T H E   U N I V E R S I T Y   O F   T U L S A

THE GRADUATE SCHOOL

HISTORY MATCHING WITH PARAMETERIZATION BASED ON THE SVD OF A DIMENSIONLESS SENSITIVITY MATRIX

by
Reza Tavakoli

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

2010

THE UNIVERSITY OF TULSA

THE GRADUATE SCHOOL

HISTORY MATCHING WITH PARAMETERIZATION BASED ON THE SVD OF A DIMENSIONLESS SENSITIVITY MATRIX

by
Reza Tavakoli

A DISSERTATION

APPROVED FOR THE DISCIPLINE OF

PETROLEUM ENGINEERING

By Dissertation Committee

______________________, Chairperson
Albert C. Reynolds

____
Gaoming Li

____
Leslie Thompson

____
Peyton Cook

ABSTRACT

Reza Tavakoli (Doctor of Philosophy in Petroleum Engineering)

History Matching With Parameterization Based on the SVD of a Dimensionless Sensitivity Matrix

Directed by Albert C. Reynolds

146 pp., Chapter 5: Conclusions

(323 words)

In this work we develop efficient parameterization algorithms for history matching based on the principal right singular vectors of the dimensionless sensitivity matrix corresponding the maximum a posteriori estimate of reservoir model's parameters. The necessary singular vectors can be computed with the Lanczos algorithm without explicit computation of the sensitivities. We provide a theoretical argument which indicates that this parameterrization provides an optimal basis for parameterization of the vector of the model parameters. We develop and illustrate two gradient-based algorithms based on this parameterization. Like the limited memory Broyden-Fletcher-Goldfarb-Shanno (LBFGS) algorithm, these algorithms avoid explicit computation of individual sensitivity coefficients. For all synthetic problems that we have considered, the reliability, computational efficiency and robustness of the methods presented here are better than those obtained with quasi-Newton methods.

We also implement the SVD parameterization algorithm to generate a suite of conditional realizations in the randomized maximum likelihood (RML) framework to characterize the uncertainty of reservoir performance predictions. We generate multiple realizations simultaneously by minimizing an ensemble of objective functions concurrently using the singular triplets of a particular realization at each iteration. We show that when combining SVD parameterization with the RML method, we can achieve significant additional computational savings compared to the standard implementation of RML using a quasi-Newton method and this algorithm gives good data matches with history-matched models that are consistent with the prior geology. We present two new algorithms based on this idea, one which relies only on updating the SVD parameterization at each iteration and one which combines an inner iteration based on an adjoint gradient where during the inner iteration the truncated SVD parameterization does not vary. Results with our algorithms are superior to those obtained from the ensemble Kalman filter (EnKF) with and without covariance localization. Finally, we show that by combining EnKF with the SVD-algorithm, we can improve the efficiency of the SVD-algorithms and the reliability of EnKF estimates.

ACKNOWLEDGEMENTS

I would like to express my sincere appreciation to Dr. Albert C. Reynolds, Jr., Professor of Petroleum Engineering and Mathematical Science of The University of Tulsa for his support, guidance, insights, and encouragement during my four years Ph.D. study at The University of Tulsa. I especially thank Dr. Gaoming Li, Associate Professor of Petroleum Engineering of The University of Tulsa, Dr. Leslie Thompson, Associate Professor of Petroleum Engineering of The University of Tulsa and Dr. Peyton Cook, Professor of Mathematical Science of The University of Tulsa for serving on my committee and for their helpful suggestions and comments. I would like to extend my thanks to all the other faculty members of the Petroleum Engineering department of The University of Tulsa for their guidance through my courses of study as a graduate student.

My appreciation extends to all the graduate students in the Department of Petroleum Engineering and students in TUPREP (Tulsa University Petroleum Reservoir Exploration Projects), who have helped me a lot in different ways. I greatly appreciate the help of Mrs. Judy Teal and Mrs. Loreta M. Watkins during my study in The University of Tulsa.

This work is dedicated to my parents, my brother, Hossein Tavakoli, and all other members of my family for their unconditional love and support.

TABLE OF CONTENTS

Page

COPYRIGHT iii

ABSTRACT iv

ACKNOWLEDGEMENTS vi

TABLE OF CONTENTS ix

LIST OF TABLES x

LIST OF FIGURES xx

CHAPTER 1: **INTRODUCTION** 1
- 1.1 **Parameterization Algorithms** 2
- 1.2 **Evaluation of Uncertainty** 6
- 1.3 **Research Contributions and Dissertation Outline** 8
  - 1.3.1 *Research Contributions* 8
  - 1.3.2 *Dissertation Outline* 9

CHAPTER 2: **GRADIENT CALCULATION IN IMPES RESERVOIR SIMULATOR WITH ADJOINT CODE AND FORWARD GRADIENT METHODS** 10
- 2.1 **IMPES Simulator** 10
- 2.2 **Adjoint Equations** 17
  - 2.2.1 *G^T Times a Vector* 22
- 2.3 **Forward Gradient Method** 23
  - 2.3.1 *G Times a Vector* 27
- 2.4 **Lanczos Algorithm for computing a partial SVD** 29
  - 2.4.1 *Stopping conditions of Lanczos Algorithm* 33

CHAPTER 3: **MAP ESTIMATE WITH SVD PARAMETERIZATION** 35
- 3.1 **The Posterior Probability Density Function** 35
- 3.2 **Approximate Posterior Covariance Matrix** 37
- 3.3 **Normalized Variance** 38
- 3.4 **Dimensionless Sensitivity Matrix** 40
- 3.5 **Confidence Regions** 41

3.6 **Reduction in Uncertainty in Terms of Singular Values of Dimensionless Sensitivity Matrix** 42
3.7 **Levenberg-Marquardt for SVD Parameterization** 43
3.8 **More Efficient Algorithms Based on SVD parameterization** . 47
3.8.1 *First Modified Algorithm; LBFGS and SVD Parameterization* 48
3.8.2 *Second Modified Algorithm* 50
3.8.3 *Steps of the SVD-Grd Algorithm* 52
3.9 **Convergence Criteria** 55
3.10 **Computational Results** 56
3.10.1 *Example 1* 56
3.10.2 *MAP Estimate Results of Example 1* 58
3.10.3 *LBFGS and LBFGS-SVD Results of Example 1* 62
3.10.4 *Example 2* 65
3.10.5 *MAP Estimate Results of Example 2* 67
3.10.6 *Comments on the Value of the Singular Value Cutoff* 70
3.10.7 *LBFGS and LBFGS-SVD Results of Example 2* 71

CHAPTER 4: **SIMULATION OF PERMEABILITY FIELDS WITH SVD PARAMETERIZATION ALGORITHMS AND ENKF** 75
4.1 **Introduction** 75
4.2 **RML for Sampling Posterior PDF** 77
4.3 **Truncated SVD Algorithms for RML** 79
4.3.1 *First RML Algorithm* 79
4.3.2 *Second RML Algorithm (SVD-EnRML)* 80
4.3.3 *Steps of the SVD-EnRML and SVD-GF-EnRML Algorithms* . 82
4.3.4 *Convergence Criteria.* 86
4.4 **Ensemble Kalman Filter (EnKF)** 86
4.4.1 *Covariance localization* 88
4.5 **Computational Results** 91
4.5.1 *SVD-GF-EnRML and SVD-EnRML Simulation Results of Example 1* 92
4.5.2 *EnKF Results of Example 1* 97
4.5.3 *Incorporation of EnKF results with the SVD-GF-EnRML algorithm, Example 1* 108
4.5.4 *SVD-GF-EnRML and SVD-EnRML Simulation Results of Example 2* 113
4.5.5 *EnKF Results of Example 2* 117
4.5.6 *Incorporation of EnKF results with SVD-GF-EnRML algorithm, Example 2* 134
4.5.7 *Toy Problem* 138

CHAPTER 5: **CONCLUSIONS** 143

NOMENCLATURE 146

BIBLIOGRAPHY 146

APPENDIX A: **SINGULAR VALUE DECOMPOSITION** 154

LIST OF TABLES

Page

3.1 Geostatistical parameters. 57

3.2 Summary of computational costs of the SVD and SVD-Grd parameterization algorithms, Example 1. 62

3.3 Summary of computational costs of the basic SVD, LBFGS and LBFGS-SVD algorithms, Example 1. 66

3.4 Summary of computational costs of the SVD and SVD-Grd parameterization algorithms, Example 2. 71

3.5 Summary of computational costs of the basic SVD, LBFGS and LBFGS-SVD algorithms, Example 2. 74

4.1 Summary of computational costs of the SVD-GF-EnRML and SVD-EnRML algorithms, Example 1. 97

4.2 Summary of computational costs of SVD-GF-EnRML and SVD-EnRML algorithms, Example 2. 117

LIST OF FIGURES

Page

3.1 True horizontal log-permeability and porosity fields, Example 1. 57

3.2 Behavior of the normalized objective function and model mismatch part of the objective function during iteration, red is from basic SVD parameterization algorithm, blue is from modified SVD parameterization algorithm, and green is from SVD-Grd algorithm, Example 1. 60

3.3 Final log-permeability fields with basic and modified SVD parameterization algorithms and SVD-Grd algorithm, Example 1. 61

3.4 Observed and predicted data with basic and modified SVD parameterization algorithms and SVD-Grd algorithm, open red circles are observed data, red is from true model, blue is from basic SVD parameterization algorithm, green is from modified SVD parameterization algorithm and orange is from SVD-Grd algorithm, Example 1. 62

3.5 Behavior of the normalized objective function and model mismatch part of the objective function during iteration, red is from basic SVD parameterization algorithm, blue is from LBFGS algorithm, and green is from LBFGS-SVD algorithm, Example 1. 64

3.6 Final log-permeability fields with basic SVD parameterization, LBFGS and LBFGS-SVD algorithms, Example 1. 64

3.7 Observed and predicted data with basic SVD parameterization, LBFGS and LBFGS-SVD algorithms, open red circles are observed data, red is from true model, blue is from basic SVD parameterization algorithm, green is from LBFGS algorithm and orange is from LBFGS-SVD algorithm, Example 1. 65

3.8 True horizontal log-permeability and porosity fields, Example 2. 66

3.9 Behavior of the normalized objective function and model mismatch part of the objective function during iteration, red is from basic SVD parameterization algorithm, blue is from modified SVD parameterization algorithm, and green is from SVD-Grd algorithm, Example 2. 68

3.10 Final log-permeability fields with basic and modified SVD parameterization algorithms and SVD-Grd algorithm, Example 2. 69

3.11 Observed and predicted data with basic and modified SVD parameterization algorithms and SVD-Grd algorithm, open red circles are observed data, red is from true model, blue is from basic SVD parameterization algorithm, green is from modified SVD parameterization algorithm and orange is from SVD-Grd algorithm, Example 2. 70

3.12 Behavior of the normalized objective function and model mismatch part of the objective function during iteration, red is from basic SVD parameterization algorithm, blue is from LBFGS algorithm, and green is from LBFGS-SVD algorithm, Example 2. 72

3.13 Final log-permeability fields with basic SVD parameterization, LBFGS and LBFGS-SVD algorithms, Example 2. 73

3.14 Observed and predicted data with basic SVD parameterization, LBFGS and LBFGS-SVD algorithms, open red circles are observed data, red is from true model, blue is from basic SVD parameterization algorithm, green is from LBFGS algorithm and orange is from LBFGS-SVD algorithm, Example 2. 74

4.1 Behavior of the normalized objective function during iteration of SVD-GF-EnRML and SVD-EnRML algorithms, red curve represents MAP estimate, gray curves represent realizations, Example 1. 93

4.2 True $\ln(k)$ field and final average of realizations of log-permeability from SVD-GF-EnRML and SVD-EnRML algorithms, Example 1. 94

4.3 Unconditional realizations of log-permeability fields, Example 1. 94

4.4 Conditional realizations of log-permeability field from SVD-GF-EnRML algorithm, Example 1. 95

4.5 Conditional realizations of log-permeability field from SVD-EnRML algorithm, Example 1. 95

4.6 Data matches and future prediction from SVD-GF-EnRML, open red circles are observed data, red is from true model, blue is the prediction mean, green is from the MAP estimate, gray curves represent predictions from realizations, Example 1. 96

4.7 Data matches and future prediction from SVD-EnRML, open red circles are observed data, red is from true model, blue is the prediction mean, green is from the MAP estimate, gray curves represent predictions from realizations, Example 1. 97

4.8 Matches and predictions of the BHP of Prod-4 and the WOR of Prod-1 and Prod-4 with initial ensemble, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 1. 99

4.9 Matches and predictions of the WOR of Prod-2, Prod-3 and Prod-5 with initial ensemble, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 1. . 99

4.10 Ensemble mean final log-permeability fields as a function of ensemble size without covariance localization, Example 1. 100

4.11 Ensemble mean final log-permeability fields as a function of ensemble size with anisotropic covariance localization, Example 1. 101

4.12 Three unconditional log-permeability fields, Example 1. 101

4.13 Three ensemble members of the final log-permeability fields obtained from EnKF without localization, Ne=150, Example 1. 102

4.14 Three ensemble members of the final log-permeability fields obtained from EnKF with anisotropic localization, Ne=150, Example 1. 102

4.15 Matches and predictions of WOR of Prod-1 as a function of ensemble size during data assimilation with EnKF without localization, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 1. 103

4.16 Matches and predictions of WOR of Prod-1 as a function of ensemble size during data assimilation with EnKF with anisotropic localization, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 1. 104

4.17 Matches and predictions of WOR of Prod-1 as a function of ensemble size for rerun from time zero with EnKF without localization, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 1. 105

4.18 Matches and predictions of WOR of Prod-1 as a function of ensemble size for rerun from time zero with EnKF with anisotropic localization, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 1. 105

4.19 Matches and predictions of WOR of Prod-4 as a function of ensemble size during data assimilation with EnKF without localization, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 1. 106

4.20 Matches and predictions of WOR of Prod-4 as a function of ensemble size during data assimilation with EnKF with anisotropic localization, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 1. 106

4.21 Matches and predictions of WOR of Prod-4 as a function of ensemble size for rerun from time zero with EnKF without localization, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 1. 107

4.22 Matches and predictions of WOR of Prod-4 as a function of ensemble size for rerun from time zero with EnKF with anisotropic localization, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 1. 107

4.23 Matches and predictions of BHP of Inj-1 as a function of ensemble size during data assimilation with EnKF without localization, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 1. 108

4.24 Matches and predictions of BHP of Inj-1 as a function of ensemble size during data assimilation with EnKF with anisotropic localization, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 1. 109

4.25 Matches and predictions of BHP of Inj-1 as a function of ensemble size for rerun from time zero with EnKF without localization, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 1. 109

4.26 Matches and predictions of BHP of Inj-1 as a function of ensemble size for rerun from time zero with EnKF with anisotropic localization, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 1. 110

4.27 Behavior of the normalized objective function for each realization during iteration of SVD-GF-EnRML with $m_{\text{initial}} = m^{\text{a}}_{\text{EnkF, Loc}}$, Example 1. . . . 111

4.28 Three conditional realizations of log-permeability fields from SVD-GF-EnRML algorithm using $m_{\text{initial}} = m^{\text{a}}_{\text{EnkF, Loc}}$, Example 1. 112

4.29 Matches and predictions of BHP and WOR from SVD-GF-EnRML using $m_{\text{initial}} = m^{\text{a}}_{\text{EnkF, Loc}}$, open red circles are observed data, red is from true model, blue is the prediction mean, gray curves represent predictions from realizations, Example 1. 112

4.30 Behavior of the normalized objective function during iteration of SVD-GF-EnRML and SVD-EnRML algorithms, red curve represents MAP estimate, gray curves represent realizations, Example 2. 114

4.31 True $\ln(k)$ field and final average of realizations of log-permeability from SVD-GF-EnRML and SVD-EnRML algorithms, Example 2. 114

4.32 Unconditional realizations of log-permeability fields, Example 2. 115

4.33 Conditional realizations of log-permeability fields from SVD-GF-EnRML algorithm, Example 2. 115

4.34 Conditional realizations of log-permeability field from SVD-EnRML algorithm, Example 2. 116

4.35 Data matches and future prediction from SVD-GF-EnRML, open red circles are observed data, red is from true model, blue is the prediction mean, green is from the MAP estimate, gray curves represent predictions from realizations, Example 2. 116

4.36 Data matches and future prediction from SVD-EnRML, open red circles are observed data, red is from true model, blue is the prediction mean, green is from the MAP estimate, gray curves represent predictions from realizations, Example 2. 117

4.37 Matches and predictions of the BHP of Inj-1 and Prod-3 and the WOR of Prod-1 with initial ensemble, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 2. 119

4.38 Matches and predictions of the WOR of Prod-2, Prod-3 and Prod-4 with initial ensemble, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 2. . . 119

4.39 Ensemble mean log-permeability fields from EnKF as a function of ensemble size without covariance localization, Example 2. 120

4.40 Ensemble mean log-permeability fields from EnKF as a function of ensemble size with anisotropic covariance localization, Example 2. 121

4.41 Ensemble mean log-permeability fields from EnKF as a function of ensemble size with Kalman gain localization, Example 2. 121

4.42 Three unconditional log-permeability fields, Example 2. 122

4.43 Three ensemble members of the final log-permeability fields obtained from EnKF without localization, Ne=150, Example 2. 123

4.44 Three ensemble members of the final log-permeability fields obtained from EnKF with anisotropic covariance localization, Ne=150, Example 2. . . . 123

4.45 Three ensemble members of the final log-permeability fields obtained from EnKF with Kalman gain localization, Ne=150, Example 2. 123

4.46 Matches and predictions of WOR of Prod-2 as a function of ensemble size during data assimilation with EnKF without localization, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 2. 124

4.47 Matches and predictions of WOR of Prod-2 as a function of ensemble size for rerun from time zero with EnKF without localization, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 2. 125

4.48 Matches and predictions of WOR of Prod-2 as a function of ensemble size during data assimilation with EnKF with anisotropic covariance localization, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 2. 126

4.49 Matches and predictions of WOR of Prod-2 as a function of ensemble size during data assimilation with EnKF with Kalman gain localization, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 2. 126

4.50 Matches and predictions of WOR of Prod-2 as a function of ensemble size for rerun from time zero with EnKF with anisotropic covariance localization, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 2. 127

4.51 Matches and predictions of WOR of Prod-2 as a function of ensemble size for rerun from time zero with EnKF with Kalman gain localization, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 2. 127

4.52 Matches and predictions of WOR of Prod-3 as a function of ensemble size during data assimilation with EnKF without localization, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 2. 129

4.53 Matches and predictions of WOR of Prod-3 as a function of ensemble size during data assimilation with EnKF with anisotropic covariance localization, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 2. 129

4.54 Matches and predictions of WOR of Prod-3 as a function of ensemble size during data assimilation with EnKF with Kalman gain localization, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 2. 130

4.55 Matches and predictions of WOR of Prod-3 as a function of ensemble size for rerun from time zero with EnKF without localization, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 2. 130

4.56 Matches and predictions of WOR of Prod-3 as a function of ensemble size for rerun from time zero with EnKF with anisotropic covariance localization, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 2. 131

4.57 Matches and predictions of WOR of Prod-3 as a function of ensemble size for rerun from time zero with EnKF with Kalman gain localization, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 2. 131

4.58 Matches and predictions of BHP of Prod-3 as a function of ensemble size during data assimilation with EnKF without localization, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 2. 132

4.59 Matches and predictions of BHP of Prod-3 as a function of ensemble size during data assimilation with EnKF with anisotropic covariance localization, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 2. 132

4.60 Matches and predictions of BHP of Prod-3 as a function of ensemble size during data assimilation with EnKF with Kalman gain localization, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 2. 133

4.61 Matches and predictions of BHP of Prod-3 as a function of ensemble size for rerun from time zero with EnKF without localization, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 2. 133

4.62 Matches and predictions of BHP of Prod-3 as a function of ensemble size for rerun from time zero with EnKF with anisotropic covariance localization, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 2. 134

4.63 Matches and predictions of BHP of Prod-3 as a function of ensemble size for rerun from time zero with EnKF with Kalman gain localization, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 2. 134

4.64 Behavior of the normalized objective function for each realization during iteration of SVD-GF-EnRML with $m_{\text{initial}} = m^{\text{a}}_{\text{EnkF, Loc}}$, Example 2. . . . 135

4.65 True log-permeability field (a) and average of first 15 realizations of final ln(k) from EnKF with anisotropic covariance localization (b) and from SVD-GF-EnRML algorithm using $m_{\text{initial}} = m^{\text{a}}_{\text{EnkF, Loc}}$ (c), Example 2. . . 136

4.66 Three conditional realizations of log-permeability fields from SVD-GF-EnRML algorithm using $m_{\text{initial}} = m^{\text{a}}_{\text{EnkF, Loc}}$, Example 2. 137

4.67 Matches and predictions of BHP and WOR from SVD-GF-EnRML using $m_{\text{initial}} = m^{\text{a}}_{\text{EnkF, Loc}}$, open red circles are observed data, red is from true model, blue is the prediction mean, gray curves represent predictions from realization, Example 2. 138

4.68 Prior pdf (red curve) and posterior pdf (blue curve) after integrating 3 data. 139

4.69 Posterior pdf (blue curve) and realizations histogram (red patterns) obtained with: (a) RML, (b) SVD-EnRML, (c) EnKF and (d) SVD-EnRML-EnKF algorithms. 140

4.70 Behavior of the normalized objective function for each realization during iteration of the SVD-EnRML algorithm with (a) $m_{\text{initial}} = m_{\text{uc}}$ and (b) $m_{\text{initial}} = m^{\text{a}}_{\text{EnkF}}$, red curve represents MAP estimate, gray curves represent realizations. 141

CHAPTER 1

INTRODUCTION

Generating a distribution of reservoir properties (model parameters) like permeability and porosity fields that is consistent with all static data (core analysis data, well logs, geology, seismic, etc.) and conditioned to dynamic data (pressure, production data) is a very important task in reservoir characterization. While it is relatively easy to condition reservoir models to the static data, generation of plausible reservoir models conditioned to the dynamic data is far more challenging and difficult. The process of estimation of unknown reservoir properties by tuning input rock properties in the reservoir simulator to match dynamic data is called *history matching*.

In automatic history matching and data integration problems of interest in reservoir characterizations, an optimization algorithm is applied to minimize an objective function which quantifies the difference between the observed data and the predictions of data computed from the model parameters. In the Bayesian approach [59, 51] used in this study, a model mismatch term is also included in the objective function which measures the distance from a prior mean or unconditional realization generated from the prior geostatistical model. It is well known that the history matching problem is ill-posed, i.e., it is not possible to obtain a unique solution and very different estimates of reservoir parameters all yield an acceptable match of the production history. Therefore, the incorporation of the prior information in the objective function provides regularization and reduces the ill-posedness of the inverse history matching problem. Minimization of the objective function yields the maximum a posteriori (MAP) estimate of the model which is the most probable model.

There are two categories of optimization algorithms, gradient based optimization algorithms and non-gradient based optimization algorithms. In this study, we consider only the gradient based optimization algorithms for automatic history matching. In gradient based optimization algorithms, the gradient information required depends on the optimization algorithm used. In the Gauss-Newton method, the sensitivity matrix, which involves the derivative of predicted data with respect to all model parameters, is required to form the Hessian matrix. For the nonlinear conjugate gradient method [42, 8], explicit computation of the full sensitivity matrix is not necessary; at each iteration, we only need to form the product of the sensitivity matrix, G, times a vector and the product of the transpose of the sensitivity matrix, G^T, and a vector. Here and in the rest of this dissertation, G denotes the sensitivity matrix. G times a vector can be computed by the direct method [63], which requires a forward solution (a run of the reservoir simulator). In the reservoir simulation literature, the direct method is usually referred to as the gradient simulator method [3]. G^T times a vector can be calculated from the adjoint method [7, 6, 38, 66], which requires one "backward in time" run of the reservoir simulator. Rodrigues [57] and Kraaijevanger et al. [34] present a very clear development of the computational equations for both the gradient-simulator and adjoint methods, and the connections between them. The development of these authors is also discussed in Oliver et al. [51].

1.1 Parameterization Algorithms

For real field history matching problems, the number of grid blocks for numerical simulation is several thousands to several tens of thousands. Therefore, the number of model parameters to be adjusted in the automatic history matching is very large. Reducing the number of model parameters to be estimated reduces the computational costs and may also mitigate the effects of the natural ill-posedness of the large scale history matching problems. However, to preserve geological reality, parameterization should yield a model that is qualitatively similar to the one that would be obtained with no parameterization, e.g., parameterization should not result in excessive smoothness and artifacts inconsistent with the geological model. Thus, the development of robust and efficient algorithms for parameterization of the history matching problem is one of the objectives of this study.

The simplest reduced parameterization is provided by the method of zonation, in which the whole reservoir is divided into a small number of zones in each of which the properties are assumed to be uniform. One advantage of the zonation method is that it is simple to apply. Jacquard and Jain [30] and Jahns [31] used the zonation method to reduce the number of parameters to be estimated in one of the first automatic history matching studies presented in the literature. The gradzone method [5] and adaptive multiscale methods [26, 25] are more modern clever implementations of zonation. The gradzones are selected based on the high sensitivity of various data with respect to model parameters in each of the selected gridblocks. Adaptive multiscale methods use a coarse parameterization of the reservoir initially and refine the parameterization in regions where the introduction of additional parameter is most likely to improve the data match. Although it is simple to apply these methods, like zonation, these methods can introduce non-geological discontinuities between zones.

Gavalas et al. [22] recognized that by using a priori statistical information on the unknown parameters, the problem becomes better determined and the variability in the set of reservoir descriptions that provide an acceptable match of production data is reduced. They propose a reduction in the number of parameters by using the eigenvectors of the model covariance matrix to define the new basis. Shah et al. [58] compared results obtained by reparameterization using zonation, reparameterization using sensitivity coefficients and Bayesian estimation. For reparameterization based on the sensitivity matrix, they proposed the use of the eigenvectors associated with the largest eigenvalues of G^TG to reduce the dimension of the parameter space. The trace of the a posteriori covariance matrix was used to define the total uncertainty in the parameter estimates. They found that the smallest total uncertainty was obtained with Bayesian estimation. They showed that a Bayesian history matching approach gave better estimates of the true permeability and porosity fields than were obtained by zonation in a simulated example of a one-dimensional reservoir. They did not, however, use reparameterization when considering Bayesian estimation.

Oliver [48] also incorporated reparameterization based on the spectral (eigenvalue-eigenvector) decomposition of the prior covariance matrix to determine two-dimensional permeability fields conditioned to well-test pressure data and prior information. Reynolds et al. [54] showed that when the overall prior covariance matrix contains information for both porosity and permeability, a straightforward application of spectral decomposition results in a reparameterization which suppresses much of the porosity information. However, Reynolds et al. [54] showed that this difficulty can be eliminated by applying spectral decomposition to the prior correlation matrix. They also showed that adding a nugget effect will decrease the rate of decay of the eigenvalues and thus make spectral decomposition less efficient in reducing the number of model parameters.

The subspace method [45, 46] can also be used to reduce the number of parameters and reduce the size of the matrix problems solved during optimization. In the subspace method, the search direction vector is expanded as a linear combination of basis vectors of a lower dimensional subspace of the model space. These vectors may, for example, be gradients of subgroups of data mismatch terms and gradients of the model misfit part of the objective function. Reynolds et al. [54] used subspace methods to reduce significantly the size of the Hessian matrix in each of the Gauss-Newton iterations for automatic history-matching problems. Abacioglu et al. [1] followed the work of Reynolds et al. [54] with a more detailed investigation. The subspace vectors were computed with the adjoint method, and the gradient of the objective function with respect to the subspace vectors were calculated with the gradient simulator method. They showed that the utility of a subspace method depends on several factors, including the choice and number of the subspace vectors to be used. In an extension of the earlier work of Shah et al. [58], they presented a theoretical argument that suggests that the eigenvectors of a re-scaled Hessian matrix would be an ideal basis for parameterization. They did not use this approach in example problems, however, as it is not feasible to either generate the Hessian or to do a Schur decomposition of the Hessian for large-scale problems. Other parameterization methods that have been considered include the pilot point method [53, 10] which can introduce non-physical artifacts in the rock property fields [51], and gradual deformation [29] which tends to yield reasonable geological models but may encounter difficulties in obtaining a good data match [18].

The expansion of unknown model parameters in terms of the right singular vectors of a linear data kernel operator for linear inverse problems was introduced by Dietrich [11]. Dietrich showed that this basis is optimum in the sense that it maximizes the amount of information passed from the solution space to the data space. In this study, for parameterization, we use the right singular vectors of the dimensionless sensitivity matrix, G_D, defined in Zhang et al. [67]. We present a theoretical argument that the principle right singular vectors of the dimensionless sensitivity matrix form an optimal basis because eliminating those corresponding to smaller singular values has a negligible effect on the reduction in uncertainty obtained by conditioning a reservoir model to dynamic data. Rodrigues [57, 56] also applied a history-matching procedure where the change in the vector of model parameters over an iteration was a linear combinations of these right singular vectors.

To obtain the singular vectors corresponding to the largest singular values of the dimensionless sensitivity matrix, we use the Lanczos algorithm [24, 47, 61] to generate the truncated singular value decomposition. The Lanczos algorithm requires forming G times a vector and G^T times a vector; the first product can be computed using the "gradient simulator method" and the second product from an adjoint solution [57, 51].

1.2 Evaluation of Uncertainty

In the presence of modeling error, noise in observed data and the inherent non-uniqueness of the inverse history matching problem, there will always be some uncertainty in dynamic reservoir modeling. Our main interest is in characterizing the uncertainty in reservoir description and reservoir performance predictions as a tool to optimize reservoir management. Quantification of the uncertainty in the future production forecast yields a better understanding of the behavior of petroleum reservoirs and reduces the risk of investment by increasing the knowledge of the reservoir so that the engineer may obtain more feasible plans.

To quantify the uncertainty in reservoir performance, it is desirable to generate a suite of plausible reservoir models (realizations) that are consistent with all static information and conditioned to observed data. The correct assessment of the uncertainty will strongly depend on the quality of the distribution of the realizations, i.e., the realizations should be drawn from the probability distribution for the reservoir model and should adequately represent the underlying uncertainties. In the Bayesian framework [59, 51], the problem of uncertainty quantification is equivalent to the formulation and sampling of the a posteriori probability density function (pdf). Calculation of the MAP estimate is really only a part of the reservoir characterization problem because the MAP estimate gives a very smooth model which does not reflect the heterogeneity that would be typical for a realization generated from the prior model. The MAP estimate provides an approximation of the conditional mode of the distribution, or in the case of a Gaussian posterior, the MAP estimate provides an approximation of the conditional mean of the posterior pdf.

To generate multiple plausible realizations to evaluate the uncertainty in reservoir description and performance prediction, we need algorithms to sample correctly and efficiently from the a posteriori probability density function. Sampling algorithms have been widely reported in the literature. The rejection algorithm and Markov chain Monte Carlo (MCMC) algorithm are rigorous sampling methods. However, the results of Liu and Oliver [40] showed that the rejection algorithm is completely impractical for the problems of conditioning a reservoir model to production data. Unfortunately, the MCMC method is also computationally expensive and impractical for uncertainty evaluation of reservoir performance [49].

Davis [9] and Alabert [2] applied square root decomposition of the covariance matrix to generate realizations of Gaussian random fields in petroleum engineering. This technique can be used to generate conditional realizations of model parameters by computing a linearized approximation to the a posteriori covariance matrix, i.e., this approach assumes that the a posteriori pdf can be approximated by a Gaussian centered at the MAP estimate (LMAP). The only advantage of this method is that only one minimization corresponding to the MAP estimate is required, however, because of this linear approximation for nonlinear problems, there is no guarantee that the realizations generated by this algorithm are sampled correctly from the a posteriori pdf. (see [39, 40]).

Randomized maximum likelihood (RML) [50, 33, 55] provides a viable procedure for generating an approximate sampling of the posterior pdf. Although RML provides only an approximate sampling procedure, by its construction, it is expected to at least generate samples around the modes of the posterior pdf. In the linear case where data are linearly related to the model parameters, it has been shown that the RML method samples the a posteriori pdf correctly [55, 51]. In this algorithm, the conditional realizations of model parameters are generated by calibration of unconditional realizations of model parameters drawn from the prior covariance matrix and unconditional realizations of observed data generated from the square root of the data covariance matrix. In this work, we incorporate the truncated singular value decomposition (SVD) parameterization algorithm into the RML method for sampling from posterior pdf. Based on this, we develop two algorithms of sampling which are significantly more efficient than the standard implementation of RML using a quasi-Newton method.

1.3 Research Contributions and Dissertation Outline

1.3.1 Research Contributions

The development of practical, robust and efficient techniques for automatic history matching is a research problem of great interest. The main contribution of this work was the development of a parameterization algorithm in which the change in the vector of model parameters is represented by the linear combination of the right singular vectors of the dimensionless sensitivity matrix. This SVD parameterization algorithm does not require the explicit knowledge of the sensitivity matrix, G. We provide a theoretical basis which shows that the parameterization based on the principal singular triplets of the dimensionless sensitivity matrix is ideal if the objective is to minimize the posterior uncertainty in the model parameters. We show that some savings in computational expense can be achieved by starting with a small number of singular vectors initially and gradually increasing the number during the iterative process.

In this study, we consider both the problem of generating the MAP estimate as well as procedures for sampling from the posterior pdf to generate conditional realizations of the log-permeability field. Another contribution of this work was to develop efficient procedures for sampling from the posterior pdf. We provide two algorithms for sampling from the posterior pdf based on incorporation of truncated SVD parameterization into the RML method. The purpose of these algorithms is to reduce the computational costs of generating a suite of plausible samples compared to the standard implementation of RML using a quasi-Newton method.

1.3.2 Dissertation Outline

There are five chapters and one appendix in this dissertation. In Chapter 2, we briefly present the basic equations of an implicit pressure explicit saturation (IMPES) reservoir simulator, the adjoint method and the forward gradient simulator used with the IMPES simulator. A detailed procedure of the Lanczos algorithm is also presented in this chapter. Chapter 3 includes a discussion of Bayesian inversion, the theoretical development of the MAP estimate and the formulation of the SVD parameterization algorithms. The computational results of the MAP estimate with different algorithms are compared in this chapter. Chapter 4 covers the detailed development of the sampling algorithms based on incorporation of the SVD truncated parameterization into the RML. This chapter also includes the results of the uncertainty of both model parameters and future performance predictions obtained with SVD sampling algorithms compared to results generated using the ensemble Kalman filter (EnKF). Chapter 5 presents the conclusions and summarizes the research contribution of this study. In Appendix A, the critical properties of the singular value decomposition needed in this study are provided.

CHAPTER 2

GRADIENT CALCULATION IN IMPES RESERVOIR SIMULATOR WITH ADJOINT CODE AND FORWARD GRADIENT METHODS

In gradient-based optimization methods for automatic history matching, calculation of the gradient information, i.e., the sensitivity of dynamic data to reservoir model parameters, is the most challenging part. As we mentioned in the introduction chapter, we use the Lanczos algorithm to obtain the singular vectors corresponding to the largest singular values of the dimensionless sensitivity matrix. The Lanczos algorithm requires forming the sensitivity matrix, G, times a vector and G^T times a vector; the first product can be computed using the "gradient simulator method" and the second product from an adjoint solution. In the first section of this chapter, the details of two-phase (oil and water), two-dimensional (x-y coordinate system) IMPES reservoir simulator is described. In the second section, an outline of the derivation of the adjoint and the gradient simulator methods used to generate the product of the sensitivity matrix and the transpose of the sensitivity matrix with arbitrary vectors are presented. Finally, we present the details of the bidiagonalized Lanczos algorithm procedure which is used to compute the truncated singular value decomposition of a dimensionless sensitivity matrix.

2.1 IMPES Simulator

The simulator used to generate finite-difference solutions is a standard "implicit pressure and explicit saturation" (IMPES) simulator. This IMPES simulator is based on finite-difference formulation black-oil equations, two phase and two-dimensional (x-y coordinate system) flow of oil and water in an isotropic reservoir of uniform thickness. Capillary pressures, gravity effects and rock compressibility are assumed to be negligible. We assume no-flow outer reservoir boundaries and specify uniform initial pressure and water saturation as the initial conditions. We suppose there are N_x and N_y gridblocks in the x and y directions respectively. We let N denote the total number of gridblocks, i.e., $N = N_x \times N_y$. At each gridblock, two basic finite-difference equations apply which represent the mass balance for each of the two components, oil and water. In addition, a constraint is applied at each of the N_w wells to yield N_w additional equations. At each well at each time step, either an individual phase flow rate, the total flow rate or the wellbore pressure may be specified as a well constraint.

Based on the mass balance equations, we can drive the finite-difference equation for each component in each grid block at each time step. We partition the reservoir into rectangular gridblocks using a standard block-centered grid and let (x_i,y_j), $i = 1, 2, \ldots, N_x$ and $j = 1, 2, \ldots, N_y$ denote the areal coordinates of the gridblock centers. Using standard notation, the difference equations can be written

$$\begin{aligned} f_{u,i,j}^{n+1} =& T_{ux,i+1/2,j}^{n+1}(p_{i+1,j}^{n+1} - p_{i,j}^{n+1}) - T_{ux,i-1/2,j}^{n+1}(p_{i,j}^{n+1} - p_{i-1,j}^{n+1}) + \\ & T_{uy,i,j+1/2}^{n+1}(p_{i,j+1}^{n+1} - p_{i,j}^{n+1}) - T_{uy,i,j-1/2}^{n+1}(p_{i,j}^{n+1} - p_{i,j-1}^{n+1}) - q_{u,i,j}^{n+1} \\ & - V_{\phi,i,j}^{n+1}\left[(b_u S_u)_{i,j}^{n+1} - (b_u S_u)_{i,j}^{n}\right] = 0, \end{aligned} \tag{2.1}$$

for $u = o, w$ (oil or water), $i = 1, 2, \ldots, N_x$, $j = 1, 2, \ldots, N_y$, and $n = 0, 1, 2, \ldots$, where

$$V_{\phi,i,j}^{n} = \frac{\triangle x_i \triangle y_j h \phi_{i,j}}{5.615 \triangle t^{n-1}}. \tag{2.2}$$

Note that to impose no-flow outer boundaries, for all time steps $n = 0, 1, 2, \ldots$, we set

$$T^{n+1}_{ux,1/2,j} = T^{n+1}_{ux,N_x+1/2,j} = 0, \tag{2.3}$$

for $j = 1, 2, \ldots, N_y$, and

$$T^{n+1}_{uy,i,1/2} = T^{n+1}_{uy,i,N_y+1/2} = 0, \tag{2.4}$$

for $i = 1, 2, \ldots, N_x$. Throughout, n refers to the time index and $\triangle t^n$ refers to the time step size with $t^{n+1} = t^n + \triangle t^n$, for $n = 0, 1, 2, \ldots$ where $t^0 = 0$. The term b_u refers to the inverse of formation volume factor (FVF) of phase u, i.e., $b_u = 1/B_u$, where B_u is the FVF of phase u. The terms $q_{u,i,j}$ represent source/sink terms in STB/D for phase u and are zero unless grid block (i,j) contains a production or injection well.

We define $\Delta_t h^{n+1} = h^{n+1} - h^n$ for any arbitrary function h. Therefore, for two arbitrary functions a and b, we can simply show that

$$\Delta_t(ab)^{n+1} = b^{n+1}\Delta_t(a)^{n+1} + a^n\Delta_t(b)^{n+1}. \tag{2.5}$$

Using a trick given by Eq. 2.5 in the last term of Eq. 2.1 and doing some simple manipulations, we can rewrite Eq. 2.1 as

$$\begin{aligned} f^{n+1}_{u,i,j} =& T^{n+1}_{ux,i+1/2,j}(p^{n+1}_{i+1,j} - p^{n+1}_{i,j}) - T^{n+1}_{ux,i-1/2,j}(p^{n+1}_{i,j} - p^{n+1}_{i-1,j}) + \\ & T^{n+1}_{uy,i,j+1/2}(p^{n+1}_{i,j+1} - p^{n+1}_{i,j}) - T^{n+1}_{uy,i,j-1/2}(p^{n+1}_{i,j} - p^{n+1}_{i,j-1}) - q^{n+1}_{u,i,j} \\ & - V^{n+1}_{\phi,i,j} b^{n+1}_{u,i,j}\big[S^{n+1}_{u,i,j} - S^{n}_{u,i,j}[1 - C_u(p^{n+1}_{i,j} - p^{n}_{i,j})]\big] = 0, \end{aligned} \tag{2.6}$$

where C_u denotes phase compressibility. Note that to obtain Eq. 2.6, we assume that the phase FVF does not vary over a time step.

