THE UNIVERSITY OF TULSA

THE GRADUATE SCHOOL

THE USE OF SUBSPACE METHODS FOR EFFICIENT CONDITIONING OF RESERVOIR MODELS TO PRODUCTION DATA

by
Yafes Abacioglu

A dissertation submitted in partial fulfillment of
the requirements for the degree of Doctor of Philosophy
in the Discipline of Petroleum Engineering
The Graduate School
The University of Tulsa

2001

THE UNIVERSITY OF TULSA

THE GRADUATE SCHOOL

THE USE OF SUBSPACE METHODS FOR EFFICIENT CONDITIONING OF RESERVOIR MODELS TO PRODUCTION DATA

by
Yafes Abacioglu

A DISSERTATION
APPROVED FOR THE DISCIPLINE OF
PETROLEUM ENGINEERING

By Dissertation Committee

_____ , Co-Chairperson

_____ , Co-Chairperson

_____

_____

ABSTRACT

Yafes Abacioglu (Doctor of Philosophy in Petroleum Engineering)

The Use of Subspace Methods for Efficient Conditioning of Reservoir Models to Production Data

(116 pp.-Chapter V)

Co-Directed by Dr. Albert C. Reynolds and Dr. Dean S. Oliver

(357 words)

It has been shown previously that realizations of rock property fields (simulator gridblock log-permeabilities and porosities) conditional to production data can be obtained by minimizing an objective function which includes a sum of production data mismatch terms squared plus a regularization term obtained from a prior geostatistical model. It is also well known that minimization using the Gauss-Newton method with restrictions on step length can be applied to generate such realizations. If one wishes to simulate permeability and porosity values at thousands of gridblocks by conditioning to large amounts of production data, then computation of sensitivity coefficients, solution of the Gauss-Newton matrix problem and related matrix multiplications become computationally expensive. In order to reduce the computational effort required for large-scale inversion with large amounts of data, it is necessary to use a reparameterization technique. The purpose of this study is then to develop an effective method of reparameterization for solving large-scale reservoir inverse problems with large amounts of production data using the subspace methodology.

Unlike other methods such as the pilot point method, the proposed parameterization in terms of gradients of sub-objective functions preserves the high rate of

convergence of the conventional Gauss-Newton method (with the full parameterization) while keeping the features of the resulting realizations. The product of the prior model covariance matrix with gradients of the data sub-objective functions provides a good set of subspace vectors for reservoir inverse problems. Partitioning the data by well and then by time interval is an effective method of choosing subspace vectors.

Although computation of the optimal number of subspace vectors to maintain the fast convergence of the standard Gauss-Newton method may be expensive, we show that it is desirable to start with a small number of subspace vectors and gradually increase the number at each Gauss-Newton iteration until an acceptable level of data mismatch is obtained.

An efficient implementation that uses an adjoint method to compute the subspace vectors and the "gradient simulator" method to compute sensitivities to coefficients of subspace vectors is described. These techniques eliminate the need of forming the entire sensitivity matrix directly and more importantly make the proposed subspace method applicable to multiphase problems.

TABLE OF CONTENTS

LIST OF FIGURES **viii**

LIST OF TABLES **xi**

CHAPTER 1: INTRODUCTION **1**

1.1 Automatic History Matching 3

1.2 Reparameterization Techniques 5

1.3 Scope 7

CHAPTER 2: BAYESIAN INVERSION **9**

2.1 Bayes Estimation Theory 9

2.2 Minimization of the Objective Function 11

CHAPTER 3: SUBSPACE METHODOLOGY **17**

3.1 Computational Scheme 19

3.2 The "Ideal" Reduced Basis 21

3.3 Choice and Number of Subspace Vectors 23

3.4 Levenberg-Marquardt Method with Optimal Selection of λ 27

3.5 Computation of Subspace Vectors 32

3.6 Sensitivity to Coefficients of the Subspace Vectors 34

3.7 Summary of the Recommended Procedure 38

CHAPTER 4: COMPUTATIONAL RESULTS **41**

4.1 Description of the Example Problems 41

4.2 Results from Simple Partitioning with Constant Basis Dimension . . 45

4.3 The Effect of Small and Large Basis Dimensions 52

4.4 Importance of the Choice of Subspace Vectors 58

4.5 Gradual Increase in Dimension of Basis 63

4.6 Computational Analysis 71

CHAPTER 5: CONCLUSIONS **76**

NOMENCLATURE **78**

REFERENCES **80**

APPENDIX A: SINGLE-PHASE FLOW SOFTWARE FOR CONDITIONING A GEOSTATISTICAL MODEL TO WELL-TEST PRESSURE DATA **87**

A.1 General Information 87

A.2 Source Code Files 89

A.3 Input Data Files 94

A.4 Output Files 97

APPENDIX B: ADJOINT METHOD **100**

B.1 Flow Equations 100

B.2 Adjoint Equations 105

B.3 Calculation of Subspace Vectors 108

APPENDIX C: SENSITIVITY TO COEFFICIENTS OF SUBSPACE VECTORS **110**

C.1 The Product $G_l A_l$ 110
C.2 Computation of $G_l A_l$ 111

LIST OF FIGURES

3.1 Log-permeability part of 6th pre-subspace vector of Well 1 from, (a) the direct method, and (b) the adjoint method. 34

3.2 Porosity part of 6th pre-subspace vector of Well 1 from, (a) the direct method, and (b) the adjoint method. 35

3.3 Pre-subspace vector from, (a) the direct method, and (b) the adjoint method by layer from the top of the reservoir (top) to the bottom. . . 36

3.4 70th row of GB. 38

3.5 115th row of GB. 39

4.1 Schematic of the 2-D reservoir model grid and well locations. 42

4.2 The true log-permeability field (left) and porosity field (right) for the 2-D problem. 43

4.3 Observed versus calculated pressure data for the 2-D problem. 44

4.4 Areal grid, well locations and well numbers for the 3-D problem. . . . 46

4.5 Observed pressure data at the wells for the 3-D problem. 47

4.6 True log-permeability field (above) and true porosity field (below). . . 48

4.7 Comparison of MAP estimates for 2-D example. 49

4.8 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 Gauss-Newton iteration. 51

4.9 Unconditional realizations of log-permeability (above) and porosity (below). 53
4.10 Conditional realizations of log-permeability (above) and porosity (below) from the conventional method. 54
4.11 Conditional realizations of log-permeability (above) and porosity (below) from the subspace method with 47 subspace vectors. 55
4.12 Comparison of average permeabilities. 56
4.13 The first 430 eigenvalues of the matrix $L^T G^T C_D^{-1} G L$ for 3-D example problem. 57
4.14 The number of Gauss-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. 58
4.15 The convergence behavior of the objective function for two different methods of partitioning the data with 22 subspace vectors. 61
4.16 Comparison of the objective functions for two different methods of partitioning the data with 47 subspace vectors. 62
4.17 Comparison of the results obtained by simple partitioning and SVD. . 63
4.18 Log-permeability field after first iteration. 64
4.19 The convergence is only slightly slower, but the total work is reduced when the number of subspace vectors is gradually increased at each iteration with Levenberg-Marquardt method. 66
4.20 Comparison of $O_M(m)$ versus iteration for the three methods. 67
4.21 Reduction in the objective function for a wide range of starting values of the Levenberg-Marquardt damping factor. 68
4.22 Effect of λ on the roughness for the 2-D problem with 27 subspace vectors. 69

4.23 The rates of reduction in the objective function (above) and in the optimal value of λ (below) are much different for the 2-D and 3-D problems. 72
4.24 Effect of γ on the convergence of the objective function for the 2-D problem. 73

A.1 3-D schematic of the reservoir. 97
A.2 Areal schematic of the reservoir. 98

LIST OF TABLES

4.1 Schedule of λ. 67

A.1 Input data. 91

ACKNOWLEDGMENTS

I wish to express my sincere appreciation and gratitude to Dr. Albert C. Reynolds, Jr., Professor of Petroleum Engineering and Mathematical Sciences of the University of Tulsa, and Dr. Dean S. Oliver, Associate Professor of Petroleum Engineering of the University of Tulsa for their invaluable assistance and guidance during this study. I also thank Dr. Jack R. Jones, Reservoir Engineer at BP Canada Energy, and Dr. Peyton J. Cook, Associate Professor of Mathematical Sciences of the University of Tulsa for participating in my dissertation committee and for their comments and suggestions.

I gratefully acknowledge the financial support from TUPREP (Tulsa University Petroleum Reservoir Exploitation Projects) for this study.

This work is dedicated to my wife Saadet Akgul-Abacioglu for her love and support. I especially dedicate this work to my newly born son Mustafa Selami Abacioglu.

CHAPTER I

INTRODUCTION

Estimating heterogeneous rock property fields is an important task in reservoir characterization and may have a dramatic effect on reservoir performance predictions. Unfortunately, petroleum reservoirs are often inaccessible to direct sampling of the spatial distributions of the reservoir parameters. However, pressure and/or production data (dynamic data) from wells and static data (data from core analysis, well logs, geology, seismic, etc.) are important sources of information for estimation. The reservoir parameters can be inferred from static data and from matching the history of pressure and production to those calculated by solving a mathematical model that describes the physics of fluid flow in porous media.

The mathematical model is usually too complex to be solved analytically. Instead it is approximated by a "reservoir simulator", whose input is the physical description of the reservoir. The reservoir simulator used here is a single-phase purely implicit seven-point finite difference simulator, and the gridblock values of log-permeabilities and porosities are referred to as the "model parameters." The process of finding a reservoir description by adjusting the model parameters in the reservoir simulator, so that simulated pressure and production match the observed data is referred to as "history matching" in petroleum engineering. This process is called "calibration" in the groundwater hydrology field.

When considering the problem of the determination of gridblock values of permeability and porosity from integration of static and dynamic data, it is necessary to solve an "inverse problem," which is typically underdetermined, as the number of

parameters to be estimated far exceeds the number of independent data. Therefore, the reservoir description obtained as an inverse solution by history matching is non-unique; i.e. there exist multiple descriptions of rock property fields which will reproduce the data with reasonable accuracy. As a result of non-uniqueness, we can only expect to generate probabilistic answers to questions of reservoir performance.

Our approach to the inverse problem follows the philosophy of Tarantola [53]; i.e., we use a Bayesian interpretation of probability and apply Bayes' theorem to derive the a posteriori probability density function (pdf) for the model parameters. This pdf is based on all information and data. Thus, samples of this pdf represent realizations of the rock property fields which honor all data in a statistical sense. All such realizations represent solutions to the inverse problem. Once we have a sufficient number of realizations, we can draw conclusions about the statistics of the reservoir description and, hence, the statistics of the reservoir performance. By simulating from these realizations, instead of predicting only one future performance, one can determine the variability in predictions and quantify the uncertainty in predicted reservoir performance.

The maximum a posteriori estimate of the model parameters can be obtained by minimizing a particular objective function, and a method proposed by Oliver et al. [40] and Kitanidis [28] can be used to sample the a posteriori pdf. Here, the sampling procedure given in Ref. [40] is referred to as the randomized maximum likelihood (RML) method. RML method is technically correct only for the case where the data are linearly related to the model; however, the results of Oliver et al. [40] suggest that it generates approximately correct sampling of the a posteriori pdf in the nonlinear case. In a recent study, Betancourt and Oliver [2] have shown that RML method appears to do the best job of sampling among the approximate methods that they compared.

If one wishes to generate realizations of permeability and porosity fields consisting of thousands of gridblocks by conditioning to large amounts of production

data (e.g., permanent pressure gauge data), then computational efficiency of the inversion method becomes very important. In order to reduce the computational effort required for large-scale inversion with large amounts of data, it is necessary to use a reparameterization technique. The main objective of this study is then to develop a practical procedure for reparameterization using the subspace methodology.

1.1 Automatic History Matching

History matching can be carried out either manually or automatically. The method of classical history matching used most often in practice is manual trial and error. The reservoir engineer, using experience and personal judgement, runs the reservoir simulator several times with different values of the parameters until a satisfactory match is obtained. There are various reasons that make manual history matching unattractive. Firstly, the method is recognized to be time-consuming. Secondly, it seldom honors all static data and usually results in a poor quality of the history match. Thirdly, it leads to results of questionable quality when the resulting reservoir description is used to predict future performance. Finally, it may not give any characterization of uncertainty.

Automating the history matching process does not eliminate the need for involvement of reservoir engineer's experience and skills. It is expected that partial or fully automatic history matching will reduce cycle times to complete a simulation study and enable history matching to be performed for large number of unknown parameters. Therefore, recently, the petroleum industry's interest in automatic history matching has increased significantly.

Automatic history matching is usually cast in the form of a minimization problem. A so-called objective function which quantifies the difference between the observed data and the predictions of data computed from the model parameters is minimized to compute the set of reservoir model parameters. Thus, it is impor-

tant to choose an efficient optimization algorithm to minimize the objective function. Non-gradient optimization algorithms such as simulated annealing (see, for example, Ouenes et al. [41] and Vasco et al. [55]) have often been applied as minimization methods because they are simple to implement. However, they require many thousands of iterations to converge. Since at each iteration it is required to run the reservoir simulator to evaluate the objective function, these methods are considered as computationally inefficient. Nowadays, gradient based algorithms such as Gauss-Newton method are becoming widely used for history matching because of their rapid convergence properties.

Another consideration is that reservoir history matching problems are typically ill-posed; i.e. different sets of reservoir parameter estimates may yield nearly identical matches of data. Therefore, attempts to automate the history matching process have been only partially successful. As discussed by Hadamard [17], a problem is ill-posed if the solution is not unique or if it is not a continuous function of the data (i.e., a "small" perturbation of data causes an arbitrarily "large" perturbation of the solution). Mathematically, the nonuniqueness in the history matching procedure occurs because we do not have a sufficient number of independent pressure data to determine all reservoir parameters uniquely. Because the history matching problem in a natural mathematical formulation is ill-posed, regularization is required in order to obtain a meaningful solution [38]. However, depending on the regularization procedure, one can obtain different solutions, i.e., different history matches. Moreover, in such a procedure, uncertainty in the resulting reservoir description cannot be easily characterized.

Our approach for solving the inverse problem of history matching is based on inverse problem theory [53] which provides a methodology to determine realizations of rock property fields that honor prior information when history matching pressure and/or production data. The incorporation of prior information is important, because additional independent data reduces the ill-posedness of the inverse problem and the

uncertainty in the reservoir decriptions obtained. In our work, prior information is incorporated in the form of a prior geostatistical model. We assume that this prior geostatistical model is generated from all available data except production data. The use of a prior geostatistical model acts as a regularization term.

1.2 Reparameterization Techniques

Field application of inverse problem techniques to capture the true complexity of reservoir heterogeneity will require an increase in the size of the inverse problem. A limit to the increase in the size of inverse problems is dictated by computing power. The main computational difficulties are forward modeling, calculation of the sensitivity coefficients (gradients of the computed data with respect to the model parameters) and performing the numerical linear algebra. Thus, an ideal inversion methodology requires a minimum number of forward modeling computations, a minimum amount of sensitivity information and an efficient procedure to perform the numerical linear algebra. In this regard, reparameterization methods may have some benefits in reducing the computational burden.

The basic idea of reparameterization techniques is that perturbations to the model parameters (the permeability and porosity fields), δm, are expanded as a linear combination of a relatively small number (N_B) of basis vectors a_i, i.e. $\delta m = \sum_{i=1}^{N_B} \alpha_i a_i$ or $\delta m = A\alpha$ where A is the matrix with ith column equal to the vector a_i and α is a vector of coefficients to be determined by the minimization of the objective function. In this section, we present the methods that have been published in the literature to reparameterize the reservoir model to improve the computational efficiency.

The simplest reparameterization technique is known as the method of zonation, in which the reservoir is divided into a relatively small number of zones over which the parameter is assumed to have uniform values. Thus, a basis vector a_i corresponding a specific zone has entries equal to one for the gridblocks in that zone

and zero for the other gridblocks. To the best of our knowledge Jacquard and Jain [22] and Jahns [23] used the zonation approach first for history matching purposes. A decade later, Gavalas et al. [13] and Shah et al. [52] showed that Bayesian history matching approach gave better estimates of the true permeability and porosity fields than were obtained by zonation in a simulated case of a one-dimensional reservoir. In a recent study, Bissell et al. [5] proposed a type of zonation in which gridblocks are grouped together into sets what they call gradzones. They provided a procedure to select gradzones based on the high sensitivity of various data with respect to parameters in each of the selected gridblocks. Although it is simple to apply zonation for reservoir inverse problems, it is difficult to obtain a good data match because of the small number of degrees of freedom. More importantly, any coarse zonation method yields discontinuous reservoir properties at zonation boundaries.

Gavalas et al. [13], Oliver [39] and Reynolds et al. [48] have investigated the use of eigenvectors associated with the largest eigenvalues of the covariance matrix as basis vectors for reparameterization. In cases for which the eigenvalues decay rapidly (such as the Gaussian covariance) this method was found to be highly effective in reducing the number of parameters. However, the benefits of this approach seem limited since the most common variogram models (spherical and exponential) for permeability and porosity have slowly decaying eigenvalues. Application of this approach is also limited if the variogram includes a nugget. Reynolds et al. [48] gave detailed discussion of this issue.

Shah et al. [52] proposed the use of the eigenvectors associated with the largest eigenvalues of $G^{\mathrm{T}}G$ as basis vectors. Here and in the rest of this dissertation, G denotes the sensitivity coefficient matrix. Because the dimension of $G^{\mathrm{T}}G$ is $N_{\mathrm{M}} \times N_{\mathrm{M}}$, where N_M is the number of model parameters, computation of eigenvalues and eigenvectors can clearly be expensive for large models.

The pilot point method of parameterization which perturbs reservoir properties only at selected pilot point locations to match the production data was originally

proposed by de Marsily et al. [10] in the groundwater hydrology field. This is a reduced parameterization whose basis vectors are simply the columns of the prior covariance matrix corresponding to the pilot point locations. The pilot point method has been applied to synthetic and field cases by several researchers [29, 47, 30, 16] in the groundwater hydrology field. In recent years, some researchers [59, 56, 4, 50] have adapted the same idea to history matching. The drawback of this method is that, however, it can result in overshoot at the pilot point locations (see, for example, Xue and Datta-Gupta [59]).

In the geophysical literature, Kennett and Williamson [26], Oldenburg and others [36, 37] suggested parameterizations based on subdividing the objective function. Reynolds et al. [48] applied this technique to history matching.

1.3 Scope

To quantify uncertainty in future reservoir performance and optimize reservoir management, it is required to have computationally efficient automatic history-matching techniques that can accommodate both static and dynamic data for generating plausible reservoir models. In this study, we describe an efficient method of history-matching in which changes to the reservoir model are constructed from a limited set of basis vectors. The purpose of this reparameterization is to reduce the cost of a Gauss-Newton iteration, without altering the final estimate of model parameters and without substantially slowing the rate of convergence. The utility of a subspace method depends on several factors, including the choice and number of the subspace vectors to be used. Computational gains in efficiency result partly from a reduction in the size of the matrix system that must be solved in a Gauss-Newton iteration. More important contributions, however, result from a reduction in the number of sensitivity coefficients that must be computed, reduction in the dimensions of the matrices that must be multiplied, and elimination of matrix products involving the

inverse of the prior model covariance matrix. These factors affect the efficiency of each Gauss-Newton iteration. Although computation of the optimal set of subspace vectors may be expensive, we show that the rate of convergence and the final results strongly depend on the number of subspace vectors but they are somewhat insensitive to the choice of subspace vectors. We also show that it is desirable to start with a small number of subspace vectors and gradually increase the number at each Gauss-Newton iteration until an acceptable level of data mismatch is obtained. However, the process of increasing the number of subspace vectors at each Gauss-Newton iteration makes the minimization process more unstable. We use the Levenberg-Marquardt method with optimal selection of damping factor to resolve this problem.

In this study, we consider the problem of generating the maximum a posteriori (MAP) estimates and realizations of log-permeability and porosity fields conditioned to pressure data for only single-phase flow for both two-dimensional and three-dimensional cases. However, the subspace methodology developed in this study should be applicable to multiphase flow problems.

The dissertation is structured as follows. In Chapter II, we present Bayes' theorem to derive the objective function and discuss the methods to minimize the objective function. In Chapter III, we present the equations for the computational implementation of the subspace method and discuss several aspects related with the methodology. In Chapter IV, we present the computational examples. Finally, Chapter V presents the conclusions of this study.

CHAPTER II

BAYESIAN INVERSION

Our objective is to derive the a posteriori probability density function (pdf) which represents the solution of the inverse problem, and to generate a set of realizations of the model parameters which represents a sampling of this pdf. We use a Bayesian interpretation of probability and apply Bayes' theorem to derive the a posteriori pdf for the model parameters.

M denotes the random vector of model parameters, with specific realizations denoted by m. We assume that the prior model for the rock property fields is multivariate Gaussian, characterized by means and covariances (arising from a specified variogram) with permeability assumed to be log-normal, porosity assumed to be normal with correlation between log-permeability and porosity determined by specifying a correlation coefficient. Cross covariances are modeled by using the "screening hypothesis" of Xu et al. [58] which implies that the variograms for porosity and log-permeability are of the same type, and have identical ranges, but different sills (variances).

2.1 Bayes Estimation Theory

Bayes' theorem is applied to update prior uncertain knowledge on parameters by conditioning the parameters to new data that have become available. For our purpose, Bayes' theorem can be expressed as

$$p_{M|D}\left(m|d\right) = \frac{p_{D|M}\left(d|m\right) p_M(m)}{p_D(d)}. \tag{2.1}$$

Thus, the prior pdf $p_M(m)$ of the model parameters is updated according to Eq. 2.1 to obtain the a posteriori pdf $p_{M|D}(m|d)$ of the model parameters, given the data, d. The two ingredients needed for the update are $p_{D|M}(d|m)$, called the likelihood function, and $p_D(d)$ which is a scaling constant in our estimation problem since it does not depend on the model parameters.

Based on our assumption of a multinormal distribution, the prior distribution has a probability relation

$$p_M(m) \propto \exp\left[-\frac{1}{2}(m - m_p)^T C_M^{-1}(m - m_p)\right], \tag{2.2}$$

where m_p is the vector containing the estimates of the prior means of the model parameters and C_M is the prior covariance matrix. We let d be a data vector where the relationship between the data and the model m is given by

$$d = g(m), \tag{2.3}$$

and let D denote the random data vector (D is a random vector because M is random). In our application, d contains calculated wellbore pressures obtained from the model using the reservoir simulator. The functional relationship of Eq. 2.3 represents the effect of generating d from our reservoir simulator.

For a given set of data, d_{obs}, the likelihood function for the model is given by the following equation

$$p_{D|M}(d_{obs}|m) \propto \exp\left[-\frac{1}{2}(g(m) - d_{obs})^T C_D^{-1}(g(m) - d_{obs})\right]. \tag{2.4}$$

In Eq. 2.4, d_{obs} refers to the vector of observed or measured wellbore pressures. We model measurement errors as independent identically distributed Gaussian random variables with zero mean and variance σ_d^2. Thus, the covariance matrix for the measurement errors is a diagonal matrix C_D with all diagonal entries equal to σ_d^2.

Using Eqs. 2.2 and 2.4 in Eq. 2.1, the a posteriori probability density function

is obtained as

$$p_{M|D}\left(m|d_{obs}\right) = a \exp\Big[-\frac{1}{2}\left(m-m_p\right)^T C_M^{-1}\left(m-m_p\right) - \frac{1}{2}\left(g(m)-d_{obs}\right)^T C_D^{-1}\left(g(m)-d_{obs}\right)\Big], \quad (2.5)$$

where a simply denotes a generic normalizing constant which ensures that pdf integrates to one.

The most probable model, referred to as the maximum a posteriori estimate (MAP), is the one that maximizes Eq. 2.5, or equivalently, minimizes the objective function

$$O(m) = \frac{1}{2}\left(m-m_p\right)^T C_M^{-1}\left(m-m_p\right) + \frac{1}{2}\left(g(m)-d_{obs}\right)^T C_D^{-1}\left(g(m)-d_{obs}\right). \quad (2.6)$$

Realizations of rock property fields conditioned to pressure data can be generated by sampling the a posteriori pdf using the procedure suggested by Oliver et al. [40] and Kitanidis [28]. In this procedure, m_p is replaced by an unconditional simulation, m_{uc}, of m from the prior model and d_{obs} is replaced by an unconditional simulation of the pressure data, d_{uc}, in Eq. 2.6. Thus, Eq. 2.6 is replaced by

$$O(m) = \frac{1}{2}\big(m-m_{uc}\big)^T C_M^{-1}\big(m-m_{uc}\big) + \frac{1}{2}\big(g(m)-d_{uc}\big)^T C_D^{-1}\big(g(m)-d_{uc}\big). \quad (2.7)$$

This procedure is referred to as Randomized Maximum Likelihood (RML) method. In the remainder of the dissertation, the equations are presented in terms of generating realizations using the RML method.

2.2 Minimization of the Objective Function

In the previous section, we have shown that history-matching realizations of a stochastic model to nonlinear data can be formulated as an optimization problem; i.e. we defined automatic history matching as the process of minimizing the objective function of Eq. 2.7 by means of an optimization algorithm. Then, the objective

becomes to find a solution using the most efficient algorithm. In other words, we wish to use an algorithm which converges to a solution as quickly as possible.

The optimization algorithms can be classified depending on whether they use gradient information or not. The most commonly used non-gradient algorithms for reservoir description problems are simulated annealing [41, 55], genetic algorithms [51], Monte Carlo methods [54, 21] and neural networks [42]. These methods seem to be attractive because they are simple to implement and do not require the computation of gradients of the objective function or data. However, they are very expensive in terms of computation since they require the value of the objective function at a large number of points in the parameter space. This issue becomes very important when objective function evaluations involve the use of a reservoir simulator. On the other hand, gradient methods require the gradient of the objective function or sensitivity coefficients. This additional gradient information reduces the number of objective function evaluations and makes gradient methods converge much more rapidly, especially in the neighborhood of a minimum. The well-known gradient algorithms are steepest descent, conjugate gradients, Newton's method, the Gauss-Newton method and the quasi-Newton method (variable-metric method).

Gradient methods that do not require computation of data sensitivities or the Hessian matrix are standard for solving large unconstrained minimization problems. Jahns [23], Chen et al. [8] and others have used conjugate gradient or steepest decent for history matching. While the computational requirement for each iteration is relatively low, the number of iterations required can be large, especially for multiphase flow problems [34]. In a recent study done at the University of Tulsa, Kalita [25] investigated the use of the conjugate gradient method as an alternative to commonly used Gauss-Newton method with restricted-step. He considered the problem of generating the maximum a posteriori estimates and realizations of log-permeability and porosity fields conditioned to pressure data for single-phase flow of gas for both two-dimensional and three-dimensional cases. In most cases he con-

sidered, the Gauss-Newton method required far fewer iterations than the conjugate gradient algorithm to obtain convergence. He concluded that even though one iteration in Gauss-Newton is much more time consuming than one conjugate gradient iteration, the high number of iterations required in conjugate gradient pose a threat to its implementation in the current form for a large scale problem. Quasi-Newton or variable-metric mehods have some of the convergence properties of Gauss-Newton without the need to compute the Hessian. In this method, the algorithm computes an approximate Hessian which is updated after each iteration based on the change in the gradient of the objective function. Yang et al. [60] investigated the use of two variable-metric methods – the Broyden/Fletcher/Goldfarb/Shanno (BFGS) method and a self-scaling variable-metric (SSVM) method – for hypothetical two-phase reservoir history-matching problems. Both methods yielded accurate results with less computer effort than the steepest decent method. Deschamps et al. [11] performed a similar study in which various combinations of Gauss-Newton, quasi-Newton and steepest descent were compared. They concluded that the hybrid methods combining steepest descent with Gauss-Newton were most efficient for the relatively small problems and that quasi-Newton methods would be required for large practical problems.

In this work we use Gauss-Newton and Levenberg-Marquardt algorithms for minimization of the objective function. The Levenberg-Marquardt algorithm can be thought of as a modification of the Gauss-Newton algorithm. These algorithms will be respectively explained in detail in the following two sub-sections.

2.2.1 Gauss-Newton Method

Newton's method can be derived for minimizing the objective function of Eq. 2.7 using the second-order approximation to the objective function using the Taylor-series expansion about m^l, i.e.,

$$O(m^l + \delta m) \approx q^l(\delta m) = O(m^l) + \nabla O(m^l)^T \delta m + \frac{1}{2}\delta m^T H_l \delta m, \tag{2.8}$$

where $\delta m = m - m^l$; H_l is the Hessian matrix (second derivative of the objective function) and $q^l(\delta m)$ is the resulting quadratic approximation for iteration l. Then the lth iteration of Newton's method can be written

$$H(m^l)\delta m^l = -\nabla O(m^l), \tag{2.9}$$

and the iterate m^{l+1} is obtained by

$$m^{l+1} = m^l + \delta m^l, \tag{2.10}$$

where the correction δm^l minimizes $q^l(\delta m)$, i.e. δm^l is defined by the condition that $\nabla q^l(\delta m^l) = 0$.

In Eq. 2.9, it is necessary to compute the gradient and the Hessian of the objective function. Using the basic vector calculus, the gradient of the objective function (Eq. 2.7) is obtained as

$$\nabla O(m^l) = C_{\mathrm{M}}^{-1}(m^l - m_{uc}) + G_l^{\mathrm{T}} C_{\mathrm{D}}^{-1}\big(g(m^l) - d_{uc}\big), \tag{2.11}$$

and the Hessian matrix of $O(m^l)$ is obtained as

$$H(m^l) = \nabla[(\nabla O(m^l))^T] = C_{\mathrm{M}}^{-1} + G_l^{\mathrm{T}} C_{\mathrm{D}}^{-1} G_l + (\nabla G_l^T) C_{\mathrm{D}}^{-1}\big(g(m^l) - d_{uc}\big). \tag{2.12}$$

However, in the Gauss-Newton algorithm, the Hessian matrix of Eq. 2.12 is approximated by

$$H(m^l) \cong C_{\mathrm{M}}^{-1} + G_l^{\mathrm{T}} C_{\mathrm{D}}^{-1} G_l, \tag{2.13}$$

where G, is the gradient of the computed data with respect to the model parameters; i.e. $G_l = (\nabla g^{\mathrm{T}})^{\mathrm{T}}$ evaluated at m^l. Substituting Eqs. 2.11 and 2.13 into Eq. 2.9 gives the Gauss-Newton equation, i.e.

$$\big(C_{\mathrm{M}}^{-1} + G_l^{\mathrm{T}} C_{\mathrm{D}}^{-1} G_l\big)\,\delta m^l = -C_{\mathrm{M}}^{-1}(m^l - m_{uc}) - G_l^{\mathrm{T}} C_{\mathrm{D}}^{-1}\big(g(m^l) - d_{uc}\big). \tag{2.14}$$

Determination of δm^l using Eq. 2.14 requires the solution of a system of $N_{\rm M}$ equations where $N_{\rm M}$ is the number of model parameters. An alternative formulation of the Gauss-Newton equation,

$$\delta m^l = (m_{uc} - m_l) - C_{\rm M} G_l^T \left(C_{\rm D} + G_l C_{\rm M} G_l^T\right)^{-1} \left[g(m) - d_{uc} - G_l(m^l - m_{uc})\right], \quad (2.15)$$

can be obtained from the Sherman-Morrison-Woodbury matrix inversion formula [14, 18]. A historical survey of the Sherman-Morrison-Woodbury formulas is given by Hager [18] in the literature. This alternative formulation of Eq. 2.15 has an inverse matrix on the right side which has a dimension of $N_D \times N_D$ where N_D is the number of conditioning data. We generally have $N_D < N_M$, thus Eq. 2.15 will normally be more efficient. 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 N_D and N_M are large. In this case, the standard Gauss-Newton approach is generally too expensive to use for minimization.

