Report Title:	**Mapping of Reservoir Properties and Facies Through Integration of Static and Dynamic Data**
Report Type:	Annual Technical Report
Reporting Period Start Date:	October 1, 2000
Reporting Period End Date:	September 30, 2001
Principal Authors:	Dean S. Oliver, Albert C. Reynolds, Fengjun Zhang, Ruijian Li, Yafes Abacioglu & Yannong Dong
Date Report Issued:	December 2001
DOE Award Number:	DE-FC26-00BC15309

Petroleum Engineering Department
The University of Tulsa
600 South College Avenue
Tulsa, Oklahoma 74104

ABSTRACT

Knowledge of the distribution of permeability and porosity in a reservoir is necessary for the prediction of future oil production, estimation of the location of bypassed oil, and optimization of reservoir management. But while the volume of data that can potentially provide information on reservoir architecture and fluid distributions has increased enormously in the past decade, it is not yet possible to make use of all the available data in an integrated fashion. While it is relatively easy to generate plausible reservoir models that honor static data such as core, log, and seismic data, it is far more difficult to generate plausible reservoir models that honor dynamic data such as transient pressures, saturations, and flow rates. As a result, the uncertainty in reservoir properties is higher than it could be and reservoir management can not be optimized. The goal of this project is to develop computationally efficient automatic history matching techniques for generating geologically plausible reservoir models which honor both static and dynamic data. Solution of this problem is necessary for the quantification of uncertainty in future reservoir performance predictions and for the optimization of reservoir management.

Facies (defined here as regions of relatively uniform petrophysical properties) are common features of all reservoirs. Because the flow properties of the various facies can vary greatly, knowledge of the location of facies boundaries is of utmost importance for the prediction of reservoir performance and for the optimization of reservoir management. When the boundaries between facies are fairly well known, but flow properties are poorly known, the average properties for a facies can be determined using traditional techniques. Traditional history matching honors dynamic data by adjusting petrophysical properties in large areas, but in the process of adjusting the reservoir model ignores the static data and often results in implausible reservoir models. In general, boundary locations, average permeability and porosity, relative permeability curves, and local flow properties may all need to be adjusted to achieve a plausible reservoir model that honors all data. In this project, we will characterize the distribution of geologic facies as an indicator random field, making use of the tools of geostatistics as well as the tools of inverse and probability theory for data integration.

Contents

List of Figures

EXECUTIVE SUMMARY

Automatic history matching of multi-phase flow production data can be used to construct estimates or realizations of reservoir properties that are consistent with time-lapse seismic data, production data and static data obtained from logs, cores and geologic and geophysical interpretation. The automatic history matching procedure used here requires the minimization of an objective function which consists of the sum of a regularization term and production data mismatch terms. The regularization term represents a geostatistical model constructed from static data. A Levenberg-Marquardt algorithm is applied to minimize the appropriate objective function. Currently, we are able to estimate or construct realizations of the porosity, horizontal permeability and vertical permeability fields as well as estimates of the well skin factors. Here, the porosity, horizontal permeability and vertical permeability fields consist of gridblock permeabilities and porosities. In fact, we actually construct estimates or realizations of log-permeabilities and then convert these results to permeabilities. The reason for this is that the theoretical justification of our approach to automatic history matching is based on Bayesian statistics and in this setting the permeability fields are assumed to be log-normal.

As currently implemented, the Levenberg-Marquardt algorithm requires the calculation of individual sensitivity coefficients. Adjoint equations for three-dimensional, three-phase flow problems are developed and implemented to calculate the sensitivity of production data to permeability fields and well skin factors. Typically, the development of adjoint equations is tedious and lengthy, but with the formulation given here, the adjoint equations can easily be incorporated into a fully-implicit finite-difference simulator to obtain code for automatic history matching.

Procedures to ascertain the value of particular types of data (pressure, producing gas-oil ratio and water-oil ratio) for reducing the uncertainty in estimates of reservoir properties are applied and discussed.

To calculate all sensitivity coefficients required to history match a data set consisting of N_d data, requires the solution of N_d adjoint systems. If the N_d production data are

equally spaced in time, the solution of all the adjoint systems requires roughly the same computational time as $N_d/2$ forward simulation runs. It may be possible to reduce the computational time expended in solving the adjoint systems by a factor of five or more by viewing the total set of adjoint equations at each time step as one matrix problem with multiple right-hand sides (see Wu et al. (1999) and Killough et al. (1995)). We have not implemented such a scheme, because we currently believe that optimization schemes which require only the gradient of the objective function hold more promise. In any case, if N_d, is large, the generation of all sensitivity coefficients using the adjoint method may not be feasible. One possible solution is to search for solutions within a subspace of the model parameters. In Chapter 3 we describe a subspace approach that is fairly efficient and results in nearly the same solution as is obtained from the full parameter space.

Later in this report, we explore variable metric and preconditioned conjugate gradient methods for minimizing the appropriate objective function. These methods do not require the calculation of all sensitivity coefficients. They require only the gradient of the objective function, which can be calculated from a single adjoint solution. Thus, even if these algorithms require significantly more iterations to converge, they may still require a small fraction of the computer time required to obtain convergence with the Levenberg-Marquardt algorithm. Preconditioned conjugate gradient methods and some variable metric methods also require far less computer memory than the Levenberg-Marquardt algorithm. In this report, we explore the application of variable metric and conjugate gradient methods to history-match pressure data from a gas reservoir. We have formulated a limited memory Broyden-Fletcher-Goldfarb-Shannon (LBFGS) algorithm and a preconditioned conjugate gradient algorithm which we believe may be efficient for large scale optimization problems. For history matching data from a single-phase gas reservoir, the results suggest that these methods may be far more efficient than the Levenberg-Marquardt algorithm when the number of production data and the number of model parameters are both large. The LBFGS and preconditioned conjugate gradient methods, however, have not yet been implemented for history matching multiphase flow production data.

One of the primary motivations for the development of methods of history matching that can deal with large amounts of data and large models, is the need to incorporate time-lapse seismic data into reservoir characterization. In Chapter 4 we discuss some preliminary work on the problem of incorporating time-lapse seismic, and its sensitivity to uncertainty in extraneous parameters.

Chapter 1

INTRODUCTION

Automatic history matching is based on minimizing an objective function which includes a sum of production data mismatch terms squared. Typically, minimization is done based on a derivative based optimization routine, such as the Gauss-Newton and Levenberg-Marquardt algorithms, because algorithms which do not use derivative information converge too slowly for practical applications. We refer to the reservoir parameters to be estimated as model parameters. The model parameters typically represent reservoir simulator gridblock porosities and permeabilities (or log-permeabilities) but may also represent well skin factors, transmissibility multipliers, parameters describing relative permeability curves or parameters describing facies boundaries. Standard implementations of the Gauss-Newton method or Levenberg-Marquardt algorithm require calculation of sensitivity coefficients, which formally represent the derivative of predicted production data with respect to the model parameters.

For automatic history matching problems of interest to us, the number of model parameters is greater than the number of independent production data and thus the history matching problem does not have a unique solution. If the Gauss-Newton procedure is applied to minimize an objective function consisting of only the sum of squared production data misfit terms, the Hessian matrix will be singular and the optimization algorithm will be unstable. This instability problem can be avoided by adding a regularization term to the objective function to be minimized; see Tikhonov (1963) and Parker (1994). With a proper regularization, the Hessian matrix in the Gauss-Newton method will be real symmetric positive definite and hence nonsingular. In this work, we use a prior geostatistical model to provide regularization. With this approach, the history matching problem is equivalent to a Bayesian estimation problem (Gavalas et al., 1976; Tarantola, 1987; He et al., 1997; Wu et al., 1999).

The Gauss-Newton method is popular because it converges quadratically in the neigh-

borhood of a minimum; see, for example, Fletcher (1987). Sometimes, however, if the initial guess in the Gauss-Newton method results in a large initial data mismatch, the Gauss-Newton will converge to a reservoir model which represents a local minimum and does not give an acceptable match of production data; see, Wu et al. (1999). For this reason, we often apply a Levenberg-Marquardt algorithm instead of the Gauss-Newton method.

Automatic history matching traces its roots to research conducted in the 1960's by Jacquard (1964), Jacquard and Jain (1965) and Jahns (1966). To the best of our knowledge, Jacquard and Jain (1965) presented the first procedure for numerically computing sensitivity coefficients for history matching purposes. They applied their method to the estimation of permeability in a two-dimensional reservoir from pressure data obtained under single-phase flow conditions. They used a combination of zonation (less than twenty distinct values) and an algorithm conceptually similar to the Levenberg-Marquardt algorithm to provide regularization. Jahns estimated transmissibility (kh/μ) values and storativity ($\phi c_t h$) simultaneously by history matching single-phase flow pressure data. He used the finite difference method to compute sensitivity coefficients and applied the Gauss-Newton method with an exact line search to estimate the rock property fields by minimizing an objective function consisting only of the sum of squared pressure mismatch terms. Zonation was used to provide regularization. Jahns actually used a sequence of minimization steps where the number of zones, and hence the number of parameters was increased at each minimization step. The maximum number of parameters estimated was nine, eight zonal transmissibilities (or permeabilities) and total storativity. The finite difference method used to compute sensitivities requires $N_m + 1$ runs of the simulator where N_m is the number of model parameters estimated. This procedure would not be feasible when thousands of model parameters are estimated.

Jacquard and Jain (1965) based their procedure for computing sensitivity coefficients on an electric-circuit analogue. Later, motivated by Jacquard and Jain's ideas, Carter et al. (1974) presented an elegant derivation of a method to compute sensitivity coefficients for two-dimensional single-phase flow problems. As originally presented, the Carter et al. procedure can be applied to compute the sensitivity of simulator gridblock pressures to all gridblock permeabilities and porosities. If each well penetrates only a single gridblock, one can compute the sensitivity of the wellbore pressure to model parameters from the well's gridblock pressure sensitivities, provided the simulator uses a formula like the one of Peaceman (1978) to relate wellbore pressure and gridblock pressure. For two-dimensional single-phase flow problems with pressure measurements at N_w wells, this procedure requires $N_w + 1$ reservoir simulation runs to compute all sensitivity coefficients regardless of the number of model parameters and regardless of the number of pressure data. For three-dimensional problems, the number of

simulation runs required would be equal to one plus the number of gridblocks penetrated by wells. If the number of such gridblocks is large, the Carter et al. procedure becomes less computationally attractive. However, He et al. (1996) have developed an approximate three-dimensional version of the Carter et al. method which is computationally efficient. Regardless of the number of gridblocks penetrated by wells, the He et al. method requires only $N_w + 1$ reservoir simulation runs to compute the sensitivity of all well pressure data to all gridblock permeabilities and porosities. The method is only approximate and does not always yield accurate results if vertical flow is significant in gridblocks penetrated by wells. It is not clear, however, whether the fact that the sensitivity coefficients are approximate leads to significant errors in the history-matching process.

For nonlinear problems, e.g., multiphase flow problems, the derivations of Carter et al. (1974) and He et al. (1996) do not apply. Thus, we are forced to seek other alternatives. One possible choice is the adjoint or optimal control method, introduced independently for the single-phase history matching problem by Chen et al. (1974) and Chavent et al. (1975). (For single-phase flow problems, Carter et al. (1982) have shown that their method is equivalent to the adjoint method.)

Unlike the Carter et al. (1982) method, however, the adjoint method can be applied to compute sensitivity coefficients in multiphase flow problems. Unfortunately, the procedure requires N_d adjoint solutions where N_d is the number of production data to be history matched. The sensitivities can be calculated easily once the adjoint variables have been computed. Solving an adjoint problem is similar to solving the simulation finite difference equations with two distinct differences: (i) to find the adjoint variables needed to computed the sensitivity of a particular production data at the time t_l, the appropriate adjoint problem is solved backward in time, from time t_l to time zero; (ii) unlike the forward problem (simulator problem), the adjoint problem is linear. At each time step in the adjoint solution, a matrix problem is solved. The coefficient matrix is independent of the production data but the right hand side of the matrix problem is determined directly from the specific production data. If N_d production data are uniformly spaced in time and the final time at which we have measured production data is t_L, then computing all adjoint solutions needed to compute the sensitivities of all production data effectively requires solving a sequence of matrix problems related to solving the adjoint problem backward in time from t_L. At each time step, the matrix problem is solved with an average of $(N_d + 1)/2$ right-hand side vectors; see Wu et al. (1999) for additional discussion. Even if one uses a procedure based on solving a matrix problem with multiple right hand sides, it is likely that solution of the adjoint systems needed to compute sensitivities for N_d production data will not be feasible

when N_d is large. If one assumes that solving the adjoint matrix systems with an average of $(N_d + 1)/2$ right hand side vectors is equivalent to 0.05 times $(N_d + 1)/2$ simulation runs, the number of equivalent simulation runs required is prohibitive if there are several hundred production data to be matched. Because of this, the adjoint method traditionally has been used only in conjunction with optimization methods which require only the gradient of the objective function, e.g., conjugate gradient or variable metric methods; see, for example, Wasserman et al. (1975), Lee and Seinfeld (1987a,b), Yang and Watson (1988), Makhlouf et al. (1993). Computation of the gradient of the objective function requires only the solution of a single adjoint system and thus requires no more computational time than one reservoir simulation run. Unfortunately, the implementations of these methods have resulted in slow convergence. For example, Makhlouf et al. (1993) reported that history matching a two-phase flow 450 cell reservoir model required 6400 CPU seconds on a CRAY X-MP/48. In their work, a conjugate gradient method was used as the optimization algorithm. For one three-phase flow problem with 450 grid blocks, 222 iterations of the conjugate gradient algorithm were required to obtain convergence.

Largely because of the results of Makhlouf et al. (1993), until recently, our work on automatic history matching has focused on using the Gauss-Newton and Levenberg-Marquardt algorithms instead of conjugate gradient or variable metric algorithms. Wu et al. (1999) were the first to use the adjoint method in conjunction with the Gauss-Newton method to perform history matching. They implemented the adjoint method to compute the sensitivity of all production data to gridblock permeabilities and porosities. In their work, they constructed estimates and realizations of permeability and porosity fields by conditioning a prior geostatistical model to pressure and water-oil ratio data. They considered only two-dimensional, two-phase flow (water-oil) systems. In this work, we extend the procedure of Wu et al. to three-dimensional, three-phase flow problems. This minimization procedure will definitely not be practically feasible if both the number of data and the number of model parameters exceed a few hundred.

Perhaps because it is simple to implement, the so-called gradient method is frequently used to compute sensitivity coefficients needed for automatic history matching. This method was introduced into the petroleum engineering literature by Anterion et al. (1989), but was known earlier in the ground water hydrology literature as the sensitivity coefficient method; see, for example, the review of parameter identification methods by Yeh (1986). In this procedure, the sensitivity of pressures and saturations to model parameters at the end of a simulator time-step can be obtained by solving a matrix problem obtained by differentiating the matrix form of the finite-difference equations with respect to a model parameter, e.g., a

gridblock value of permeability or porosity. From the pressure and saturation sensitivities, one can easily construct other sensitivity coefficients, e.g., the sensitivity of gas-oil ratio to model parameters. The advantage of the gradient simulator method is that the matrix problem solved to obtain these sensitivity coefficients involves the same coefficient matrix as the one used to solve for pressures and saturations at this time step. Moreover, the coefficient matrix does not depend on the model parameters; only the right hand side of the matrix problem depends on the model parameters. Thus, the problem reduces to solving a matrix problem with multiple right-hand side vectors, one right-hand side vector, for each model parameter. The difficulty is that if we wish to estimate (or construct realizations of) permeabilities and porosities at several thousand gridblocks, then we have several thousand right-hand sides. The number of right-hand sides is equal to the number of model parameters to be estimated. With the fast iterative solver developed by Killough et al. (1995), it appears that the computational time to compute a single sensitivity coefficient is on the order of 10% of a forward simulation. For the gradient simulator to be practical, the number of model parameters must be small. This means, if the underlying reservoir simulation problem involves tens of thousands or gridblocks, one must reduce the number of parameters estimated directly in the optimization algorithm by some form of reparameterization, e.g., zonation (Jacquard and Jain, 1965) or gradzones (Bissell et al., 1994; Bissell, 1994; Tan, 1995), pilot points (de Marsily et al., 1984; RamaRao et al., 1995; Bissell et al., 1997) or subspace methods (Kennett and Williamson, 1988; Oldenburg et al., 1993; Reynolds et al., 1996; Abacioglu et al., 2000).

When the number of model parameters and number of production data to be matched are both large and can not be reduced by some reparameterization technique without incurring a significant loss of information, one must seek an alternative to computing and storing the full sensitivity coefficient matrix, G. One can write the Gauss-Newton method such that each iteration requires the solution of an $N_d \times N_d$ matrix problem where N_d represents the number of production data to be matched. If this matrix problem is solved by a conjugate gradient method (Hestenes and Stiefel, 1952; Fletcher and Reeves, 1964; Shanno, 1978a,b), the explicit direct computation of G is not required. Each iteration requires only the product of G times a vector and the product of the transpose of G times a vector. A procedure for computing these matrix vector products without first computing G was introduced into the petroleum engineering literature by Chu et al. (2000) although the basic idea appeared earlier in a somewhat simpler context in the geophysics literature; see Mackie and Madden (1993). Although computation of the matrix products is relatively efficient, the conjugate gradient method may require up to N_d iterations to obtain convergence if the matrix is poorly condi-

tioned and no good preconditioning matrix is available (see Axelsson (1994)). If N_d is large, this would render the algorithm impractical. Although the convergence of the conjugate gradient method can be considerably accelerated by the choice of a good preconditioner, it is not clear that one can construct a good preconditioner since the coefficient matrix for the matrix problem that is solved is not explicitly computed. To compute this coefficient matrix would require the explicit computation of the full sensitivity coefficient matrix G.

One can also avoid explicit computation of all sensitivity coefficients if history matching is done using a nonlinear optimization method, that requires only the gradient of the objective function. As mentioned previously, Makhlouf et al. (1993) found that a nonlinear conjugate gradient algorithm could require over two hundred iterations to converge even for a small three-phase flow history matching problem. As each conjugate gradient iteration requires roughly the equivalent of three reservoir simulation runs, history matching a large problem using a nonlinear conjugate gradient method does not appear to be feasible based on the results of Makhlouf et al. (1993). However, Makhlouf et al. (1993) did not apply preconditioning. If a good preconditioning matrix can be found for nonlinear conjugate gradients, it is conceivable that convergence could be considerably accelerated. The most straightforward choice of a preconditioning matrix (Kalita and Reynolds, 2000) yielded some improvement in the convergence rate, but also often led to poorer matches of production data than were obtained with the Gauss-Newton method. In this work, we explore the use of approximations to the inverse Hessian matrix constructed from a variable metric method as a preconditioning matrix for the nonlinear conjugate gradient method. We also explore the direct use of variable metric methods as nonlinear optimization algorithms for automatic history matching.

Quasi-Newton or variable metric methods, which are based on generating an approximation to the inverse of the Hessian matrix, require only the gradient of the objective function. The methods differ in how they correct or update the inverse Hessian approximation at each iteration. The rank one correction formula was first suggested by Broyden (1967). Another formula, now called the DFP algorithm, was first suggested by Davidon in 1959 and later presented by Fletcher and Powell (1963). The BFGS correction formula, suggested independently by Broyden (1970), Fletcher (1970), Goldfarb (1970) and Shanno (1970), and several variants of the BFGS formula (like the self-scaling variable metric (SSVM) by Oren (1973), limited memory BFGS by Nocedal (1980) and Liu and Nocedal (1989)) have also been advanced as useful variable metric methods.

The conjugate gradient method was originally proposed by Hestenes and Stiefel (1952) for solving linear systems and extended to nonlinear optimization by Fletcher and Reeves

(1964) to obtain the Fletcher-Reeves algorithm. Later Polak (1971) proposed a different formula to calculate the coefficient involved in the search direction update equation. Powell (1977) presented some numerical results and theoretical reasons which indicate that the Polak-Ribière algorithm is superior to the Fletcher-Reeves algorithm. The efficiency of the conjugate gradient method depends primarily on the preconditioner used.

The limited memory BFGS (LBFGS) was designed for the purpose of solving large scale problems which involve thousands of variables. Limited memory methods originated with the work of Shanno (1978a), and were subsequently developed and analyzed by Buckley (1978), Nazareth (1979), Nocedal (1980), Shanno (1978b), and Buckley and Lenir (1983). Liu and Nocedal (1989), and Nash and Nocedal (1991) tested LBFGS method with a set of problems. They concluded that LBFGS performs better than conjugate gradient in terms of computational efficiency, except in cases where the function evaluation is inexpensive. Nash and Nocedal (1991) also tested a truncated-Newton method in their work. From their comparison, none of the algorithms is clearly superior to the other.

The self-scaling variable metric (SSVM) method was used by Yang and Watson (1988) on hypothetical water floods of both 1D and 2D reservoir models. The 1D reservoir model consisted of 10 gridblocks with an injection well at one end and a producing well at the other end. Sixty data from each well were used for history matching. Four cases based on this 1D reservoir model were tested. The reservoir was characterized by different parameters in different cases. The number of model parameters varied from 9 to 19. Two other cases were based on a quarter of a five-spot 2D model which consisted of a 10×10 grid. Again sixty data from each well were history matched. The number of model parameters for these two cases were 4 and 11 respectively. In this paper, the authors tested four different algorithms, BFGS, SSVM, conjugate gradient and steepest descent. They concluded that (i) the self-scaling variable metric method is significantly more efficient than the BFGS method; (ii) the SSVM and BFGS methods are more efficient and robust than the conjugate gradient method, except in the case where the objective function is nearly quadratic; and (iii) both SSVM and BFGS methods perform significantly better than the steepest descent method.

Masumoto (2000) applied the SSVM method to a water-oil two phase fluid flow problem. The author considered a 1D reservoir model with 20 gridblocks. With a fixed porosity field, the author estimated the gridblock permeabilities. The objective function he minimized included a pressure mismatch part and the pressure derivative mismatch part. The author did not give any information about how many data he history matched or any assessment of the minimization algorithm. Savioli and Grattoni (1992) compared four different minimization algorithms: Davidon-Fletcher-Powell (DFP), Fletcher-Reeves (FR), BFGS and Levenberg-

Marquardt (LM). The authors presented two examples. In the first example, they estimated one permeability value and one porosity value by applying these four algorithms. The second example they considered was an oil-water two phase water flooding problem. They estimated the exponent used to define the relative permeability and capillary curves with a power law function (only one adjustable parameter for each curve). They concluded that among these four algorithms, BFGS performed best in terms of computational efficiency and stability. Given the small number of parameters estimated, it is difficult to know whether these results will extrapolate to large scale history matching problems.

In this work, we present the results of experiments on history matching pressure data from a gas reservoir using preconditioned conjugate gradient and variable metric (quasi-Newton) methods. Results are compared in terms of three metrics, computational time (measured by number of equivalent reservoir simulation runs) required for convergence, computer memory required, and quality of history match. The results identify a specific preconditioned conjugate gradient method and a limited memory BFGS as promising optimization algorithms for automatic history matching of multiphase flow production data.

Chapter 2

HISTORY MATCHING OF PRODUCTION DATA

2.1 Model Estimation and Simulation

Here, we define the reservoir model parameters and the a posteriori probability density function (pdf) for these parameters. This pdf, which is conditional to production data, defines the set of plausible reservoir descriptions. We discuss computation of the maximum a posteriori (MAP) estimate. The MAP estimate is the model which maximizes the a posteriori pdf and is thus conveniently referred to as the most probable model. Methods for sampling this pdf to characterize the uncertainty in model parameters and the uncertainty in performance predictions are discussed only briefly.

2.1.1 The Prior Model.

For simplicity, the reservoir is assumed to be a rectangular parallelepiped which occupies the region

$$\Omega = \{(x, y, z) \mid 0 < x < L_x, 0 < y < L_y, 0 < z < L_z\}. \tag{2.1}$$

The forward model is a fully-implicit finite-difference simulator based on a block centered grid. The principle permeability directions are assumed to be aligned with the coordinate directions so that the permeability tensor is diagonal. Fluid properties are assumed to be known. Given two-phase oil-water and two-phase oil-gas relative permeabilities, the three-phase oil relative permeability is constructed from Stone's Model II; see Aziz and Settari (1979). Wellbore constraints are handled using the equation of Peaceman (1983).

In the current version of our inverse code, we can compute sensitivity of production data to gridblock porosities, horizontal permeabilities, vertical permeabilities and the skin factor at each well. For the specific example considered here, we only estimate the horizontal and vertical permeability fields and the well skin factors, so we present our equations in terms of these model parameters. Thus, if there are N simulator gridblocks and N_w wells, the total number of model parameters is equal to $M = 2N + N_w$. Specifically, the vector of model parameters is given by

$$m = [m_k^T, m_{k_z}^T, m_s^T]^T, \tag{2.2}$$

where m_k is an N-dimensional column with its jth entry equal to the horizontal permeability for gridblock j, m_{k_z} is an N-dimensional column with its jth entry equal to the vertical permeability for gridblock j, and m_s is an N_w dimensional column vector with its jth entry given by the skin factor at the jth well. These reservoir parameters are modeled as random variables, so m is a random vector. From a purely history matching point of view, we wish to construct an estimate of m from production data (dynamic data) and static data. However, there are an infinite number of models which will give equally reasonable matches of the data, and it is desirable to define a procedure for generating a particular estimate or to characterize the uncertainty in reservoir descriptions. From both the philosophical and practical points of view (see Tarantola (1987) and Omre et al. (1993)), the most challenging part of the inverse problem is the determination of a representative pdf for reservoir parameters. Similar to the recent work on automatic history matching by He et al. (1997) and Wu et al. (1999), we follow ideas that can be found in Tarantola (1987) and simply assume that a prior geostatistical model for $m_r = [m_k^T, m_{k_z}^T]^T$ can be constructed from static data. In our work, we assume this prior geostatistical model can be represented by a multivariate Gaussian distribution for m_r with a given mean and covariance matrix. In practice, the prior covariance matrix for the rock property fields can be generated from semivariograms by assuming that horizontal, vertical permeability can be modeled as stationary random functions. In our implementation, we make this assumption and then apply the Xu et al. (1992) screening hypothesis to generate the prior covariance matrix for m_r; see, Chu et al. (1995b). In the prior model, each well skin factor is treated as an independent Gaussian variable with specified mean and variance. If the skin factor was estimated by fitting pressure data with a classical well testing model solution using nonlinear regression, then the estimate of the skin factor would be its prior mean and its variance can be constructed directly from the same information used to construct confidence intervals.

The vector of prior means is given by

$$m_{\text{prior}} = \begin{bmatrix} m_{k,\text{prior}} \\ m_{k_z,\text{prior}} \\ m_{s,\text{prior}} \end{bmatrix}. \tag{2.3}$$

We let C_k denote the prior covariance matrix for m_k, C_{k_z} denote the prior covariance for k_z, C_{k,k_z} denote the cross covariance matrix between k and k_z and let C_s denote the $N_w \times N_w$ model covariance matrix for the vector of well skin factors. Then the prior model covariance matrix is given by

$$C_M = \begin{bmatrix} C_k & C_{k,k_z} & O \\ C_{k,k_z} & C_{k_z} & O \\ O & O & C_s \end{bmatrix}, \tag{2.4}$$

where the O's denote null submatrices of the appropriate size. If horizontal and vertical permeability are not correlated, then C_{k,k_z} is also a null matrix.

The prior pdf for m is then given by

$$\pi_p(m) = a \exp\Big\{ -\frac{1}{2}(m - m_{\text{prior}})^T C_M^{-1}(m - m_{\text{prior}})\Big\}, \tag{2.5}$$

where a is the normalizing constant. Note the model which has the highest probability based on Eq. 2.5 is $m = m_{\text{prior}}$, thus it is convenient to think of m_{prior} as the best estimate of the model based on static data.