For a two-phase flow problem, there are two unknowns, or two primary variables, in each gridblock. The primary variables for a gridblock (i,j) are the gridblock pressure, $p_{i,j}$, and oil saturation, $S_{o,i,j}$. Note that in each grid block, (i,j), we impose the auxiliary condition $S_{o,i,j} + S_{w,i,j} = 1.0$. In addition to the gridblock pressures and saturations, the flowing wellbore pressure, $p_{wf,l}$, at the lth well is also a primary variable. At each time step, the primary variables, p and S_o, of each individual gridblock and p_{wf} in each well are calculated by an IMPES simulator and are saved for future use in constructing the adjoint system and forward gradient.

For IMPES simulator, we combine oil and water equations given by Eq. 2.6, respectively, with $u = o$ for oil phase and $u = w$ for water phase. Note that for oil and water two-phase problems $S_{o,i,j} + S_{w,i,j} = 1.0$ for (i,j)'th gridblock. Therefore, by using this auxiliary equation in the combined equation, we can eliminate the unknown phase saturation. Using some simple algebraic manipulations and simplification, we obtain the following combined equation for phase pressure denoted by $f^{n+1}_{p,i,j}$

$$\begin{aligned} f^{n+1}_{p,i,j} =& (T^n_{ox,i+1/2,j} + T^n_{wx,i+1/2,j})(p^{n+1}_{i+1,j} - p^{n+1}_{i,j}) - (T^n_{ox,i-1/2,j} + T^n_{wx,i-1/2,j})(p^{n+1}_{i,j} - p^{n+1}_{i-1,j}) + \\ & (T^n_{oy,i,j+1/2} + T^n_{wy,i,j+1/2})(p^{n+1}_{i,j+1} - p^{n+1}_{i,j}) - (T^n_{oy,i,j-1/2} + T^n_{wy,i,j-1/2})(p^{n+1}_{i,j} - p^{n+1}_{i,j-1}) - \\ & (B^n_{o,i,j} q^{n+1}_{o,i,j} + B^n_{w,i,j} q^{n+1}_{w,i,j}) - V^{n+1}_{\phi,i,j} \left[C_o S^n_{o,i,j} + C_w (1 - S^n_{o,i,j}) \right] (p^{n+1}_{i,j} - p^n_{i,j}) = 0, \end{aligned} \tag{2.7}$$

for $i = 1, 2, \ldots, N_x$, $j = 1, 2, \ldots, N_y$, and $n = 0, 1, 2, \ldots$, where C_o and C_w denote oil and water compressibilities, respectively, and B_o and B_w denote oil and water FVF's. We solve this combined equation implicitly for phase pressure of all gridblocks, $p^{n+1}_{i,j}$ for $i = 1, 2, \ldots, N_x$ and $j = 1, 2, \ldots, N_y$. Note that to solve the pressure equations given by Eq. 2.7, we backdate all nonlinear terms, e.g., the transmissibilities of oil and water backdated to previous time step t^n. After having obtained phase pressure, we choose to obtain oil saturation of each gridblock explicitly from the oil phase equations. We let $f^{n+1}_{S_o,i,j}$ denote the oil phase saturation equation for gridblock (i,j). Therefore, we can write

$$\begin{aligned} f^{n+1}_{S_o,i,j} =& T^n_{ox,i+1/2,j}(p^{n+1}_{i+1,j} - p^{n+1}_{i,j}) - T^n_{ox,i-1/2,j}(p^{n+1}_{i,j} - p^{n+1}_{i-1,j}) + \\ & T^n_{oy,i,j+1/2}(p^{n+1}_{i,j+1} - p^{n+1}_{i,j}) - T^n_{oy,i,j-1/2}(p^{n+1}_{i,j} - p^{n+1}_{i,j-1}) - B^n_{o,i,j} q^{n+1}_{o,i,j} \\ & - V^{n+1}_{\phi,i,j}\left[S^{n+1}_{o,i,j} - S^n_{o,i,j}[1 - C_o(p^{n+1}_{i,j} - p^n_{i,j})]\right](p^{n+1}_{i,j} - p^n_{i,j}) = 0. \end{aligned} \tag{2.8}$$

We solve Eq. 2.8 explicitly to obtain $S^{n+1}_{o,i,j}$ for $i = 1, 2, \ldots, N_x$ and $j = 1, 2, \ldots, N_y$ for each time step of the simulation, $n = 0, 1, 2, \ldots$. In contrast to a fully-implicit simulator, we can partition the pressure and saturation equations in the IMPES simulator, i.e., the pressure and saturation equations are solved independently.

In our IMPES simulator, after calculation of the phase pressures and saturations, we can use well constraint equations to obtain well bottom-hole pressure (BHP) and oil and water flow rates. If a well flowing BHP is specified, phase flow rates of that well are computed by Peaceman's equation [52]. The oil and water flow rates of a producer well l located at gridblock (i, j) at time step $n + 1$ can be evaluated as

$$q^{n+1}_{o,i,j} = WI_{i,j}\Big(\frac{k_{ro}}{B_o\mu_o}\Big)^{n+1}_{i,j}(p^{n+1}_{i,j} - p^{n+1}_{wf,l}), \tag{2.9}$$

$$q^{n+1}_{w,i,j} = WI_{i,j}\Big(\frac{k_{rw}}{B_w\mu_w}\Big)^{n+1}_{i,j}(p^{n+1}_{i,j} - p^{n+1}_{wf,l}), \tag{2.10}$$

where $WI_{i,j}$ denotes the well index term which represents the well geometry and $p^{n+1}_{wf,l}$ is the specified BHP of a producer. Similarly, the total water injection rate for a injection well l at gridblock (i, j) at time step $n + 1$ can be evaluated as

$$q^{n+1}_{w,i,j} = WI_{i,j}\Big(\frac{k_{ro}}{B_o\mu_o} + \frac{k_{rw}}{B_w\mu_w}\Big)^{n+1}_{i,j}(p^{n+1}_{i,j} - p^{n+1}_{wf,l}), \tag{2.11}$$

Note that for injection wells, we use total mobility (oil+water) to compute injection rate. After initial simulation time steps, the total mobility at injection well grid block is equal to the water mobility. Therefore, depending on the specified constraint for well l, we can calculate the phase flow rates or BHP with Eqs. 2.9, 2.10 or 2.11 for $l = 1, 2, \ldots, N_w$. Note that when we solve Eqs. 2.7 and 2.8 for phase pressures and oil saturations, we use Eqs. 2.9, 2.10 and 2.11 to replace $q_{o,i,j}^{n+1}$ and $q_{w,i,j}^{n+1}$.

From this point on, we will assume that the gridblocks are reordered in one dimension from $k = 1, 2, \ldots, N = N_x \times N_y$. This reordering is done such that for a gridblock (i, j) in two-dimension, the index k in one dimension ordering is obtained with $k = (j-1)N_x + i$, where $i = 1, 2, \ldots, N_x$ and $j = 1, 2, \ldots, N_y$. We let y^{n+1} denote a column vector which contains the set of primary variables (pressures, saturations and well bottom hole pressures) at time step $n+1$. Since we solve gridblock pressure and saturation equations separately in the IMPES simulator and then we obtain well BHP's, we write y^{n+1} as

$$\begin{aligned} y^{n+1} =& [(P^{n+1})^T, (S_o^{n+1})^T, (P_{wf}^{n+1})^T]^T \\ &= [p_1^{n+1}, p_2^{n+1}, \ldots, p_N^{n+1}, S_{o,1}^{n+1}, S_{o,2}^{n+1}, \ldots, S_{o,N}^{n+1}, p_{wf,1}^{n+1}, \ldots, p_{wf,N_w}^{n+1}]^T, \end{aligned} \tag{2.12}$$

where

$$P^{n+1} = [p_1^{n+1}, p_2^{n+1}, \ldots, p_N^{n+1}]^T, \tag{2.13}$$

$$S_o^{n+1} = [S_{o,1}^{n+1}, S_{o,2}^{n+1}, \ldots, S_{o,N}^{n+1}]^T, \tag{2.14}$$

and

$$P_{wf}^{n+1} = [p_{wf,1}^{n+1}, p_{wf,2}^{n+1}, \ldots, p_{wf,N_w}^{n+1}]^T. \tag{2.15}$$

Therefore, the finite-difference combined (pressure) equation given by Eq. 2.7 and oil saturation equation given by Eq. 2.8 in gridblock k at time t^{n+1}, respectively, can be written as

$$f_{p,k}^{n+1} = f_{p,k}^{n+1}(y^{n+1}, y^n, m) = 0, \tag{2.16}$$

and

$$f_{S_o,k}^{n+1} = f_{S_o,k}^{n+1}(y^{n+1}, y^n, m) = 0, \tag{2.17}$$

for $k = 1, 2, \ldots, N$ and $n = 1, 2, \ldots, L$. Note that m denotes the vector of model parameters like gridblock porosities and permeabilities given by

$$m = [m_1, m_2, \ldots, m_{N_m}]^T. \tag{2.18}$$

Also, the well constraints equations given by Eqs. 2.9, 2.10 and 2.11 can be represented by

$$f_{wf,l}^{n+1} = f_{wf,l}(y^{n+1}, y^n, m) = 0, \tag{2.19}$$

for $l = 1, 2, \ldots, N_w$ and $n = 1, 2, \ldots, L$. Therefore, similar to a fully implicit simulator, the complete system of equations for IMPES simulator for all gridblocks and all simulation time steps can be written as

$$f^{n+1} = f(y^{n+1}, y^n, m) = \begin{bmatrix} f_p^{n+1} \\ f_{S_o}^{n+1} \\ f_{wf}^{n+1} \end{bmatrix} = \begin{bmatrix} f_{p,1}^{n+1} \\ \vdots \\ f_{p,N}^{n+1} \\ f_{S_o,1}^{n+1} \\ \vdots \\ f_{S_o,N}^{n+1} \\ f_{wf,1}^{n+1} \\ \vdots \\ f_{wf,N_w}^{n+1} \end{bmatrix} = 0, \tag{2.20}$$

where

$$f_p^{n+1} = [f_{p,1}^{n+1}, f_{p,1}^{n+1}, \ldots, f_{p,N}^{n+1}]^T, \tag{2.21}$$

$$f_{S_o}^{n+1} = [f_{S_o,1}^{n+1}, f_{S_o,1}^{n+1}, \ldots, f_{S_o,N}^{n+1}]^T, \tag{2.22}$$

and

$$f_{wf}^{n+1} = [f_{wf,1}^{n+1}, f_{wf,1}^{n+1}, \ldots, f_{wf,N_w}^{n+1}]^T. \tag{2.23}$$

Note that unlike a fully implicit simulator, for the IMPES simulator, we have partitioned the combined pressure equations, saturation equations and well constraints equations. The total number of unknowns to be solved is $N_e = 2N + N_w$ for each time step of simulation.

2.2 Adjoint Equations

In gradient-based optimization algorithms, minimization of objective function requires computation of the gradient of the objective function, $O(m)$. In quasi-Newton algorithms and in the SVD parameterization algorithm based on the Lanczos method, the calculation of the product of the transpose of the sensitivity matrix, G^T, times a vector is required. This product can be obtained by adjoint solution [66, 57]. In this section, we describe the detail formulation of the adjoint code for the IMPES simulator. The predicted data $g(m)$ and the objective function, $O(m)$, are functions of the primary variables, $y^1, y^2, \ldots, y^L$, and model parameters, m. Note that the primary variables, y_i's, for $i = 1, 2, \ldots, L$ are implicit functions of the model parameters m, where this implicit relation is represented by the mass balance equations given by Eq. 2.20.

Following Oliver et al. [51], Gao [17], Li et al. [38], we define a general scalar function by

$$\beta = \beta(y^1, \ldots, y^L, m), \tag{2.24}$$

where L corresponds to the last time step t^L at which one wishes to compute sensitivities. Our objective is to compute the product of the transpose of the sensitivity matrix by a vector. Therefore, in the SVD parameterization algorithm, β will represent the vector of predicted data, $g(m)$ times a vector v. Here, the objective is to compute the sensitivity of β to the model parameter, m, i.e. $d\beta/dm$. We now adjoin Eq. 2.20 to the function β to obtain the adjoint functional J given by

$$J = \beta + \sum_{n=0}^{L} (\lambda^{n+1})^T f^{n+1}, \tag{2.25}$$

where λ^{n+1}, $n = 0, 1, \ldots, L$, is the vector of adjoint variables at time step $n+1$, and is given by

$$\lambda^{n+1} = \begin{bmatrix} \lambda_p^{n+1} \\ \lambda_{S_o}^{n+1} \\ \lambda_{wf}^{n+1} \end{bmatrix} = \begin{bmatrix} \lambda_{p,1}^{n+1} \\ \vdots \\ \lambda_{p,N}^{n+1} \\ \lambda_{S_o,1}^{n+1} \\ \vdots \\ \lambda_{S_o,N}^{n+1} \\ \lambda_{wf,1}^{n+1} \\ \vdots \\ \lambda_{wf,N_w}^{n+1} \end{bmatrix}. \tag{2.26}$$

Here, $\lambda_{u,j}^n$ denotes the scalar adjoint variable (Lagrange multiplier) for the u simulator equation at gridblock j at time t^n. As shown in Li et al. [37] and Oliver et al. [51], the adjoint problem is

$$\lambda^{n+1} = 0, \tag{2.27}$$

and

$$[\nabla_{y^n}(f^n)^T]\lambda^n = -[\nabla_{y^n}(f^{n+1})^T]\lambda^{n+1} - \nabla_{y^n}\beta, \tag{2.28}$$

for $n = L, L-1, \ldots, 1$. Based on Eqs. 2.12 and 2.20, we can partition pressure, oil saturation and well constraints equations. Therefore, we can write the following

$$\nabla_{y^n}(f^n)^T = \begin{bmatrix} \nabla_{p^n}(f_P^n)^T & \nabla_{p^n}(f_{S_o}^n)^T & \nabla_{p^n}(f_{wf}^n)^T \\ \nabla_{S_o^n}(f_P^n)^T & \nabla_{S_o^n}(f_{S_o}^n)^T & \nabla_{S_o^n}(f_{wf}^n)^T \\ \nabla_{p_{wf}^n}(f_P^n)^T & \nabla_{p_{wf}^n}(f_{S_o}^n)^T & \nabla_{p_{wf}^n}(f_{wf}^n)^T \end{bmatrix}. \tag{2.29}$$

Note that in the IMPES simulator formulation based on Eqs. 2.7 and 2.8 and well constraints equations (Eqs. 2.9, 2.10 and 2.11), some of the block matrix entries of $\nabla_{y^n}(f^n)^T$ given by Eq. 2.29 are zero, e.g., $\nabla_{S_o^n}(f_P^n)^T = 0$, $\nabla_{S_o^n}(f_{wf}^n)^T = 0$, $\nabla_{p_{wf}^n}(f_P^n)^T = 0$ and $\nabla_{p_{wf}^n}(f_{S_o}^n)^T = 0$. Also, $\nabla_{S_o^n}(f_{S_o}^n)^T$ and $\nabla_{p_{wf}^n}(f_{wf}^n)^T$ are diagonal matrices. Similarly, using Eqs. 2.12 and. 2.20, we can rewrite $\nabla_{y^n}(f^{n+1})^T$ by partitioning as follows

$$\nabla_{y^n}(f^{n+1})^T = \begin{bmatrix} \nabla_{p^n}(f_P^{n+1})^T & \nabla_{p^n}(f_{S_o}^{n+1})^T & \nabla_{p^n}(f_{wf}^{n+1})^T \\ \nabla_{S_o^n}(f_P^{n+1})^T & \nabla_{S_o^n}(f_{S_o}^{n+1})^T & \nabla_{S_o^n}(f_{wf}^{n+1})^T \\ \nabla_{p_{wf}^n}(f_P^{n+1})^T & \nabla_{p_{wf}^n}(f_{S_o}^{n+1})^T & \nabla_{p_{wf}^n}(f_{wf}^{n+1})^T \end{bmatrix}. \tag{2.30}$$

Note that based on the same equations we developed for the IMPES simulator in the previous section (Eqs. 2.7, 2.8, 2.9, 2.10 and 2.11), there are some null and diagonal matrices in $\nabla_{y^n}(f^{n+1})^T$ given by Eq. 2.30. For example, the gradients of $f^{n+1} = [f_p^{n+1}, f_{S_o}^{n+1}, f_{wf}^{n+1}]^T$ with respect to p_{wf}^n are null matrices ($\nabla_{p_{wf}^n}(f_P^{n+1})^T = 0$, $\nabla_{p_{wf}^n}(f_{S_o}^{n+1})^T = 0$, $\nabla_{p_{wf}^n}(f_{wf}^{n+1})^T = 0$). The matrices $\nabla_{p^n}(f_P^{n+1})^T$ and $\nabla_{p^n}(f_{S_o}^{n+1})^T$ are diagonal. We can rewrite $\nabla_{y^n}\beta$ based on partitioned primary variables (Eq. 2.12) as follows

$$\nabla_{y^n}\beta = \begin{bmatrix} \nabla_{p^n}\beta \\ \nabla_{S_o^n}\beta \\ \nabla_{p_{wf}^n}\beta \end{bmatrix}. \tag{2.31}$$

We let RHS denote a column vector which contains the right hand side of Eq. 2.28 as $RHS = [RHS_p^T, RHS_{S_o}^T, RHS_{wf}^T]^T$. Therefore, using Eqs. 2.26 and. 2.29, we can rewrite Eq. 2.28 as

$$\begin{bmatrix} \nabla_{p^n}(f_P^n)^T & \nabla_{p^n}(f_{S_o}^n)^T & \nabla_{p^n}(f_{wf}^n)^T \\ 0 & \nabla_{S_o^n}(f_{S_o}^n)^T & 0 \\ 0 & 0 & \nabla_{p_{wf}^n}(f_{wf}^n)^T \end{bmatrix} \begin{bmatrix} \lambda_p^n \\ \lambda_{S_o}^n \\ \lambda_{wf}^n \end{bmatrix} = \begin{bmatrix} RHS_p \\ RHS_{S_o} \\ RHS_{wf} \end{bmatrix}. \tag{2.32}$$

From Eq. 2.32, it is clear that we obtain λ_{wf}^n, $\lambda_{S_o}^n$ and λ_p^n from following equations respectively

$$\nabla_{p_{wf}^n}(f_{wf}^n)^T \lambda_{wf}^n = RHS_{wf}, \tag{2.33}$$

$$\nabla_{S_o^n}(f_{S_o}^n)^T \lambda_{S_o}^n = RHS_{S_o}, \tag{2.34}$$

and

$$\nabla_{p^n}(f_P^n)^T \lambda_p^n = RHS_p - \nabla_{p^n}(f_{S_o}^n)^T \lambda_{S_o}^n - \nabla_{p^n}(f_{wf}^n)^T \lambda_{wf}^n, \tag{2.35}$$

We solve adjoint equations (Eqs. 2.33, 2.34 and 2.35) backward in time for $n = L, L-1, \ldots, 1$ where Eq. 2.27 gives the starting condition for the backward solution in time. Note that the forward simulation equation is solved forward in time. Also note that the coefficients in Eqs. 2.33, 2.34 and 2.35 are independent of the adjoint variables λ_p, λ_{S_o} and λ_{wf}, which means that the adjoint equations are linear. The information needed in building the coefficients in the adjoint equations must be saved from the forward simulation run. When the λ^ns satisfy the adjoint system of Eq. 2.28, as shown in Li et al. [37] the total differential of J can be written as

$$dJ = \Big\{ [\nabla_m \beta]^T + \sum_{n=1}^{N} (\lambda^n)^T [\nabla_m (f^n)^T]^T \Big\} dm. \tag{2.36}$$

It follows that the total derivative of J with respect to m, i.e. the sensitivities are given by

$$\frac{dJ}{dm} = \begin{bmatrix} \frac{dJ}{dm_1} \\ \vdots \\ \frac{dJ}{dm_{N_m}} \end{bmatrix} = \nabla_m \beta + \sum_{n=1}^{N} \big[\nabla_m (f^n)^T\big](\lambda^n), \tag{2.37}$$

or equivalently based on the partitioned λ^n and f^n given by Eqs. 2.26 and 2.20, we can write

$$\frac{dJ}{dm} = \nabla_m \beta + \sum_{n=1}^{N} \Big\{ \big[\nabla_m (f_p^n)^T\big](\lambda_p^n) + \big[\nabla_m (f_{S_o}^n)^T\big](\lambda_{S_o}^n) + \big[\nabla_m (f_{wf}^n)^T\big](\lambda_{wf}^n) \Big\}, \tag{2.38}$$

where

$$\nabla_m \beta = \left[\frac{\partial \beta}{\partial m_1}, \frac{\partial \beta}{\partial m_2}, \ldots, \frac{\partial \beta}{\partial m_{N_m}} \right]^T, \tag{2.39}$$

and

$$\nabla_m [f_p^n]^T = \begin{bmatrix} \frac{\partial f_{p,1}^n}{\partial m_1} & \frac{\partial f_{p,2}^n}{\partial m_1} & \cdots & \frac{\partial f_{p,N}^n}{\partial m_1} \\ \frac{\partial f_{p,1}^n}{\partial m_2} & \frac{\partial f_{p,2}^n}{\partial m_2} & \cdots & \frac{\partial f_{p,N}^n}{\partial m_2} \\ \vdots & \vdots & \vdots & \vdots \\ \frac{\partial f_{p,1}^n}{\partial m_{N_m}} & \frac{\partial f_{p,2}^n}{\partial m_{N_m}} & \cdots & \frac{\partial f_{p,N}^n}{\partial m_{N_m}} \end{bmatrix}. \tag{2.40}$$

In Eq. 2.40, if we use $f_{S_o}^n$ instead of f_p^n, we will obtain equation for $\nabla_m [f_{S_o}^n]^T$. The matrices $\nabla_m [f_p^n]^T$ and $\nabla_m [f_{S_o}^n]^T$ are $N_m \times N$ sparse matrices. The matrix $\nabla_m [f_{wf}^n]^T$ is an $N_m \times N_w$ sparse matrix. Note that in the preceding two equations, the gradients involve the explicit partial derivatives of terms with respect to model parameters. If the expression for a term does not explicitly involve a model parameter, the partial derivative of that term with respect to that model parameter is zero.

In order to apply for example a quasi-Newton algorithm in minimization of an objective function, we need only to compute the gradient of the objective function and this can be done by setting β equal to the data mismatch part of the objective function in the adjoint procedure. The detailed description of how to obtain the gradient of the data mismatch part of the objective function by the adjoint procedure is given in Zhang [68] and Oliver et al. [51].

2.2.1 G^T Times a Vector

The implementation of the Lanczos algorithm to obtain the truncated SVD of the dimensionless sensitivity matrix requires computation of product of G^T times a vector v. As shown in Oliver et al. [51], if $g(m)$ is the vector of all predicted data obtained by running the simulator forward corresponding to all observed data, then for any fixed vector v we denote β as

$$\beta = [g(m)]^T v. \tag{2.41}$$

The total derivative (sensitivity) vector of β is given by

$$\frac{d\beta}{dm} = G^T v. \tag{2.42}$$

Thus to compute $G^T v$, we simply apply the adjoint procedure to compute $d\beta/dm$.

2.3 Forward Gradient Method

In this section, we will show how the forward gradient method for an IMPES reservoir simulator can be used to calculate the sensitivity matrix. Also, in the process, we show how to calculate the product of the sensitivity matrix, G with an arbitrary vector u.

In order to provide the derivation of the forward gradient method for computing G times a vector, we rewrite the system of reservoir simulator finite difference equations given by Eq. 2.20 as

$$f^n = f(y^n, y^{n-1}, m) = 0, \tag{2.43}$$

for $n = 1, 2, \ldots, L$. Note that y^0 represents given fixed initial conditions. Next, we combine vectors of primary variables given by Eq. 2.12 at all time steps into one overall primary column vector Y with size of $N_Y = N_e \times L$ as

$$Y = [(y^1)^T, (y^2)^T, \ldots, (y^L)^T]^T, \tag{2.44}$$

and combine the simulation equations given by Eq. 2.43 at all time steps into one overall simulation equation $F = F(Y, m)$ as

$$F = F(Y, m) = \begin{bmatrix} f^1(y^1, y^0, m) \\ f^2(y^2, y^1, m) \\ \vdots \\ f^L(y^L, y^{L-1}, m) \end{bmatrix}, \tag{2.45}$$

so the complete functional relationship defining the reservoir simulator is written as

$$F(Y, m) = 0. \tag{2.46}$$

To calculate the total derivatives (sensitivities) of primary variables with respect to the model parameters, we take the total differential of F. From Eq. 2.46, it follows that

$$(\nabla_Y F^T)^T dY + (\nabla_m F^T)^T dm = 0. \tag{2.47}$$

It follows from Eq. 2.47 that

$$(\nabla_Y F^T)^T \frac{dY}{dm} = -(\nabla_m F^T)^T, \tag{2.48}$$

where the matrix of total derivatives, dY/dm, contains sensitivity matrices for all time steps and is given by

$$\frac{dY}{dm} = \begin{bmatrix} \frac{dy^1}{dm} \\ \frac{dy^2}{dm} \\ \vdots \\ \frac{dy^L}{dm} \end{bmatrix} = \begin{bmatrix} [\nabla_m (y^1)^T]^T \\ [\nabla_m (y^2)^T]^T \\ \vdots \\ [\nabla_m (y^L)^T]^T \end{bmatrix}, \tag{2.49}$$

and

$$(\nabla_Y F^T)^T = \begin{bmatrix} [\nabla_{y^1}(f^1)^T]^T & O & O & \dots & O \\ [\nabla_{y^1}(f^2)^T]^T & [\nabla_{y^2}(f^2)^T]^T & O & \dots & O \\ O & [\nabla_{y^2}(f^3)^T]^T & [\nabla_{y^3}(f^3)^T]^T & \dots & O \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ O & O & \dots & [\nabla_{y^{L-1}}(f^L)^T]^T & [\nabla_{y^L}(f^L)^T]^T \end{bmatrix}, \tag{2.50}$$

where the O's denote null matrices and

$$(\nabla_m F^T)^T = \begin{bmatrix} [\nabla_m(f^1)^T]^T \\ [\nabla_m(f^2)^T]^T \\ \vdots \\ [\nabla_m(f^L)^T]^T \end{bmatrix}. \tag{2.51}$$

As shown in Rodrigues [57] and Oliver et al. [51], from Eqs. 2.48- 2.51 it follows that the total derivatives of primary variables with respect to model parameters can be written forward in time recursively as

$$(\nabla_{y^n}[(f^n)^T]^T)\hat{G}_n = -(\nabla_{y^{n-1}}[(f^n)^T]^T)\hat{G}_{n-1} - [\nabla_m(f^n)^T]^T, \tag{2.52}$$

where $\hat{G}_n = [\nabla_m(y^n)^T]^T$ for $n = 1, 2, \dots, L$. Note that $\hat{G}_0$ is a null matrix because y^0 is a fixed vector and hence independent of m.

The vector of simulator predicted data corresponding to the observed data is denoted by g and it is actually a function of primary variables, Y and model parameters, m which can be written as

$$g = g(Y, m) = [g_1(Y, m), g_2(Y, m), \dots, g_{N_d}(Y, m)]^T, \tag{2.53}$$

where N_d is the number of observed data and $g_i(Y, m)$'s for $i = 1, 2, \dots, N_d$ are the predicted data, like bottom hole pressure (BHP), water oil ratio (WOR), ..., obtained by running the reservoir simulator forward. The matrix of total derivatives of the predicted data, $g_i(Y, m)$, $i = 1, 2, \ldots, N_d$, with respect to the model parameters is called the sensitivity matrix and is defined as

$$G = \Big[G_{i,j}\Big] = \Big[\frac{dg_i}{dm_j}\Big] \tag{2.54}$$

for $i = 1, 2, \ldots, N_d$ and $j = 1, 2, \ldots, N_m$, i.e., G is an $N_d \times N_m$ matrix. The (i, j)'s entry of G can be obtained as follows

$$\frac{dg_i}{dm_j} = \sum_{k=1}^{N_Y} \frac{\partial g_i}{\partial y_k} \frac{\partial y_k}{\partial m_j} + \frac{\partial g_i}{\partial m_j}, \tag{2.55}$$

where y_k denotes a single entry of the vector Y defined by Eq. 2.44. Using Eq. 2.55, one can write the sensitivity matrix, G, as

$$G = [\nabla_Y g^T]^T \frac{dY}{dm} + [\nabla_m g^T]^T, \tag{2.56}$$

where the matrix $\nabla_m g^T$ is an $N_m \times N_d$ matrix and defined as

$$\nabla_m g^T = \begin{bmatrix} \frac{\partial g_1}{\partial m_1} & \frac{\partial g_2}{\partial m_1} & \cdots & \frac{\partial g_{N_d}}{\partial m_1} \\ \frac{\partial g_1}{\partial m_2} & \frac{\partial g_1}{\partial m_2} & \cdots & \frac{\partial g_{N_d}}{\partial m_2} \\ \vdots & \vdots & \vdots & \vdots \\ \frac{\partial g_1}{\partial m_{N_m}} & \frac{\partial g_2}{\partial m_{N_m}} & \cdots & \frac{\partial g_{N_d}}{\partial m_{N_m}} \end{bmatrix}. \tag{2.57}$$

where the (j, i) entry of the matrix $\nabla_m g^T$, denoted as $\frac{\partial g_i}{\partial m_j}$ is the partial derivative of the ith predicted data with respect to the jth model parameter. Note that if the formula for g_i does not explicitly involve m_j, then $\frac{\partial g_i}{\partial m_j} = 0$. The detailed formulas for calculation of entries of $\nabla_m g^T$ is given in Li et al. [38].

Using Eq. 2.48 to solve for dY/dm and using the result in Eq. 2.56 gives

$$G = -[\nabla_Y g^T]^T [\nabla_Y F^T]^{-T} [\nabla_m F^T]^T + [\nabla_m g^T]^T, \tag{2.58}$$

where the superscript $-T$ denotes the dual operation of taking the transpose and then the inverse.

2.3.1 G Times a Vector

Here, we show in details how to calculate the product of G times a vector in the process of gradient simulator without explicitly computing the entries of the G. Implementation of this procedure into an IMPES reservoir simulation is also described. Multiplying Eq. 2.58 by an arbitrary N_m-dimensional column vector u gives

$$Gu = -[\nabla_Y g^T]^T [\nabla_Y F^T]^{-T} [\nabla_m F^T]^T u + [\nabla_m g^T]^T u. \tag{2.59}$$

If we define a column vector Z by

$$Z = [\nabla_Y F^T]^{-T} [\nabla_m F^T]^T u, \tag{2.60}$$

or equivalently

$$[\nabla_Y F^T]^T Z = [\nabla_m F^T]^T u. \tag{2.61}$$

then Eq. 2.59 becomes

$$Gu = -[\nabla_Y g^T]^T Z + [\nabla_m g^T]^T u. \tag{2.62}$$

Note that Z consists of L column subvectors of length N_e, the number of primary variables for each time step, so that

$$Z = [(z^1)^T, (z^2)^T, \ldots, (z^L)^T]^T. \tag{2.63}$$

Recalling the structure of $(\nabla_Y F^T)^T$ given by Eq. 2.50, the z^n's can be obtained forward in time as

$$[\nabla_{y^1}(f^1)^T]^T z^1 = [\nabla_m (f^1)^T]^T u, \tag{2.64}$$

and

$$[\nabla_{y^n}(f^n)^T]^T z^n = [\nabla_m (f^n)^T]^T u - [\nabla_{y^{n-1}}(f^n)^T]^T z^{n-1}, \tag{2.65}$$

for $n = 2, 3, \ldots, L$. Note when we solve the two preceding equations to obtain z^i's with IMPES reservoir simulator, we can use Eqs. 2.12 and 2.20 to partition the coefficients matrices. For example, we simply replace $\nabla_{y^n}(f^n)^T$ by Eq. 2.29 for $n = 1, 2, \ldots, L$. The z^n vector for the IMPES simulator in partitioned form is given by

$$z^n = [(z_p^n)^T, (z_{S_o}^n)^T, (z_{wf}^n)^T]^T, \tag{2.66}$$

for $n = 1, 2, \ldots, L$. After Z has been calculated, we can calculate Gu by rewriting Eq. 2.62 as

$$Gu = -\sum_{n=1}^{L} [\nabla_{y^n} g^T]^T z^n + [\nabla_m g^T]^T u. \tag{2.67}$$

The ith component of the vector Gu is given by

$$(Gu)_i = -\sum_{n=1}^{L} \sum_{k=1}^{N_e} \frac{\partial g_i}{\partial y_k^n} z_k^n + \sum_{j=1}^{N_m} \frac{\partial g_i}{\partial m_j} u_j. \tag{2.68}$$

Note that the partial derivatives on the right-hand side of Eq. 2.68 are computed directly from the explicit expression for the predicted data. For an IMPES reservoir simulator, we can partition $\nabla_{y^n} g^T$ by using Eq. 2.12. Therefore, we can rewrite Eq. 2.67 as

$$Gu = -\sum_{n=1}^{L} \left\{ [\nabla_{p^n} g^T]^T z_p^n + [\nabla_{S_o^n} g^T]^T z_{S_o}^n + [\nabla_{p_{wf}^n} g^T]^T z_{wf}^n \right\} + [\nabla_m g^T]^T u. \tag{2.69}$$

2.4 Lanczos Algorithm for computing a partial SVD

We implement parameterization based on singular value decomposition (SVD) of a dimensionless sensitivity matrix in order to obtain a robust algorithm in the gradient-based automatic history matching. The dimensionless sensitivity matrix, G_D, involves the sensitivity matrix G. For a large scale history matching problem, explicit construction of G is not feasible. Therefore, in our application of SVD parameterization algorithm, we need to obtain singular triplets with SVD algorithms which do not require explicit calculation of all individual entries of the sensitivity matrix. If singular values of the dimensionless sensitivity matrix decay rapidly to very small values (close to zero), one does not need to compute many singular values and it suffices to obtain a partial SVD consisting of only the largest singular values and their corresponding singular vectors because the information content of small singular triplets is typically quite low. Therefore, another desirable property of the algorithm chosen to compute SVD is that the algorithm converges quickly to the largest singular values.

To obtain the singular vectors corresponding to the largest singular values of the dimensionless sensitivity matrix, we use the Lanczos algorithm (see [24, 47, 61]) to generate the truncated singular value decomposition. The Lanczos algorithm has two main benefits. First, the algorithm iteratively approximates the largest singular values, the ones carrying most of the information content of the dimensionless sensitivity matrix. Second, this algorithm does not require explicit knowledge of the sensitivity matrix, i.e., the matrix G need not be explicitly computed. The matrix G is only referenced through matrix-vector multiplication. The Lanczos algorithm only requires calculation of the product of the sensitivity matrix with an arbitrary vector

(Gu) and the product of the transpose of the sensitivity matrix with an arbitrary vector $(G^T v)$. To obtain Gu, we use the forward (or gradient simulator) method as described in the previous section ([56]). Calculation of $G^T v$ is done with the adjoint method ([51, 56]), which was also described previously.

Next, we describe details of the Lanczos bidiagonalization algorithm. The Lanczos algorithm proceeds in two steps (see [47]). In the first step, the problem is reduced to a problem on a subspace which is computationally much more economical. This is a bidiagonalization algorithm. In this step, the original matrix is converted to a bidiagonal matrix during the fundamental recurrences that define the Lanczos bidiagonalization. In the second step, the singular triplets of the original domain are calculated based on the decomposition of the reduced problem defined by the bidiagonal matrix. In the paragraphs below we describe in further detail how the iterative bidiagonalization process may be used in the SVD calculations for the dimensionless sensitivity matrix, G_D, which were introduced initially in Zhang et al. [67] and we will define it in Chapter 3.

For a rectangular $N_d \times N_m$ matrix, G_D the Lanczos bidiagonalization computes a sequence of Lanczos vectors $u_j \in R^{N_d}$ and $v_j \in R^{N_m}$ and scalars α_j and β_j for $j = 1, 2, \ldots, k$. This can be done using very elementary manipulations; we need to form the product of G_D and G_D^T with various vectors and to take linear combinations and inner product of vectors. Furthermore, rather than doing the full bidiagonalization algorithm, the Lanczos algorithm can be terminated early to give a truncated factorization of the matrix G_D. The Lanczos bidiagonalization algorithm proceeds as follows;

1. Let v_1 be an $N_m \times 1$ starting unit vector. Compute $y = G_D v_1$ and let $\alpha_1 = \parallel y \parallel$, $u_1 = \frac{y}{\alpha_1}$.

2. For iterations $\ell = 1, 2, ...$ do

   a. $w = G_D^T u_\ell - \alpha_\ell v_\ell$

```
        • Reorthogonalize: for $i = 1, 2, ..., \ell$ do
        • $w = w - (v_i^T w)v_i$
        • end
    b. $\beta_\ell = \| w \|$
    c. $v_{\ell+1} = \frac{w}{\beta_\ell}$
    d. $y = G_D v_{\ell+1} - \beta_\ell u_\ell$
        • Reorthogonalize: for $i = 1, 2, ..., \ell$ do
        • $y = y - (u_i^T y)u_i$
        • end
    e. $\alpha_{\ell+1} = \| y \|$
    f. $u_{\ell+1} = \frac{y}{\alpha_{\ell+1}}$
end.
```

After $L-1$ steps (Lanczos iterations), we have generated the upper bidiagonal $L \times L$ matrix B_L given by

$$B_L = \begin{bmatrix} \alpha_1 & \beta_1 & & & \\ & \alpha_2 & \beta_2 & & \\ & & \ddots & \ddots & \\ & & & \alpha_{L-1} & \beta_{L-1} \\ & & & & \alpha_L \end{bmatrix}. \tag{2.70}$$

We let $V_L = [v_1, v_2, ..., v_L]$ where v_j, $j = 1, 2, \ldots, L$ are the right Lanczos vectors and $U_L = [u_1, u_2, ..., u_L]$ where u_j, $j = 1, 2, \ldots, L$ are the left Lanczos vectors. In exact arithmetic, the Lanczos vectors are orthonormal so that

$$U_L^T U_L == U_L U_L^T = I_L, \tag{2.71}$$

and

$$V_L^T V_L = V_L V_L^T = I_L, \tag{2.72}$$

where I_L is the $L \times L$ identity matrix. In the presence of roundoff errors, the orthogonality among the left and right Lanczos vectors is gradually lost so that Eqs. 2.71 and 2.72 no longer hold. The reorthogonalize steps in the algorithm prevent significant accumulation of round off error. Based on the above Lanczos bidiagonalization algorithm (see step 2, a and c, and d and f), we can write the following recurrences

$$\alpha_{\ell+1} u_{\ell+1} = G_D v_{\ell+1} - \beta_\ell u_\ell, \tag{2.73}$$

and

$$\beta_\ell v_{\ell+1} = G_D^T u_\ell - \alpha_\ell v_\ell. \tag{2.74}$$

We can write Eqs. 2.73 and 2.74 in compact matrix form as

$$G_D V_L = U_L B_L, \tag{2.75}$$

and

$$G_D^T U_L = V_L B_L^T + \beta_L v_{L+1}. \tag{2.76}$$

Using the orthogonality property of Eq. 2.71, we can rewrite Eq. 2.75 as follows

$$U_L^T G_D V_L = B_L. \tag{2.77}$$

It can be shown [23] that the matrix B_L has singular values that are close to certain singular values of matrix G_D, typically the largest and smallest singular values of G_D. Having obtained the matrix B_L with the Lanczos bidiagonalization algorithm, the estimates for the singular values and corresponding singular vectors of G_D can be obtained by calculation of the SVD of the matrix B_L, i.e.,

$$B_L = \tilde{U}_L \Lambda_L \tilde{V}_L^T, \tag{2.78}$$

where Λ_L is a $L \times L$ diagonal matrix with diagonal entries denoted by λ_j^ℓ, $j = 1, 2, ..., L$. Using Eq. 2.78 in Eq. 2.77 and performing simple algebraic manipulation gives

$$G_D = U_L \tilde{U}_L \Lambda_L \tilde{V}_L^T V_L^T = U \Lambda_L V^T \tag{2.79}$$

where, $U = U_L \tilde{U}_L$ and $V = V_L \tilde{V}_L$. The diagonal entries of Λ_L are the approximations to the L largest singular values of G_D. The columns of U and V approximate the corresponding left and right singular vectors [61].

2.4.1 Stopping conditions of Lanczos Algorithm

The number of Lanczos iterations depends on; i) the assigned number of retained converged singular values, i.e., the level of truncation of singular value decomposition (p) and ii) the assigned singular cutoff (sv-cut). The singular cutoff, sv-cut, is defined as the ratio of the smallest converged singular value to the largest converged singular value. Since with Lanczos algorithm, the first obtained singular value is the largest one, therefore we can write

$$\text{sv-cut} = \frac{\lambda_j^\ell}{\lambda_1^\ell}. \tag{2.80}$$

With a specified value of the singular cutoff, we terminate the Lanczos algorithm at the smallest value of p such that $\frac{\lambda_p}{\lambda_1} \leq$ sv-cut. To obtain converged singular values, we do iterations of Lanczos algorithm and check the convergence criterion. For convergence of each of the singular values we require that the relative change in the singular values from current iteration to the next iteration be small, i.e.,

$$\frac{|\lambda_j^\ell - \lambda_j^{\ell-1}|}{\lambda_j^\ell} \le \epsilon_{sv} \tag{2.81}$$

where ℓ is the iteration index and the λ_j^ℓ's are the approximate singular values at iteration ℓ. At convergence, Eq. 2.81 should be satisfied for $j = 1, 2, ..., p$. In our applications, we use $\epsilon_{sv} = 10^{-5}$.