2.2.2 Levenberg-Marquardt Method

Reynolds and Oliver and their coworkers (see, for example, [20, 48]) have used the restricted-step procedure [12] in the Gauss-Newton method to control the step size at each iteration of minimization. In some of their applications of conventional Bayesian inversion, they found that using the Gauss-Newton method with restricted-step gives a rough model at the first iteration and the roughness does not go away at the later iterations [57]. In such cases, the Gauss-Newton method converges to a local minimum which yields a rough model. Similarly, in our applications of the subspace method where we start with a small number of subspace vectors in the early iterations and gradually increase the number of subspace vectors as needed, the addition of new subspace vectors usually resulted in abrupt increase in the model roughness at the next iteration. Again, the Gauss-Newton method may converge to a local minimum with a rough model. This problem can be avoided using the ideas due to Levenberg

[32] and Marquardt [35]. We implement the Levenberg-Marquardt algorithm in a slightly non-standard way. Instead of adding a diagonal matrix, λI (I is an identity matrix) to the Hessian, we multiplied the inverse of the model covariance by a factor of $1 + \lambda$ in the Hessian. Thus, the Levenberg-Marquardt equation is obtained by modifying the Gauss-Newton iteration (Eq. 2.14) to

$$\left[(1+\lambda)C_{\mathrm{M}}^{-1} + G_l^{\mathrm{T}} C_{\mathrm{D}}^{-1} G_l\right] \delta m^l = -C_{\mathrm{M}}^{-1}(m^l - m_{uc}) - G_l^{\mathrm{T}} C_{\mathrm{D}}^{-1}\big(g(m^l) - d_{uc}\big), \quad (2.16)$$

where λ is a positive number. Similar to the Gauss-Newton equation (Eq. 2.15), an alternative formulation can be obtained as

$$\delta m^l = -\frac{m^l - m_{uc}}{1+\lambda} + C_{\mathrm{M}} G_l^{\mathrm{T}} \left[(1+\lambda)C_{\mathrm{D}} + G_l C_{\mathrm{M}} G_l^{\mathrm{T}}\right]^{-1} \left\{\frac{G_l(m^l - m_{uc})}{1+\lambda} - \big(g(m^l) - d_{uc}\big)\right\}. \quad (2.17)$$

A detailed derivation of Eq. 2.17 can be found in Ref. [3]. The question with the Levenberg-Marquardt method is how to choose and update λ. There are several algorithms developed to solve this problem. The most commonly used one is Marquardt's method. In this method, initially a small positive value is taken for λ. However, in our applications, we use a larger starting value of λ to make the model change smoother from the beginning of the algorithm. The parameter λ is divided by a factor after each successful iteration, and is multiplied by the factor if the new parameter leads to an increased value of the objective function, i.e., if an unsuccessful step was proposed. If λ is very large, a step in the direction $-C_M \nabla O$ is performed. A lambda value of zero is equivalent to a Gauss-Newton step. The former is robust but inefficient, the latter has a quadratic convergence rate near the minimum but may lead to unsuccessful steps. The parameter λ therefore determines how fast the step size and step direction changes from steepest descent to Gauss-Newton and vice versa.

CHAPTER III

SUBSPACE METHODOLOGY

Subspace methods reduce the size of the matrix problem that must be solved at each iteration of the Gauss-Newton method. In a subspace method, the search direction vector is expanded as a linear combination of basis vectors for a lower dimensional subspace of the model space. The order of the matrix problem to be solved at each iteration of the Gauss-Newton procedure is thereby reduced to the dimension of the subspace. In section 3.1, the computational scheme for the subspace method is presented.

As addressed previously by Oldenburg et al. [36] the success or failure of a subspace method depends on the selection of the subspace vectors. Thus, the questions of "which type" and "how many" must be answered if the subspace method is to be applied successfully. In section 3.2, we show that there is an optimal number of basis vectors such that the convergence of the Gauss-Newton method is unaffected by the reparameterization. Unfortunately, computation of the optimal set of basis vectors requires computation of eigenvalues of a large matrix which will normally be too expensive to be practical. Considering the choice of subspace vectors, we use a procedure for partitioning the data mismatch of the objective function into sub-objective functions to form the subspace vectors. This procedure, which will be explained in section 3.3, does not give the optimal number of subspace vectors. To overcome the problem of the need to know optimal number of subspace vectors in advance, we use a new technique in which the Gauss-Newton algorithm is started with a few subspace vectors at early iterations and the number of subspace vectors is gradually increased at each Gauss-Newton iteration until an acceptable level of data

mismatch is obtained. When we first attempted to use this technique, we were not successful because of addition of new subspace vectors often resulted in an increase in the model roughness at the next iteration. This problem was resolved by using the implementation of the Levenberg-Marquardt method with optimal selection of damping factor which is explained in section 3.4.

Effective implementation of a subspace method to generate realizations of rock property fields conditioned to large amounts of production data and geostatistical information will also require an efficient way to generate sensitivity coefficients. In Ref. [48], a modified generalized pulse spectrum technique (MGPST) is used to generate sensitivity coefficients. Even though this technique is quite efficient, it does not yield good estimates of the sensitivity coefficients related to the porosity field. Thus, in Ref. [19], a procedure introduced by Carter et al. [6] was extended to generate efficiently sensitivity coefficients related to three-dimensional permeability and porosity fields. The drawback of this method is that it is applicable only for single-phase flow problems. Ref. [57] contains a recent review of the methods for calculating sensitivity coefficients. In sections 3.5 and 3.6, we show that the adjoint method can be used to compute the subspace vectors and the "gradient simulator" method can be implemented to compute sensitivity to coefficients of the subspace vectors. These techniques will eliminate the need to form the entire sensitivity matrix directly. In other words, these techniques will require only $2(N_B - 1)$ (N_B is the number of subspace vectors) simulation runs for the sensitivity calculations at each iteration (compared to N_D for the conventional approach). More importantly, the use of these techniques will make the proposed subspace method applicable to multiphase flow problems.

Finally, in section 3.7, we summarize the recommended procedure for use of subspace vectors in history matching.

3.1 Computational Scheme

The basic idea of the subspace procedure is that at the lth iteration of the Gauss-Newton method, one can approximate the search direction δm^l as a linear combination of a relatively small number of subspace vectors, which are denoted as a_j^l, $j = 1, 2, \ldots, N_B$, without significantly changing the δm^l obtained in solving Eq. 2.14. The change in the model estimate or search direction vector at the lth iteration of the Gauss-Newton method is then written as

$$\delta m^l = \sum_{j=1}^{N_B} \alpha_j^l a_j^l = A_l \alpha^l, \tag{3.1}$$

where A_l is the matrix with jth column equal to the column vector a_j^l, i.e.

$$A_l = \left[a_1^l, a_2^l, \ldots, a_{N_B}^l\right], \tag{3.2}$$

and

$$\alpha^l = \left[\alpha_1^l, \alpha_2^l, \ldots, \alpha_{N_B}^l\right]^T. \tag{3.3}$$

Note that α_j^ls are scalars and the superscript l is used to indicate that the subspace and its basis vectors are recomputed at each iteration of the Gauss-Newton method.

Using Eq. 3.1 in Eq. 2.9 and multiplying the resulting equation by the transpose of A_l gives

$$\left(A_l^T H_l A_l\right) \alpha^l = -A_l^T \nabla O_l. \tag{3.4}$$

Substituting Eqs. 2.11 and 2.13 into Eq. 3.4 gives

$$\left[A_l^T (C_M^{-1} + G_l^T C_D^{-1} G_l) A_l\right] \alpha^l = A_l^T [-C_M^{-1}(m^l - m_{uc}) - G_l^T C_D^{-1}\left(g(m^l) - d_{uc}\right)]. \tag{3.5}$$

The actual set of subspace vectors used at the lth iteration is defined by

$$a_j^l = C_M b_j^l, \tag{3.6}$$

$j = 1, 2, \ldots, N_B$, where the b_j^ls are defined in subsection 3.3.1. Defining B_l as the $N_M \times N_B$ matrix with jth column given by b_j^l, it follows that

$$A_l = C_M B_l. \tag{3.7}$$

Then Eq. 3.1 becomes

$$\delta m^l = A_l \alpha^l = C_M B_l \alpha^l. \tag{3.8}$$

Substituting Eq. 3.8 into Eq. 3.5 gives

$$\begin{aligned}(C_M B_l)^T (C_M^{-1} + G_l^T C_D^{-1} G_l)(C_M B_l)\alpha^l \\ = (C_M B_l)^T [-C_M^{-1}(m^l - m_{uc}) - G_l^T C_D^{-1}\big(g(m^l) - d_{uc}\big)],\end{aligned} \tag{3.9}$$

or

$$\begin{aligned}\left[B_l^T C_M B_l + (G_l C_M B_l)^T C_D^{-1} (G_l C_M B_l)\right] \alpha^l \\ = -B_l^T (m^l - m_{uc}) - \big(G_l C_M B_l\big)^T C_D^{-1}\big(g(m^l) - d_{uc}\big),\end{aligned} \tag{3.10}$$

which forms the computational scheme for the subspace method and can be solved for α^l. Once α^l is obtained, δm^l can be computed from Eq. 3.8. The primary computational advantage of Eq. 3.10 is that it avoids inversion of C_M (or the solution of linear systems with C_M as the coefficient matrix). Note that it is still necessary to evaluate the product $(m - m_{uc})^T C_{\mathrm{M}}^{-1} (m - m_{uc})$ in the objective function, to determine whether or not to stop iterating. We use a diagonal approximation to the covariance in this case without affecting the results significantly. The same approximation is used in the restricted-step procedure when we need to evaluate the objective function; however, inversion of C_M is eliminated when we evaluate the Hessian for the subspace method. Also, the matrix problem of Eq. 3.10 is an $N_B \times N_B$ problem, whereas the original Hessian is $N_M \times N_M$; see Eqs. 2.9 and Eq. 2.13. Another advantage of Eq. 3.10 is that the product $G_l C_M B_l$ can be calculated fairly efficiently using the "gradient simulator" method.

3.2 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 $C_{\mathrm{M}} = LL^{\mathrm{T}}$ be a square-root decomposition of the model covariance matrix which has a dimension of $N_M \times N_M$ and define a dimensionless vector of model corrections

$$\alpha = L^{-1}\delta m. \tag{3.11}$$

The Gauss-Newton equation for estimation of α is

$$\begin{aligned} L^{\mathrm{T}}\big(C_{\mathrm{M}}^{-1} + G^{\mathrm{T}}C_{\mathrm{D}}^{-1}G\big)L\alpha &= -L^{\mathrm{T}}\nabla O, \\ \big(I + L^{\mathrm{T}}G^{\mathrm{T}}C_{\mathrm{D}}^{-1}GL\big)\alpha &= -L^{\mathrm{T}}\nabla O, \\ \big(I + U\Lambda U^{\mathrm{T}}\big)\alpha &= -L^{\mathrm{T}}\nabla O, \end{aligned} \tag{3.12}$$

where $U\Lambda U^{\mathrm{T}}$ is the Schur decomposition of $L^{\mathrm{T}}G^{\mathrm{T}}C_{\mathrm{D}}^{-1}GL$. Λ is a diagonal matrix of eigenvalues, which are ordered such that $\lambda_i \geq \lambda_{i+1}$ and U is orthogonal. $UU^{\mathrm{T}} = I$ so

$$U(I + \Lambda)U^{\mathrm{T}}\alpha = -L^{\mathrm{T}}\nabla O. \tag{3.13}$$

The solution of this equation is clearly

$$\alpha = -U(I + \Lambda)^{-1}U^{\mathrm{T}}L^{\mathrm{T}}\nabla O, \tag{3.14}$$

or

$$\alpha = -U \begin{bmatrix} 1/(1+\lambda_1) & & & 0 \\ & 1/(1+\lambda_2) & & \\ & & \ddots & \\ 0 & & & 1/(1+\lambda_{N_M}) \end{bmatrix} U^{\mathrm{T}}L^{\mathrm{T}}\nabla O. \tag{3.15}$$

Suppose that only the first p of the eigenvalues are greater than 0.1. Then

$$\begin{aligned}\alpha &\approx -U \begin{bmatrix} 1/(1+\lambda_1) & & & & & & 0 \\ & \ddots & & & & & \\ & & 1/(1+\lambda_p) & & & & \\ & & & 1 & & & \\ & & & & \ddots & & \\ 0 & & & & & & 1 \end{bmatrix} U^{\mathrm{T}} L^{\mathrm{T}} \nabla O \\ &\approx -U \left(I - \begin{bmatrix} \lambda_1/(1+\lambda_1) & & & & & & 0 \\ & \ddots & & & & & \\ & & \lambda_p/(1+\lambda_p) & & & & \\ & & & 0 & & & \\ & & & & \ddots & & \\ 0 & & & & & & 0 \end{bmatrix} \right) U^{\mathrm{T}} L^{\mathrm{T}} \nabla O. \end{aligned} \tag{3.16}$$

Note again that $UU^T = I$ so

$$\alpha \approx - \left(I - U \begin{bmatrix} \lambda_1/(1+\lambda_1) & & & & & & 0 \\ & \ddots & & & & & \\ & & \lambda_p/(1+\lambda_p) & & & & \\ & & & 0 & & & \\ & & & & \ddots & & \\ 0 & & & & & & 0 \end{bmatrix} U^{\mathrm{T}} \right) L^{\mathrm{T}} \nabla O. \tag{3.17}$$

Defining U_p as the matrix with columns equal to the eigenvectors of $L^{\mathrm{T}} G^{\mathrm{T}} C_{\mathrm{D}}^{-1} G L$ associated with the p eigenvalues whose magnitudes are of order 0.1 or greater, Eq. 3.17

can be written as

$$\begin{aligned}\alpha &\approx -\left(I - U_p \begin{bmatrix} \lambda_1/(1+\lambda_1) & & 0 \\ & \ddots & \\ 0 & & \lambda_p/(1+\lambda_p) \end{bmatrix} U_p^{\mathrm{T}} \right) L^{\mathrm{T}} \nabla O \\ &\approx -\left[I - \sum_{k=1}^{p} \frac{\lambda_k}{\lambda_k + 1} U_k U_k^{\mathrm{T}} \right] L^{\mathrm{T}} \nabla O,\end{aligned} \tag{3.18}$$

where U_k is the kth column of U. Substituting the final term of Eq. 3.18 into Eq. 3.11 to obtain δm gives

$$\begin{aligned}\delta m &\approx -L \left[I - \sum_{k=1}^{p} \frac{\lambda_k}{\lambda_k + 1} U_k U_k^{\mathrm{T}} \right] L^{\mathrm{T}} \nabla O \\ &\approx -C_M \nabla O + \sum_{k=1}^{p} L U_k \left[\frac{\lambda_k}{\lambda_k + 1} U_k^{\mathrm{T}} L^{\mathrm{T}} \nabla O \right].\end{aligned} \tag{3.19}$$

Hence, we can say that a good approximation to δm can be constructed from the columns of LU_p. 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.

3.3 Choice and Number of Subspace Vectors

The effectiveness of the subspace methodology strongly depends on a judicious choice of subspace vectors. In the previous section, we have shown that the Schur decomposition of $L^{\mathrm{T}} G^{\mathrm{T}} C_{\mathrm{D}}^{-1} G L$ provides a way to determine an optimal number of subspace vectors. As we noted, decomposition of this matrix is too computationally inefficient to be practical. By making the number of subspace vectors small, the size of the system to be solved can be significantly reduced. However, 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. In Chapter IV, we will illustrate the effect of various choices of basis vectors with two- and three-dimensional single-phase reservoir models.

In some cases, an improper choice of subspace vectors can result in visible artifacts. This is most noticeable with methods like pilot point. When a good set of subspace vectors are chosen, the differences should be imperceptible. Refs. [36], [26] and [37] provide several procedures for choosing subspace vectors. The simplest choice is to use one subspace vector (gradient of the objective function). This choice makes the subspace method equal to the method of steepest decent which is known to be inefficient. On the other hand, subspace vectors associated with gradients of the pressure data mismatch and the model mismatch parts of the objective function of Eq. 2.7 can be used to form a two-dimensional subspace; however, this choice is also too simple to yield a rapidly converging algoritm. In this study, we use the product of the prior covariance matrix with gradients of the sub-objective functions as set of subspace vectors.

3.3.1 Partitioning of the Objective Function

Based on the ideas of Kennett and Williamson [26], Oldenburg et al. [36] and Oldenburg and Li [37], we partition the total objective function into a term that arises from model roughness and distance from the prior model and a term that arises from pressure data mismatch for the problem of conditioning rock property fields to transient pressure data. The two partial objective functions are

$$O_{\mathrm{M}}(m) = \frac{1}{2}\big(m - m_{uc}\big)^{\mathrm{T}} C_{\mathrm{M}}^{-1}\big(m - m_{uc}\big), \tag{3.20}$$

and

$$O_{\mathrm{D}}(m) = \frac{1}{2}\big(g(m) - d_{uc}\big)^{\mathrm{T}} C_{\mathrm{D}}^{-1}\big(g(m) - d_{uc}\big). \tag{3.21}$$

The data mismatch objective function can be further partitioned as

$$O_{\mathrm{D}}(m) = \sum_k O_{\mathrm{D}}^k(m), \tag{3.22}$$

where

$$O_{\mathrm{D}}^k(m) = \frac{1}{2}\big(g^k(m) - d_{uc}^k\big)^{\mathrm{T}}[C_{\mathrm{D}}^k]^{-1}\big(g^k(m) - d_{uc}^k\big), \tag{3.23}$$

d_{uc}^k is the kth set of unconditional simulation of data, and C_{D}^k is the data covariance matrix for the kth set of data. Here we have assumed that the measurement errors are either independent (in which case C_{D} is diagonal), or that the partitioning is done in such a way that data with correlated measurement errors are included in the same group. Note also that the partitioning is carried out first by well and then by time interval. Data from different wells are never grouped together.

The gradients of these sub-objective functions are given by

$$\nabla O_{\mathrm{M}}(m) = C_{\mathrm{M}}^{-1}(m - m_{uc}), \tag{3.24}$$

and

$$\nabla O_{\mathrm{D}}^k(m) = G_k^{\mathrm{T}}[C_{\mathrm{D}}^k]^{-1}\big(g^k(m) - d_{uc}^k\big). \tag{3.25}$$

This procedure is done at each iteration of the Gauss-Newton algorithm but, for now, we have deleted the iteration index l from the notation. One set of subspace vectors is formed by multiplying ∇O_{M} by C_{M} and then partitioning the resulting vector into two parts (one vector for porosity and one for permeability), i.e.

$$w_1 = \begin{bmatrix} m_\phi - m_{uc,\phi} \\ 0 \end{bmatrix}, \tag{3.26}$$

and

$$w_2 = \begin{bmatrix} 0 \\ m_k - m_{uc,k} \end{bmatrix}, \tag{3.27}$$

where, m_ϕ is the N-dimensional column vector of gridblock porosities and m_k is the N-dimensional column vector of gridblock log-permeabilities; $m_{uc,\phi}$ and $m_{uc,k}$ are the corresponding vector of unconditional realizations for these attributes. Here, N is the total number of gridblocks.

The gradients of the data sub-objective functions (Eqs. 3.25) give vectors of the form

$$w_{2+k} = G_k^{\mathrm{T}}[C_{\mathrm{D}}^k]^{-1}\big(g^k(m) - d_{uc}^k\big). \tag{3.28}$$

The set of all ws defined in Eqs. 3.26, 3.27 and 3.28 are now used to form an $N_M \times N_B$ matrix W where N_B is the number of w vectors used. These vectors, which are referred as pre-basis or pre-subspace vectors, are now denoted by w_j^l, $j = 1, 2, \ldots, N_B$ where l refers to the Gauss-Newton iteration index. Thus, at the lth iteration, the $N_M \times N_B$ matrix of pre-subspace vectors is given by

$$W_l = \left[w_1^l, w_2^l, \ldots, w_{N_B}^l\right]. \tag{3.29}$$

It is possible that the pre-subspace vectors of Eq. 3.29 may not be linearly independent, and if they are not, direct application of the subspace procedure would yield an ill-conditioned matrix problem. To ensure the pre-subspace vectors used are linearly independent and have similar magnitudes, one can apply a Gram-Schmidt procedure [26, 48], but this procedure suffers from the accumulation of round-off error if the number of subspace vectors is large. Thus, we compute a singular value decomposition (or QR decomposition) of the matrix W_l to obtain orthonormal vectors. Singular value decomposition of the matrix W_l is given by

$$W_l = B_l S_l V_l^T, \tag{3.30}$$

where B_l is an $N_M \times N_B$ column-orthogonal matrix; S_l is an $N_B \times N_B$ diagonal matrix of singular values and V_l^T is the transpose of an $N_B \times N_B$ orthogonal matrix V_l. Then, we multiply the matrix B_l by C_M to create the basis vectors, i.e.

$$A_l = C_M B_l. \tag{3.31}$$

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 decent. At the other extreme, the Gauss-Newton equation (Eq. 2.15) can be written in such a way that the basis vectors are the columns of $C_{\mathrm{M}}G^{\mathrm{T}}$, i.e.

$$\delta m^l = (m_{uc} - m_l) - \sum_{i=1}^{N_{\mathrm{D}}} \left[C_{\mathrm{M}} G_l^T\right]_i \alpha_i. \tag{3.32}$$

This is exactly the basis that would be obtained if each set of data were to include only one measurement. Hence this method of selecting basis vectors satisfies the suggestion of Parker [43] that a good depleted basis should at least be capable of approaching the true optimal solution if enough basis functions are used. The second conclusion that can be drawn from this comparison is 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. Even though Eq. 3.32 suggests that we should multiply the vectors arising from ∇O_M by C_M once, we multiply them twice. By doing so we eliminate need of C_M^{-1} without affecting the results substantially.

3.4 Levenberg-Marquardt Method with Optimal Selection of λ

As we explained earlier, it is computationally inefficient to determine the optimal number of subspace vectors in advance. To eliminate the need to know the optimal number in advance, we use an efficient strategy in which we begin with a small number of subspace vectors in the early iterations of minimization by grouping more data together to generate the data sub-objective functions and gradually increase the number of subspace vectors (i.e., partition the data mismatch objective function into more groups) at subsequent iterations. Gauss-Newton method did not work well in this case because the addition of new subspace vectors resulted in an increase in the model roughness at the next iteration. An increase in the model roughness makes

the minimization process unstable. As an alternative to Gauss-Newton method, the standard Levenberg-Marquardt algorithm (explained in subsection 2.2.2) can be used. In the subspace parameterization, the Levenberg-Marquardt equation is obtained from Eq. 3.10

$$\left[(1+\lambda)B_l^T C_M B_l + (G_l C_M B_l)^T C_D^{-1}(G_l C_M B_l)\right]\alpha^l = -B_l^T(m^l - m_{uc}) - \left(G_l C_M B_l\right)^T C_D^{-1}\left(g(m^l) - d_{uc}\right). \quad (3.33)$$

Similar to the Gauss-Newton method, the standard Levenberg-Marquardt method did not work well in our application of the subspace method because the parameters (initial λ, growth and decay factors of λ) required for efficient convergence of the Levenberg-Marquardt algorithm were problem dependent and difficult to determine without much experimentation. Thus, we sought a more robust implementation of the Levenberg-Marquardt algorithm. In this implementation, we search for the Levenberg-Marquardt damping factor λ that gives the minimum of data mismatch with a smooth model. More specifically, at each Levenberg-Marquardt iteration we perform a one-dimensional search for the value of λ that minimizes

$$O_{\mathrm{F}} = O_{\mathrm{D}} + \gamma_i O_{\mathrm{M}}, \quad (3.34)$$

where O_D is the data mismatch objective function, i.e.

$$O_D = \frac{1}{2}\left(g(m) - d_{uc}\right)^T C_D^{-1}\left(g(m) - d_{uc}\right), \quad (3.35)$$

O_M is the regularization or smoothing term, i.e.

$$O_M = \frac{1}{2}\left(m - m_{uc}\right)^T C_M^{-1}\left(m - m_{uc}\right), \quad (3.36)$$

and γ_i is the weighting factor for the regularization or smoothing term. We use the weighting since at early Levenberg-Marquardt iterations O_D is extremely large and the effect of the regularization term or smoothing term is little, i.e. much smaller than O_D. We chose to base the magnitude of the weighting factor for regularization

or model mismatch term on the magnitude of the data mismatch so that the effect of the data mismatch term is reduced. In the ith Levenberg-Marquardt iteration, we define

$$\gamma_i = \max[1, O_{\mathrm{D}}^{i-1}/(8N_{\mathrm{D}})], \tag{3.37}$$

where O_{D}^{i-1} denotes the value of the data mismatch objective function (Eq. 3.35) at the previous Levenberg-Marquardt iteration. Motivation for this choice of weighting is as follows; based on the chi-squared assumption, O_D is expected to be on the order of $N_D/2$. The variance of the chi-squared distribution is N_D. We want the O_D term to be limited to being a few standard deviations from the mean. Thus, we apply this weighting factor which reduces the magnitude of the data mismatch term when it is far outside the normal range. Even though the choice of weighting factor or more specifically $8N_D$ is ad hoc, 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 [32] who proposed using a Newton-like method to estimate the best value of λ that minimizes the objective function 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 need to search for the λ that minimizes the total objective function of Eq. 3.34. For this one-dimensional minimization problem, we tried both the Golden Section search method and the parabolic interpolations of Brent's method as described in Numerical Recipes [46]. Neither of these methods require the calculation of derivatives. In the following two sub-sections we will briefly explain these methods. Detailed information about these methods can be found in Numerical Recipes [46]. We assume that the optimal value of λ is bracketed by $10^{\log(\gamma_i)-3.5}$ and $10^{\log(\gamma_i)+3.5}$. For each trial value of λ, we must solve the system of equations in Eq. 3.33, update the model and evaluate the objective function of Eq. 3.34 again using the diagonal of C_M. Evaluation of the objective

function requires one solution of the forward problem (one simulation run).

3.4.1 Golden Section Search

This method simply describes the successive bracketing of a minimum of a function f to provide an improved guess for the real value of the minimum.

A minimum is bracketed when there is a triplet of points, $a < b < c$, such that $f(b)$ is less than both $f(a)$ and $f(c)$. In this case, the function has a minimum in the interval (a, c). The bracketing can be successively improved by performing the following steps.

- Evaluate f at some new point x in the larger of the two intervals (a, b) or (b, c). Suppose we choose (b, c) to be specific.

- If $f(b) > f(x)$ then x replaces the midpoint b, and b becomes an end point, then, the new bracketing of triplet is $b < x < c$.

- If $f(b) < f(x)$ then b remains the midpoint with x replacing the end point, i.e. the new bracketing triplet of points is $a < b < x$.

Either way the width of the bracketing interval will reduce and the position of the minima will be better defined. This process is repeated until the distance between the two outer points becomes sufficiently small (i.e., approaches some set limit, TOL).

An important question remains to be answered is how to choose the new point x each time. Suppose that b is some fraction w of the distance between a and c then $w = (b-a)/(c-a)$, and, consequently, $1-w = (c-b)/(c-a)$. Also suppose that the new point, x, is some distance z beyond b, then $z = (x-b)/(c-a)$. The next bracketing interval will, therefore, either be of length $w + z$ relative to the current one, or of length $1-w$. The optimal strategy would make these equal, i.e. $z = 1-2w$. This will give the symmetric point to b in the interval a to c, i.e. $|a-b| = |x-c|$.

How does the value of w get chosen? It must come presumably from the previous stage of applying the same strategy, so if z is chosen to be optimal, then so

was w. This scale similarity implies that z should be the same fraction of the distance from b to c (if that is the bigger interval) as was b from a to c, i.e., $w = z/(1-w)$. These two equations can be used in w and z to form the quadratic equation $w^2 - 3w + 1 = 0$ yielding $w \approx 0.38197$, and $1 - w \approx 0.61803$. Therefore the optimal bracketing interval (a, b, c) has its middle point b a fractional distance 0.38197 from one end, and 0.61803 from the other. These are the so-called golden mean fractions. In each step of the golden section search, therefore, a new point which is a fraction 0.38197 into the larger of the two intervals (from the central point of the triplet) is selected.

3.4.2 Parabolic Interpolation and Brent's Method

The Golden Section Search is a method designed to handle the least favorable of function minimizations: no assumptions are made about the shape of the function in the region of the minimum, and the minimum is bracketed with greater and greater precision with each iteration. However, if the function is smooth near the minimum, then a parabola fitted through any three points near to the minimum should give a value very close to the actual minimum in one step.

This method is known as inverse parabolic interpolation, as we want to find a point x at which our function $f(x)$ is a minimum (not just a value of $f(x)$). The formula for the abscissa x that is the minimum of a parabola through three points $f(a)$, $f(b)$ and $f(c)$ is

$$x = b - \frac{1}{2}\frac{(b-a)^2[f(b)-f(c)] - (b-c)[f(b)-f(a)]}{(b-a)[f(b)-f(c)] - (b-c)[f(b)-f(a)]}. \tag{3.38}$$

This equation fails only if the three points are collinear.

A technique likely to be both stable and efficient is one that will use a slow-but-sure method such as the Golden Section Search in unfavorable areas, but can also switch to parabolic interpolation when it comes sufficiently close to a minimum. In this way, we can be sure we are approaching a minimum rather than a maximum (or vice versa) before switching to the more efficient parabolic method.

Brent's method is such a scheme. At each stage, it keeps track of six function points, a, b, u, v, w, and x. The minimum is bracketed between a and b; x is the point with the least function value found so far (or the most recent in the case of a tie); w is the point with the second least function value; v is the previous value of w; u is the point at which the function was evaluated most recently.

A typical ending configuration for Brent's method is that a and b are $2 \times x \times TOL$ apart with x (the best abscissa) at the point of a and b, and therefore fractionally accurate to $\pm 2 \times TOL$.

3.5 Computation of Subspace Vectors

Although the proposed pre-subspace vectors could be calculated using the formula in Eq. 3.28, i.e. $G_k^{\mathrm{T}}[C_{\mathrm{D}}^k]^{-1}\left(g^k(m) - d_{uc}^k\right)$, this would not be efficient in general as it would require computation of the sensitivity coefficient for each datum. One of the advantages of the subspace method is the possibility of avoiding computation of the sensitivities of data to model parameters. It is only necessary to compute the sensitivities of data sub-objective functions to model parameters. This can be done using the adjoint method.

The adjoint equations were derived independently for the single-phase history matching problem by Chen et al. [8] and Chavent et al. [7]. The adjoint method has been applied to multiphase flow problems [31, 60, 34, 57]. The most detailed description of the use of the adjoint system for multiphase flow can be found in Li et al. [33]. Appendix-B contains a detailed description of the equations for single-phase flow problems.

