Numerical Simulation of Pattern Miscible Displacement in a Heterogeneous Formation with Varied Well Position

Elhassan Abdalla

Submitted in Partial Fulfillment of the Requirement for the Degree of Master of Science in Petroleum Engineering

New Mexico Institute of Mining and Technology
Socorro, New Mexico
December, 1990

PRRC LIBRARY COPY
Abstract

In enhanced oil recovery schemes where an expensive fluid is normally injected underground to displace crude oil, there has been a need to discern the effect of the spatial statistical structure of heterogeneous formations on reservoir recovery characteristics. Synthesized permeability fields were used in solving the governing partial differential equation of miscible displacement at unit mobility ratio assuming two-dimensional steady incompressible flow in a five-spot quadrant. The numerical model was formulated using both the finite difference and finite element methods. The permeability distribution is obtained using either a Monte-Carlo type approach or the Discrete Fourier Transform method. Because of the geometrical effect in pattern displacement, the relative location of heterogeneities may influence the pattern sweep efficiency. An ensemble of subrealizations obtained by translating the pattern boundaries of a base realization within a larger segment of a heterogeneous formation has been used to study the effect of boundary translation on recovery characteristics. The magnitude of boundary displacement is such that the regional influence for which the assumptions of stationarity hold is not exceeded. The effect of the magnitude of the heterogeneity correlation scale on pressure distribution has been separately investigated.
Acknowledgements

The author is much indebted to Dr. J.P. Heller for his supervision of this research project and his continual encouragement. The guidance and assistance of Dr. Allan Gutjahr and Dr. Steven Schaffer are gratefully acknowledged. Appreciation is expressed to Dr. Stefan Miska and Dr. John Griffin for their valued suggestions.

This research project would not have been possible without the interest and financial support of the U.S. Department of Energy, the New Mexico Research and Development Institute, and the consortium of oil companies sponsoring the PRRC research project "Improvement of CO₂ Flood Performance."

Gratitude is due to Mrs. Catherine Lambert and to several others who undertook the task of typesetting the manuscript.
# Table of Contents

Abstract .................................................................................................................. ii  
Acknowledgements .............................................................................................. iii  
List of Tables .......................................................................................................... v  
List of Figures ......................................................................................................... vi  
Chapter 1. Introduction ......................................................................................... 1  
Chapter 2. Literature Review ................................................................................ 4  
Chapter 3. Formulation of the Model ................................................................... 23  
  3.0 Problem Statement ......................................................................................... 23  
  3.1 Derivation of the Governing Equation ............................................................ 25  
  3.2 Generation of the Subrealizations of Random Permeability Fields .......... 30  
    3.2.1 The Source Point Method ...................................................................... 30  
    3.2.1.1 Generalization of Subrealizations using SPM ................................. 32  
    3.2.2 Spectral Method Algorithm ................................................................. 34  
  3.3 Solution of the Flow Equation ....................................................................... 41  
    3.3.1 Finite Difference Formulation ............................................................... 41  
    3.3.1.1 Pressure Approach ......................................................................... 41  
    3.3.1.2 Stream Function Approach ............................................................ 46  
    3.3.2 Finite Element Formulation .................................................................. 50  
    3.3.2.1 Pressure Approach ......................................................................... 50  
    3.3.2.2 Stream Function Approach ............................................................ 63  
  3.4 Velocity Field, Streamlines, and Front Tracing Procedure ......................... 64  
  3.5 Numerical Model Testing ............................................................................... 67  
Chapter 4. Results and Discussion ...................................................................... 78  
  4.1 Variation of Recovery Characteristics with Boundary Translation .......... 78  
  4.2 Effect of Heterogeneity Characteristics on Pressure Distribution ........... 96  
Chapter 5. Conclusions and Recommendations .................................................. 99  
Appendix A ............................................................................................................. 101  
Appendix B ............................................................................................................. 122  
References ............................................................................................................. 135
List of Tables

<table>
<thead>
<tr>
<th>Table 4.1</th>
<th>Means of the Permeability Distribution</th>
<th>82</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table 4.2</td>
<td>Subrealizations Recovery History</td>
<td>90</td>
</tr>
<tr>
<td>Table 4.3</td>
<td>Input Data and Results of the Spectral Method</td>
<td>97</td>
</tr>
</tbody>
</table>
## List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1(a)</td>
<td>Covariance Function</td>
<td>6</td>
</tr>
<tr>
<td>2.1(b)</td>
<td>Semi-variogram Function</td>
<td>6</td>
</tr>
<tr>
<td>2.2</td>
<td>Definition of Porosity and Representative Elementary Volume</td>
<td>7</td>
</tr>
<tr>
<td>3.1</td>
<td>The Five-Spot (Homogeneous Medium) Flood Network</td>
<td>28</td>
</tr>
<tr>
<td>3.2</td>
<td>Source Points &amp; Base Realization Boundaries</td>
<td>33</td>
</tr>
<tr>
<td>3.3</td>
<td>Bilinear Interpolation Function</td>
<td>58</td>
</tr>
<tr>
<td>3.4</td>
<td>Exact and Approximate Pressure Profile Along Diagonal Streamline Finite Difference Solution</td>
<td>68</td>
</tr>
<tr>
<td>3.5</td>
<td>Exact and Approximate Pressure Profile Along Diagonal Streamline Finite Element Solution, 32x32 Grid</td>
<td>69</td>
</tr>
<tr>
<td>3.6</td>
<td>Exact and Approximate Pressure Profile Along Diagonal Streamline Finite Difference (64x64 Grid) vs. Finite Element Solution (40x40 Grid)</td>
<td>71</td>
</tr>
<tr>
<td>3.7</td>
<td>Orthogonal Curvilinear Grid (Upper Octant) (26x27 Grid)</td>
<td>72</td>
</tr>
<tr>
<td>3.8</td>
<td>Exact and Approximate Pressure Profile Along Diagonal Streamline Finite Element (26x27 Grid)</td>
<td>73</td>
</tr>
<tr>
<td>3.9</td>
<td>Uniform Medium Flownet</td>
<td>74</td>
</tr>
<tr>
<td>3.10</td>
<td>Uniform Medium Isochrones</td>
<td>75</td>
</tr>
<tr>
<td>3.11</td>
<td>Uniform Medium Streamlines (Stream Function Solution)</td>
<td>77</td>
</tr>
<tr>
<td>4.1</td>
<td>Relative Injector Location in the Different Subrealizations</td>
<td>80</td>
</tr>
<tr>
<td>4.2</td>
<td>Variogram Plot for Base Realization</td>
<td>81</td>
</tr>
<tr>
<td>4.3</td>
<td>Flownet for Base Realization</td>
<td>83</td>
</tr>
<tr>
<td>4.4</td>
<td>Flownet for Subrealization 5</td>
<td>85</td>
</tr>
<tr>
<td>4.5</td>
<td>Flownet for Subrealization 14</td>
<td>86</td>
</tr>
<tr>
<td>4.6</td>
<td>Isochrones for Base Realization</td>
<td>87</td>
</tr>
<tr>
<td>4.7</td>
<td>Isochrones for Subrealization 5</td>
<td>88</td>
</tr>
<tr>
<td>4.8</td>
<td>Isochrones for Subrealization 14</td>
<td>89</td>
</tr>
<tr>
<td>4.9</td>
<td>Recovery History Plots</td>
<td>92</td>
</tr>
</tbody>
</table>
Chapter 1. Introduction

Adequate management of a reservoir requires its accurate description and knowledge of its permeability characteristics. The engineer can then confidently predict the effect of permeability variability on the pressure distribution and ultimately on recovery. Despite the recognition of the statistical nature of reservoir rock heterogeneity, the conventional practice of petroleum reservoir modeling is deterministic in nature and does not discern the spatial variability of permeability as a factor in its own right with, for instance, the lateral variation in permeability being largely understated. Fluid flow laboratory-scale models of porous materials are based on uniform pore geometry, whereas the true porous structure of sedimentary media is spatially random and can rarely be regarded as homogeneous over any length of any field significance.

The aim of this thesis is to investigate the statistical behavior of a heterogeneous but physically isotropic formation in which the resident fluid is being displaced by another fluid miscible with it, employing pattern displacement. A two-dimensional random permeability field was synthesized using either a Monte-Carlo type model or the spectral method. The resultant governing partial differential equation, assuming steady state, was then solved numerically. The effect of the degree of heterogeneity and its correlation scale can be simulated by generating a sufficiently large ensemble of realizations; however, because of the added geometrical effect in pattern displacement, the relative location of heterogeneities may influence the sweep efficiency. Hence, it is of particular interest to investigate the effect of translating the boundaries of a representative element of pattern within a larger segment of a single realization of a random permeability field. This will enable generating subrealizations having fixed parameters of degree of heterogeneity and correlation scale. The magnitude of the displacement of the element boundaries would be such that the regional influence, for which the assumptions of stationarity hold, is not exceeded.
While the point-to-point rock properties exhibit wide variation, none of the macroscopic derived properties as permeability show significant changes within a characteristic length peculiar to each formation. The variation is, however, not completely random. Two features manifest themselves as departure from randomness: the occurrence of hills and valleys in the mapped surface of the macroscopic property and the correlation between neighboring points in space that may be expressed as the spatial autocovariance.

The variation is also not discrete, but the correlation from block to block diminishes slowly with distance. This accords with the process of the deposition cycle itself which is an evolutionary and continuous process. Because of the complexity of the porous structure, no formulation based on all the details the geologic microscopic properties would be mathematically tractable. Facies models that conceptually reconstruct the history of deposition and diagenesis provide only a mental picture to the practicing engineer. Many natural processes, however, can be viewed as realizations of a stochastic process that is described statistically by a synthesized variable whose behavior is analogous to that of the actual variable. The overall behavior of such a process in a large number of experiments can be predicted with a high degree of certainty by calculations performed on a digital computer.

By viewing the sedimentary porous medium as a realization of a stochastic process, the probabilistic methods (Monte-Carlo, spectral) account quantitatively for the covariance behavior of the generated permeability field, or equivalently, its heterogeneity correlation scale. Seen as a stochastic process, the permeability fields generated by the foregoing methods possess the property of stationarity. The physical meaning of stationarity is that the expected value of a macroscopic property remains constant in different subregions within a larger zone of influence and that the spatial covariance of this property can be approximated by the same function in the different subregions. The degree of heterogeneity and its correlation scale may conceivably
affect such factors as the time-to-breakthrough, sweep efficiency, and the pressure distribution characteristics. These parameters consequently would have far-reaching effects on production scheduling and sizing of production facilities.

Numerical simulation of fluid flow in a heterogeneous formation, similar to that in an inhomogeneous fluid-system, poses its own difficulty. One such attendant problem is that of singularity in the mathematical solution at the corners where the source and sink exist. This problem pertains to the finite difference rectilinear formulation. The singularity affects the accuracy of the solution near the wellbore and will possibly be minimized by excessive mesh refinement, however, that may render the scheme inefficient. Alternatively, a finite element formulation on orthogonal curvilinear co-ordinates using a non-uniform mesh would eliminate the singularity; however, some of the probabilistic models that generate the random permeability field are normally based on uniform grid spacing.
Chapter 2. Literature Review

A sedimentary porous medium is a result of complex cycles of the geologic processes of weathering to produce particulate residue and dissolved constituents, transportation to sedimentary basin, and deposition in a particular environment. Subsequently, deposits are altered by diagenesis to produce lithified sedimentary rock with different characteristics than the unaltered medium. Diagenesis is a result of reworking in the sediments caused by physical and chemical reactions in response to the overburden pressure and increasing temperature downward.

Two broad classifications of depositional environment are used in reservoir description. These are continental environments and marine environments that are characterized by different texture, structural, and composition properties, and which form distinctive beds of sedimentary rocks referred to as facies. In a continental environment, because most particulates are composed in part of silica and were transported to the depositional basin, the sediments are referred to as siliciclastic. Some siliciclastic structures yield directional trend as a result of paleocurrent patterns. The trend may be measurable in spite of the scatter in the data, as in a modern meandering river deposits. Other structures show strongly marked randomness in their lithofacies properties as in a braided river system which is characterized by numerous channels separated by bars (small islands). A braided river system normally occurs where water discharge is high and the stream is overloaded with sediments leading to rapid deposition and erosion of banks; repeated channel branching generates such a braided network of bars of varying lithofacies.

In contrast to siliciclastic deposits, marine environment formations are a result of a topographic buildup that created a barrier to incoming waves; formation constituents are mainly carbonate sediments. Diagenesis is more intensive due to a combination of physical, chemical as well as biological processes. The spatial distribution of rock properties show more
irregularities than siliciclastic deposits; however, the lack of directional trend would suggest a weak spatial autocovariance.

Boggs (1987) reported that environmental interpretation would be improved if facies association and sequences are studied rather than individual facies in which case attention must be given to the nature of facies boundaries and the degree of randomness and non-randomness of the sequences themselves. These factors have bearing on the interpretation of the depositional and post-depositional conditions. On lateral and vertical facies association, Boggs added that such association is a valuable criterion for environmental discrimination. The concept involved is an application of the stratigraphic principle known as Walther's Law.

The irregularity in rock properties is termed "formation heterogeneity." Associated with heterogeneity, in reservoir quantitative description, is a characteristic length referred to as the correlation scale. While, on the core dimensions, the correlation scale may be related to the pore size, the scale of interest on the inter-well level would be of the order of the simulation-block linear dimensions. Reservoir heterogeneity may be structured or random. Structured heterogeneity covers such features as anisotropy or flow discontinuities, whereas random heterogeneity refers to lateral and vertical variation in rock properties caused by the nature of deposition and post-depositional processes.

When a derived rock property like permeability is described by a probability density function, \( f(k) \), the ordered set \( \{k(i,j)\} \) in space where \( \{i, j=1, 2, ...\infty\} \) is called a stochastic process. Simulating a stochastic process, either theoretically or experimentally, results in the generation of a realization sometimes called a sample path; a sufficiently large number of realizations constitutes an ensemble. The more general definition of a stochastic process allows different probability density functions at each point and hence requires the joint density function,
The statistical process is homogeneous and is said to be second order stationary when the probability density function is the same for all points. For a realization $V(x; \alpha)$, where $\alpha$ denotes randomness, stationarity requires:

(i) $E[V(x; \alpha)] = \mu$ for all $x$, that is, the expectation of $V(x; \alpha)$ is constant and is given by the arithmetic mean, $\mu$.

(ii) $C(x, x + \xi) = C(\xi) = E[(V(x; \alpha) - \mu)(V(x + \xi; \alpha) - \mu)]$

defines the autocovariance function for two pairs of points which are $\xi$ apart in space. That is, $C(\xi)$ is assumed to be a function of the separation distance $\xi$ (the lag) and is independent of the location $x$.

Another related function is that of the semi-variogram defined by

$$\gamma(\xi) = \frac{1}{2} E \left[ (V(x + \xi) - V(x))^2 \right]$$

$$= \frac{1}{2} \left[ V(x + \xi)^2 - E[V(x + \xi)V(x)] + \frac{1}{2} E[V(x)^2] \right]$$

$$= C(0) - C(\xi)$$ (2.1)
where \( C(0) \) is the magnitude of the variance given by \( C(\xi) \) at zero lag. Equation 2.1 shows that the semi-variogram is the complementary function to the autocovariance function as illustrated in Figure 2.1. The semi-variogram approaches an asymptotic value defined as the sill and is given by the magnitude of the variance.

In higher dimensions, \( \xi \) is a vector that may have a characteristic orientation relative to an arbitrary co-ordinate system. If the porous medium is physically isotropic, the function defining \( \xi \) will be the same in all directions and \( \xi \) reduces to a scalar.

The governing equations of fluid flow in porous media are founded on the continuum approach. Essential to the concept of continuum is the notion of representative elementary volume introduced by Bear (1968). When a porous medium is heterogeneous with, for example, the porosity varying in space, an upper limit of the length dimension of the representative elementary volume should be a characteristic length that indicates the rate at which changes in porosity takes place. The lower limit of the length is related to the size of the pore. This is illustrated graphically in Figure 2.2, where the dependent variable \( (\phi_i = \Delta V_l / \Delta V) \) is the ratio of void volume to the sample bulk volume as a function of the sample volume, \( \Delta V \).

![Figure 2.2 Definition of Porosity and Representative Elementary Volume.](image-url)
Two sub-domains are identified in Figure 2.2: the first of which is unstable due to insufficiency of sample size, $\Delta V$, in which case, the experimental error will dominate; and a stable portion of the curve in which the fluctuation has diminished to a small amplitude reflecting only the random variation of the pore size and is given by $\Delta V'$ which is the lower limit of the representative elementary volume. Bear and Braester (1972) defined the characteristic length for a given property, $P$, of the porous medium by

$$L_i = \frac{P}{\frac{\partial P}{\partial x_i}} = \frac{1}{\left(\frac{\partial \ln P}{\partial x}\right)}$$

(2.2)

where $L_i$ is the upper limit of the magnitude of $L$ in the $i^{th}$-direction of the elementary volume. The consequences of this definition is that the size of the representative elementary volume should be beyond the range of microscopic effects, but less than a characteristic scale of heterogeneity.

When the reservoir is dominated by macroscopic formation irregularities, the underlying assumptions in the conventional theory of petroleum reservoir analysis limit its accuracy in modeling reservoir behavior. Because of the complexity of the factors involved in the microscopic geologic properties, no tractable field-scale model based directly on them is possible. Facies models that conceptually reconstruct the history of deposition and diagenesis provide, on one hand, a method of simplifying, categorizing, and interpreting data that otherwise appear complex and random; a probabilistic model, on the other hand, offers a trustworthy quantitative method of solution.
The Monte-Carlo type model and spectral methods, when applied to generation of random fields, have central in their formulation the autocovariance function, \( C(\xi) \). When this function expresses the product moment between neighboring points, it is of order one, and the order increases as more surrounding points are considered. A related, sometimes simpler, function is the spectral density, \( S(k) \), which is the Fourier transform of \( C(\xi) \) and is given by

\[
S(\omega) = \frac{1}{2\pi} \int_{-\infty}^{\infty} e^{-i\omega \xi} C(\xi) d\xi
\]

\[
= \frac{1}{\pi} \int_{0}^{\infty} \cos(\omega \xi) C(\xi) d\xi
\]

where \( \xi \), as defined earlier, is the lag distance and \( \omega \) is the wave number. In practice, the discrete Fourier transform is used; for an \( N^{th} \)-order sequence \( C(\xi) \), we have

\[
S(\omega) \equiv \frac{1}{N} \sum_{j=-N}^{N-1} C(\xi) e^{-2\pi i j \Delta \xi \xi / N}
\]

where \( \xi = j \Delta \xi \) and \( N \Delta \xi \) is the maximum lag.

Equation (2.4) is referred to as the discrete Fourier transform whose inverse is given by the discrete form of the autocovariance function

\[
C(\xi) \equiv \sum_{j=-N}^{N-1} S(\omega) e^{2\pi i j \Delta \omega \xi / N}
\]

where \( \omega = j \Delta \omega \) and \( N \Delta \omega \) is the maximum lag.
The discrete Fourier transform is a linear operator that maps an $N$th-order sequence in the time domain to another $N$th-order sequence in the frequency domain. The operator possesses the property of reciprocity defined by

\[
\mathcal{F}\{x(k)\} = \mathcal{F}\{X(n)\}
\]

In the complex field of numbers, $C(\omega)$ may also be written in terms of the complex conjugate,

\[
[C(\omega)]_1^{\ast} = \mathcal{F}\{C(\omega)\} = \mathcal{F}\{C(\omega)\}^{\ast}
\]

By the shifting theorem, it is possible to compute the Fourier transform or its inverse with the same computer code. A third applicable property of the Fourier transform is that if $I(x, y)$ has a Fourier transform given by $F(u, v)$,

\[
I(x-a, y-b) = \mathcal{F}\{-F(u, v)\}
\]

which has a Fourier transform $F(n) = \mathcal{F}\{C(n)\}$.

A variant of the Fourier transform method is the Spectral Representation Theorem which can be applied to a second order stationary stochastic process $Y(x)$ that is correlated in the spatial domain. The theorem defines a unique complex stochastic process $Z(\xi)$ in the wave number domain whose components are uncorrelated and are related to the spectral distribution.

In the complex field of numbers, $C(\omega)$ may also be written in terms of the complex conjugate,
function via the Fourier-Stieltjes integral. Gutjahr (1989) presented a procedure that uses the spectral representation theorem to generate random fields in one and higher dimensions. The algorithm may take as input either the spectral density or the autocovariance function for a prescribed class of functions and may apply to either isotropic or anisotropic medium. The theorem states:

If \( V(\vec{x}) \) is a statistically homogeneous random field that is zero-mean and has a spectral density \( S(\vec{\omega}) \), then there exists a unique (with probability one) complex stochastic process \( Z(\vec{\omega}) \) with the following properties:

\begin{align*}
(i) & \quad V(\vec{x}) = \int_{-\infty}^{\infty} e^{i\vec{\omega} \cdot \vec{x}} dZ(\vec{\omega}) \\
(ii) & \quad E[dZ(\vec{\omega})] = 0 \\
(iii) & \quad E[dZ(\vec{\omega}_1), Dz(\vec{\omega}_2)] = \begin{cases} 0 & \text{if } \vec{\omega}_1 \neq \vec{\omega}_2 \\ S(\vec{\omega})d\vec{\omega} & \text{if } \vec{\omega}_1 = \vec{\omega}_2 = \vec{\omega} \end{cases}
\end{align*}

where \( \vec{\omega} \) is the wave number vector. The matrix \( dZ(\vec{\omega}) \) is the equivalent of \( S(\vec{\omega})d\vec{\omega} \) in the ordinary Fourier transform.

McKay (1988) applied the Spectral Representation Theorem to generate one- and two-dimensional random fields and studied the associated spatial covariance behavior.

Methods of the discrete Fourier transform are normally formulated on evenly spaced grid points to correspond to uniform multiples of wave numbers. The grid points on which the problem is implemented corresponds to sampling of the true continuous population. When the
sampling interval does not account for the smallest significant wave number, distortion of information is likely to occur through the phenomenon of aliasing; moreover in reality, the fundamental wavelength of the data is generally an unknown.

Alternative to studying the spatial autocovariance behavior through the discrete Fourier transform is a Monte-Carlo type approach to the problem in which the equivalent population sampling is performed at random. Monte-Carlo simulation is normally a scheme employing random numbers that are uniformly distributed \([U(0, 1)]\) which can be used for solving a certain class of stochastic or deterministic problems. As was originally conceived, the Monte-Carlo method aimed at solving a complex deterministic system by formulating the equivalent stochastic process, wherein changing the parameters of the random numbers corresponded to changes of the physical system parameters and an estimate of the original solution would be obtained through sampling from the stochastic process (Brandt, 1976).

A simplified example is the evaluation of an ordinary integral:

$$\theta = \int_0^1 f(x) \, dx$$

and if \([X_1, X_2, \ldots, X_n]\) is a random sample \(-U(0, 1)\), then the estimator

$$\hat{\theta} = \frac{1}{n} \sum_{i=1}^{n} f_i(X)$$
is an unbiased estimator of $\theta$ (Kennedy and Gentle, 1980). The idea can be applied to the solution of the more general linear operator equations including multiple integrals that are not analytically tractable.

If $f(X)$ is the probability density function from which the random variable values are to be generated and $F(X)$ is the corresponding probability distribution function, then in order to generate a random value for the stochastic variable $X$, a uniform random number, $r$, $0 \leq r \leq 1$, is produced and $X$ is obtained via the transformation

$$F^{-1}(r) = X$$

which requires the use of an inverse probability distribution function the solution of which may be mathematically complicated; the inverse probability distribution function has been solved for a wide class of probability density functions that include the log-normal distribution density and are installed in the mathematics library of many computer systems.