Therefore, the three input parameters of the Lanczos algorithm which control the computational cost of the algorithm are p, sv-cut and ϵ_{sv}. The strategy of using the Lanczos bidiagonalization algorithm to compute the partial SVD with truncation level p is that for iterations $\ell = 1, 2, ..., p$, we perform only the bidiagonalization algorithm. Then, for further iterations ($\ell > p$), in addition to the bidiagonalization process, we use Eq. 2.78 to compute singular values of the matrix B_L and we perform the convergence check given by Eq. 2.81. If the condition given by Eq. 2.81 holds or if $\frac{\lambda_j^\ell}{\lambda_1^\ell} <$ sv-cut, then we use Eq. 2.79 to calculate the right and left singular vectors of G_D and terminate the Lanczos algorithm.

In the implementation of the Lanczos algorithm, the computational cost of applying matrix-vector multiplication to calculate $G_D v_j$ and $G_D^T u_j$ far exceeds other computational costs. The Lanczos algorithm with k iterations requires $k+1$ forward gradient runs to obtain $G_D v_j$ and k adjoint runs to obtain $G_D^T u_j$.

CHAPTER 3

MAP ESTIMATE WITH SVD PARAMETERIZATION

In this chapter, we present the basic equations and notation for the history matching problem in a Bayesian setting by developing an equation for the posterior probability density function (pdf). We discuss the computation of the maximum a posteriori (MAP) estimate of reservoir model parameters which is the model that maximizes the posterior pdf. Based on the linearization of the posterior pdf in the neighborhood of the MAP estimate, we provide a theoretical argument which indicates that parameterization based on the principle right singular vectors of the dimensionless sensitivity matrix provides an optimal basis for parameterization of the vector of model parameters. We also develop parameterization algorithms based on the singular value decomposition (SVD) of the dimensionless sensitivity matrix to compute the MAP estimate. In the last section of this chapter, we present the computational results of the MAP estimate with different SVD parameterization algorithms for two 2-dimensional synthetic examples.

3.1 The Posterior Probability Density Function

In the Bayesian domain, suppose that our N_m-dimensional random column vector of model parameters m are uncertain, and that we can represent the uncertainty through an estimate of the prior mean m_{prior} and the $N_m \times N_m$ prior model covariance matrix, C_M. The prior uncertainty in the model parameters is described by the following Gaussian probability density function (pdf)

$$f(m) = a_1 \exp\big[-\frac{1}{2}(m - m_{\text{prior}})^T C_M^{-1} (m - m_{\text{prior}})\big], \tag{3.1}$$

The theoretical relationship between an N_d-dimensional column vector of predicted data d and the vector m of model parameters is given by

$$d = g(m), \tag{3.2}$$

If m is the true model parameters, then the difference between d and corresponding vector of observed data, d_{obs}, represents measurement error ϵ^d, i.e.,

$$\epsilon^d = d - d_{\text{obs}}, \tag{3.3}$$

Assuming measurement error is Gaussian with mean zero and an $N_d \times N_d$ covariance matrix C_D, then the probability of observed data d_{obs} given the model parameters is simply the probability of measurement error ϵ^d, i.e.,

$$f(d_{\text{obs}}|m) = f(\epsilon^d) = a_2 \exp\Big[-\frac{1}{2}(d_{\text{obs}} - g(m))^T C_D^{-1}(d_{\text{obs}} - g(m))\Big], \tag{3.4}$$

Note if we assume d_{obs} is given, then Eq. 3.4 gives the likelihood of m given d_{obs} denoted by $L(m|d_{\text{obs}})$. By Bayes Theorem [59, 51], the posterior pdf of the model parameters conditional to the observed data is proportional to the product of the prior pdf and the likelihood function for the model parameters, i.e.

$$f(m|d_{\text{obs}}) \propto f(m) L(m|d_{\text{obs}}). \tag{3.5}$$

Using Eqs. 3.1 and 3.4 in Eq. 3.5 gives the posterior pdf of model parameters conditioned to observed data as follows

$$f(m|d_{\text{obs}}) = a \exp(-O(m)), \tag{3.6}$$

where a is the normalizing constant and

$$O(m) = \frac{1}{2}(m - m_{\text{prior}})^T C_M^{-1}(m - m_{\text{prior}}) + \frac{1}{2}(g(m) - d_{\text{obs}})^T C_D^{-1}(g(m) - d_{\text{obs}}), \quad (3.7)$$

$O(m)$ in Eq. 3.7 is referred to as the total objective function.

3.2 Approximate Posterior Covariance Matrix

Let m_{MAP}, denote the MAP estimate obtained by minimizing the objective function of Eq. 3.7. As m_{MAP} corresponds to a minimum of $O(m)$, the gradient must be zero at the MAP estimate, i.e.,

$$\nabla O(m_{\text{MAP}}) = C_M^{-1}(m_{\text{MAP}} - m_{\text{prior}}) + G_{\text{MAP}}^T C_D^{-1}(g(m_{\text{MAP}}) - d_{\text{obs}}) = 0, \quad (3.8)$$

where G_{MAP} denotes the sensitivity matrix evaluated at m_{MAP}. Using standard matrix inversion lemmas [59, 24], it follows easily that

$$\begin{aligned} m_{\text{MAP}} = m_{\text{prior}} - C_M G_{\text{MAP}}^T (C_D + G_{\text{MAP}} C_M G_{\text{MAP}}^T)^{-1} & \\ [g(m_{\text{MAP}}) - d_{\text{obs}} - G_{\text{MAP}}(m_{\text{MAP}} - m_{\text{prior}})]. & \quad (3.9) \end{aligned}$$

In some neighborhood of m_{MAP}, $g(m)$ is well approximated by the following first order Taylor series:

$$g(m) = g(m_{\text{MAP}}) + G_{\text{MAP}}(m - m_{\text{MAP}}). \quad (3.10)$$

When Eqs. 3.8 and 3.10 hold, the following second order Taylor's series expansion holds exactly:

$$O(m) = O(m_{\text{MAP}}) + \frac{1}{2}(m - m_{\text{MAP}})^T H_{\text{MAP}}(m - m_{\text{MAP}}), \quad (3.11)$$

where H_{MAP} denotes the Hessian evaluated at the MAP estimate of the model, i.e.,

$$H_{\mathrm{MAP}} = C_M^{-1} + G_{\mathrm{MAP}}^T C_D^{-1} G_{\mathrm{MAP}}. \tag{3.12}$$

Using Eq. 3.11 in Eq. 3.6 gives the following approximation of the posterior pdf

$$f(m|d_{\mathrm{obs}}) = \hat{a} \exp\Big[-\frac{1}{2}(m - m_{\mathrm{MAP}})^T H_{\mathrm{MAP}} (m - m_{\mathrm{MAP}}) \Big], \tag{3.13}$$

where $\hat{a}$ is equal to the original normalizing constant a multiplied by $\exp\big(-O(m_{\mathrm{MAP}})\big)$. Eq. 3.13 is in the form of a Gaussian pdf with mean m_{MAP} and (posterior) covariance matrix given by

$$C_{\mathrm{MAP}} = H_{\mathrm{MAP}}^{-1} = (C_M^{-1} + G_{\mathrm{MAP}}^T C_D^{-1} G_{\mathrm{MAP}})^{-1}. \tag{3.14}$$

Eq. 3.13 applies only in the neighborhood of m_{MAP} where the approximation of Eq. 3.10 is accurate and in other regions of the model space, the posterior pdf may be very different. However, if Eq. 3.10 is exact for all m, i.e., the relation between the model m and predicted data $g(m)$ is linear, then Eq. 3.13 holds for all m so the posterior pdf is Gaussian. Finally, we note that using the matrix inversion lemmas [59], the posterior covariance matrix of Eq. 3.14 can be rewritten as

$$C_{\mathrm{MAP}} = C_M - C_M G_{\mathrm{MAP}}^T (G_{\mathrm{MAP}} C_M G_{\mathrm{MAP}}^T + C_D)^{-1} G_{\mathrm{MAP}} C_M. \tag{3.15}$$

3.3 Normalized Variance

The diagonal entries of C_M and C_{MAP}, respectively, represent the prior and posterior variances of the ith model parameter, m_i, for $i = 1, 2, \cdots N_m$. Denoting the diagonal entries of C_M and C_{MAP} by $c_{i,i}$ and $c'_{i,i}$, respectively, we define the ith normalized posterior variance by

$$\mathrm{var}_{n,i} \equiv \frac{c'_{i,i}}{c_{i,i}} = 1 - \frac{b_{i,i}}{c_{i,i}}. \tag{3.16}$$

for $i = 1, 2, \cdots N_m$, where $b_{i,i}$ is the ith diagonal entry of the matrix B defined by

$$B = C_M G_{\mathrm{MAP}}^T (G_{\mathrm{MAP}} C_M G_{\mathrm{MAP}}^T + C_D)^{-1} G_{\mathrm{MAP}} C_M. \tag{3.17}$$

The diagonal entries of B are nonnegative because B is real-symmetric positive semi-definite. C_{MAP} is clearly real symmetric positive definite and hence its diagonal entries, which are given by $c'_{i,i} = c_{i,i} - b_{i,i} > 0$. From this fact and the fact noted above that $b_{i,i} \geq 0$, it follows that

$$0 \leq b_{i,i} < c_{i,i} \text{ for } i = 1, 2, \cdots N_m, \tag{3.18}$$

and

$$0 < \mathrm{var}_{n,i} \leq 1 \text{ for } i = 1, 2, \cdots N_m, \tag{3.19}$$

Note the normalized variance gives a measure of the reduction in the variance obtained by integrating the observed data. If, for example, the normalized variance of m_i is 1, then we have not reduced uncertainty by conditioning to production data, whereas if its value is 0.5, we have obtained a 50% reduction in uncertainty by conditioning the prior model to the observed data, d_{obs}. Several authors [22, 27, 37, 36] have used the normalized variance in some form, e.g., the sum of the normalized variances, to measure the reduction in uncertainty obtained by integrating production data. However, considering only the reduction in uncertainty due to a reduction in the variances neglects the reduction in uncertainty due to a higher correlation between model parameters that often results from integrating production data. To obtain a better indication of how conditioning to production data modifies uncertainty in the model parameters, Zhang et al. [67] introduced dimensionless sensitivity coefficients. They, however, did not provide a precise characterization of the reduction of uncertainty. We provide this characterization below.

3.4 Dimensionless Sensitivity Matrix

The dimensionless sensitivity matrix G_D is defined by

$$G_D = C_D^{-1/2} G C_M^{1/2}, \tag{3.20}$$

where G is the sensitivity coefficient evaluated at some particular m, $C_M^{1/2}$ denotes the square root of the covariance matrix and $C_D^{-1/2}$ denotes the inverse of the square root of C_D. For the purpose of characterizing the change in uncertainty obtained by conditioning to data, any square root will suffice. In our applications, we use the square roots obtained by the Cholesky decomposition. Here, we use $C_M^{1/2}$ to denote "the square root of C_M and write

$$C_M = C_M^{1/2} C_M^{T/2}, \tag{3.21}$$

where $C_M^{T/2}$ denotes the transpose of $C_M^{1/2}$. If $C_M = LL^T$ is the Cholesky decomposition of C_M where L is lower triangular, then $C_M^{1/2} = L$. If the Schur decomposition (eigenvalue-eigenvector decomposition) is used to generate the square root, then $C_M^{T/2} = C_M^{1/2}$ and $C_M = (C_M^{1/2})^2$, a result which better fits our notion of a square root. From Eq. 3.21, it follows that

$$C_M^{-1} = C_M^{-T/2} C_M^{-1/2}, \tag{3.22}$$

where $C_M^{-T/2}$ represents the inverse of $C_M^{T/2}$. Similarly,

$$C_D = C_D^{1/2} C_D^{T/2}, \tag{3.23}$$

and

$$C_D^{-1} = C_D^{-T/2} C_D^{-1/2}. \tag{3.24}$$

Introducing the square roots of the covariance matrices and letting G and G_D represent the sensitivity and dimensionless sensitivity matrices evaluated at the MAP estimate, and letting I_{N_m} and I_{N_d}, respectively, denote the $N_m \times N_m$ and $N_d \times N_d$ identity matrices, Eq. 3.15 can be rewritten as

$$\begin{aligned} C_{\text{MAP}} &= C_M^{1/2}\Big(I_{N_m} - C_M^{T/2} G^T C_D^{-T/2}\Big[C_D^{-1/2} G C_M G^T C_D^{-T/2} + I_{N_d}\Big]^{-1} C_D^{-1/2} G C_M^{1/2}\Big) C_M^{T/2} \\ &= C_M^{1/2}\Big(I_{N_m} - G_D^T\Big[G_D G_D^T + I_{N_d}\Big]^{-1} G_D\Big) C_M^{T/2} \end{aligned}. \tag{3.25}$$

3.5 Confidence Regions

Assuming the posterior pdf is Gaussian (Eq. 3.13) with covariance matrix C_{MAP}, a surface of the form

$$(m - m_{\text{MAP}})^T C_{\text{MAP}}^{-1} (m - m_{\text{MAP}}) = r^2 \tag{3.26}$$

is a surface of constant probability density and the interior of this ellipsoid represents a confidence region [4]. The volume of this ellipsoid is given by

$$V' = \frac{\sqrt{r^2 \pi^{N_m}}}{\Gamma(1 + (N_m/2))} \sqrt{\det C_{\text{MAP}}}, \tag{3.27}$$

where Γ is the Gamma (generalized factorial) function. For a fixed value of r^2, the volume of this ellipsoid reflects the uncertainty in m. The smaller the volume of this ellipsoid, the smaller the uncertainty in m_{MAP} as an estimate of the true model, or the more likely that a sample of the pdf of Eq. 3.13 will be close to m_{MAP}. For the same value of r^2, the corresponding volume of the ellipsoid

$$(m - m_{\text{prior}})^T C_{\text{M}}^{-1} (m - m_{\text{prior}}) = r^2 \tag{3.28}$$

is given by

$$V = \frac{\sqrt{r^2 \pi^{N_m}}}{\Gamma(1 + (N_m/2))} \sqrt{\det C_M}. \tag{3.29}$$

The ratio of V' to V represents the reduction in uncertainty obtained by conditioning the prior model to d_{obs}. From Eqs. 3.27, 3.29 and 3.25, it follows that

$$\frac{V'}{V} = \sqrt{\frac{\det C_{\mathrm{MAP}}}{\det C_M}} = \sqrt{\det \left(I_{N_m} - G_D^T \left[G_D G_D^T + I_{N_d} \right]^{-1} G_D \right)}. \tag{3.30}$$

Next we show that the characterization of Eq. 3.30 can be rewritten in terms of the singular values of the dimensionless sensitivity matrix.

3.6 Reduction in Uncertainty in Terms of Singular Values of Dimensionless Sensitivity Matrix

In the discussion of this section, we assume that $N_d < N_m$ as this is almost always the case in history matching problems of interest. This assumption simply allows us to be more specific regarding the SVD of the $N_d \times N_m$ dimensionless sensitivity matrix, G_D. However, the results presented here can also be established in $N_d \geq N_m$.

The critical properties of singular value decomposition (SVD) needed for our discussion can be found in Golub and van Loan [24] and are summarized in Appendix A. Here, we let u_i and v_i, respectively denote the left and right singular vector corresponding to the singular value λ_i determined from a SVD of the dimensionless sensitivity matrix G_D. Eq. A.14 gives

$$\left(I_{N_d} + G_D G_D^T \right)^{-1} u_i = \frac{1}{(1 + \lambda_i^2)} u_i, \quad \text{for } i = 1, 2, \cdots N_d, \tag{3.31}$$

so the eigenvalue-eigenvector pairs of the $N_d \times N_d$ matrix $\left(I_{N_d} + G_D G_D^T \right)$ are $(1 + \lambda_i^2, u_i)$ for $i = 1, 2 \cdots N_d$ where $\lambda_1^2 \geq \lambda_2^2 \geq \cdots \geq \lambda_{N_d}^2 \geq 0$. From Eqs. A.15 and A.16, it follows easily that the eigenvalue-eigenvector pairs of the $N_m \times N_m$ matrix

$I_{N_m} - G_D^T\big(I_{N_d} + G_D G_D^T\big)^{-1} G_D$ are $\big\{\beta_i, v_i\big\}_{i=1}^{N_m}$ where

$$\beta_i = \begin{cases} 1 - \frac{\lambda_i^2}{(1+\lambda_i^2)} \text{ for } i = 1, 2, \cdots N_d \\ 1 \qquad \text{for } i = N_d + 1, \cdots N_m. \end{cases} \tag{3.32}$$

As the determinant of a matrix is equal to the product of its eigenvalues, it follows that

$$\det\Big(I_{N_m} - G_D^T\big(I_{N_d} + G_D G_D^T\big)^{-1} G_D\Big) = \prod_{i=1}^{N_m} \beta_i = \prod_{i=1}^{N_d} \frac{1}{1+\lambda_i^2}. \tag{3.33}$$

Using Eq. 3.33 in Eq. 3.30 gives

$$\frac{V'}{V} = \sqrt{\frac{\det C_{\text{MAP}}}{\det C_M}} = \sqrt{\prod_{i=1}^{N_d} \frac{1}{1+\lambda_i^2}}, \tag{3.34}$$

where λ_i, $i = 1, 2, \cdots N_d$ are the singular values of the dimensionless sensitivity matrix, G_D. Eq. 3.34 indicates that the singular values of the dimensionless sensitivity matrix determine the reduction in uncertainty in m obtained by conditioning to observed data and very small singular values have a small effect on the reduction in uncertainty obtained by conditioning to data. For $N_d \leq N_m$, there can be at most N_d nonzero singular values. These results suggest that the right singular vectors corresponding to the largest singular values may provide an optimal parameterization of the change in the model during an iteration of a gradient-based optimization algorithm. Moreover, if the singular values of the dimensionless sensitivity matrix decay rapidly, we may need only a few singular vectors in the parameterization. Thus, the development given above provides theoretical support for the parameterization method used by Rodrigues [57].

3.7 Levenberg-Marquardt for SVD Parameterization

We use a gradient-based optimization algorithm to minimize the objective function of Eq. 3.7. For large scale problems, we typically minimize $O(m)$ by the implementation of the limited memory Broyden-Fletcher-Goldfarb-Shanno (LBFGS) method discussed in Zhang and Reynolds [66] and Gao and Reynolds [19] but in this study we use a modified Levenberg-Marquardt (LM) [35, 43] algorithm described in Oliver et al. [51] for minimization. In order to avoid overshooting and undershooting in the final model, it is sometimes necessary to control the magnitude of change in model parameters over an iteration at least during the early iterations [38, 20]. For this purpose we use the modified LM algorithm [35, 43] which is basically a regularization procedure.

In generating the MAP estimate, the search direction in the modified LM algorithm is calculated as

$$\delta m^{l+1} = -[(1+\gamma_l)C_M^{-1} + G_l^T C_D^{-1} G_l]^{-1}\{C_M^{-1}(m^l - m_{\text{prior}}) + G_l^T C_D^{-1}(g(m^l) - d_{obs})\}. \tag{3.35}$$

where l denotes the iteration index and γ_l denotes LM parameter. The preceding LM can be applied successfully [38] if the number of data is sufficiently small so that it is computationally feasible to calculate all entries of the sensitivity matrix with an adjoint method. Here, following Rodrigues [57], we seek to improve computational efficiency by expanding $\delta \tilde{m}^{l+1}$ in terms of the right singular vectors of the dimensionless sensitivity matrix at each iteration for cases where the computation of all entries of the sensitivity matrix is not feasible.

Using the Cholesky decomposition of C_M given by $C_M = LL^T$, it follows that $C_M^{-1} = L^{-T}L^{-1}$ where L^{-T} denotes the inverse of L^T. C_D could be factored in the same way but as C_D is diagonal in the examples considered here, we simply use $C_D^{-1} = C_D^{-\frac{1}{2}} C_D^{-\frac{1}{2}}$. We let $G_{D,l}$ denote the dimensionless sensitivity matrix corresponding to G_l, i.e., $G_{D,l} = C_D^{-\frac{1}{2}} G_l L$. Then, we left multiply Eq. 3.35 by L^{-1} and simplify to obtain

$$L^{-1}\delta m^{l+1} =$$

$$-L^{-1}[(1+\gamma_l)L^{-T}L^{-1}+G_l^T C_D^{-\frac{1}{2}} C_D^{-\frac{1}{2}} G_l]^{-1}\{L^{-T}L^{-1}(m^l - m_{\text{prior}}) + G_l^T C_D^{-1}(g(m^l) - d_{obs})\}$$

$$= -[(1+\gamma_l)I_{N_m} + G_D^T G_D]^{-1}\{L^{-1}(m^l - m_{\text{prior}}) + G_D^T C_D^{-\frac{1}{2}}(g(m^l) - d_{obs})\}, \quad (3.36)$$

where I_{N_m} denotes $N_m \times N_m$ identity matrix. Defining the transformed model, $\tilde{m}^l$, by

$$\tilde{m}^l = L^{-1}(m^l - m_{\text{prior}}) \quad (3.37)$$

for iteration l, we must also have

$$\delta\tilde{m}^{l+1} = \tilde{m}^{l+1} - \tilde{m}^l = L^{-1}(m^{l+1} - m^l) = L^{-1}\delta m^{l+1}. \quad (3.38)$$

Using this notation, Eq. 3.36 becomes

$$\delta\tilde{m}^{l+1} = -[(1+\gamma_l)I_{N_m} + G_{D,l}^T G_{D,l}]^{-1}\{\tilde{m}^l + G_{D,l}^T C_D^{-\frac{1}{2}}(g(m^l) - d_{obs})\}, \quad (3.39)$$

or

$$[(1+\gamma_l)I_{N_m} + G_{D,l}^T G_{D,l}]\delta\tilde{m}^{l+1} = -\{\tilde{m}^l + G_{D,l}^T C_D^{-\frac{1}{2}}(g(m^l) - d_{obs})\}. \quad (3.40)$$

For the case where the number of data N_d is less than the number of model parameters N_m, it follows from Eqs. A.11 and A.12, that

$$[(1+\gamma_l)I_{N_m} + G_{D,l}^T G_{D,l}]v_i = \begin{cases} (1+\gamma_l+\lambda_i^2)\, v_i & \text{if} \quad 1 \le i \le N_d \\ (1+\gamma_l)v_i & \text{if} \quad N_d < i \le N_m. \end{cases} \tag{3.41}$$

Thus, the eigenvalue-eigenvector pairs of the $N_m \times N_m$ matrix $[(1+\gamma_l)I_{N_m}+G_{D,l}^T G_{D,l}]$ are $(1+\gamma_l+\lambda_i^2, v_i)$ for $i = 1, 2 \cdots N_d$ and $(1+\gamma_l, v_i)$ for $i = N_d+1, \cdots N_m$.

We seek an approximation to $\delta\tilde{m}^{l+1}$ contained in the subspace spanned by the right singular vectors. Specifically, we write the change in $\tilde{m}$ as a linear combination of the right singular vectors corresponding to the p largest singular values, i.e.,

$$\delta\tilde{m}^{l+1} = \sum_{j=1}^{p} \alpha_j v_j. \tag{3.42}$$

where we assume that $p \le N_d$. Substituting Eq. 3.42 into the left hand side of Eq. 3.40, using the truncated SVD of $G_{D,l}$ in the form of $G_{D,l}^T = V_p \Lambda_p^T U_p^T$, and using Eq. 3.41, Eq. 3.40 can be approximated by

$$\sum_{j=1}^{p} \alpha_j (1+\gamma_l+\lambda_j^2) v_j = -\{\tilde{m}^l + V_p \Lambda_p^T U_p^T C_D^{-\frac{1}{2}} (g(m^l) - d_{\text{obs}})\}, \tag{3.43}$$

where $p \le N_d$. Left multiplying Eq. 3.43 by v_k^T and using the orthogonality of singular vectors, it follows easily that

$$\alpha_k = \frac{-v_k^T \tilde{m}^l - \lambda_k u_k^T C_D^{-\frac{1}{2}} (g(m^l) - d_{obs})}{1+\gamma_l+\lambda_k^2} \tag{3.44}$$

for $k = 1$ to p. Using Eq. 3.44 in Eq. 3.42, yields

$$\delta\tilde{m}^{l+1} = \sum_{j=1}^{p} \Big[\frac{-v_j^T \tilde{m}^l - \lambda_j u_j^T C_D^{-\frac{1}{2}} (g(m^l) - d_{obs})}{1+\gamma_l+\lambda_j^2} \Big] v_j. \tag{3.45}$$

So from Eq. 3.38,

$$\delta m^{l+1} = L\delta \tilde{m}^{l+1} = L \sum_{j=1}^{p} \Big[\frac{-v_j^T \tilde{m}^l - \lambda_j u_j^T C_D^{-\frac{1}{2}} (g(m^l) - d_{obs})}{1 + \gamma_l + \lambda_j^2} \Big] v_j. \tag{3.46}$$

Once δm^{l+1} has been computed, we can compute new model parameter from

$$m^{l+1} = m^l + \delta m^{l+1}. \tag{3.47}$$

The parameter γ_l varies from iteration to iteration based on the variation of objective function. We use a simple procedure to obtain the value of γ_l for each iteration. If $O(m^{l+1}) < O(m^l)$, we accept m^{l+1} and $\tilde{m}^{l+1}$ as the new models in original and transformed space, respectively, decrease γ_l by a factor of 10 for the next iteration ($\gamma_l = \gamma_l/10$). Otherwise, we increase γ_l by a factor of 10 and redo the iteration. This is repeated until we obtain convergence. As suggested in Oliver et al. [51], the initial value of γ_l should be between $\sqrt{O(m^0)/N_d}$ and $O(m^0)/N_d$, where $O(m^0)$ is the initial value of the objective function. We use the larger of these two values. After applying Eq. 3.45, we apply Eq. 3.46 to obtain the search direction in the original domain and then update the vector of original model parameters by Eq. 3.47.

At each iteration of Eq. 3.45, the truncated SVD must be updated, which is computationally expensive. As we mentioned in Chapter 2, we use Lanczos algorithm to compute the singular triplets of the dimensionless sensitivity matrix. Moreover, the computational cost of the Lanczos increases linearly with the level of truncation, p. In the examples, presented here, at each iteration, the truncated SVD is based on specification of the value of sv-cut.

3.8 More Efficient Algorithms Based on SVD parameterization

Even though truncated SVD for parameterization appears to provide a viable approach for reducing the computational cost of history matching, the calculation and updating of singular triplets iteratively with the Lanczos algorithm, involves a significant cost as each iteration of Lanczos requires one forward simulation solution to obtain $G_{D,l}x$ and one adjoint solution to obtain $G_{D,l}^T y$. To obtain p singular triplets, the Lanczos algorithm requires approximately $p + n$ iterations of Lanczos. The value of n depends on the magnitude of assigned relative error (ϵ_{sv}) for convergence of each of the singular values. The details on convergence criteria for the Lanczos algorithm are given in Chapter 2. Therefore, each call of Lanczos requires roughly the equivalent of $(p+n)/2$ reservoir simulation runs. In all examples we have done, we have used $\epsilon_{sv} = 10^{-5}$. With this choice, the value of n varied from 3 to 7. Therefore, we can not afford to do extensive updates of the singular triplets during the minimization or the SVD will become more computationally expensive than a quasi-Newton method. Great savings in computationally efficiency can be achieved if the number of truncated singular value decompositions is significantly reduced. In the following two subsections, we present two new algorithms which try to enhance efficiency of the SVD parameterization algorithm.

3.8.1 First Modified Algorithm; LBFGS and SVD Parameterization

One option for obtaining a computationally efficient method is to use a quasi-Newton method, such as LBFGS, and SVD parameterization as a combined algorithm. When downhill conditions are satisfied, each iteration of LBFGS requires one forward simulation run and one adjoint solution. Therefore, the computational cost of each iteration of LBFGS algorithm is much less than the cost of one iteration of SVD parameterization algorithm. In our computational examples in this dissertation, we have used the LBFGS algorithm described by Zhang and Reynolds [66] with the line search algorithm given by Gao and Reynolds [19]. In this approach, the periodic use of truncated SVD keeping 10 to 20 singular triplets may ameliorate the effects of ill-conditioning by smoothing results whereas the LBFGS algorithm promotes computationally efficiency.

We have tried various combinations of the two algorithms. Unfortunately, we have been able to achieve at best only modest improvements in computational efficiency and thus, we present here only a simple combination of the two algorithms. Far more computationally efficient procedures are presented in the next chapter. In this method, we start the minimization by using the LBFGS algorithm and continue the LBFGS algorithm until we obtain a value of the normalized objective function less than 100. Then we do one iteration of SVD parameterization to update the model parameter using Eq. 3.45 and then switch to the LBFGS algorithm again. The normalized objective function is denoted by $O_N(m)$ and as defined by Gao et al. [20] for the MAP estimate is given by

$$O_N(m) = \frac{2 \times O(m)}{N_d}. \tag{3.48}$$

Note that in all the examples in this study for the LBFGS algorithm we have used the prior covariance matrix as an initial guess for the inverse of the Hessian matrix, i.e., $\tilde{H}_0^{-1} = C_M$. Also, in the first call of the LBFGS algorithm, we terminate the LBFGS when $O_N(m) \leq 100$, at which point, we compute the singular triplets for the first time in the combined algorithm. Using the singular triplets, we update the model using Eq. 3.45. Then we switch to the LBFGS algorithm again with the initial guess equal to the updated model obtained by the SVD algorithm and $\tilde{H}_0^{-1} = C_M$. We continue the LBFGS algorithm as long as

$$\frac{O(m^l) - O(m^{l+1})}{O(m^l)} \times 100 > 0.1, \tag{3.49}$$

is satisfied. If Eq. 3.49 is not satisfied, and the convergence criteria are not satisfied, we use the SVD algorithm again to update the model parameters once and then again switch back to the LBFGS algorithm. The condition given by Eq. 3.49 implies that when the relative change in the objective function is small, we should switch to the SVD algorithm if we still have a high value of the objective function. Throughout we refer to this combined algorithm simply as LBFGS-SVD.

3.8.2 Second Modified Algorithm

In this algorithm, to further improve computational efficiency of the SVD parameterization algorithm, we add an inner iteration in which we explicitly compute the gradient of the objective function with the adjoint method, where during the inner iteration the truncated SVD parameterization (computed in the outer loop) does not vary.

The right and left hand sides of Eq. 3.40, respectively, represent the gradient of the objective function and the "Levenberg-Marquardt Hessian" for minimization in terms of the transformed model parameter $\tilde{m}$, Eq. 3.37. By using the actual definition of the dimensionless sensitivity matrix, $G_{D,l}^T$, given by Eq. 3.20 as $G_{D,l}^T = L^T G_l^T \tilde{C}_D^{-\frac{1}{2}}$, in the right hand side of Eq. 3.40, we obtain

$$[(1+\gamma_l)I_{N_m} + G_{D,l}^T G_{D,l}]\delta\tilde{m}^{l+1} = -\{\tilde{m}^l + L^T G_l^T \tilde{C}_D^{-1}(g(m^l) - d_{obs})\}. \tag{3.50}$$

In the inner loop, we use a fixed set of SVD-triplets to evaluate the Hessian at each iteration, but update the gradient on the right-hand side. Note that here we introduce the possibility of damping the model changes by artificially inflating the noise level along the lines suggested by Gao and Reynolds [19] which is effectively done by replacing the measurement error covariance matrix C_D by a matrix $\tilde{C}_D \geq C_D$ where this inequality refers to an element by element inequality. We use $\tilde{C}_D$ to restrict the size of the change in the model parameters and to avoid overshooting/undershooting problems [62, 19]. Using the eigenvalue-eigenvector properties given in Eq. 3.41 and expanding $\delta\tilde{m}^{l+1}$ as a linear combination of right singular vectors, Eq. 3.42, it follows easily from Eq. 3.50 that

$$\delta\tilde{m}^{k+1} = \sum_{j=1}^{p}\left[\frac{-v_j^T\tilde{m}^k - v_j^T L^T G_k^T \tilde{C}_D^{-1}(g(m^k) - d_{\text{obs}})}{1+\gamma_k+\lambda_j^2}\right]v_j, \tag{3.51}$$

where in this inner loop iteration, k denotes the iteration index. Note this inner iteration also uses an LM algorithm with parameter γ_k for minimization. In Eq. 3.51, we need to calculate the product of the transpose of sensitivity matrix, G_k^T, with the vector $\tilde{C}_D^{-1}(g(m^k) - d_{\text{obs}})$. This product can be obtained with one adjoint solution [66].

For the examples considered in this dissertation, C_D is a diagonal matrix and following Gao and Reynolds [19], we specify different damping factors for each data based on the difference between the observed data and predicted data, i.e., the matrix $\tilde{C}_D$ is obtained by

$$\tilde{C}_D = \Psi C_D \Psi \tag{3.52}$$

where Ψ is a diagonal matrix with its diagonal elements equal to the damping factors, ψ_i for $i = 1, 2, \cdots, N_d$. The ith damping factor, ψ_i, is calculated by

$$\psi_i = \max\left[1, \left|\frac{g_i(m^0) - d_{\text{obs},i}}{3\sigma_{d,i}}\right|\right], \tag{3.53}$$

where $\sigma_{d,i}$ denotes the standard deviation of the ith measurement error, m^0 is the last updated model obtained at the outer iteration, $g_i(m^0)$ is the ith component of the predicted data vector evaluated at m^0 and $d_{\text{obs},i}$ denotes the ith entry of the observed data vector d_{obs}. In applying Eq. 3.51 during inner loop iterations for $k = 0, 1, 2, \cdots$, $\tilde{C}_D$ does not change from iteration to iteration. Therefore, we effectively trying to find model parameter, m^{k+1}, that minimize the "damped" objective function

$$O_{\mathrm{damp}}(m) = \frac{1}{2}(m - m_{\mathrm{prior}})^T C_M^{-1}(m - m_{\mathrm{prior}}) + \frac{1}{2}(g(m) - d_{\mathrm{obs}})^T \tilde{C}_D^{-1}(g(m) - d_{\mathrm{obs}}), \tag{3.54}$$

using as the initial guess the model, m^0 obtained from the outer iteration. As we wish to use the correct data covariance, C_D, in constructing the final estimate of model parameters, we use this damping strategies in the inner loop as long as the value of the original normalized objective function (Eq. 3.48) is greater than 3. Otherwise, we use the original data covariance matrix in the inner loop and we minimize the original objective function to construct the final estimate of the model parameters. This modified SVD parameterization algorithm is referred to as the SVD-Gradient algorithm which is denoted simply by SVD-Grd.

3.8.3 Steps of the SVD-Grd Algorithm

Here, the detail of specific steps for the implementation of the SVD-Grd algorithm is given. After its presentation, we will show how to convert the steps to the basic SVD parameterization algorithm.

1. Here l is the iteration index for the outer loop. Set $l = 0$ and assign the initial guess of $m^0 = m_{\mathrm{prior}}$ for the MAP estimate. Set the initial value of the sv-cut.

2. To obtain the initial value of the objective function, we run the simulator forward to the final data assimilation time with an initial guess of the model parameters (as assigned in the previous step) for the MAP estimate and compute $O(m^l)$ from Eq. 3.7. Set the initial value of the LM parameter equal to $\gamma = O(m^l)/N_d$.

3. Call the Lanczos algorithm to compute the truncated-SVD of the dimensionless sensitivity matrix associated with the MAP estimate at outer loop iteration l, pertaining to m^l.

4. Use Eq. 3.45 together with

$$m^{l+1}_{\text{temp}} = m^l + L\delta\tilde{m}^{l+1} \tag{3.55}$$

to calculate a proposed new update of model parameter, m^{l+1}_{temp}.

5. Calculate $O(m^{l+1}_{\text{temp}})$ by running the simulator. If $O(m^{l+1}_{\text{temp}}) \geq O(m^l)$, then increase γ by a factor of 10 and return to step 4. If $O(m^{l+1}_{\text{temp}}) < O(m^l)$, then set

$$m^{l+1} = m^{l+1}_{\text{temp}}, \tag{3.56}$$

$$\tilde{m}^{l+1} = \tilde{m}^l + \delta\tilde{m}^{l+1}, \tag{3.57}$$

and decrease γ by a factor of 10

6. Check for convergence using the criteria discussed later. If the algorithm has not converged, increase iteration index l by 1 and go to the next step which is the first step of the inner loop. Otherwise, go to step 7, i.e., terminate the algorithm.

   (a) With k denoting the iteration index of the inner loop, set $k = 0$ and set the initial guess of the model parameter for the inner loop as $m^k = m^l$, where m^l denotes the last updated model obtained at the outer loop.

   (b) Compute the damping factors from Eq. 3.53, then compute the value of the damped objective function $O_{\text{damp}}(m^k)$ with Eq. 3.54.

   (c) Call the adjoint gradient subroutine to calculate the product of G_k^T with the vector $\tilde{C}_D^{-1}(g(m^k) - d_{\text{obs}})$.

   (d) Use Eq. 3.51 together with

$$m^{k+1}_{\text{temp}} = m^k + L\delta\tilde{m}^{k+1} \tag{3.58}$$

to calculate a proposed new update of model parameter, m^{k+1}_{temp}.

(e) Run the simulator forward to calculate the corresponding value of the damped objective function, $O_{\text{damp}}(m^{k+1}_{\text{temp}})$.

(f) if $O_{\text{damp}}(m^{k+1}_{\text{temp}}) \geq O_{\text{damp}}(m^k)$, then increase γ by a factor of 10 and return to step d. If $O_{\text{damp}}(m^{k+1}_{\text{temp}}) < O_{\text{damp}}(m^k)$, then set $m^{k+1} = m^{k+1}_{\text{temp}}$, decrease γ by a factor of 10 and go to the next step.

(g) Check for convergence of inner loop. The computational cost of each successful iteration in the inner loop is one forward simulation run and one adjoint solution, which is much less than the cost of one iteration of the outer loop. Therefore, it is feasible to do inner loop iterations as long as the decrease of objective function is reasonable compared to the computational cost of iterations. In our history matching examples, we have used the following convergence criterion

$$\frac{O_{\text{damp}}(m^k) - O_{\text{damp}}(m^{k+1})}{O_{\text{damp}}(m^k)} < 5 \times 10^{-3}. \tag{3.59}$$

If the condition given by Eq. 3.59 is not satisfied, increase the k index by one and return to step c. Otherwise, set $m^l = m^k$ and go to step 3. When we return to the outer loop, we may choose to increase the number of singular vectors used, i.e., decrease the value of the sv-cut for the next call of the Lanczos algorithm.

7. end

In the basic SVD parameterization algorithm, we need to update the truncated SVD in each iteration and there is no inner loop. Therefore, to implement the basic SVD parameterization algorithm, we delete step 6 of the algorithm described above and replace step 5 by

Step 5: Check for convergence using the criteria discussed later. If the algorithm has not converged, increase iteration index l by 1 and go to step 3. Otherwise terminate the algorithm.

3.9 Convergence Criteria

We use the following convergence criteria for all of the algorithms described in this chapter in computation of MAP estimate. For convergence, we require both the change in the objective function and the change in the model to be small, i.e.,

$$\frac{|O(m^{l+1}) - O(m^l)|}{O(m^{l+1})} < \epsilon_o \tag{3.60}$$

and

$$\frac{\|m^{l+1} - m^l\|_2}{\|m^{l+1}\|_2} < \epsilon_m \tag{3.61}$$

where for the examples considered in this study,

$$\epsilon_o = 10^{-3}, \tag{3.62}$$

and

$$\epsilon_m = 10^{-2}. \tag{3.63}$$

Based on results given in Tarantola [59] which are discussed in more detail in Oliver et al. [51], if m_c is the model obtained at convergence, we expect $O_N(m_c)$, the normalized objective function evaluated at m_c, will satisfy,

$$1 - 5\sqrt{2/N_d} \le O_N(m_c) \le 1 + 5\sqrt{2/N_d}. \tag{3.64}$$

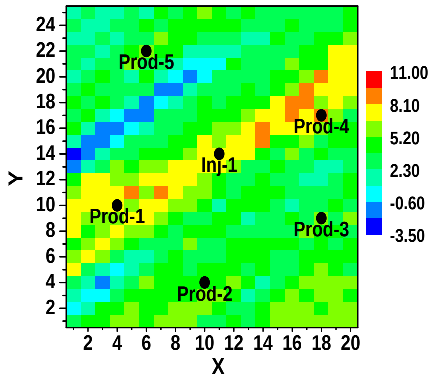
When we are generating a MAP estimate, the normalized objective function is defined by Eq. 3.48. This result will be used to provide an indication of the quality of our match.