We implemented the adjoint solution for computation of pre-subspace vectors, then compared the results from the adjoint method with pre-subspace vectors obtained by direct or finite perturbation method for a 2-D, single-phase flow problem with five wells. The direct method in which model parameters of each gridblock were

altered by a small amount and changes in sub-objective functions were calculated is known to be inefficient. There are 441 gridblocks (21×21) in the system. The number of pressure data at each well is equal to 29 (corresponding to 29 timesteps in the simulation), thus the total number of data is 145. Other related information about this example will be presented in detail in subsection 4.1.1. We partitioned the data objective function first by well then by time interval. For each well, we partitioned the data objective function into 9 parts (3 or 4 data grouped together). Thus, the total number of data sub-objective functions are equal to 45. The gradients of these 45 data sub-objective functions give 45 pre-subspace vectors. The pre-subspace vectors numbered from 1 to 9 involve the data from Well 1, the ones numbered from 10 to 18 involve the data from Well 2 and so on. As an example, Fig. 3.1 shows the 2-D plot of the gradient of the sixth data sub-objective function (corresponding to build-up data) from Well 1 (located at the center) with respect to gridblock log-permeabilities obtained from the adjoint method and the direct method. The results are in excellent agreement. Fig. 3.2 shows a similar comparison for the gradient of the same data sub-objective function with respect to gridblock porosities. These two gradients, i.e. gradient of sixth data sub-objective function with respect to gridblock log-permeabilities and the gradient with respect to gridblock porosities, form a 882-dimensional sixth pre-subspace vector.

We have also compared the results from the adjoint method with the direct method for 3-D reservoir models. The example problem is very similar to the 2-D problem considered. The only differences are the size, the dimension of the model ($11 \times 11 \times 3$) and the locations of five wells. Fig. 3.3 shows similar excellent agreement between the two methods for the gradient of the sixth sub-objective function (corresponding to build-up data) from Well 1 with respect to gridblock log-permeabilities.

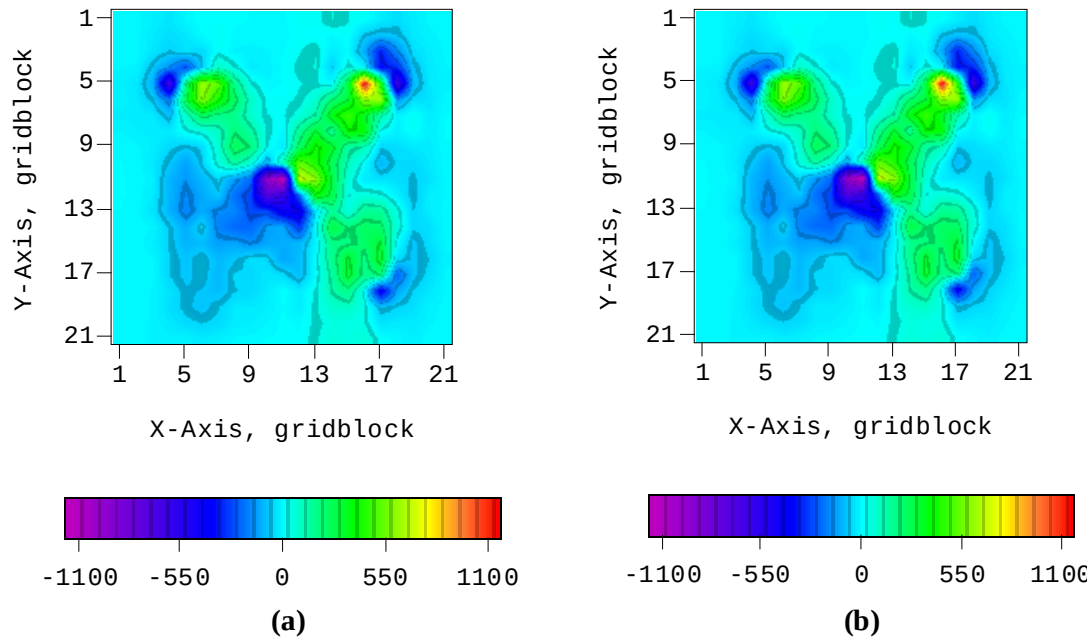


Figure 3.1: Log-permeability part of 6th pre-subspace vector of Well 1 from, (a) the direct method, and (b) the adjoint method.

3.6 Sensitivity to Coefficients of the Subspace Vectors

Once the subspace vectors have been chosen, we need to calculate the product $G_l C_M B_l$ or $G_l A_l$ (see Eqs. 3.7 and 3.10) efficiently. This product, which is simply the sensitivity of the data to the coefficients of the subspace vectors, can be calculated efficiently using the direct or "gradient simulator" method instead of multiplying A_l by G_l directly. Direct multiplication is not preferred because it requires computation of the sensitivity coefficient of each datum to form the matrix G_l.

The direct or gradient simulator method was introduced to the petroleum engineering literature by Anterion et al. [1], but was known earlier in the groundwater hydrology literature as the sensitivity equation method [61]. The matrix problem solved to obtain the product $G_l A_l$ involves the same coefficient matrix as the one

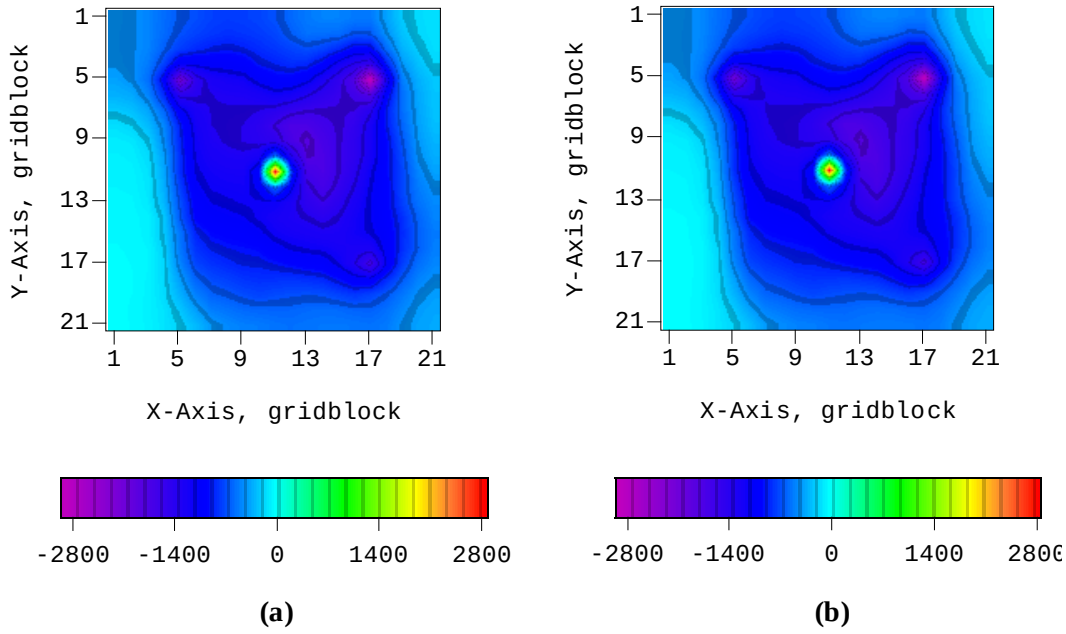


Figure 3.2: Porosity part of 6th pre-subspace vector of Well 1 from, (a) the direct method, and (b) the adjoint method.

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 subspace vector. Killough et al. [27] developed a fast iterative solver for this type of problem; the effort required for each sensitivity is of the order of 10% of a forward simulation. Detailed equations are given in Appendix-C to calculate the sensitivity to coefficients of the subspace vectors.

Here we present an example for the computation of the product $G_l A_l$ (sensitivity of the data to the coefficients of the subspace vectors) which is an $N_D \times N_B$

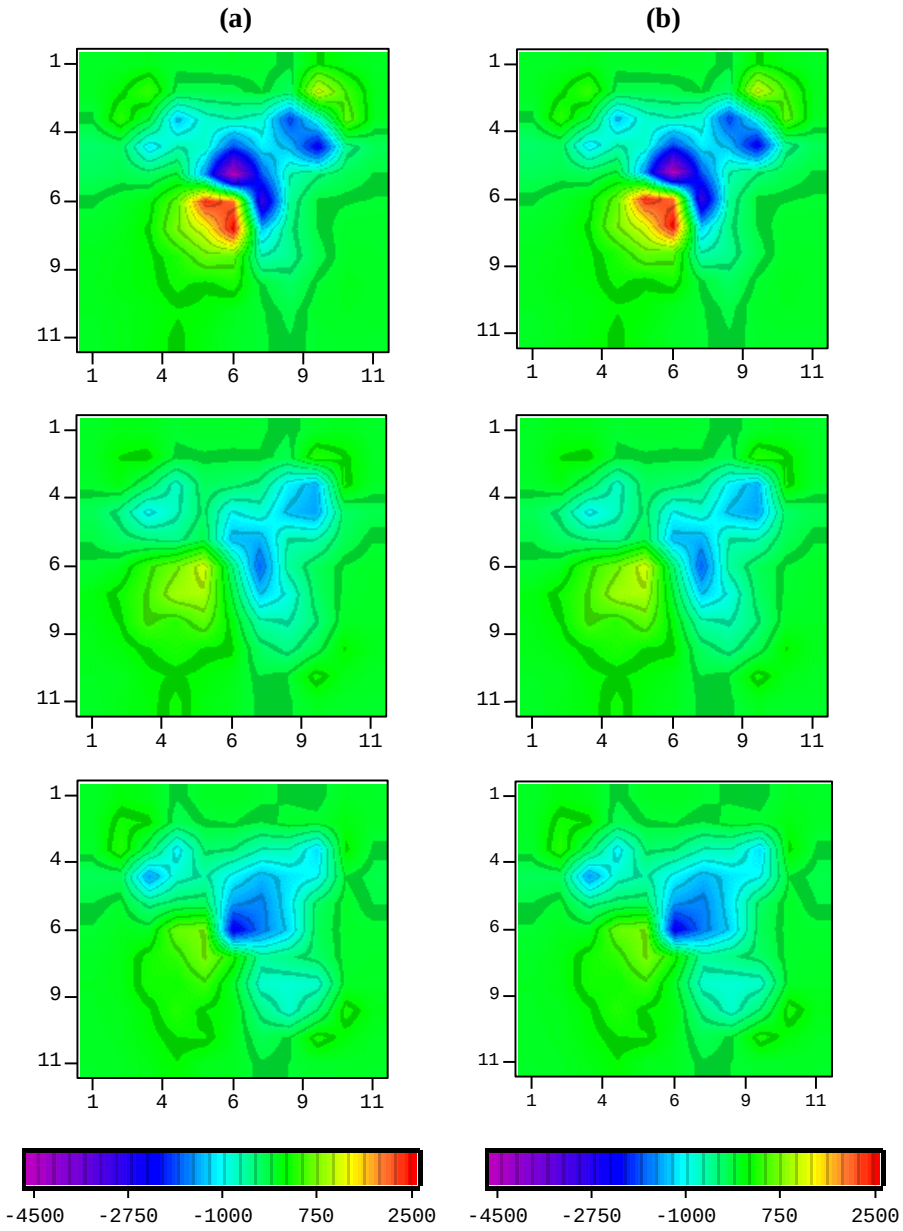


Figure 3.3: Pre-subspace vector from, (a) the direct method, and (b) the adjoint method by layer from the top of the reservoir (top) to the bottom.

matrix, i.e.

$$GA = \left(\nabla_\alpha g^T\right)^T = \begin{bmatrix} \frac{\partial g_1}{\partial \alpha_1} & \frac{\partial g_1}{\partial \alpha_2} & \cdots & \frac{\partial g_1}{\partial \alpha_{N_B}} \\ \frac{\partial g_2}{\partial \alpha_1} & \frac{\partial g_2}{\partial \alpha_2} & \cdots & \frac{\partial g_2}{\partial \alpha_{N_B}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial g_{N_D}}{\partial \alpha_1} & \frac{\partial g_{N_D}}{\partial \alpha_2} & \cdots & \frac{\partial g_{N_D}}{\partial \alpha_{N_B}} \end{bmatrix}, \tag{3.39}$$

where we have deleted the iteration index from our notation to simplify the equations. Appendix-C contains detailed equations to obtain Eq. 3.39. The example problem considered here is the same as the 2-D one used for the comparison of computation of pre-subpsace vectors in the previous section. In our comparisons, we actually calculated the product GB where B is the matrix formed by the 45 pre-subspace vectors explained in the previous section. We compare the results obtained from the gradient simulator method with the product GB formed by multiplying the G matrix by the B matrix. In the later case, the sensivity coefficient matrix G is obtained using Carter's method. We compared the two GB matrices obtained from the gradient simulator method and the direct multiplication by plotting each row of these matrices and observed excellent agreement. As an example, Fig. 3.4 shows the comparison of the 70th row of GB calculated from the gradient simulator method (illustrated by diamonds) and the GB product formed by multiplying the G matrix by the B matrix. The results are in excellent agreement. Recall that we have 29 pressure data at each well and the total number of data is equal to 145, so the GB matrix has a dimension of 145×45. Thus, 70th row of GB corresponds to the sensitivities of the 12th pressure at Well 3 with respect to the coefficients of the pre-subspace vectors. Since the pre-subspace vectors numbered from 19 to 27 involve the data at Well 3, the considered pressure data is more sensitive to these coefficients of the pre-subspace vectors.

Fig. 3.5 shows a similar comparison for another row which is the 115th row of the GB product. Again, the two sets of results are in excellent agreement. This row represents the sensitivity of the 28th pressure at Well 4 (the observation well) to the coefficients of the subspace vectors.

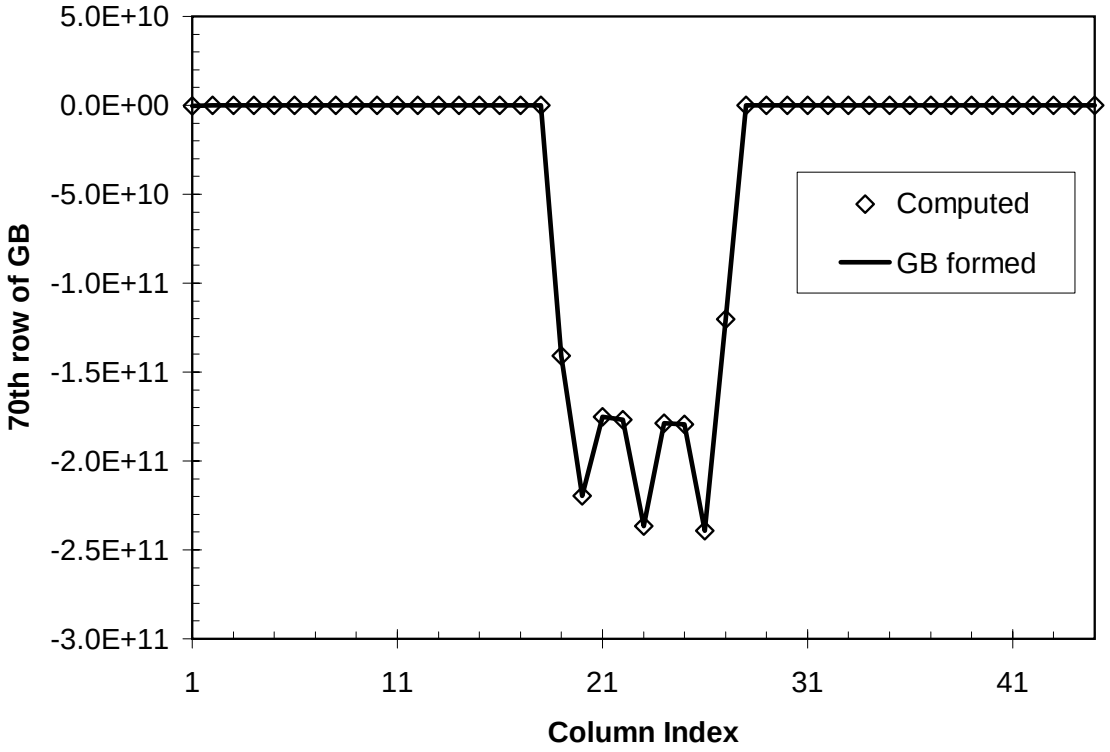


Figure 3.4: 70th row of GB.

3.7 Summary of the Recommended Procedure

The recommended procedure for use of subspace vectors in history matching is as follows.

1. Generate pre-basis vectors by the following procedure.

   - Partition the total objective function into a term that arises from the model roughness and distance from the prior model and a term that arises from data mismatch.

   - One set of pre-basis vectors are formed by multiplying the gradient of the partial objective function from the model part by C_M and then partitioning the resulting vector into two parts (one vector for porosity and one vector for

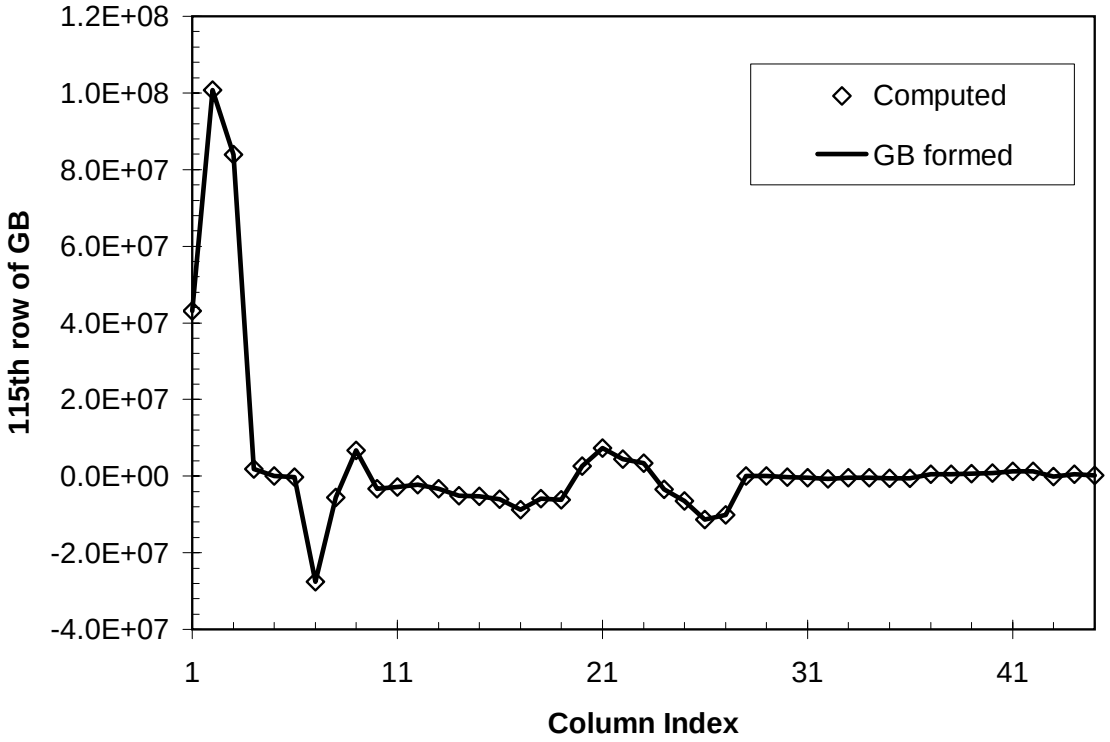


Figure 3.5: 115th row of GB.

permeability).

- Partition the data mismatch partial objective function further by well and time interval to generate data sub-objective functions. Gradients of these data sub-objective functions provide another set of pre-basis vectors. Use the adjoint method to compute the gradients of sub-objective functions with respect to all model parameters (one system solve per sub-objective function). Use a small number of partitions at early iterations, and increase the number as the minimum is approached. (Based on the limited cases we tried, we recommend an ad hoc procedure in which we start with one subspace vector per well at the first iteration and then increase the number by adding one subspace vector per well at each subsequent iteration until the convergence is obtained.)

2. Use SVD (or QR decomposition) to ensure that the pre-basis vectors are independent and have similar magnitudes, then multiply by C_M to create the basis vectors.

3. Compute matrix products of the form GA using the gradient simulator method (one system solve per column of A), see Appendix-C.

4. Solve the system

$$\left[(1+\lambda)B_l^T C_M B_l + (G_l C_M B_l)^T C_D^{-1}(G_l C_M B_l)\right]\alpha^l = -B_l^T(m^l - m_{uc}) - \left(G_l C_M B_l\right)^T C_D^{-1}\left(g(m^l) - d_{uc}\right), \quad (3.40)$$

   for α^l using the optimal λ determined by the procedure explained in section 3.4.

5. Compute the correction to the model, $\delta m^l = A_l \alpha^l$.

6. Update the model estimate, $m^{l+1} = \delta m^l + m^l$.

7. Iterate until convergence is obtained.

CHAPTER IV

COMPUTATIONAL RESULTS

In this chapter, we illustrate the use of the subspace method on synthetic two- and three-dimensional example problems. In the 2-D case, we generate the maximum a posteriori estimate of the model parameters. We use the randomized maximum likelihood method to generate conditional realizations of the 3-D model. We compare the results of the subspace method with the conventional Gauss-Newton history-matching approach that uses the full parameterization of the model. Most of the investigations are carried out using the 2-D example problem because it is small enough that we can run many cases to study the effects of several parameters and issues. However, the 3-D example problem has a fairly large number of model parameters and involves over 400 conditioning pressure data.

4.1 Description of the Example Problems

4.1.1 Two-Dimensional Example Problem

The areal dimensions of the 2-D synthetic reservoir are 2,100 feet by 2,100 feet; the reservoir thickness is uniform and equal to 25 feet. A 21 by 21 grid with $\Delta x = \Delta y = 100$ feet is used in the numerical solution of the flow equations. Fig. 4.1 shows the schematic of the reservoir model grid and the locations of five wells. Well 1 produces at a constant rate 700 rb/day for the first 1 day, followed by a 1 day shut-in period, after which it is returned to production at 700 rb/day for 1 day; Wells 2 and 3 produce at fixed rates of 500 rb/day and 800 rb/day, respectively, for all times; Well 4 is an observation well (no production); Well 5 produces at a constant rate of 600

rb/day for 2 days after which it is shut-in for 1 day.

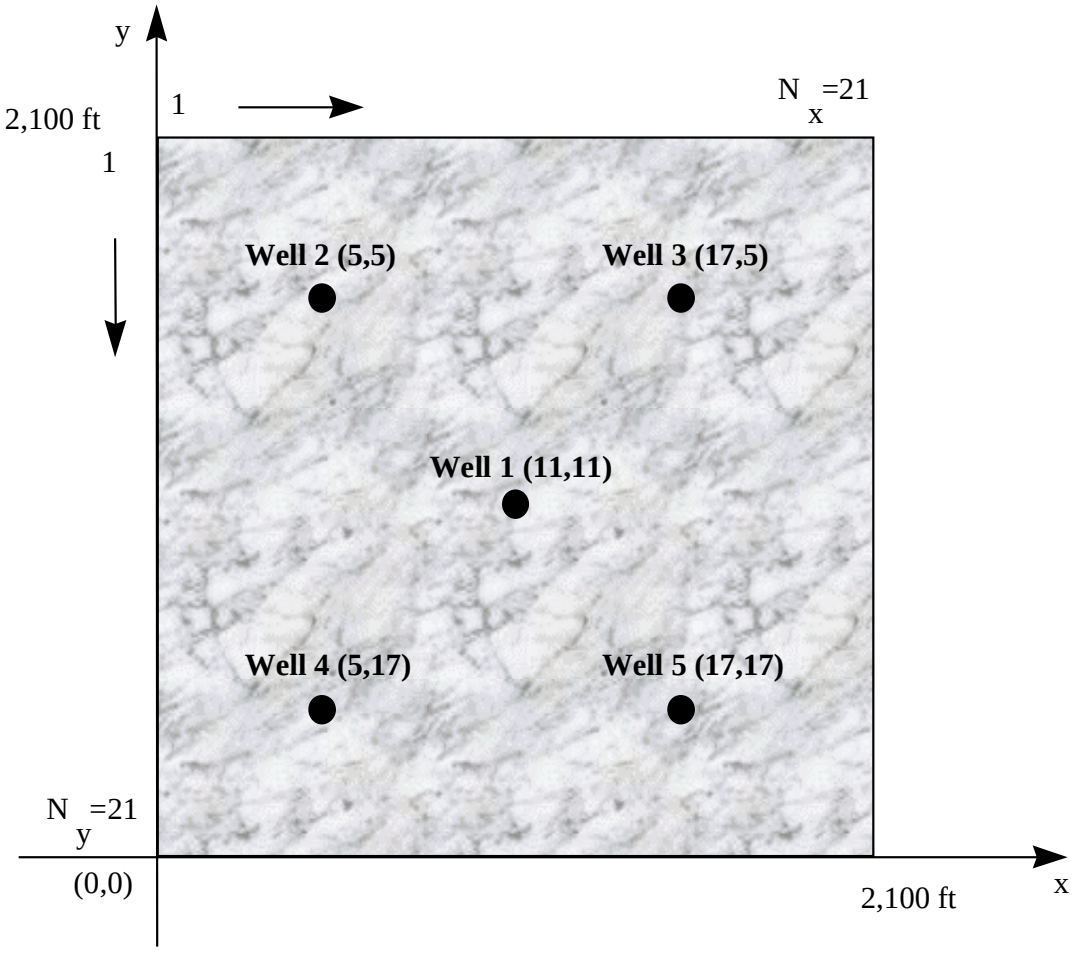


Figure 4.1: Schematic of the 2-D reservoir model grid and well locations.

The "true" distributions of porosity and log-permeability, which were used to generate synthetic pressure data, are correlated Gaussian random fields with anisotropic spherical variograms [24]. The correlation coefficient between porosity and log-permeability is 0.5. The range of the variogram is 400 feet in the x-direction and 900 feet in the y-direction. The prior mean for log-permeability is 4.0 and the prior variance is 1.0. The prior mean for porosity is 0.25 and the prior variance

is 0.0025. Fig. 4.2 shows the true log-permeability and porosity fields. The standard deviation of measurement errors for all pressure data was assumed to be 0.1 psi. Although it is possible to estimate skin and permeability simultaneously, for

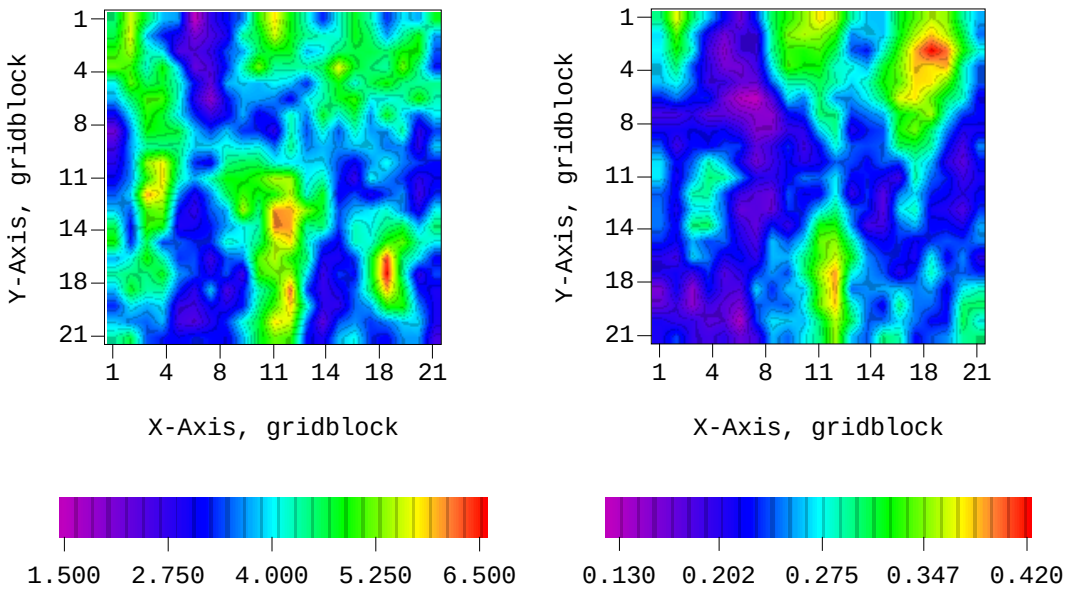


Figure 4.2: The true log-permeability field (left) and porosity field (right) for the 2-D problem.

this illustration, the skin factors were assigned prior values of 3.0, 5.0, 5.0, 0.0 and 3.0 respectively for each well and given a very small uncertainty ($\sigma_s^2 = 10^{-7}$). The fluid is assumed to remain as a single phase at all pressures; system compressibility, $c_t = 10^{-5}$ 1/psi, fluid viscosity, $\mu = 0.8$ cp, all wellbore radii, $r_w = 0.3$ feet; and the initial pressure, $p_i = 3,230$ psi. Synthetic pressure data are then computed from the reservoir flow simulator using the true reservoir description as input data. The data for each well are shown by solid symbols in Fig. 4.3. There are 29 pressure measurements at each well so N_{D} is 145.

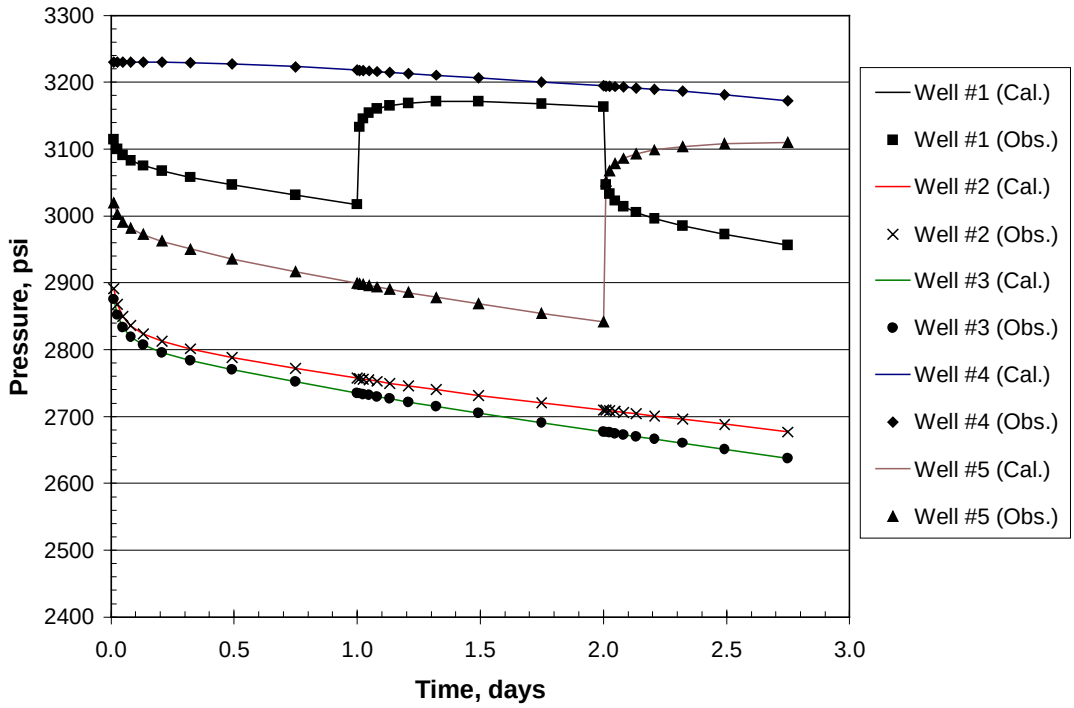


Figure 4.3: Observed versus calculated pressure data for the 2-D problem.