Law (1944) reported that most reservoirs have permeability data that fit the log-normal probability function given by

$$f(x) = \frac{100}{\sigma \sqrt{2\pi}} \exp \left[ -\frac{1}{2} \left( \frac{\log_a X - \log_a X_m}{\sigma} \right)^2 \right]$$

where
14

\[ a = \text{any arbitrary logarithmic base} \]
\[ X_m = \text{geometric mean permeability} \]
\[ \sigma = \text{standard deviation} \]

Loudon (1979) reported that many geological processes operate in a multiplicative (or so-called geometrical) manner. The degree of mineralization of a rock body may decrease exponentially with distance from the source of mineralization. The size of sedimentary particles which a stream can carry varies in proportion to the stream velocity. Loudon added that it would be likely that many geological distributions follow nearly the log-normal distribution.

When the data follow closely the log-normal distribution a simple approach to the evaluation of the integral is the application of the Central Limit Theorem which is stated as follows:

The distribution of the sums of random samples from any distribution with finite mean and variance approaches the standard normal distribution as the individual sample size increases.

A corollary of the theorem can be stated as

The distribution of random samples approaches log-normal distribution as the individual sample size increases.

Cardwell and Parsons (1945) investigated the practical problem of estimating a single equivalent permeability for a heterogeneous reservoir or portion thereof to that of a homogeneous segment of the same dimensions that would pass the same flux under the same
pressure drop. They showed that for a reservoir divided into homogeneous blocks, the equivalent permeability lies between a harmonic and an arithmetic average of the actual block permeabilities weighted volumetrically. For radial flow, they showed that the equivalent permeability of the blocks lies between a volume-harmonic and a volume-arithmetic average weighted as the inverse square law of their distances from the axis of the well and that, for the permeabilities in the region immediately surrounding the wellbore, the equivalent permeability, $k_{eq}$, lies between a harmonic and an arithmetic average weighted by their depth interval, $\Delta h_i$, as follows:

$$\frac{\sum_i \Delta h_i}{\sum_i \frac{\Delta h_i}{k_i}} < k_{eq} < \frac{\sum_i k_i \Delta h_i}{\sum_i \Delta h_i}$$

The closeness between the harmonic and arithmetic averages will depend on the dispersion of the permeability distribution.

Warren and Price (1961) extensively investigated the problem of random permeability variation for both transient and steady state single phase flow in a 3-dimensional numerical model. They formulated a Monte-Carlo type model for generating a spatially random permeability field in which the lateral and vertical variation were of the same order of magnitude. The numerical model consisted of homogeneous porous blocks in which the permeabilities of the respective blocks were assigned according to a probability frequency distribution obtained from core samples. The frequency distribution was preserved in the model such that the number of blocks was always a multiple of the number of core samples. The spatial distribution for the different realizations were obtained by choosing a set of $N$ values, corresponding to the number of simulation blocks, from a specified frequency function, assigning
a random number to each and ordering the permeabilities according to the magnitude of the
random numbers. A one-to-one correspondence was then established between the ordered
permeabilities and the grid points. A mechanism of permutation of a base realization, of fixed
parameters, was employed to generate an ensemble of realizations of presumably completely
randomized blocks. It was found that the resulting frequency function of the realizations
asymptotically formed a log-normal frequency function irrespective of the parent distribution.
Results of the numerical experiment showed that the behavior of a heterogeneous system in
three-dimensions approaches that of a homogeneous with a permeability equal to the geometric
mean of the individual permeabilities.

The model formulated by Warren and Price did not take into consideration the covariance
behavior in the random permeability field; however, it is likely that, as the number of blocks
increased relative to the number of core samples, a spatial covariance structure of a limited range
will develop. Reported results showed reduction in the estimated variance of the permeability
distribution as the number of blocks sharing the same value of core permeability increased. The
reduction in the estimate of the variance may be explained in part by the variation in the number
of grid blocks (sample size) which varied between 27 and 1728; the smaller number of grid
blocks may be seen as an insufficient sample size. Further, choosing a core sample to represent
the simulation block dimension or even the drainage volume of a well will be prone to a high
magnitude of experimental error for, relatively speaking, the core dimensions would be regarded
of a microscopic dimension, a point which was alluded to in Figure 2.2. Hence the possible
covariance in the model was equivocally confounded in the variance estimate.

Heller (1972) put forward a method for generating a mathematically continuous and
spatially correlated random permeability field on a two-dimensional grid. The method involves
generating $N_{sp}$ triplets of random numbers. The first and second set are generated from a
uniform distribution and are used as a location parameter in $x$ and $y$ co-ordinates of a point in a flow system. The third random set follows a prescribed probability density function that gives the concentration parameter in a final formula. The concentration parameter represents the height of what may be referred to as source points or "permeability extrema." The number of source points, $N_{sp}$, (which is an input parameter) determine the spatial covariance structure of the synthesized random permeability field. These source points are viewed as points of maxima and minima on the hills and valleys of the contoured map of the logarithm of the permeability. These $N_{sp}$ source points, along with the $x$ and $y$ co-ordinates of a point, are used to calculate what may be called "permeability functions" at each mesh point. The "permeability functions" are calculated by using the inverse square law relationship. This involves relating all of the random points throughout the system to all of the mesh points, by using the inverse of the square of the differences between each mesh point an all of the introduced random values. This is achieved through the formula

$$s_{l}(N, M) = \sum_{j=1}^{N_{sp}} \frac{s(j)}{\sum_{j=1}^{N_{sp}} \left( \frac{1}{(x(j) - x_{l}(N))^{2} + (y(j) - y_{l}(M))^{2}} \right)}$$

where

- $N_{sp}$ = number of source points
- $s(j)$ = normally distributed random numbers (permeability extremum of the $j^{th}$ point)
- $x, y$ = random numbers representing $x$ and $y$ location of the $j^{th}$ point
- $x_{l}(N), y_{l}(M)$ = $x$ and $y$ location of a grid point
- $s_{l}(N, M)$ = permeability exponent (log($k$)) at grid point $(N, M)$
In order to obtain permeability values that are log-normally distributed, the exponential of the "f" values multiplied by a constant are evaluated. These are then assumed to represent the permeability values at each mesh point in the system. The value of permeability at each mesh point is thus obtained by:

\[
\text{permeability value at each mesh point (Heller, 1972)} = (W', N) \gamma
\]

The mean of the log-permeability field, \( \gamma \), is set to zero to obtain the limiting case of uniformly distributed

- Setting the value of \( \alpha \) to zero will obtain the limiting case of uniformly distributed

Where

\[
(W', N) \gamma = (W', N) \gamma
\]

In order to obtain permeability values that are log-normally distributed, the exponential of

Heravi (1988) investigated miscible displacement in a heterogeneous five-spot quadrant by generating a random permeability field using the Source Point method (SPM). The discrete log-permeability field, obtained by following closely the normal frequency function. The work of Heravi further showed:

(a) As the number of source points decreased, the correlation length increased

(b) The variogram of the random permeability field bears similarity to that of the spherical model, which lead to early time of breakthrough, 

(c) The variation of the random permeability field

(f) The discretization of the log-permeability field using the Source Point method (SPM)
King (1989) reported on the "renormalization procedure" method for improved estimate of
the effective permeability in rectilinear co-ordinates of a heterogeneous formation that would be
equivalent to a homogeneous formation. The method is in contrast to the simple usage of
geometric mean that does not apply where there is an impervious section in the formation. It
involves averaging over smaller regions of the reservoir to form a new averaged permeability
probability distribution with lower variance than the original as a result of the reduction of the
amount of fluctuations. The probability transformation is repeatedly done on the newer grid
until a stable result is found whose permeability approximates that of the region. For a
correlated medium, the variance in permeability will be reduced as would the correlation length
as a result of the renormalization of the permeability probability distribution. Repeated
renormalization would eventually give a single value of effective permeability, the probability
density of which has reduced to the Dirac delta function. In application of the method, King
found that in both two- and three-dimensions, the geometric mean was a good estimate of
effective permeability.

Gutjahr et al. (1978) derived analytically expressions for the effective hydraulic
conductivity for linear flow geometry in different dimensions. The effective conductivity was
found to be the harmonic mean for one-dimensional flow, the geometric mean for
two-dimensional flow, and \((1 + \sigma^2/6)\) times the geometric mean for three-dimensional flow,
where \(\sigma^2\) is the variance of the logarithm of conductivity.

Analytical solution for the effective permeability is not applicable when the flow regime
differs from unidirectional. An approximate expedient way for estimating the effective
permeability for an element of a five-spot pattern is to base the grid on an orthogonal curvilinear
co-ordinate system of isopotentials and stream functions in which isopotential are taken at
uniform pressure decrements in order to effectively linearize the flow regime. The stream function is defined as the complex conjugate of pressure and is given, in rectilinear co-ordinates \((x, y)\), by

\[
\frac{\partial \psi}{\partial x} = -\frac{\partial P}{\partial y} \quad ; \quad \frac{\partial \psi}{\partial y} = \frac{\partial P}{\partial x}
\]

and in orthogonal curvilinear co-ordinates, \((s, n)\), by

\[
\frac{\partial \psi}{\partial s} = -\frac{\partial P}{\partial n}
\]

where \(s\) is arc length along the curve of constant \(\psi\) (streamline) and \(n\) is the distance along the curve normal to the streamline.

Superimposing the pressure solution on the stream function solution may generate a flow-net of curvilinear squares. Muskat (1937) gave a graphical procedure to compute the flux in such a network for single-phase flow which shows that the volumetric flux \(Q\) in a particular segment of the formation bounded by two streamlines and two isobars is given by

\[
Q = \frac{m}{n} \Delta P
\]

where \(m\) and \(n\) are the number of unit streamlines and unit pressure drops, respectively. This amounts to calculation of the number of unit squares in the segment. Such a procedure is useful in computing ultimate recovery, however, in order to compute recovery history, a different
procedure should be instituted. Hurst (1981) gave a front track procedure based on the theory of electrical models. A numerical computational scheme based on the same analogue is given by Craft and Hawkins (1959).

The analytical solutions of the governing partial differential equation for pattern miscible displacement exist only when the medium is uniform and the mobility ratio is unity. Muskat (1937) derived the analytical pressure solution for an element of a five-spot pattern by consideration of an infinite array of regularly spaced sources and sinks. Hurst (1981) gave the corresponding stream function solution. Both solutions are in the form of infinite series. When the properties of the medium vary, only approximate methods of solution are possible. Among these methods, the finite difference is the most widely used, while the finite element method, traditionally used in structural mechanics, is gaining inroads in modeling fluid dynamics flow problems.

The partial differential equation for steady state miscible displacement is elliptic which for the case of constant coefficient (uniform medium) results is a well-posed numerical form. However for the variable coefficient form, abrupt changes in the properties of the media may perturb the numerical scheme significantly. Whereas the finite difference method approximates the governing partial differential equation by a differential approach, the finite element uses an integral representation of the partial differential equation. Both methods convert the problem from continuous space to discrete space utilizing "blocks" and "elements," respectively. The finite difference method normally uses arbitrary rectangular grid and hence can treat heterogenieties if they are described by simple geometry or by the spatial domain into a large number of presumably homogeneous blocks. The finite element method is more flexible in this
respect, permitting variable spatial resolution and flexible geometry. This flexibility would allow exact treatment of source functions.

It is customary in the finite element method to perform integration over the element assuming constant medium properties; then the heterogeneity is accounted for when element integrations are combined globally. Pinder et al. (1973) reported that variable medium properties within an element (simulation block) can be handled easily by isoparametric quadratic elements formulation. This should permit more flexibility in discretizing the solution domain by using an appropriate approximation function of permeability within the element. Results reported by them showed high accuracy that matched the analytic solution.

The finite element representation of the partial differential equation is such that it satisfies the underlying physical principle and can be formulated differently. Two of the most popular methods are: (a) the method of weighted residual, and (b) the variational method. For the same class of problems that include those described by Laplace's Equation, the two methods yield the equivalent form (Wang and Anderson, 1982).

Settari et al. (1977) applied the variational method for the slug dilution problem in a homogeneous medium five-spot pattern on a 10x10 finite element grid. Results showed accuracy comparable to that which would require 20,000 grid blocks using finite difference.
Chapter 3. Formulation of the Model

3.0 Problem Statement

The more practical problem of pattern miscible displacement of an inhomogenous fluid system in a heterogeneous formation is simplified in this investigation to that of miscible displacement at unit mobility ratio in a representative element of a heterogeneous but isotropic five-spot quadrant. The assumptions made are those of two-dimensional steady state flow of a slightly compressible fluid system and negligible gravity effect. The governing partial differential equation (using standard notation) is

$$\nabla \cdot (k(x, y) \nabla P) = \nabla^2 P + \nabla P \cdot \nabla \ln k(x, y) = 0$$  \hspace{1cm} (3.1)

The governing equation is solved numerically with a synthesized permeability field using the "Source Point" method (SPM) or the Spectral Method algorithm. These methods account quantitatively for the covariance behavior of the generated random field and possess the property of stationarity. The effect of the degree of heterogeneity and its correlation scale can be simulated by generating a sufficiently large ensemble of realizations; however, because of the added geometrical effect in pattern displacement, the relative location of heterogeneities may influence the sweep efficiency. Hence, the aim here is to investigate, in one given realization, the effect of translating the boundaries of a representative element of pattern within a larger segment of the random permeability field. This will enable generating subrealizations having fixed parameters of degree of heterogeneity and correlation scale. The magnitude of the displacement of the element boundaries would be such that the regional influence, for which the assumptions of stationarity hold, is not exceeded. The generated ensemble of subrealizations will
enable investigating pattern production recovery characteristics as a function of boundaries translation.

For the study of the invariance of recovery characteristics with translation, sixteen subrealizations generated by SPM based on one chosen realization will be investigated.

The numerical procedure applied to each subrealization is comprised of the following different stages:

1. Solution of the flow equation for pressure/stream function;
2. Computation of the velocity field, streamlines, and front tracking;
3. Calculation of recovery history.

A statistical test on the invariance of recovery characteristics with translation is performed by testing the significance in the variance of the recovery history for the different subrealizations. This procedure is given in Chapter 4 of this thesis.

Four realizations will be generated using the Spectral Method to broadly infer the heterogeneity characteristics of the random permeability field and that of the resultant pressure distribution.
3.1 Derivation of the Governing Equation

By applying the principle of conservation of mass to an element of volume of a heterogeneous but isotropic porous medium of dimensions $\Delta x$, $\Delta y$, $\Delta z$, porosity $\phi$, and permeability $k(x, y, z)$, where the dimensions of the element are of scale greater than the microscopic scale of heterogeneity, and assuming unsteady flow of a fluid of density $\rho$, viscosity $\mu$, and with velocity components $u_x$, $u_y$, $u_z$, and by neglecting gravity effects, one obtains

$$
\frac{\partial}{\partial x} (\rho u_x) + \frac{\partial}{\partial y} (\rho u_y) + \frac{\partial}{\partial z} (\rho u_z) - \dot{m}(x, y) = - \frac{\partial}{\partial t} (\rho \phi) \tag{3.2}
$$

where $\dot{m}$ is the source/sink mass flux at point $(x, y, z)$.

Applying Darcy's Law:

$$
\begin{align*}
   u_x &= - \frac{k(x, y, z)}{\mu} \frac{\partial P}{\partial x} \\
   u_y &= - \frac{k(x, y, z)}{\mu} \frac{\partial P}{\partial y} \\
   u_z &= - \frac{k(x, y, z)}{\mu} \frac{\partial P}{\partial z}
\end{align*} \tag{3.3}
$$

Substituting Equation (3.3) into Equation (3.2), we have

$$
\nabla \cdot \left( \frac{k(x, y, z)}{\mu} \rho \nabla P \right) + \dot{m}(x, y) = \frac{\partial}{\partial t} (\phi \rho) \tag{3.4}
$$
\[ 0 = (\kappa \cdot \nabla) b + \left[ \frac{\eta}{(\kappa' \cdot \nabla) \Delta} + \frac{d}{\Delta} \frac{\eta}{(\kappa' \cdot \nabla) \Delta} \right] \cdot \Delta + d \Delta \frac{\eta}{(\kappa' \cdot \nabla) \Delta} = (\kappa' \cdot \nabla) b + \left( d \Delta \frac{\eta}{(\kappa' \cdot \nabla) \Delta} \right) \cdot \Delta \frac{d}{\Delta} \]

where \( b \) is the volumetric flux at source or sink points. Expanding the divergence and gradient:

\[ 0 = (\kappa' \cdot \nabla) b + \left( d \Delta \frac{\eta}{(\kappa' \cdot \nabla) \Delta} \right) \cdot \Delta \frac{d}{\Delta} \]

or

\[ 0 = \frac{d}{\Delta} \left( \frac{\eta}{(\kappa' \cdot \nabla) \Delta} \right) \cdot \Delta \frac{d}{\Delta} \]

Dividing Equation (3.9) by \( \phi \) and neglecting the vertical variation in \( \phi \), we have

\[ 0 = \frac{10}{(d \phi) \epsilon} \]

at a steady state.

(9.9)

\[ d \Delta \phi \Delta = d \Delta \frac{d \phi}{d \epsilon} = d \Delta \]

(9.9)

\[ \frac{10}{\phi \epsilon} \Delta \phi + \frac{10}{d \epsilon} \phi \Delta \phi = \frac{10}{(d \phi) \epsilon} \]

Hence

\[ (d \phi - d') \phi + 1 = \frac{d}{d'} \]

of state is approximated by

Here \( \phi \) is pressure dependent and hence Equation (3.9) is non-linear. For liquid, the equation

26
Dividing by \( \frac{k(x, y)}{\mu} \), we have

\[
\nabla^2 P + \nabla \cdot \nabla \ln \rho + \nabla \cdot \nabla \ln k + \frac{\mu q(x, y)}{k(x, y)} = 0
\]

(3.7)

Applying Equation (3.6), the second term may be expressed as

\[
\nabla \cdot \nabla \ln \rho = c \rho_0 \nabla \cdot \nabla \ln P
\]

or

\[
\nabla \cdot \nabla \ln \rho = c \rho_0 (\nabla P)^2 / P
\]

It can be seen that \( \left( \frac{c \rho_0 (\nabla P)^2}{P} \right) \) is of relatively small magnitude in liquid flow and if it is ignored, Equation (3.7) reduces to

\[
\nabla^2 P + \nabla \cdot \nabla \ln k(x, y) + \frac{\mu q(x, y)}{k(x, y)} = 0
\]

(3.8)

which is the governing equation for two-dimensional miscible displacement at unit mobility ratio in a heterogeneous formation. However, the practical problem described by Equation (3.8) has no analytic solution for arbitrary spatial variability of \( k \).

For uniform media in which \( \nabla \ln k(x, y) \) is zero, Equation (3.8) reduces to

\[
\frac{\partial^2 P}{\partial x^2} + \frac{\partial^2 P}{\partial y^2} + \frac{\mu q(x, y)}{k(x, y)} = 0.
\]

(3.9)
Muskat (1937) gave the analytical solution of Equation (3.9) for the boundary conditions imposed by a five-spot flood network represented by an infinite array of regularly spaced point sources and sinks in which the x-axis was chosen so as to go through the diagonal of the element square as shown in Figure 3.1.

![Five-Spot Flood Network Diagram](image)

Figure 3.1. The Five-Spot (Homogenous Medium) Flood Network

The pressure distribution with the origin taken either as a source (injection well) or a sink (production well) is given by

\[
P(x, y) = q \log \frac{\cosh \pi y/d - \cos \pi x/d}{\cosh \pi y/d + \cos \pi x/d}
\]

\[
- q \sum_{m=1}^{\infty} (-1)^m \log \frac{4 \left[ \cosh \frac{\pi (y - md)}{d} + \cos \frac{\pi x}{d} \right]}{d e^{2\pi x}}
\]

\[
+ q \sum_{m=1}^{\infty} (-1)^m \log \frac{4 \left[ \cosh \frac{\pi (y + md)}{d} - \cos \frac{\pi x}{d} \right]}{d e^{2\pi x}}
\]

(3.10)

where \(d\) is the distance between the injection well and production well, \(q\) is the steady state fluid injection rate, \(k\) is the permeability of the formation, \(\mu\) is the viscosity of the fluid, and \(h\) is the formation thickness. Muskat further showed that the limiting value of the pressure differential existing between the injection well and the production well is
\[ \Delta P = P(0, d - r_w) - P(0, r_w) \]

is given by

\[ \Delta P = \frac{\mu q \left[ \log_e \left( \frac{d}{r_w} \right) - 0.6190 \right]}{3.54 kh} \]

where \( r_w \) is the wellbore radius and the constant 3.54 is the conversion factor to the oil-field system of units.

Hurst (1981) developed the relationship that determines the streamline distribution for a uniform medium as

\[ S_T(x, y) = \frac{\mu q}{2 \pi kh} \left\{ \frac{\pi}{2} - \left[ \tan^{-1} \frac{\sin \pi x/d}{\sinh \pi y/d} \right] \right\} + \sum_{n=1}^{\infty} (-1)^n \left[ \tan^{-1} \frac{\sin \pi x/d}{\sinh \pi (y + md)/d} + \tan^{-1} \frac{\sin \pi x/d}{\sinh \pi (y - md)/d} \right] \]

(3.11)

The Muskat and Hurst solutions of the pressure and streamline distributions may be used as a reference check for approximate methods of solution for the homogeneous medium case. Further, the flow-net generated by the two solutions [(3.10) and (3.11)] may be used as an orthogonal curvilinear co-ordinate system in this ideal case.
3.2 Generation of the Subrealizations of Random Permeability Fields

3.2.1 The Source Point Method

The Source Point Method involves generating three sets of random numbers and conditioning them so as to represent random permeability field. The first and second set are generated from a uniform distribution and are used as a location parameter in the \( x - y \) co-ordinates of a point in a flow system while the third set is used as the concentration parameter to give the permeability extrema (maxima and minima) of the system.

The SPM procedure comprises the following steps:

Step 1: Generate two sets of uniform random numbers distributed over the interval [0,100] and generate a third associated set from the standard normal distribution. These numbers are pseudorandom numbers obtained from such scientific software packages as the IMSL software.

Step 2: Adopt a grid system and divide appropriately into blocks or elements. Choose a value for the number of source points \( N_{sp} \) commensurate with the number of blocks or elements.

Step 3: Compute the permeability exponent \( sI(N,M) \) at the grid points via

\[
sI(N,M) = \frac{\sum_{j=1}^{N_{sp}} \frac{s(j)}{((x(j) - xI(N))^2 + (y(j) - yI(M))^2)}}{\sum_{j=1}^{N_{sp}} \frac{1}{((x(j) - xI(N))^2 + (y(j) - yI(M))^2)}}
\]
Repeat steps 1 to 4.

Confirm that the blocks or elements dimensions are within the scale of heterogeneity or else.

Step 4: Generate a random variation of the two-dimensional permability field thus obtained.

\[ \mu = (N, \nu) \gamma \]

where

\[ \nu = (N, \mu) \gamma \]

and assign a base permability value \( \gamma \) and permability distribution scale factor \( \alpha \) and

\[ \mu = (N, \nu) \gamma \]

where

\[ \gamma = (N, \mu) \]

Normally distributed random numbers (permability extremes) and location of source points

\[ d_N s \]

Repeat steps 1 to 4.

Confirm that the blocks or elements dimensions are within the scale of heterogeneity or else.