2.1.2 The a Posteriori Probability Density Function.

We wish to determine the conditional pdf for m given observed production data. Here, we consider only three types of production data, wellbore pressure (p_{wf}), producing water-oil ratio (WOR) and producing gas-oil ratio (GOR). The WOR and GOR data are not actually measured directly but are constructed from rate measurements. Nevertheless, we will refer to the values of WOR and GOR as measured or observed data. The column vector $d_{obs,w}$ contains all observed WOR data that will be used as conditioning data. The column vector $d_{obs,g}$ contains the set of GOR conditioning data and $d_{obs,p}$ contains all conditioning pressure data. Throughout, the N_d dimensional column vector d_{obs} includes all production data that will be used to condition the model m. This may include one type of data, e.g., only GOR data or multiple types of data, e.g., pressure, WOR and GOR data.

Pressure measurements errors are modeled as independent identically distributed Gaussian random variables with mean zero and variance σ_p^2. GOR measurement errors are modeled as independent identically distributed Gaussian random variables with mean zero and

variance σ_g^2. WOR ratio measurement errors are modeled by the procedure introduced by Wu et al. (1999). In this model, the WOR measurement error depends on the magnitude of the measurement. Specifically, the variance of a particular measurement error is defined as

$$\mathrm{Var}(e_{WOR}) = WOR_{obs}^2 \epsilon_o + \frac{1}{q_{o,obs}^2} \max\Big[\epsilon_w^2 q_{w,obs}^2,\, \sigma_{w,min}^2\Big], \tag{2.6}$$

where e_{WOR} denotes the error in the "measurement" of WOR constructed from the observed oil and water rates, $q_{o,obs}^2$ and $q_{w,obs}^2$. Here, ϵ_m denotes the relative measurement error for the flow rate of phase m. For example, if the relative measurement error in the oil flow rate is two per cent, then $\epsilon_o = 0.02$. The term $\sigma_{w,min}$ is used so that we do not prescribe unrealistically small measurement errors for the WOR when the WOR is small. To use this model, one must specify values of ϵ_w, ϵ_o, and $\sigma_{w,min}$. The three diagonal matrices, $C_{D,p}$, $C_{D,w}$ and $C_{D,g}$, respectively, denote the covariance matrices for pressure data measurement errors, WOR measurement errors and GOR measurement errors. If the total number of conditioning data is N_d, i.e., the dimension of d_{obs} is N_d, then the overall data covariance matrix is given by the following $N_d \times N_d$ diagonal matrix:

$$C_D = \begin{bmatrix} C_{D,p} & O & O \\ O & C_{D,w} & O \\ O & O & C_{D,g} \end{bmatrix}. \tag{2.7}$$

We of course do not need to use all types of data as conditioning data. For example, if we wish to history match only GOR data, then $d_{obs} = d_{obs,g}$ and $C_D = C_{D,g}$.

For a given model m, d denotes the predicted, true or calculated data corresponding to d_{obs}. If m is the true reservoir from which d_{obs} was obtained and there are no measurement errors, then $d = d_{obs}$. As d depends on the model, we write

$$d = g(m), \tag{2.8}$$

to represent the operation of calculating d given m. In our work, Eq. 2.8 represents the operation of running the reservoir simulator to calculate d.

Bayes' theorem (see Tarantola (1987)) implies that the a posteriori pdf for the model m conditional to the observed data is proportional to the product of the prior pdf and the likelihood function for the model, and is thus given by

$$f(m|d_{\mathrm{obs}}) = a \exp\{-O(m)\}, \tag{2.9}$$

where a is the normalizing constant and

$$O(m) = \frac{1}{2}\Big[\big(m - m_{\mathrm{prior}}\big)^T C_M^{-1} \big(m - m_{\mathrm{prior}}\big) + \big(g(m) - d_{\mathrm{obs}}\big)^T C_D^{-1} \big(g(m) - d_{\mathrm{obs}}\big)\Big]. \tag{2.10}$$

Construction of the MAP Estimate and Realizations.

The maximum a posteriori (MAP) estimate is denoted by m_∞ and is defined to be the model that maximizes the pdf of Eq. 2.9, or equivalently minimizes the objective function of Eq. 2.10. Although gradient based methods appear to be the only feasible way to construct a minimum of $O(m)$, there is no guarantee that Eq. 2.10 has a unique global minimum, or that a gradient-based optimization procedure will converge to a global minimum. In fact, if a gradient method is applied to minimize $O(m)$, it is important to check the results to ensure that the method did not converge to a local minimum which yields an unacceptable match of production data, or unreasonable reservoir properties.

If one wishes to generate multiple realizations of the model, it is necessary to sample the conditional pdf of Eq. 2.9. The most common way to do this is to apply the method proposed by Oliver et al. (1996) and Kitanidis (1995). In our work this method is referred to as the randomized maximum likelihood method. To generate a realization with this procedure, we calculate a unconditional realization $m_{\rm uc}$ from

$$m_{\rm uc} = m_{\rm prior} + C_M^{1/2} z_M, \tag{2.11}$$

where z_M is N_m-dimensional column vector of independent standard random normal deviates. The matrix $C_M^{1/2}$ is a square root of C_M and is normally chosen as $C_M^{1/2} = L$ where

$$C_M = LL^T, \tag{2.12}$$

is the Cholesky decomposition of C_M. For large problems, generation of the Cholesky decomposition is not feasible, and we apply sequential Gaussian co-simulation to generate an unconditional realization of the model; see Gómez-Hernández and Journel (1992). Similarly a realization of the data is generated from

$$d_{\rm uc} = d_{\rm obs} + C_D^{1/2} z_D, \tag{2.13}$$

where z_D is an N_d-dimensional column vector of standard random normal deviates. The conditional realization of m is then obtained by minimizing

$$O_r(m) = \frac{1}{2}(m - m_{\rm uc})^T C_M^{-1}(m - m_{\rm uc}) + \frac{1}{2}(d - d_{\rm uc})^T C_D^{-1}(d - d_{\rm uc}). \tag{2.14}$$

It can be argued (see Zhang et al. (2001)) that $O(m)$ can be approximated as a chi-squared distribution with expectation given by $E(O(m)) = N_d$ and standard deviation given approximately by $\sigma(O(m)) \approx \sqrt{2N_d}$. Virtually all samples should be within five standard deviations

of the mean. Thus, if applying an optimization algorithm to minimize Eq. 2.14 gives a result m_c, we accept m_c as a legitimate realization if and only if

$$N_d - 5\sqrt{2N_d} \leq O(m_c) \leq N_d + 5\sqrt{2N_d}. \tag{2.15}$$

If Eq. 2.15 is not satisfied, the minimization algorithm has failed. Normally this failure indicates that the algorithm has converged to a local minimum or converges so slowly that the decrease in the objective function is so small that the convergence criteria, which is based on the change in the objective function, is satisfied before we actually reach a minimum.

Ultimately, we wish to be able to history match several hundred production data to generate realizations of tens of thousands of model parameters. Thus computational efficiency is an extremely important consideration. It is equally important that the algorithm be robust, i.e., convergence failures should be extremely small. If minimization of $O_r(m)$ frequently result in value $O_r(m_c)$ which does not satisfy Eq. 2.15, the utility of the optimization algorithm is diminished.

The Gauss-Newton method with restricted step has often been used to minimize $O(m)$; see Chu et al. (1995a) However, if the initial guess for the model yields a very poor match of the observed production data, a straightforward application of the method may converge extremely slowly or may converge to a model which yields an unacceptable match of production data; see Wu et al. (1999). Wu et al. (1999) overcame this problem by using an artificially high value for the variance of data measurement errors at early iterations. Here, we avoid this difficulty by using a form of the Levenberg-Marquardt algorithm introduced by Bi (1999). This algorithm can be written in two different forms. The first comes from a modification of the standard Gauss-Newton method and is given by

$$\left[(1+\lambda_l)C_M^{-1} + G_l^T C_D^{-1} G_l\right]^{-1} \delta m^{l+1} = -\left[C_M^{-1}(m^l - m_{\text{prior}}) + G_l^T C_D^{-1}(g(m^l) - d_{\text{obs}})\right], \tag{2.16}$$

$$m^{l+1} = m^l + \alpha_l \delta m^{l+1} \tag{2.17}$$

where $\alpha_l = 1$. Here l, as either a subscript or superscript, refers to the iteration index. The matrix G_l denotes the $N_d \times M$ sensitivity coefficient matrix evaluated at m^l. The entry in the ith row and jth column of G_l represents the sensitivity of the ith calculated data g_i to the jth model parameter evaluated at m^l, i.e., this entry is $\partial g_i(m^l)/\partial m_j$, where m_j is the jth entry of m. If $O(m^{l+1}) < O(m^l)$, we set $\lambda_{l+1} = \lambda_l/10$, and if the objective function does not decrease, we increase the Levenberg-Marquardt parameter by a factor of 10. We start with an initial value of $\lambda = 10,000$. For the multiphase flow problems we have considered to date, this simple procedure works well.

Although we focus on the generation of the MAP estimate in this work, it is possible to use a similar procedure to generate a realization with the randomized maximum likelihood method introduced by Kitanidis (1995) and Oliver et al. (1996). To do so, we simply replace $d_{\rm obs}$ by d_{uc} and $m_{\rm prior}$ by m_{uc} in Eq. 2.16.

The second form of the Levenberg-Marquardt algorithm can be obtained from Eq. 2.16 by applying standard matrix inversion lemmas (see Tarantola (1987)) and is given by

$$\delta m^{l+1} = \frac{m^l - m_{\rm prior}}{1+\lambda_l} + C_M G_l^T \Big[(1+\lambda_l)C_D + G_l C_M G_l^T\Big]^{-1}\Big[\frac{G_l(m^l - m_{\rm prior})}{1+\lambda_l} - \big(g(m^l) - d_{\rm obs}\big)\Big]. \tag{2.18}$$

Again m^{l+1} is obtained from Eq. 2.17 with $\alpha_l = 1$. Choosing $\lambda = 0$ in Eqs. 2.16 and 2.18 gives two forms of the Gauss-Newton method. When the Gauss-Newton algorithm is applied, we use the restricted step method Fletcher (1987) to calculate the damping factor α_l in Eq. 2.17.

The formula of Eq. 2.16 requires calculation of C_M^{-1} and then solving an $N_m \times N_m$ matrix problem where N_m is the number of model parameters. If the number of model parameters is small, this may be a computationally efficient procedure. Applying Eq. 2.18 requires solving an $N_d \times N_d$ matrix problem where N_d is the number of data. If the number of production data that will be history matched is small, this is the preferred procedure. If N_m exceeds a few thousand and N_d exceeds a few hundred, the computational resources required to apply either form of the LM algorithm are too large for routine practical application.

Eq. 2.18 requires solving

$$\big((1+\lambda_l)C_D + G_l C_M G_l^T\big)x = G_l(m^l - m_{\rm prior}) - (g(m^l) - d_{\rm obs}), \tag{2.19}$$

for

$$x = \big(C_D + G_l C_M G_l^T\big)^{-1}\big(G_l(m^l - m_{\rm prior}) - (g(m^l) - d_{\rm obs})\big). \tag{2.20}$$

If the matrix problem of Eq. 2.19 is solved iteratively by the conjugate gradient method, then one does not need to explicitly compute G; one only needs to be able to calculate Gu and $G^T v$ for vectors u and v at each iteration of the Gauss-Newton or Levenberg-Marquardt algorithm; see Chu et al. (2000) et al. A clear presentation of how one may compute Gu and $G^T v$ is given in Abacioglu (2001). Computation of Gu requires a forward run of the simulation. Computation of $G^T v$ requires one solution of the adjoint system. As the solution of the adjoint system requires roughly the same computational time as one simulation run, each iteration of the conjugate gradient method requires roughly two reservoir simulation runs. Computation of the right hand side of Eq. 2.19 also requires one simulator run but must be done only once for each Gauss-Newton iteration. To apply the conjugate gradient algorithm, we need to provide the initial estimate for the solution and calculate the residual

corresponding to the initial estimate which requires one operation of Gu and one operation of $G^T v$. To accomplish one Levenberg-Marquardt iteration, we need one more operation of $G^T v$ outside the inner iteration; see Eq. 2.18. Thus, if the inner iteration (the solution of Eq. 2.19 by the conjugate gradient method) requires on average k_{cg} iterations for convergence and k_{LM} iterations are required to obtain convergence of the Levenberg-Marquardt, roughly

$$I_{LM} = k_{LM}(2k_{cg} + 4), \tag{2.21}$$

reservoir simulation runs are required to generate a MAP estimate or realization. For the overall procedure to be feasible k_{cg} must be quite small. If an extremely good preconditioning matrix could be found for the conjugate gradient step, it is likely that the method could be quite effective. Since the matrix $(1 + \lambda_l)C_D + G_l C_M G_l^T$ is never explicitly formed, however, it is difficult to see how a good preconditioner could be generated. (For the case where C_D is a constant diagonal matrix, using C_D as a preconditioning matrix would not alter the condition number.)

For the multiphase flow history matching examples presented in this work, the form of the LM algorithm given by 2.18 is applied. The convergence tolerance is defined by

$$\frac{O(m^l) - O(m^{l+1})}{O(m^l) + 10^{-14}} \leq 10^{-3}. \tag{2.22}$$

2.1.3 Evaluation of Uncertainty.

The best way to evaluate uncertainty in reservoir properties would be to construct a large suite of realizations by sampling the pdf of Eq. 2.9; see, for example, Hegstad and Omre (1997), Omre et al. (1996), He (1997) or Wu (1999). Although sampling the pdf is preferable, the computational expense of doing so even with the relatively efficient randomized likelihood method can be considerable. For the case where we simply construct the MAP estimate of the model, the a posteriori covariance matrix can be used to provide an approximate evaluation of uncertainty in individual model parameters. Tarantola (1987) has shown that the a posteriori covariance matrix can be written as

$$C_{MP} = C_M - C_M G^T \left(G C_M G^T + C_D\right)^{-1} G C_M. \tag{2.23}$$

In this equation, G is the matrix of sensitivity coefficients evaluated at the maximum a posteriori estimate of the model. The matrix C_{MP} gives the exact characterization of the uncertainty in model parameters if the relationship between data and model parameters is linear. In this case, the diagonal elements of this matrix give the a posteriori variances of the

model parameters, which represents one measure of the uncertainty in the model parameters. One should note, however, that the ith diagonal element of C_{MP} gives only the variance of the marginal distribution of m_i and neglects the effect that the correlation between parameters has on the reduction in uncertainty. For three-dimensional problems, conditioning to production data may yield a negligible reduction in the uncertainty in individual model parameters, but may significantly reduce the uncertainty in predicted performance; see He (1997) and He et al. (1997)

As model parameters have dimensions and the differences in variances between different types of parameters may be large, one should scale the a posteriori variances to evaluate the relative reduction in uncertainty; see He et al. (1997). For the problems considered here, we will simply use the diagonal entries of C_{MP} divided by the corresponding prior variances (diagonal entries of C_M) to measure the reduction in model uncertainty.

Sensitivity coefficients indicate how strongly a model parameter influences a particular data. In particular, the sensitivity of the ith calculated data $d_i = g_i(m)$ to the jth model parameter m_j is given by

$$\hat{s}_{i,j} = \frac{\partial g_i}{\partial m_j}. \tag{2.24}$$

Recently, Zhang et al. (2000) showed that examination of raw sensitivity coefficients can result in misleading interpretations of which model parameters are best resolved by observed data. This is particularly true for parameters which have radically different magnitudes, such as porosity and permeability. For example, the sensitivity of active well transient pressure data to porosity may be much greater than the sensitivity of pressure data to permeability although flowing wellbore pressure data resolves permeability better than it does porosity. To understand the relative reduction in the uncertainty in model parameters that is obtained by history matching observed data, Zhang et al. (2000) introduced dimensionless sensitivity coefficients. The dimensionless sensitivity of the ith data, g_i, to the jth model parameter is defined by

$$s_{i,j} = \frac{\partial g_i}{\partial m_j}\frac{\sigma_{m_j}}{\sigma_{d,i}}, \tag{2.25}$$

where σ_{m_j} denotes the prior standard deviation of model parameter m_j and $\sigma_{d,i}$ denotes the standard deviation of the measurement error for the ith observed data. At least in an approximate sense, if $s_{i,j} > s_{i,l}$, then conditioning a model to observed data $d_{\text{obs},i}$ will reduce the uncertainty in parameter m_j more than parameter m_l. If m_l represents a gridblock porosity and m_j a gridblock log-permeability, then the standard deviation for m_l typically will exceed one hundred times the standard deviation of porosity, i.e., $\sigma_{m_j} > 100\sigma_{m_l}$. Thus, we may have $s_{i,j} > s_{i,l}$ even though $\hat{s}_{i,l} > \hat{s}_{i,j}$. Dimensionless sensitivities give only an

approximate measure of the reduction in uncertainty in model parameters because they do not account for any reduction in uncertainty due to prior or posterior correlations between parameters.

2.2 Calculation of Sensitivity Coefficients

The equations that must be solved to compute sensitivity coefficient with the adjoint method are presented in this section.

2.2.1 The Reservoir Simulator.

The simulator used is based on a fully-implicit, finite-difference formulation of the three-phase flow, black-oil equations expressed in an x–y–z coordinate system which apply on Ω; see Eq. 2.1. At every gridblock, three finite difference equations apply. These $3N$ equations represent the mass balances for each of the three components. In addition, a constraint is applied at each of the N_w wells to yield N_w additional equations. At each well at each time step, either an individual phase flow rate, the total flow rate or the wellbore pressure may be specified as a well constraint. In the results considered here, capillary pressures are assumed to be negligible.

For gridblock i, the primary variables are $p_i, S_{w,i}$ and $S_{g,i}$. In addition, the flowing wellbore pressure, $p_{wf,j}$ at the jth well is a primary variable. We let y^n denote a column vector which contains the set of primary variables at time step n. At gridblock i, the finite difference equation for component u can be written

$$f_{u,i}(y^{n+1}, y^n, m) = f_{u,i}^{n+1} = 0, \tag{2.26}$$

for $u = o, w, g$ and $i = 1, \ldots, N$. The well constraints are represented by

$$f_{wf,j}(y^{n+1}, y^n, m) = f_{wf,j}^{n+1} = 0, \tag{2.27}$$

for $j = 1, 2, \ldots, N_w$. If the flowing wellbore pressure at well j at time t^{n+1} is specified to be equal to $p_{wf,j,0}^{n+1}$, then Eq. 2.27 is given by

$$f_{wf,j}(y^{n+1}, y^n, m) = p_{wf,j}^{n+1} - p_{wf,j,0}^{n+1} = 0. \tag{2.28}$$

Eqs. 2.26 and 2.27 represent the system of $3N + N_w$ equations that are solved to obtain the values of the primary variables at time $t^{n+1} = t^n + \Delta t^n$. For wells at which the flowing bottom hole pressure or total flow rate is specified, phase flow rates at each well are computed by

using the equation of Peaceman (1983). The complete system of equations can formally be written as

$$f^{n+1} = f(y^{n+1}, y^n, m) = \begin{bmatrix} f_{o,1}^{n+1} \\ f_{w,1}^{n+1} \\ f_{g,1}^{n+1} \\ f_{o,2}^{n+1} \\ \vdots \\ f_{g,N}^{n+1} \\ f_{wf,1}^{n+1} \\ \vdots \\ f_{wf,N_w}^{n+1} \end{bmatrix} = 0, \tag{2.29}$$

where

$$y^{n+1} = \begin{bmatrix} p_1^{n+1} \\ S_{w,1}^{n+1} \\ S_{g,1}^{n+1} \\ p_2^{n+1} \\ \vdots \\ S_{g,N}^{n+1} \\ p_{wf,1}^{n+1} \\ \vdots \\ p_{wf,N_w}^{n+1} \end{bmatrix}. \tag{2.30}$$

Eq. 2.29 is solved for y^{n+1} by the Newton-Raphson method (Aziz and Settari (1979)) which can be written as

$$J^\nu \delta y^{n+1,\nu+1} = -f(y^{n+1,\nu}, y^n, m) \tag{2.31}$$

$$y^{n+1,\nu+1} = y^{n+1,\nu} + \delta y^{n+1,\nu+1}, \tag{2.32}$$

where ν is the iteration index and

$$J^\nu = \left([\nabla_{y^{n+1}} f^T]^T \right)_{y^{n+1,\nu}}, \tag{2.33}$$

is the Jacobian matrix evaluated at $y^{n+1,\nu}$,which represents the νth approximation for y^{n+1}.

2.2.2 Adjoint Equations.

We define a general scalar function by

$$\beta = \beta(y^1, ..., y^L, m), \tag{2.34}$$

where L corresponds to the last time t^L at which one wishes to compute sensitivity coefficients. The objective is to compute the sensitivities coefficients for β. As is well known, the adjoint functional J is obtained by adjoining Eq. 2.29 to the function β to obtain

$$J = \beta + \sum_{n=0}^{L} (\lambda^{n+1})^T f^{n+1}, \tag{2.35}$$

where λ^{n+1} is the vector of adjoint variables at time step $n+1$, and is given by

$$\lambda^{n+1} = \left[\lambda_1^{n+1}, \quad \lambda_2^{n+1}, \quad \ldots, \quad \lambda_{3N+N_w}^{n+1}\right]^T. \tag{2.36}$$

Taking the total differential of Eq. 2.35, and doing some simple rearranging gives

$$\begin{aligned} dJ &= d\beta + \sum_{n=0}^{L} \left\{ (\lambda^{n+1})^T [\nabla_{y^{n+1}} (f^{n+1})^T]^T dy^{n+1} + [\nabla_m (f^{n+1})^T]^T dm \right\} + \sum_{n=0}^{L} (\lambda^{n+1})^T [\nabla_{y^n} (f^{n+1})^T]^T dy^n \\ &= d\beta + BT + \sum_{n=1}^{L} \{ [(\lambda^n)^T [\nabla_{y^n} (f^n)^T]^T + (\lambda^{n+1})^T [\nabla_{y^n} (f^{n+1})^T]^T] dy^n + (\lambda^n)^T [\nabla_m (f^n)^T]^T dm \}, \end{aligned} \tag{2.37}$$

where

$$BT = (\lambda^{L+1})^T \{ [\nabla_{y^{L+1}} (f^{L+1})^T]^T dy^{L+1} + [\nabla_m (f^{L+1})^T]^T dm \} + (\lambda^1)^T [\nabla_{y^0} (f^1)^T]^T dy^0. \tag{2.38}$$

The total differential of β can be written as

$$d\beta = \sum_{n=1}^{L} [\nabla_{y^n} \beta]^T dy^n + [\nabla_m \beta]^T dm. \tag{2.39}$$

The initial conditions are fixed, so

$$dy^0 = 0. \tag{2.40}$$

Choosing

$$\lambda^{L+1} = 0, \tag{2.41}$$

it follows that $BT = 0$. Using this result and Eq. 2.39 in Eq. 2.37 and rearranging the resulting equation gives

$$\begin{aligned} dJ = \sum_{n=1}^{L} \left[\left\{ (\lambda^n)^T [\nabla_{y^n} (f^n)^T]^T + (\lambda^{n+1})^T [\nabla_{y^n} (f^{n+1})^T]^T + [\nabla_{y^n} \beta]^T \right\} dy^n \right] \\ + \{ [\nabla_m \beta]^T + \sum_{n=1}^{N} (\lambda^n)^T [\nabla_m (f^n)^T]^T \} dm. \end{aligned} \tag{2.42}$$

To obtain the adjoint system, the coefficients multiplying dy^n in Eq. 2.42 are set equal to zero, i.e., we require that the adjoint variables satisfy

$$(\lambda^n)^T[\nabla_{y^n}(f^n)^T]^T + (\lambda^{n+1})^T[\nabla_{y^n}(f^{n+1})^T]^T + [\nabla_{y^n}\beta]^T = 0. \tag{2.43}$$

Taking the transpose of Eq. 2.43, gives the adjoint system

$$[\nabla_{y^n}(f^n)^T]\lambda^n = -[\nabla_{y^n}(f^{n+1})^T]\lambda^{n+1} - \nabla_{y^n}\beta. \tag{2.44}$$

This system is solved backwards in time for $n = L, L-1, \ldots, 1$.

Note that the coefficient matrix $(\nabla_{y^n}(f^n)^T)$ in the adjoint system is simply the transpose of the Jacobian matrix of Eq. 2.33 evaluated at y^n. The matrix multiplying λ^{n+1} in Eq. 2.44 is a diagonal band matrix which is only related to the accumulation terms in the reservoir simulation equations. This is an important result as it means that one can extract the matrices involved in the adjoint equations directly from the Jacobian matrices used in the simulator. This avoids the tedious process of directly deriving the individual adjoint equations. In this sense, our derivation shows that the adjoint method is somewhat similar to the gradient simulator method (Anterion et al. (1989)) in that the coefficient matrices that appear in both problems can be formed directly from Jacobian matrices used in solving the finite-difference equations by the Newton-Raphson method. As the adjoint system is solved backwards in time, information needed to compute the transpose of the Jacobian matrices must be saved from the simulation run, whereas, in the gradient simulator method, the desired sensitivity coefficients are computed at each time step during the simulation run.

Considering J as a function of m , we can write its total differential as

$$dJ = (\nabla_m J)^T dm. \tag{2.45}$$

By comparing Eq. 2.42 and Eq. 2.45, it follows that the desired sensitivity coefficients for J, or equivalently, β are given by

$$\nabla_m J = \nabla_m \beta + \sum_{n=1}^{L} [\nabla_m (f^n)^T](\lambda^n). \tag{2.46}$$

In Eq. 2.46, the gradient $\nabla_m \beta$ involves the partial derivatives of β with respect to the model parameters. If the jth model parameter does not explicitly appear in the expression for β, then $\partial\beta/\partial m_j = 0$. For example, if $\beta = p^n_{wf}$, then we set $\nabla_m \beta = 0$ in Eq. 2.46.

For the results considered here, the choices of β are restricted to the wellbore pressure, GOR and WOR at time steps where observed data for these variables are used as conditioning data. If one wishes to use a conjugate gradient (Makhlouf et al. (1993)) or variable

metric method (Yang and Watson (1988)), then one needs only compute the gradient of the objective function and this can be done by setting $\beta = O(m)$ in the adjoint procedure. In this case, one only needs to solve the adjoint system Eq. 2.44 once to obtain the gradient.

2.3 Dimensionless Sensitivity Coefficients, An Example

The example presented here is one which is sufficiently complex to be instructive but small enough so that we were able to check the sensitivity coefficients computed by the adjoint method with those computed by the finite-difference method. Although some of the sensitivity coefficient results presented are not easy to explain from a purely physical viewpoint, all of them have been compared with results generated with the finite-difference method. (In some previous publications, the finite-difference method was referred to as the direct method, see, for example, Chu and Reynolds (1995) or He et al. (1997).) For all cases that we have considered, the adjoint method and finite-difference method gave results that agreed to two significant digits. Thus plots of the sensitivities obtained from the two methods are graphically indistinguishable and we present only plots of sensitivities obtained by the adjoint method.

The areal extent of the reservoir is 600 feet by 600 feet and contains two layers. Layer 1 refers to the top layer and layer 2 refers to the bottom layer. The thickness of each layer is uniform and equal to 30 feet. An uniform $15 \times 15 \times 2$ finite difference grid with $\Delta x = \Delta y = 40$ ft and $\Delta z = 30$ ft is used in all reservoir simulation runs.

