) then
    dstrm = dstr(1,1)
  else
    dstrm = dstr(1,2)
  endif
  if(sj.lt.dstrm*factor) then
    sbt(nb(j),nbs(nb(j))) = sc(j)
  else
    sbt(nb(j),nbs(nb(j))) = sc(j) - sign(dstrm,s_jump)
  endif
endif

c if(iter.eq.ikk)write(2,66)palfa(j),nb(j),sc(j),
c     & sbt(nb(j),nbs(nb(j))),sbt(nb(j),1),s_jump1,s_jump2,s_jump
66 format(1x,f4.0,1x,i2,10(1x,f7.4))
600 continue
```

```
do 700 l = 2, nw
if(nbs(l).eq.0) goto 700

c --- set the streamlines in sequence
sq(0) = -1.e5
do 630 m = 1, nbs(l)
smin = 1.e5
do 620 i = 1, nbs(l)
if(sbt(l,i).lt.smin.and.sbt(l,i).gt.sq(m-1)) smin = sbt(l,i)
620 continue
sq(m) = smin
630 continue

c -- isolated pattern --
if(kpat.eq.0) then
  do i = 2, nbs(l)
  fw(l) = fw(l) + sq(i) - sq(i-1)
  end do
  fwa = fwa + fw(l)

c -- repeated pattern --
else
  do i = 2, nbs(l)
  fw(l) = fw(l) + (sq(i) - sq(i-1)) * real(nprd)
  end do
  fwa = fwa + fw(l) / real(nprd)
endif

700 continue

c------ calculation of net profit
720 continue
dq_inj = - qw(1) * dt
dq_oil = 0.
dq_wat = 0.
do 750 l = 2, nw
```

```
dq_oil = dq_oil + qw(l) * (1-fw(l)) * dt
dq_wat = dq_wat + qw(l) * fw(l) * dt
750 continue

c - repeated pattern -
if(kpat.eq.1) then
  dq_oil = dq_oil / real(nprd)
  dq_wat = dq_wat / real(nprd)
endif

q_inj = q_inj + dq_inj
q_oil = q_oil + dq_oil
q_wat = q_wat + dq_wat
profit = profit +
     & (price*dq_oil - cost1*dq_inj - cost2*dq_oil - cost3*dq_wat) *
     & (1+rate) ** (-tt/period)

c if(iter/kp.eq.real(iter)/real(kp)) then
c       write(5,123)iter,tt,(nbs(l),fw(l), l=2,nw)
c   write(7,222)tt,fwa,profit/scale,q_inj,q_oil,q_wat
c     & ,dq_oil/dt,dq_wat/dt
c endif
123 format(1x,i5,1x,f8.4,4(1x,i3,1x,f6.4))

if(tt.gt.tt1.and.tt.lt.tt2)
     & write(11,333)b,c,d,tt,profit/scale,tbm,fwa

c --- search for the optimal shut-in time ---
if(profit.gt.profit_max) then
  tp = tt
  fwp = fwa
  profit_max = profit
endif

c --- check if all the streamlines reach the producers
c ns_t = 0
c do 790 l = 2, nw
```

```
c ns_t = ns_t + nbs(l)
c790 continue
c if(ns_t.eq.ns) goto 900
if(tt.ge.tt_ref) goto 900

c------ end of one time step
800 continue

900 continue
c write(2,*)
c do 950 j = 1, ns, 6
c950 write(2,109)(palfa(j+l),nb(j+l), l = 0,5)
c109 format(6(2x,f4.0,1x,i2))
c write(5,333)b,c,d,tt,profit/scale,tbm,fwa

write(11,333)b,c,d,tt,profit/scale,tbm,fwa
write(11,333)b,c,d,tp,profit_max/scale,tbm,fwp
333 format(3(1x,f5.2),10(1x,f8.4))

c close(3)
c close(5)
c close(7)

c------ end of one pattern ( for certain values of a,b,c,d )
1000 continue

111 format(1x,i4,1x,f5.0,10(1x,f7.2))
222 format(16(1x,f9.4))
c close(2)
close(1)
close(11)

stop
end
```

```
c**************************************************************************
c  File: well.f Mar.  3, 1995 *
c-------------------------------------------------------------------------*
c Subroutine well of Waterflooding Project *
c written by: Yan Pan *
c *
c set up locations, flow rates of wells & their images with  *
c same sign of flow rates for repeated pattern, closed bdries *
c**************************************************************************
subroutine well
include '/ekofisk/suprid/pan/waterfld/decl.cm'

c------ well positions

xw(1) = c
yw(1) = d
xw(2) = a
yw(2) = b
xw(3) = -a
yw(3) = b
xw(4) = -a
yw(4) = -b
xw(5) = a
yw(5) = -b

c------ flow rates ( production q>0; injection q<0 )