Step 4: Generate a random variation of the two-dimensional permability field thus obtained.

\[ \mu = (N, \nu) \gamma \]

where

\[ \nu = (N, \mu) \gamma \]

and assign a base permability value \( \gamma \) and permability distribution scale factor \( \alpha \) and

\[ \mu = (N, \nu) \gamma \]

where

\[ \gamma = (N, \mu) \]

Normally distributed random numbers (permability extremes) and location of source points

\[ d_N s \]

where

\[ d_N s \]
3.2.1.1 Generalization of Subrealizations using SPM

In the application of SPM, the first two sets of random numbers are chosen from the uniform distribution using pseudo-random numbers. Each number $r$ chosen lies in the range

$$0.0 \leq r \leq 100.0.$$ 

This range pertains to a base realization having external boundaries coinciding with the quadrant boundaries. Assuming the larger segment of the heterogeneous formation containing the base realization extends 10% of the side of the quadrant in both the negative and positive $x$ and $y$ directions so that the larger segment area is 1.44 times the area of the quadrant, then the uniform random number relationship is rescaled to fit the larger segment according to

$$R = A + (B - A)r$$

where

$$A = -10.0$$  
$$B = 110.0$$

$$-10.0 \leq R \leq 110.0$$

A finite difference $64 \times 64$ grid was chosen with $N_{sp}$ of 288 for this investigation. The base realization was moved within the larger segment using two parameters, $d$ and $\theta$, where $d$ gives the displacement in the negative or positive $x$-direction and $\theta$ is the angle between the positive $x$-direction and the line joining the old origin to the new origin (injector location) as shown in Figure 3.2. The displacement was made in steps of 4% of the length of the side of the quadrant (660.0 feet), and $\theta$ ranged $0.0^\circ$ to $90.0^\circ$ in steps of $22.5^\circ$. Sixteen such displacements were made that included the base realization.
3.2.2 Spectral Method Algorithm

The spectral representation theorem is defined in the complex vector space. A complex random variable $Z$, defined on the wave number domain $(\bar{\omega})$, is a process of assigning to every outcome $\bar{\omega}$ a complex vector $Z(\bar{\omega})$,

$$Z(\bar{\omega}) = R(\bar{\omega}) + iT(\bar{\omega})$$

where $R$ and $T$ are independent real random variables.

The importance of the spectral representation theorem is that it takes a correlated structure in the spatial domain, $\bar{x}$, and replaces it with an uncorrelated process defined in the wave number domain. According to the theorem, a statistically homogeneous random field $V(\bar{x})$ is given by a complex stochastic process $Z(\bar{\omega})$ as the multiple integral

$$V(\bar{x}) = \int \int \cdots \int e^{i\bar{\omega} \cdot \bar{x}} dZ(\bar{\omega})$$

abbreviated as

$$V(\bar{x}) = \int e^{i\bar{\omega} \cdot \bar{x}} dZ(\bar{\omega})$$

The spectral representation theorem is stated as follows:

Let $V(\bar{x})$ be a statistically homogeneous random field that is zero-mean and that has a spectral density $S(\bar{\omega})$, then there exists a unique (with probability one) complex stochastic process $Z(\bar{\omega})$ with the following properties:
\[ 2P \left( \frac{1}{2} \right) C_{1, \frac{1}{2}, 0} \int_{-\infty}^{\infty} = (\pi) \phi \]

Rewriting the spectral density as

\[ 2P \left( \frac{1}{2} \right) C_{1, \frac{1}{2}, 0} \int_{-\infty}^{\infty} = (\pi) \phi \]

Transformation of the computational procedure.

Analytic solution. The algorithm is intended for large fields, and it uses the Fast Fourier
of spectral density and covariance function. The discrete solution may be compared with the
may take an input either the spectral density or the covariance function. In case of known
pairs

(1999). Making use of the reciprocity property of the Fourier Transformation, the algorithm

The algorithm for generating a two-dimensional random field is bridged from Cuglar

Direct delta function that picks up jumps of the Z(\omega).

\[ \omega = \omega = 1 \omega \int P(\omega) \mathcal{E} = \left[ (\omega) \mathcal{E}, (1, \omega) \mathcal{E} \right] \mathcal{E} (\Pi) \]

\[ 0 = 0 \int \mathcal{E} \mathcal{E} (\Pi) \]

formulas

\[ (\omega) \mathcal{E} \mathcal{E} \int_{-\infty}^{\infty} = (\omega) \mathcal{E} (\Pi) \]

35
\[
\left[ \left( \frac{T}{I} + \sqrt{w} \right) \varphi \left( \frac{T}{I} + \sqrt{w} \right) \right] \nabla = \left( \frac{T}{I} + \sqrt{w} \right) \varphi
\]

where is taken to be

\[
\left( \frac{T}{I} + \sqrt{w} \right) \varphi
\]

In a two-dimensional square region such that \( x \cdot x \cdot x \) and \( \varphi \cdot \varphi \cdot \varphi \) are taken at the midpoints of the intervals, we have

\[
\varphi \cdot \varphi \cdot \varphi \cdot \varphi = \sum_{1 \leq i \leq N} \varphi \cdot \varphi \cdot \varphi \cdot \varphi
\]

In discrete form, \( \varphi \) is approximated using a finite set of spectra given by an order

\[
\left( \varphi \right) \Lambda = \left( x \right) \Lambda
\]

\[
\left( \varphi \right) \Lambda
\]

where denotes the space dimension. Hence,

\[
\left( \varphi \right) \Lambda
\]
where the contraction of the $sZP$ implies

\[
\left((f^* f)^{\varphi}_{\Delta} + (f^* f)^{\varphi}_{\Delta} + (f^* f)^{\varphi}_{\Delta} + (f^* f)^{\varphi}_{\Delta}\right) = \\
\left((w^* w)^{\varphi}_{\Delta} ZP \sum_{x_{\Delta}} \sum_{y_{\Delta}} \sum_{z_{\Delta}} \sum_{w_{\Delta}} m_{z_{\Delta}} \sum_{1=1}^{N} \sum_{1=1}^{M} m_{w_{\Delta}} \sum_{1=1}^{N} \right) + \\
\left((w^* w)^{\varphi}_{\Delta} ZP \sum_{x_{\Delta}} \sum_{y_{\Delta}} \sum_{z_{\Delta}} \sum_{w_{\Delta}} m_{z_{\Delta}} \sum_{1=1}^{N} \sum_{1=1}^{M} m_{w_{\Delta}} \sum_{1=1}^{N} \right) + \\
\left((w^* w)^{\varphi}_{\Delta} ZP \sum_{x_{\Delta}} \sum_{y_{\Delta}} \sum_{z_{\Delta}} \sum_{w_{\Delta}} m_{z_{\Delta}} \sum_{1=1}^{N} \sum_{1=1}^{M} m_{w_{\Delta}} \sum_{1=1}^{N} \right) + \\
\left((w^* w)^{\varphi}_{\Delta} ZP \sum_{x_{\Delta}} \sum_{y_{\Delta}} \sum_{z_{\Delta}} \sum_{w_{\Delta}} m_{z_{\Delta}} \sum_{1=1}^{N} \sum_{1=1}^{M} m_{w_{\Delta}} \sum_{1=1}^{N} \right) = (f^* f)^{\varphi}_{\Delta}
\]

(2.13)

The sum for $(\forall x)(x)_A$ can be broken into parts

\[
\left((w^* w)^{\varphi}_{\Delta} ZP \sum_{x_{\Delta}} \sum_{y_{\Delta}} \sum_{z_{\Delta}} \sum_{w_{\Delta}} m_{z_{\Delta}} \sum_{1=1}^{N} \sum_{1=1}^{M} \sum_{1=1}^{N} m_{w_{\Delta}} \sum_{1=1}^{N} \right)
\]

Next, let $\mathcal{M}_{x_{\Delta} / y_{\Delta}} = M$. Then

\[
(\forall x)(x)_A \quad \text{is real and has the same covariance as}
\]

\[
(\forall x)(x)_A
\]

insures that $\mathcal{M}$ is real and has the same covariance as $\mathcal{M}$.

\[
\left((n)^{\varphi}_{\Delta} - (n)^{\varphi}_{\Delta} = (n^\text{-})(n^\text{-})_{\Delta} = (n^\text{-})(n^\text{-})_{\Delta}
\]

and

\[
\left((n)^{\varphi}_{\Delta} - (n)^{\varphi}_{\Delta} = (n^\text{-})(n^\text{-})_{\Delta} = (n^\text{-})(n^\text{-})_{\Delta}
\]

Random variables that have variances equal to 1/2 of the choice of

where denotes the complex conjugate of $n$ and $(n^\text{-})_\Delta$ are zero-mean independent.

\[
(n)_\Delta ZP = (n^\text{-})_\Delta ZP
\]

Further,

\[
37\]
\[ dZ(-m_x, -m_y) = dZ^*(m_x-1, m_y-1) \quad \text{for } m_x = 1, ..., M; \quad m_y = 1, ..., M. \]

which in turn yields \( \hat{V}_1(j_x, j_y) = \hat{V}_1^*(j_x, j_y) \) and \( \hat{V}_2(j_x, j_y) = \hat{V}_2^*(j_x, j_y) \). As a result, only \( \hat{V}_1(j_x, j_y) \) and \( \hat{V}_2(j_x, j_y) \) will need to be calculated. These can be combined and computed with a single two-dimensional Fast Fourier Transform (FFT).

However, it is noted that the FFT is generally of the form

\[
\sum_{m=0}^{2M-1} e^{-2\pi i m/m} dZ(m)
\]

while the sum in (3.12) only goes to \( M-1 \), hence the \( dZ \) matrix must be augmented in each dimension with complex zeros in positions \( m = M, ..., 2M - 1 \) such that the zeros occupy the first and fourth quadrants in the solution space

\[
dZ = \begin{pmatrix}
    \begin{pmatrix}
        dZ_{0,0} & \cdots & dZ_{0,M-1} \\
        \vdots & \ddots & \vdots \\
        dZ_{M-1,0} & \cdots & dZ_{M-1,M-1}
    \end{pmatrix} & \begin{pmatrix}
        0 & \cdots & 0 \\
        \vdots & \ddots & \vdots \\
        0 & \cdots & 0
    \end{pmatrix}
\end{pmatrix}
\]

The first field will be \( 2 \text{ Re} \{ \hat{U}(j_x, j_y) \} \) where

\[
\hat{U}(j_x, j_y) = \sum_{m_x=0}^{2M-1} \sum_{m_y=0}^{2M-1} W^{2m_x j_x + 2m_y j_y + j_x + j_y} dZ(m_x, m_y)
\]
\[ \mathcal{W}/1 = x \mathcal{V} \quad \mathcal{W}/1 = x \mathcal{V} \]
\[ \mathcal{W}/\mathcal{V} = n \mathcal{V} \quad \mathcal{W}/\mathcal{V} = n \mathcal{V} \]

:\textit{Step I} : 

The major computational steps in the algorithm for an input spectral density \( S(\mu) \) are:

\[
\mathcal{W} \prec z^w \mathcal{P} \quad 0 = (z^w \mathcal{w}) \mathcal{Z} \mathcal{P} \\
1 - \mathcal{W} \cdots \cdots 0 = z^w \quad 1 - \mathcal{W} \cdots \cdots 0 = z^w \quad (z^w - \mathcal{w}) \mathcal{Z} \mathcal{P} = \]

\[
(z^w \mathcal{w} \mathcal{L}! + z^w \mathcal{P}) \\
\left\{ z^{n \mathcal{V}} [z^{n \mathcal{V}} \left( \frac{z}{1} + z^w \right) z^{n \mathcal{V}} \left( \frac{z}{1} + z^w \right)] \phi \right\} = (z^w + \mathcal{W} z^w) \mathcal{Z} \mathcal{P} \\
1 - \mathcal{W} \cdots \cdots 0 = z^w \quad 1 - \mathcal{W} \cdots \cdots 0 = z^w \\
\text{for} \\
\left\{ z^w \mathcal{w} \mathcal{L}! + z^w \mathcal{P} \right\} \\
\left\{ z^{n \mathcal{V}} [z^{n \mathcal{V}} \left( \frac{z}{1} + z^w \right) z^{n \mathcal{V}} \left( \frac{z}{1} + z^w \right)] \phi \right\} = (z^w \mathcal{w}) \mathcal{Z} \mathcal{P}
\]
Step 2: Form the array.

\[ dZ(m_x, m_y) = \left\{ \phi \left[ \left( m_x + \frac{1}{2} \right) \Delta u_x, \left( m_y + \frac{1}{2} \right) \Delta u_y \right] \Delta u_x \Delta u_y \right\}^{1/2} \left( R_{m_x, m_y} + iT_{m_x, m_y} \right), \]

for \( 0 \leq m_x \leq M_x - 1, \ 0 \leq m_y \leq M_y - 1. \)

\[ dZ(m_x, 2M_y + m_y) = \]
\[ \left\{ \phi \left[ \left( m_x + \frac{1}{2} \right) \Delta u_x, \left( m_y + \frac{1}{2} \right) \Delta U_y \right] \Delta u_x \Delta u_y \right\}^{1/2} \left( R_{m_x, m_y} + iT_{m_x, m_y} \right), \]

for \( 0 \leq m_x \leq M_x - 1, \ -M_y \leq m_y \leq -1. \)

\[ dZ(m_x, m_y) = 0, \quad \text{otherwise.} \]

where \( R_{m_x, m_y}, T_{m_x, m_y} \) are independent, zero-mean, variance \(-\frac{1}{2}\) random normal variables.

Step 3: Perform a two-dimensional Fast Fourier Transform of the \( dZ \) array and take twice the real part of the lower fourth of the output array.

Usage was made of the FORTRAN coded program of this algorithm which is available on the SUN computer system of New Mexico Tech to generate a two-dimensional permeability field for the input given in Chapter 4 of this thesis.
3.3 Solution of the Flow Equation

3.3.1 Finite Difference Formulation

3.3.1.1 Pressure Approach

The basic objective of the finite difference calculation method is to represent the time-space continuum by a set of discretely-spaced points. Equation (3.8) is an elliptic pde defining a two-dimensional spatial continuum. The finite difference approximation of Equation (3.8) is developed as follows:

\[
\frac{\partial}{\partial x} \left( \frac{k(x,y)}{\mu} \frac{\partial P}{\partial x} \right) + \frac{\partial}{\partial y} \left( \frac{k(x,y)}{\mu} \frac{\partial P}{\partial y} \right) + q(x,y) = 0
\]

Now, letting \( \Delta x \Delta y \) represent the simulation block dimension, and representing the formation thickness by \( h \), we have the approximation for a five-point differencing scheme,

\[
\begin{align*}
\frac{k_{i+1/2} \Delta \left( P_{i+1,j} - P_{i,j} \right) - k_{i-1/2} \Delta \left( P_{i,j} - P_{i-1,j} \right)}{2 \Delta x^2} \\
+ \frac{k_{j+1/2} \Delta \left( P_{i,j+1} - P_{i,j} \right) - k_{j-1/2} \Delta \left( P_{i,j} - P_{i,j-1} \right)}{2 \Delta y^2} = q_{i,j}
\end{align*}
\]

Multiplying by the block volume \( \Delta x \Delta y \Delta z \), where \( \Delta z \) represents the formation thickness (h), we have
\[
\mathbf{r}_{\mathcal{D}} = \begin{bmatrix}
1 + r_d' \\
1 + r_d' \\
r_d' \\
1 - r_d'
\end{bmatrix}
\begin{bmatrix}
\frac{\mathbf{I}'}{z} \\
\frac{\mathbf{I}'}{z} \\
\frac{\mathbf{I}'}{z} \\
\frac{\mathbf{I}'}{z}
\end{bmatrix} + \begin{bmatrix}
\frac{\mathbf{I}'}{z} \\
\frac{\mathbf{I}'}{z} \\
\frac{\mathbf{I}'}{z} \\
\frac{\mathbf{I}'}{z}
\end{bmatrix} - \begin{bmatrix}
\mathbf{I}' \\
\mathbf{I}' \\
\mathbf{I}' \\
\mathbf{I}'
\end{bmatrix}
\]

Expressing Equation (3.12) in matrix form, we have:

\[
\left(\frac{\mathbf{I}'}{z} + \frac{\mathbf{I}'}{z}\right) = \mathbf{I}' + \mathbf{I}'
\]

\[
\left(\frac{\mathbf{I}'}{z} + \frac{\mathbf{I}'}{z}\right) = \mathbf{I}' + \mathbf{I}'
\]

Now, in series, the inter-block transmissibilities are evaluated using the harmonic average:

\[
\frac{\mathbf{I}}{z} \frac{\mathbf{I}}{z} \frac{\mathbf{I}}{z} = \mathbf{I}' + \mathbf{I}'
\]

\[
\frac{\mathbf{I}}{z} \frac{\mathbf{I}}{z} \frac{\mathbf{I}}{z} = \mathbf{I}' + \mathbf{I}'
\]

Denoting the transmissibilities by:

\[
\mathbf{I}_d = \frac{z \mathbf{V} \mathbf{V} \mathbf{V} \mathbf{V}}{\mathbf{I}}
\]

where \( \mathbf{I}_d \) is the source/sink term in units of volume per unit time:

\[
\mathbf{I}_d = \begin{bmatrix}
(1 - r_d') (1 - r_d') \\
(1 - r_d') (1 - r_d') \\
(1 - r_d') (1 - r_d') \\
(1 - r_d') (1 - r_d')
\end{bmatrix}
\]

(3.12)
where
\[
\sum T = \frac{\Delta x}{\Delta y} T_{i,j-1/2} + \frac{\Delta y}{\Delta x} T_{i-1/2,j} + \frac{\Delta y}{\Delta x} T_{i+1/2,j} + \frac{\Delta x}{\Delta y} T_{i,j+1/2}
\]
This formulation results in a symmetric discretized form of the given elliptic problem. The coefficient matrix, however, will not be positive definite and, hence, no unique pressure solution will be obtained unless a pressure value is specified at a boundary point.

The truncation error (from Taylor's series expansion) results in an approximation to \(\frac{\partial^2 P_{i,j}}{\partial x^2}\) that is first order correct for unequally spaced nodal values and is second order for equally spaced values (Pinder and Gray, 1977).

More accurate finite difference schemes can be obtained by representing the differential equation by a higher-order finite difference approximation designed to reduce the truncation error. In such a scheme, the number of pivoted values in the computation stencil are increased. A nine-point finite difference approximation to Poisson's equation given by the computational stencil (Smith, 1965)
\[
\begin{bmatrix}
1 & 4 & 1 \\
4 & -20 & 4 \\
1 & 4 & 1
\end{bmatrix} P,
\]
has a truncation error of order \(O(\Delta x)^4 + O(\Delta y)^4\), as opposed to the five-point stencil
\[
\begin{bmatrix}
1 \\
1 & -4 & 1 \\
1
\end{bmatrix} P,
\]
that has a truncation error of order \(O(\Delta x)^2 + O(\Delta y)^2\).

Both the five-point and nine-point schemes are used in the numerical model.
Incorporation of Boundary Conditions

Assuming Neumann no-flow boundary along the four exterior boundaries, we have

\[
\begin{align*}
\frac{\partial P}{\partial x} &= 0 \quad (y = 0; \quad y = y_e) \\
\frac{\partial P}{\partial y} &= 0 \quad (x = 0; \quad x = x_e)
\end{align*}
\]

and that the sandface injection pressure is prescribed at \((x = 0, \ y = 0)\)

\[
Q(x, y) = Q_{producer} \quad \delta(x - x_e, y - y_e)
\]

where \(Q(x, y)\) is a sink function, \(Q_{producer}\) is the magnitude of the sink, and \(\delta(x - x_e, y - y_e)\) is the Dirac delta function at the "Producer" location \(x_e, y_e\).

A central difference approximation of (3.13) whose truncation error is of order \(O(\Delta x)^2\) and \(O(\Delta y)^2\), is given by

\[
\begin{align*}
\frac{\delta (kP)}{\partial x} &= \frac{k_{i+1,j} P_{i+1,j} - k_{i-1,j} P_{i-1,j}}{2\Delta x} = 0 \\
\frac{\delta (kP)}{\partial y} &= \frac{k_{i,j+1} P_{i,j+1} - k_{i,j-1} P_{i,j-1}}{2\Delta y} = 0
\end{align*}
\]

which suggest

\[
\begin{align*}
P_{i+1,j} &= P_{i-1,j} \\
P_{i,j+1} &= P_{i,j-1} \\
k_{i+1,j} &= k_{i-1,j} \\
k_{i,j+1} &= k_{i,j-1}
\end{align*}
\]
In the main computer program, given in Appendix A, to the boundary conditions involving both the five-point formula and nine-point formula is included may be very slow. A FORTRAN coded procedure for solving the governing equation subject successively over-relaxation (SOR) was used although the point SOR method lacks the method of point system of equations is normally easier to code using iterative methods. The method of point thousands of equations that would need large computer memory for any "direct" method. Such methods, however, the typical application considered herein may require the solution of equation resulting from the finite difference may be solved iteratively or by using "direct" equation defined at the nodal points rather than the block centers was adopted. The algebraic system of equations, in a five-point pattern, the pressure and sink are prescribed at the boundaries, a grid

**Computer Implementation**

\[ 1 - f'1 - f'1 = 1 + f'1 + f' \]

\[ 1 - f'1 - f'1 = 1 + f'1 + f' \]

Further, for the nine-point differencing scheme

45
3.3.1.2 Stream Function Approach

The governing equation given by Equation (8) can be reformulated using the stream function approach. For laminar flow, particles follow a series of lines called streamlines. For steady flow, the streamlines do not change with time. At steady state and for an incompressible fluid, the continuity equation reduces to

\[
\frac{\partial v_x}{\partial x} + \frac{\partial v_y}{\partial y} = 0
\]  

(3.14)

Recall Darcy's Law in a heterogeneous medium,

\[

\begin{align*}
  v_x &= -\frac{k(x,y) \partial P}{\mu \partial x} \\
  v_y &= -\frac{k(x,y) \partial P}{\mu \partial y}
\end{align*}

\]  

(3.15)

If \( k(x,y) \) is constant and substituting (3.15) into (3.14) will yield

\[
\frac{\partial^2 P}{\partial x^2} + \frac{\partial^2 P}{\partial y^2} = 0
\]

which is Laplace's Equation.

Equation (3.14) can also be satisfied by defining the stream function \( \psi \) such that:

\[
\begin{align*}
  v_x &= \frac{\partial \psi}{\partial y} \\
  v_y &= -\frac{\partial \psi}{\partial x}
\end{align*}

\]  

(3.16)
because the first and second derivatives of $\psi$ are continuous; the order of differentiation should be immaterial, thus

$$\frac{\partial^2 \psi}{\partial y \partial x} - \frac{\partial^2 \psi}{\partial x \partial y} = 0$$

Substituting (3.16) into (3.15) and dividing both sides by $\frac{k(x,y)}{\mu}$ gives

$$\frac{\partial P}{\partial x} = \frac{\partial \psi}{\partial y} \frac{\mu}{k(x,y)} \quad (3.17)$$

$$\frac{\partial P}{\partial y} = -\frac{\partial \psi}{\partial x} \frac{\mu}{k(x,y)} \quad (3.18)$$

Now, differentiating Equation (3.17) with respect to $y$ and Equation (3.18) with respect to $x$ and subtracting gives

$$\frac{\partial}{\partial x} \left[ \frac{\mu}{k(x,y)} \frac{\partial \psi}{\partial x} \right] + \frac{\partial}{\partial y} \left[ \frac{\mu}{k(x,y)} \frac{\partial \psi}{\partial y} \right] = 0 \quad (3.19)$$

because

$$\frac{\partial^2 P}{\partial x \partial y} = -\frac{\partial^2 P}{\partial y \partial x}.$$  

Equation (3.19) shows that $\psi$ also satisfies a modified Laplace Equation.