3.10 Computational Results

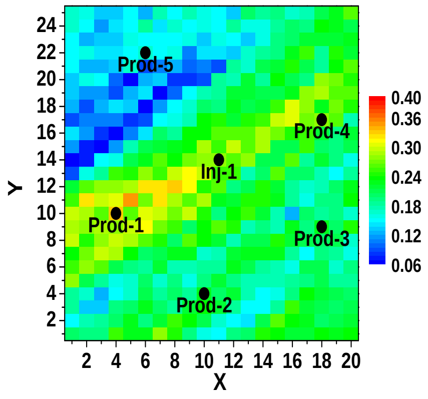
3.10.1 Example 1

This synthetic example pertains to a two-dimensional horizontal reservoir with a 20×25 grid and 500 active grid blocks. The areal dimensions of the reservoir are 6000 feet by 7500 feet, with a uniform grid size of $\triangle x = \triangle y = 300$ feet. The reservoir thickness is 10 feet thick. The true rock property fields were generated using sequential Gaussian co-simulation. The key geostatistical parameters used to generate the truth are listed in Table 3.1. In this table, φ_{mean} and $[ln(k)]_{\text{mean}}$, respectively, denote the prior mean of porosity and log-permeability. The standard deviations of porosity and log-permeability, respectively, are denoted by σ_φ and $\sigma_{ln(k)}$. Here, $\rho_{\varphi,ln(k)}$ denotes the correlation coefficient between porosity and log-permeability, α is the angle measured counterclockwise from the x-axis to the principal direction of the covariance function, r_1 is the correlation range in the principal direction and r_2 is the correlation range in the orthogonal direction. The prior mean of horizontal log-permeability is uniform and equal to 4.5, i.e, all entries of m_{prior} are equal to 4.5. The true horizontal log-permeability and true porosity maps with the location of wells are shown in Figures 3.1(a) and 3.1(b).

There are 5 production wells and one injection well in the reservoir. As can be seen from Figure 3.1(a), there is essentially a high permeability channel in the reservoir. The injection well (Inj-1), Prod-2 and Prod-4 are located within the channel. Also there is a barrier with low permeability values in the north-west part of the reservoir. Prod-5 is located behind this barrier. The initial reservoir pressure is 4800 psi. The injection well starts injecting water with a specified injection rate of $q_w = 1000$ STB/day, and at the same time, all producers start production. The



(a) True ln(k).



(b) Porosity.

Figure 3.1: True horizontal log-permeability and porosity fields, Example 1.

production rate at each well is specified to be $q_o = 200$ STB/day. This schedule is continued until we reach a total time of $t = 3300$ days. The observation data in this example include the flowing bottom hole pressure (BHP) and water-oil ratio (WOR) recorded at 30 days intervals. At the end time of simulation ($t = 3300$ days), only Prod-1 and Prod-4 have experienced water breakthrough.

Table 3.1: Geostatistical parameters.

Parameters	Values
φ_{mean}	0.2
$[ln(k)]_{\text{mean}}$	4.50
σ_φ	0.05
$\sigma_{ln(k)}$	2.0
$\rho_{\varphi,\ln(\text{k})}$	0.80
α	40^o
$r_1(ft)$	8400
$r_2(ft)$	1500

The true synthetic data are generated with a simulator and the observation data are obtained by adding Gaussian random noise to the true synthetic data. The standard deviations of the Gaussian noises are $\sigma_{BHP} = 10$ psi for BHP of all wells,

$\sigma_{WOR1,i} = 0.02 WOR1_{\text{true},i}$ (STB/STB), and $\sigma_{WOR4,i} = 0.04 WOR4_{\text{true},i}$ (STB/STB). We only use the set of gridblock $\ln(k)$ as model parameters with the porosity field fixed equal to the true porosity field. Therefore, the number of model parameters are the same as the number of gridblocks, i.e., equal to $N_m = 500$. The number of observed data is $N_d = 880$. Note the variation in the original $\ln(k)$ field is from -3 to 9, thus we may need a fairly large number of singular vectors to approximate this variability.

3.10.2 MAP Estimate Results of Example 1

We have implemented the SVD and SVD-Grd parameterization algorithms to obtain the MAP estimate for this example. To compute the truncated-SVD, we need to specify the value of the sv-cut for each call of the Lanczos algorithm. In the basic implementation of the SVD parameterization algorithm, we use fixed value of sv-cut = 0.003125 during all the iterations. One obvious possibility for improving the efficiency of SVD parameterization is to start with a fairly low number of singular triplets and gradually increase the number during iterations, i.e., we can start the SVD parameterization algorithm with a fairly large value of sv-cut, say 0.1, and gradually decrease the sv-cut during iterations. Here, we show results for one implementation of this idea where we simply start with sv-cut = 0.1 and decrease sv-cut at each iteration by dividing by two until we reach sv-cut = 0.003125. Then from this point on, we iterate with the fixed value sv-cut = 0.003125 until we obtain convergence. Throughout, this implementation of the SVD algorithm is referred to as the modified SVD parameterization algorithm. In implementation of the SVD-Grd algorithm, we use the same strategy for changing sv-cut as was used in the modified SVD parameterization algorithm.

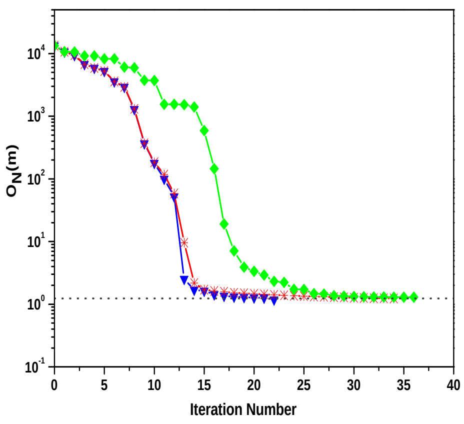
Figure 3.2(a) shows the behavior of the normalized objective function during iteration obtained with the SVD and SVD-Grd parameterization algorithms. As shown in this figure, the basic SVD parameterization algorithm (red curve) and the modified SVD parameterization algorithm (blue curve) converged in 34 and 22 iterations, respectively, to the final value of the normalized objective functions of $O_N(m) = 1.24$ and $O_N(m) = 1.18$. Note that for the SVD parameterization algorithms the number of updating of the singular triplets is the same as the number of iterations. Therefore, with the modified SVD algorithm, we have improved the computational efficiency of the SVD parameterization algorithm. For the example under consideration, $N_d = 880$, so the right-side of Eq. 3.64 is equal to 1.24. A lower value of the objective function at convergence indicates a more probable model (higher value of the pdf). For the SVD-Grd parameterization algorithm in Figure 3.2, the x-axis represents the iteration number of both the outer and the inner loops. Specifically, odd values of the iteration index correspond to the result from the outer loop and even values of the iteration index correspond to the final result obtained from an inner loop iteration. Thus, the behavior of the normalized objective function in Figure 3.2(a) shows that the SVD-Grd parameterization algorithm (green curve) converged to the final value of the normalized objective function of $O_N(m) = 1.29$ using 18 updates of the singular triplets (18 calls to the Lanczos algorithm). Note that with the value of sv-cut $= 0.003125$, for all three algorithms 21 number of singular triplets retained at convergence.

The first term in the objective function (Eq. 3.7) is the model mismatch term denoted by $O_m(m)$, which is defined by

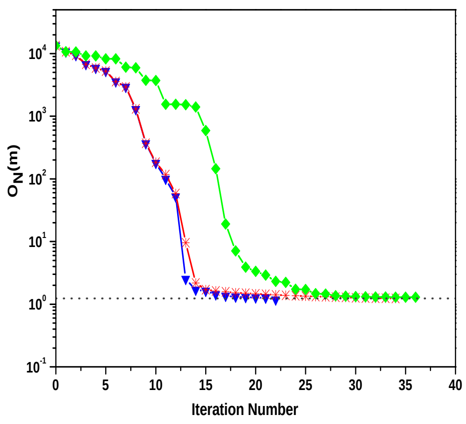
$$O_m(m) = \frac{1}{2}(m - m_{\text{prior}})^T C_M^{-1}(m - m_{\text{prior}}). \tag{3.65}$$

$O_m(m)$ gives a measure of how much we have had to change the prior best estimate, m_{prior}, to match the data during iterations of the optimization algorithm, i.e., gives a measure of the smoothness of the model. Since in the example considered here, the permeability field represented by m_{prior} is uniform, larger values of $O_m(m)$ correspond to rougher models. The behavior of $O_m(m)$ during iteration of the SVD and SVD-

Grd algorithms is shown in Figure 3.2(b).



(a) $O_N(m)$.

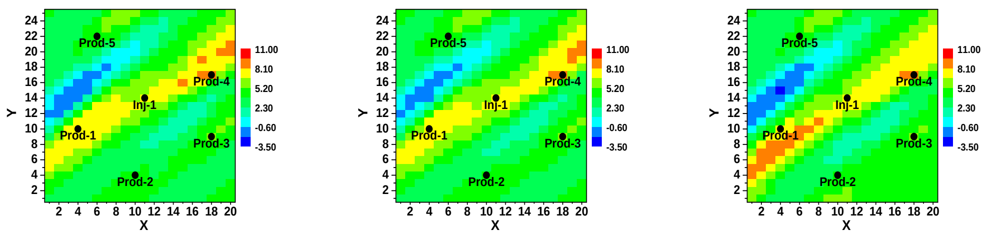


(b) $O_m(m)$.

Figure 3.2: Behavior of the normalized objective function and model mismatch part of the objective function during iteration, red is from basic SVD parameterization algorithm, blue is from modified SVD parameterization algorithm, and green is from SVD-Grd algorithm, Example 1.

The final estimate of the log-permeability fields, $\ln(k)$, obtained with SVD and SVD-Grd parameterization algorithms are shown in Figure 3.3. Note that we use m_{prior} as an initial guess of the model parameters in both algorithms. As can be seen from this figure, both SVD and SVD-Grd parameterization algorithms captured the essential features of the truth and the existence of the channel and low permeability barrier is visible in the final fields and is comparable with the truth (Figure 3.1(a)).

Figure 3.4 shows the history matching results and future forecasts of the BHP of Inj-1 and the WOR of Prod-1 and Prod-4 obtained with the SVD and SVD-Grd parameterization algorithms. Note that in this and similar figure the vertical line denotes the end of the history matching period. As can be seen from Figure 3.4(a), good BHP history matches of Inj-1 are obtained and also the prediction of BHP obtained with both the SVD and SVD-Grd algorithms are in good agreement with the true prediction. The BHP data matches and future performance predictions for other producers (not shown) are similar to Inj-1. The history matching results

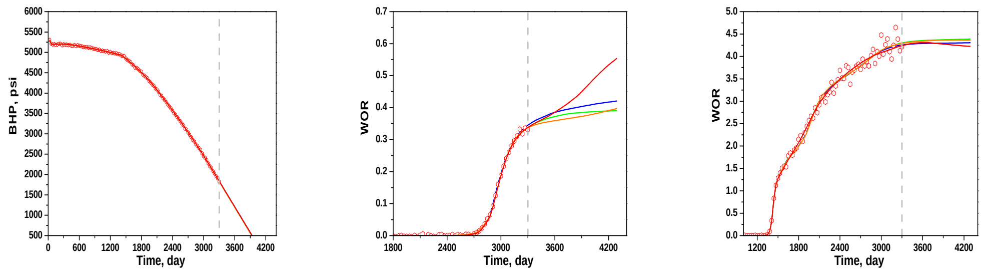


(a) Basic SVD. (b) Modified SVD. (c) SVD-Grd.

Figure 3.3: Final log-permeability fields with basic and modified SVD parameterization algorithms and SVD-Grd algorithm, Example 1.

and future forecasts of the WOR of Prod-1 and Prod-4 are shown in Figures 3.4(b) and 3.4(c), respectively. We observe that both algorithms performed very well during the history matching period for both producers. The good history matching results were expected from the low values of the normalized objective function obtained at convergence with both the SVD and SVD-Grd parameterization algorithms (Figure 3.2(a)). Although we obtained very good history matches of the WOR of Prod-1, at later times, the WOR prediction results of Prod-1 obtained from the algorithms deviated from the true prediction. The WOR prediction performances of Prod-4 is comparable to the truth for both algorithms. There is no water breakthrough for other producers up to 4300 days of prediction for the true model or any of the estimated permeability fields.

Table 3.2 summarizes the total computational costs of the SVD and SVD-Grd parameterization algorithms based on the number of the simulation runs, adjoint solutions, and forward gradient. Note that for each call of the Lanczos algorithm we only need one complete (nonlinear) reservoir simulation run. During this run, we save all information required for building the relevant matrices for subsequent Lanczos iterations in which we need to calculate G times a vector with a "forward run" (gradient simulator) and G^T times a vector with an adjoint solution. The computational cost of a forward gradient and an adjoint solution are approximately



(a) BHP Inj-1. (b) WOR Prod-1. (c) WOR Prod-4.

Figure 3.4: Observed and predicted data with basic and modified SVD parameterization algorithms and SVD-Grd algorithm, open red circles are observed data, red is from true model, blue is from basic SVD parameterization algorithm, green is from modified SVD parameterization algorithm and orange is from SVD-Grd algorithm, Example 1.

equal and the computational cost of each is approximately equal to one fourth of one reservoir simulation run. The total computational costs are given in the fifth column of the Table 3.2 based on the number of equivalent simulation runs. As can be seen, the basic SVD parameterization algorithm has the highest computational cost. Implementations of the SVD-Grd algorithm increased the computational efficiency of the SVD parameterization algorithm. However, the modified SVD parameterization algorithm has the lowest computational cost for this example.

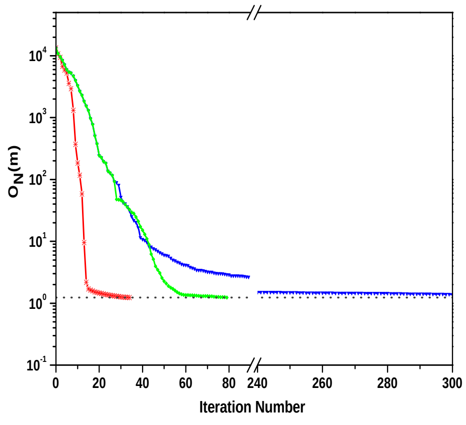
Table 3.2: Summary of computational costs of the SVD and SVD-Grd parameterization algorithms, Example 1.

Algorithm	Simulations	Forward Runs	Adjoint	Equ. Sim. Runs	$O_N(m)$
Basic SVD	98	929	895	554	1.24
Modified SVD	63	535	513	325	1.18
SVD-Grd	138	414	458	356	1.29

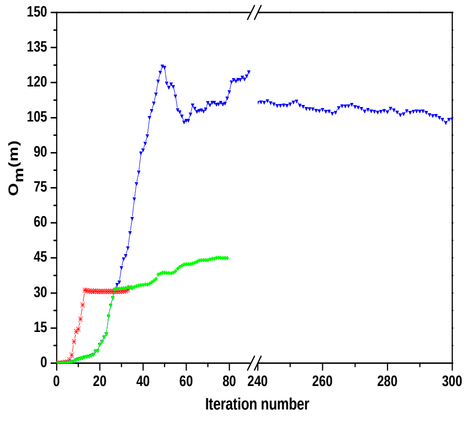
3.10.3 LBFGS and LBFGS-SVD Results of Example 1

Next, we consider results generated from the LBFGS algorithm and the combination of LBFGS and SVD parameterization algorithms (LBFGS-SVD). Figure 3.5 illustrates the behavior of the normalized objective function and the model mismatch part of the objective function obtained with LBFGS, basic SVD, and LBFGS-SVD algorithms. The implementation of the SVD parameterization algorithm is with sv-cut = 0.003125 fixed during iteration and is the same result shown in Figure 3.2 with red curve. The LBFGS algorithm has not converged in 300 iterations and at iteration 300 gives an estimate of the log-permeability field such that $O_N = 1.39$. The LBFGS-SVD algorithm converged in 79 iterations with two updates of singular values to an estimate of the model parameters such that $O_N = 1.23$. The number of retained singular values are 20 and 18 in the first and the second call of the Lanczos algorithm, respectively. In the combined algorithm, we used sv-cut = 0.00625 for the SVD cutoff, but results are essentially identical if we also use for this singular cutoff of 0.003125. The behavior of the model mismatch part of the objective function (Figure 3.5(b)) shows that with the LBFGS algorithm, the model obtained is further from m_{prior} than is the final model obtained with the other two algorithms. This is consistent with the results of Figure 3.6, which shows the final log-permeability maps obtained with the LBFGS, basic SVD and LBFGS-SVD algorithms. As we can see, LBFGS algorithm has captured some of the characteristics of the channel and the barrier in the final model, however, we also obtained relatively high permeability regions that do not appear in the true permeability field. We obtained a more reasonable final permeability filed with LBFGS-SVD and the pure SVD algorithm than we obtained with the LBFGS algorithm.

Figure 3.7 shows the data matches and predictions obtained with the basic SVD, LBFGS and LBFGS-SVD algorithms. As we expect from the low value of the normalized objective function, we obtained reasonably good data matches with all three algorithms. The future predictions are also reasonable although the future prediction of WOR at Prod-1 is not as accurate as one would hope to obtain. Note the implementation of the LBFGS-SVD algorithm has improved the prediction results

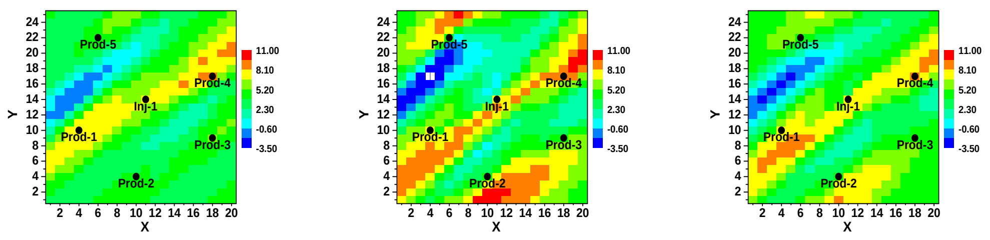


(a) $O_N(m)$.



(b) $O_m(m)$.

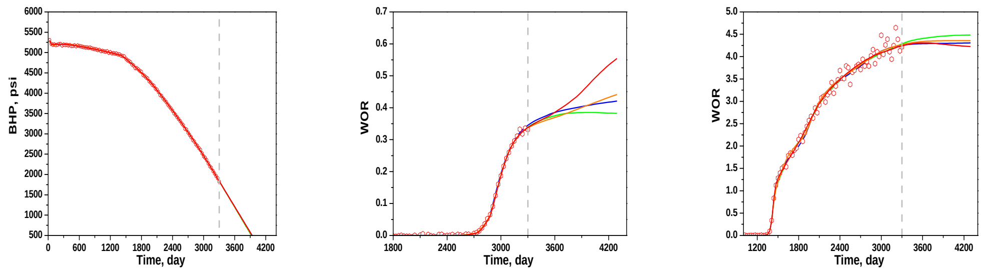
Figure 3.5: Behavior of the normalized objective function and model mismatch part of the objective function during iteration, red is from basic SVD parameterization algorithm, blue is from LBFGS algorithm, and green is from LBFGS-SVD algorithm, Example 1.



(a) Basic SVD. (b) LBFGS. (c) LBFGS-SVD.

Figure 3.6: Final log-permeability fields with basic SVD parameterization, LBFGS and LBFGS-SVD algorithms, Example 1.

of the WOR of Prod-1 compared to the results obtained with the LBFGS method.



(a) BHP Inj-1. (b) WOR Prod-1. (c) WOR Prod-4.

Figure 3.7: Observed and predicted data with basic SVD parameterization, LBFGS and LBFGS-SVD algorithms, open red circles are observed data, red is from true model, blue is from basic SVD parameterization algorithm, green is from LBFGS algorithm and orange is from LBFGS-SVD algorithm, Example 1.

Table 3.3 summarizes the total computational costs of LBFGS, basic SVD and LBFGS-SVD algorithms based on the number of simulation runs, gradient simulator and adjoint solutions. In Table 3.3, the overall computational cost of each algorithm is given in the last column based on the number of equivalent simulation runs. The total computational cost of the LBFGS algorithm is the highest and the computational cost of the combined algorithm (LBFGS-SVD) is the lowest. In fact, the combined algorithm is significantly more efficient than the LBFGS and basic SVD parameterization algorithms. The comparison of the results of this table with Table 3.2 indicates that the LBFGS-SVD combined algorithm is the most efficient algorithm for this example.

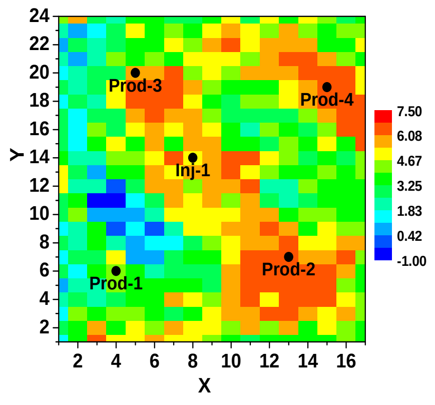
3.10.4 Example 2

This example pertains to a two-dimensional horizontal reservoir model with a $17 \times 24 \times 1$ uniform grid. The areal dimensions of the reservoir are 6800 feet by 9600 feet and the reservoir thickness is uniform and equal to 10 feet. The sizes of the grid blocks are $\Delta x = \Delta y = 400$ feet. We use an anisotropic spherical covariance function

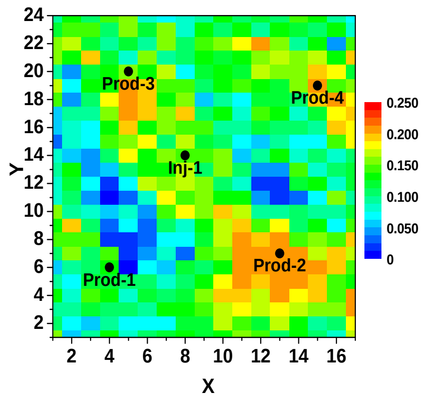
Table 3.3: Summary of computational costs of the basic SVD, LBFGS and LBFGS-SVD algorithms, Example 1.

Algorithm	Simulations	Forward Runs	Adjoint	Equ. Sim. Runs	$O_N(m)$
Basic SVD	98	929	895	554	1.24
LBFGS	449	——	449	561	1.39
LBFGS-SVD	180	50	221	248	1.23

to generate the prior covariance matrix, C_M. The principle direction has an angle of 120° with the x-direction measured counter clockwise from the x-axis. We use a correlation length equal to $25 \times \Delta x = 10,000$ feet in the principal direction and a shorter correlation length equal to $10 \times \Delta x = 4,000$ feet in the orthogonal direction. The prior mean for log-permeability is 4.15 and the prior standard deviation is 1.35. The prior mean for porosity is 0.12 and the prior standard deviation is 0.056. The correlation coefficient between porosity and log permeability is assumed to be 0.90.



(a) True ln(k).



(b) Porosity.

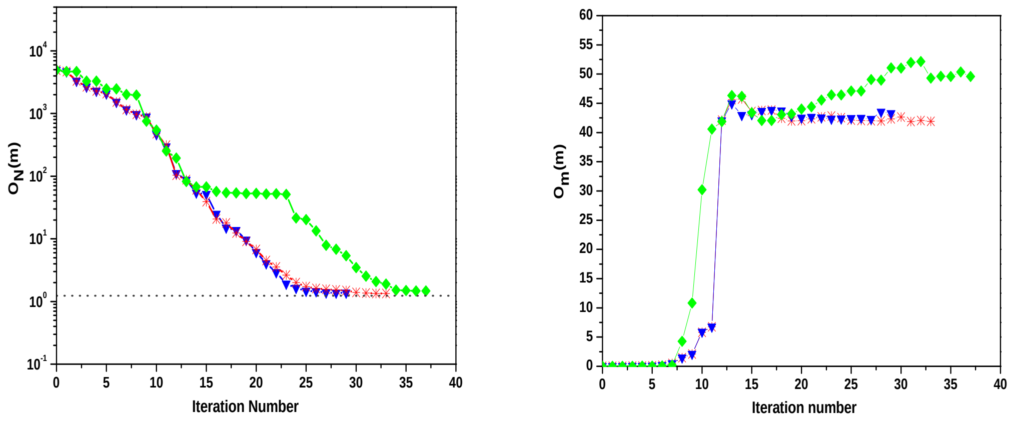
Figure 3.8: True horizontal log-permeability and porosity fields, Example 2.

The true horizontal log-permeability field and the true porosity field with the location of wells are respectively shown in Figures 3.8(a) and 3.8(b). As can be seen from the truth maps, there is high permeability streak in the south-east to north-west direction connecting Prod-2 and Prod-3. Also, there is a high permeability-porosity channel in the north-east corner of the reservoir where Prod-4 is located. There are five wells in this reservoir. A water injection well (Inj-1) is located near the center of the reservoir at grid block $(8, 14)$, and four production wells (Prod-1, Prod-2, Prod-3, and Prod-4) are located, respectively, at grid blocks of $(4, 6)$, $(13, 7)$, $(5, 20)$ and $(15, 19)$. Only two-phase flow (oil and water) is considered. The initial reservoir pressure is 6500 psi. The initial reservoir saturation is equal to irreducible water saturation, which is given by $S_{iw} = 0.2$. The residual oil saturation is $S_{or} = 0.1$. The water injection rate is $q_w = 1000$ STB/day from time zero onward and the oil rate at each producer is specified as $q_o = 250$ STB/day.

Observed data for the flowing bottom hole pressure (BHP) and water-oil ratio (WOR) are obtained by adding noise to corresponding production data obtained from a forward simulator run to $t = 3000$ days. Noise is generated by assuming measurement errors are independent Gaussian random variables with all means equal to zero. The standard deviations of measurement error for all bottom hole pressures, WOR of Prod-2, and WOR of Prod-3, respectively, were specified as $\sigma_p = 15$ (psi), $\sigma_{\mathrm{WOR2},i} = 0.02\ \mathrm{WOR2}_{\mathrm{true},i}$ (STB/STB), and $\sigma_{\mathrm{WOR3},i} = 0.10\ \mathrm{WOR3}_{\mathrm{true},i}$ (STB/STB). Water breakthrough occurs only at producers 2 and 3. Measured data are recorded every thirty days so for example, at each production well, we have 100 pressure data. For this example the set of model parameters includes only the gridblock log-permeability values with the porosity field fixed equal to the true porosity field. Therefore, the number of model parameters are the same as the number of gridblocks, i.e., equal to $N_m = 408$. The number of observed data is $N_d = 700$.

3.10.5 MAP Estimate Results of Example 2

The implementations of the SVD and SVD-Grd parameterization algorithms for this example are similar to those used for the previous example, i.e., we use the same fixed value of sv-cut $= 0.003125$ during iteration for the basic SVD parameterization algorithm and same strategy of changing sv-cut for the modified SVD parameterization and SVD-Grd algorithms. The behavior of the normalized objective function and the model mismatch part of the objective function obtained with the SVD and SVD-Grd parameterization algorithms are shown in Figures 3.9(a) and 3.9(b), respectively. As can be seen from these figures, the behavior of the $O_N(m)$ and $O_m(m)$ for the basic (red curves) and modified (blue curves) implementations of the SVD parameterization algorithm are almost identical. The basic and modified SVD parameterization algorithms converged to a final estimate of the model such that $O_N(m) = 1.34$ and $O_N(m) = 1.37$, respectively, in 33 and 29 iterations. Also, the SVD-Grd algorithm (green curves) converged to a final estimate of the model such that $O_N(m) = 1.47$ and required 18 updates of the singular triplets. At convergence, the number of retained singular triplets are 18, 17 and 18 for the basic SVD parameterization, modified SVD and SVD-Grd algorithms, respectively.

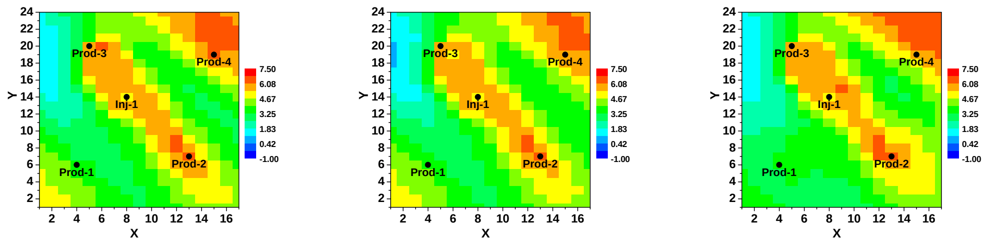


(a) $O_N(m)$. (b) $O_m(m)$.

Figure 3.9: Behavior of the normalized objective function and model mismatch part of the objective function during iteration, red is from basic SVD parameterization algorithm, blue is from modified SVD parameterization algorithm, and green is from SVD-Grd algorithm, Example 2.

Based on Eq. 3.64, for this example, we expect that the value of the normalized objective function to be less than 1.27 at convergence. As we presented here, with the final value of sv-cut = 0.003125, both algorithms converged to the model such that Eq. 3.64 was not quite satisfied although we obtained a normalized objective function close to one in all cases. We then tried a smaller value of sv-cut to obtain slightly smaller values of the objective function at convergence for this example. The results obtained with the final value of sv-cut = 0.00078125 (not shown here) satisfied Eq. 3.64.

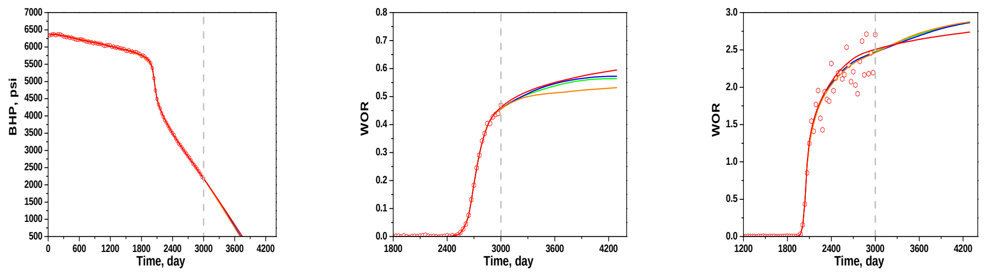
Figure 3.10 shows the final model estimates of log-permeability, $\ln(k)$, obtained with implementations of SVD and SVD-Grd parameterization algorithms. Note in all cases, a uniform log-permeability of $\ln(k) = m_{\text{prior}}$ is used as an initial guess of the model parameters. As can be seen from Figures 3.10(a), 3.10(b), and 3.10(c) the MAP estimates obtained with both SVD and SVD-Grd algorithms illustrate the basic features of the truth. In all log-permeability fields, the main characteristics of the channel have been captured compared to the true log-permeability field (Figure 3.8(a)).



(a) Basic SVD. (b) Modified SVD. (c) SVD-Grd.

Figure 3.10: Final log-permeability fields with basic and modified SVD parameterization algorithms and SVD-Grd algorithm, Example 2.

The history matching results and future prediction performances of the BHP of Prod-3 and the WOR of Prod-2 and Prod-3 are shown in Figure 3.11 for the SVD and SVD-Grd parameterization algorithms. As can be seen from this figure, we have obtained very good history matches of BHP and WOR with both algorithms. Also, the BHP future forecasts are comparable with the true predictions. We have obtained identical history matches and prediction results of BHP for other wells (not shown). The prediction of WOR for Prod-2 obtained with the SVD-Grd algorithm is not as good as those obtained with the basic and modified SVD parameterization algorithms. The WOR prediction for Prod-3 obtained with both algorithms deviates from the truth at later times.



(a) BHP Prod-3. (b) WOR Prod-2. (c) WOR Prod-3.

Figure 3.11: Observed and predicted data with basic and modified SVD parameterization algorithms and SVD-Grd algorithm, open red circles are observed data, red is from true model, blue is from basic SVD parameterization algorithm, green is from modified SVD parameterization algorithm and orange is from SVD-Grd algorithm, Example 2.

The computational costs in terms of the number of simulation runs, forward gradient and adjoint solutions are summarized in Table 3.4 for the SVD and SVD-Grd parameterization algorithms. In Table 3.4, the overall computational cost of each implementation is given in the fifth column based on the number of equivalent simulation runs. As indicated in this table, the SVD-Grd parameterization algorithm is more computationally efficient than the basic and modified SVD parameterization algorithms.

3.10.6 Comments on the Value of the Singular Value Cutoff

Based on the overall results, the singular value cutoff on the order of 0.003 is sufficient to obtain good data matches, a reasonably small value of the objective function and a reasonable geological description of the permeability field. Based on

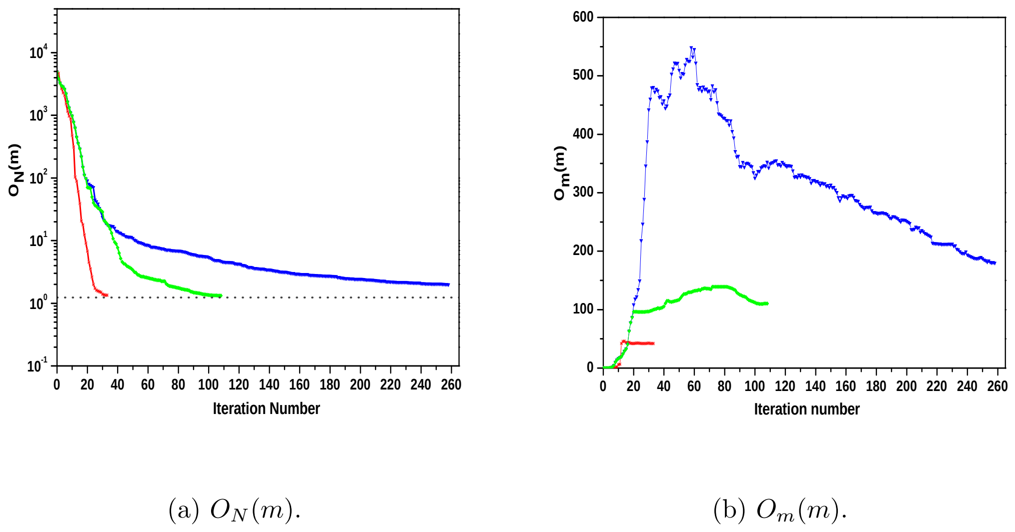
Table 3.4: Summary of computational costs of the SVD and SVD-Grd parameterization algorithms, Example 2.

Algorithm	Simulations	Forward Runs	Adjoint	Equ. Sim. Runs	$O_N(m)$
Basic SVD	95	766	733	469	1.34
Modified SVD	81	644	615	395	1.37
SVD-Grd	185	342	397	370	1.47

our experiments with different examples, a singular cutoff value on the order of 0.001 has always proved sufficient to obtain good data matches, a sufficiently small value of the objective function (see Eq. 3.64) and a geological description consistent with the true geology. However, if this cutoff does not give a sufficient low value of the objective function at convergence of the SVD algorithm, we simply divide the cutoff value by 4 and continue the algorithm.

3.10.7 LBFGS and LBFGS-SVD Results of Example 2

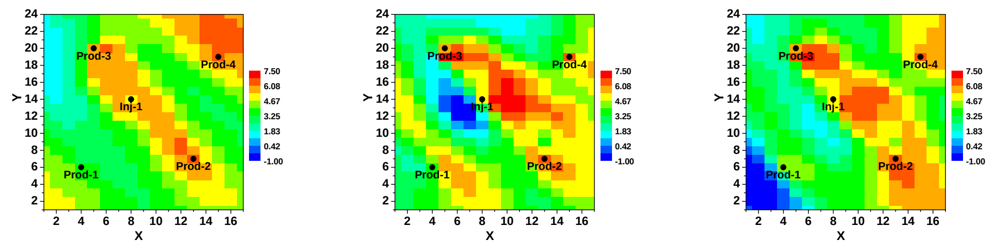
Here, the results obtained from the LBFGS and LBFGS-SVD algorithms are presented for the example considered previously in subsection Example 2; see Figure 3.8(a) for the true log-permeability field and well locations. Figure 3.12 shows the behavior of the normalized objective function (Figure 3.12(a)) and the model mismatch part of the objective function (Figure 3.12(b)) during the iterations obtained from the basic SVD parameterization algorithm (red curves), LBFGS algorithm (blue curves), and LBFGS-SVD algorithm (green curves). The results of the basic SVD parameterization algorithm are the same results given in Figure 3.9 which are shown again here for comparison. As shown from Figure 3.12(a), the LBFGS algorithm converged in 258 iterations to an estimate of the vector of model parameters, m, such that $O_N = 1.98$. The combined algorithm, LBFGS-SVD converged in 108 iterations with the final normalized objective function, $O_N = 1.32$ and required three updates of singular triplets with the Lanczos algorithm. Note that for the SVD part of the combined LBFGS-SVD algorithm, we use sv-cut = 0.003125. We have also used sv-cut = 0.00625 in the LBFGS-SVD algorithm and the results are very similar to those obtained with sv-cut = 0.003125. Even though the algorithm based solely on SVD parameterization converged in 33 iterations, the basic SVD parameterization algorithm is computationally expensive. In this example, LBFGS-SVD algorithm is far more efficient than the LBFGS and SVD parameterization algorithms as discussed in more detail later.



(a) $O_N(m)$. (b) $O_m(m)$.

Figure 3.12: Behavior of the normalized objective function and model mismatch part of the objective function during iteration, red is from basic SVD parameterization algorithm, blue is from LBFGS algorithm, and green is from LBFGS-SVD algorithm, Example 2.

The final log-permeability fields obtained with LBFGS, basic SVD, and LBFGS-SVD algorithms are shown in Figure 3.13. We can see that the LBFGS algorithm gives a model which has captured basic features of the truth, but there is a high permeability region around Prod-1 which does not appear in the truth. The model obtained with the LBFGS-SVD algorithm is reasonably close to the truth map, as the channel features and high permeability region around Prod-4 have been captured correctly, however the final model shows a low permeability regions in the lower left corner which does not exist in the truth.



(a) Basic SVD. (b) LBFGS. (c) LBFGS-SVD.

Figure 3.13: Final log-permeability fields with basic SVD parameterization, LBFGS and LBFGS-SVD algorithms, Example 2.

Figure 3.14 shows the history matches and future forecast of the BHP of Prod-3 and the WOR of Prod-2 and Prod-3 obtained from the basic SVD, LBFGS, and LBFGS-SVD algorithms. As expected from the low values of the objective function, we obtained reasonably good matches of WOR and BHP data with all three algorithms. For the BHP of Prod-3, the predictions from the three algorithms are comparable with the true prediction. The BHP results of other wells (not shown) are of the same quality. The WOR history matches of Prod-2 and Prod-3 obtained with both the LBFGS and LBFGS-SVD algorithms are very good. At later times, the prediction of the WOR for Prod-2 obtained with the LBFGS is lower than the truth. The permeability region around Prod-2 obtained with LBFGS algorithm has lower values of $\ln(k)$ compared to the truth, i.e., we under predict WOR for Prod-2 at later times. The prediction of the WOR of Prod-3 obtained with the LBFGS and LBFGS-SVD algorithms are comparable to the truth.

The summary of the computational costs of the LBFGS, basic SVD and LBFGS-SVD algorithms is given in Table 3.5. The computational cost is based on the number of simulation runs, gradient simulator runs and adjoint solutions. For the LBFGS algorithm, at each iteration, we need to ensure that a downhill direction is found and also do a line search to find an approximate step size. Each check on whether we have found a proper step size requires a forward run and an adjoint

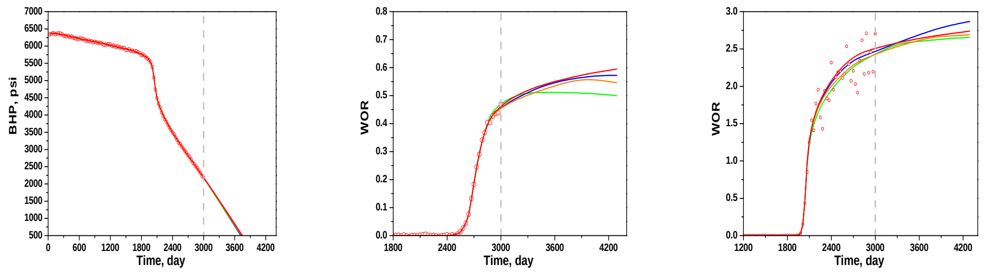

(a) BHP Prod-3. (b) WOR Prod-2. (c) WOR Prod-3.

Figure 3.14: Observed and predicted data with basic SVD parameterization, LBFGS and LBFGS-SVD algorithms, open red circles are observed data, red is from true model, blue is from basic SVD parameterization algorithm, green is from LBFGS algorithm and orange is from LBFGS-SVD algorithm, Example 2.

solution. The results of Table 3.5 indicate that the total computational cost of the LBFGS-SVD algorithm is much lower than the cost of the basic SVD and LBFGS algorithms.

Table 3.5: Summary of computational costs of the basic SVD, LBFGS and LBFGS-SVD algorithms, Example 2.