4.1.2 Three-Dimensional Example Problem

Potential strengths and weaknesses in the methodology should be more apparent in a larger model. For this reason, a three-dimensional model with $25 \times 25 \times 10$ gridblocks was also used for evaluation. 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. The prior mean for porosity is 0.20 and the prior variance is 0.002. The prior mean for log-permeability is 4.0 and the prior variance is 0.5. 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).

Fig. 4.4 shows the areal grid and the locations of five wells for the 3-D model. Well 1 produces at a constant rate 3,000 rb/day for the first 1 day, followed by a 1 day buildup, then production is 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. 4.5) are generated using the true log-permeability and porosity fields shown in Fig. 4.6. 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 skin factors at the wells were assumed to be known. Other reservoir and fluid properties are the same as the two-dimensional example.

4.2 Results from Simple Partitioning with Constant Basis Dimension

4.2.1 Two-Dimensional Maximum a Posteriori Estimates

Using the conventional Gauss-Newton history-matching approach without subspace vectors, the maximum a posteriori (MAP) estimates of log-permeability and porosity (conditioned to pressure data from the true 2-D model) were generated. Recall that we have shown (Chaper III, Eq. 3.32) that this conventional approach is equivalent to the subspace method in which each set of data includes only one measurement. The MAP estimates of log-permeability and porosity from the conventional approach are shown in the left column of Fig. 4.7. As expected, these MAP estimates are smooth and they capture the trend of the true fields. Generation of the MAP estimates using the conventional approach (Eq. 2.15) requires computation of 145

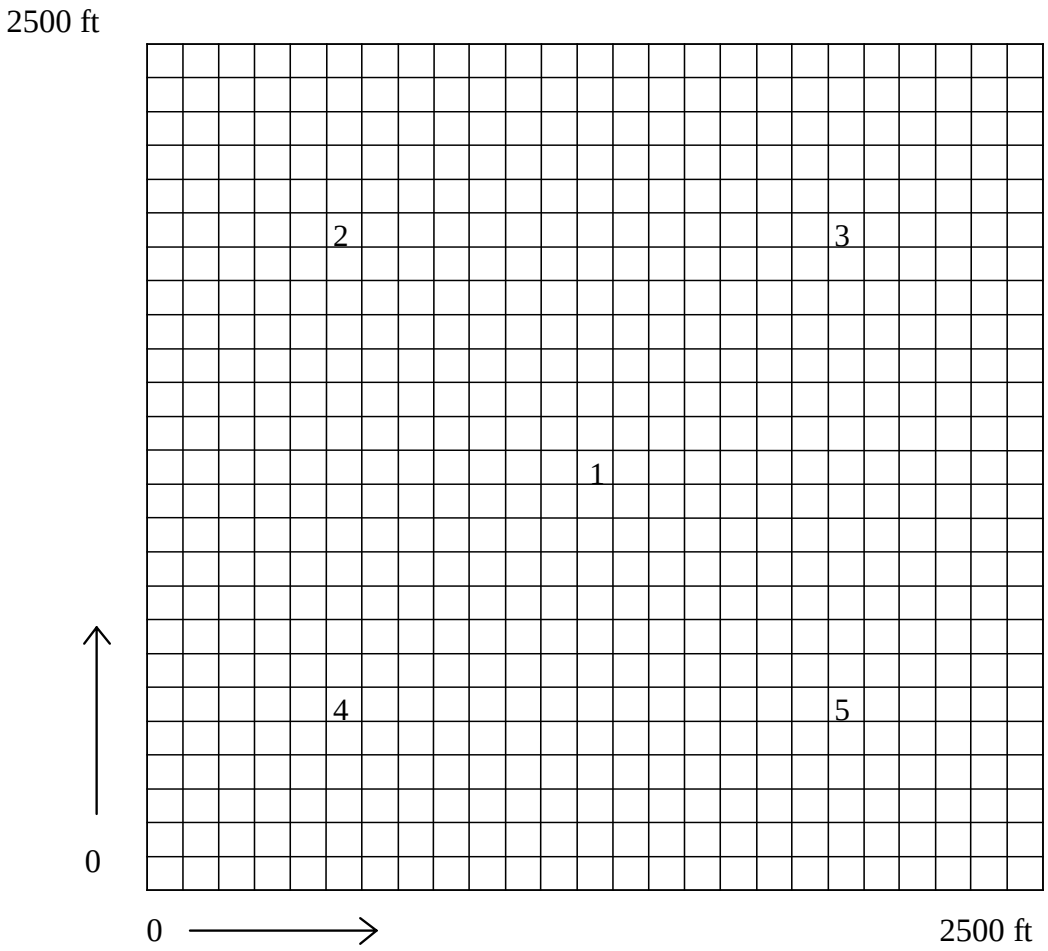


Figure 4.4: Areal grid, well locations and well numbers for the 3-D problem.

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 work equivalent to approximately 6 flow simulations using Carter's method [6]. For a multiphase flow problem, the computational effort required to generate one sensitivity vector would be equivalent to approximately one flow simulation using the adjoint method.

A straightforward approach to selecting subspace vectors is to partition the

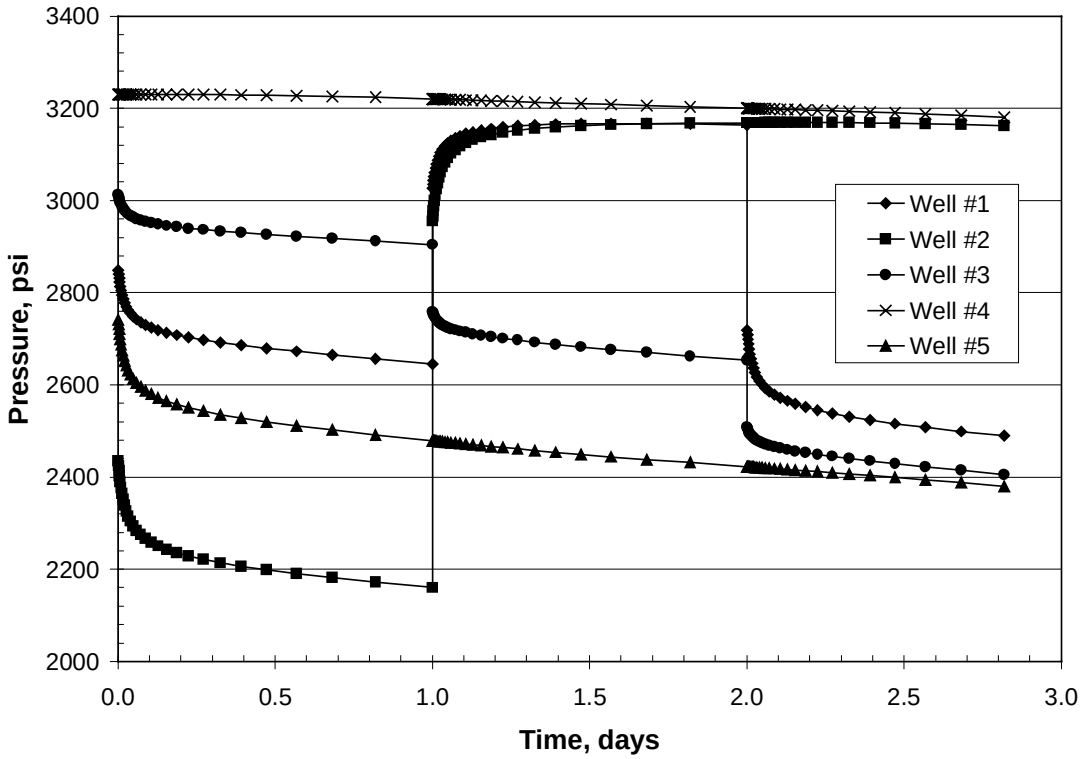


Figure 4.5: Observed pressure data at the wells for the 3-D problem.

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. This partitioning produced 45 subspace vectors from the data mismatch part of the objective function and there are 2 subspace vectors from the model mismatch part of the objective function, thus the total number subspace vectors are equal to 47. The resulting estimates for log-permeability and porosity were quite similar to the MAP estimates obtained using the conventional method (Fig. 4.7). The results from the subspace approach were obtained in far less time, however (see section 4.6 for detailed computational efficiency gains). In this 2-D example, although the total number of model parame-

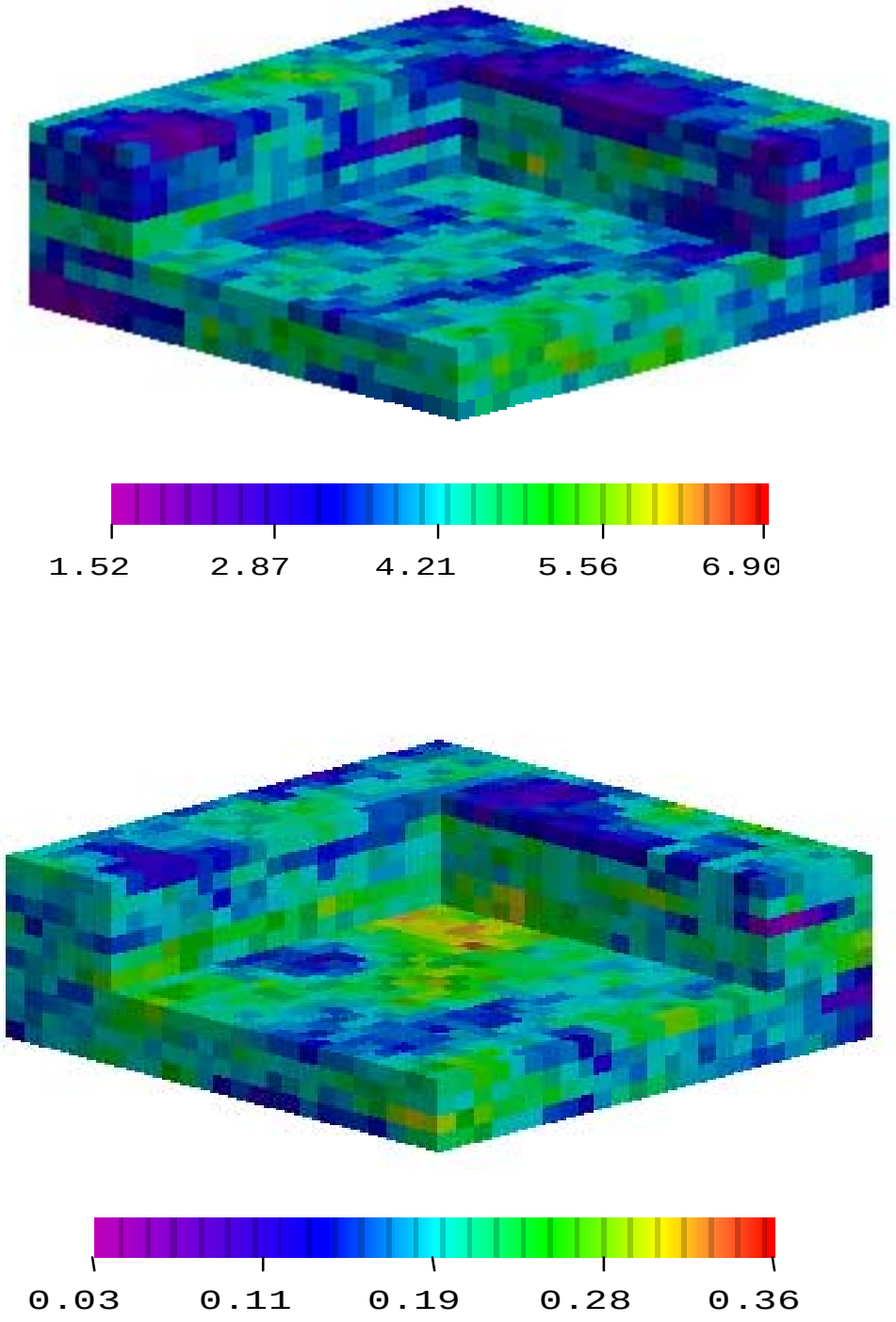
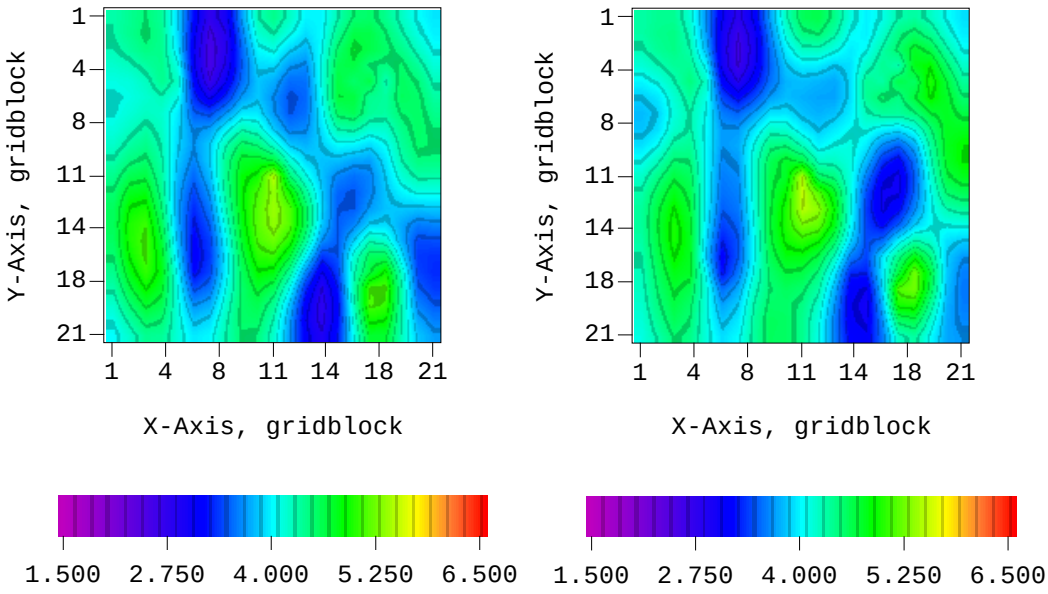
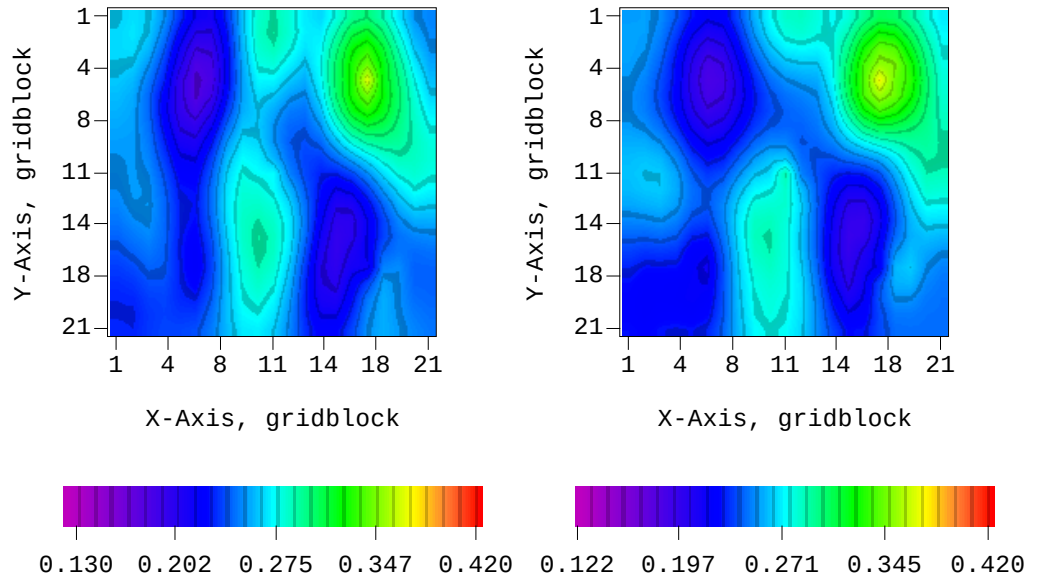


Figure 4.6: True log-permeability field (above) and true porosity field (below).



(a) Log-permeability field using 145 basis vectors

(b) Log-permeability field using 47 basis vectors



(c) Porosity field using 145 basis vectors

(d) Porosity field using 47 basis vectors

Figure 4.7: Comparison of MAP estimates for 2-D example.

ters 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.

The decision to use 47 subspace vectors in this example was somewhat ad hoc but an analysis of the spectrum of the dimensionless Hessian, $L^T G^T C_D^{-1} GL$, 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. 4.8) based on the cut-off value of 0.1 which was discussed in section 3.2. So, the use of 47 subspace vectors is probably close to optimal number. Although this matrix changes as the minimum is approached, we observed that the number of eigenvalues that are greater than 0.1 was relatively constant in these examples.

4.2.2 Three-Dimensional Reservoir Realizations

In the 3-D example, we generate conditional realizations using the randomized maximum likelihood method. Fig. 4.9 shows the unconditional realizations of log-permeability and porosity generated using sequential Gaussian cosimulation [15]. Fig. 4.10 shows the conditional (conditioned to pressure data) log-permeability and porosity realizations obtained from the conventional Gauss-Newton history-matching approach. The conditional realizations from the subspace method using 47 vectors (i.e. 9 data sub-objective functions per well plus 2 model mismatch vectors) are shown in Fig. 4.11. Comparison of Figs. 4.10 and 4.11 shows that the reservoir model from the conventional method is nearly identical to the model created using the subspace method. The quality of the comparison is easier to see in Fig. 4.12, which shows the vertically averaged log-permeability in the true model, the unconditional model that was used as the starting point, and the conditional realizations from the conventional and the subspace methods. Vertically averaged conditional realization of log-permeability from subspace method is very close to the one obtained from the

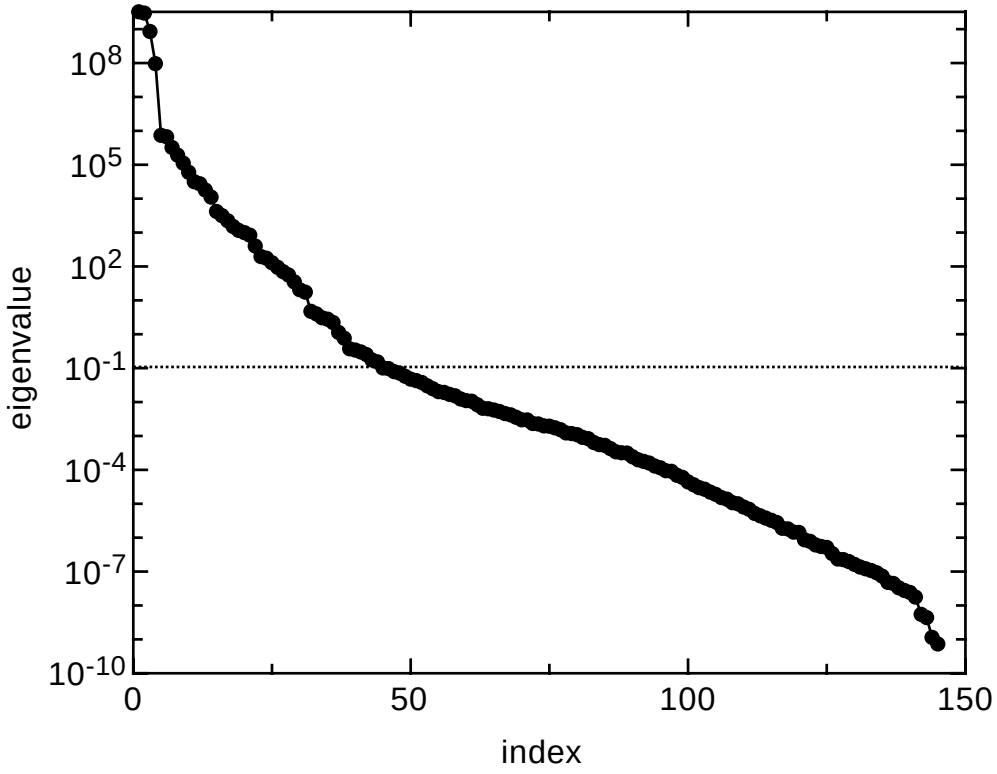


Figure 4.8: The first 145 eigenvalues of the matrix $L^T G^T C_{\mathrm{D}}^{-1} G L$ showing that approximately 40–50 basis vectors are sufficient to accurately construct δm in a Gauss-Newton iteration.

conventional method. Both the conventional and subspace cases terminated with a fairly low value of the objective function after 13 iterations. We should note that the subspace method requires much less work per iteration. We will discuss the computational efficiency issues in section 4.6.

In this fairly large 3-D example, we again show that it is possible to parameterize changes in the model in a subspace whose dimension was only 47, although the total number of parameters is 12,500 and the number of pressure data is 430. Fig. 4.13 shows the eigenvalues of the dimensionless Hessian, $L^T G^T C_{\mathrm{D}}^{-1} G L$, at the first Gauss-Newton iteration. The number of subspace vectors required to span the

data space is on the order of 50 based on the cut-off value of 0.1 so the ad hoc use of 47 vectors is also close to optimal number in this 3-D example.

4.3 The Effect of Small and Large Basis Dimensions

We observed that the number of iterations required to obtain an acceptable level of data mismatch in a Gauss-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. Fig. 4.14 illustrates the effect of the number of subspace vectors used in the representation of δm in the Gauss-Newton method for the 2-D example problem. With 47 subspace vectors (i.e. 9 data sub-objective functions per well plus two model mismatch vectors) convergence was achieved in 4 iterations, which was the same as in the conventional Bayesian inverse approach (Eq. 2.15). When 7 subspace vectors were used (i.e. 1 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 Gauss-Newton iterations the rate of reduction slowed substantially. Even after 50 iterations, the total objective function was 9206 which is still very far from the value obtained in 4 iterations with 47 subspace vectors.

Results with 12 (i.e. 2 data sub-objective functions per well plus 2 model mismatch vectors) and 22 subspace vectors (i.e. 4 data sub-objective functions per well plus 2 model mismatch vectors) were intermediate between 7 and 47 subspace vector cases. In these cases, the initial rate of reduction in the data mismatch was large, but after three iterations the rate of reduction again slowed substantially (see curves marked with black triangles and squares in Fig. 4.14). 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.

The case with 87 subspace vectors (i.e. 17 data sub-objective functions per well plus 2 model mismatch vectors) uses more subspace vectors than optimum num-

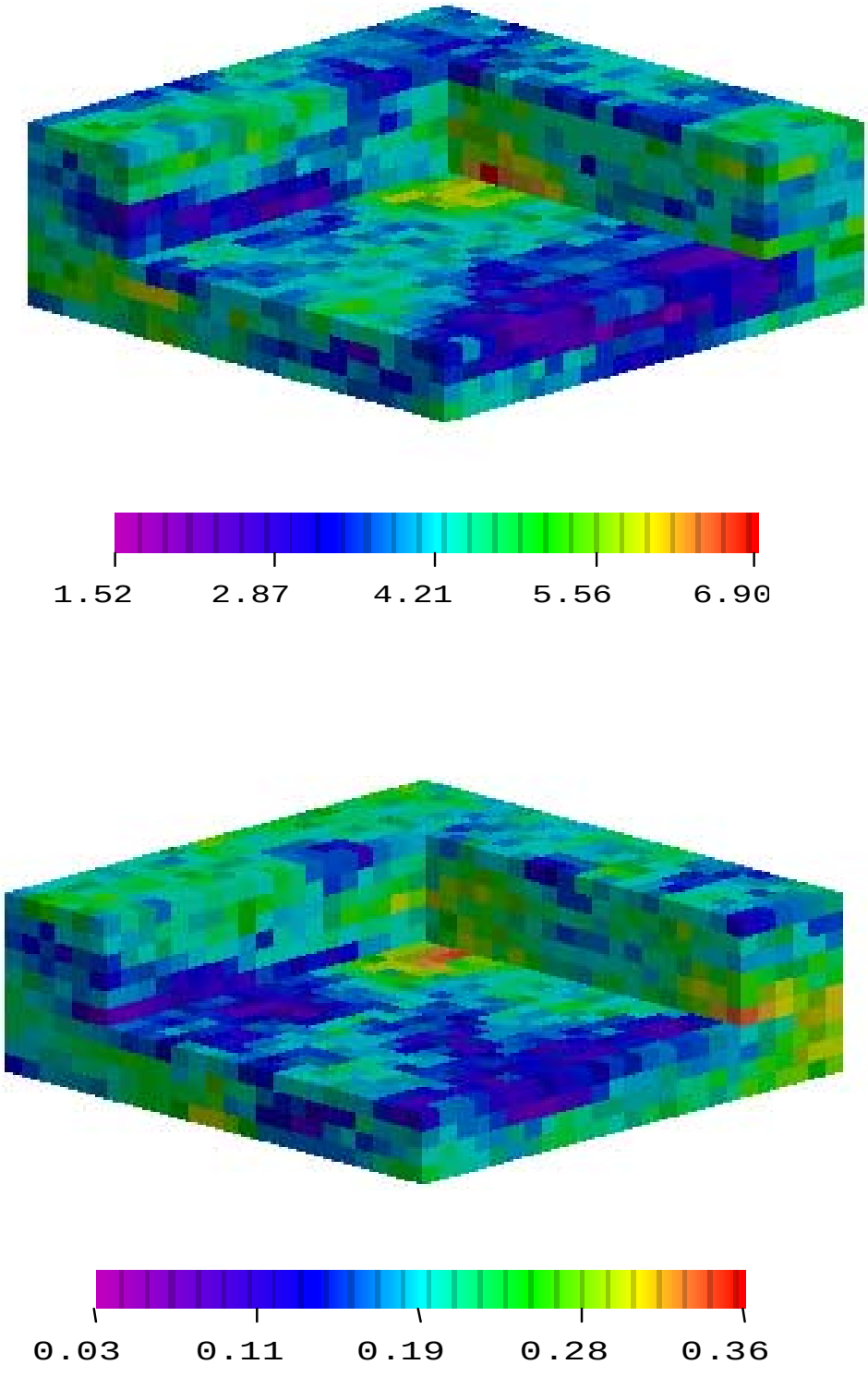


Figure 4.9: Unconditional realizations of log-permeability (above) and porosity (below).

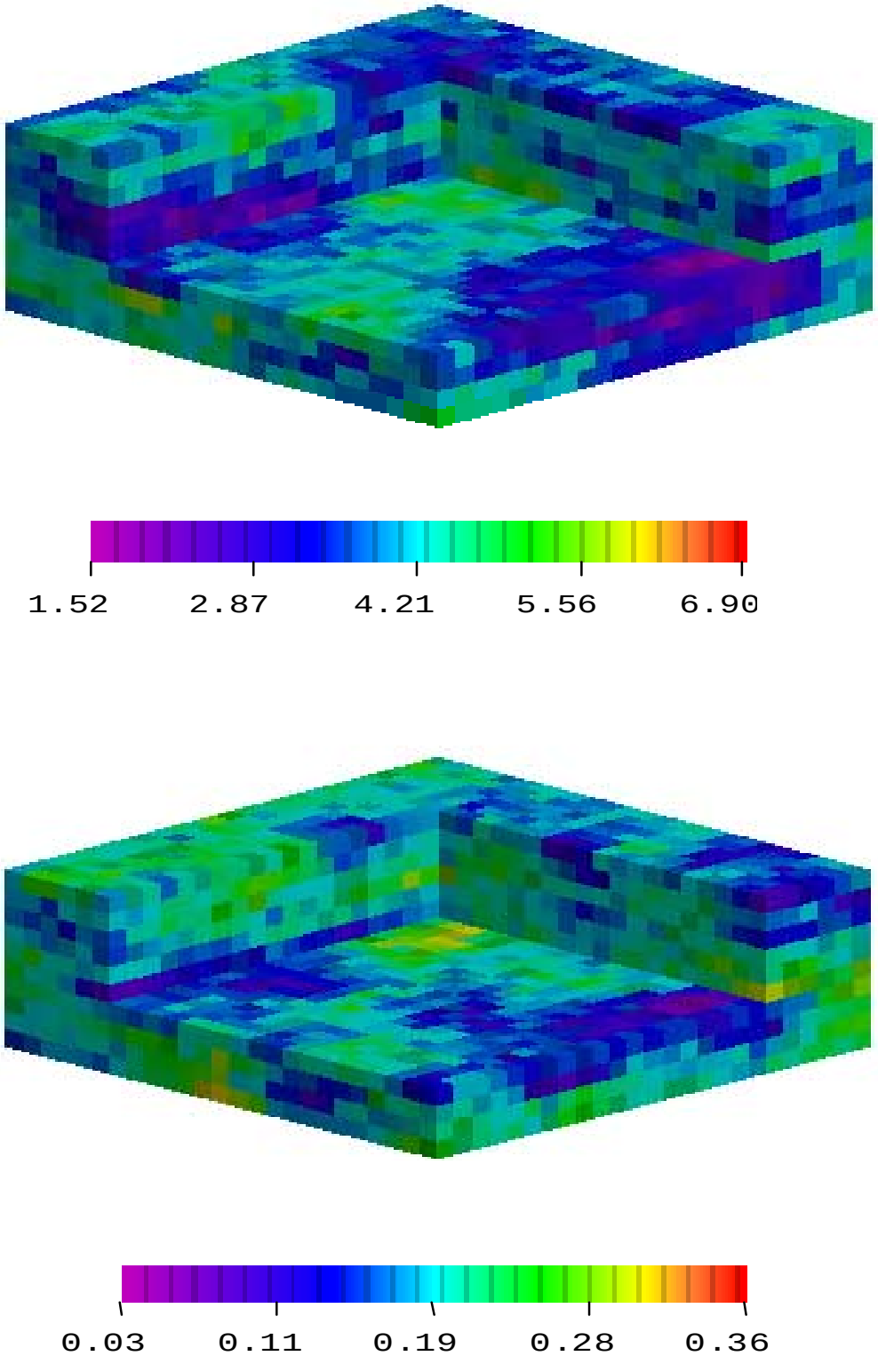


Figure 4.10: Conditional realizations of log-permeability (above) and porosity (below) from the conventional method.