Computing the inner product

$$\nabla P \cdot \nabla \psi = \frac{\partial P}{\partial x} \frac{\partial \psi}{\partial x} + \frac{\partial P}{\partial y} \frac{\partial \psi}{\partial y}$$
and substituting for $\frac{\partial P}{\partial x}$ and $\frac{\partial P}{\partial y}$ from Darcy's equation, we see that $\nabla P \cdot \nabla \psi = 0$. Hence, $P$ and $\psi$ are orthogonal curves for both the uniform and heterogeneous medium alike.

**Boundary Conditions**

Defining boundary conditions in terms of $P$ and $\psi$ for a quadrant of a five-spot pattern we have the boundary conditions along the lower and right boundaries

$$\frac{\partial P}{\partial y} = 0 \quad \left( \frac{\partial \psi}{\partial x} = 0 \right) \quad \text{and} \quad \frac{\partial P}{\partial x} = 0 \quad \left( \frac{\partial \psi}{\partial y} = 0 \right)$$

By integrating

$$\left( \frac{\partial \psi}{\partial x} \right)_{y=0} = 0 \quad \text{and} \quad \left( \frac{\partial \psi}{\partial y} \right)_{x=x_e} = 0$$

we have,

$$\psi_{y=0} = \psi_{x=x_e} = \text{constant},$$

that is, a Dirichlet boundary condition is obtained and is assigned arbitrarily a value of $-\frac{Q}{2}$. Similarly, for the left and upper boundaries, the boundary streamlines are assigned a value of $\frac{Q}{2}$. And, at the source/sink locations, a value of 0.0 was assigned corresponding to the diagonal streamline.
Discretizing the governing equation, as before, and omitting $\mu$,

\[
\frac{\partial}{\partial x} \left[ \frac{1}{k} \frac{\partial \psi}{\partial x} \right] \equiv \frac{1}{k_{L+1/2,J}} \frac{\psi_{L+1,J} - \psi_{L,J}}{\Delta x} - \frac{1}{k_{L-1/2,J}} \frac{\psi_{L,J} - \psi_{L-1,J}}{\Delta x}
\]

\[
\frac{\partial}{\partial y} \left[ \frac{1}{k} \frac{\partial \psi}{\partial y} \right] \equiv \frac{1}{k_{L,J+1/2}} \frac{\psi_{L,J+1} - \psi_{L,J}}{\Delta y} - \frac{1}{k_{L,J-1/2}} \frac{\psi_{L,J} - \psi_{L,J-1}}{\Delta y}
\]

Hence the stream function governing equation can be put in Equation (3.8) form and solved to obtain the stream function distribution at the grid points which may then be used to calculate the velocity field and hence compute the isochrones and recovery history. The streamlines obtained from this approach can be compared with those obtained from the pressure formulation to assess quantitatively of numerical approximations.

The same computer coded procedure developed to obtain the pressure field can be utilized to solve for the stream function by changing the boundary conditions.
3.3.2 Finite Element Formulation

3.3.2.1 Pressure Approach

Minimum energy dissipation governs the process of fluid flow in porous media. Based on the results of the calculus of variations, the problem of solving Laplace's equation so as to satisfy preassigned boundary conditions is equivalent to that of finding a function $P(x, y)$ satisfying those boundary conditions for which

$$
\iint \left[ \frac{1}{2} k(x, y) \left( \frac{\partial P}{\partial x} \right)^2 + \frac{1}{2} k(x, y) \left( \frac{\partial P}{\partial y} \right)^2 \right] \, dx \, dy
$$

is a minimum (Muskat, 1937).

There are several approaches to formulating the approximating integral equations describing the finite element method. Of the various approaches, the Rayleigh-Ritz based on the calculus of variations has been used extensively; Rayleigh-Ritz, however, requires the existence of a functional which requires that the physical problem has a variational formulation. Most engineering problems are expressed, not in terms of a functional, but in terms of a set of governing differential equations and boundary conditions, the approximate solution of which may be obtained by the weighted residual methods. The Galerkin method is a special case of the general class of the weighted residual methods. The Galerkin method is more general and those problems solved using Rayleigh-Ritz procedure represent a subclass of those amenable to solution by the Galerkin method (Pinder and Gray, 1977). The weighted residual method starts with the governing differential equation
method, one attempts to approximate \( (x) \) by a trial solution \( u(x) \) which may be made up of

with the boundary \( \Gamma \), and \( x \) represents the space co-ordinates \( (x,y) \). In the weighted residual

where \( I \) is a differential operator and \( n \) is the dependent variable defined in a region \( \Omega \)

With the boundary condition

\[
(\Omega \), \quad (x)f = (x)n
\]

Problem whose governing differential equation is given by

To illustrate the application of the Galerkin approach, we consider the boundary value

Galerkin's Method approximates the governing differential equation by substituting a trial

\[
 \mathbf{R} = (\Omega)(x)f - (\Omega)(x)n
\]

and assumes an approximate solution \( u \) which defines a residual given by

\[
(\Omega)(x)f = (\Omega)(x)n
\]
a linear combination of suitable functions and satisfies the boundary conditions of the problem, and is given by

$$\hat{u}(\vec{x}) = g(\vec{x}) + \sum_{i=1}^{n} c_i N_i(\vec{x})$$

where $N_i(\vec{x})$ are interpolating functions (also called shape functions) and $c_i$ are unknown constants.

Substitution of the trial solution $\hat{u}(\vec{x})$ into the governing differential equation will result in a residual function defined by

$$R(\vec{x}) = L \hat{u}(\vec{x}) - f(\vec{x})$$

or

$$R(\vec{x}) = L \left[ g(\vec{x}) + \sum_{i}^{n} c_i N_i(\vec{x}) \right] - f(\vec{x})$$

where $n$ is the number of elements.

In the weighted residuals method, the approximate solution is sought such that the integral of the residuals weighted by the function $\omega(\vec{x})$ is

$$\int_{\Omega} \omega_i(\vec{x}) R(\vec{x}) d\Omega = 0,$$

where $\Omega$ pertains to the subregions of the element.
In the Galerkin method, the weights $\omega_i(x)$ are the same interpolation functions, $N_i(\vec{\tau})$; that is, the weight functions are taken to be the same spatial function used to approximate $u(\vec{\tau})$, hence

$$\int_{\Omega} N_i(\vec{\tau}) R(\vec{\tau}) d\Omega = 0. \quad (3.20)$$

where $i$ designates the node index in the element.

That is, the Galerkin method forces the residual to be zero by making it to be orthogonal to each member $N_i(\vec{\tau})$ of a complete set of interpolation functions.

**Computation of the Element Matrices**

Let

$$L u(\vec{\tau}) = f(\vec{\tau})$$

be

$$\frac{\partial}{\partial x} \left( k(x,y) \frac{\partial u}{\partial x} \right) + \frac{\partial}{\partial y} \left( k(x,y) \frac{\partial u}{\partial y} \right) + \mu q(x,y) = 0$$

where $u$ stands for the pressure, $P$ in this case, and $q(x,y)$ is a sink function. The boundary conditions are given by:

$$u = u_0 \quad \text{on } \Gamma_1 \text{ (Dirichlet Boundary Condition)}$$

$$k(x,y) \left[ \frac{\partial u}{\partial x} n_x + \frac{\partial u}{\partial y} n_y \right] + q(x,y) = 0 \quad \text{on } \Gamma_2 \text{ (Neumann Boundary Condition)}$$
where $\Gamma_1$ and $\Gamma_2$ form the boundary of $\Omega$. The boundary condition on $\Gamma_2$ represents the normal flux through the boundary.

Let the trial solution on the element ($e$) be defined by an interpolation function:

$$\hat{u}(x, y) \equiv N'u^e(x, y)$$

where $N'$ is the vector transpose of interpolation function and $u^e(x, y)$ are the element known values of $u$ at its nodes.

Then Equation (3.20) may be written as

$$\int \int N \left[ \frac{\partial}{\partial x} \left( k(x, y) \frac{\partial \hat{u}}{\partial x} \right) + \frac{\partial}{\partial y} \left( k(x, y) \frac{\partial \hat{u}}{\partial y} \right) + q(x, y) \right] \, dx \, dy \quad (3.21)$$

The trial solution is required to be twice differentiable. The Galerkin Method, however, uses a piece interpolation function for $\hat{u}$, hence the first derivatives of $u$ will be piecewise continuous, while the second derivatives do not exist. The problem can be restated using integration by parts in two dimensions (which is the application of Green's Theorem of Equation (3.21)) to give

$$- \int \int_{\Omega} \left[ k(x, y) \frac{\partial N}{\partial x} \frac{\partial \hat{u}}{\partial x} + k(x, y) \frac{\partial N}{\partial y} \frac{\partial \hat{u}}{\partial y} - N q(x, y) \right] \, dx \, dy \quad (3.22)$$

$$+ \int_{\Gamma} \left[ k(x, y) \frac{\partial \hat{u}}{\partial x} n_x + k(x, y) \frac{\partial \hat{u}}{\partial y} n_y \right] \, N \, d\Gamma = 0$$
where \( \mathcal{U} \) stands for an approximate solution and \( n_x \) and \( n_y \) are the directional cosines in the \( x \) and \( y \) directions.

Further, the derivatives \( \frac{\partial \mathcal{U}}{\partial x} \) and \( \frac{\partial \mathcal{U}}{\partial y} \) are to be replaced by

\[
\begin{align*}
\frac{\partial \mathcal{U}}{\partial x} &= \frac{\partial N}{\partial x} u_i^e \\
\frac{\partial \mathcal{U}}{\partial y} &= \frac{\partial N}{\partial y} u_i^e
\end{align*}
\]

where \( M \) is the number of nodes in the element, \( e \).

It is remarked here Equation (3.22) is the counterpart of Equation (3.8) in the finite difference formulation. If further, the derivatives of \( N^e \) are written in terms of the nodal co-ordinates, an algebraic system of equations is obtained for each element which will be three algebraic equations for a triangular element and four algebraic equations for the rectangular element. For an elliptical problem, the operator \( L \) will result in a symmetric coefficient matrix for the element. The coefficient matrix is referred to as the element stiffness matrix.

Incorporation of Boundary Conditions

The no-flow boundary imposed before in Section 3.1 is restated by the second integral on the L.H.S. of Equation (3.22).

\[
\int_{\Gamma} \left[ k(x,y) \frac{\partial \mathcal{U}}{\partial x} n_x + k(x,y) \frac{\partial \mathcal{U}}{\partial y} n_y \right] N \, d\Gamma
\]
For no-flow boundary, $\frac{\partial u}{\partial x} n_x = 0$ and $\frac{\partial u}{\partial y} n_y = 0$ in which case, the value of the contour integral is zero.

Specification of $u$ at any location in the grid constitutes the Dirichlet boundary condition.

**Interpolation functions**

The integral equation obtainable from the variational form shows that the interpolation function for the trial solution be should at least bilinear (Reddy, 1984). The interpolation function usually determines a value within the element in terms of its nodal values. In two-dimensional second-order problems, the correspondence between the number of nodes and the degree of the polynomial of the interpolation function is not unique. For instance, the polynomial

$$u(x,y) = c_1 + c_2x + c_3y$$

contains three linearly independent terms and is linear in both $x$ and $y$. On the other hand, the polynomial

$$u(x,y) = c_1 + c_2x + c_3y + c_4xy$$

contains four linearly independent terms, but is algebraically non-linear in the last term. The former requires an element with three nodes while the latter requires four nodes. However for the same nodal distribution, since the triangular element is smaller than the quadrilateral elements, less discretization error would result for the triangular form.
Linear Interpolation Function for the Three-node Triangular Element

A trial solution \( \tilde{u}(x, y) \) is defined within the triangular element by linearly interpolating the nodal values as given by Equation (3.23) where \( c_1, c_2, c_3 \) are coefficients that need to be determined from

\[
\begin{align*}
  u_1 &= u(x_1, y_1) = c_1 + c_2 x_1 + c_3 y_1 \\
  u_2 &= u(x_2, y_2) = c_1 + c_2 x_2 + c_3 y_2 \\
  u_3 &= u(x_3, y_3) = c_1 + c_2 x_3 + c_3 y_3
\end{align*}
\]

or in matrix form

\[
\begin{bmatrix}
  u_1 \\
  u_2 \\
  u_3
\end{bmatrix} =
\begin{bmatrix}
  1 & x_1 & y_1 \\
  1 & x_2 & y_2 \\
  1 & x_3 & y_3
\end{bmatrix}
\begin{bmatrix}
  c_1 \\
  c_2 \\
  c_3
\end{bmatrix}
\]

in which the nodes are numbered counterclockwise. Solving for \( c, c_2, c_3 \), we obtain

\[
\begin{align*}
  c_1 &= \frac{1}{2A_e} \left[ u_1(x_2y_3 - x_3y_2) + u_2(x_3y_1 - x_1y_3) + u_3(x_1y_2 - x_2y_1) \right] \\
  c_2 &= \frac{1}{2A_e} \left[ u_1(y_2 - y_3) + u_2(y_3 - y_1) + u_3(y_1 - y_2) \right] \\
  c_3 &= \frac{1}{2A_e} \left[ u_1(x_3 - x_2) + u_2(x_1 - x_3) + u_3(x_2 - x_1) \right]
\end{align*}
\]

(3.26)

where \( A_e \) is the arc of the triangle, expressed in terms of its coordinates at the vertices.

\[
2A_e =
\begin{bmatrix}
  1 & x_1 & y_1 \\
  1 & x_2 & y_2 \\
  1 & x_3 & y_3
\end{bmatrix}
\]

(3.27)

\[
= (x_2y_3 - x_3y_2) + (x_3y_1 - x_1y_3) + (x_1y_2 - x_2y_1)
\]
Consider the approximation given below by Eq. (3.28) for the rectangular element with linear interpolation functions for the four-node rectangular element.

\[ q^2c + c = (q, n) \]

\[ q^2 + q + c = (q, n) \]

\[ q^2c + c = (0, n) \]

\[ q + c = (0, n) \]
Solving for \( c_i (i = 1, 2, 3, 4) \), we obtain

\[
\begin{bmatrix}
  c_1 \\
  c_2 \\
  c_3 \\
  c_4
\end{bmatrix} =
\begin{bmatrix}
  1 & 0 & 0 & 0 \\
  1 & a & 0 & 0 \\
  1 & a & b & ab \\
  1 & 0 & b & 0
\end{bmatrix}^{-1}
\begin{bmatrix}
  u_1 \\
  u_2 \\
  u_3 \\
  u_4
\end{bmatrix} = \frac{1}{ab}
\begin{bmatrix}
  ab & 0 & 0 & 0 \\
  -b & b & 0 & 0 \\
  -a & 0 & 0 & a \\
  1 & -1 & 1 & 1
\end{bmatrix}
\begin{bmatrix}
  u_1 \\
  u_2 \\
  u_3 \\
  u_4
\end{bmatrix}
\]

(3.30)

Substituting Equation (3.30) into Equation (3.28), we obtain

\[
\begin{bmatrix}
  u_1 \\
  u_2 \\
  u_3 \\
  u_4
\end{bmatrix} = \begin{bmatrix}
  \phi_1 & \phi_2 & \phi_3 & \phi_4
\end{bmatrix}
\begin{bmatrix}
  u_1 \\
  u_2 \\
  u_3 \\
  u_4
\end{bmatrix}
\]

(3.31)

\[= \sum_{i=1}^{4} u_i \phi_i(\xi, \eta)\]

where

\[
\begin{align*}
\phi_1(\xi, \eta) &= \left(1 - \frac{\xi}{a}\right) \left(1 - \frac{\eta}{b}\right) \\
\phi_2(\xi, \eta) &= \frac{\xi}{a} \left(1 - \frac{\eta}{b}\right) \\
\phi_3(\xi, \eta) &= \frac{\xi}{a} \frac{\eta}{b} \\
\phi_4(\xi, \eta) &= \left(1 - \frac{\xi}{b}\right) \frac{\eta}{b}
\end{align*}
\]

(3.32)

which is a bilinear form of interpolation function. Quadratic and higher order polynomials for rectangular elements may be obtained by taking the tensor product of the x-direction one-dimensional interpolation function with y-direction one-dimensional interpolation function (Reddy, 1984).
Isoparametric Mapping for Curvilinear Coordinates

The versatility of the finite element method to handle complicated geometry was made use of to solve the governing equation and its boundary conditions on the curvilinear grid described by the isopotential stream function flow-net. The interpolation function for the rectangular element is given by Equation (3.32) in local coordinates. Attempting to construct similar interpolation functions in global coordinates \((x, y)\) for the general quadrilateral will require considerable algebraic manipulation because the sides of the quadrilateral are inclined to the axes and the interpolation function will not be linear along the element boundaries. Instead, it is easier to work in a local coordinate \((\xi, \eta)\) and transform to global coordinates by mapping. Derivatives are converted from \((\xi, \eta)\) to \((x, y)\) coordinates by applying the chain rule, and are expressed in the form of the Jacobian matrix \(J\).

\[
J = \begin{bmatrix}
\frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} \\
\frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta}
\end{bmatrix}
\]

and

\[
\begin{bmatrix}
\frac{\partial}{\partial x} \\
\frac{\partial}{\partial y}
\end{bmatrix} = J^{-1} \begin{bmatrix}
\frac{\partial}{\partial \xi} \\
\frac{\partial}{\partial \eta}
\end{bmatrix}
\]

The procedure is carried out in determining the terms of the stiffness matrix \(G_{Li}^{\xi}\) given by

\[
G_{Li}^{\xi} = \iint \left( \frac{\partial N_i^\xi}{\partial x} \frac{\partial N_j^\xi}{\partial x} + \frac{\partial N_i^\xi}{\partial y} \frac{\partial N_j^\xi}{\partial y} \right) dx \, dy
\]

by applying the inverse Jacobian.
\[
\begin{bmatrix}
\frac{\partial N_i^*}{\partial x} \\
\frac{\partial N_i^*}{\partial y}
\end{bmatrix} = [J]^{-1}
\begin{bmatrix}
\frac{\partial \tilde{N}_L^*}{\partial \xi} \\
\frac{\partial \tilde{N}_L^*}{\partial \eta}
\end{bmatrix}
\]

where \( \tilde{N}_L^* \) denotes the interpolation function in local coordinates. The other requirement for the transformation is that the area \( dx \ dy \) in the quadrilateral corresponds to \( |J| \ d\xi \ d\eta \) in the local coordinates. Hence, the transformed integral becomes

\[
G_{L,i}^* = \int_0^1 \int_0^1 \left[ \left( \frac{\partial \tilde{N}_L^*}{\partial \xi} \frac{\partial \tilde{N}_L^*}{\partial \eta} \right) \left( \frac{\partial \tilde{N}_L^*}{\partial \xi} \frac{\partial \tilde{N}_L^*}{\partial \eta} \right) + \left( \frac{\partial \tilde{N}_L^*}{\partial \xi} \frac{\partial \tilde{N}_L^*}{\partial \eta} \right) \left( \frac{\partial \tilde{N}_L^*}{\partial \eta} \frac{\partial \tilde{N}_L^*}{\partial \eta} \right) \right] |J| \ d\xi \ d\eta.
\]

The integral is easier to evaluate numerically using Gaussian quadrature which has the form

\[
\int \int f(\xi, \eta) \, d\xi \ d\eta = \sum_{i=1}^n \sum_{j=1}^n N_i N_j f(\xi_i, \eta_j)
\]

where \( N_i, N_j \) are the weighting functions and \( \xi_i, \eta_i \) are coordinate positions. The numerical integration procedure is included in the finite element computer program, given in Appendix B, that was coded in FORTRAN.

**Truncation Error**

The truncation for the finite element approximation scheme can be obtained by examining the error associated with the numerical differentiation and integration in Equation (3.21). The
truncation error is second order for the discretization and fourth order for the integration resulting in a second-order accurate scheme (Pinder and Gray, 1977). Thus the finite element accuracy, using linear interpolation functions, appears to be of the same order as that of the finite difference, however, better accuracy in the finite element approximation can be obtained by using higher order interpolation functions.

Assembly of Global Stiffness Matrix

The individual element matrices overlap at locations corresponding to common global coordinates. This in turn shows how, in the finite element method, the dependent variable at a particular node interacts with the other nodes. The local element coordinates are related to the global coordinates through the Boolean connectivity matrix in which the correspondence between the element and the global numbering is established in a predetermined order.

The element stiffness matrix for an elliptic partial differential equation is symmetric and its size is given by the number of its nodes. The global stiffness matrix is obtained by summing up the element contribution one at a time. When the element stiffness matrix is symmetric the global matrix is also symmetric and possesses the property of bandedness. The symmetry of the global matrix can be preserved after the incorporation of boundary conditions imposed by the problem by manipulation of the row and column corresponding to a prescribed variable location.

In anticipation of the less computer memory required by the finite element formulation as opposed to the finite difference formulation a "direct" method for solving the resultant symmetric banded matrix was used. The procedure for assembling and solving the resulting system of algebraic equations is included in the finite element computer program given in Appendix B.
3.3.2.2 Stream Function Approach

The governing equation given by Equation (3.16) based on the finite element formulation and pressure approach can be re-derived using the stream function approach. The boundary condition in the stream function approach is the specification of $\psi$ along the quadrant boundaries. Hence, the new equation takes the form

$$
\iint_{\Omega} \left[ k(x,y) \frac{\partial N}{\partial x} \frac{\partial \psi}{\partial x} + k(x,y) \frac{\partial N}{\partial y} \frac{\partial \psi}{\partial y} - N q(x,y) \right] \, dx \, dy = 0
$$

for the element. This approach, however, was not included in the computer implementation.
3.4 Velocity Field, Streamlines, and Front Tracing Procedure

The relative time of travel of the front between any two isopotential curves is directly proportional to the distance between the curves and inversely proportional to the pressure gradient. The time interval \( dt \) that a particle takes along a streamline to cover a differential distance \( ds \) is given by

\[
ds = \nu \, dt
\]

where \( \nu \) is the particle displacement velocity.

The cumulative time for traversing a streamline of length \( S \) is given by

\[
t = \int_{0}^{S} \frac{ds}{\nu}
\]

where \( \nu \) is given by Darcy's Law:

\[
\nu = \frac{k}{\phi \mu} \frac{d(\Delta P)}{ds}
\]

where \( \phi \) denotes porosity, hence

\[
t = \phi \mu \int_{0}^{S} \frac{ds}{k(x, y) \frac{d(\Delta P)}{ds}}
\]
or, in discrete form,

\[ t = \sum_i \Delta t_i = \phi \mu \sum_i \frac{(\Delta S_i)^2}{k(x, y) \Delta P_i} \]  \hspace{1cm} (3.33)

Equation (3.33) is based on streamline-isopotential coordinates. The solution of (3.33) would require the development of an interpolation function and contouring for both the stream function and pressure. A simpler approach is to use \((x, y)\) Cartesian coordinates and develop a continuous interpolation scheme for the components of velocities, \((v_x, v_y)\).