do l = 1, nw
qw(l) = q * qw(l)
end do

if(kw.eq.0) return
c------ images of wells

ll = nw
c --- images of injector ---
```

```
k = 1
do 100 i = -kw, kw
do 100 j = -kw, kw
ll = ll + 1
xw(ll) = i * 2*a + (-1)**i * xw(k)
yw(ll) = j * 2*b + (-1)**j * yw(k)
qw(ll) = qw(k)
if(xw(ll).eq.xw(k).and.yw(ll).eq.yw(k)) ll = ll - 1
100 continue

c --- images of producers ---

do 200 k = 2, nw
if(xw(k).gt.0) then
  fx = 1
else
  fx = -1
endif
if(yw(k).gt.0) then
  fy = 1
else
  fy = -1
endif
do 200 i = 0, 2*kw
do 200 j = 0, 2*kw
ll = ll + 1
xw(ll) = (-1)**i * (xw(k) + fx * 2*a * i)
yw(ll) = (-1)**j * (yw(k) + fy * 2*b * j)
qw(ll) = qw(k)
if(xw(ll).eq.xw(k).and.yw(ll).eq.yw(k)) ll = ll - 1
200 continue
111 format(1x,i4,3(1x,f7.1),3(1x,i4))

return
end
```

```
c**************************************************************************
c  File: strm.f Jan.  7, 1995 *
c-------------------------------------------------------------------------*
c    Function strm(x,y) of Waterflooding Project *
c written by: Yan Pan *
c *
c calculation of the value of stream function at point(x,y) *
c in a homogeneous reservoir with multi-wells *
c *
c Vr = -q/(2pie*r*h) = - 1/r*d(strm)/d(cita) *
c =>  strm = q*cita/(2pie*h) *
c [   production q>0; injection q<0   ] *
c**************************************************************************
real function strm(xx,yy)
include '/ekofisk/suprid/pan/waterfld/decl.cm'

ss = 2*pie*h
strm = 0.
do 50 k = 1, img
strm = strm + cita(xw(k),yw(k),xx,yy)*qw(k)/ss
50 continue
return
end

c**************************************************************************
c  File: cita.f Mar.  9, 1995 *
c-------------------------------------------------------------------------*
c Function cita(x1,y1,x2,y2) of Waterflooding Project *
c written by: Yan Pan *
c *
c calculation of the angle between 2 pts *
c (x1,y1) -- position of well (x2,y2) -- any point  *
c - 0.5pie <= cita <= 1.5pie *
c**************************************************************************
real function cita(x1,y1,x2,y2)
common /const/pie
```

```
dy = y2 - y1
dx = x2 - x1
if(dx.gt.0) then
  cita = atan(dy/dx)
else if(dx.lt.0) then
  cita = atan(dy/dx) + pie
else if(dx.eq.0.and.dy.eq.0) then
  cita = 0.
else if(dy.gt.0) then
  cita = .5 * pie
else
  cita = -.5 * pie
endif
return
end

c**************************************************************************
c  File: pote.f Feb. 15, 1995 *
c-------------------------------------------------------------------------*
c    Function pote(x,y) of Waterflooding Project *
c written by: Yan Pan *
c *
c calculation of the value of potential function at point(x,y) *
c in a homogeneous reservoir with multi-wells *
c *
c Vr = -q/(2pie*r*h) = - d(pote)/dr *
c =>  pote = q/(2pie*h)*ln(r) *
c [   production q>0; injection q<0   ] *
c**************************************************************************
real function pote(xx,yy)
include '/ekofisk/suprid/pan/waterfld/decl.cm'
ss = 2*pie*h
pote = 0.
do 50 k = 1, img
50 pote = pote + .5*log((xx-xw(k))**2+(yy-yw(k))**2)*qw(k)/ss
return
end
```

```
************************************************************************
c  File:        delr.f                                  Nov. 15, 1994    *
c-----------------------------------------------------------------------*
c       Subroutine delr(xx,yy,dr) of Waterflooding Project          *
c                        written by:     Yan Pan                        *
c                                                                       *
c       calculation of the distance between water front & producers *
c & identification of the producer at which the streamline ends  *
c************************************************************************
        subroutine delr(xx,yy,dr,nr)

        include '/ekofisk/suprid/pan/waterfld/decl.cm'
real ddr(nw)

dr = 1.e9
do 50 l = 2, nw
ddr(l) = (xx-xw(l))**2+(yy-yw(l))**2
if(dr.gt.ddr(l)) then
  dr = ddr(l)
  nr = l
endif
50 continue
        dr = sqrt(dr)

return
end
```

```
c**************************************************************************
c  File: newx.f Apr.  9, 1995 *
c-------------------------------------------------------------------------*
c Subroutine newx(c,y,x,yi,dstrm,dlim) of Waterflooding Project *
c written by: Yan Pan *
c *
c calculation of the coordinates along a stream line  *
c  applying quasi-Newton's Method *
c c -- value of stream function yi -- initial estimation *
c x -- given coordinate   y  -- unknown coordinate *
c dstrm -- discont. stream value  dlim -- constraint of dy *
c**************************************************************************
subroutine newx(cc,yy,xx,yi,dstrm,dlim)
common err,factor,maxcvg

iter = 0
y1 = yy
f1 = strm(y1,xx) - cc
y0 = yi
f0 = strm(y0,xx) - cc
if(dstrm.ne.0.and.abs(f0).ge.dstrm*factor) then
  f0 = f0 - sign(dstrm,f0)
endif
if(abs(f0).lt.err) then
  yy = y0
  return
endif