Each layer of the true model consists of three permeability regions. Fig. 2.1 shows the distribution of values of horizontal log-permeability in layer 1. For layer 1, $\ln(k) = 5.2$ ($k = 181$ md) in the lower left quadrant, $\ln(k) = 5.8$ ($k = 330$ md) in the lower right quadrant and $\ln(k) = 5.5$ ($k = 245$ md) in the upper half. For layer 2, $\ln(k) = 3.7$ ($k = 40$ md) in the lower left quadrant, $\ln(k) = 4.3$ ($k = 74$ md) in the lower right quadrant and $\ln(k) = 4.0$ ($k = 55$ md) in the upper half. In the truth case, vertical permeability is equal to one-tenth horizontal permeability in the top layer and is equal to two-tenths of horizontal permeability in the bottom layer. Thus, for layer 1, $\ln(k_z) = 2.9$ ($k_z = 18$ md) in the lower left quadrant, $\ln(k_z) = 3.5$ ($k_z = 33$ md) in the lower right quadrant and $\ln(k_z) = 3.2$ ($k_z = 24.5$ md) in the upper half. For layer 2, $\ln(k_z) = 2.1$ ($k_z = 8$ md) in the lower left quadrant, $\ln(k_z) = 2.7$ ($k_z = 15$ md) in the lower right quadrant and $\ln(k_z) = 2.4$ ($k_z = 11$) in the upper half. Here reservoir porosity is assumed to be uniform with $\phi = 0.22$. Even though the truth case consists of zones, permeabilities of each grid cell are estimated. The

simplicity of the example chosen allows one to easily visualize the quality of estimates. Some of the pertinent information is summarized in Table 2.1 where the first row gives the zone and the layer number. Throughout, Zone 1 refers to the upper half of the system as depicted in Fig. 2.1, Zone 2 refers to the lower right quadrant and Zone 3 refers to the lower left quadrant. The zone descriptions for layer 2 are the same. Var $\ln(k)$ denotes the variance of $\ln(k)$ and $E[\ln(k)]$ denotes its mean.

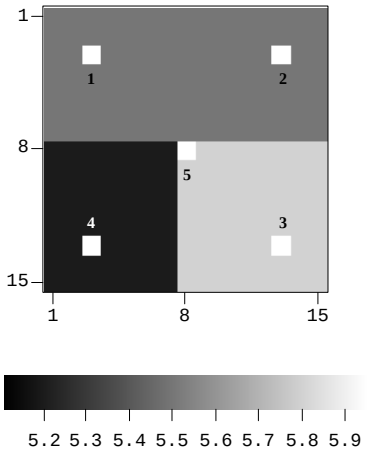


Figure 2.1: True model of horizontal log permeability, layer 1.

Table 2.1: True log-permeabilities and statistical parameters for example problem.

Zone/layer	1/1	2/1	3/1	1/2	2/2	3/2
true $ln(k)$	5.5	5.8	5.2	4.0	4.3	3.7
true k, md	245	330	181	55	74	40
true $ln(k_z)$	5.5	5.8	5.2	4.0	4.3	3.7
true k_z, md	24.5	33	18	11	15	8
Var $[ln(k)]$	0.5	0.5	0.5	0.5	0.5	0.5
$E[ln(k)]$	5.5	5.5	5.5	4.0	4.0	4.0
Var $[ln(k_z)]$	0.5	0.5	0.5	0.5	0.5	0.5
$E[ln(k_z)]$	3.2	3.2	3.2	2.4	2.4	2.4

Capillary effects are not included. Initial reservoir pressure at the depth corresponding to the center of the top layer is specified as $p_i = 4500$ psi. Initial bubble point pressure is

set equal to 4515 psi. Initial water saturation is equal to irreducible water saturation which is equal to 0.2. Initial oil saturation is $S_{o,i} = 0.8$. As mentioned previously, Stone's second model is used to calculate the relative permeability to oil from the two sets of two-phase relative permeability curves. For the two-phase oil-water system, residual oil saturation is 0.2. For the two-phase gas-oil system, residual oil saturation is 0.3 and critical gas saturation is equal to 0.05.

The reservoir contains four producing wells which are completed only in the top layer. The white gridblocks in Fig. 2.1 show the location of the gridblocks that contain wells. These wells are located near the four corners of the reservoir. Well 1 is completed in gridblock $(3, 3, 1)$, well 2 is completed is gridblock $(13, 3, 1)$, well 3 is completed in gridblock $(13, 13, 1)$ and well 4 is completed in gridblock $(3, 13, 1)$. Each of these four wells is produced at total flow rate of 350 RB/D. A single water injection well (well 5) completed in gridblock $(8, 8, 2)$ is used to inject water into the bottom layer at a rate of 1100 STB/D. At initial reservoir pressure, this is equivalent to an injection rate of 1107 RB/D. Note that the injection well (well 5) is located in the lower right quadrant of the bottom layer, which corresponds to the highest permeability zone ($\ln(k) = 4.3$) of the bottom layer. The true skins factors at wells 1 through 5, respectively, are specified as 3.0, 4.0, 5.0, 2.0 and 0.0.

By running the simulator using data from the truth case as input, the production response shown in Figs. 2.2 and 2.3 were obtained. As the injection rate in RB/D is less than the producing rate, the pressure at all four producing wells (not shown) continually decreases with time. Except at very early times, the pressure at the injection well also decreases with time.

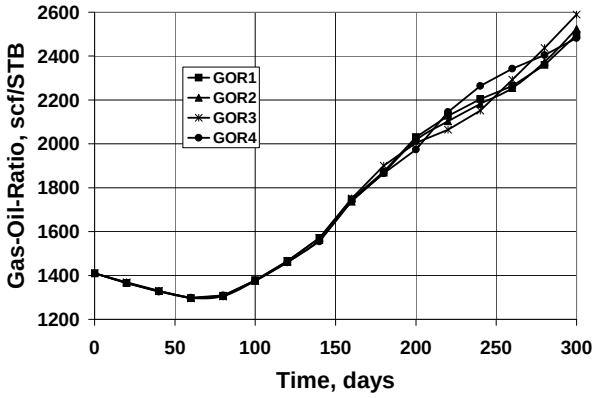


Figure 2.2: The gas-oil ratio.

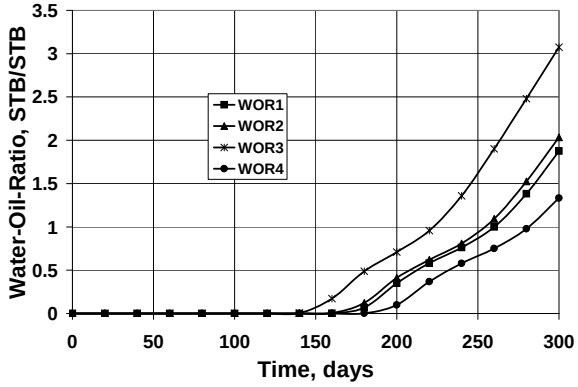


Figure 2.3: The water-oil ratio.

For the example considered, wells produce from only one gridblock so the water-oil ratio and gas-oil ratio, respectively, are given by

$$WOR = \frac{k_{rw}\mu_o B_o}{k_{ro}\mu_w B_w}, \tag{2.47}$$

and

$$GOR = R_s + \frac{k_{rg}\mu_o B_o}{k_{ro}\mu_g B_g}. \tag{2.48}$$

These equations are applied at each producing well and are evaluated using the pressure and saturations of the gridblock containing the well.

For three-dimensional multiphase flow problems, sensitivity coefficients are difficult to understand physically. If one wishes to perform history matching by manually adjusting parameters instead of using a fully automated procedure, the availability of a procedure to compute sensitivity coefficients should prove valuable in those cases where we are unable to apply physical intuition to predict how a change in a model parameter will change production data. The sensitivity of GOR to the permeability field is particularly hard to predict. The reason for this difficulty is that the gas production rate consists of two parts, gas dissolved in the oil phase (reflected by R_s) and the free gas flow rate which is largely controlled by gas relative permeability. If, as should be expected, an increase in permeability in the well's gridblock results in an increase in gridblock pressure, then R_s increases, but gas saturation will typically decrease resulting in a decrease in the production rate of free gas. If the incremental increase in the production rate of dissolved gas is greater than the incremental decrease in the rate of production of free gas, then the increase in permeability will result in an increase in GOR so the sensitivity of GOR to gridblock permeability is positive. On

the other hand if the incremental decrease in the production rate of free gas is greater than the incremental increase in the production rate of dissolved gas, then the sensitivity will be negative. The interpretation of sensitivity coefficients is further complicated when injected water is displacing both oil and gas and flow occurs in both the horizontal and vertical directions. In this case, the sensitivity of WOR and GOR to a gridblock permeability depend on gridblock pressure, water saturation and gas saturation and changes in these variables depend on how a change in permeability affects flow of each phase in all three directions.

2.3.1 Sensitivity to Horizontal Permeability.

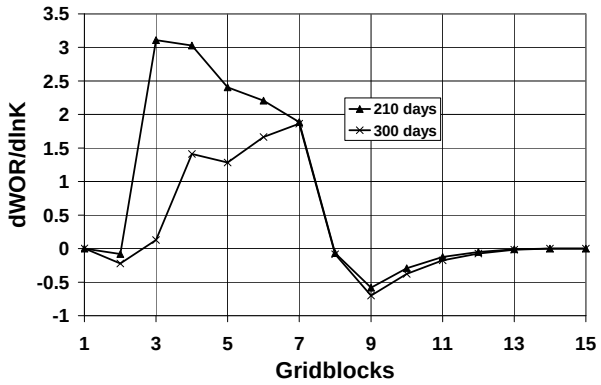


Figure 2.4: Dimensionless sensitivity of WOR at well 1 to horizontal log-permeability of layer 1.

Fig. 2.4 shows a plot of the dimensionless sensitivity of the producing WOR at well 1 to the horizontal log-permeability field of layer 1 at two times after breakthrough. In this figure, and in similar ones presented later, sensitivities are shown along a diagonal line of gridblocks from the upper left corner to the lower right corner of the reservoir as oriented in Fig. 2.1. Thus gridblock 3 pertains to the areal location of well 1, gridblock 8 pertains to the areal location of the injection well, and gridblock thirteen represents the areal location of well 3. As shown in Fig. 2.4, the sensitivity of the water-oil ratio to layer 1 horizontal log-permeability is negative at gridblocks that are near the injection well and between well 3 and the injection well. This makes sense because increasing these permeabilities causes more of the injected water to flow towards well 3 thus decreasing the WOR at well 1. On the other hand, increasing the permeability in the interwell region between well 1 and the

injector causes more of the injected water to flow towards well 1 and increases the velocity of flow. Thus, the water saturation and WOR at well 1 increase. The sensitivity of the producing GOR to these permeabilities (Fig. 2.5) is not so easily explained and the behavior of the dimensionless sensitivity of GOR to horizontal log-permeability in layer 2 (Fig. 2.6) would be extremely difficult to predict based on physical intuition. The results shown in Fig. 2.6 pertain to the diagonal row of gridblocks in layer 2 that correspond to the layer 1 diagonal row of gridblocks shown in Fig. 2.5.

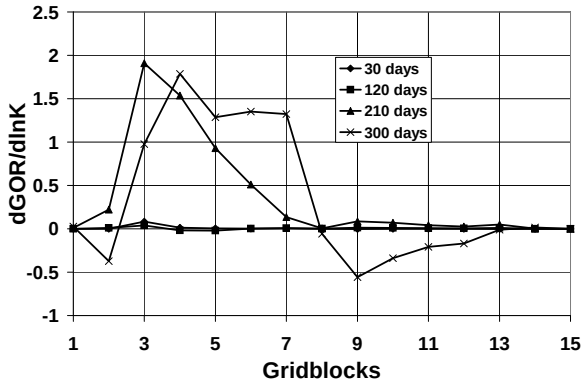


Figure 2.5: Dimensionless sensitivity of GOR at well 1 to horizontal log-permeability of layer 1.

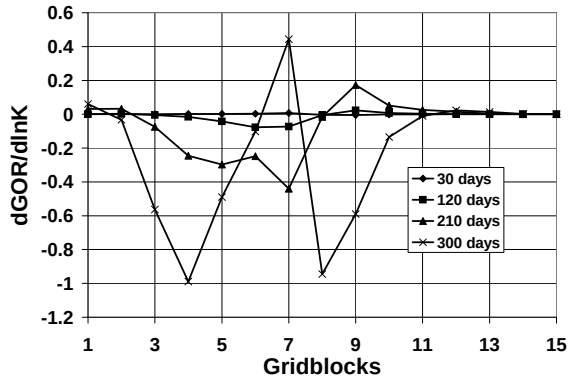


Figure 2.6: Dimensionless sensitivity of GOR at well 1 to horizontal log-permeability of layer 2.

2.3.2 Sensitivity to Vertical Permeability.

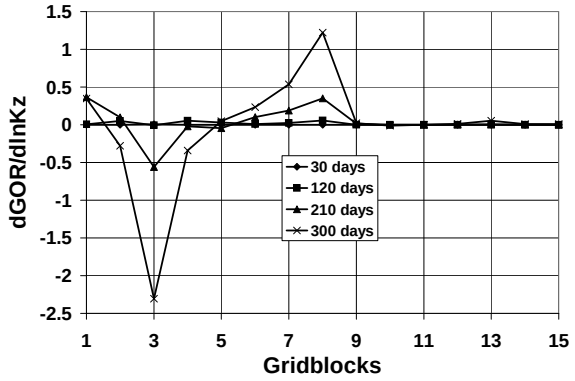


Figure 2.7: Dimensionless sensitivity of GOR at well 1 to vertical log-permeability of layer 2.

Fig. 2.7 shows a plot of the dimensionless sensitivity of the producing GOR at well 1 to layer 2 vertical log-permeability at four times. The results pertain to the same diagonal row of gridblocks as the results of Fig. 2.6. Increasing k_z in gridblock 3 of Fig. 2.7 increases flow from layer 2 to layer 1 at the areal location of well 1 which produces only from layer 1. As this gridblock vertical permeability is increased, a greater percentage of the fluid flowing into the layer 1 gridblock containing well 1 comes from layer 2; at times after breakthrough, this results in a lower gas saturation in the wellblock and a lower GOR. This explains why the sensitivity of GOR to $\ln(k_z)$ for gridblock 3 is negative at 210 and 300 days. Note at the same times, the sensitivity of GOR to $\ln(k_z)$ at the injection well (gridblock 8) is positive. Although increasing this vertical permeability would cause more of the injected water to flow into the top layer and thus increase the producing WOR at well 1, we do not believe that one could predict a priori that this would also increase the GOR for the specific problem under consideration.

The reservoir model under consideration contains only two layers and the layer thicknesses coincide with the height of the two vertical gridblocks used in the finite-difference model. Thus, the reservoir simulators involves only the value of the harmonic average vertical permeability at the boundary between layers. Thus, even though we later present results on the estimates of layer vertical permeabilities, one should recognize that any two sets of layer vertical permeabilities that result in the same value of the harmonic average vertical permeability will yield the same production data.

2.3.3 Sensitivity to Skin Factor.

The flowing wellbore pressure of producing well j is highly sensitive to the skin factor for well j but is insensitive to the skin factor at all other producing wells. Increasing the skin factor at a flowing well results in a decreased wellbore pressure ($\partial p_{wf,j}/\partial s_j < 0$), but has a negligible effect on the well's gridblock pressure. Since producing GOR and WOR are based on the well's gridblock pressure, GOR and WOR are insensitive to the wells skin factor. This indicates that history matching only to GOR or only to WOR data can not resolve well skin factors. Reasonable estimates of well skin factors can be obtained only by history matching pressure data.

Since the injection rate is fixed at the water injection well, the flow rate at the water injection well is insensitive to the skin factor at the injection well. It follows that the wellbore pressure at each producing well is insensitive to the skin factor at the injection well. (This would not be the case if the wellbore pressure was specified as the well constraint at the injection well.)

The injection wellbore pressure is highly sensitive to the skin factor at the injection well. This sensitivity coefficient is positive because as the skin factor increases, a higher wellbore pressure is required to maintain the specified injection rate. The injection well pressure is insensitive to the skin factors of all producing wells because the total flow rate is specified as a well constraint at the producing wells.

2.3.4 Comments.

The sensitivity of a specific production data (e.g., GOR at a specified time) to a particular model parameter (e.g., layer 1 vertical log-permeability for a gridblock containing a producing well) gives a measure of the magnitude of the change in this data that will result from a change in this model parameter. If this sensitivity is small, then we expect that the particular model parameter can not be reliably determined by the particular observed data, i.e., we expect the uncertainty in the model parameter will be not be significantly reduced by history matching the model to this single data point. However, to compare how different types of data affect the estimates of different models parameters, sensitivity results must be scaled properly. If the measurement error is very small, then the range of values of the model parameter that yield an acceptable match of the single observed data will be smaller and we expect the uncertainty in the model parameter to be smaller. Also if the prior variance is very small, then the model parameter is resolved well before history matching the data. Thus, even if the particular data is highly sensitive to the model parameter, we should not expect

history matching to yield a big reduction in the uncertainty. The dimensionless sensitivity coefficients introduced by Zhang et al. (2000)(see Eq. 2.25) attempts to scale sensitivity coefficients to account for measurement errors and the prior variances in model parameters. Qualitatively, we expect that the higher dimensionless sensitivity coefficients correspond to a greater reduction in the uncertainty in model parameters. This simple concept ignores the correlation between model parameters, however. For example, if vertical permeability were nonzero, but no vertical flow occurred in a particular region of the reservoir, production data would be insensitive to k_z in that region. If, however, k_z were strongly correlated to k in that region and the data reduced the uncertainty of k, we would expect the uncertainty in k_z to be reduced also.

2.4 Automatic History Matching Example

2.4.1 The Truth Case.

The truth case is the same one for which sensitivity coefficients were previously calculated. The true skin factor at well 1 through 5 respectively, are given by 3.0, 4.0, 5.0 2.0 and 0.0 respectively. The observed data consists of data obtained by running the reservoir simulator for the truth case to predict reservoir performance for a 300 day time period. At each producing well, we selected 10 WOR data, 10 GOR data and 10 pressure data to use as conditioning data. At the injection well, we selected 10 pressure data to use as conditioning data. The data are uniformly distributed throughout the 300 days time period with the earliest time conditioning data corresponding to $t = 30$ days. We assumed pressure measurement errors to be independent, identically-distributed, normal random variables with mean zero and variance equal to 1 psi^2. GOR measurement errors were modeled similarly except the variance was set equal to 100 (scf/STB)^2. The variances of WOR measurement errors were specified by Eq. 2.6 with $\sigma_{w,\min} = 2.0$ STB/STB, $\epsilon_o = 0.01$ and $\epsilon_w = 0.02$.

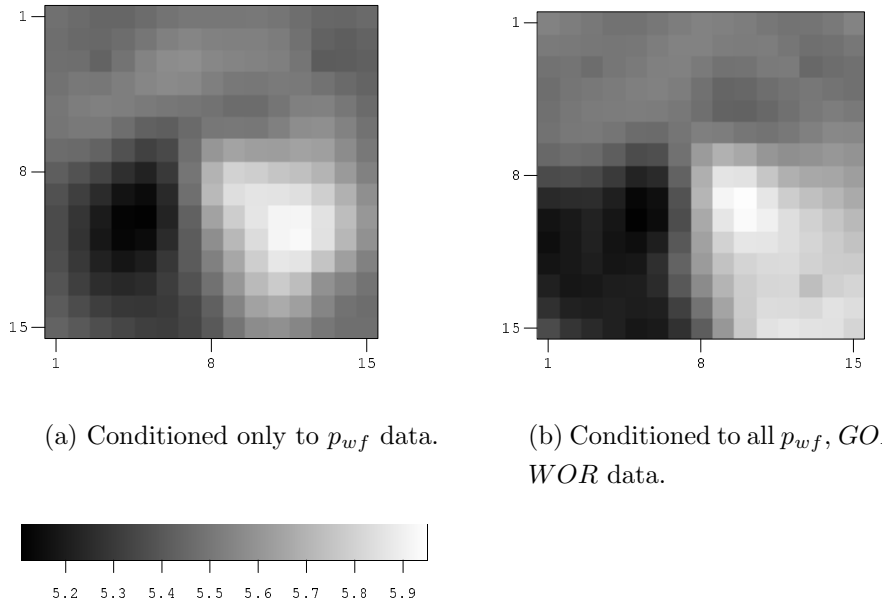
For the top layer, the prior means for $\ln(k)$ and $\ln(k_z)$, respectively, were specified as 5.5 and 3.2 with the variances of both random variable equal to 0.5. For the second layer, the prior means for $\ln(k)$ and $\ln(k_z)$, respectively, were specified as 4.0 and 2.4 with the variances of both random variable equal to 0.5. This information and permeability values for the true model were presented previously in Table 2.1.

The same semivariogram was used for each of the four log-permeability fields. The semivariogram is an isotropic spherical semivariogram with range equal to 160 ft and sill equal to 0.5. As the areal dimensions of simulation gridblocks are $\Delta x = \Delta y = 40$ feet, if

the distance between the centers of two gridblocks at the same elevation is greater than or equal to $4\Delta x$, the two gridblock permeabilities are uncorrelated. There is no correlation between layer 1 permeabilities and layer 2 permeabilities. In each layer, the correlation coefficient between $\ln(k)$ and $\ln(k_z)$ is set equal to 0.7. Each well skin factor is modeled as an independent random variable with mean equal to 2.0 and variance equal to 25.

2.4.2 The MAP Estimate.

As discussed previously, the MAP estimate was generated using the Levenberg-Marquardt algorithm to minimize the objective function of Eq. 2.10. The vector of prior means was used as the initial guess. Note even though each layer actually consists of three zones, horizontal and vertical log-permeability are estimated at each gridblock. This simple model is used only because it makes it easy to evaluate the quality of the estimate. We consider results obtained by history matching only pressure data, pressure plus WOR data, pressure plus GOR data and all three types of data.



(a) Conditioned only to p_{wf} data.

(b) Conditioned to all p_{wf}, GOR and WOR data.

Figure 2.8: Maximum a posteriori estimate of horizontal log-permeability, layer 1.

The left plot in Fig. 2.8 shows the MAP estimate of layer 1 horizontal log-permeability obtained by history-matching only pressure data and the right plot shows results obtained by conditioning to pressure, GOR and WOR data. Comparing results to the truth case shown in Fig. 2.1, we see that the MAP estimate obtained by conditioning to all three types of data

is closer to the truth case. The estimates of layer 2 $\ln(k)$ are not shown but qualitatively look very similar to those shown in Fig. 2.8. Conditioning to pressure, WOR and GOR data gave a better estimate of the layer 2 horizontal log-permeability field than was obtained by conditioning only to pressure.

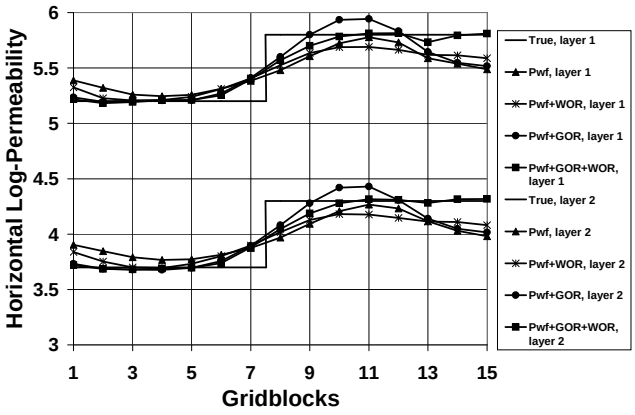


Figure 2.9: MAP estimate of horizontal log-permeability along the line through wells 3 and 4.

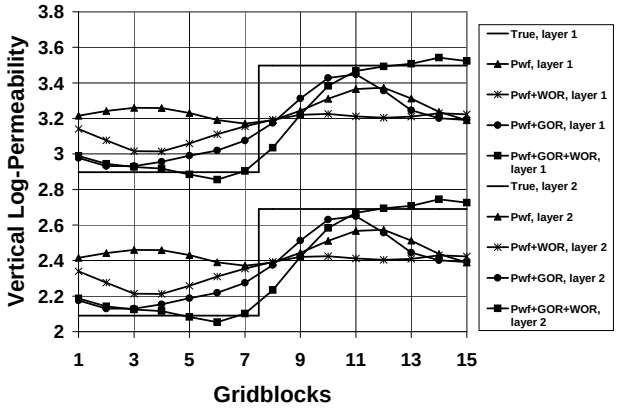
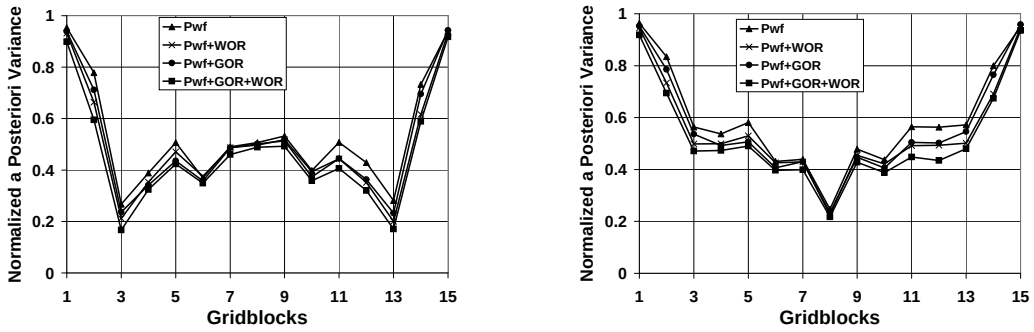


Figure 2.10: MAP estimate of vertical log-permeability along the line through wells 3 and 4.

Fig. 2.9 confirms the preceding results and also shows results obtained by conditioning to pressure and WOR data only and pressure and GOR data only. The results in Fig. 2.9 pertain to results along the line of gridblocks in layer 1 that pass through wells 3 and 4 (see Fig. 2.1, and along the corresponding line of gridblocks in layer 2. Fig. 2.10 shows the cor-

responding estimates of $\ln(k_z)$. In Figs. 2.9 and 2.10 and in similar figures, curves through solid triangular data points refer to results obtained by conditioning only to pressure data, curves through data points denoted by an asterisk refer to results obtained by conditioning only to pressure and WOR data, curves through solid circular data points refer to results obtained by conditioning only to pressure and GOR data and curves through solid square data points represent results obtained by conditioning to all observed data, pressure, GOR and WOR. Considering the overall results, it is clear the worst estimates of the true permeability fields are obtained when only pressure data is history-matched and the best estimates are obtained when the estimate is obtained by history-matching all the pressure, WOR and GOR data. History matching only GOR data plus pressure data gives better estimates of vertical permeability than history matching pressure and WOR data. On a visual basis, it does not appear that history matching pressure plus water-oil ratio data gives significantly better results than matching only pressure data. This conclusion is similar to one reported by Landa and Horne (1997) and Wu et al. (1999).

Conclusions qualitatively similar to those shown in Figs. 2.9 and 2.10 were obtained by examining estimates along a diagonal line of gridblocks in layer 1 that passes through wells 1, 5 and 3 and the corresponding diagonal row of gridblocks in layer 2. Figs. 2.11 and 2.12 show the corresponding normalized a posteriori variances constructed from the a posteriori covariance matrix, C_{MP}.



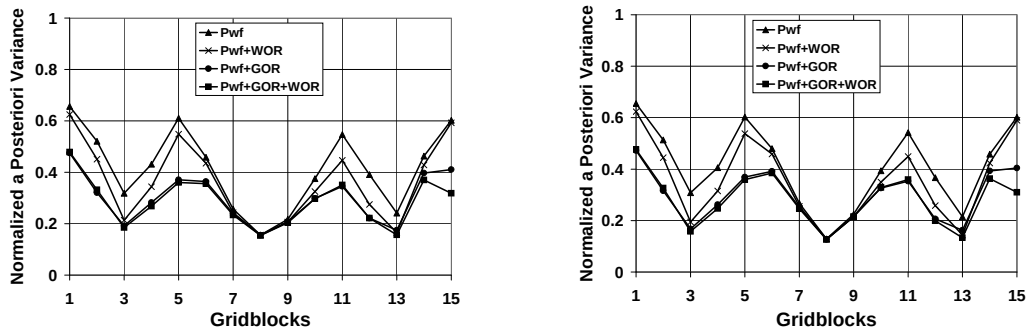
(a) Layer 1 .

