APPLICATION OF ARTIFICIAL INTELLIGENCE TO RESERVOIR CHARACTERIZATION: AN INTERDISCIPLINARY APPROACH

Annual Report

By
D. R. Kerr
L. G. Thompson
S. Shenoi

May 1997

Performed Under Contract No. DE-AC22-93BC14894

The University of Tulsa
Tulsa, Oklahoma

Bartlesville Project Office
U. S. DEPARTMENT OF ENERGY
Bartlesville, Oklahoma

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED
DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, expressed or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government.

This report has been reproduced directly from the best available copy.

Available to DOE and DOE contractors from the Office of Scientific and Technical Information, P.O. Box 62, Oak Ridge, TN 37831; prices available from (615) 576-8401.

Available to the public from the National Technical Information Service, U.S. Department of Commerce, 5285 Port Royal Rd., Springfield VA 22161
Application Of Artificial Intelligence To Reservoir Characterization: An Interdisciplinary Approach

Annual Report

By
D. R. Kerr
L. G. Thompson
S. Shenoi

May 1997

Work Performed Under Contract No. DE-AC22-93BC14894

Prepared for
U.S. Department of Energy
Assistant Secretary for Fossil Energy

Robert E. Lemmon, Project Manager
Bartlesville Project Office
P.O. Box 1398
Bartlesville, OK 74005

Prepared by:
The University of Tulsa
Tulsa, OK
DISCLAIMER

Portions of this document may be illegible in electronic image products. Images are produced from the best available original document.
**Objectives**

The basis of this research is to apply novel techniques from Artificial Intelligence and Expert Systems in capturing, integrating and articulating key knowledge from geology, geostatistics, and petroleum engineering to develop accurate descriptions of petroleum reservoirs. The ultimate goal is to design and implement a single powerful expert system for use by small producers and independents to efficiently exploit reservoirs.

The main challenge of the proposed research is to automate the generation of detailed reservoir descriptions honoring all the available "soft" and "hard" data that ranges from qualitative and semi-quantitative geological interpretations to numeric data obtained from cores, well tests, well logs and production statistics. In this sense, the proposed research project is truly multi-disciplinary. It involves significant amount of information exchange between researchers in geology, geostatistics, and petroleum engineering. Computer science (and artificial intelligence) provides the means to effectively acquire, integrate and automate the key expertise in the various disciplines in a reservoir characterization expert system. Additional challenges are the verification and validation of the expert system, since much of the interpretation of the experts is based on extended experience in reservoir characterization.

The overall project plan to design the system to create integrated reservoir descriptions begins by initially developing an AI-based methodology for producing large-scale reservoir descriptions generated interactively from geology and well test data. Parallel to this task is a second task that develops an AI-based methodology that uses facies-biased information to generate small-scale descriptions of reservoir properties such as permeability and porosity. The third task involves consolidation and integration of the large-scale and small-scale methodologies to produce reservoir descriptions honoring all the available data. The final task will be technology transfer. With this plan, we have carefully allocated and sequenced the activities involved in each of the tasks to promote concurrent progress towards the research objectives. Moreover, the project duties are divided among the faculty member participants. Graduate students will work in teams with faculty members.

The results of the integration are not merely limited to obtaining better characterizations of individual reservoirs. They have the potential to significantly impact and advance the discipline of reservoir characterization itself.
Summary of Technical Progress

1. Decomposition of System

We have decomposed the overall system development into smaller component parts to allow us to focus on the expert knowledge required for that component. In addition, the decomposition will facilitate the implementation of the system and its validation and verification. The three component systems will be representative of how each of the experts in geology, geostatistics, and engineering characterizes the reservoir. Figure 1 describes a model for this breakdown. The concurrent development of these component systems fits into the development of the large and small scale aspects of the system as originally stated in the proposal.

The geostatistical system continues to be tested and updated. This system includes the use of wavelet transforms to determine the effect of compression to some part of the original data on the overall performance of the reservoir. Concentration on the geology system has been placed on upgrading the neural network output for log facies recognition. In addition, we have developed an automated system for correlation of zones among wells. The marker bed recognition system is considered complete at this time, though later enhancements may be added. The individual components (completion rules, type curve matching, and linear regression components) are currently being integrated to form a complete well test interpretation system. The graphical system is currently being designed for implementation to visualize correlations between wells. This system will be augmented as the other system components mature. The designing of the overall user interface to integrate all of the systems has begun.
Figure 1: Overall System Components
2. Geostatistical System: Integrated Lithofacies and Petrophysical Properties Simulation

2.1 Overview

This report presents the new procedure that simultaneously generates consistent distributions of categorical lithofacies and continuous petrophysical properties, i.e., porosity and permeability. The technique used is the conditional simulation method which is capable of honoring the original distribution of the data and the associated spatial relationship. The simulation of lithofacies is conducted using the combination of indicator simulation and truncated Gaussian simulation techniques whereas the porosity simulation is conducted using the sequential Gaussian simulation. To generate the permeability simulation, the conditional distribution technique is used.

Previously, to generate petrophysical properties consistent with geological description, investigators have used a two-step approach where geological description are constructed first, and the petrophysical properties are described in the next step through a filtering process. In contrast, in the approach used in this report, each grid block is visited only once. Using the same search neighborhood, the geological facies is estimated first, followed by porosity and permeability values. The method accounts for correlations among these variables as well as the spatial relationships. This reduces the storage requirements and makes the process computationally efficient.

The method was successfully validated using the field data. Both sandstone and carbonate fields were used to generate the facies/rock type, porosity, and permeability distributions. The simulated geological descriptions matched well with the geologists’ interpretation. Further the results were consistent with the observed production performance in terms of quality of rocks and petrophysical properties.

The implementation of this new procedure is done using the C++ language. The implementation includes the user interface program which operates in Windows 95 / NT system. Using this program, the user is able to prepare the required parameter file, to run the simulation, and to view the graphical results, i.e., the cross sections, directly from the screen.

2.2 Introduction

Reservoir characterization is the process of generating reservoir properties, mainly porosity and permeability, by integrating many types of data. An objective of performing reservoir characterization is to better predict the future performance of the reservoir. A tool that can be used to perform reservoir characterization is geostatistics. Various geostatistical techniques have been developed in the industry to perform this task. Some of
the most important techniques are conditional simulation techniques which include sequential Gaussian simulation, sequential indicator simulation, truncated Gaussian simulation, probability field simulation, etc.

A common practice in the industry in developing reservoir properties with a consistent underlying geological description is to apply the two-stage approach\textsuperscript{25,26}. In the first stage, the geological description is simulated using a conditional simulation technique such as sequential indicator simulation or Gaussian truncated simulation to produce the detailed geological simulation. In the second stage, petrophysical properties are simulated for each type of the geological facies/unit using a conditional simulation technique such as sequential Gaussian simulation or simulated annealing. The simulated petrophysical properties are then filtered using the generated geological simulation to produce the final description. The main problem of this approach is efficiency, since it requires intensive computation time.

The objective of the study presented in this report is the development of new cosimulation procedure that simultaneously simulates the geological, i.e., lithofacies description and the associated petrophysical properties, i.e., porosity and permeability. There are two advantages that are expected from this new procedure, namely the consistent petrophysical properties with the underlying geological description and computation efficiency.

As in any other engineering program, most of the algorithms in geostatistics are implemented using the FORTRAN language. The problem that is encountered with FORTRAN is the difficulty in extending or reusing the previously-developed code to implement the newly-developed techniques. Following the progress achieved in computer science technology, the object oriented paradigm is thought to be appropriate for implementation in this field. This is mainly due to the philosophy of the object oriented paradigm which is extendibility and reusability of a computer program. The most common computer language that uses this paradigm is the C++ language. This language is used to implement the cosimulation program, which is further referred as COSIM program, presented in this report.

The other reason for using the C++ language instead of FORTRAN is the ease in the development of a user-friendly program. The necessity of user friendliness may be appreciated by the fact that many sophisticated techniques have not been frequently used because they are very difficult to use. Therefore, in addition to having a good technique, a computer program should also possess the characteristics of being easy to use.

2.3 Background Theory

2.3.1 Conditional Simulation Methods

Conditional simulation is a geostatistical method to generate description of reservoir properties which uses the available quantitative and qualitative data. This method is a stochastic approach because reservoir properties are represented by random variables. The description of the properties generated by this method are conditional since the
available data are honored at the sampled locations. And, the method simulates several equiprobable descriptions of the actual distribution of a property in the reservoir. In constructing the possible reservoir descriptions, the constraints imposed on the simulation process may include prior distribution of the simulated variables, spatial relationships in various directions and geometry of geological shapes and sizes. As more constraints are incorporated in a conditional simulation process, more similar would be the equiprobable images.

There are three ways in which conditional simulation techniques differ from conventional techniques:

- **Sample Distribution Data Honored**:
  Unlike simple interpolation or extrapolation, conditional simulation honors the entire sample data distribution rather than reducing the spread of the data distribution. This is important for retaining extreme values (outliers) in the sample data set, which form a very small part of the overall sample, but which may greatly influence the flow performance of the reservoir. An example would be a small streak of high permeability, which can have significant influence on waterflood performance, and still constitute a very small part of the entire productive intervals in terms of the total sample distribution.

- **Data Spatial Relationships Honored**:
  The second advantage of the conditional technique is that it honors the spatial relationships developed from the sample data. Many conventional interpolation methods generate smooth distributions which do not satisfy the spatial relationships established using the sample data.

- **Reservoir Description Uncertainties Quantified**:
  The last advantage of the conditional simulation method is its ability to quantify uncertainties in the reservoir description through multiple, equiprobable images of the reservoir. Conditional simulations allow construction of multiple pictures of the reservoir, all observing the same constraint(s).

Several conditional simulation techniques have been proposed in the literature. The method used to generate the facies and petrophysical properties in this report is the combination of indicator and Gaussian simulation methods as well as the conditional distribution technique which is the subject of the following section.

### 2.3.1 Co-Simulation of Facies and Petrophysical Properties

As mentioned previously, common practice in generating reservoir description in the industry is the two stage approach where at the first stage the rock type or the geological facies is simulated followed by the simulation of the petrophysical properties at the second stage. The process at the second stage requires a lot of computation time and computer storage to hold the temporary results which will be discarded after combining with the results of the first stage through filtering process. Therefore, if we can combine these two processes in one, an efficient simulation will be obtained.

To eliminate the two stage approach and to reduce the computation time, in this approach, the grid block is visited only once. Using the same search neighborhood, the
geological facies is estimated first, followed by porosity and permeability. The method accounts for correlations among these variables as well as the spatial relationships. This reduces the storage requirements and makes the process computationally efficient while maintaining the consistency between the generated petrophysical properties with the underlying geology.

The use of indicator technique for facies simulation is appropriate based on the nature of the facies itself. Facies, similar to indicators, can be described by discrete variables. That is, facies can only take finite number of values. Therefore, an indicator variable can be defined for each grid block by assigning a value of 1 for the facies present and 0 for the absent facies. For example, for a system with four facies an indicator value of [0,1,0,0] can be interpreted as facies 2 is present at that location and all the other facies are absent. For each available data (i.e., facies at the well location) we can convert it into corresponding indicator variable. Then, we can model the spatial relationship for each facies using variograms. These variograms are used to generate the proportion curves of the facies, through an indicator kriging technique, at each grid block. The proportion curve are used later to back transform the facies from Gaussian space as explained in the following paragraph.

