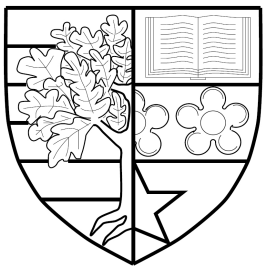
APPLICATION OF MULTILEVEL CONCEPTS FOR UNCERTAINTY QUANTIFICATION IN RESERVOIR SIMULATION

by

Doaa Mostafa Ali Elsakout



Submitted for the degree of
Doctor of Philosophy

School of Energy, Geoscience, Infrastructure and Society
Heriot-Watt University

April 2016

Abstract

Uncertainty quantification is an important task in reservoir simulation and is an active area of research. The main idea of uncertainty quantification is to compute the distribution of a quantity of interest, for example oil rate. That uncertainty, then feeds into the decision making process.

A statistically valid way of quantifying the uncertainty is a Markov Chain Monte Carlo (MCMC) method, such as Random Walk Metropolis (RWM). MCMC is a robust technique for estimating the distribution of the quantity of interest. RWM is can be prohibitively expensive, due to the need to run a huge number of realizations, $45\% - 70\%$ of these may be rejected and, even for a simple reservoir model it may take 15 minutes for each realization. Hamiltonian Monte Carlo accelerates the convergence for RWM but may lead to a large increase computational cost because it requires the gradient.

In this thesis, we present how to use the multilevel concept to accelerate convergence for RWM. The thesis discusses how to apply Multilevel Markov Chain Monte Carlo (MLMCMC) to uncertainty quantification. It proposes two new techniques, one for improving the proxy based on multilevel idea called Multilevel proxy (MLproxy) and the second one for accelerating the convergence of Hamiltonian Monte Carlo is called Multilevel Hamiltonian Monte Carlo (MLHMC).

The idea behind the multilevel concept is a simple telescoping sum: which represents the expensive solution (e.g., estimating the distribution for oil rate on finest grid) in terms of a cheap solution (e.g., estimating the distribution for oil rate on coarse grid) and 'correction terms', which are the difference between the high resolution solution and a low resolution solution. A small fraction of realizations is then run on the finer grids to compute correction terms. This reduces the computational cost and simulation errors significantly.

MLMCMC is a combination between RWM and multilevel concept, it greatly reduces the computational cost compared to the RWM for uncertainty quantification. It makes Monte Carlo estimation a feasible technique for uncertainty quantification in reservoir simulation applications. In this thesis, MLMCMC has been implemented on two reservoir models based on real fields in the central Gulf of Mexico and in North Sea.

MLproxy is another way for decreasing the computational cost based on constructing an emulator and then improving it by adding the correction term between the proxy and simulated results.

MLHMC is a combination of Multilevel Monte Carlo method with a Hamiltonian Monte Carlo algorithm. It accelerates Hamiltonian Monte Carlo (HMC) and is faster than HMC. In the thesis, it has been implemented on a real field called Teal South to assess the uncertainty.

Dedicated to the souls of my parents...

Acknowledgements

I would like to express my deepest gratitude to my supervisors, Prof. Mike Christie and Prof. Gabriel Lord for leaving me alone to study and be independent. Thanks for providing freedom to explore different ideas throughout my PhD journey. Also, I would like to thank the Uncertainty quantification group for improving my presentation skills and giving feedback. I would like to thank Uncertainty quantification sponsors for their useful comments on my work especially, Prof. Jonathan Carter. Furthermore, I would like to thank the examiners for this thesis, Prof Peter King and Dr. James Cruise for their useful comments and incredible feedback to improve the thesis. Moreover, thanks to computer support team and the best magician Jack Talbot for fixing software problem issues during my PhD.

I would like to thank Ali Danesh Scholarship for funding my PhD study at Heriot-Watt University. I would like to thank the African Institute for Mathematical Science for providing funding to conference participation. Thanks to Faculty of Science, Cairo University for giving me a study leave to study PhD.

I would like to thank my best friend Samah Alhafian for supporting me during my PhD study. Also, I would like to thank my friends Laila, Maha, Radiha, Razan, Alyaa, and Sohad for their effort to make me enjoying the time here in Edinburgh during my PhD. Moreover, my friends at Uncertainty quantification group Zainab, Alexandra, Junko and Behzad for supporting me during my study.

Last but not least, I would first like to thank my mother without her continuous support and encouragement I never would have been able to achieve my goals. I dedicate this PhD for my parents' souls.

Contents

List of Tables

List of Figures

Nomenclature

$\mathbb{E}$ Expectation

$\mathbb{V}$ Variance

$\mathcal{N}$ Normal distribution

$\mathcal{U}$ Uniform distribution

CFL Courant-Friedrichs-Lewy

FOPR Field Oil Production Rate

HMC Hamiltonian Monte Carlo

LHS Latin Hypercube Sampling

MCI Monte Carlo Integration

MCMC Markov Chain Monte Carlo

MH Metropolis-Hastings Algorithm

MLHMC Multilevel Hamiltonian Monte Carlo

MLMC Multilevel Monte Carlo

MLMCMC Multilevel Markov Chain Monte Carlo

MLproxy Multilevel proxy

NA Neighbourhood Algorithm

NAB Neighbourhood Algorithm Bayes

PSO Particle Swarm Optimization

RBF Radial Basis Function

RS Rejection Sampling

RWM Random Walk Metropolis

List of Publications from Thesis

Elsakout, D., Christie, M., & Lord, G. (2015). Multilevel Markov Chain Monte Carlo (MLMCMC) For Uncertainty Quantification. Society of Petroleum Engineers. doi:10.2118/175870-MS [presented in SPE North Africa Technical Conference and Exhibition, 14 − 16 September, Cairo]–submitted to SPE journal.

Elsakout, D., Christie, M., & Lord, G. Multilevel Hamiltonian Monte Carlo (MLHMC) for Quantifying Uncertainty in Reservoir Simulation [presented in IMA Conference on Numerical Methods for Simulation, 1 − 4 September 2015, Oxford University and The International Conference for Mathematics and Applications, 27 − 29 December 2015 Cairo, 6th October, Egypt]–In progress for submission to Journal.

Chapter 1

Introduction

The oil industry is a multi-trillion dollar business that has to deal with many sources of uncertainty. The principal source of uncertainty is limited data on rock properties (porosities and permeabilities) that govern flow through the reservoir.

The consequence of uncertainty in reservoir modelling is business risk. Decisions about the reservoir often cost hundred of millions of dollars. The two choices a company has in the face of business risk are either to pay to acquire additional data, or to create a development plan that is robust to risk. Doing this effectively requires a good understanding of uncertainty.

The main idea of modelling the reservoir is to understand the subsurface system. Reservoir simulation solves a system of partial differential equations governing the motion of porous media flow. These equations describe conservation mass, momentum and energy. The output of the code is predictions quantities of interest such as the oil rate. This reservoir simulation needs spatial information about the reservoir such as porosity and permeability. Since these quantities are known precisely at very few points, the reservoir simulator predictions are uncertain.

In order to reduce uncertainty, the model is calibrated to observed production data in a process called history matching. History matching updates the reservoir model until a reasonable match between the observed and simulated data is obtained. Manual history matching, in which quantities are adjusted by hand, is a time consuming trial-error approach. An effective way is assisted history matching using optimization techniques, which removes the routine tasks from the engineer.

History matching is an ill-posed inverse problem, which means that there is no unique solution. Therefore, a good history matched model is not evidence of a robust model for forecasting, which is important for decision making (Kabir and Young, 2004; Tavassoli et al., 2005; Busby et al., 2007a; Carter and White, 2013).

Many techniques have been proposed in the literature for quantifying uncertainty. Recently published algorithms include stochastic optimisers, Randomised Maximum Likelihood and the ensemble Kalman filter. Stochastic optimisers include techniques such as particle swarm optimisation (Mohamed et al., 2010a,b), genetic algorithms (Erbas and Christie, 2007; Carter and Ballester, 2004), evolutionary search strategies (Schulze-Riegert et al., 2001), differential evolution (Storn and Price, 1995, 1997), gradient algorithms (Mohamed et al., 2010a) and neighbourhood algorithm (Sambridge, 1999a; Christie et al., 2002). For any stochastic optimiser (genetic algorithms, particle swarm optimisation), the samples obtained are characteristic of the algorithm, not of any probability distribution quantifying uncertainty. In order to obtain an estimate of uncertainty, an approximation has to be made: in the case of the Neighbourhood Algorithm Bayes sampler (Sambridge, 1999b), the approximation is based on assuming the misfit surface is flat in a Voronoi cell around the sample.

Randomised Maximum Likelihood (Chen and Oliver, 2012) combines gradient techniques with a stochastic resampling of measured data to obtain an algorithm that quantifies uncertainty. The main problem with using gradient techniques is that they can easily get trapped in local minima and fail to locate the global minimum. However, Randomised Maximum Likelihood fixed the problem of trapping by using stochastic sampling. The Ensemble Kalman filter (Evensen, 2007) uses an ensemble of solutions and assimilates one measured data point at a time to get good agreement. The Ensemble Kalman filter is exact if the problem is linear and Gaussian. For nonlinear problems, it sometimes suffers from ensemble collapse.

The ultimate aim of quantifying uncertainty is to have a posterior distribution for each uncertain parameter and a posterior distribution for the quantity of interest e.g. oil rate. The posterior distribution is based on the prior knowledge about the

reservoir and the description of the measurement errors and has all the information about the reservoir. The accuracy of uncertainty estimation for the posterior distribution is affected by prior knowledge, the description of the measurement errors and stochastic sampling. In oil production, we are primarily interested in the uncertainty in produced quantities rather than the values of the parameters (porosity and permeability).

There are two rigorous statistical methods for sampling from an arbitrary probability distribution: rejection sampling, and Markov Chain Monte Carlo (MCMC) (Robert and Casella, 1999). In rejection sampling (Neal, 1993), an approximation to the desired probability distribution is chosen, and points are sampled from that distribution, and then accepted or rejected according to whether they are below or above the desired probability distribution. The difficulty in applying rejection sampling is that, for many distributions, the efficiency of the sampling is low. However, any valid sample generated by rejection sampling is a valid sample from the desired distribution (Gilks et al., 1996). In MCMC, a Markov Chain is created which samples from given probability distribution. A Markov Chain is constructed such that it has the desired distribution as the equilibrium distribution (probability distribution is invariant with respect to moving between two consecutive times) of the Markov Chain. Any finite number of samples from the Markov Chain are then an approximation to the desired distribution, with the sampling error related to the length of the chain.

A common MCMC algorithm is random walk Metropolis (Metropolis et al., 1953). The principal difficulty in applying random walk Metropolis method to quantify uncertainty in reservoir simulation is that the number of samples needed is often large and with expensive flow simulations, the cost is prohibitive. One way of reducing cost is generating a proxy model that gives an approximation to the flow simulator response, yet runs orders of magnitude more quickly (Goodwin and Powell, 2012). However, the results obtained using this technique depends on the quality of the proxy model.

Because prediction of uncertainty quantification is an important task, it is impor-

tant to find fast, efficient methods to quantify uncertainty using MCMC algorithms.

1.1 Thesis Objectives and Statement

The main aim of the thesis is to demonstrate an application of the multilevel concept for uncertainty quantification and to develop reliable techniques based on stochastic sampling. Figure 1.1 shows how the thesis contributions fits into the field of uncertainty quantification.

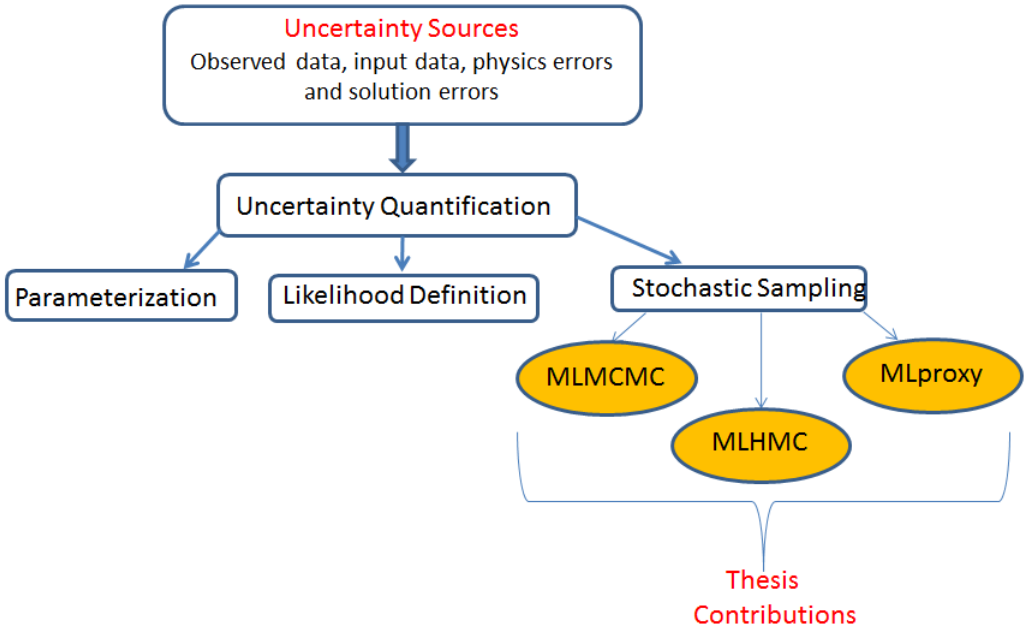


Figure 1.1: Thesis contribution to the field of uncertainty quantification.

The thesis demonstrates an application of a new technique which is called Multilevel Markov Chain Monte Carlo (MLMCMC) for quantifying uncertainty in reservoir simulation. This technique avoids the approximations involved in proxy methods by always running the flow simulation. It gains speed by decomposing the desired results into a component calculated with a coarse model, and corrections obtained on a sequence of finer models.

Another thesis objective is to combine the multilevel concept with Hamiltonian Monte Carlo. The thesis compares the new techniques with other methods such as particle swarm optimisation with neighbourhood algorithm Bayes, Hamiltonian Monte Carlo and random walk Metropolis. Multilevel Hamiltonian Monte Carlo

(MLHMC) successfully improves the convergence rate and decreases the computational cost for uncertainty quantification without loss of efficiency.

The thesis proposes a new technique for improving a proxy by adding the error model term between the simulated and the proxy result. It is a combination of MLMC idea, constructing the proxy and two-stage MCMC method (Efendiev et al., 2005).

Overall, we develop three ways that uncertainty quantify more efficiently than random walk Metropolis or Hamiltonian Monte Carlo. We achieve a estimate of probability distributions as efficiently as Hamiltonian Monte Carlo, but in much less time.

The thesis employs the Bayesian framework to quantify uncertainty. A Bayesian framework is a statistical framework used Bayes' theorem to update the belief related to reservoir as a distribution for each uncertain parameter. Bayes' theorem is a way of understanding how the probability that a theory is true is affected by a new piece of evidence.

1.2 Thesis Outline

The thesis is structured as follows:

Chapter 2 reviews the mathematical equations which govern the flow motion and statistics. It discusses the numerical reservoir simulation, the errors in the modelling and uncertainty quantification for history matching and forecasting. Finally, the chapter reviews the Bayes' theorem and objective function.

Chapter 3 reviews and discusses different stochastic algorithms, e.g., neighbourhood algorithm, particle swarm optimisation, neighbourhood algorithm Bayes, rejection sampling, Markov Chain Monte Carlo and Gibbs sampler. The chapter discusses the advantages and disadvantages of Markov Chain Monte Carlo methods.