Applying Darcy's Law using the central difference approximation

\[ (v_x)_{ij} = \frac{-2(k_{i-1,j})(k_{i+1,j})}{\Delta x[(k_{i-1,j})(\phi \mu) + (k_{i+1,j})(\phi \mu)]} \left[ \frac{P_{i+1,j} - P_{i-1,j}}{2} \right] \]

\[ (v_y)_{ij} = \frac{-2(k_{i,j-1})(k_{i,j+1})}{\Delta y[(k_{i,j-1})(\phi \mu) + (k_{i,j+1})(\phi \mu)]} \left[ \frac{P_{i,j+1} - P_{i,j-1}}{2} \right] \]

These two equations will define the velocity field at the grid points. A bilinear interpolation scheme is then used to define a continuous interpolation function for the velocity components. The function, developed in local coordinates \((\xi, \eta)\) in Section 3.3.2 on finite element formulation, is given by

\[ \phi_1(\xi, \eta) = \left( 1 - \frac{\xi}{a} \right) \left( 1 - \frac{\eta}{b} \right) \]

\[ \phi_2(\xi, \eta) = \frac{\xi}{a} \left( 1 - \frac{\eta}{b} \right) \]

\[ \phi_3(\xi, \eta) = \frac{\xi}{a} \frac{\eta}{b} \]

\[ \phi_4(\xi, \eta) = \left( 1 - \frac{\xi}{b} \right) \frac{\eta}{b} \]  \hspace{1cm} (3.34)
and

\[
\nu(x, \eta) = (\phi_1 \phi_2 \phi_3 \phi_4)
\]

\[
\begin{bmatrix}
\nu_{x1} \\
\nu_{x2} \\
\nu_{x3} \\
\nu_{x4}
\end{bmatrix}
\]

\[
\nu(y, \eta) = (\phi_1 \phi_2 \phi_3 \phi_4)
\]

\[
\begin{bmatrix}
\nu_{y1} \\
\nu_{y2} \\
\nu_{y3} \\
\nu_{y4}
\end{bmatrix}
\]

The instantaneous particle location in the x and y direction will be

\[
x_{i+1} = x_i + \int_{t_i}^{t_{i+1}} \nu_x \, dt
\]

\[
y_{i+1} = y_i + \int_{t_i}^{t_{i+1}} \nu_y \, dt
\]

The \(x_{i+1}, y_{i+1}\) values determine the streamline coordinates at time \(t_{i+1}\). Finding the successive front locations requires determining the travel time of all particles at all times. The front at a particular time is found by joining all particle locations sharing the same time level. Particles are considered to have broken through if they fall within a magnitude of the wellbore radius near the production well.

The front tracking procedure was coded in FORTRAN and is included in the main computer code given in Appendix A.
3.5 Numerical Model Testing

Finite Difference Formulation

The finite difference procedure of the numerical model was tested for accuracy assuming:

(a) quadrant length = 660.0 ft. (40-acre spacing)
(b) uniform porous medium of transmissibility, \( kh/\mu = 1.0 \) (md. ft/cP)
(c) pressure at the Injector = 1400.0 psia
(d) flow rate at the Producer = 600.0 B/D

The computer program employing the five-point formula was tested for discretization error using the 32x32 and 64x64 grids. Both grids showed significant discrepancies in the absolute magnitude of the pressure solution as shown in Figure 3.4. However, the pressure profile of both the 32x32 and 64x64 grids track closely the analytical solution pressure profile. Based on the assumptions of steady state and incompressible flow, the resultant elliptic form of partial differential equation is independent of the absolute magnitude of the pressure and the accuracy is largely influenced by the derivative of the pressure.

The nine-point implementation of the finite difference algorithm showed a very slow rate of convergence using the point SOR method and was not considered a good candidate for our problem.

Finite Element Formulation

Results of the finite element computer program based on rectilinear coordinates for the triangular and rectangular elements are shown in Figure 3.5. Results show, in regards to the pressure spatial derivative, that the rectangular elements yielded more accurate results. The numerical
Figure 3.4 Exact and Approximate Pressure Profile Along Diagonal Streamline Finite Difference Solution
Figure 3.5 Exact and Approximate Pressure Profile Along Diagonal Streamline
Finite Element Solution, 32x32 Grid
accuracy of the 40x40 rectangular element compares closely to that of the 64x64 finite difference solution as shown in Figure 3.6.

The rectilinear formulation of both the finite difference and the finite element methods shows that the greater part of pressure drop occurs near the source and sink. Alternatively the orthogonal-curvilinear grid shown in Figure 3.7 will offer a flexible spatial resolution. The grid of Figure 3.7 was based on uniform pressure drop spacing of 42.65 psi on 26x26 grid of isobars and streamlines. The calculated uniform pressure drop spacing was based on a prescribed pressure at the injector of 1400.0 and flowing pressure at the producer of 333.7 psia assuming a wellbore radius of 0.66 feet. An additional isobar in the middle was added to enable following the natural curvature of the streamlines. In this grid, the sink will be defined by a uniform string of 26 sinks, each with a strength equal to 1/26th of that in the rectilinear form. Similarly the sandface pressure at the injector well defined at 26 points. The resultant pressure solution given in Figure 3.8 shows a discrepancy of less than 0.5%. The difficulty with the varying grid spacing is the possible lack of compatibility with a heterogeneity correlation scale defined by the geostatistical problem. However, such a grid may be used to estimate the pattern effective permeability since it can formulate the flow system as a bundle of streamtubes flowing in parallel and within each streamtube flow may be regarded as linear by the choice of uniform pressure drop spacing.

The 64x64 finite difference grid was favored over the 40x40 finite element grid because the heterogeneity correlation scale considered in our problem was of relatively short length. The 64x64 finite difference grid was used for the solution of both the pressure and stream function approaches.

The uniform medium solution of the governing equation for pressure and the computed streamlines are shown in the flownet of Figure 3.9. The consequent isochrone computations is shown in Figure 3.10.
Figure 3.7 Orthogonal-Curvilinear Grid (Upper Ocean) (26x27 Grid)
Figure 3.8  Exact and Approximate Pressure Profile Along Diagonal Streamline  
Finite Element (26x27 Grid)
Figure 3.9 Uniform Medium Flownet.
The stream function solution of the governing partial differential equation for the uniform medium case is shown in Figure 3.11. Attempts to solve the equation for the heterogeneous medium case did not meet with success because of the method of linear search for locations sharing the same magnitude of stream function. Refinement of the computational procedure by using the conjugate gradient method is recommended.
Figure 3.11 Uniform Medium Streamlines  
(Stream Function Solution)
Chapter 4: Results and Discussion

4.1 Variation of Recovery Characteristics with Boundary Translation

Input Data:

The following data pertains to the dimensions of the five-spot quadrant, formation, and fluid properties and operating boundary conditions.

- Quadrant length = 660.0 feet (40-acres spacing)
- Formation thickness, $h$ = 20.0 feet
- Effective porosity, $\phi$ = 0.20
- Base permeability, $k_o$ = 50 md.
- Fluid viscosity, $\mu$ = 1.0 cP
- Base transmissibility, $T_o$ = 1.0 (md. ft./cP)
- Pressure at injection well = 1400 psia
- Steady state producing rate = 600 B/D

Generation of Subrealizations of Random Permeability Field

Sixteen subrealizations of the natural logarithm of the permeability field were generated following the procedure described in Section 3.2.1. A value of 2.0 was chosen for $\alpha$, the scale factor which determines the width of the distribution and a value of 288 was assigned for the number of source points, $N_{sp}$. A seed for random numbers generation was assigned. As given in Section 3.2.1, the magnitude of displacement was made in steps of 0.04 of the quadrant length. Hence, for a quadrant of length of 660.0 feet, the magnitude of the displacements are
(-26.4, 0.0, 26.4, 52.8 feet) with the angle formed by the positive x-axis and the line joining the
injector location for the base case to those of subrealizations ranging from 0.0 to 90.0 in steps
of 22.5°. The locations of the injection well for the different subrealization relative to the base
realization are shown to scale in Figure 4.1.

The variogram plot of the \( \log_{10} k(x,y) \) for the base realization is shown in Figure 4.2. The
plot suggests a heterogeneity correlation scale in the range of 40.0 to 50.0 feet. The 64x64 grid
blocks used to discretize the quadrant have a square block of length 10.4762 feet. Hence, this
permits shifting the quadrant boundaries within the larger of the formation by order of magnitude
of the simulation block dimension.

In computing the pressure field or stream function, a value of 1.0 was used for the base
transmissibility. The value, of the different kinds of mean transmissibility, namely the
arithmetic, the harmonic, and the geometric generated by the SPM are given in Table 4.1. The
values of the geometric means for the different subrealization were invariably close to 1.00 while
arithmetic and harmonic showed varying but small spread. The magnitudes of the three means
were such that the arithmetic is always the highest while the harmonic was the lowest. This fact
can be illustrated by a multi-block flow system in which the effective mean permeability would
approach the arithmetic if blocks were arranged in a parallel fashion and would approach the
harmonic if arranged in series. Computation of the geometric mean was performed using a word
size of 64 bit.

Pressure Field, Streamlines, and Front Tracking

The isobars-streamlines plots (flow-net) for three subrealizations are given in uniform intervals
of pressure and streamlines decrements. The pressure steps were 40 psia. The flow-net for the
base realization is shown in Figure 4.3. The flow-net for subrealization 5, which is one of the
Figure 4.1 Relative Injector Location in the Different Subcalculations
Table 4.1: Means of the Permeability Distribution

<table>
<thead>
<tr>
<th>Subrealization</th>
<th>Arithmetic</th>
<th>Geometric</th>
<th>Harmonic</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1.6372</td>
<td>1.0371</td>
<td>0.7431</td>
</tr>
<tr>
<td>2</td>
<td>1.4310</td>
<td>0.9890</td>
<td>0.7324</td>
</tr>
<tr>
<td>3</td>
<td>1.6023</td>
<td>1.0121</td>
<td>0.7296</td>
</tr>
<tr>
<td>4</td>
<td>1.5131</td>
<td>1.0052</td>
<td>0.7251</td>
</tr>
<tr>
<td>5</td>
<td>1.5325</td>
<td>0.9904</td>
<td>0.7278</td>
</tr>
<tr>
<td>6</td>
<td>1.5313</td>
<td>1.0038</td>
<td>0.7282</td>
</tr>
<tr>
<td>7</td>
<td>1.4873</td>
<td>0.9859</td>
<td>0.7283</td>
</tr>
<tr>
<td>8</td>
<td>1.6311</td>
<td>1.0283</td>
<td>0.7374</td>
</tr>
<tr>
<td>9</td>
<td>1.4682</td>
<td>0.9947</td>
<td>0.7284</td>
</tr>
<tr>
<td>10</td>
<td>1.5591</td>
<td>0.9980</td>
<td>0.7206</td>
</tr>
<tr>
<td>11</td>
<td>1.5432</td>
<td>1.0206</td>
<td>0.7334</td>
</tr>
<tr>
<td>12</td>
<td>1.3743</td>
<td>0.9608</td>
<td>0.7068</td>
</tr>
<tr>
<td>13</td>
<td>1.7237</td>
<td>1.0569</td>
<td>0.7486</td>
</tr>
<tr>
<td>14</td>
<td>1.4261</td>
<td>0.9875</td>
<td>0.7225</td>
</tr>
</tbody>
</table>
Figure 4.3: Power for Base Realizations
closest subrealizations to the base realization, is shown in Figure 4.4 while that for subrealization 14, being one of the farthest, is shown in Figure 4.5.

The streamlines computation for subrealizations 15 and 16 (the farthest subrealizations from the base case) yielded anomalous results depicted as some stagnant streamlines at the boundaries closer to the producer. These two subrealizations were discarded from the ensemble considered.

The computed frontal locations prior to breakthrough, at breakthrough and past breakthrough are shown for the base realization and subrealizations 5 and 14 in Figures 4.6, 4.7, and 4.8, respectively. Tabular values of the recovery history at the indicated frontal times are summarized in Table 4.2 for the fourteen subrealizations considered while the recovery history for the three representative cases of the base realization and subrealization 5 and 14 are given in Figure 4.9.

The time-to-breakthrough of the different subrealizations showed some variability with values generally exceeding but not greatly different from 589.21 days, the value for a uniform medium. This is to be expected since in a heterogeneous medium the streamline paths become more sinuous. The variability in breakthrough times may be explained in part by the fact that breakthrough would not always be along the streamline of shortest distance as in the uniform medium case. On the extreme side, subrealization 8 showed a relatively early breakthrough time of 505.08 days while subrealization 13 showed the greatest breakthrough time of 698.95 days. On the whole, the average time-to-breakthrough of the 14 subrealizations was 597.73 days which is to be compared with 589.21 days for the uniform medium case.

As can be seen from the graph of the data in Table 4.2, the variations in breakthrough time do not describe all features of the differences between these subrealizations. Areal sweep efficiency for a miscible flood may be defined as the fraction of the quadrant area invaded by
Figure 4.5: Flownet for Submergence I4
Figure 4.7: Isochrones for Subrealization 5
Figure 4.8: Isoclines for Subtraction
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>622.6</td>
<td>.72</td>
<td>.72</td>
<td>7</td>
<td>635.4</td>
<td>.71</td>
<td>.71</td>
</tr>
<tr>
<td></td>
<td>250.0</td>
<td>.28</td>
<td>.28</td>
<td>250.0</td>
<td>28</td>
<td>.28</td>
<td>.28</td>
</tr>
<tr>
<td></td>
<td>375.0</td>
<td>.43</td>
<td>.42</td>
<td>375.0</td>
<td>42</td>
<td>.42</td>
<td>.41</td>
</tr>
<tr>
<td></td>
<td>500.0</td>
<td>.57</td>
<td>.55</td>
<td>500.0</td>
<td>.56</td>
<td>.56</td>
<td>.54</td>
</tr>
<tr>
<td></td>
<td>750.0</td>
<td>.86</td>
<td>.80</td>
<td>750.0</td>
<td>.84</td>
<td>.84</td>
<td>.76</td>
</tr>
<tr>
<td></td>
<td>1000.0</td>
<td>1.15</td>
<td>.92</td>
<td>1000.0</td>
<td>1.11</td>
<td>.92</td>
<td>.87</td>
</tr>
<tr>
<td></td>
<td>1250.0</td>
<td>1.44</td>
<td>.95</td>
<td>1250.0</td>
<td>1.39</td>
<td>.92</td>
<td>.92</td>
</tr>
<tr>
<td></td>
<td>1500.0</td>
<td>1.72</td>
<td>.97</td>
<td>1500.0</td>
<td>1.67</td>
<td>.97</td>
<td>.97</td>
</tr>
<tr>
<td>2</td>
<td>582.3</td>
<td>.67</td>
<td>.67</td>
<td>8</td>
<td>505.1</td>
<td>.67</td>
<td>.67</td>
</tr>
<tr>
<td></td>
<td>125.0</td>
<td>.14</td>
<td>.12</td>
<td>125.0</td>
<td>.17</td>
<td>.17</td>
<td>.17</td>
</tr>
<tr>
<td></td>
<td>250.0</td>
<td>.28</td>
<td>.28</td>
<td>250.0</td>
<td>.33</td>
<td>.33</td>
<td>.32</td>
</tr>
<tr>
<td></td>
<td>375.0</td>
<td>.43</td>
<td>.41</td>
<td>375.0</td>
<td>.50</td>
<td>.50</td>
<td>.49</td>
</tr>
<tr>
<td></td>
<td>500.0</td>
<td>.57</td>
<td>.56</td>
<td>500.0</td>
<td>.67</td>
<td>.67</td>
<td>.67</td>
</tr>
<tr>
<td></td>
<td>750.0</td>
<td>.86</td>
<td>.78</td>
<td>750.0</td>
<td>1.00</td>
<td>.88</td>
<td>.88</td>
</tr>
<tr>
<td></td>
<td>1000.0</td>
<td>1.15</td>
<td>.88</td>
<td>1000.0</td>
<td>1.33</td>
<td>.95</td>
<td>.95</td>
</tr>
<tr>
<td></td>
<td>1250.0</td>
<td>1.43</td>
<td>.95</td>
<td>1250.0</td>
<td>1.67</td>
<td>.98</td>
<td>.98</td>
</tr>
<tr>
<td></td>
<td>1500.0</td>
<td>1.72</td>
<td>.97</td>
<td>1500.0</td>
<td>2.00</td>
<td>.98</td>
<td>.98</td>
</tr>
<tr>
<td>3</td>
<td>614.2</td>
<td>.68</td>
<td>.68</td>
<td>9</td>
<td>556.1</td>
<td>.65</td>
<td>.65</td>
</tr>
<tr>
<td></td>
<td>125.0</td>
<td>.14</td>
<td>.14</td>
<td>125.0</td>
<td>.15</td>
<td>.15</td>
<td>.13</td>
</tr>
<tr>
<td></td>
<td>250.0</td>
<td>.28</td>
<td>.28</td>
<td>250.0</td>
<td>.28</td>
<td>.28</td>
<td>.28</td>
</tr>
<tr>
<td></td>
<td>375.0</td>
<td>.43</td>
<td>.43</td>
<td>375.0</td>
<td>.44</td>
<td>.44</td>
<td>.40</td>
</tr>
<tr>
<td></td>
<td>500.0</td>
<td>.56</td>
<td>.56</td>
<td>500.0</td>
<td>.58</td>
<td>.58</td>
<td>.55</td>
</tr>
<tr>
<td></td>
<td>750.0</td>
<td>.83</td>
<td>.80</td>
<td>750.0</td>
<td>.88</td>
<td>.88</td>
<td>.75</td>
</tr>
<tr>
<td></td>
<td>1000.0</td>
<td>1.11</td>
<td>.90</td>
<td>1000.0</td>
<td>1.17</td>
<td>.90</td>
<td>.89</td>
</tr>
<tr>
<td></td>
<td>1250.0</td>
<td>1.39</td>
<td>.94</td>
<td>1250.0</td>
<td>1.46</td>
<td>.95</td>
<td>.95</td>
</tr>
<tr>
<td></td>
<td>1500.0</td>
<td>1.67</td>
<td>.97</td>
<td>1500.0</td>
<td>1.75</td>
<td>.97</td>
<td>.97</td>
</tr>
<tr>
<td>4</td>
<td>544.9</td>
<td>.63</td>
<td>.63</td>
<td>10</td>
<td>579.2</td>
<td>.66</td>
<td>.66</td>
</tr>
<tr>
<td></td>
<td>250.0</td>
<td>.28</td>
<td>.28</td>
<td>250.0</td>
<td>.28</td>
<td>.28</td>
<td>.28</td>
</tr>
<tr>
<td></td>
<td>375.0</td>
<td>.43</td>
<td>.40</td>
<td>375.0</td>
<td>.43</td>
<td>.43</td>
<td>.43</td>
</tr>
<tr>
<td></td>
<td>500.0</td>
<td>.57</td>
<td>.55</td>
<td>500.0</td>
<td>.57</td>
<td>.57</td>
<td>.56</td>
</tr>
<tr>
<td></td>
<td>750.0</td>
<td>.86</td>
<td>.77</td>
<td>750.0</td>
<td>.85</td>
<td>.85</td>
<td>.81</td>
</tr>
<tr>
<td></td>
<td>1000.0</td>
<td>1.15</td>
<td>.90</td>
<td>1000.0</td>
<td>1.14</td>
<td>.92</td>
<td>.92</td>
</tr>
<tr>
<td></td>
<td>1250.0</td>
<td>1.43</td>
<td>.95</td>
<td>1250.0</td>
<td>1.42</td>
<td>.95</td>
<td>.95</td>
</tr>
<tr>
<td></td>
<td>1500.0</td>
<td>1.72</td>
<td>.96</td>
<td>1500.0</td>
<td>1.70</td>
<td>.98</td>
<td>.98</td>
</tr>
<tr>
<td>5</td>
<td>606.2</td>
<td>.69</td>
<td>.69</td>
<td>11</td>
<td>612.8</td>
<td>.68</td>
<td>.68</td>
</tr>
<tr>
<td></td>
<td>250.0</td>
<td>.28</td>
<td>.27</td>
<td>250.0</td>
<td>.28</td>
<td>.28</td>
<td>.28</td>
</tr>
<tr>
<td></td>
<td>375.0</td>
<td>.43</td>
<td>.44</td>
<td>375.0</td>
<td>.42</td>
<td>.42</td>
<td>.41</td>
</tr>
<tr>
<td></td>
<td>500.0</td>
<td>.57</td>
<td>.56</td>
<td>500.0</td>
<td>.56</td>
<td>.56</td>
<td>.54</td>
</tr>
<tr>
<td></td>
<td>750.0</td>
<td>.86</td>
<td>.80</td>
<td>750.0</td>
<td>.84</td>
<td>.84</td>
<td>.80</td>
</tr>
<tr>
<td></td>
<td>1000.0</td>
<td>1.14</td>
<td>.92</td>
<td>1000.0</td>
<td>1.11</td>
<td>.90</td>
<td>.90</td>
</tr>
<tr>
<td></td>
<td>1250.0</td>
<td>1.43</td>
<td>.96</td>
<td>1250.0</td>
<td>1.39</td>
<td>.95</td>
<td>.95</td>
</tr>
<tr>
<td></td>
<td>1500.0</td>
<td>1.71</td>
<td>.98</td>
<td>1500.0</td>
<td>1.67</td>
<td>.97</td>
<td>.97</td>
</tr>
<tr>
<td>6</td>
<td>616.0</td>
<td>.68</td>
<td>.68</td>
<td>12</td>
<td>598.0</td>
<td>.67</td>
<td>.67</td>
</tr>
<tr>
<td></td>
<td>250.0</td>
<td>.28</td>
<td>.28</td>
<td>250.0</td>
<td>.28</td>
<td>.28</td>
<td>.28</td>
</tr>
<tr>
<td></td>
<td>375.0</td>
<td>.41</td>
<td>.42</td>
<td>375.0</td>
<td>.42</td>
<td>.42</td>
<td>.41</td>
</tr>
<tr>
<td></td>
<td>500.0</td>
<td>.55</td>
<td>.54</td>
<td>500.0</td>
<td>.56</td>
<td>.56</td>
<td>.54</td>
</tr>
<tr>
<td></td>
<td>750.0</td>
<td>.84</td>
<td>.80</td>
<td>750.0</td>
<td>.84</td>
<td>.84</td>
<td>.75</td>
</tr>
<tr>
<td></td>
<td>1000.0</td>
<td>1.11</td>
<td>.84</td>
<td>1000.0</td>
<td>1.12</td>
<td>.90</td>
<td>.90</td>
</tr>
<tr>
<td></td>
<td>1250.0</td>
<td>1.38</td>
<td>.94</td>
<td>1250.0</td>
<td>1.40</td>
<td>.96</td>
<td>.96</td>
</tr>
<tr>
<td></td>
<td>1500.0</td>
<td>1.66</td>
<td>.97</td>
<td>1500.0</td>
<td>1.68</td>
<td>.98</td>
<td>.98</td>
</tr>
</tbody>
</table>
Table 4.2: Subrealizations Recovery History

Continued

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>13</td>
<td>698.9</td>
<td>.82</td>
<td>.82</td>
<td>14</td>
<td>596.7</td>
<td>.66</td>
<td>.66</td>
</tr>
<tr>
<td></td>
<td>125.0</td>
<td>.15</td>
<td>.15</td>
<td></td>
<td>125.0</td>
<td>.14</td>
<td>.13</td>
</tr>
<tr>
<td></td>
<td>250.0</td>
<td>.29</td>
<td>.27</td>
<td></td>
<td>250.0</td>
<td>.28</td>
<td>.26</td>
</tr>
<tr>
<td></td>
<td>375.0</td>
<td>.44</td>
<td>.44</td>
<td></td>
<td>375.0</td>
<td>.42</td>
<td>.39</td>
</tr>
<tr>
<td></td>
<td>500.0</td>
<td>.58</td>
<td>.58</td>
<td></td>
<td>500.0</td>
<td>.56</td>
<td>.52</td>
</tr>
<tr>
<td></td>
<td>750.0</td>
<td>.88</td>
<td>.84</td>
<td></td>
<td>750.0</td>
<td>.84</td>
<td>.81</td>
</tr>
<tr>
<td></td>
<td>1000.0</td>
<td>1.17</td>
<td>.93</td>
<td></td>
<td>1000.0</td>
<td>1.11</td>
<td>.90</td>
</tr>
<tr>
<td></td>
<td>1250.0</td>
<td>1.46</td>
<td>.96</td>
<td></td>
<td>1250.0</td>
<td>1.39</td>
<td>.94</td>
</tr>
<tr>
<td></td>
<td>1500.0</td>
<td>1.75</td>
<td>.97</td>
<td></td>
<td>1500.0</td>
<td>1.67</td>
<td>.98</td>
</tr>
</tbody>
</table>
92

Base Realization

Subrealization 5

Subrealization 14

Figure 4.9 Recovery History Plots
the displacing fluid at different times. The variation in the shape of these curves for the different subrealizations was not so great. For instance, subrealization 13, in which the breakthrough was delayed, showed a high sweep efficiency at that time. The variation among these curves may be explained by the possible influence of the pattern geometrical factor ensuing from the relative location of the particular heterogeneities in the medium and the magnitude of the heterogeneity correlation scale.