To estimate the facies value at the unsampled location, the truncated Gaussian simulation technique is used. This technique requires the simulation of facies value in the Gaussian space and the proportion curves at each grid block to back transform the simulated facies from the Gaussian space. The use of Gaussian simulation indicates that the program requires a variogram model for the facies in the Gaussian space in addition to the variogram model for each facies as mentioned previously. The simulation is conducted sequentially where each unsampled location is visited only once. As in any sequential Gaussian simulation technique, the number of points used in the kriging process to determine the value of the unsampled point is given by the user through the definition of search neighborhood. Once the simulated Gaussian facies is obtained at each grid block, the proportion curve can be used to truncate the Gaussian facies into the original facies definition. Detail information about the Gaussian truncated simulation technique can be found elsewhere.28

The porosity simulation is also conducted using the sequential Gaussian simulation technique. This means that the program also requires the information about the variogram model of the porosity in the Gaussian space. The porosity simulation is conducted simultaneously with the facies simulation. Same data points from the same search neighborhood are used in the kriging process when simulating porosity. This technique saves computational time and effort.

To ensure that the porosity value is consistent with the underlying geological facies, the program generates the correlation between facies and its corresponding porosity value of the sampled data. This correlation is then used when back transforming the result of porosity simulation from the Gaussian field. This way, the resulting porosity distribution is consistent with the facies simulation, i.e., with the underlying geological description.
The next step in the simulation is the simulation of the permeability values. The conventional way of generating permeability distribution is through the linear relationship between porosity and log of permeability. In many situations, this linear relationship is not well defined. The technique used in this report is the conditional distribution technique where the correlation between porosity and permeability is first developed without trying to fit it into certain functionality such as linear relationship. Instead, for a particular facies, porosity data are divided into several classes. For each class, all the corresponding permeability values are gathered, and a distribution function of permeability for that class is created. Once a porosity is assigned at a particular location, we know the porosity class from which the porosity value comes from. Permeability is then randomly sampled from the conditional distribution. This way, the relationship between the porosity and permeability, as evident from the sample data, is honored. For example, if we have four facies and for each facies we divide the corresponding porosity data into four porosity classes, we will get sixteen conditional distributions of permeability corresponding to each porosity class. Reference 29 provides more detail procedure for conditional distribution technique.

2.4 Program Structure

The implementation of the simulation technique discussed in the previous section is done using the C++ language. The choice of this language over the traditional FORTRAN language is based on the following two reasons:

1. Extendibility and reusability of the source code.

2. User friendly interface.

It is expected that the source code developed for this application can be adapted easily to the future development of other simulation technique. At the same time this program is also expected to be widely used since it uses the state of the art programming style, i.e., Windows based program for PC users.

2.4.1 Calculation Module

The list of the classes used in the program is summarized in Table 1. The interrelationship among these classes is shown in Figure 2. The arrowed line indicates the inheritance relationship, e.g., between class Application and class Cosim, whereas the line ending with a circle indicates where the class is being used. For example, class Variogram is used in class Kriging, and class Kriging is used in class GausSim.
<table>
<thead>
<tr>
<th>Class Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application</td>
<td>Main driver for the cosim program</td>
</tr>
<tr>
<td>Cosim</td>
<td>Define the simulation technique to be used</td>
</tr>
<tr>
<td>IndSim</td>
<td>Provide the procedure to perform indicator simulation.</td>
</tr>
<tr>
<td>GausSim</td>
<td>Provide the procedure to perform the Gaussian simulation.</td>
</tr>
<tr>
<td>CondDist</td>
<td>Provide the procedure to perform the conditional distribution technique in generating the permeability distribution and the storage of the related correlation between porosity and permeability</td>
</tr>
<tr>
<td>Kriging</td>
<td>Provide the procedure to estimate the node value either by Simple kriging or Ordinary kriging technique with or without covariance table.</td>
</tr>
<tr>
<td>Variogram</td>
<td>Provide the calculation of variogram and/or covariance value between any two points in 3D for a given Variogram model.</td>
</tr>
<tr>
<td>CovTab</td>
<td>Provide the calculation and storage for the covariance table.</td>
</tr>
<tr>
<td>Grid</td>
<td>Provide the grid block network of the simulation system that includes the neighborhood searching technique such as super block search.</td>
</tr>
<tr>
<td>Point3D</td>
<td>Provide the structure to represent a 3D point. This class is generated using a template that can accept any data-type.</td>
</tr>
<tr>
<td>Utility</td>
<td>Provide several utility functions that are common in geostatistical simulation technique such as random number generator, inverse of Gaussian data, normal transformation, etc.</td>
</tr>
<tr>
<td>listClass</td>
<td>Provide the link list of the data to store variable with unknown size. This class is developed using template facility to accept any data-type.</td>
</tr>
<tr>
<td>Matrix</td>
<td>Provide the procedure related to matrix operation. This class uses template facility</td>
</tr>
</tbody>
</table>

Table 1  C++ Class Summary Used in the Program
Figure 2  Program Structure
Class Application is the main driver for the program. It is developed using the principal of polymorphism and dynamic binding where it contains a virtual function called DoSimulation. This virtual function is defined inside some other classes which are inherited from the Application class and dynamically binded during the execution of the program. Therefore, common features of the Application class, such as development of the grid system, variogram definition, correlations among variables, etc., can be used for different techniques of simulation. At this time, there is only one class derived from the Application class, i.e., the Cosim class. Future extension of this program would take advantage of this structure.

The Cosim class consists of the functions to perform the cosimulation of lithofacies and petrophysical properties. There are 3 main classes which are used in this class, i.e., the IndSim for Indicator Simulation, the GausSim for sequential Gaussian simulation, and the CondDist for conditional distribution technique for generating the permeability distribution. The class GaussSim can be used for either generating porosity distributions alone or for both rock type and porosity in which the technique of truncated Gaussian simulation (GTSIM) is applied. In the case where it is used for GTSIM, the IndSim is used for generating the proportion curve which is required in transforming the Gaussian simulated result back to its indicator value.

The kriging class is used to perform the kriging of the unsampled value. In developing the covariance matrix, this class can either use the covariance table which is provided by class CovTab, or directly calculate the covariance between two points given the variogram model. The class stores the information about the variogram model input by the user and provide the routine to calculate the variogram or covariance value between two given points in three dimensional space. The CovTab class stores the covariance value between two points which is defined by the super block searching technique.

Several other classes which are grouped together as the Miscellaneous class are used by almost all of other classes. This is due to its functionality that basically provides the utility function and common data structure.

2.4.2 Interface Module

The compiler used to develop the interface for this program is Microsoft Visual C++ 4.0. This compiler is found to be the best for current development of Windows programming. This compiler requires Windows 95 or Windows NT operating system. It is anticipated that most of the PC users use this operating system. The layout of the program once it is executed is shown in Figure 3. It can be seen that in addition to several default menus of Windows program such as File, Edit, View, and Help, this program has 3 main menus, namely Pre Simulation, Simulation, and Simulation.
For simplicity, the first version of this program is built using the Single Document Interface (SDI) technique. This means that the program has only one window and only one document can be loaded at a time. Even though this technique restricts the number of windows that can be opened at one time, its performance will not be significantly affected. The future version of this program is planned to be developed using the Multiple Document Interface (MDI) technique.

The File menu consists of several sub-menus those control the loading and unloading of the document. This document contains the information of parameter file which is saved in the binary format. The Print command, which is part of File menu, is fully operational, but it has not been configured to give the “good” size. Some work is needed to obtain good hard copy.

The only command that is fully operational from the Edit menu is the Clear Screen command. This command is used to clear the screen. When the program is first executed, blank white screen appears. After the user creates the image on the screen, the Clear Screen command or the corresponding toolbar, i.e., the scissor, can be used to clear the entire screen to obtain the blank white screen.

The View menu consists of two fully operational commands; the View Tool Bar and the View Status Bar. These commands can be used to enable or disable the tool bar and/or the status bar. Disabling these tool bars will generate bigger view area and vice versa.

2.4.3 Pre-Simulation

The Pre-Simulation menu consists of two sections. The first section is related to general geostatistics tool such as the variogram analysis, the statistical analysis of the data, and the normal transformation which is used for Gaussian type simulation. The second
section is specifically related to the COSIM program, i.e., the building of parameter file. This parameter file is the main input for the COSIM calculation. Current version of the program consists of the second section only.

In building the parameter file, the user can either open the previously generated file or create the new one. The dialog is designed using the property sheet where there are 8 properties defined for the whole parameters. Effort has been made to make the use of this facility as easily as possible. Figure 4 presents the dialog that guides the user in building the parameter file.

![Parameter File Dialog](image)

**Figure 4** Dialog window to generate parameter file

### 2.4.4 Simulation

The Simulation menu is the menu to use when the user ready to execute the simulation calculation. The default or the latest parameter file selected by the user is prompted by the program to be used for simulation. The user can change this file as required. Once the simulation is finished, the message box appear on the screen to inform the user. The output of the simulation is written out into the text file specified in the parameter file.

### 2.4.5 Post-Simulation

The post simulation menu is designed to help the user in evaluating the simulation results. Current version is capable of performing qualitative evaluation only by presenting the 2D cross section of the simulation result. To generate a 2D cross section, the program provides the dialog to guide the user in selecting specific cross section that he/she wants to see. The user can choose to draw the image on the screen or create an ASCII file which...
is suitable for other commercial drawing package such as Spyglass Transform. Figure 5 shows the dialog to create the 2D cross section on the screen.

To generate the cross section, the user needs to supply output file name and grid dimension of the system. The user can select to plot the XY, XZ, or YZ plane, the grid block level/depth, and the scale of each variable. The default color scheme will be used by the program for the first image. Once the image is generated the user has the flexibility to change the drawing attributes such as plot scale, color scheme, or grid block size. These options are available from floating menu that can be accessed by clicking the mouse’s right button anywhere inside the drawing area.

2.5 Result

To demonstrate the capability of the program in simulating the consistent petrophysical properties with the underlying geological description, two cases are presented in this section. These two cases are generated based on the field data. The first data set are taken from the Self Unit, Glenn Pool field, Oklahoma, which is the sandstone reservoir whereas the second data set are taken from the North Robertson Unit, West Texas, which is the carbonate reservoir. In both cases, the geologist has generate the geological description at well location. A set of porosity logs are available at several wells and a correlation of facies/rock type, porosity, and permeability are available from the core data.
2.5.1 Sandstone Reservoir

The geological unit used by the geologist in describing the Glenn Sandstone of Self Unit is the Discrete Genetic Interval (DGI). Reference 6 provides detail information about this geological description. Kerr and Ye36 have defined 6 DGI’s in the vicinity of Self Unit, namely DGI A through F. Each DGI may consist of several facies such as channel fill, crevass splay, and floodplain mudstone. Two results are presented in this section. The first one is the simulation of all DGI without further differentiating the DGI into facies whereas in the second result the distribution of channel fill and crevass splay of DGI D is shown together with its associated petrophysical properties. These results show the user about the ability of the program in simulating any geological unit used by the geologist.

Figures 6 through 8 show the Vertical cross section of the DGI, porosity, and permeability distributions, respectively. The number of grid blocks used in this example is 256,000 (40x40x160) grid blocks where the size of each block is 66 x 66 x 1 ft. From these figures we can see clearly how the DGIs are varied from top to bottom with shale on the top and bottom borders. We can also observe how the petrophysical properties change from top to bottom. The increase in porosity and permeability from top to bottom are matched very well with the field observation, i.e., from well log and core data.