Chapter 4 focusses on finding the solutions of the advection equation, Buckley-Leverett equation and the pressure equation. It discusses how to find the exact solution, when it is possible. Moreover, it studies the numerical solution using different numerical schemes and different initial conditions, it discusses the stability

and determine the stability conditions and analyse the behaviour of a number of finite difference schemes for solving differential equations governing the miscible displacement.

Chapter 5 introduces the major concepts of the Monte Carlo Integration and the Multilevel Monte Carlo method, and compares the two techniques on 4 different problems: the exponential growth and decay equation as a toy example equipped with a random initial condition, the advection equation with a random initial condition, the Buckley-Leverett equation with random flux function and random initial condition and the pressure equation with a random diffusion coefficient.

Chapter 6 demonstrates an application of a new technique, Multilevel Markov Chain Monte Carlo (MLMCMC), for quantifying uncertainty in reservoir simulations. It applies MLMCMC to solve multi-phase flow and show results for two fields. The first is Teal South in the Gulf of Mexico and the second is Scapa in the UK North Sea. In addition, the chapter reviews random walk Metropolis, how to analyse the output result and how to use the sensitivity analysis for the reservoir model.

Chapter 7 proposes a new approach, based on the multilevel concept, for improving the proxy to increase the confidence if we use it for inference, with less computational cost. Moreover, it reviews some experimental design techniques and discusses Radial Basis Function (RBF), how to build a proxy using experimental design with RBF and how to use this proxy to construct an error model.

Chapter 8 presents a new technique for uncertainty quantification and accelerates the convergence of random walk Metropolis called Multilevel Hamiltonian Monte Carlo. The technique is tested on Teal South model to assess uncertainty in the oil rate. The chapter compares MLMCMC and Hamiltonian Monte Carlo.

Chapter 9 concludes the thesis contributions, major findings and suggests recommendations for future research.

In addition to main chapters,two appendices include the classification of the first and second partial differential equations based on the eigenvalues and the proof of detailed balance for Metropolis-Hastings, Hamiltonian Monte Carlo and MLMCMC.

Chapter 2

Background Material

This chapter provides the foundation of the thesis. The chapter reviews some basic background material from petroleum engineering and statistics. It starts by reviewing the mathematical equations, which govern the flow in a porous media flow. Following this, it discusses the numerical reservoir simulation and the errors in the modelling. After that, it discusses uncertainty quantification for history matching and forecasting. Finally, it reviews the Bayes' theorem and objective function.

2.1 Mathematics of a Flow in Porous Media Flow

The generic conservation equation for a system has the following form,

$$\frac{\partial \mathbf{u}}{\partial t} + \bigtriangledown . F(\mathbf{u}) = 0, \tag{2.1}$$

where, $\mathbf{u}(\mathbf{x}, t)$ is mass, energy, momentum or saturation and the flux function is F. Equation (2.1) is augmented with some initial and boundary conditions. The integral form for the conservation law (2.1) is valid even for a discontinuous solution. One of the applications of (2.2) is in petroleum engineering.

$$\frac{\mathrm{d}}{\mathrm{dt}} \int_a^b u(x,t)\mathrm{dx} = F(u(b,t)) - F(u(a,t)). \tag{2.2}$$

Porous media flow has the conservation of mass and the conservation of momentum is replaced by Darcy's law, conservation of energy (isothermal equation) and equation of state (Trangenstein, 1986; Carter, 2010).

2.1.1 Darcy's Law

Darcy's law describes the motion of a fluid through a porous medium. The law was formulated by Henry Darcy in 1856 (Darcy, 1856) based on the results of experiments on the flow of water through beds of sand under homogeneous incompressible flow. Darcy determined experimentally the flow rate (Hubbert, 1957). Its a slow flow approximation to conservation of momentum in Navier−Stokes. Darcy's law is commonly used to describe oil, water, and gas flows through petroleum reservoirs (reservoir is a body of underground rocks that contains a mixture of hydrocarbon fluid and water trapped in porous rocks). Figure 2.1 shows a graphical representation of Darcy's Law.

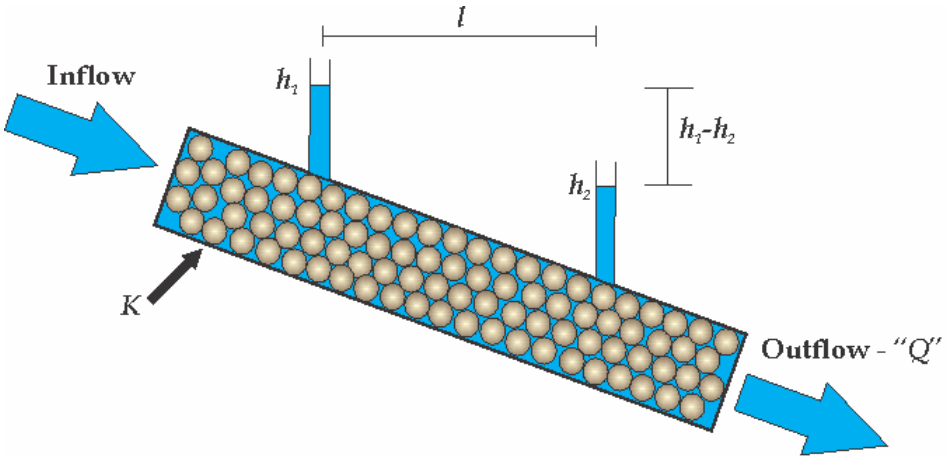


Figure 2.1: Graphical representation of Darcy's Law (Matt Herod, 2011).

Darcy's law is expressed for one phase flow (Hubbert, 1957) by the formula:

$$\mathbf{Q} = KA\frac{h_1 - h_2}{l}, \tag{2.3}$$

where, $\mathbf{Q}$ is the flow rate and its unit is $[L^3/T]$, A is the cross section area to flow, h_1 and h_2 are the heights from the reference level of the water above and below the sand respectively, l is the horizontal distance between h_1 and h_2, $\frac{h_1 - h_2}{l}$ is called

hydraulic gradient, and K is hydraulic conductivity (its unit is $[L/T]$) which equals to,

$$K = \kappa\rho g/\mu, \tag{2.4}$$

where, κ is the medium permeability, ρ is the water density, g is the gravity and μ is the viscosity of the fluid. Moreover the pressure can be written as follows,

$$P = \rho g h. \tag{2.5}$$

Substituting with (2.5) and (2.4) into (2.3), Darcy's law can be written as follows,

$$\mathbf{v} = \mathbf{Q}/\phi(x)A = -\frac{\kappa(x)}{\mu\phi(x)} \bigtriangledown P, \tag{2.6}$$

where $\mathbf{v}$ is vector flow rate per unit area, ϕ is the porosity, μ is the viscosity, κ is the permeability of the medium and P is the pressure. There is a negative sign in the formula because the flow is from high to low pressure.

Assuming the medium is isotropic (identical in all directions) and the permeability is uniform, then Darcy's law (Hubbert, 1957) can be written as follows,

$$\mathbf{v} = -\frac{\kappa(x)}{\mu\phi(x)}(\bigtriangledown P - \rho\mathbf{g}).$$

2.2 Reservoir Simulation

Reservoir simulation is a powerful tool for the management and forecasting of the reservoir. Reservoir simulation is the process of solving the conservation equations on a grid and computing oil, water and gas rates. The equations governing the flow are nonlinear because the fluid properties such as viscosity and density depend on the pressure. Reservoir simulation solves discrete, nonlinear equations, and uses Newton Raphson method to approximate the solution by linearizing the system of equations governing the flow (Pettersen, 2006; Farmer, 2005).

The requirements for the reservoir simulation can be summarised as follows,

- Dimensions for each grid block, length, width and thickness

- Parameters for each grid block e.g., porosity, permeability, saturation, pressure and fluid (oil, gas and water)
- Fluid parameters for e.g., viscosities, densities, compressibilities and formation volume factors
- Well data for e.g., location, production or injection rate and limitation

Reservoir simulation is a hard process in terms of computational performance and it is time consuming, based on the model size and the number of time steps.

Most commercial codes for reservoir simulation use finite volume methods (see (Leveque, 1992) for overview of finite volume methods). On any finite grid there are two sources of uncertainties: lack of knowledge of reservoir properties (e.g., porosity and permeability) and discretization error. Usually, lack of knowledge of reservoir properties is a more significant overall uncertainty, except in the case of every coarse grids (SPE10 (Christie and Blunt, 2001)).

To quantify uncertainty involves running many simulations. To avoid the coarsest grid error shown in (Christie and Blunt, 2001), we have to finer grids, which increases computational cost dramatically (Farmer, 2005).

The goal of this thesis is to reduce the costs associated with the fine grids simulations as mentioned in Chapter 1.

2.3 Deterministic Versus Stochastic Modelling

A deterministic model is a model with known input data initial and boundary conditions and it does not involve randomness. If the initial condition or any parameter or even the unknown variable is randomly distributed, then we call the model stochastic. For example, in the Buckley-Leverett equation, when the permeabilities are uncertain (see Section 4.4). Hence, stochastic modelling becomes an important area of applied mathematics. For examples, financial mathematics (Asmussen and Glynn, 2007), biochemical reactions (Anderson and Higham, 2012), plasma physics (Rosin et al., 2014) and uncertainty quantification in engineering and science (Christie et al., 2006; Carter and White, 2013; Floris and Peersmann, 1998; Oliver et al., 2008).

2.3.1 Fractional Flow in a Stochastic Setting

The fractional flow function depends on the relative permeability. Relative permeability is measured in a laboratory using core samples. The measurements depend on lab conditions and are uncertain because different core samples have different measured relative permeability. Therefore, there is uncertainty in relative permeability and the fractional flow as well. This uncertainty affects the reservoir performance.

Different analytical models have been used for estimating the relative permeability of a two-phase flow. For example, the modified Brooks and Corey **MBC** model (Alpak and Lake, 1999), is an expression that is able to fit most experimental data for water and oil flow:

$$\kappa_{rw}(S) = \kappa_{rw}^0\Big(\frac{S - S_{w_c}}{1 - S_{w_c} - S_{o_c}}\Big)^{n_1} \tag{2.7a}$$

$$\kappa_{ro}(S) = \kappa_{ro}^0\Big(\frac{1 - S - S_{o_c}}{1 - S_{w_c} - S_{o_c}}\Big)^{n_2}, \tag{2.7b}$$

where, S water saturation, n_1, n_2 are constants, κ_{rw}^0, κ_{ro}^0 are endpoint relative permeabilities for water and oil respectively and S_{o_c}, S_{w_c} are critical oil saturation and connate water saturation. The fractional flow F in the reservoir is defined by,

$$F = \frac{\kappa_{rw}/\mu_w}{\kappa_{rw}/\mu_w + \kappa_{ro}/\mu_o}, \tag{2.8}$$

where μ_w is water viscosity and μ_o is oil viscosity. Substituting $S^* = \dfrac{S - S_{w_c}}{1 - S_{w_c} - S_{o_c}}$ into (2.7) and then substituting from (2.7) into (2.8) with $\dfrac{\mu_w \kappa_{ro}^0}{\mu_o \kappa_{rw}^0} = B$ then, we obtain the following formula for the fractional flow,

$$F = \frac{(S^*)^{n_1}}{(S^*)^{n_1} + B(1 - S^*)^{n_2}}.$$

If we are calibrating a model, we can use n_1, n_2 and B are unknown parameters (as in Chapter 5).

2.3.2 Uncertainty Sources in Reservoir Simulations

How to assess the reliability of the reservoir model? A reliable reservoir model is able to predict the unknown future performance of the reservoir with a specified degree of accuracy.

The sources of uncertainty arise from three types of input data: the spatially varying rock properties; the fluid properties; and the rock-fluid interaction properties such as relative permeability and capillary pressure. The spatially varying rock properties are the biggest source of uncertainty, as data is taken at a small number of wells (order 10s), covering a minute fraction of the reservoir volume. We are able to apply loose constraints on the properties through use of outcrop data.

2.4 An Inverse Problem

There are two problems to deal with in reservoir simulation: forward and inverse problems. The forward problem can be described mathematically as a map from reservoir properties to production data,

$$\text{Production} = f(m),$$

where f is known function and m are the parameters. The forward problem, contains all the properties (all the physics of the situation) of the reservoir and it is straightforward to solve (Dadashpour, 2009).

An inverse problem is defined as follows: Given the production data, can we find the input parameters corresponding to these observed data corrupted by noise. In reality, we do not have the whole information about the reservoir, hence, the parameters m are unknown. An inverse problem can be described mathematically by

$$\mathcal{D} = g(m) + \text{noise}, \tag{2.9}$$

where $\mathcal{D}$ is the observed data, g is known function, m are the unknown parameters and noise is the error measurements. An ill-posed inverse problem does not have a unique solution m, which satisfies (2.9) (Hadamard, 1902).

Because the reservoir properties are generally unknown, we have to solve an inverse problem to determine them.

Possible goals of an inverse problem as in (Richardson and Zandt, 2009) are:

- Estimating the model parameters.
- Estimating the range for each parameter.
- Studying the sensitivity of the model to the data.
- Uncertainty quantification to predict the quantity of interest.

History matching is an example of an inverse problem (Suzuki and Caers, 2006). Forecasting oil production depends on solving the inverse and forward problems (Glimm and Sharp, 1999).

2.4.1 History Matching

Traditionally, the goal of history matching is to find a reservoir model that reproduces the observed data of the field (Floris et al., 2001). The process of tuning the model until obtaining a reasonable match between the observed data and simulated data is referred to as history matching (Oliver et al., 2008). We minimize the discrepancy between the observed data and simulated data by changing the parameter values until we get the best match.

There are two ways of history matching: manual (traditional which uses trial and error) and assisted history matching. Manual history matching provides a single forecast. Assisted history matching (automatic) reduces the manual effort by the specialist to obtain a reservoir model consistent with the observed data.

Generally, any of the reservoir properties can be adjusted in the history matching. For example, initial saturation is not directly measurable at every location in the reservoir. It can be measured at the sparse locations of wells and locations in between the values can be interpolated.

There has been about 55 years of research proposing different methods for assisted history matching. In assisted history matching, the algorithm searches for a

valid parameter set, which has a good match with the observed data. In these frameworks, stochastic optimization algorithms become one of the techniques for studying history matching problems. Figure 2.2 shows the 55 years of history matching algorithms adapted from (Hajizadeh, 2011). In Figure 2.2, Multilevel Markov Chain Monte Carlo (MLMCMC) and Multilevel Hamiltonian Monte Carlo (MLHMC) are the main contributions of this thesis.