Test of Significance of the Variability of Recovery History

Testing the significance of the variability of the recovery history of the different subrealizations may be approached statistically. Experimental sets of data, which are believed to come from the same parent population, may be tested for the equality of variances by formulating a statistical hypothesis and a level of significance for the test. The hypotheses are based on statistical distributions, while the level of significance is a matter of judgment of what probability level we take as being significant to reject or accept a certain hypothesis. A widely used test is that based on the Fisher variance ratio distribution (F-test), which may be used to test successive pairs of variances or simply the ratio of the highest to lowest variance. Alternatively, Brownlee (1953) suggests the use of Bartlett’s test of comparing several variances. The Bartlett’s test as applicable to our problem is outlined as follows:

(a) compute the natural log of the variance for each set
(b) compute the mean-variances of the sets, \( S^2 \)
(c) compute the degrees of freedom as the number of observations less one for each set
(d) choose a probability level of significance based on which we reject the hypothesis of equality of variances
(e) compute two terms B and C as follows:
\[ B = kn \log_e S^2 - n \sum \log_e \sigma_i^2 \]
\[ C = 1 + \frac{k+1}{3nk} \]

where:
- \( \sigma_i^2 \) = ith set variance
- \( S^2 \) = mean variance
- \( k \) = number of variances (not to be confused with permeability \( k(x, y) \))
- \( n-1 \) = degrees of freedom

The criterion is such that if \( B/C \) is greater than the value of chi-square (obtained from \( X^2 \) distribution) at the assigned level of significance and degrees of freedom, then we conclude that the sets (here subrealizations) are not drawn from a statistically homogeneous population. Inputting a value of \( k = 14 \) and \( (n-1) = 8 \), the parameters \( B \) and \( C \) were computed as 0.3140 and 1.0446, respectively. Entering an \( X^2 \)-table at a level of significance of 0.05 and \( B/C \) value of 0.3006 shows that \( B/C \) is far less than the \( X^2 \)-value which gives evidence that these subrealizations are drawn from the same parent population.

**Fisher Variance-Ratio Test (F-test)**

Since the F-test compares the spread of two sets of data, we take the largest variance and smallest variance for this test. Associated with the variance ratio test are two values of degrees of freedom of the larger variance, \( n_1 \), and that of the smaller variance, \( n_2 \). Here \( n_1 \) and \( n_2 \) are equal to 8 since we have 9 points in the recovery curve. Entering the F-distribution tables with these values of \( n_1 \) and \( n_2 \) and choosing a level of significance of 0.05, we obtain
From the same statistically homogeneous population, the theoretical value, the F-test further corroborates that the ensemble of subrealization are drawn compared to an experimental value of 1.82. Since the experimental value is less than the

\[ F_{0.05} = 3.44 \]
4.2 Effect of Heterogeneity Characteristics on Pressure Distribution

The effect of spatial auto covariance structure of a heterogeneous permeability field on the reduction of its permeability variance and consequently on the variance of the resultant pressure field in a flow system was also investigated by applying the spectral representation theorem that incorporates the Fast Fourier algorithm (Gutjahr, 1989). In the discrete Fourier method, the shortest wave length is dictated by the spacing in the spatial domain \((\Delta x, \Delta y)\). In order to investigate the effect of the autocovariance structure on the variance of the pressure field, we would choose Fourier block dimensions large enough to allow varying the range of the correlation scale. The accuracy dictated by the finite difference formulation in solving the pressure field calls for fine spacing that is small grid blocks in discretizing the spatial domain.

Four realizations of a two-dimensional permeability field of varying Fourier block spacing and a fixed correlation length were studied. In the first and second, the spacing was kept smaller than the correlation length while in the third and fourth realization the spacing was allowed to exceed the correlation length as shown in Table 4.3. The same input data of quadrant dimensions, formation and fluid properties and boundary conditions for solving the governing equation were also used here. In addition, in the spectral method, we have to specify:

(a) the type of covariance function and the magnitude of its variance
(b) the magnitude of the correlation scale
(c) spacing in x- and y-directions
(d) seed for random numbers generation.
Table 4.3: Input Data and Results of the Spectral Method

<table>
<thead>
<tr>
<th>Realization</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Variance Value of Covariance Function</td>
<td>0.55</td>
<td>0.55</td>
<td>0.55</td>
<td>0.55</td>
</tr>
<tr>
<td>Heterogeneity Scale (feet)</td>
<td>42.0</td>
<td>42.0</td>
<td>42.0</td>
<td>42.0</td>
</tr>
<tr>
<td>Spacing ($\Delta x = \Delta y$) (feet)</td>
<td>16.8</td>
<td>33.6</td>
<td>84.0</td>
<td>210.0</td>
</tr>
<tr>
<td>Means of $\log_e$ Transmissibility</td>
<td>0.0032</td>
<td>0.0189</td>
<td>0.0814</td>
<td>0.0034</td>
</tr>
<tr>
<td>Variance of $\log_e$ Transmissibility</td>
<td>0.4468</td>
<td>0.4201</td>
<td>0.2798</td>
<td>0.1021</td>
</tr>
<tr>
<td>Pressure Variance (psia)$^2$</td>
<td>4368</td>
<td>4229</td>
<td>4079</td>
<td>3783</td>
</tr>
</tbody>
</table>
The program determines the natural logarithm of the random permeability field with mean-zero distribution. Here, in order to solve the flow equation we take a value of 1.0 to represent the base transmissibility \((kh/\mu)\) rather than a base permeability. An exponential type covariance function was chosen with a variance of 0.55 and correlation length of 42.0 feet and keeping the seed for random numbers generation the same for all realizations. The input data of the varying block size and the corresponding output of the variances of the natural logarithm of permeability and pressure distributions are included in Table 4.3. Results show that the mean of the logarithm of permeability is very close to zero which corresponds to a geometric mean of 1.0. Results also show a trend in the reduction of the magnitude of the variance of \(\log_e\) transmissibility and that of the pressure distribution as the block size increases however the rate of decrease in the case of the pressure distribution is not pronounced as in the case of \(\log_e\) transmissibility. However, understandably a long correlation length would result in a larger reduction of the magnitude of variance.

In this approach of allowing the spacing \((\Delta x, \Delta y)\) to exceed the correlation length, there is merit to the view that columns 3 and 4 of Table 4.3 may represent artifact cases for which the Discrete Fourier Transform method is not strictly adhered to since in effect we are truncating the spectral components of high wave number.
Chapter 5: Conclusions and Recommendations

Because of the geometrical effect in pattern displacement, the relative locations of heterogeneities may affect the pattern sweep efficiency. The behavior of a heterogeneous but physically isotropic medium with miscible displacement at unit mobility ratio was investigated on an ensemble of subrealizations obtained by translating the boundaries of a five-spot quadrant within a larger segment of a heterogeneous formation. With each of these subrealizations, the history of a miscible displacement was computed. This investigation has led to the conclusion that the five-spot quadrant recovery characteristics in a medium with relatively short correlation length tends to approach those of a homogeneous medium. Any apparent variation was shown to be statistically insignificant. However, this conclusion can not be generalized for varying correlation lengths. Results of varying the correlation length on the variance of permeability field and that of the resultant pressure field showed a definite trend. Hence, the recommendation is made that the influence of the correlation length should be investigated.

An inherent assumption in this investigation using the pressure approach to solving the governing partial differential equation is that the no-flow boundaries will be straight lines. However, the boundaries for a medium with a relatively long correlation length will be significantly perturbed. In this case, it may be necessary to investigate a large part or all of the field rather than a single approximate quadrant. Alternatively, the stream function approach to solving the governing partial differential equation, that results in a Dirichlet boundary condition, will make it easier to deal with the perturbed no-flow boundary. The finite difference formulation of the governing partial differential equation results in the problem of singularity in the mathematical solution which will affect the accuracy of the solution near the sandface. Such a drawback is highly undesirable in a heterogeneous medium. Alternatively, a finite element
formulation on orthogonal curvilinear coordinates using non-uniform mesh spacing offers flexible
spatial resolution and may be used in conjunction with the Source Point method which generates
a random field as a mathematically continuous function with spatial covariance structure.

In practice, the criteria used for locating the wells of a pattern do not take into
consideration the relative locations of heterogeneities since these are unknown. The question that
this investigation addresses is, would the recovery history be significantly different had the wells
been located in slightly different places? This investigation has ascertained that the difference
is insignificant for the range of displacement in well locations considered. The principal
restriction to this conclusion is that it applies only for a correlation length that is a fairly small
fraction of pattern spacing. Further calculation with greater correlation lengths is strongly
recommended.

The major limitations of this work, that should be kept in mind in applying it, are:

1. Straight lines were presumed for the no-flow boundaries marking the region of recovery
   between a producer-injector pair. Although this is incorrect, it is plausible to assume the
discrepancy will not be felt within the economic life of the project.

2. The area around the wellbore is not well approximated by the numerical method that is
   used. Further work in clarifying this for the variable permeability case is also suggested.

3. Porosity has been assumed to be constant; it was expected that error introduced by this
   idealization would be of minor proportion compared to the variation in permeability.
Finite Difference Source List

APPENDIX A
Appendix A: Finite Difference Source Listing

This program is designed to:

(i) generate an ensemble of 2-D random permeability field based on the Source Method Algorithm (it calls the IMSL library to obtain pseudorandom numbers)

(ii) calculates either the pressure distribution or the stream function distribution for steady-state incompressible Miscible Displacement in a Five-Spot quadrant and using either the 5-point formula or the 9-point formula (Finite Difference)

(iii) calculates the velocity field and streamlines

(iv) tracks the front before and after breakthrough

To compile on the SUN computer use:
f77 -f68881 fivespot.f -limsl

Program variables & constants are defined in the respective Subroutines:

Program fivespot
Implicit real (a-z)
Integer size,nlx,nsx,nly,nsy,numit,ngx,ngy,maxn
Parameter(nlx=0,nsx=11,nly=0,nsy=11,size=100)

Dimension k(nlx:nsx,nly:nsy)
Dimension vx(nlx:nsx,nly:nsy),vy(nlx:nsx,nly:nsy)
Dimension p(nlx:nsx,nly:nsy),pi(nlx:nsx,nly:nsy)
Dimension a(nlx:nsx,nly:nsy),b(nlx:nsx,nly:nsy)
Dimension c(nlx:nsx,nly:nsy)
Dimension f(nlx:nsx,nly:nsy),g(nlx:nsx,nly:nsy)
Dimension xc(size),yc(size),allrand(size)
Dimension sl(nlx:nsx,nly:nsy),s(nlx:nsx,nly:nsy)

Common /block1/ngy,xf,yf,alpha,w,tol,ko,phi,viscosity,h,
Oproducer
Common /alk2/maxn,dt,rad,btr
Common /iter/maxit
Common /datnum/seed,numran

Open(unit=80,file='input25.dat')

Subroutine to read in input file containing data needed for solution of the problem call input

Subroutine to generate the random numbers using IMSL math library call rnum(xc,yc,s,allrand)

Subroutine to calculate the Permeability function values ("SL")
call subrealzn(xc,yc,s,nlx,nsx,nly,nsy,sl,sbar,sivar)
c*** Subroutine to calculate the Permeability values ("K")
call permibility(nlx,nsx,nly,nsy,sl,k,slbar,slvar)

Subroutine to calculate the pressure distribution field
   call pressure(numit,dx,dy,succes,rlx,nsx,nly,nsy,k,
                   p,pi,a,b,c,d,f,g,c91,c92,c93,c94)

Subroutine to calculate velocity components in X & Y-directions
   call velocity(dx,dy,rlx,nsx,nly,nsy,vx,vy,p,k)

Subroutine to calculate the front location
   call front(dx,dy,rlx,nsx,nly,nsy,vx,vy,k)
end

c*
subroutine input

******************************************************************************
c* Define variables & constants
******************************************************************************
c** ngx, ngy = size of the discretized grid in the X and Y direction

Subroutine input

******************************************************************************
real perm,dt
character*20 filern
common /block1/ngx,ngy,xf,yf,alpha,w,tol,ko,phi,viscosity,h,
   
Subroutine input

******************************************************************************
read(80,*)ngx,ngy
read(80,*)xf,yf
read(80,*)alpha,w,tol,maxit
read(80,*)numran
read(80,*)perm,phi,viscosity,h,Pinject,Qproducer
read(80,100)filern
read(80,*)maxn
read(80,*)dt
read(80,*)btr
read(80,*)seed
100 format(a)
return
end

** subroutine subrealizn

---

This subroutine uses the inverse square law relationship in the text to generate the permeability functions (the log permeability values)

*** sl() = log permeability values 
*** numran = Number of generated random numbers or the number of source points in this study 
*** nlx,nsx,nly,nsy = Array sizes for the two dimensional field 
*** slbar = mean of "sl" values 
*** slvar = variance of "sl" values 
*** ko = base transmissibility 
*** x,y = X-coordinate and the Y-coordinate of a grid point 

 subroutine subrealizn(xc,yc,s,nlx,nsx,nly,nsy,sl,slbar,slvar)

-----------------------------------------------------------------------------------------------------------------------------

 real sl(nlx,nsx,nly,nsy)
 common /datnum/seed,numran 
 common /block/ngx,ngy,xf,yf,alpha,w,tol,ko,phi,viscosity,h,
 1 Qproducer

 real xc(numran),yc(numran),s(numran)
 real xl,yl,ystep,xstep,k0

 open(unit=81,file='sourcepts')

* loops 30,114,119 are for generating the subrealizations and commented out in this documentation to allow generating a base realization only

*** rescale random numbers for the larger heterogeneous segment X & Y-coordinates
*   do 30 i = 1,numran
*      xc(i)=xc(i)*1.2-10.0
*      yc(i)=yc(i)*1.2-10.0
*   write(6,*i,xc(i),yc(i),s(i)
*30 continue

 Calculate the "sl" values

 xstep = 100./(ngx-1)
ystep = 100./(ngy-1)

*** Generate diff subrealizations or random fields & store for future use by calling subroutine store
***
***
 ccc do 119 k=-4,8,4
 c      delta=fl ot(k)
 c do 114 jk=1,5
 c      theta=22.5*(float(jk)-1.0)
 c      if(k.eq.0.and.jk.gt.1) then
 c      go to 114
 ccc endif

 xl = 0.0
do 20 n = 1, ngx
   yl = 0.0
   do 19 m = 1, ngy
      sum1 = 0.0
      sum2 = 0.0
      do 10 jj = 1, numran
         ccc
            delx = delta * cos(theta)
            dely = delta * sin(theta)
            den = (xc(jj) - xl)**2 + (yc(jj) - yl)**2
            den = (xc(jj) - delx - xl)**2 + (yc(jj) - dely - yl)**2
            sum1 = sum1 + s(jj)/den
            sum2 = sum2 + 1./den
          10
         continue
         sl(n, m) = sum1/sum2
         yl = yl + ystep
      19
      continue
      xl = xl + xstep
      20 continue
      c call store(sl, ngx, ngy, theta, delta)
      c continue
      ccc continue
do 98 i = nx, 1, -1
      write(72, *)'ROW', i
      write(81, 100)(sl(i, j), j = 1, nxy)
     100 format(10f7.2)
      continue
c call stats(sl, ngx, ngy, slbar, geomean, harmean, slvar, std,
      1 nlx, nsx, nly, nsy)
      return
end
*****************************************************************************
subroutine store(sl, nx, ny, theta, delta)
c****** run interactively to store results in 16 files
*****************************************************************************
character *20 inputfil
real sl(0:65, 0:65)
write(*, *)'IN STORE nx and ny = ', nx, ny
write(*, *)' INPUT filename'
read(5, 125) inputfil
125 format(a)
open(unit=72, file = inputfil)
open(unit=82, file = 'files1')
write(72, *)
write(72, *)'Log-permeability values'
write(72, *)
write(72, *)'theta=', theta, '  delta=', delta
write(72, *)
do 99 i = nx, 1, -1
   write(72, *)'ROW', i
   write(72, 100)(sl(i, j), j = 1, nxy)
99 continue
close(unit=72)
return
end
This routine calculates the permeability field using the previously generated log-k values.

**sl( )** = log-Permeability field

**k( )** = Permeability field

**alpha** = magnitude of heterogeneity

**ko** = Uniform permeability value

```fortran
subroutine permibility(nlx,nsx,nly,nsy,sl,k,slbar,slvar)
  common /block1/ngx,ngy,xf,yf,alpha,w,tol,ko,phi,viscosity,h,
  Qproducer
  common /datrnum/seed,numran
  character *20 output
  real ko,k(nlx:nsx,nly:nsy),sl(nlx:nsx,nly:nsy)
  open(unit=83,file='fileperm')
  Read in the "sl" values from stored files
  c*
  c*** loop 1 is used to read stored data of sl(i,j) of the different subrealizations and is skipped here
  goto 102
  write(*,*)'INPUT outputfile'
  read(*,126)output
  126 format(a)
  open(unit=73,file=output)
  do 1 i = ngx,1,-1
    read(73,100)(sl(i,j),j=1,ngy)
  100 format(10f7.2)
  continue
  close(unit=73)
  c Calculate the permeability values
  102 ko=1.
    do 3 n = ngx,1,-1
    do 2 m = 1,ngy
      k(n,m)=ko*exp(alpha*sl(n,m))
    2 continue
    3 continue
  c Print the permeability values into a file for later use
    write(83,*)
    write(83,*)' Permeability Field'
    write(83,*)' Permeability Field'
    write(83,*)
    do 98 i=ngx,1,-1
    write(83,'(10f7.2)') k(i,1:ngy)
    98 continue
    return
end
```
This routine calculates the pressure distribution field for the
quarter of the five-spot pattern under study. Finite difference
approximation is used, and the resulting equations are solved using
the SOR iterative method
and either of the arithmetic mean, harmonic or geometric
to define interblock transmissibilities
numit = maximum number of iterations for the SOR iterative method
dx , dy = grid block length in X and Y direction
xf , yf = length of the domain in X and Y direction
p( ) = pressure values
pi( ) = initial pressure values
a( ),b( ),c( ),d( ),f( ),g( ) coefficients of the pressure values
at each grid point
c91( ),c92( ),c93( ),c94( ) additional coefficients for the 9 point
formula

subroutine pressure(numit,dx,dy,succes,nlx,nsx,nly,nsy,k,
p,pi,a,b,c,d,f,g,c91,c92,c93,c94)
  integer r,numit,maxit
  real k(nlx:nsx,nly:nsy),ko,phi,viscosity,h
  real p(nlx:nsx,nly:nsy)
  real pi(nlx:nsx,nly:nsy)
  real a(nlx:nsx,nly:nsy),b(nlx:nsx,nly:nsy),c(nlx:nsx,nly:nsy)
  real f(nlx:nsx,nly:nsy),g(nlx:nsx,nly:nsy)
  real c91(nlx:nsx,nly:nsy),c92(nlx:nsx,nly:nsy)
  real c93(nlx:nsx,nly:nsy),c94(nlx:nsx,nly:nsy)
  logical succes
  common /datnumnum/seed,numran
  common /iter/maxit
  common /block1/ngx,ngy,xf,yf,alpha,w,tol,ko,phi,viscosity,h,
  Qproducer
  open(unit=84,file='filepress')
  distx = xf
  disty = yf
  dx = xf/(ngx-1)
  dy = yf/(ngy-1)

  c  Initialize the pressure values as the first guess for the iterative
  method
  pinc = 50.0 / (ngx-1)
  r=2*(ngx-1)
  do 13 i=1,ngx
   do 14 j=1,ngy
     p(i,j)=pinc*(r-(j-1))
     pi(i,j)=pinc*(r-(j-1))
  14 continue
  r=r-1
  13 continue

  do 542 i = 1,ngx
   do 543 j = 1,ngy

\[ p(i,j) = \pi(i,j) \]

543  continue

542  continue

\[ c*** \text{ specify type of formula & type of mean} \]
\[ c\]

\[ \text{harm}=1 \]
\[ \text{arith}=1 \]
\[ \text{ninep}=1 \]

\[ \text{if(ninep .eq. 1)print*, ' 9 point formula chosen '} \]

\[ \text{Right and left image boundaries (no flow)} \]

562  continue

\[ \text{do 562} \ i=0,\text{ngx+1} \]
\[ k(i,0)=k(i,2) \]
\[ k(i,\text{ngx+1})=k(i,\text{ngx-1}) \]

\[ \text{Bottom and top boundaries (no flow)} \]

563  continue

\[ \text{if(arith .eq. 1)then} \]
\[ \text{print*, ' arithmean is chosen '} \]

561  continue

\[ \text{do 601} \ i=1,\text{ngx} \]
\[ \text{do 601} \ j=1,\text{ngy} \]
\[ c(i,j) = -(\text{dy/dx})*0.5*(k(i,j)+k(i-1,j)) \]
\[ b(i,j) = -(\text{dy/dx})*0.5*(k(i,j)+k(i+1,j)) \]
\[ g(i,j) = -(\text{dx/dy})*0.5*(k(i,j)+k(i,j-1)) \]
\[ f(i,j) = -(\text{dx/dy})*0.5*(k(i,j)+k(i,j+1)) \]
\[ a(i,j) = -(g(i,j)+f(i,j)+c(i,j)+b(i,j)) \]

\[ \text{if(ninep .eq. 1)then} \]
\[ c91(i,j) = -(\text{dy/dx})*0.5*(k(i,j)+k(i+1,j)) \]
\[ c92(i,j) = -(\text{dy/dx})*0.5*(k(i,j)+k(i+1,j)) \]
\[ c93(i,j) = -(\text{dy/dx})*0.5*(k(i,j)+k(i-1,j)) \]
\[ c94(i,j) = -(\text{dy/dx})*0.5*(k(i,j)+k(i-1,j)) \]
\[ \text{end if} \]

601  continue

\[ \text{end if} \]

\[ \text{if(harm .eq. 1)then} \]
\[ \text{print*, ' harmean is chosen '} \]

600  continue

\[ \text{end if} \]
if(ninep .eq. 1) then
    c91(i,j) = -(dy/dx)*(2./((k(i,j)+1.)/k(i+1,j+1)))
    c92(i,j) = -(dy/dx)*(2./((k(i,j)+1.)/k(i+1,j-1)))
    c93(i,j) = -(dy/dx)*(2./((k(i,j)+1.)/k(i-1,j-1)))
    c94(i,j) = -(dy/dx)*(2./((k(i,j)+1.)/k(i-1,j+1)))
end if