Total computation time to complete the simulation is about 2 hours on the PC Pentium 150 MHz compared to about 2 days work that is needed for the two stage approach where porosity simulation is required for each facies defined in the simulation. This is not to mention about preparation that has to be made independently for the filtering process and permeability simulation. It is realized that the 2 hours computation time is not yet optimum. The program can be made more efficient in the future.
Figure 9 presents the 2D cross section of DGI D created by simulation and its comparison with the geologist’s interpretation. This figure is shown to demonstrate the capability of the program in simulating the geological facies, e.g., channel fill and crevass splay. The corresponding porosity and permeability distribution are shown in Figure 10. We can observe that the generated petrophysical properties are consistent with the underlying geological description.
Figure 9  
Comparison between simulated and geologist's interpretation of DGI D

Figure 10  
Porosity and Permeability distribution of DGI D.
2.5.2 Carbonate Reservoir

As mentioned previously, the data for this case study is obtained from the North Robertson Unit, West Texas. The geological unit used by the geologist in describing the reservoir is rock type. Therefore, the term facies will be replaced by rock type in this case. There are 8 rock types defined in the system, namely rock types 1 through 8, but only 4 of them are reservoir rocks, i.e., Rock Type 1, 2, 3, and 5. The other 4 rock types, i.e., Rock Type 4, 6, 7, and 8, are classified as non-reservoir rock. In this simulation, all non-reservoir rocks are combined and renamed as Rock Type 4. Thus, for the simulation purposes, only 5 rock types are simulated, namely Rock Type 1 through 5.

Figures 11 through 13 show the 3D view of the Rock Type distribution with its associated porosity and permeability distribution. The number of grid blocks used in this case study is 715,473 (99 x 99 x 73) with the size of each grid block is 55 x 55 x 1 ft. As in sandstone case, we can also observe that petrophysical properties are consistent with the underlying geological description. In this particular example, Rock Type 5 is not part of the conditioning data. Therefore, none of the grid block has the property of Rock Type 5.
Figure 11  3D View of Rock Type distribution

Figure 12  3D View of Porosity distribution
Several conclusions can be drawn based on this study:

- The new procedure to simultaneously simulate the lithofacies and petrophysical properties, i.e., porosity and permeability, has been developed.
- The geological simulation is conducted using combination of indicator simulation and truncated Gaussian simulation.
- Porosity simulation is conducted using sequential Gaussian simulation.
- Permeability simulation is conducted using conditional distribution technique.
- The efficiency of the program is obtained by performing single search neighborhood which is used for both facies and porosity simulation.
- The implementation of the technique is done using the object oriented paradigm, i.e., C++ language, in a user friendly environment.
- The program is tested using field data, both sandstone and carbonate reservoir. It has been observed that the simulated geological description agrees very well with the geologist's description and the petrophysical properties are consistent with the underlying geological description.
3. Geostatistical System: Incorporation of Dynamic Constraints in a Reservoir Description Process

3.1 Overview

Throughout the life of a petroleum reservoir, information about the reservoir -- typically seismic, core and log data, rates and pressures -- is collected. These data represent perspectives of the reservoir at widely-differing scales. The challenge therefore, is to integrate these different scales of data into a reservoir description process while honoring the scale level they represent. In addition, algorithmic efficiency may prevent the consideration of certain types of data, for example 'dynamic' data, such as rate and pressure data. These represent valuable information because they represent a heterogeneity scale which is closer to simulation gridblock scales than the typical well core/log data. So far, dynamic data have been used on a limited basis because it then becomes necessary to use flow simulation, making the algorithm inefficient.

An efficient algorithm for using stochastic conditional simulation in which we directly incorporate dynamic information -- namely, rate and pressure information is presented. Modeling the spatial description on a fine scale and the flow on an upscaled grid reduces the flow simulation execution time and allows for a faster conditional simulation algorithm. Upscaling approaches are described which result in flow performance matching between the fine or 'true' scale and the upscaled grid. The simulated annealing method is used with a 2-part objective function consisting of a variogram constraint and a flow simulation constraint with each part being appropriately weighted.

Improvements in the reservoir description when the dynamic constraint is included are demonstrated. Also the upscaling techniques used are shown to be effective in matching the flow performance of the reservoir grid between scales. The procedure simultaneously generates fine scale description which can be used for future evaluation, and a coarse scale description which honors the production data. Using this procedure, up to a 10,000-gridblock description has been simulated.

3.2 Introduction

Modern reservoir characterization allows the estimation of reservoir properties at particular spatial locations using geostatistical techniques. Geostatistics quantifies the spatial relationships of neighboring values for a particular reservoir characteristic, and uses that relationship to estimate the value at unsampled locations.

It is important and becoming more popular to integrate the static and dynamic data for reservoir characterization. By integrating different types of data, the final reservoir characterization will be more precise or unique. In other words, the uncertainty of the reservoir description will be reduced. However, for optimal reservoir management, it is more critical to understand the uncertainty of the future performance prediction.
geostatistics, the norm is to generate multiple realizations of the reservoir description and conduct the reservoir simulation. The multiple realizations of the future performance prediction form the prediction error bound. This is a more flexible approach than the traditional serial modeling approach for performance prediction. Conventionally, the geostatistical modeling incorporates the static data -- such as seismic, geological data -- quite well, but may not reproduce the dynamic performance. The incorporation of the dynamic data into the modeling may yield better history matching. However, the prediction characteristics need to be understood.

The constraints imposed on the simulation process may include a prior distribution function of the simulated variable, spatial relationships in various directions and the geometry of geologic shapes. In addition, conditioning data may be used to further assist in the development of a reservoir model which better approximates the truth. These data are usually actual field measurements. The differences among the resulting equiprobable images become smaller as more conditioning and constraining are applied.

An important distinction which must be made is that between static and dynamic data. The two data types previously cited -- permeability and porosity -- represent static data; i.e., their values do not (significantly) change over time. This facilitates their use as conditioning data. On the other hand, production data are an example of dynamic data, because they are a function of time. This complicates their use for conditioning, because it now becomes necessary to solve an inverse problem. This is unfortunate, since production and pressure data are reservoir characteristics of primary interest in reservoir studies. The main reason being the economic necessity to predict present and future reservoir performance.

For full-field studies, the incorporation of dynamic constraints has been stymied because of the prohibitively-high cost in computing time required for the flow simulation phase. The major reason for this is the necessity to repeat the calculations in time-stepping between the initial time and the final flow period. It seems reasonable, therefore, that the ability to utilize only one timestep over the same period would significantly reduce the computing time requirements.

This suggested an approach which could be used. It is based on the Laplace Transform Finite Difference (LTFD) method, a procedure in which the problem is solved in Laplace space. This procedure removes the timestep size constraint as exists for an IMPES-type method, because the time variable is transformed into the Laplace variable, 'u'. Thus it is semi-analytical in time and numerical in space. Hence, stability is conserved without having the timestep size limitation of explicit methods. Instead, one can use just one timestep between the initial time and the time of interest. As regards more implicit methods, e.g., Newton's method, the LTFD method has the advantage in that the calculation of gradients is not required, making it considerably less computationally-intensive. It is expected therefore that this approach, in terms of computational time, will be less expensive. Also, by using the flow simulation results in Laplace space, we obviate the need for numerical inversion (via the Stehfest algorithm) to real time and space.

In addition, the smaller the number of grids used for the flow simulation, the faster is the execution time. Thus another way of accelerating the execution time of the
algorithm would be to perform the flow simulation on an upscaled grid. We have to ensure, however, that the flow simulation performance of the coarser grid is representative of the "true" or fine scale grid within the time range used in the objective function of the simulated annealing algorithm.

The flow simulation is incorporated into the simulated annealing procedure as part of the objective function, in a general sense, as a difference between the observed and estimated flow rates or pressure values at discrete time intervals. Thus the reservoir model will be constrained by the production data in this manner. It is expected that reducing the computation time will result in a viable solution procedure in which production data -- or more generally one or more dynamic constraints -- may be incorporated.

As a summary, the objective of this study is to incorporate dynamic constraints, in this case production and flowing bottomhole pressure data, for reservoir characterization using a stochastic conditional simulation method, specifically the simulated annealing (SA) method.

3.3 Approach

The approach involves the coupling of a numerical flow simulator to the simulated annealing algorithm. The flow simulator will constitute a dynamic constraint as part of the simulated annealing objective function. Our objective function will then consist of a variogram constraint and a flow simulation constraint.

Simulated annealing is an example of adaptive heuristics for multivariate or combinatorial optimization. Its implementation requires, among other parameters, the definition of an objective function which represents a measure of the difference between an input model and our "fit" of that model. The goal of the algorithm is thus to determine the configuration of independent variables -- for example, permeability values -- that minimizes the objective function. Note that the objective function can be comprised of more than one model, in which case we attempt to minimize the sum of the differences.

The Laplace Transform Finite Difference (LTFD) method is used for the flow simulation. This method is based on solving the single-phase flow equations in Laplace space. It has the advantage of using only one timestep between the initial time and the time of interest. As a result this method is computationally far less expensive than other numerical methods.

3.3.1 Input Data

Analysis of the behavior of the composite objective function led to the conclusion that the rate of convergence was too slow. Additionally, the terminal objective function value of about 0.1 was too high for one to conclude that convergence to a globally-optimal solution had occurred.

For the flow simulation part of the objective function, we compare observed pressures to pressures calculated by the flow simulator as a means of judging how closely the performance of the estimated model approximates the "true" reservoir description.
This comparison is done in Laplace space, and so a routine to convert the real time and space data to Laplace space is used.

We determined that the precision of the Laplace transform of the input data could be improved by the inclusion of more data values for better interpolation and by choosing the appropriate data extrapolation procedures (which depends on the flow regime characteristics at the time from which we have to extrapolate). This improvement in precision produced better results in that convergence was faster and the terminal objective function value reached about one order of magnitude lower at approximately 0.01. Also the maximum absolute percentage error between the 'observed' and calculated Laplace space pressure values was reduced to about 0.8%. It is thus necessary to undertake some pre-analysis of the input data to ensure that the data quality and quantity are adequate and also that the appropriate algorithms are used.

3.3.2 Training Images

In recognition of the direct relationship between the time required for running the simulated annealing algorithm (especially when the flow simulation is included) and the level of grid discretization, it was decided to undertake runs with smaller number of grids (upscaled grids). The algorithm would then run faster, enabling speedier analysis of the optimization schemes. The use of various approaches was considered for generating the "training images" used in testing our algorithm. The turning bands method, sequential Gaussian simulation (SGS) and simulated annealing (SA) were some of these. Ultimately, SA was used to generate the training image.

3.3.3 Algorithmic Considerations

By far, the major effort was expended in developing and improving the algorithm. The three main packages analyzed were the flow simulation code, the matrix solver and the SA code. Changes to the SA algorithm were considered only in terms of modifications made to implement the flow simulation constraint. The basic, variogram constraint-handling SA code was considered sufficiently optimized. Changes to the matrix solver were made, not by modifying the code, but by considering alternate solvers. The flow simulation code, however, was closely scrutinized. From the above comments, it can be seen that all these efforts were directed at improving and optimizing the flow simulation performance, since this was recognized as the critical area needing attention.