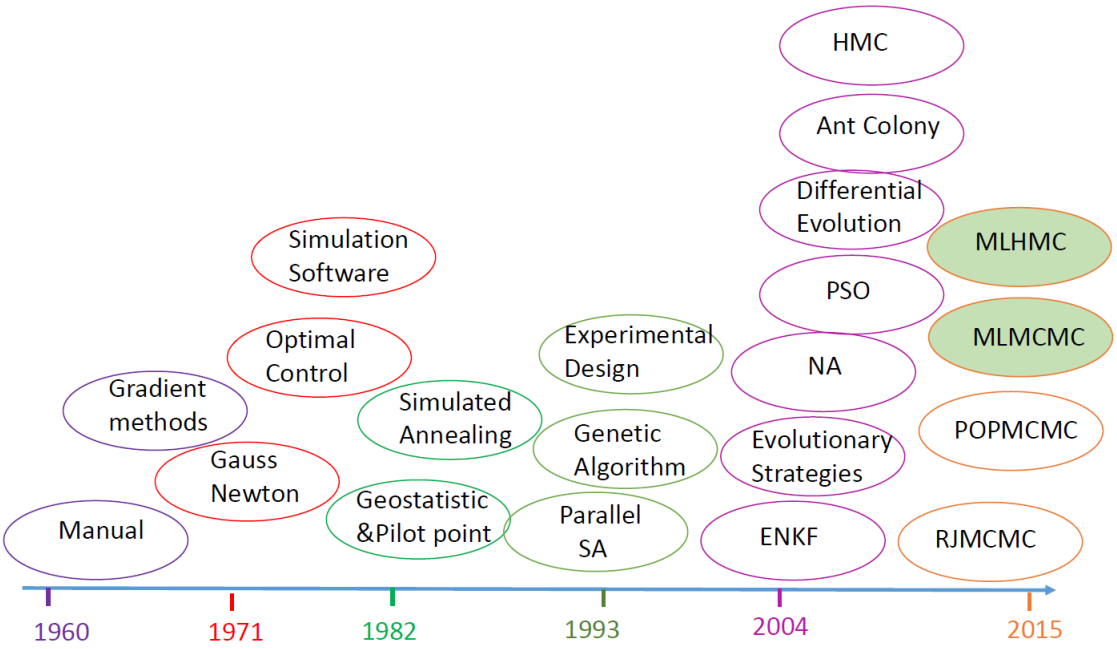


Figure 2.2: 55 years of history matching.

In a history matching study, our ultimate goal is to be able to update the reservoir model in such a way, that it is able to predict the future reasonably (Oliver et al., 2008). Figure 2.3 shows a history matching workflow.

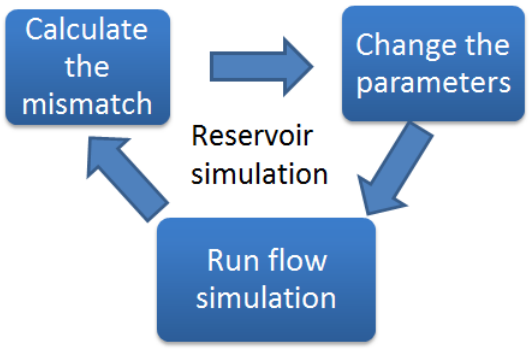


Figure 2.3: History matching workflow.

We expect a simulation model, which is able to capture the past life of the reservoir is the most likely to capture the future, meaning it can make confident predictions. However, sometimes the worst forecasting is based on the best history matched model (TNO, 2003; Tavassoli et al., 2005).

2.5 Uncertainty Quantification

There are several sources of uncertainty in reservoir simulation, which make it difficult to get confident predictions. Incorporating all the uncertainty sources is the main issue with estimating the uncertainty in the prediction of any quantity of interest. Uncertainty has a direct impact on the decision making process in reservoir forecasting (Begg et al., 2001). Based on estimating the uncertainty of the prediction, we can assess the decision risks. In other words, providing information to evaluate the risks of making decisions is a key objective in uncertainty quantification.

The complexity of the model and the method for uncertainty quantification should be determined based on the question that need to be answered. Normally, we start with the simplest method and model it, then decide whether it is suitable for the decision being made. However, for a particular problem, deciding what is appropriate can be unclear and difficult (Williams et al., 2004).

To estimate the uncertainty distribution from the history matching models, there are three different ways. One way is to use a single (best fitting) model which has the minimum error corresponding to observed data (Oliver, 1996). The second way is to use a subset of history matching models–for example from randomized maximum likelihood (Oliver et al., 1996). The third way is to use all the models, the Markov Chain Monte Carlo method (Behrenbruch et al., 1985). Markov Chain Monte Carlo algorithms which sample from a probability distribution by constructing a Markov chain, which is a sequence of random variables where the probability for the next state depends only the current one that has the desired distribution as its equilibrium distribution (Gilks et al., 1996).

2.5.1 Why has Probability Theory been Used for Uncertainty quantification?

Our aim is to find a confident forecast for the quantity of interest such as oil rate based on finding accurate parameter values for reservoir properties. Since the problem is ill-posed, it has multiple solutions. Therefore, it is better to use probability theory to estimate the distribution of the quantity of interest. Moreover, history matching based on one realization is almost certain to predict oil rate incorrectly– we just do not know by how much. Therefore, an ensemble of realizations has been used, which leads to increase in CPU time. Decreasing the CPU time is one of the research problems in the research area and it is a goal of this thesis.

The thesis uses a Bayesian framework. A Bayesian framework is a statistical framework using Bayes' theorem to update the beliefs about the unknown reservoir parameters. The Bayesian approach calculates the posterior distribution of the properties given the observed production data.

2.5.2 Bayesian Inference

Bayesian approach has several advantages, one being that insufficient data can be represented easily as probability distributions (Sivia, 1996).

2.5.2.1 Bayes' Theorem

Bayes' theorem is given by

$$\mathbb{P}(m|\mathcal{D}) = \frac{\mathbb{P}(\mathcal{D}|m)\mathbb{P}(m)}{\mathbb{P}(\mathcal{D})}, \tag{2.10}$$

where the posterior density is $\mathbb{P}(m|\mathcal{D})$, the likelihood is $\mathbb{P}(\mathcal{D}|m)$ and $\mathbb{P}(m)$ is the prior density. The denominator of (2.10) is the normalization constant and can be defined by

$$\mathbb{P}(\mathcal{D}) = \int_{\mathcal{M}} \mathbb{P}(\mathcal{D}|m)\mathbb{P}(m)\mathrm{dm}$$

where $\mathcal{M}$ is the parameter space. Bayes' theorem is a useful tool because it links the posterior density and the likelihood, which can be calculated (Sivia, 1996).

The prior distribution describes the beliefs of reservoir knowledge that we had before knowing the observed data. Estimating the prior distribution is much easier than guessing the best setting of the reservoir parameters. The choice of the prior depends on geological descriptions and geological data for the reservoir. However, since there is rarely enough data to swamp the prior. An inadequate prior distribution could be a reason for an inaccurate uncertainty estimation.

The likelihood interoperates the observed data. It is a joint probability density of the observed data given the model. It is not a probability density function for parameter m, but it assumes the parameters m are true and assesses the likelihood that the observed data is consistent with the model. We often assume measurement errors are Gaussian, independent and identically distributed at any time t, which leads to a least square formulation.

The posterior distribution is a probability distribution for the model m, as constrained by the observed data $\mathcal{D}$ (Glimm et al., 2004), which updates our beliefs about the set of models, given by some observed data.

The normalization constant can be used to compare between different reservoir models. If a model has a small normalizing constant, the model does not fit well. The normalization constant, can be extremely hard to calculate because it involves high dimensional integration.

Figure 2.4 shows the difference between the prior density, the likelihood and the posterior density is included. In the beginning, the prior is wide and once the observed data is included, described by likelihood, then the belief becomes more confident.

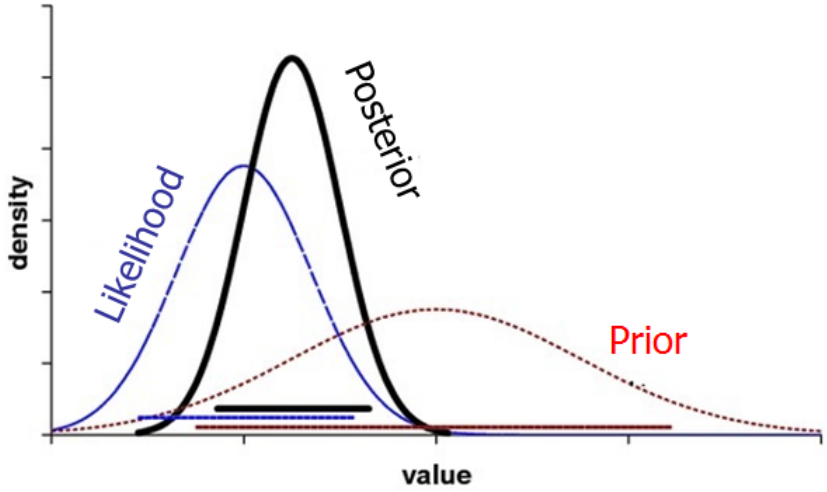


Figure 2.4: Comparison between prior and posterior densities.

In the maximum likelihood estimates, the parameter $\hat{m}$, in the presence of the data given, is the maximising the likelihood.

2.6 Objective Function

As we mentioned, the goal of history matching is to obtain a model such that the discrepancy between the observed data and simulated data is minimized. The discrepancy is defined as the difference between the observed and simulated data. The misfit is defined as negative Log likelihood, and if we assume independent Gaussian errors is given by

$$\text{misfit} = \sum_{j=1}^{N} \frac{(q_{t_j}^o - q_{t_j}^s)^2}{2\sigma_{t_j}^2} \tag{2.11}$$

where $q_{t_j}^o$ and $q_{t_j}^s$ refer to observed data and simulated data respectively at time t_j, N refers to the number of data points and σ_{t_j} is the standard deviation of the observed data. There are different algorithms used to minimize the misfit, for example, particle swarm optimization, which will be discussed in Chapter 3.

2.6.0.2 How to Estimate σ_{t_j}

The misfit definition has a value of σ_{t_j} that must be estimated. There are several ways to do that. One way, the engineer provides values for a maximum likelihood

estimate for σ_{t_j}, assuming σ_{t_j} does not depend on time, i.e. $\sigma_{t_j} = \sigma$. Under these assumptions, the likelihood can be defined by,

$$\mathcal{L}(\mu, \sigma) = \frac{1}{(2\pi)^{N/2}\sigma^N} \exp(-\frac{1}{2\sigma^2} \sum_{j=1}^{N} (q_{t_j}^o - q_{t_j}^s)^2). \tag{2.12}$$

Taking logarithms of both sides, we get,

$$\log \mathcal{L} = -N \log \sigma - \frac{N}{2} \log(2\pi) - \frac{1}{2\sigma^2} \sum_{j=1}^{N} (q_{t_j}^o - q_{t_j}^s)^2. \tag{2.13}$$

To estimate σ, we differentiate equation (2.13) with respect to σ,

$$\frac{\partial \log \mathcal{L}}{\partial \sigma} = -\frac{N}{\sigma} + \frac{1}{\sigma^3} \sum_{j=1}^{N} (q_{t_j}^o - q_{t_j}^s)^2.$$

Setting $\dfrac{\partial \log \mathcal{L}}{\partial \sigma} = 0$, we obtain the following,

$$N\sigma^2 = \sum_{j=1}^{N} (q_{t_j}^o - q_{t_j}^s)^2$$

$$\sigma_{\text{opt}} = \sqrt{\frac{\sum_{j=1}^{N} (q_{t_j}^o - q_{t_j}^s)^2}{N}}.$$

where, q^s is the maximum likelihood simulated result, σ_{opt} is a biased estimator for σ. If we use $\sigma^2 = \dfrac{1}{N} \sum_{i=1}^{N} (q_{t_j}^o - q_{t_j}^s)^2$, we get the minimum misfit of order $N/2$. However, the unbiased estimator for σ is

$$\sigma^2 = \frac{1}{N-1} \sum_{i=1}^{N} (q_{t_j}^o - q_{t_j}^s)^2$$

If the simulated data are time dependent, then Equation (2.13) becomes

$$\log \mathcal{L} = -\sum_{j=1}^{N} \log \sigma_{t_j} - \frac{N}{2} \log(2\pi) - \frac{1}{2} \sum_{j=1}^{N} \frac{(q_{t_j}^o - q_{t_j}^s)^2}{\sigma_{t_j}^2}, \tag{2.14}$$

which we minimise for each σ_{t_j} to get

$$\sigma_{t_j} = (q^o_{t_j} - q^s_{t_j}).$$

In the next chapter, we will discuss different sampling algorithms. Some of them are based on optimizations and others are based on Bayesian framework.

Chapter 3

Stochastic Algorithms–Literature

The aim of this chapter is to review different stochastic algorithms for uncertainty quantification, e.g., Neighbourhood Algorithm, Particle Swarm Optimisation, Neighbourhood Algorithm Bayes, Rejection Sampling, Markov Chain Monte Carlo and Gibbs sampler.

There are many sampling algorithms that have been used in the petroleum literature to generate calibrated models and for uncertainty quantification. Some of the algorithms used stochastic optimisation and others are based on Bayesian inference. Optimisation methods include Particle Swarm Optimisation (PSO) (Mohamed et al., 2010a,b), Genetic Algorithms (GA) (Erbas and Christie, 2007; Carter and Ballester, 2004), Evolutionary search strategies (Schulze-Riegert et al., 2001), Differential Evolution (Storn and Price, 1995, 1997), gradient algorithms (Mohamed et al., 2010a) and Neighbourhood Algorithm (NA) (Sambridge, 1999a; Christie et al., 2002).

Stochastic algorithms generate uncertainty well matched models as the number of iterations increases. However, they do not sample fro many specific probability distribution and additional calculations are needed to assess uncertainty. The NAB sampler (Neighbourhood Algorithm Bayes) is an example of resampling algorithm based on Gibbs sampler that computes appropriate probability (Sambridge, 1999a,b).

On the other hand, the Bayesian approach, e.g., the Markov Chain Monte Carlo (MCMC) family approaches, such as the Metropolis Algorithm and Rejection Sampling (RS) produces a natural way for estimating posterior distributions for reservoir

properties.

3.1 Optimisation Algorithms

In this section, we review two different optimisation techniques: Neighbourhood Algorithm and Particle Swarm Optimisation (PSO) because we will compare a new result with PSO combined with Neighbourhood Algorithm Bayes (NAB).

3.1.1 Neighbourhood Algorithm (NA)

The Neighbourhood Algorithm was developed in (Sambridge, 1999a). The idea of this algorithm is to find a way of obtaining multiple acceptable models rather than finding a single solution. The key assumption in the Neighbourhood Algorithm is that the misfit is constant in the voroni cell around any given model.

To quantify the uncertainty using the NA, we have two stages, the search stage, in which we have the acceptable models for the inverse problem, and the appraisal stage (Sambridge, 1999b). The NA has two parameters: n_s, which is the number of models generated at each stage, and n_r, which is the number of models selected for refinement. The ratio n_s/n_r control the algorithm behaviour from more explorative $n_r = n_s$ to more exploitive $n_r = 1$.

The search stage procedure can be summarised as follows,

1. Initialize with a random set of n_s models and calculate the misfit for each model (2.11).

2. Select the n_r models with the lowest misfit values and generate n_s models randomly in those n_r cells.

3. Returns to Step 3 and repeat the processes until reaching the number of iterations, which the user defined it.

The NA is applicable in the petroleum industry (Stephen et al., 2005; Subbey et al., 2004) as it is efficient in terms of exploring the parameter space. We can use this algorithm to estimate the uncertainty envelopes $P10$, $P50$ and $P90$.

3.1.2 Particle Swarm Optimisation (PSO)

Particle Swarm optimisation was introduced by Kennedy and Eberhart in 1995 (Kennedy and Eberhart, 1995). PSO was inspired by the social behaviour of a flock of birds (Bratton and Kennedy, 2007). The main idea is to search for the global minimum of a function $f : \mathbb{R}^d \to \mathbb{R}$, which means: find $a \in \mathbb{R}^d : f(a) \leq f(b), \forall\, b \in \mathbb{R}^d$. PSO simulates the social behaviour of a group of individuals by sharing information between them while they are exploring the parameter space. Initial models refer to particles, with each particle having its own memory, which stores the best location seen by a particle. The swarm also has its own collective memory which records the best location by any of the particles (Chun, 2010). The initial values for the parameters in the parameter space are random. The dynamic process for PSO has two operations, updating the velocity of each particle, and then updating the position of each particle. Algorithm 1 summarises PSO.