701   continue
end if

if(geom .eq. 1) then
    print*, ' gemean is chosen '
    do 801 i=1,ngx
        do 801 j=1,ngy
            c(i,j) = -(dy/dx)*((k(i,j)*k(i-1,j))**.5)
            b(i,j) = -(dy/dx)*((k(i,j)*k(i+1,j))**.5)
            g(i,j) = -(dx/dy)*((k(i,j)*k(i,j-1))**.5)
            f(i,j) = -(dx/dy)*((k(i,j)*k(i,j+1))**.5)
            a(i,j) = -(g(i,j)+f(i,j)+c(i,j)+b(i,j))
        end do 801 j
    end do 801 i
end if

801   continue
end if

Start of the iteration process to solve the governing equation

succes = .false.
numit = 0
amax = 0.0
numit = numit + 1
IF(numit .gt. maxit)GO TO 45

101   continue
    sum = 0.0
    oldmax = 0.0
    botmax = 0.0
    kount = 1

Right and left image boundaries (no flow)

do 522 i=0,ngx+1
    p(i,0)=p(i,2)
    k(i,0)=k(i,2)
    ! left
    p(i,ngx+1)=p(i,ngx-1)
    k(i,ngx+1)=k(i,ngx-1)
    ! right
end do 522

522   continue

Bottom and top image boundaries (no flow)

do 530 j=0,ngy+1
    p(ngy+1,j)=p(ngy-1,j)
\[
k(n_{gy}+1,j) = k(n_{gy}-1,j) \\
p(0,j) = p(2,j) \\
k(0,j) = k(2,j)
\]

! top

! bottom

530 continue

c** set Boundary Condition
c*
p(1,1) = Pinject
c Main computation loop for the pressure values
do 60 i=1,ngx
do 61 j=1,ngy

c Set the value of the injection and production rate at appropriate grid
c points. Flow rate at all other nodal points is set to zero
c**
c*** the constant "1.127" converts the consistent set of units
c*** of the problem to field units
c**
d = 0.0
if((i.eq.nqx).and. (j.eq.ngy)) then
  d = -Qproducer/1.127
end if

if(ninep .eq. 1) then

c Pressure calculation using the nine point formula
if((i .ne. 1) .or. (j .ne. 1)) then

c Calculate pressure values iteratively
\[
den = -(4*(g(i,j)+c(i,j)+b(i,j)+f(i,j)) + c91(i,j)+c92(i,j)+
c93(i,j)+c94(i,j))
\]

\[
p(i,j) = (w/den)*(d-4*b(i,j)*p(i+1,j)-4*c(i,j)*p(i-1,j)-
4*f(i,j)*p(i,j+1)-4*g(i,j)*p(i,j-1)-c91(i,j)*p(i+1,j+1)-
c92(i,j)*p(i+1,j-1)-c93(i,j)*p(i-1,j+1)-
c94(i,j)*p(i-1,j-1))
\]

ccc p(i,j) = p(i,j)+(l-w)*pi(i,j)
p(i,j) = p(i,j)*ko*h/viscosity
end if

else

c Pressure calculation using the five point formula
if((i .ne. 1) .or. (j .ne. 1)) then
\[
p(i,j) = (w/a(i,j))*(d-c(i,j)*p(i-1,j)-b(i,j)*p(i+1,j)-
g(i,j)*p(i,j-1)-f(i,j)*p(i,j+1))
\]

ccc p(i,j) = p(i,j)+(l-w)*pi(i,j)
ccc p(i,j) = p(i,j)*ko*h/viscosity
else
  p(1,1) = Pinject
end if
end if

Calculate the maximum difference between a calculated value and its previous value. If this value is less than the tolerance the pressure calculation phase is halted

\[
\text{err} = \text{abs}(p(i,j) - p(i,j))
\]

if (err .gt. amax) amax = err

continue

continue

Print the current pressure distribution when the number of iterations has exceeded the maximum number of iterations

if (numit .gt. maxit) then
  write(84,*)
  write(84,*)' Pressure Field (exceeded iterations)'
  write(84,*)
  do 398 i = ngx, 1, -1
  ccccc
  do 398 i = 1, ngx
    write(84,*)' ROW ', i
    write(84,1000)(p(i,j), j = 1, ngy)
    write(*,*)' ROW ', i
    write(*,*)(p(i,j), j = 1, ngy)
  ccccc
  format(10f7.2)
end if

return

if (amax .lt. tol) then
  succes = .true.

Printing pressure values into a data file

write(84,*)
write(84,*)' Pressure Field'
write(84,*)

do 98 i = ngx, 1, -1
 cccc
 do 98 i = 1, ngx
    write(84,*)' ROW ', i
    write(84,*)(p(i,j), j = 1, ngy)
 cccc
 continue

write(84,*)
write(84,1)alfa, w, tol, ngx, ngy

format(5x, 'alfa = ', f5.3, ', 4x, ' w = ', f5.2, ', 4x,

$'
tolerance = ', e15.8, ', ngx = ', i5, ', ngy = ', i5//)

write(84,*)' ko = ', ko
write(84,79)qproducer, ngx, ngy

format(2x, 'for the Injection Flow rate = ', f9.2, ',

$'
for the Production Flow rate = ', f9.2, ',

$'
Prescribed Pressure is = ', f9.2, ',

$'
Pressure values for the grid size : ', i3, ' x ', i3/

write(84,68)numit
write(84,122)

format(1x, 70(' - '))

format(3x, 'number of iteration is = ', i10)
format (lx, 6f10.5)
format (lx, 2f6.2)
write (84, 23) ' seed = ', seed, ' Nsp = ', numran
format (/a, f16.1, a, i9)
write (84, *)
if (ninep .eq. 1) write (92, *) ' 9 point formula chosen '
else
   Replace the newly calculated values and repeat the iteration
   do 65 i = 1, ngx
      do 65 j = 1, ngy
         pi (i, j) = p (i, j)
      end do
   continue
   go to 35
   end if
   return
end

******************************************************************************

C C This subroutine will calculate the velocity field for the area
C C under the study, using the pressure values already generated by
C C "Pres" subroutine
C C *** vx( , ) = velocity for each grid point in the X-direction
C C *** vy( , ) = velocity for each grid point in the Y-direction
C C *** p( , ) = Pressure values at each grid point
C C *** k( , ) = Permeability values at each grid point

   subroutine velocity(dx, dy, nlx, nlx, nly, nly, vx, vy, p, k)
   real p(nlx:nlx, nly:nly), k(nlx:nlx, nly:nly), ko
   real vx(nlx:nlx, nly:nly), vy(nlx:nlx, nly:nly)
   common /datnum/ seed, numran
   common /iter/maxit
   common /block1/ ngx, ngy, xf, yf, alpha, w, tol, ko, phi, viscosity, h,
                Qproducer
   open (unit=85, file='filevx')
   open (unit=86, file='filevy')

   C Read in the permeability values
   dx = xf / (ngx-1)
   dy = yf / (ngy-1)

   C Set up the image bounderries for pressure and permeability for the
   C no flow boundaries
   do 72 i = 0, ngx+1
      p(i, 0) = p(i, 2)
      p(i, ngx+1) = p(i, ngx-1)
      k(i, 0) = k(i, 2)
      k(i, ngx+1) = k(i, ngx-1)
   72 continue

******************************************************************************
72 continue
do 73 j=0,ngy+1
   p(0,j)=p(2,j)
p(ngy+1,j)=p(ngy-1,j)
k(0,j)=k(2,j)
k(ngy+1,j)=k(ngy-1,j)
73 continue
c Horizontal displacement velocity calculation for the interior grid points.
do 471 i = 1, ngx
do 471 j = 1, ngy
   vx(i,j) = -2.*((k(i,j+1)+k(i,j-1))/(dx*(k(i,j+1)+k(i,j-1))))
   vx(i,j) = -(ko/h)*(p(i,j+1)-p(i,j-1))/(2.0*dx)/(viscosity*phi)
   vx(i,j) = (k(i,j-1)+k(i,j))*p(i,j+1)-(k(i,j+1)+k(i,j))*p(i,j-1)
   vx(i,j)=vx(i,j)/(4.0*dx)
471 continue
c Vertical displacement velocity calculation for the interior grid points.
do 981 i = 1, ngx
do 981 j = 1, ngy
   vy(i,j) = -2.*((k(i+1,j)+k(i-1,j))/(dy*(k(i+1,j)+k(i-1,j))))
   vy(i,j) = -(ko/h)*(p(i+1,j)-p(i-1,j))/(2.0*dy)/(viscosity*phi)
   vy(i,j) = (k(i-1,j)+k(i,j))*p(i+1,j)-(k(i+1,j)+k(i,j))*p(i-1,j)
   vy(i,j)=vy(i,j)/(4.0*dy)
981 continue
c Printing velocity in x and y direction.
write(85,' Velocity Field X-dir')
do 297 i = npx, 1, -1
   write(85,' ROW ',j)
   write(85,100)(vx(i,j),j=1,ngy)
297 continue
write(86,' Velocity Field Y-dir')
do 542 i = ngy, 1, -1
   write(86,' ROW ','j
   write(86,100)(vy(i,j),j=1,ngy)
542 continue
100 format(10f7.2)

return
deend

**********************************************************************
c c This routine uses the velocity field and tracks the position of the
c c particles from the injection to the production well. The tracking
c c procedure starts by placing a set of points around the injection well
c c and track their movement from the injector to the producer. The points
c c are moved due to the pressure gradients in the field. A streamline is
c c the path of the movement of the introduced point from the injection to
c c the production well. Connecting the points at a particular time along
each streamline give us the location of the flood front at that time.

*** xp( ) = moving point coordinate in the X-direction
*** yp( ) = moving point coordinate in the Y-direction
*** vx,xp, vy,yp = velocity components of moving point
*** distip = distance between the injector and the producer
*** dx, dy = grid block length in X and Y direction
*** xf, yf = length of the domain in X and Y direction
*** ni, nj = column number and row number of grid in which point is
*** located
*** ftime = total time for a point to move from the injector to the
*** producer
*** maxn = maximum number of moving points
*** allwrite = variable allowing for printing of the particle positions
*** *** moving from the injection to the production well
*** *** nfinal = counter for the number of streamlines which extend from the
*** *** injection to the production well
*** *** radius = radius used to position the starting points about the
*** *** injection well
*** *** ist1 = counter for the number of streamlines
*** *** nall = counter for the total number of moving points along all
*** *** streamlines
*** *** xall( ) = array containing the X coordinate of all the moving points
*** *** yall( ) = array containing the Y coordinate of all the moving points
*** *** ptall( ) = array containing the time associated with a particular point
*** *** degarray( ) = array containing the angles referring to a streamline
*** *** finaltime( ) = array containing the time that takes for a particle to
*** *** reach the production well for all the streamlines
*** *** finalx( ) = array containing the X coordinate of a particle that
*** *** has reached the production well for all the streamlines
*** *** finaly( ) = array containing the Y coordinate of a particle that
*** *** has reached the production well for all the streamlines
*** *** degree( ) = array containing a list of the streamlines
*** *** deg = an angle which gives the starting position of a particle, hence,
*** *** determining a particular streamline
*** *** dt = time increment used to move the points to their next position
*** *** btr = a factor used to define a range on the time at which breakthrough
*** *** along the fastest streamline has occurred

subroutine front(dx,dy,nlx,nsx,nly,nsy,vx,vy,k)
real vx(nlx:nsx,nly:nsy),vy(nlx:nsx,nly:nsy)
real finaltime(100),finalx(100),finaly(100),degree(100)
real dt,ftime,dt,vx,vx,p
real kxy,kx,k(nlx:nsx,nly:nsy)
real xp(70002),yp(70002)
common /oblk/xall(149900),yall(149900),ptall(149900)
integer kount

common /block1/ngx,ngy,xf,yf,alpha,w,tol,ko,phi,viscosity,h,
Qproducer
common /alk2/maxn,dt,rad,btr
common /datnum/seed,numran
open(unit=87, file='filestr')
Image boundary on velocity values for the no flow boundaries

```fortran
  do 612 i = 1, ngx
      vx(i,0) = - vx(i,2) ! left
      vx(i,ngx+1) = - vx(i,ngx-1) ! right
  continue

  do 613 j = 1, ngy
      vy(ngy+1,j) = - vy(ngy-1,j) ! top
      vy(0,j) = - vy(2,j) ! bottom
  continue
```

distip = sqrt(xf*xf + yf*yf)
p1 = acos(-1.0)
radius = 1.0
istl = 1

```fortran
  write(87,142) ngx, ngy
  format(3x,' grid size = ',i3,' x ',i3/)
  write(87,143) alpha, w, tol, ko
  format(3x,' alpha = ',f8.2, ' w = ',f8.2, ' tol = ',f15.5 '/../,
      * 3x,' ko = ',f8.2)
  write(87,79) producer
  format(2x,’ for the Injection Flow rate = ',f9.2,'$/
  $ for the Production Flow rate = ',f9.2,'$/
  $ prescribed pressure = ',f9.2)
  write(87,*)' Nsp = ', numran, ' seed = ', seed
  write(87,*)' dt = ', dt
  continue
```

```fortran
c ndtv = 1
c nall = 1
nfinal = 0
dx = xf / (ngx-1)
print*, 'dx=', dx
dy = yf / (ngy-1)

Read in an angle which basically gives the location of the starting
point around the injection well
deg = 0.0
kountpt = 0
do 1119 index = 1,19
  if (deg.gt.4..and.deg.lt.84.) then
    deg = deg + 4.0
  else
    deg = deg + 4.0
  endif
  print*, ' index=', index, nall, ' degree=', deg
1119 isumold = (nall/10)*10
```
\[\begin{align*}
\text{xp}(1) &= \text{radius} \times \sin(\text{deg} \times \pi/180.0) \\
\text{yp}(1) &= \text{radius} \times \cos(\text{deg} \times \pi/180.0) \\
\text{kxy} &= k(1,1) \\
\text{frtime} &= 0.0 \\
\text{xall}(\text{nall}) &= \text{xp}(1) \\
\text{yall}(\text{nall}) &= \text{yp}(1) \\
\text{ptall}(\text{nall}) &= \text{frtime}
\end{align*}\]

c Main loop to track the position of the particles from the injection well to the production well along a streamline.

c*** A bilinear velocity interpolation scheme is used

c*** open file to store number of points in the individual streamlines

\[
\begin{align*}
\text{open(unit=91, file='june.91')} \\
\text{w1} &= 0.0 \\
\text{w2} &= 0.0 \\
\text{w3} &= 0.0 \\
\text{w4} &= 0.0
\end{align*}\]

do 10 i = 1, \text{maxn}

\[
\begin{align*}
\text{n1} &= (\text{yp}(1)/\text{dy}) + 1 \\
\text{n2} &= (\text{xp}(1)/\text{dx}) + 1 \\
\text{remainx} &= \text{mod}(\text{xp}(1), \text{dx}) \\
\text{remainx} &= \text{remainx} / \text{dx} \\
\text{remainy} &= \text{mod}(\text{yp}(1), \text{dy}) \\
\text{remainy} &= \text{remainy} / \text{dy} \\
\text{w1} &= (1.0 - \text{remainx}) \times (1.0 - \text{remainy}) \\
\text{w2} &= \text{remainx} \times (1.0 - \text{remainy}) \\
\text{w3} &= \text{remainx} \times \text{remainy} \\
\text{w4} &= (1.0 - \text{remainx}) \times \text{remainy}
\end{align*}\]

c Check for the point falling within a small radius around the production well. If a point falls within such a distance, it is assumed that the point has reached the production well, and tracking for the next streamline begins.

\[
\begin{align*}
\text{if}((\text{xp}(1), \text{ge.xf} = 0.5 \text{ .and. } \text{xp}(1), \text{le.xf} = 0.5) \\
\text{and.} ((\text{yp}(1), \text{ge.yf} = 0.5 \text{ .and. } \text{yp}(1), \text{le.yf} = 0.5) \text{or.} \\
\text{xp}(1), \text{gt.xf} = 0.5 \text{ .or. } \text{xp}(1), \text{gt.yf} = 0.5)) \text{then}
\end{align*}\]

\[
\begin{align*}
\text{ntfinal} &= \text{ntfinal} + 1 \\
\text{finaltime}(\text{istl}) &= \text{ptall}(\text{nall}) \\
\text{finalx}(\text{istl}) &= \text{xall}(\text{nall}) \\
\text{finaly}(\text{istl}) &= \text{yall}(\text{nall}) \\
\text{degree}(\text{istl}) &= \text{deg} \\
\text{xall}(\text{nall}) &= \text{xf} \\
\text{yall}(\text{nall}) &= \text{yf} \\
\text{ptall}(\text{nall}) &= \text{finaltime}(\text{istl})
\end{align*}\]

c Advance to the next streamline

\[
\text{go to 22}
\]
e else
The point is located within the field, and it is moved due to the velocity field. Velocity components of the moving points in X and Y directions are calculated:

\[
\begin{align*}
\text{vx}_p &= w_1 \text{vx}(n1, n12) + w_2 \text{vx}(n1, n12+1) + w_3 \text{vx}(n1+1, n12+1) + w_4 \text{vx}(n1+1, n12) \\
\text{vy}_p &= w_1 \text{vy}(n1, n12) + w_2 \text{vy}(n1, n12+1) + w_3 \text{vy}(n1+1, n12+1) + w_4 \text{vy}(n1+1, n12) \\
\text{nall} &= \text{nall} + 1
\end{align*}
\]

Print the position of the moving points:

```fortran
if((nall/10)*10 .eq. nall) then
  kountpt = kountpt + 1
  write(87, 313) xall(nall-1), yall(nall-1), ptall(nall-1),
              vx, vy, nall, deg
  write(87, 313) xall(nall-1), yall(nall-1), deg
end if
```

Move the point to its new location:

```fortran
xp(i+1) = xp(i) + vx * dt
yp(i+1) = yp(i) + vy * dt
ftime = ftime + dt
xall(nall) = xp(i+1)
yall(nall) = yp(i+1)
ptall(nall) = ftime
```

```
if(xall(nall) .gt. xf. and. yall(nall) .gt. yf. and. ptall(nall) .gt. btr) then
  print*, ' degree ?', deg, xall(nall), yall(nall), ptall(nall)
endif
continue
```

Advance to tracking of the points along the next streamline:

This routine finds the time at which the shortest streamline breaks through:

```fortran
call minmax(finaltime, nfinal, timemax, timemin, imax, imin)
write(87, *) finalx(imin), finaly(imin), finaltime(imin),
        degree(imin)
```

This routine searches for the data sharing the same time level, thus producing the front locations at different time levels.
call search(btr, timemin, nall)
return
end

This routine finds the location of the flood front by choosing the points sharing a common time level

**xall( )** = array containing the X coordinate of all the moving points

**yall( )** = array containing the Y coordinate of all the moving points

**ptall( )** = array containing the time associated with a particular point

**degsarray( )** = array containing the angles referring to a streamline

**ntlevel** = number of time levels for which the front is to be located

**upper, lower** = parameters used to define a range for the times at which the front is to be located

**kmin** = counter defining the number of points found at the fastest time level

**counter( )** = array containing the number of points found at a particular time level

**timemin** = the fastest time at which fluid particles along a streamline has reached the production well

subroutine search(btr, timemin, nall)
common /oblk/xall(149900), yall(149900), ptall(149900)
real upper, lower
integer counter(20)
open(unit=88, file='filefront')
print*,'
NALL-1=' , nall-1
print*,' enter the number of time levels for the front ' 'construction'
read*, ntlevel
ntlevel=11
write(88,*),' Front positions at selected time levels are :
write(88,*)
if(timemin .ne. 0.0)then

Print the coordinates of the points lying on the flood front that first cusps into the production well

kmin = 0
print*,' nall=', nall
do 3 i = 1, nall-1
if(ptall(i).ge.timemin-btr.and.ptall(i).le.timemin+btr)then
write(88,8)xall(i), yall(i), ptall(i), i
end if
3 continue

kmin = kmin + 1
write(88,*),' Total = ', kmin

Print the coordinates of the points lying on the same flood front. This can be done for as many different time levels desired (up to 20, here)
upper1 = .1
lowerl = -.1
do 2 k = 1, ntelevel

counter(k) = 1

if(k.le.3) then
  upper = upper1 + 125.0*float(k)
  lower = lower1 + 125.0*float(k)
else
  upper = 250.05 + 250.0*(k-3)
  lower = 249.95 + 250.0*(k-3)
endif

  do 1 l = 1, nall - 1
     if(ptall(l).ge. lower .and. ptall(l) .le. upper) then
       write(88,8)xall(l), yall(l), ptall(l), l
       counter(k) = counter(k) + 1
     end if
   1 continue
write(88,*) ' Total = ', counter(k) - 1
2 continue
end if
end if
end if

format(3f12.2, i10)
return
end

The following routines calculate some statistical properties, and obtain
minimum and maximum of a given list. They are self explanatory.