3.3.3.1 Flow Simulation Code

Coefficient Matrix Code. The heart of the flow simulation code is the development and solution of the matrix of coefficients for the gridblock pressures. We did a detailed study of the structure of that matrix and determined that, for the given boundary conditions, the matrix is a 7-band, sparse, symmetric system. There is little scope for improving the matrix formulation code, but we were able to optimize the code which updates that matrix (which is required when we perturb the description). We determined that on update only 19 matrix elements are affected (13 for a 2-D grid), thus we need to change only those elements. Initial testing has shown that the changes have resulted in some gain in efficiency, but not of the magnitude hoped for.
Variable Rates or Constant Rates. When variable rates are used — i.e., rates which are step functions of time — it has been observed that the LTFD method results in pressure changes, coincident with the rate change, which are "smoother" than they should be. This smoothing problem is confirmed when we compare the results from the LTFD simulator with the results from the ECLIPSE-100 (ECL) simulator. In fact, in general the use of variable rates with Laplace transforms is problematic. One solution has been to use a constant rate solution in Laplace space and use superposition in real space to correct for the variable rates. This is not a viable approach for our case, since we solve our problem entirely in Laplace space. Recently a procedure was presented for correcting this anomaly. This has led to our consideration of two approaches:

- the use of the Chen et al. approach to correct the variable rate problem,
- the use of constant rates to avoid the Laplace transform problems and the problem of the introduction of new transients at each rate change.

The equation given in reference 12 for the "corrected" bottomhole pressure in Laplace space is:

$$-ar{p}_{wD}(s) = \frac{q_{D_i}}{s_1^2 \left[ C_D + \frac{1}{s_1 G(s_1)} \right]} + \sum_{j=2}^{n} \frac{t_D - t_{D_{j-1}}}{s_j^2 \left[ C_D + \frac{1}{s_j G(s_j)} \right]}$$

Firstly, it seems that this equation may be incorrect, so we will have to verify its accuracy before any further consideration of its usage. Secondly, the structure of the equation suggests that, in order to determine the variable rate, Laplace space, bottomhole pressure for a particular value of $s$, we have to determine the constant rate, Laplace space solutions for a number of $s$ values equal to the number of times the rates are changed. This would mean introducing a significant extra computing "load" which would add to the algorithm's inefficiency. For this reason, we have suspended further consideration of this approach and for the moment we are concentrating on the constant rate approach.

3.3.3.2 Matrix Solver

We considered matrix solvers from the SLAP, ITPACK and Templates packages. Some solvers promise faster execution times; however, because of the specialized nature of those approaches, they have not yet been tested. For example, red-black ordering implies parallelization of the code — something which we are not considering at present. We did not test the Templates codes because of time constraints and there seemed to be minimal advantage in exploring this alternative. Table 2 summarizes the results obtained with the packages tested. A 224-block dataset was used for the testing.
<table>
<thead>
<tr>
<th>Package</th>
<th>Method</th>
<th>Run Time (CPU Sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ITPACK</td>
<td>Jacobi Conjugate Gradient (JCG)</td>
<td>14096</td>
</tr>
<tr>
<td></td>
<td>Symmetric Successive Overrelaxation</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Semi-Iteration (SSORSI)</td>
<td>44084</td>
</tr>
<tr>
<td>SLAP</td>
<td>Conjugate Gradient (CG)</td>
<td>13824</td>
</tr>
<tr>
<td></td>
<td>Generalized Minimum Residual (GMRES)</td>
<td>22045</td>
</tr>
<tr>
<td></td>
<td>Bi-Conjugate Gradient Squared (BCGS)</td>
<td>14283</td>
</tr>
</tbody>
</table>

Table 2: Comparison of Run Times for Various Matrix Solvers

Based on the above results, we have selected the SLAP CG code for use. Consultation with the Mathematics Department has suggested that our choice was the appropriate one.

3.3.3.3 SA Code

\( NVAL \) Parameter. \( NVAL \) is the parameter used for the number of pressure values per well used for comparison in the flow simulation part of the objective function of the simulated annealing algorithm. This is defined in the following equation:

\[
\text{Flow Simulation Part for Iteration, } k = \frac{1}{E_2} \sqrt{\sum_{i=1}^{NDIFF} \left[ \frac{L[\Delta p(i)]_0^0}{L[\Delta p(i)]_0} - 1 \right]^2} 
\]  

(2)

where \( wgt2 \) =weighting applied to the flow simulation part

\[
E_2^0 = \sqrt{\sum_{i=1}^{NDIFF} \left[ \frac{L[\Delta p(i)]_0^0}{L[\Delta p(i)]_0} - 1 \right]^2} = \text{Initial Flow Simulation Energy} 
\]  

(3)

\( NVAL \) = Number of summation terms per well
\( NWELL \) = Number of wells
\( NDIFF = NWELL \times NVAL \) = number of terms used in the summation
\( L[\Delta p] \) = the Laplace transform of the change in the flowing bottomhole pressure

Ten was the \( NVAL \) value being used. The run time of one iteration in the algorithm is proportional to \( NVAL \), so we analyzed the effect of the value of \( NVAL \) on the performance of the code. Values of \( NVAL \) ranging from five to twenty were used. The observations are:

- The lower the value of \( NVAL \) the shorter the cycle time, but, in some cases, more cycles are needed for convergence. This suggests that there is/are optimum values for \( NVAL \). It seems that this value is about six or seven. However, these results are for just one dataset so far.
- The relative “goodness of fit” of the distributions obtained using the different \( NVAL \) values is difficult to determine by visual inspection (i.e., whether one description is better than another) and, since we are still researching an appropriate objective measure of goodness of fit, has not yet been determined.
Variogram Pre-Selection Approach. It was suggested\(^7\) that we could significantly improve the execution time of the SA algorithm if we could determine before running the flow simulation whether a perturbation would be "beneficial". We subsequently determined that using the variogram-based objective function change as a criterion for deciding whether a perturbation would be accepted, was a good approach since the acceptance of the variogram-only change was highly correlated with that of the two-part objective function change (conservatively over 60%).

Two-Scale Approach. If we could use an upscaled grid for the flow simulation part of the algorithm, then we would be able to reduce the execution time of the algorithm considerably. In addition, this approach would allow the consideration of grid sizes (at the fine scale) which were previously prohibitively large for flow simulation. Note that we would still perform the variogram analysis at the finer scale. The upscaling methodology is the major problem to be solved for this approach to be viable. While the upscaled distribution "resembles" the true distribution, matching the flowing bottomhole pressures obtained from flow simulation at the different scales is a problem.

Two-Scale, Variogram Pre-Check Approach. This is a combination of the ideas in previous two sections. It is the approach on which we are currently focusing. In this approach the objective function consists of two parts: a variogram component and a flow simulation component. The perturbation strategy is single-point. The algorithm proceeds as follows:

1. An initial distribution is generated which honors the spatial statistics and the conditioning data.
2. The initial temperature level is determined by the procedure suggested by Aarts and Korst.\(^8\)
3. A coarse scale gridblock location is randomly selected. Based on the upscaling factor, the corresponding fine scale grid locations are determined, and one of these is perturbed. We evaluate the variogram part of the objective function on the fine scale grid system and test the energy change via the Metropolis condition. This is the pre-selection stage.
4. If the perturbation is accepted then the average of the permeabilities of the fine scale blocks -- which make up the upscaled block selected in part 3 above -- is determined. In this way the upscaled grid system is perturbed and this modified system is flow simulated. The flow simulation-derived change in the objective function (energy) value is thus determined on the upscaled grid system. The approach is thus a 2-scale one.
5. The composite (combined variogram and flow simulation) change in the objective function is determined. Again the Metropolis condition test is used and if this test is passed then the distribution is modified by updating with the new permeability value at the perturbed fine scale location. The energy (objective function) value is also updated.
6. The convergence tests are conducted. If not satisfied, another location is selected and we loop again.
3.3.4 Upscaling

An integral part of the SA approach is the use of an upscaled grid for the flow simulation part of the algorithm. Also, we focus on permeability upscaling since the porosity variations have little effect on primary production. We propose two modified upscaling approaches -- one from the published literature and one we empirically developed -- which perform significantly better. Other upscaling technique can be found elsewhere.19

3.3.4.1 Modified Geometric Averaging

*Development of Procedure.* The major errors when conventional approaches, such as Renormalization, Geometric Averaging, Harmonic Averaging, or Tensor Method, were used for upscaling, were obtained at early time, corresponding to the time at which the pressure transient is influenced by the near-well region,20,21 and, as was observed during the development of this study, dominated by the wellblock permeability. It seemed reasonable therefore to concentrate on improving the upscaling approach for this region to be able to better match the pressure behavior at early time. Further to this, the upscaling procedure should favor the fine scale wellblock permeability value. By numerical experiments, we determined that an appropriate procedure was to assign a weighted arithmetic average of the fine scale wellblock permeability and the mean of the remaining fine scale values making up the coarse scale wellblock permeability value. Thus

\[
\bar{k} = w_1 \cdot k_{wb} + w_2 \cdot \frac{1}{n-1} \sum_{i \neq \text{wb}}^n k_i,
\]

where \( w_1 \) is the weighting applied to the fine scale wellblock permeability, \( k_{wb} \) is the fine scale wellblock permeability value, \( w_2 \) (equal to \( 1 - w_1 \)) is the weighting applied to the mean permeability of the other fine scale, near-well gridblocks which are included in the coarse scale wellblock and \( \frac{1}{n-1} \sum_{i \neq \text{wb}}^n k_i \) represents that mean where \( n \) is the upscaling ratio. An appropriate value for \( w_1 \) was empirically determined to be 0.9 (thus \( w_2 = 0.1 \)). The upscaling procedure therefore consists of using conventional geometric averaging to upscale all coarse scale blocks except the coarse scale wellblocks. For wellblocks, Equation (4) is used.

3.3.4.2 Ding's Approach22

This approach recognizes the need for a specific treatment of scale-up in the vicinity of wells due to the 'radial' flow pattern resulting from a high pressure gradient. This is in contrast to 'linear' flow, normally associated with low pressure gradients. He distinguishes between effective and equivalent permeability. The former is due to the spatial distribution or correlation of permeability values, while the latter is that required so
that simulations on a coarse grid are equivalent to those on a fine grid -- a purely numerical problem, which this scale-up approach addresses. In this approach the upscaled parameters consist of the transmissibilities and numerical productivity indices (PIs) around the well(s). The procedure may be outlined as follows:

- Upscale the entire grid via conventional geometric averaging.
- Determine the well locations.
- For the well locations, determine the upscaled transmissibilities for the 4 (2-D) near-well interfaces, $T_i$ to $T_4$.
- Develop a square grid with $9n^2$ fine scale gridblocks with the well at the center, where $n^2$ is the upscaling factor. Find the part of the reservoir to which this grid belongs.
- Run a steady-state flow simulation on this $9n^2$ sub-grid. Save all the fine scale gridblock pressures and transmissibilities (in both the x- and y-directions).
- Calculate the fine scale flows, using $F_{eq} = T_{f} \cdot \Delta p_{f}$, where $F_{eq}$ is the equivalent fine scale flow between two fine scale blocks, $T_{f}$ is the fine scale transmissibility between those blocks and $\Delta p_{f}$ is the gridblock pressure differential between the same blocks. Then calculate the equivalent coarse scale flows by using $F_{eq} = \sum F_{eq}$, where $F_{eq}$ is the equivalent flow across the interface between two coarse scale blocks and the fine scale equivalent flows are those that constitute that coarse scale one.
- Calculate the coarse scale gridblock pressures and the coarse scale wellblock, using $p_{co} = \sum_{i} storage-weighted p_{f}$, where $p_{co}$ is the coarse scale gridblock pressure and the storage-weighted $p_{f}$ are the corresponding fine scale values.
- Calculate the equivalent coarse scale transmissibilities, $T_{1}$ to $T_4$, using $T_i = \left( \frac{F_{eq}}{\Delta p_{co}} \right)$, $i=1,...,4$ where $\Delta p_{co}$ represents the coarse scale gridblock pressure differential between the outer block and the wellblock.
- Finally, calculate the coarse scale numerical productivity index for all wells using $PI_{co} = \frac{q}{(p_{co} - p_{wf})}$, where $q$ is the flow rate, $p_{co}$ is the coarse scale wellblock pressure and $p_{wf}$ is the flowing bottomhole pressure (BHP).