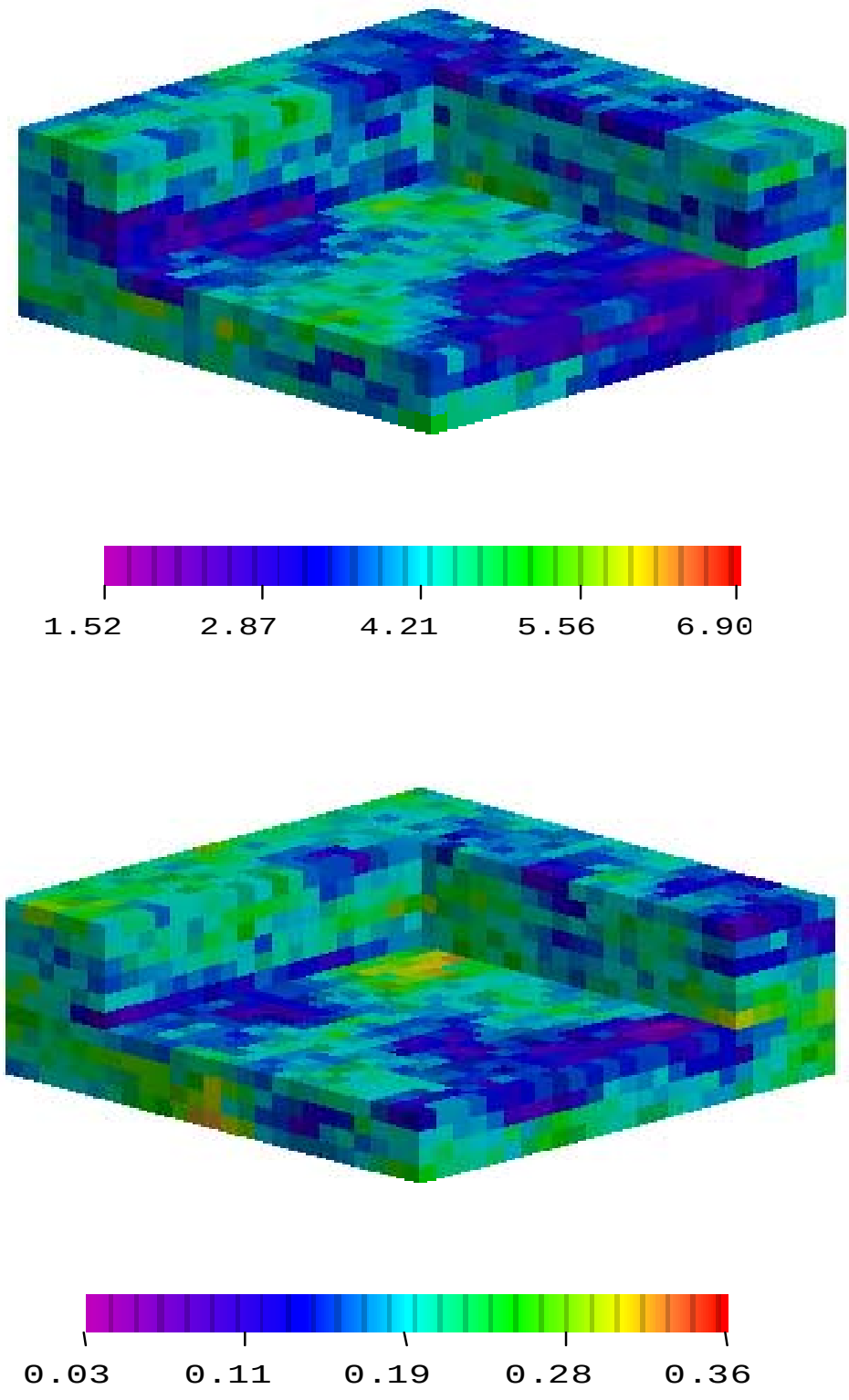
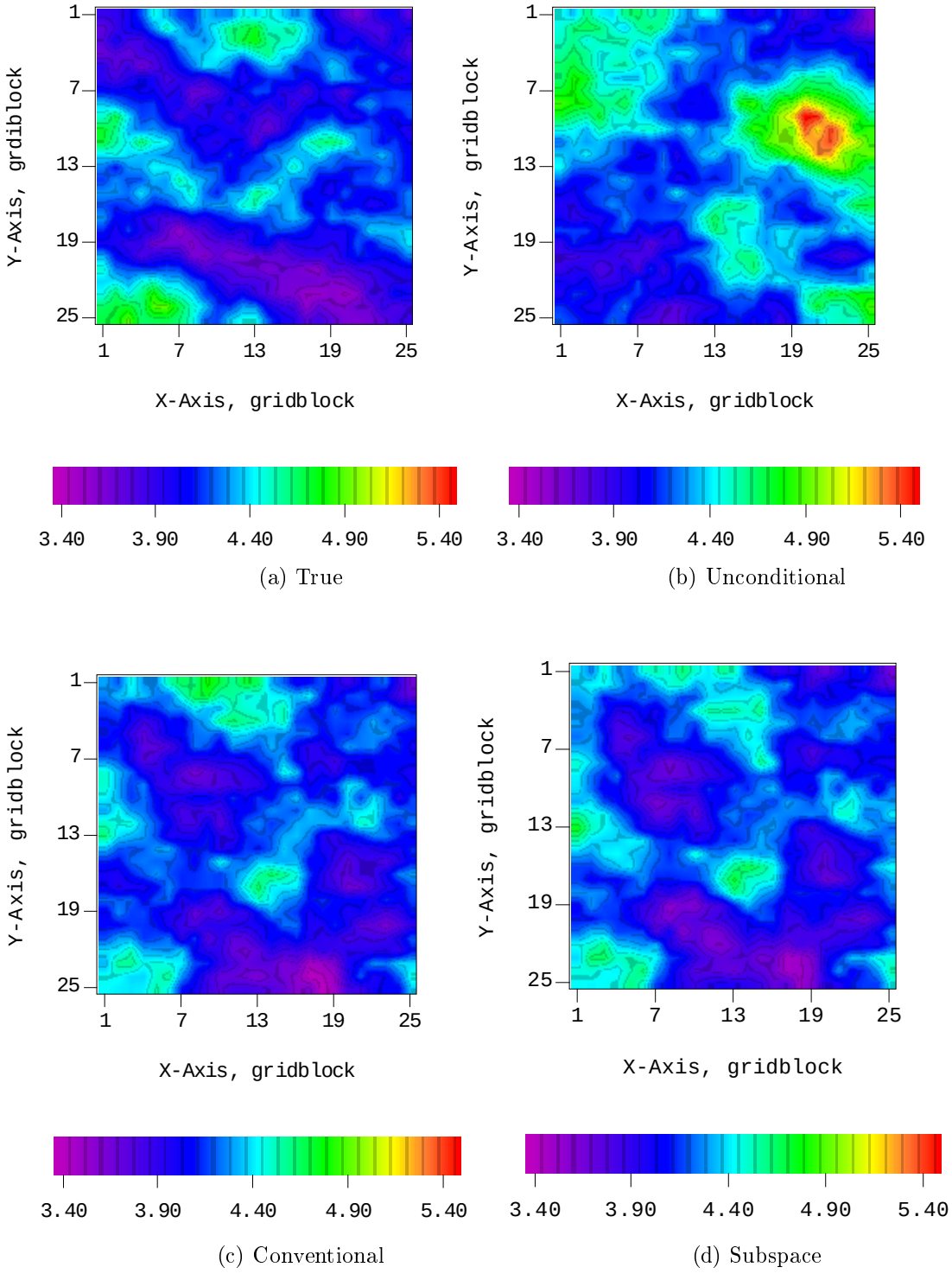


Figure 4.11: Conditional realizations of log-permeability (above) and porosity (below) from the subspace method with 47 subspace vectors.



(a) True (b) Unconditional

(c) Conventional (d) Subspace

Figure 4.12: Comparison of average permeabilities.

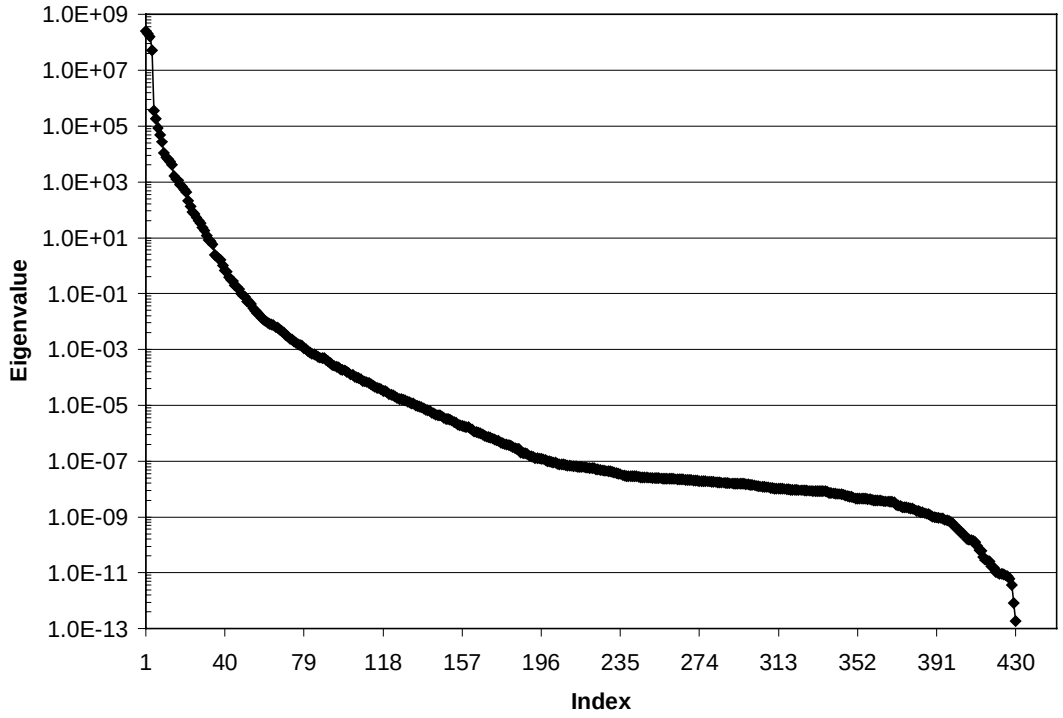


Figure 4.13: The first 430 eigenvalues of the matrix $L^T G^T C_{\mathrm{D}}^{-1} G L$ for 3-D example problem.

ber of 47. In this case, convergence was obtained in 4 iterations as in the subspace case with 47 subspace vectors and conventional method. Final MAP estimates obtained with 87 subspace vectors are identical to those obtained with 47 subspace vectors. Note that this case requires more computational time than the 47 subspace vector case.

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 mismatch 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 $L^T G^T C_{\mathrm{D}}^{-1} G L$ that are greater than 0.1. Because it is unlikely that the eigenvalues of the dimensionless Hessian will be computed for

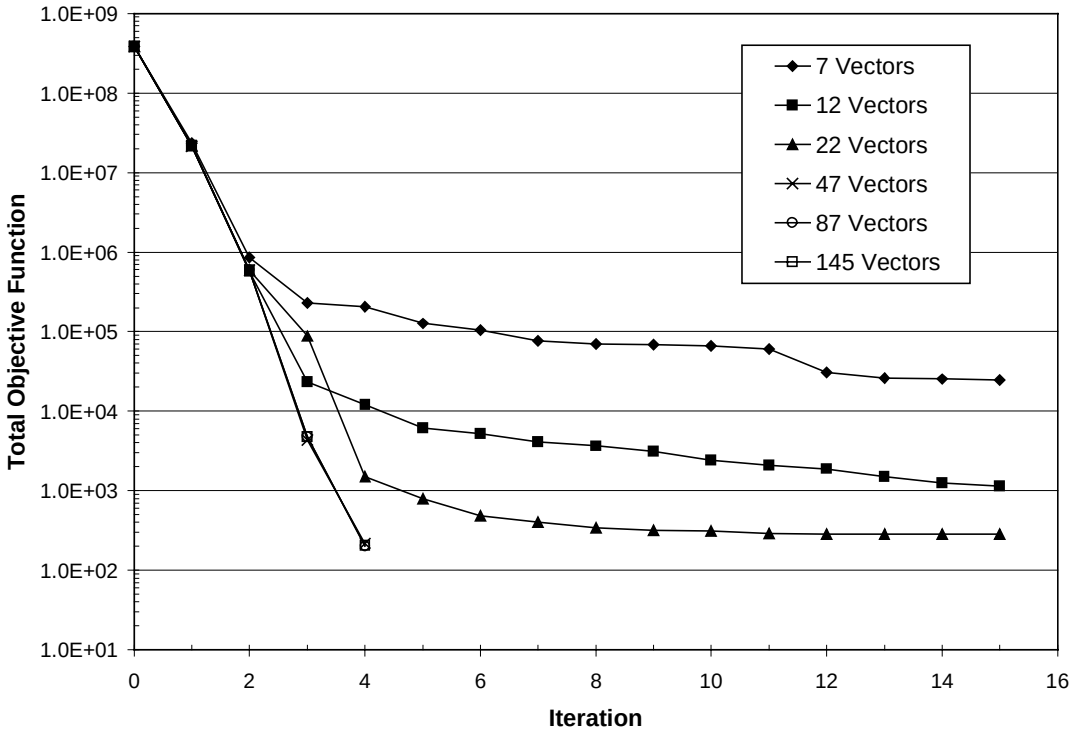


Figure 4.14: The number of Gauss-Newton iterations required to reduce the objective function to the desired level depends on the number of subspace vectors used in the expansion of δm.

large problems, however, we explored the consequences of using a smaller than optimal number of subspace vectors which resulted in slower convergence. 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.

4.4 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. Thus, we never combined data from two flow periods (such as buildup and drawdown). If this had been true it

would have had severe consequences on the utility of the method in practice where the measurements generally show frequent rate changes. 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. Fig. 4.15 contains a comparison of the results from two different subspace vector selection schemes, both of which have 22 subspace vectors (i.e. 4 data sub-objective functions per well plus two model mismatch vectors). In the first case, (*across* flow periods), we simply partitioned the data from each well into 4 sets of approximately equal size, i.e. at each well data numbered from 1 to 7 form the first set, from 8 to 14 form the second set, from 15 to 21 form the third set and from 22 to 29 form the fourth set. By this partitioning, data from different flow periods (such as buildup and drawdown) are combined. For example, at Well 1 data numbered from 1 to 10 correspond to drawdown period and data numbered from 11 to 20 correspond to buildup period so the second set from this well combines data from both drawdown and buildup periods. In the second case, (*within* flow periods), we did not allow data from different flow periods to be combined. For example, for Well 1, the 4 sets are generated by grouping the data numbered 1-5, 6-10, 11-20, 21-29 so the first two sets group the data only from the first drawdown period, the third set involves the data only from the buildup period and the last set groups the data only from the second drawdown period. The results were nearly identical from these two partitioning cases (Fig. 4.15). Fig. 4.16 shows similar comparison of the objective function using 47 subspace vectors. The results were again almost identical. Similarly, following the ideas of Oldenburg and Li [37], we also initially thought that a systematic approach would be to select data partitioning based on the magnitude of the data mismatch, i.e. partitioning the data into more groups in the wells that have larger data mismatches. However, we observed that whether an equal number of subspace vectors were chosen for each well, or whether the data were partitioned based on the magnitude of the data mismatch, the resulting convergence rate and model estimates were approximately the same. These

results are encouraging as it suggests that the subspace spanned by the gradients of sub-objective functions for any reasonable partitioning of the data are similar.

We also initially attempted to use a technique in which we proposed a set of trial subspace vectors and eliminated the singular vectors corresponding to the smallest singular values from a SVD. More specifically, if the singular values are ordered from largest to smallest as

$$\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_{N_B}, \tag{4.1}$$

with the associated orthonormal vectors denoted by $b_1^l, b_2^l, \ldots, b_{N_B}^l$ and ordered the same way as the singular values, then upon return from the singular value decomposition (SVD) routine, we retained only the b_j^l's, $j = 1, 2, \ldots, r$ where r is the smallest interger such that

$$\sum_{j=1}^{r} \sigma_j \geq \beta \sum_{j=1}^{N_B} \sigma_j, \tag{4.2}$$

and β is a factor which determines the level of approximation. We tried $\beta = 0.98$ which means that we were retaining ninety-eight percent of the total energy in the spectrum and filtering out the subspace vectors corresponding to high frequencies. In most cases, this choice of β yielded too few subspace vectors (approximately $N_B/4$ vectors). Thus, the results obtained from this technique were very similar to those obtained by simple partitioning of the objective function into a small number of groups. Fig. 4.17 shows an example for this conclusion. In the first case, we simply partitioned the data mismatch objective function into 10 groups (2 per well) to form the subspace vectors from the data mismatch part of the objective function and used 2 subspace vectors from the model mismatch part of the objective function. Thus, the total number of subspace vectors is equal to 12. The second case had more-or-less 12 singular vectors ($\beta = 0.98$) from a SVD of 47 subspace vectors at each Gauss-Newton iteration. The 47 subspace vectors are the same ones used in the previous section. Both techniques show similar slow convergence rate after 3 iterations. These results

also support our conclusion of the importance of the number of subspace vectors, not the exact choice.

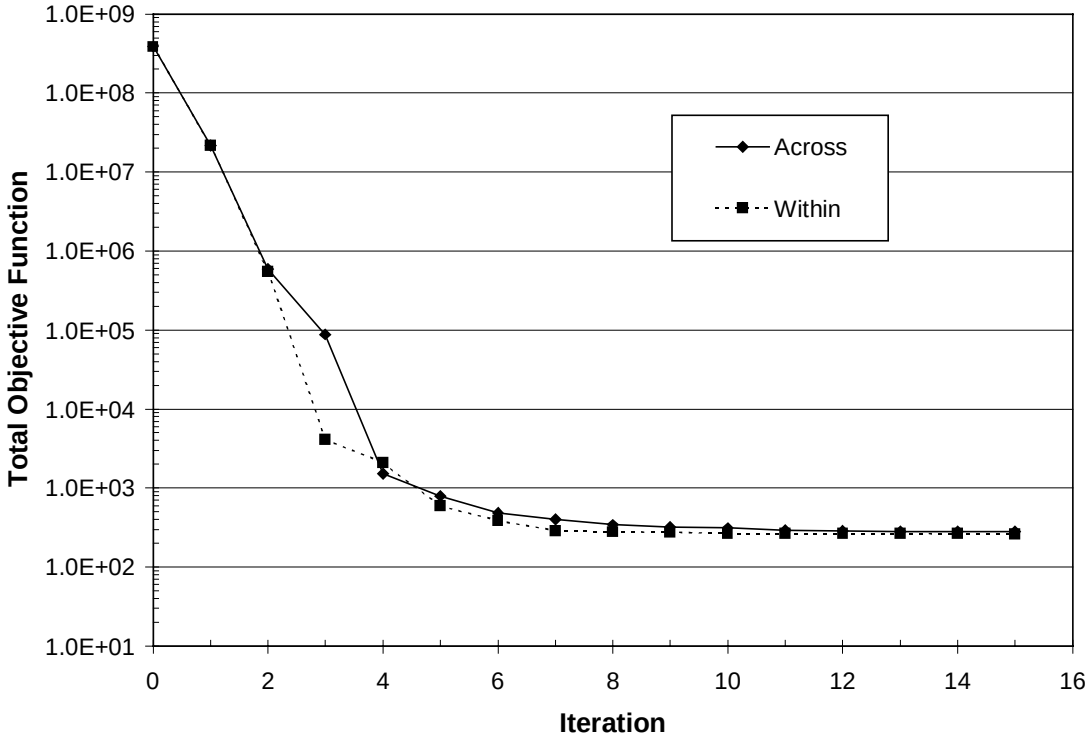


Figure 4.15: The convergence behavior of the objective function for two different methods of partitioning the data with 22 subspace vectors.

We should note that not every partitioning of the data provides an equally good set of subspace vectors. A particularly poor choice of initial subspace vectors often gives a large reduction in the data mismatch, but results in a large increase in the model roughness. For example, we ran a case using two subspace vectors for the data in Well 4, none for data in Well 3, and one subspace vector for each of the other 3 wells. We knew that this was not a true partitioning of the data as the data from Well 3 were not represented in any subspace vector. However, the purpose of running this case is to show that poor partitioning of the data mismatch objective function does

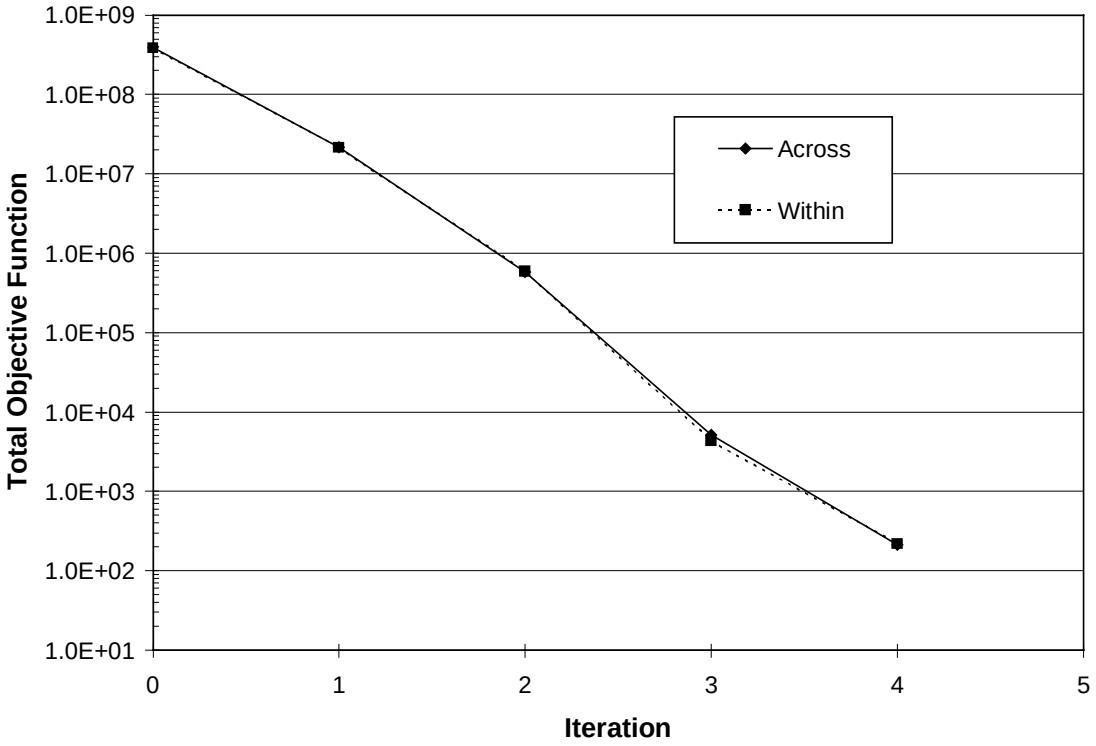


Figure 4.16: Comparison of the objective functions for two different methods of partitioning the data with 47 subspace vectors.

not yield good results. After 15 iterations, the data objective function was 1.7×10^6. The model roughness introduced in the first iteration, remained at the end. Fig. 4.18 shows the log-permeability field after the first Gauss-Newton iteration. This field is rough, meaning that it has a large deviation from the prior model, and has a large difference between the lowest and highest values of the gridblock log-permeabilities. In this case, recall that the prior model for log-permeability is an homogeneous field with each gridblock value equal to 4.0. A similar rough field is observed for porosity.

We also explored the use of constant vectors in our methodology. This idea was originally used by Oldenburg et al. [36] who have noted that convergence of the subspace method is accelerated by adding a constant vector. However, for the

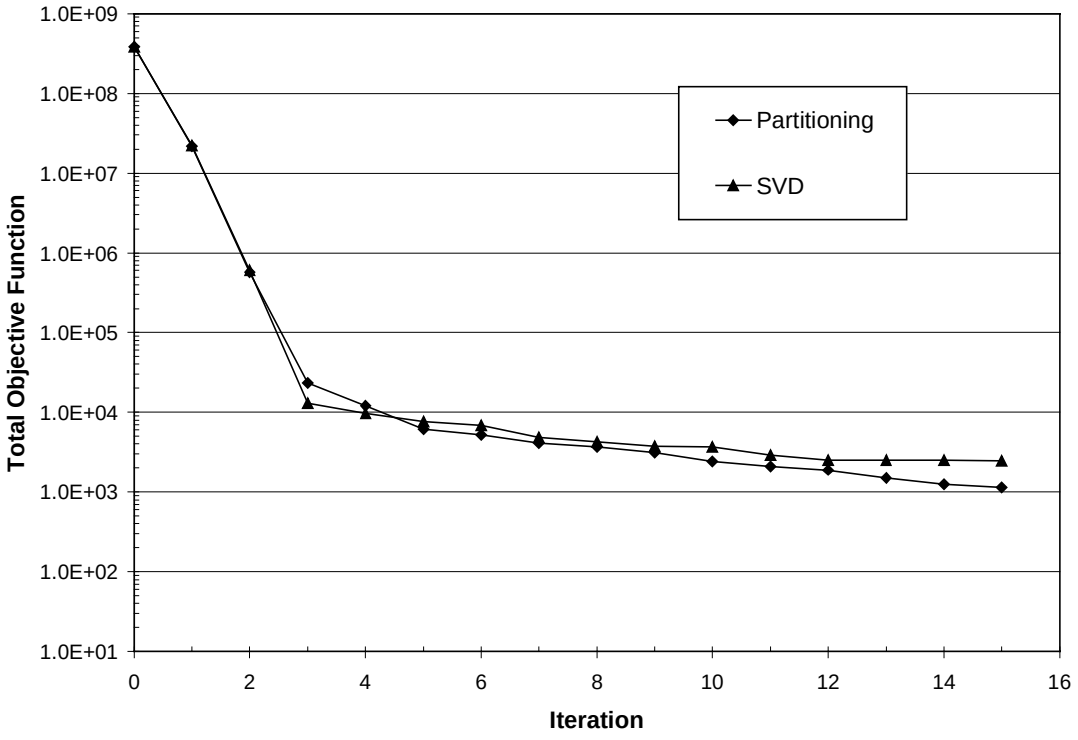


Figure 4.17: Comparison of the results obtained by simple partitioning and SVD.

subspace methodology we present here, including the constant vectors did not improve the convergence and did not yield any significant improvement in the resulting model estimates.

4.5 Gradual Increase in Dimension of Basis

In section 4.3, we showed 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 mismatch was always large, even for small numbers of subspace vectors. This suggests that an efficient strategy for minimizing the computational effort is to

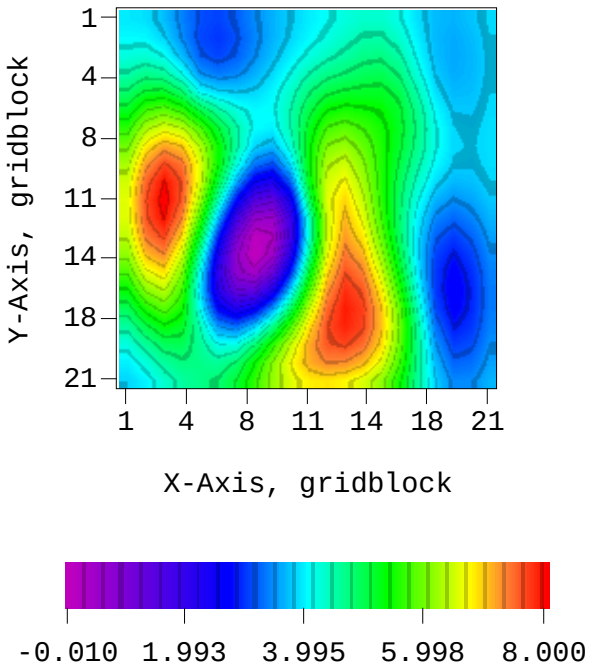


Figure 4.18: Log-permeability field after first iteration.

begin with a small number of subspace vectors in the early iterations, adding more when needed to maintain a high rate of convergence.

Gauss-Newton had worked quite well for most cases in which the number of subspace vectors was constant (e.g., 47 subspace vectors at all iterations), 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 this case, we typically found that the addition of new subspace vectors at some iterations resulted in an increase in the model roughness at the next iteration. Although the restricted step method was tried to cure this problem, it was difficult to remove the model "roughness" in subsequent iterations and reduction in the step size caused slow convergence. Fig. 4.19 depicts this case in terms of the reduction in the total objective function for the 2-D problem (labeled with G-N). 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 Gauss-Newton iteration. In Fig. 4.19, when we increased the number of subspace vectors from 17 to 27, we had the model roughness problem. Fig. 4.20, which is a plot of $O_M(m)$ (model mismatch objective function with the diagonal of C_M) versus iteration, illustrates the model roughness problem that occurred at the third iteration. As a result, convergence slowed down and the model became rough.

To solve this problem, we tried to use the Levenberg-Marquardt algorithm. However, in the standard implementation of the Levenberg-Marquardt algorithm with growth and decay factors of 10 for λ 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 again acquired "roughness" at the third iteration which was difficult to remove at later iterations (see the curve for $\lambda = 10^{-1}$ in Fig. 4.21). 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 $\lambda = 10^9$ in Fig. 4.21), or the rate at later iterations was small (see $\lambda = 10^5$ to 10^9 in Fig. 4.21). Although starting values for λ in the range 10^2 to 10^3 worked well for this 2-D example, it was necessary to start with λ between 10^6 and 10^7 and growth and decay factors of 2 to achieve a small value of the objective function for the 3-D reservoir example. So, we concluded that the value of initial λ and growth and decay factors of λ for efficient convergence of the Levenberg-Marquardt algorithm are problem dependent and difficult to determine without much experimentation. Thus, we required to seek a more robust implementation of the Levenberg-Marquardt algorithm.

In the standard Levenberg-Marquardt method, by trial and error, we tried to find a value of λ that gives the minimum of data mismatch with a smooth model at each iteration. By doing so, when the number of subspace vectors was gradually increased, the objective function was reduced nearly as fast as when 47 subspace

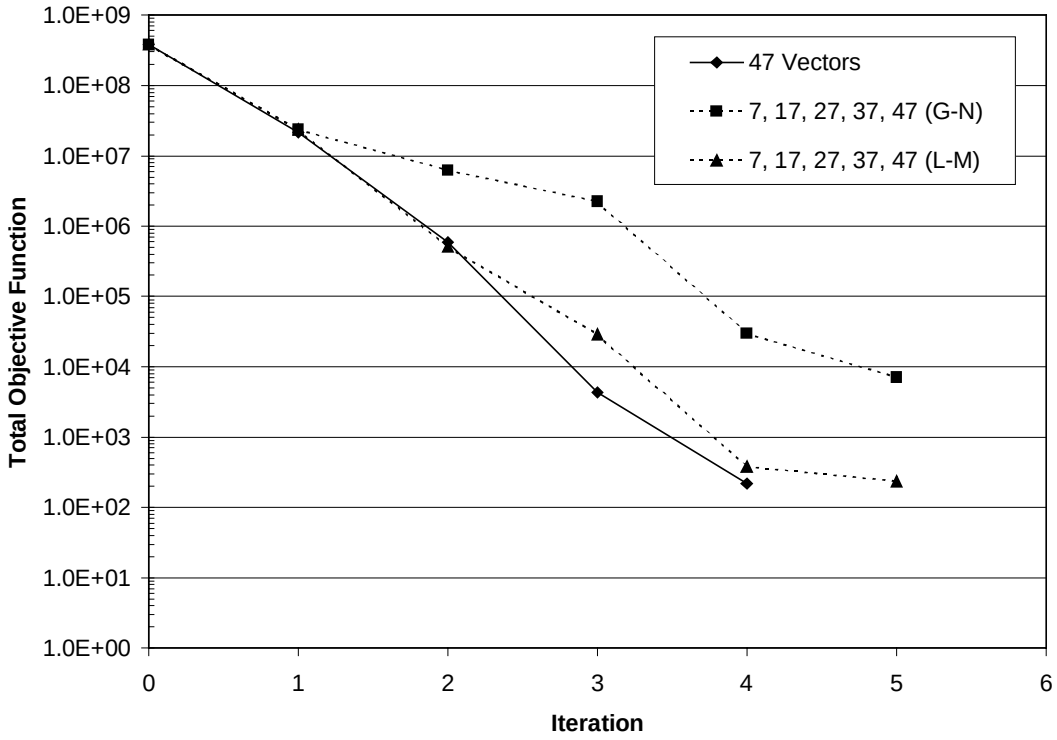


Figure 4.19: The convergence is only slightly slower, but the total work is reduced when the number of subspace vectors is gradually increased at each iteration with Levenberg-Marquardt method.

vectors were used in all iterations (see curve labeled with L-M in Fig. 4.19). Table 4.1 contains the schedule of the values of λ used in this case. Note that in this trial and error procedure, we start searching for optimal lambda at the second iteration because using 7 subspace vectors at the first iteration with Gauss-Newton method did not produce any problem in terms of convergence and roughness. However, this was not the case in the 3-D problem in which using 7 subspace vectors at the first iteration required a big value of λ to enhance the convergence and reduce the roughness. The success of this manual method motivated us to develop an automated method to find the optimum value of λ at each iteration. This procedure was explained in section

Iteration	λ
1	0.0
2	75.0
3	200.0
4	35.0
5	20.0

Table 4.1: Schedule of λ.