Algorithm 1: PSO (Christie et al., 2011)

1: Initialize the original position (parameter space) and velocity for each particle randomly.
2: For each particle, solve and calculate the misfit value M.
3: Evaluate the fitness of individual particles at each iteration.
4: For updating the best position p_{best}, if the current fitness value of one particle is better than the old value, update the old one with the current one.

$$\text{if}\,(p^k < p^k_{\text{best}}) : p^k_{\text{best}} = p^k$$

5: The current global best fitness value and the corresponding best position must be found and updated if is required.
6: Update the velocity of each particle.

$$v_i^{k+1} = \omega v_i^k + c_1 r_1 (p^k_{\text{best}} - x_i^k) + c_2 r_2 (g^k_{\text{best}} - x_i^k)$$

where, ω is the inertia of the particles, controlling the impact of the previous velocities on the current one. $r_1\, r_2 \sim U(0,1)$, c_1, c_2 are cognitive and social components, representing the particle's attraction towards to local best and global best.
7: Update the position of each particle using equation

$$x_i^{k+1} = x_i^k + v_i^{k+1}$$

8: Repeat steps 3 to 7 until the stopping criterion is achieved.

The choice of the parameters ω, c_1 and c_2 is an active research area. Some ap-

plications of PSO have shown that ω can be decreased linearly from 0.8 to 0.4 (Mohamed et al., 2010b,a). Other approaches about how to tune the parameters can be found in (Engelbrecht, 2005). In the literature, there are different versions of the PSO algorithm (Mu et al., 2009; Settles, 2005; Mohamed et al., 2010b; Bratton and Kennedy, 2007).

3.1.2.1 PSO Advantages

- Easy to implement and fast compared with other stochastic algorithms (Mohamed et al., 2010b).
- The derivative of the function is not required.
- It is a heuristic (the particles learn from each other) method.
- It is an effective optimisation algorithm for different applications.
- It is popular for history matching because it is easy to of parallelise.
- On average, PSO reduces the misfit in each generation more rapidly than the NA.

3.2 Neighbourhood Algorithm Bayes (NAB)

NAB is a stochastic sampling algorithm that was developed by (Sambridge, 1999b) in order to correct for the unknown sampling distribution from a stochastic algorithm. The main idea of NAB is to divide the parameter space into Voronoi cells, where the misfit is assumed to be constant in each cell. NAB is the appraisal stage of NA. Sambridge used a Gibbs sampler to estimate the posterior distribution. The critical observation that makes NAB works is that you only need to construct 1-D sections through the voroni cells. Once each model has an associated posterior probability, we can estimate $P10$, $P50, P90$. For more details about NAB see (Sambridge, 1999b).

How NAB works

1. Start with a random sample x_1 as shown in Figure 3.1 and draw a horizontal line through x_1

2. Propose a new location x_{new} on the horizontal line, and determine associated misfit M_{new}

3. Find minimum misfit M_{min} along that line as shown in Figure 3.1

4. Generate $\alpha \sim \mathcal{U}(0, 1)$

5. The accept criterion for the new point is

$$\alpha \leq \min\{1, \exp(M_{\text{min}} - M_{\text{new}})\}.$$

   otherwise go to 2

6. Cycle $(2) - (5)$ through dimensions

7. Based on the result, the posterior distribution can be estimated from the frequency of the resampled models, then we can run a few forecast simulations based on the resampled models. At the end, $P10$, $P50$, $P90$ could be estimated.

The advantage of using NAB is to avoid running the reservoir simulation for all the models created from the optimisation algorithms and uses the complete ensemble to infer the information about the reservoir (Sambridge, 1999b; Erbas, 2006). It only requires a few simulations of the forecasting period to be run. We will use it in Chapter 6.

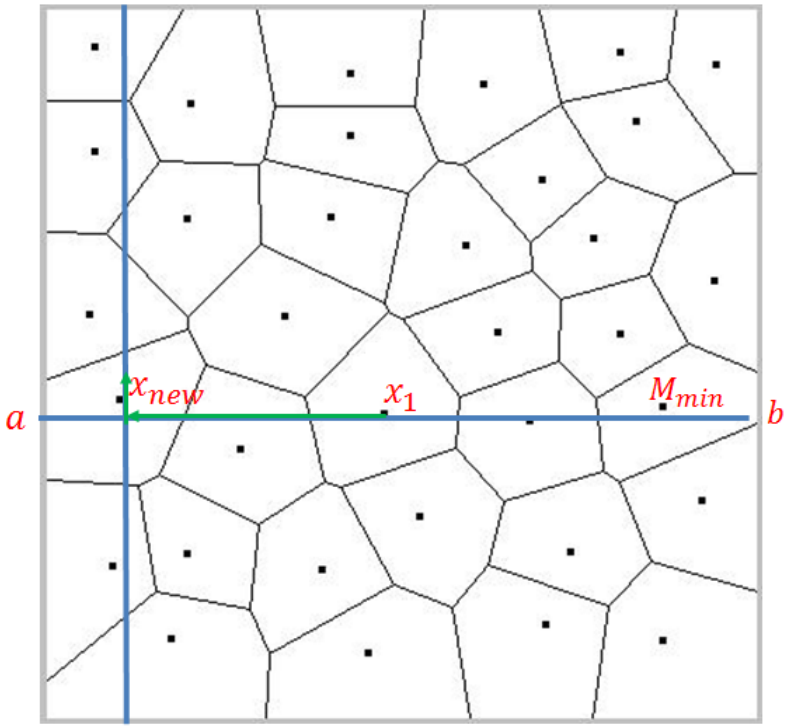


Figure 3.1: NAB resampling adapted from (Sambridge, 1999b).

Often in Bayesian approaches, to estimate the posterior distribution, requires solving the integration of high-dimensional functions. This can be costly (Andrieu et al., 2003), but we have several approaches to perform this integration, e.g., Rejection Sampling (RS) and Markov Chain Monte Carlo (MCMC).

3.3 Rejection Sampling (RS)

The Rejection Sampling technique is based on replacing the target density function, $f(x)$ by another density function, $g(x)$ which is relatively simple to generate samples from (Neal, 1993).

The idea of RS is as follows. Let us assume $f(x)$ is too complicated to sample from directly. We assume that we have a simpler density $g(x)$, and from it, we can generate samples. Moreover, we assume that we know the value of a constant c, such that $cg(x) > f(x)$; $\forall x$. The idea of the RS algorithm is: First generate x^* from $g(x)$ then calculate $cg(x^*)$. Secondly, generate a uniformly distributed random variable u from the interval $[0, cg(x^*)]$. These two random numbers can be viewed as selecting a point on a two-dimensional plane. If $u > f(x^*)$, then x^* is rejected,

otherwise it is accepted. The acceptance rate is based on the ratio between the area under $f(x)$ and the area under $cg(x)$. Therefore, the acceptance rate will be higher if c is close to 1 (MacKay, 2002; Bishop, 2006).

RS is not generally a good practical technique for generating samples from high-dimensional distributions (Neal, 1993). It works well only if the proposal density, $cg(x)$, is very close to $f(x)$. Figure 3.2 shows the functions involved in rejection sampling–adapted from (Bishop, 2006). The following example is a toy problem to explain how to apply RS.

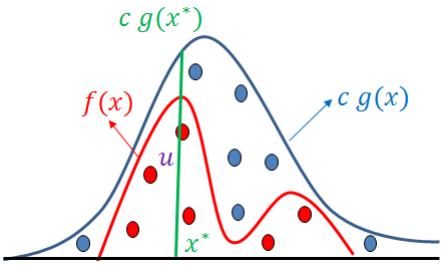


Figure 3.2: The functions involved in rejection sampling.

Example 3.3.0.1 *Estimate π using RS.*

Solution: The area of the unit circle is π. Figure 3.3 shows that the unit circle inside a square.

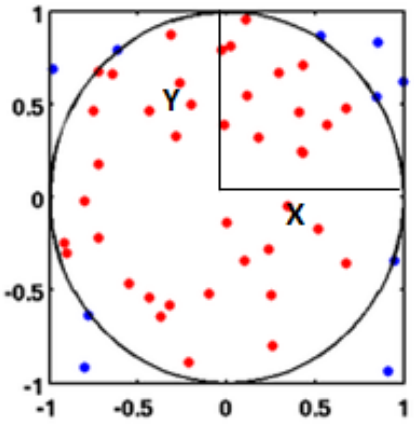


Figure 3.3: Estimate the area of quarter circle.

Then, we estimate the area of the quarter of the circle, which equals to $\pi/4$. To achieve this, choose two random variables X and Y, such that X and Y represent the coordinates of a point and, $X, Y \sim \mathcal{U}(0,1)$. The radius is $R = \sqrt{x^2 + y^2}$. Generate

N samples and the acceptance-rejection condition for the samples is based on the decision function (3.1) : set $m = 0$*, if* $R < 1$ *then* (x, y) *accepted,* $m = m + 1$*, else rejected. Therefore,* $\pi = 4 \times \frac{m}{N}$.

$$decision\ function = \begin{cases} accept(x, y) & R \leq 1 \\ reject(x, y) & R > 1 \end{cases} \tag{3.1}$$

3.4 Markov Chain Monte Carlo (MCMC)

Markov Chain Monte Carlo samples from the posterior distribution. It uses a Markov chain and does not require calculation of the normalization constant, which is very difficult to compute. To estimate the target distribution perfectly, we should have an infinite MCMC (number of iterations tends to infinity), which is impossible. A finite MCMC creates a truncation error, which can be decreased by increasing the sample size, but it is impossible to decrease the bias error.

MCMC is a common technique for modelling in different disciplines to explore the probability distributions, especially, in high dimensional problems.

3.4.1 Introduction to Markov Chains

Definition 3.4.1.1 *(Gilks et al., 1996) Let* $(X_n)_{n \in \mathbb{Z}^+}$ *be a stochastic process taking values in a countable set* S*. We say that the* ***Markov property*** *holds if*

$$\mathbb{P}(X_{n+1} = j | X_n = i, X_{n-1} = i_{n-1}, \cdots, X_0 = i_0) = \mathbb{P}(X_{n+1} = j | X_n = i) \tag{3.2a}$$

$$= P_{ij},\ \forall n \in \mathbb{Z}^+, i_0, i_1, \cdots, i, j \in S. \tag{3.2b}$$

where, the transition probability from i to j is P_{ij}. The Markov process has no memory, in the sense that the next state only depends on the current state. The chain is time homogeneous because the transition probability is independent of time (Gamerman, 1997). A Markov Chain has three components: state space S, the distribution for the initial state $\mathbf{p}(0)$ and transition probability from one state to another $\mathbf{P}$.

Let, $p_j(n) = \mathbb{P}(X_n = j)$ and $\boldsymbol{p}(n) = (p_j(n))$. In general, we start the chain by

specifying a starting vector, $\boldsymbol{p}(0)$. Often all the elements of $\boldsymbol{p}(0)$ are zeros except for a single element. The probability that the chain has state value i at time (or step) $n+1$ is given by the **Chapman-Kolmogorov equation** (Gamerman, 1997),

$$p_i(n+1) = \sum_k P_{ki} p_k(n). \tag{3.3}$$

Equation (3.3) describes the evolution of the chain. The **Chapman-Kolmogorov equation** in a matrix form can be written as follows

$$\mathrm{p}(n+1) = \mathrm{p}(n)\mathbf{P}.$$

Since, the transition probability does not change from one state to another, $\boldsymbol{p}(n) = \boldsymbol{p}(0)\mathbf{P}^n$.

The chain is irreducible if all states communicate.

Definition 3.4.1.2 *A Markov chain is said to be* ***irreducible****, if* $\exists\, n \in \mathbb{N}^* \mid P_{ij}^n > 0, \forall i, j \in S$.

If the system returns to state s after leaving it at some time in the future, the state s is **recurrent**. *A Markov Chain is irreducible if the corresponding graph is strongly connected (and thus all its states are recurrent)* (Gilks et al., 1996).

A state s has a period k, if k is the greatest common divisor (gcd) of all the cycle lengths that pass via s. If all the states in a Markov Chain have a period $k > 1$, then the chain is **periodic**. Otherwise, it is aperiodic. In other words, the chain is aperiodic if it does not get trapped in cycles.

Definition 3.4.1.3 *(Gamerman, 1997) A Markov chain is said to be* ***aperiodic****, if* $\forall i, j \in S\, gcd\{n : p_{ij}^n > 0\} = 1$.

If $\mathbf{P}$ has no eigenvalue equal to -1, then it is aperiodic (Gilks et al., 1996).

A Markov chain may reach **a stationary distribution $\boldsymbol{\pi} = \boldsymbol{\pi}\ \mathbf{P}$**.

- $\boldsymbol{\pi}$ is the left eigenvector associated with the eigenvalue $\lambda = 1$ of $\mathbf{P}$. An **invariant distribution** is an eigenvector of the transition probability matrix with eigenvalue 1 and $\pi(x)$ is an invariant distribution.

- The conditions for a unique stationary distribution are that the chain is irreducible and aperiodic.

- When a chain is periodic, it can cycle in a deterministic fashion between states and hence never settles down to a stationary distribution.

A sufficient condition for a unique stationary distribution, π is that the detailed balance equation holds $\forall\, k,\, j \in S,\, P_{jk}\pi_j = P_{kj}\pi_k$ but this is not necessary. The reason why the detailed balance property is of interest is that it implies invariance of the distribution $\pi(x)$ under the Markov chain. This is a necessary condition of the key property that we want from our MCMC simulation, this being that the probability distribution of the chain should converge to $\pi(x)$ (Gamerman, 1997).

A detailed balance Markov chain satisfies the detailed balance equation and is also called a reversible Markov chain. This means that the probability of the chain moving from state i to state j is the same as the probability of moving from state j to state i. The chain has an equilibrium distribution when the chain is reversible (Gilks et al., 1996).

If the Markov chain has stationary distribution, irreducibility and aperiodicity, then the chain is called an **ergodic** chain. Ergodic Markov Chains are important because they guarantee the convergence to a unique distribution, in which each state has a strictly positive probability (Bishop, 2006). Reasons why a chain might not be ergodic are: its matrix might be reducible, the transition probability matrix of such a chain has more than one eigenvalue equal to 1 or the chain might have a periodic set.

The continuous extension of the **Chapman-Kolmogorov equation** becomes,

$$p_t(y) = \int p_{t-1}(x)P(x,y)\mathrm{dx}.$$

At equilibrium, this stationary distribution satisfies,

$$\pi(y) = \int \pi(x)P(x,y)\mathrm{dx}.$$

3.4.2 Metropolis-Hasting Algorithm (MH)

The Metropolis algorithm was proposed by physicists to compute complex integrals. They expressed the integrals as expectations for some distribution (Metropolis and Ulam, 1949; Metropolis et al., 1953). The Metropolis algorithm (Metropolis et al., 1953) assumes that the proposal distribution is symmetric, i.e. $q(\theta_1, \theta_2) = q(\theta_2, \theta_1)$.

Hastings generalized the Metropolis algorithm, i.e. the proposal distribution is symmetric or non symmetric (Hastings, 1970). MH generates a sequence drawing from the proposal distribution, as in Algorithm 2. It constructs a Markov chain that converges to the posterior distribution.

MH was one of the top ten algorithms to have the greatest influences on the practice of science in the 20th century (Beichl and Sullivan, 2000). It is one of the most common and simplest methods for estimating the target distribution. It requires a proposal distribution q and an accept-reject criterion.

$$\text{Acceptance probability} = \min\{1, \frac{q_{ij}\pi_j}{q_{ji}\pi_i}\}$$

where π is the posterior distribution.

