OPTIMUM DAMPING FACTOR FOR LEVENBERG-MARQUARDT ALGORITHM WITH APPLICATION TO RESERVOIR PARAMETER ESTIMATION

BY

Xian Zhang

A Thesis Presented to the

DEANSHIP OF GRADUATE STUDIES

KING FAHD UNIVERSITY OF PETROLEUM & MINERALS

DHAHRAN, SAUDI ARABIA

In Partial Fulfillment of the
Requirements for the Degree of

MASTER OF SCIENCE

In

PETROLEUM ENGINEERING

JANUARY, 2015

KING FAHD UNIVERSITY OF PETROLEUM & MINERALS

DHAHRAN- 31261, SAUDI ARABIA

DEANSHIP OF GRADUATE STUDIES

This thesis, written by XIAN ZHANG under the direction his thesis advisor and approved by his thesis committee, has been presented and accepted by the Dean of Graduate Studies, in partial fulfillment of the requirements for the degree of **MASTER OF SCIENCE IN PETROLEUM ENGINEERING.**

Dr. Abeeb A. Awotunde
(Advisor)

Dr. Abdullah S. Sultan
Department Chairman

Dr. Roland N. Horne
(Member)

Dr. Salam A. Zummo
Dean of Graduate Studies

Dr. Hasan Y. Al-Yousef
(Member)

14/1/15
Date

عمادة الدراسات العليا
DEANSHIP OF GRADUATE STUDIES

This work is dedicated to my father, mother, brother and sister for their great love, encouragement and support. Also, I dedicate this work and give my special thanks to my friends studying at KFUPM for their help throughout the process.

ACKNOWLEDGMENTS

Thank you Dr. Abeeb Awotunde. Because of your wonderful lectures on Mathematics and Reservoir Simulation, I started to learn reservoir simulation as my major interest. Also, I should thank you for teaching me a lot of programming skills, which enlighten my interests in the use of MATLAB and other programming languages.

I wish to thank Dr. Roland N. Horne and Dr. Hasan Y. Al-Yousef for serving as my committee members. Actually, the thesis took them a lot of time reading and thinking, but they were always very patient with me. In addition, I really appreciate the technical support of SimOpt Lab towards the completion of my MS thesis at King Fahd University of Petroleum & Minerals.

Finally, I am very grateful to my beloved father, mother, brother and sister. My parents work very hard and give me great financial support through out the years. Special thanks to my brother and sister for reminding me of every traditional Chinese festival while I am studying in the Kingdom of Saudi Arabia. I should also thank my friends, colleagues and entire Chinese, Nigerian, Pakistani, and Cameroonian communities for their support and good wills.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

$\vec{R}^{n+1}$	:	Residual at time level n+1
$\vec{f}^{n}$	:	Solution of mathematical equations at time level n
$\vec{f}^{n+1}$	:	Solution of mathematical equations at time level n+1
β	:	Reservoir properties
δt	:	Time step
$\vec{\alpha}$	:	Model parameters in vector form
M	:	Total number of unknown model parameters
$p_{o,M}$	:	Pressure at gridblock M
$S_{w,M}$	:	Saturation at gridblock M
$P_{wf,N}$	:	Bottom-hole pressure at well N
$\vec{R}_{glk}$	:	Residual of gridblock
$\vec{R}_{well}$	:	Residual of well
ϕ	:	Initial porosity distribution
$\vec{k}$	:	Permeability field in vector form
q	:	Flow rate
γ	:	Mobility Ratio
F	:	Objective function
d_{cal}	:	Calculated data
d_{obs}	:	Observed data
$\vec{d}_{cal}$	:	Calculated data in vector form

$\vec{d}_{obs}$	:	Observed data in vector form
D_{cal}	:	Calculated data space
D_{obs}	:	Observed data space
$C_{D_{obs}}^{-1}$	:	Covariance matrix in data space
$C_{D_{cal}}^{-1}$	:	Covariance matrix in model space
$\vec{s}$	:	Newton search direction
$\vec{g}$	:	Gradient of objective function
H	:	Hessian matrix
B	:	Approximated Hessian matrix
H_{GN}	:	Gauss-Newton Hessian matrix
H_{LM}	:	Levenberg-Marquardt Hessian matrix
v	:	Optimal step size
S	:	Sensitivity coefficients
I	:	Identity matrix
λ	:	Damping factor
χ	:	Scaling factor chosen by modelers
G	:	Generation used for DE algorithm
r_i	:	DE operation at i^{th} position
M_F	:	Mutation factor used for DE algorithm
CR	:	Crossover factor used for DE algorithm
k_{norm}	:	Relative closeness from initial guess to estimate
k_{cal}	:	Calculated permeability field
k_{obs}	:	Observed permeability field

ABSTRACT

Full Name : [Xian Zhang]

Thesis Title : [Optimum Damping Factor for Levenberg-Marquardt Algorithm with Application to Reservoir Parameter Estimation]

Major Field : [Petroleum Engineering]

Date of Degree : [January 2015]

The accurate evaluation of production performance over entire life of reservoir is very important nowadays. With rapid development of technology, automatic history matching allows us to estimate reservoir properties to effectively guide the future development of oilfield. Since the accuracy of history matching heavily depends on the methods that are computationally reliable and efficient, the choice of appropriate optimization algorithms is of great importance for engineers. Practically, the gradient-based algorithms are widely used in reservoir parameter estimation, especially that the Levenberg-Marquardt algorithm gives us faster convergence than other gradient-based algorithms. Unfortunately, the use of this algorithm is limited to small and medium-scale problems where the estimation of damping factor can be done very quickly based on computation of eigenvalues of sensitivity matrix.

For large-scale inverse problems, however, such estimation is difficult and a less-effective method known as trial-and-error approach is used. To overcome this drawback, this work tries to present a new way that uses Differential Evolution, a global search method, to estimate optimum damping factor for Levenberg-Marquardt algorithm. Then, we develop a new correlation between damping factor and statistical values of Hessian.

Final results show that our proposed algorithm shows much faster convergence rate due to elimination of uncertainty in estimating damping factor. More importantly, when our algorithm is used, it yields smaller residuals of objective function compared to that of traditional Levenberg-Marquardt algorithm. In addition, results also prove that correlation derived by our algorithm works well in dealing with reservoirs of different sizes.

ملخص الرسالة

الاسم الكامل: تشيانغ زانغ

عنوان الرسالة: عامل التضاؤل الأمثل لخوارزمية ماركواردت-ليفنبيرغ في مجال التنبؤ بمعايير المكامن

التخصص: هندسة البترول

تاريخ الدرجة العلمية: يناير 2015

يعتبر التقييم الدقيق لأدائية المكامن على طول فترة الإنتاج من أهم المتطلبات في وقتنا الحاضر. مع الوتيرة المتسارعة في التطور التكنلوجي أصبحت عمليات مطابقة التاريخ الإنتاجي من اهم الوسائل المعينة على التقدير الدقيق لخواص المكامن، وهو من الأهمية بمكان في تطوير حقول البترول. يعتبر الاختيار الأمثل لخوارزمية التحسين في غاية من الأهمية للمهندسين وذلك لتأثر دقة عمليات المطابقة على جودة وفعالية الأسلوب الحسابي المستخدم.

تعتبر خوارزميات الأساس المتناقص من أكثر الخوارزميات استخداماً في عمليات التنبؤ بمعايير المكامن حيث تعتبر خوارزمية ماركواردت-ليفنبيرغ الأفضل من بين الكل نسبة لسرعة التقارب العالية لهذه الخوارزمية. بالرغم من ذلك، فأن هذه الخوارزمية تمتاز فقط في المسائل الصغيرة والمتوسطة النطاق والتي يتم فيها التنبؤ بمعامل التضاؤل سريعا وذلك بالاعتماد على حساب معاملات التحول الخطية لمصفوفة الحساسية.

في المسائل كبيرة النطاق يكون من الصعب ومن غير المجدي القيام بعمليات التنبؤ باستخدام أسلوب التجربة والخطاء. للتغلب على هذه المعضلة، قامت الدراسة الحالية باقتراح طريقة جديدة باستخدام التقييم التفاضلي وذلك للتنبؤ بالقيمة المثلى لمعامل التضاؤل لخوارزمية ماركواردت-ليفنبيرغ. من بعد ذلك قامت الدراسة بتطوير علاقة رياضية تربط بين معامل التضاؤل والقيم الإحصائية لمصفوفة هيسيان. أظهرت النتائج المتحصل عليها افضلية الخوارزمية المقترحة نسبة سرعة التقارب العالية والتي اعتمدت على حذف عدم التيقن في حسابات معامل التضاؤل. أيضا أظهرت النتائج أن استخدام الخوارزمية المقترحة ينتج عنه متبقيات أصغر لدالة الهدف مقارنة

بخوارزمية ماركواردت-ليفنبيرغ. إضافة الى ذلك فقد وجد أن العلاقة الحسابية التي تم تطويرها تعمل بصورة فعالة مع مختلف احجام المكامن.

CHAPTER 1

INTRODUCTION

1.1 Overview of Levenberg-Marquardt Algorithm used in History Matching

To predict field production performance, modeling technique is an essential aspect of the project. The general procedures include establishing mathematical model, applying fitting techniques, and minimizing the objective function which is calculated as the median of the square residuals. The smaller the value of an objective function is, the better the result will be. In terminology, such small value is called global minimum corresponding to the best approximation to the unknown parameters or model parameters. In addition, most mathematical models accounting for fluid flow in the porous media are often nonlinear problems with respect to the model parameters. Currently, we have to use iterative methods to search for global minimum. Take the least square method for instance, it requires a starting value at the beginning of the optimization, then the successive iterations are executed by minimization requirement until convergence criteria is satisfied.

It is well known that the advantage of the least square method is the broad range of functions that can be fitted and the efficient use of data. However, the shortcoming of

this method is the starting value which needs to be close to the region of confidence interval where global minimum is located to ensure the convergence. Once the starting value is far from that region, the least square method will be trapped in local minimum or may not converge at all. Thus, modelers have to switch to the use of Newton-type algorithms such as steepest descent and Gauss-Newton method. Steepest descent is a first-order optimization algorithm. To find a local minimum of an objective function, it uses gradient descent that takes steps proportional to the negative of the gradient of the function at the current point, then gradually iterating to the best approximation. Contrary to steepest descent, another method is the use of Taylor expansion, which is known as Gauss or Gauss-Newton. This method corrects model parameters by introducing the Hessian, which gives much faster convergence than Newton algorithm. Nevertheless, both steepest and Gauss-Newton are not good at searching global minimum. They are often trapped in local minimum. Moreover, steepest descent shows slow convergence rate while Gauss-Newton method has a problem of divergence after successive iterations.

In fact, Levenberg (1944) studied the disadvantages of steepest descent and Gauss-Newton algorithm. He developed a new method which combines steepest descent and Gauss-Newton, by introducing a product of an identity matrix and a non-negative value which is known as damping factor. Normally, if the iteration shows sufficient reduction in residuals, damping factor will be decreased, thus giving a step closer to Gauss-Newton direction. Instead, damping factor will be increased if the iteration gives insufficient reduction in residuals. In this case, the algorithm gives a step closer to gradient descent direction. However, Levenberg's algorithm has a disadvantage that when damping factor is too large, the inverse of Hessian is no use at all. Later, Marquardt

(1963) modified Levenberg's algorithm by replacing identity matrix with a diagonal matrix, resulting in Levenberg-Marquardt algorithm (LM). This new algorithm is able to effectively prevent Hessian matrix to be close to singular. In this way, the nonlinear problems will always have solution in the following successive iterations. According to the LM algorithm, it is found that if the damping factor used at first iteration reduces the error, the damping factor will be divided by a reduction number before next iteration. If the damping fails to reduce the error, it will be multiplied by an amplification number. In this way, the LM algorithm has the capacity of controlling the convergence rate to be slow or fast to ensure that a solution can be found. In other words, the LM algorithm automatically switches from steepest descent to Gauss-Newton. The way Marquardt used to estimate damping factor is so called trial-and-error approach.

Initially, trial-and-error approach enables LM to perform very well in solving small and medium-scale problems. Nowadays, large-scale problems frequently appear in history matching. Such trial-and-error approach may become less effective. Therefore, the implementation of LM may become unreliable for large-scale history matching problems. To find an optimization algorithm that is computationally efficient and reliable, we investigate the application of new algorithm to estimate damping factor for the LM algorithm.

1.2 Objectives of Thesis

In Section 1.1, we have simply overviewed the use of LM algorithm in history matching problems. Based on the drawback of LM, we try to use Differential Evolution, a global search method, to estimate optimum damping for LM. Then, we propose a new algorithm

that uses DE to estimate damping factor for LM so that it is able to solve inverse problems of different size. Therefore, the objectives of this thesis mainly include the following aspects:

(1) Develop an algorithm that uses DE to estimate optimum damping factor for LM;
(2) Apply this method to three synthetic reservoirs of different size representing small, medium, and large-scale inverse problems, to check if the proposed method is feasible for history matching;
(3) If the developed algorithm is more reliable and efficient than traditional LM algorithm, we furthermore develop some correlations based on statistical values of Gauss-Newton Hessian and optimum damping factors;
(4) Apply those correlations to test small and large-scale inverse problems again to see if there exists a generalized correlation that could be used throughout the oilfields in the world.

1.3 Thesis Organization

The thesis comprises of five chapters. Chapter 1 gives a brief introduction of thesis background and objectives of thesis. Chapter 2 gives us detailed literature reviews that summarize the existing researches that are close to our topic. Particularly, it focuses on gradient-based Newton-type optimization algorithms and evolutionary algorithms. Chapter 3 summarizes mathematical formulation related to gradient-based optimization algorithms and Differential Evolution. Chapter 4 mainly introduces the proposed algorithm and how it is implemented during the optimization. Chapter 5 presents the major results of applying this method to three synthetic reservoir models of different

sizes which represent the small, medium, and large-scale inverse problems. It also shows details of the correlation development and results of using correlation for reservoir parameter estimation. Chapter 6 briefly summarizes the conclusions that were obtained during the working of this thesis.

CHAPTER 2

LITERATURE REVIEW

In this Chapter, we will review the existing literatures which are closely linked with our topic. The literatures may include history matching, computation of sensitivity coefficients, descent algorithms, and evolution algorithms. The purpose of this Chapter is to help us to present appropriate academic research background and build up a general research framework.

2.1 History Matching

History matching is a powerful tool referred to the act of adjusting a model of a reservoir until it closely reproduces the past behavior of a reservoir. It is often used in predicting the field performance associated with inverse problems in which the observed data is used to estimate reservoir parameters or model parameters that have some physical interpretation. For instance, we want to estimate permeability distribution throughout the whole reservoir based on matching production data such as bottom-hole flowing pressure and flow rate from production wells. But two major concerns which pose great influence on performance of history matching are reliability and efficiency. Reliability largely depends on the mathematical model we are using. If the mathematical model used in history matching is not logically consistent or impractical to describe reservoir system, the matching results will be definitely useless. With a reliable mathematical model, the selection of optimization strategies furthermore determines the efficiency of history

matching. If both of the two aspects are well studied and selected, modelers may achieve excellent matching results.

In the first aspect, the establishment of mathematical model should be closely associated with model parameters. Moreover, model parameters also play significant roles in a numerical reservoir simulator because of their contribution to the model evaluation during the optimization. In fact, without proper starting values chosen by modelers at the beginning of the optimization, the numerical simulator can not run. Since parameter and data are essential for building up equation system, the specification of parameter and data are of great importance. Oliver and Chen (2011) stated that the required parameters in history matching often involve permeability, porosity, density, oil-water contact, gas-oil ratio, net-to-gross rations, transmissibility, aquifer size, and fluid composition, etc. Other parameters serving as auxiliary parameters to estimate reasonable distribution of reservoir properties include the absence of tar mats, presence of other type of rock, location of high permeability channels, and location of shale barriers, etc. In manual history matching, modelers often need to adjust some parameters such as pore volume, and vertical to horizontal permeability ratio to match reservoir performance. But the number of adjusted parameters is not very big, perhaps several variables at most. For automatic history matching, modelers are flexible to adjust a large number of variables, perhaps dozens of them. Often the selection of variables needed to be adjusted depends on the method that modelers use and should not exceed the range of ultimate use.

In history matching, one type of required parameters for mathematical models is seismic data obtained by seismic surveys, which gives us a general description of reservoir geostatistical information and reservoir properties. Take 4D seismic survey for

instances, it is very common in the exploration that happened in the frontier regions where the hydrocarbon resources are buried at a deep depth with hash surrounding environment. The seismic survey mainly involves the data acquisition, processing, interpretation of repeated seismic survey over a producing field. So, seismic data can help engineers to understand the changes of reservoir over time, especially its production behavior. This understanding is very important, perhaps a few percent in increasing the recovery factor means great revenue implications for the reservoir exploration. Thus, the seismic data allows us to observe changes in subsurface without well constraints because drilling wells is not practical compared to invest cost. In fact, seismic data nowadays becomes convenient for engineers to indirectly investigate the reservoir properties governed by physical laws and fluid behavior. Dadashpour et al. (2008) showed that the reservoir parameters such as porosity and permeability are often converted to seismic data in terms of P-wave and S-wave by using Gassmann's equation and Hertz-Mindlin model, both of which are widely used to estimate seismic data caused by fluid saturation and pore pressure, respectively. In addition, Schiozer and Ferreira (2013) showed that seismic data can provide us field-wide information by identifying producing zones and bypassed oil. They pointed out that a special consideration needs to be applied in the case of 4D seismic information since it is related to the movement of fluids in the reservoir.

In practice, the seismic data can be acquired from either the surface seismic survey or borehole at different times over the target area to evaluate changes in the subsurface. Thus, seismic data and production data are often obtained at the same time. In history matching, different types of data have different implications. Landa and Horne (1997), Landa (1999) classified those data into two groups: static data and dynamic data.

Static data contains data obtained from geology, electrical logs, core analysis, geostatistics, and seismic. While, dynamic data contains data obtained from seismic survey referred to the movement of fluids in the reservoir, well testing, and pressure shut-in survey, etc. Among those dynamic data, only seismic data is areally distributed while others either come from production wells or injection wells. As for data that have been used in history matching, Landa and Horne (1997) also gave us suggestions that using all or at least most of the information simultaneously. In terminology, the process of handling those different data simultaneously is known as data integration. Huang (1999) classified the method of data integration as data-based integration and model-based integration. The first method involves rock physics study, modeling, time lapse seismic data processing and reconciling the time lapse data with engineering data. While, the second method involves model optimization that requires finding the global minimum of an objective function in terms of discrepancy between observed and calculated data. In this way, the seismic data can be estimated by forward modeling using Gassmann's equation. The model is updated by either trial-error or gradient-based algorithm. So far, static information has already obtained great success in data integration while completely integrating dynamic data with static data is not very common in reality because building a proper model honoring both types of data is rather difficult.

In the second aspect, once model and parameters are well prepared, the selection of optimization strategy is of vital importance that determines the computation efficiency. Often the specific problems for inverse model are ill-posed, resulting in extensive computation time for optimization. Therefore, many efforts have been done to improve history matching to estimate reservoir parameters. Kruger (1961) calculate the areal

permeability distribution for a two-phase reservoir when analyzing flooding and cycling projects, which is the first study on history matching. His method allows verification of the model parameter by matching the past reservoir production behavior. In this work, the key aspect is to match two sets of pressure data which are observed data and calculated data based on iteratively updating permeability. Later, Jacquard (1965) developed a method to automatically interpret the pressure measurement. His method aimed at reduce the error of the least squares between observed and calculated pressure obtained from the mathematical model, by defining a permeability map and a porosity map, then reproducing the production history of wells. Chavent et al (1975) applied the optimal control theory that formulates the history matching problem as a control problem in advance and then calculates permeability distribution. The data to be matched is the pressure of a two-dimensional single-phase oil reservoir. The method he used consists of steepest descent method and the use of an adjoint equation which allows gradient to be obtained in minimum computing time. Especially, the optimal control theory can determine Jacobian matrix without calculation of sensitivity coefficients for every parameter to be estimated. But, the limitation of this work refers to (1) non-dependency of viscosity or compressibility on pressure, (2) simplification of boundary conditions, and (3) acceptance of storage coefficients as known quantities. Fasanino et al. (1986) applied adjoint method combined with geostatistical information and pilot technique for a single-phase history matching of a two-dimensional gas reservoir. The sensitivity coefficients are calculated in terms of product of permeability and thickness. Li et al. (2003) studied a three-dimensional three-phase reservoir. They modified the adjoint equation so that the computation of sensitivity coefficients becomes faster. Their work also shows theoretical

formulation of probability theory and gives a brief introduction of gradient-based Newton algorithms such as Levenberg-Marquardt algorithm. Cheng et al. (2006) applied a compressible streamline formulation and streamline-derived analytic sensitivity to matching permeability distribution, water cut, and gas-oil ratio for a three-phase reservoir. Awotunde (2010a) in his dissertation applied wavelet transform as parameterization technique to reduce data space in order to reduce the cost of computation of sensitivity coefficients. More importantly, he showed an analytical method referred to calculate the damping factor for the Levenberg-Marquardt algorithm.

Different from gradient-based algorithms, the ensemble kalman filter (EnKF) is developed as a sequential data assimilation method which has the ability to estimate a large number of model parameters. It provides multiple simultaneous history matching models and calibration of state variables such as saturation and pressure. For instance, Zhang and Oliver (2009) applied EnKF to estimate more than 200,000 unknown parameters based on matching pressure, gas-oil ratio, and water cut.

2.2 Computation of Sensitivity Coefficients

Sensitivity coefficients are used for observing the changes of output of calculated data with respect to small changes in estimates. They can help us to search for errors during the optimization, and find the trust regions in the space where input variables and output variables are reasonable without exceeding their limits. Especially, the efficiency of gradient-based Newton algorithms largely depends on the computation of sensitivity coefficients. Once a tremendous amount of work is required by computation of sensitivity coefficients, methods such as Newton algorithm and Gauss-Newton may become less efficient. To avoid computing sensitivity coefficients, modelers have to use conjugate

gradient and quasi-Newton algorithms to replace gradient-based algorithms. However, both conjugate gradient and quasi-Newton algorithms show very slow convergence rate and thus time consuming. This fact results in less efficient use in real large-scale problems.

Currently, three methods are often used to calculate sensitivity coefficients including finite difference, the forward sensitivity, and the adjoint method. The finite difference method involves perturbing one model parameter at a time to calculate corresponding changes in sensitivity coefficients. After finishing all perturbations of model parameters, modelers can obtain a matrix storing those sensitivity coefficients which is known as Jacobian matrix. Although the finite difference method is simple to implement, it may become very time consuming if the number of model parameter is very huge. Assume that there are M grid blocks in total. If we want to estimate permeability in each grid block, there will be M unknown model parameters. So we need a number of times equating M to run the forward simulator to perturb all of model parameters. Then, one extra operation is needed to calculate sensitivity coefficients. So the total number of running the forward simulator will equate $M+1$. Apparently, this method is cumbersome when M is too big. To overcome this shortcoming of the finite difference method, the forward sensitivity method has been developed. It requires a solution in the form of multiple right-hand side vectors of a linear system. The number of right-hand side vectors equates that of model parameters. When numerical simulator is working, the sensitivity of variables at each grid block will be calculated. But this also brings modelers new problem that we may only need the sensitivity coefficients at locations of production wells. So the forward sensitivity method may have some

redundant parts associated with calculation. If the problems are small or medium-scale, this shortcoming doesn't affect the performance of the simulator. Rather, the forward sensitivity method has been proved to be more efficient when model space is small. But when model space is very large or we are dealing with high dimensional problems, the forward sensitivity method will become cumbersome, too. The work referred to the application of forward sensitivity method have been done by Sun and Yeh (1985), Chen et al. (1989), Bissell (1994), Sun et al, (2001), Rodrigues (2006a), and Awotunde (2010a). From these work, we find that the forward sensitivity method first used in ground water inverse problems and then it is furthermore improved in field of mathematics. Only Bissell (1994), Rodrigues (2006a), and Awotunde (2010a) presented the application of forward sensitivity method in history matching.

The finite difference method and forward sensitivity method may not be suitable for large-scale problems in which the model space is very large. The adjoint approach is developed to compute the sensitivity coefficients and nowadays it becomes very popular in history matching. Jacquard (1965) first used the adjoint method for matching 14 unknown parameters from a two-dimensional single-phase transient flow model. Then, Chavent et al. (1978) computed gradient of an objective function by using adjoint method for a single-phase problem. Anterion et al. (1989) used adjoint method to reduce model space and then calculate sensitivity coefficients based on a solution of the adjoint equation. Oliver et al. (1999) proposed a discrete adjoint method for calculating sensitivity coefficients of production data for a two-phase flow reservoir. They modified the form of Gausss-Newton algorithm and generate maximum a posteriori estimates. Moreover, they obtained the realization of the reservoir property fields which are

conditioned to a prior geostatistical model, pressure, and water-oil ratio. Later, Oliver et al. (2001) calculated the sensitivity coefficients of production data to permeability distribution and skin factors by using adjoint equations. Oliver et al. (2003) proposed a method using adjoint equations to calculate sensitivity coefficients of wellbore pressure, water-oil ratio, and gas-oil ratio with respect to changes in permeability and porosity at each grid block. In this work, they directly pointed out that this method is very suitable for large number of model parameters. Eydinov et al. (2009) presented a procedure to estimate relative permeability curves with grid block permeability and porosity for a three-phase reservoir. The procedure involves using the Low-memory Broyden-Fletcher-Goldfarb-Shanno (LBFGS) algorithm for optimization and the adjoint method for calculating sensitivity coefficients. In addition, they proposed using B-spline approximation in history matching to estimate relative permeability. Awotunde (2010a) stated that the computation time required for the adjoint method doesn't depend on the number of model parameters. Instead, it depends on the number of data to be matched. He also applied wavelet transform to reduce model space with adjoint method, the computation of sensitivity coefficients becomes more efficient than that of forward sensitivity method and finite difference.

Besides the gradient-based algorithms, new methods proposed for calculating sensitivity coefficients include streamline-based analytical sensitivities and simultaneous perturbation stochastic approximation (SPSA). Vasco and Datta-Gupta (1997) first applied a streamline-based simulator to history matching. They formulated the sensitivities in terms of one-dimensional integrals of analytic functions along stationary streamlines. The advantages of this method are simplifying the adjoint system for an ideal

tracer and easily evaluating the integral of the sensitivity coefficients. Spall (1992) applied SPSA to calculate gradient based on a simultaneous perturbation of each component of the model vector during the optimization. Although this method acts similar to finite difference method, the equation form of gradient is quite different. From literature study, we notice that this method has very limited use in history matching and thus it is not popular.