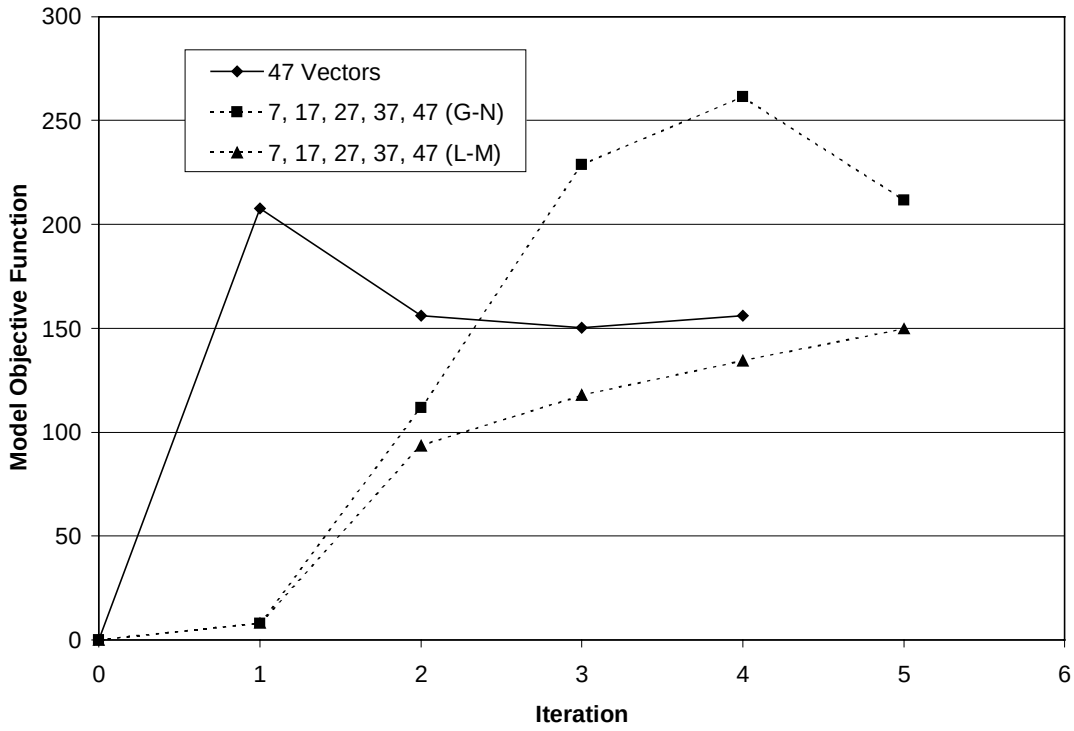


Figure 4.20: Comparison of $O_M(m)$ versus iteration for the three methods.

3.4. Recall that in this automated procedure we perform a one-dimensional search for the value of the λ that minimizes a weighted objective function, i.e.

$$O_{\mathrm{F}} = O_{\mathrm{D}} + \gamma_i O_{\mathrm{M}}. \tag{4.3}$$

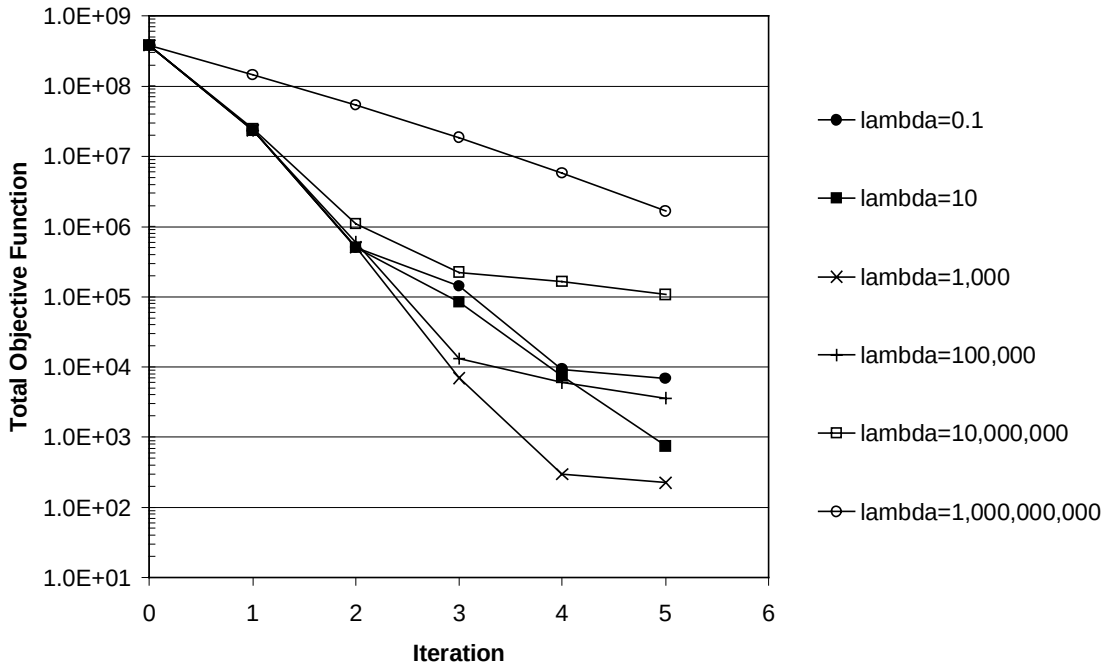


Figure 4.21: Reduction in the objective function for a wide range of starting values of the Levenberg-Marquardt damping factor.

As an example, Fig. 4.22 shows the values of O_D, O_M and O_F for different values of λ at the third iteration (27 subspace vectors) of the 2-D model with 7, 17, 27, 37, 47 and 57 subspace vectors used at iterations 1 through 6, respectively. As is expected, we see that the larger values of λ give smoother models. However, for the smooth models the data mismatch function is high. The trade-off between the data mismatch and model roughness can be obtained with the corresponding λ (which is $5,000$ for this case) at the minimum of O_F. For this case, γ obtained by applying Eq. 3.37 is equal to 560. Note that in this automated method we start searching for the optimal λ at the first iteration.

For one-dimensional minimization, we found that Brent's method is the most computationally efficient because it requires fewer iterations to find the optimal λ. It

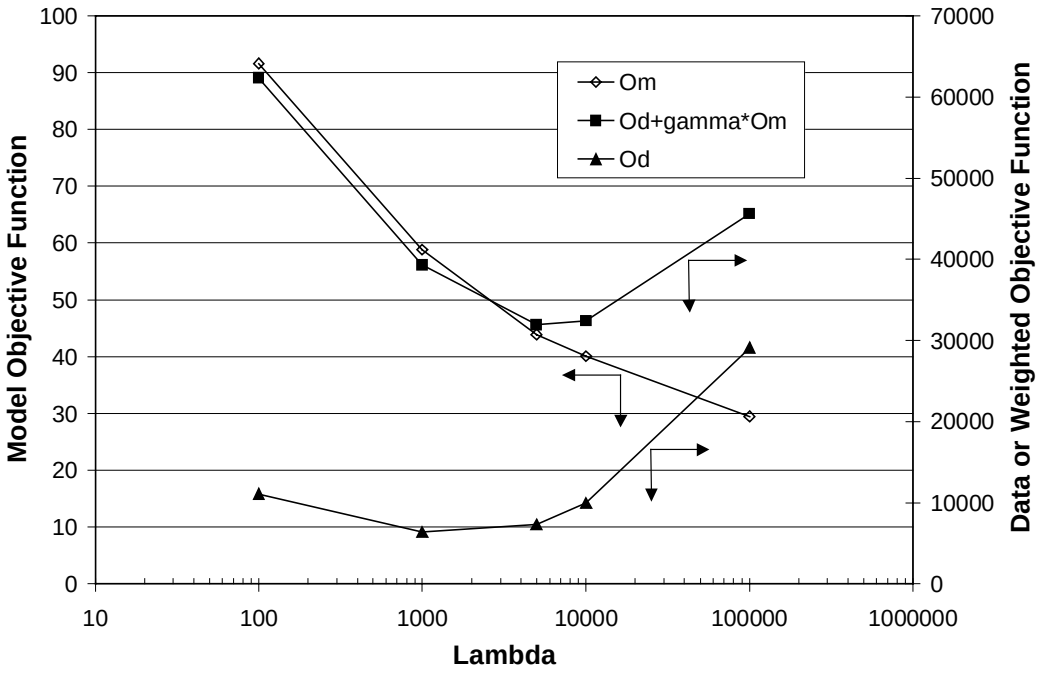


Figure 4.22: Effect of λ on the roughness for the 2-D problem with 27 subspace vectors.

is critical to reduce the number of iterations to find λ because each iteration requires an evaluation of the objective function which requires solution of the forward problem (i.e. a simulation run). With Brent's method we were able to find the optimal λ at each iteration of the Levenberg-Marquardt algorithm in 6-8 iterations for the 2-D problem and in 7-14 iterations for the 3-D problem with tolerance, TOL, equal to 0.01 (see subsection 3.4.2). We should note that the number of iterations required for the Brent's method to find optimal λ strongly depends on the tolerance. At some iterations, we observed that objective function, O_F, of Eq. 4.3 behaves more-or-less constant around the actual minimum. In this case, Brent's method required more iterations to find optimum λ. Thus, we introduced a new stopping criterion in Brent's method which reduces the number of iterations without significantly changing

the results. The new stopping criterion is based on the objective function, i.e.

$$\frac{O_F(\lambda_w) - O_F(\lambda_x)}{O_F(\lambda_w)} < \epsilon \tag{4.4}$$

where $O_F(\lambda_x)$ is the least objective function value found so far and $O_F(\lambda_w)$ is the second least objective function value. We found that using $\epsilon = 0.05$ works well for both 2-D and 3-D problems. Using this new stopping criterion number of iterations required to find the approximate optimum λ reduced to 3-4 iterations in the 2-D problem and reduced to 5-9 iterations in the 3-D problem without any changes in the convergence and model estimates.

The convergence behavior and schedule of optimal values of λ for the 2-D and 3-D problems were much different (Fig. 4.23). 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. Detailed time comparisons will be given in section 4.6. In the 3-D case, we have started with 1 subspace vector per well plus the 2 subspace vectors from the model mismatch part of the objective function. Thus, the total number of subspace vectors used at the first iteration is equal to 7. Then, we added 1 subspace vector per well at each Levenberg-Marquardt iteration. Thus, we ended up with 67 vectors at the 13th iteration. The work done in this variable subspace dimension case to compute the subspace vectors and the sensitivities to coefficients of subspace vectors was about 78% 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.

We have investigated the effect of γ or more specifically the effect of the constant in the denominator of γ on the results. Fig. 4.24 shows the total objective function for the constants 4, 8, and 16 in the denominator of γ for the 2-D model. The results are not so different. Similar results were obtained for the 3-D model

under consideration.

4.6 Computational Analysis

In this section, we make a general computational analysis for generating the MAP estimates or realizations of the rock property fields with N_M model parameters conditioned to N_D data by the conventional Bayesian inversion method and the suggested subspace reparameterization method. We will compare the methods in terms of the computational components which take the most time in one minimization iteration. In our comparisons, we use the conventional Bayesian inversion method in the form of the alternative formulation of the Gauss-Newton equation (Eq. 2.15) which requires the solution of a system of N_D equations. This alternative formulation is more efficient than the original Gauss-Newton equation (Eq. 2.14 which requires the solution of a system of N_M equations) since we generally have $N_D < N_M$. It is obvious that if we had compared the results from our subspace method with the original Gauss-Newton method we would have shown more efficiency gain.

Computational components in the conventional method that take a significant amount of time are:

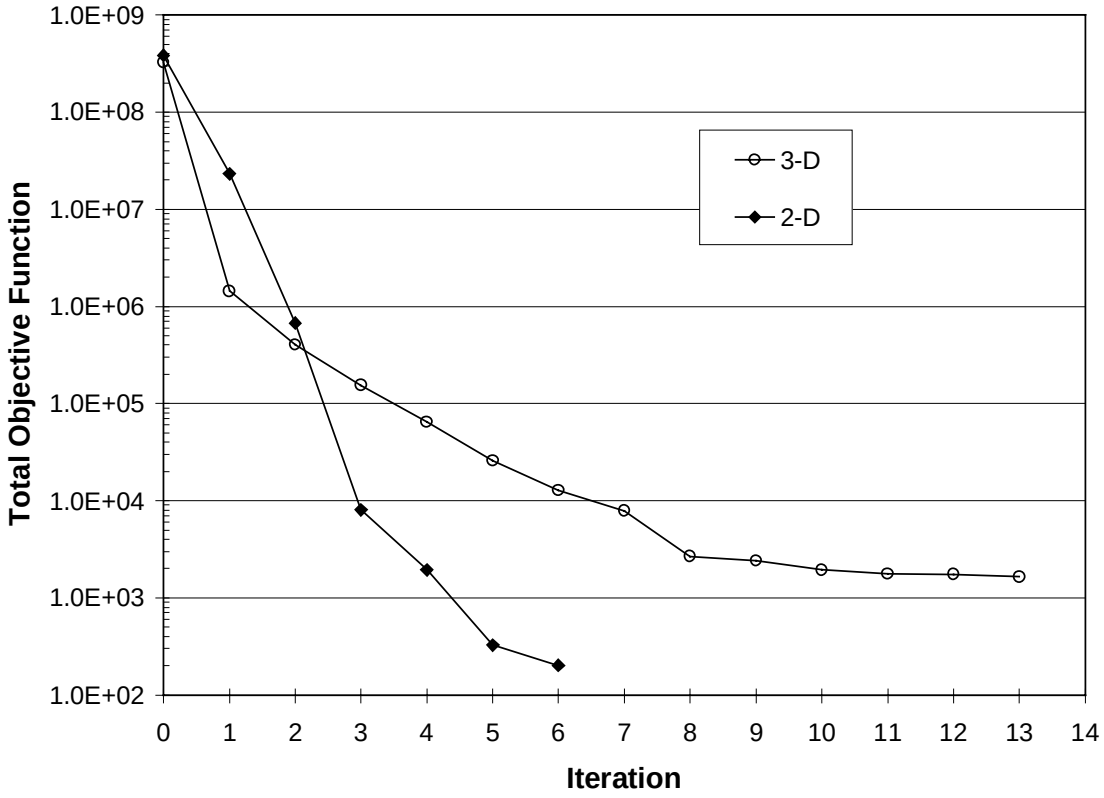
1. Calculation of sensitivity coefficients.

2. Computation of $C_M G^T$.

3. Computation of $GC_M G^T$.

4. Solution of $Ax = b$ where A denotes $N_D \times N_D$ matrix $(C_D + G_l C_M G_l^T)$ (see Eq. 2.15).

Similarly, computational components in the subspace method that take a significant amount of time are:



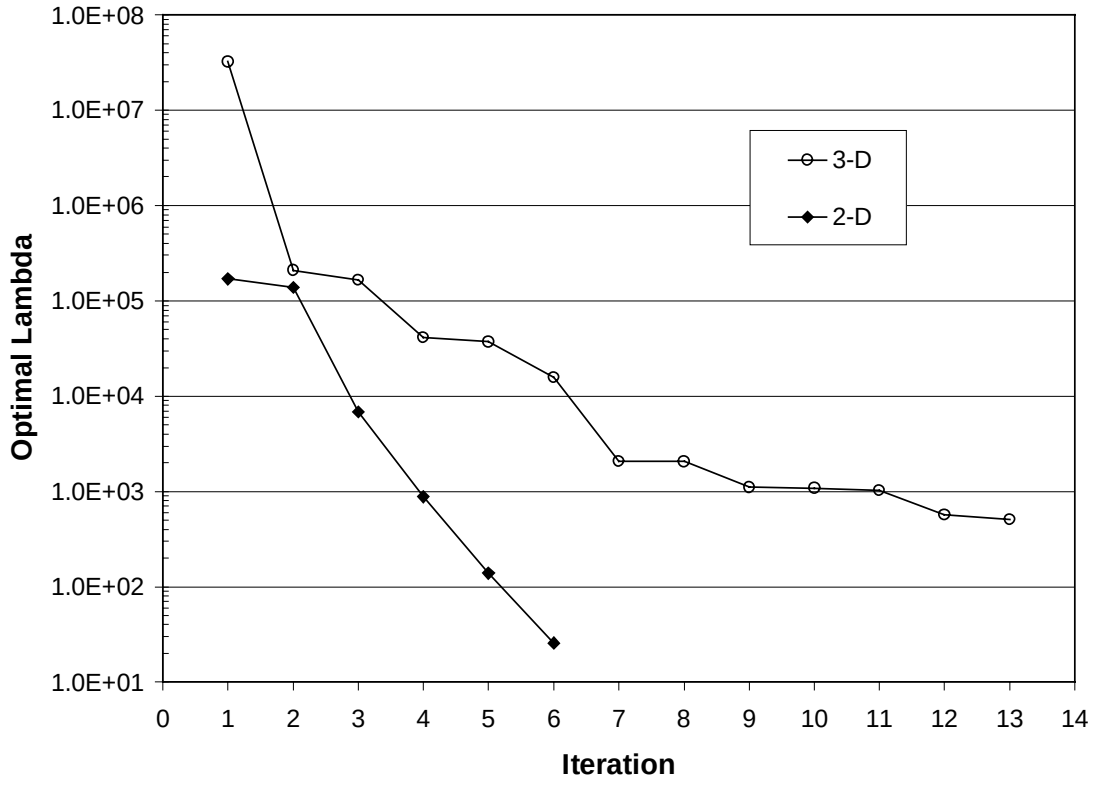


Figure 4.23: The rates of reduction in the objective function (above) and in the optimal value of λ (below) are much different for the 2-D and 3-D problems.

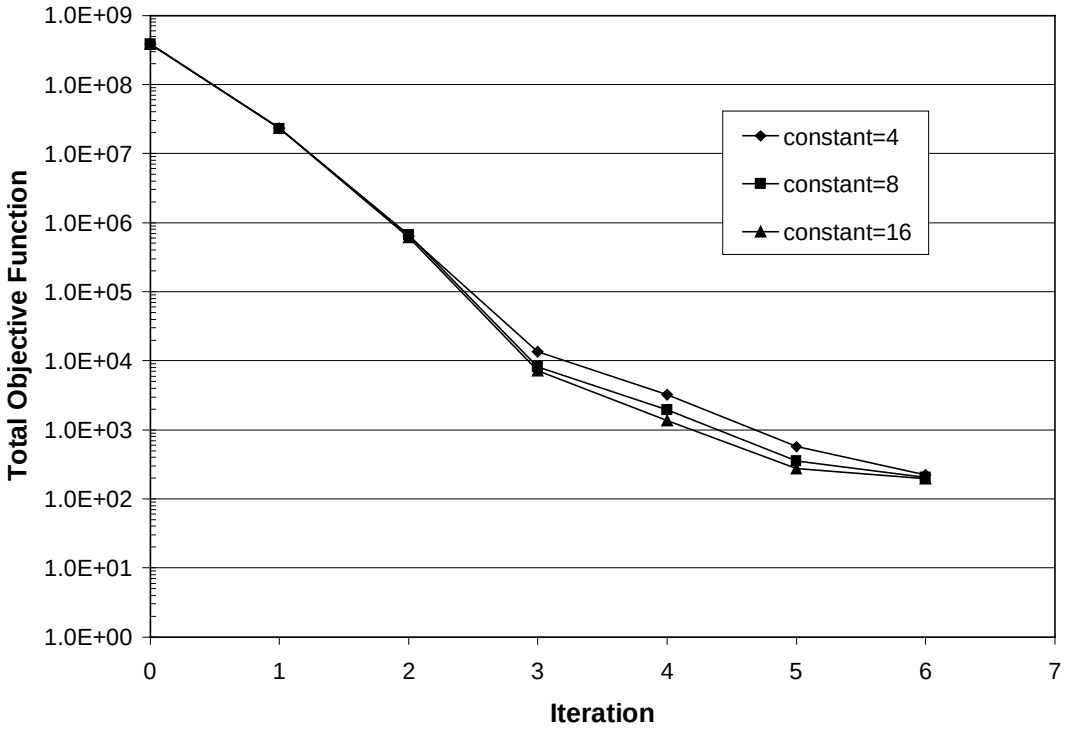


Figure 4.24: Effect of γ on the convergence of the objective function for the 2-D problem.

1. Calculation of subspace vectors.

2. Computation of $C_M B$.

3. Computation of sensitivity to subspace vectors, i.e. GA.

4. Solution of $Ax = b$ which is $N_B \times N_B$ (see Eq. 3.10) where N_B is the number of subspace vectors.

5. Finding optimum λ by one-dimensional minimization.

Generation of the MAP estimates or conditional realizations from the RML method using the conventional method requires computation of N_D sensitivity vec-

tors for each iteration of the minimization procedure. For single-phase flow in a fluid with small compressibility, a three-dimensional extension [19] of a method developed by Carter et al. [6] is known to be the most efficient method for computing sensitivity coefficients. With this method computation of N_D sensitivity vectors for N_w wells requires work equivalent to approximately $N_w + 1$ simulation runs. However, for multiphase flow problems Carter's technique is not applicable. Even though we have demonstrated the use of the subspace method by considering single-phase flow problems, we would like the method to be applicable to multiphase flow problems. For multiphase flow problems, the adjoint method is an efficient method for calculating sensitivity coefficients. Using the adjoint method, computational effort required to generate N_D sensitivity vectors would be equivalent to approximately N_D flow simulation runs.

In our subspace methodology, the main computational components are the computation of the subspace vectors (the ones generated from the data mismatch objective function) and the computation of the product GA which is the sensitivity to the coefficients of the subspace vectors. Computation of subspace vectors requires the solution of one adjoint system per subspace vector (Appendix-B). Roughly speaking, the computational time required to solve one adjoint system is equivalent to one simulation run. Thus, calculation of $(N_B - 2)$ subspace vectors (2 vectors from the model mismatch objective function are easily computed) requires approximately $(N_B - 2)$ simulation runs at each minimization iteration. The computational effort required to compute the sensitivity of pressure data to the coefficients of subspace vectors (GA) is approximately equivalent to N_B simulation runs. Thus, these techniques require only $2(N_B - 1)$ simulation runs for the sensitivity calculations at each iteration of the subspace method. Recall that $N_B < N_D$, so the subspace method is approximately $N_D/[2(N_B - 1)]$ times faster than conventional approach in terms of simulation runs for the sensitivity calculations. For example, for the 3-D example considered ($N_D = 430$ and $N_B = 47$ assuming 47 vector case at every iteration), this

ratio is equal to about 4.67.

In terms of matrix multiplications, the conventional method requires calculating $C_M G^T$ and $G C_M G^T$ whereas the subspace methodology requires only the computation of $C_M B$ at each minimization iteration. Recall that G^T is an $N_M \times N_D$ matrix and B is an $N_M \times N_B$ matrix, so the computational time required to compute $C_M B$ is N_D/N_B times less than the computational time required to compute $C_M G^T$. The time spent for the calculation of $G C_M G^T$ in the conventional method is totally saved in the subspace method.

In terms of solving $Ax = b$, the conventional method has $N_D \times N_D$ dimensional A matrix whereas in the subspace method the dimension of A is reduced to $N_B \times N_B$. Although the solution of 430 equations took a few seconds in the 3-D example problem that we considered, this would not be the case when N_D is very large.

In the subspace method, efficiency is increased by starting with as few subspace vectors as possible at early iterations, then gradually increasing the number of the subspace vectors in order to achieve the required reduction in the objective function. In this case, as we noted it is required to use the Levenberg-Marquardt method with optimal selection of λ to eliminate model roughness problems. Thus, finding the optimum λ by one-dimensional minimization requires extra simulation runs (5-9 runs for the 3-D problem we considered) at each iteration of the subspace method. However, reduction in the computational time required for the sensitivity calculations and matrix multiplications in the early iterations is more than the computational time required to find the optimum λ at all iterations.

CHAPTER V

CONCLUSIONS

In this study, an efficient method of reparameterization for solving large-scale reservoir inverse problems with large amounts of production data using the subspace methodology has been developed. We demonstrated the use of the subspace methodology for the problems of generating maximum a posteriori estimates and realizations of log-permeability and porosity fields conditioned to synthetic pressure data for single-phase flow for both 2-D and 3-D cases. However, the subspace methodology developed in this study is applicable to multiphase flow problems.

Based on this study, we arrive at the following conclusions.

1. We have shown that it is possible to construct the model correction vector, δm from a reduced basis, whose dimension is much less than the number of data (N_D) or the number of model parameters (N_M), without affecting the result or the number of iterations required to obtain a minimum.

2. We have shown that 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 partitioning. Any reasonable partitioning of the data gave similar results.

3. On the other hand, we observed that the method is sensitive to number of subspace vectors. If too few subspace vectors are chosen, the number of Gauss-Newton iterations required is very large. The initial rate of reduction in the

objective function is largely independent of the number of subspace vectors, however.

4. We have shown that there is an optimal number of basis vectors such that the convergence of the Gauss-Newton method is unaffected by the reparameterization. Unfortunately, computation of the optimal set of basis vectors requires computation of eigenvalues of the dimensionless Hessian matrix which could be too expensive to be practical.

5. We have shown that an efficient strategy for minimizing the computational effort is to begin with a small number of subspace vectors in the early iterations and to gradually increase the number of subspace vectors at subsequent iterations. Unfortunately, the process of increasing the number of subspace vectors at each Gauss-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 is computed at each iteration.

NOMENCLATURE

C_D covariance matrix for pressure measurement errors.

C_M prior covariance matrix.

d vector of observed data.

G_l sensitivity coefficient matrix at lth Gauss-Newton iteration.

$g(m)$ calculated pressure data from simulation.

k permeability, md.

N_M total number of model parameters.

m_p vector of prior means of model parameters.

N number of simulator gridblocks.

N_D number of conditioning pressure data.

N_w number of wells at which data are measured.

N_B number of subspace vectors.

ϕ porosity, fraction.

Subscripts

k related to log-permeability field.

uc unconditional.

ϕ related to porosity field.

Superscripts

T matrix transpose.

-1 matrix inverse.

REFERENCES

[1] 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.

[2] S. Betancourt and D. S. Oliver. Comparison of sampling methods for uncertainty evaluation. In *TUPREP Research Report 17 (May 22, 2000)*, pages 91–106, 2000.

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

[4] Robert Bissell, O. Dubrule, P. Lamy, P. Swaby, and O. Lepine. Combining geostatistical 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.

[5] 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.

[6] 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.

[7] 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.

[8] 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.

[9] 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.

[10] 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.

[11] T. Deschamps, T. Grussaute, D. Mayers, and R. Bissel. The results of testing six different gradient optimisers on two history matching problems. In *Proceedings of the 6th European Conference on the Mathematics of Oil Recovery*, pages B–24, 1998.

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

[13] 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.

[14] Gene H. Golub and Charles F. van Loan. *Matrix Computations*. The Johns Hopkins University Press, Baltimore, second edition, 1989.

[15] 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.

[16] J. Jaime Gómez-Hernández, Andrés Sahuquillo, and José E. Capilla. Stochastic simulation of transmissivity fields conditional to both transmissivity and piezometric data. 1. Theory. *Journal of Hydrology*, (203):162–174, 1997.

[17] J. Hadamard. Sur les problémes aux dérivées partielles et leur signification physique. *Princeton University Bulletin*, 13:49–52, 1902.

[18] William W. Hager. Updating the inverse of a matrix. *SIAM Review*, 31(2):221–239, 1989.

[19] 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.

[20] Nanqun He, Dean S. Oliver, and Albert C. Reynolds. Conditioning stochastic reservoir models to well-test data (SPE-38655). In *1997 SPE Annual Technical Conference and Exhibition*, 1997.

[21] Bjørn Kåre Hegstad, Henning Omre, Håkon Tjelmeland, and Kelly Tyler. Stochastic simulation and conditioning by annealing in reservoir description. In M. Armstrong and P. A. Dowd, editors, *Geostatistical Simulation*, pages 43–55. Kluwer Acad., 1994.

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

[23] 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.

[24] André Journel and C. J. Huijbregts. *Mining Geostatistics*. Academic Press, New York, 1978. 600 p.

[25] Rintu Kalita. Conditioning a three dimensional reservoir model to gas production data. Master's thesis, The University of Tulsa, 2000.

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

[27] J. E. Killough, Yogeshwar Sharma, Alain Dupuy, Robert Bissel, 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.

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

[29] A. Marsh LaVenue and John F. Pickens. Application of a coupled adjoint sensitivity and kriging approach to calibrate a groundwater flow model. *Water Resour. Res.*, 28(6):1543–1569, 1992.

[30] A. Marsh LaVenue, Banda S. RamaRao, Ghislain de Marsily, and Melvin G. Marietta. Pilot point methodology for automated calibration of an ensemble of conditionally simulated transmissivity fields, 2. Application. *Water Resour. Res.*, 31(3):495–516, 1995.

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

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

[33] R. Li, A. C. Reynolds, and D. S. Oliver. Computation of sensitivity coefficients for three-phase flow production data. In *TUPREP Research Report 17 (May 22, 2000)*, pages 127–158, 2000.

[34] 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.

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

[36] 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.

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

[38] Dean S. Oliver. Incorporation of transient pressure data into reservoir characterization. *In Situ*, 18(3):243–275, 1994.

[39] Dean S. Oliver. Multiple realizations of the permeability field from well-test data. *Soc. Petrol. Eng. J.*, 1(2):145–154, 1996.

[40] 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.

[41] Ahmed Ouenes, B. Bréfort, G. Meunier, and Dupéré. A new algorithm for automatic history matching: Application of simulated annealing method (SAM) to reservoir inverse modeling. *SPE 26297*, 1993.

[42] Ahmed Ouenes, Rao S. Doddi, Yinghua Lin, George Cunningham, and Naji Saad. A new approach combining neural networks and simulated annealing for solving petroleum inverse problems. In *4th European Conference on the Mathematics of Oil Recovery*, 1994.

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

[44] D. W. Peaceman. Interpretation of well block pressures in numerical reservoir simulation. *Soc. Pet. Eng. J.*, pages 183–194, June 1978.

[45] D. W. Peaceman. Interpretation of well-block pressures in numerical reservoir simulation with non-square grid blocks and anisotropic permeability. *Soc. Pet. Eng. J.*, pages 531–543, June, 1983.

[46] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes in FORTRAN: The Art of Scientific Computing.* Cambridge University Press, Cambridge, England, 1992.

[47] 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.

[48] 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.

[49] Albert C. Reynolds, Nanqun He, and Dean S. Oliver. Reducing uncertainty in geostatistical description with well testing pressure data. In *Fourth International Reservoir Characterization Technical Conference*, 2–4 March 1997.