Definition 3.4.2.1 *The **acceptance rate** for the chain is the ratio between the number of accepted samples and the number of samples after the burning-in period.*

Running MH with n iterations does not produce n independent samples from the target distribution. Since the successive samples are dependent, we need to run the algorithm for enough time to get independent samples. Metropolis-Hastings ensures that the target distribution is invariant, but it does not ensure that the chain is aperiodic and irreducible (MacKay, 2002).

MH is different to RS because if we reject a sample, it gives weight to the previous sample, but in the case of RS, we discard the sample completely.

3.4.2.1 Choosing the Proposal Distribution

The choice of the proposal distribution is one of the hardest tasks. It should have the same structure as the target distribution and have a positive probability den-

Algorithm 2: Metropolis-Hasting (Hajian, 2007).

Data: Initialize x_0.
Result: Vector containing all the acceptance states $x = (x_0, x_1, \cdots)$.
for $i = 1$ **to** $N_{samples}$ **do**
  Draw $x_{new} \sim q(.|x_o)$ which is a proposal distribution;
  Draw $\alpha' \sim \mathcal{U}(0, 1)$;
  if $\alpha' < \min\{\frac{\pi(x_{new})q(x_{new}, x_o)}{\pi(x_o)q(x_o, x_{new})}, 1\}$ **then**
    $x[i] = x_{new}$
    $x_o = x_{new}$;
  else
    $x[i] = x_o$;
  end
end

sity (MacKay, 2002; Walsh, 2004). If the proposal distribution is non-zero, then the Markov chain is ergodic (Neal, 1993).

The choice of the standard deviation of the proposed distribution should be compatible with the target distribution shape. The acceptance rate for MH is very low if the width of the target distribution is too small compared to the standard deviation of the proposal distribution (Kuzmanovska, 2012).

The following is example 3 in (Walsh, 2004).

Example 3.4.2.1 *How does the choice of the proposal distribution effect on the output result? The target distribution is* $\pi(x) = cx^{-5/2}\, e^{-2/x}$. *We use two proposal distributions* χ^2_2 *and* χ^2_{10} *and compare the results.*

Recall for $x \sim \chi^2_n$ *, that* $q(y, x) \propto x^{n/2-1}e^{-x/2}$. *Figure 3.4 shows the results of a single run of the sampler under two different proposal distributions (*χ^2_2 *and* χ^2_{10} *). *χ^2_2 *has a smaller standard deviation than* χ^2_{10} *, and thus a higher acceptance probability. The chain using* χ^2_{10} *mixes poorly (for long periods of time, explore small regions of the parameter space) unlike the other chain which mixes well.*

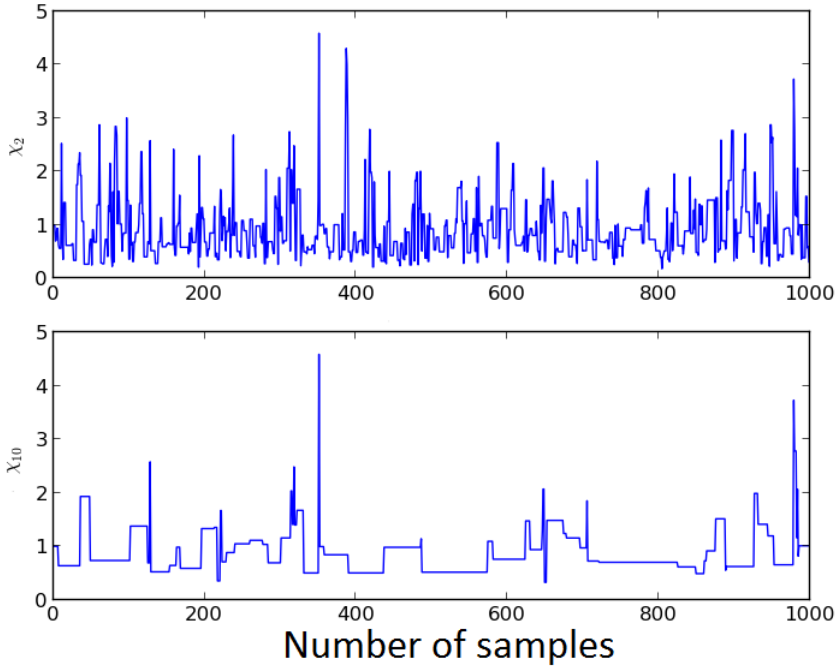


Figure 3.4: (top): Good mixing using proposal distribution χ^2_2. (bottom): Poor mixing using proposal distribution χ^2_{10}.

3.4.3 Gibbs Sampling

The Gibbs sampling algorithm was named after the physicist Josiah Gibbs. Gibbs sampling was proposed in (Geman and Geman, 1984) to study a problem related to image processing. It is a special case of MH, where the random value is always accepted (Robert and Casella, 1999). Gibbs sampling can be viewed as a Metropolis method in which a sequence of proposal distributions is defined in terms of the conditional distributions of the joint distribution (Walsh, 2004).

To explain how Gibbs sampling works, we consider a bivariate random variable (x, y), and suppose we want to compute the marginals, $p(x)$ and $p(y)$. The sampler starts with the initial value x_0 for x and obtains y_0 by generating a random variable from the conditional distribution $p(y|x = x_0)$. The sampler uses y_0 to generate a new value of x_1 , drawing from the conditional distribution based on the value y_0, $p(x|y = y_0)$ (Walsh, 2004). The sampler proceeds as follows,

$$x_i \sim p(x|y = y_{i-1}) \qquad y_i \sim p(y|x = x_i).$$

Algorithm 3 shows the steps for Gibbs sampling for a d dimensional problem.

The convergence of Gibbs sampling can be improved by involving fewer variables than the full set of variables in the conditional distribution (Gelfand and Smith, 1990).

Algorithm 3: Gibbs Sampling (MacKay, 2002)

Data: Initialize $t = 0$, N_{samples}, initial state for the chain
$X_0 = (x_1^{(o)}, x_2^{(o)}, \cdots, x_d^{(o)})$.
Result: Vector contains all the acceptance states $x = (X_0, X_1, \cdots)$.
for $t = 1$ **to** $N_{samples}$ **do**
  Draw $x_1^{(t)} \sim p(x_l | x_2^{(t-1)}, \cdots, x_d^{(t-1)})$
  Draw $x_2^{(t)} \sim p(x_2 | x_1^{(t)}, x_3^{(t-1)}, \cdots, x_d^{(t-1)})$
  $\vdots$
  Draw $x_d^{(t)} \sim p(x_d | x_1^{(t)}, x_3^{(t)}, \cdots, x_{d-1}^{(t)})$
  $X_t = (x_1^{(t)}, \cdots, x_d^{(t)})$
end

3.4.4 Advantages and Disadvantage of MCMC Methods

We summarise the advantages and disadvantages of RS, Metropolis, MH and Gibbs sampling methods (MacKay, 2002; Hajian, 2007) as follows,

- Monte Carlo methods are powerful tools that allow us to sample from any probability distribution.
- RS requires the proposal distribution to be very close to the target distribution and is unsuited to problems in high dimensions.
- MH is relatively easy to implement, even when the target distribution is complicated.
- If the MH method suffers a from low acceptance rate, then it takes a long time and a large numbers of steps to converge in general.
- Gibbs sampling is easy to implement, but it requires sampling from the conditional distribution of each parameter. Sometimes it will be hard if we have complex distributions, therefore it is less applicable in practice.
- Gibbs sampling suffers from the same defect as simple Metropolis algorithms–the state space is explored by a slow random walk.

- A common problem of MCMC techniques is the high correlation between the samples.

- In high-dimensional problems, Metropolis algorithms have slow mixing and low efficiency, but Gibbs sampling is an attractive method because it has no adjustable parameters but its use is restricted to certain cases.

3.4.5 Chain Set-up

One current research topic is how to choose the initial state of the chain. Based on the definition of the Markov chain, the chain forgets the initial state. However, the choice of initial state significantly impacts the convergence rate of the chain. The common choice of initial state is based on the prior knowledge of each uncertain parameter. One suggestion of the initial state is the most central point of the distribution, for example, the distribution's **mode**.

How long do we need to run the chain? There are three possibilities:

- Making one long run, this has the best chance of obtaining the convergence to the stationary distribution.

- Making a few medium length runs with different initial conditions.

- Making multiple short runs, each starting from different random initial positions. The advantage is that the correlations between the recorded samples are smaller. The motivation for this is that we obtain independent samples (Gilks et al., 1996).

Note that, with parallel processing machines, utilizing multiple chains is computationally more efficient than a single long chain. When we have long flat periods (corresponding to values being rejected), the chain is mixing poorly (if it takes time to cover small regions of the parameter space) and it arises because the target distribution is multimodal and the choice of initial state is close to one of the peaks. To deal with this problem, we should start with several different chains or use the simulated annealing on a single-chain.

3.5 Summary

This chapter provides different sampling techniques. Some of them are based on optimisation, e.g. NA and PSO. Some of them are Bayesian, e.g. RS, MH, and Gibbs sampling. Due to the slow convergence performance of MH, there are different techniques to accelerate convergence, e.g. HMC, (Section 8.1) based on Hamiltonian dynamics. In the Chapters 6 and 8, we will explain two methods, Multilevel Markov Chain Monte Carlo and Multilevel Hamiltonian Monte Carlo, for speeding up convergence and the time between independent samples. The following chapter will review the numerical solution for conservation equations.

Chapter 4

Numerical Solution for Conservation Equations–Literature

In Chapter 2, we discussed the system of equations governing the flow in a porous media. The types of equations were: hyperbolic and parabolic, namely mass conservation and the pressure equation respectively. We obtained the differential and integral forms of the equations.

In this chapter, we focus on reviewing different finite difference schemes and finding the solutions of the advection equation, Buckley-Leverett equation and the pressure equation. We find the exact solution, when it is possible. Moreover, we study the numerical solution using different numerical schemes and different initial conditions. Also, we check the stability and determine the stability conditions and analyse the behaviour of a number of finite difference schemes for solving differential equations governing the miscible displacement (Rakhib, 2004; Shu, 2006).

These equations have many applications in science and engineering for example in weather prediction, meteorology and petroleum engineering. Also, there are many complicated issues associated with solving these systems, such as shock formation, which are not seen elsewhere. In the following section, we review finite difference schemes.

4.1 Finite Difference Schemes

We have a vast number of numerical techniques for approximating the solution of the hyperbolic system, (2.1). We are searching for a numerical scheme that is consistent with the conservation law (2.1) that is also convergent, stable and propagates physically valid fronts. We discuss some of these, including the stability. Also, we study the numerical dispersion and the numerical diffusion of the schemes. Before discussing any numerical schemes, we should first recap what convergence is and the stability of the numerical schemes.

For a linear PDE, the Lax equivalence theorem states that for a consistent finite difference method for a well-posed linear initial value problem, the method is convergent if and only if it is stable (Strikwerda, 1989). A numerical scheme is convergent when the approximate solution tends to the exact solution as the time step tends to zero and the mesh size tends to zero as well. If we do not know the exact solution, we can increase the number of time steps, decrease the mesh size and use the solution as a reference to approximate the error. Based on the error, we can decide if the scheme is convergent or divergent. The numerical scheme can be divergent if a discontinuity appears. The Courant-Friedrichs-Lewy (CFL) condition is necessary but insufficient for the stability condition. This means that when the numerical scheme satisfies the CFL condition, we can apply the stability test- this is sufficient. Sometimes, the numerical scheme can generate oscillations, however, the analytical solution does not have any (Trangenstein, 1986; Shu, 2006; Leveque, 1992).

We have several ways of approximating first and second derivatives using finite difference methods. The popular ones are:

$$\dot{f}_j \simeq \frac{f_{j+1} - f_j}{h} \qquad \rightarrow \text{Forward with accuracy } \mathcal{O}(h)$$

$$\dot{f}_j \simeq \frac{f_j - f_{j-1}}{h} \qquad \rightarrow \text{Backward with accuracy } \mathcal{O}(h)$$

$$\dot{f}_j \simeq \frac{f_{j+1} - f_{j-1}}{2h} \qquad \rightarrow \text{Center with accuracy } \mathcal{O}(h^2)$$

$$\ddot{f}_j \simeq \frac{f_{j+1} - 2f_j + f_{j-1}}{h^2} \qquad \rightarrow \text{Center with accuracy } \mathcal{O}(h^2).$$

Where $\dot{f}$ denotes the derivative of f with respect to time and $h = \triangle t$. If it is with respect to space, $h = \triangle x$ (Leveque, 1992; Rakhib, 2004). In the following section, we discuss how to find the analytical solution of the first order hyperbolic equation.

4.2 Analytical Solution of the First Order Hyperbolic Equation

The first order $1D$ hyperbolic equation (see Appendix A) is written as follows

$$\frac{\partial s}{\partial t} + \frac{\partial f}{\partial x}(s) = 0,\ x \in \mathbb{R},\ t > 0 \tag{4.1a}$$

$$s(x, 0) = s_0(x), \tag{4.1b}$$

where, $s : \mathbb{R} \times \mathbb{R}^+ \to \mathbb{R}$ is a mass or saturation variable $f : \mathbb{R} \to \mathbb{R}$ is the flux function and $s_0(x)$ is an initial condition. We can rewrite (4.1) using the chain rule to obtain the following form

$$\frac{\partial s}{\partial t} + \frac{\mathrm{df}}{\mathrm{d}s}\frac{\partial s}{\partial x} = 0. \tag{4.2}$$

We can use the method of characteristics to solve (4.2) analytically. The main advantage of using this method is converting a PDE to an ODE. The characteristic curves $x(t)$ in the $x - t$ plane satisfy

$$\frac{\mathrm{dx}}{\mathrm{d}t} = \frac{\mathrm{df}}{\mathrm{d}s} \implies x = x_0 + \frac{\mathrm{df}}{\mathrm{d}s}(s_0)t.$$

By substituting this into (4.2), we obtain

$$\frac{\partial s}{\partial t} + \frac{\mathrm{df}}{\mathrm{d}s}\frac{\partial s}{\partial x} = \frac{\partial s}{\partial t} + \frac{\mathrm{dx}}{\mathrm{d}t}\frac{\partial s}{\partial x} = \frac{\mathrm{d}s}{\mathrm{d}t} = 0.$$

Therefore, s is constant along the characteristic curves. In the following section, we study the case when $f(s) = vs$ of (4.1) where v is a constant speed.

4.3 Advection Equation

In this section, we consider the one-dimensional linear advection (scalar transport) equation. The linear advection equation, with speed v, can be written as shown by (4.3) below.

$$\frac{\partial s}{\partial t} + v\frac{\partial s}{\partial x} = 0,\ x \in \mathbb{R},\ t > 0 \tag{4.3a}$$

$$s(x, 0) = s_0(x). \tag{4.3b}$$

Equation (4.3) is a special case of (4.2) when $\frac{\mathrm{df}}{\mathrm{d}s} = v$. The advection equation describes the motion of a conserved scalar field s, for example the gas motion is advected by a known velocity vector field. The meaning of the advection equation is: if we start from initial point (x_0, t_0) and subsequently move with a constant speed v, then $s(x, t)$ will never change from its initial position. The characteristics for (4.3) are $x - vt$. If the characteristic curves satisfy $\frac{\mathrm{dx}}{\mathrm{d}t} = v$, then $x = x_0 + vt$ and s is constant along the characteristic curves, i.e. $s(x, t) = s(x_0, 0) = s(x - vt)$. Therefore, if v is positive constant speed, then the analytical solution of (4.3) is a wave propagating to the right.

$$s(x, t) = s_0(x - vt),$$

where $s_0 = s_0(x)$ is an initial condition. The solution is a set of straight parallel lines (Leveque, 1992; Trangenstein, 1986). We will discuss the stochastic version of the advection equation using single Point Upstream Weighting and Lax-Wendroff schemes in Section 5.4.2. We next discuss the numerical solution of the advection equation.