For the flow simulation at the coarse scale, calculate the coarse scale transmissibilities as usual, except for the near-well blocks; in these cases we use $T_1$ to $T_4$ as calculated above for each well. We then use the numerical $PI_{co}$ to calculate the BHPs from the gridblock pressures (instead of using the Peaceman correction).
Because the procedure requires the numerical solution of a steady-state problem using a fine scale sub-grid around each well, the solution is somewhat more complex and time-consuming than the modified geometric average upscaling method; however, it consistently performs better than that approach for upscaling.

3.4 Result

Three synthetic 100x100-gridblock datasets were generated. We present the results obtained using these datasets for evaluating the performance of the proposed composite objective function SA algorithm. The approach used in evaluating the results is to compare the ‘truth case’ description to that obtained by our algorithm and also to that obtained when a variogram-only objective function is used. Note that in testing our modified approach, we use both upscaling approaches (modified geometric averaging and ‘Ding’\textsuperscript{22}) in the runs for the first two synthetic images. Note also that the same variogram models are used for the composite objective function SA run (our modified SA algorithm) and for the variogram-only objective function SA run. Hence the only difference between these SA algorithms is that the flow simulation constraint is included in our modified approach. The comparisons made are:

- Visual comparisons of the images.
- Comparison of the dynamic behavior. This is accomplished by flow simulating all descriptions using ECLIPSE-100\textsuperscript{10} and calculating the relative (percentage) errors in the flowing BHPs using the truth case flow simulation results as the standard or basis of comparison.

The purpose, of course, is to validate that

- the realizations generated by the modified SA algorithm more closely ‘resemble’ the truth case image than that generated by the variogram-only objective function, and that
- the dynamic behavior of the distribution obtained from the modified approach also matches that of the truth case more closely than that of the variogram-only objective function SA run.

It should be pointed out that in the modified SA run, a flow simulation time range of 1-100 days was used in all cases. However, in these comparisons, we flow simulate to 150 days, i.e., we are no longer using history matching but are in fact observing how well the future performance of the results of both descriptions match that of the truth case.

3.4.1 Dataset #1

3.4.1.1 Visual Comparisons for Dataset #1

Figure 14 presents a visual comparison of the results. It shows 4 maps: the first, (a), is the truth case, the second, (b), is the modified SA run in which modified geometric averaging upscaling is used to create the upscaled (flow simulation) grid, the third, (c), is
the modified SA run in which Ding's upscaling approach is used and the fourth, (d), is the result for the variogram-only objective function SA run. It can be seen that the major features of the truth case description are captured by both Figures 14(b) and (c), while Figure 14(d) does a relatively poor job.
Figure 14: (a) Truth Case (b) Modified SA Run Results - 1
(c) Modified SA Run Results - 2 (d) Variogram-Only SA Run Results for Dataset #1
3.4.1.2 Flow Performance Comparisons for Dataset #1

Composite Objective Function vs Variogram-Only Objective Function. For this first dataset, nine wells were used for flow simulation. As stated above, all grids were flow simulated under the same set of conditions and the flowing BHPs compared. Figures 15(a)-(b) compare the errors in these pressures (actually in the change in the flowing BHPs) as a function of time for the variogram-only objective function SA run and that of the composite objective function SA run. In these plots we have used the geometric averaging approach. The relative errors are calculated as given in Equation 1 below:

\[
\text{Relative Error} = 100 \times \frac{\text{"Truth Case" } \Delta p_{wf} - \text{ SA Run Result } \Delta p_{wf}}{\text{"Truth Case" } \Delta p_{wf}}.
\] (1)

Other wells show a similar behavior. These plots show that the errors are significantly smaller when the modified approach is used, even in the predictive part of the time scale (>100 days). These plots further illustrate that the upscaling approaches used are adequate.

![Figure 15: Pressure Error Comparisons Between the Modified SA Approach and the Variogram-Only SA Approach for Dataset #1](image)

(a) (b)

Figure 15: Pressure Error Comparisons Between the Modified SA Approach and the Variogram-Only SA Approach for Dataset #1
Comparison of Upscaling Approaches. Figures 16(a)-(b) give the results of the analyses conducted to ascertain whether one upscaling approach was better than the other. It can be seen from these plots that the methods performed about the same, thus validating the use of either. Figure 17 shows that the maximum absolute relative errors from the variogram-only objective function results are larger than those of the results from the modified approaches. Figure 17 also shows that when the maximum absolute relative errors are compared, the two approaches appear to be on par, and in addition, these results are at least an order of magnitude better than those of the variogram-only results.

![Graphs showing pressure error comparisons between modified geometric averaging and 'Ding' upscaling approaches for Dataset #1](image)

(a) (b)

Figure 16: Pressure Error Comparisons Between the Modified Geometric Averaging and 'Ding' Upscaling Approaches for Dataset #1
Figure 17: Comparisons of Maximum Absolute Relative Errors from Variogram-Only Objective Function and the Composite Objective Function Using the Two Upscaling Approaches For Dataset #1

3.4.2 Dataset #2

3.4.2.1 Visual Comparisons for Dataset #2

Figure 18 below shows the true image for dataset #2 in (a), the SA composite objective function run results using modified geometric averaging upscaling and Ding upscaling in (b) and (c) respectively, and the SA variogram-only objective function run result in (d). As for dataset #1, it can be seen that the composite objective function SA runs results match the true image much better than that of the variogram-only SA run. Although not shown, the bottomhole pressure comparisons are very similar to Dataset #1.
Figure 18: (a) Truth Case (b) Modified SA Run Results - 1

(c) Modified SA Run Results - 2 (d) Variogram-Only SA Run Results for Dataset #2
3.4.3 Dataset #3

3.4.3.1 Visual Comparisons for Dataset #3

Having verified that the two upscaling approaches are equally good, we focussed on a comparison of the modified SA algorithm (in which upscaling is performed via modified geometric averaging) and the variogram-only algorithm for dataset #3. For this case also it can be seen that the composite objective function results are better in matching the true image visually (Figure 19).
Figure 19: (a) Truth Case (b) Modified SA Run Results - 1
(c) Variogram-Only SA Run Results for Dataset #3
3.4.4 Sensitivities on Variogram Component

Previously the variogram models were based on the exhaustive dataset. Cases were run in which we attempted to model the variogram using the conditioning data (9 point values of permeability) only. This sparsity of data resulted in a variogram model which is very approximate and may even be inaccurate. Also while an exhaustive dataset allows anisotropy modeling, such a meager dataset is hard-pressed to give even an isotropic model. Figure 20(b) shows the resulting image obtained for the modified SA approach. It is obvious from these results that although there was insufficient information, we still get a reasonable image of the truth case. However, as shown in Figure 20(c), the results for a variogram-only SA run -- using a conditioning data-based (isotropic) variogram model -- are unable to capture the image of the truth case. This last result is dramatic proof of the utility of the flow simulation constraint in the SA process.
Figure 20: Results from Sensitivity Tests on Variogram Component
3.5 Conclusions

The results presented in this report suggest that the inclusion of dynamic constraints significantly improves the performance of SA in matching the reservoir description as well as the dynamic behavior of the reservoir, even with respect to its future behavior. It should be noted, however, that the location of the wellblocks plays an important role in these results. For example, if the wells are located in the relatively low-permeability areas, the matches obtained are not as good as when the wells were situated in the higher-permeability locations. One possible explanation for this can be related to the limitations of the upscaling methods used. When there is significant contrast between the wellblock permeability and the surrounding permeability values, these methods do not work well. Another explanation may be that, as Vasco et al. concluded, transient pressure data seem to be most sensitive to structure in the immediate vicinity of the wells and do not seem to have as much resolving power as, e.g., tracer data. Thus it would be difficult for the wells in the low-permeability areas to resolve structural features of the high-permeability areas. Note however that our pressure data can include post-transient information, and so we should expect to capture some features which may be relatively remote from the wells, but which influence their pressure behavior at later times. In addition, it may be argued that from a practical standpoint we are primarily interested in the regions of higher potential in the reservoir, i.e., the higher permeability regions. The question arises therefore as to how do we obtain beforehand, knowledge on where these areas are. One source may be seismic information from which we may have, in a general sense, a feel for the 'lay of the land'. Another source of information may be existing well data. In this case we would consider only those data which satisfy the specified constraints.

3.5.1 Effectiveness of Upscaling Approaches Used

3.5.1.1 Modified Geometric Averaging Approach

This procedure has been shown to perform well in upscaling the reservoir description for flow simulation purposes. The flowing BHPs obtained at the coarse scale agree with those at the fine scale within about ±10% even at very early times (<0.1 day) and better than ±5% after 1 day flowing time. Thus this approach may be used in a 2-scale SA approach with reasonable accuracy if pressure data after about 1 day flowing time is used. There is however, a limitation on the degree of contrast which may be accommodated by this approach.

3.5.1.2 Ding's Approach

As in the case of the modified geometric averaging approach, this methodology was found to perform quite well in upscaling the reservoir description for flow simulation, especially after a flowing time of 1 day. The algorithm however is somewhat more complex than that for the modified geometric averaging approach, and thus is slightly more computationally-intensive. For this reason we opted for the modified geometric averaging approach.
3.5.1.3 Effectiveness of the LTFD Approach

This algorithm is theoretically appealing in the sense that it suggests an approach for flow simulation which can efficiently timestep to given times of interest. However, there are severe limitations to the current implementation of this approach. The major ones were related to the algorithm's limitation to single-phase flow and the trend towards matrix singularity at longer flow periods, especially for large grids.

3.5.1.4 Effectiveness of the Modified SA Algorithm

The variogram-only SA algorithm for a grid size of 100x100 cells may be run in about 10-15 minutes while, as we have seen, the composite objective function run takes on the order of about one day. This difference in run times may be acceptable only if the results obtained via the more lengthy run are much better than what is obtained from the variogram-only run. The results presented support this claim. Nevertheless, we perceive this work as a beginning. There is much improvement possible by refinements in the methodology. However, the conceptual framework has been developed and has also been shown to be a viable strategy. There may be tremendous benefits to be derived from a hybrid approach which considers this approach and, for example, a parallelized implementation.
4. Testing the Geological System Components

We have completed reasonable testing of the subsystems comprising the geological system. This testing resulted in the fine tuning of the subsystems. The results are promising and we plan to continue the fine-tuning as integration of the subsystems continues and as we build the 3-D model of the information. We begin the next section by providing updates to the subsystem. The bulk of the section presents the testing of the results against two geology expert interpretations.

4.1 Updating the Well Log Segmentation Process

Previously, we discussed the filtering approach to detecting the cuts in the gamma ray logs. Figure 21 shows a basic result of the approach. Figure 22 shows the geologist interpretation of the same log showing that the filtering approach for this log detected a high percentage of the correct cuts.