Algorithm	Simulations	Forward Runs	Adjoint	Equ. Sim. Runs	$O_N(m)$
Basic SVD	95	766	733	469	1.34
LBFGS	377	——	377	471	1.98
LBFGS-SVD	182	102	271	275	1.32

CHAPTER 4
SIMULATION OF PERMEABILITY FIELDS WITH SVD PARAMETERIZATION ALGORITHMS AND ENKF

4.1 Introduction

In the Bayesian framework [59, 51], the problem of uncertainty quantification is equivalent to the formulation and sampling of the a posteriori probability density function (pdf). Calculation of the MAP estimate is really only a part of the reservoir characterization problem because the MAP estimate gives a very smooth model which does not reflect the extremes in heterogeneity that would be typical for a realization generated from the prior model. The MAP estimate provides an approximation of the conditional mode of the distribution, or in the case of a Gaussian posterior, the MAP estimate provides an approximation of the conditional mean of the posterior pdf.

In this chapter, our main focus is on the generation of multiple plausible realizations of reservoir model parameters that are consistent with all static information and conditioned to observed data in order to quantify the uncertainty in reservoir performance. Randomized maximum likelihood (RML) [50, 33, 55] provides a viable procedure for generating an approximate sampling of the posterior pdf. Although RML provides only an approximate sampling procedure, by its construction, it is expected to at least generate samples around the modes of the posterior pdf.

In Chapter 3, we provided a theoretical basis for parameterization of reservoir model parameters based on the truncated-SVD of the dimensionless sensitivity matrix [67]. Specifically, we showed that parameterization based on the principal singular triplets of the dimensionless sensitivity matrix is **ideal** if the objective is to minimize the posterior uncertainty in model parameters. We implemented two SVD parameterization algorithms for generation of only the MAP estimate. The generation of multiple realizations was not considered in the previous chapter. In this chapter, we provide two algorithms for sampling from the posterior pdf based on the incorporation of truncated SVD parameterization into the RML method. In the first algorithm, we generate N_e realizations simultaneously by minimizing an ensemble of objective functions concurrently using the singular triplets of a particular realization for parameterization at each iteration. Even though multiple realizations are generated, the algorithm only updates one SVD parameterization per iteration, instead of N_e SVD updates, and thus results in considerable computational savings. In the second algorithm, to further improve computational efficiency, we add an inner iteration in which we explicitly compute the gradient of each objective function that is being minimized with the adjoint method, where during the inner iteration the truncated SVD parameterization does not vary.

We also present results obtained with implementation of the ensemble Kalman filter (EnKF). Despite recent intense focus on the application of the EnKF to history matching problems, Gao et al. [20] appears to be the only paper in the petroleum engineering literature that compares EnKF with gradient-based history matching. For the well known PUNQ-S3 benchmark problem [15], Gao et al. [20] showed that the ensemble mean obtained by assimilating production data with EnKF was fairly similar to the conditional mode (maximum a posteriori (MAP) estimate) obtained with a quasi-Newton method and both methods gave similar estimates of future performance predictions. In this work, we further extend this comparison by comparing history-matching results obtained with our new algorithms to those obtained with the ensemble Kalman filter (EnKF) without and with covariance localization [28].

Finally, we present a combined algorithm where EnKF is used to generate an initial guess for the truncated-SVD algorithms. This combination yields better data matches and performance predictions than are obtained with EnKF with a computational cost that is less than is required by either of the pure SVD algorithms.

4.2 RML for Sampling Posterior PDF

Throughout, we use the notation $x \sim N(\mu, C)$ to indicate that x is a Gaussian random vector with mean (expectation) μ and covariance matrix C. A vector x is always a column vector whereas its transpose, x^T, is a row vector.

The randomized maximum likelihood (RML) algorithm generates a set of realizations of the model, m_i, $i = 1, 2 \cdots$ that represent an approximate sampling of the posteriori pdf. To construct the realizations m_i, $i = 1, 2 \cdots, N_e$ using RML, generate N_e samples $m_{\text{uc},i}$ from the prior Gaussian $N(m_{\text{prior}}, C_M)$ and N_e samples $d_{\text{uc},i}$ from $N(d_{\text{obs}}, C_D)$ as follows

$$m_{\text{uc},i} = m_{\text{prior}} + C_M^{\frac{1}{2}} Z_{m,i}, \tag{4.1}$$

and

$$d_{\text{uc},i} = d_{\text{obs}} + C_D^{\frac{1}{2}} Z_{d,i}, \tag{4.2}$$

where $Z_{m,i}$ is an N_m-dimensional vector of independent standard random normal deviates and $Z_{d,i}$ is an N_d-dimensional vector of independent standard random normal deviates. Then set

$$m_{c,i} = \operatorname*{argmin}_m O_{r,i}(m), \tag{4.3}$$

for $i = 1, 2, \cdots N_e$ where we have used the subscript c to denote that these are conditional realizations generated from RML and the objective functions, $O_{r,i}(m)$, $i = 1, 2, \cdots N_e$ are defined by

$$O_{r,i}(m) = \frac{1}{2}(m - m_{\text{uc},i})^T C_M^{-1}(m - m_{\text{uc},i}) + \frac{1}{2}(g(m) - d_{\text{uc},i})^T C_D^{-1}(g(m) - d_{\text{uc},i}). \tag{4.4}$$

The minimization of $O_{r,i}(m)$ is similar to the computation of the maximum a posteriori (MAP) estimate using a minimization algorithm like the LM algorithm. However, we use $m_{\text{uc},i}$ and $d_{\text{uc},i}$, respectively, instead of m_{prior} and d_{obs}, respectively, to generate the ith realization. Using the same procedure as was used for the MAP estimate in Chapter 3, the search direction in the modified LM algorithm for computing ith RML realization is given by

$$\big[(1+\gamma_l)I_{N_m} + G_{D,l}^T G_{D,l}\big]\delta\tilde{m}_{c,i}^{l+1} = -\{\tilde{m}_{c,i}^l + G_{D,l}^T C_D^{-\frac{1}{2}}(g(m_i^l) - d_{\text{uc},i})\}, \tag{4.5}$$

where l is the iteration index, I_{N_m} denotes $N_m \times N_m$ identity matrix, $G_{D,l}$ is the dimensionless sensitivity matrix at the lth iteration, γ_l is the Levenberg-Marquardt parameter and $\tilde{m}_i^l$ denotes the vector of transformed model parameters of the ith realization at iteration l defined as

$$\tilde{m}_{c,i}^l = L^{-1}(m_{c,i}^l - m_{\text{uc},i}). \tag{4.6}$$

Once $\delta\tilde{m}_{c,i}^{l+1}$ is calculated, we set

$$\tilde{m}_{c,i}^{l+1} = \tilde{m}_{c,i}^l + \delta\tilde{m}_{c,i}^{l+1}, \tag{4.7}$$

and apply Eq. 4.6 to calculate the updated model parameters of the ith realization as

$$m_{c,i}^{l+1} = m_{\text{uc},i} + L\tilde{m}_{c,i}^{l+1}. \tag{4.8}$$

The RML procedure has a high computational cost as a different minimization problem must be solved individually to generate each conditional realization. In this study, to reduce the computational costs of RML algorithm, we incorporate the SVD parameterization algorithms developed in the previous chapter with RML method to develop two efficient sampling algorithms.

4.3 Truncated SVD Algorithms for RML

The SVD parameterization algorithms developed in Chapter 3 could be applied directly to generate a suite of N_e conditional realizations of the pdf of Eq. 3.6 by minimizing the set of objective functions given by Eq. 4.4. However, each of these individual N_e minimizations would require its own independent sequence of truncated SVD computations using the Lanczos algorithm. Each call of Lanczos requires roughly the equivalent of $(p+5)/2$ reservoir simulation runs, and thus, great savings in computationally efficiency can be achieved if the number of truncated singular value decompositions is significantly reduced. One way to do this is to use the same SVD parameterization for all realizations. This means we effectively minimize all N_e objective functions of Eq. 4.4 simultaneously, but at the lth iteration, all $\delta\tilde{m}_{c,i}$, $i = 1, 2, \cdots N_e$ are required to lie in the same subspace, the subspace spanned by the right singular vectors of one specific dimensionless sensitivity matrix.

There appear to be many ways to select the specific sensitivity matrix used at each iteration. In the results presented here, we use the truncated SVD associated with the MAP estimate at every iteration until we obtain $O_{N,\mathrm{MAP}}(m_c^{l+1}) < 1 + 5\sqrt{2/N_d}$ (see Eq. 3.64) and then from this point on, we switch to using the truncated SVD corresponding to the $m_{c,i}^{l+1}$ which gives the highest value of its normalized objective function.

4.3.1 First RML Algorithm

Throughout, we let m_c^l denote the approximation of the MAP estimate at the lth iteration obtained by applying the LM algorithm to minimize the objective function of Eq. 3.7, and let $m_{c,i}^l$, $i = 1, 2, \cdots N_e$ denote the estimate of the ith RML realization at the l iteration when applying the LM algorithm to minimize $O_{r,i}(m)$ given by Eq. 4.4. We let (λ_j, u_j, v_j), $j = 1, 2, \cdots p$ denote the j singular triplets from the truncated SVD decomposition of the dimensionless sensitivity matrix, $G_D(m_c^l)$, associated with the MAP estimate at the lth iteration. In applying the LM algorithm to generate the MAP estimate the search direction $\delta\tilde{m}_c^{l+1}$ is obtained from Eq. 3.45 whereas, in applying LM to generate the ith RML realization (Eq. 4.5), the search direction is updated by the obvious analogue of Eq. 3.45,

$$\delta\tilde{m}_{c,i}^{l+1} = \sum_{j=1}^{p} \left[\frac{-v_j^T \tilde{m}_{c,i}^l - \lambda_j u_j^T C_D^{-\frac{1}{2}} (g(m_{c,i}^l) - d_{\text{uc},i})}{1 + \gamma_l + \lambda_j^2} \right] v_j, \tag{4.9}$$

for $i = 1, 2, \cdots N_e$. A detailed step-by-step description of this and a second algorithm are provided later.

Throughout we refer the method just described for using SVD-based parameterization when generating realizations with RML as SVD-gradient-free-ensemble-RML which is denoted simply by SVD-GF-EnRML. Note that although the algorithm was derived directly from a gradient-based LM algorithm, neither sensitivities or gradients of the objective functions are explicitly computed. In the next subsection, in an attempt to enhance efficiency, we add an inner iteration in which we explicitly compute the gradient of each objective function that is minimized.

4.3.2 Second RML Algorithm (SVD-EnRML)

Here, we use the same idea as was used for the SVD-Grd algorithm in Chapter 3 to develop a new algorithm for generation of multiple conditional realizations. The main part of the description of this new algorithm is the same as the description of the SVD-Grd algorithm used to compute the MAP estimate. However, in this chapter for computation of multiple conditional realizations, we present the second sampling algorithm in a slightly different form as follows;

As the software we use allows us to easily compute the gradient of the objective function using the adjoint method, we add an inner iteration to the basic procedure computed above in which we attempt to improve the data match by iterating using the gradient of the objective function that is being minimized, while keeping the SVD parameterization (computed in the "outer loop") fixed.

The right and left hand sides of Eq. 4.5, respectively represent the gradient of the objective function and the "Levenberg-Marquardt Hessian" for minimization in terms of the transformed model $\tilde{m}_i$, Eq. 4.6. In the inner loop, we use a fixed set of SVD-triplets to evaluate the Hessian at each iteration, but update the gradient on the right-hand side. We also use $\tilde{C}_D$ instead of data covariance matrix C_D in the inner loop, where $\tilde{C}_D$ contains artificially high variance of production data measurement errors in the main diagonal [19]. Under these conditions, Eq. 4.5 applied to the estimation of a single realization can be written as

$$\delta\tilde{m}_{c,i}^{k+1} = \sum_{j=1}^{p} \left[\frac{-v_j^T \tilde{m}_{c,i}^k - v_j^T L^T G_k^T \tilde{C}_{D,i}^{-1}(g(m_{c,i}^k) - d_{\mathrm{uc},i})}{1 + \gamma_k + \lambda_j^2} \right] v_j, \tag{4.10}$$

where in this inner loop iteration, k denotes the iteration index. Note this inner iteration also uses an LM algorithm with parameter γ_k for minimization and is applied for $i = 1, 2, \cdots N_e$. The inflated covariance matrix may also depend on i so we have added a subscript i to it also. In Eq. 4.10, we need to calculate the product of the transpose of sensitivity matrix, G_k^T, with the vector $\tilde{C}_{D,i}^{-1}(g(m_{c,i}^k) - d_{\mathrm{uc},i})$. This product can be obtained by one adjoint solution [66].

The data covariance matrix, C_D, is diagonal for the examples considered in this study and following Gao and Reynolds [19], $\tilde{C}_{D,i}$ is obtained by multiplying the nth diagonal entry of C_D by ψ_n^2 where

$$\psi_n = \max\Big[1, \Big|\frac{g_n(m_{c,i}^0) - [d_{\mathrm{uc},i}]_n}{3\sigma_{d,n}}\Big|\Big], \tag{4.11}$$

where $\sigma_{d,n}$ denotes the standard deviation of the nth measurement error, $m_{c,i}^0$ is the last updated model obtained at the outer iteration, $g_n(m_{c,i}^0)$ is the nth component of the predicted data vector evaluated at $m_{c,i}^0$ and $[d_{\mathrm{uc},i})]_n$ denotes the nth entry of the perturbed data vector $d_{\mathrm{uc},i}$. In applying the iterative scheme of Eq. 4.10 for $k = 0, 1, 2, \cdots$, $\tilde{C}_{D,i}$ does not change from iteration to iteration. This effectively means that in the inner loop, we are effectively trying to find a $\delta\tilde{m}_{c,i}$ in the subspace spanned by the right singular vectors $\{v_j\}_{j=1}^{p}$ and corresponding $m_{c,i}$ that minimizes the "damped" objective function

$$O_{\mathrm{damp},i}(m) = \frac{1}{2}(m - m_{\mathrm{uc},i})^T C_M^{-1}(m - m_{\mathrm{uc},i}) + \frac{1}{2}(g(m) - d_{\mathrm{uc},i})^T \tilde{C}_{D,i}^{-1}(g(m) - d_{\mathrm{uc},i}), \tag{4.12}$$

using as the initial guess the model, $m_{c,i}^0$ obtained from the outer iteration. This is applied for each RML-generated realization, i.e,, for $i = 1, 2, \cdots N_e$. At early iterations, data mismatch terms often are extremely large and if some damping is not done, the regularization effect of the prior model is completely lost. Even though the LM algorithm can control the step size from iteration to iteration, we nevertheless find that some form of data damping at early iterations often improves the convergence performance of gradient based optimization algorithms. If the initial C_D gives a true representation of the noise in the data as it does in the examples presented here, as the iterations proceed, we obtain a sufficiently good match of data so that $\psi_n = 1$ for all n, $\tilde{C}_{D,i} = C_D$ and then the damped objective function becomes equal to the actual objective function we wish to minimize to obtain the ith conditional realization. Finally, we note that at early iterations in the inner loop, we are using a fairly small number of singular triplets in our parameterization so it may not be necessary to enforce tight convergence tolerances during minimization of the damped objective function. This modified algorithm is referred to as the SVD-EnRML algorithm.

4.3.3 Steps of the SVD-EnRML and SVD-GF-EnRML Algorithms

Here we give the specific steps for the implementation of our SVD-EnRML algorithm. After its presentation, we will show how to convert the steps to the SVD-GF-EnRML algorithm.

1. Using l as iteration index for outer loop. Set $l = 0$ and assign the initial guess of $m_{c,i}^0 = m_{\mathrm{uc},i}$ for $i = 1, 2, \cdots N_e$ for the RML realizations and an initial guess of $m_c^0 = m_{\mathrm{prior}}$ for the MAP estimate. Set the sv-cut = 0.10. Also, set the final value of the singular cutoff, sv-cut$_{\mathrm{f}}$. In the synthetic examples presented in this dissertation, we use sv-cut$_{\mathrm{f}}$ = 0.0004.

2. To obtain the initial values of the objective functions, we run the simulator forward to the final data assimilation time with initial guesses of the model parameters (as assigned in the previous step) for the MAP estimate and all realizations and compute $O_{r,i}(m_{c,i}^l)$ with Eq. 4.4 for $i = 1, 2, \cdots N_e$ and compute $O(m_c^l)$ from Eq. 3.7.

3. If $O_{N,\mathrm{MAP}}(m_c^l) < 1 + 5\sqrt{2/N_d}$ or if convergence criteria of Eqs. 3.60 and 3.61 are both satisfied, call the Lanczos algorithm to compute the truncated-SVD of the dimensionless sensitivity matrix pertaining to $m_n^l \equiv m_{c,k}^l$ for some k such that

$$O_{r,n}(m_n^l) = \max_{1 \le i \le N_e} O_{r,i}(m_{c,i}^l). \tag{4.13}$$

   Otherwise, use the Lanczos algorithm to compute the truncated-SVD of the dimensionless sensitivity matrix associated with MAP estimate at iteration l, pertaining to m^l.

4. For $i = 1, 2, \cdots N_e$

   (a) Use Eq. 4.9 together with

$$m_{c,\mathrm{temp},i}^{l+1} = m_{c,i}^l + L\delta\tilde{m}_{c,i}^{l+1} \tag{4.14}$$

   to calculate a proposed new update of the ith RML realization.

(b) Calculate $O_{r,i}(m^{l+1}_{c,\text{temp},i})$ by running the simulator. If $O_{r,i}(m^{l+1}_{c,\text{temp},i}) \geq O_{r,i}(m^{l}_{c,i})$, then increase γ by a factor of 10 and return to step a. If $O_{r,i}(m^{l+1}_{c,\text{temp},i}) < O_r(m^{l}_{c,i})$, then set

$$m^{l+1}_{c,i} = m^{l+1}_{c,\text{temp},i}, \tag{4.15}$$

$$\tilde{m}^{l+1}_{c,i} = \tilde{m}^{l}_{c,i} + \delta\tilde{m}^{l+1}_{c,i}. \tag{4.16}$$

and decrease γ by a factor of 10 for the next iteration of the ith realization. Note each realization is generated by minimizing a different objective function and the value of γ may be different in each of the N_e minimizations. This same procedure (steps a and b) is used to obtain the new estimate, m^{l+1}_c of the MAP estimate. (Once we obtain a decrease in each of the $N_e + 1$ objective functions, if convergence has not been obtained, we go to the inner iteration.)

5. Check for convergence using the criteria discussed later. If the algorithm has not converged, increase the iteration index l by 1 and go to step 6 which is the first step of the inner loop. Otherwise, go to step 7, i.e., terminate the algorithm.

6. For $i = 1, 2, \cdots N_e$

   (a) With k denoting the iteration index of the inner loop, set $k = 0$ and set the initial guess of the model parameter for the inner loop as $m^0_{c,i} = m^l_{c,i}$, where $m^l_{c,i}$ denotes the last updated model obtained at the outer loop.

   (b) Compute damping factors by using Eq. 4.11, then compute damped objective function $O_{\text{damp},i}(m^k_{c,i})$ by Eq. 4.12 for each i.

   (c) Call the adjoint gradient subroutine to calculate the product of G^T_k with the vector $\tilde{C}^{-1}_{D,i}(g(m^k_{c,i}) - d_{\text{uc},i})$.

   (d) Use Eq. 4.10 together with

$$m_{c,\text{temp},i}^{k+1} = m_{c,i}^{k} + L\delta\tilde{m}_{c,i}^{k+1} \tag{4.17}$$

to calculate a proposed new update of the ith RML realization, $m_{c,\text{temp},i}^{k+1}$.

(e) Run the simulator forward to calculate the corresponding value of the damped objective function, $O_{\text{damp},i}(m_{c,\text{temp},i}^{k+1})$.

(f) if $O_{\text{damp},i}(m_{c,\text{temp},i}^{k+1}) \geq O_{\text{damp},i}(m_{c,i}^{k})$, then increase γ by a factor of 10 and return to step d. If $O_{\text{damp},i}(m_{c,\text{temp},i}^{k+1}) < O_{\text{damp},i}(m_{c,i}^{k})$, then set $m_{c,i}^{k+1} = m_{c,\text{temp},i}^{k+1}$, decrease γ by a factor of 10 and go to the next step.

(g) Check for convergence of inner loop. The computational cost of each successful iteration in the inner loop is one forward simulation run and one adjoint solution, which is much less than the cost of one iteration of the outer loop. In our history matching examples, we have used the following convergence criterion

$$\frac{O_{\text{damp},i}(m_{c,i}^{k}) - O_{\text{damp},i}(m_{c,i}^{k+1})}{O_{\text{damp},i}(m_{c,i}^{k})} < 5 \times 10^{-3}. \tag{4.18}$$

If the condition given by Eq. 4.18 is not satisfied, increase the k index by one and return to step c. We implement the same steps of the inner loop to obtain the new update of the model parameters for the MAP estimate. Once we have obtained convergence of all damped objective functions, replace sv-cut by $\max\left\{\frac{\text{sv-cut}}{2}, \text{sv-cut}_\text{f}\right\}$, set $m_{c,i}^{l} = m_{c,i}^{k}$ for each i and go to step 3.

7. end

To implement SVD-GF-EnRML, we delete step 6 of the algorithm described above and replace step 5 by

Step 5: Check for convergence using the criteria discussed later. If the algorithm has not converged, increase iteration index l by 1 and go to step 3. Otherwise terminate the algorithm.

4.3.4 Convergence Criteria.

The convergence criteria for termination of the SVD-GF-EnRML and the SVD-EnRML algorithms are based on the requirement that both the maximum relative change in the objective function and the maximum relative change in the model parameters of all realization be small, i.e.,

$$dO(m) = \max_{1 \le j \le N_e} \left\{ \frac{|O_{r,j}(m_{c,j}^{l+1}) - O_{r,j}(m_{c,j}^{l})|}{O_{r,j}(m_{c,j}^{l+1})} \right\} < \eta_o \tag{4.19}$$

and

$$dm = \max_{1 \le j \le N_e} \left\{ \frac{\|m_{c,j}^{l+1} - m_{c,j}^{l}\|_2}{\|m_{c,j}^{l+1}\|_2} \right\} < \eta_m \tag{4.20}$$

In the examples presented in this work, we have used the following values:

$$\eta_o = 10^{-3}, \tag{4.21}$$

and

$$\eta_m = 10^{-2}. \tag{4.22}$$

Note that when we are calculating a conditional realization with RML or any truncated-SVD parameterization algorithms, the normalized objective function is defined by

$$O_{N,i}(m) = \frac{O_{r,i}(m)}{N_d}. \tag{4.23}$$

4.4 Ensemble Kalman Filter (EnKF)

Recently, the ensemble Kalman filter (EnKF) has been investigated as a reservoir management tool for data assimilation and assessment of uncertainty in future forecasts. The EnKF method was originally introduced by Evensen [12] as a sequential data assimilation algorithm. The EnKF sequential algorithm consist of two major steps, first, a forecast step which is equivalent to running the simulation to predict data at the next data assimilation time step, second, an analysis or updated step which includes the implicit calculation of the Kalman gain matrix and update of the model parameters and simulation variable so that they are consistent with data and with each other. The EnKF can easily be coupled with any reservoir simulator. Neither adjoint code nor specific knowledge of simulator is required for implementation of the EnKF. Furthermore, dynamic data (production data) is assimilated continuously in time as they become available and covariances are updated.

When applying EnKF, we consider the reservoir model parameter m and state variables (grid block pressures and saturations) p_n in an augmented state vector, y_n, i.e.,

$$y_n = \begin{bmatrix} m \\ p_n \end{bmatrix}. \tag{4.24}$$

Note that the subscript n denotes the time at which we wish to assimilate data. All vectors are column vectors with the dimension of m given by N_m and the dimension of p_n given by N_p. Thus, the dimension of y_n, denoted by N_y is given by $N_y = N_m + N_p$.

For simplicity in the notations, we assume that we assimilate data at every simulation time step. Therefore, the update equation for each ensemble member by the traditional Kalman filter [32] update is given by

$$y^a_{n,j} = y^f_{n,j} + C_{Y^f_n D^f_n}(C_{D^f_n D^f_n} + C_{D_n})^{-1}(d^j_{n,\mathrm{uc}} - d^{f,j}_n) \quad \text{for } j = 1, 2, \cdots N_e, \tag{4.25}$$

where $C_{Y^f_n D^f_n}$ denotes the covariance between the state vector y_n and predicted data at time n, $C_{D^f_n D^f_n}$ represents auto-covariance for predicted data, $d^{f,j}_n$ denotes predicted data at time n corresponding to the vector of observed data we wish to assimilate and $d^j_{n,\mathrm{uc}}$ is a sample of the normal distribution $N(d_{n,\mathrm{obs}}, C_{D_n})$ where $d_{n,\mathrm{obs}}$ represents observed data and C_{D_n} denotes the data covariance matrix at data assimilation time n. Here, the superscript f denotes forecast or predicted and the superscript a refers to the result of the analysis or update data assimilation step, e.g., $y^a_{n,j}$ represents updated state vector at time n by assimilating data $d^j_{n,\mathrm{uc}}$.

The vector consisting of the first N_m components of $y^a_{n,j}$ pertains to the model parameter m. It is well known [12] and straightforward to show that the model part of $y^a_{n,j}$ is given by

$$m^a_j = m^f_j + C_{MD^f_n}(C_{D^f_n D^f_n} + C_{D_n})^{-1}(d^j_{n,\mathrm{uc}} - d^{f,j}_n), \quad \text{for } j = 1, 2, \cdots N_e, \tag{4.26}$$

where $C_{MD^f_n}$ denotes the covariance between the model m and predicted data at time n.

4.4.1 Covariance localization

In the implementation of the EnKF technique, we use a limited number of realizations to represent approximations of the relevant covariance matrices. As is well known [28, 16], this can lead to spurious correlations between an individual datum and components of the state vector separated by large distances. When assimilating data, such spurious correlations can result in changes in components of the state vector which are insensitive to the data. A second problem occurs when the number of data are large. The limited degrees of freedom available to assimilate data due to the limited number of ensemble members can result in loss of rank [41] and filter divergence where eventually all updated state vectors become essentially the same but do not give a good prediction of subsequent data. The result is that the filter becomes unable to assimilate additional data and biased predictions with unrealistically low uncertainty are obtained.

Increasing the ensemble size could reduce spurious correlations and increase the degrees of freedom available to assimilate data but at the cost of reduced computational efficiency. A more standard way to achieve these effects is to apply covariance localization [21, 44, 28, 16]. In this approach, we form the Schur or Hadamard product of a positive-definite matrix ρ with the ensemble-based estimate to produce a localized covariance estimate

$$C_{y,\rho} = \rho \circ C_Y. \tag{4.27}$$

The entries of the matrix ρ are generated from a correlation function with compact support. By replacing C_y by $\rho \circ C_Y$, we zero long distance covariances corresponding to spurious correlation. The (i,j) entry of $C_{y,\rho}$ is the product of the (i,j) entry of ρ and the (i,j) entry of C_Y. Note that the dimension of the correlation matrix ρ is $N_y \times N_y$.

With covariance localization, the EnKF update equation of Eq. 4.25 is approximated by

$$y^a_{n,j} = y^f_{n,j} + \rho_{yd} \circ C_{Y^f_n D^f_n}(\rho_{dd} \circ C_{D^f_n D^f_n} + C_{D_n})^{-1}(d^j_{n,\mathrm{uc}} - d^{f,j}_n), \tag{4.28}$$

for $j = 1, 2, \cdots N_e$, where ρ_{yd} denotes the correlation matrix between the state vector and data and ρ_{dd} represents auto-correlation matrix for predicted data.

Covariance localization allows one to eliminate long distance spurious correlations between entries of the EnKF state vector due to sampling error. In addition, covariance localization increases the rank of the forecast covariance matrix and thus provides more degrees of freedom for assimilating production data [28, 14]. In our application of covariance localization, the correlation function (ρ) used is the two-dimensional form of the fifth-order compact function of Gaspari and Cohn [21] defined as

$$\rho(L_D) = \begin{cases} -\frac{1}{4}L_D^5 + \frac{1}{2}L_D^4 + \frac{5}{8}L_D^3 - \frac{5}{3}L_D^2 + 1,\ 0 \le L_D \le 1; \\ \frac{1}{12}L_D^5 - \frac{1}{2}L_D^4 + \frac{5}{8}L_D^3 + \frac{5}{3}L_D^2 - 5L_D + 4 - \frac{2}{3}L_D^{-1}, \quad 1 < L_D \le 2; \\ 0, \quad L_D > 2, \end{cases} \tag{4.29}$$

where L_D is the dimensionless correlation length which is calculated as shown below in the general anisotropic case. We let α denote the angle between the positive x-axis and the principal direction (u-direction) which refers to the direction in which the correlation length is the greatest. The v-direction is perpendicular to the u-direction; a_u and a_v, respectively, denote the correlation lengths in the u and v directions. (In the isotropic case, the correlation lengths a_u and a_v are identical and we may take $\alpha = 0$.)

Suppose that $c_{i,j}$, which denotes here the (i,j) entry of $C_{Y_n^f,D_n^f}$, represents the covariance between a component of the state vector located at areal location (x_1, y_1) and data at observation location (x_2, y_2) and let $L_x = |x_2 - x_1|$ and $L_y = |y_2 - y_1|$. To compute $\rho_{y,i,j}$ we calculate

$$\begin{cases} L_u = L_x \cos\alpha + L_y \sin\alpha \\ L_v = -L_x \sin\alpha + L_y \cos\alpha \end{cases}, \tag{4.30}$$

$$L_D = \left[\left(\frac{L_u}{a_u}\right)^2 + \left(\frac{L_v}{a_v}\right)^2\right]^{0.5}, \tag{4.31}$$

and then calculate

$$\rho_{y,i,j} = \rho(L_D), \tag{4.32}$$

using Eq. 4.29. Then the (i,j) entry of $\rho_{yd} \circ C_{Y_n^f,D_n^f}$ is given by $\rho_{yd,i,j}c_{i,j}$. The entries of $\rho_{dd} \circ C_{D_n^f D_n^f}$ in Eq. 4.28 are calculated similarly. Note that to obtain the (i,j) entry of ρ_{yd}, we need to calculate the distance between the ith entry of the state vector and the jth entry of the predicted data.

We can also apply covariance localization in a slightly nonstandard form in which the Schur product of ρ is applied directly to the Kalman gain matrix defined as

$$K_n^f = C_{Y_n^f} H_n^T (H_n C_{Y_n^f} H_n^T + C_{D_n})^{-1}. \tag{4.33}$$

Therefore, with the Kalman gain localization, the EnKF update equation of Eq. 4.25 is approximated by

$$y_n^{a,j} = y_n^{f,j} + (\rho_{yd} \circ K_n^f)(d_{n,\mathrm{uc}}^j - d_n^{f,j}) = y_n^{f,j} + \rho_{yd} \circ \left[C_{Y_n^f, D_n^f} (C_{D_n^f D_n^f} + C_{D_n})^{-1} \right] (d_{n,\mathrm{uc}}^j - d_n^{f,j}), \tag{4.34}$$

for $j = 1, 2, \cdots N_e$.

4.5 Computational Results

In this section, we present results obtained from implementation of the SVD-GF-EnRML and SVD-EnRML algorithms for the same 2-dimensional synthetic examples described in Chapter 3. The objective function behavior, final estimates of the log-permeability fields, history matches and prediction results and the computational costs are compared for both algorithms.

We also present the results obtained from implementation of the EnKF without and with covariance localization for different ensemble sizes. We have tried to investigate the effects of covariance localization on the final estimate of model parameters, data matches and future forecasts with anisotropic correlation function (Example 1) as well as the standard implementation of the covariance localization versus Kalman gain localization (Example 2). Results with SVD parameterization algorithms are compared with results obtained with EnKF.

Finally, we present the results obtained with combined EnKF algorithm and SVD parameterization algorithms where EnKF is used to generate an initial guess for the truncated-SVD algorithms.

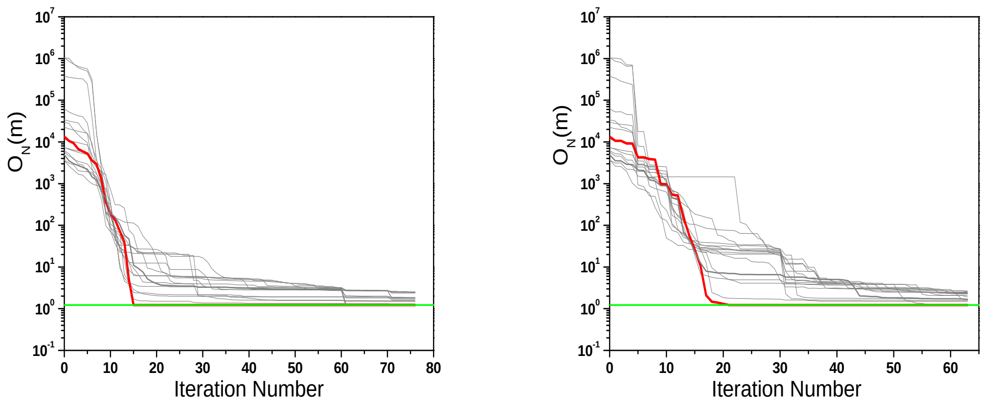
4.5.1 SVD-GF-EnRML and SVD-EnRML Simulation Results of Example 1

In order to quantify the uncertainty in history matching and future forecasts, we need to generate plausible realizations of model parameters that honor the observed data. We have implemented the SVD-GF-EnRML and SVD-EnRML techniques for Example 1 to generate 15 realizations ($N_e = 15$). We have generated prior unconditional realizations from the square root (Cholesky decomposition) of the prior covariance matrix.

We use the same strategy to compute the truncated SVD in both SVD-GF-EnRML and SVD-EnRML algorithms. We start with sv-cut $= 0.1$ at the first iteration and decrease sv-cut at each (outer) iteration by dividing by two until we reach the final value of sv-cut $= 0.0004$. Then from this point on, we iterate with fixed value sv-cut $= 0.0004$ until we obtain convergence of the MAP estimate. Once the MAP estimate has converged, we switch to using the truncated SVD corresponding to the realization which gives the highest value of the normalized objective function at each subsequent iteration. At this point, we set sv-cut $= 0.1$ and repeat the strategy of changing the sv-cut from iteration to iteration in the same way as we did for the MAP estimate.

Figures 4.1(a) and 4.1(b), respectively, show the behavior of the normalized objective function from iteration to iteration for all the realizations using the SVD-GF-EnRML and SVD-EnRML algorithms. During the early iterations, where the singular triplets correspond to the dimensionless sensitivity matrix associated with calculation of the MAP estimate, the normalized objective function of each realization decreases sharply. The results of Figure 4.1(a) indicate that the MAP has converged in 15 iterations based on the criteria given in Eqs. 3.60 and 3.61. From this point on, we calculate the singular triplets of the dimensionless sensitivity matrix of the realization with the highest value of the normalized objective function at each subsequent iteration. The final value of the normalized objective function is less than 2.40 for all of the realizations at the final iteration, but only three of the realizations satisfy the condition given by Eq. 3.64 which indicates that all normalized objective function values should be less than or equal to 1.24.

Note in Figure 4.1(b), the x-axis represents the iteration number of both the outer and the inner loops. Specifically, odd values of the iteration index correspond to the result from the outer loop and even values of the iteration index correspond to the final result obtained from an inner loop iteration. At convergence, the maximum value of the normalized objective function among all realizations is $O_{N,\max} = 2.70$ for the SVD-EnRML algorithm.



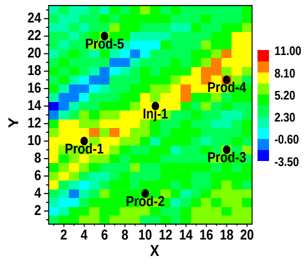
(a) SVD-GF-EnRML.

(b) SVD-EnRML.

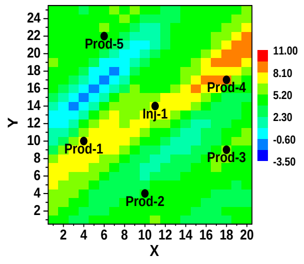
Figure 4.1: Behavior of the normalized objective function during iteration of SVD-GF-EnRML and SVD-EnRML algorithms, red curve represents MAP estimate, gray curves represent realizations, Example 1.

The final average of log-permeability fields (the average of all realizations of $\ln(k)$ field) obtained from the SVD-GF-EnRML and SVD-EnRML algorithms are respectively shown in Figures 4.2(b) and 4.2(c). Note that the final fields are comparable to the truth, there is no overshooting and undershooting in the fields and the features of the channel and low permeability barrier have been captured correctly compared to the truth (Figure 4.2(a)).

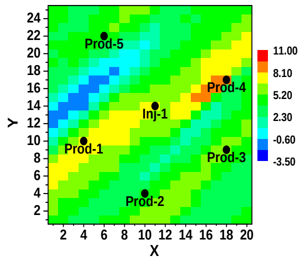
Three unconditional log-permeability fields generated from the prior covari-



(a) True Ln(k).



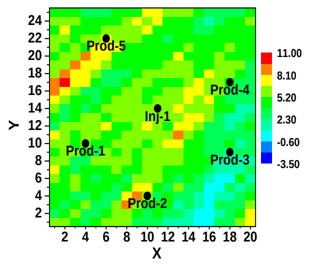
(b) SVD-GF-EnRML.



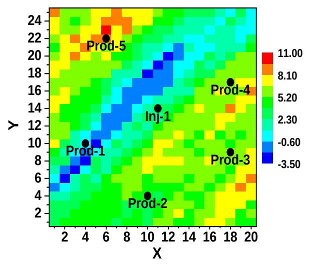
(c) SVD-EnRML.

Figure 4.2: True $\ln(k)$ field and final average of realizations of log-permeability from SVD-GF-EnRML and SVD-EnRML algorithms, Example 1.

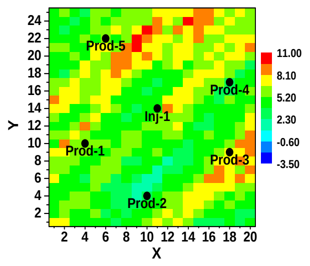
ance matrix are shown in Figure 4.3. The final conditional log-permeability fields corresponding to these three realizations obtained with the SVD-GF-EnRML and SVD-EnRML algorithms, respectively, are shown in Figures 4.4 and 4.5. Although the three realizations exhibit distinct differences, the essential features of the channel and the barrier that exist in the truth have been captured in each individual conditional realization of Figures 4.4 and 4.5.



(a) Realization 5.



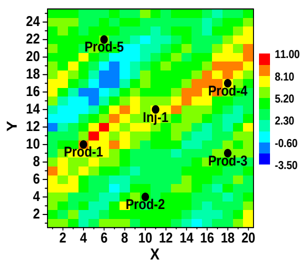
(b) Realization 10.

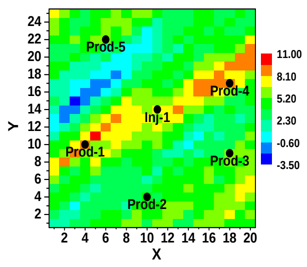


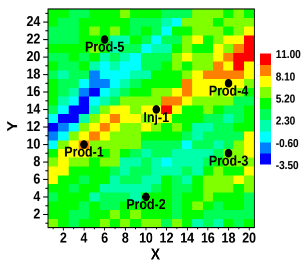
(c) Realization 13.

Figure 4.3: Unconditional realizations of log-permeability fields, Example 1.

The BHP data matches and future forecast for Inj-1 obtained with the SVD-GF-EnRML algorithm are shown in Figure 4.6(a). Note that in this and similar figures, the vertical line denotes the end of the history matching period. As can be





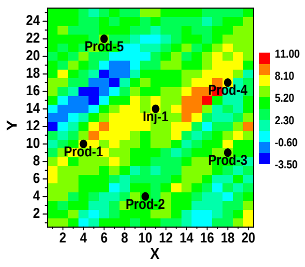


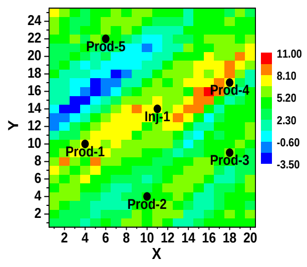
(a) Realization 5.

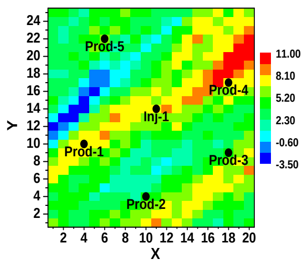
(b) Realization 10.

(c) Realization 13.

Figure 4.4: Conditional realizations of log-permeability field from SVD-GF-EnRML algorithm, Example 1.