4.3.1 Numerical Solution of the Advection Equation

In this section, we discuss different numerical schemes for solving the advection equation. We discuss the solution using first and second order schemes, namely the single point upstream and Lax-Wendroff, respectively. Moreover, we discuss the

stability of each scheme.

4.3.1.1 Single Point Upstream Weighting (Upwind) Scheme

The general formula for solving Equation (4.3) is:

$$s_j^{n+1} = \begin{cases} s_j^n - v\dfrac{\triangle t}{\triangle x}\left(s_j^n - s_{j-1}^n\right) & ;\ v > 0 \\ s_j^n - v\dfrac{\triangle t}{\triangle x}\left(s_{j+1}^n - s_j^n\right) & ;\ v < 0 \end{cases} \quad j = 0, 1, 2, \cdots, M \tag{4.4}$$

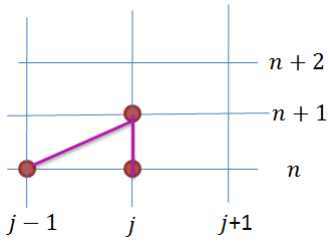


Figure 4.1: The schematic visualization of the upwind method when $v > 0$.

To obtain the differential dispersion relation (Christie, 1987), we substitute the Fourier mode $s(x,t) = e^{i(\omega t - kx)}$ into (4.3) as follows,

$$i\omega s - ikvs = 0,\ u \neq 0 \Rightarrow \omega = kv, \tag{4.5}$$

where k is the wave number. Since there is no imaginary component, the wave is undamped. Also, Equation (4.5) says that all Fourier modes travel at the same speed.

To study the stability of the scheme, we apply the concept of Von-Neumann stability (Leveque, 1992; Trefethen, 1996; Christie, 1987). Letting $s_j^n = e^{i(\omega n\triangle t - kj\triangle x)}$ and $\alpha = s_j^{n+1}/s_j^n$ and substituting into (4.3) we obtain

$$e^{i(\omega(n+1)\triangle t - kj\triangle x)} = e^{i(\omega n\triangle t - kj\triangle x)} - v\frac{\triangle t}{\triangle x}\left(e^{i(\omega n\triangle t - kj\triangle x)} - e^{i(\omega n\triangle t - k(j-1)\triangle x)}\right),$$

and dividing by $e^{i(\omega n\triangle t - kj\triangle x)}$, we obtain the following

$$e^{i\omega\triangle t} = \alpha = 1 - v\frac{\triangle t}{\triangle x}\left(1 - e^{ik\triangle x}\right).$$

Therefore,

$$e^{i\omega\triangle t} = \alpha = \left(1 - v\frac{\triangle t}{\triangle x}\right) + v\frac{\triangle t}{\triangle x}e^{ik\triangle x}. \tag{4.6}$$

Equation (4.6) is the equation of a circle with centre $1 - v\frac{\triangle t}{\triangle x}$ and radius $v\frac{\triangle t}{\triangle x}$ in the complex plane. The scheme is stable when $|\alpha| < 1$. To get this, $v\frac{\triangle t}{\triangle x}$ should be less than 1. This means that the scheme is conditionally stable if and only if the physical velocity v is smaller than the spreading velocity $\triangle x/\triangle t$. We examine this scheme with periodic boundary condition i.e. $s^n_{-1} = s^n_M$.

In conclusion, the upwind method is a first order accurate in time and space i.e., $\mathcal{O}(\triangle x, \triangle t)$. The scheme is stable if $v > 0$ and thus $0 < v\frac{\triangle t}{\triangle x} < 1$, otherwise $-1 < v\frac{\triangle t}{\triangle x} \leq 0$. The scheme gives a poor resolution at discontinuities.

Example 4.3.1.1 *Use the single point upstream method for solving the advection equation, (4.3), at $t = 1$ assuming velocity $v = 1$, a periodic boundary condition and an initial condition,*

$$s_0 = \begin{cases} 1 & ; 0.4 \leq x \leq 0.6 \\ 0 & ; \textit{otherwise} \end{cases}. \tag{4.7}$$

Figure 4.2 shows the approximate solution is smeared out and the maximum height is about 0.85 compared with the exact solution, using $CFL = v\frac{\triangle t}{\triangle x} = 0.5$.

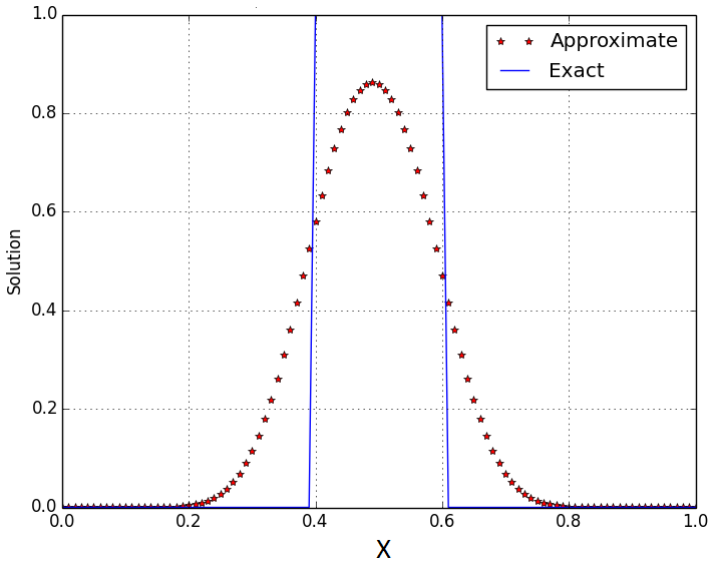


Figure 4.2: Advection of pulse with the single point upstream with CFL= 0.5.

4.3.1.2 Lax-Wendroff Scheme

The Lax-Wendroff scheme is a second order accurate in space and time i.e. $\mathcal{O}(\triangle^2 x, \triangle^2 t)$. The general formula for the Lax-Wendroff scheme for solving (4.3) is

$$s_j^{n+1} = \frac{v\ \triangle t}{2\ \triangle x}\big(v\frac{\triangle t}{\triangle x}+1\big)s_{j-1}^n+\big(1-v^2\frac{\triangle t^2}{\triangle x^2}\big)s_j^n+\frac{v\ \triangle t}{2\ \triangle x}\big(v\frac{\triangle t}{\triangle x}-1\big)s_{j+1}^n,\ j=0,1,\cdots,M. \tag{4.8}$$

To use Lax-Wendroff scheme, we examine the scheme assuming periodic boundary conditions, $s_{-1}^n = s_M^n,\ s_{M+1}^n = s_0^n$.

The sonic point produces an oscillation to the right of the shocks. Sometimes, for initial data, we can not see the oscillation. This is probably because of the combination of the shock and the fan, as the fan is damping the oscillation produced by the shock.

To study the stability of the scheme, we apply the concept of Von-Neumann (Trefethen, 1996; Christie, 1987). Letting $s_j^n = e^{i(\omega n\triangle t - kj\triangle x)}$ and $\alpha = s_j^{n+1}/s_j^n$, and substituting into (4.3), we obtain that the scheme is stable when $v\dfrac{\triangle t}{\triangle x} < 1$.

In general, ω is a complex

$$\omega = \Omega + i\gamma$$

where,

$$\tan\Omega t = \frac{-v\dfrac{\triangle t}{\triangle x}\sin(k\triangle x)}{1+v^2\dfrac{\triangle t^2}{\triangle x^2}\big(\cos(k\triangle x)-1\big)}$$

$$e^{-2\gamma t} = v^2\frac{\triangle t}{\triangle x}^2\sin^2(k\triangle x)+\Big(1+v^2\frac{\triangle t^2}{\triangle x^2}\big(\cos(k\triangle x)-1\big)\Big)^2.$$

The phase error is $\Omega - vk$ and from that, we know the dispersion Ω and γ, the diffusion.

Example 4.3.1.2 *Find the solution of the advection equation at $t = 1$, using the Lax-Wendroff scheme (4.8), with a periodic boundary condition and an initial condition (4.7). Assuming the velocity, $v = 1$ and $CFL = 0.9$, Figure 4.3 shows the solution has sharp resolution because the Lax-Wendroff scheme has a low numerical*

diffusion. Also, the solution overshoots and undershoots (below zero) and we observe large oscillations in the solution (Christie, 1987; Strauss, 2007).

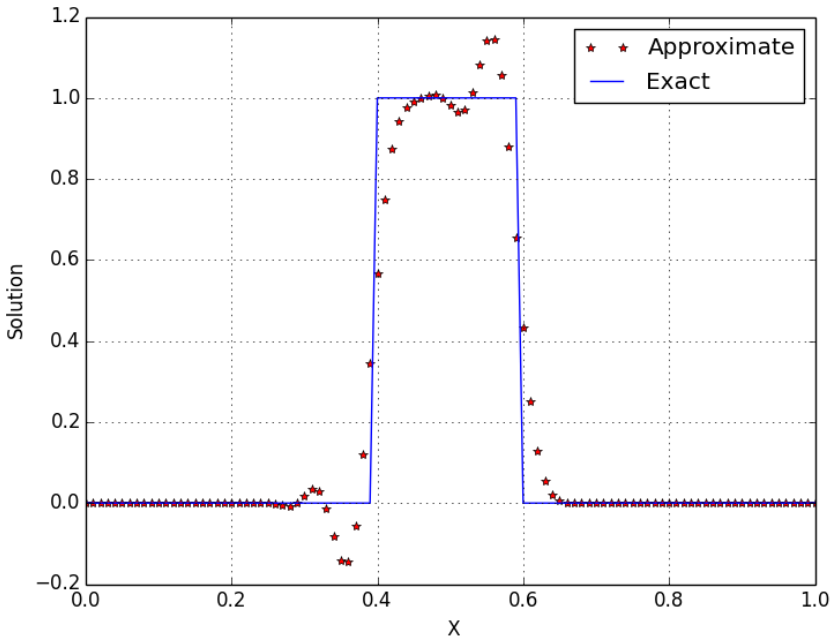


Figure 4.3: Advection of a pulse with the Lax-Wendroff scheme with CFL= 0.9.

We next discuss a nonlinear example of (4.1).

4.4 Buckley-Leverett Equation

The theory of oil displacement established by Buckley and Leverett, is based on the relative permeability (Buckley and Leverett, 1942). Relative permeabilities only depend on the saturation. It is a non-convex and non-linear scalar equation (Trangenstein, 1986; Carter, 2010; Rakhib, 2004). Buckley-Leverett is a very simplified version of the two-phase flow equations under specific assumptions, which are the effects of the capillary pressure gradient and gravity forces have been ignored, flow is linear and horizontal, water is injected into an oil reservoir, oil and water are both incompressible and oil and water are immiscible (Weizhong Luo, 1986).

4.4.1 Derivation of the Buckley-Leverett Equation

When the water displaces oil, by applying the mass balance of water around the control volume length, $\triangle x$ for a time period $\triangle t$, we can write the mass balance as follows:

$$((q\rho)_w|_x - (q\rho)_w|_{x+\triangle x}) \triangle t = A\phi \triangle x((s\rho)_w|_{t+\triangle t} - (s\rho)_w|_t),$$

where q is the rate, ρ_w is the water density, ϕ is porosity, s_w is the water saturation and A is the area. By taking limits, when $\triangle x$, $\triangle t \to 0$, we get the following:

$$-\frac{\partial(q\rho)_w}{\partial x} = A\phi\frac{\partial(s\rho)_w}{\partial t}. \tag{4.9}$$

Assuming the fluid is incompressible, ρ_w is a constant and $q_w = q_{\text{total}} f_w$. Therefore, the one dimensional Buckley-Leverett equation can be written as follows

$$\frac{\partial s_w(x,t)}{\partial t} + \frac{q_{\text{total}}}{A\phi}\frac{\partial f(s_w)}{\partial x} = 0.$$

If $s_w = 0$ ($s_w = 1$) that means, we have a pure oil (water) respectively (Trangenstein, 1986). The distance along the flow path is x, t is time, and q_{total} is total rate of the flow.

Buckley and Leverett found the saturation profile for a water-flood model, but their solution is physically unrealistic as shown in Figure 4.4. Due to this, they got multiple-valued saturated solutions. Therefore, we should replace it by a discontinuity profile. To do this, we should find the position of the discontinuity, determined by material balance. If the flux flow function has more inflection points, the solution may have several shocks (Rakhib, 2004).

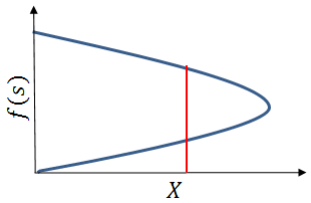


Figure 4.4: Water saturation profile.

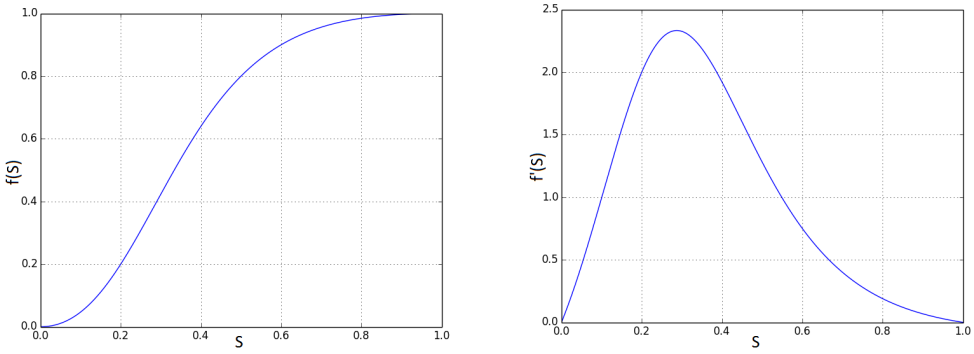


Figure 4.5: (a): Flux function (b): The derivative of the flux function for Buckley-Leverett equation.

4.4.2 Analytical Solution

The Buckley-Leverett equation in one dimension has the following form:

$$\frac{\partial s}{\partial t} + v\frac{\partial f}{\partial x}(s) = 0 \Leftrightarrow \frac{\partial s}{\partial t} + \frac{\partial s}{\partial x} v \frac{\mathrm{df}}{\mathrm{ds}} = 0. \tag{4.10}$$

The velocity of a fixed plane with saturation s is $\frac{\mathrm{dx}}{\mathrm{dt}} = v\frac{\mathrm{df}}{\mathrm{ds}}$. Due to some values of x having multiple values for the saturation. We find a weak solution by using the entropy condition, which says $\frac{\mathrm{df}}{\mathrm{ds}} = \frac{[f]}{[s]}$, where $[f] = f_r - f_l$ and $[s] = s_r - s_l$ means change in f and s respectively. First we find s and then substitute this into $x = x_0 + v\frac{\mathrm{df}}{\mathrm{ds}}(s_0)(t - t_0)$. Thus, we can find x as well. The Buckley-Leverett equation does not have a general form for the flux function, this usually comes from experimentation and this makes the problem interesting. One popular example of flux function (Leveque, 1992) is

$$f(s) = \frac{s^2}{s^2 + \frac{1}{4}(1-s)^2}. \tag{4.11}$$

Figure 4.5 shows the flux function (4.11) and its derivative.