2.3 Decent Algorithms

Descent methods are closely associated with gradient-based Newton algorithms in which the gradient of an objective function is required to be calculated during the optimization. The decent methods mainly contain steepest descent, conjugate gradient, Newton's algorithm, Gauss-Newton, Quasi-Newton, and the LM algorithm.

Steepest descent can be derived from expansion of Taylor series truncated at first-order. Since the algorithm doesn't require the calculation of second-order derivatives, it becomes very simple and easy to implement. Theoretically, steepest descent is a line search method in which a search direction is opposite the gradient descent direction. When the value of an objective function is minimized during the optimization, this algorithm will generate rapid convergence rate. Welty et al. (1979) described a method for automated history matching of well tests that such method can use Taylor series point and steepest descent direction to define the search direction whenever the classical Gauss-Newton least squares procedure is inefficient.

The conjugate gradient is an algorithm used for solving systems of linear equations. Because it is also an iterative method which produces the exact solution after a finite number of iterations, conjugate gradient can be applied to solve nonlinear partial

differential equations with the use of finite difference method. Compared to steepest descent method, the conjugate gradient converges faster. But every iteration of the conjugate gradient is more expensive than that of steepest descent. Moreover, if the pre-conditioners are used, there will be a significant slow down on convergence rate of conjugate gradient. As a result, conjugate gradient converges much slower than steepest descent. In reservoir simulation, Towler and Killough (1979) applied conjugate gradient to solve for the pressure distribution for large-scale reservoir in which the traditional Gauss elimination is impractical because of round-off errors. In the work, they also tested performance of five different pre-conditioners in a large and heterogeneous reservoir.

Newton's algorithm is an iterative method which is also known as Newton-Raphson method. It can be derived from Taylor series with first-order derivative. When implementing Newton's algorithm, we need to provide a good initial guess and then calculate Jacobian matrix based on first-order derivative of model parameters. However, such good initial guess is not easy to be found out since it covers a wide range of estimates. When our initial guess falls upon the region of true values, it will iterate to the solutions. When our initial guess is far from this region, it will be trapped in local minima or never generate the solutions at all. So the initial guess of Newton's algorithm largely dominates the performance of convergence. Another issue in the use of Newton's algorithm is singularity of the Jacobian matrix. To prevent the Jacobian to be singular, pre-condition is often introduced during the implementation. An application of this algorithm in petroleum industry can be found in Khalid et al. (1998). They developed a procedure that can minimize the impact of the facility model on the global solution

procedure based on using pre-conditioned Newton's algorithm in which a local grid refinement pre-conditioner is used to accelerate the Newton step.

Gauss-Newton algorithm is a modification of Newton's algorithm. It is often used to solve nonlinear least squares problems. The major difference between Gauss-Newton and Newton's algorithm is in the use of first-order derivative to calculate Jacobian. The former calculates Jacobian indirectly based on sensitivity coefficients with respect to model parameters and then it successively iterates to a minimum of an objective function. While the latter achieves this purpose directly by differentiating the objective function with respect to model parameters. For some small problems, the Jacobian is simple to be calculated by Newton's algorithm. But for some medium or large-scale problems, the calculation of the Jacobian analytically becomes extremely expensive, which results in inefficient use of Newton's algorithm. In other words, the success of implementation of Newton's algorithm depends on the accuracy and efficiency of the computation of the Jacobian. So the computation of the Jacobian obtained by Newton's algorithm poses a significant impact on the amount of work and time. To simplify the computation of Jacobian, Gauss-Newton algorithm is used to calculate sensitivity coefficients in the form of a matrix in advance. Then, according to basic matrix operations such as transpose and multiplication, we can obtain the Hessian easily. In this way, we reduce the difficulties of computing the Jacobian to a certain degree. More importantly, Gauss-Newton has been proved to be more efficient than Newton's algorithm in convergence. Additionally, Gauss-Newton allows us a wide range of the selection of initial guess at the beginning of optimization which makes Gauss-Newton more flexible than Newton's algorithm. Unfortunately, direct use of Gauss-Newton in history matching is not very popular

because of difficulties in the control of convergence. Sometimes, the diagonal elements of the Hessian are either too big or too small which can not give a good step for next iteration. In other word, the Hessian becomes singular during the optimization, resulting in failure in finding the solution. To efficiently apply Gauss-Newton algorithm in automated history matching, the modification is needed. For instance, Tan and Kalogerakis (1991) modified Gauss-Newton algorithm by indirectly applying linearization of first-order derivative. Then they applied this modified algorithm to test a fully-implicit three-dimensional three-phase reservoir model and possessed a quadratic convergence rate. Later, Tan (1995) modified the calculation of sensitivity coefficients matrix by separating irrelevant state variables during the linearization. Again, he tested this modified Gauss-Newton algorithm with a fully-implicit three dimensional three-phase simulator. He gained a significant improvement in saving time compared to previous work in 1991.

Quasi-Newton algorithm is developed based on Newton's algorithm. As we discussed before, Newton's algorithm is able to calculate first and second order derivatives based on Taylor series expansion. For higher dimensional problems, Newton's algorithm must use gradient and Hessian to update the Newton step. But one of shortcomings in the use of Newton's algorithm is to calculate Hessian and this Hessian must be invertible at each iteration. To eliminate the huge work of computation of the Hessian, Quasi-Newton has the capacity of updating Hessian only by analyzing the successive gradient vectors, which means the Hessian calculated by quasi-Newton algorithm is not an exact Hessian but an approximated one and it is invertible, too. Since the cost of computation is dramatically reduced by the approximated Hessian, Quasi-

Newton is the most efficient and reliable algorithm used for large-scale inverse problems in history matching. Yang and Watson (1988) first introduced variable-metric minimization methods including the Broyden-Fletcher-Goldfarb-Shanno (BFGS) and a self-scaling variable-metric method (SSVM). They tested BFGS and SSVM in hypothetical two-phase reservoir history-matching problems. Kiyoshi (2000) combined the adjoint method and Quasi-Newton algorithm (self-scaling variable-metric method) to a two-phase fluid flow in a heterogeneous reservoir. The matched data is pressure and its derivative is from production wells. Rodrigues et al. (2006b) used the adjoint method to calculate gradient with a trust-region quasi-Newton algorithm for solving history matching problems.

Nevertheless, if the dimension of problems is not very high, take medium-scale problems for instance, the LM algorithm perhaps can give us the fastest convergence in a limited number of iterations. Even if stopping at the designated iteration, the LM algorithm is able to show the smallest value of the objective function. So the use of the LM algorithm for automated history matching flourished in the petroleum industry. Generally, the LM algorithm is a modification of the Gauss-Newton algorithm which is based on a local linearization of the residuals. The Gauss-Newton algorithm often converges very quickly if the initial guess begins sufficiently near the region of minimum. Transtrum and Sethna (2012) pointed out that good starting points can prevent the Hessian of Gauss-Newton algorithm from being ill-posed so that the eigenvalues can span a range of acceptable orders of magnitude. For example, a range changes from 0.0001 to 100. Instead, the eigenvalues of ill-posed problems are often beyond 1000000 or more. Consequently, the Gauss-Newton algorithm is restricted by its staring points.

Furthermore, the eigenvalues are closely associated with magnitude of diagonal elements of Hessian. To modify the Gauss-Newton algorithm, Levenberg first added a product in terms of a damping factor and identity matrix to perturb the diagonal elements of the original Hessian. This damping factor has the ability to make the Hessian always be positive-definite diagonal matrix. Meanwhile, the damping factor can be adjusted by the LM algorithm at each iteration during the optimization. When damping factor is large, the LM algorithm works as steepest descent which is able to take a large step in the gradient direction. When damping factor is small, the LM algorithm iterates to the solution and acts as Gauss-Newton algorithm which takes nearly a zero step length at the end of optimization. Obviously, the selection of damping factor will determine the reliability and efficiency of the LM algorithm.

The basic strategy of selecting a damping factor just uses observations, which is a type of direct method. For example, Marquardt (1963) initially suggested that if a step is acceptable, then decreasing the damping factor by dividing by a reduction constant. When a step is not acceptable, then increasing the damping factor by multiplying the amplification constant. Often the value of reduction constant is suggested to be 0.01 while amplification constant is 10. Sometimes given a specified problem, keeping damping factor to be constant, say 10, is also preferable during the optimization. Watson and Lee (1986) applied LM for history matching problems which have moderate to small number of unknowns. Grattoni and Bidner (1990) applied LM to determine capillary pressure and relative permeability. Matsui and Tanaka (1994) analyzed the relationship between eigenvalues of the product of the Jacobian matrix of performance functions and a damping factor. They found that the value of damping factor should be the median of

the eigenvalues of the Hessian. Based on this idea, they developed a new analytical method for setting an adequate initial value for the damping factor in damped least squares problems.

Later, Zhang et al (2003) proposed a procedure to generate good initial guess by conditioning a stochastic channel to pressure data and well observations of channel facies, channel thickness, and channel top depth. This technique shows improved computational efficiency when the LM algorithm is used for generating realizations of the model by the randomized maximum likelihood method. Araneda (2004) executed many different experimental schemes to find the damping factor. His contributions include investigation of eigenvalues of the Hessian, maximum and minimum elements along the main diagonal elements of the Hessian. But he didn't develop any correlations for optimum damping factor. And all of the cases tested in the experimental work are small-sale simple inverse problems. None of high-dimensional problems are considered. Awotunde and Horne, (2008, 2010, 2011, 2012a, 2014) adopted LM as the major optimization algorithms for history matching problems.

2.4 Evolutionary Algorithms

Evolutionary Algorithms (EAs) are a class of stochastic search and optimization methods that include genetic algorithms, evolutionary programming, evolution strategies, genetic programming, and their variants. The EAs cover a wide range of applications including design, scheduling, control and others. Compared to traditional algorithms such as gradient-based Newton-type algorithms and enumerative strategies, the EAs are relatively robust and more straightforward to the situations that lack priori knowledge. EAs are often based on certain characteristics and behaviors of biological, molecular, swarm of

insects, and neurological systems. The general classification of the EAs is divided into six groups which are (1) genetic algorithm, (2) simulated annealing, (3) particle swarm optimization, (4) ant colony optimization, (5) fuzzy optimization, and (6) neural-network-based methods. So far, many modelers have successfully implemented those algorithms in history matching. Zheng et al. (2002) used genetic algorithm to establish a relationship between permeability and field production data including bottom hole flowing pressure, production rate, water flooding injection rate and water cut. The new correlation can give better results of permeability estimation. Especially, the new correlation is suitable for well pattern or fault-block with balanced injection and production. Prais and Portellaand (1999) presented a method that combines simulated annealing with geostatistical modeling to provide reservoir images given production data. The conditioning technique is used in order to generate geostatistical images. Then images are used to obtain the confidence interval of production prediction by performing a flow simulation. Awotunde (2010b) showed a local-global optimization method that generates multiple realizations of reservoir parameters at coarse scale. The method involves the use of local search optimization algorithm to parameterize the model space at a coarse scale followed by particle swarm optimization method for better estimates in the vicinity of the local estimate. Demyanov et al. (2009) introduced a new stochastic approach for automatic history matching based on a continuous ant colony optimization algorithm. Li et al. (2013) proposed a novel rule-based framework based on fuzzy reasoning to integrate engineering knowledge and assisted history matching workflows. Silva et al. (2007) investigated different types of neural networks as proxies to reservoir simulator. During the matching process, some responses of the simulator are utilized like training sets of the

neural networks resulting in full trained neural networks which substituted the reservoir simulator with high accuracy.

Besides the algorithms mentioned above, Lee and El-Sharkawi (2008) stated that perhaps differential evolution (DE) is the simplest algorithm for solving real value and multi-model objective functions. DE was first proposed by Storn and Price (1996) at Berkeley as a new evolution algorithm. They proposed using a non-uniform crossover together with child vector parameters to guide the minimization process. The mutation operation with DE is executed by arithmetical combinations of individuals rather than perturbing the genes in individuals with small probability, which is very different from genetic algorithms. Another feature of DE is in the search with floating point representation instead of binary presentation. However, fast convergence rate of DE may lead to a high probability towards obtaining a local optimum. This fact is caused by diversity of population descending faster during the solution process. To overcome this, we may need to generate a large population size at the beginning of the optimization. But by doing this, much computational time will be spent to evaluate the fitness function. To save CPU time in solving equations, Wang and Chiou (1998) developed a hybrid algorithm of differential evolution. Later, lots of work related to DE have been proposed over 1999-2009, mainly by modifying either mutation factor or crossover operation. Then, Neri and Tirronen (2010) summarized the advances in DE and also presented a survey related to comparison of efficiency of different versions of DE. He showed a classification with two macro-groups. The first group of DE modification includes algorithms integrating additional components within DE structure. The representative algorithms in this group can be found in work done by Zhang and Sanderson (2007),

Chen and Teng (2008). They use DE as an evolutionary framework assisted by introducing some components such as local searchers and surrogate assisted models. Thus, the algorithms in this group can also be decomposed as DE framework plus extra components. Contrary to the first group, the second group mainly contains algorithms which develop a modified DE structure. For example, Coello (2007) showed the application of such modified DE structure for solving multiple objective functions in the book. Whereas he listed the techniques going from a simple linear aggregating function to the most popular ones on Pareto ranking. Brest (2008) proposed a self-adaptive DE for large-scale problems and constrained optimization. The new modification has a mechanism for changing mutation factor with some probability based on fitness values of randomly chosen vectors.

In petroleum industry, Demyanov et al. (2009) applied DE and neighborhood algorithm to estimate reservoir properties based on matching oil production rate, water production rate, and gas production rate towards a case in Gulf of Mexico. Later, Demyanov et al. (2010) investigated other variants of DE to solve history matching problems. The results of those variants are obtained from DE/Rand, DE/Best, DE/Rand-Best, and DE/Best 2, in which the selection of individual members use different strategies. According to comparison given the same case, DE/Best and DE/Best 2 significantly improved the speed of convergence to find good history-matched models. Awotunde (2014) applied DE to estimate reservoir parameters such as permeability, skin factor, wellbore storage and external reservoir radius in well test analysis. The performance of DE is compared to the Levenberg-Marquardt algorithm, covariance matrix adaptation evolution strategy (CMA-ES), and particle swarm optimization.

CHAPTER 3

MATHEMATICAL FORMULATION

In this section, theoretical foundations related to this work are presented. These concepts are associated with inverse problem theory, probability theory, descent methods, and differential evolution. They provide us the fundamental mathematical principles on which subsequent formulations in this work are presented.

3.1 Inverse Problem Theory

In the petroleum industry, modelers frequently encounter the inverse problems in many branches. It is unavoidable for us to apply inverse analysis to estimate the spatial distribution of reservoir properties. Particularly, when exploration lacks enough contact with reservoir through wells, the production data will be very limited. This fact will lead us to come up with an indirect measurement of reservoir properties. Given such situation, inverse analysis becomes indispensable a tool that allows modelers to estimate reservoir parameters. Or at least, it can help modelers to get a confidence interval for some parameters to a certain degree. For example, inverse analysis has been successfully applied to estimate reservoir properties by using transient pressure measurements taken in the wells. It may also give a good estimation of permeability distribution in history matching.

In general, inverse problems comprise of three parts including parameterization of the system, forward modeling, and inverse modeling. In this section, we will briefly introduce some concepts used in inverse problems theory.

3.1.1 Parameterization

Parameterization is the process of characterization of a system. The concepts of parameterization introduced here mainly involve the parameter, model, and data spaces.

The model parameters are the variables required to be estimated for a specified mathematical model. They are often estimated through optimization. For example, automated history matching often treats permeability distribution as a set of model parameters, in which the permeability in each grid block is what modelers need to estimate by applying optimization algorithms such as gradient-based algorithms and global optimization algorithms.

The data are the values obtained by either solving a system of equations for the model or from measurements taken in production or injection wells. Often modelers classify the data obtained from solving a system of equations for the model as simulated data or calculated data. While the data obtained from the measurements of tools in the field or wells are classified as measured data or observed data. In history matching, observed data often comes from the production wells. Such data include production rates, pressure profile, water cut, and so on. Here, the abbreviations of calculated data and observed data are $\vec{d}_{cal}$ and $\vec{d}_{obs}$ respectively.

The model space, D_{cal}, is the space of all parameters that completely characterize the mathematical model that describes the physical system. Assume that the reservoir is

characterized by permeability distribution. We can represent the vector of model parameters as $\vec{\alpha}$, since permeability in each grid block is required to be estimated. The data space is the space of all physical responses obtained from the measurement of tools. In other word, data space comprises of data including calculated data and observed data. Here, the abbreviation of data space is D_{obs} .

3.1.2 Forward Modeling

Usually, a reservoir is located at a certain depth with very complex configurations and properties. Because of the large size of the reservoir and its complex geological structures, it is impractical to drill wells at every location. So given limited contact with reservoir, the data source is very limited. To obtain the data efficiently, some wells needs to be drilled into the reservoir. Then, gauges are installed in those wells. If the wells are mainly used for producing, gauges will keep a record of flow rates and pressure regimes of wells in the reservoir. And physical measurement tools such as well logging and seismic survey are often used to measure other reservoir parameters. The data measured by the gauges or tools are treated as measured or observed data, $\vec{d}_{obs}$.

If the forward modeling has been established by mathematical methods that describe the reservoir situation very well, modelers need to determine which methods can be used to solve the mathematical equations. Often the equations accounting for fluid flow in the porous media are nonlinear equations. Therefore, the numerical methods are suitable. There are three popular types of numerical methods: finite-difference, finite-element, and finite-volume methods. In this work, a finite-volume reservoir simulator governed by conservation of mass, isothermal fluid phase behavior, and Darcy's law is

used to solve the model problems. The simulator has the ability to deal with three-dimensional, oil-water, oil-gas, and oil-water-gas models. A fully-implicit simulator (Awotunde, 2010a) is adopted for this work. The simulator provides a built-in function to calculate the sensitivity of data to model parameters using adjoint method. Given a three-dimensional reservoir with two-phase flow, the reservoir is discretized into M grid blocks and it has N wells. The general form of discrete system as a residual equation is

$$\vec{R}^{n+1}\left(\vec{f}^{n+1}, \vec{f}^{n}, \beta, \delta t, \vec{\alpha}\right) = \vec{0} \tag{3.1}$$

where $\vec{f}^{n+1}$ represents solution of mathematical equations, such as pressure, water cut, flow rates, saturation, and so on. $\vec{f}^{n}$ is a vector comprising of primary unknowns and bottom hole flowing pressure of wells. For a two phase simulator, $\vec{f}^{n}$ is given by:

$$\vec{f}^{n} = [p_{o,1}, S_{w,1}, \dots, p_{o,M}, S_{w,M}, p_{wf1}, p_{wf2}, \dots p_{wfN}] \tag{3.2}$$

And β is the vector of a known reservoir property, such as porosity, compressibility, fluid density, viscosity, and so on. δt is time step required by simulator. $\vec{\alpha}$ is the model parameter needed to be estimated. In the Equation (3.1), $\vec{R}^{n+1}$ consists of residuals derived from conservation of mass from grid block and wells. Thus, $\vec{R}^{n+1}$ is given by:

$$\vec{R}^{n+1} = \vec{R}^{n+1} = \begin{pmatrix} \vec{R}_{blk}^{n+1} \\ \vec{R}_{well}^{n+1} \end{pmatrix} \tag{3.3}$$

where,

$$\vec{R}_{blk}^{n+1} = [\vec{R}_{o,1}^{n+1}, \vec{R}_{w,1}^{n+1}, \dots, \vec{R}_{o,M}^{n+1}, \vec{R}_{w,M}^{n+1}]^{T} \tag{3.4}$$

$$\vec{R}_{well}^{n+1} = [\vec{R}_{well,1}^{n+1}, \vec{R}_{well,2}^{n+1}, \dots, \vec{R}_{well,N}^{n+1}]^T \tag{3.5}$$

$\vec{R}_{blk}^{n+1}$ consists of residuals referred to the two-phase present in the reservoir:

$$\vec{R}_{o}^{n+1} = [\vec{p}_{o}^{n+1}, \vec{S}_{w}^{n+1}, \vec{p}_{wf}^{n+1}, \vec{p}_{o}^{n}, \vec{S}_{w}^{n}, \emptyset, \delta\mathrm{t}, \vec{k}]^T \tag{3.6}$$

$$\vec{R}_{w}^{n+1} = [\vec{p}_{o}^{n+1}, \vec{S}_{w}^{n+1}, \vec{p}_{wf}^{n+1}, \vec{p}_{o}^{n}, \vec{S}_{w}^{n}, \emptyset, \delta\mathrm{t}, \vec{k}]^T \tag{3.7}$$

where $\emptyset$ is the initial porosity distribution and $\vec{k}$ is the permeability distribution. If a fixed total production rate is treated as constraint, we may have:

$$\vec{R}_{well,i}^{n+1} = \sum\nolimits_{l=o,w} \left(\sum\nolimits_{j}^{Nc} q_{l,j}^{well} - q_{t,i} \right) = \vec{0} \tag{3.8}$$

where $q_{l,j}^{well}$ is the flow rate of l-phase at the j^{th} completion given by:

$$q_{l,j}^{well} = \lambda_{l,j}^{n+1} WI_j (p_{l,j}^{n+1} - p_{wf}^{n+1} - \gamma_{l,j}^{n+1} \Delta z_j) \tag{3.9}$$

Where $\lambda_{l,j}^{n+1}$ and $\gamma_{l,j}^{n+1}$ are mobility ratio, and specific gravity of l-phase. WI_j is the well index at the j^{th} completion.

Obviously, those equations in the forward modeling are nonlinear equations. Thus, we have to use iterative methods to solve them.

3.1.3 Inverse Modeling

Inverse modeling is often used to estimate the model parameters. In the forward modeling, given the permeability distribution, the model will generate unique solution for observed data. While in the inverse modeling, the permeability distribution is unknown and must be determined. So a random initial guess has to be input in the model. Then,

model will generate different solutions at different iterations. The last solution in a gradient-based optimization is taken at the estimate of the model parameters. However, there is no unique solution to the problem. The general procedure for an inverse process can be summarized as follows:

(1) Input the initial guess for model parameters;

(2) Calculate the responses of a system from the forward modeling technique;

(3) Construct an explicit objective function;

(4) Update model parameters by minimizing the value of objective function through optimization methods;

(5) Check stopping criteria. If unacceptable, repeat step (2) to (5).

In inverse modeling, an optimization algorithm is needed to estimate the unknown system parameters. Existing popular algorithms are gradient-based algorithms and global (stochastic) algorithms. In this work, we use Levenberg-Marquardt algorithm to estimate the system parameters.

3.2 Gradient-based Optimization Algorithms

In this section, the mathematical formulations related to gradient-based optimization algorithms are presented.

3.2.1 Objective Function of Optimization Problems

Principally, optimization is the minimization or maximization of a function subject to constraints on its variables. In this work, $\vec{a}$ is the vector of variables, also called

unknowns or parameters. F is the objective function required to be maximized or minimized. $\vec{c}$ is the vector of constraints that the parameters must satisfy. Therefore, the optimization problem can be expressed as minimizing F subject to $\vec{c}$. The optimization problem can be written as

$$\min F\left(\vec{a}\right) \text{ subjct to } F(\vec{a}) = \{\begin{matrix} \vec{c}_i(\vec{a}) = 0 & i \in 1,2,\ldots,n \\ \vec{c}_j(\vec{a}) \geq 0 & j \in 1,2,\ldots,n \end{matrix} \tag{3.10}$$

Problem with the form of Equation (3.10) (Nocedal and Wright, 2006) can be classified according to the nature of the objective function and constraints (linear, nonlinear, convex), the number of parameters, the smoothness of the functions (differentiable or non-differentiable), and so on. If the optimization problems have the constraints on the variables, they are classified as the unconstrained optimization problems. These problems arise directly in my practical applications. If the optimization problems have no constraints on the variables, they are classified as the constrained optimization problems which arise from models that have explicit constraints on the variables. When the objective function and constraints are linear function of parameter, the problem is called linear programming problem. However, if the objective function or some of the constraints are nonlinear functions which are often encountered in engineering, the problem is called nonlinear programming problems.

If given the prior information which expresses specific, definite information about a parameter, the general objective function can be written as (Awotunde, 2010a)

$$F(\vec{\alpha}) = \frac{1}{2}[(\vec{d}_{cal} - \vec{d}_{obs})^T C_{D_{obs}}^{-1}(\vec{d}_{cal} - \vec{d}_{obs}) + (\vec{\alpha} - \vec{\alpha}_{prior})^T C_{D_{cal}}^{-1}(\vec{\alpha} - \vec{\alpha}_{prior})] \tag{3.11}$$

where $\vec{d}_{cal}$, $\vec{d}_{obs}$ are calculated data and observed data, respectively. $C_{D_{cal}}^{-1}$, $C_{D_{obs}}^{-1}$ are covariance matrices in model space and data space, respectively. $\vec{\alpha}_{prior}$ is the prior information about model parameter, $\vec{\alpha}$. T indicates transpose operation. If there is no prior information about the variable, the second term on the right-hand-side can be ignored and Equation (3.11) has uniform variance. Thus, Equation (3.11) can be written as

$$F(\vec{\alpha}) = \frac{1}{2}(\vec{d}_{cal} - \vec{d}_{obs})^T(\vec{d}_{cal} - \vec{d}_{obs}) \tag{3.12}$$

Another common form used to describe the process of the minimization can be expressed as

$$\min_{\vec{\alpha}}\{F(\vec{\alpha})\} = \min_{\vec{\alpha}}\{\frac{1}{2} \parallel \vec{d}_{cal} - \vec{d}_{obs} \parallel_2^2\} \tag{3.13}$$

Since the form of formulation may pose a significant influence on the performance of an algorithm, one issue in problem formulation is scaling. In unconstrained optimization, if changes to parameter, $\vec{\alpha}$, in a certain direction produces much larger variations in the objective function value than that in another direction, the problem is said to be poorly scaled. In the petroleum industry, this problem often arises in fluids flow in the porous media where different wells have very different rates. To avoid the problem of poor assimilation of some data relative to others, when these data occur at different scales, we may use a scaling factor in different objectives as follows:

$$\min_{\vec{\alpha}}\left\{F\left(\vec{\alpha}\right)\right\} = \min_{\vec{\alpha}}\left\{\frac{1}{2}\left[\frac{1}{\chi_1}\left\|\vec{d}_{cal,1} - \vec{d}_{obs,1}\right\|_2^2 + \frac{1}{\chi_2}\left\|\vec{d}_{cal,2} - \vec{d}_{obs,2}\right\|_2^2 + \ldots + \frac{1}{\chi_L}\left\|\vec{d}_{cal,L} - \vec{d}_{obs,L}\right\|_2^2\right]\right\} \tag{3.14}$$

where $\frac{1}{\chi_1}, \frac{1}{\chi_2}, \ldots, \frac{1}{\chi_L}$ are the scaling factors corresponding to L different datasets.