(b) Layer 2

Figure 2.11: The normalized a posteriori variance of horizontal log-permeability along the diagonal line through wells 1, 3 and 5.

These variances pertain to the same diagonal rows of gridblocks considered in the sensitivity

coefficient plots of Figs. 2.4-2.7. In these figures, gridblock 3 pertains to the areal location of well 1, gridblock 8 pertains to the areal location of well 5 and gridblock 13 represents the areal location of well 3. The normalized variances at well locations are roughly equal to 0.2 which may be interpreted to mean that we have reduced the uncertainty of these log-permeabilities by eighty percent. Except at the areal location of the injection well (gridblock 8) history matching WOR and/or GOR data plus pressure data reduced the uncertainty (as measured by the normalized a posteriori variance) below the uncertainty obtained by history matching only to pressure data. Note this occurs even at layer 1 gridblock 3 which contains well 1 (see Fig. 2.11(a)) even though the pressure at well 1 is highly sensitive to this gridblock permeability and largely insensitive to other layer 1 $\ln(k)$ values. Note, however, well gridblock $\ln(k)$ and the well skin factor are both unknown and must be resolved by the data. We believe that the fact that both the WOR (Fig. 2.4) and the GOR (Fig. 2.5) are highly sensitive to this gridblock $\ln(k)$ at certain times but insensitive to the well skin factor explains why adding observed GOR and WOR data as conditioning data reduces the uncertainty in $\ln(k)$ at the gridblock containing well 1.



(a) Layer 1 .

(b) Layer 2

Figure 2.12: The normalized a posteriori variance of vertical log-permeability along diagonal line, well 1-3.

As the WOR and GOR (Figs. 2.4 and 2.5) are almost completely insensitive to the horizontal gridblock permeability for gridblock 8 of layer 1, it is not completely surprising that conditioning to WOR and/or GOR plus wellbore pressures does not significantly reduce the uncertainty below that obtained by conditioning only to pressure data. Any additional reduction in uncertainty in $\ln(k)$ obtained at this gridblock by adding WOR and/or GOR

as conditioning data must come from the correlation between this $\ln(k)$ random variable and the log-permeabilities in neighboring gridblocks where the GOR and WOR sensitivity coefficients are non-negligible. It is interesting to note, however, that even though at later times, the GOR (Fig. 2.6) and WOR (not shown) are relatively sensitive to $\ln(k)$ for gridblock 8 of layer 2, the reduction in uncertainty in this horizontal log-permeability is essentially independent of which type of data is history matched as long as pressure data is included as conditioning data. This is because the injection wellbore pressures (not shown) and flowing wellbore pressures at all producing wells are quite sensitive to this gridblock permeability but flowing wellbore pressures are insensitive to the skin factor at the injection well. Thus, using GOR and WOR data in addition to pressure data as conditioning data does not reduce the uncertainty in this gridblock $\ln(k)$ below the level obtained by conditioning only to pressure data.

Estimates of well skin factors obtained by matching various combinations of production data are shown in Table 2.2. Note the best estimates are obtained by history matching pressure, WOR and GOR data.

Table 2.2: The true and estimated skin factors.

Well No.	1	2	3	4	5
True skin factors	3.00	4.00	5.00	2.00	0.00
Initial guess	2.00	2.00	2.00	2.00	2.00
p_{wf}	2.60	3.30	2.97	2.61	-0.19
p_{wf}+WOR	2.67	3.39	3.18	2.22	0.24
p_{wf}+GOR	2.73	3.39	3.54	2.07	0.00
p_{wf}+GOR+WOR	2.83	3.65	4.37	1.96	-0.27

The history matches of some of the observed GOR and WOR data are shown in Fig. 2.13. Solid circular and triangular data points represent the observed data used as conditioning data. Curves through the cross data points indicate the data predicted from the initial guess (m_{prior}) of the model parameters. (Some initial pressure mismatches exceeded 400 psi.) The other solid curves represent predicted data based on the model obtained by simultaneously matching pressure, WOR and GOR data. Matches of similar quality were obtained for all wells for all pressure, WOR and GOR observed data.

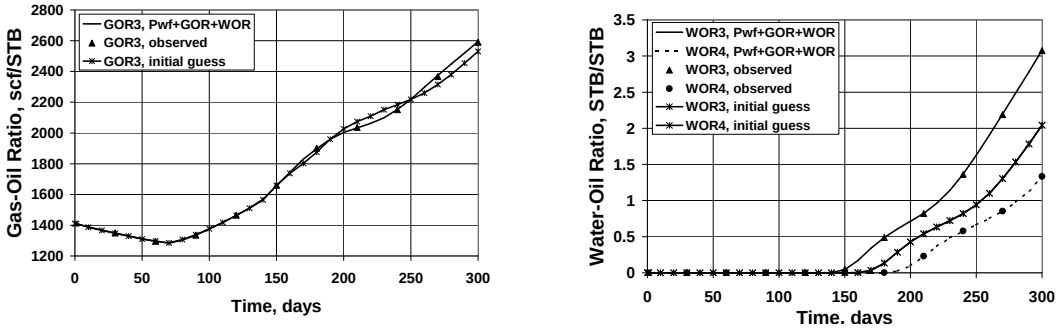


Figure 2.13: The GOR match (left)and WOR match (right) for the model conditioned to p_{wf}, GOR, and WOR data.

2.4.3 Remarks.

The results and discussion of this history matching example serves to illustrate the following observations. (i) If particular production data are completely insensitive to a model parameter, changing this model parameter does not change the values of production data predicted by reservoir simulation. Thus, conditioning to these data can reduce the uncertainty in this model parameter only by reducing the uncertainty in the parameters which are correlated with this model parameter. The reduction in uncertainty due to either a prior correlation (determined by C_M) or a posterior correlation (determined by C_{MP}) can not be predicted by examining sensitivity coefficients. (ii) If a set of model parameters is well resolved by a given data set, adding a second set of conditioning data to the first may not give an additional reduction in uncertainty even if the second set of data is highly sensitive to the model parameters.

Regarding observation (i), one might guess that the good estimates of $\ln(k_z)$ obtained (see Fig. 2.10) are partially due to a high correlation between the random functions $\ln(k)$ and $\ln(k_z)$ (correlation coefficient equal to 0.7) and the fact that horizontal log-permeability is fairly well resolved by the data. However, in dimensionless form, the sensitivity of production data to horizontal log-permeability and the sensitivity of production data to vertical log-permeability variables show peaks of similar magnitude; compare, for example Fig. 2.6 with Fig. 2.7. Thus, it is reasonable to conjecture that the good estimates of $\ln(k_z)$ obtained in Fig. 2.10 are not solely due to the correlation between $\ln(k)$ and $\ln(k_z)$. To investigate this conjecture, we generated the MAP estimate assuming that horizontal and vertical permeability are not correlated in the prior model. The resulting corresponding MAP estimate of $\ln k_z$ did not provide a significantly worse estimate of the true $\ln(k_z)$ field than the MAP

estimate shown in Fig. 2.10.

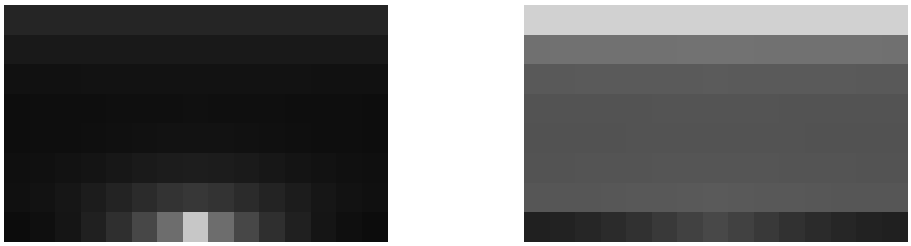
2.5 Comments on Sensitivity Coefficients

Although our objective is to develop methodology for automatic history matching, it is important to note that sensitivity coefficients help understanding reservoir physics and can be useful even when history matching is done manually. In the manual approach, the reservoir engineer has to decide which parameters to adjust and the magnitude of the adjustment based on physical intuition and trial and error. For three-phase problems where gas production or injection is significant, our experience suggest that physical intuition is insufficient to predict how modifications to the permeability field will affect the producing gas-oil ratio. Calculated sensitivity coefficients, however, provide a clear indication of how changes in gridblock permeabilities will affect the the producing gas-oil ratio. To illustrate these points, we consider a simple two-dimensional problem.

2.5.1 A Cross Section Example.

We consider a vertical cross section of a solution-gas-drive reservoir. The simulation grid consists of 15 gridblocks in the x-direction (horizontal direction) and 8 gridblocks in the z-direction (vertical direction). No flow occurs in the y-direction and only one gridblock is used in the y-direction. The dimension of each gridblock is 40 feet in the x-direction, and 30 feet in the z-direction. The horizontal and vertical permeabilities are 40 mD and 4 mD in each gridblock, respectively. The porosity field is uniform with a value of 0.22. A single well located at the bottom center (gridblock (8,8)) is produced at a constant oil rate of 50 STB/d. Water is immobile with initial water saturation equal to irreducible water saturation, $S_{iw} = 0.2$. Initial oil saturation is equal to 0.8. The initial reservoir pressure is 4500 psi in the bottom row of gridblocks. Initial bubble point pressure is equal to 4417 psi. The reservoir pressure drops below bubble point soon after the beginning of the simulation run. Critical gas saturation is 0.05. After 30 days of production, gas saturation is still below critical (Fig 2.14(a)), whereas after 400 days of production (Fig 2.14(b)), gas saturation ranges from 0.26 to 0.33 and is highest in the top most row of gridblocks. This indicates that counter current flow has occurred; some of the gas that has been liberated from solution has flowed to the top of the reservoir due to the effect of gravity.

For this type of problem it is difficult to predict a priori whether an increase in a gridblock



(a) Gas saturation at 30 days. (b) Gas saturation at 400 days.

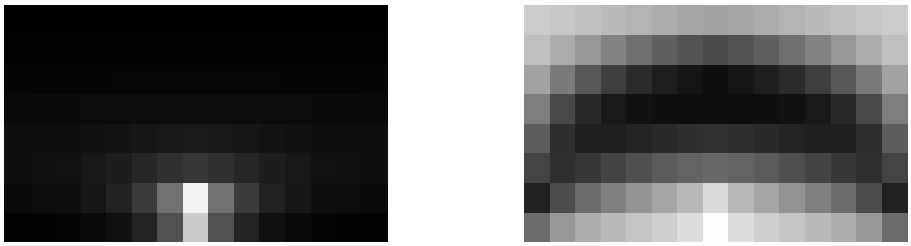
Figure 2.14: Gas saturation at early and late times.

permeability will increase or decrease the producing gas-oil ratio (GOR) which is given by

$$GOR = R_s + \frac{k_{rg}\mu_o B_o}{k_{ro}\mu_g B_g}, \tag{2.49}$$

where R_s denotes the dissolved GOR, B_o and B_g, respectively, denote the oil and gas formation volume factor, μ_o and μ_g, respectively, denote the oil and gas viscosity, and k_{ro} and k_{rg}, respectively, denote oil and gas relative permeability. All terms in Eq. 2.49 are evaluated at the pressure and saturation of the gridblock containing the producing well. Increasing permeability tends to result in higher gridblock pressure which increases the dissolved gas part of the producing GOR, but results in a decrease in gas saturation which tends to decrease the free gas part of the producing GOR. Additional difficulty in predicting the behavior arises from the interaction between gravity and viscous forces. It is not clear a priori whether increasing a particular gridblock permeability will cause more gas to flow to the top, or cause more gas to flow to the producing well.

At 30 days, the pressure in the reservoir is below the bubble point pressure (Fig. 2.14(a)), but all gridblock gas saturations are below critical gas saturation. As a result of the low gas saturation, all produced gas comes from the dissolved gas in the oil phase. An increase in the horizontal and vertical permeabilities results in higher pressure, and as a result, more gas will be dissolved in the oil. Because no free gas is produced, increasing the dissolved GOR results in an increase in the producing GOR. Thus the sensitivity of GOR to vertical permeability is positive everywhere; see Fig. 2.15(a). The sensitivity of pressure to vertical permeability is highest in the region of the wellbore gridblock because vertical pressure gradients are largest there. The sensitivity is somewhat higher in the gridblock above the well completion than in the wellbore gridblock itself because the permeability of that gridblock affects two gridblock transmissibilities while the vertical permeability of the wellbore gridblock affects only one. After 400 days of production, gas saturation is above critical everywhere, and because of



(a) Sensitivities at 30 days. (b) Sensitivites at 400 days.

Figure 2.15: Sensitivity of producing GOR to vertical permeability

gravity segregation, gas saturation is highest in the upper layers (Fig. 2.14(b)). Fig. 2.15(b) shows the sensitivity of producing GOR to vertical permeability at 400 days. The sensitivity of GOR to gridblock vertical permeability is positive for gridblocks near the well indicating that increasing vertical permeability in this region will result in an increase in GOR. On the other hand, for gridblock slightly above the center of the reservoir, the sensitivities are large in magnitude but negative. This indicates that increasing k_z in these gridblocks will result in increased transport of gas to the top of the reservoir and a decrease in the producing GOR.

2.6 LM Versus Gauss-Newton

As noted before, the MAP estimate that was presented was obtained using a Levenberg-Marquardt algorithm. A limited set of experiments indicates that the Levenberg-Marquardt method has better convergence properties than the standard Gauss-Newton method especially for cases where the initial data mismatch is large. Here, we compare these two algorithms for a two-dimensional, two-phase (oil-water) flow problem. The true model for the 2D example is shown in Fig. 2.16. The value of log-permeability is 3.7 (40.4 md) in the lower left quadrant, 4.3 (73.7 md) in the lower right quadrant and 3.9 (49.4 md) in the upper part. The porosity is constant throughout the model with value 0.22. A $21 \times 21 \times 1$ grid is used for reservoir simulation. Well 5 is a water injection well which is located at the center of the reservoir (in gridblock (11,11)). Wells 1 through well 4 are producing wells and are located, respectively, in gridblocks $(4, 4)$, $(18, 4)$, $(18, 18)$ and $(4, 18)$. At well 5, water is injected at rate of 785 STB/D. Each producing well produces at total fluid rate of 200 RB/D. We generated eight p_{wf} observed data for each producing well uniformly spaced in a 320 day period. The MAP estimate is conditioned to the p_{wf} data only.

For this problem, the Gauss-Newton method required 18 iterations to converge with an

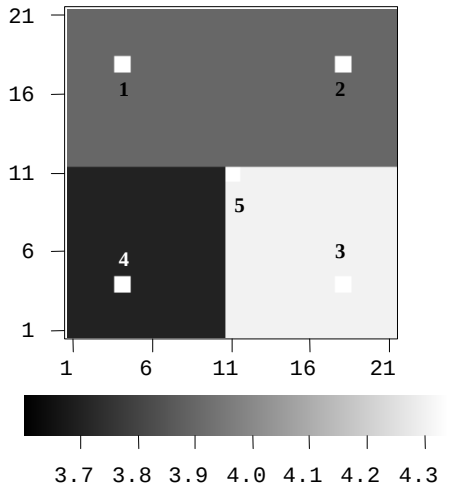


Figure 2.16: The true log-permeability field.

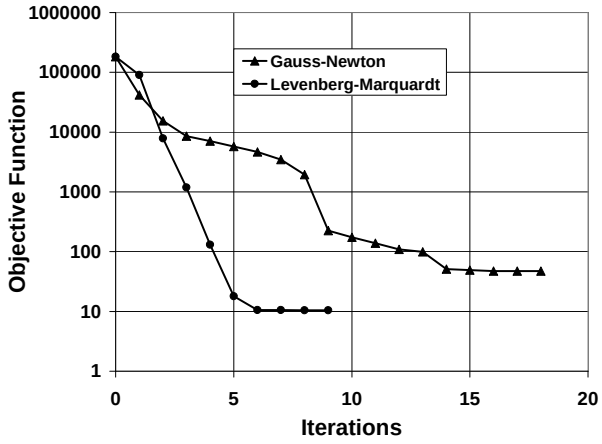
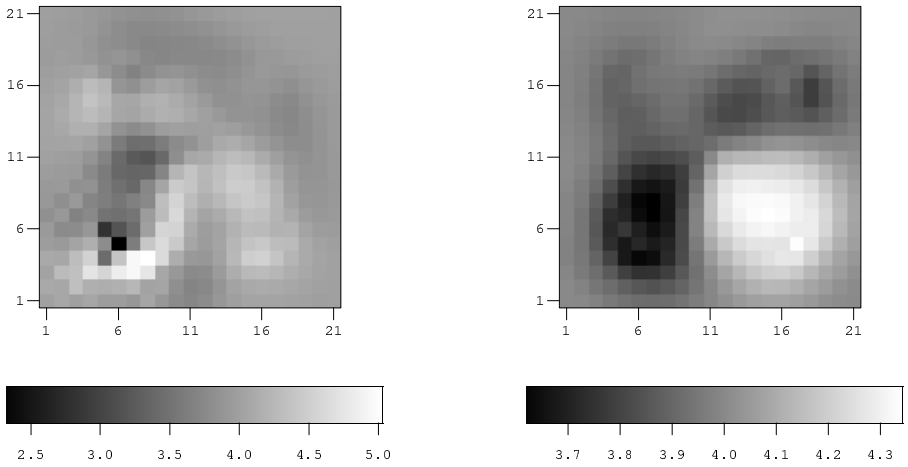


Figure 2.17: The convergence rate of LM and GN.

objective value function of about 60 at convergence, whereas the Levenberg-Marquardt algorithm converged in 9 iterations with an objective function value of about 10 at convergence; see Fig. 2.17. Even more importantly, the Levenberg-Marquardt method converged to a reasonable estimate (Fig. 2.18(b)), of the true log-permeability field of Fig. 2.16. On the other hand, the MAP estimate generated from Gauss-Newton method (Fig. 2.18(a)) is very rough and is quite different from the true model. For example, in the lower left quadrant of the reservoir, the true value of log-permeability is 3.7, but in the results obtained from the Gauss-Newton method, gridblock values of log-permeability range from from 2.5 to 5.0 in



(a) Gauss-Newton method.

(b) Levenberg-Marquardt method.

Figure 2.18: Comparison of MAP estimate of log-permeability using Gauss-Newton and Levenberg-Marquardt algorithms.

this quadrant.

2.7 Application of Quasi-Newton and Conjugate Gradient Methods

Here, we explore the conjugate gradient and quasi-Newton (variable metric) methods. The advantage of these methods is that they require only the computation of the gradient of the objective function with respect to the model parameters. The sensitivity coefficient matrix G is not needed. Our results suggest that these methods can lead to considerable reductions in computer time and memory required for large scale history matching problems. To date, however, we have only implemented thess procedures to history match data from single phase gas reservoirs.

2.7.1 Quasi-Newton Methods

The well known Newton's equation is given by

$$H(m^l)\delta m^{l+1} = -g(m^l), \tag{2.50}$$

where the $H(m^l)$ and $g(m^l)$, respectively, denote the second derivative (Hessian matrix) and the gradient of the objective function evaluated at m^l and l is the iteration index. With either the objective function of Eq. 2.10 or the one of Eq. 2.14, the Hessian matrix is given by

$$H_l = C_M^{-1} + G_l^T C_D^{-1} G_l \tag{2.51}$$

where G_l is the sensitivity matrix evaluated at m^l. As noted before, if both the number of model parameters and the number of data are large, the evaluation of G_l is computationally expensive. In quasi-Newton methods, H_l^{-1} is approximated by a symmetric positive definite matrix D_l which is corrected or updated from iteration to iteration. Different quasi-Newton methods use different formulas to calculate D_{l+1} from D_l. All updating formulas satisfy the quasi-Newton condition given by

$$D_{l+1} y_l = s_l, \tag{2.52}$$

where $y_l = g_{l+1} - g_l$ and $s_l = m^{l+1} - m^l$. Various possible updating formulas honor this quasi-Newton condition. In our procedure, we use Broyden-Fletcher-Goldfarb-Shanno (BFGS) correction equation given by

$$D_{l+1} = D_l + \frac{s_l s_l^T}{s_l^T y_l} - \frac{D_l y_l y_l^T D_l}{y_l^T D_l y_l} + v_l v_l^T, \tag{2.53}$$

where

$$v_l = (y_l^T D_l y_l)^{1/2} \Big(\frac{s_l}{s_l^T y_l} - \frac{D_l y_l}{y_l^T D_l y_l} \Big). \tag{2.54}$$

The limited memory BFGS (LBFGS), which uses a limited number of previous vectors (g_l and s_l) to construct the inverse Hessian approximation at each iteration, can be applied to large scale problems. In our work, the LBFGS algorithm proposed by Nocedal (1980) was implemented and applied. The BFGS or LBFGS algorithm is given below:

Step 1 Initialization

Provide an initial guess of the model m_0, calculate the objective function corresponding to m_0 and evaluate the gradient of the objective function at m_0, set l=0.

Step 2 Calculate the search direction $d_l = -D_l g_l$.

Step 3 Calculate the step size α_l.

Step 4 Update the model $m^c = m^l + \alpha_l d_l$.

Step 5 Calculate the objective function based on m^c.

Step 6 If $O(m^c) < O(m^k)$, set $m^{l+1} = m^c$, $l = l + 1$ and go to step 7;
otherwise cut the step size by a specified factor and go to step 4.

Step 7 Determine if the stopping criteria are satisfied or not. If satisfied, then stop; otherwise go to step 2.

Note that when we implement BFGS, we form the inverse Hessian approximation explicitly, whereas in LBFGS we calculate the product of $H_l g_l$ directly and do not store or calculate the full matrix D_l; see Nocedal (1980).

Convergence Criteria

In our results, the following stopping criteria are used to terminate the algorithm:

1. $$\frac{\mid O^{l+1} - O^{kl} \mid}{O^l + 10^{-14}} < \varepsilon \tag{2.55}$$
where l denotes the iteration index. For Gauss-Newton and Levenberg-Marquardt algorithm, we used $\varepsilon = 10^{-3}$. When we used the same value for the other algorithms, the algorithm frequently obtained a model which satisfied Eq. 2.55 but not Eq. 2.15. This problem was avoided by setting $\varepsilon = 10^{-7}$.

2. Specify a maximum iteration number. If the number of iterations exceeds the specified number, we terminate the algorithm. Here, the maximum number of iterations is specified as 100. Note that reaching the maximum number of iterations does not imply that the algorithm has converged. However, all the examples we tested converged in fewer than 100 iterations even though we used $\varepsilon = 10^{-7}$ in Eq. 2.55.

Scaling

Generally speaking, scaling is obtained by multiplying the old D^l by a factor γ and then substituting γD^l instead of D^l itself into the update equation to calculate D^{l+1}. The purpose of scaling the matrix D_l is to decrease the condition number of matrix $R_l = H_l^{1/2} D_l H_l^{1/2}$ where H_l is the true Hessian matrix, and therefore to improve the convergence behavior of BFGS. When D_l is close to H_l^{-1} which is the inverse of the true Hessian, R_l will be close to the identity matrix and the condition number of R_l will be close to 1 which is ideal. There are many options we can choose to perform scaling. Shanno and Phua (1978) and Yang and Watson (1988) use the scaling factor $\gamma = s^T y/(y^T H y)$ and only scale the initial matrix D_0, in their work. Oren (1973) and Oren (1974a) provided several switching rules which are

used to calculate the scaling factor. In our implementation, at iteration l, the scaling factor depends on the value of τ_l where τ_l can be calculated by

$$\tau_l = \frac{s_l^T D_l^{-1} s_l}{s_l^T y_l} = -\frac{\alpha_l g_l^T s_l}{s_l^T y_l} = \frac{g_l^T s_l}{g_l^T D_l y_l}. \tag{2.56}$$

If $\tau_l < 1$ we let scaling factor γ_l equal to τ_l, otherwise we let γ_l equal to σ_l where

$$\sigma_l = \frac{s_l^T y_l}{y_l^T D_l y_l}.$$

In LBFGS, only D_0 is explicitly provided, and D_l for $l \geq 1$ is never calculated directly. So when we use the first and the last expression of Eq. 2.56 to calculate τ_l, we use D_0 instead of D_l.

2.7.2 Conjugate Gradient Method

It is well known that the conjugate gradient method can be applied to minimize non-quadratic objective functions. Although the method has been applied for the history matching of production data (see, for example Makhlouf et al. (1993)), its slow rate of convergence has precluded its use in large scale history matching problems. The success of the conjugate gradient method for nonlinear optimization depends on whether we are able to construct a good preconditioner. A good preconditioning matrix at the lth iteration is a matrix M_l which is a good approximation to the Hessian H_l so that

$$M_l^{-1} H_l \approx I. \tag{2.57}$$

For our problem, the search direction obtained by the Gauss-Newton method is calculated from Eq. 2.16 with $\lambda = 0$ and the Hessian at the lth iteration is given by

$$H_l = C_M^{-1} + G_l^T C_D^{-1} G_l. \tag{2.58}$$

An optional preconditioner for the conjugate gradient method would be

$$M_l = H_l, \tag{2.59}$$

but the conjugate gradient method requires calculating

$$p_l = M_l^{-1} r_l, \tag{2.60}$$

at each iteration, If $M_l = H_l$, Eq. 2.60 requires the same computational effort as the direct application of Newton's method and does not improve computational efficiency. If we choose $M_l = C_M^{-1}$, however, then Eq. 2.60 becomes

$$p_l = C_M r_l. \tag{2.61}$$

So the calculation of p_l from r_l requires only multiplication by the prior covariance matrix C_M. Kalita (2000) and Kalita and Reynolds (2000) considered the problem of conditioning a gas reservoir model to well test pressure data by automatic history matching. Both the Gauss-Newton method and the conjugate gradient method with C_M^{-1} as the preconditioner were used to minimize the relevant objective function (Eq. 2.10 or Eq. 2.14). Kalita's results indicate that the conjugate gradient method was not always more efficient than the Gauss-Newton method. Moreover, in most cases, the conjugate gradient method converged to a value of the objective function which was significantly higher than the converged value of the objective function obtained by the Gauss-Newton method. In our work, we show that these difficulties can be overcome by using a better preconditioner.

In the preconditioned conjugate gradient algorithm, the preconditioning matrix M_l is used only in an equation like Eq. 2.60. Thus, it is preferable to estimate M_l^{-1} directly instead of estimating M_l. We would like M_l^{-1} to be an approximation for the inverse Hessian. This suggests that M_l^{-1} can be constructed from a quasi-Newton method. Our work indicated that $D_l \approx H_l^{-1}$ calculated from either a BFGS or a scaled limited memory BFGS (LBFGS) method provides preconditioners superior to using $D_l \approx C_M$. The LBFGS preconditioner can be implemented in a way so that we do not need to calculate D_l directly; instead, we simply form the product $D_l r_l$ when needed in computations like the one of Eq. 2.60.

2.8 Evaluation of Computational Efficiency

Here we assess the computational efficiency of GN (Gauss-Newton), LM (Levenberg-Marquardt), PCG (preconditioned conjugate gradient), BFGS and LBFGS. In the evaluation of computational efficiency, we count only the number of adjoint solutions and the number of reservoir simulation runs required by each method. Moreover, we count one adjoint solution over the total time interval of a simulation run as one equivalent simulation run. We do not keep track of the computational effort incurred when a proposed model update is rejected because it results in an increase in the objective function. We only keep track of the number of iterations which correspond to a decrease in the objective function.

In GN and LM, if the data are evenly distributed in the time domain, the computational cost of calculating sensitivity of all data to all model parameters requires $(N_d + 1)/2$ adjoint solutions which is equivalent to $(N_d + 1)/2$ simulation runs. GN and LM require one additional simulation run to calculate the objective function. The new objective function value is compared with the old objective function values. If the objective function decreases, the new model will be accepted. Otherwise the new model will be rejected and the iteration is

repeated. So a total of $(N_d+1)/2+1$ simulation runs are needed to accomplish one GN or LM iteration.