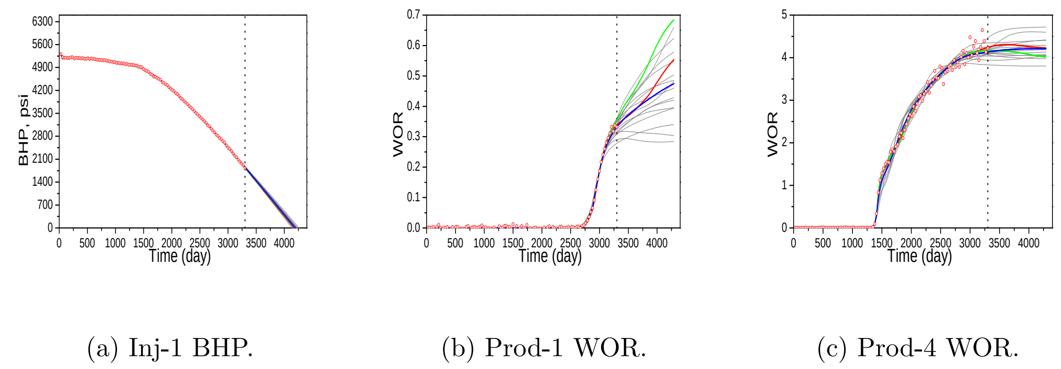
(a) Realization 5.

(b) Realization 10.

(c) Realization 13.

Figure 4.5: Conditional realizations of log-permeability field from SVD-EnRML algorithm, Example 1.

seen from this figure, good BHP matches are obtained for all realizations and also the prediction of BHP is in good agreement with the true prediction. The BHP data matches and performance predictions for other producers (not shown) are similar to Inj-1. The history matching results and future forecasts of the WOR of Prod-1 and Prod-4 are shown in Figures 4.6(b) and 4.6(c), respectively. Again good data matches are obtained and the predicted data from the true model falls near the center of the span of predictions from the set of realizations.

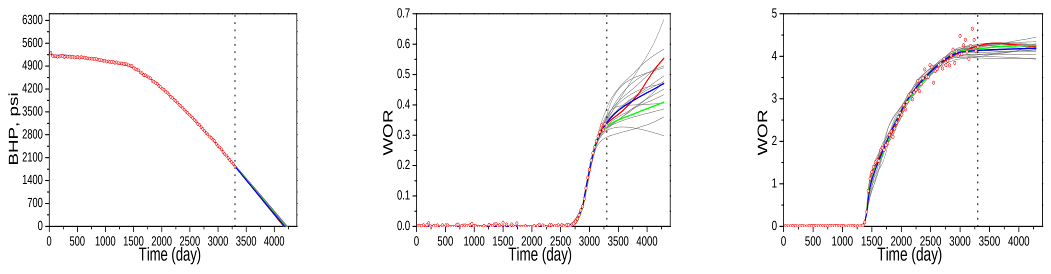


(a) Inj-1 BHP. (b) Prod-1 WOR. (c) Prod-4 WOR.

Figure 4.6: Data matches and future prediction from SVD-GF-EnRML, open red circles are observed data, red is from true model, blue is the prediction mean, green is from the MAP estimate, gray curves represent predictions from realizations, Example 1.

Comparable matches and predictions obtained with the SVD-EnRML method are shown in Figure 4.7. These results are essentially of the same quality as to those shown in Figure 4.6.

Table 4.1 gives the number of simulation runs, number of adjoint runs and the number of forward gradient runs for the SVD-GF-EnRML and SVD-EnRML algorithms. The total computational cost based on the number of equivalent simulation runs is given in the last column of the table. From the results, we see that the SVD-GF-EnRML algorithm requires about 35% more computational effort than the SVD-EnRML algorithm. This result is consistent with other examples that we have done in that SVD-EnRML is always more efficient, but is some cases, the com-



(a) Inj-1 BHP. (b) Prod-1 WOR. (c) Prod-4 WOR.

Figure 4.7: Data matches and future prediction from SVD-EnRML, open red circles are observed data, red is from true model, blue is the prediction mean, green is from the MAP estimate, gray curves represent predictions from realizations, Example 1.

putational cost of the two algorithms is essentially the same. The results in the third row of Table 4.1 are based on using EnKF to generate an initial guess for SVD-GF-EnRML, a procedure which is discussed in detail later.

Table 4.1: Summary of computational costs of the SVD-GF-EnRML and SVD-EnRML algorithms, Example 1.

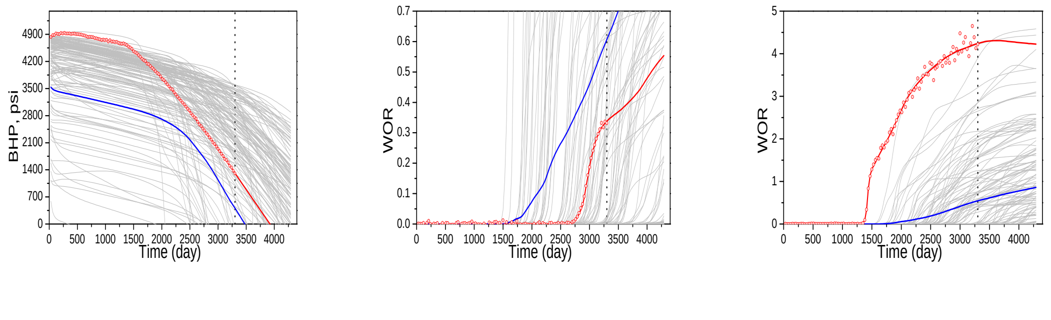
Algorithm	Simulations	Forward Runs	Adjoint	Equ. Sim. Runs
SVD-GF-EnRML	2463	2732	2658	3810
SVD-EnRML	2078	1067	1618	2749
SVD-GF-RML-EnKF	1131	1020	992	1634

4.5.2 EnKF Results of Example 1

We now consider comparable results obtained with the ensemble Kalman filter (EnKF) with three different ensemble sizes of $N_e = 50$, $N_e = 100$ and $N_e = 150$. We first generated an initial ensemble for the $N_e = 150$ case. Then for the $N_e = 50$, we selected the first 50 realizations of these 150 unconditional realizations. For the $N_e = 100$ case, we selected the first 100 realizations of these 150 unconditional realizations. In the $N_e = 150$ case, EnKF requires 150 simulation runs to assimilate the data and another 150 reruns from time zero to ensure that we have not introduced inconsistency due to nonlinearity or non-Gaussianity [60]. Note that we have one additional realization corresponding to the $m_{\mathrm{uc},N_e+1} = m_{\mathrm{prior}}$ and $d_{\mathrm{uc},N_e+1} = d_{\mathrm{obs}}$ in all three ensemble sizes used here for EnKF. We consider both the standard implementation of EnKF for the combined parameter-state estimation [13, 14] as well as EnKF with covariance localization [28] based on the anisotropic form of the Gaspari and Cohn [21] correlation function. We generate the anisotropic Gaspari-Cohn correlation function so that its principle directions and correlation lengths are aligned with the prior covariance function for the log-permeability field. In the Gaspari-Cohn correlation function used in this example, the critical length in the principle direction is set equal to 8400 feet and the critical length in the orthogonal direction is set equal to 1500 feet. Although not shown, experiments with larger and smaller critical lengths did not yield improved results.

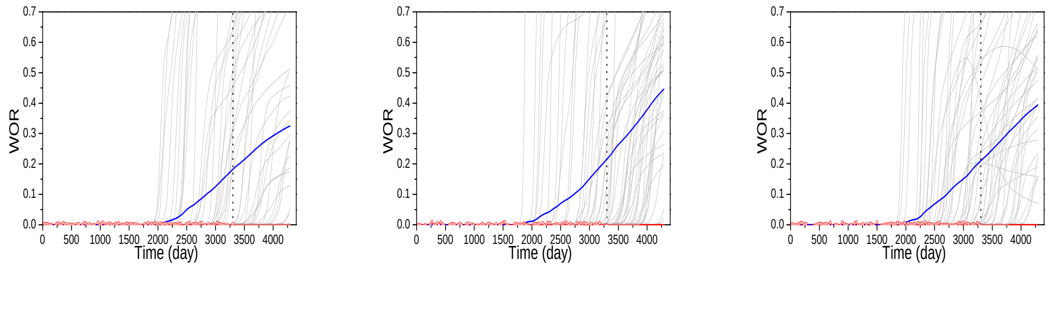
In order to obtain an estimate of the initial (prior) uncertainty of the predicted data compared to the observed data, we have run the simulator forward up to end time with the initial realizations of the log-permeability field. The uncertainty in the BHP of Prod-4 based on predictions from the initial ensembles with $N_e = 150$ compared to the true data (red curve) is shown in Figure 4.8(a). The WOR predicted data for Prod-1 and Prod-4 are respectively shown in Figures 4.8(b) and 4.8(c). Prod-1 has early breakthrough with high values of WOR data and Prod-4 has late breakthrough with low values of WOR compared to the truth. The WOR predicted data from initial ensembles model parameters for Prod-2, Prod-3, and Prod-5, respectively, are shown in Figures 4.9(a), 4.9(b) and 4.9(c). For these three producers, there is no water breakthrough with the true model but the ensemble mean and most realizations exhibit breakthrough.

Figures 4.10 and 4.11 show the final update of the ensemble mean of the log-permeability fields as a function of ensemble size obtained with the standard EnKF method without and with covariance localization, respectively. As can be seen from



(a) Prod-4 BHP. (b) Prod-1 WOR. (c) Prod-4 WOR.

Figure 4.8: Matches and predictions of the BHP of Prod-4 and the WOR of Prod-1 and Prod-4 with initial ensemble, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 1.



(a) Prod-2 WOR. (b) Prod-3 WOR. (c) Prod-5 WOR.

Figure 4.9: Matches and predictions of the WOR of Prod-2, Prod-3 and Prod-5 with initial ensemble, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 1.

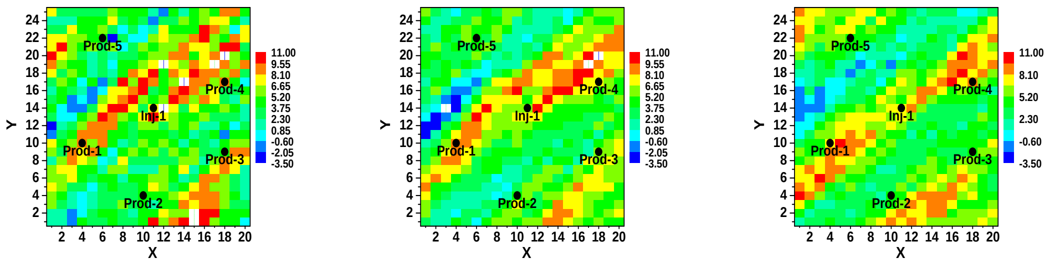
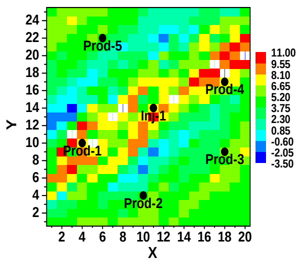

(a) Ne=50. (b) Ne=100. (c) Ne=150.

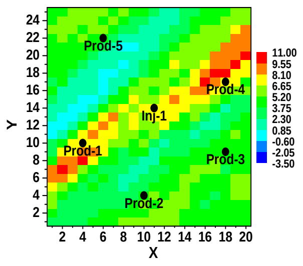
Figure 4.10: Ensemble mean final log-permeability fields as a function of ensemble size without covariance localization, Example 1.

Figures 4.10(a), 4.10(b), and 4.10(c), when we do not use covariance localization, the final estimate of the average model gives a reasonable good approximation of the true log-permeability field only when $N_e = 150$. In this figures, grid blocks colored white indicate $\ln(k)$ values greater or smaller than the truth. Even in the $N_e = 150$ case, there are some high permeability regions near the upper left and lower right corners that do not exist in the truth (Figure 4.2(a)). As Figure 4.11 shows, this problem is removed with covariance localization and we obtain a better estimate of the final model compared to the truth. Although, the covariance localization greatly improved the final model, with ensemble size of $N_e = 50$, we still have some high permeability values in the channel location compared to the truth.

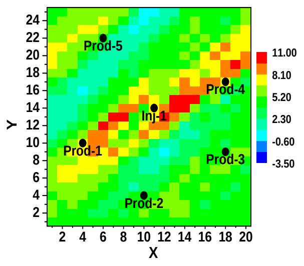
Figure 4.12 shows three different unconditional ensemble members of the log-permeability field. The final updated log-permeability fields corresponding to these unconditional realizations (Figure 4.12) are respectively shown in Figures 4.13 and 4.14 for the case without localization and with anisotropic localization with an ensemble size of $N_e = 150$. As illustrated in Figure 4.13, we obtained realizations of the log-permeability field which shows some basic features of the truth with no localization, however, again there are some regions of high permeability at the south-east part of the reservoir which do not exist in the truth. Moreover, there is a small



(a) Ne=50.



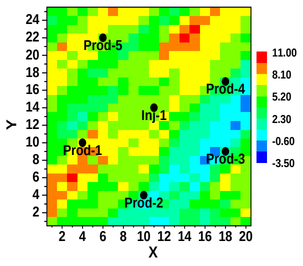
(b) Ne=100.



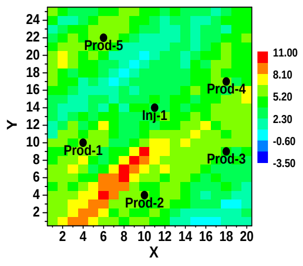
(c) Ne=150.

Figure 4.11: Ensemble mean final log-permeability fields as a function of ensemble size with anisotropic covariance localization, Example 1.

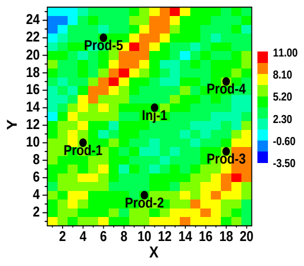
variability between ensemble members when no localization is used. With implementation of covariance localization, (Figure 4.14) the variability between different ensemble members increases and the regions of high permeability values at the south-east corner of the reservoir tends to disappear and we still obtain log-permeability fields which are plausible compared to the true field.



(a) Ensemble 5.



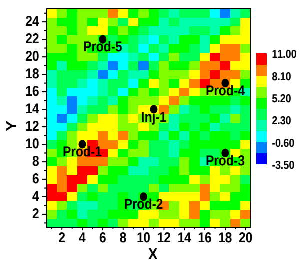
(b) Ensemble 10.



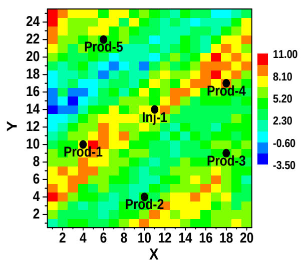
(c) Ensemble 13.

Figure 4.12: Three unconditional log-permeability fields, Example 1.

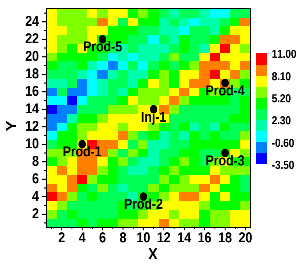
The history matching results during data assimilation and future prediction of the WOR of Prod-1 without using covariance localization, for three different ensemble sizes are presented in Figure 4.15. It is evident from Figures 4.15(a) and 4.15(b) that ensemble sizes of 50 and 100 without covariance localization are insufficient to



(a) Ensemble 5.

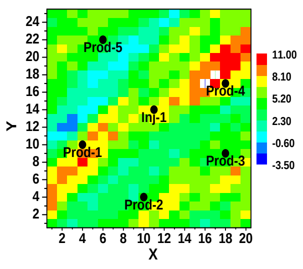


(b) Ensemble 10.

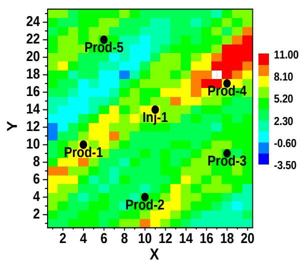


(c) Ensemble 13.

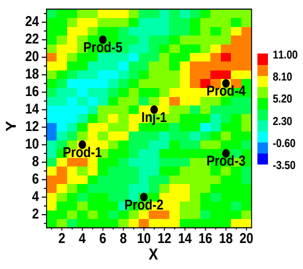
Figure 4.13: Three ensemble members of the final log-permeability fields obtained from EnKF without localization, Ne=150, Example 1.



(a) Ensemble 5.



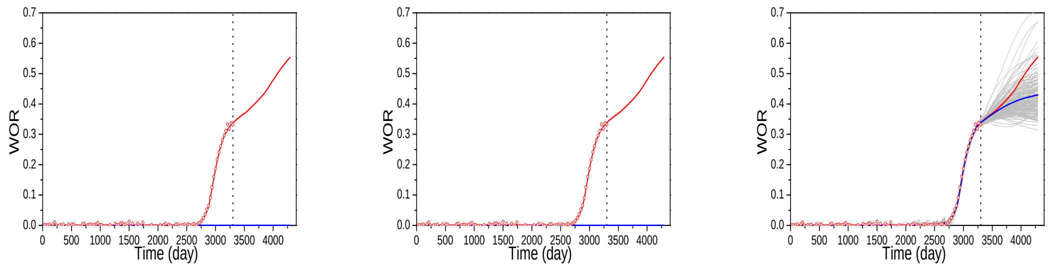
(b) Ensemble 10.



(c) Ensemble 13.

Figure 4.14: Three ensemble members of the final log-permeability fields obtained from EnKF with anisotropic localization, Ne=150, Example 1.

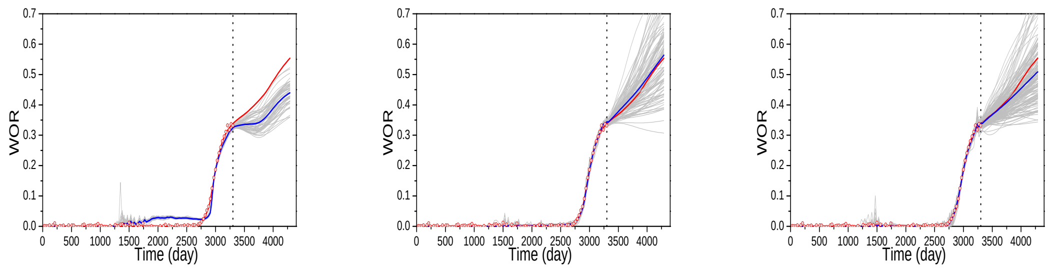
match or predict water breakthrough for Prod-1. However, we obtained a good WOR match with an ensemble size of 150 without localization (Figure 4.15(c)) and future predictions are reasonable and within the band of ensemble prediction. Figure 4.16 shows the history matches and performance prediction of the WOR of Prod-1 obtained with covariance localization during data assimilation period as a function of the ensemble size. With covariance localization, the ensemble means match the data even for an ensemble size of 50 (Figure 4.16(a)) and 100 (Figure 4.16(b)) and better predictions are obtained in all cases than for the comparable results obtained without localization. However, the predictions with an ensemble size of 50 (Figure 4.16(a)), are completely unreliable.



(a) Ne=50. (b) Ne=100. (c) Ne=150.

Figure 4.15: Matches and predictions of WOR of Prod-1 as a function of ensemble size during data assimilation with EnKF without localization, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 1.

In order to investigate the EnKF consistency issue [60], we rerun the simulation forward from time zero with the ensemble of final updated models and compare the performance of the predicted data with the history matching results obtained during the data assimilation phase. The predicted performances of WOR of Prod-1 from the final model obtained by running from time zero as a function of ensemble size are shown in Figure 4.17 for the case without localization. For ensemble size of 150, even though we obtained a good history match during data assimilation (Fig-



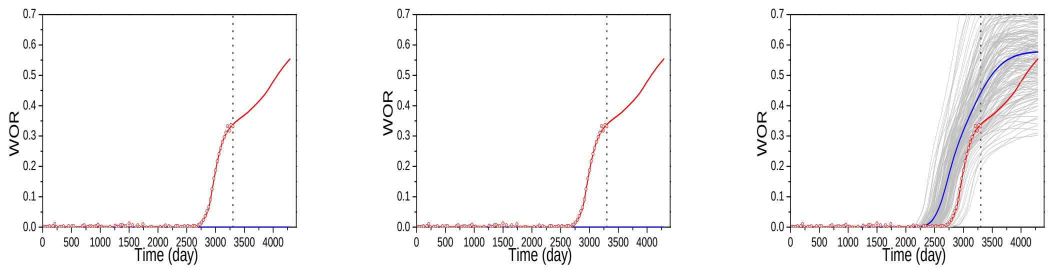
(a) Ne=50. (b) Ne=100. (c) Ne=150.

Figure 4.16: Matches and predictions of WOR of Prod-1 as a function of ensemble size during data assimilation with EnKF with anisotropic localization, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 1.

ure 4.15(c)), when the simulator is rerun from time zero with the final ensemble of permeability fields obtained by assimilating all production data, the WOR predictions are biased and not in good agreement with the true WOR of of Prod-1; the ensemble mean prediction is not close to the truth (Figure 4.17(c)). Inconsistency between the statistical properties of the updated ensemble of primary reservoir simulation variables and the statistical properties of the ensemble of predictions generated by rerunning the reservoir simulator from time zero using the updated realizations of model parameter is a problem that can occur in EnKF when the relation between dynamical variables and model parameters is highly nonlinear [60].

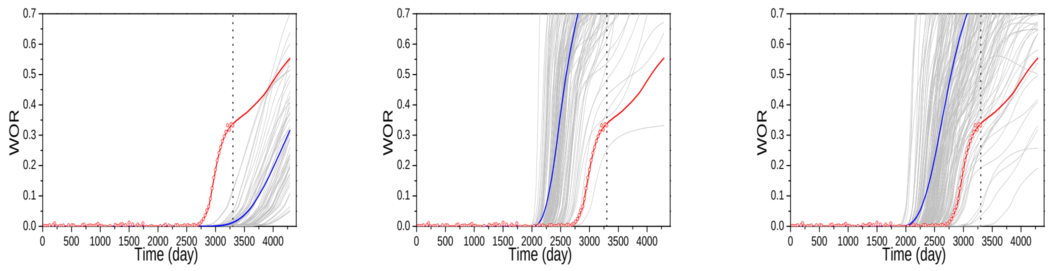
Figure 4.18 shows the predicted performances of WOR of Prod-1 from the final model obtained by running from time zero as a function of ensemble size for the case with anisotropic covariance localization. As we can see from this figure, severe inconsistency occurs with all three ensemble sizes compared to the results of Figure 4.16.

Figures 4.19 and 4.20 show the results of history matching during data assimilation and future forecasts of the WOR of Prod-4, for three different ensemble sizes, without and with anisotropic covariance localization, respectively. As we can



(a) Ne=50. (b) Ne=100. (c) Ne=150.

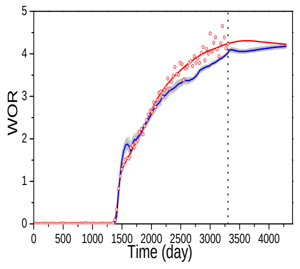
Figure 4.17: Matches and predictions of WOR of Prod-1 as a function of ensemble size for rerun from time zero with EnKF without localization, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 1.



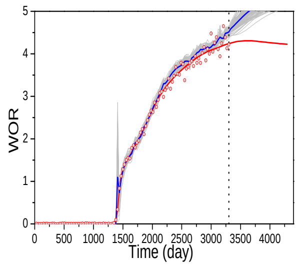
(a) Ne=50. (b) Ne=100. (c) Ne=150.

Figure 4.18: Matches and predictions of WOR of Prod-1 as a function of ensemble size for rerun from time zero with EnKF with anisotropic localization, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 1.

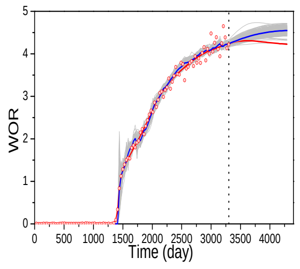
see from Figures 4.20(a), we have obtained very good history matches of WOR data with ensemble size of 50 by using anisotropic covariance localizations unlike the case without localization (Figure 4.19(a)). Again for each values of N_e, covariance localization resulted in the best characterization of future predictions.



(a) Ne=50.

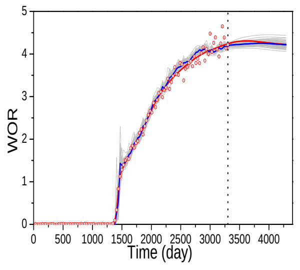


(b) Ne=100.

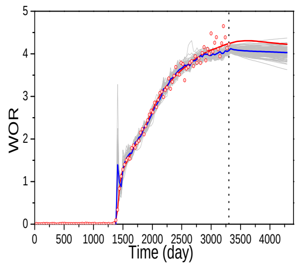


(c) Ne=150.

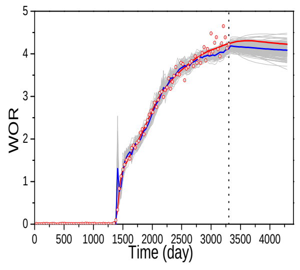
Figure 4.19: Matches and predictions of WOR of Prod-4 as a function of ensemble size during data assimilation with EnKF without localization, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 1.



(a) Ne=50.



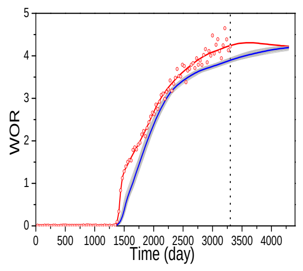
(b) Ne=100.



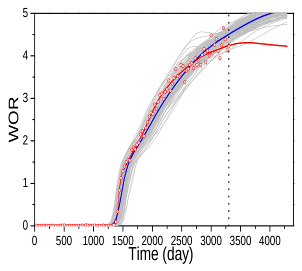
(c) Ne=150.

Figure 4.20: Matches and predictions of WOR of Prod-4 as a function of ensemble size during data assimilation with EnKF with anisotropic localization, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 1.

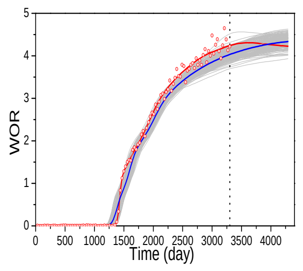
As we can see from Figure 4.21, the predicted performances of the WOR of Prod-1 from the final model obtained by running from time zero as a function of ensemble size for the case without localization are almost consistent with the results obtained during the data assimilation (Figure 4.19). However, the comparison of Figures 4.20 and 4.22 shows that the severe inconsistency occurs in the WOR of Prod-4 with anisotropic covariance localization for ensemble sizes of 100 and 150.



(a) Ne=50.

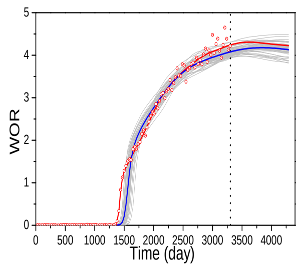


(b) Ne=100.

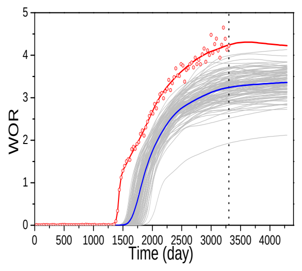


(c) Ne=150.

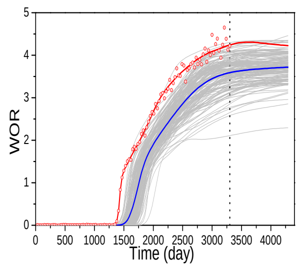
Figure 4.21: Matches and predictions of WOR of Prod-4 as a function of ensemble size for rerun from time zero with EnKF without localization, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 1.



(a) Ne=50.



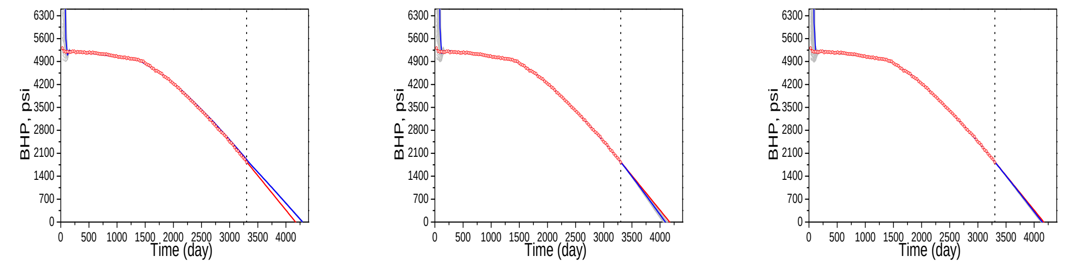
(b) Ne=100.



(c) Ne=150.

Figure 4.22: Matches and predictions of WOR of Prod-4 as a function of ensemble size for rerun from time zero with EnKF with anisotropic localization, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 1.

The history matching results during data assimilation and the future performance prediction of BHP of Inj-1 as a function of ensemble size are shown in Figures 4.23 and 4.24 for the cases without localization and with covariance localization, respectively. As we can see from these figures, the history matching of BHP is very good for all cases with and without localization. With ensemble sizes of 50 and 100, for the prediction phase, we obtained better performances when we use covariance localization than when no localization is used. The BHP history matches and future forecasts for other (producer) wells (not shown) are similar to the BHP of Inj-1. The corresponding results obtained by rerunning from time zero with the final model are shown in Figures 4.25 and 4.26. For the case without localization, the inconsistency does not appear to significantly affect the predictions of the BHP of Inj-1; compare Figures 4.23 and 4.25. However, for the case with covariance localization, there is inconsistency between results, see Figures 4.24 and 4.26.

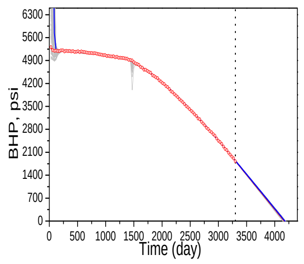


(a) Ne=50. (b) Ne=100. (c) Ne=150.

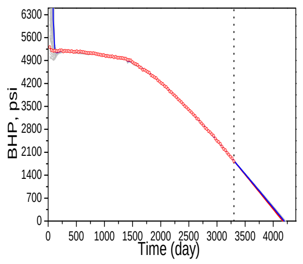
Figure 4.23: Matches and predictions of BHP of Inj-1 as a function of ensemble size during data assimilation with EnKF without localization, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 1.

4.5.3 Incorporation of EnKF results with the SVD-GF-EnRML algorithm, Example 1

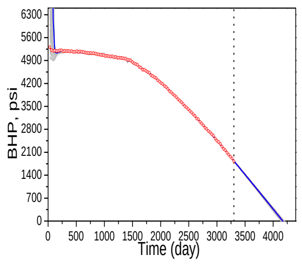
The results of the synthetic example presented earlier showed that we greatly improve the computational efficiency of the RML method by implementation of the SVD algorithms. However, we still require on the order of three thousand (equiv-



(a) Ne=50.

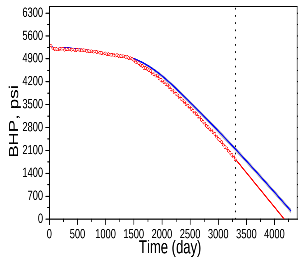


(b) Ne=100.

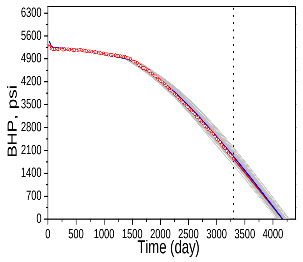


(c) Ne=150.

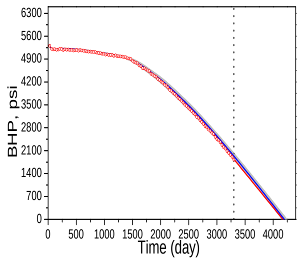
Figure 4.24: Matches and predictions of BHP of Inj-1 as a function of ensemble size during data assimilation with EnKF with anisotropic localization, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 1.



(a) Ne=50.

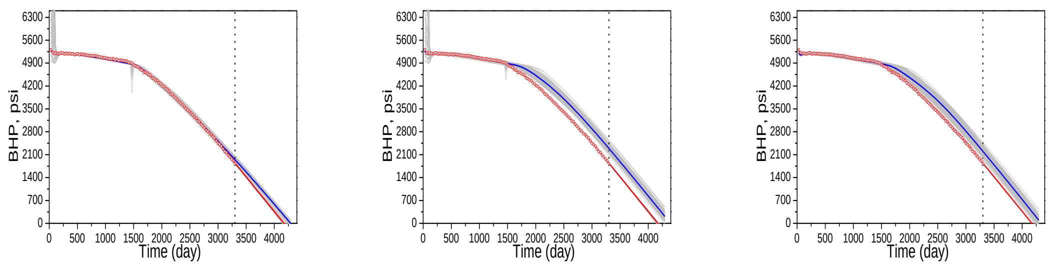


(b) Ne=100.



(c) Ne=150.

Figure 4.25: Matches and predictions of BHP of Inj-1 as a function of ensemble size for rerun from time zero with EnKF without localization, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 1.



(a) Ne=50. (b) Ne=100. (c) Ne=150.

Figure 4.26: Matches and predictions of BHP of Inj-1 as a function of ensemble size for rerun from time zero with EnKF with anisotropic localization, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 1.

alent) reservoir simulation runs to obtain a set of 15 or so plausible realizations of the reservoir model parameters. Here we demonstrate that if we use the final updated EnKF realizations of the permeability field as initial guesses in the SVD-GF-EnRML, computational efficiency is significantly improved. Similar gains in computational efficiency to those shown here were obtained by using the EnKF results as initial guesses for the SVD-EnRML algorithm. As we only directly estimate reservoir model parameters (permeability field) with the SVD algorithms, the statistical inconsistency sometimes experienced with EnKF is not an issue when using SVD-GF-EnRML or SVD-EnRML. Here, we refer to the final realization corresponding to the m_{prior} and d_{obs} obtained with EnKF as a MAP realization and use the previous described algorithm for new implementation of the SVD-GF-EnRML algorithm. We use the same strategy of changing of sv-cut during iterations as was used in the previous implementation of the SVD-GF-EnRML algorithm, i.e., we start with sv-cut $= 0.1$ at the first iteration and decrease sv-cut at each (outer) iteration by dividing by two until we reach the final value of sv-cut $= 0.0004$. Then from this point on, we iterate with fixed value sv-cut $= 0.0004$.

The notation $m_{\text{initial}} = m^{\text{a}}_{\text{EnkF, Loc}}$ in figure captions and elsewhere indicates that in applying SVD-GF-EnRML, the set of initial guesses (realizations of the permeability field) were selected as the first 15 realizations obtained from EnKF with anisotropic localization with ensemble size of $N_e = 150$. Here, we refer to the combined algorithm as SVD-GF-EnRML-EnKF. The behavior of the normalized objective function during iteration with this algorithm is shown in Figure 4.27. Note that the initial values of the normalized objective function (value at iteration zero with $m^{\mathrm{a}}_{\text{EnkF, Loc}}$) are very large indicating a poor data match which is expected from the results of Figures 4.18(c), 4.22(c) and 4.26(c). During iteration of SVD-GF-EnRML-EnKF, the value of $O_{N,\max}(m)$ decreases from 9991.39 to 2.36 and the number of retained singular triplets is 36 at convergence.

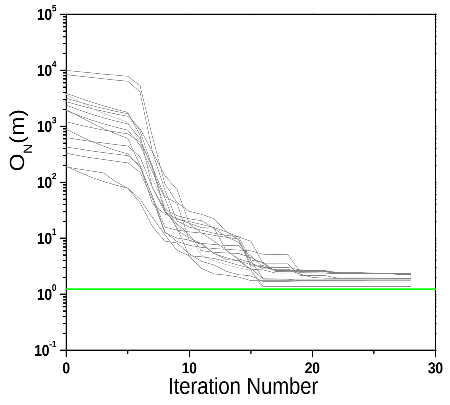
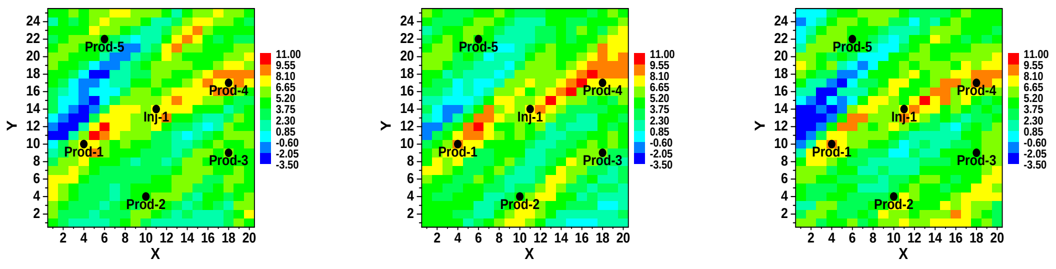


Figure 4.27: Behavior of the normalized objective function for each realization during iteration of SVD-GF-EnRML with $m_{\text{initial}} = m^{\mathrm{a}}_{\text{EnkF, Loc}}$, Example 1.

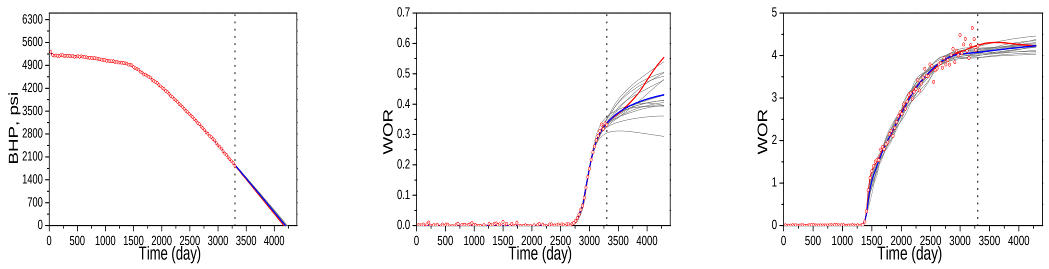
Figure 4.28 shows the final conditional log-permeability fields obtained with the SVD-GF-EnRML algorithm for the case of $m_{\text{initial}} = m^{\mathrm{a}}_{\text{EnkF, Loc}}$. The corresponding unconditional realizations are shown in Figure 4.12. As can be seen from this figure, the essential features of the channel and the barrier that exist in the true model have been captured in each individual conditional realization; however, each realization is distinctly different.

The history matching results and future forecasts of the BHP of Inj-1 and the



(a) Realization 5. (b) Realization 10. (c) Realization 13.

Figure 4.28: Three conditional realizations of log-permeability fields from SVD-GF-EnRML algorithm using $m_{\text{initial}} = m^{\text{a}}_{\text{EnkF, Loc}}$, Example 1.



(a) BHP Inj-1. (b) WOR Prod-1. (c) WOR Prod-4.

Figure 4.29: Matches and predictions of BHP and WOR from SVD-GF-EnRML using $m_{\text{initial}} = m^{\text{a}}_{\text{EnkF, Loc}}$, open red circles are observed data, red is from true model, blue is the prediction mean, gray curves represent predictions from realizations, Example 1.

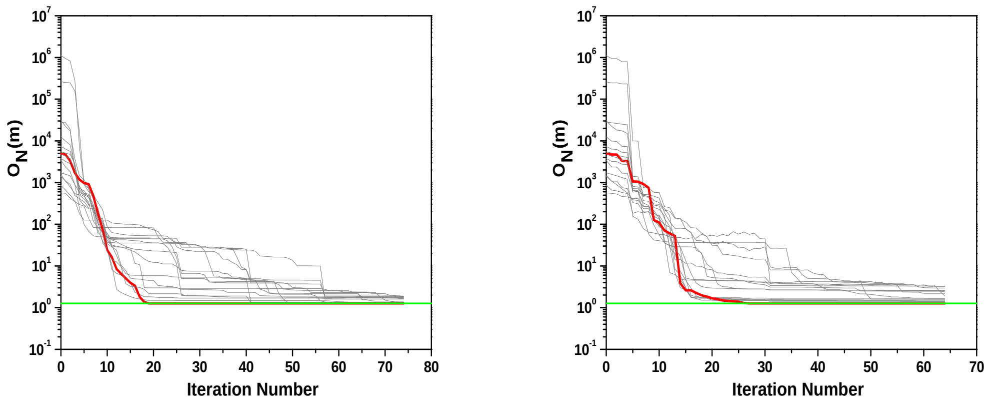
WOR of Prod-1 and Prod-4 obtained with the SVD-GF-EnRML-EnKF algorithm ($m_{\text{initial}} = m^{\text{a}}_{\text{EnkF, Loc}}$) are shown in Figure 4.29. Good matches of the BHP and WOR are obtained. The future performance predictions ($t > 3300$ days) of BHP and WOR are also reasonably good although after 3600 days the predicted WOR at producer 1 deviates from the WOR calculated from the true permeability field. The data matches and future predicted obtained with SVD-GF-EnRML-EnKF algorithm are far better than the corresponding results obtained with EnKF (Figures 4.18(c), 4.22(c) and 4.26(c)).

The summary of the computational cost of the SVD-GF-EnRML algorithm with $m_{\text{initial}} = m^{\text{a}}_{\text{EnkF, Loc}}$ is given in the last row of Table 4.1. Note that we have significantly improved the computational efficiency of the SVD-GF-EnRML algorithm by generating a set of initial guesses of the log-permeability field from EnKF.