3.2.2 Sensitivity Coefficients

Often the gradient-based algorithms require computing sensitivity coefficients. Differentiating Equation (3.14) with respect to model parameter, α, the sensitivity coefficients are given by:

$$S = \frac{\partial \vec{d}_{cal}}{\partial \vec{\alpha}} \tag{3.15}$$

However, one concern about computation of the sensitivity coefficients in history matching is the determination of the form of the model parameters to use. If the proper form of model parameter is not used, convergence may become elusive and unacceptable. To illustrate this point, we take permeability distribution estimation for example. When we try to match production data such as pressure data and water cut data, the gradient-based optimization algorithms require determining the form of permeability before optimization starts. If natural permeability is given to the simulator, some negative values of permeability may be obtained during the optimization. As a result, simulator diverges after only several iterations. Some modelers may use pre-condition to force these negative values of permeability to be positive ones, but such behavior won't improve the result quality in nature. To remove the influence of these negative values, model

parameters can be represented by the logarithmic form (Landa, 1997; Awotunde, 2010a). Thus, the model parameters in this case, k, can be expressed by:

$$\vec{\alpha} = ln(\vec{k}) \tag{3.16}$$

Then, the Equation (3.15) becomes:

$$S = \vec{k} \cdot \frac{\partial \vec{d}_{cal}}{\partial ln(\vec{k})} \tag{3.17}$$

In practice, each sensitivity coefficient can be calculated by a substitution method which requires perturbing a control parameter, say one element in α. After all variables of control parameters in α finishes perturbation, the sensitivity coefficients are obtained. The substitution method can be described by the following equation:

$$S_{i,j} = \frac{d_{\mathrm{cal,i}}(\alpha_j + \delta\alpha_j) - d_{\mathrm{cal,i}}(\alpha_j)}{\delta\alpha_j} \tag{3.18}$$

This approach perhaps is the simplest method to calculate sensitivity coefficients. It is easy for us to implement when dealing with complex problems in reservoir simulation. But there is a shortcoming in the use of this method that substitution method may become cumbersome when the number of model parameters is very big. Assume that we discrete a reservoir into M grid blocks and we are required to estimate permeability distribution throughout the reservoir. So the number of model parameters will be equal to M. If substitution method is adopted to calculate sensitivity coefficients, it will run forward simulator M times to perturb each parameter in α. Finally, the output of sensitivity coefficient requires

the number of times at least equal to $M + 1$ to run the simulator. Besides that, we need to be concern how to choose the amount of perturbation, δα. The range of the amount of perturbation is often based on the personal choice. If an excessively large value of δα is used, the approximation of sensitivity matrix will be inaccurate while too small a value may lead to large numerical cancellation errors. To save the computation time of history matching, the adjoint method is adopted to calculate the sensitivity coefficients in this work.

3.2.3 Gradient of the Objective Function

In mathematics, the gradient is a generalization of the concept of derivative of a function in one dimension to a function in several dimensions. It is often a vector-valued function and represents the slope of the tangent of the graph of the function.

Based on the sensitivity coefficients obtained from Equation (3.17), the formula used to compute gradient, $\vec{g}$, of the objective function is given by:

$$\vec{g} = \frac{\partial F}{\partial \vec{\alpha}} = S^T(\vec{d}_{cal} - \vec{d}_{obs}) \tag{3.19}$$

3.3 Descent Algorithms

In section 3.2, the formulation of the gradient of an objective function and sensitivity coefficients are discussed. In the following subsections, we will show the formulas and implementing procedures related to gradient-based optimization algorithms.

3.3.1 Newton's Algorithm

Newton's algorithm is also known as Newton-Raphson method. It is an algorithm used for finding successively better approximations to the roots of a real-valued function.

Consider a multi-dimensional optimization problem that is expanded at second-order approximation:

$$F(\vec{\alpha}+\vec{s}) \approx F(\vec{\alpha}) + \vec{s}\vec{g} + \vec{s}^{T} H \vec{s} \tag{3.20}$$

where $\vec{s}$ is the Newton direction, T is the transpose operation. H is the exact Hessian matrix which is given by

$$H = \frac{\partial \vec{g}}{\partial \vec{\alpha}} \tag{3.21}$$

Differentiating Equation (3.20) with respect to $\vec{s}$, we have

$$\vec{g} + H\vec{s} = \vec{0} \tag{3.22}$$

At k^{th} iteration, we have

$$H_k \vec{s}_k = -\vec{g}_k \tag{3.23}$$

Solving for $\vec{s}_k$, the solution is updated by

$$\vec{\alpha}_{k+1} = \vec{\alpha}_k + \vec{s}_k \tag{3.24}$$

where $k = 0,1,2, \ldots,$ n. In this way, a sufficiently accurate solution may be obtained after a finite number of iterations.

The steps to implement Newton's algorithm include:

(1) Make a random initial guess, $\vec{\alpha}$;

(2) Calculate gradient, $\vec{g}$, and the exact Hessian matrix, H, using Equation (3.20);

(3) Calculate new step, $\vec{s}$, using Equation (3.22);

(4) Update $\vec{\alpha}$ using Equation (3.24) ;

(5) Check stopping criteria. If not acceptable, repeat (2)-(4).

3.3.2 Steepest Descent

The steepest descent is one of the few algorithms that avoid huge calculation of Hessian and storage of any matrix. The second-order approximation of Taylor series for a multi-dimensional problem is almost the same as that in Newton's method. The only difference is that the steepest decent uses an approximated Hessian matrix instead of an exact one because the exact Hessian matrix is impractical to calculate during the optimization. The approximated Hessian used by steepest descent algorithm is given by

$$H = ||\vec{g}||_2 \cdot I \tag{3.25}$$

Then, Equation (3.22) becomes

$$\vec{g} + (||\vec{g}||_2 \cdot I) \cdot \vec{s} = \vec{0} \tag{3.26}$$

at k^{th} iteration, we have

$$||\vec{g}_k||_2 \cdot I_k \cdot \vec{s}_k = -\vec{g}_k \tag{3.27}$$

Solving for $\vec{s}_k$, we have

$$\vec{s}_k = \frac{-\vec{g}_k}{||\vec{g}_k||_2} \tag{3.28}$$

Then, the solution is updated by

$$\vec{\alpha}_{k+1} = \vec{\alpha}_k + \vec{s}_k \tag{3.29}$$

In steepest descent, the search direction is the opposite direction to the gradient of the objective function. The direction of steepest ascent is the same as that of gradient. Thus, the search direction is opposite the direction of steepest ascent. Since the Hessian matrix is the definite, Equation (3.27) will always have a solution, $\vec{s}_k$. However, the convergence rate of steepest descent is very slow in many cases. The general steps of implementation of steepest descent algorithm include:

(1) Make a random initial guess, $\vec{\alpha}$;

(2) Calculate gradient, $\vec{g}$, and approximated Hessian matrix using Equation (3.20) and Equation (3.25), respectively;

(3) Calculate new step, $\vec{s}$, using Equation (3.28);

(4) Update $\vec{\alpha}$ using Equation (3.29) ;

(5) Check stopping criteria. If not acceptable, repeat (2)-(4).

3.3.3 Quasi-Newton

Quasi-Newton algorithms are a widely used optimization algorithms for nonlinear optimization. They are effective in solving inverse problems where the number of parameters is large. Although there are many different variations in quasi-Newton algorithms, they are all based on approximating the Hessian. The formula used to calculate the approximated Hessian matrix by quasi-Newton algorithm is given by

$$H = B \tag{3.30}$$

where B is the approximated Hessian matrix. Then, Equation (3.22) becomes

$$\vec{g} + B\vec{s} = \vec{0} \tag{3.31}$$

at k^{th} iteration, we have

$$B_{k+1}\vec{s}_k = \vec{y}_k \tag{3.32}$$

where $\vec{s}_k$ and $\vec{y}_k$ are given by

$$\vec{s}_{k-1} = \vec{\alpha}_k - \vec{\alpha}_{k-1} \tag{3.33}$$

$$\vec{y}_{k-1} = \vec{g}(\vec{\alpha}_k) - \vec{g}(\vec{\alpha}_{k-1}) \tag{3.34}$$

If Symmetric Rank 1 update is used, B_{k+1} is updated by

$$B_{k+1} = B_k + \frac{(\vec{y}_k - B_k\vec{s}_k)(\vec{y}_k - B_k\vec{s}_k)^T}{(\vec{y}_k - B_k\vec{s}_k)^T\vec{s}_k} \tag{3.35}$$

But in practice, there may be no Symmetric Rank 1 update that can maintain symmetry and positive definiteness of the approximated Hessian. At this time, a more widely used update formula called BFGS is given used. The update formula is:

$$B_{k+1} = B_k - \frac{(B_k\vec{s}_k)(B_k\vec{s}_k)^T}{\vec{s}^T B_k \vec{k}_k} + \frac{\vec{y}_k\vec{y}_k^T}{\vec{y}_k^T\vec{s}_k} \tag{3.36}$$

Based on the above equations, we solve for $\vec{s}_k$ and then update $\vec{\alpha}_k$. The steps to implement quasi-Newton algorithm include:

(1) Make a random initial guess, $\vec{\alpha}$, and Hessian approximation (i.e. $B_0 = I$);

(2) Calculate gradient, $\vec{g}$, using Equation (3.20);

(3) Calculate new search direction $\vec{s}$ using Equation (3.32);

(4) Update $\vec{\alpha}$ and the approximated Hessian, B, using Equation (3.35) or (3.36);

(5) Check stopping criteria. If not acceptable, repeat (2)-(4).

3.3.4 Gauss-Newton

In Newton's method, the computation of the exact Hessian may become expensive and impractical. For some small and medium-scale problems, Gauss-Newton algorithm may give us fast convergence with the use of first-order derivative to calculate the Hessian matrix. The formula used for approximating Hessian is given by:

$$H_{GN}(\vec{\alpha}) = S^T S \tag{3.37}$$

Then, the new search direction for Gauss-Newton algorithm is computed by

$$\vec{g} + H_{GN}\vec{s} = \vec{0} \tag{3.38}$$

Thus, at k^{th} iteration, the solution is updated by

$$\vec{\alpha}_{k+1} = \vec{\alpha}_k + \vec{s}_k \tag{3.39}$$

When the Hessian of Gauss-Newton algorithm is positive-definite, the algorithm will give a quadratic convergence in the neighborhood of true solution. The general procedure to implement Gauss-Newton algorithm include:

(1) Make a random initial guess, $\vec{\alpha}$;

(2) Calculate gradient, $\vec{g}$, and the Hessian, H_{GN}, using Equation (3.20) and Equation (3.37);

(3) Calculate new step, $\vec{s}$, using Equation (3.38);

(4) Update $\vec{\alpha}$ using Equation (3.39);

(5) Check stopping criteria. If not acceptable, repeat (2)-(4).

3.3.5 Levenberg-Marquardt

For the Hessian of Gauss-Newton algorithm, there is no guarantee that the Hessian is positive-definite all the time. Once the starting points are out of the region of the true solutions, the Hessian obtained by Gauss-Newton algorithm will be not symmetric and positive-definite. As a result, Gauss-Newton algorithm diverges even at the first iteration and yields inaccurate results.

According to literature review (Levenberg 1944; Marquardt 1963), we know that the diagonal elements of the Hessian have the abilities to prevent Hessian to be singular. Thus, the Levenberg-Marquardt algorithm is developed to ensure that the Hessian is the definite. In Levenberg-Marquardt algorithm, the Gauss-Newton Hessian matrix is prevented to be singular by adding a diagonal matrix which is given by:

$$H_{LM}(\vec{\alpha}) = H_{GN}(\vec{\alpha}) + \lambda \cdot I \tag{3.40}$$

where H_{GN} is the Hessian obtained from Gauss-Newton algorithm. λ is the damping factor. Thus, H_{LM} is the Hessian obtained from the perturbation of the main diagonal elements of H_{GN}.

The new step at k^{th} iteration is calculated by:

$$\vec{g}_k + H_{LM}\vec{s}_k = \vec{0} \tag{3.41}$$

And the new search step is updated by:

$$\vec{\alpha}_{k+1} = \vec{\alpha}_k + \vec{s}_k \tag{3.42}$$

Equation (3.40) ensures the Hessian obtained from the LM algorithm is positive-definite. But we need to ensure that λ is neither too small nor too big. Often modelers prefer to use trial-and-error approach to update damping factor. The general procedures to implement LM algorithm include:

(1) Make a random initial guess, $\vec{\alpha}$;

(2) Calculate gradient, $\vec{g}$, using Equation (3.20);

(3) Calculate new step, $\vec{s}$, using Equation (3.41);

(4) Update H_{LM} and $\vec{\alpha}$ using Equation (3.40) and Equation (3.42);

(5) Check stop criteria. If not acceptable, repeat (2)-(4).

3.4 Line Search Approach

During the implementation of the above gradient-based algorithms, all of them need to choose a direction and search along this direction. Then, the initial guess will iterate to a lower value after a certain number of iterations. But the distance to move along new direction can be either large or small. As a result, all of gradient-based optimization algorithms mentioned above can not determine an optimal steplength for new search direction. To navigate the steplength, line search approach is often coupled with gradient-based optimization algorithms. The formula accounting for line search approach is

$$\vec{\alpha}_{k+1} = \vec{\alpha}_k + \nu_k \vec{s}_k \tag{3.43}$$

where ν_k is larger than zero and it satisfies the objective function given by:

$$F(\vec{\alpha}_k + \nu_k \vec{s}_k) < F(\vec{\alpha}_k) \tag{3.44}$$

The condition given in Equation (3.44) is the basic backtracking line search. However, this condition is a weak condition. It may fail because the steps may become too small or too big relative to the amount of decrease in objective function that they provided. Thus, a stricter condition called the Armijo-Goldstein to backtracking line search. The formula is given by:

$$F\left(\vec{\alpha}_k + \nu_k \vec{s}_k\right) < F\left(\vec{\alpha}_k\right) + \varpi \cdot \nu_k \langle \vec{g}_k, \vec{s}_k \rangle \tag{3.45}$$

where ϖ ranges from zero to one. When backtracking line search fails, Equation (3.45) is often used to give sufficient decrease.

3.5 Global Optimization Algorithm

To solve optimization problems that were previously difficult to solve, several global optimization algorithms have been developed in the past decades. This section will briefly summarize the global optimization algorithms and introduce the fundamentals of Differential Evolution (DE). Furthermore, the theoretical formulations of DE are presented.

3.5.1 Introduction of Global Optimization Algorithms

Global optimization algorithms have been proposed and developed for several decades. They are: evolutionary computation, genetic algorithms, evolution strategies and evolutionary programming, different evolution, particle swarm optimization, ant colony search algorithm, tabu search, simulated annealing, and so forth.

Evolutionary computation simulates a hypothetical population-based optimization process on a computer, which is a stochastic optimization technique that can perform

better than classic gradient-based optimization algorithms. Genetic algorithm is a search algorithm based on the conjecture of natural selection. It needs to evaluate fitness function or objective function to guide its search in parallel with a coding of parameters instead of parameters themselves. Evolution strategies (ES) share many features with the genetic algorithms (GAs). The differences are ES operates on floating-point vectors whereas GAs use binary strings; ES relies on mutation as the search operator whereas GAs mainly rely on recombination to search the space. Evolutionary programming is similar to a GA that emphasizes behavioral linkage between parents and offspring, rather than seeking to emulate specific operators. DE is a stochastic direct search method and it is considered as an accurate, reasonably fast, and robust optimization algorithm. Particle swarm optimization (PSO) is an exciting new algorithm in evolutionary computation which is somewhat similar to GAs. The ant colony search algorithms mimic the behavior of real ants which have the abilities to find the shortest path from food sources to the nest without using visual cues. Tabu search is basically a gradient-descent search with memory that preserves a number of previously visited states. Simulated annealing simulates relaxing the system to a state with minimum free energy.

Recently, applications of each of global optimization algorithms have been reported in many works. They are either combined among themselves or with traditional approaches to solve challenging problems. In this work, DE is chosen for the estimation of λ and we present a description of this algorithm in the following subsections.

3.5.2 DE Fundamentals

DE is a type of evolutionary algorithms. It is efficient at solving a wide range of the optimization problems.

DE uses a population of candidate solutions, also known as agents, to search for the optimum solution. Often the agents in the population are generated initially from a uniform distribution. The formula accounting for initializing population is

$$\vec{\alpha}_i^G = \vec{\alpha}_{i,low} + rand_i[0,1] \cdot (\vec{\alpha}_{i,high} - \vec{\alpha}_{i,low}) \tag{3.46}$$

Where $\vec{\alpha}_{i,low}$ and $\vec{\alpha}_{i,high}$ are the lower and upper bounds of d-dimensional vector. If a priori information is available, the preliminary solution can be integrated to the initial population by adding distributed random deviations to the nominal solution.

To find out the optimum solution, each agent needs to repeatedly undergo mutation, crossover, and selection. This procedure will be repeated until a preset stopping criterion is met. Finally, the agent that yields the best fitness or objective function value will be saved as the optimum solution.

3.5.3 Key Operators for DE

This subsection summarizes mathematical formulations of key operators of DE, including mutation, crossover, and selection.

(1) Mutation – In DE, the agents in the population are generated by Equation (3.46). The best agent in the population is determined by evaluating the fitness values or the objective function values of the member of the population. It is stored externally as the best solution found, $\vec{\alpha}_{best}^G$, up to the current generation, G, and it is updated when a better agent is found in subsequent generations. Mutation involves generate a mutant vector, $\vec{\alpha}_i^{G+1}$, for each candidate solution using the formula (Awotunde, 2014):

DE/rand/2: $\vec{\alpha}_i^{G+1} = \vec{\alpha}_{r1}^{G} + M_F(\vec{\alpha}_{r2}^{G} - \vec{\alpha}_{r3}^{G}) + M_F(\vec{\alpha}_{r4}^{G} - \vec{\alpha}_{r5}^{G})$ (3.47)

DE/best/1: $\vec{\alpha}_i^{G+1} = \vec{\alpha}_{best}^{G} + M_F(\vec{\alpha}_{r1}^{G} - \vec{\alpha}_{r2}^{G})$ (3.48)

DE/best/2: $\vec{\alpha}_i^{G+1} = \vec{\alpha}_{best}^{G} + M_F(\vec{\alpha}_{r1}^{G} - \vec{\alpha}_{r2}^{G}) + M_F(\vec{\alpha}_{r3}^{G} - \vec{\alpha}_{r4}^{G})$ (3.49)

DE/rand-to-best: $\vec{\alpha}_i^{G+1} = \vec{\alpha}_{r1}^{G} + M_F(\vec{\alpha}_{best}^{G} - \vec{\alpha}_{r2}^{G}) + M_F(\vec{\alpha}_{r3}^{G} - \vec{\alpha}_{r4}^{G})$ (3.50)

Where $\vec{\alpha}_{r_1}^{G}, \vec{\alpha}_{r_2}^{G}, \vec{\alpha}_{r_3}^{G}, \vec{\alpha}_{r_4}^{G}$, and $\vec{\alpha}_{r_5}^{G}$ are different agents. They are all different from $\vec{\alpha}_i^{G+1}$, selected randomly from the population in the current generation. $M_F \in [0,2]$ is the mutation factor.

(2) Crossover – It is performed to generate a child vector, $\vec{\tau}_i^{G+1}$, from each parent vector, $\vec{\alpha}_i^{G}$, in the current generation so that the search diversity is enhanced. The crossover constant, $\mathrm{CR} \in [0,1]$, is chosen by the user and a random number between zero and one is drawn from the uniform distribution for every coordinate of a trial vector. The formula accounting for crossover operation is:

$$\vec{\tau}_i^{G+1} = \vec{\alpha}_i^{G} + CR \cdot (\vec{\alpha}_{r1}^{G} - \vec{\alpha}_i^{G}) + M_F \cdot (\vec{\alpha}_{r2}^{G} - \vec{\alpha}_{r3}^{G}) \quad (3.51)$$

In practice, if the random number drawn is less than CR, the value of the respective coordinate is taken from the mutant vector. Otherwise, the value is taken from the parent vector.

(3) Selection – This is the final step of implementing DE, involving the selection of the better vector between the parent vector, $\vec{\alpha}_i^{G}$, and child vector, $\vec{\tau}_i^{G+1}$. If the fitness or the objective function value of the offspring is better than that of the parent, the parent

will be replaced with the offspring so that a new generation of population is formed. The procedure will be repeated until the stopping criterion is met.

CHAPTER 4

OPTIMIZATION ALGROITHMS FOR INVERSE PROBLEMS

In this Chapter, we will provide our optimization algorithms that combine LM and DE. Meanwhile, the details of implementing those algorithms are presented so that the readers can know how the algorithms are used for solving history matching problems.

4.1 LM+LSCH Algorithm

History matching problems are often inverse problems in reservoir simulation. To estimate reservoir parameter efficiently with certain reliability, choice of the optimization algorithms is very important for engineers. The Levenberg-Marquardt (LM) algorithm is often used in conjunction with line search (LSCH). While LM computes the downhill to take in a minimization procedure; LSCH algorithm finds an optimum step in this direction, making the new estimate better than the one that would have been obtained in many cases where LSCH was absent. Thus, the combination of these two algorithms (LM+LSCH) constitutes an efficient tool for parameter estimation. Both algorithms are iterative, and the LSCH algorithm is run inside the every iteration of the LM. In this work, we first choose LM1+LSCH and LM2+LSCH as optimization algorithms. LM1 indicates traditional LM method in which the damping factor is divided by 4 if the residual decreases but multiplied by 4 if the residual increases. Similarly, LM2 indicates LM method in which the damping factor is divided or multiplied by 10.

Take permeability estimation for instance, LM starts with an initial guess of the permeability field and calculates the objective function from the initial guess. The starting point of damping factor can be chosen as a random positive value. LM first calculates gradient and sensitivity matrix through adjoint method. The sensitivity matrix, S, is used to compute the Gauss-Newton Hessian matrix by $S^T S$. Then, the initial guess of damping factor is used to perturb Gauss-Newton Hessian matrix. LM calculates new step and LSCH is used to determine a suitable length for it. Note that the number of iterations to run LSCH should not be large. After updating initial guess of permeability field successfully, LM evaluates the objective function again. If the objective function changes, LM will update damping factor by trial-and-error either dividing by a constant (4 or 10) if the residual decreased or multiplying by the constant if the residual increased. The new estimate allows LM to update gradient and Hessian, optimization process goes to next iteration.

The procedure to implement the LM+LSCH algorithm is given by:

1. An initial damping factor is chosen as a random positive constant;
2. LM begins with an initial guess of the reservoir parameter field at the first iteration and calculates the objective function.
3. LM uses the current iterate to calculate gradient and sensitivity coefficients, respectively. Furthermore, LM uses sensitivity coefficients to calculate Gauss-Newton Hessian matrix. Then, LM perturbs Hessian using the current value of the damping factor and calculates an approximate Newton direction;

4. Line search approach is used to determine steplength and update the current iterate;

5. LM uses new iterate to calculate residual and compare it to the one from Step 2. If residual decreases, we update initial damping factor by trial-and-error approach.

6. LM repeats Step 3 to Step 5 until optimization completes.

4.2 LM+DE Algorithm

For estimation of damping factor, trial-and-error approach often requires an initial guess before optimization starts while DE requires upper and lower bounds to do so. To integrate DE with LM, LSCH must be replaced by DE and the bounds should be provided in advance. In the LM+DE algorithm, LM calculates the gradient and Gauss-Newton Hessian matrix using the most recent estimate of model parameters. DE then performs a global search for the optimum value of damping factor to be used at the current iteration. The damping factor obtained from DE is used to perturb the Gauss-Newton Hessian matrix and an approximate Newton direction is evaluated for the current iteration. Because the damping factor in this case is expected to be optimal, no steplength is needed to be evaluated as a good value of λ. Thus, this automatically yields a Newton direction that has an optimal steplength.

The general procedures to implement LM+DE algorithm include:

1. LM begins with an initial guess of the reservoir parameter field;

2. LM uses the current iterate to calculate gradient and sensitivity coefficients, respectively. Furthermore, LM uses sensitivity coefficients to calculate Gauss-Newton Hessian matrix;

3. DE searches for an optimal value of damping factor

4. LM repeats Steps 2 and 3.

4.3 LM+DE+LSCH Algorithm

To check that if DE has the ability to find an optimum damping factor at each of LM iteration without using LSCH, we use LM+DE+LSCH to solve the same history matching problems again. The general procedure of implementing LM+DE+LSCH is the same as LM+DE while inserting an extra step between Step 3 and Step 4. That is using LSCH to determine the suitable steplength for new search direction.

CHAPTER 5

APPLICATION TO SYNTHETIC RESERVOIRS

5.1 Feasibility Analysis for Optimization Algorithms

LM+LSCH is a gradient-based optimization algorithm for solving inverse problems. While DE is a simple global optimization algorithm developed for multidimensional real-valued functions but doesn't use gradient of the problem being optimized. The free computation of gradient indicates that DE can deal with optimization problems that are not differentiable. Normally, it is difficult for engineers to use analytical method to calculate values of damping factor and then apply them to LM+LSCH to complete the history matching process because of the huge cost of computation. To avoid this, most users of the LM algorithm use an empirical method, trial-and-error approach, to estimate the damping factor (Watson and Lee 1986; Grattoni and Bidner 1990; Zhang et al. 2003; Awotunde and Horne, 2008; Awotunde and Horne, 2010; Awotunde and Horne, 2011; Awotunde and Horne, 2012a; Awotunde, 2014). However, for high-dimensional problems, such method becomes less effective.