In LBFGS and PCG, the total computational cost of implementing one iteration is equivalent to 3 simulation runs, which included one equivalent simulation run for calculating the gradient of the objective function by using the adjoint method, one simulation run for calculating the step size by using Newton-Raphson method to do the line search and another simulation run for calculating the objective function. Thus, LBFGS and PCG are $((N_d+1)/2+1)/3$ times faster than GN and LM for each iteration. For example if we have 1000 data, LBFGS and PCG will be roughly 167 times faster than GN or LM. In terms of the total time, if GN or LM requires n_1 iterations to converge on average, while LBFGS or PCG needs n_2 iterations to converge on average, then LBFGS or PCG will be t times faster than GN or LM where

$$t = \frac{n_1}{\frac{n_2}{((N_d+1)/2+1)/3}} = \frac{n_1}{n_2} \times \frac{(N_d+1)/2+1}{3}. \tag{2.62}$$

Although BFGS requires slightly more time than LBFGS and PCG to perform the matrices operations involved in the update equation, it is the memory requirement that makes the standard BFGS method inferior to LBFGS and PCG for large scale problems.

2.8.1 Memory

For large scale problems, the memory required by an optimization algorithm is also a key issue that needs to be considered. Because we are only concerned with the difference between algorithms, we only consider the memory used by the algorithm itself. Table 2.3 gives a rough estimate of the number of double precision real numbers used by each algorithm. N_d is the number of data, N_m is the number of model parameters, and L is the number of previous vectors used in the LBFGS algorithm. For convenience, we use one memory unit to stand for the memory occupied by one double precision real number. In the GN or LM method, $(2+2N_d)\times N_m$ (δm, one N_m-dimension work array, sensitivity coefficient matrix G and $C_M G^T$) memory units are used. In PCG, $6\times N_m$ ($6\times N_m$: s, dir, r, $m-m_{pr}$, ∇O_d, one work array) memory units are used. In BFGS or SBFGS $(7+N_m)\times N_m$ ($7\times N_m$: v_k, $m-m_{pr}$, δm, ∇O_d, y_k, dir, one work array; inverse Hessian approximation matrix) memory units are used. In LBFGS, $(4+2\times L)\times N_m$ ($4\times N_m$: δm, diagonal inverse Hessian approximation, dir, one work array; y_k and s_k for $k=1,2,\cdots,L$) memory units are used. When the number of data and the number of model parameters are both large GN, LM BFGS and SBFGS use extremely large amounts of memory which make them impractical. LBFGS uses

slightly more memory than PCG depending on how big L is. From the above comparison, the LBFGS and PCG algorithm seems more promising than the other methods we have tried for large scale problems. The memory used by both methods are independent of the number of data.

Table 2.3: Memory used by each algorithm

	No. of DP real numbers
GN/LM	$(2+2\times N_d)\times\ N_m$
PCG	$6\times N_m$
BFGS/SBFGS	$(7+N_m)\times N_m$
LBFGS	$(4+2\times \mathrm{L})\times N_m$

2.9 Preliminary Results

In order to investigate the reliability and computational efficiency of the optimization algorithms discussed previously, we consider history matching data from a synthetic three-dimensional gas reservoir. The reservoir is 2000 ft $\times$ 2000 ft $\times$ 40 ft. Gridblock sizes are $\Delta x = \Delta y = 100$ ft and $\Delta z = 10$ ft. Thus there are 20, 20 and 4 gridblocks in the $x-$, $y-$ and $z-$directions respectively.

A spherical variogram was used to generate the prior covariance matrix. The correlated lengths in the $x-$, $y-$ and $z-$ direction are 400 ft, 200 ft and 10 ft respectively. Horizontal permeability, vertical permeability, porosity and skin factor are the model parameters for this example. We assume that porosity field is correlated with horizonal permeability field and the correlation coefficient is 0.7 and that vertical permeability field is uncorrelated with the porosity and horizonal permeability fields. The prior information for the model parameters are given in Table 2.4.

The initial pressure is 3230 psi. All six boundaries are assumed to be no-flow boundaries. The reservoir is produced by two completely penetrating wells. Well 1 is located in areal gridblock (5,5) and well 2 is located in areal gridblock (15,15). Well 1 was shut in for two days and then was produced at the rate of 4×10^4 Mscf/day for two days. Well 2 produced at the rate of 3.5×10^3 Mscf/day for the first two days and was then shut in for the following two days. Fig. 2.19 shows the pressure response of the two wells. We used 22 measured data from each well as conditioning data. Thus the total number of data to be history matched is 44. The number of model parameters is $20 \times 20 \times 4 \times 3 + 8 = 4808$. The observed data

Table 2.4: Prior information of model parameters

	Mean	Variance
$\ln(k_x)$	4.0	0.5
$\ln(k_z)$	-2.9	0.5
ϕ	0.25	0.002
S_{skin}	4.0	0.0001

are obtained by adding random noise to the simulated pressure data predicted from the true reservoir.

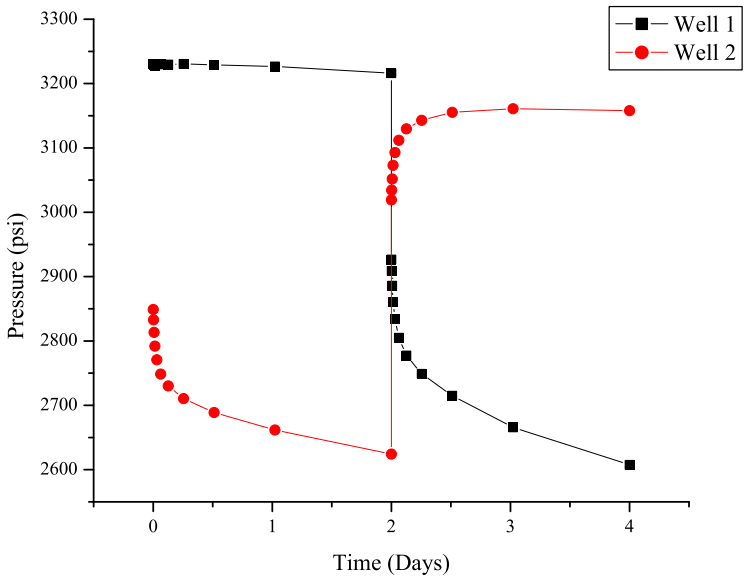


Figure 2.19: Pressure response from the true model.

2.9.1 Comparison of Six Methods

Six algorithms, Gauss-Newton (GN) with restricted step, Levenberg-Marquardt (LM), preconditioned conjugate gradient (PCG), BFGS, scaling BFGS (SBFGS) and limited memory BFGS (LBFGS), are tested and the behavior of them is compared. In LM, we simply use the value of 1000 for the initial damping factor and multiply by 10 when the objective function increases and divide by 10 when the objective function decreases. In PCG, the full matrix C_M was chosen as the preconditioner and fixed for each iteration. In BFGS and scaling BFGS, the full prior covariance matrix C_M is used as the initial inverse Hessian approxi-

mation. In scaling BFGS, we only scale the initial inverse Hessian approximation and it is scaled by a factor of γ which is equal to τ when $\tau = g_1^T s_1 / g_1^T D_0 y_1 < 1$ and is equal to $\sigma = s_1^T y_1 / y_1^T D_0 y_1$ when $\tau \geq 1$. The LBFGS we used is the algorithm proposed by Nocedal (1980). In the LBFGS method, at most 30 previous vectors were used to construct the inverse Hessian approximation D_l, and at each iteration, D_0, which is an identity matrix, is scaled by a factor of $s_l^T y_l / (y_l^T y_l)$ where l represents the l^{th} iteration. All 6 algorithms were applied to the same 10 unconditional realizations of data and the model, when doing history matching, i.e. the same m_{uc} and d_{uc} were used in the objective function of Eq. 2.14.

Table 2.5: Comparison between algorithms

		R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	Average
GN	Obj.	37	31	21	33	43	28	27	38	40	33	33.1
	No. Iter.	7	13	6	8	12	8	12	12	6	6	9
LM	Obj.	38	30	21	33	43	28	27	38	40	34	33.2
	No. Iter.	8	14	12	8	13	13	21	8	14	10	12
BFGS	Obj.	59	52	35	41	70	41	39	53	54	87	53
	No. Iter.	33	17	18	19	33	38	18	42	35	52	30.5
SBFGS	Obj.	77	50	33	43	56	36	43	105	64	38	54.5
	No. Iter.	13	51	25	38	17	41	29	14	32	31	29.1
LBFGS	Obj.	53	47	63	131	138	44	66	174	87	42	84.5
	No. Iter.	36	41	17	42	27	50	10	39	34	43	33.9
PCG	Obj.	153	146	70	94	347	42	184	213	230	45	152
	No. Iter.	12	20	11	23	19	64	5	17	19	23	21.3

Table 2.5 shows the objective function value at convergence and the number of iterations required to obtain convergence. In terms of the number of iterations, the GN and LM are the best algorithms and the objective function converges to a small value (33 on average) for each of the 10 realizations. Both the GN and LM methods, however, require considerable work for each iteration due to evaluating the sensitivity of each data to all model parameters. From the results shown in Table 2.5, we can make these conclusions.

1. In terms of total machine time, algorithms which only require the gradient of the objective function (especially LBFGS and PCG) are much faster than GN and LM method. As discussed previously, theoretically, LBFGS and PCG should be roughly $[(N_D + 1)/2 + 1]/3$ times faster than GN and LM per iteration. In this example,

we history matched 44 data to generate each realization. Thus, LBFGS and PCG algorithms should be $[(N_d+1)/2+1]/3 \approx 8$ times faster than GN and LM per iteration.

2. Based on results of Zhang et al. (2000), we believe that the average value of the objective function at the minima should be on the order of $N_d = 44$ and should not exceed $N_d+5\sqrt{2N_d} = 91$. However the average objective function value at convergence is higher than this value for the preconditioned conjugate gradient method where the preconditioner is the full matrix C_M. Thus, it appears that the PCG method does not yield an appropriate history matched model. Moreover, LBFGS converges to a high value (close to 91). Thus we need to seek modified algorithms that will have better convergence properties.

To further investigate the convergence property of LBFGS, we have tried different initial H_0 and different scaling scheme, which are described below.

OPT1. Use the identity matrix as the initial inverse Hessian approximation; scale D_0 for each iteration by a factor γ, i.e.,
$\hat{D}_0 = \gamma D_0 = \gamma I = s_l^T y_l/(y_l^T y_l) I = \sigma$.

OPT2. Use the identity matrix as the initial inverse Hessian approximation; scale D_0 for each iteration by a factor γ which is determined by the following scheme:
$\gamma = \tau$ for $\tau < 1$ where $\tau = g_l^T s_l/(g_l^T y_l)$,
$\gamma = \sigma$ otherwise.

OPT3. Use the identity matrix as the initial inverse Hessian approximation; scale D_0 for each iteration by a factor γ which is determined by the following scheme:
$\gamma = \tau$ for $\tau < 1$ where $\tau = -\alpha_l g_l^T s_l/(s_l^T y_l)$,
$\gamma = \sigma$ otherwise.

OPT4. Use the identity matrix as the initial inverse Hessian approximation; scale D_0 for each iteration by a factor γ which is determined by the following scheme:
$\gamma = \tau$ for $\tau < 1$ where $\tau = s_l^T s_l/(s_l^T y_l)$,
$\gamma = \sigma$ otherwise.

OPT5. Use the diagonal of C_M instead of the identity matrix as the initial inverse Hessian approximation and only scale the initial matrix by a factor γ which is determined by the following scheme:
$\gamma = \tau$ for $\tau < 1$ where $\tau = g_1^T s_1/(g_1^T D_0 y_1) = -\alpha_1 g_1^T s_1/(s_1^T y_1) = s_1^T D_0^{-1} s_1/(s_1^T y_1)$,
$\gamma = \sigma$ otherwise.

OPT6. Use the diagonal of C_M as the initial inverse Hessian approximation; scale D_0 for each iteration by a factor γ which is determined by the following scheme:
$\gamma = \tau$ for $\tau < 1$ where $\tau = g_l^T s_l / (g_l^T D_0 y_l)$,
$\gamma = \sigma$ otherwise.

OPT7. Use the diagonal of C_M as the initial inverse Hessian approximation; scale D_0 for each iteration by a factor γ which is determined by the following scheme:
$\gamma = \tau$ for $\tau < 1$ where $\tau = -\alpha_l g_l^T s_l / (s_l^T y_l)$,
$\gamma = \sigma$ otherwise.

OPT8. Use the diagonal of C_M as the initial inverse Hessian approximation; scale D_0 for each iteration by a factor γ which is determined by the following scheme:
$\gamma = \tau$ for $\tau < 1$ where $\tau = s_l^T D_0^{-1} s_l / (s_l^T y_l)$,
$\gamma = \sigma$ otherwise.

Table 2.6: Comparison of LBFGS algorithm with different options

		R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	Average
OPT1	Obj.	53	47	63	131	138	44	66	174	87	42	84.5
	No. Iter.	36	41	17	42	27	50	10	39	34	43	33.4
OPT2	Obj.	53	51	41	70	114	42	65	F	95	42	63.7
	No. Iter.	41	33	40	52	40	53	11	F	24	43	37.4
OPT3	Obj.	47	51	41	62	115	43	66	F	54	40	57.6
	No. Iter.	41	34	25	51	39	41	10	F	41	40	35.7
OPT4	Obj.	47	55	30	309	94	36	36	F	58	38	78
	No. Iter.	38	50	36	12	43	44	35	F	39	39	37
OPT5	Obj.	90	F	60	100	70	58	59	110	156	60	84.8
	No. Iter.	40	F	14	39	25	39	16	50	11	46	31.1
OPT6	Obj.	55	55	78	66	F	41	60	163	61	41	69
	No. Iter.	21	26	8	46	F	46	16	36	50	32	31.2
OPT7	Obj.	59	F	30	38	83	42	36	67	50	36	49
	No. Iter.	10	F	20	32	16	24	23	16	34	34	23.2
OPT8	Obj.	43	41	31	38	55	33	35	54	54	36	42
	No. Iter.	34	22	18	38	21	35	26	24	21	34	27.3

All the results corresponding to the above options are summarized in Table 2.6. Comparing the first four options, OPT 1, which is used by Shanno and Phua (1978) and Yang and Watson (1988), is the worst one, in the sense of results in the highest value of the objective function. Theoretically (see Oren (1974b)) we have

$$\tau_l = \frac{s_l^T D_l^{-1} s_l}{s_l^T y_l} = -\frac{\alpha_l g_l^T s_l}{s_l^T y_l} = \frac{g_l^T s_l}{g_l^T D_l y_l}. \tag{2.63}$$

These three expressions give three equivalent formulae for τ_l. In general the motivation for using the last two formulae to calculate τ_l is to avoid calculating the inverse of D_l. In the LBFGS method only D_0 is directly provided; D_l for $l \geq 1$ is never calculated directly. Thus, we tried implementing the first and last forms for τ given in Eq. 2.63 with D_l replaced by D_0. OPT 4 uses the formula

$$\tau_l = \frac{s_l^T D_0^{-1} s_l}{s_l^T y_l} = \frac{s_l^T s_l}{s_l^T y_l}, \tag{2.64}$$

OPT 3 uses

$$\tau_l = -\frac{\alpha_l g_l^T s_l}{s_l^T y_l} \tag{2.65}$$

and OPT 2 uses

$$\tau_l = \frac{g_l^T s_l}{g_l^T D_0 y_l} = \frac{g_l^T s_l}{g_l^T y_l}. \tag{2.66}$$

In Table 2.6, an F entry indicates that the algorithm converged to a very large value. In our examples, F corresponds to a value greater than or equal to 700. Note that OPT1, in which a "fixed" scaling factor was used converged in slightly fewer iterations than options 2, 3 and 4, but it converged to a higher average objective function value (84.5). In OPT 5, the diagonal of C_M was used as the initial inverse Hessian approximation and we only scale the initial D_0. For initial scaling, all the three formula which are used to calculate τ (Eq. 2.63) are identical. Thus, in the OPT 5 case, it does not matter which formula is used to calculate τ. In OPT 6, 7, and 8, the diagonal of C_M was used as the initial inverse Hessian approximation and we scale the initial D_0 at each iteration. The difference between them is that different formulas were used to calculate τ. Comparing these results with those from OPT 5, we can conclude that scaling H_0 at each iteration is better than just scaling at only the initial iteration. (We obtained the same conclusion for the case where the identity matrix was used as D_0 even though we did not show these results in Table 2.6.) Comparing the results of OPT 6 through 8, we also can conclude that using $(s_l^T D_0^{-1} s_l)/(s_l^T y_l)$ (OPT 8) to calculate τ provides the best results. Our current recommendation is to use $\tau = s_l^T D_0^{-1} s_l / s_l^T y_l$ in LBFGS.

From the above analysis, we can draw several conclusions.

1. In LBFGS, scaling D_0 at each iteration is much better than only scaling the initial D_0.

2. In LBFGS, if the same scaling scheme is used, using the diagonal of C_M as the initial D_0 is superior to using the identity matrix.

3. Using formulas which depend on the gradient and search direction at each iteration to calculate the scaling factor γ is better than using a "fixed" formula to calculate γ.

4. If a diagonal matrix is used as the initial inverse Hessian D_0, the best choice for calculating τ is to choose $(s_l^T D_0^{-1} s_l)/(s_l^T y_l)$.

2.9.2 Improved Preconditioned Conjugate Gradient Method

In the results shown previously, we found that the conjugate gradient method with C_M as the preconditioner does not work well. Here, we denote this method by P-CG which means the preconditioner is given by the full matrix C_M. For the 10 realizations tested, this conjugate gradient algorithm converged to higher objective function values than the BFGS algorithm with C_M as the initial inverse Hessian approximation (see Table 2.5) and the LBFGS algorithm. As discussed previously, the inverse Hessian approximation generated from the quasi-Newton method can be incorporated into the conjugate gradient algorithm as a preconditioner. Here we tested two preconditioners. One is generated from BFGS with the full matrix C_M as the initial inverse Hessian approximation. For simplicity, we call this algorithm BFGS-P-CG which means the preconditioner is generated from BFGS. The other preconditioner is generated from LBFGS using OPT 8, i.e. the diagonal of C_M was used as H_0 and the optimal scaling was used at each iteration. We refer to this algorithm as LBFGS-P-CG which means the preconditioner is generated from LBFGS. The final objective function value and the number of iterations required to converge for both algorithms are shown in Table 2.7. For the purpose of comparison, we also include the results from BFGS, LBFGS and P-CG in this table. The convergence behavior of LBFGS and LBFGS-P-CG are similar. Compared to the BFGS algorithm, we can see that on average the BFGS-P-CG converged to slightly lower objective function values in fewer iterations although BFGS-P-CG failed for one realization. Both BFGS-P-CG and LBFGS-P-CG have much better convergence properties than P-CG. However, to implement BFGS or BFGS-P-CG, we have to form explicitly the $N_m \times N_M$ matrix which approximates the inverse Hessian approximation. Recall that N_m is the number of model parameters. For the large scale problems we wish to consider, the number of model parameters might be on the order of tens of thousands. For these problems, BFGS methods become impractical.

Table 2.7: Results for BFGS-P-CG and LBFGS-P-CG

		R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	Average
BFGS-P-CG	Obj.	90	F	28	40	54	41	39	46	53	47	48.7
	No. Iter.	35	F	25	31	10	33	25	19	24	26	25.3
LBFGS-P-CG	Obj.	44	51	29	38	57	33	36	47	54	36	42.5
	No. Iter.	23	17	23	31	23	38	21	35	20	30	26.1
P-CG	Obj.	153	146	70	94	347	42	184	213	230	45	152
	No. Iter.	12	20	11	23	19	64	5	17	19	23	21.3
BFGS	Obj.	59	52	35	41	70	41	39	53	54	87	53
	No. Iter.	33	17	18	19	33	38	18	42	35	52	30.5
LBFGS	Obj.	43	41	31	38	55	33	35	54	54	36	42
	No. Iter.	34	22	18	38	21	35	26	24	21	34	27.3

From the limited examples that we have considered, we found that LBFGS and LBFGS-P-PCG are the most effective and efficient methods among all the algorithms. So these two algorithms were used to generate 50 different realizations. For the LBFGS algorithm, the average value of the objective function at the convergence and the average number of iterations required to converge are 44.7 and 27.2 respectively. For the LBFGS-P-CG algorithm, these two average values are 44.3 and 26.3, respectively. Note these results are consistent with those shown in Table 2.7. Based on these results, we believe that these two minimization algorithms will be effective for history matching problems.

To further confirm the effectiveness of the preconditioner generated from the LBFGS, we used LBFGS-P-CG algorithm in the restricted-entry case where P-CG works very poorly (see Kalita (2000)). The restricted-entry example is a slight modification of the previous example. In the restricted entry case, only the top layer is open to flow. For this case we also generated 10 realizations using the P-CG and the LBFGS-P-CG algorithm. The value of the objective function at convergence and the number of iterations required to converge for both methods are summarized in Table 2.8. When P-CG was used, all the 10 realizations converged to a very high value of the objective function. On average, 35.3 iterations were required to reduce the average objective function value to 447. For the same 10 realizations, LBFGS-P-CG performs very well. The objective function converged to an average value of 43.1 in 29.3 iterations.

Table 2.8: Results for restricted entry case

		R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	Average
P-CG	Obj.	237	578	159	372	252	119	776	1381	504	93	447
	No. Iter.	46	63	30	12	17	52	33	39	19	42	35.3
LBFGS-P-CG	Obj.	55	37	32	41	52	38	38	51	50	37	43.1
	No. Iter.	35	34	23	32	21	30	34	23	31	30	29.3

Chapter 3

SUBSPACE METHODOLOGY FOR HISTORY MATCHING

History matching is usually cast in the form of a minimization problem: Compute the set of reservoir model parameters, m, that minimize the mismatch between the observed data, d (the historical pressure and rate observations), and the predictions of data computed from the model parameters, $g(m)$. If the measurement errors are Gaussian, modeling errors are negligible, and the prior probability density function (pdf) for the model parameters is multi-Gaussian, the set of parameters with the highest probability density can be found by minimizing the following functional of the model;

$$O(m) = \frac{1}{2}\big(m - m_{prior}\big)^T C_{\mathrm{M}}^{-1}\big(m - m_{prior}\big) + \frac{1}{2}\big(g(m) - d\big)^T C_{\mathrm{D}}^{-1}\big(g(m) - d\big), \tag{3.1}$$

where m_0 is the vector of prior estimates of model parameter values. C_{M} is the prior covariance matrix of the model parameters; it describes the variability and correlation of the model parameters. C_{D} is the covariance matrix of the measurement errors. The use of the covariance matrix in the objective function provides a dimensionless measure of data mismatch and model roughness, so that a reasonable tradeoff between the two terms is achieved in minimization.

When the number of model parameters is large, as it is in history matching, the natural choice for minimization of the objective function is conjugate gradient. Conjugate gradient methods typically converge too slowly to be practical, however. Methods that make use of the curvature information in the objective function typically converge much more rapidly, especially in the neighborhood of a minimum. If there are relatively few data (N_{D} is small), an expansion of this form can be an efficient form of solution. The problem is considerably more difficult when both the number of data and the number of model parameters are

large. In this case, the standard Gauss-Newton approach is generally too expensive to use for minimization. Because of measurement errors and redundancy in the data, a good approximation of the Newton direction can often be found in a subspace of much smaller dimension than N_D, however.

3.1 Reparameterization

The basic idea of any subspace procedure is that at the lth iteration of the Newton method, one can approximate the search direction δm^{l+1} as a linear combination of a relatively small number of subspace vectors without significantly changing the value of δm^{l+1} obtained.

In some approaches, the set of basis vectors or functions are changed at every iteration of the minimization procedure. This has the advantage of potentially using basis vectors that are optimal at every iteration. The disadvantage is that the computation of the optimal basis vectors can be expensive by itself. Shah et al. (1978) proposed the use of the eigenvectors associated with the largest eigenvalues of $G^T G$ as basis vectors. Because the dimension of $G^T G$ is large, computation of eigenvalues and eigenvectors can be expensive for realistic models. For a nonlinear history-matching problem, however, one must compute the gradients, or the action of the gradients on vectors, as well as the eigenvalue/eigenvector decomposition of a large matrix at several minimization iterations in which case the repeated computation of G will be expensive. Also, it is seldom clear how many of the singular vectors should be retained from the decomposition of G or $G^T G$.

3.1.1 Partitioning of the Objective Function

We found that when too small a set of subspace vectors is chosen, the result is a very low convergence rate. Also, in some cases, a poor choice of subspace vectors can result in visible artifacts. This is most noticeable with methods like pilot point and zonation. When a good set of subspace vectors are chosen, the effect on the solution should be imperceptible.

If one were to use only one subspace vector in each Newton iteration, an appropriate choice at iteration $l+1$ would be the gradient of the objective function. With this choice, the subspace method would be equivalent to the steepest descent algorithm. A better subspace method, which uses more subspace vectors but is still easy to implement and converges more rapidly for large problems, is one in which the subspace vectors are obtained by partitioning the data objective function (Reynolds et al., 1996). This method, which has been discussed in the geophysical literature (Oldenburg et al., 1993), tends to group data with similar

information content together, thus removing some of the redundancy, without the expense of computing many sensitivity coefficients or of performing a singular value decomposition.

For the problem of conditioning rock property fields to transient pressure data, we partition the total objective function into a term that arises from model roughness and distance from the prior model and terms that arise from pressure misfit, similar to the approach suggested by Kennett and Williamson (1988), Oldenburg et al. (1993), and Oldenburg and Li (1994).

The gradients of these objective functions are given by

$$\nabla O_{\mathrm{M}} = C_{\mathrm{M}}^{-1}(m - m_{prior}), \tag{3.2}$$

and

$$\nabla O_{\mathrm{D}}^{k}(m) = G_{k}^{T}[C_{\mathrm{D}}^{k}]^{-1}\big(g^{k}(m) - d^{k}\big). \tag{3.3}$$

A partitioning of the objective function is done at each iteration of the Gauss-Newton algorithm. The gradients of the partitioned data objective function clearly provide a useful set of basis vectors for minimization. If all of the data are grouped together to generate a single basis vector the method is equivalent to steepest descent. At the other extreme, we know that the Gauss-Newton method can be written in such a way that the basis vectors are the columns of $C_{\mathrm{M}}G^{T}$. This is exactly the basis that would be obtained if each set of data were to include only one measurement. Hence the partitioning of the objective function as a method of selecting basis vectors satisfies the recommendation of Parker (1994) that a good depleted basis should at least be capable of approaching the true optimal solution if enough basis functions are used. It also seems clear from the limiting cases that it is desirable to premultiply the gradients of the partial objective functions by the prior covariance matrix to generate subspace vectors for the expansion of δm^{l}.

Although it is clear that the number of basis vectors should be between 1 (steepest descent) and N_{D} (Gauss-Newton), it is not clear either how many basis vectors to use or how to choose them.

3.1.2 The Ideal Reduced Basis

In the Bayesian framework for solving inverse problems, the solution is based on a tradeoff between honoring the observations and closeness to the prior model estimate. It is possible to show in this case that there is an optimal number of basis vectors such that the convergence of the Gauss-Newton method is unaffected by the reparameterization. Let $C_{\mathrm{M}} = LL^{T}$ be a square-root decomposition of the prior model covariance matrix and define a dimensionless vector of model corrections $\alpha = L^{-1}\delta m$.

We can say that a good approximation to δm can be constructed from the columns of LU_p where the columns of U_p are the eigenvectors of $L^T G^T C_{\mathrm{D}}^{-1} GL$ associated with the p eigenvalues whose magnitudes are of order 1 or greater. Thus, a decomposition of this form would provide an optimal number of subspace vectors to use, and an optimal set in the sense that the vectors that are not used would not contribute to the solution. Unfortunately, computation of this set of basis vectors is probably too expensive to be practical.