4.5.4 SVD-GF-EnRML and SVD-EnRML Simulation Results of Example 2

Here, we present results obtained from truncated-SVD parameterization algorithms and EnKF method for a second example, Example 2. We have implemented the SVD-GF-EnRML and SVD-EnRML algorithms for Example 2 to generate 15 realizations ($N_e = 15$). We use the same procedure to generate unconditional realizations as was used for Example 1. The same procedure is used here for computation of the singular triplets as was used in the previous example. Figures 4.30(a) and 4.30(b), respectively, show the behavior of the normalized objective function during iteration for all realizations that were obtained using the SVD-GF-EnRML and SVD-EnRML algorithms. At convergence, the maximum value of the normalized objective function among all realizations is $O_{N,\max} = 1.34$ for the SVD-GF-EnRML algorithm compared to $O_{N,\max} = 3.26$ for the SVD-EnRML algorithm.

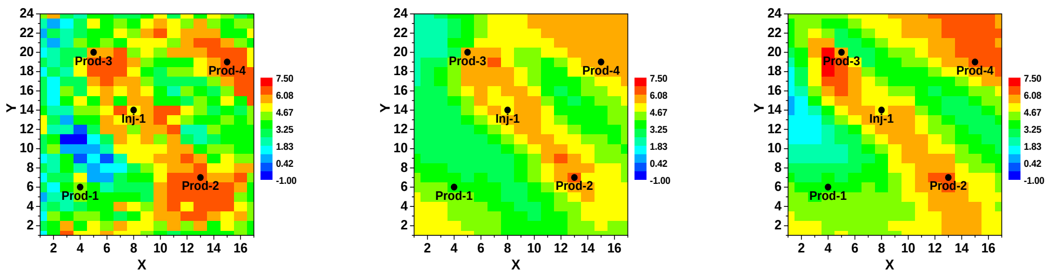
Figures 4.31(b) and 4.31(c) show the final average of the realizations of log-permeability obtained by the SVD-GF-EnRML and SVD-EnRML algorithms, respectively. As can be seen from these figures, both fields capture some basic features



(a) SVD-GF-EnRML. (b) SVD-EnRML.

Figure 4.30: Behavior of the normalized objective function during iteration of SVD-GF-EnRML and SVD-EnRML algorithms, red curve represents MAP estimate, gray curves represent realizations, Example 2.

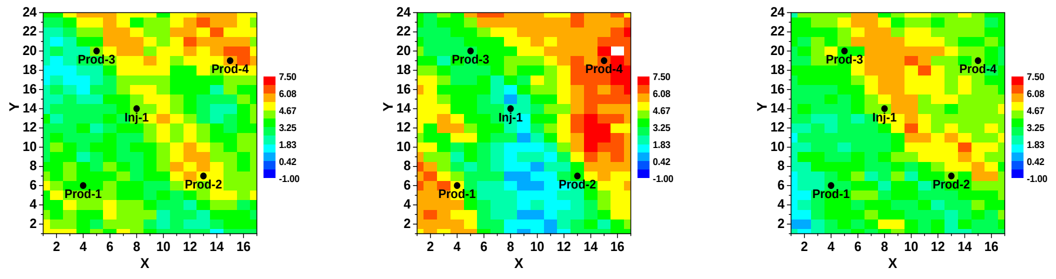
of the true log permeability field (Figure 4.31(a)). Specially, some characteristics of the true channel have been captured in the final average field of the realizations with both algorithms.



(a) True Ln(k). (b) SVD-GF-EnRML. (c) SVD-EnRML.

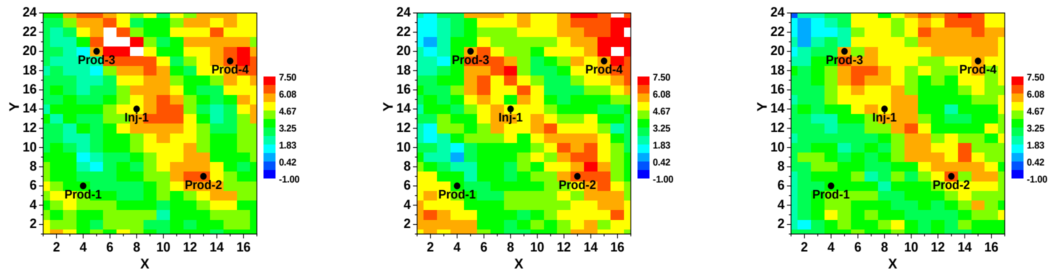
Figure 4.31: True $\ln(k)$ field and final average of realizations of log-permeability from SVD-GF-EnRML and SVD-EnRML algorithms, Example 2.

Figure 4.32 shows three unconditional realizations of log-permeability fields obtained from the Cholesky decomposition of the prior covariance matrix. The corresponding final conditional realizations of the log-permeability fields that were obtained by implementation of the SVD-GF-EnRML and the SVD-EnRML algorithms are shown, respectively, in Figures 4.33 and 4.34. Note that the main features of the truth have been captured in all realizations but there are distinct differences between realizations.



(a) Realization 6. (b) Realization 10. (c) Realization 13.

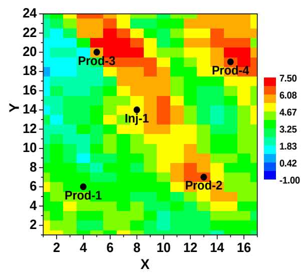
Figure 4.32: Unconditional realizations of log-permeability fields, Example 2.



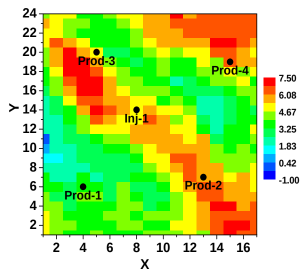
(a) Realization 6. (b) Realization 10. (c) Realization 13.

Figure 4.33: Conditional realizations of log-permeability fields from SVD-GF-EnRML algorithm, Example 2.

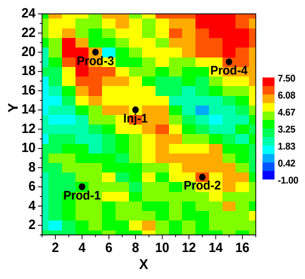
The history matching results and future prediction performances of BHP of Prod-3 and WOR of Prod-2 and Prod-3 are shown in Figure 4.35 for all realizations. Note that very good history matching and prediction results for BHP of Prod-3 were obtained. The results of BHP for other wells (not shown) are very similar to those shown for Prod-3. As can be seen in Figures 4.35(b) and 4.35(c), we obtained



(a) Realization 6.



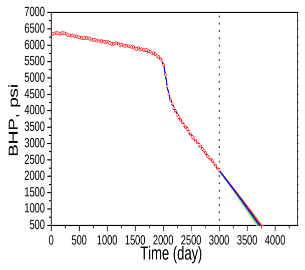
(b) Realization 10.



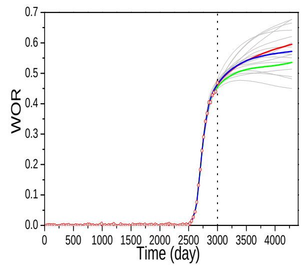
(c) Realization 13.

Figure 4.34: Conditional realizations of log-permeability field from SVD-EnRML algorithm, Example 2.

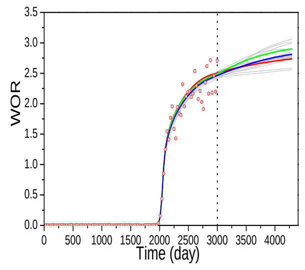
very good results for WOR for both producers during the history matching period and the predicted data from truth (red curves) lies near the center of the band of predictions generated from the set of realizations, i.e., the SVD-GF-EnRML method gives unbiased history matching and prediction results. The corresponding match and prediction results obtained with SVD-EnRML shown in Figure 4.36 are fairly similar.



(a) Prod-3 BHP.



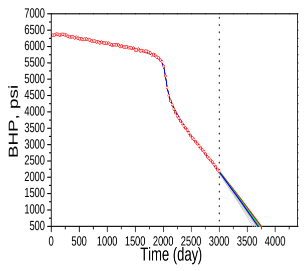
(b) Prod-2 WOR.



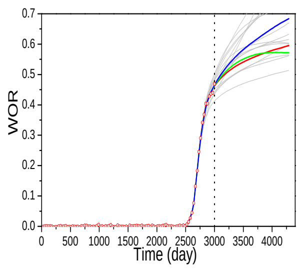
(c) Prod-3 WOR.

Figure 4.35: Data matches and future prediction from SVD-GF-EnRML, open red circles are observed data, red is from true model, blue is the prediction mean, green is from the MAP estimate, gray curves represent predictions from realizations, Example 2.

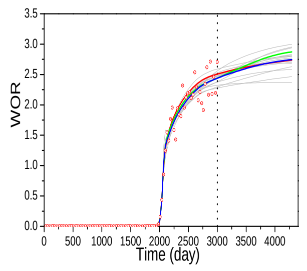
The summary of the computational costs for the SVD-GF-EnRML and SVD-EnRML sampling algorithms is given in Table 4.2. As shown in the last column



(a) Prod-3 BHP.



(b) Prod-2 WOR.



(c) Prod-3 WOR.

Figure 4.36: Data matches and future prediction from SVD-EnRML, open red circles are observed data, red is from true model, blue is the prediction mean, green is from the MAP estimate, gray curves represent predictions from realizations, Example 2.

of the Table 4.2, the overall computational costs of the SVD-EnRML algorithm is less than that of SVD-GF-EnRML algorithm. This is comparable to the result obtained in Example 1. Although we need a large number of simulation runs to obtain 15 conditional realizations, the SVD-GF-EnRML and SVD-EnRML algorithms are significantly more efficient than the standard implementation of RML using a quasi-Newton method. The results of the third row of Table 4.2 are based on incorporation of the EnKF results with the SVD-GF-EnRML algorithm which will be discussed later.

Table 4.2: Summary of computational costs of SVD-GF-EnRML and SVD-EnRML algorithms, Example 2.

Algorithm	Simulations	Forward Runs	Adjoint	Equ. Sim. Runs
SVD-GF-EnRML	3420	3177	3105	4990
SVD-EnRML	3092	1239	1947	3885
SVD-GF-RML-EnKF	818	1274	1244	1447

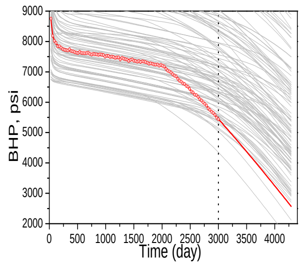
4.5.5 EnKF Results of Example 2

For the EnKF method, 150 unconditional ensembles were generated from the

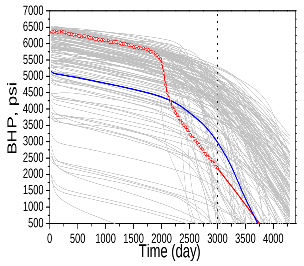
Cholesky decomposition of the prior covariance matrix. We use this set of 150 initial ensembles to sample different ensemble sizes of 50, 100 and 150. Note that like previous example, we add one additional realization corresponding to the m_{prior} to the initial ensemble members. The observed data to be assimilated corresponding to m_{prior} is d_{obs}. The main objective here is to investigate the effects of ensemble size and covariance localization on the final estimate of the model parameters, the history matching and prediction results of EnKF algorithm. We consider both the standard implementation of EnKF as well as EnKF with anisotropic covariance localization using the Gaspari and Cohn [21] correlation function. We implement the covariance localization in two different forms for this example. The first form is the standard covariance localization given by Eq. 4.28 and the second form is the Kalman gain localization given by Eq. 4.34. The correlation length in the principle direction (u-axis) and orthogonal direction (v-axis) are respectively assumed to be $a_u = 25 \times \triangle x = 10,000$ feet and $a_v = 10 \times \triangle y = 4,000$ feet. The principal direction is aligned with the principle direction of the prior covariance matrix of the log-permeability field ($\alpha = 120°$).

The initial estimate of the uncertainty in the prediction performance is obtained by running the reservoir simulation forward with all the unconditional ensembles. The variability of BHP and WOR production data from the initial models compared to the true prediction are shown in Figures 4.37 and 4.38 for an ensemble size of 150. As can be seen from these figures, the initial spread (uncertainty) of the ensemble predictions is very large. The initial WOR prediction data for Prod-1, Prod-2, Prod-3, and Prod-4 are respectively shown in Figures 4.37(c)- 4.38. As shown, there is no water breakthrough for Prod-1 and Prod-4 in the truth case but the initial ensemble mean shows nonzero WOR data.

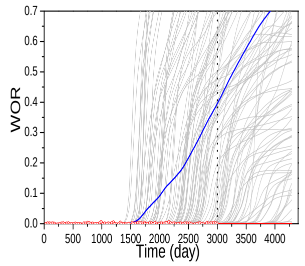
The final estimates of the ensemble mean of log-permeability field as functions of the ensemble size are presented in Figures 4.39, 4.40 and 4.41, respectively, for standard EnKF without localization, with anisotropic covariance localization and



(a) Inj-1 BHP.

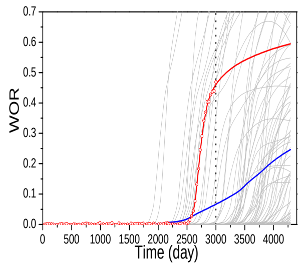


(b) Prod-3 BHP.

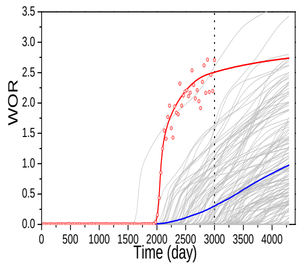


(c) Prod-1 WOR.

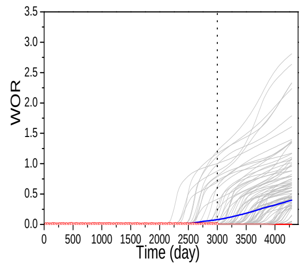
Figure 4.37: Matches and predictions of the BHP of Inj-1 and Prod-3 and the WOR of Prod-1 with initial ensemble, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 2.



(a) Prod-2 WOR.



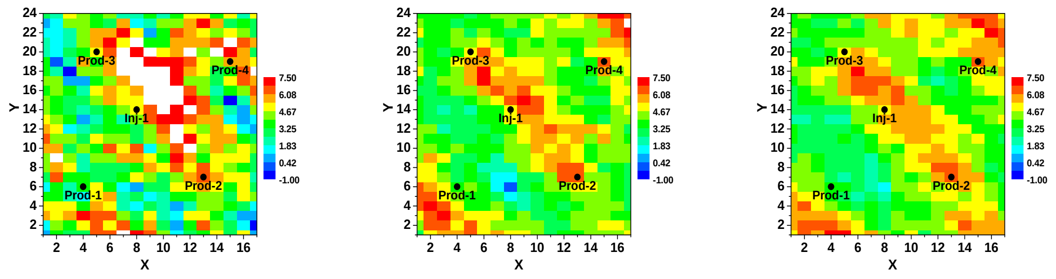
(b) Prod-3 WOR.



(c) Prod-4 WOR.

Figure 4.38: Matches and predictions of the WOR of Prod-2, Prod-3 and Prod-4 with initial ensemble, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 2.

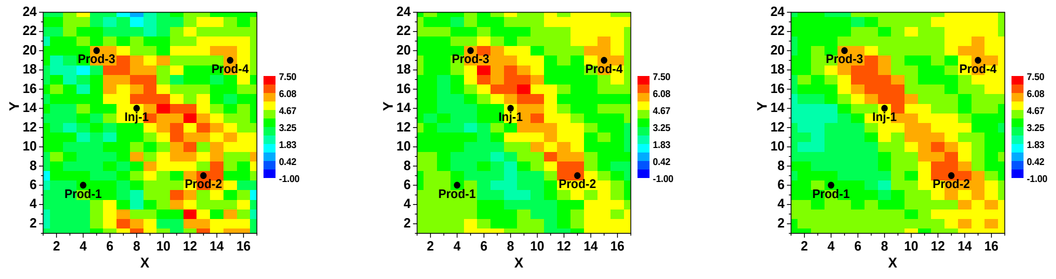
with Kalman gain localization. As shown in Figure 4.39(a), we have obtained very poor log-permeability field with the ensemble size of 50 with standard EnKF method without covariance localization. The final log-permeability filed exhibits overshooting (higher values of $\ln(k)$ than are reasonable). Values of lnk greater than 7.5 were obtained which are inconsistent with the truth. Increasing the ensemble size to 100 and 150 members improves the results and the final estimate of mean of the log-permeability fields capture some features of the truth as shown in Figures 4.39(b) and 4.39(c). However, these fields show a high permeability region in the lower left corner which do not appears in the truth (Figure 4.31(a)). As Figures 4.40 and 4.41 show, this problem is removed and the final ensemble mean fields provide a good estimate of the true log-permeability field for all three ensemble sizes. The channel features have been captured and there is no overshooting and undershooting values in log-permeability. Also, note that there is not much differences between the results of the anisotropic covariance localization and the Kalman gain localization.



(a) Ne=50. (b) Ne=100. (c) Ne=150.

Figure 4.39: Ensemble mean log-permeability fields from EnKF as a function of ensemble size without covariance localization, Example 2.

Figure 4.42 shows three of the initial (unconditional) ensemble members generated using the Cholesky decomposition of the prior covariance matrix. The final conditional ensembles of model parameters corresponding to these unconditional realizations (Figure 4.42) are shown, respectively, in Figures 4.43, 4.44 and 4.45 for

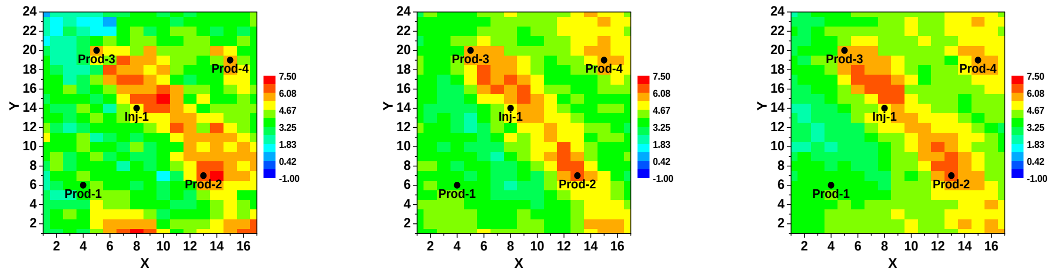


(a) Ne=50.

(b) Ne=100.

(c) Ne=150.

Figure 4.40: Ensemble mean log-permeability fields from EnKF as a function of ensemble size with anisotropic covariance localization, Example 2.



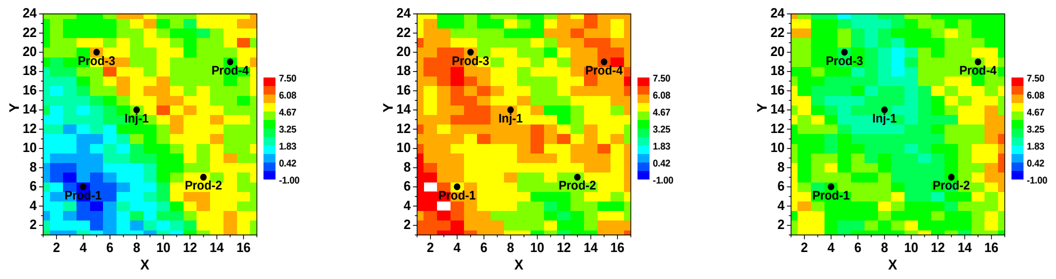
(a) Ne=50.

(b) Ne=100.

(c) Ne=150.

Figure 4.41: Ensemble mean log-permeability fields from EnKF as a function of ensemble size with Kalman gain localization, Example 2.

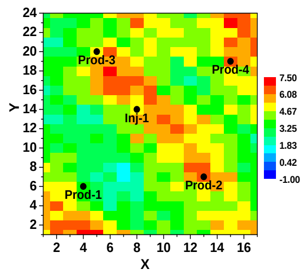
the case without covariance localization, with anisotropic covariance localization and with Kalman gain localization for the ensemble size of $N_e = 150$. As can be seen in Figure 4.43, some features of the truth (Figure 4.31(a)) have been captured in the final ensemble members with no localization. However, there is a small variability between ensemble members. Moreover, again there is a region of high permeability values in the lower left corner of each of the realization which does not appear in the truth (Figure 4.31(a)). As Figures 4.44 and 4.45 show, with implementation of covariance localization, the variability between the ensembles increases, the region of high permeability values at the lower right corner of the reservoir disappears and the essential features of the truth are still captured correctly.



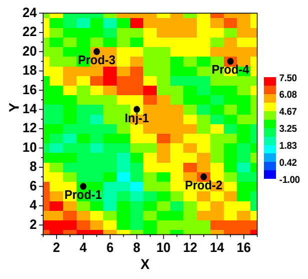
(a) Ensemble 4. (b) Ensemble 8. (c) Ensemble 12.

Figure 4.42: Three unconditional log-permeability fields, Example 2.

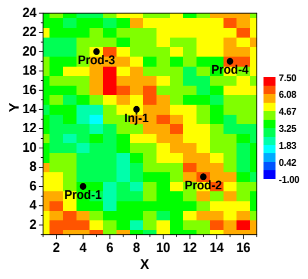
Figure 4.46 shows the history matching results during data assimilation and prediction performances of the WOR of Prod-2 for the 50, 100, and 150 ensemble sizes obtained for the case without covariance localization. It is clear from Figure 4.46(a) that with an ensemble size of 50 the history matching results are reasonable with small variability in the ensemble. However, the WOR prediction from all ensemble members deviates from the truth after 3300 days. Note that the sharp reduction of WOR at later time of the prediction phase is not physically correct for this particular example. We have obtained the same behavior of the WOR for a particular realization with a commercial simulator (ECLIPSE). As shown in Figure 4.46(b),



(a) Ensemble 4.

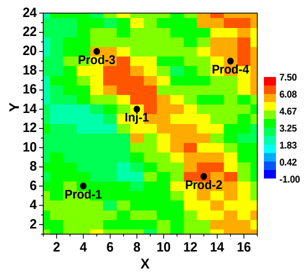


(b) Ensemble 8.

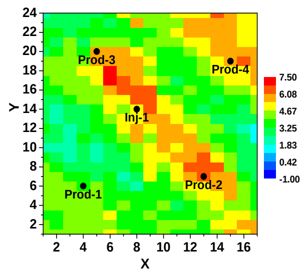


(c) Ensemble 12.

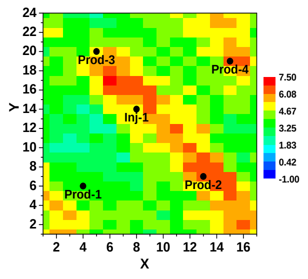
Figure 4.43: Three ensemble members of the final log-permeability fields obtained from EnKF without localization, Ne=150, Example 2.



(a) Ensemble 4.

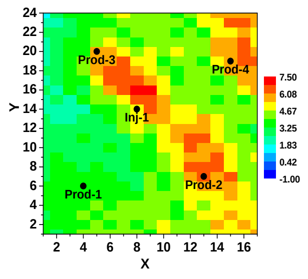


(b) Ensemble 8.

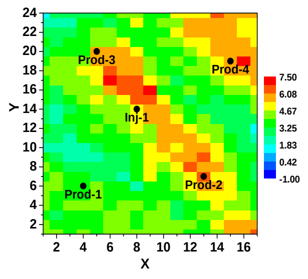


(c) Ensemble 12.

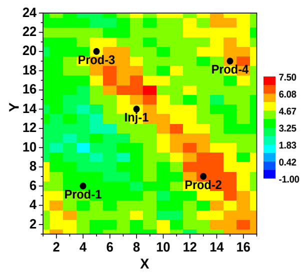
Figure 4.44: Three ensemble members of the final log-permeability fields obtained from EnKF with anisotropic covariance localization, Ne=150, Example 2.



(a) Ensemble 4.



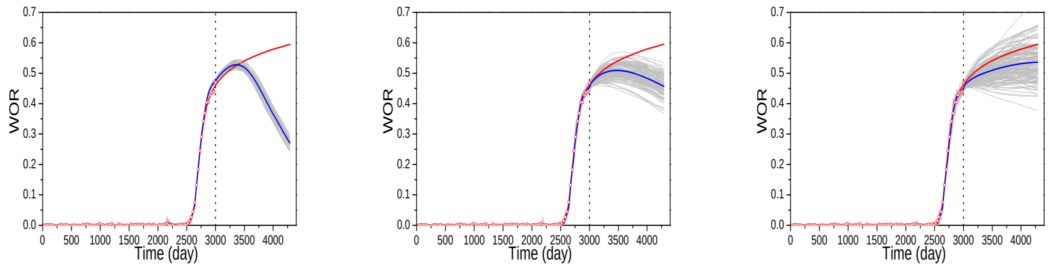
(b) Ensemble 8.



(c) Ensemble 12.

Figure 4.45: Three ensemble members of the final log-permeability fields obtained from EnKF with Kalman gain localization, Ne=150, Example 2.

increasing the ensemble size to 100 members improved the history matching results and slightly improved the prediction performances but at latter times the ensemble mean prediction deviates from the truth. The history matching results are good with ensemble size of 150 without using localization and the ensemble mean prediction is in good agreement with the truth (Figure 4.46(c)).

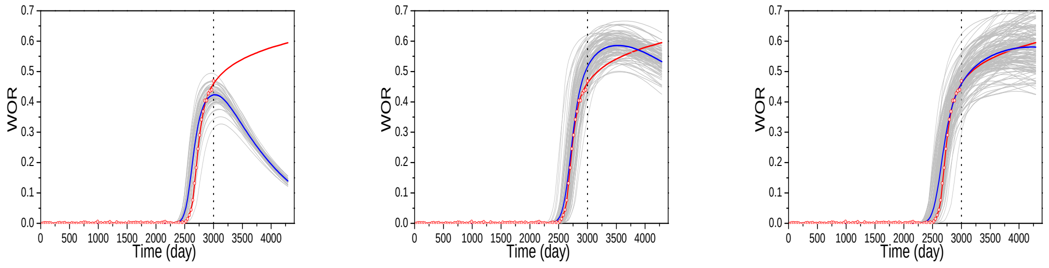


(a) Ne=50. (b) Ne=100. (c) Ne=150.

Figure 4.46: Matches and predictions of WOR of Prod-2 as a function of ensemble size during data assimilation with EnKF without localization, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 2.

The consistency of the results have been checked by rerunning the simulation from time zero with the ensemble of permeability fields obtained by assimilating all data and comparing the resulting data matches and future predictions with the results obtained during the data assimilation. Figure 4.47 shows the predicted performances of WOR of Prod-2 from the final model obtained by running from time zero as a function of ensemble size for the case without localization. From Figure 4.47, we observe that the ensemble mean prediction is in good agreement with the truth for times less than 3000 days for ensemble sizes of 50 and 100 and all the results for three ensemble sizes are comparable with the results of Figure 4.46. However, the spread of the ensembles increases when we rerun from time zero.

The history matching results during data assimilation period and future forecasts of WOR of Prod-2 as a function of the ensemble size are shown in Figures 4.48



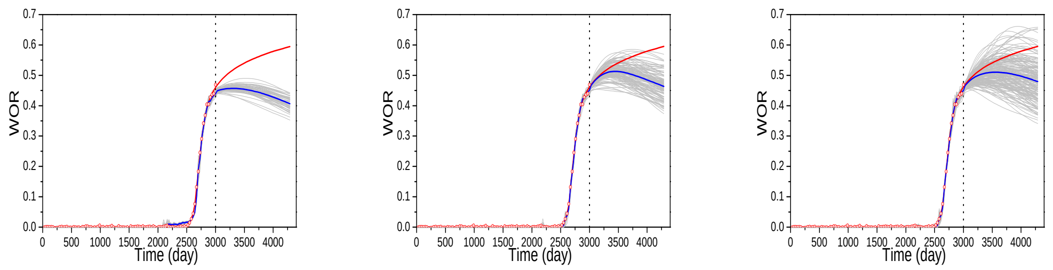
(a) Ne=50. (b) Ne=100. (c) Ne=150.

Figure 4.47: Matches and predictions of WOR of Prod-2 as a function of ensemble size for rerun from time zero with EnKF without localization, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 2.

and 4.49, respectively, for the case with anisotropic covariance localization and the Kalman gain localization, respectively. As shown in these figures, we have obtained very good history matches of the WOR with all three ensemble sizes and as we increase the ensemble size, the prediction performance improves and the mean ensemble prediction moves toward the true prediction. The comparison of these results with the results of Figure 4.46 shows that the covariance localization slightly improves the future forecast results for the case with 50 and 100 ensemble members. Note that the results are almost similar for both implementations of covariance localization.

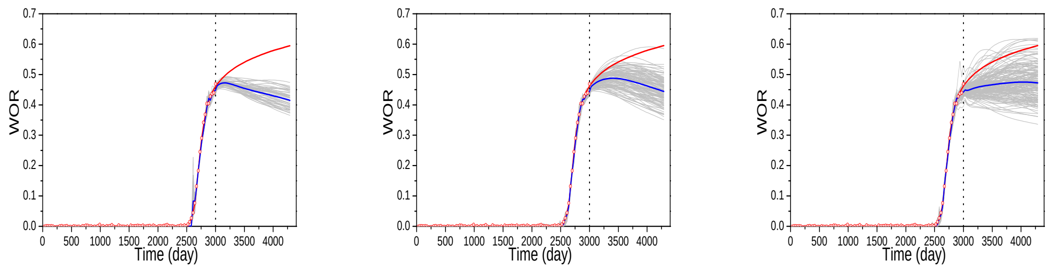
Figures 4.50 and 4.51 show the predicted performances of WOR of Prod-2 from the final model obtained by running from time zero as a function of ensemble size for the case with covariance localization and Kalman gain localization. As we can see from these figures, the rerun results are inconsistent with those obtained during the data assimilation period especially for ensemble sizes of 50 and 100 and the the uncertainty of the forecasts increased when we rerun from time zero. However, the variances of prediction are lower than those obtained from initial models (Figure 4.38(a)).

In general, the behavior of the WOR and BHP history matches and prediction



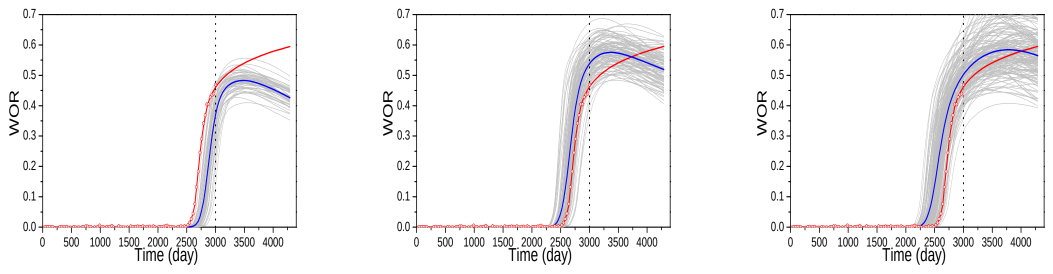
(a) Ne=50. (b) Ne=100. (c) Ne=150.

Figure 4.48: Matches and predictions of WOR of Prod-2 as a function of ensemble size during data assimilation with EnKF with anisotropic covariance localization, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 2.



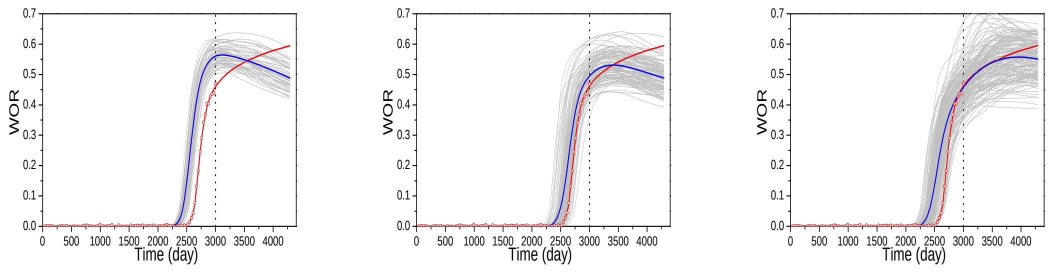
(a) Ne=50. (b) Ne=100. (c) Ne=150.

Figure 4.49: Matches and predictions of WOR of Prod-2 as a function of ensemble size during data assimilation with EnKF with Kalman gain localization, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 2.



(a) Ne=50. (b) Ne=100. (c) Ne=150.

Figure 4.50: Matches and predictions of WOR of Prod-2 as a function of ensemble size for rerun from time zero with EnKF with anisotropic covariance localization, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 2.



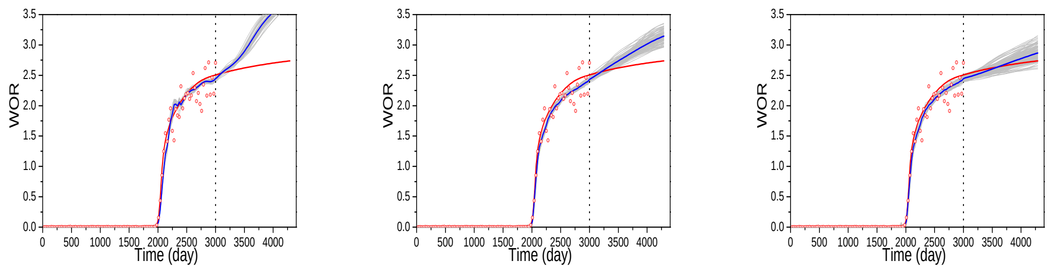
(a) Ne=50. (b) Ne=100. (c) Ne=150.

Figure 4.51: Matches and predictions of WOR of Prod-2 as a function of ensemble size for rerun from time zero with EnKF with Kalman gain localization, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 2.

results from EnKF for other wells is similar to those presented for the WOR of Prod-2. As we increase the ensemble size, the history matching results improve during the data assimilation, future performance predictions improve and the mean of the prediction moves toward the true prediction and the final log-permeability fields capture the most essential features of the truth. Also, the implementation of the covariance localization mitigates the spurious correlations issue and improves the history matches and prediction results especially for the case where the ensemble size is small. Our results show that the anisotropic covariance localization and the Kalman gain localization gave almost similar results with all ensemble sizes. We check the consistency of the results by rerunning the simulation from time zero with the ensemble of permeability fields obtained by assimilating all data and comparing the resulting data matches and future predictions with the results obtained during the data assimilation. Because we update the ensemble of primary reservoir simulation variables along with the model parameters sequentially with EnKF and the problem is highly non-linear, the inconsistency usually occurs when we rerun the simulation with the updated models.

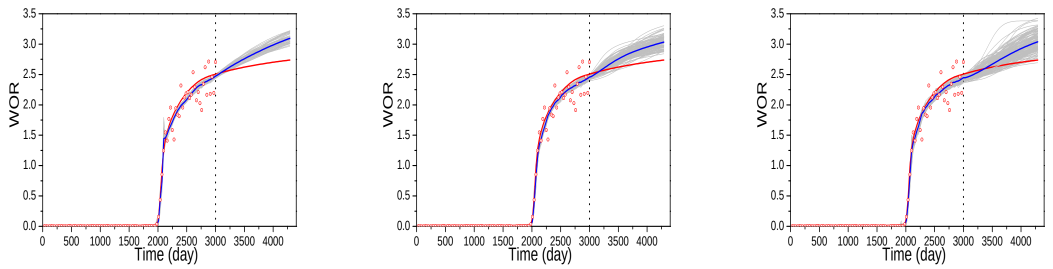
The results of history matching during data assimilation and the future performance prediction of the WOR of Prod-3 as a function of ensemble size are shown in Figures 4.52, 4.53 and 4.54, respectively, for the case without using covariance localization, with anisotropic covariance localization and the Kalman gain localization. The corresponding history matches and prediction results obtained by rerunning the simulation from time zero with the ensemble of permeability fields obtained by assimilating all data are shown, respectively, in Figures 4.55, 4.56 and 4.57.

The history matches during data assimilation period and prediction performance of the BHP of Prod-3 as a function of the ensemble size are shown in Figures 4.58, 4.59 and 4.60, respectively, for the case without covariance localization, with anisotropic covariance localization and the Kalman gain localization, respectively.



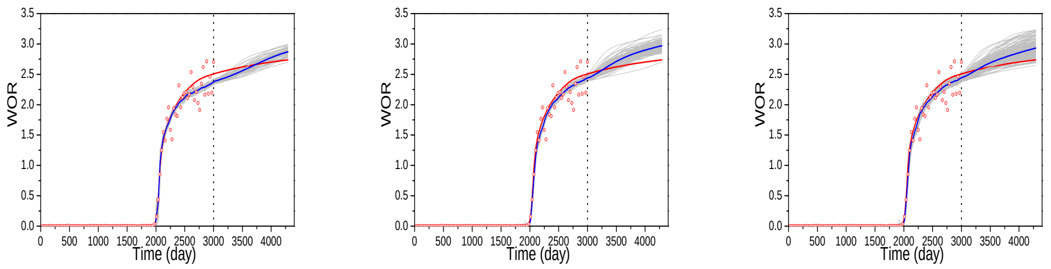
(a) Ne=50. (b) Ne=100. (c) Ne=150.

Figure 4.52: Matches and predictions of WOR of Prod-3 as a function of ensemble size during data assimilation with EnKF without localization, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 2.



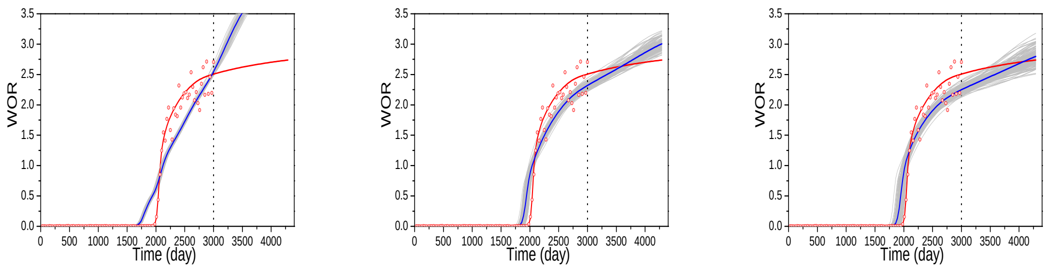
(a) Ne=50. (b) Ne=100. (c) Ne=150.

Figure 4.53: Matches and predictions of WOR of Prod-3 as a function of ensemble size during data assimilation with EnKF with anisotropic covariance localization, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 2.



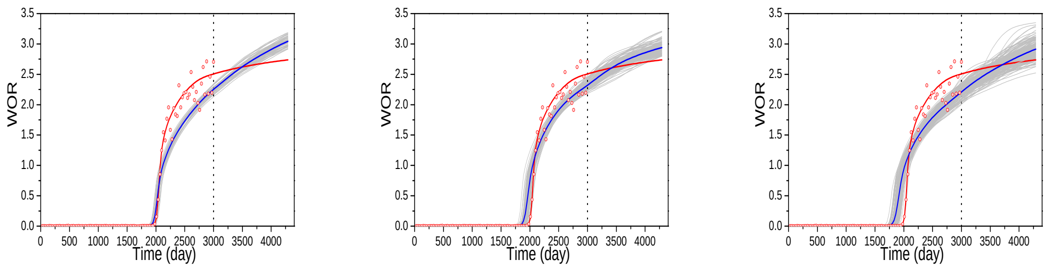
(a) Ne=50. (b) Ne=100. (c) Ne=150.

Figure 4.54: Matches and predictions of WOR of Prod-3 as a function of ensemble size during data assimilation with EnKF with Kalman gain localization, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 2.



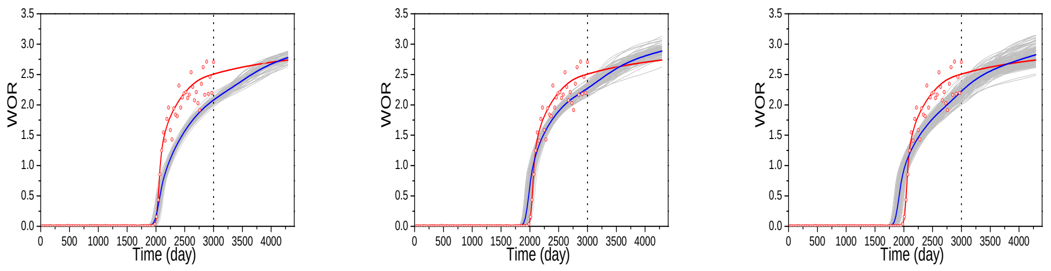
(a) Ne=50. (b) Ne=100. (c) Ne=150.

Figure 4.55: Matches and predictions of WOR of Prod-3 as a function of ensemble size for rerun from time zero with EnKF without localization, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 2.