Here, we propose using DE, a global search method, to estimate the damping factor for LM because DE may have the potential to find out optimum damping factor so as to avoid uncertainty in estimating damping factor. In addition, other advantages of using DE instead of trial-and-error approach include: first, trial-and-error approach doesn't have any bounds for damping factor. It can only estimate damping factor by providing an appropriate value in advance. If the initial guess of damping factor is far

away from an appropriate value, LM+LSCH may fail even at first iteration. But DE can avoid such problem in way of generating upper and lower bounds for damping factor. Second, if DE is able to estimate optimum damping factor for LM, it may have the potential to eliminate the necessity to determine the length of new step. So DE can replace LSCH approach and avoid extra computation cost of the gradient, especially where an adjoint formulation for gradient is not available. Third, since optimum damping factor is more reliable than that estimated by trial-and-error approach, DE should enable LM to generate much smaller residuals of objective function within limited iterations compared to that of LM+LSCH. Thus, the simulator should converge rapidly and save us a lot of time. Finally, because the DE uses a population of candidate solutions, it parallelizes naturally, thus ensures that only a small time is spent on estimating the damping factor.

To validate our idea, we provide three reservoir models of different sizes: 16×16, 32×32 and 64×64 to test those algorithms and correlations. The requirement is to estimate permeability field based on matching three different cases including pressure data, water cut data, and combined data (pressure and water cut together). For convenience, we label these cases as Case 1, Case 2, and Case 3. Once matching is done, we analyze accuracy and efficiency of matched results obtained by different algorithms.

Synthetic noisy data are used as production for inverse analysis will be calculated by running a forward simulator. Noise is the synthesized data by (Awotunde, 2014):

$$d_{obs} = d_{meas} + d_{meas} \times (\sqrt{NSR} \times randn) \tag{5.1}$$

where, NSR is noise-signal-ratio (10^{-6} for pressure data and 10^{-5} for water cut data). d_{meas} is the synthetic data without noise.

Besides, the relative closeness of the estimate from the initial guess to final estimation is measured by the use of l_1-norm of the difference between calculated log permeability field and true log permeability field given by:

$$k_{norm} = \frac{\|\log(\vec{k}_{\mathrm{cal}}) - \log(\vec{k}_{\mathrm{obs}})\|_1}{L_k} \tag{5.2}$$

where, k_{norm} is the relative closeness. $\vec{k}_{cal}$, and $\vec{k}_{obs}$ are the calculated permeability field and the true permeability field, respectively. L_k is the number of unknown permeability values required to be estimated.

5.1.1 Example 1

A synthetic reservoir of size 16×16 was used for the test. Table 5-1, Table 5-2, and Table 5-3 present the details of discretization, well information, and fluid properties. Figure 5-1 shows that the reservoir is heterogeneous with fully distributed permeability field. It also shows three producers and two injectors over the entire reservoir. All wells were put on operation simultaneously after simulation starts. The data required to be matched are pressure measured at all wells and water cut at producers. Totally, there are 8 datasets. Three bottomhole pressure datasets (BHP) and three water cut datasets from producers while two BHP datasets from injectors. Each dataset has 256 data points. Total number of data points required to be matched is 2048. In addition, LM is made up to 10 iterations while the maximum number of iterations used to run line search is 5. As for DE, we set up suitable upper and lower bounds for candidates of damping factors. Then, the

iterations to run DE are made up to 5 at each of LM iteration. The function evaluations in each of the iteration in DE were done in parallel, saving significant time.

Table 5-4 shows the details of the implementation of four different optimization algorithms including LM1+LSCH, LM2+LSCH, LM+DE, and LM+DE+LSCH. Values of residual of the objective function are shown in Fig. 5-2a, Fig. 5-2b, and Fig. 5-2c. From those figures, we observe that the proposed algorithm (LM+DE) performed much better than LM1+LSCH and LM2+LSCH in terms of efficiency and error reduction. Firstly, when the proposed algorithm is used, DE eliminates the uncertainty on estimating damping factor. For example, at each of LM iterations, DE requires user-provided upper and lower bounds of damping factor, which gives us more freedom to choose a good initial damping factor. Once DE finds the optimum damping factor, optimization rapidly generates the best search step in current iteration and this step guarantees the convergence of algorithm at the beginning of optimization. However, in the trial-and-error approach, the initial value of damping factor is given by a random real positive constant. If such value is too large or too small, LM may diverge even at the first iteration. Even if LM converges, it may require some iterations to perturb the values of damping factor due to the uncertainty. Thus, we may observe a phenomenon that the decay of residual perturbs up and down. Secondly, checking Fig. 5-2a, Fig. 5-2b, and Fig. 5-2c, we observe that the curves of decay of residual using LM+DE and LM+DE+LSCH are perfectly overlapped. This shows DE can eliminate the use of line search since optimum damping factor enables LM to determine the Newton direction and suitable steplength simultaneously. Without line search, LM is able to avoid extra computation of gradient, thus making the algorithm be more computationally efficient. However, if using

trial-and-error approach, LM must use line search to calculate the gradient and furthermore determine the suitable steplength, otherwise algorithm may diverge due to the inappropriate steplength. Thirdly, if setting a certain value as optimization stopping criteria, we observe that LM+DE only need to run 4 iterations while LM1+LSCH and LM2+LSCH have to run up to 8 iterations to meet the tolerance. Since the computation time required for implementing LM algorithm largely depends on the calculation of sensitivity matrix and gradient, LM1+LSCH and LM2+LSCH may become more time consuming compared to LM+DE. To achieve the same value of the residual of the objective function, the proposed algorithm definitely performs much more efficiently than the standard LM algorithms within limited number of iterations. Table 5-5 shows the details of running algorithms at a certain number of iterations. In addition, comparing the values of residuals shown in Fig. 5-2a, Fig. 5-2b, and Fig. 5-2c, we observe that when DE is used, LM yields the smallest value of residual. The trends of the damping factor are presented in Fig. 5-2d to 5-2f. As expected from the trial-and-error method, we observe generally decreasing trends from LM1+LSCH and LM2+LSCH. However, the trends of λ from LM+DE and LM+DE+LSCH are not generally decreasing. This shows that λ does not have to always decrease to yield lower values of the residual.

Finally, Figure 5-3 shows the measured data obtained from different algorithms. Note that the dotted lines represent observed data while solid lines are calculated data. We observe that all algorithms are able to generate good matches. This fact shows that LM+DE is reliable in dealing with small-scale history matching problem. Furthermore, integrating DE with LM is shown to be a good combination of global search and local search. Figure 5-4 shows the estimated permeability fields obtained by different

algorithms. However, none of the estimated permeability field is close to the true field. This may be caused by lacking of enough information content of data.

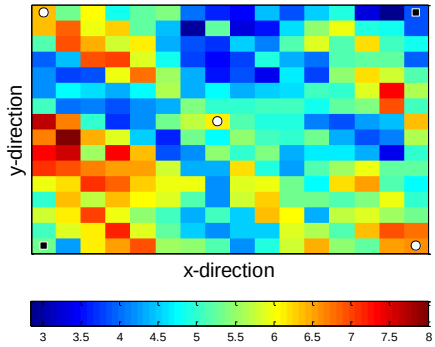
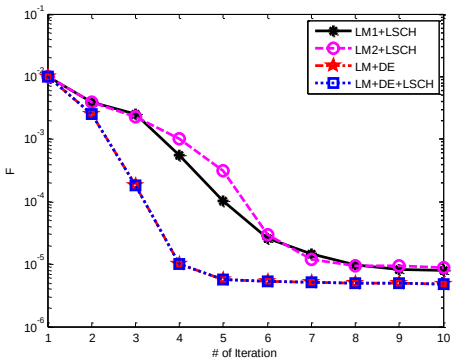
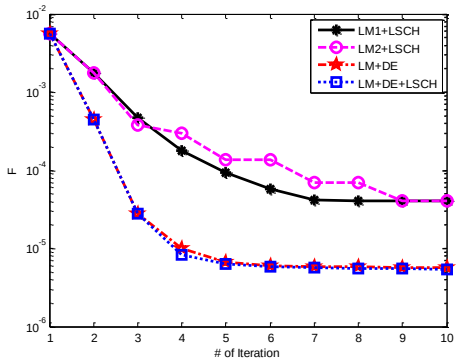


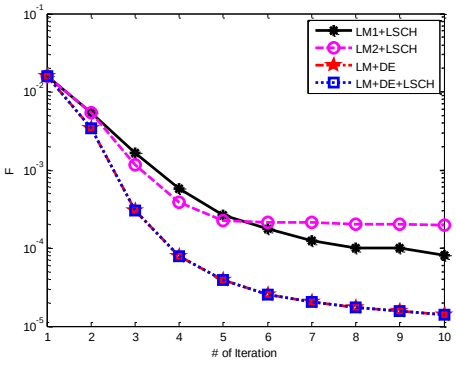
Figure 5-1 True permeability field and well locations: squares indicate injectors and circles indicate producers (Example 1)



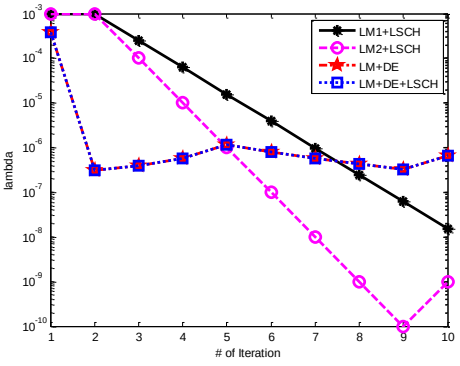
(a) decay of residual of Case 1



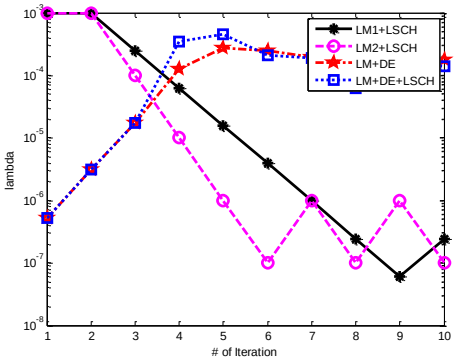
(b) decay of residual of Case 2



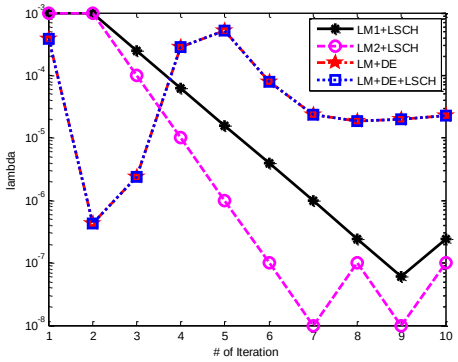
(c) decay of residual of Case 3



(d) values of lambda of Case 1

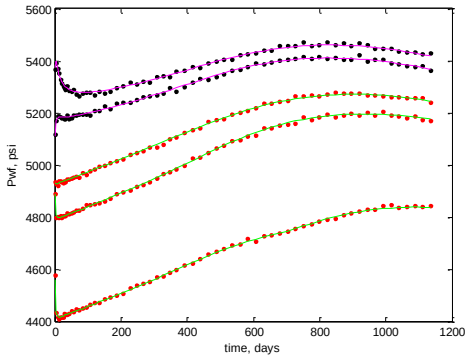


(e) values of lambda of Case 2

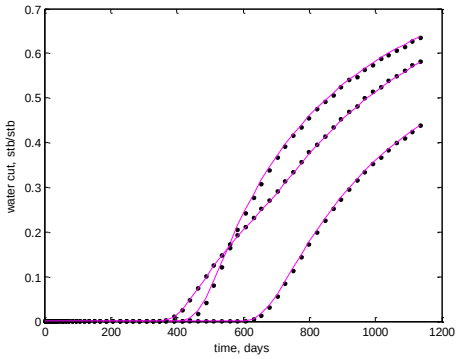


(f) values of lambda of Case 3

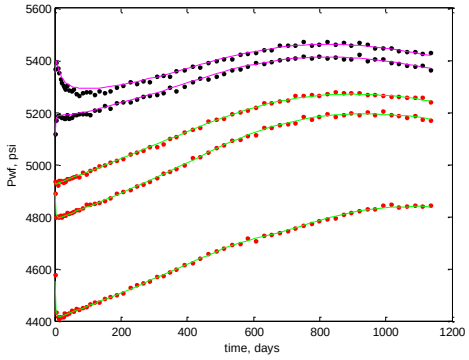
Figure 5-2 Decay of residual and values of damping factor (Example 1)



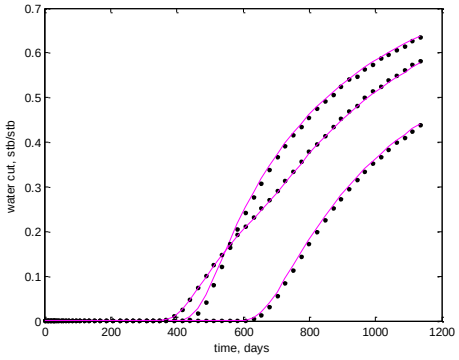
(a) match to pressure data using LM1+LSCH



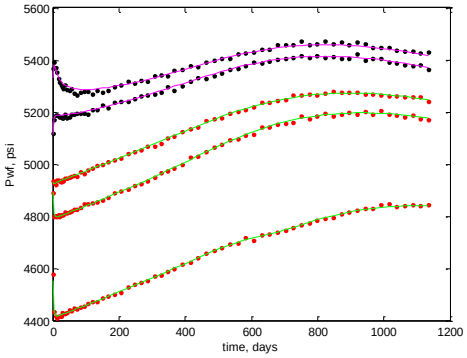
(b) match to water cut data using LM1+LSCH



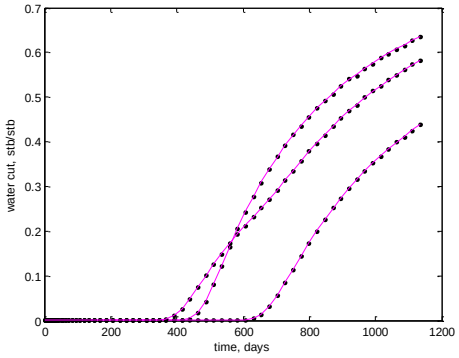
(c) match to pressure data using LM2+LSCH



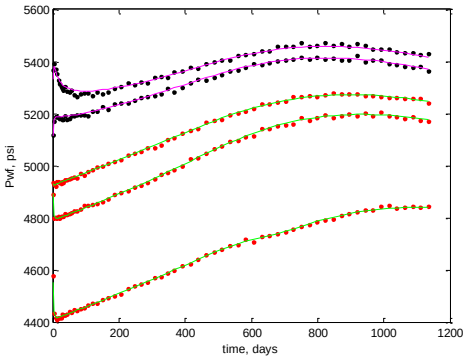
(d) match to water cut data using LM2+LSCH



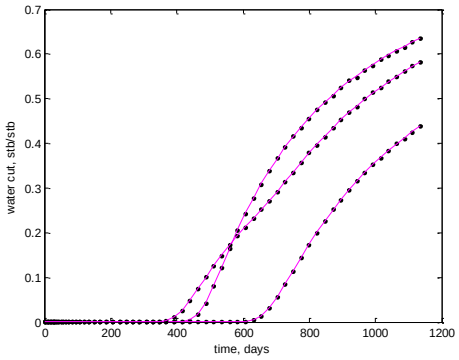
(e) match to pressure data using LM+DE



(f) match to water cut data using LM+DE

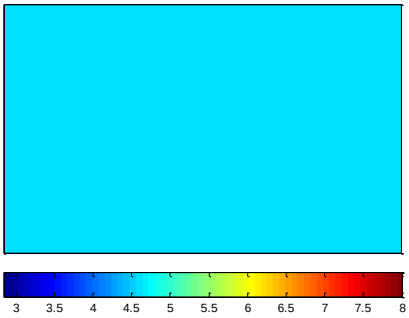


(g) match to pressure data using LM+DE+LSCH

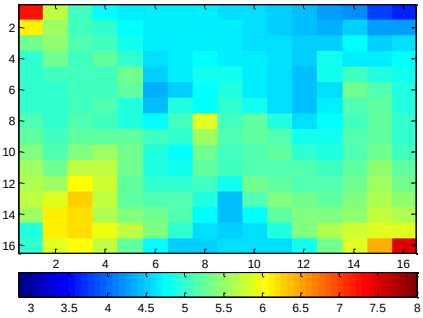


(h) match to water cut data using LM+DE+LSCH

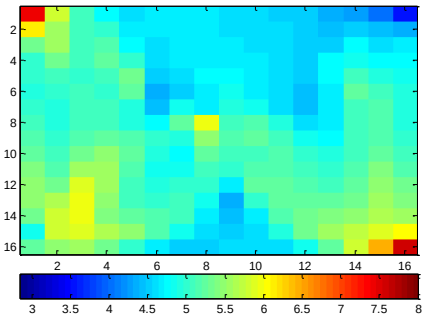
Figure 5-3 Use of different algorithms to match data (Example 1)



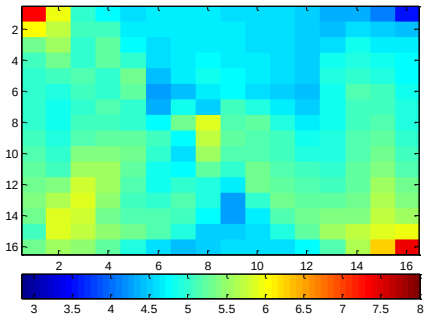
(a) initial guess



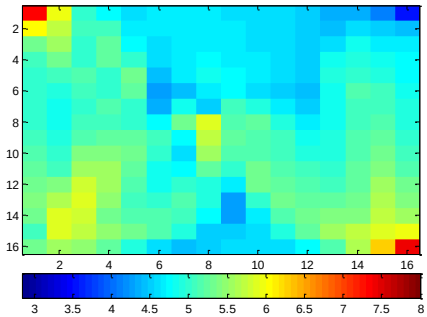
(b) estimate from LM1+LSCH



(c) estimate from LM2+LSCH



(d) estimate from LM+DE



(e) estimate from LM+DE+LSCH

Figure 5-4 Estimated permeability field using different algorithms (Example 1)

Table 5-1 Discretization of Example 1

x-axis	y-axis	z-axis
16	16	1
Lx	Ly	Lz
800	800	100

Table 5-2 Well information of Example 1

well index	x-coordinate	y-coordinate	z-coordinate	flow rate	well type
1	8	8	1	500	producer
2	16	16	1	-550	injector
3	1	1	1	600	producer
4	16	1	1	350	producer
5	1	16	1	-850	injector

Table 5-3 Fluid properties of Example 1

fluid properties	value
initial density of oil, lbm/ft^3	40
contant used to compute oil vicosity	1.20E-05
isothermal compressibity of oil	2.00E-06
initial water density,lbm/ft^3	62.238
isothermal compressibity of water	5.00E-07
contant used to compute water viscosity	6.00E-08
initial pressure, psi	5000
initial water saturation	1.00E-06
well radius, ft	0.25
skin factor	0
initial porosity	0.25

Table 5-4 Details of implementing optimization algorithms (Example 1)

Case #	algorithm	fevals/iter	# of LM iterations	total fevals	initial k_{norm}	k_{norm} of estimates	time(sec)
Case 1	LM1 + line search	4.6/iter	10	46	0.9565	0.7077	1390
Case 2	LM1 + line search	3.3/iter	10	33	0.9565	0.9542	1112
Case 3	LM1 + line search	2.7/iter	10	27	0.9565	0.6945	984
Case 1	LM2 + line search	4.3/iter	10	43	0.9565	0.6843	1325
Case 2	LM2 + line search	3.3/iter	10	33	0.9565	0.954	1104
Case 3	LM2 + line search	3.9/iter	10	39	0.9565	0.7275	1292
Case 1	LM + DE	6.1/iter	10	61	0.9565	0.7366	1639
Case 2	LM + DE	6.1/iter	10	61	0.9565	0.9435	1642
Case 3	LM + DE	6.1/iter	10	61	0.9565	0.7102	1643
Case 1	LM + DE + line search	9.5/iter	10	95	0.9565	0.7366	2327
Case 2	LM + DE + line search	9.2/iter	10	92	0.9565	0.9435	2229
Case 3	LM + DE + line search	7.3/iter	10	73	0.9565	0.7102	1908

Table 5-5 Running optimization at the designated iteration (Example 1)

Case #	algorithm	# of LM iteration	total fevals	initial k_{norm}	k_{norm} of estimates	time(sec)
Case 1	LM1 + line search	8	36	0.9565	0.7142	1207
Case 2	LM1 + line search	8	26	0.9565	0.9538	987
Case 3	LM1 + line search	8	20	0.9565	0.7189	879
Case 1	LM2 + line search	8	33	0.9565	0.6735	1096
Case 2	LM2 + line search	8	26	0.9565	0.9533	918
Case 3	LM2 + line search	8	29	0.9565	0.7349	919
Case 1	LM + DE	4	25	0.9565	0.7128	688
Case 2	LM + DE	4	25	0.9565	0.9428	684
Case 3	LM + DE	4	25	0.9565	0.7384	687
Case 1	LM + DE + line search	4	35	0.9565	0.7128	895
Case 2	LM + DE + line search	4	32	0.9565	0.9429	832
Case 3	LM + DE + line search	4	29	0.9565	0.7384	773

5.1.2 Example 2

Table 5-6, Table 5-7, and Table 5-8 shows the discretization, well information, and fluid properties, respectively. Figure 5-5 shows the wells location in the reservoir and the true permeability field. There are 6 producers and 4 injectors. The measured data comprise of BHP datasets from all wells and water cut datasets from producers. There are 16 datasets in total. These are 10 bottomhole pressure datasets from producers and injectors and 6 water cut datasets from producers. Each of dataset has 256 data points. Thus, the total number of data points required to be matched is 4096. In addition, LM is made up to run 10 iterations and line search approach is made up to a maximum of 5. If DE is used at each of LM iteration, the number of iterations is 5. The line search has the ability to exit the iteration early (before reaching 5 iterations) if a good steplenth is found during the first few iterations.

Table 5-9 presents the details of implementation of four different optimization algorithms. Fig. 5-6a shows the decay of residual for the match of Case 1. Under the condition that we provide an appropriate initial damping factor, LM1+LSCH and LM2+LSCH decrease the value of residual continuously but not rapidly compared to the proposed algorithm. However, if an appropriate value of initial damping factor is not selected, both LM1+LSCH and LM2+LSCH will diverge. Since DE requires the bounds of damping factor, the proposed algorithm will always guarantee the convergence of optimization. Similar phenomenon is also observed for Case 2 and Case 3.

Figure 5-6a, Fig. 5-6b, and Fig. 5-6c reveal that the decay of residuals from LM+DE and that from LM+DE+LSCH are almost in perfect agreement. This fact shows that DE has the ability to estimate the optimum damping factor which enables LM to

navigate suitable steplength without the use of line search. In addition, if we set a certain value of residual as tolerance, LM+DE, within iterations, decreases rapidly to reach such value while LM1+LSCH and LM2+LSCH have to run up to iterations. Table 5-10 is used to illustrate the performance of four different optimization algorithms at truncated iterations. With the use of parallel computation, the proposed algorithm (LM+DE), saves a lot of time for the match of Case 1 and Case 3 compared to LM1+LSCH and LM2+LSCH. For the Case 2 accounting for the match of water cut data, the proposed algorithm seems to work almost the same as that of the standard LM algorithm. Besides, the last value of residual in Fig. 5-6a shows that the proposed algorithm performs much better than the standard LM algorithm because the residual obtained by LM+DE is much more smaller than that of LM1+LSCH and LM2+LSCH. In Fig. 5-6b and Fig. 5-6c, we observe that the last value of the residual obtained by the proposed algorithm is nearly the same as that of LM1+LSCH when matching water cut data and combined data.

Checking Fig. 5-6d, Fig. 5-6e, and Fig. 5-6f, we observe that the values of damping factor obtained by LM+DE and LM+DE+LSCH do not always decrease. Thus, we may conclude that values of damping factor do not always decrease even if we have a good decline trend of residuals of objective function. Thus, the proposed algorithm is more efficient and gives lower error. Finally, the matched results are shown on Fig. 5-7 in which all subfigures generate good matches to data. Figure 5-8 shows the estimated permeability field after optimization. None of the estimated permeability field is close to the true permeability field. This goes to show that the information content of the data matched is not enough to resolve the parameter of the system independently.