![Figure 21: Results obtained from previous filtering approach](image-url)
With the additional testing, it was determined that the approach used in some logs can leave big gaps without a cut. Therefore, a new rule had to be included in the filtering algorithm. This new rule calculates the shape distance between the cuts. If a distance is longer than a prefixed valued, it is possible that a cut is missed. Therefore, all the rules used to find cuts are applied again over this gap, but using the unfiltered log. In Figure 23 is shown an well log with a big gap between depth 1454 and 1486, thus the new rule determines any cuts could be missed. Figures 21 and 22 show a missing cut at depth 1478. In the well log shows in Figure 24 the new rule was applied and the missing cut was found.
Figure 23: Results obtained without gap detection.

Figure 24: Results obtained with gap detection.
4.2 Updating the 2-D Correlation Process

In this section, we describe the expert system that performs the correlation given the matrix values obtained by the previous ranking approach to the correlation of log curves. A rule set, described in the March 1996 report has been used to generate a matrix of compatibilities of zones in wells. In addition, we introduce a visualization module to depict and manipulate the system’s correlations.

4.2.1 The Expert System for Correlation

The system for correlating and zones in wells and visualizing them can be considered as three modules each performing distinct tasks. They are the correlation rank matrix module, matrix analysis module and the visualization module as depicted in Figure 25.

![Figure 25](image)

**Figure 25** The system model for correlation of wells

**Correlation rank matrix module.** This module uses a set of rules based on similarities in well log trace shapes, thickness and vertical position of zones. The segmentation of well logs and the log-facies identification by the neural network and depths of identified marker beds will be given as input. The matrix obtained from this module is used as input for the module analyzing the matrix.

**Matrix analysis module.** Using the matrix obtained from the correlation rank matrix module, this module reports the correlation of the zones. A set of rules are used to analyze the ranks in the matrix to decide which zones correlate and which ones merge. These rules are given below.

1. Find the row and column maximums for each of the rows and columns in the matrix.
2. If a particular entry in the matrix is a row and column maximum, then the zones corresponding to that entry in the matrix are correlated. This is done since if two zones being compared have the highest rank both on the row and column, then there is a obvious correlation between them.
3. If there is a row/column maximum and if the respective column/row corresponding to the maximums is not correlated then the zones are correlated. This allows the freedom of correlating rows and columns in which there is
either a row or column maximum but the corresponding entry may not be the maximum by a couple of ranks.

4. For each of the remaining rows/columns that have not been correlated, going down from the first maximum to each of the successive lower maximums, it is checked to see if any of the adjacent rows or columns have been correlated. If they have then the corresponding zones are merged and the correlation extended to cover that zone. Since the neural network comes up a varying number of cuts for the two wells being correlated, there is a likelihood that zones merge before they can be correlated.

5. A rule which checks if an exterior row/column has been correlated and if it has does not permit that correlation with an interior row/column. This is done to limit the number of cross-overs at the end, and to allow the fact that there may be some zones that won't be correlated at the very end probably because of one of the logs being longer than the other.

6. Detection and removal of cross-over: A cross-over occurs when the correlation of the zones intersect.

The above steps are repeated until all the zones have been correlated with no cross-over. Initially all the cross-overs were detected and removed and only then were the zones recorrelated. This led to the problem that several zones would remain uncorrelated. To remove this problem, cross-overs were detected one at a time and zones recorrelated after every removal. Another problem was that the system was picking certain cross-overs which were in fact zones merging with others. To handle this problem, rules were added to check for merging before testing for cross-overs. Finally, there may still be zones in which have not been correlated which probably occurred because of a shorter log data which will have to be resolved by the user during visualization of the correlation.

4.2.2 Visualization module.

To allow the user the freedom to modify the correlation obtained by the system, a 2-dimensional view of the gamma-ray logs of two wells which are being correlated is plotted against depth and is displayed in a window. In order to do that, a window was created and the log values were rescaled so that they fit the window dimensions. This is done by finding the minimum depth and gamma ray values for the log, assigning them minimum X and Y axis values for the window and for the remaining values of the log the window values are calculated relative to the minimum values. The correlation of the zones is displayed by drawing lines between the corresponding depths again taking into account the rescaling. The user can use the mouse to click on the end-points of the line to delete and add lines where they think there should be a correlation. The depth values of the logs are displayed along the y-axis.

A problem was that to delete inappropriate correlation, the user had to click exactly on the pixel corresponding to the end point of the line for the system to recognize the line chosen. Clicking on a pixel in the vicinity was not sufficient. To provide a better interface, a rectangular region was defined around the end points and drawn. So the user
now has to click anywhere inside that rectangle for that correlation to be deleted. A print option is provided in the form of a button which allows the user to view the hard copy as well. Also, once the user has decided on a correlation, the data is written back to a file with the changes, since some of the correlation between the zones may have been deleted or modified.

4.3 Comparison of the system with geology experts

4.3.1 Cut comparison

The cut recognition of the system has been compared with the cuts identified by the geologists. They are compared on the following basis: (1) the number of cuts identified by the system that correspond exactly with the cuts identified by the geologist is counted, (2) the number of cuts that the geologist has marked but which the system did not identify is noted, (3) the number of additional cuts added by the system is also noted as it will affect the correlation of well logs, and (4) the percentage is calculated based on the number of cuts identified correctly by the system to the total number of cuts identified by the geologist. The results appear in Table 3.
<table>
<thead>
<tr>
<th>Well name</th>
<th>Compared to Geologist 1</th>
<th></th>
<th>Compared to geologist 2</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>correct</td>
<td>Missed</td>
<td>Added</td>
<td>%</td>
</tr>
<tr>
<td>Well 11-75</td>
<td>8</td>
<td>5</td>
<td>1</td>
<td>62%</td>
</tr>
<tr>
<td>Well 11-86</td>
<td>8</td>
<td>4</td>
<td>2</td>
<td>67%</td>
</tr>
<tr>
<td>Well 11-89</td>
<td>12</td>
<td>4</td>
<td>3</td>
<td>75%</td>
</tr>
<tr>
<td>Well 679</td>
<td>13</td>
<td>3</td>
<td>1</td>
<td>81%</td>
</tr>
<tr>
<td>Well 672</td>
<td>9</td>
<td>2</td>
<td>0</td>
<td>82%</td>
</tr>
<tr>
<td>Well 1051</td>
<td>10</td>
<td>5</td>
<td>2</td>
<td>67%</td>
</tr>
<tr>
<td>Self 82</td>
<td>8</td>
<td>1</td>
<td>3</td>
<td>89%</td>
</tr>
<tr>
<td>Self 78</td>
<td>8</td>
<td>6</td>
<td>0</td>
<td>57%</td>
</tr>
<tr>
<td>Bg18-32</td>
<td>8</td>
<td>6</td>
<td>1</td>
<td>57%</td>
</tr>
<tr>
<td>Bg18-33</td>
<td>5</td>
<td>4</td>
<td>3</td>
<td>55%</td>
</tr>
<tr>
<td>Self 81</td>
<td>8</td>
<td>2</td>
<td>0</td>
<td>80%</td>
</tr>
<tr>
<td>Well 11-88</td>
<td>12</td>
<td>4</td>
<td>1</td>
<td>75%</td>
</tr>
</tbody>
</table>

Table 3: The comparison of cut recognition

On average, the number of cuts correctly identified is about 74%. the number of cuts missed is approximately 26% and the number of cuts added by the system is about 14%. The best recognition of cuts is for Self 81 with 1 missed cut and 1 added. The worst is Bg 18-33 with 4 cuts missed out of 9. The best and worst recognition of cuts based on the number missed is shown in Figure 26.
Figure 26: Comparison of Cuts
4.3.2 Comparison of the facies

To analyze the facies recognition of the neural network system, the facies identified by the network was compared to the facies identified by the geologists considering the zones identified by the system, as this would allow a direct comparison. The results are presented in Table 4.

<table>
<thead>
<tr>
<th>Well name</th>
<th>Compared to Geologist 1</th>
<th>Compared to geologist 2</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Correct facies</td>
<td>Percentage</td>
</tr>
<tr>
<td>Well 11-75</td>
<td>4/7</td>
<td>57%</td>
</tr>
<tr>
<td>Well 11-86</td>
<td>3/8</td>
<td>38%</td>
</tr>
<tr>
<td>Well 11-89</td>
<td>5/12</td>
<td>41%</td>
</tr>
<tr>
<td>Well 679</td>
<td>9/13</td>
<td>69%</td>
</tr>
<tr>
<td>Well 672</td>
<td>3/8</td>
<td>38%</td>
</tr>
<tr>
<td>Well 1051</td>
<td>7/12</td>
<td>58%</td>
</tr>
<tr>
<td>Self 82</td>
<td>7/9</td>
<td>78%</td>
</tr>
<tr>
<td>Self 78</td>
<td>4/7</td>
<td>57%</td>
</tr>
<tr>
<td>Bg18-32</td>
<td>3/7</td>
<td>43%</td>
</tr>
<tr>
<td>Bg18-33</td>
<td>2/6</td>
<td>33%</td>
</tr>
<tr>
<td>Self 81</td>
<td>5/7</td>
<td>71%</td>
</tr>
<tr>
<td>Well 11-88</td>
<td>6/12</td>
<td>50%</td>
</tr>
</tbody>
</table>

Table 4: The facies recognition comparison

As with the cut comparison, the best and worst facies recognition is shown in Figure 27. The average recognition of the facies is about 58%.
Figure 27: Comparison of Facies
4.3.3 Correlation Comparison

To evaluate the performance of the system’s correlation, the system’s correlation was presented to a geologist who classified the correlation of zones under 3 categories: (1) A good correlation: A correlation which agrees with the expert’s correlation of the wells, (2) An acceptable correlation: A correlation which does not exactly match with the expert but which is a reasonable correlation, (3) A bad correlation: A correlation which cannot be justified by the expert. Based on the above 3 categories, the system’s correlation was analyzed and the results are presented in Table 5.

<table>
<thead>
<tr>
<th>Wells being correlated</th>
<th>Geologist 1</th>
<th></th>
<th></th>
<th></th>
<th>Geologist 2</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Total</td>
<td>Good</td>
<td>Acceptable</td>
<td>Bad</td>
<td>Total</td>
<td>Good</td>
<td>Acceptable</td>
<td>Bad</td>
</tr>
<tr>
<td>1175 vs. 1188</td>
<td>7</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>7</td>
<td>6</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1186 vs. 1189</td>
<td>5</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>5</td>
<td>3</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>1175 vs. 1186</td>
<td>5</td>
<td>4</td>
<td>1</td>
<td>0</td>
<td>5</td>
<td>5</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>self 81 vs. self 78</td>
<td>4</td>
<td>3</td>
<td>0</td>
<td>1</td>
<td>4</td>
<td>2</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>self 81 vs. self 82</td>
<td>6</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>6</td>
<td>5</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>672 vs. 1051</td>
<td>5</td>
<td>0</td>
<td>2</td>
<td>3</td>
<td>5</td>
<td>5</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1833 vs. 1832</td>
<td>6</td>
<td>2</td>
<td>0</td>
<td>4</td>
<td>6</td>
<td>5</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>679 vs. 672</td>
<td>4</td>
<td>0</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>3</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Total</td>
<td>42</td>
<td>14</td>
<td>12</td>
<td>16</td>
<td>42</td>
<td>34</td>
<td>5</td>
<td>3</td>
</tr>
</tbody>
</table>

Table 5: Comparison of Correlation

On an average the system comes up with 78% of good and acceptable correlation. One problem that needs to be modified is the merging of zones. Since the number of zones for one of the wells may be considerably greater than the other, the zones have to be merged while correlating. Incorrect merging is the cause of some problems when the correlation is bad. Also with the input to the system is from other modules which perform automated recognition of marker bed, cuts and facies which may not be the most suitable choices, the correlation may be bad because of such inconsistencies. Figures 28 and 29 present good and bad correlations performed by the expert system
Figure 28: A good correlation by the system: Wells 11-75 and 11-86
Figure 29: A bad correlation by the system: Wells 672 and 1051
Though the 2-D results were promising, when extended to a allow for 3-D characterization of the reservoir, some changes were needed in the correlation algorithm. We present these updates and there associated results in the next section.