*****************************************************************************

This routine calculates some statistical properties. Two dimensional
array.

subroutine stats(a, ngx, ngy, amean, gmean, hmean, variance, sdev,
   nlx, nsx, nly, nsy)

real a(nlx:nsx, nly:nsy)
sumgeo = 1.0
sumsl = 0.0
do 334 i = 1, ngx
   do 334 j = 1, ngy
      sumsl = sumsl + a(i, j)
   end if
334 continue

sumharm = sumharm + 1.0/a(i, j)
end if

sumgeo = sumgeo * a(i, j)

amean = sumsl / (ngx*(ngy))
sumsq = 0.0
do 124 i = 1, ngx
   do 124 j = 1, ngy
      sumsq = sumsq + (a(i, j) - amean)**2
    124 continue
if (sumgeo .lt. 0.0) then
    write(6,'(A10,1X,F10.5,1X,F10.5)') 'geometric mean not possible'
else
    gmean = sumgeo **(1./float(ngx*ngy))
end if

hmean = float(ngx*ngy)/sumharm

variance = sumsq/(float(ngx*ngy))
sdev = sqrt(variance)
return
end

******************************************************************************
c
c Subroutine to find the maximum and minimum value in a one
c dimensional array.

c subroutine minmax(a,n,bmax,bmin,imax,imin)
  real a(n)
bmax=1.e-6
bmin=1.e+6
imax = 1
imin = 1
do 2 i=1,n
  if(a(i). ne. 0) then
    if(a(i).gt.bmax) then
      bmax=a(i)
imax = i
    end if
    if(a(i).lt.bmin) then
      bmin=a(i)
imin = i
    end if
  end if
2 continue
return
end

******************************************************************************
c The following subroutine generates two sets of random numbers
c from a uniform distribution, and a set of numbers from a normal
c distribution, using IMSL library routines.

c *** xc( ) = Random numbers representing the X coordinate of a point (uniform)
c *** yc( ) = Random numbers representing the Y coordinate of a point (uniform)
c *** s( ) = Random numbers representing the Z coordinate of a point (normal)
c *** seed = Seed for the random number generator
c *** numran = Number of generated random numbers or the number of source
c *** points as used in this study
c *** allrand( ) = Array holding generated random numbers from the uniform
c *** distribution
c *** ranset, rnum, rnoa = Standard IMSL routines for generating random numbers

c subroutine rnum(xc,yc,s,allrand)
  implicit real (a-z)
dimension xc(numran), yc(numran), s(numran)
dimension allrand(2*numran)
integer i,j,numran,ntotal,iseed
common /datrnum/seed,numran
external rnest,rnum,rrnoa
iseed = int(seed)
ntotal = 2*numran

c IMSL routines to generate a set of uniformly distributed random numbers
    call rnest(iseed)
    call rnum(ntotal,allrand)

c Scale the generated values to range from 0 to 100 (IMSL)
    call sscl(ntotal,100.,allrand,1)

c Generate random numbers from a standard normal distribution (IMSL)
    call rrnoa(numran,s)
    j = 1
    do 204 i = 1,numran
        xc(j) = allrand(i)
        j = j + 1
    204          continue
    j = 1
    do 205 i = numran+1,ntotal
        yc(j) = allrand(i)
        j = j + 1
    205          continue

c Print the values of x, y, and s into a file, for the determination
of the permeability function (sl) values
    do 34 i=1,numran
        write(8,*)xc(i),yc(i),s(i)
    34          continue
    return
end
Appendix B

Finite Element Source Listing
This Program is designed to solve the pressure field of the
2-D governing elliptic partial differential equation
in a five-spot Quadrant using the finite element method:

(i) linear triangular and rectangular elements
(ii) isoparametric quadrilateral element

**Appendix B : Finite Element Source Listing **

<table>
<thead>
<tr>
<th>Variable/Constant</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>gstif(,)</td>
<td>element of Global stiffness [M]</td>
</tr>
<tr>
<td>estif(,)</td>
<td>element of Element stiffness [M]</td>
</tr>
<tr>
<td>gf( )</td>
<td>holds elements of Load vector &amp; Solution vector</td>
</tr>
<tr>
<td>ibdf( )</td>
<td>holds index of prescribed pressure (the Unknown)</td>
</tr>
<tr>
<td>vbdf( )</td>
<td>holds value of prescribed pressure (the Unknown)</td>
</tr>
<tr>
<td>nod(, )</td>
<td>element of connectivity [M]</td>
</tr>
<tr>
<td>inod(, )</td>
<td>element node number</td>
</tr>
<tr>
<td>x( ),y( )</td>
<td>coordinates of nodes</td>
</tr>
<tr>
<td>dx( ),dy( )</td>
<td>intervals in X &amp; Y-directions</td>
</tr>
<tr>
<td>cl</td>
<td>material property</td>
</tr>
<tr>
<td>c5</td>
<td>source strength</td>
</tr>
<tr>
<td>iel</td>
<td>element type</td>
</tr>
<tr>
<td>npe</td>
<td>number of node in element</td>
</tr>
<tr>
<td>imesh</td>
<td>type of mesh</td>
</tr>
</tbody>
</table>

implicit real*8(a-h,o-z)
parameter(nsdfl=26)
dimension gstif(1600,100), gf(1600), inod(20,2),  
1    ibdf(nsdfl), vbdf(nsdfl)
common/blok/ iel,npe,nx,ny,c1,c5,ibdf,vbdf
common/msh/ nod(3200,9), x(1600),y(1600), dx(40), dy(40)
common/stf/ elstif(18,18),elxy(9,2),f(18)
data nmax, nmax/1600,100/
open(unit=14,file='input26.dat'
open(unit=19,file='press26.dat')
call input

Processing Starts Here

nxl=iel*nx+1
nyl=iel*ny+1
if(iel.eq.0) nxl=nx+1
if(iel.eq.0) nyl=ny+1
call mesh(iel, nx, ny, npe, nmm, nem)

ndf=1
neq=nnm*ndf
nn=npe*ndf
do 75  i = 1, nem
write(*,440) i, (nod(i,j), j= 1, npe)
write(*,460)nsdf, (ibdf(i), i= 1, nsdf)

c*--------------------------------------------------------
c* compute the half band width
 c*--------------------------------------------------------
100 nhbw = 0
do 110 n = 1, nem
do 110 i = 1, npe
do 110 j = 1, npe
nw = (iabs(nod(n,i) - nod(n,j)) +1) * ndf
110 if(nhbw.lt.nw) nhbw=nw
write(*,590) nhbw

c*--------------------------------------------------------
c* initialize the global stiffness matrix ad force vector
 c*--------------------------------------------------------
130 do 130 i = 1, neq
gf(i) = 0.0
130 do 130 j = 1, nhbw
gstif(i,j) = 0.0

c*--------------------------------------------------------
c* assembly of the element matrices
 c*--------------------------------------------------------
150 do 300 n = 1, nem
kou=kou+1
do 150 i = 1, npe
ni = nod(n,i)
elxy(i,1) = x(ni)
elxy(i,2) = y(ni)
150 continue
if(iel.gt.0) call stiffq(n,npe,nn,iel,cl,cl,c5)
*if(iel.eq.0) call stifft(n,npe,nn,iel,cl,cl,c5)
do 160 i = 1, nn
160 continue
write(*,670) (elstif(i,j), j= 1, nn)
c*--------------------------------------------------------
c* assemble Element matrices to obtain Global matrix
 c*--------------------------------------------------------
200 do 280 i = 1, npe
nr = (nod(n,i)-1)*ndf
do 280 ii = 1, ndf
nr = nr + 1
l = (i-1)*ndf + ii
gf(nr) = gf(nr) + f(l)
do 260 j = 1, npe
nc=(nod(n,j)-1)*ndf
do 260 jj = 1, ndf
m = (j-1)*ndf + jj
nc = nc + jj + 1 - nr
if(nc) 260, 260, 250
250 g stif(nr,nc) = g stif(nr,nc) + elstif(l,m)
260 continue
280 continue
300 continue
print*, 'GLOBAL STIFFNESS'
write(*,999)((gstif(i,j),j=1,nhbw),i=1,neq)
format(6f7.2)

call bdry to adjust Global [M] for the boundary conditions
ires = 0
do 330 i = 1, neq
write(*,480) (gstif(i,j), j = 1, nhbw)
print*, 'GLOBAL STIFFNESS (half band)'
write(*,480) (gf(i), i = 1, neq)
if(ndf.eq.0) goto 350
call bdry(nmax,ncmax, neq, nhbw, gstif, gf, nsdf, ibdf, vbdf)

call subroutine 'solve' to solve the system of equations
(The solution is returned in array gf.)
call solver(nmax, ncmax, neq, nhbw, gstif, gf, ires)
write(19,481)(gf(i),i=1,nmm)
STOP

*** formats
format(10x,20i5)
format(10x,'ACT. NUMBER OF ELEMENTS IN THE MESH...=',i3,/,
* 10x,'NUMBER OF NODES IN THE MESH....................=',i3,/,
* 10x,'TOTAL NUMBR OF EQUATIONS IN THE MODEL...=',i3,/
format(/,5x,'NO. OF SPECIFIED DEGREES OF FREEDOM.....=',i3,/,
* 5x,'ARRAY OF THE SPECIFIED DEGREES OF FREEDOM=',10i5,/
* 10x,20i5,,/)
format(/,10x,'CONVEXTIVE BOUNDARY DATA: NBE...=',i3,/,10x,
* 'ARRAY IBN=',10i5)
format(13f6.1)
format(13f6.1)
format(2(16,2e15.5,10x))
format(/,5x,'BOOLEAN (CONNECTIVITY) MATRIX NOD(I,J) ',/
format(10x,'PARAMETERS, C1, C2, C3, C4, AND C5: ',/
* 15x,'C1 =',e10.3,/,15x,'C2 =',e10.3,/,15x,
* 'C3 =',e10.3,/,15x,'C4 =',e10.3,/,15x,'C5 =',e10.3)
format(16i5)
format(/,3x,'SOLUTION VECTOR: ',/
format(3x,'COORDINATES OF THE GLOBAL NODES: ',/
format(8f10.4)
format(5x,'HALF BAND WIDTH OF GLOBAL STIFFNESS MATRIX =',i3)
format(/,5x,'VALUES OF THE SPECIFIED FORCES: ',/
format(5e15.5)
format(3x,'ELEMENT MATRICES: ',/
format(3x,'GLOBAL STIFFNESS MATRIX: ',/)

END

*** SUBROUTINE input
implicit real*8(a-h,o-z)
dimension ibdf(26), vbdf(26)
common/blokl/ iel,npe,nx,ny,c1,c5,ibdf,vbdf
write(*,*), iel,npe,imesh
read(14,*) iel,npe,imesh
write(*,*), nx,ny
read(14,*) c1,c5
write(*,*), cl,c5
read(14,*) (ibdf(i),i=1,26)
write(*,*), (ibdf(i),i=1,26)
read(14,*) P
10 do 1=1,26
    vbdf(i)=P
    do 20 i=1,26
20    write(*,*) vbdf(i)
end

******************************************************************************
SUBROUTINE stiffe(n,npe,nn,iel,ak1,ak2,q)
******************************************************************************

computes the element coefficient matrices for triangular element

******************************************************************************
implicit real*8(a-h,o-z)
common/stf,elstif(18,18),elxy(9,2),f(18)
dimension b(3,6), bt(6,3), str(3,6), alpha(3), beta(3), gama(3)
1   ,x(3), y(3)

******************************************************************************
define the coefficients of the interpolation functions
******************************************************************************
do 10 i = 1, npe
    x(i) = elxy(i,1)
y(i) = elxy(i,2)
do 10 j = 1, nn
10   b(i,j) = 0.0

   do 20 i = 1, npe
       j = i+1
       if (j.gt.npe) j = j - npe
       k = j+1
       if (k.gt.npe) k = k - npe
       alpha(i) = x(j)*y(k) - x(k)*y(j)
y(i) = y(j) - y(k)
       beta(i) = x(k) - x(j)
20   gama(i) = x(1)*(y(2)-y(3)) +x(2)*(y(3)-y(1)) +x(3)*(y(1)-y(2))

******************************************************************************
compute the coefficient matrix
******************************************************************************
100 xbar = (x(1) + x(2) + x(3))/3.0
        ybar = (y(1) + y(2) + y(3))/3.0
        a00 = 0.5*det
        a01 = a00*ybar
        a10 = a00*xbar
all
  =a00*(x(1)*y(1)+x(2)*y(2)+x(3)*y(3)+9.0*xbar*ybar)/12.0
a20
  =a00*(x(1)*x(1)+x(2)*x(2)+x(3)*x(3)+9.0*xbar*xbar)/12.0
a02
  =a00*(y(1)*y(1)+y(2)*y(2)+y(3)*y(3)+9.0*ybar*ybar)/12.0

C* only constant source term is assumed
C*
do 120  i = 1, npe
if(i.eq.1.and.n.lt.3) then
  f(i)=.5*q*(alpha(i)+beta(i)*xbar+gama(i)*ybar)
else
  f(i)=0.0
endif
  do 120  j = 1, npe
  elsestif(i,j) = (ak1*beta(i)+beta(j)+ak2*gama(i)*gama(j))/det/2.0
  continue
120  RETURN
END

C***********************************************************************
C***********************************************************************
SUBROUTINE stifq(n,npe, nn, iel, ak1, ak2, q)

C*** stiffness matrix for isoparametric quadrilateral elements
implicit real*8(a-h,o-z)
common/stf/ elstif(18,18), elxy(9,2), f(18)
dimension  sf(9), gdsf(2,9), gauss(4,4), wt(4,4),
  ss(18,18), s(9,9), sx(9,9), sy(9,9)
C***********************************************************************
data gauss  /4*0.0d0, -.57735027d0, .57735027d0, 2*0.0d0,
  1   -.77459667d0, 0.0d0, .77459667d0, 0.0d0,
  2   -.86113631d0, -.3399014d0, .3399014d0, .86113631d0/
C***********************************************************************
data wt /2.0d0, 3*0.0d0, 2*1.0d0, 2*0.0d0, .55555555d0,
  1   .88888888d0, .55555555d0, 0.0d0, .34785485d0,
  2   2* .65214515d0, .34785485d0/
C***********************************************************************
ndf = nn/npe
ngp = iel + 1

C***********************************************************************
C* initialize the arrays
C***********************************************************************
do 20  i = 1, npe
  do 20  j = 1, npe
    s(i,j) = 0.0
    sx(i,j) = 0.0
    sy(i,j) = 0.0
    sxy(i,j) = 0.0
  20  do 30  i = 1, nn
    f(i) = 0.0
  do 30  j = 1, nn
    ss(i,j) = 0.0
  30
do-loops on numerical (Gauss) quadrature begin here

  do 100 ni = 1, np
  do 100 nj = 1, np
    xi = gauss(ni,np)
    eta = gauss(nj,np)
    call shape(npe, xi, eta, sf, gdsf, det, elxy)
    const = det*wt(ni,np)*wt(nj,np)
    do 80 i = 1, np
      if(i.eq.(npe-1).and.n.gt.625) then
    c* evaluate source term
    c* f(i) = f(i) + q*sf(i)*const
      else
        f(i)=0.0
      endif
      cccc>> sf(i)=0.0
      60 do 80 j = 1, np
        s(i,j) = s(i,j) + const*sf(i)*sf(j)
        sx(i,j) = sx(i,j) + const*gdsf(1,i)*gdsf(1,j)
        sy(i,j) = sy(i,j) + const*gdsf(2,i)*gdsf(2,j)
        sxy(i,j) = sxy(i,j) + const*gdsf(1,i)*gdsf(2,j)
      80 continue
      100 continue

  c* element calculations
  c*
  do 110 i = 1, np
    do 110 j = 1, np
      endif
      return
    j=1
  110 continue

subroutine shape(npe, xi, eta, sf, gdsf, det, elxy)

  The subroutine evaluates the interpolatin functions (sf(i)) and
  its derivatives with respect to natural coordinates (dscf(i,j)),
  and the derivatives of sf(i) with respect to global coordinates
  for four node rectangular linear & isoparametric elements.
  c*

  implicit real*8(a-h,o-z)
  dimension elxy(9,2), xnode(9,2), np(9), gdsf(2,9),
  1 gj(2,2), gjinv(2,2), sf(9), gdsf(2,9)
  data xnode,/-1.0d0, 2*1.0d0, -1.0d0, 0.0d0, 1.0d0, 0.0d0, -1.0d0,
  1 0.0d0, 2*1.0d0, 2*1.0d0, -1.0d0, 0.0d0, 1.0d0, 0.0d0, 1.0d0, 2*0.0d0/
  data np, /1,2,3,4,5,6,7,8,9/
  c*
  fnc(a,b) = a*b

  linear interpolation functions (for four-node element)
c*-----------------------------------------------------
60      do 70  i = 1, npe
      xp = xnode(i,1)
      yp = xnode(i,2)
      xi0 = 1.0 + xi*xp
      eta0 = 1.0 + eta*yp
      sf(i) = 0.25*fnc(xi0,eta0)
      dsf(1,i) = 0.25*fnc(xp,eta0)
      dsf(2,i) = 0.25*fnc(yp,xi0)
    70      do 140 i = 1, 2
      do 140  j = 1, 2
      gj(i,j) = 0.0
    140      do 150 k = 1, npe
      gj(i,j) = gj(i,j) + dsf(i,k)*elxy(k,j)
      det = gj(1,1)*gj(2,2) - gj(1,2)*gj(2,1)
      gjinv(1,1) = gj(2,2)/det
      gjinv(2,2) = gj(1,1)/det
      gjinv(1,2) = gj(1,2)/det
      gjinv(2,1) = gj(2,1)/det
    150      do 150 i = 1, 2
      do 150  j = 1, npe
      gdsf(i,j) = 0.0
    150      do 150 k = 1, 2
      gdsf(i,j) = gdsf(i,j) + gjinv(i,k)*dsf(k,j)
    RETURN
END

c*****************************************************************
subsection mesh(iel, nx, ny, npe, nmm, nem)
c*****************************************************************

c*** The subroutine generates array nod(i,j), coordinates x(i), y(i),
c*** and mesh information (nmm,nem,npe) for rectangular domains. The
** domain is divided into linear triangular elements or quadrilateral
** element (nx by ny nonuniform mesh in general).
c*****************************************************************
implicit real*8(a-h,o-z)
common/msh/ nod(3200,9), x(1600), y(1600), dx(40), dy(40)
if(iel.gt.0) goto 100

mesh of triangular elements

nem = 2(nx*ny)
nxl = nx + 1
nyl = ny + 1
nxnl = 2*nx
nyyl = 2*ny
nmm = nxl*nyl
nod(1,1) = 1
nod(1,2) = 2
nod(1,3) = nxl + 1
nod(2,1) = 1
nod(2,2) = nxl + 2
nod(2,3) = nxl + 1
k = 3

DO 60 iy = 1, ny
   l = iy*nxx1
   m = (iy-1)*nxx1
IF(nx.eq.1) GOTO 40

DO 30 n = k,1,2
DO 20 i = 1, npe
   nod(n,i) = nod(n-1,i) + 1
   nod(n+1,i) = nod(n-1,i) + 1
20  CONTINUE

40 IF(ny.eq.1) GOTO 60

DO 50 i = 1, npe
   nod(1+1,i) = nod(m+1,i) + nxl
   nod(1+2,i) = nod(m+2,i) + nxl
50 K = 1 + 3
L = 0
YC = 0.0

DO 90 j = 1, nyl
   XC = 0.0
DO 80 i = 1, nyl
   L = l+1
   X(L) = XC
   Y(L) = YC
80 XC = XC + DX(I)
90 YC = YC + DY(J)

RETURN

C*---------------------------------------------------------------
C* mesh of quadrilateral elements with four nodes
C*---------------------------------------------------------------
100 NEXL = NX + 1
NEYL = NY + 1
NXX = IEL*NX
NYY = IEL*NY
NXX1 = NXX + 1
NYY1 = NYY + 1
NEM = NX*NY
NNM = NXX1*NYY1 - (IEL-1)*NX*NY
IF(NPE.EQ.9) NNM = NXX1*NYY1
K0 = 0
IF(NPE.EQ.9) K0 = 1

C*---------------------------------------------------------------
C* generate the array nod(i,j)
C*---------------------------------------------------------------
NOD(1,1) = 1
NOD(1,2) = IEL + 1
NOD(1,3) = NXX1 + (IEL-1)*NEX1 + IEL + 1
IF(NPE.EQ.9) NOD(1,3) = 4*NX + 5
NOD(1,4) = NOD(1,3) - IEL
IF(NPE.EQ.4) GOTO 200
NOD(1,5) = 2
NOD(1,6) = NXX1 + (NPE-6)
nod(1,7) = nod(1,3) - 1
nod(1,8) = nxx1 + 1
if(npe .eq. 9) nod(1,9) = nxx1 + 2
200 if(ny .eq. 1) goto 230
   m = 1
   do 220 n = 2, ny
   l = (n-1)*nx + 1
   do 210 i=1,npe
   nod(l,i) = nod(m,i) + nxx1 + (iel-1)*nexl + k0*nx
   220 m = 1
   do 260 ni = 2, nx
   do 240 i = 1, npe
   kl = iel
   if(i .eq. 6 .or. i .eq. 8) kl = 1 + k0
   nod(ni,i) = nod(ni-1, i) + kl
   240 m = ni
   do 260 nj = 2, ny
   l = (nj-1)*nx + ni
   do 250 j = 1, npe
   nod(l,j) = nod(m,j) + nxx1 + (iel-1)*nexl + k0*nx
   250 m = 1
   c*-------------------------------------
c*  generate the coordinates x(i) and y(i)
c*-------------------------------------
270 open(unit=114, file='convert26.dat')
   do 271 l=1,702
   read(114,*)x(i), y(i)
   write(*,*)x(i), y(i)
   271 continue
   RETURN
ccccccc270 yc = 0.0
   if(npe .eq. 9) goto 310
   do 300 ni = 1, neyl
   i = (nxx1 + (iel-1)*nexl) *(ni-1) + 1
   j = (ni-1)*iel + 1
   x(i) = 0.0
   y(i) = yc
   do 280 nj = 1, nxx
   i = i + 1
   x(i) = x(i-1) + dx(nj)
   y(i) = yc
   if(ni .gt. ny .or. iel .eq. 1) goto 300
   j = j + 1
   yc = yc + dy(j-1)
   i = i + 1
   x(i) = 0.0
   y(i) = yc
   do 290 ii = 1, nx
   k = 2*ii - 1
   i = i + 1
   x(i) = x(i-1) + dx(k) + dx(k+1)
290     y(i) = yc
300     yc = yc + dy(j)
   write(*,*)  ??????
   write(*,*)x(i),i=1,702
   write(*,*)  ??????
   write(*,*)y(i),i=1,702
   return

310   do 330 ni = 1, nyy1
      i = nxx1*(ni-1)
      xc = 0.0
   do 320 nj = 1, nxx1
      i = i + 1
      x(i) = xc
   y(i) = yc
      xc = xc + dx(nj)
320   yc = yc + dy(ni)
   return
   end

************ subroutine bndry(nrmax,ncmax,neq,nhbw,s,sl,nbdy,ibdy,vbdy)************

* subroutine used to impose boundary conditions on banded equations

implicit real*8(a-h,o-z)
dimension s(nrmax, ncmax), sl(nrmax)
dimension ibdy(nbdy), vbdy(nbdy)

print*, ' IN subroutine bndry'
   do 300 nb = 1, nbdy
      ie = ibdy(nb)
      sval = vbdy(nb)
      it = nb - nhbw
      i = ie - nhbw
   do 100 ii = 1, it
      i = i + 1
      if (i .lt. 1) goto 100
      j=ie-i+1
      sl(i) = sl(i) - s(i,j)*sval
      s(i,j) = 0.0
100   continue
   do 200 ii = 2, nhbw
      if (i .gt. neq) goto 200
      sl(i) = sl(i) - s(i,ii)*sval
      s(i,ii) = 0.0
200   continue
300   continue

RETURN
END

c*****************************************************************************
solver(nrm, ncm, neqns, nbw, band, rhs, ires)
c*****************************************************************************

c* Solving a banded symmetric system of equations  
c* in resolving, ires .gt. 0, lhs elimination is skipped.
c*
implicit real*8(a-h,o-z)
dimension band(nrm, ncm), rhs(nrm)
*print*, ' IN SUBROUTINE solver'
print*, ' COEFFICIENT MATRIX'
print*, ' LOAD VECTOR after adjustment'
write(*,777) (rhs(j),j=1,neqns)
777 format(f8.2)
meqns = neqns - 1
if (ires .gt. 0) goto 90

do 500 npiv = 1, meqns
   npivot = npiv + 1
   lstsub = npiv + nbw - 1
   if (lstsub .gt. neqns) lstsub = neqns
   do 400 nrow = npivot, lstsub

   c* invert rows and columns for row factor
   ncol = nrow - npiv + 1
   factor = band(npiv, ncol)/band(npiv, 1)
   do 200 ncol = nrow, lstsub
      icol = ncol - nrow + 1
      jcol = ncol - npiv + 1
      band(nrow,icol) = band(nrow, icol) - factor*band(npiv, jcol)
      200 continue
   rhs(nrow) = rhs(nrow) - factor*rhs(npiv)
500 continue
   goto 101
90   do 100 npiv = 1, meqns
      npivot = npiv + 1
      lstsub = npiv + nbw - 1
      if (lstsub .gt. neqns) lstsub = neqns
      do 110 nrow = npivot, lstsub
         ncol = nrow - npiv + 1
         factor = band(npiv, ncol)/band(npiv, 1)
         110 continue
      rhs(nrow) = rhs(nrow) - factor*rhs(npiv)
100  continue

   c* back substitution
   do 800 ijk = 2, neqns
      npiv = neqns - ijk + 2
      rhs(npiv) = rhs(npiv)/band(npiv, 1)
134

lstsub = npiv - nbw + 1
if (lstsub .lt. 1) then lstsub = 1
npivot = npiv - 1
do 700 jki = lstsub, npivot
nrow = npivot - jki + lstsub
ncol = npiv - nrow + 1
factor = band(nrow, ncol)
700 rhs(nrow) = rhs(nrow) - factor*rhs(npiv)
800 continue
rhs(1) = rhs(1)/band(1,1)
RETURN
END
References


Heravi, N.E., 1988, "Five-Spot Displacement in a Heterogeneous Reservoir at Unit Mobility Ratio," M.S. Thesis, New Mexico Institute of Mining and Technology, Socorro, New Mexico.