If too few basis vectors are used or if the basis vectors are poorly chosen, it seems intuitive that the convergence rate will be affected. On the other hand, the total computational effort is more important than the number of Gauss-Newton iterations so it is necessary to investigate the tradeoff between the number and choice of subspace vectors and convergence rate. We illustrate the effect of various choices of basis vectors with several two- and three-dimensional single-phase reservoir models.

3.2 Synthetic Examples

A three-dimensional model with $25 \times 25 \times 10$ gridblocks was created to evaluate the potential strengths and weaknesses in the methodology. Each gridblock in the 3-D model is 10 ft thick and 100 ft in the x and y directions. The log-permeability and porosity fields were modeled as Gaussian random fields with anisotropic spherical variograms. The correlation coefficient between log-permeability and porosity is 0.7. The range of the variogram is 1,200 ft in the x-direction, 800 ft in the y-direction and 30 ft in the vertical direction. Prior to the incorporation of pressure data, the porosity was expected to lie in the range 0.20 ± 0.09 with 95% confident. Similarly, the prior mean for log-permeability is 4.0 and the prior uncertainty is ± 1.4. The permeability was assumed to be isotropic, i.e., $k = k_x = k_y = k_z$. In this model, there are thus 12,500 model parameters (6,250 gridblocks and two model parameters per gridblock).

The 3-D model also contains five wells. Well 1 produces at a constant rate 3,000 rb/day for the first 1 day, followed by a 1 day buildup, then production at a constant rate of 3,500 rb/day for 1 day; Well 2 produces at a constant rate of 3,400 rb/day for 1 day, then a 2 day buildup; Well 3 produces at a constant rate of 1,500 rb/day for 1 day, then 2,500 rb/day for 1 day and finally 3,500 rb/day for 1 day; Well 4 is an observation well; Well 5 produces at a fixed rate of 3,200 rb/day, for all times. Synthetic pressure data (Fig. 3.2) are generated using the true log-permeability and porosity fields shown in Fig. 3.1. In this case, we have 86 pressure data at each well so the total number of data is equal to 430. Pressure measurement errors were assumed to be independent, with variance, $\sigma_d^2 = 0.01$ psi^2. For simplicity, the

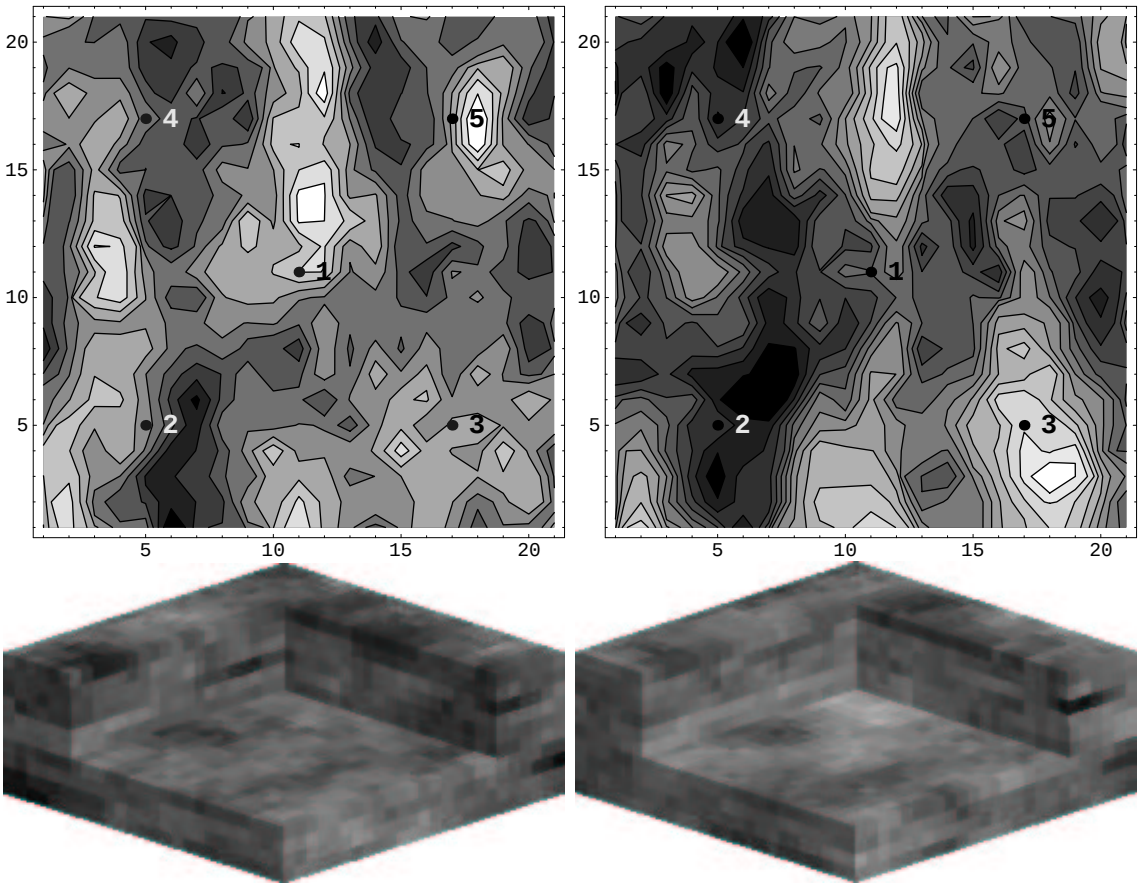


Figure 3.1: The true log-permeability and porosity fields for the 2-D problem (upper row) and 3-D problem (lower row). Pressure measurements are recorded at the five well locations.

skin factors at the wells were assumed to be known. Other reservoir and fluid properties are the same as the two-dimensional example.

3.2.1 Results from Simple Partitioning with Constant Basis Dimension

Using the conventional Gauss-Newton history-matching approach without subspace vectors, the maximum a posteriori model estimates of log-permeability and porosity (conditioned to pressure data from the true 2-D model) were generated. These are shown in the left column of Fig. 3.3. Generation of the maximum a posteriori model estimates using the conventional approach (Eq. 2.16) requires computation of 145 sensitivity vectors for each iteration of the Gauss-Newton procedure, and "inversion" of a 145×145 matrix. For single-phase flow in a fluid with small compressibility, the computation of all 145 sensitivity vectors requires

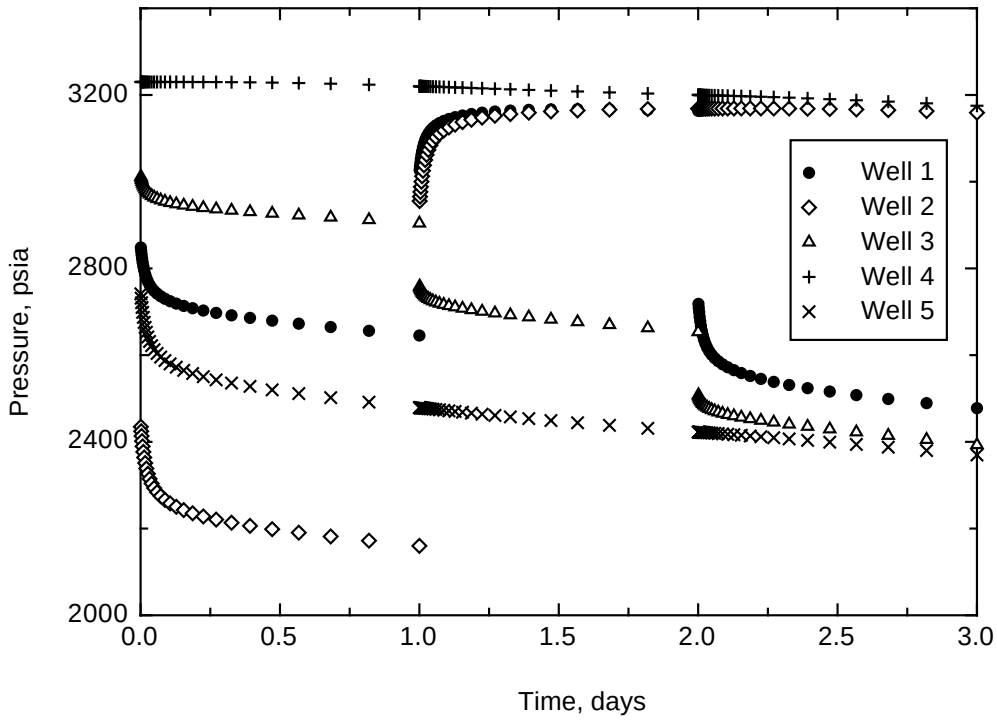


Figure 3.2: The pressure history used for conditioning the permeability and porosity in the three-dimensional model.

work equivalent to approximately 5 flow simulations (Carter et al., 1974). For a multiphase flow problem, the computational effort required to generate one sensitivity vector would be equivalent to approximately one flow simulation.

A straightforward approach to selecting subspace vectors is to partition the data in the objective function first by the well at which the data were measured, and then by time period. By trial-and-error, we found that partitioning the data from each well into 9 subsets resulted in convergence to a minimum value of the objective function that was just as rapid as when all the data were used. The resulting estimates for log-permeability and porosity were quite similar to the MAP estimates obtained using the conventional methods (Fig. 3.3). The results from the subspace approach were obtained in far less time, however. In this example, although the total number of model parameters is 882 and the number of pressure data is 145, it was possible to parameterize the changes in the model in a subspace whose dimension was only 47, without significantly affecting the number of iterations required for convergence or the quality of the match.

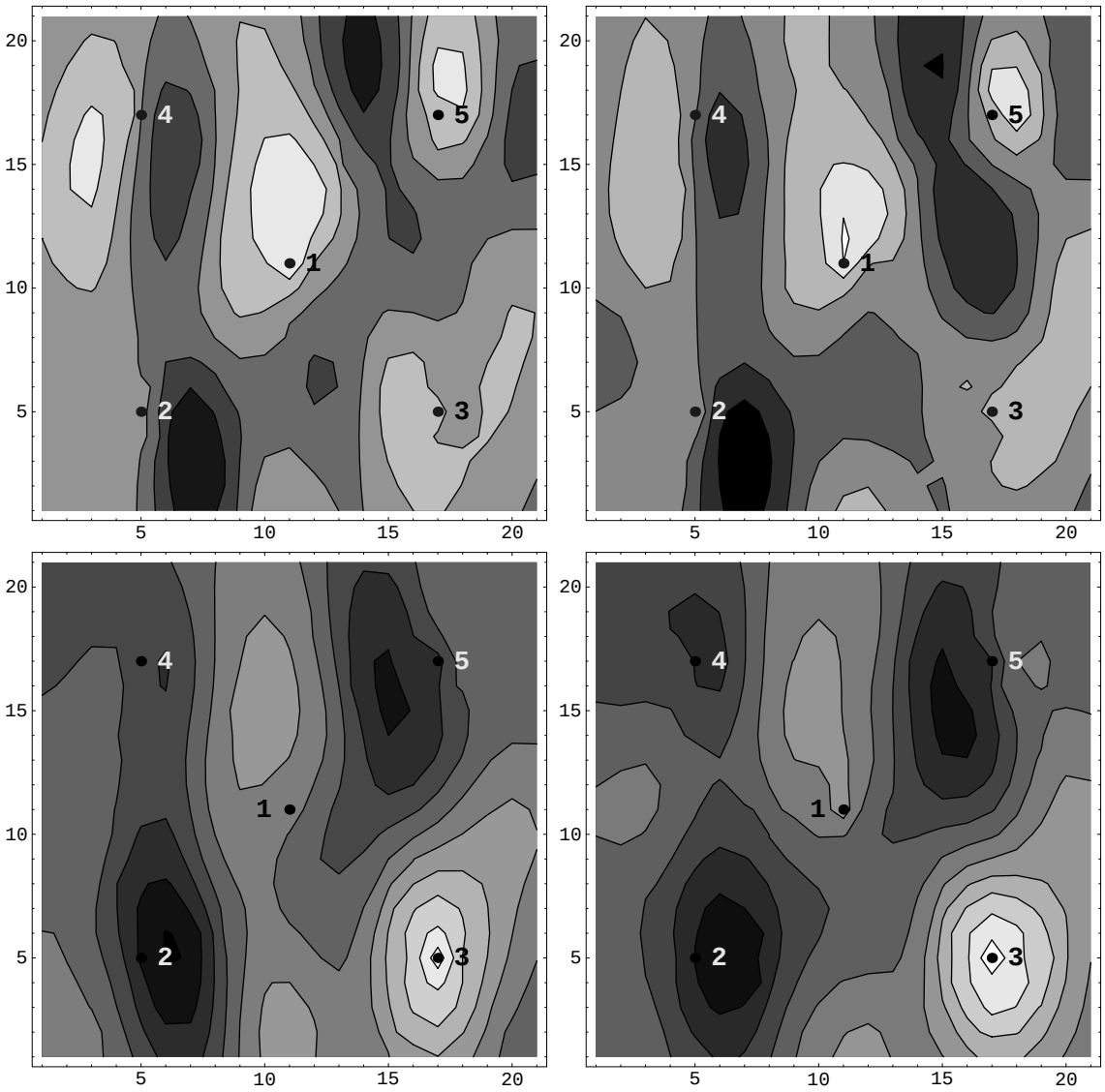


Figure 3.3: MAP estimates of the porosity field using 145 basis vectors (upper left) and 47 basis vectors (upper right). MAP estimates of the log-permeability field using 145 basis vectors (lower left) and 47 basis vectors (lower right).

3.2.2 Dimension of Basis from Eigenvalue Analysis of Hessian

The decision to use 47 subspace vectors in this example was somewhat ad hoc but an analysis of the spectrum of the dimensionless Hessian, $L^T G^T C_{\mathrm{D}}^{-1} G L$, shows that the number of subspace vectors required to span the data space in the first iteration is on the order of 50 (see Fig. 3.4) so the use of 47 is probably close to optimal. Because it is unlikely that the eigenvalues of the Hessian will be computed for large problems, however, we explored

the consequences of using a smaller than optimal number of subspace vectors. Incidentally, the consequence of using more than the optimal number is simply an additional expense in computer time. There is no detrimental effect on the resulting estimate when additional subspace vectors are used.

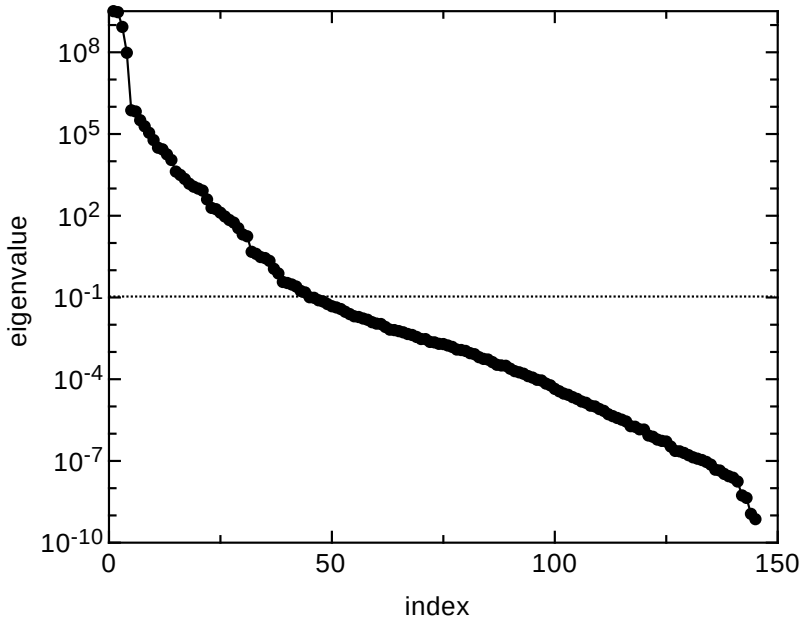


Figure 3.4: The first 145 eigenvalues of the matrix $L^T G^T C_{\mathrm{D}}^{-1} G L$ showing that approximately 40–50 basis vectors are sufficient to accurately construct δm in a Newton iteration.

3.2.3 The Effect of Small and Large Basis Dimensions

The number of iterations required to obtain an acceptable level of data mismatch in a Newton method is a function of the number of subspace vectors used in the representation of δm and, to a lesser extent, a function of the actual choice of subspace vectors. With 47 subspace vectors (i.e. 9 data sub-objective functions per well plus two model mismatch vectors) convergence was achieved in four iterations, which was the same as in the conventional Bayesian inverse approach. When 7 subspace vectors were used (i.e. one data sub-objective function per well plus two model mismatch vectors), the initial rate of reduction in the data mismatch was large but after two Newton iterations the rate of reduction slowed substantially (Fig. 3.5). Even after 50 iterations, the total objective function was still very far from the value obtained in 4 iterations with 47 subspace vectors.

Results with 12 and 22 subspace vectors were intermediate between the previously described cases. In these cases, the pressure data were subdivided first by well, and then by flow period. Thus, for Well 1 which produced at 700 B/D for one day, then was shut-in for one day, and was produced for a final day at 700 B/D, the data for day 1 were subdivided into two periods, the data from day 2 were all grouped together, and the data from day 3 were also subdivided into two periods. This gave five subspace vectors per well. Data from two flow periods (such as shut-in and drawdown) were never combined. The initial rate of reduction in the data misfit was large, but after four iterations the rate of reduction again slowed substantially (see curve marked with black squares in Fig. 3.5). After 15 iterations, the value of the objective function obtained with 22 subspace vectors was 280 while the value obtained with 47 subspace vectors in 4 iterations was 220.

In general, we observed that when the number of subspace vectors used was smaller than some limiting value, the rate of reduction in the data misfit function would slow to an unacceptably slow rate after a few iterations. The number of subspace vectors required to achieve rapid convergence seems to be approximately equal to the number of eigenvalues of $L^T G^T C_{\mathrm{D}}^{-1} GL$ that are greater than 0.1. Although this matrix changes as the minimum is approached, the number of eigenvalues that are greater than one was relatively constant in these examples.

3.2.4 Importance of the Choice of Subspace Vectors

We initially believed that it would be important to group data with similar information content together to generate basis vectors. If this had been true it would have had severe consequences on the utility of the method. In fact, we found that the rate of reduction in the objective function seems to be dictated largely by the number of subspace vectors, and not by the exact choice of vectors. Thus, when we used the first 15 singular vectors from a SVD of a large set of trial subspace vectors, the results were similar to those obtained by simple partitioning the objective function. Similarly, Fig. 3.6 contains a comparison of the results from two subspace vector selection schemes, both of which have 22 subspace vectors. In the first case, (*across* flow periods), we simply partitioned the data from each well into sets of approximately equal size. In the second case, (*within* flow periods), we did not allow data from different flow periods to be combined. The results were again nearly identical. This is encouraging as it suggests that the subspace spanned by the gradients of sub-objective functions for any reasonable partitioning of the data are similar.

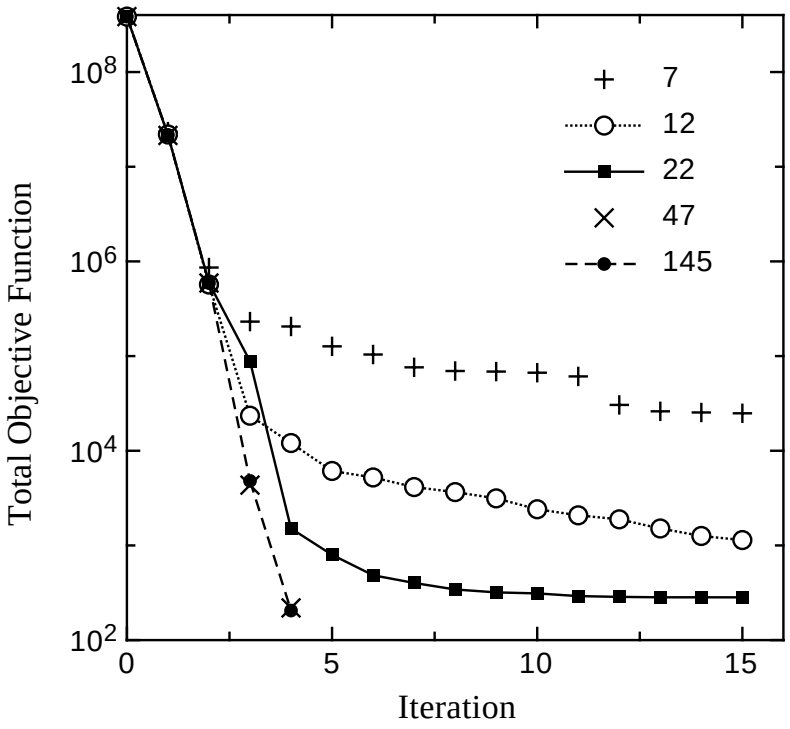


Figure 3.5: The number of Newton iterations required to reduce the objective function to the desired level depends on the number of subspace vectors used in the expansion of δm.

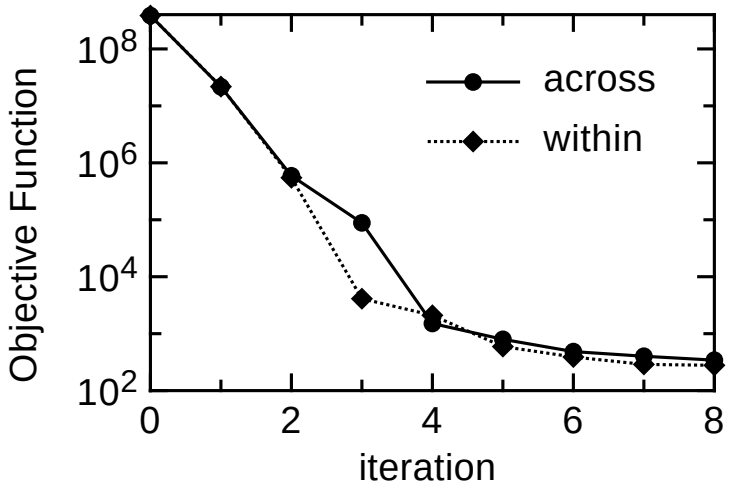


Figure 3.6: The convergence behavior is similar for two different methods of partitioning the data.

3.3 Gradual increase in dimension of basis

In the discussion of Fig. 3.5, we observed that unless we chose enough subspace vectors, the rate of convergence could become quite slow. It was unclear, however, how to estimate the proper number in advance without computing eigenvalues or singular values of a large matrix. Secondly, we observed that the initial rate of reduction in the data misfit was always large, even for small numbers of subspace vectors. This suggests that an efficient strategy for minimizing the computational effort is to begin with a small number of subspace vectors in the early iterations, adding more when needed to maintain a high rate of convergence. Unfortunately, when this method was first attempted, it was never very efficient, largely because the addition of new subspace vectors always resulted in an increase in model roughness at the next iteration. This problem was eventually resolved by using Levenberg-Marquardt with a fairly large damping factor.

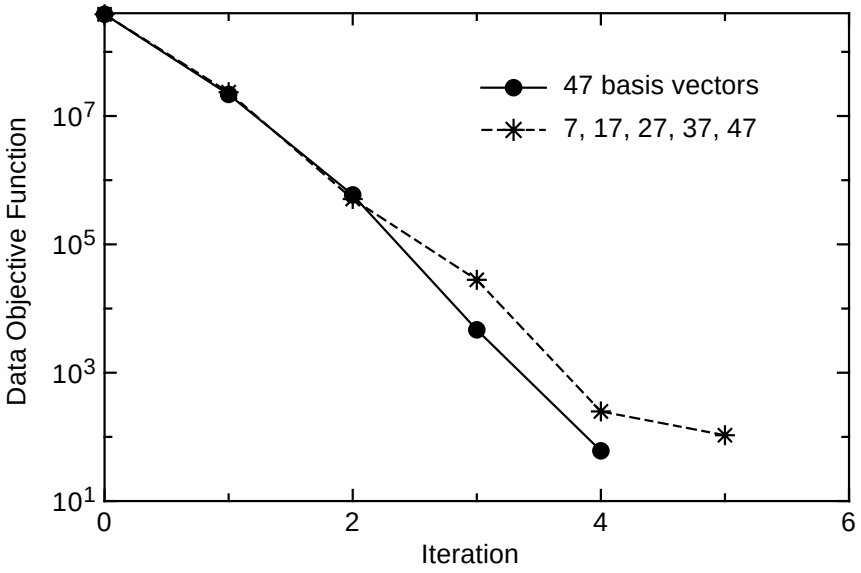


Figure 3.7: The convergence is only slightly slower, but the total work is reduced when the dimension of the basis vectors is increased at each iteration.

Fig. 3.7 compares the reduction in the objective function for a 2-D problem in which the number of subspace vectors is increased in each Newton iteration to a case in which the number of subspace vectors was constant. In this case, the number of subspace vectors used was 7, 17, 27, 37 and 47 in iterations 1 through 5, respectively. An equal number of subspace vectors (i.e. 1 per well, 3 per well, 5 per well, 7 per well and 9 per well) were used for each well at each Newton iteration. Fig. 3.7 shows that when the number of subspace vectors

was gradually increased, the pressure mismatch part of the objective function was reduced nearly as fast as when 47 subspace vectors were used in all iterations. The work done in the variable subspace dimension case was about 70% of the work required for the case with 47 subspace vectors in all iterations. The savings from using an increasing number of subspace vectors would probably be larger in practice as the number of subspace vectors to use in a constant dimension method would likely be estimated suboptimally.

3.3.1 Traditional Levenberg-Marquardt

Gauss-Newton had worked quite well for most cases in which the number of subspace vectors was constant, but when the number of subspace vectors was increased at every iteration it became more difficult to achieve convergence to a small value of the objective function. In these cases, we typically found that the model mismatch (or regularization) part of the objective function had become large in the early iterations. It was difficult to remove the model "roughness" in subsequent iterations and the final value of the objective function was unsuitably large. To solve this problem, we implemented the Levenberg-Marquardt algorithm in a slightly non-standard way. Instead of adding a diagonal matrix, λI to the Hessian, we multiplied the inverse of the model covariance by a factor of $1 + \lambda$ in the Hessian. In the subspace parameterization, the Levenberg-Marquardt equation is obtained from Eq. 2.16,

$$\left((1+\lambda)B_l^T C_{\mathrm{M}} B_l + B_l^T C_{\mathrm{M}} G_l^T C_{\mathrm{D}}^{-1} G_l C_{\mathrm{M}} B_l\right)\alpha^l = -B_l^T C_{\mathrm{M}} \nabla O_l. \tag{3.4}$$

Otherwise, the implementation was similar to the algorithm described by Marquardt (1963) with a growth factor of 4 for λ and a decay factor of 2.

In this fairly standard implementation of the Levenberg-Marquardt algorithm, the problems we experienced were similar to those we experienced with the Gauss-Newton method; if the value of λ was too small at an early iteration, the model acquired "roughness" which was difficult to remove at later iterations (see the curve for $\lambda_0 = 10^{-1}$ in Fig. 3.8). On the other hand, when we started with a value of λ that was too large, the initial rate of reduction in the objective function was small (see $\lambda_0 = 10^9$ in Fig. 3.8), or the rate at later iterations was small (see $\lambda_0 = 10^5$ to 10^9 in Fig. 3.8). Although starting values for λ in the range 10^2 to 10^3 worked well for the 2-D example, it was necessary to start with λ between 10^6 and 10^7 to achieve a small value of the objective function for the 3-D reservoir example.