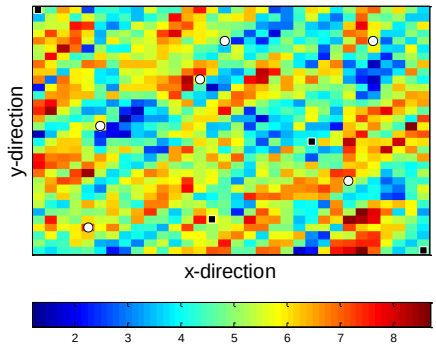
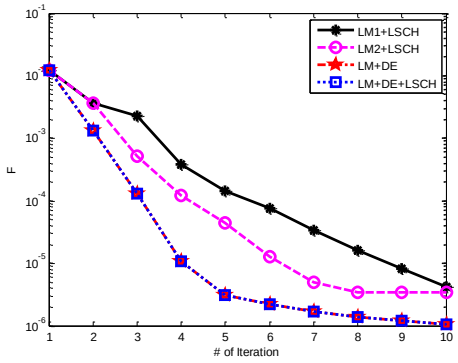
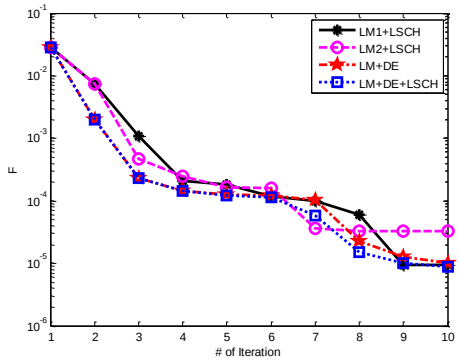


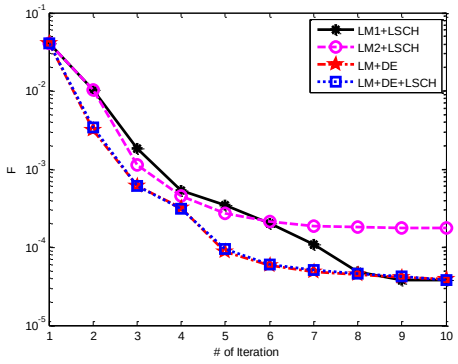
Figure 5-5 Permeability field and well locations: squares indicate injectors and circles indicate producers (Example 2)



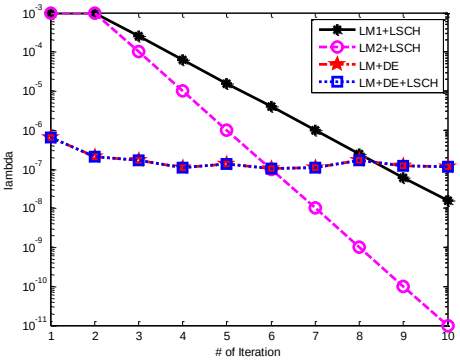
(a) decay of residual of Case 1



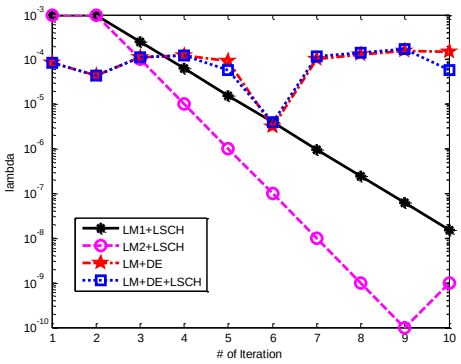
(b) decay of residual of Case 2



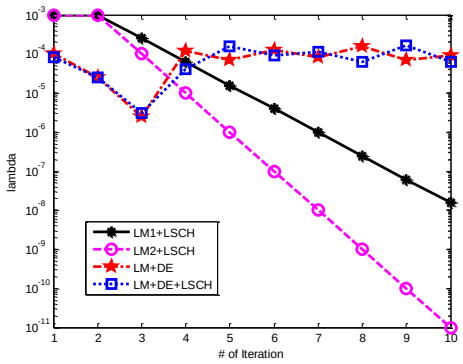
(c) decay of residual of Case 3



(d) values of lambda of Case 1

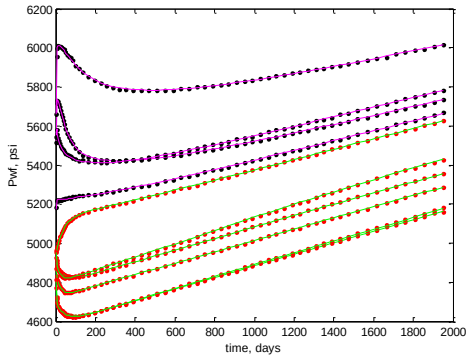


(e) values of lambda of Case 2

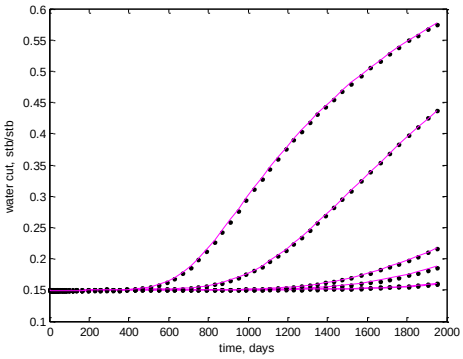


(f) values of lambda of Case 3

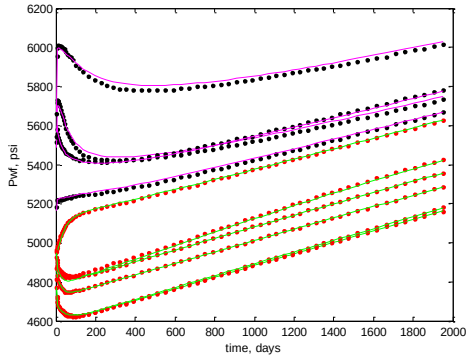
Figure 5-6 Decay of residual and values of damping factor (Example 2)



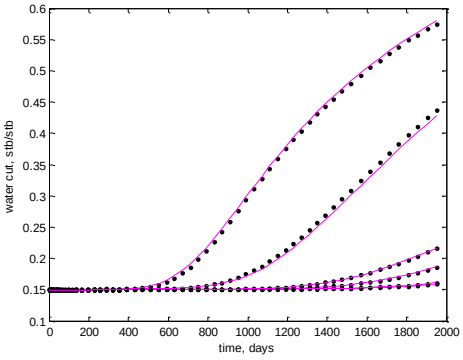
(a) match to pressure data using LM1+LSCH



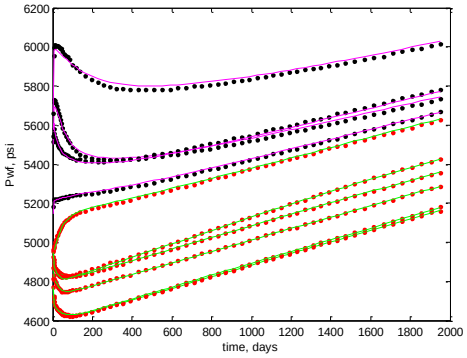
(b) match to water cut data using LM1+LSCH



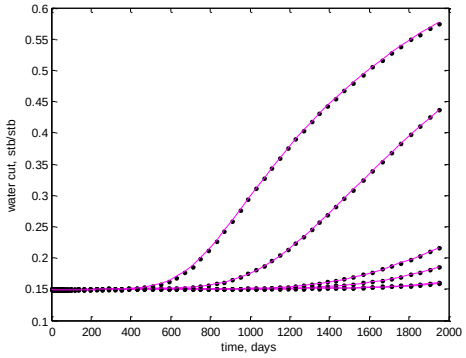
(c) match to pressure data using LM2+LSCH



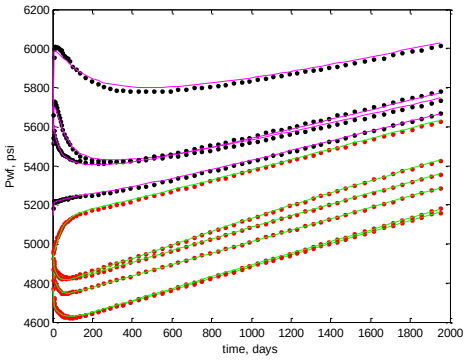
(d) match to water cut data using LM2+LSCH



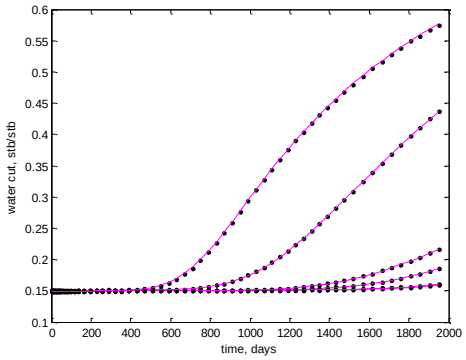
(e) match to pressure data using LM+DE



(f) match to water cut data using LM+DE

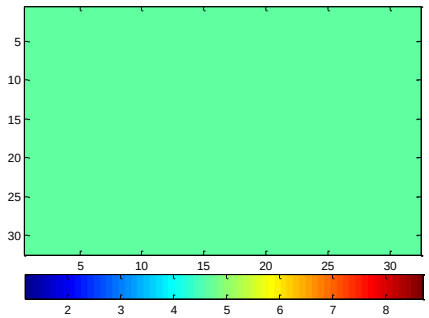


(g) match to pressure data using LM+DE+LSCH

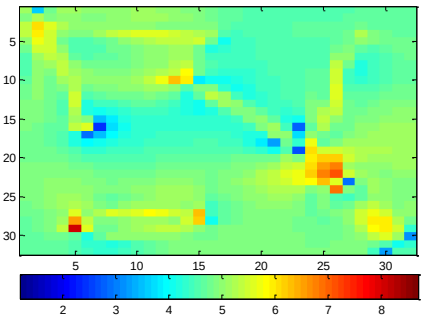


(h) match to water cut using LM+DE+LSCH

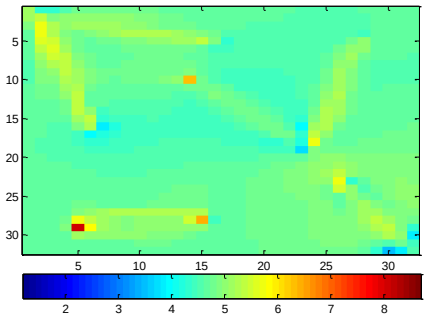
Figure 5-7 Use of different algorithms to match data (Example 2)



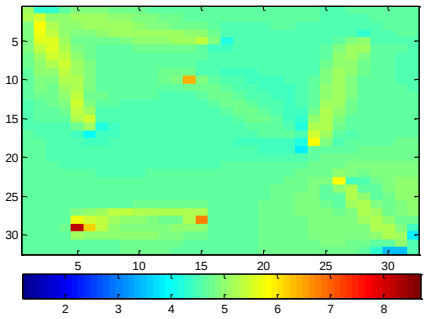
(a) initial guess



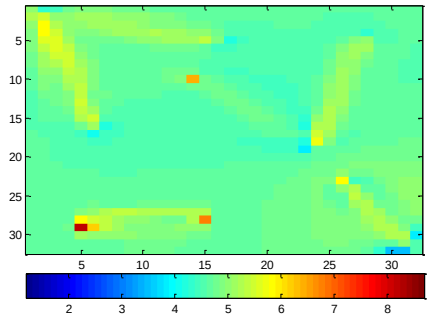
(b) estimate from LM1+LSCH



(c) estimate from LM2+LSCH



(d) estimate from LM+DE



(e) estimate from LM+DE+LSCH

Figure 5-8 Estimated permeability field using different algorithms (Example 2)

Table 5-6 Discretization of Example 2

x-axis	y-axis	z-axis
32	32	1
Lx	Ly	Lz
3200	3200	100

Table 5-7 Well information of Example 2

well index	x-coordinate	y-coordinate	z-coordinate	flow rate	well type
1	16	5	1	700	producer
2	6	16	1	900	producer
3	14	10	1	600	producer
4	26	23	1	400	producer
5	5	29	1	1000	producer
6	28	5	1	500	producer
7	23	18	1	-900	injector
8	15	28	1	-1100	injector
9	1	1	1	-1200	injector
10	32	32	1	-1000	injector

Table 5-8 Fluid properties of Example 2

properties	value
initial density of oil, lbm/ft^3	52
contant used to compute oil vicosity	1.20E-05
isothermal compressibity of oil	2.00E-06
initial water density,lbm/ft^3	62.238
isothermal compressibity of water	5.00E-07
contant used to compute water viscosity	6.00E-08
initial pressure, psi	5000
initial water saturation	0.1
well radius, ft	0.25
skin factor	0
initial porosity	0.25

Table 5-9 Details of implementing optimization algorithms (Example 2)

Case #	algorithm	fevals/iter	# of LM iterations	total fevals	initial k_{norm}	k_{norm} of estimates	time(sec)
Case 1	LM1 + line search	4.3/iter	10	43	1.1911	1.0426	9320
Case 2	LM1 + line search	4.2/iter	10	42	1.1911	1.0646	9170
Case 3	LM1 + line search	2.9/iter	10	29	1.1911	1.1849	7415
Case 1	LM2 + line search	4.7/iter	10	47	1.1911	1.0334	8782
Case 2	LM2 + line search	4.2/iter	10	42	1.1911	1.1712	8103
Case 3	LM2 + line search	4.0/iter	10	40	1.1911	1.1322	7872
Case 1	LM + DE	6.1/iter	10	61	1.1911	1.0349	8137
Case 2	LM + DE	6.1/iter	10	61	1.1911	1.1947	8093
Case 3	LM + DE	6.1/iter	10	61	1.1911	1.1403	8043
Case 1	LM + DE + line search	9.8/iter	10	98	1.1911	1.0221	11631
Case 2	LM + DE + line search	8.2/iter	10	82	1.1911	1.1948	10572
Case 3	LM + DE + line search	7.1/iter	10	71	1.1911	1.14	9486

Table 5-10 Running optimization at the designated iteration (Example 2)

Case #	algorithm	# of LM iterations	total fevals	k_{norm} before	k_{norm} of estimates	time(sec)
case 1	LM1 + line search	8	33	1.1911	1.0877	7456
case 2	LM1 + line search	8	32	1.1911	1.185	7332
case 3	LM1 + line search	8	20	1.1911	1.0909	5926
case 1	LM2 + line search	8	37	1.1911	1.0334	8101
case 2	LM2 + line search	8	32	1.1911	1.1715	7576
case 3	LM2 + line search	8	30	1.1911	1.1322	7258
case 1	LM + DE	5	31	1.1911	1.0759	4211
case 2	LM + DE	5	31	1.1911	1.1894	4246
case 3	LM + DE	5	31	1.1911	1.1429	4240
case 1	LM + DE + line search	5	48	1.1911	1.0759	5694
case 2	LM + DE + line search	5	39	1.1911	1.1949	5435
case 3	LM + DE + line search	5	36	1.1911	1.1429	4761

5.1.3 Example 3

The third reservoir used in the test is distcretized into 64×64 gridblocks with fully distributed permeability field shown in Fig. 5-9. The detailed information referred to discretization, well information, and fluid properties are presented in Table 5-11, Table 5-12, and Table 5-13. According to Fig. 5-9, we observe that reservoir has 24 producers and 16 injectors. The matched data in this example comprise of pressure data measured at all wells and water cut data measured at producers. Thus, there are 64 datasets. Each dataset has 140 data points. Total number of data points required to be matched is 8960. LM is made up to run 10 iterations. At each of LM iteration, we still run DE in 5 iterations.

Table 5-14 shows the detailed information of performance of optimization algorithms. The residuals obtained by different algorithms are shown in Fig. 5-10a, Fig. 5-10b, and Fig. 5-10c, respectively. Firstly, when DE is used, it has the ability to eliminate the uncertainty of estimating damping factor since DE searches for the appropriate value of damping factor over the entire solution domain at each of LM iteration. This enables LM to avoid divergence due to bad initial damping factor at the beginning of optimization. Secondly, even if given an appropriate value of initial damping factor that benefits the trial-and-error approach in estimating damping factor in the remaining iterations, the proposed algorithm perform better than the standard LM algorithm. On one hand, values of residual in Fig. 5-10a and Fig. 5-10b show a perfect overlap when LM+DE and LM+DE+LSCH are used. This goes to show that DE has the ability to find the optimum damping factor for LM algorithm in dealing with large-scale inverse problem, removing the use of line search. On the other hand, within 5 iterations,

the error norms obtained by LM+DE decrease more rapidly than those of LM1+LSCH and LM2+LSCH. If a certain value of residual is given as tolerance for stopping criteria, we observe that the proposed algorithm only needs to run 5 iterations to reach the tolerance while the standard LM algorithm has to run up to 8 iterations. As discussed before, the computation time of applying LM algorithm heavily relies on the computation of sensitivity matrix and gradient, a large number of iterations will yield a huge cost in computation time. Table 5-15 is used to illustrate the performance of four algorithms at the truncated iterations. From this table, we conclude that the proposed algorithm is more efficient than the standard LM algorithm. Besides, by checking the last value of residual in Fig. 5-10a and Fig. 5-10c, we observe that the proposed algorithm yields the smallest value of residual. In other words, the data match from LM+DE should be more accurate than those from LM1+LSCH and LM2+LSCH. However, as observed in Fig. 5-10b, the difference between residual from LM+DE and that from LM+DE+LSCH illustrates that DE may not necessarily find the optimum damping factor in some of the LM iteration. In that case, LM+DE would give a poorer performance than LM+DE+LSCH. But the determination in performance is not significant and the LM+DE is still more efficient than the LM+LSCH.

Figures 5-10d to 5-10f show the values of damping factor corresponding to different cases. We observe that the values of damping factor in LM+DE and LM+DE+LSCH do not necessarily decline with reduction in residual error. Figure 5-11 shows the matched results. From all the algorithms, we observe good matches. Finally, Figure 5-12 shows the estimates of permeability field obtained from the algorithms. None

of the estimated permeability field is close to true field. This is because the information of the observed data is not enough to resolve the large number of unknown parameters.

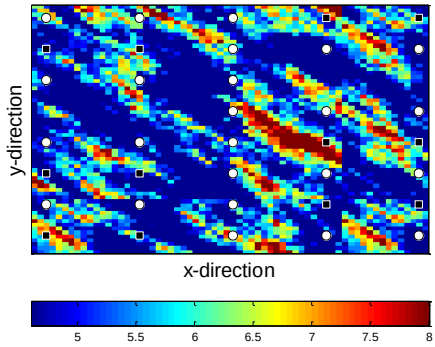
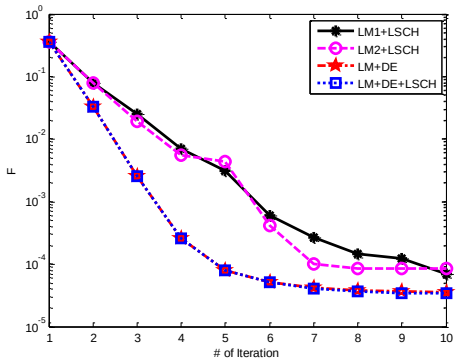
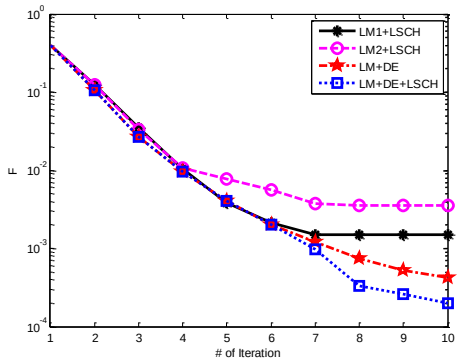


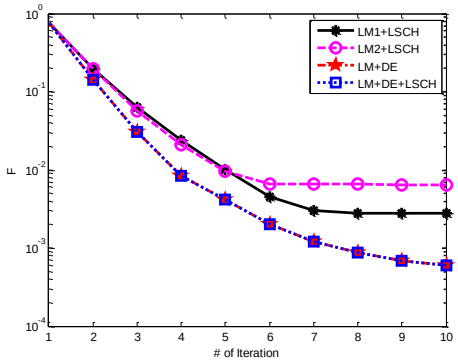
Figure 5-9 Permeability field and well locations: squares indicate injectors and circles indicate producers (Example 3)



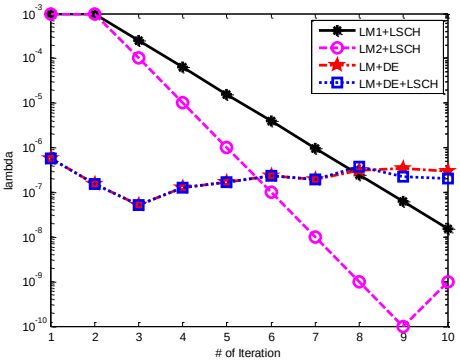
(a) decay of residual of Case 1



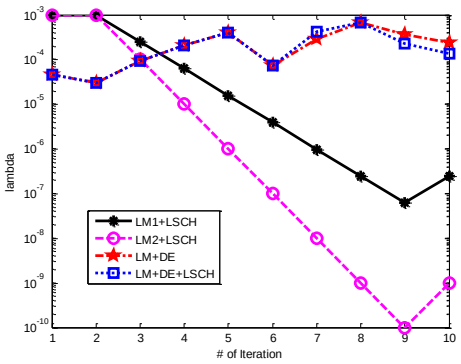
(b) decay of residual of Case 2



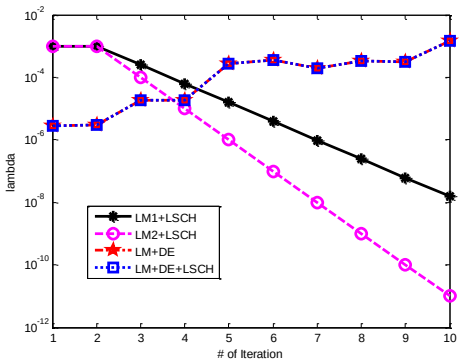
(c) decay of residual of Case 3



(d) values of lambda of Case 1

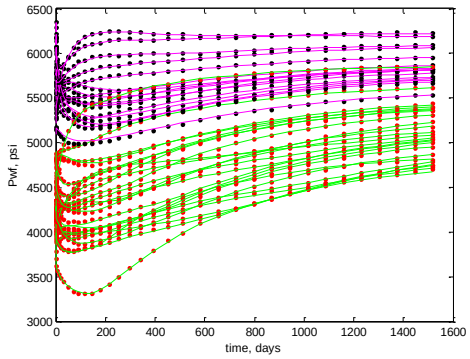


(e) values of lambda of Case 2

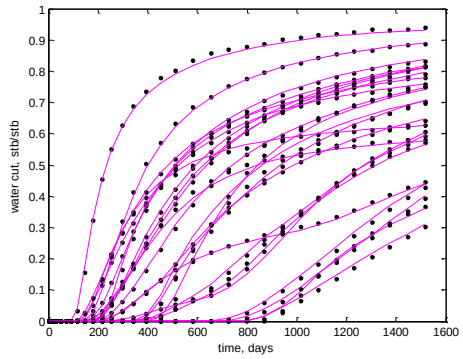


(f) values of lambda of Case 3

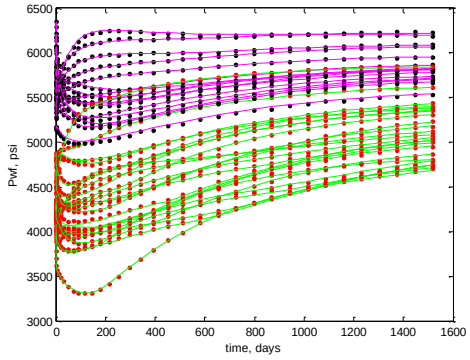
Figure 5-10 Decay of residual and values of damping factor (Example 3)



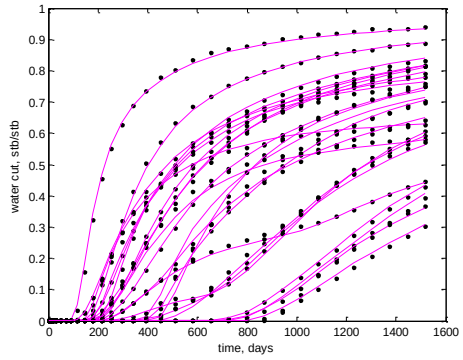
(a) match to pressure data using LM1+LSCH



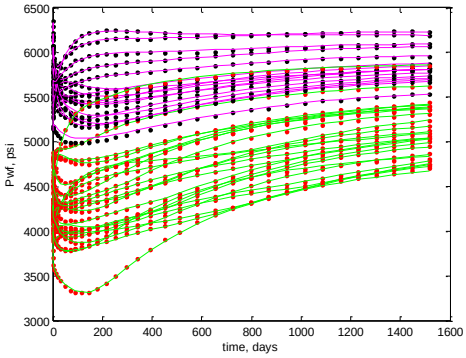
(b) match to water cut data using LM1+LSCH



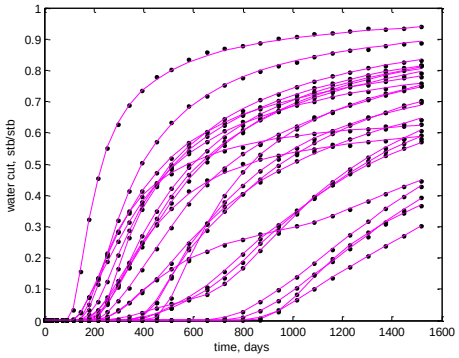
(c) match to pressure data using LM2+LSCH



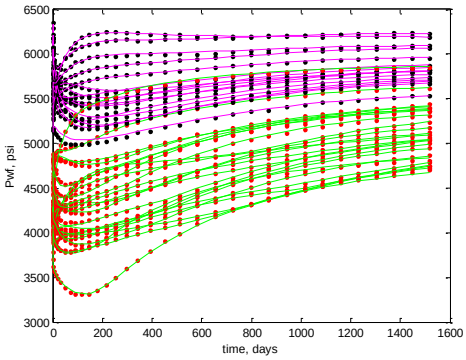
(d) match to water cut data using LM2+LSCH



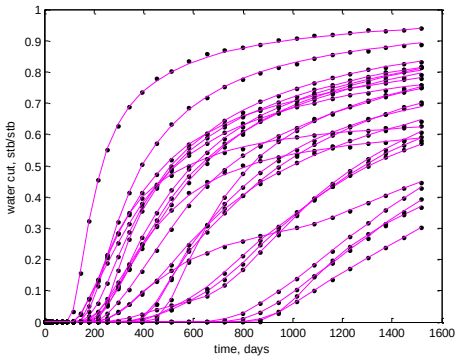
(e) match to pressure data using LM+DE



(f) match to water cut data using LM+DE

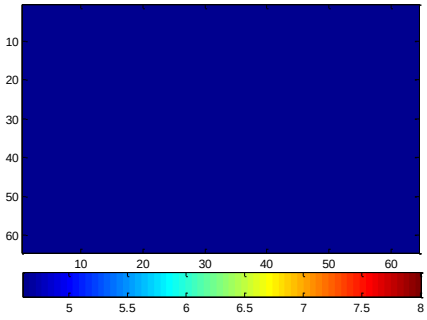


(g) match to pressure data using LM+DE+LSCH

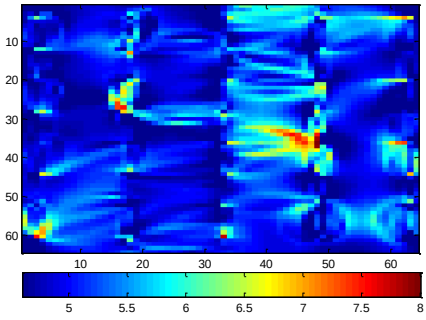


(h) match to water cut using LM+DE+LSCH

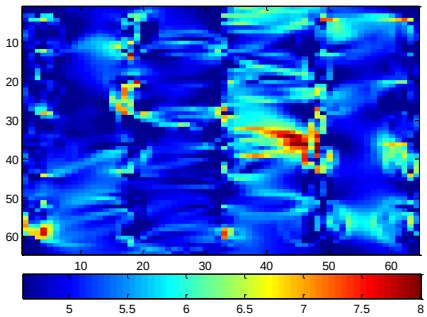
Figure 5-11 Use of different algorithms to match data (Example 3)



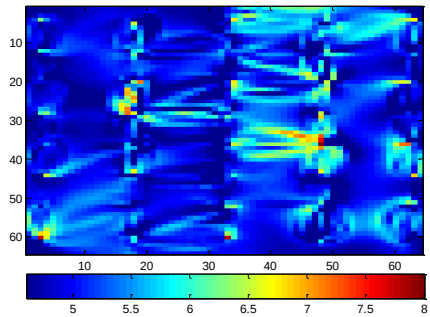
(a) initial guess



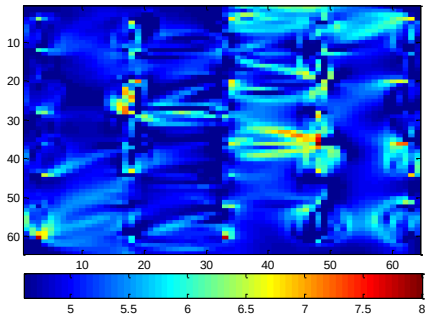
(b) estimate from LM1+LSCH



(c) estimate from LM2+LSCH



(d) estimate from LM+DE



(e) estimate from LM+DE+LSCH

Figure 5-12 Estimated permeability field using different algorithms (Example 3)