100 iter = iter + 1
if(f1.ne.f0) then
  dy = -f1/(f1-f0)*(y1-y0)
else
  dy = (y0 - y1)*.5
endif
dy = min(dy,dlim)
dy = max(dy,-dlim)
yc = y1 + dy
```

```
fc = strm(yc,xx) - cc
if(dstrm.ne.0.and.abs(fc).ge.dstrm*factor) then
  fc = fc - sign(dstrm,fc)
endif

c write(2,111)iter,y0,y1,yc,f0,f1,fc
111 format(' newx ',i3,10(1x,f10.4))

if(f1*fc.ge.0) then
  if(f1+fc.ne.0) then
    z = f1/(f1+fc)
  else
    z = .5
  endif
  f0 = z*f0
  y1 = yc
  f1 = fc
else
  y0 = y1
  f0 = f1
  y1 = yc
  f1 = fc
endif
if(abs(dy).ge.err.or.abs(fc).ge.err.and.iter.lt.maxcvg) then
  goto 100
else if(iter.ge.maxcvg) then
c   write(2,99)' f=',f1,' y=',y1,'yi=',yi,'yy=',yy,'ds=',dstrm
  yy = 99
else
c write(2,22)iter,yy,xx,f1
  yy = y1
endif
99 format(10(1x,a3,1x,f8.4))
22 format(1x,i2,8(1x,f9.4))

return
end
```

```
c**************************************************************************
c  File: newy.f Apr.  9, 1995 *
c-------------------------------------------------------------------------*
c Subroutine newy(c,x,y,yi,dstrm,dlim) of Waterflooding Project *
c written by: Yan Pan *
c *
c calculation of the coordinates along a stream line  *
c  applying quasi-Newton's Method *
c c -- value of stream function yi -- initial estimation *
c x -- given coordinate   y  -- unknown coordinate *
c dstrm -- discont. stream value dlim -- constraint of dy *
c**************************************************************************
subroutine newy(cc,xx,yy,yi,dstrm,dlim)
common err,factor,maxcvg

iter = 0
y1 = yy
f1 = strm(xx,y1) - cc
y0 = yi
f0 = strm(xx,y0) - cc
        if(dstrm.ne.0.and.abs(f0).ge.dstrm*factor) then
          f0 = f0 - sign(dstrm,f0)
        endif
if(abs(f0).lt.err) then
  yy = y0
  return
endif

100 iter = iter + 1
if(f1.ne.f0) then
  dy = -f1/(f1-f0)*(y1-y0)
else
  dy = (y0 - y1)*.5
endif
dy = min(dy,dlim)
dy = max(dy,-dlim)
yc = y1 + dy
```

```
fc = strm(xx,yc) - cc
if(dstrm.ne.0.and.abs(fc).ge.dstrm*factor) then
          fc = fc - sign(dstrm,fc)
        endif

c write(2,111)iter,y0,y1,yc,f0,f1,fc
111 format(' newy ',i3,10(1x,f10.4))

if(f1*fc.ge.0) then
  if(f1+fc.ne.0) then
    z = f1/(f1+fc)
  else
    z = .5
  endif
  f0 = z*f0
  y1 = yc
  f1 = fc
else
  y0 = y1
  f0 = f1
  y1 = yc
  f1 = fc
endif
if(abs(dy).ge.err.or.abs(fc).ge.err.and.iter.lt.maxcvg) then
  goto 100
else if(iter.ge.maxcvg) then
c   write(2,99)' f=',f1,' y=',y1,'yi=',yi,'yy=',yy,'ds=',dstrm
  yy = 99
else
c write(2,22)iter,xx,yy,f1
  yy = y1
endif
99 format(10(1x,a3,1x,f8.4))
22 format(1x,i2,8(1x,f9.4))

return
end
```

```
c*************************************************************************
c  File: decl.cm Mar.  9, 1995 *
c-------------------------------------------------------------------------*
c    Variable Declaration of Waterflooding Project *
c written by: Yan Pan *
c *
c nw -- # of wells rw -- well radius *
c q  -- flow rate  h  -- reservoir depth *
c kw -- # of images of one well at each side
c img -- # of all the sources & sinks in the repeated pattern *
c *
c a,b -- geometry parameters of the producers *
c c,d -- geometry parameters of the injector *
c *
c xw,yw,qw -- positions & flow rates of wells *
c dstr(nw,2) -- two jumps of stream values at 90 & 270 degree *
c *
c pie = 3.1415926535 am,bm -- boundary constrains *
c*************************************************************************
parameter(nw = 5, q = 1, h = 1, nn = 1000)
common
     & /vari/a,b,c,d
     & /wells/rw,kw,img,xw(nn),yw(nn),qw(nn),dstr(nn,2)
     & /const/pie
```