[50] Frédéric Roggero. Direct selection of stochastic model realizations constrained to historical data. *SPE 38731, Annual Technical Conference*, pages 155–165, 1997.

[51] M. K. Sen, Akhil Datta Gupta, P. L. Stoffa, L. W. Lake, and G. A. Pope. Stochastic reservoir modeling using simulated annealing and genetic algorithm. In *Proceedings from the 67th Annual Technical Conference and Exhibition of the Society of Petroleum Engineers*, pages 939–950, October 1992.

[52] 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.

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

[54] Håkon Tjelmeland, Henning Omre, and Bjorn Kåre Hegstad. Sampling from Bayesian models in reservoir characterization. Technical Report Statistics No. 2/1994, Norwegian Institute of Technology, Trondheim, Norway, 1994.

[55] D. W. Vasco, A. Datta-Gupta, and J. C. S. Long. Integrating field production history in stochastic reservoir characterization, SPE-36567. In *Proccedings of the 1996 Annual Technical Conference of the SPE*, pages 829–840, 1996.

[56] Xian-Hun Wen, Clayton V. Deutsch, and A. S. Cullick. High resolution reservoir models integrating multiple-well production data, SPE-38728. In *Proccedings of the 1997 Annual Technical Conference of the SPE*, pages 115–129, 1997.

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

[58] 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.

[59] Guoping Xue and Akhil Datta-Gupta. Structure preserving inversion: An efficient approach to conditioning stochastic reservoir models to dynamic data. *1997 SPE Annual Technical Conference*, SPE 35412, 1997.

[60] Pin-Huel Yang and A. Ted Watson. Automatic history matching with variable-metric methods. *SPE Reservoir Engineering*, pages 995–1001, August 1988.

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

APPENDIX A

SINGLE-PHASE FLOW SOFTWARE FOR CONDITIONING A GEOSTATISTICAL MODEL TO WELL-TEST PRESSURE DATA

A.1 General Information

The program, written in standard FORTRAN 77, facilitates generation of realizations of three-dimensional rock property fields (simulator gridblock values of log-permeability and porosity) conditioned to multiwell pressure data and prior information using inverse problem theory. The reservoir model is assumed to be a rectangular parallelepiped of uniform thickness h. Reservoir boundaries are assumed to be no-flow boundaries. We assume either two- or three-dimensional single-phase flow in a Cartesian coordinate system. The reservoir can contain any number of complete-penetration or restricted-entry wells. Each well is produced at a specified sequence of rates where the rate may vary with time and may be different at each well; pressure buildup at a well is simulated by setting the rate to zero subsequent to a producing period. Interference or observation wells are simulated by setting the rate to zero at all times. The permeability and porosity fields are assumed to be heterogeneous. Since log-permeability is often modeled as a Gaussian random variable, the natural logarithm of gridblock values of permeability is used. The code assumes that the prior model for the rock property fields is multivariate Gaussian, characterized by means and covariances (arising from a specified variogram) with permeability assumed to be log-normal, porosity assumed to be normal with correlation between log-permeability and porosity determined by specifying a correlation coefficient. As the model is currently written, permeability is assumed to be isotropic. However, the code can be easily extended to anisotropic permeability case. In the three-dimensional

case, one can specify that vertical permeability in each gridblock is equal to horizontal permeability multiplied by an input constant factor. If the prior geostatistical model is non-Gaussian, then one can still condition a realization generated from the prior model to pressure data, but to do so, one has to first estimate means and variograms (or covariances) from the geostatistical realization. Uncertainty in the prior means is incorporated using a partially, doubly-stochastic model; see Reynolds et al. [49]. The program can be used to generate either the maximum a posteriori estimate (most probable model) or realizations conditioned to the prior model and multiwell pressure data. The maximum a posteriori estimate is generated by minimizing the objective function derived from the a posteriori pdf using the procedure described by He [20]. A complete description of the procedure used to generate sensitivity coefficients can be found in He [20]. In generating the maximum a posteriori estimate and in generating realizations, one can incorporate hard data for porosity at any gridblock penetrated by a well. One must specify the variance of the measurement error in hard data. This variance can be small but not zero. Hard data for permeability is not allowed.

Skin factors at active wells can be estimated directly as part of the conditioning process. The recommended procedure is to estimate one skin factor per well. For three-dimensional problems, one can estimate a skin factor for each gridblock penetrated by a well, but these individual "layer" skin factors can not be resolved accurately, unless layer flow rates are available.

Pressure responses are obtained by a standard purely-implicit, seven-point finite difference simulator, where wellbore pressure is related to the well's gridblock pressures by Peaceman's method. Detailed flow equations are given in the first section of Appendix-B.

A.2 Source Code Files

There are 8 source code files (INVS.FOR, SUBR.FOR, SIMSEN.FOR, COVCAL.FOR, SUBSP.FOR, MXSOLV.FOR, QR.FOR and DIRECT.FOR) in the program. In the following we briefly describe these files.

INVS.FOR: This file includes the main code in which minimization of the objective function using the restricted-step Gauss-Newton method (or using Levenberg-Marquardt) is carried out. All variables are defined at the beginning of the code. Throughout the code, the ongoing computations are fully explained using comment statements. In applying the Gauss-Newton method, iteration stops when any one of the following stopping criteria is chosen (user has these options to choose):

1) Stop if data mismatch is satisfied to a specified tolerance, i.e.

$$RMSE = \sqrt{\frac{1}{N_d}\sum_{i=1}^{N_d}\big(d_i - g_i(m)\big)^2} \leq \varepsilon_1 \sigma_d, \tag{A.1}$$

where $RMSE$ stands for the root-mean-square error, d_i represents the ith data measurement when the MAP estimates are generated or represents the ith unconditional simulation of the data when the RML method is applied and $g_i(m)$ represents the ith calculated data, and σ_d is the standard deviation of the data measurement errors. In the code EP1, which is an input, corresponds for ε_1 and is currently chosen as 1.0.

2) Stop if

$$\frac{\|\delta m^{l+1}\|}{\|m^l\| + 10^{-12}} \leq \varepsilon_2, \tag{A.2}$$

where m is the vector of the model parameters and the norm in this equation is the two-norm. ε_2 (EP2 in the code) is currently set to 10^{-5}.

3) Stop if the maximum number of iterations (MNI in the code) exceeds an input value (currently set as 13).

4) Stop if

$$\frac{O(m^l) - O(m^{l+1})}{O(m^l) + 10^{-14}} \leq \varepsilon_3, \tag{A.3}$$

where $O(m^l)$ and $O(m^{l+1})$ are the values of the objective function at the previous and current iterations, respectively. Currently, the code uses a diagonal approximation to the covariance matrix for evaluation of these objective functions. In Eq. A.3, ε_3 is denoted by EP3 in the code and its input value is currently set to 10^{-3}.

SUBR.FOR: This file includes several subroutines (namely INPUT, PIKSRT, MEANC, LUDCMP, LUBKSB and TIMEINT) for different purposes. Subroutine INPUT is basically used for reading the data for the reservoir simulator. If non-uniform grid sizes in any of the directions are used, the data are also read in this subroutine. Subroutine PIKSRT sorts an array into ascending numerical order, by straight insertion. Subroutine MEANC is called by the main code when we want to incorporate the uncertainty in prior means of porosity and log-permeability. This file also contains subroutines LUDCMP and LUBKSB from Numerical Recipes to do LU decomposition.

Construction of the time steps for the reservoir simulator is done in the subroutine TIMEINT. Details on the construction of time steps is explained next.

We model the reservoir with multiple wells where rate changes may occur at different times at each well. Thus, we need to change time steps whenever there is a discrete rate change (or pressure change if the well constraint represents the specification of the flowing bottom-hole pressure) at any well. Let T_{f_j}, $j = 1, 2, ...$ denote the duration of each time interval (total times). In subroutine TIMEINT, we construct the T_{f_j}'s using the duration of each rate at each well. For example, suppose we have three wells, well 1 produces at fixed rate of 80.0 rb/day for all times, well 2 produces at a constant rate 150.0 rb/day for the first 1.5 days, which is followed by a two day buildup test, then we bring it on production at a constant rate 200.0 rb/day for 6.5 days; well 3 produces at a constant rate 750.0 rb/day for 8 days and then we run a two day buildup test. Note the total simulation run is ten days. Table A.1 contains the input data.

From this input the subroutine TIMEINT computes the total time sequence

Well Number or Index	Rate, rb/day	Duration, days
1	80.0	10.0
2	150.0	1.5
2	0.0	2.0
2	200.0	6.5
3	750.0	8.0
3	0.0	2.0

Table A.1: Input data.

for T_{f_j}'s as $T_{f_1} = 1.5$, $T_{f_2} = 3.5$, $T_{f_3} = 8.0$ and $T_{f_4} = 10.0$ days and the total number of flow periods as 4. Note that a new $\bar{T}_{f_j}$ value is defined whenever there is a rate change at any well.

To construct the time steps, we input for all flow periods an initial time step size, (DT0 in the program); a maximum allowable time step change (DTMAX) and a time step multiplier (TSM) which is denoted by α. Then, time steps are calculated as

$$t^{n+1} = t^n + \Delta t^n, \tag{A.4}$$

where

$$\Delta t^n = \alpha \Delta t^{n-1}. \tag{A.5}$$

Suppose for concreteness, we are constructing the solution on the time interval $[T_{f_1}, T_{f_2}] = [1.5, 3.5]$ so we set $t^0 = 1.5$ days and set Δt^0 equal to the input value. At the first step, we simply apply Eq. A.4 to get t^1, but at the subsequent steps, we use the multiplier, i.e. apply Eq. A.5 before applying Eq. A.4. After computing t^{n+1}, we perform a check before computing the solution at the new time step as follows:

If $t^{n+1} > T_{f_2}$ then set $t^{n+1} = T_{f_2} (\Delta t^n = T_{f_2} - t^n)$.

If $t^{n+1} < T_{f_2}$ but $t^{n+1} + 0.5\alpha\Delta t^n \geq T_{f_2}$, then set $t^{n+1} = T_{f_2}(\Delta t^n = T_{f_2} - t^n)$.

SIMSEN.FOR: This file includes subroutines to perform reservoir simulation and the sensitivity coefficient calculations. The reservoir simulator can be applied to two-dimensional or three-dimensional, single-phase flow in an $x - y - z$ coordinate system. The flow equation is discretized using a standard seven-point finite-difference scheme. Subroutine SENS can be considered as the main code for simulation and sensitivity coefficient calculation. It calls subroutine TRANS to generate transmissibilities, subroutine AMAT to form the coefficient matrix and right hand side vector for the finite difference equations, subroutine WELLIND to calculate well indices and derivatives of the well indices, subroutine SOURG to set the sink term for gridblocks penetrated by a well and subroutine SENSIT to calculate sensitivity coefficients using Carter's method. In the subroutine SENSIT, one should note that we interpolate the pressure data calculated from the simulator to the times at which we want to calculate the sensitivity coefficients - the times at which we have conditioning pressure data do not need to correspond to simulator time steps.

COVCAL.FOR: This file contains several subroutines and functions to generate the prior covariance matrix, C_M, for a specified variogram structure. Subroutine COVANCE, which is essentially identical to subroutine LUSIM from GSLIB (Geostatistical Software Library), is used to generate the prior covariance matrix. Since the prior covariance matrix is a sparse matrix (number of nonzero entries depends on the variogram range), a significant reduction in memory requirements is obtained by storing only non-zero entries of the prior covariance matrix. In the program, we have taken advantage of the sparseness of the prior covariance matrix by only storing the non-zero elements using the following procedure: Let the maximum correlation length (measured in units of gridblock length) in the three gridblock directions be L_x, L_y and L_z. For any parameter in the grid, only the covariance with parameters in two regions (there are two parameters per cell; permeability and porosity) of dimension $L_T = (2L_x - 1) \times (2L_y - 1) \times (2L_z - 1)$ must be stored because all other

values of the covariance are essentially zero. Instead of storing a sparse covariance matrix of dimension $2N_T \times 2N_T$ (where N_T is the total number of cells), we store a dense matrix whose dimensions are $2N_T \times 2L_T$ where, typically, $L_T \ll N_T$. If in this storage procedure, one wishes to use non-uniform grid sizes in any of the directions, then L_x, L_y and L_z must be based on the finest gridblock size. In the subroutine COVANCE, the array MGXYZ stores the limits of nonzero covariance.

SUBSP.FOR: This file includes subroutines and functions to apply our current implementation of the subspace method. If one does not want to apply the subspace method, this file can be discarded. Subroutine SUBSPACE contains the main code for the subspace method. In this subroutine, minimization of the objective function is also carried out. For minimization, we have three options to use: Gauss-Newton with restricted step, standard Levenberg-Marquardt method and modified Levenberg-Marquardt method (with optimal selection of λ). In the modified Levenberg-Marquardt method, we need to search for the λ as a one-dimensional minimization problem. For this one-dimensional minimization, there are two options for the user to specify: Golden Section search method and Brent's method. These options are chosen by specifying flags in the input file STAT.PAR which will be explained later. We suggest to use modified Levenberg-Marquardt method for minimization of the objective function and Brent's method for one-dimensional minimization of λ with the current implementation of the subspace methodology.

As noted in Chapter-III of this dissertation, we have formulated the subspace method such that it is unnecessary to solve linear systems with the prior covariance matrix as the coefficient matrix. Formally, C_M^{-1} is required to incorporate the restricted-step method into the Gauss-Newton procedure, but in the current implementation of the restricted-step, we simply replace C_M^{-1} by the inverse of the diagonal of C_M.

The recommended procedure for the subspace method is outlined in section 3.7 of Chapter-III. One of the important features of this procedure is that we begin

with a small number of subspace vectors in the early iterations and gradually increase the number of subspace vectors at subsequent iterations. We usually start with one subspace vector per well by grouping all the data corresponding to a particular well and increase the number of vectors by adding 1 or 2 (user can specify) more subspace vectors per well at each iteration. This somewhat ad hoc increasing procedure worked well for all synthetic examples that we considered.

MXSOLV.FOR: This file includes several subroutines to compute the solution of the linear system $Ax = b$ using a preconditioned iterative solver. There are four solvers available in this file. These are Conjugate Gradient method, Generalized Minimal Residual (GMRES), Bi-Conjugate Gradient and Minimal Residual. We always use the Conjugate Gradient method.

DECOMP.FOR: This file contains several subroutines to do matrix decomposition. Subroutine SVDCMP and function PYTHAG are used to do singular value decomposition. QR decomposition is done using several other subroutines in this file.

DIRECT.FOR: This file includes the subroutines to calculate the sensitivity coefficients using the direct or finite difference method. This file is just for checking the calculations of sensitivity coefficients.

A.3 Input Data Files

Most of the FORTRAN source code files utilize the FORTRAN include file "INCLUDE.DAT" which contains parameters that define the sizes of the various arrays at compile time. Definitions of the parameters are given in this file. In the event any of the parameters are changed, the entire program must be recompiled. This file also contains the common blocks.

Data are input to the program via three data files: STAT.PAR, SIM.PAR and PRESS.PAR. These files are needed for all the cases.

All data files contain the variables describing the case to be run. Data are input in free format, i.e., individual data entries are separated either by spaces or commas within the first 72 columns of the file. Comment lines may be added arbitrarily throughout the data files. Comment lines may begin with 'c' or 'C'. Any line which starts with 'c' or 'C' in the first column is removed from the data file by the program before the run is started.

Three temporary files STAT.DAT, SIM.DAT and PRESS.DAT are created on the user directory; these temporary files contain the uncommented user data in the input files STAT.PAR, SIM.PAR and PRESS.PAR, respectively. In the following, we briefly explain the data files STAT.PAR, SIM.PAR and PRESS.PAR since all these files are self-documented.

STAT.PAR: This file contains mainly statistical and geostatistical data (variogram information). The first two cards are used for specifying the optimization procedure. In Card 1, the flag MTHD that can take three different integer values (namely 1, 2 and 3) specifies the optimization algorithm. If MTHD=1, Gauss-Newton with restricted step is used as the method of minimization. In this case Card 2 is not included. If MTHD=2, standard Levenberg-Marquardt method is the minimization algorithm. In this case, initial value of λ, and growth and decay factors of λ must be specified in Card 2. Finally, if MTHD=3, modified Levenberg-Marquardt method (with optimal selection of λ) is used. This option can be chosen if and only if subspace method is used. In this case, the method of one-dimensional search and tolerance for the termination of one-dimensional search must be specified in Card 2. We can either choose Golden Section search (ISRCH=1) or Brent's method (ISRCH=2) for one-dimensional minimization. (We recommend to use ISRCH=2.) The third card in this file consists of several flags; each of which can be assigned a value of either 0 or 1. As an example, the program can generate either maximum a posteriori estimates (MAPE=1) or realizations (MAPE=0). In the program, we can condition to hard data for porosity (i.e., IHARD=1). Card 4 must be supplied if we have hard data for

porosity. We assume that the hard data are collected at the same locations as the observed pressure (or rate) data. At each well, a porosity value (hard data) is read in for each gridblock in the z-direction.

SIM.PAR: This file includes the data required for the reservoir simulation run. We specify the type of production (i.e., constant flowrate production or constant wellbore pressure production) via the flag ICQP. We have an option to estimate one skin factor at each well (flag ISKIN=1) or one skin factor for each gridblock in the z-direction (ISKIN=0).

PRESS.PAR: This file is used for reading observed pressure (or rate) data. Time values at which we have the data are also supplied by this file.

In addition, three input files KXINI.DAT, PINI.DAT and SINI.DAT are used to input prior models (unconditional realizations) for the log-permeability field, porosity field and skin factors respectively, for the case where we wish to generate realizations. If we wish only to generate maximum a posteriori estimates, then we do not need to include the files KXINI.DAT and PINI.DAT. Note that if one wishes to generate more than one realization, all the unconditional realizations for log-permeability and porosity must be stored in the same files KXINI.DAT and PINI.DAT respectively. In this case, the parameter NR in the INCLUDE.DAT file must be set equal to the number of realizations which we wish to generate. In the input files, KXINI.DAT and PINI.DAT, unconditional realizations for both log-permeability field and porosity field must be stored as follows: Let i, j, and k denote the indices for the x, y, and z-directions, respectively. We first start with first gridblock in the z-direction (i.e., $k = 1$, top layer, see Fig. A.1), then the y-direction index, j, increases from top to bottom (see Fig. A.2). For each index j, the x-direction index increases from left to right. The same ordering is repeated for $k = 2, 3, ..., N_z$, where N_z is the total number of gridblocks in z-directions. Well locations are indexed in the same order. Note that this same storage scheme is used in the output files which contain model parameters. This is done for plotting purposes, but log-permeability and porosity fields are stored

as 1-D arrays in the code. Skin factors, in both input and output files, are stored in a similar way, i.e., starting from the first gridblock in the z-direction. Skin factors should be read in the same order as well numbers, i.e. first skin factor value corresponds to first well, etc. Well numbers are determined by the order of the reading of the well indices in the input file SIM.PAR.

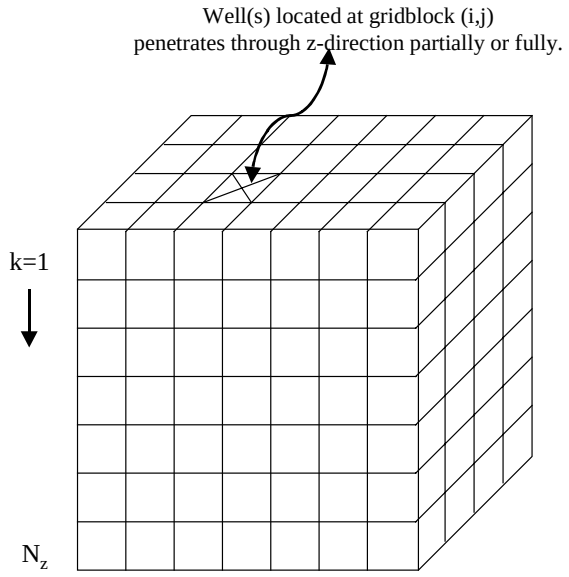


Figure A.1: 3-D schematic of the reservoir.

If one wants to use non-uniform grid sizes in any of the directions, the data has to be read in from the files GRIDX.DAT, GRIDY.DAT, and GRIDZ.DAT for the x, y, and z-directions, respectively. Otherwise, these are not be created. In all examples that we have presented, we have used uniform grids.

A.4 Output Files

Results from the run are contained in several output files. These files are:

CHECK.DAT: This file stores various information such as number of iter-

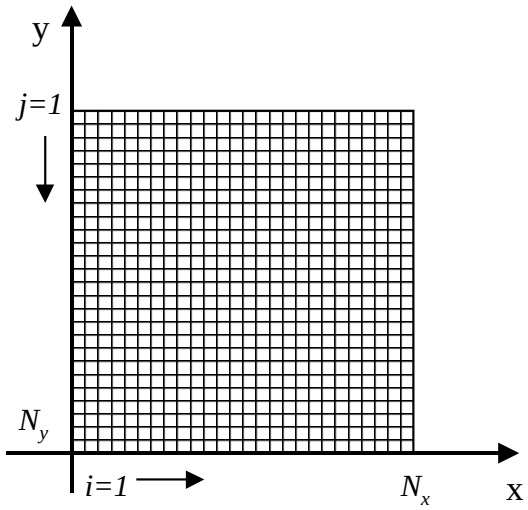


Figure A.2: Areal schematic of the reservoir.

ation, value of the objective function, etc.

KREAL.DAT: This file stores realizations (or the maximum a posteriori estimate) of the log-permeability field after conditioning to both pressure data and hard data for porosity. (It is not necessary to use any hard data).

PREAL.DAT: This file stores realizations (or maximum a posteriori estimate) of porosity field after conditioning to both pressure data and hard data for porosity. (It is not necessary to use any hard data).

SREAL.DAT: This file stores skin factors after conditioning to both pressure data and hard data for porosity. (It is not necessary to use any hard data).

PITER.DAT: This file stores times versus observed and calculated pressure responses at each iteration of the Gauss-Newton procedure.

PFINAL.DAT: This file stores times versus observed and calculated pressure responses at convergence of the Gauss-Newton procedure.

HITER.DAT: This file stores the location (gridblock number) versus observed and calculated porosity values at the hard data locations at each iteration of

the Gauss-Newton procedure (if IHARD=1). This file is used only for testing of the code.

UCOR.DAT: This file stores the corrections to the prior means for porosity and log-permeability in the case IMEAN=1.

APPENDIX B

ADJOINT METHOD

B.1 Flow Equations

Here, we consider three-dimensional single phase flow of a slightly compressible fluid of constant viscosity and constant compressibility. We use an x-y-z Cartesian coordinate system. Neglecting the gravity effects, the governing flow equation can be written in oil field units as

$$\frac{C_1}{\mu}\nabla \cdot \Big([k]\nabla p(x,y,z,t)\Big) = \frac{\phi c_t}{C_2}\frac{\partial p}{\partial t} + \hat{q}_w(x,y,z,t), \tag{B.1}$$

where $C_1 = 1.127 \times 10^{-3}$ and $C_2 = 5.615$ and $[k]$ denotes the permeability tensor. We assume that the principal axes of permeability coincide with the directions of the coordinate system so that

$$[k] = \begin{bmatrix} k_x & 0 & 0 \\ 0 & k_y & 0 \\ 0 & 0 & k_z \end{bmatrix}. \tag{B.2}$$

The term $\hat{q}_w(x,y,z,t)$ is the source or sink term at time t at well w in units of RB/ft^3-day and for production at well w, $\hat{q}_w$ is positive; while for injection, $\hat{q}_w$ is negative and is nonzero only if the point (x,y,z) is intersected by a well. The reservoir model is assumed to be a rectangular parallelepiped, i.e., Eq. B.1 applies for all $t > 0$ on

$$\Omega = \Big[(x,y,z) \mid 0 < x < L_x, 0 < y < L_y, 0 < z < L_z\Big], \tag{B.3}$$

where the boundary of Ω is denoted by $\partial\Omega$. Assuming a uniform initial pressure, p_i, the initial conditions are given by

$$p(x,y,z,0) = p_i, \tag{B.4}$$

for (x, y, z) in Ω. We also assume no flow boundary conditions at outer edges of the reservoir.

We partition Ω into gridblocks using a standard block-centered grid and let (x_i, y_j, z_k) for $i = 1, 2, \ldots, N_x$, $j = 1, 2, \ldots, N_y$ and $k = 1, 2, \ldots, N_z$ denote the gridblock centers. Using a purely implicit finite-difference scheme, Eq. B.1 can be differenced. Multiplying the resulting equation by $\Delta x_i \Delta y_j \Delta z_k$, where Δx_i, Δy_j and Δz_k are the dimensions of the gridblock centered at (x_i, y_j, z_k) gives

$$T_{z,i,j,k-1/2}p^n_{i,j,k-1} + T_{y,i,j-1/2,k}p^n_{i,j-1,k} + T_{x,i-1/2,j,k}p^n_{i-1,j,k} - (T_{i,j,k} + V^n_{i,j,k})p^n_{i,j,k} + T_{x,i+1/2,j,k}p^n_{i+1,j,k} + T_{y,i,j+1/2,k}p^n_{i,j+1,k} + T_{z,i,j,k+1/2}p^n_{i,j,k+1} - q^n_{i,j,k} = -V^n_{i,j,k}p^{n-1}_{i,j,k}, \tag{B.5}$$

for $i = 1, 2, \ldots, N_x$, $j = 1, 2, \ldots, N_y$ and $k = 1, 2, \ldots, N_z$. Note that in Eq. B.5 $q^n_{i,j,k}$ represents the internal sink or source term at time t^n in gridblock (i, j, k) and has unit of RB/D. The transmissibilities, denoted by T's in Eq. B.5, are defined as

$$T_{i,j,k} = T_{x,i+1/2,j,k} + T_{x,i-1/2,j,k} + T_{y,i,j+1/2,k} + T_{y,i,j-1/2,k} + T_{z,i,j,k+1/2} + T_{z,i,j,k-1/2}, \tag{B.6}$$

$$T_{x,i+1/2,j,k} = \frac{C_1 \Delta y_j \Delta z_k k_{x,i+1/2,j,k}}{\mu(x_{i+1} - x_i)}, \tag{B.7}$$

for $i = 1, 2, \ldots, N_x - 1$ and all j and k,

$$T_{x,1/2,j,k} = T_{x,N_x+1/2,j,k} = 0, \tag{B.8}$$

for all j and k;

$$T_{y,i,j+1/2,k} = \frac{C_1 \Delta x_j \Delta z_k k_{y,i,j+1/2,k}}{\mu(y_{j+1} - y_j)}, \tag{B.9}$$

for $j = 1, 2, \ldots, N_y - 1$ and all i and k,

$$T_{y,i,1/2,k} = T_{y,i,N_y+1/2,k} = 0, \tag{B.10}$$

for all i and k;

$$T_{z,i,j,k+1/2} = \frac{C_1 \Delta x_i \Delta y_j k_{z,i,j,k+1/2}}{\mu(z_{k+1} - z_k)}, \tag{B.11}$$

for $k = 1, 2, \ldots, N_z - 1$ and all i and j,

$$T_{z,i,j,1/2} = T_{z,i,j,N_z+1/2} = 0, \tag{B.12}$$

for all i and j. Permeabilities at gridblock interfaces are computed as harmonic averages.

In Eq. B.5, $V^n_{i,j,k}$ is defined by

$$V^n_{i,j,k} = \frac{\phi_{i,j,k} c_t \Delta x_i \Delta y_j \Delta z_k}{C_2 \Delta t^n}. \tag{B.13}$$

The relation between a gridblock source or sink term, the gridblock pressure and flowing bottom hole pressure is specified by applying Peaceman's equation [44, 45], i.e.,

$$q^n_{i,j,k} = WI_{i,j,k}(p^n_{i,j,k} - p^n_{wf,i,j}), \tag{B.14}$$

where $WI_{i,j,k}$, the well index term, is given by

$$WI_{i,j,k} = \frac{2\pi C_1 \Delta z_k \sqrt{k_{x,i,j,k} k_{y,i,j,k}}}{\mu[\ln(r_{o,i,j,k}/r_{w,i,j}) + s_{i,j,k}]}. \tag{B.15}$$

Here, $r_{w,i,j}$ is the wellbore radius of the well k and $s_{i,j,k}$ is the skin factor at the well for model layer k and

$$r_{o,i,j,k} = \frac{0.28073 \Delta x_i \sqrt{1 + \frac{k_{x,i,j,k} \Delta y_j^2}{k_{y,i,j,k} \Delta x_i^2}}}{1 + \sqrt{k_{x,i,j,k}/k_{y,i,j,k}}}. \tag{B.16}$$

The individual gridblock rates must sum to the total rate, i.e.

$$q^n_{i,j} = \sum_{k=l1}^{l2} q^n_{i,j,k} = \sum_{k=l1}^{l2} WI_{i,j,k}(p^n_{i,j,k} - p^n_{wf,i,j}), \tag{B.17}$$

where the sum is over all gridblocks penetrated by the well.