4.4 Changing the Correlation to accommodate 3-D Information and Display

"3-D" correlation (correlation among multiple wells) extends correlation between two wells. The technique for correlating two wells has been described in detail in the previous reports. 3-D correlation implies that a geologically reasonable correlation framework be established to correlate wells, so that the correlation is not only corresponds between two individual wells but also matches the general geological trends for the location.

The main problem faced in extending the correlation between two wells to "3-D" correlation, is that the pairwise correlation results between wells A and B may not be consistent with the correlation results between wells B and C. As the geological condition gets more complicated, the inconsistencies are likely to be prominent. There has been no published methodologies dealing with automated correlation of multiple wells.

As a starting point to overcome the challenge, a relatively simple algorithm for "3-D" correlation is proposed and henceforth called "proximity-based algorithm". This algorithm is based on the concept that proximity is first control for continuity and/or similarity of geological features. In general if well B is closer to well A than well C to well A, then well A is assumed to be more similar to B rather than C. This implies that the best correlation is obtained by correlating a well with its nearest well.

4.4.1 Description of the proximity-based algorithm:

The steps involved in the process are discussed below:

1. Select reference well/wells for which all necessary information is provided. (user input)
2. Set a matrix which displays the distance between the uncorrelated wells (columns in the matrix) and the correlated wells (rows of the matrix). This matrix henceforth called as the correlation distance matrix, consists of the distances between the reference wells and the other wells in the field.
3. Correlate reference well with its closest uncorrelated well.
4. Update the correlation distance matrix to include the newly correlated well and remove it from the columns representing the uncorrelated wells.
5. Choose an uncorrelated well which has the least distance to any of the correlated wells and correlate that pair of wells.
6. Repeat steps 4 and 5 until all wells have been correlated.

For example, in Figure 30 a map of the well locations in the Glenpool field is given. Well 11-86 and Self 82 have been chosen the reference wells. Well 11-75 and well
11-86 are correlated first (least distance). The next pair would be self 82 and self 81. The third correlation would be between Well 11-88 and well 11-75. This process is continued until all the wells have been correlated.

Another approach that is being considered is an algorithm of correlating each well with the reference well. This produces different results from the algorithm described above. This is natural since the wells further away when being correlated do not produce as distinct demarcations as the wells that are closer. So the zones in the wells tend to merge.

The correlation tests were carried out for both the algorithms. Well 11-86 was taken as reference well. It was correlated with well 11-75 and then well 11-1 with Well 11-88. This produces a different result than when well 11-86 is directly correlated with Well 11-88 and well 11-75. The results are shown in Figure 31 (proximity based algorithm) and Figure 32 (correlation with reference well).

The proximity based algorithm stresses on the proximity factor which makes geological sense. Both algorithms start from a reference well and expand in different directions. This meets a geologist's appreciation of overall reasonability. For the proximity based algorithm, there is no need to change the algorithm for multiple reference wells. For the second approach however contradictions may arise when multiple reference wells are present. Therefore the area may have to be divided, such that areas in a specific compartment are correlated with the reference well in its compartment.

One of the key factors that played a part in the correlation is the sensitivity of the rule set to the distance-dip angle factor. If this factor is large, the correlation matrix produces high ranks for many zones, and if small, the ranks are low across several zones. This has to be studied to make the effect of this factor consistent.

The rules used to correlate two wells may have to be modified to reduce the merging of zones of the reference well. This arises because in the "2-D" correlation, neither of the wells were considered to be a reference well. If a cut is missed and therefore zones merged, then at each successive phase the number may become progressively smaller which has to be avoided. Once a well has been correlated, it acts like a reference well. Since the zones obtained after correlation may have changed, the facies for the zones have to be identified once again.

4.4.2 Future Work:

The above framework using either of the two algorithms, is the first step in the overall "3-D" correlation process. Once the zones have been identified, each zone has to be considered individually to identify if it is a reasonable correlation in the geological sense. The facies geometry has to be identified across the area. If the correlation turns out to be unreasonable based on a set of factors, backtracking may have to be incorporated to make the correlation reasonable.

Directional heterogeneity which may exist in many geological situations has to be considered. Whenever directional heterogeneity occurs, it would be more desirable to correlate wells which are aligned parallel to the direction of minimum variation first, even though they are not necessarily closest to each other.
Figure 31
4.5 Geological Systems Component: Facies Geometry Database

The facies geometry database is the first step in developing rules for predicting the three-dimensional geometry of facies given the known thickness from well log analysis. The following is based on a number of empirical studies that are directed relating thickness to other geometrical dimensions for meandering channels, crevasse splays, and fluvial dominated deltas.

4.5.1 Meandering Channels

There are several formulas that relate width to depth of a meandering channel bed. The following section will highlight just a few.

One of the more commonly used formulas is from M. R. Leeder (1973). Leeder’s formula is \( w = 6.8h^{1.54} \). The standard deviation Leeder gives 0.35 log units. This formula is widely used (Bridge and Mackey 1993, Lorenz et al. 1985, Stalcuppe and Adams 1986, Collinson 1978, and Williams 1988), but it does have one limitation. It is only for streams with sinuosity (channel thalweg length/valley length) greater than 1.7.

S.A. Schumm (1963, 1978) also determined a formula that relates width and depth of stream channels. This formula involves another parameter, the silt-clay content. Schumm’s equation is \( F = 225 M^{-1.08} \). F is the width-depth ratio and M is the weighted mean percent silt-clay content. Schumm doesn’t give an error for his formula but Cotter (1971) uses a standard error of 0.20 for Schumm’s equation. In Schumm’s 1978 article meander wavelength is also related to F by the equation \( l = 18 \left( F^{0.53} \times w^{0.69} \right) \) or in log form \( \log l = 1.27809 + 0.52822 \log F + 0.68774 \log w \). The standard error is 0.21 log units, \( l = \) meander wavelength and \( w = \) bankfull width.

Garnett P. Williams offers another equation relating width and depth (1986). For streams with sinuosity greater than 1.7 he found with his data the relation to be \( w = 15.5d^{1.4} \) (30 sites). Using all his data regardless of sinuosity he found \( w = 21.3d^{1.45} \) (67 sites). In these equations \( w = \) bankfull width and \( d = \) bankfull depth.

A different approach was taken by Fielding and Crane (1987) to determine the geometry of fluvial channels. First they determine the width of a channel by using an equation suitable for the depositional environment. For meandering streams they suggest using Collinson’s equation of channel belt width = 64.6 (channel depth)\(^{1.54})\). The best fit line for the data that Fielding and Crane used gives an equation of channel belt width = 12.1 (channel depth)\(^{1.85}). They suggest using this equation when unsure of the depositional setting of a sand body. Next they compute a cross sectional area for all sandstones that will be intersected by a well, excluding any that are less than 2 meters thick or that would be unsuitable reservoirs. cross sectional area = \( \sum_{i=1}^{n} t_i W_i \), this is for a well that intersects \( n \) channel belt sandstones, their individual widths (\( w \)) are computed and \( t \) is the observed thickness. There is also an equation for a success rate which “expressed as a percentage, is defined as the proportion of the total channel belt
sandstone reservoir volume in a given interval which will be penetrated by planned development wells a certain distance apart (p.324)."

\[
\text{success rate} = \frac{\sum_{i=1}^{n} t_i \text{minimum} (w_i, W)}{\sum_{i=1}^{n} t_i W}.
\]

\(W\) is the proposed spacing between wells. For secondary or tertiary development the success rate equation was rewritten as

\[
\text{success rate} = \frac{\sum (w_i \geq W) t_i W}{\sum t_i W}.
\]

A simple relationship is also found between channel width and point bars deposits. A common estimate is the two-thirds rule, the width of the point bar deposit is about two-thirds the width of the channel. This can be beneficial if an outcrop is available. It gives a rough estimate (Cotter, 1971)\(^39\).

Using 20 data points from Williams (1986)\(^40\), Puigdefabregas (1973)\(^42\), Schumm (1978)\(^38\) a graph was made plotting width and depth of meandering streams (Figure 33). the relationship it produced is

\[
w = 19.735d^{1.3572}.
\]

This is in the same order of magnitude as the other equations.

4.5.2 Crevasse Splays

Crevasse splays are overbank deposits that occur when a stream is flooded and a levee is breached. A crevasse channel forms and sand and silt is deposited in a fan or lobate shape. Deposits can range from a few centimeters in thickness to several feet. Smith et al. in their 1989 article\(^43\) related stages of crevasse splays to geometry. Stage I
is an immature splay that is small and produces lense or wedge shaped deposits. Stage II forms less tabular deposits that are disconnected by channelization. The cross sections are irregular. Stage III produces linear and more isolated sand bodies as channelization is even more common. With each stage the size of the splay increases.

Few empirical formulas were offered in the literature that related the depth to width. R. Mjos et al. in their 1993 article\(^4\) that the maximum length to thickness ratio is 2000. Generally the width to thickness ratios are in the range of 150-1500 and thicknesses range 0.3 to 4.5m, but usually less than 2m. Crevasse channel sandstones have width/thickness ratios of 5-60, and thickness are usually less than 4m, but can range up to 7m. With data given in Rhee and Chough (1993)\(^5\), O'Brien and Wells (1986)\(^6\), Flores (1984)\(^7\), and Chrzastowski et al. (1994)\(^8\), a graph relating width to thickness came up with a slope of 426.87 (Figure 34). This fits in with Mjos et al. assumptions, but is not conclusive because of the few data points. The larger splays gave ratios that were closer to the published norms.

**Figure 34**

4.5.4 Fluvial Deltas

The literature seems to contain little information on the geometry of entire deltas. There is some research on deltaic channels and bar deposits. Lowry and Jacobson (1993)\(^9\) give an empirical formula for channel width to thickness ratio for isolated channels that incise mouth-bars. The relationship they give is \(w = 12.7t^{1.2}\). They also found distributary channel width to thickness ratios range from 10:1 to 20:1. Tyler et al. (1991)\(^10\) found that in the Ferron Sandstone basal channels have fairly consistent width to thickness ratios. The average is 9.5. These are small features at the base of the distributary complex. Falkner and Fielding (1993)\(^11\) found in their study area that proximal mouth bars have width to thickness ratios of 5 to 25.

From “The Atchafalaya river Delta Report 4, Generic Analysis of Delta Development” by Wells et al.\(^12\) data on the size of deltas was used. Maps of five deltas in different stages of their development were measured to find length perpendicular to the main river channel, width parallel to the main river channel, and the channel width of the main distributary channel to the delta (c). The length to width ratio was charted versus c
There seems to be two main groups in this data. The deltas where the l/w ratio is greater than 1 and the group where it is less than 1. The cause of this difference seems to rely on the size of the delta. The smaller two deltas have ratios greater than 1, and the other larger deltas have l/w ratios smaller than 1. For smaller deltas (less than 10 km in length) the relationship between c and l/w is $c = -0.0302l/w + 0.1995$ (Figure 36). The relationship for larger deltas (greater than 10 km in length) is $c = 0.3663l/w + 0.3458$ (Figure 37).