Using the method of characteristics, assuming $t_0 = 0$, the analytical solution is found to be:

$$x = t\frac{8s_0 - 8s_0^2}{(5s_0^2 - 2s_0 + 1)} + x_0,\ s(x_0, 0) = s_0$$

The shock speed $= \frac{[f]}{[s]} = \frac{f_R - f_L}{s_R - s_L}$. The fractional flow curve is not affected by

gravity, but it is affected by the mobility ratio $\frac{k_{rw}}{\mu_w}/\frac{k_{r0}}{\mu_0}$. The lowest ratio has the most efficient displacement resulting in the fractional flow curves having no inflection point (Carter, 2010). We will use the analytical solution for the Buckley-Leverett equation to test the efficiency of the stochastic technique in Chapter 5. We will discuss the stochastic version of the Buckley-Leverett equation later in Section 5.4.3.

4.4.3 The Numerical Solution of the Buckley-Leverett Equation

The general formula for a single point upstream weighting (upwind) scheme for solving (4.10) is:

$$s_j^{n+1} = s_j^n - \frac{\triangle t}{2 \triangle x}\left(f_j^n - f_{j-1}^n\right),\ f = \frac{s^2}{s^2 + 0.25\,(1-s)^2} \tag{4.12}$$

The initial condition is $s(x,0) = 1$. We have a periodic boundary condition. We find the solution at $t = 1$ as in the Figure 4.6. Figure 4.6 shows the difficulties due to the nonlinear flux function–the numerical diffusion close to the shock (Blunt and Christie, 1989). It shows that the solution with the single point upstream method is reliable and physically consistent because of a satisfactory agreement with the Buckley-Leverett results (Weizhong Luo, 1986). Moreover, it shows that there is a problem near the discontinuity and the shock front is much sharper with little numerical diffusion. (Barenblatt, 1996; Duffy, 2004; Strauss, 2007)

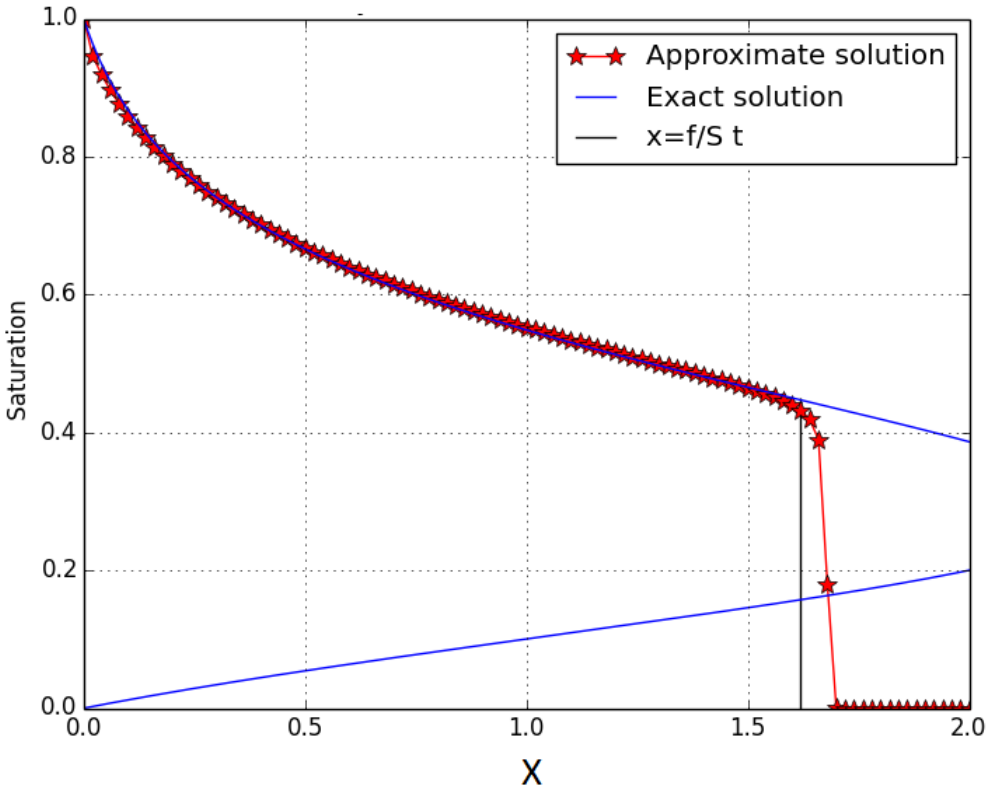


Figure 4.6: Buckley-Leverett solution with the single point upstream scheme with CFL= 0.4.

4.5 Pressure Equation

The one dimensional pressure diffusion equation can be expressed as follows,

$$\frac{\partial P}{\partial t} = D\frac{\partial^2 P}{\partial x^2},\ 0 < x < L,\ t > 0, \tag{4.13}$$

where $D = \dfrac{\lambda_t}{\phi c_t}$ is the hydraulic diffusivity, ϕ is the porosity, λ_t is the total fluid mobility and c_t is the total compressibility. The initial condition is $P(x,0) = P_0$, $0 < x < L$ and the boundary conditions are $P(0,t) = P_i$ and $P(L,t) = P_f$. Equation (4.13) is a parabolic equation (see Appendix A) (Zwhalen and Patzek, 1997).

4.5.1 Analytical Solution

First of all, let us find the steady state solution P_s for (4.13) which satisfies

$$\frac{\mathrm{d}^2 P_s}{\mathrm{d}x^2} = 0$$

by integrating the last equation two times and using the boundary condition, $P(0,t) = P_i$ and $P(L,t) = P_f$, we obtain

$$P_s = P_i + \frac{P_f - P_i}{L}x.$$

Let

$$\nu(x,t) = P(x,t) - P_s = P(x,t) - P_i - \frac{P_f - P_i}{L}x. \tag{4.14}$$

By using the boundary conditions for $P(0,t) = P_i$ and $P(L,t) = P_f$, then the boundary conditions for $\nu(x,t)$ are $\nu(0,t) = v(L,t) = 0$ and by using the initial condition for $P(x,0) = P_0$, $0 < x < L$. The initial condition for ν is then

$$\nu(x,0) = (P_0 - P_i) - \frac{P_f - P_i}{L}x.$$

Using separation of variables, we can easily obtain the following exact solution (Sorbie et al., 2015)

$$P(x,t) = \underbrace{P_i + \frac{P_f - P_i}{L}x}_{\text{Steady state solution}} + 2\sum_{n=1}^{\infty} \frac{1}{n\pi} C e^{-\frac{n^2\pi^2}{L^2}Dt} \sin(\frac{n\pi}{L}x). \tag{4.15}$$

We will use the analytical solution for the pressure equation to test the efficiency of the stochastic techniques in Chapter 5. We will discuss the stochastic version of the pressure equation later in Section 5.4.4.

4.5.2 Similarity Solution

We solve (4.13) by finding a similarity solution using the substitution $P(x,t) = y(z)$, $z = x/\sqrt{t}$. We obtain

$$D\frac{\mathrm{d}^2\mathrm{y}}{\mathrm{dz}^2} + \frac{1}{2}\frac{\mathrm{dy}}{\mathrm{dz}}z = 0$$

and by integrating the previous equation, we obtain

$$\frac{\mathrm{dy}}{\mathrm{dz}} = Ae^{-z^2/4D}$$

and by integrating again, we obtain

$$y = B + A \int_0^z e^{-z^2/4D} \mathrm{dz}.$$

The initial condition is $t \to 0 \implies z \to \infty$, then $P = P_0$. The boundary condition is if $x = 0 \implies z = 0$, then $P = P_i$. Therefore, plugging in the initial condition and the boundary condition, we obtain the following

$$y = P_i + \frac{P_0 - P_i}{\sqrt{\pi D}} \int_0^z e^{-z^2/4D} \mathrm{dz}$$

$$P(x,t) = P_i + (P_0 - P_i) erf(x/\sqrt{4Dt}).$$

4.5.3 The Numerical Solution of the Pressure Equation

Before finding the numerical solution, we discuss the stability for explicit and implicit time stepping schemes for solving (4.13).

4.5.3.1 Stability Condition for Explicit and Implicit Schemes

The explicit scheme can be expressed as follows

$$P_j^{n+1} = P_j^n + \frac{\triangle t}{\triangle x^2} D(P_{j+1}^n - 2P_j^n + P_{j-1}^n). \tag{4.16}$$

To study the stability for the scheme, we use the concept of Von-Neumann (Christie, 1987) by calculating the amplification factor, α, using (4.16). Then, we find the condition to get $|\alpha| < 1$ assuming $P_j^n = e^{i(\omega n \triangle t - kj \triangle x)}$. Therefore, we obtain $\alpha = 1 - 4\frac{\triangle t}{\triangle x^2} D \sin^2 \frac{k \triangle x}{2}$. The stability condition for the explicit scheme (4.16) is

$$\frac{\triangle t}{\triangle x^2} \leq \frac{1}{2D}.$$

The implicit scheme is

$$P_j^{n+1} = P_j^n + \frac{\triangle t}{\triangle x^2} D(P_{j+1}^{n+1} - 2P_j^{n+1} + P_{j-1}^{n+1}). \tag{4.17}$$

Using the concept of Von-Neumann as in the explicit scheme, we have $\alpha = 1 - 4\frac{\triangle t}{\triangle x^2} D \sin^2 \frac{k \triangle x}{2} \alpha \implies \alpha = \frac{1}{1 + 4\frac{\triangle t}{\triangle x^2} D \sin^2 \frac{k \triangle x}{2}} \leq 1$. Therefore, the scheme is unconditionally stable. We solve the pressure equation (4.13) numerically under an initial condition, $P_0 = 0$, and boundary conditions $P(0,t) = 1$ and $P(1,t) = 0.5$ with $D = 1$. We use the implicit scheme (4.17). Figure 4.7 shows that the absolute error for the pressure solution in range $10^{-6} - 10^{-4}$, which means the approximation gives an accurate solution.

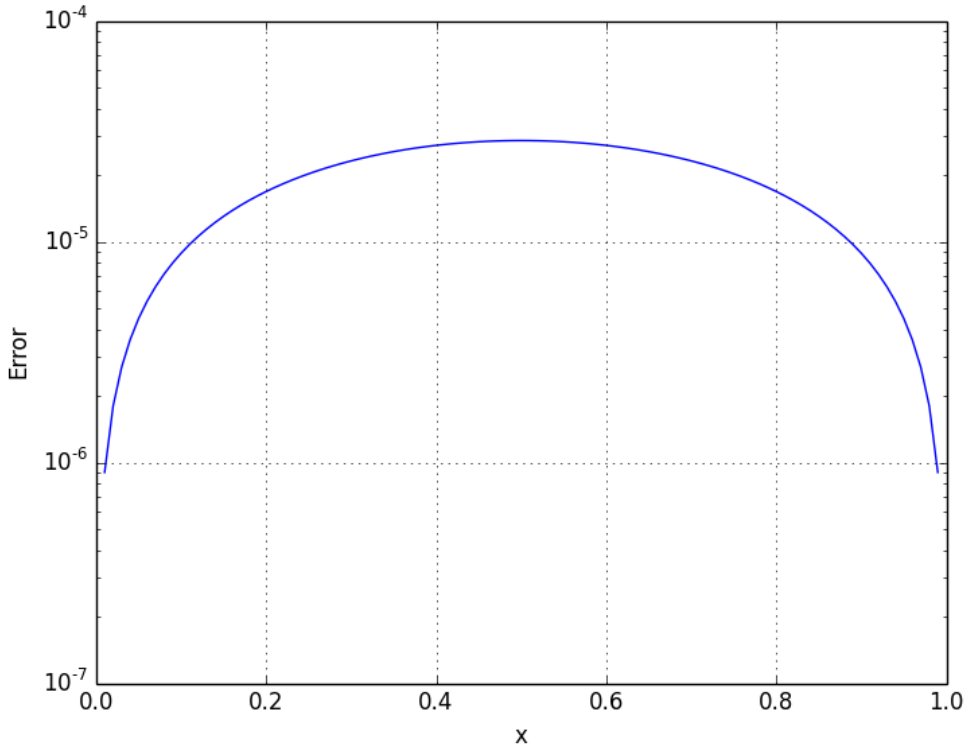


Figure 4.7: Error for the pressure solution using implicit scheme (4.17).

4.6 Summary

This chapter reviews how to find the analytical and numerical solutions for hyperbolic and parabolic differential equations (Advection, Bucley-Leverett and Pressure). In the next chapter, we will discuss how to solve these equations within a stochastic frame. This means either the initial condition or the diffusion parameter or the flux function are random. In the next chapter, we will discuss how to use Monte Carlo Integration and Multilevel Monte Carlo and compare these.

Chapter 5

Multilevel Monte Carlo for Porous Media Flow

This chapter introduces the major concepts of Monte Carlo Integration and the Multilevel Monte Carlo method and compares the two techniques on 4 different problems: the exponential growth and decay equation as a toy example equipped with random initial condition, the advection equation with random initial condition, the Buckley-Leverett equation with random flux function and random initial condition and the pressure equation with a random diffusion coefficient.

Integration and optimization problems are the major numerical problems that arise in statistical inference. Monte Carlo is a powerful method for solving these two problems. The Monte Carlo method is used for approximating integrals, which cannot be solved analytically (MacKay, 2002), for example, it can be used to calculate the normalization constant and the maximum likelihood (Robert and Casella, 2010). Moreover, Monte Carlo integration is better than the numerical techniques for estimating the integration in high dimensions as, the convergence rate for the Simpson's rule and the Trapezoidal rule are $\mathcal{O}(1/M^{4/d})$ and $\mathcal{O}(1/M^{2/d})$ respectively, where d is the dimension and M is the number of samples (H. Gould and Tobochnik, 2010). For example, the Simpson's rule converges slower than Monte Carlo when $d > 8$. In the case of high dimensional space, the numerical approximation can become very expensive and the result may not be efficient.

Monte Carlo methods are computationally expensive because they require a very

large number of samples from a given distribution to be generated before then studying the statistical properties of the samples. For this reason, "Monte Carlo is an extremely bad method; it should be used only when all alternative methods are worse" (Sokal, 1997).

5.1 Monte Carlo Method

The term "Monte Carlo" refers to the Monte Carlo Casino and was first used by Ulam and Von Neumann in 1944 for stochastic simulation to build atomic bombs (Manhattan Project). It is virtually impossible to find a formal definition of Monte Carlo in the literature (Anderson, 1999; Stoian, 1965).

Monte Carlo can be defined as a family name for a variety of stochastic techniques based on random numbers. Any problem that has been solved using the repeated generators of random numbers, can be said to be solved using Monte Carlo techniques. Monte Carlo is used to find answers to problems that may or may not be related to probability (Stoian, 1965).

In petroleum engineering, Monte Carlo methods have been used for more than four decades. For example, (Walstrom et al., 1967) used Monte Carlo simulation to evaluate uncertainty in calculation of water saturation from well logs. However, the disadvantage of using Monte Carlo is the number of simulations that need to be run to obtain a specific accuracy of the solution is huge. In practice, for a simple reservoir model it may be take $1/4 - 6$ hours for each realization, so may be impractical to run a huge number of simulations.