Table 5-11 Discretization of Example 3

x-axis	y-axis	z-axis
64	64	1
Lx	Ly	Lz
5120	5120	100

Table 5-12 Well information of Example 3

well index	x-coordinate	y-coordinate	z-coordinate	flow rate	well type
1	3	4	1	2300	producer
2	18	4	1	2500	producer
3	33	4	1	2200	producer
4	48	4	1	-2900	injector
5	63	4	1	-3000	injector
6	3	12	1	-3200	injector
7	18	12	1	-2900	injector
8	33	12	1	2800	producer
9	48	12	1	2400	producer
10	63	12	1	1900	producer
11	3	20	1	2300	producer
12	18	20	1	1500	producer
13	33	20	1	2400	producer
14	48	20	1	-3000	injector
15	63	20	1	-2400	injector
16	3	28	1	-2700	injector
17	18	28	1	-4800	injector
18	33	28	1	2200	producer
19	48	28	1	1700	producer
20	63	28	1	2000	producer
21	3	36	1	2400	producer
22	18	36	1	1800	producer
23	33	36	1	2600	producer
24	48	36	1	-4500	injector
25	63	36	1	-2800	injector
26	3	44	1	-2600	injector
27	18	44	1	-2700	injector
28	33	44	1	1800	producer

29	**48**	**44**	**1**	**2200**	**producer**
30	**63**	**44**	**1**	**1700**	**producer**
31	**3**	**52**	**1**	**1500**	**producer**
32	**18**	**52**	**1**	**1900**	**producer**
33	**33**	**52**	**1**	**2400**	**producer**
34	**48**	**52**	**1**	**-3100**	**injector**
35	**63**	**52**	**1**	**-2600**	**injector**
36	**3**	**60**	**1**	**-4200**	**injector**
37	**18**	**60**	**1**	**-3200**	**injector**
38	**33**	**60**	**1**	**2500**	**producer**
39	**48**	**60**	**1**	**1800**	**producer**
40	**63**	**60**	**1**	**2700**	**producer**

Table 5-13 Fluid properties of Example 3

properties	value
initial density of oil, lbm/ft^3	**40**
contant used to compute oil vicosity	**1.20E-05**
isothermal compressibity of oil	**2.00E-06**
initial water density,lbm/ft^3	**62.238**
isothermal compressibity of water	**5.00E-07**
contant used to compute water viscosity	**6.00E-08**
initial pressure, psi	**5000**
initial water saturation	**1.00E-06**
well radius, ft	**0.25**
skin factor	**0**
initial porosity	**0.25**

Table 5-14 Details of implementing optimization algorithms (Example 3)

Case #	algorithm	fevals/iter	# of LM iterations	total fevals	initial k_{norm}	k_{norm} of estimates	time(sec)
Case 1	LM1 + line search	4.4/iter	10	44	0.7965	0.62	40430
Case 2	LM1 + line search	3.9/iter	10	39	0.7965	0.8549	38166
Case 3	LM1 + line search	3.3/iter	10	33	0.7965	0.6727	37091
Case 1	LM2 + line search	4.4/iter	10	44	0.7965	0.6475	41393
Case 2	LM2 + line search	3.6/iter	10	36	0.7965	0.879	38293
Case 3	LM2 + line search	2.8/iter	10	28	0.7965	0.6617	34162
Case 1	LM + DE	6.1/iter	10	61	0.7965	0.6222	37087
Case 2	LM + DE	6.1/iter	10	61	0.7965	0.8704	37507
Case 3	LM + DE	6.1/iter	10	61	0.7965	0.7161	36265
Case 1	LM + DE + line search	9.8/iter	10	98	0.7965	0.621	49893
Case 2	LM + DE + line search	8.5/iter	10	85	0.7965	0.8719	45702
Case 3	LM + DE + line search	7.1/iter	10	71	0.7965	0.7161	39738

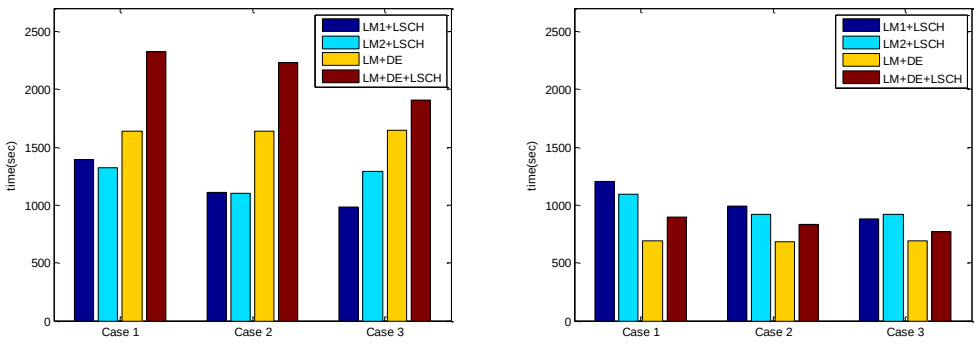
Table 5-15 Running optimization at the designated iteration (Example 3)

Case #	algorithm	# of LM iteration	total fevals	initial k_{norm}	k_{norm} of estimates	time(sec)
Case 1	LM1 + line search	8	34	0.7965	0.6376	32114
Case 2	LM1 + line search	8	29	0.7965	0.8545	30393
Case 3	LM1 + line search	8	23	0.7965	0.6734	28440
Case 1	LM2 + line search	8	34	0.7965	0.634	32508
Case 2	LM2 + line search	8	26	0.7965	0.8782	30173
Case 3	LM2 + line search	8	23	0.7965	0.6617	28793
Case 1	LM + DE	5	36	0.7965	0.6329	19862
Case 2	LM + DE	5	36	0.7965	0.8539	20092
Case 3	LM + DE	5	36	0.7965	0.7155	19247
Case 1	LM + DE + line search	5	48	0.7965	0.6329	25726
Case 2	LM + DE + line search	5	39	0.7965	0.8561	22939
Case 3	LM + DE + line search	5	36	0.7965	0.7155	21068

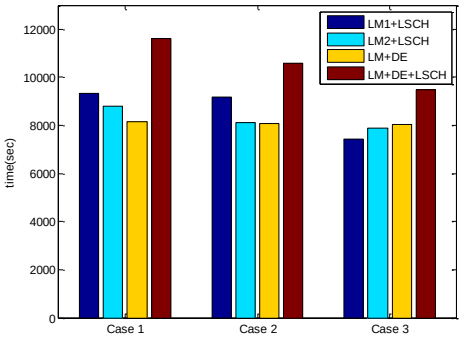
5.1.4 Summary of Time Consumed by Different Algorithms

Figure 5-13 shows the summary of time taken by different algorithms for different problem sizes which are 16×16, 32×32, and 64×64.

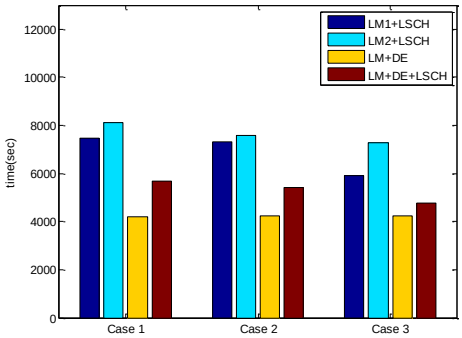
We observe that when we run algorithms in 10 LM iterations for all Examples, the proposed algorithm (LM+DE) shown in Fig. 5- 13a takes longer time than that of the standard LM algorithms (LM1+LSCH and LM2+LSCH). However, when the size of problems increases, the time taken by the proposed algorithm is almost the same as or a bit longer than that of the standard LM algorithms (i.e. Fig 5-13c and Fig. 5-13e). If we set the stopping criterion as a certain value of residual of the objective function, the proposed algorithm will reach this certain value within the limited number of iterations at the early stage of the optimization as mentioned in previous subsection. The results are shown in Fig. 5-13b, Fig. 5-13d, and Fig. 5-13f. From these figures, it is observed that the proposed algorithm does perform better than the standard LM algorithms in all Examples because the time consumed by the proposed algorithm is shorter than that of the latter ones. In other words, the proposed algorithm saves us significant computation time and ensures accuracy for history matching. Thus, we conclude that the proposed algorithm is more efficient than the standard LM algorithms in this work.



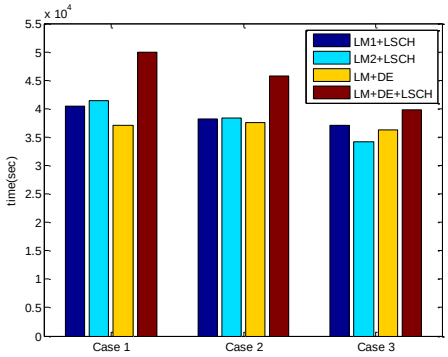
(a) 10 LM iterations (Example 1)



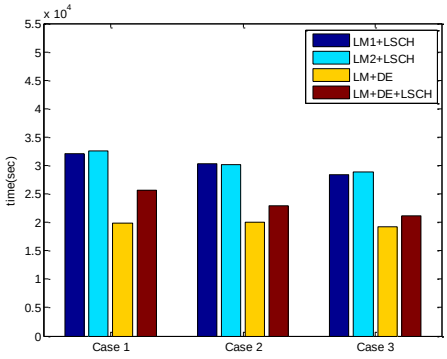
(b) at the designated LM iterations (Example 1)



(c) 10 LM iterations (Example 2)



(d) at the designated LM iterations (Example 2)



(e) 10 LM iterations (Example 3)

(f) at the designated LM iterations (Example 3)

Figure 5-13 Summary of time consumed by different algorithms for different problem sizes

5.2 Correlation Development

In section 5.1, we applied four different algorithms to three reservoir models. In each model, we had three different cases. We observe that the time taken by LM+DE is still significant and comparable to that taken by LM+LSCH. Thus, we develop correlations based on the statistics of the main diagonal elements of the Gauss-Newton Hessian matrix.

5.2.1 Data Selection Criteria

Since the reliability of correlation heavily relies on selection of statistical values of Hessian and values of optimum damping factor, the choice of statistical values becomes

significant.

Initially, we choose maximum, minimum, standard deviation, mean, median, l_1-norm, l_2-norm of main diagonal elements of the Gauss-Newton Hessian matrix. At every iteration of LM (LM+DE), we save values of the main diagonal elements of the Gauss-Newton Hessian matrix in advance. After saving these statistical values, we calculate maximum, minimum, standard deviation, mean, median, l_1-norm, and l_2-norm, respectively. Later, we save them as well as damping factors in each of LM iteration. Once optimization is done, say 10 iterations, there will be 10 datasets containing statistical values and their associated damping factors.

Then, we plot statistical values versus iterations on semi-log graph for all Examples. We observe that some of the data points in statistical values clearly show either decreasing or increasing trend during the optimization. However, some of them are noisily scattered. Especially, the larger the case is, the more limited number of effective points will be. By the demand of developing a generalized correlation, the log-log plot of lambda versus statistical values is then used to select the appropriate data points. We observe that statistical values of maximum, standard deviation, mean, l_1-norm, and l_2-norm, generally show either increasing or decreasing trend. Therefore, the selected statistical values to develop correlation are maximum, standard deviation, mean, l_1-norm, and l_2-norm in this work.

However, not all data points from maximum, standard deviation, mean, l_1-norm, and l_2-norm show us clear and reasonable trend. Practically, only several data points from log-log plot of lambda versus statistics show a decreasing or increasing trend. We save these data points and remove nosily scattered ones. Based on the further selection,

only Case 1 and Case 2 in all examples can be used to develop correlation. But a challenge comes out that there are three examples. Each of them has two suitable cases including the match to pressure data and the match to water cut data. Thus, there will be 6 different groups of the selected data points. In this work, we only choose to develop correlations for small-scale and large-scale reservoirs while ignoring the medium-scale reservoir model. Therefore, there will be 4 groups of data that can be used to develop correlations. The following equation is our proposed form of the correlation:

$$ln(\lambda_C) = a_1 \cdot |ln(max)|^{b_1} + a_2 \cdot |ln(std)|^{b_2} + a_3 \cdot |ln(mean)|^{b_3} + a_4 \cdot |ln(l_1)|^{b_4} + a_5 \cdot |ln(l_2)|^{b_5} \quad (5.3)$$

where λ_C is simulated damping factor. $a_i, b_i, i = 1,2,\ldots,5$ are coefficients of the proposed correlation. $max, std, mean, l_1, l_2$ are maximum, standard deviation, mean, l_1-norm, and l_2-norm, of the main diagonal elements of the Gauss-Newton Hessian matrix.

5.2.2 Correlation from Case 1

In Case 1, we match only the pressure data. The statistical values and their corresponding damping factors are shown in Table 5-16. These damping factors are the optimum values found by running DE at every iteration of the LM method (i.e. in the LM+DE algorithm). Since the log-log plot of lambda versus statistical parameters determines our selection, we first plot lambda versus statistical parameters based on Table 5-16. As mentioned before, there are some points badly scattered. So we have to remove bad points but keep good ones. The data after selection is shown in Table 5-17 including statistical parameters and their corresponding damping factors. Figure 13 is used to show the trends of lambda verses statistical parameters after selecting good points. From Fig. 5-14a to Fig. 5-14e, it

is observed that the selected data on log-log plot of lambda versus statistical parameters show clear trends.

Then, we developed the correlation based on Equation (5.3) and the selected data. The results of simulated lambda is shown in figure 5-14f. From the Fig. 5-14f, we observe that the result generate a good match between the simulated and measured lambda. The formula of correlation is given by

$$ln(\lambda_C) = 0.1250 \cdot |ln(max)|^{-0.4242} - 0.0352 \cdot |ln(std)|^{2.9288} + 0.7832 \cdot$$
$$|ln(mean)|^{-7.4688 \cdot 10^{-5}} + 0.6499 \cdot |ln(l_1)|^{-0.3082} + 0.8501 \cdot |ln(l_2)|^{0.1316} \quad (5.4)$$

Similar to Case 1 of Example 1, we repeat the procedure and develop the correlation from Example 3. Table 5-18 shows original statistical parameters and lambda. Table 5-19 gives the selected statistical parameters and lambda. Figure 5-15a, Fig. 5-15b, Fig. 5-15c, Fig. 5-15d, Fig. 5-15e were used to show lambda versus selected statistical parameters. Figure 5-15f shows the result of the simulated lambda for Example 3 and we obtain a good match between the simulated and measured lambda. The correlation for Case 1 of Example 3 is given by:

$$ln(\lambda_C) = -2.2121 \cdot |ln(max)|^{-3.8695} - 2.7409 \cdot |ln(std)|^{-2.4332} + 1.0256 \cdot$$
$$|ln(mean)|^{-0.1303} - 1.808 \cdot |ln(l_1)|^{3.1037} - 6.9919 \cdot |ln(l_2)|^{3.0251} \quad (5.5)$$

Table 5-16 Statistical parameters of Hessian from Case 1 of Example 1

# of iter	max	min	std	$mean$	$median$	l_1-norm	l_2-norm	λ
1	0.001463	5.78E-10	0.000111	1.49E-05	3.88E-07	0.003812	3.19E-06	0.000381
2	0.00523	1.01E-09	0.000327	2.40E-05	3.63E-07	0.006151	2.74E-05	3.15E-07
3	0.005071	8.34E-10	0.000317	2.18E-05	2.47E-07	0.005581	2.57E-05	3.96E-07
4	0.004936	5.20E-10	0.000309	2.12E-05	1.91E-07	0.005423	2.44E-05	5.61E-07
5	0.004753	5.10E-10	0.000297	2.06E-05	1.87E-07	0.005272	2.26E-05	1.15E-06
6	0.004634	5.30E-10	0.00029	2.02E-05	1.86E-07	0.005172	2.15E-05	7.95E-07
7	0.004478	5.34E-10	0.00028	1.97E-05	1.85E-07	0.005043	2.01E-05	5.82E-07
8	0.004287	5.38E-10	0.000268	1.91E-05	1.84E-07	0.004884	1.84E-05	4.32E-07
9	0.004085	5.45E-10	0.000256	1.84E-05	1.83E-07	0.004715	1.68E-05	3.18E-07
10	0.003899	5.55E-10	0.000244	1.78E-05	1.82E-07	0.004561	1.53E-05	6.42E-07

Table 5-17 Selected Data for Case 1 of Example 1

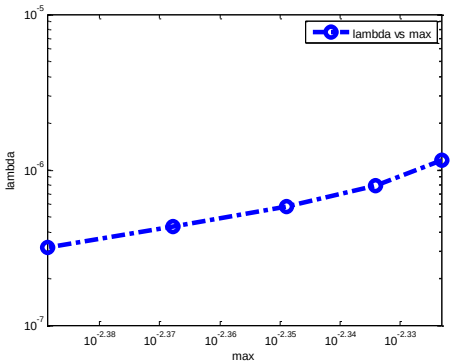
# on the points	max	std	$mean$	l_1-norm	l_2-norm	λ
5	0.004753	0.000297	2.06E-05	0.005272	2.26E-05	1.15E-06
6	0.004634	0.00029	2.02E-05	0.005172	2.15E-05	7.95E-07
7	0.004478	0.00028	1.97E-05	0.005043	2.01E-05	5.82E-07
8	0.004287	0.000268	1.91E-05	0.004884	1.84E-05	4.32E-07
9	0.004085	0.000256	1.84E-05	0.004715	1.68E-05	3.18E-07

Table 5-18 Statistical parameters of Hessian from Case 1 of Example 3

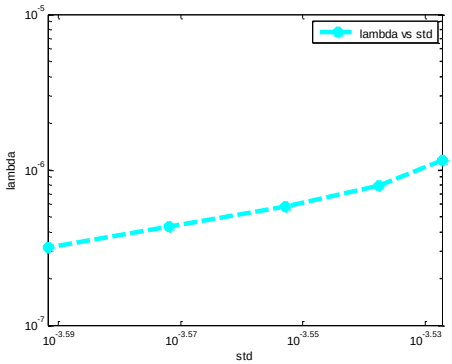
# of iter	max	min	std	$mean$	$median$	l_1-norm	l_2-norm	λ
1	0.037785	1.04E-10	0.001492	0.000125	4.23E-07	0.510496	0.009178	5.76E-07
2	0.02712	2.80E-11	0.001016	7.26E-05	2.18E-07	0.297404	0.00425	1.55E-07
3	0.028583	4.58E-11	0.00103	6.89E-05	1.54E-07	0.282387	0.004364	5.12E-08
4	0.029345	3.25E-11	0.001072	7.11E-05	1.39E-07	0.291283	0.004727	1.29E-07
5	0.029434	2.88E-11	0.001083	7.18E-05	1.41E-07	0.294249	0.004821	1.64E-07
6	0.02966	2.42E-11	0.00109	7.23E-05	1.41E-07	0.296254	0.004887	2.30E-07
7	0.029849	1.89E-11	0.001099	7.30E-05	1.34E-07	0.298967	0.004969	1.89E-07
8	0.029809	1.87E-11	0.0011	7.31E-05	1.38E-07	0.299221	0.004978	3.75E-07
9	0.029783	1.83E-11	0.001101	7.31E-05	1.39E-07	0.299294	0.004982	2.22E-07
10	0.029861	1.77E-11	0.001102	7.32E-05	1.38E-07	0.299777	0.004999	2.02E-07

Table 5-19 Selected Data for Case 1 of Example 3

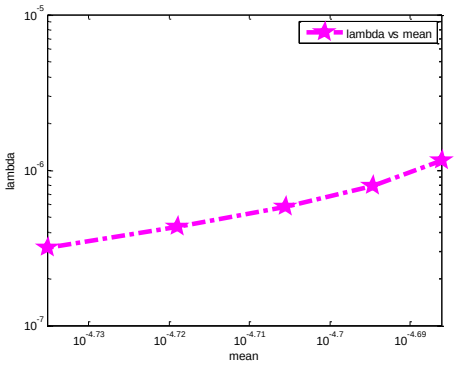
# on the points	max	std	$mean$	l_1-norm	l_2-norm	λ
3	0.028583	0.00103	6.89E-05	0.282387	0.00436	5.12E-08
4	0.029345	0.001072	7.11E-05	0.291283	0.00473	1.29E-07
5	0.029434	0.001083	7.18E-05	0.294249	0.00482	1.64E-07
6	0.02966	0.00109	7.23E-05	0.296254	0.00489	2.30E-07
8	0.029809	0.0011	7.31E-05	0.299221	0.00498	3.75E-07



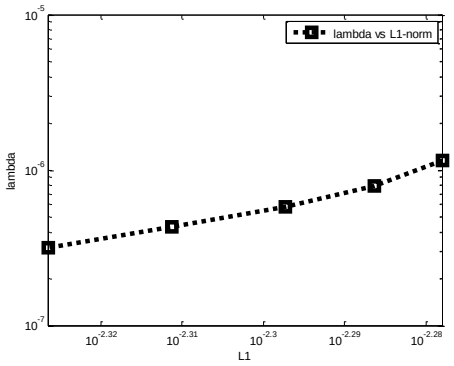
(a) lambda vs max



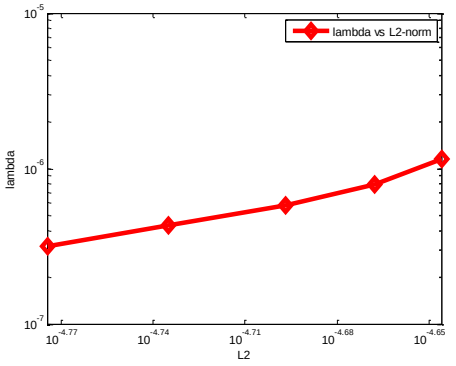
(b) lambda vs standard deviation



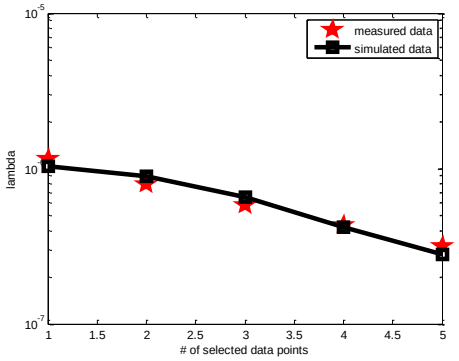
(c) lambda vs mean



(d) lambda vs L1-norm

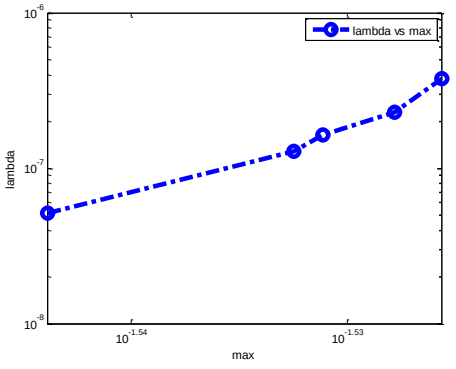


(e) lambda vs L2- norm

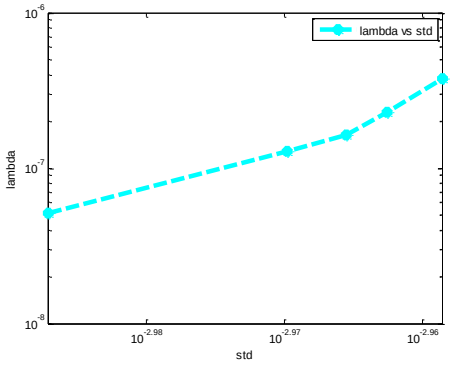


(f) results of simulated lambda

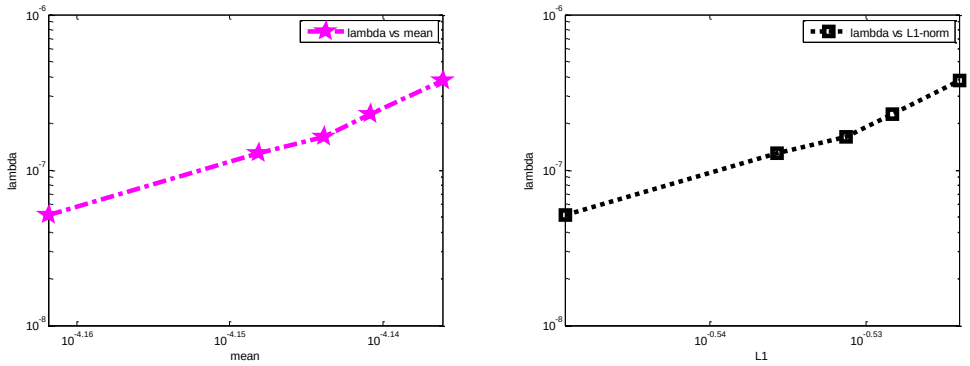
Figure 5-14 lambda vs statistical parameters of Case 1 of Example 1 and results of simulated lambda



(a) lambda vs max

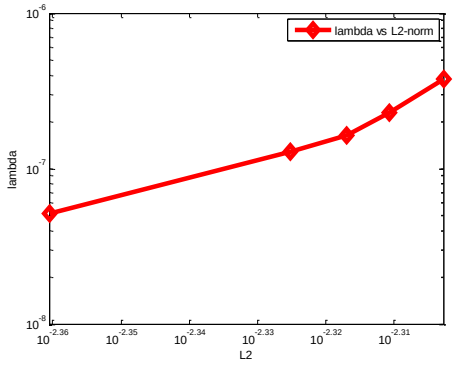


(b) lambda vs standard deviation

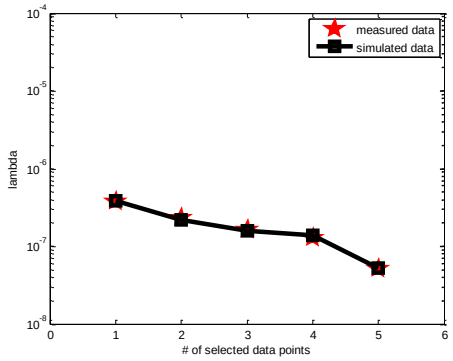


(c) lambda vs mean

(d) lambda vs L1-norm



(e) lambda vs L2- norm



(f) results of simulated lambda

Figure 5-15 lambda vs statistical parameters of Case 1 of Example 3 and results of simulated lambda

5.2.3 Correlation from Case 2

In this subsection, we developed correlation from Case 2 of Example 1. Table 5-20 shows the statistical parameters and lambda obtained by using LM+DE when matching water cut data. Table 5-21 shows the statistical parameters and lambda after selection based on plotting lambda versus statistical parameters on log-log graph. Figure 5-16a, Fig. 5-16b, Fig. 5-16c, Fig. 5-16d, Fig. 5-16e are the selected lambda versus statistical parameters. We observe that the selected data points are not continuous in the sequence. The reason may be the complexity of matching to water cut data. Figure 5-16f is the result of the simulated lambda matching to measured ones. The formula accounting for this correlation is:

$$ln(\lambda_C) = 2.1845 \cdot |ln(max)|^{3.1157} - 1.9035 \cdot |ln(std)|^{-2.8587} - 0.3646 \cdot |ln(mean)|^{-0.0630} - 1.5434 \cdot |ln(l_1)|^{2.1501} - 0.2372 \cdot |ln(l_2)|^{0.0339} \quad (5.6)$$

As for correlation from Example 3, Table 5-22 shows the statistical parameters and their associated values of lambda. The matched data is water cut data. Table 5-23 shows the selected parameters according to log-log graph of lambda versus statistical parameters. As shown in Fig. 5-17a, Fig. 5-17b, Fig. 5-17c, Fig. 5-17d, and Fig. 5-17e, we observe that the general trend for plotting lambda versus statistical parameters show clear trends along certain direction. But we also observe that the number of data points plotted on the log-log graph is not the same among different datasets of statistical parameters. The reason is that Example 3 is a high-dimensional inverse problem and the log-log graph of lambda versus statistical parameters hardly show any trends if we select the fixed data points in sequence. For example, Fig. 5-17a, Fig. 5-17b and Fig. 5-17c show lambda versus maximum, lambda versus standard deviation, and lambda versus l_2-norm using 5 data points. But Fig. 5-17d and Fig. 5-17e use 4 data points in log-log graph of lambda versus mean, and lambda versus l_1-norm. Moreover, the data points in Fig. 5-17a, Fig. 5-17b, and Fig. 5-17c are mutually different. Thus, they are not fixed data in sequence. This phenomenon could be observed in Fig. 5-17d and Fig. 5-17e. If we keep using fixed data points for all these figures, the data points will be badly scattered and shows useless information. To select the data effectively, we have to plot them with different number of data points based on their individual trend.