This relationship would be useful if the depth of the distributary channel can be determined. This could be accomplished by using the two-thirds rule or the ratios for crevasse splay channels. Deltas are for the most part crevasse splays, so that is reasonable.
4.5.5 Conclusion

Meandering streams offer the best opportunity for estimation of depth based on thickness. The relationship is well documented, and the relationships take into account other factors such as sinuosity. Crevasse splays and deltas pose more of a problem. The relationships between depth and width are inconclusive.

5. Well Model Identification System

The Well Model Identification System reported in the March 1996 quarterly report has integrated successfully all of the subcomponents which were encoded using both C++ and FORTRAN. The remaining parts to be developed within this system are:

- the incorporation and integration of a graphing facility to display the information
- the automatic transfer of data between the subsystems and the non-linear regression algorithm
- the integration of the well model identification system with the whole reservoir integration system.

The final task involves using information obtained from the geological interpretation system as input to the well model identification system. In addition, the output of the well model identification system will be used to with the results from the geostatistical system to determine the final reservoir description.

6. Integrating the Component Systems into a Single Reservoir Characterization System

To facilitate the construction of the reservoir characterization system, we decomposed the system into smaller parts as described in Section 2. This decomposition allowed us to apply multiple artificial intelligence techniques, such as expert systems and
neural networks, as well as utilize many numerical techniques. Because of the interdisciplinary nature of the project multiple languages were used in the development, along with multiple platforms. The language and platform decisions were also made to expedite development and testing. Because the target architecture is the PC, it was necessary to port all code from workstations for PCs.

We had chosen C++ as the final language in which all the systems would be encoded. However, some of the very computationally intensive code was written in FORTRAN. This gave us three choices in integration: (1) manually convert all of the code to C++, (2) use a conversion tool to convert all of the code to C++, or (3) find software that claims to integrate C++ and FORTRAN without any conversion. Manual conversion would be extremely difficult because there are thousands of lines of code in FORTRAN. Automatic conversion is available, but it produces poor quality and hard to read code, some of which will not compile. The final option seemed most feasible because of MicroSoft's claims that their Visual C++ and PowerStation FORTRAN were fully integratable. We discuss the integration of the whole system using this approach in the next section.
6.1 Integration of the Subsystems

Despite the fact that the code was written in both FORTRAN and C++, integrating the components of the geological, geostatistical, and well test systems should be achieved using the following approach.

1. Any needed modifications are done to the files to allow them to be used together.
2. A driver is created to call the individual systems.
3. The code is compiled and debugged until working object modules are produced.
4. A single executable file is created by linking the object files.

Due to problems with the compilers, libraries, and code itself, integration was not as easy as it should have been. In the above steps, modification needed to be made to avoid the errors associated with each step.

**Step 1.** Making the modifications to the files to allow compilation is not a complicated matter. Errors in this step are easily located and fixed. Errors that fall into this category are problems such as duplicate function names in the code of different components, error messages that terminate execution prematurely, and path names to data files that are not compatible between machines.

**Step 2.** Coding the driver is also a relatively simple step. Because some of the code is written in C++ and some in FORTRAN, a special declaration statement is needed for the C++ compiler to be able to recognize FORTRAN subroutines. If this statement is coded correctly, it will appear that the FORTRAN and C++ are seamlessly integrated.

**Step 3.** It is during compilation and linking that most of the problems with integration appear. Using Microsoft FORTRAN PowerStation and Microsoft Visual C++ with the Microsoft Developer Studio, the programmer can write code in either language, and during compilation, the Developer Studio will call the needed compiler. In some cases, however, using both compilers together can cause the FORTRAN object files to become corrupted. There can also be conflicts between the required FORTRAN libraries and the required C++ libraries resulting in a link error. These problems can be resolved by turning off the optimization on the FORTRAN compiler and removing the conflicting sections of the libraries from the C++ libraries.

Additional problems result from using the Developer Studio itself. The Developer Studio contains two platforms through which you can write and compile code. The Win 32 - Debug platform creates files that record execution and make debugging the code easier. These files must be run within the Developer Studio. The Win - 32 Release platform creates no such files. It simply creates the object files from the code. The libraries and programs used by the Debug platform can cause problems. For example, when executing C++ code that contains linked constructs and has been compiled under Debug, there can appear run-time memory access violations. The easiest solution to this problem, assuming that the code has been thoroughly tested to be correct, is to use the Release platform to compile C++ code containing linked constructs.
Step 4. Once the code has been successfully compiled and linked, a executable is created that can have problems at run-time. Any time a potential divide by zero is encountered in a piece of code, a run time math error is produced and the computer stops execution. The compiler does not necessarily locate this error in the code because the values of the divisors may not be set until execution. When the FORTRAN and C++ code are compiled and linked, a divide by zero error occurs during execution that does not happen when the pieces of code are compiled and run separately. Apparently, the C++ compiler can handle this problem and allows an executable to be created that prevents this run-time math error. The FORTRAN compiler, however, does not overlook the potential divide by zero. When executed together, the FORTRAN libraries linked to the code cause an error statement to be produced on the C++ code. The problem, though difficult to identify originally, can be easily fixed with a conditional statement in the C++ code wherever potential divide by zeros exist.

An unusual problem that had been encountered during integration also relates to the use of the Developer Studio. FORTRAN code that compiles, links, and executes fine under the Debug platform will not execute properly under the Release platform. Though much debugging has been done on the code, no improvements have been made. The code still fails to execute under Release. Because the C++ code contains linked structures and must use Release, it is important that this problem be solved before any integration can take place. At this time, the solution to this problem requires further research.

Most of the problems that have arisen thus far during integration have been solved. It is likely that more problems will be encountered as programming continues, but the experience gained thus far should make solving future problems easier.

### 6.2 Integrating Subsystems (ANSIM and COSIM)

This report describes the steps to be taken in integrating the lithofacies and petrophysical description with the dynamic data. COSIM (the program which simulates the lithofacies and petrophysical properties) described in the ‘Integrated Lithofacies and Petrophysical Properties Simulation’ section of this report. It is written in C++. All the algorithms described in ‘The Incorporation of Dynamic Constraints in Stochastic Conditional Simulation’ section have been coded into ANSIM. ANSIM is written in FORTRAN.

Even though the two codes are written in two different languages their integration should not be difficult. Problems arise however in the two compilers, when they are used together. A second more important problem arises from the FORTRAN code ANSIM itself. The environment chosen for integration is the Microsoft Developer Studio, which contains the Visual C++ (Version 4.0) compiler and the FORTRAN PowerStation compiler. It should be noted that during installation of the two compilers care should be taken in loading the two into the same directory. This helps the Microsoft Developer Studio to locate the compilers without any errors. In the case of the compilers being located in two different directories appropriate paths have to be specified during the setup of the Microsoft Developer Studio.
The overall procedure of the execution of the two codes is shown in Figure 38. The COSIM program provides grid block values as an output. These values are input into ANSIM program. The data were flow simulated and the simulated results are compared with the observed performance. If the match between the two is good, the program is terminated. Otherwise, the permeability values at individual grid blocks are perturbed by honoring the local relationship between porosity and permeability, and the procedure is repeated till a good match between the observed production data and the simulated production data.

The code ANSIM is written in FORTRAN Visual Workbench Version 1.0, in the Windows 3.11 environment. The first task accomplished in the integration effort was the successful linking and execution of ANSIM in Windows 95, in the Microsoft Developer Studio in the presence of both compilers. The next task was to run the code using a driver written in C++. For this purpose declaration statements had to be incorporated in the FORTRAN subroutines as and when incorrect file specification errors were reported during linking.

It was observed that parts of the FORTRAN code were getting corrupted when compiling the driver and the subroutines together on the same Build. Turning off optimization on the FORTRAN compiler avoided this corruption. Platform chosen for execution was Debug.

Although successful compilation of ANSIM was achieved, a run-time error was being generated upon execution of the program. The error F6700 - heap space exceeded, was generated after approximately twenty temperature levels of successful execution. A very important point to be noted here is that ANSIM was written in FORTRAN 77 while the run-time error of heap space violation is generated when dynamic allocation of memory has exceeded. This is a feature of FORTRAN 90. It was observed that the same run-time error was generated in the FORTRAN PowerStation without any C++ components. On comparison of the output generated for the first twenty temperature levels using both versions of ANSIM the two outputs were observed to be identical. Without any alterations in the input data files and without any C++ components the code was executed using the Microsoft Developer Studio Debugger. It was observed that the code was crashing only in one subroutine after execution for the first twenty temperature levels.

ANSIM was not executing after a certain point in the subroutine SANSP3.FOR. This subroutine contains a matrix solver, SLAP CG code incorporated, but not developed specifically for this code. SLAP CG template code was not tested independently before its inclusion into ANSIM. Part of the problem may be from this template code. But more significantly as the error generated is for FORTRAN 90 it is very much possible that the FORTRAN PowerStation compiler may have an intrinsic bug. This problem will be investigated further to find the optimal solution.
COSIM
Co-Simulation

Facies  Porosity

\ln K

\phi

Permeability

ANSIM
Dynamic Matching Using Simulated Annealing

Simulated Data

Observed Production

\text{Simulated data - Observed data} < \varepsilon

Yes

Output

No

Figure 38
This section presents the initial development of a graphical user interface. We are experimenting with the use of Microsoft Visual C++ interface capabilities, along with basic MS Windows programming. The graphical interface will allow users to handle different graphic formats to present data, and to process the data through the individual tools. Figure 39 shows how it looks in MS Windows95 (it also can be used on MS Windows NT 3.51). In Figure 39 also shows the current graphical formats that are allowed: well log representation (at right) and two dimension representations (at left).

![Figure 39: The application.](image-url)
Figure 40 shows the application allows one to load and save data in multiple formats. The current formats are:

- .log  only use for well log data.
- .2d   only use for 2d graphics.
- .txt  use for well log and 2d data but using text format.

Figure 41 shows the application and the well log graphic representation. On this representation is possible to use two tools. One of them is Set Cuts, used to identify cuts in the log. The other is Set Facies used for facies recognition. Figure 42 shows the same well log with cuts and identified facies.

Figure 40: The application and well log graphic with unidentified facies.
Figure 41: The application and well log graphic with unidentified facies.
Figure 42: The application and well log graphic with identified facies.
Two new representation formats have been included into the graphics interface. As the Figure 43 shows the first format is used to represent physical location of the oil well. On this representation the user can select a sequence of oil well (the red lines among the oil log) to obtain information about the correlation of their Gr. logs. The correlation of the well logs is the other representation added to interface. Figure 44 shows how it looks in the graphic interface.

Figure 43: The application and a physical location of the oil well.
Figure 44: The application and a well log correlation.
8. References

8. Yang, A.-P.: “Turning Bands Method to Generate 2-D Random Field with Autocorrelation,” University of Texas at Austin (April 1987)


44. Mjos, R. et al., 1993, “Crevasse Splay Sandstone Geometries in the Middle Jurassic Ravenscar Group of Yorkshire, UK” in Alluvial Sedimentation International Association of Sedimentologists Special Publication 17 ed. M.Marzo and C. Puigfabregas pp.167-184


53. From Greenbaum, A. et al., ‘Sparse Linear Algebra Package Version 2.0′, Lawrence Liverpool National Laboratory, Liverpool Computing Center (1986))