Substituting Eq. B.14 into Eq. B.5 gives

$$T_{z,i,j,k-1/2}p^n_{i,j,k-1} + T_{y,i,j-1/2,k}p^n_{i,j-1,k} + T_{x,i-1/2,j,k}p^n_{i-1,j,k} - (T_{i,j,k} + WI_{i,j,k} + V^n_{i,j,k})p^n_{i,j,k} + T_{x,i+1/2,j,k}p^n_{i+1,j,k} + T_{y,i,j+1/2,k}p^n_{i,j+1,k} + T_{z,i,j,k+1/2}p^n_{i,j,k+1} + WI_{i,j,k}p^n_{wf,i,j} = -V^n_{i,j,k}p^{n-1}_{i,j,k}, \quad \text{(B.18)}$$

for $i = 1, 2, \ldots, N_x$, $j = 1, 2, \ldots, N_y$ and $k = 1, 2, \ldots, N_z$. Eq. B.17 can be rewritten as

$$\sum_{k=l1}^{l2} WI_{i,j,k}p^n_{i,j,k} - \left(\sum_{k=l1}^{l2} WI_{i,j,k}\right) p^n_{wf,i,j} = q^n_{i,j}. \quad \text{(B.19)}$$

Assuming flow rates are specified at N_w wells, Eq. B.19 is applied at each well, i.e., Eq. B.19 represents N_w equations. Thus Eqs. B.18 and B.19 represent $N + N_w$ equations in $N + N_w$ unknowns. The unknowns are the N gridblock pressures and the N_w wellbore pressures. Combining Eqs. B.18 and B.19, we obtain the following matrix equation for the flow problem with well rates specified:

$$\begin{bmatrix} & & \\ A_{1,1} & & A_{1,2} \\ & & \\ & & \\ A_{2,1} & & A_{2,2} \\ & & \end{bmatrix} \begin{bmatrix} p^n_1 \\ \vdots \\ p^n_N \\ p^n_{wf,1} \\ \vdots \\ p^n_{wf,N_w} \end{bmatrix} = - \begin{bmatrix} & & \\ V & & O \\ & & \\ & & \\ O & & O \\ & & \end{bmatrix} \begin{bmatrix} p^{n-1}_1 \\ \vdots \\ p^{n-1}_N \\ p^{n-1}_{wf,1} \\ \vdots \\ p^{n-1}_{wf,N_w} \end{bmatrix} + \begin{bmatrix} 0 \\ \vdots \\ 0 \\ q_1 \\ \vdots \\ q_{N_w} \end{bmatrix}, \quad \text{(B.20)}$$

where the gridblock pressures are ordered in a one-dimensional array as follows. We first start with first gridblock in the z-direction (i.e., $k = 1$) and with first gridblock in the y-direction (i.e., $j = 1$), then the x-direction index increases from 1 to N_x. The same ordering of the x-direction index is repeated for $j = 2, 3, ..., N_y$. Then, the same ordering of the x-direction and the y-direction indices is repeated for $k = 2, 3, ..., N_z$. In Eq. B.20, $A_{1,1}$ is an $N \times N$ dimensional 7-band matrix with diagonal entries given by terms $-\left(T_{i,j,k} + WI_{i,j,k} + V^n_{i,j,k}\right)$ and the other 6 nonzero diagonals given by

gridblock transmissibilities; $A_{1,2}$ is an $N \times N_w$ matrix whose entries are either zero or equal to a well index term. Each column of $A_{1,2}$ which corresponds to a particular well has non-zero entries equal to well index terms, $WI_{i,j,k}$, at gridblocks corresponding to that particular well; $A_{2,1}$ is an $N_w \times N$ matrix which is equal to $A_{1,2}^T$; $A_{2,2}$ is an $N_w \times N_w$ diagonal matrix with a particular diagonal entry corresponding to negative of the sum of well index terms corresponding to a particular well, i.e. $-\sum_{k=l1}^{l2} WI_{i,j,k}$; V is an $N \times N$ diagonal matrix with diagonal entries equal to $V_{i,j,k}^n$ terms and O's are null matrices. In general matrix notations, Eq. B.20 can be written as

$$A_n p^n = -D_n p^{n-1} + Q, \tag{B.21}$$

where p^n and p^{n-1} are the vectors of pressures in all gridblocks and wellbores at the nth and $(n-1)$th timesteps, respectively. A_n and D_n are matrices of flow and accumulation coefficients, i.e.

$$A_n = \begin{bmatrix} A_{1,1} & A_{1,2} \\ A_{2,1} & A_{2,2} \end{bmatrix}, \tag{B.22}$$

and

$$D_n = \begin{bmatrix} V & O \\ O & O \end{bmatrix}. \tag{B.23}$$

Q contains source and sink terms. The matrix A_n can be written as

$$A_n = B_n - D_n, \tag{B.24}$$

where B_n is the matrix obtained from replacing the diagonal entries of A_n which are $-\left(T_{i,j,k} + WI_{i,j,k} + V_{i,j,k}^n\right)$ terms by the terms $-\left(T_{i,j,k} + WI_{i,j,k}\right)$. Thus, Eq. B.21 can also be written as

$$B_n p^n = D_n(p^n - p^{n-1}) + Q. \tag{B.25}$$

B.2 Adjoint Equations

Assume we wish to compute the gradient of some real-valued function $\hat{g}(p)$ with respect to a vector m of model parameters. We need one Lagrange multiplier or adjoint variable λ for each pressure. Thus, we let λ_s be an $(N + N_w)$-dimensional column vector, i.e.,

$$\lambda_s^n = [\lambda_{s,1}^n, \lambda_{s,2}^n, ..., \lambda_{s,N}^n, \lambda_{s,N+1}^n, \lambda_{s,N+2}^n, ..., \lambda_{s,N+N_w}^n]^T, \tag{B.26}$$

and adjoin Eq. B.21 to the function $\hat{g}$ to obtain the functional J given by

$$J = \hat{g} + \sum_{n=1}^{L} (\lambda_s^n)^T [A_n p^n + D_n p^{n-1} - Q]. \tag{B.27}$$

It is obvious that the terms in the sum are zero for any values of the adjoint variables. This implies that

$$\nabla J = \nabla \hat{g}, \tag{B.28}$$

where jth component of $\nabla \hat{g}$ is the derivative of $\hat{g}$ with respect to the jth model parameter, i.e., sensitivity coefficient. Considering J as a function of k_x, k_y, k_z and ϕ, its total differential can be written as

$$\delta J = (\nabla_{k_x} J)^T \, \delta k_x + \left(\nabla_{k_y} J\right)^T \delta k_y + (\nabla_{k_z} J)^T \, \delta k_z + (\nabla_{\phi} J)^T \, \delta \phi, \tag{B.29}$$

where δ denotes the total differential operator and k_x, k_y, k_z and ϕ are respectively the vectors of gridblock permeabilities and porosities, i.e.,

$$k_x = [k_{x,1}, k_{x,2}, \ldots, k_{x,N}]^T, \tag{B.30}$$

$$k_y = [k_{y,1}, k_{y,2}, \ldots, k_{y,N}]^T, \tag{B.31}$$

$$k_z = [k_{z,1}, k_{z,2}, \ldots, k_{z,N}]^T, \tag{B.32}$$

$$\phi = [\phi_1, \phi_2, \ldots, \phi_N]^T, \tag{B.33}$$

and their differentials are given by

$$\delta k_x = [\delta k_{x,1}, \delta k_{x,2}, \ldots, \delta k_{x,N}]^T, \tag{B.34}$$

$$\delta k_y = [\delta k_{y,1}, \delta k_{y,2}, \ldots, \delta k_{y,N}]^T, \tag{B.35}$$

$$\delta k_z = [\delta k_{z,1}, \delta k_{z,2}, \ldots, \delta k_{z,N}]^T, \tag{B.36}$$

$$\delta \phi = [\delta \phi_1, \delta \phi_2, \ldots, \delta \phi_N]^T. \tag{B.37}$$

In Eq. B.29 the terms $(\nabla_{k_x} J)^T$, $\left(\nabla_{k_y} J\right)^T$, $(\nabla_{k_z} J)^T$ and $(\nabla_\phi J)^T$ give the sensitivity coefficients for the functional. Taking the total differential of Eq. B.27 using standard operational formulas and notation, we obtain

$$\begin{aligned}\delta J = \delta \hat{g} + \sum_{n=1}^{L} (\lambda_s^n)^T \Big[& A_n \, \delta p^n + \left(\nabla_{k_x} (A_n p^n)^T\right)^T \delta k_x \\ & + \left(\nabla_{k_y} (A_n p^n)^T\right)^T \delta k_y + \left(\nabla_{k_z} (A_n p^n)^T\right)^T \delta k_z \\ & + \left(\nabla_\phi (A_n p^n)^T\right)^T \delta \phi + D_n \delta p^{n-1} + \left(\nabla_\phi (D_n \, p^{n-1})^T\right)^T \delta \phi \Big].\end{aligned} \tag{B.38}$$

For the problems considered here the expression for $\hat{g}$ will involve only pressures so the total differential of $\hat{g}$ can be written as

$$\delta \hat{g} = \sum_{n=1}^{L} (\nabla_{p^n} \hat{g})^T \, \delta p^n, \tag{B.39}$$

and δp^n denotes differential of p^n. By changing the index of summation, we have

$$\sum_{n=1}^{L} (\lambda_s^n)^T \, D_n \delta p^{n-1} = \sum_{n=1}^{L-1} \left(\lambda_s^{n+1}\right)^T D_{n+1} \delta p^n. \tag{B.40}$$

Note that since p^0 represents fixed specified initial condition, its differential is zero, i.e. $\delta p^0 = 0$. Since the entries of the matrix B_n do not depend on ϕ, it follows that

$$\nabla_\phi \, (A_n \, p^n)^T = \nabla_\phi \, ((B_n - D_n) \, p^n)^T = -\nabla_\phi \, (D_n \, p^n)^T. \tag{B.41}$$

Using Eqs. B.40 and B.41 and making rearrangements, Eq. B.38 can be written as

$$\delta J = \sum_{n=1}^{L} \left[(\lambda_s^n)^T A_n + \left(\lambda_s^{n+1}\right)^T D_{n+1} + (\nabla_{p^n} \hat{g})^T \right] \delta p^n + \sum_{n=1}^{L} (\lambda_s^n)^T \left(\nabla_{k_x} (A_n p^n)^T \right)^T \delta k_x + \sum_{n=1}^{L} (\lambda_s^n)^T \left(\nabla_{k_y} (A_n p^n)^T \right)^T \delta k_y + \sum_{n=1}^{L} (\lambda_s^n)^T \left(\nabla_{k_z} (A_n p^n)^T \right)^T \delta k_z - \sum_{n=1}^{L} (\lambda_s^n)^T \left(\nabla_\phi \left(D_n (p^n - p^{n-1}) \right)^T \right)^T \delta \phi. \quad \text{(B.42)}$$

In Eq. B.42, choosing the adjoint variables to insure that the coefficients multiplying δp^n vanish, we obtain

$$(\lambda_s^n)^T A_n + \left(\lambda_s^{n+1}\right)^T D_{n+1} + (\nabla_{p^n} \hat{g})^T = 0, \quad \text{(B.43)}$$

for $n = 1, 2, \ldots, L-1$. Taking the transpose in Eq. B.43 and rearranging the resulting equation gives the discrete system of adjoint equations

$$A_n \lambda_s^n + D_{n+1} \lambda_s^{n+1} = -\nabla_{p^n} \hat{g}, \quad \text{(B.44)}$$

for $n = 1, 2, \ldots, L-1$. The additional constraint is given by

$$\lambda_s^L = 0. \quad \text{(B.45)}$$

Eq. B.44 is solved for the adjoint variables backward in time by applying Eq. B.45 as a constraint. Once we solve for adjoint variables, sensitivity of any function $\hat{g}$ with respect to model parameters are calculated using the following equations obtained by comparing Eqs. B.29 and B.42,

$$\nabla_{k_x} J = \sum_{n=1}^{L} \nabla_{k_x} \left[A_n p^n \right]^T \lambda_s^n, \quad \text{(B.46)}$$

$$\nabla_{k_y} J = \sum_{n=1}^{L} \nabla_{k_y} \left[A_n p^n \right]^T \lambda_s^n, \quad \text{(B.47)}$$

$$\nabla_{k_z} J = \sum_{n=1}^{L} \nabla_{k_z} \left[A_n p^n\right]^T \lambda_s^n, \tag{B.48}$$

and

$$\nabla_{\phi} J = -\sum_{n=1}^{L} \nabla_{\phi} \left[D_n (p^n - p^{n-1})\right]^T \lambda_s^n. \tag{B.49}$$

If the permeability is isotropic, i.e. $k_x = k_y = k_z = k$ then we simply use

$$\nabla_{k} J = \sum_{n=1}^{L} \nabla_{k} \left[A_n p^n\right]^T \lambda_s^n. \tag{B.50}$$

B.3 Calculation of Subspace Vectors

We wish to compute gradients of partial data objective functions. Recall that (section 3.3.1) data objective function is partitioned as

$$O_{\mathrm{D}}(m) = \sum_{k} O_{\mathrm{D}}^{k}(m), \tag{B.51}$$

where

$$O_{\mathrm{D}}^{k}(m) = \frac{1}{2}\big(g^k(m) - d_{uc}^k\big)^{\mathrm{T}} [C_{\mathrm{D}}^k]^{-1} \big(g^k(m) - d_{uc}^k\big). \tag{B.52}$$

For the problems considered here, C_{D}^k is a diagonal matrix with all diagonal entries equal to $\sigma_{\mathrm{d},k}^2$; $g^k(m)$ contains the kth set of calculated wellbore pressures obtained from the model using the reservoir simulator and d_{uc}^k is the kth set of unconditional simulation of the observed wellbore pressure data. Eq. B.52 can be written as

$$O_{\mathrm{D}}^{k} = \frac{1}{2} \sum_{l=1}^{N_k} \frac{\big(g_l^k(m) - d_{uc,l}^k\big)^2}{\sigma_{\mathrm{d},k,l}^2}. \tag{B.53}$$

For the particular problem of computing gradients of partial data objective functions, we choose

$$\hat{g} = \frac{1}{2} \sum_{l=1}^{N_k} \frac{\big(g_l^k(m) - d_{uc,l}^k\big)^2}{\sigma_{\mathrm{d},k,l}^2}. \tag{B.54}$$

To compute the gradient (i.e. sensitivity) of $\hat{g}$ with respect to model parameters (a subspace vector) we first need to compute the source term on the right-hand side of Eq. B.44, i.e. $-\nabla_{p^n}\hat{g}$, then solve the adjoint equations. So

$$\nabla_{p^n}\hat{g} = \nabla_{p^n}\left[\frac{1}{2}\sum_{l=1}^{N_k}\frac{\left(g_l^k(m) - d_{uc,l}^k\right)^2}{\sigma_{\mathrm{d},k,l}^2}\right]. \tag{B.55}$$

The first N entries of $\nabla_{p^n}\hat{g}$ are equal to zero since $\hat{g}$ is independent of gridblock pressures. The remaining N_w entries of $\nabla_{p^n}\hat{g}$ which will be denoted by $\nabla_{p_w^n}\hat{g}$ (where p_w^n is the N_w-dimensional vector which contains wellbore pressures at the wells at nth timestep) are zero except when $t^n = t^l$. If $t^n = t^l$, then

$$\nabla_{p_w^l}\hat{g} = e_w\frac{\left(g_l^k(m) - d_{uc,l}^k\right)}{\sigma_{\mathrm{d},k,l}^2}, \tag{B.56}$$

where e_w is an N_w-dimensional vector with the entry corresponding to the location of wellbore pressure in the vector p_w^l equal to one and all other entries zero.

As a final comment we should note that for every subspace vector based on the gradient of a sub-objective function, there is one adjoint system to solve.

APPENDIX C

SENSITIVITY TO COEFFICIENTS OF SUBSPACE VECTORS

C.1 The Product $G_l A_l$

Before starting to derive the equations for computing the product $G_l A_l$ using the gradient simulator method, it might be beneficial to show the product $G_l A_l$. Recall that G_l is the sensitivity coefficients matrix evaluated at m^l, i.e.

$$G = \left(\nabla_m g^T\right)^T = \begin{bmatrix} \frac{\partial g_1}{\partial m_1} & \frac{\partial g_1}{\partial m_2} & \cdots & \frac{\partial g_1}{\partial m_{N_M}} \\ \frac{\partial g_2}{\partial m_1} & \frac{\partial g_2}{\partial m_2} & \cdots & \frac{\partial g_2}{\partial m_{N_M}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial g_{N_D}}{\partial m_1} & \frac{\partial g_{N_D}}{\partial m_2} & \cdots & \frac{\partial g_{N_D}}{\partial m_{N_M}} \end{bmatrix}. \tag{C.1}$$

Note that in Eq. C.1 we have deleted the iteration index to simplify the equations.

We want to calculate sensitivities with respect to some subspace vectors that are linear combinations of the gridblock model parameters, i.e.

$$m^l = m^{l-1} + A\alpha, \tag{C.2}$$

or

$$m^l = m^{l-1} + \sum_{j=1}^{N_B} a^j \alpha_j. \tag{C.3}$$

The kth entry of Eq. C.3 can be written as

$$m_k^l = m_k^{l-1} + \sum_{j=1}^{N_B} a_k^j \alpha_j. \tag{C.4}$$

Differentiating Eq. C.4 gives

$$a_k^j = \frac{\partial m_k}{\partial \alpha_j}, \tag{C.5}$$

so

$$A = \begin{bmatrix} \frac{\partial m_1}{\partial \alpha_1} & \frac{\partial m_1}{\partial \alpha_2} & \cdots & \frac{\partial m_1}{\partial \alpha_{N_B}} \\ \frac{\partial m_2}{\partial \alpha_1} & \frac{\partial m_2}{\partial \alpha_2} & \cdots & \frac{\partial m_2}{\partial \alpha_{N_B}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial m_{N_M}}{\partial \alpha_1} & \frac{\partial m_{N_M}}{\partial \alpha_2} & \cdots & \frac{\partial m_{N_M}}{\partial \alpha_{N_B}} \end{bmatrix}. \tag{C.6}$$

Note that in Eqs. C.5 and C.6 we have again deleted the iteration index to simplify the equations. Multiplying Eq. C.1 with Eq. C.6 gives

$$GA = \begin{bmatrix} \frac{\partial g_1}{\partial m_1}\frac{\partial m_1}{\partial \alpha_1} + \ldots + \frac{\partial g_1}{\partial m_{N_M}}\frac{\partial m_{N_M}}{\partial \alpha_1} & \cdots & \frac{\partial g_1}{\partial m_1}\frac{\partial m_1}{\partial \alpha_{N_B}} + \ldots + \frac{\partial g_1}{\partial m_{N_M}}\frac{\partial m_{N_M}}{\partial \alpha_{N_B}} \\ \frac{\partial g_2}{\partial m_1}\frac{\partial m_1}{\partial \alpha_1} + \ldots + \frac{\partial g_2}{\partial m_{N_M}}\frac{\partial m_{N_M}}{\partial \alpha_1} & \cdots & \frac{\partial g_2}{\partial m_1}\frac{\partial m_1}{\partial \alpha_{N_B}} + \ldots + \frac{\partial g_2}{\partial m_{N_M}}\frac{\partial m_{N_M}}{\partial \alpha_{N_B}} \\ \vdots & \ddots & \vdots \\ \frac{\partial g_{N_D}}{\partial m_1}\frac{\partial m_1}{\partial \alpha_1} + \ldots + \frac{\partial g_{N_D}}{\partial m_{N_M}}\frac{\partial m_{N_M}}{\partial \alpha_1} & \cdots & \frac{\partial g_{N_D}}{\partial m_1}\frac{\partial m_1}{\partial \alpha_{N_B}} + \ldots + \frac{\partial g_{N_D}}{\partial m_{N_M}}\frac{\partial m_{N_M}}{\partial \alpha_{N_B}} \end{bmatrix}. \tag{C.7}$$

Each entry of matrix of Eq. C.7 represents the chain rule, so Eq. C.7 can be rewritten as

$$GA = \left(\nabla_\alpha g^T\right)^T = \begin{bmatrix} \frac{\partial g_1}{\partial \alpha_1} & \frac{\partial g_1}{\partial \alpha_2} & \cdots & \frac{\partial g_1}{\partial \alpha_{N_B}} \\ \frac{\partial g_2}{\partial \alpha_1} & \frac{\partial g_2}{\partial \alpha_2} & \cdots & \frac{\partial g_2}{\partial \alpha_{N_B}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial g_{N_D}}{\partial \alpha_1} & \frac{\partial g_{N_D}}{\partial \alpha_2} & \cdots & \frac{\partial g_{N_D}}{\partial \alpha_{N_B}} \end{bmatrix}, \tag{C.8}$$

which is simply the sensitivity of the data to the coefficients of the subspace vectors.

C.2 Computation of $G_l A_l$

Again, we begin with the simulator equation (Eq. B.21) which is repeated here as

$$A_n p^n = -D_n p^{n-1} - Q. \tag{C.9}$$

Taking the total differential of Eq. C.9, we obtain

$$\delta\left(A_n p^n\right) = -\delta\left(D_n p^{n-1}\right) + \delta Q, \tag{C.10}$$

where δ denotes the total differential operator. We consider the case that flow rates are specified, so $\delta Q = 0$ in Eq. C.10. We let $A_{n,i}$ denote the ith row of the matrix A_n so

$$A_n = \begin{bmatrix} A_{n,1} \\ A_{n,2} \\ \vdots \\ A_{n,i} \\ \vdots \\ A_{n,N_T} \end{bmatrix}, \tag{C.11}$$

and

$$A_n p^n = \begin{bmatrix} A_{n,1}p^n \\ A_{n,2}p^n \\ \vdots \\ A_{n,i}p^n \\ \vdots \\ A_{n,N_T}p^n \end{bmatrix}, \tag{C.12}$$

where $N_T = N + N_w$. Thus, $\delta\left(A_n p^n\right)$ can be written as

$$\delta\left(A_n p^n\right) = \begin{bmatrix} \delta\left(A_{n,1}p^n\right) \\ \delta\left(A_{n,2}p^n\right) \\ \vdots \\ \delta\left(A_{n,i}p^n\right) \\ \vdots \\ \delta\left(A_{n,N_T}p^n\right) \end{bmatrix} = \begin{bmatrix} \sum_{j=1}^{N_M} \frac{\partial\left(A_{n,1}p^n\right)}{\partial m_j}\delta m_j \\ \sum_{j=1}^{N_M} \frac{\partial\left(A_{n,2}p^n\right)}{\partial m_j}\delta m_j \\ \vdots \\ \sum_{j=1}^{N_M} \frac{\partial\left(A_{n,i}p^n\right)}{\partial m_j}\delta m_j \\ \vdots \\ \sum_{j=1}^{N_M} \frac{\partial\left(A_{n,N_T}p^n\right)}{\partial m_j}\delta m_j \end{bmatrix}, \tag{C.13}$$

where the last equality of Eq. C.13 is obtained from the relationship between the total differential and partial derivatives of the model parameters.

Since each $A_{n,i}p^n$ in Eq. C.12 is simply a number,

$$\left(A_n p^n\right)^T = \begin{bmatrix} A_{n,1}p^n & \cdots & A_{n,i}p^n & \cdots & A_{n,N_T}p^n \end{bmatrix}. \tag{C.14}$$

Using basic vector calculus, the gradient of the vector $(A_n p^n)^T$ is obtained as

$$\nabla_m \left(A_n p^n\right)^T = \begin{bmatrix} \frac{\partial (A_{n,1} p^n)}{\partial m_1} & \frac{\partial (A_{n,2} p^n)}{\partial m_1} & \cdots & \frac{\partial (A_{n,N_T} p^n)}{\partial m_1} \\ \frac{\partial (A_{n,1} p^n)}{\partial m_2} & \frac{\partial (A_{n,2} p^n)}{\partial m_2} & \cdots & \frac{\partial (A_{n,N_T} p^n)}{\partial m_2} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial (A_{n,1} p^n)}{\partial m_{N_M}} & \frac{\partial (A_{n,2} p^n)}{\partial m_{N_M}} & \cdots & \frac{\partial (A_{n,N_T} p^n)}{\partial m_{N_M}} \end{bmatrix}, \tag{C.15}$$

where ∇_m represents the gradient operator with respect to the model and thus involves partial derivatives with respect to all parameters. Taking the transpose of Eq. C.15 and multiplying by the model perturbation vector, it follows that

$$\left(\nabla_m \left(A_n p^n\right)^T\right)^T \delta m = \begin{bmatrix} \sum_{j=1}^{N_M} \frac{\partial (A_{n,1} p^n)}{\partial m_j} \delta m_j \\ \sum_{j=1}^{N_M} \frac{\partial (A_{n,2} p^n)}{\partial m_j} \delta m_j \\ \vdots \\ \sum_{j=1}^{N_M} \frac{\partial (A_{n,i} p^n)}{\partial m_j} \delta m_j \\ \vdots \\ \sum_{j=1}^{N_M} \frac{\partial \left(A_{n,N_T} p^n\right)}{\partial m_j} \delta m_j \end{bmatrix}. \tag{C.16}$$

Comparing Eq. C.16 with Eq. C.13, we obtain the following equality

$$\delta \left(A_n p^n\right) = \left(\nabla_m \left(A_n p^n\right)^T\right)^T \delta m. \tag{C.17}$$

Expanding the derivatives in Eq. C.15 gives

$$\nabla_m \left(A_n p^n\right)^T = \begin{bmatrix} A_{n,1} \frac{\partial p^n}{\partial m_1} & A_{n,2} \frac{\partial p^n}{\partial m_1} & \cdots & A_{n,N_T} \frac{\partial p^n}{\partial m_1} \\ A_{n,1} \frac{\partial p^n}{\partial m_2} & A_{n,2} \frac{\partial p^n}{\partial m_2} & \cdots & A_{n,N_T} \frac{\partial p^n}{\partial m_2} \\ \vdots & \vdots & \ddots & \vdots \\ A_{n,1} \frac{\partial p^n}{\partial m_{N_M}} & A_{n,2} \frac{\partial p^n}{\partial m_{N_M}} & \cdots & A_{n,N_T} \frac{\partial p^n}{\partial m_{N_M}} \end{bmatrix} + \begin{bmatrix} \frac{\partial A_{n,1}}{\partial m_1} p^n & \frac{\partial A_{n,2}}{\partial m_1} p^n & \cdots & \frac{\partial A_{n,N_T}}{\partial m_1} p^n \\ \frac{\partial A_{n,1}}{\partial m_2} p^n & \frac{\partial A_{n,2}}{\partial m_2} p^n & \cdots & \frac{\partial A_{n,N_T}}{\partial m_2} p^n \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial A_{n,1}}{\partial m_{N_M}} p^n & \frac{\partial A_{n,2}}{\partial m_{N_M}} p^n & \cdots & \frac{\partial A_{n,N_T}}{\partial m_{N_M}} p^n \end{bmatrix}. \tag{C.18}$$

From Eqs. C.11, C.17 and C.18, it follows that

$$\delta\left(A_n p^n\right) = \left(\nabla_m \left(A_n p^n\right)^T\right)^T \delta m = \begin{bmatrix} A_{n,1} \\ A_{n,2} \\ \vdots \\ A_{n,N_T} \end{bmatrix} \begin{bmatrix} \frac{\partial p^n}{\partial m_1} & \frac{\partial p^n}{\partial m_2} & \cdots & \frac{\partial p^n}{\partial m_{N_M}} \end{bmatrix} \delta m + \begin{bmatrix} \frac{\partial A_n}{\partial m_1} p^n & \frac{\partial A_n}{\partial m_2} p^n & \cdots & \frac{\partial A_n}{\partial m_{N_M}} p^n \end{bmatrix} \delta m. \quad \text{(C.19)}$$

Applying the same arguments used to derive Eqs. C.17 and C.19, it is easy to show that

$$\delta p^n = \left(\nabla_m \left(p^n\right)^T\right)^T \delta m = \begin{bmatrix} \frac{\partial p^n}{\partial m_1} & \frac{\partial p^n}{\partial m_2} & \cdots & \frac{\partial p^n}{\partial m_{N_M}} \end{bmatrix} \delta m. \quad \text{(C.20)}$$

Using Eqs. C.11 and C.20, Eq. C.19 can be rewritten as

$$\delta\left(A_n p^n\right) = \left(\nabla_m \left(A_n p^n\right)^T\right)^T \delta m = A_n \delta p^n + \begin{bmatrix} \frac{\partial A_n}{\partial m_1} p^n & \frac{\partial A_n}{\partial m_2} p^n & \cdots & \frac{\partial A_n}{\partial m_{N_M}} p^n \end{bmatrix} \delta m. \quad \text{(C.21)}$$

Defining F_n by

$$F_n = \begin{bmatrix} \frac{\partial A_n}{\partial m_1} p^n & \frac{\partial A_n}{\partial m_2} p^n & \cdots & \frac{\partial A_n}{\partial m_{N_M}} p^n \end{bmatrix}, \quad \text{(C.22)}$$

Eq. C.21 can be rewritten as

$$\delta\left(A_n p^n\right) = \left(\nabla_m \left(A_n p^n\right)^T\right)^T \delta m = A_n \delta p^n + F_n \delta m. \quad \text{(C.23)}$$

The same basic derivation used to derive Eq. C.23 can be applied to show that

$$\delta\left(D_n p^{n-1}\right) = D_n \delta p^{n-1} + E_n \delta m, \quad \text{(C.24)}$$

where

$$E_n = \begin{bmatrix} \frac{\partial D_n}{\partial m_1} p^{n-1} & \frac{\partial D_n}{\partial m_2} p^{n-1} & \cdots & \frac{\partial D_n}{\partial m_{N_M}} p^{n-1} \end{bmatrix}. \quad \text{(C.25)}$$

Using Eqs. C.23 and C.24 in Eq. C.10 and rearranging, we obtain

$$A_n \delta p^n = -F_n \delta m - D_n \delta p^{n-1} - E_n \delta m. \tag{C.26}$$

The same derivation of Eq. C.26 was previously given by Chu and Reynolds [9].

Here, we want to calculate sensitivities with respect to some subspace vectors that are linear combinations of the gridblock model parameters, i.e. $\delta m = A\alpha = A\delta\alpha$. Substituting this into Eq. C.26 gives

$$A_n \delta p^n = -F_n A \delta\alpha - D_n \delta p^{n-1} - E_n A \delta\alpha. \tag{C.27}$$

Considering p^n as a function of α's, the following equations which can be shown using the arguments similar to those presented previously relate changes in pressure to changes in subspace coefficients,

$$\delta p^{n-1} = \left[\nabla_\alpha \left(p^{n-1}\right)^T\right]^T \delta\alpha, \tag{C.28}$$

and

$$\delta p^n = \left[\nabla_\alpha \left(p^n\right)^T\right]^T \delta\alpha. \tag{C.29}$$

Using Eq. C.28 in Eq. C.27 gives

$$A_n \delta p^n = -F_n A \delta\alpha - D_n \left[\nabla_\alpha \left(p^{n-1}\right)^T\right]^T \delta\alpha - E_n A \delta\alpha. \tag{C.30}$$

Eq. C.30 can be rearranged to obtain

$$\delta p^n = A_n^{-1} \left[-F_n A - D_n \left[\nabla_\alpha \left(p^{n-1}\right)^T\right]^T - E_n A\right] \delta\alpha. \tag{C.31}$$

Comparing Eq. C.31 with Eq. C.29, it follows that

$$\left[\nabla_\alpha \left(p^n\right)^T\right]^T = A_n^{-1} \left[-F_n A - D_n \left[\nabla_\alpha \left(p^{n-1}\right)^T\right]^T - E_n A\right], \tag{C.32}$$

or

$$A_n \left[\nabla_\alpha \left(p^n\right)^T\right]^T = -F_n A - D_n \left[\nabla_\alpha \left(p^{n-1}\right)^T\right]^T - E_n A. \tag{C.33}$$

Eq. C.33 can be written for each column of A (i.e., a_j) as

$$A_n \begin{bmatrix} \frac{\partial p_1^n}{\partial \alpha_j} \\ \vdots \\ \frac{\partial p_N^n}{\partial \alpha_j} \\ \frac{\partial p_{wf,1}^n}{\partial \alpha_j} \\ \vdots \\ \frac{\partial p_{wf,N_w}^n}{\partial \alpha_j} \end{bmatrix} = -F_n a_j - D_n \begin{bmatrix} \frac{\partial p_1^{n-1}}{\partial \alpha_j} \\ \vdots \\ \frac{\partial p_N^{n-1}}{\partial \alpha_j} \\ \frac{\partial p_{wf,1}^{n-1}}{\partial \alpha_j} \\ \vdots \\ \frac{\partial p_{wf,N_w}^{n-1}}{\partial \alpha_j} \end{bmatrix} - E_n a_j. \tag{C.34}$$

Finally, Eq. C.34 can be solved for $j = 1, 2, \ldots, N_B$ to obtain

$$GA = \begin{bmatrix} \frac{\partial p_{wf,1}}{\partial \alpha_1} & \frac{\partial p_{wf,1}}{\partial \alpha_2} & \cdots & \frac{\partial p_{wf,1}}{\partial \alpha_{N_B}} \\ \frac{\partial p_{wf,2}}{\partial \alpha_1} & \frac{\partial p_{wf,2}}{\partial \alpha_2} & \cdots & \frac{\partial p_{wf,2}}{\partial \alpha_{N_B}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial p_{wf,N_D}}{\partial \alpha_1} & \frac{\partial p_{wf,N_D}}{\partial \alpha_2} & \cdots & \frac{\partial p_{wf,N_D}}{\partial \alpha_{N_B}} \end{bmatrix}. \tag{C.35}$$