Figure 17f is the simulated result of lambda which is shown by the black line with squares. The formula of this correlation is given by:

$$ln(\lambda_C) = 2.2553 \cdot |ln(max)|^{0.8680} - 2.0849 \cdot |ln(std)|^{-0.0324} - 1.1628 \cdot |ln(mean)|^{-0.9023} + 5.0634 \cdot |ln(l_1)|^{-1.9849} - 2.2016 \cdot |ln(l_2)|^{-6.0913} \quad (5.7)$$

Table 5-20 Statistical values of Hessian from Case 2 of Example 1

# of iter	max	min	std	$mean$	$median$	l_1-norm	l_2-norm	λ
1	0.008573	7.88E-08	0.001097	0.000406	9.14E-05	0.103883	0.000349	5.22E-07
2	0.009694	1.35E-07	0.001061	3.65E-04	8.56E-05	0.093547	0.000321	3.09E-06
3	0.008879	8.09E-09	0.001008	3.47E-04	8.84E-05	0.088952	0.00029	1.73E-05
4	0.008655	4.56E-09	0.000981	3.39E-04	8.69E-05	0.086836	0.000275	3.43E-04
5	0.008675	4.85E-09	0.00098	3.39E-04	8.67E-05	0.086747	0.000274	4.56E-04
6	0.008685	5.05E-09	0.00098	3.39E-04	8.66E-05	0.086685	0.000274	2.08E-04
7	0.008697	5.27E-09	0.000979	3.38E-04	8.65E-05	0.086649	0.000274	1.87E-04
8	0.008757	6.01E-09	0.000979	3.38E-04	8.59E-05	0.086542	0.000273	6.24E-05
9	0.008773	5.59E-09	0.000979	3.38E-04	8.57E-05	0.086612	0.000274	1.62E-04
10	0.008805	5.09E-09	0.00098	3.39E-04	8.59E-05	0.086732	0.000275	1.41E-04

Table 5-21 Selected Data for Case 2 of Example 1

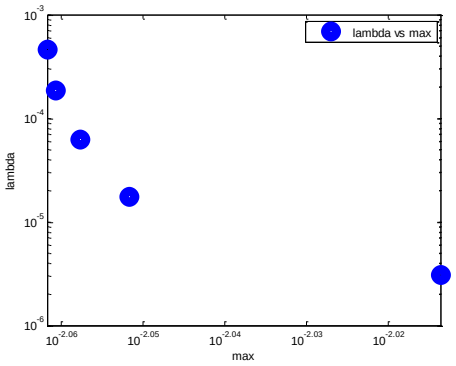
# on the points	max	std	$mean$	l_1-norm	l_2-norm	λ
1	0.008573	0.001097	4.06E-04	0.103883	0.000349	5.22E-07
2	0.009694	0.001061	3.65E-04	0.093547	0.000321	3.09E-06
3	0.008879	0.00101	3.47E-04	0.088952	0.00029	1.73E-05
4	0.008655	0.000981	3.39E-04	0.086836	0.000275	3.43E-04
5	0.008675	0.00098	3.39E-04	0.086747	0.000274	4.56E-04
7	0.008697	0.000979	0.000338	0.086649	0.000274	0.000187
8	0.008757	0.000979	0.000338	0.086542	0.000273	6.24E-05

Table 5-22 Statistical values of Hessian from Case 2 of Example 3

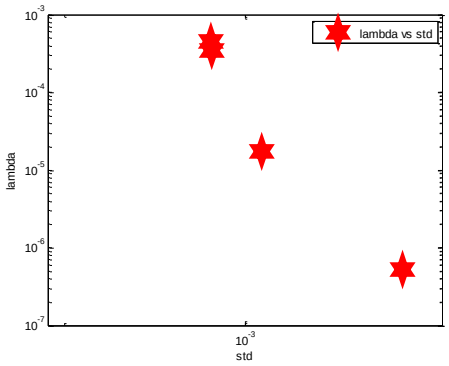
# of iter	max	min	std	$mean$	$median$	l_1-norm	l_2-norm	λ
1	0.011206	1.09E-10	0.000442	0.000103	1.09E-05	0.42169	0.000843	4.64E-05
2	0.013779	3.04E-10	0.000533	1.09E-04	1.08E-05	0.445567	0.001211	3.03E-05
3	0.017721	5.55E-10	0.000564	1.07E-04	1.06E-05	0.437023	0.001348	9.41E-05
4	0.018074	6.77E-10	0.000546	1.04E-04	1.08E-05	0.425913	0.001263	2.06E-04
5	0.017681	7.49E-10	0.00053	1.02E-04	1.08E-05	0.418012	0.001195	3.94E-04
6	0.01723	7.88E-10	0.000521	1.01E-04	1.08E-05	0.413136	0.001151	7.27E-05
7	0.016407	9.18E-10	0.000511	9.94E-05	1.07E-05	0.407245	0.001111	4.24E-04
8	0.015921	9.65E-10	0.000508	9.88E-05	1.07E-05	0.404843	0.001095	6.66E-04
9	0.015492	9.94E-10	0.000505	9.84E-05	1.07E-05	0.402955	0.001085	2.27E-04
10	0.015452	1.02E-09	0.000505	9.83E-05	1.07E-05	0.402791	0.001086	1.34E-04

Table 5-23 Selected Data for Case 2 of Example 3

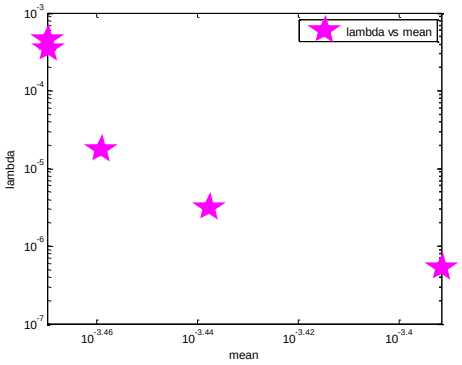
# on the points	max	std	$mean$	l_1-norm	l_2-norm	λ
1	0.011206	0.000442	1.03E-04	0.42169	0.000843	4.64E-05
2	0.013779	0.000533	1.09E-04	0.445567	0.00121	3.03E-05
3	0.017721	0.000564	1.07E-04	0.437023	0.00135	9.41E-05
4	0.018074	0.000546	1.04E-04	0.425913	0.00126	2.06E-04
6	0.01723	0.000521	1.01E-04	0.413136	0.00115	7.27E-05
7	0.016407	0.000511	9.94E-05	0.407245	0.00111	0.000424
8	0.015921	0.000508	9.88E-05	0.404843	0.0011	6.66E-04
9	0.015492	0.000505	9.84E-05	0.402955	0.00108	0.000227



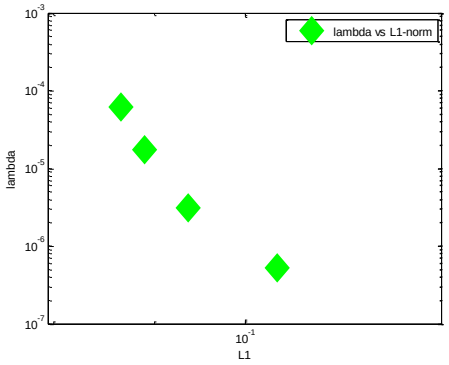
(a) lambda vs max



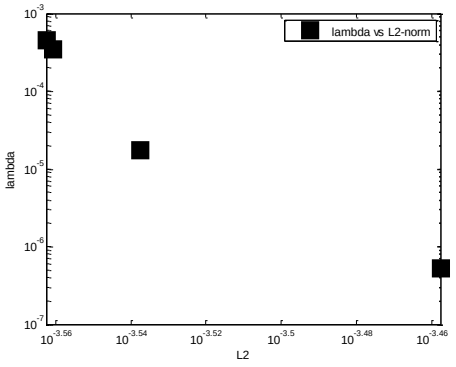
(b) lambda vs standard deviation



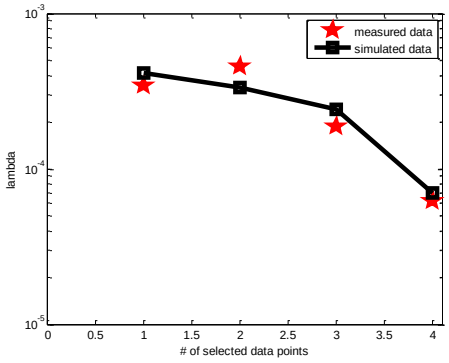
(c) lambda vs mean



(d) lambda vs L1-norm

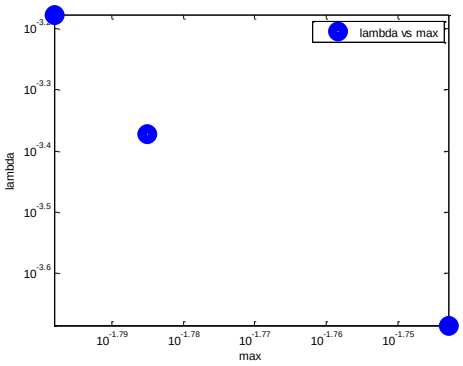


(e) lambda vs L2- norm

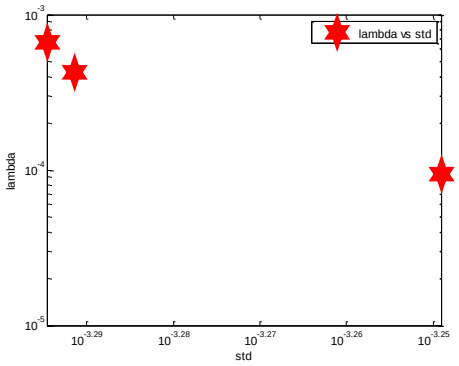


(f) results of simulated lambda

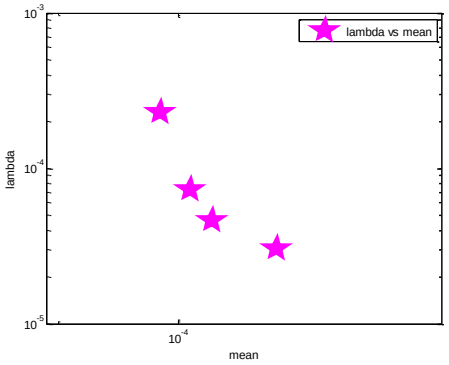
Figure 5-16 lambda vs statistical parameters of Case 2 of Example 1 and results of simulated lambda



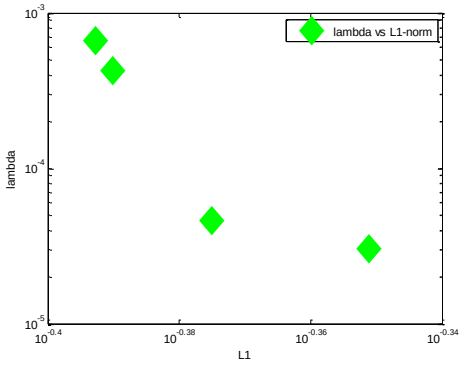
(a) lambda vs max



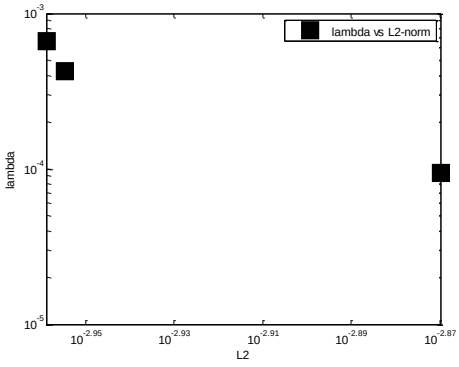
(b) lambda vs standard deviation



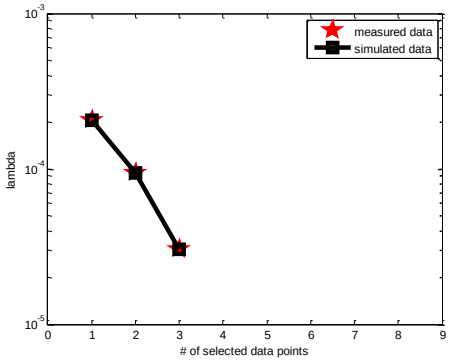
(c) lambda vs mean



(d) lambda vs L1-norm



(e) lambda vs L2- norm



(f) results of simulated lambda

Figure 5-17 lambda vs statistical parameters of Case 2 of Example 3 and results of simulated lambda

5.3 Test of the Correlations

In this section, we show the application of the correlations developed from Examples. First, we use the correlations generated from Case 1 and Case 2 of Example 1 to test all the cases of Example 1 though there is no correlation developed from Case 3. Second, we

repeat the same process to three cases of Example 3. At last, we use the correlations developed from Example 1 and Example 3 to test each other again.

5.3.1 Apply CE1 to Example 1

For convenience of explanation, we define CE1 as the correlation developed from Example 1, define P-based correlation as the correlation developed from pressure data which is Case 1, define W-based correlation as the correlation developed from water cut data which is Case 2. LMC+LSCH uses the values of lambda developed in correlation to optimize the problems. The general procedure is: First, run LM+LSCH (LM1+LSCH and LM2+LSCH) to test Examples; Second, run LM+DE to test Examples; Third, run LM+DE+LSCH to test Examples; at last, run LMC+LSCH using the developed correlations only. The maximum number of iterations to run each optimization algorithm is made up to 10.

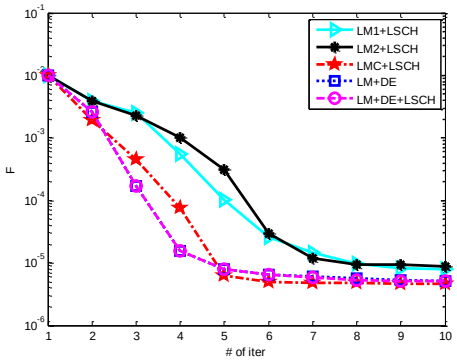
Figure 5-18 shows the decreasing trends of the objective function value obtained by applying different optimization algorithms. LMC+LSCH used the P-based CE1. We observe that LMC+LSCH performs better than standard LM algorithms. First, within five iterations at the early stage of the optimization, LMC+LSCH converges faster than that of the standard LM algorithms. Second, if checking the last several objective function values, the results of LMC+LSCH are almost the same as that of LM+DE as well as LM+DE+LSCH. This fact shows that LMC+LSCH generates more accurate results than that of the standard LM algorithms. Thus, LMC+LSCH is relatively more efficient and reliable than standard LM algorithms.

Figure 5-18b shows the decreasing trends of the objective function of Case 2 using different optimization algorithms. From the Fig. 5-18b, we observe that the

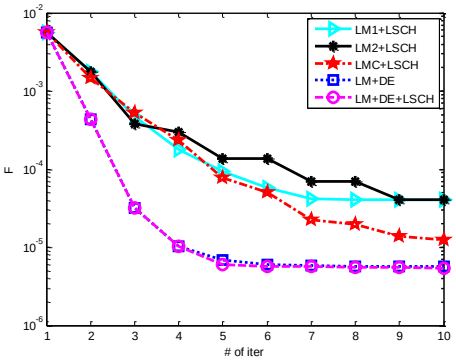
performance of LMC+LSCH is better than that of the standard LM algorithms because it generates smaller objective function values than that of the standard LM algorithms. Thus, results obtained by LMC+LSCH should be more accurate than that of the standard LM algorithms. Then, we use LMC+LSCH to test Case 3 which is the combined data. Unfortunately, LMC+LSCH performs relatively worse than the standard LM algorithms, LM+DE, and LM+DE+LSCH. Because it doesn't converge faster and the objective function values are bigger than all the other algorithms. In general, LMC+LSCH is an efficient algorithm when dealing with Case 1 and Case 2. But it is inefficient when dealing with Case 3.

Figure 18 show the decay of the objective function values obtained by W-based CE1. When implementing LMC+LSCH, we set up a pre-condition that if the residual of the objective function doesn't decrease, the value of the objective function will keep the same as previous one. If such value is unchanged for five iterations, optimization process will stop automatically and inform us that the algorithm can not converge. Under this condition, we observe that LMC+LSCH doesn't converge in all Cases. From Fig.19, we conclude that W-based CE1 is not a suitable correlation that can be applied to test Example 1.

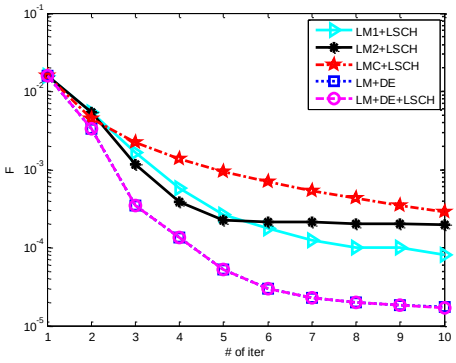
Since we have analyzed the estimated permeability field during testing the feasibility of optimization algorithms in Section 5.1 for all Examples, here we ignore the plot of estimated permeability field. As a matter of fact, checking the values of residuals of objective function may give us enough confidence on the estimation of permeability field.



(a) Case 1

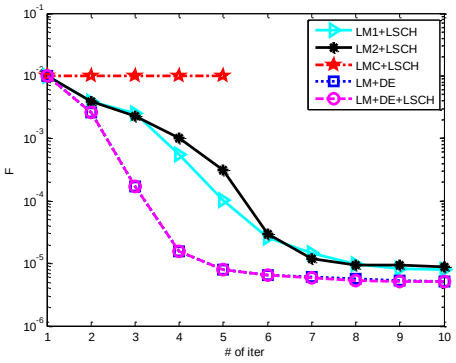


(b) Case 2

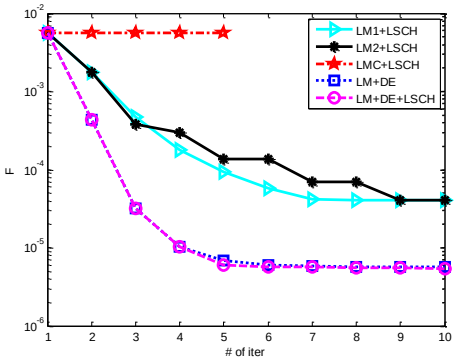


(c) Case 3

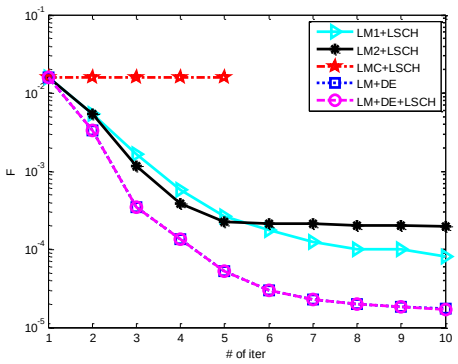
Figure 5-18 Use of P-based correlation developed by Example 1 to test Example 1



(a) Case 1



(b) Case 2



(c) Case 3

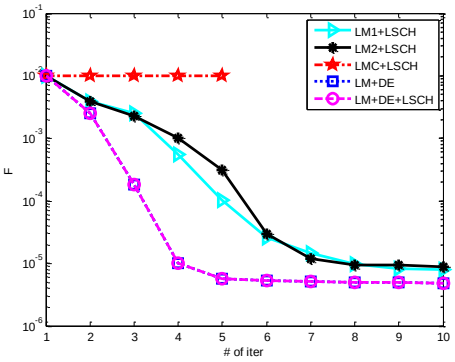
Figure 5-19 Use of W-based correlation developed by Example 1 to test Example 1

5.3.2 Apply CE3 to Example 1

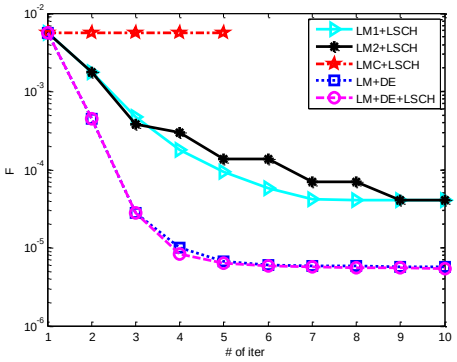
In this subsection, we define CE3 as the correlation developed from Example 3 in advance. Then, we applied P-based CE3 and W-based CE3 to test Example 1 in purpose that if the correlations developed from large-scale reservoir model have the capacity of solving small-scale inverse problem. As shown from Fig. 5-20a to Fig. 5-20c, we observe that the residuals of the objective function obtained by LMC+LSCH keep constant continuously for five iterations, which means LMC+LSCH can not converge at all. However, all the other optimization algorithms such as the standard LM algorithms, LM+DE, and LM+DE+LSCH, converge very well. This fact shows that the P-based CE3 is not a suitable correlation for solving history matching problem of Example 1. In other word, the P-based correlation developed from large-scale reservoir is incapable of being applied to small-scale reservoir.

Then, we checked the application of W-based CE3 to Example 1. Figure 21a clearly shows two advantages of using LMC+LSCH. First, within six iterations at the early stage of optimization, LMC+LSCH converges faster than the standard LM algorithms. Second, the last value of the residual of the objective function is smaller than

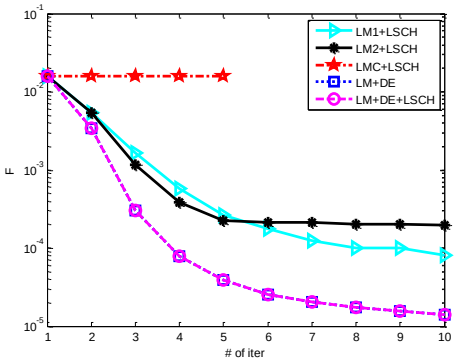
that of the standard LM algorithms. This means LMC+LSCH obtains higher accuracy compared to the standard LM algorithms. Moreover, both of the two advantages are observed in Fig. 21b which is the match to water data. As for Case 3, it is observed that the last objective function value obtained by LMC+LSCH is smaller than that of the standard LM algorithms. Therefore, we conclude that the W-based CE3 is a suitable correlation because it shows improvement on the match to all the Cases. Furthermore, correlation developed from the match of the water cut data of the large-scale reservoir is reliable for solving small-scale reservoir.



(a) Case 1

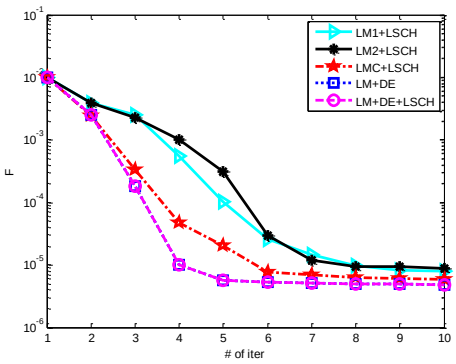


(b) Case 2

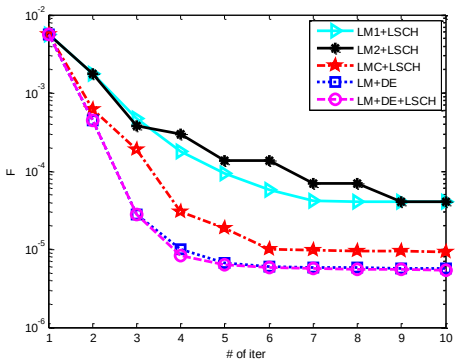


(c) Case 3

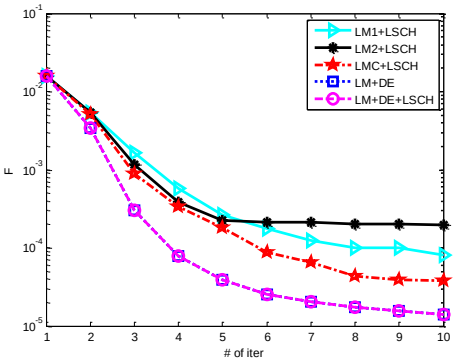
Figure 5-20 Use of P-based correlation developed by Example 3 to test Example 1



(a) Case 1



(b) Case 2



(c) Case 3

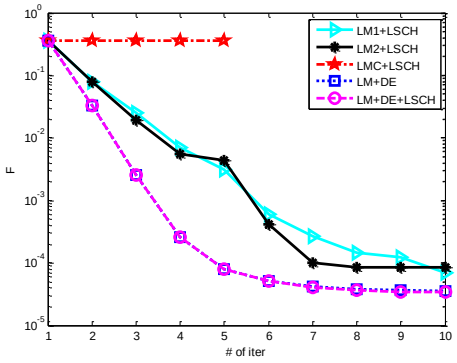
Figure 5-21 Use of W-based correlation developed by Example 3 to test Example 1

5.3.3 Apply CE3 to Example 3

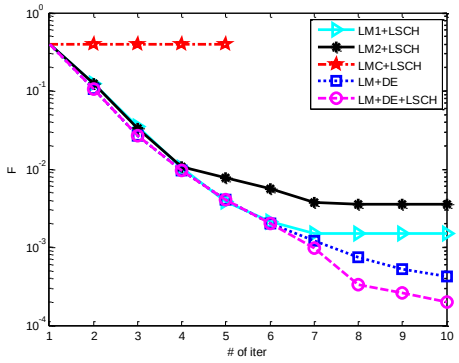
In this subsection, CE3 is applied to test all the Cases of Example 3. Figure 5-22 shows that our proposed correlation developed from the match of pressure data of Example 3 doesn't work well because the residuals of the objective function do not decrease when LMC+LSCH is used. In other words, LMC+LSCH diverges during the optimization. However, all the other optimization algorithms converge well and yield decreasing trends of the objective function values. Thus, we conclude that the P-based correlation developed from Example 3 can not be applied for history matching problems.