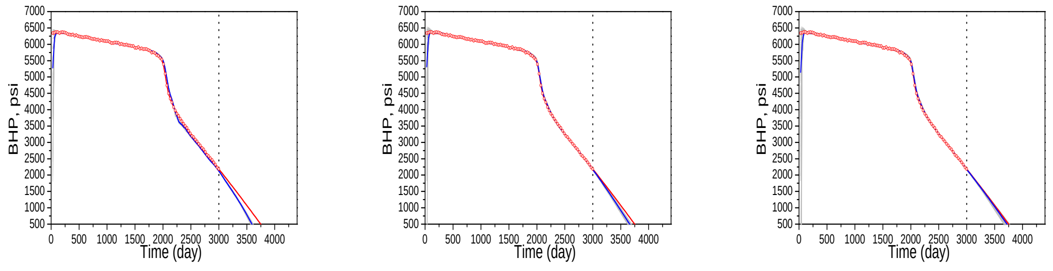
(a) Ne=50. (b) Ne=100. (c) Ne=150.

Figure 4.56: Matches and predictions of WOR of Prod-3 as a function of ensemble size for rerun from time zero with EnKF with anisotropic covariance localization, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 2.



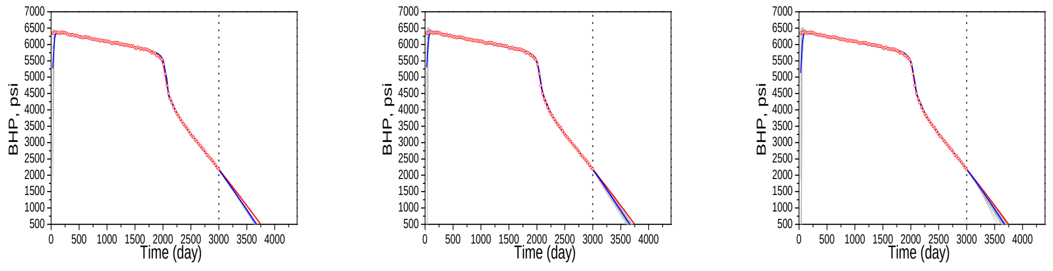
(a) Ne=50. (b) Ne=100. (c) Ne=150.

Figure 4.57: Matches and predictions of WOR of Prod-3 as a function of ensemble size for rerun from time zero with EnKF with Kalman gain localization, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 2.



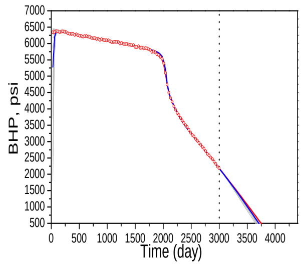
(a) Ne=50. (b) Ne=100. (c) Ne=150.

Figure 4.58: Matches and predictions of BHP of Prod-3 as a function of ensemble size during data assimilation with EnKF without localization, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 2.

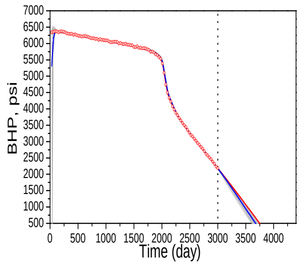


(a) Ne=50. (b) Ne=100. (c) Ne=150.

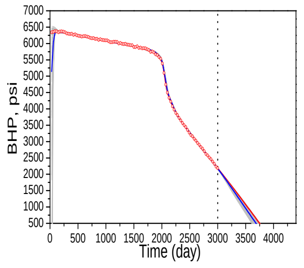
Figure 4.59: Matches and predictions of BHP of Prod-3 as a function of ensemble size during data assimilation with EnKF with anisotropic covariance localization, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 2.



(a) Ne=50.



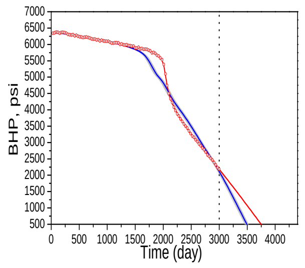
(b) Ne=100.



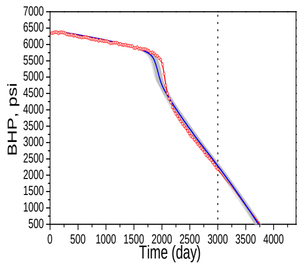
(c) Ne=150.

Figure 4.60: Matches and predictions of BHP of Prod-3 as a function of ensemble size during data assimilation with EnKF with Kalman gain localization, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 2.

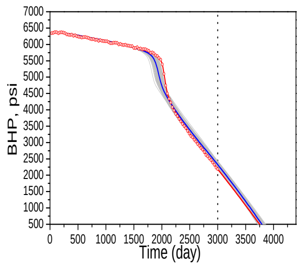
Figures 4.61, 4.62 and 4.63 respectively, show the results of predicting the BHP of Prod-3 obtained from rerunning the simulation from time zero with the ensemble of permeability field obtained by assimilating all data for the cases without localization, with anisotropic localization and the Kalman gain localization. As can be seen from this figure, we have obtained consistent results in all cases with ensemble sizes of 100 and 150. The BHP performances of other wells (not shown) are similar to the performance of the BHP of Prod-3.



(a) Ne=50.

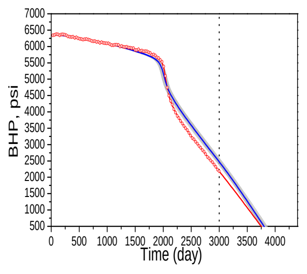


(b) Ne=100.

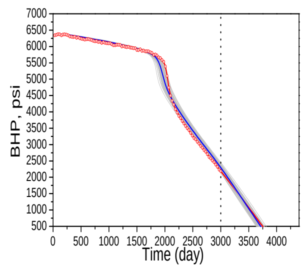


(c) Ne=150.

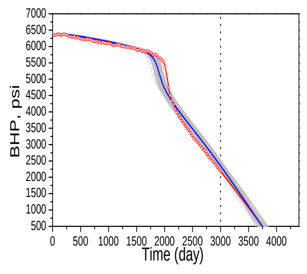
Figure 4.61: Matches and predictions of BHP of Prod-3 as a function of ensemble size for rerun from time zero with EnKF without localization, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 2.



(a) Ne=50.

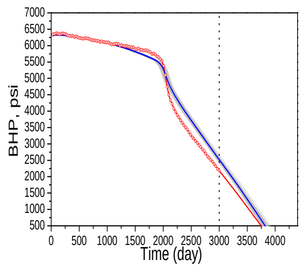


(b) Ne=100.

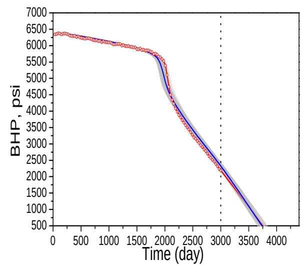


(c) Ne=150.

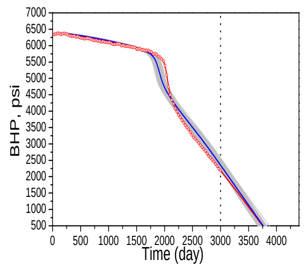
Figure 4.62: Matches and predictions of BHP of Prod-3 as a function of ensemble size for rerun from time zero with EnKF with anisotropic covariance localization, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 2.



(a) Ne=50.



(b) Ne=100.



(c) Ne=150.

Figure 4.63: Matches and predictions of BHP of Prod-3 as a function of ensemble size for rerun from time zero with EnKF with Kalman gain localization, open red circles are observed data, red is from true model, blue is ensemble mean, gray curves are ensemble predictions, Example 2.

4.5.6 Incorporation of EnKF results with SVD-GF-EnRML algorithm, Example 2

Next, we present results obtained by using final updated EnKF realizations of the permeability field as initial guesses in the SVD-GF-EnRML algorithm to generate 15 conditional realizations. We use the same procedure as described previously for this new implementation of the SVD-GF-EnRML algorithm. We initialize the SVD-GF-EnRML algorithm by using the final updated model corresponding to the $m_{\mathrm{uc}} =$

m_{prior} obtained from the EnKF as a MAP realization to compute the truncated-SVD. The strategy of changing the sv-cut during iteration is similar to the previous implementation of the SVD-GF-EnRML algorithm. As in the previous example, this implementation significantly improves the computational efficiency of the SVD-GF-EnRML algorithm and the final history matches and prediction results are very good compared to those obtained with the EnKF method.

For the results presented in this section, the set of initial guesses (realizations) of the permeability field were selected as the first 15 realizations obtained from EnKF with anisotropic covariance localization with ensemble size of $N_e = 150$ ($m_{\text{initial}} = m^{\text{a}}_{\text{EnkF, Loc}}$). Figure 4.64 shows the behavior of the normalized objective function during iteration obtained with SVD-GF-EnRML-EnKF algorithm. The initial large values of the normalized objective function at iteration zero are from final log-permeability realizations obtained with the EnKF with anisotropic covariance localization and indicate poor data matches which is expected from the results of Figures 4.50(c), 4.56(c) and 4.62(c). Note that the value of $O_{N,\max}(m)$ decreases from 822.37 to 1.38 in 30 iterations with the SVD-GF-EnRML-EnKF algorithm and the number of retained singular triplets is 37 with sv-cut = 0.0004 at convergence.

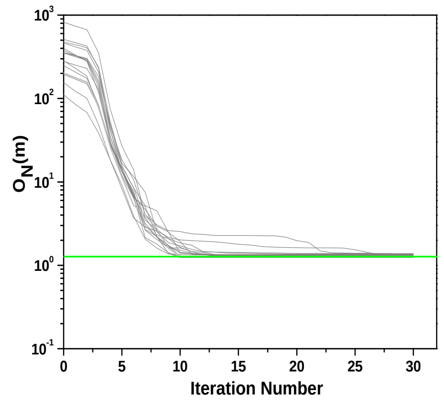
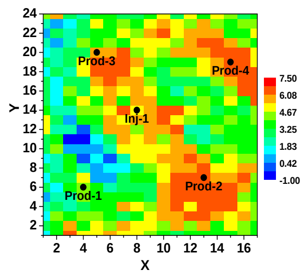
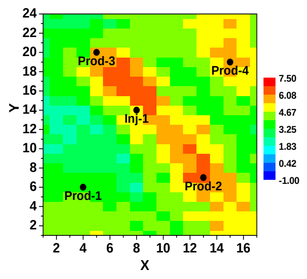


Figure 4.64: Behavior of the normalized objective function for each realization during iteration of SVD-GF-EnRML with $m_{\text{initial}} = m^{\text{a}}_{\text{EnkF, Loc}}$, Example 2.

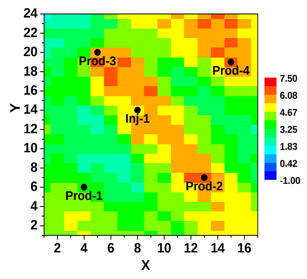
The average of the first 15 realizations of the final log-permeability fields obtained from the EnKF method, i.e., the average of the initial guesses of log-permeability fields for the SVD-GF-EnRML algorithm is shown in Figure 4.65(b). The final average of realizations of log-permeability field obtained from the SVD-GF-EnRML-EnKF algorithm is shown in Figure 4.65(c). As can be seen from this figure, the essential features of the truth have been captured but are different than the EnKF results of Figure 4.65(b).



(a) True $\ln(k)$.



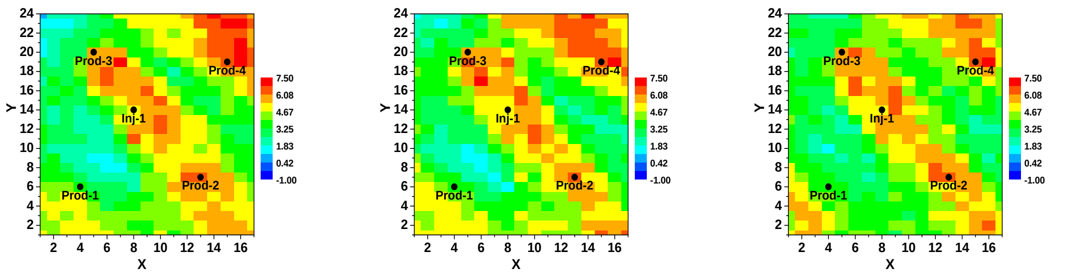
(b) EnKF with anisotropic localization.



(c) SVD-GF-EnRML-EnKF.

Figure 4.65: True log-permeability field (a) and average of first 15 realizations of final $\ln(k)$ from EnKF with anisotropic covariance localization (b) and from SVD-GF-EnRML algorithm using $m_{\text{initial}} = m^{\text{a}}_{\text{EnkF, Loc}}$ (c), Example 2.

Figure 4.66 shows the final conditional log-permeability fields obtained with the SVD-GF-EnRML-EnKF algorithm. The corresponding unconditional realizations are shown in Figure 4.42 and the corresponding initial guesses (final log-permeability realizations from EnKF method) are shown in Figure 4.44. As can be seen from Figure 4.66, the essential features of the high permeability region connecting Prod-2 and Prod-3 and the high permeability region around Prod-4 that exist in the true model have been captured in each individual conditional realization. Also, each realization shows some features of the low permeability region exist in the south-west part of the reservoir; however, there is variability between each realization.

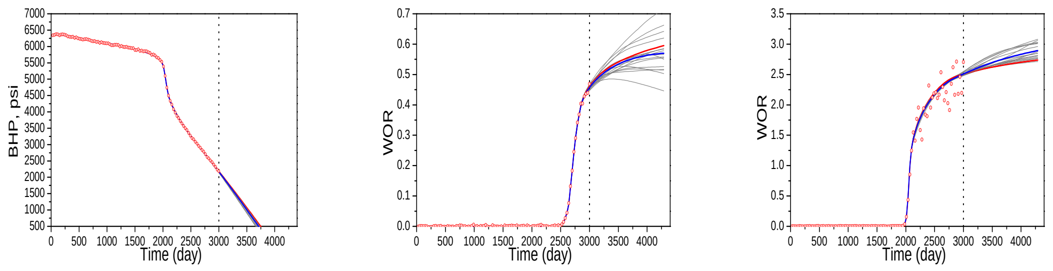


(a) Realization 4. (b) Realization 8. (c) Realization 12.

Figure 4.66: Three conditional realizations of log-permeability fields from SVD-GF-EnRML algorithm using $m_{\text{initial}} = m^{\text{a}}_{\text{EnkF, Loc}}$, Example 2.

The history matching results and future forecasts of the BHP of Prod-3 and the WOR of Prod-2 and Prod-3 obtained with the SVD-GF-EnRML-EnKF algorithm ($m_{\text{initial}} = m^{\text{a}}_{\text{EnkF, Loc}}$) are shown in Figure 4.67. As can be seen from this figure, very good matches of the BHP and WOR are obtained. The future performance predictions ($t > 3000$ days) of the BHP of Prod-3 and the WOR of Prod-2 are comparable with truth. The future forecasts of the WOR of Prod-3 are in good agreement with the truth until 3300 days after which the mean prediction of the realizations over predict the WOR of Prod-3. The data matches and future predictions obtained with SVD-GF-EnRML-EnKF algorithm are better than the corresponding results obtained with EnKF (Figures 4.50(c), 4.56(c) and 4.62(c)).

The last row of Table 4.2 gives the summary of the computational cost of the SVD-GF-EnRML-EnKF algorithm based on the number of simulation runs, forward gradient and adjoint solutions. The comparison of this row with the computational cost of the SVD-GF-EnRML algorithm (results of the first row of Table 4.2) indicates that we have significantly improved the computational efficiency of the SVD-GF-EnRML algorithm by generating a set of initial guesses of the log-permeability field from EnKF, i.e., we gain 71 percent computational savings with the combined algorithm.



(a) BHP Prod-3. (b) WOR Prod-2. (c) WOR Prod-3.

Figure 4.67: Matches and predictions of BHP and WOR from SVD-GF-EnRML using $m_{\text{initial}} = m^{\text{a}}_{\text{EnkF, Loc}}$, open red circles are observed data, red is from true model, blue is the prediction mean, gray curves represent predictions from realization, Example 2.

4.5.7 Toy Problem

Because EnKF is constructed to estimate the conditional mean of the posterior conditional pdf at each data assimilation step, the EnKF ensemble of realizations may provide a poor representation of the uncertainty if the sequence of posterior pdf's are non-Gaussian. Zafari and Reynolds [65, 64] illustrated this EnKF difficulty by considering two toy problems with multimodal conditional pdf's. Here, we reconsider the toy problem of Zafari and Reynolds [65] simply to illustrate that if we use the final ensemble of vectors of model parameters as initial guesses in SVD-EnRML or SVD-GF-EnRML, we generate an good approximation of the true posterior pdf.

For this toy problem, there is a single real parameter m which has a prior Gaussian distribution with mean equal to 2.0 and variance equal to 0.2. The forward model is given by

$$d = g(m, t) = 1 - \frac{9t}{2}(m - \frac{2\pi}{3})^2. \tag{4.35}$$

where m is a real random variable and t acts as time. This equation can be rewritten as a recursive equation so that it mimics a reservoir simulator:

$$g(m, t + \Delta t) = g(m, t) - \frac{9\Delta t}{2}(m - \frac{2\pi}{3})^2. \tag{4.36}$$

The true synthetic data are generated with $m_{\text{true}} = 1.88358$, at three times ($t_j = j$, $j = 1, 2, 3$). For each time, synthetic observed data were generated by adding normal random noise generated from $N(0, 0.01)$ to true synthetic data as measurement error. Figure 4.68 shows the prior pdf and the posterior pdf conditional to three observed data. As can be seen from this figure, the posterior pdf has two modes, one approximately at $m_{\text{true}} = 1.88358$ and another at $m = 2.3$.

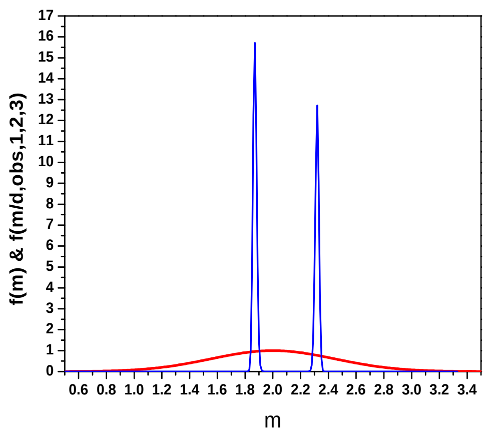
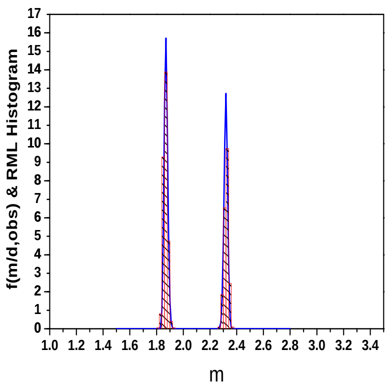
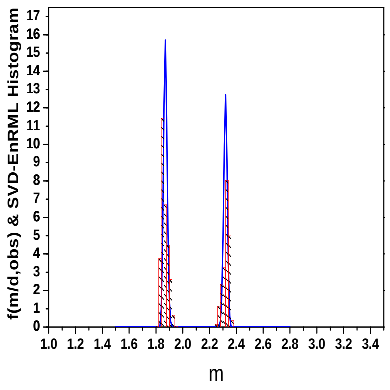


Figure 4.68: Prior pdf (red curve) and posterior pdf (blue curve) after integrating 3 data.

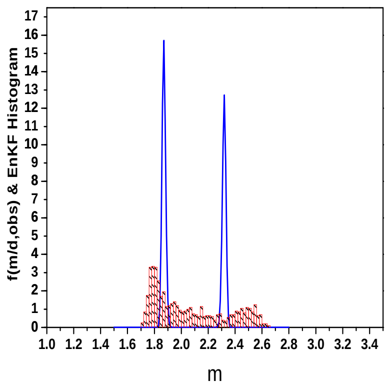
We generate 1000 unconditional realizations of the model parameters from the prior distribution and generate 1000 conditional realizations with three different algorithms of the standard RML algorithm, SVD-EnRML algorithm and by assimilating data with EnKF. We compare these distributions of m to the true posterior pdf. Then we compare the EnKF results with those obtained from SVD-EnRML and SVD-EnRML-EnKF where we use the EnKF ensemble as the ensemble of initial guesses for SVD-EnRML. The histograms of the conditional realizations of m



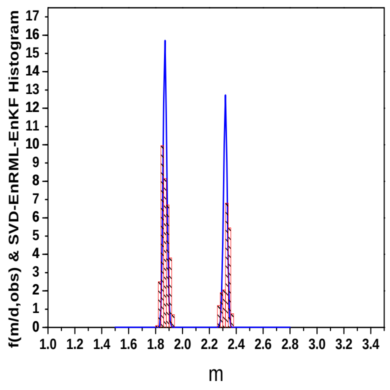
(a) RML.



(b) SVD-EnRML.



(c) EnKF.

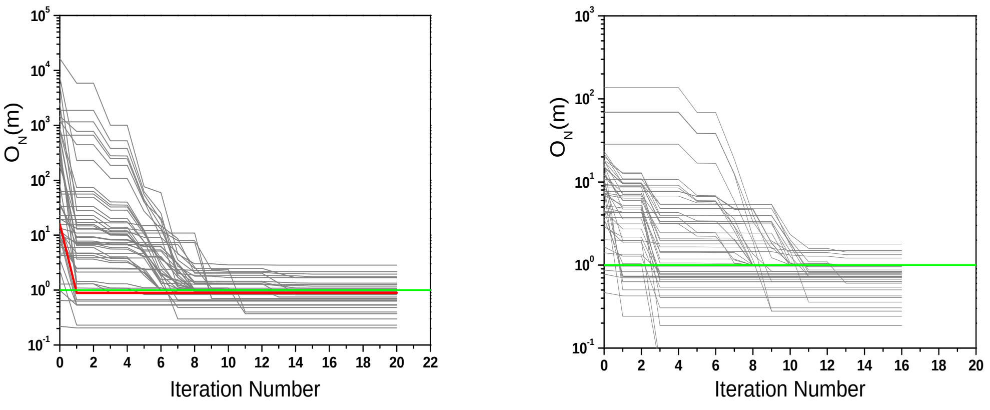


(d) SVD-EnRML-EnKF.

Figure 4.69: Posterior pdf (blue curve) and realizations histogram (red patterns) obtained with: (a) RML, (b) SVD-EnRML, (c) EnKF and (d) SVD-EnRML-EnKF algorithms.

obtained with standard RML and SVD-EnRML algorithms are shown, respectively, in Figures 4.69(a) and 4.69(b). As can be seen from these figures, both algorithms generated conditional realizations which provide good approximation of the true posterior pdf. Figure 4.69(c) shows the histogram of the ensemble of model parameters obtained by assimilating 3 data obtained with EnKF compared to the true posterior pdf. The EnKF generated ensemble provides a poor approximation to the true posterior pdf whereas SVD-EnRML-EnKF provides a good approximation, Figure 4.69(d).

The behavior of the normalized objective function during iteration for realizations that were obtained using the SVD-EnRML algorithm with $m_{\text{initial}} = m_{\text{uc}}$ and $m_{\text{initial}} = m^{\text{a}}_{\text{EnkF}}$ (SVD-EnRML-EnKF), respectively, are shown in Figures 4.70(a) and 4.70(b). As shown in these figures, the normalized objective function has decreased during iterations in both cases and at convergence we have obtained low values of the maximum of the normalized objective function.



(a) SVD-EnRML with $m_{\text{initial}} = m_{\text{uc}}$. (b) SVD-EnRML with $m_{\text{initial}} = m^{\text{EnkF}}_{\text{a}}$.

Figure 4.70: Behavior of the normalized objective function for each realization during iteration of the SVD-EnRML algorithm with (a) $m_{\text{initial}} = m_{\text{uc}}$ and (b) $m_{\text{initial}} = m^{\text{a}}_{\text{EnkF}}$, red curve represents MAP estimate, gray curves represent realizations.

We obtained qualitatively similar results by implementation of the SVD-GF-

EnRML and SVD-GF-EnRML-EnKF algorithms (not shown).

CHAPTER 5

CONCLUSIONS

The main focus of this work was to develop and implement efficient parameterization algorithms for history matching. In the basic algorithms implemented here, at each iteration of the Levenberg-Marquardt algorithm, the change in the vector of model parameters, δm, is represented by a linear combination of the right singular vectors of a dimensionless sensitivity matrix. We have presented theoretical arguments that show that this parameterization is optimal if the objective is to minimize the posterior uncertainty in model parameters. The Lanczos algorithm can be used to generate the truncated singular value decomposition in a way that avoids explicit computation of any sensitivities.

The computational cost of generating a set of singular triplets is quite high, but computational efficiency can be improved by combining parameterization based on the right singular vectors corresponding to the largest singular values with the LBFGS algorithm as a new LBFGS-SVD algorithm. We observed that the initial rate of reduction in the objective function is almost independent of the number of singular triplets. Therefore, the computational efficiency can also be improved by a modified SVD parameterization where we gradually retain more right singular vectors in the parameterization at each iteration of the optimization algorithm. To further improve computational efficiency of the SVD parameterization algorithm, we developed the SVD-Grd algorithm. In this algorithm, we added an inner loop iteration in which we explicitly compute the gradient of the objective function with the adjoint method, where during the inner loop iteration the truncated SVD parameterization computed in the outer loop does not vary. Our computational examples indicate that the SVD-

Grd algorithm requires less computational cost than the basic SVD parameterization algorithm.

For both examples presented in this work, and other examples we have considered, truncated SVD based on retaining singular triplets such that the ratio of the minimum singular value to the maximum singular value was on the order of 0.001 was sufficient to obtain a reasonable estimate of the permeability field consistent with the geology of the truth and good data matches.

We have also incorporated a truncated-SVD parameterization algorithm with the RML method to obtain two iterative algorithms (SVD-GF-EnRML and SVD-EnRML algorithm) for history-matching production data and characterizing the uncertainty in reservoir description and future performance predictions. Both algorithms require generation of the singular triplets of the dimensionless sensitivity matrix associated with a particular realization at each iteration.

For generating multiple realizations conditioned to dynamic data, these algorithms are far more efficient than previous implementations of gradient-based algorithms. When applied to characterize permeability fields, the algorithms gave reasonable characterizations of the true permeability field, good data matches and good future performance predictions.

Although the SVD-based algorithms are far less efficient than EnKF, they are far more robust. In particular, they do not suffer from the statistical inconsistency that can significantly degrade the results of EnKF for highly nonlinear problems; the SVD-based algorithms can locate multiple modes when the posterior pdf is multimodal and the SVD-based algorithms never encounter loss of rank issues that can lead to ensemble collapse when EnKF is used. The SVD-based algorithms yield better matches of production data and more reliable future performance predictions than are obtained with EnKF.

The performance of the EnKF technique and the effect of covariance localization has been investigated. The rank deficiency problem of the EnKF technique and the spurious correlations issue have been mitigated with covariance localization particularly for the case where the ensemble size is small; we have obtained good history match and prediction results and the final model parameters gave feasible approximations of the truth. Even though we obtained good history matching results during data assimilation time with covariance localization, we obtained inconsistent results when we reran the simulator forward from time zero.

By using the ensemble of realizations of model parameters obtained by EnKF with localization as the initial guesses in the SVD-based algorithms, the computational efficiency of the SVD-algorithm was significantly improved. This result should be considered carefully, however. If one used EnKF without localization and obtained ensemble collapse, using the ensemble of EnKF realizations as initial guesses in the SVD-based algorithms would probably not be appropriate.

BIBLIOGRAPHY

[1] Y. Abacioglu, D. S. Oliver, and A. C. Reynolds. Efficient reservoir history matching using subspace vectors. *Computational Geosciences*, 5(2):151–172, 2001.

[2] F. Alabert. The practice of fast conditional simulations through the LU decomposition of the covariance matrix. *Mathematical Geology*, 19(5):369–386, 1987.

[3] 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.

[4] James V. Beck and Kenneth J. Arnold. *Parameter Estimation in Engineering and Science*. John Wiley & Sons, Chichester, England, 1977.

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

[6] 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.

[7] 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.

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

[9] M. Davis. Production of conditional simulations via the LU decomposition of the covariance matrix. *Mathematical Geology*, 19(2):91–98, 1987.

[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] C. R. Dietrich. Optimum selection of basis functions for ill-posed problems. In *ModelCARE 90: Calibration and Reliability in Groundwater Modelling*, volume 195, pages 13–22. Int. Assoc. of Hydrol. Sci., 1990.

[12] Geir Evensen. Sequential data assimilation with a nonlinear quasi-geostrophic model using Monte Carlo methods to forecast error statistics. *Journal of Geophysical Research*, 99:10143–10162, 1994.

[13] Geir Evensen. The combined parameter and state problem. In *Technical Report*. Norsk Hydro Research Center, 2005.

[14] Geir Evensen. *Data Assimilation: The Ensemble Kalman Filter*. Springer, Berlin, 2007.

[15] Frans J. T. Floris, M. D. Bush, M. Cuypers, F. Roggero, and A-R. Syversveen. Methods for quantifying the uncertainty of production forecasts: A comparative study. *Petroleum Geoscience*, 7(SUPP):87–96, 2001.

[16] Reinhard Furrer and Thomas Bengtsson. Estimation of high-dimensional prior and posterior covariance matrices in Kalman filter variants. *J. Multivar. Anal.*, 98(2):227–255, 2007. ISSN 0047-259X.

[17] Guohua Gao. *Data Integration and Uncertainty Evaluation for Large Scale Automatic History Matching Problems*. Ph.D. thesis, University of Tulsa, Tulsa, Oklahoma, 2005.

[18] Guohua Gao, Gaoming Li, and A. C. Reynolds. A stochastic algorithm for automatic history matching. *SPE Journal*, 12(2):196–208, 2007.

[19] Guohua Gao and A. C. Reynolds. An improved implementation of the LBFGS algorithm for automatic history matching. *SPE Journal*, 11(1):5–17, 2006.

[20] Guohua Gao, Mohammad Zafari, and A. C. Reynolds. Quantifying uncertainty for the PUNQ-S3 problem in a Bayesian setting with RML and EnKF. *SPE Journal*, 11(4):506–515, 2006.

[21] G. Gaspari and S. E. Cohn. Construction of correlation functions in two and three dimensions. *Quarterly Journal of the Royal Meteorological Society*, 125 (554):723–757, 1999.

[22] 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.

[23] G. H. Golub, F. T. Luk, and M.L. Overton. A block lanczos method to compute the singular values and corresponding singular vectors of a matrix. Technical report, Computer Science Department, Standford University, 1977.

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

[25] A. A. Grimstad, T. Mannseth, S. A. Aanonsen, I. Aavatsmark, A. Cominelli, and S. Mantica. Identification of unknown permeability trends from history matching of production data (SPE-77485). In *2002 SPE Annual Technical Conference and Exhibition*, 2002.

[26] A. A. Grimstad, T. Mannseth, G. Naevdal, and G. Urkedal. Scale splitting approach to reservoir characterization (SPE-66394). In *2001 SPE Annual Technical Conference and Exhibition*, 2001.

[27] 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.

[28] P. L. Houtekamer and Herschel L. Mitchell. A sequential ensemble Kalman filter for atmospheric data assimilation. *Monthly Weather Review*, 129(1):123–137, 2001.

[29] L. Y. Hu, M. Le Ravalec, G. Blanc, F. Roggero, B. Noetinger, A. Haas, and B. Corre. Reducing uncertainties in production forecasts by constraining geological modeling to dynmaic data. In *Proceedings of the 1999 SPE Annual Technical Conference and Exhibition*, pages 1–8, 1999.

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

[31] 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.

[32] R. E. Kalman. A new approach to linear filtering and prediction problems. *Transactions of he ASME, Journal of Basic Engineering*, 82:35–45, 1960.

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

[34] J. F. B. M. Kraaijevanger, P. J. P. Egberts, J. R. Valstar, and H. W. Buurman. Optimal waterflood design using the adjoint method (SPE-105764). In *SPE Reservoir Simulation Symposium*, page 15, 2007.

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

[36] Ruijian Li. *Conditioning Geostatistical Models to Three-Dimensional Three-Phase Flow Production Data by Automatic History Matching.* Ph.D. thesis, University of Tulsa, Tulsa, Oklahoma, 2001.

[37] Ruijian Li, A. C. Reynolds, and D. S. Oliver. History matching of three-phase flow production data, SPE 66351. In *Proceedings of the 2001 SPE Reservoir Simulation Symposium*, 2001.

[38] Ruijian Li, A. C. Reynolds, and D. S. Oliver. History matching of three-phase flow production data. *SPE J.*, 8(4):328–340, 2003.

[39] Ning Liu, Soraya Betancourt, and Dean S. Oliver. Assessment of uncertainty assessment methods. *Proceedings of the 2001 SPE Annual Technical Conference and Exhibition*, pages 1–15, 2001.

[40] Ning Liu and Dean S. Oliver. Evaluation of Monte Carlo methods for assessing uncertainty. *SPE Journal*, 8(2):188–195, 2003.

[41] Andrew C. Lorenc. The potential of the ensemble Kalman filter for NWP—a comparison with 4D-Var. *Quarterly Journal of the Royal Meteorological Society*, 129(595):3183–3203, 2003.

[42] 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.

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

[44] Herschel L. Mitchell, P. L. Houtekamer, and Gérard Pellerin. Ensemble size, balance, and model-error representation in an ensemble Kalman filter. *Monthly Weather Review*, 130(11):2791–2808–433, 2002.

[45] 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.

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

[47] D.P. O'Leary and J.A. Simmons. A bidiagonalization-regularization procedure for large scale discretizations of ill-posed problems. *SIAM J. SCi. Stat. Comput.*, 2(4):474–489, 1981.

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

[49] Dean S. Oliver, Luciane B. Cunha, and Albert C. Reynolds. Markov chain Monte Carlo methods for conditioning a permeability field to pressure data. *Mathematical Geology*, 29(1):61–91, 1997.

[50] 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.

[51] Dean S. Oliver, Albert C. Reynolds, and Ning Liu. *Inverse Theory for Petroleum Reservoir Characterization and History Matching.* Cambridge University Press, Cambridge, UK, 2008.

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

[53] 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.

[54] 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.

[55] Albert C. Reynolds, Nanqun He, and Dean S. Oliver. Reducing uncertainty in geostatistical description with well testing pressure data. In Richard A. Schatzinger and John F. Jordan, editors, *Reservoir Characterization—Recent Advances*, pages 149–162. American Association of Petroleum Geologists, 1999.

[56] José R. P. Rodrigues. Calculating derivatives for history matching in reservoir simulators (SPE-93445). In *SPE Reservoir Simulation Symposium*, page 9, 2005.

[57] José R. P. Rodrigues. Calculating derivatives for automatic history matching. *Computational Geosciences*, 10:119–136, 2006.

[58] 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.

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

[60] Kristian Thulin, Gaoming Li, Sigurd Ivar Aanonsen, and Albert C. Reynolds. Estimation of initial fluid contacts by assimilation of production data with EnKF. In *Proceedings of the 2007 SPE Annual Technical Conference and Exhibition*, 2007.

[61] C. R. Vogel and J. G. Wade. Iterative SVD-based methods for ill-posed problems. *SIAM J. Sci. Computing*, 15(3):736–754, 1994.

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

[63] William W-G Yeh, Young S. Yoon, and K. S. Lee. Aquifer parameter identification with kriging and optimum parameterization. *Water Resour. Res.*, 19(1): 225–233, 1983.

[64] Mohammad Zafari and Albert C. Reynolds. EnKF versus RML, theoretical comments and numerical experiments. *TUPREP Research Report 22*, pages 195–228, 2005.

[65] Mohammad Zafari and Albert C. Reynolds. Assessing the uncertainty in reservoir description and performance predictions with the ensemble Kalman filter. *SPE Journal*, 12(3):382–391, 2007. SPE 95750-PA.

[66] F. Zhang and A. C. Reynolds. Optimization algorithms for automatic history matching of production data. *Proceedings of 8th European Conference on the Mathematics of Oil Recovery*, pages 1–10, 2002.

[67] F. Zhang, A. C. Reynolds, and D. S. Oliver. Evaluation of the reduction in uncertainty obtained by conditioning a 3D stochastic channel to multiwell pressure data. *Mathematical Geology*, 34(6):713–740, 2002.

[68] Fengjun Zhang. *Automatic History Matching of Production Data for Large Scale Problems*. Ph.D. thesis, University of Tulsa, Tulsa, Oklahoma, 2002.

APPENDIX A

SINGULAR VALUE DECOMPOSITION

Without loss of generality, we assume that G_D is $N_d \times N_m$ where $N_d < N_m$. Then the singular value decomposition of the dimensionless sensitivity matrix is given by

$$G_D = U\Lambda V^T, \tag{A.1}$$

where U is an $N_d \times N_d$ orthogonal matrix, V is an $N_m \times N_m$ orthogonal matrix. Λ is an $N_d \times N_m$ matrix which can be partitioned as

$$\Lambda = \begin{bmatrix} \Lambda_s & O \end{bmatrix}, \tag{A.2}$$

where O is the $N_d \times (N_m - N_d)$ null matrix and Λ_s is the $N_d \times N_d$ diagonal matrix which has the singular values of G_D as its diagonal entries, i.e.,

$$\Lambda_s = \begin{bmatrix} \lambda_1 & 0 & 0 & \cdots & 0 \\ 0 & \lambda_2 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \cdots & 0 \\ 0 & 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & \cdots & \lambda_{N_d} \end{bmatrix}, \tag{A.3}$$

where λ_j is the jth singular value of G_D. We let u_i be the ith column of U and v_j be the jth column of V so that

$$U = \begin{bmatrix} u_1, u_2, \cdots u_{N_d} \end{bmatrix}, \tag{A.4}$$

and

$$V = \big[v_1, v_2, \cdots v_{N_m}\big]. \tag{A.5}$$

The set of u_i's are the left singular vector and the set of v_i's are the right singular vectors.

Because U and V are orthogonal matrices $UU^T = U^TU = I_{N_d}$, $VV^T = V^TV = I_{N_m}$,

$$u_i^T u_j = \delta_{i,j} \text{ for } i,j = 1,2,\cdots N_d, \tag{A.6}$$

and

$$v_i^T v_j = \delta_{i,j} \text{ for } i,j = 1,2,\cdots N_m, \tag{A.7}$$

where $\delta_{i,j}$ is the Kronecker delta function. It is well known [24] and easy to show using Eq. A.1 and Eqs. A.4 through A.7 that

$$G_D v_i = \lambda_i u_i \text{ for } i = 1,2,\cdots N_d, \tag{A.8}$$

$$G_D v_i = 0 \text{ for } i = N_d+1,\cdots N_m, \tag{A.9}$$

and

$$G_D^T u_i = \lambda_i v_i \text{ for } i = 1,2,\cdots N_d. \tag{A.10}$$

Multiplying Eqs. A.8 and A.9, respectively, by G_D^T and using Eq. A.10 gives

$$G_D^T G_D v_i = \lambda_i^2 v_i \text{ for } i = 1,2,\cdots N_d, \tag{A.11}$$

and

$$G_D^T G_D v_i = 0 \text{ for } i = N_d+1,\cdots N_m, \tag{A.12}$$

Thus, the eigenvalue-eigenvector pairs of the $N_m \times N_m$ matrix $G_D^T G_D$ are (λ_i^2, v_i) for $i = 1,2\cdots N_d$ and $(0, v_i)$ for $i = N_d+1,\cdots N_m$.

By multiplying Eq. A.10 by G_D and using Eq. A.8, it follows that

$$G_D G_D^T u_i = \lambda_i^2 u_i, \text{for } i = 1, 2, \cdots N_d, \tag{A.13}$$

so the eigenvalue-eigenvector pairs of the $N_d \times N_d$ matrix $G_D G_D^T$ are (λ_i^2, u_i) for $i = 1, 2 \cdots N_d$.

From Eq. A.13, it follows that

$$\left(I_{N_d} + G_D G_D^T\right)^{-1} u_i = \frac{1}{(1+\lambda_i^2)} u_i, \quad \text{for } i = 1, 2, \cdots N_d, \tag{A.14}$$

so the eigenvalue-eigenvector pairs of the $N_d \times N_d$ matrix $\left(I_{N_d} + G_D G_D^T\right)$ are $(1+\lambda_i^2, u_i)$ for $i = 1, 2 \cdots N_d$.

Multiplying Eq. A.8 by $G_D^T\left(I_{N_d} + G_D G_D^T\right)^{-1}$ and using Eqs. A.14 and A.10 gives

$$G_D^T\left(I_{N_d} + G_D G_D^T\right)^{-1} G_D v_i = \frac{\lambda_i}{(1+\lambda_i^2)} G_D^T u_i = \frac{\lambda_i^2}{(1+\lambda_i^2)} v_i \tag{A.15}$$

for $i = 1, 2, \cdots N_d$.

Similarly, multiplying Eq. A.9 by $G_D^T\left(I_{N_d} + G_D G_D^T\right)^{-1}$, it follows easily that

$$G_D^T\left(I_{N_d} + G_D G_D^T\right)^{-1} G_D v_i = 0 \tag{A.16}$$

for $i = N_d + 1, \cdots N_m$.