3.3.2 Levenberg-Marquardt with 1-D search

Because the parameters required for efficient convergence of the Levenberg-Marquardt algorithm were problem dependent and difficult to determine without much experimentation,

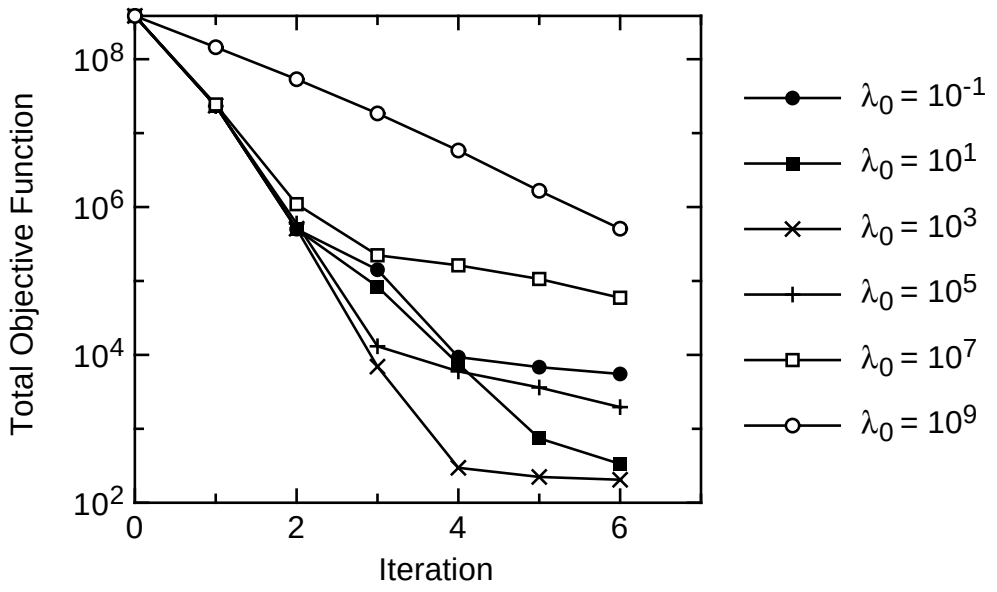


Figure 3.8: Reduction in the objective function for a wide range of starting values of the Levenberg-Marquardt damping factor.

we sought a more robust implementation of the algorithm. One simple way to prevent the incidence of model roughness is to multiply the model mismatch term in the objective function by a large factor before minimization. We chose to base the magnitude of the model mismatch multiplication factor on the magnitude of the data mismatch so that the two terms would be of comparable magnitude. In the ith Levenberg-Marquardt iteration, we define

$$\gamma_i = \max[1, O_{\mathrm{D}}^i/(8N_{\mathrm{D}})],$$

then perform a one-dimensional search for the value of the Levenberg-Marquardt damping factor λ that minimizes

$$O_{\mathrm{F}} = O_{\mathrm{D}} + \gamma_i O_{\mathrm{M}}. \tag{3.5}$$

Note that γ_i goes to one as the data mismatch, O_{D}, is reduced so that the correct objective function is minimized. This procedure has some features in common with the recommendations of Levenberg (1944) who proposed using a Newton-like method to estimate an optimal value of λ at each iteration. The primary difference is that we modify the objective function for minimization depending on the magnitude of the data mismatch. At every iteration of Levenberg-Marquardt minimization, we use Brent's method (Brent, 1973) to search for the λ that minimizes the total objective function of Eq. 3.5. We assume that the optimal value of λ is bracketed by $10^{\gamma_i - 3.5}$ and $10^{\gamma_i + 3.5}$. For each trial value of λ, we must solve the system of

equations in Eq. 3.4, update the model and evaluate the objective function of Eq. 3.5. Evaluation of the objective function requires one solution of the forward problem (one simulation run). Because approximately 6 evaluations of the objective function are typically required to estimate the optimal value of λ by Brent's method, the method will only be efficient if it results in relatively few Levenberg-Marquardt iterations compared to a method that does not attempt to optimize λ.

The convergence behavior and schedule of optimal values of λ for the 2- and 3-D problems were much different (Fig. 3.9). In both cases, however, the procedure of gradually increasing the number of basis vectors and solving for the best λ at each iteration was more efficient than traditional Gauss-Newton or Levenberg-Marquardt.

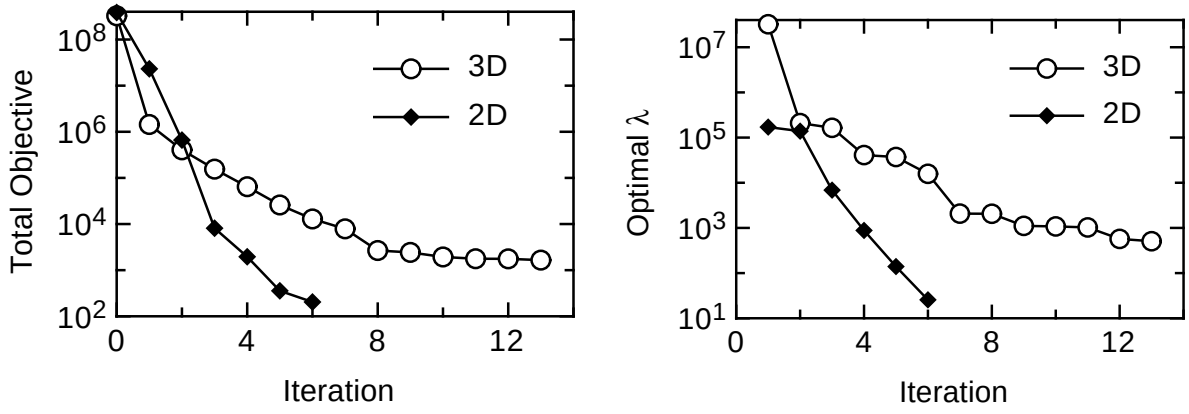


Figure 3.9: The rates of reduction in the objective function (left) and in the optimal value of lambda (right) are much different for the 2- and 3-D problems.

3.4 Computation Details

Consider the work involved in the computation of a MAP estimate using the Gauss-Newton Method with subspace vectors. For each Gauss-Newton iteration, Eq. 2.29 must be solved to determine the correction to the model parameter estimates. The main computational components are the computation of the subspace vectors (if required), and the computation of the product GA.

If the subspace vectors are chosen a priori, independently of the data, there would be no significant effort involved in their computation (other than multiplication by C_{M} perhaps). If the subspace vectors are chosen based on the data, as we propose, then one solution of the

adjoint system is required per subspace vector. Because the adjoint problem is linear, while the forward equations are typically nonlinear, the time required for the solution of the adjoint system should be less than the time required for the forward solution. The computation of the product GA has been addressed by Chu et al. (2000) and is summarized in Abacioglu et al. (2001). From the discussion of Killough et al. (1995) we see that the time required for computation of the product of G with each column of A is again some fraction at the time required for one simulation run. In their case, the fraction was on the order of 15%. Other parts of the computation are relatively insignificant, or can be made so with some care.

One interesting feature of the convergence plots is that the rate of convergence at early Newton iterations appears to be independent of the number of subspace vectors. Thus whether we use 7 subspace vectors or 145, the objective function is reduced by a factor of 10^4 in the first two iterations. The most efficient methods started with as few subspace vectors as possible at early iterations, then gradually increased the number in order to achieve the required reduction in the objective function.

Chapter 4

TIME-LAPSE SEISMIC

The goal of this part of the project is to develop methods for integration of time-lapse seismic data into the history matching code. Work on this aspect began in June 2001 with the arrival of a PhD student, Yannong Dong. At this stage much of the work is of a preliminary nature. The actual development of software and algorithms will begin in Spring 2002.

There have been many recent papers on the use of time-lapse seismic data for improved reservoir characterization and reservoir management. In most of these papers, the time-lapse seismic data has been used qualitatively to identify regions of the reservoir which are undrained or to identify movement of fluid contacts (Koster et al., 2000). Other researchers have assumed that the time-lapse seismic data provides saturation or an indicator for saturation change (Landa and Horne, 1997; Gosselin et al., 2001). This may not be a bad approximation in some reservoirs but it does ignore the effects of pressure on the seismic impedance and oversimplifies the complexity of the relationship.

In inverse theory approaches to parameter estimation, a relationship between data and the model parameters must be specified. For the problem of reservoir characterization from time-lapse seismic data, the parameters are the permeability and the porosity of every gridblock in a reservoir simulation model. At the most basic level, the data are the pre-stack seismic amplitude traces recorded at several times during the life of the reservoir. This, however, makes for a very difficult problem because the relationship is complex. The relationship can be thought of as being given by the following sequence of relationships between intermediate quantities.

1. Given permeability and porosity at every gridblock, the gridblock saturations and pressures can be computed using the reservoir simulator.

2. Given saturations and pressures (and some additional elastic parameters for the rock),

the poro-elastic modulii can be computed using Gassmann's equations or something equivalent.

3. Given the poro-elastic modulii, the seismic amplitudes can be computed using forward seismic modeling.

The first and last steps in this sequence can be very time consuming. In our research we will assume that the last step can be partially eliminated by "inverting" the seismic data one time to obtain maps of seismic impedance corresponding to the times at which the surveys were acquired. By taking this approach, the seismic impedance is treated as data even though in a strict sense it is actually an interpretation of the data. One difficulty with this approach is that it is somewhat more difficult to quantify the level of errors in seismic impedance data than in the actual seismic data, largely because it is difficult to quantify the propagation of errors through the processing and because the processing introduces additional errors of unknown magnitude.

The main features of this approach are shown in Fig. 4.1. In addition, to the time-lapse data, which are shown as change in impedance, ΔZ_{obs}, it is likely that log data for porosity and production data such as WOR, GOR, and pressures at wells will be available. If the initial choice of porosities and permeabilities result in saturations and pressures that give an adequate match to the data, the algorithm stops, otherwise the permeabilities and porosities are adjusted to improve the match. We expect to be able to use many of the algorithms that we have developed for automatic history matching. The performance of minimization methods is quite specific to the particular function being minimized, however, and the spatial distribution of data for time-lapse seismic is much different from the spatial distribution of production data, so modifications are to be expected.

The inital work, specific to the problem of incorporating time-lapse seismic data, consists of two parts. The first task involves computation of effective seismic velocity and bulk modulus in saturated porous media as functions of pressure and saturation. The second task will involve the development of numerical methods for computation of sensitivities of seismic impedance to gridblock permeability, porosity, and relative permeability parameters. We will focus on the progress on the first task in this report.

4.1 Computation of seismic impedance

Seismic impedance of the reservoir is important, because the amplitude of seismic reflections is a direct function of the contrast in seismic impedance. Usually, the material that forms the

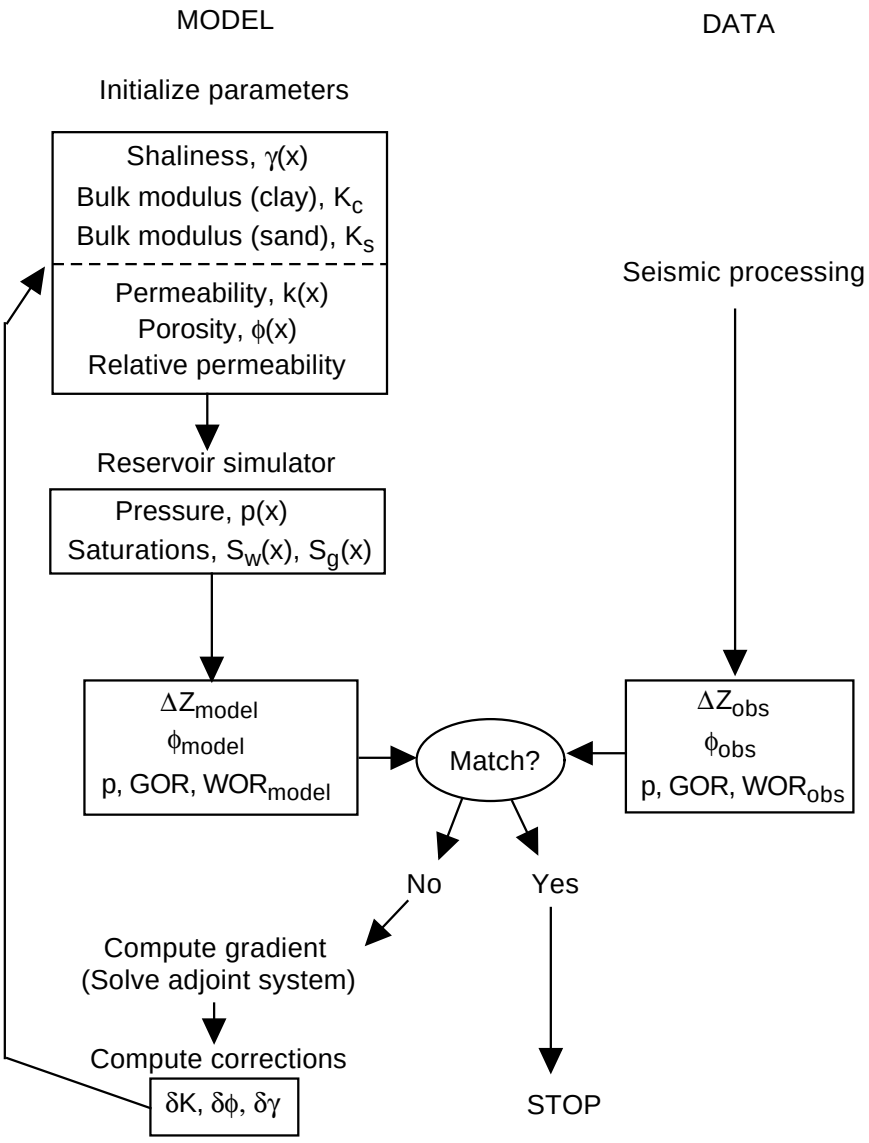


Figure 4.1: Flow chart for history matching time-lapse seismic data.

sealing cap for a reservoir has a much different seismic velocity or density than the porous material that holds the reservoir fluids. Because of the contrast, a strong seismic reflection may occur at the reservoir boundary. The time-lapse seismic method is based on the idea that the accoustic impedance may change with time because of changes in saturation and pressure due to production of the reservoir fluids.

Accoustic impedance is defined as the product of the density and the primary wave velocity, V_p. It is convenient, however, to write the impedance in the following way as a function of density, ρ, bulk modulus, K, and secondary (or shear) wave velocity, V_s.

$$Z = \rho V_p = \sqrt{\rho K + \frac{4}{3}\rho^2 V_s^2}. \tag{4.1}$$

To compute the bulk modulus of fluid saturated porous medium, we use the Gassmann equation which is applicable to low frequency oscillations.

$$K = K_g \frac{K_f + \frac{K_w\left(K_g - K_f\right)}{\phi(K_g - K_w)}}{K_g + \frac{K_w\left(K_g - K_f\right)}{\phi(K_g - K_w)}}. \tag{4.2}$$

In this equation, K is the bulk modulus of the saturated porous medium, K_f is the bulk modulus of the unsaturated porous medium, K_g is the bulk modulus of the grains, K_w is the bulk modulus of the pore fluids, and ϕ is the porosity.

The bulk modulus of the dry frame is assumed to be related to the bulk modulus of the grains, but reduced because of the porosity, i.e.,.

$$\log K_f = \log K_g - 4.25\phi. \tag{4.3}$$

If the reservoir consisted of grains of a single lithology, the grain modulus would simply depend depend on the composition of the grains. For most sandstone reservoirs, the effective grain bulk modulus depends on both the sand and clay modulii and on the shaliness, γ. One relation for computing the effective grain modulus is

$$K_g = \frac{1}{2}\left[\gamma K_c + (1-\gamma)K_s + \frac{K_s K_c}{K_s\gamma + K_c(1-\gamma)}\right]. \tag{4.4}$$

The inverse of the bulk modulus of the pore fluid is a simple weighted average of the inverse bulk modulii of the phases:

$$\frac{1}{K_w} = \frac{S_{\text{brine}}}{K_{\text{brine}}} + \frac{S_{\text{gas}}}{K_{\text{gas}}} + \frac{S_{\text{oil}}}{K_{\text{oil}}}. \tag{4.5}$$

Finally, we use the Han equation (Han, 1986) to compute shear wave velocity as a function of porosity and shaliness:

$$V_s = 3.52 - 4.91\phi - 1.89\gamma. \tag{4.6}$$

One troubling feature of these equations is the dependence on parameters such as shaliness that we may not wish to map. (Every additional variable adds to the complexity of the problem.) The reservoir flow simulator will have fluid saturations, pressures, and porosity available for every grid block in the reservoir. It is unclear whether or not it will be necessary to also map shaliness, γ, and how important the uncertainty in K_c, K_s, and γ will be for computing change in impedance.

4.2 The sensitivity of impedance to variability in shaliness

When the pressure decreases due to production, in a solution-gas drive reservoir, gas evolves and oil is displaced from the reservoir, but the saturation of water is usually approximately constant. In this case, the saturation of oil and gas will change with time. If we are to attempt to infer saturations and pressures directly from seismic impedance data, we need to compare the magnitude of the sensitivity of impedance to uncertainty or variability in shaliness, and in the elastic modulii of the minerals that make up the reservoir rock. Table 4.1 shows a reasonable set of base values for seismic impedance.

We selected what we believed to be reasonable upper and lower bounds for many of the parameters. For example, in reservoir rock, the shaliness might be expected to vary between ($0 \leq \gamma \leq 0.4$), the bulk modulus of the clay mineral to vary between ($1 \times 10^{10} \leq K_c \leq 3 \times 10^{10}$), the bulk modulus for the sand ($2.8 \times 10^{10} \leq K_s \leq 4.8 \times 10^{10}$) and gas saturation between ($0 \leq S \leq 0.2$). Figure 4.2 shows the variation of seismic impedance for those ranges of parameter inputs. Note that the sensitivity of impedance to the bulk modulus of the mineral is nearly as large as the sensitivity to a reasonable variation in gas saturation. The sensitivity to shaliness is considerably larger. From this comparison, it seems reasonable to assume that variability in shaliness would be important when attempting to invert seismic impedance data for saturation.

Parameter	Base Value
Porosity	0.2
Modulus of Sand (Pa)	3.8×10^{10}
Modulus of clay (Pa)	21.2×10^{9}
Density of Solid (kg/m^3)	2650
Density of Gas (kg/m^3)	214
Density of Water (kg/m^3)	986
Density of Oil (kg/m^3)	707
Modulus of Gas (Pa)	3.94×10^{7}
Modulus of Water (Pa)	2.39×10^{9}
Modulus of Oil (Pa)	6.71×10^{8}
Water Saturation	0.2
Oil Saturation	0.8
Gas Saturation	0.0
Shaliness	0.2
Seismic Impedance	7.00×10^{6}

Table 4.1: Base value of reservoir parameters(gas drive).

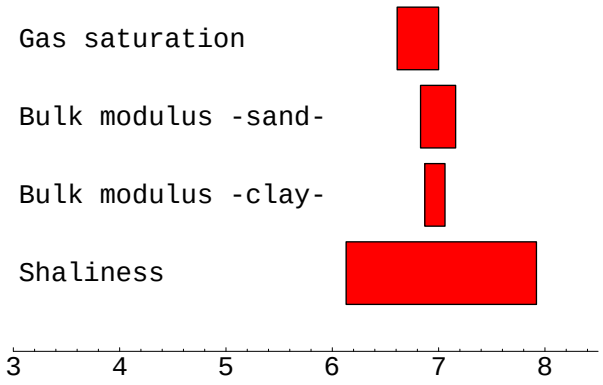


Figure 4.2: The sensitivity of impedance to variability in rock properties.

4.3 Sensitivity of change in impedance to variability in shaliness

If, instead of using the impedance, we use the change of impedance with time to estimate saturation and pressure, the situation as likely to be much different. In this case, we should

expect that some of the influence of uncertainty of reservoir properties may be reduced because only the saturation and pressure will change with time.

Table 4.2 shows the bounds that were used in this part of the investigation. To compute the influence of uncertainty in these parameters, we computed the seismic impedance at the initial time ($S_g = 0$) with the low value of the parameter (e.g. $\gamma = 0.0$), then computed the seismic impedance at the later time ($S_g = 0.5$) with the same value of the parameter ($\gamma = 0.0$). The difference tells us the magnitude of the change in seismic impedance for a clean reservoir. We then repeat the computation with $\gamma = 0.4$. This tells the magnitude of the effect of uncertainty in shaliness (γ) on the time-lapse change in seismic impedance.

The results of all computations are shown in Table 4.3. Increasing the gas saturation from 0 to 0.5 reduces impedance by about 7.5%. If the reservoir is perfectly clean the reduction is only 7.2%. If it is particularly shaly, the reduction is 8%. The variation due to uncertainty in mineral properties is much less than this. Similar results (Table 4.4) were obtained for a water flood example, in which case the change in impedance is due only to a replacement of oil by water in the pore space ($0.2 < S_w < 0.6$). As a result, we can conclude that the effect of variability in the shaliness, and bulk modulii of the rock minerals are relatively unimportant when we are considering the change in seismic impedance.

Parameter	**Low Bound Value**	**High Bound**
Gas Saturation	0.0	0.5
Shaliness	0.0	0.4
Clay Modulus (Pa)	1.0×10^{10}	3.0×10^{10}
Sand Modulus (Pa)	2.8×10^{10}	4.8×10^{10}

Table 4.2: Low bound and high bound value of reservoir parameters(gas drive).

4.4 Forward simulation of time-lapse impedance changes

In order to check the likelihood of the reservoir model being correct, it is necessary to compare the observed changes in seismic impedance with those predicted from the reservoir model. This requires that the formulas from section 4.1 be written into the reservoir simulator. One additional problem is that the thickness of grid blocks in the reservoir model is typically much less than the thickness that can be resolved by the seismic data. In order to compare the data (from seismic) with the predictions (from the model) we need to approximate the upscaling inherent in the reflection of seismic waves. For this preliminary study, we used

Parameter	Change of Seismic Impedance	Change Ratio
In Base Value	−526631	−7.5%
Shaliness in Low Bound	−502306	−7.2%
Shaliness in High Bound	−562770	−8.0%
Clay Modulus in Low Bound	−528152	−7.5%
Clay Modulus in High Bound	−525463	−7.5%
Sand Modulus in Low Bound	−528241	−7.5%
Sand Modulus in High Bound	−523081	−7.5%

Table 4.3: Change of seismic impedance by change of gas saturation.

Parameter	Change of Seismic Impedance	Change Ratio
In Base Value	251978	3.6%
Shaliness in Low Bound	242795	3.5%
Shaliness in High Bound	265121	3.8%
Clay Modulus in Low Bound	249037	3.6%
Clay Modulus in High Bound	252823	3.6%
Sand Modulus in Low Bound	247481	3.5%
Sand Modulus in High Bound	253692	3.6%

Table 4.4: Change of seismic impedance by change of water saturation.

Backus upscaling of the reservoir blocks (Backus, 1962; Lindsay and van Koughnet, 2001) to generate an effective impedance map. The actual procedure is arithmetic averaging of bulk density in the vertical direction, and arithmetic averaging in slowness (or equivalently harmonic averaging of p-wave velocity).

We used a three-dimensional reservoir model with 21 gridblocks in the x and y directions, and 8 gridblocks in the z direction. We suppose that the reservoir initially contains oil and water and that water is at the irreducible saturation equal to 0.2. There are five wells in the reservoir model. Production at four well near the corners is balanced by injection at the center well (see Figure 4.3).

The permeability of the reservoir model is heterogeneous. We assumed a log-normal distribution with mean of $\ln k$ equal to 4.5 (for permeability k measured in md) and a variable of $\ln k$ equal to 1.0. An exponential variogram model was assumed; the range in the horizontal directions is 600 ft and in the vertical directions 160 ft.

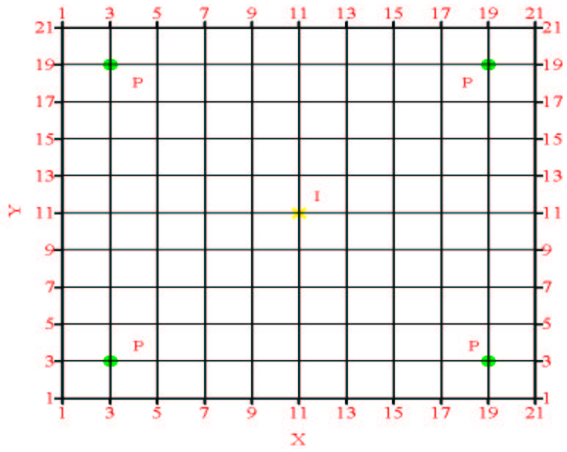


Figure 4.3: The base map for the reservoir simulation model showing the areal grid and well locations.

The simulator runs about 560 days before reaching the limiting water cut. Saturations and seismic impedance are examined at three times during the life of the reservoir. The first plot of seismic impedance change and water saturation distribution is in 10th day (Figure 4.4). The second one is in 212th day (Figure 4.5) and the last one is in the final day (Figure 4.6).

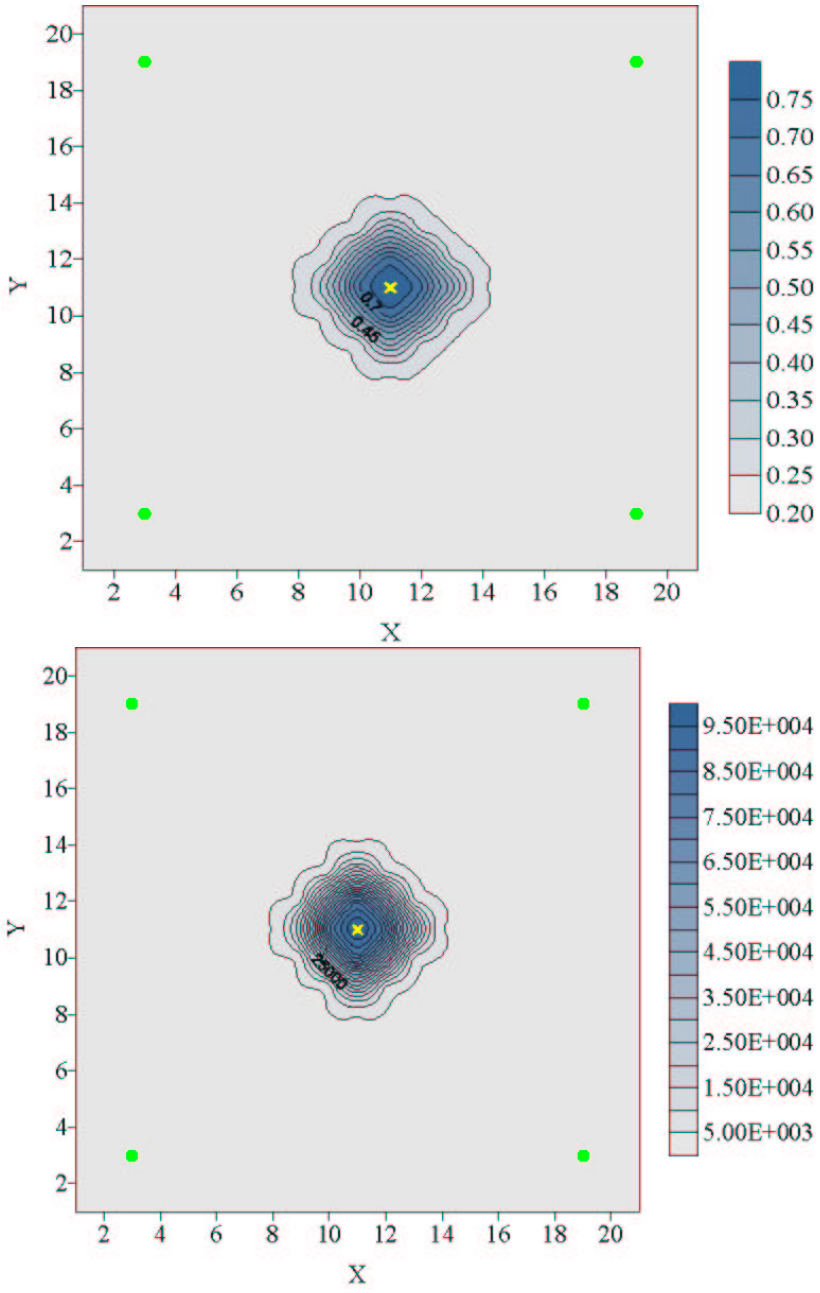


Figure 4.4: The vertically averaged water saturation (top) and the upscaled change in seismic impedance (bottom) after 10 days of injection.