Then, we check the residuals of the objective function obtained by W-based

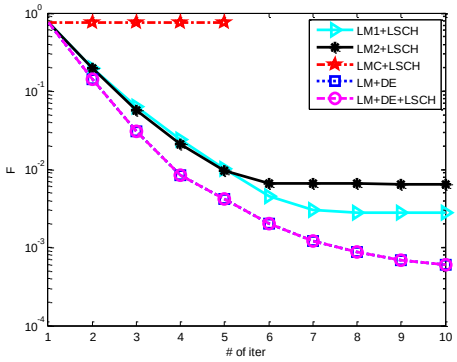
correlation. Figure 5-23 shows that the W-based correlation developed from the match of water cut data of Example 3 doesn't work well, too. LMC+LSCH diverges after five iterations while all the other algorithms converge well during the optimization. Thus, the conclusion is made that W-based correlation is not suitable for solving the inverse problems of Example 3.



(a) Case 1

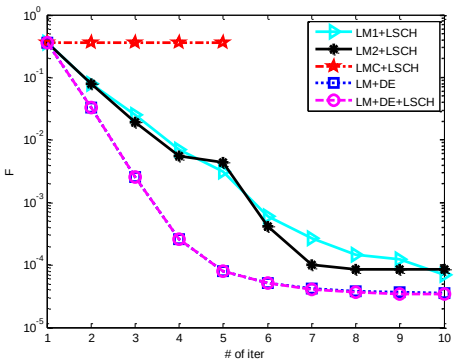


(b) Case 2

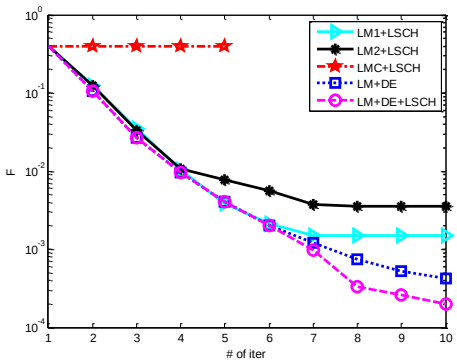


(c) Case 3

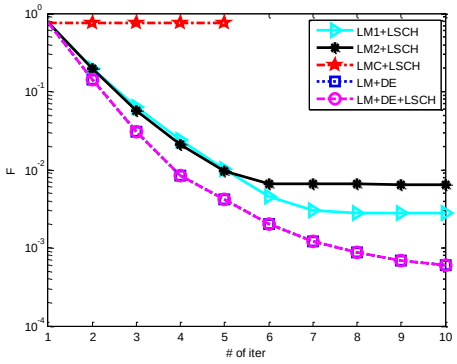
Figure 5-22 Use of P-based correlation developed by Example 3 to test Example 3



(a) Case 1



(b) Case 2



(c) Case 3

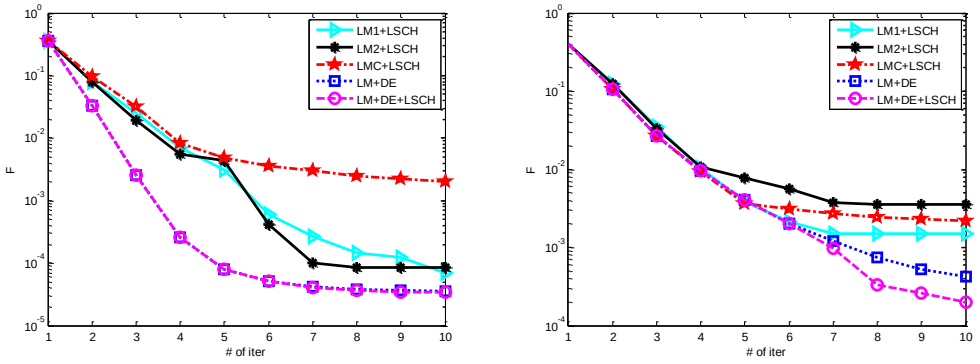
Figure 5-23 Use of W-based correlation developed by Example 3 to test Example 3

5.3.4 Apply CE1 to Example 3

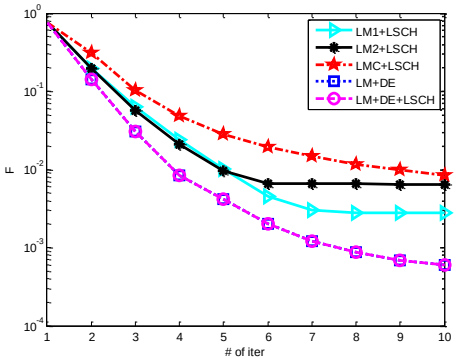
The last test is applying correlation developed from Example 1 to test Example 3 and checking if correlation derived from small-scale reservoir can be used for large-scale reservoir. Figure 5-24a presents the decay of residuals of the objective function using different algorithms. LMC+LSCH doesn't perform well compared to all the other optimization algorithms. Similar phenomenon is observed in Fig. 5-24c which is the match of the combined data. In Fig. 5-24a and Fig. 5-24c, both of the LMC+LSCH converge well. But the performances of LMC+LSCH in two figures are not as well as that of the standard LM algorithms, LM+DE, and LM+DE+LSCH. However, Fig. 5-24b

shows that LMC+LSCH yields more accurate data matches compared to LM2+LSCH. Thus, the P-based correlation developed from small-scale reservoir has capacity of solving large-scale reservoir but it may not that efficient and accurate compared to the standard LM algorithms, LM+DE, and LM+DE+LSCH.

As for W-based correlation developed from Example 1, it is not a suitable for solving large-scale history matching problem in Example 3 because LMC+LSCH diverges in all Cases (shown in Figure 5-25).

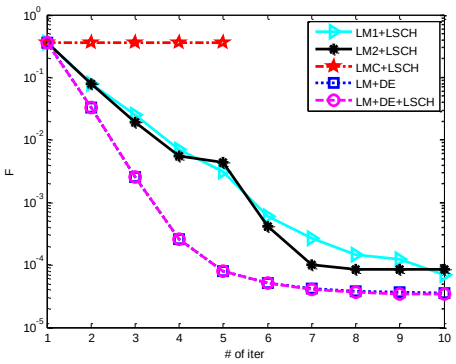


(a) Case 1 **(b) Case 2**

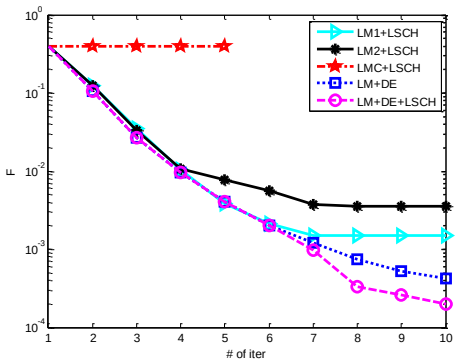


(c) Case 3

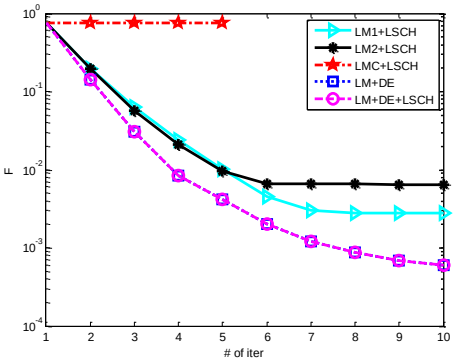
Figure 5-24 Use of P-based correlation developed by Example 1 to test Example 3



(a) Case 1



(b) Case 2



(c) Case 3

Figure 5-25 Use of W-based correlation developed by Example 1 to test Example 3

CHAPTER 6

CONCLUSION AND RECOMMENDATION

Based on the results shown in Chapter 5, we address several important conclusions referred to the application of Differential Evolution which is coupled with Levenberg-Marquardt algorithm and application of correlations used for small and large-scale inverse models. Also, we present our recommendations for future research.

6.1 Application of DE

In this work, we proposed an algorithm that uses DE to estimate the damping factor for LM algorithm. Then, we applied the developed algorithm to three different examples representing small, medium, and large-scale reservoirs.

First, the application of DE to estimate damping factor eliminates the uncertainty of trial-and-error approach in estimating damping factor and the use of line search. Because DE is capable of estimating optimum damping factor for LM algorithm and the proposed algorithm has capacity of solving all inverse problems of different sizes as shown in this work. Moreover, with the use of optimum damping factor, DE has the ability to determine the suitable steplength without line search approach, especially when we match pressure data and combined data in all examples.

Second, DE enables LM to converge very fast and yield more accurate data match. It has been shown in all examples that within limited number of the iterations at the early stage, the decay of residuals of objective function obtained by LM+DE or LM+DE+LSCH is much faster than that of standard LM algorithm, especially when we

match only pressure data and combined data. In addition, in all the examples considered, DE ensured LM to achieve the smallest residuals at the early stage of optimization.

Third, DE enables LM to achieve more accurate match to water cut data in all examples. Often, the exact match to water cut is very difficult because the magnitude of water cut data is quite small. With the use of DE, LM is able to obtain the smaller objective function values than the standard LM algorithms.

Finally, we conclude that DE is efficient and reliable for estimating damping factor when LM is used for reservoir parameter estimation.

6.2 Application of Correlation

During the implementation of LM+DE, we saved statistical values of the main diagonal elements of the Hessian at each of LM iteration and their associated values of damping factor. Based on the data selection, we chose the appropriate data points which show clear trends on log-log graph of lambda versus statistical parameters. Then, based on our proposed correlation formula, we developed correlation from small and large-scale inverse problems. Furthermore, we applied four different correlations to test small and large-scale problems, checking that if our correlations could be generalized or not.

First, we observed that correlation generated by the selected statistical parameters and lambda can not give us a generalized formula. The major reason is that the coefficients of formula are not consistent.

Second, the performance of the P-based correlation developed from Example 1 has the ability to solve both of the small and large-scale inverse problems. Moreover, it performs relatively more efficient and accurate than the standard LM algorithms in small-

scale reservoir. For large-scale reservoir, the P-based correlation allows LMC+LSCH to converge but with relative lower efficiency compared to other algorithms. The W-based correlation developed from Example 1 doesn't work well because LMC+LSCH often diverges during the optimization. The W-based correlation developed from Example 3 performs better than the standard LM algorithms when it is applied to Example 1. However, it is not suitable for solving history matching problems of Example 3. As for the W-based correlation developed from Example 1, it doesn't work for both Examples because LMC+LSCH diverges during the optimization.

Third, the P-based correlation developed from small-scale reservoir is currently the best correlation we developed in this work. It works efficient when dealing with small-scale inverse problems and also allows LMC+LSCH to converge for solving large-scale inverse problems. The W-based correlation developed from Example 3 only works for small-scale reservoir. Compared it to the P-based correlation, it is not as robust as the latter one.

Based on our study, we conclude that the P-based correlation developed from small-scale reservoir is more robust than the other correlations we developed in this work. However, it doesn't perform well always and we do no advise that it should be used for any other problem.

6.3 Recommendation

The initial purpose of this work is to develop a new efficient optimization algorithm that uses DE to estimate damping factor for the LM algorithm. According to the application of the developed algorithms applied on three different reservoir models, we furthermore developed some correlations of damping factor for the LM algorithm. However, the

proposed algorithm is not robust in all cases. In addition, the correlations developed from all the examples did not give us consistent correlation coefficients. We did not obtain a generalized correlation formula. Therefore, the recommendation for the future work is that we may need to find another way to speed up the LM+DE algorithm to further reduce the computation time. Perhaps, we may need to check other parameters that posses influence on the performance of the proposed algorithm.

References

Anterion F., Eymard R., and Karcher B.: "Use of Parameter Gradients for Reservoir History Matching," Paper SPE 18433 presented at the SPE Symposium on Reservoir Simulation, Houston, Texas, USA, 6-8 February 1989

Araneda E.: "A Variation of the Levenberg-Marquardt Method: An Attempt to Improve Efficiency," Master's thesis, Massachusetts Institute of Technology, May 2004

Awotunde A. A., and Horne R. N.: "A Multiresolution Analysis of the Relationship between Spatial Distribution of Reservoir Parameters and Time Distribution of Data Measurements," SPE Annual Technical Conference and Exhibition, Denver, Colorado, USA. 21-24 September, 2008

Awotunde A. A., and Horne R. N.: "An Improved Adjoint Sensitvity Computation for Multiphase Flow Using Wavelets," SPE Annual Technical Conference and Exhibition, Florence, Italy. 19-22 September, 2010

Awotunde A. A.: "Relating Time Series in Data to Spatial Variation in the Reservoir Using Wavelets," Ph.D. dissertation, Stanford University, June 2010a

Awotunde A. A.: "Reservoir Parameter Estimation with Improved Particle Swarm Optimization," Paper was prepared for presentation at the SPE Annual Technical Conference and Exhibition held in Sand Antonio, Texas, USA, vol. 14, 8-10 October 2010b

Awotunde A. A., and Horne R. N.: "A Multiresolution Analysis of the Relationship between Spatial Distribution of Reservoir Parameters and Time Distribution of Well-Test Data," SPE Reservoir Evaluation & Engineering. 2011

Awotunde A. A., and Horne R. N.: "An Improved Adjoint-Sensitvity Computation for Multiphase Flow Using Wavelets," SPE Journal. 2012a

Awotunde A. A.: "Reservoir Parameter Estimation with Improved Particle Swarm Optimization," SPE Annual Technical Conference and Exhibition. Society of Petroleum Engineers, 2012b

Awotunde A. A.: "A Multi-Resolution Adjoint Sensitivity Analysis of Time-lapse Saturation Maps," Computational Geosciences, 2014

Awotunde A. A.: "Estimation of Well Test Parameters Using Global Optimization Techniques," Journal of Petroleum Science and Engineering, 2014

Bissell R.: “Calculating Optimal Parameters for History Matching,” 4th European Conference on the Mathematics of Oil Recovery, 1994

Byer T. J., Edwards M. J., and Aziz K.: “A Preconditioned Adaptive Implicit Method for Reservoirs with Surface Facilities,” SPE Reservoir Simulation Symposium. Houston, Texas, 3-5 February,1998

Brest J., Zamuda A., Boskovic B., Maucec M. S., and Zumer V.: “High-dimensional Real-Parameter Optimization Using Self-Adaptive Differential Evolution Algorithm with Population Size Reduction,” IEEE world congress on computational intelligence, pp. 2032-2039, 2008

Chavent G., Dupuy M., and Lemonnier P.: “History Matching by Use of Optimal Theory,” Old SPE Journal, vol. 15, no. 1, pp. 74-86, 1975

Chiou J. P., and Wang F. S.: “A Hybrid Method of Differential Evolution with Application to Optimal Control Problems of a Bioprocess System,” IEEE Conference on Evolutionary Computation Proceeding. New York, NY, USA, pp. 627-632, 1998

Cheng H., Oyerinde A. S., Datta-Gupta A., and Milliken W. J.: “Compressible Streamlines and Three-phase History Matching,” SPE/DOE Symposium on Improved Oil Recovery, Tulsa, Oklahoma, USA, 22-26, April, 2006

Coello C. A. C., Lamont G. B., and Van Veldhuisen D. A.: “Evolution Algorithms for Solving Multi-Objective Problems,” Springer, 2007

Chen W., Shi Y. Y., and Teng H. F.: “An Improved Differential Evolution with Local Search for Constrained Layout Optimization of Satellite Module,” Advanced Intelligent Computing Theories and Applications. With Aspects of Articial Intelligence, Berlin, pp. 742-749, 2008

Dadashpour M., Landr M., and Kleppe J.: “Nonlinear Inversion for Estimating Reservoir Parameters from Time-lapse Seismic Data,” Journal of Geophysics and Engineering, vol. 5, no. 1, pp. 54-66, 2008

Ertekin T., Abou-Kassem Jamal H., and King G. R.: “Basic Applied Reservoir Simulation,” SPE Textbook Series. 2001

Eydinov D., Gao G., Li G., and Reynolds A.: “Simultaneous Estimation of Relative Permeability and Porosity/Permeability Fields by History Matching Production Data,” Journal of Canadian Petroleum Technology, vol. 48, no. 12, pp. 13-25, 2009

Fasanino G., Molinard J., and Pelce V.: “Inverse Modeling in Gas Reservoirs,” SPE Annual Technical Conference and Exhibition,New Orleans, Texas, 5-8 October 1986

Huang X. R.: “Integration of Production and Time-lapse Seismic Data,” SEG Technical Program Expanded Abstracts, vol. 18, 1999

Hajizadeh Y., Christie M., Demyanov V.: “Application of Differential Evolution as A New Method for Automatic History Matching,” Kuwait International Petroleum Conference and Exhibition. Kuwait City, Kuwait, 14-16 December 2009

Hajizadeh Y., Christie M., and Demyanov V.: “Ant Colony Optimization Algorithm for History Matching,” EUROPEC/EAGE Conference and Exhibition, Amsterdam, Netherlands, 8-11 June 2009

Jacquard P.: “Permeability Distribution from Field Pressure Data,” SPE Journal, vol. 5, no. 4, pp. 281-294, 1965

Kruger W.: "Determining areal Permeability Distribution by Calculation,” Journal of Petroleum Technology, vol. 13, no. 7, pp. 691-696, 1961

Kiyoshi M.: “Pressure Derivative Matching Method for Two Phase Fluid Flow in Heterogeneous Reservoir,” SPE Asia Pacific Conference on Integrated Modelling for Asset Management, Yokohama, Japan. 25-26 April, 2000

Levenberg K.: “A Method for Least-Squares Estimation of Nonlinear Problems”, Quart. Appl. Math, no. 2, pp. 164-168, 1944

Landa J. L., and Horne R. N.: “A Procedure to Integrate Well Test Data, Reservoir Performance History and 4-D Seismic Information into a Reservoir Description,” Paper was prepared for presentation at the 1997 SPE Annual Technical Conference and Exhibition held in San Antonio, Texas, 5-8, October 1997

Landa J. L.: “Integration of 4D Seismic and Production Data for Reservoir Parameter Estimation,” Seg Technical Program Expanded Abstracts, vol. 18, 1999

Li R., Reynolds A. C., and Oliver D. S.: “Sensitivity Coefficients for Three-Phase Flow History Matching,” Canadian International Petroleum Conference, Calgary, Alberta. 12-14 June 2001

Li R., Reynolds A. C., and Oliver D. S.: “History Matching of Three-phase Flow Production Data,” SPE Journal, vol. 8, pp. 328-340, 2003

Lee K. Y., and El-Sharkawi M. A.: “Modern Heuristic Optimization Techniques: Theory and Applications to Power Systems,” John Wiley & Sons, Inc., Hoboken, New Jersey, 2008

Li H., Mirzabozorg A., Hajizadeh Y., Nghiem L., Sousa M. C., and Yang C.: “A Soft and Law-abiding Framework for History Matching and Optimization under Uncertainty,” 2013 SPE Reservoir Simulation Symposium. The Woodlands, TX, USA, 18-20 Febrary 2013

Marquardt D. W.: “An Algorithm for Least-Squares Estimation of Nonlinear Parameters”. Journal of the Society for Industrial & Applied Mathematics, vol. 11, no. 2, pp. 431-441, 1963

Matsui H., and Tanaka K.: “Determination method of an initial damping factor in the damped-least-squares problem,” Applied optics, vol. 33, no. 13, pp. 2411-2418, 1994

Musrrat A., Pant M., and Abraham A.: “Simplex Differential Evolution,” Acta Polytechnica Hungarica. Vol. 6, No. 5, 2009

Montgomery, and Runger: “Applied Statistics and Probability for Engineers,” John Wiley and Sons. 2011

Nocedal P. J., and Wright S.: “Numerical Optimization,” Springer-Veriag, New York, 2006

Neri F., and Tirronen V.: “Recent Advances in Differential Evolution: A Survey and Experimental Analysis,” Artificial Intelligence Review, vol. 33, no. 1-2, pp. 61-106, 2010

Oliver D. S., and Chen Y.: “Recent Progress on Reservoir History Matching: A Review,” Computational Geosciences, vol. 15, no. 1, pp. 185-221, 2011

Portellaand R., and Prais F.: “Use of Automatic History Matching and Geostatistical Simulation to Improve Production Forecast,” Latin American and Caribbean Petroleum Engineering Conference, Caracas, Venezuela, 21-23 April 1999

Rodrigues J. R. P.: “Calculating Derivatives for Automatic History Matching,” Computational Geoscience, vol. 10, no. 1, pp. 119-136, 2006a

Rodrigues J. R. P., Wächter A., Conn A.,Watson T. J., de Moraes R. J., da Silva F. P. T.: “Combining Adjoint Calculations and Quasi-Newton Methods for Automatic History Matching,” SPE Europec/EAGE Annual Conference and Exhibition. Vienna, Austria, 12-15 June 2006b

Rao S. S., and Rao S.: “Engineering Optimization: Theory and Practice,” John Wiley and Sons, 2009

Sun N. Z., and Yeh W. W. G.: “Identication of Parameter Structure in Groundwater Inverse Problem,” Water Resources Research, vol. 21, no. 6, pp. 869-883, 1985

Spall J. C.: “Multivariate Stochastic Approximation Using a Simultaneous Perturbation Gradient Approximation,” Automatic Control, IEEE Transactions on, vol. 37, no. 3, pp. 332-341, 1992

Storn R., and Price K.: “Minimizing the Real Functions of the ICEC'96 Contest by Differential Evolution,” IEEE Conference on Evolutionary Computation, Nagoya, pp. 842-844, 1996

Sun N. Z., Elimelech M., and Ryan J. N.: “Sensitivity Analysis and Parameter Identiability for Colloid Transport in Geochemically Heterogeneous Porous Media,” Water Resources Research, vol. 37, no. 2, pp. 209-222, 2001

Silva P. C., Maschio C., and Schiozer D. G.: “Use of Neural-Simulation Techniques as Proxies to Reservoir Simulator: Application in Production History Matching,” Journal of Petroleum Science and Engineering, vol. 57, pp. 273-280, 2007

Schiozer D., and Ferreira C.: “Time-lapse Seismic and Engineering Data Integration to Estimate Best Time for Seismic Acquisition Data,” 75th EAGE Conference & Exhibition Incorporating SPE EUROPEC 2013, vol. 15, 10-13, June, 2013

Towler B. F. and Killough J. E.: “Comparison of Preconditioners for the Conjugate Gradient Method in Reservoir Simulation,” SPE Reservoir Simulation Symposium, New Orleans, Louisiana, 31 January-3 Feburary, 1982

Tang Y. N., Chen Y. N., Chen W. H., and Wasserman M. H.: “Generalized Pulse-Spectrum Technique for 2-d and 2-phase History Matching,” Applied Numerical Mathematics, vol. 5, no. 6, pp. 529-539, 1989

Tan T. B., and Kalogerakis N.: “A Fully Implicit Three-dimensional Three-phase Simulator with Automatic History-Matching Capability,” SPE Symposium on Reservoir Simulation, Anaheim, California, 17-20 February, 1991

Tan T. B.: “A Computationally Efficient Gauss-Newton Method for Automatic History Matching,” SPE Reservoir Simulation Symposium. San Antonio, Texas, 12-15 February, 1995

Transtrum M. K., and Sethna J. P.: “Improvements to the Levenberg-Marquardt Algorithm for Nonlinear Least-Squares Minimization,” Journal of Computational Physics, 27, January 2012

Vasco D. W., and Datta-Gupta A.: "Integrating Field Production History in Stochastic Reservoir Characterization," SPE Formation Evaluation, vol. 12, pp. 149-156, 1997

Welty D. H., and William C. M.: "Automated History Matching Of Well Tests," SPE Reservoir Simulation Symposium, Denver, Colorado. 31 January-2 Feburary 1979

Yang P. H., and Watson A. T.: "Automatic History Matching With Variable-Metric Methods," SPE Reservoir Engineering. vol. 3, no. 3, pp. 995-1001, 1988

Zhan W., Reynolds A. C., and Oliver D. S.: "Conditioning Geostatistical Models to Two-phase Production Data," SPE Annual Technical Conference and Exhibition, New Orleans, Louisiana. 27-30 September, 1999

Zheng J., Liu Y., Li J., Zhang D., Chen D., and Hong Y.: "Application Of Permeability Predictions In Profile Modification And Water Shutoff Using Genetic Algorithms," SPE Asia Pacific Oil and Gas Conference and Exhibition, Melbourne, Australia, 8-10 October, 2002

Zhang J., and Sanderson A. C.: "DE-AEC: A Differential Evolution Algorithm based on Adaptive Evolution Control," IEEE international conference on evolutionary computation, pp. 3824-3830, 2007

Zhang Y., and Oliver D. S.: "History Matching Using a Hierarchical Stochastic Model with the Ensemble Kalman Filter: A Field Case Study," SPE Reservoir Simulation Symposium. The Woodlands, Texas, USA, 2-4, February 2009

Vitae

Name	: Xian Zhang
Nationality	: China
Date of Birth	:9/20/1985
Email	: wszhangxian@hotmail.com
Area of Interest	: Reservoir Simulation, Reservoir Management
Academic Background	: MS (Petroleum Engineering), January 2015
	BS (Petroleum Engineering), June 2009