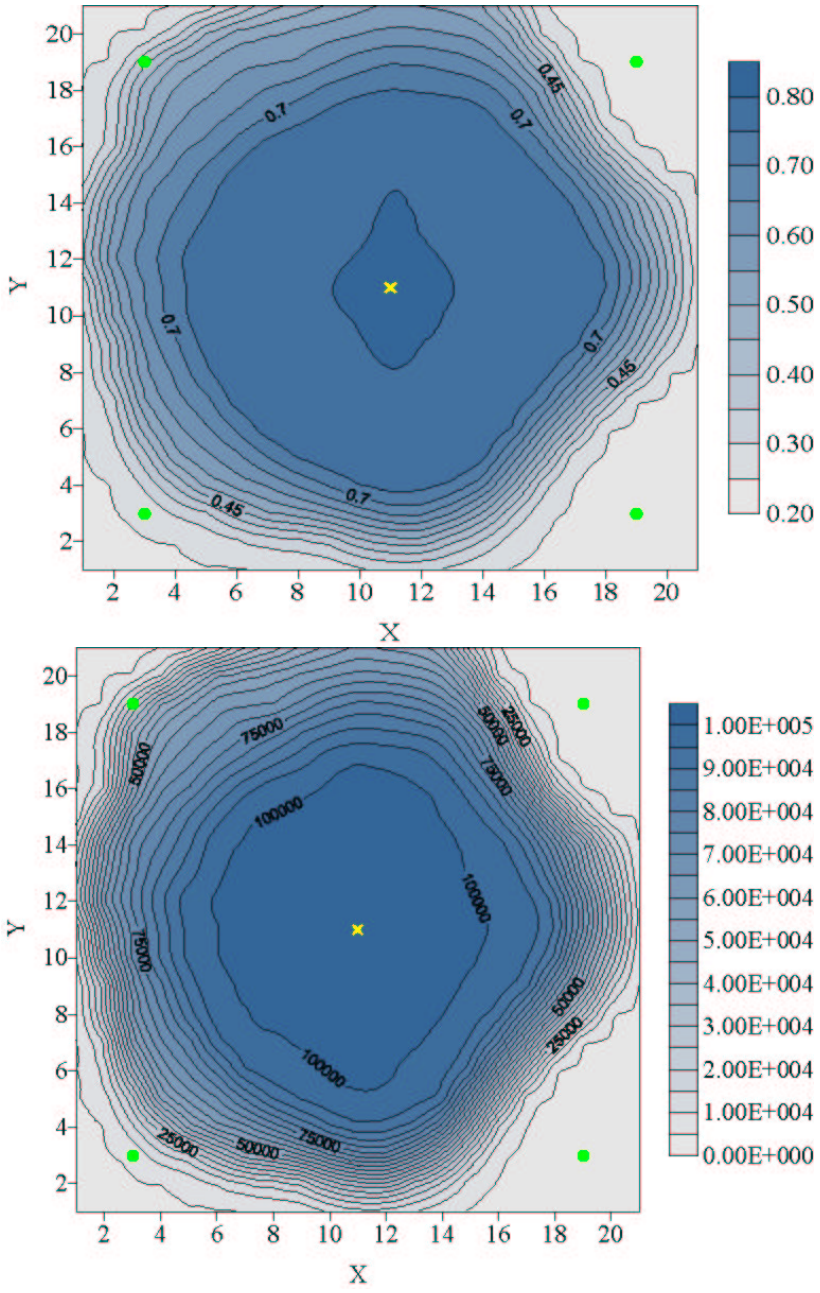


Figure 4.5: The vertically averaged water saturation (top) and the upscaled change in seismic impedance (bottom) after 212 days of injection.

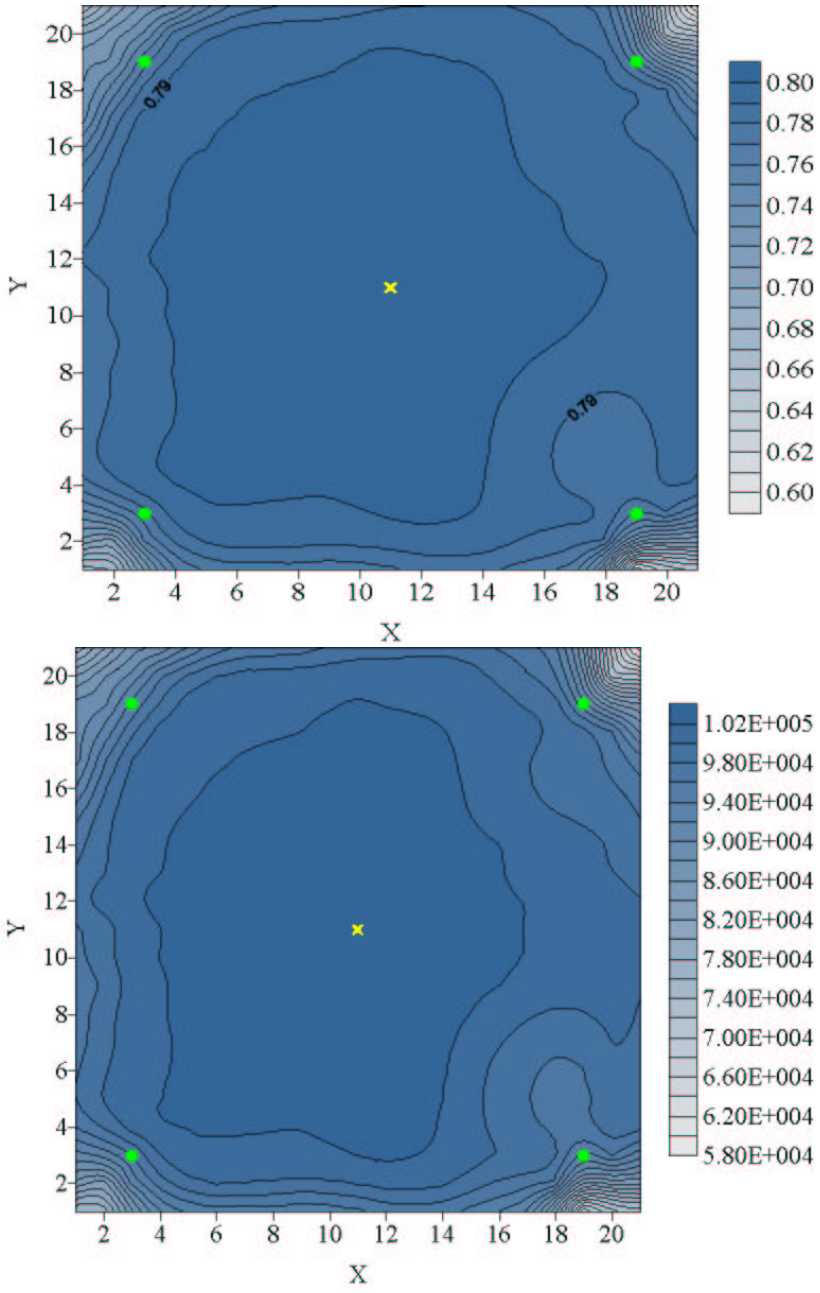


Figure 4.6: The vertically averaged water saturation (top) and the upscaled change in seismic impedance (bottom) after 560 days of injection.

Chapter 5

CONCLUSIONS

We have presented a general formulation for generating sensitivity coefficients that is compatible with a fully-implicit, finite-difference solution of the three-phase flow black-oil equations. The method can be applied to single-phase oil or gas flow or to multiphase flow problems and is not restricted to cases where the physical assumptions necessary in order to employ highly efficient streamline simulators in the history matching process apply. The formulation given here allows one to construct the adjoint equations directly from information computed in solving the finite-difference equations using the Newton-Raphson algorithm. The advantage of the adjoint method is that the number of matrix solutions required to compute the sensitivity coefficients is independent of the number of reservoir model parameters to be estimated. Computation of these sensitivities allows one to perform history matching using the Levenberg-Marquardt or Gauss-Newton method which are approximately quadratically convergent as opposed to more slowly converging conjugate gradient or variable metric methods. With these sensitivity coefficients, one can calculate the a posteriori covariance matrix, evaluate the reduction in uncertainty obtained by conditioning to production data and evaluate the information content of different types of data.

Dimensionless sensitivity coefficients allow one to determine which model parameters have the greatest influence on production data. Dimensionless sensitivity coefficients by themselves, however, do not always clearly indicate the value of a particular data type, i.e., do not always give a clear indication of the reduction in the uncertainty in model parameters that will be achieved by conditioning to the particular data. This is mainly due to the fact that dimensionless sensitivity coefficients do not consider the correlation between model parameters in either the prior or a posteriori model. Examination of the a posteriori variances gives one measure of the reduction in uncertainty achieved by history matching production data using the prior model for regularization. Based on this measure of

uncertainty, conditioning to all production data (pressure, GOR and WOR) gives a greater reduction in uncertainty than is obtained by conditioning to only pressure, only pressure and GOR or only pressure and WOR. One should bear in mind, however, that these a posteriori variances only provide an approximation of the value of conditioning data, in that they represent the variances of the marginal distributions of individual variables and ignore the a posteriori correlation between model parameters.

Six minimization algorithms, GN, LM, PCG, BFGS, SBFGS and LBFGS were implemented into our history matching code and compared based on the convergence rate and memory. Our theoretical analysis and results indicate that LBFGS and LBFGS-P-CG are the most promising algorithms for large problems in which the number of data and number of model parameters are both large. An improved scaling procedure for the LBFGS algorithm is proposed. Our results show that using the approximation to the inverse of the Hessian matrix generated from LBFGS as a preconditioner can improve the convergence properties of conjugate gradient method significantly.

For history-matching problems in which large amounts of data are acquired, we showed that it is possible to construct a model correction vector δm from a reduced basis, whose dimension is much less than the number of data or the number of model parameters, without affecting the result or the number of iterations required to obtain a minimum. The main problems for implementation were to determine how many basis vectors are required and how to choose a good set of basis vectors. Our solution was to partition the data objective function (the sum of the squared data mismatch) into a relatively small number of sub-objective functions. The product of the prior model covariance matrix with gradients of the data sub-objective functions provides a good set of subspace vectors for history matching. Partitioning the data by well and then by time interval is an effective method of choosing subspace vectors. We found that the efficiency of the method is not very sensitive to the details of the partitioning. Any reasonable partitioning of the data gave similar results.

When we used a fixed number of subspace vectors for every Newton iteration, the dimension of the basis had a large impact on the efficiency of the method. If too few subspace vectors were chosen, the number of Newton iterations required became very large. The initial rate of reduction in the objective function is largely independent of the number of subspace vectors, however. Because the computation of the gradient of sub-objective functions is expensive, it is clear that the computational effort can be reduced by starting with a small number of basis vectors for the first Newton iteration, and gradually increasing the number of basis vectors at subsequent iterations. Unfortunately, the process of increasing the number of basis vectors at each Newton iteration made the minimization process more

unstable. We found that the efficiency could be improved by using a modified form of the Levenberg-Marquardt algorithm in which an optimal damping parameter was computed at each Newton iteration.

The work on time-lapse seismic is at a preliminary stage, but we have concluded that it will be possible to incorporate change in impedance data into reservoir characterization using methodologies that are quite similar to the methodologies we have used to incorporate production data. Importantly, we found that it will not be necessary to add additional model parameters for shaliness, because while the effect of variation in shaliness on impedance is quite large, it has a much smaller effect on the change of impedance data that we will use in our studies.

Technical References

Y. Abacioglu, D. S. Oliver, and A. C. Reynolds. Efficient history-matching using subspace vectors. In *TUPREP Research Report 17 (May 22, 2000)*, pages 69–90, 2000.

Y. Abacioglu, D. S. Oliver, and A. C. Reynolds. Efficient reservoir history matching using subspace vectors. *Computations Geosciences*, 5:151–172, 2001.

Yafes Abacioglu. *The Use of Subspace Methods for Efficient Conditioning of Reservoir Models to Production Data*. Ph.D. thesis, University of Tulsa, Tulsa, Oklahoma, 2001.

F. Anterion, B. Karcher, and R. Eymard. Use of parameter gradients for reservoir history matching, SPE-18433. In *10th SPE Reservoir Simulation Symp.*, pages 339–354, 1989.

O. Axelsson. *Iterative Solution Methods*. Cambridge University Press, New York, 1994.

Khalid Aziz and A. Settari. *Petroleum Reservoir Simulation*. Elsevier Applied Science Publishers, London, 1979.

George E. Backus. Long-wave elastic anisotropy produced by horizontal layering. *Journal of Geophysical*, 67(11):4427–4440, 1962.

Zhuoxin Bi. *Conditioning 3D Stochastic Channels to Well-Test Pressure Data*. Ph.D. thesis, University of Tulsa, Tulsa, Oklahoma, 1999.

Robert Bissell. Calculating optimal parameters for history matching. In *4th European Conference on the Mathematics of Oil Recovery*, 1994.

Robert Bissell, O. Dubrule, P. Lamy, P. Swaby, and O. Lepine. Combining goestatistical modelling with gradient information for history matching: The pilot point method, SPE

38730. In *Proceedings of the 1997 SPE Annual SPE Technical Conference and Exhibition*, pages 139–154, 1997.

Robert Bissell, Yogeshwar Sharma, and J. E. Killough. History matching using the method of gradients: Two case studies. *SPE 69th Annual Technical Conference and Exhibition*, SPE 28590:275–289, 1994.

Richard P. Brent. *Algorithms for Minimization without Derivatives*. Prentice-Hall, Englewood Cliffs, NJ, 1973.

C. G. Broyden. Quasi-Newton methods and their application of function minimization. *Maths. Comp.*, 21:368–381, 1967.

C. G. Broyden. The convergence of a class of double rank minimization algorithm parts i and ii. *J. of Institute of Mathematics and its Applications*, 6:76–90 and 222–231, 1970.

A.G. Buckley. A combined conjugate-gradient quasi-Newton minimization algorithm. *Mathematical Programming*, 15:200–210, 1978.

A.G. Buckley and A. Lenir. QN-like variable storage conjugate gradients. *Mathematical Programming*, 27:155–175, 1983.

R. D. Carter, L. F. Kemp, and A. C. Pierce. Discussion of comparison of sensitivity coefficient calculation methods in automatic history matching. *Soc. Petrol. Eng. J.*, pages 205–208, 1982.

R. D. Carter, L. F. Kemp, A. C. Pierce, and D. L. Williams. Performance matching with constraints. *Soc. Petrol. Eng. J.*, 14(4):187–196, 1974.

Guy M. Chavent, M. Dupuy, and P. Lemonnier. History matching by use of optimal control theory. *Soc. Petrol. Eng. J.*, 15(1):74–86, 1975.

W. H. Chen, G. R. Gavalas, John H. Seinfeld, and Mel L. Wasserman. A new algorithm for automatic history matching. *Soc. Petrol. Eng. J.*, pages 593–608, 1974.

L. Chu and A. C. Reynolds. A general efficient method for generating sensitivity coefficients. In *Well Testing, Reservoir Characterization and Reservoir Simulation*, Petroleum Reservoir Exploitation Projects, pages 100–133. University of Tulsa, 1995.

Lifu Chu, M. Komara, and R. A. Schatzinger. An efficient technique for inversion of reservoir properties using iteration method. *SPE Journal*, 5(1):71–81, 2000.

Lifu Chu, Albert C. Reynolds, and Dean S. Oliver. Computation of sensitivity coefficients for conditioning the permeability field to well-test data. *In Situ*, 19(2):179–223, 1995a.

Lifu Chu, Albert C. Reynolds, and Dean S. Oliver. Reservoir description from static and well-test data using efficient gradient methods. *SPE 29999 (1995 SPE International Meeting in Beijing)*, page 16 pages, 1995b.

G. de Marsily, G. Lavedan, M. Boucher, and G. Fasanino. Interpretation of interference tests in a well field using geostatistical techniques to fit the permeability distribution in a reservoir model. In *Geostatistics for Natural Resources Characterization, Part 2*, pages 831–849. D. Reidel, 1984.

R. Fletcher. A new approach to variable metric algorithms. *Computer Journal*, 13:317–322, 1970.

R. Fletcher and M. J. D. Powell. A rapidly convergent descent method for minimization. *Computer Journal*, 6:163–168, 1963.

R. Fletcher and C. M. Reeves. Function minimization by conjugate gradient. *Computer Journal*, 7:149–154, 1964.

Roger Fletcher. *Practical Methods of Optimization*. John Wiley & Sons, New York, second edition, 1987.

G. R. Gavalas, P. C. Shah, and John H. Seinfeld. Reservoir history matching by Bayesian estimation. *Soc. Petrol. Eng. J.*, 16(6):337–350, 1976.

D. Goldfarb. A family of variable-metric methods derived by variational means. *Maths. Comp.*, 24:23–26, 1970.

J. Jaime Gómez-Hernández and André G. Journel. Joint sequential simulation of multigaussian fields. In A. Soares, editor, *Geostatistic Troia 92*, pages 133–144. 1992.

Olivier Gosselin, Alberto Cominelliand Stanislas van den Berg, and Sudip Dey Chowdhury. A gradient-based approach for history-matching of both production and 4d seismic data. Technical report, preprint, 2001.

D. Han. *Effects of porosity and clay content on acoustic properties of sandstones and unconsolidated sediments*. PhD thesis, Stanford University, 1986.

N. He, A. C. Reynolds, and D. S. Oliver. Three-dimensional reservoir description from multiwell pressure data and prior information. *Soc. Pet. Eng. J.*, pages 312–327, 1997.

Nanqun He. *Three Dimensional Reservoir Description by Inverse Theory using Well-Test Pressure and Geostatistical Data*. PhD thesis, University of Tulsa, 1997.

Nanqun He, Albert C. Reynolds, and Dean S. Oliver. Three-dimensional reservoir description from multiwell pressure data and prior information (SPE-36509). In *1996 SPE Annual Technical Conference and Exhibition*, 1996.

Bjørn Kåre Hegstad and Henning Omre. A comparison of rejection sampling and Metropolis-Hastings algorithm. Technical report, Norwegian University of Science and Technology, 1997.

M. R. Hestenes and E. S. Stiefel. Methods of conjugate gradients for solving linear systems. *J. Res. Nat. Bur. Stand*, 46:409–536, 1952.

P. Jacquard. Théorie de l'interprétation des mesures de pression. *Revue de L'Institut Français du Pétrole*, 19(3):297–334, 1964.

P. Jacquard and C. Jain. Permeability distribution from field pressure data. *Soc. Petrol. Eng. J.*, 5(4):281–294, 1965.

Hans O. Jahns. A rapid method for obtaining a two-dimensional reservoir description from well pressure response data. *Soc. Petrol. Eng. J.*, 6(12):315–327, 1966.

R. Kalita and A. C. Reynolds. Application of the conjugate gradient method to conditioning a gas reservoir model to pressure data. In *Well Testing, Reservoir Characterization and Reservoir Simulation*, Petroleum Reservoir Exploitation Projects, pages 107–126. University of Tulsa, 2000.

Rintu Kalita. *Conditioning a Three Dimensional Reservoir Model to Gas Production Data*. M.S. thesis, University of Tulsa, Tulsa, Oklahoma, 2000.

B. L. N. Kennett and P. R. Williamson. Subspace methods for large-scale nonlinear inversion. In *Mathematical Geophysics*, pages 139–154. D. Reidel, 1988.

J. E. Killough, Yogeshwar Sharma, Alain Dupuy, Robert Bissell, and John Wallis. A multiple right hand side iterative solver for history matching SPE 29119. In *Proceedings of the 13th SPE Symposium on Reservoir Simulation*, pages 249–255, 1995.

Peter K. Kitanidis. Quasi-linear geostatistical theory for inversing. *Water Resour. Res.*, 31 (10):2411–2419, 1995.

Klaas Koster, Pieter Gabriels, Matthias Hartung, John Verbeek, Geurt Deinum, and Rob Staples. Time-lapse seismic surveys in the North Sea and their business impact. *The Leading Edge*, 19(3):286–293, 2000.

Jorge L. Landa and Roland N. Horne. A procedure to integrate well test data, reservoir performance history and 4-D seismic information into a reservoir description (SPE-38653). In *1997 SPE Annual Technical Conference and Exhibition*, 1997.

T. Y. Lee and John H. Seinfeld. Estimation of two-phase petroleum reservoir properties by regularization. *J. Computational Physics*, 69:397–419, 1987a.

Tai-Yong Lee and John H. Seinfeld. Estimation of absolute and relative permeabilities in petroleum reservoirs. *Inverse Problems*, 3(4):711–728, 1987b.

Kenneth Levenberg. A method for the solution of certain non-linear problems in least squares. *Quarterly of Applied Mathematics*, 2:164–168, 1944.

Rick Lindsay and Rod van Koughnet. Sequential Backus Averaging: Upscaling well logs to seismic wavelengths. *The Leading Edge*, 20(2):188–191, 2001.

D. Liu and J. Nocedal. On the limited memory BFGS method for large scale optimization. *Math. Programming*, 45:503–528, 1989.

Randall L. Mackie and Theodore R. Madden. Three-dimensional magnetotelluric inversion using conjugate gradients. *Geophys. J. Int.*, 115:215–229, 1993.

Eliana M. Makhlouf, Wen H. Chen, Mel L. Wasserman, and John H. Seinfeld. A general history matching algorithm for three-phase, three-dimensional petroleum reservoirs. *SPE Advanced Technology Series*, 1(2):83–91, 1993.

Donald W. Marquardt. An algorithm for least-squares estimation of nonlinear parameters. *J. Soc. Indust. Appl. Math.*, 11(2):431–441, 1963.

Kiyoshi Masumoto. Pressure derivative matching method for two phase fluid flow in heterogeneous reservoir. *SPE-59462*, 2000.

Stephen G. Nash and Jorge Nocedal. A numerical study of the limited memory BFGS method and the truncated-Newton method for large scale optimization. *SIAM. J. OPTM.*, 1(3): 358–372, 1991.

L. Nazareth. A relationship between the BFGS and conjugate gradient algorithms and its implications for new algorithms. *SIAM. J. NUMER. ANAL.*, 16(5):794–800, 1979.

Jorge Nocedal. Updating quasi-Newton matrices with limited storage. *Math. Comp.*, 35: 773–782, 1980.

D. W. Oldenburg, P. R. McGillivray, and R. G. Ellis. Generalized subspace methods for large-scale inverse problems. *Geophys. J. Int.*, 114(1):12–20, 1993.

Douglas W. Oldenburg and Yaoguo Li. Subspace linear inverse method. *Inverse Problems*, 10:915–935, 1994.

Dean S. Oliver, Nanqun He, and Albert C. Reynolds. Conditioning permeability fields to pressure data. In *European Conference for the Mathematics of Oil Recovery, V*, pages 1–11, 1996.

Henning Omre, Bjørn Kåre Hegstad, and Håkon Tjelmeland. Alternative history matching approaches. Technical report, Department of Mathematical Sciences, Norwegian University of Science & Technology, Trondheim, Norway, 1996.

Henning Omre, Håkon Tjelmeland, Yuanchang Qi, and Leif Hinderaker. Assessment of uncertainty in the production characteristics of a sand stone reservoir. In *Reservoir Characterization III*, pages 556–603. PennWell Books, Tulsa, OK, 1993.

S. S. Oren. Self-scaling variable metric algorithm without line-search for unconstrained minimization. *Mathematics of Computation*, 27:873–885, 1973.

S. S. Oren. On the selection of parameters in self-scaling variable metric algorithms. *Mathematical Programming*, 7:351–367, 1974a.

S. S. Oren. Self-scaling variable metric (SSVM) algorithms II: Implementation and experiments. *Management Science*, 20:863–874, 1974b.

Robert L. Parker. *Geophysical Inverse Theory*. Princeton University Press, Princeton, New Jersey, 1994.

D. W. Peaceman. Interpretation of well block pressures in numerical reservoir simulation. *Soc. Pet. Eng. J.*, 18(6):183–194, 1978.

D. W. Peaceman. Interpretation of well-block pressures in numerical reservoir simulation with non-square grid blocks and anisotropic permeability. *Soc. Pet. Eng. J.*, 23(6):531–543, 1983.

E. Polak. *Computational methods in optimization: a unified approach*. Academic Press, London, 1971.

M.J.D. Powell. Restart procedures for the conjugate gradient method. *Mathematical Programming*, 12:241–254, 1977.

Banda S. RamaRao, A. Marsh LaVenue, Ghislain de Marsily, and Melvin G. Marietta. Pilot point methodology for automated calibration of an ensemble of conditionally simulated transmissivity fields, 1. Theory and computational experiments. *Water Resour. Res.*, 31 (3):475–493, 1995.

Albert C. Reynolds, Nanqun He, Lifu Chu, and Dean S. Oliver. Reparameterization techniques for generating reservoir descriptions conditioned to variograms and well-test pressure data. *Soc. Petrol. Eng. J.*, 1(4):413–426, 1996.

G. B. Savioli and C. A. Grattoni. On the inverse problem application to reservoir characterization. *SPE-025522*, 1992.

P. C. Shah, G. R. Gavalas, and J. H. Seinfeld. Error analysis in history matching: The optimum level of parameterization. *Soc. Petrol. Eng. J.*, 18(6):219–228, 1978.

D. F. Shanno. Conditioning of quasi-Newton method for function minimization. *Maths. Comp.*, 24:647–656, 1970.

D. F. Shanno. Conjugate gradient methods with inexact searches. *Mathematics of Operation Research*, 3:244–256, 1978a.

D. F. Shanno. On the convergence of a new conjugate gradient algorithm. *SIAM Journal on Numerical Analysis*, 15:1247–1257, 1978b.

D. F. Shanno and Kang-Hoh Phua. Matrix conditioning and nonlinear optimization. *Mathematical Programming*, 14:149–160, 1978.

Thomas B. Tan. A computationally efficient gauss-newton method for automatic history matching, SPE-29100. *Proceedings of the 13th SPE Symposium on Reservoir Simulation*, pages 61–70, 1995.

Albert Tarantola. *Inverse Problem Theory: Methods for Data Fitting and Model Parameter Estimation*. Elsevier, Amsterdam, The Netherlands, 1987.

A. N. Tikhonov. Regularization of incorrectly posed problems. *Soviet Math. Dokl.*, 4:1624–1627, 1963.

M. L. Wasserman, A. S. Emanuel, and J. H. Seinfeld. Practical applications of optimal-control theory to history-matching multiphase simulator models. *Soc. Petrol. Eng. J.*, 15 (4):347–355, 1975.

Zhan Wu. *Conditioning Geostatistical Models to Two-Phase Flow Production Data*. PhD thesis, University of Tulsa, 1999.

Zhan Wu, A. C. Reynolds, and D. S. Oliver. Conditioning geostatistical models to two-phase production data. *Soc. Petrol. Eng. J.*, 3(2):142–155, 1999.

W. Xu, T. T. Tran, R. M. Srivastava, and A. G. Journel. Integrating seismic data in reservoir modeling: the collocated cokriging approach, (SPE-24742). In *1992 SPE Annual Technical Conference and Exhibition*, 1992.

Pin-Huel Yang and A. Ted Watson. Automatic history matching with variable-metric methods. *SPE Reservoir Engineering*, 3(3):995–1001, 1988.

William W-G Yeh. Review of parameter identification in groundwater hydrology: The inverse problem. *Water Resour. Res.*, 22(2):95–108, 1986.

F. Zhang, A. C. Reynolds, and D. S. Oliver. Evaluation of the reduction in uncertainty obtained by conditioning a 3D channel to multiwell pressure data. *Mathematical Geology*, page submitted, 2000.

F. Zhang, A. C. Reynolds, and D. S. Oliver. Evaluation of the reduction in uncertainty obtained by conditioning a 3D channel to multiwell pressure data. TUPREP report, The University of Tulsa, 2001.