5.1.1 Monte Carlo Integration (MCI)

The generic problem is about evaluating the following integral.

$$I = \int_{\Omega} f(x)\, p(x) \mathrm{dx}, \tag{5.1}$$

where Ω is the sample space, p is the density of a probability measure defined on measurable space, f is measurable function defined on the measurable space which is

integrable with respect to p. The integral is equal to the expectation of the function f with respect to p,

$$I = \mathbb{E}_p(f).$$

To estimate the expectation, we can draw a large number of random variables $\{x_i\}_{i=1}^M$ from the density $p(x)$, assuming that $x_1,\ x_2, \cdots,\ x_M$ are independent and identically distributed. Then, we can compute

$$\widehat{f}(x) = \frac{1}{M}\sum_{i=1}^{M} f(x_i),\ \ x_i \sim p(x). \tag{5.2}$$

When the number of samples increases, the approximated value of the expectation converges to the exact expectation. This process is called Monte Carlo integration. Monte Carlo integration is based on the fact that using the law of large numbers (Robert and Casella, 1999) we can get the following,

$$I \approx \widehat{f}(x).$$

By taking the variance of (5.2),

$$\begin{aligned}\mathbb{V}(\widehat{f}(x)) &= \mathbb{V}(\frac{1}{M}\sum_{i=1}^{M} f(x_i)) \\ &= \frac{1}{M^2}\mathbb{V}(\sum_{i=1}^{M} f(x_i)).\end{aligned}$$

Since $x_1, x_2, \cdots, x_M$ are independent and identically distributed, then

$$\mathbb{V}(\sum_{i=1}^{M} f(x_i)) = \sum_{i=1}^{M}\mathbb{V}(f(x_i)) = M\mathbb{V}(f(x_1)).$$

Therefore, the variance for the estimator is equal to

$$\mathbb{V}(\widehat{f}(x)) = \frac{1}{M}\mathbb{V}(f(x_1)). \tag{5.3}$$

The root mean squared error of $\widehat{f}$ is

$$\text{RMSE}(\widehat{f}) = \sqrt{\mathbb{V}(\widehat{f}(x))} = \sqrt{\frac{1}{M}\mathbb{V}(f(x_1))} \tag{5.4}$$

To obtain the root mean squared error of order ϵ, it is required that $M = \mathcal{O}(\epsilon^{-2})$ (Gilks et al., 1996). We give an example how to use Monte Carlo Integration to solve an improper integral (Khouider, 2008).

Example 5.1.1.1 *Calculate* $I = \int_{-\infty}^{\infty} \cos(x)\exp(-x^2/2)\text{dx}$.

The exact solution is $I = \sqrt{\frac{2\pi}{e}} \approx 1.52035$. *Compared to (5.1),* $f(x) = \cos(x)$ *and* $p(x) = 1/\sqrt{2\pi}\exp(-x^2/2)$. *Therefore, the approximate solution using Monte Carlo integration is:*

$$I = \sqrt{2\pi}\int_{-\infty}^{\infty} \cos(x) \underbrace{\frac{\exp(-x^2/2)}{\sqrt{2\pi}}}_{\textit{Normal density}} \text{dx} = \mathbb{E}_{\mathcal{N}}(\cos(x))$$
$$\approx \frac{\sqrt{2\pi}}{M}\sum_{i=1}^{M}\cos(x_i),\ x_i \sim \mathcal{N}(0,1)$$

Number of samples (M)	Approximate solution	Absolute error
100	1.41911	0.1012
10000	1.50843	0.0119
1000000	1.52143	0.00108

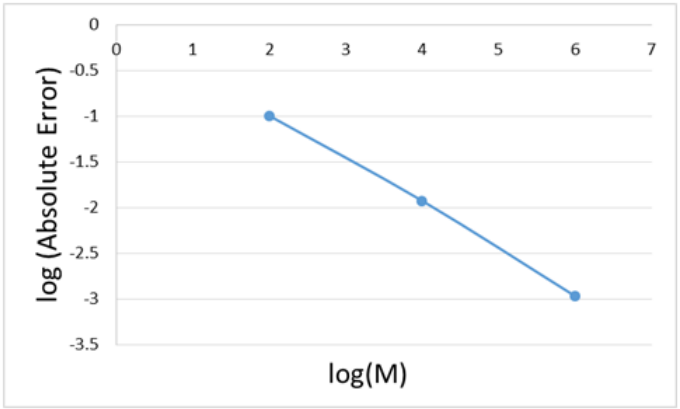


Figure 5.1: Absolute error as a function of the number of samples.

Figure 5.1 shows that the Monte Carlo error is $\mathcal{O}(1/\sqrt{M})$. Before giving another example, we define Brownian motion.

Definition 5.1.1.1 *(Oksendal, 2002) A family of random variables $W = (W_t)_{t\geq 0}$ is called Brownian motion if the following conditions are satisfied:*

- *The random variables $W_{t_1} - W_{t_0}; W_{t_2} - W_{t_1}; \cdots; W_{t_n} - W_{t_{n-1}}$ are independent for every finite sequence of $0 \leq t_0 < t_1 < t_2 < \cdots < t_n$ for every integer $n \geq 1$*
- *For $0 \leq s < t$ the random variable $W_t - W_s \sim \mathcal{N}(0, t-s)$*
- *$W_t(\omega)$ is continuous in t for every $\omega \in \Omega$*
- *$W_0 = 0$.*

Example 5.1.1.2 *Estimate the expectation of the solution of the following stochastic differential equation using MCI (Monte Carlo integration)*

$$\mathrm{dX(t)} = 0.06X\mathrm{dt} + 0.4X\mathrm{dW(t)} \tag{5.5}$$

where, $W(t)$ is Brownian motion. The Euler-Maruyama scheme (Lord et al., 2014) for (5.5) is

$$X^{n+1} = X^n + 0.06\ \triangle t\, X^n + 0.4\, X^n \sqrt{\triangle t}\, V^n,\ V^n \sim \mathcal{N}(0,1). \tag{5.6}$$

We use the MCI to estimate the mean of the solution, $\mathbb{E}(X(t))$, $t \in [0,1]$ with $\triangle t = 0.01$, $X_0 = 1$. Moreover, the exact expectation for the problem can be calculated based on the one-dimensional Itô formula (Lord et al., 2014). The exact solution can be written as follows,

$$X(t) = \exp((0.06 - 0.4^2/2)t + 0.4W(t)).$$

The expectation is

$$\mathbb{E}(X(t)) = \exp(0.06\, t). \tag{5.7}$$

Figure 5.2 shows that the absolute error between the exact expectation (5.7) and the approximate expectation of (5.6) using MCI is of order $\mathcal{O}(10^{-3})$.

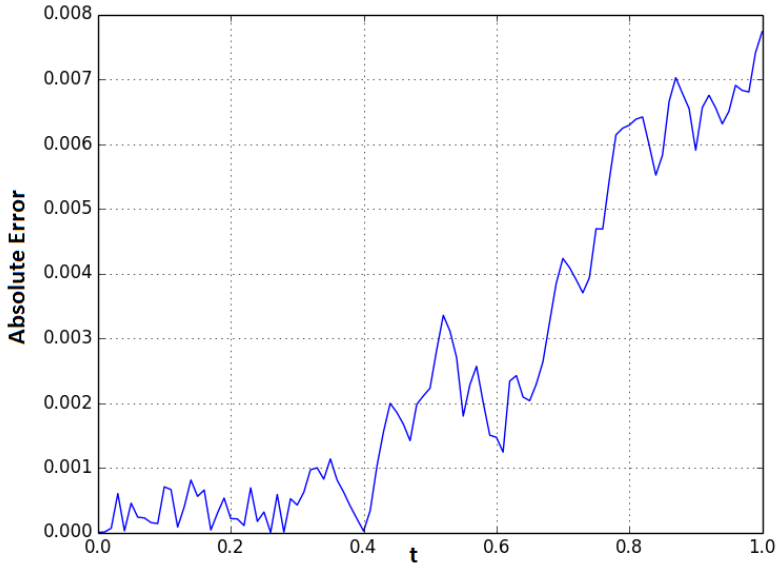


Figure 5.2: Absolute error for estimating the mean of (5.6) using MCI with 10000 samples.

The sample mean becomes more accurate when the sample variance decreases. However, there is no rule to say this is a good estimation for the sample mean. There are several techniques based on variance reduction to achieve a better estimation of the sample mean e.g., Multilevel Monte Carlo (MLMC). Before we explain MLMC, we explain Two-level Monte Carlo, then we can generalize the idea.

5.2 Two-level Monte Carlo

The idea is expressing the stochastic variable U_{l_1} in terms of another stochastic variable U_{l_0} (functions of the solution of stochastic differential equation has random parameters or random initial conditions) but the computational cost to calculate U_{l_0} is cheaper than U_{l_1} as follows

$$U_{l_1} = U_{l_0} + (U_{l_1} - U_{l_0})$$

Since the expectation is a linear operator, then

$$\mathbb{E}(U_{l_1}) = \mathbb{E}(U_{l_0}) + \mathbb{E}(U_{l_1} - U_{l_0}). \tag{5.8}$$

To estimate $\mathbb{E}(U_{l_1})$, we use the sample average for each terms in (5.8) as follows

$$\mathbb{E}(U_{l_1}) \approx \frac{1}{M_{l_0}} \sum_{i=1}^{M_{l_0}} U_{l_0}^i + \frac{1}{M_{l_1}} \sum_{i=1}^{M_{l_1}} (U_{l_1}^i - U_{l_0}^i) \tag{5.9}$$

where, M_{l_0} and M_{l_1} are the number of samples required to estimate the first and the second term in (5.8) respectively. For the differences between U_{l_1} and U_{l_0}, we generate the parameters $m_j, j = 1, \cdots, d$, where d is the number of uncertain parameter, we calculate $(U_{l_1}(m_i) - U_{l_0}(m_i))$. If we assume the variance of U_{l_0} is V_{l_0}, the variance of $U_{l_1} - U_{l_0}$ is V_{l_1} and the cost of computing per sample for U_{l_0} and $U_{l_1} - U_{l_0}$ are C_{l_0} and C_{l_1} respectively, the total computational cost can be defined as follows,

$$C_{tot} = C_{l_0} M_{l_0} + C_{l_1} M_{l_1}.$$

Because U_{l_0} and $U_{l_1} - U_{l_0}$ are independent, the variance of the estimator can be written as,

$$\mathbb{V}(\mathbb{E}(U_{l_1})) = \frac{V_{l_0}}{M_{l_0}} + \frac{V_{l_1}}{M_{l_1}}.$$

Minimizing the variance of the estimator by using Lagrange multiplier λ under the constraint

$$C_{tot} = \text{constant},$$

within M_{l_0}, $M_{l_1} \in \mathbb{R}^+$ yields

$$\frac{\partial}{\partial M_{l_i}} \left(\frac{V_{l_0}}{M_{l_0}} + \frac{V_{l_1}}{M_{l_1}} + \lambda(C_{l_0} M_{l_0} + C_{l_1} M_{l_1} - \text{constant})\right) = 0,\ i = 0,\ 1.$$

From this, we obtain the following constraints

$$M_{l_0}^2 = \frac{V_{l_0}}{\lambda\, C_{l_0}},\ M_{l_1}^2 = \frac{V_{l_1}}{\lambda\, C_{l_1}}.$$

Which leads to our estimate for the ratio of samples to maximise efficiency

$$\frac{M_{l_1}}{M_{l_0}} = \frac{\sqrt{V_{l_1} / C_{l_1}}}{\sqrt{V_{l_0} / C_{l_0}}} = \sqrt{\frac{V_{l_1}}{V_{l_0}}} \sqrt{\frac{C_{l_0}}{C_{l_1}}}.$$

Since the computational cost per sample for U_{l_0}, C_{l_0} is less than the computing cost per sample for $U_{l_1} - U_{l_0}$, C_{l_1} and $V_{l_1} < V_{l_0}$ (Lord et al., 2014) then,

$$\frac{M_{l_1}}{M_{l_0}} < 1,$$

which means by minimizing the variance of the mean estimator we obtain the number of samples required to estimate the correction term M_{l_1} is less than the number of samples required to estimate the first term in (5.8).

5.3 Multilevel Monte Carlo (MLMC)

The philosophy of MLMC is "fine grid accuracy with coarse cost" (Giles, 2015). MLMC minimizes the computational cost by combining simulations with low and high accuracy in the most efficient manner.

The basic ideas for MLMC are: using the linearity of the expectation; introducing a hierarchy of computational models, which are convergent. MLMC replaces the expensive estimate of the expectation on the fine grid, with a cheaper but less accurate estimate of the mean on the coarse grid and estimate the mean for the differences of output quantities from two consecutive models. This correction term vanishes when the model resolution increases. In that sense, the variance of the difference becomes smaller, making it cheaper as well. We replaced one estimator on the finest grid by $L+1$ estimators, if we have L levels (Giles, 2008, 2015; Dodwell et al., 2015)

The general form for MLMC, based on equation (5.8) is:

$$\mathbb{E}(U_L) = \mathbb{E}(U_{l_0}) + \sum_{l=l_0+1}^{L} \mathbb{E}(U_l - U_{l-1}) \tag{5.10a}$$

$$\approx \frac{1}{M_{l_0}} \sum_{i=1}^{M_{l_0}} U_{l_0}^i + \sum_{l=l_0+1}^{L} (\frac{1}{M_l} \sum_{i=1}^{M_l} (U_l^i - U_{l-1}^i)) \tag{5.10b}$$

$$= \mu_{l_0} + \sum_{l=l_0+1}^{L} \mu_l \tag{5.10c}$$

$$= \sum_{l=l_0}^{L} \mu_l, \tag{5.10d}$$

where, $l = l_0, l_1, \cdots, L$ refers to level (corresponding to different grid size), μ_l is the estimator for the expectation over level l. We define C_{l_0}, C_l for the computational cost of one sample of U_{l_0}, and for $U_l - U_{l-1}$ respectively and let V_{l_0}, V_l be the variance of U_{l_0}, $U_l - U_{l-1}$ respectively. Because of the independence between the estimators therefore, the total cost and the variance of multilevel estimator can be written as follows,

$$C_{tot} = \sum_{l=l_0}^{L} C_l \, M_l$$

$$\mathbb{V}(\mathbb{E}(U_L)) = \sum_{l=l_0}^{L} \frac{V_l}{M_l}.$$

(Giles, 2015), to achieve overall variance of order ϵ^2, the Lagrange multiplier must be equal to

$$\lambda = \epsilon^4 / (\sum_{l=l_0}^{L} \sqrt{V_l C_l})^2$$

when we use the same procedures to obtain the number of samples as in the Two-level Monte Carlo. The number of samples required for every level is

$$M_l = \frac{1}{\epsilon^2} \sqrt{\frac{V_l}{C_l}} \sum_{l=l_0}^{L} \sqrt{V_l C_l}. \tag{5.11}$$

Substituting into total computational cost, we get

$$C_{tot} = \sum_{l=l_0}^{L} C_l M_l = \frac{1}{\epsilon^2}(\sum_{l=l_0}^{L} \sqrt{V_l C_l})^2. \tag{5.12}$$

The performance of MC depends on whether the product $V_l C_l$ increases or decreases or stays 'almost constant' when l increases. If $V_l C_l$ decreases (increases), then the dominant term for the cost is $V_{l_0} C_{l_0}$ ($V_L C_L$) i.e. $C_{tot} \approx \epsilon^{-2} V_{l_0} C_{l_0}$ ($\epsilon^{-2} V_L C_L$). For the Monte Carlo, the cost is approximated by $\epsilon^{-2} V_{l_0} C_L$, assuming $V_{l_0} \approx V(U_L)$. In the case of $V_l C_l$ increase (decrease), MLMC reduces the computational cost compared with the Monte Carlo (Giles, 2015).

The complexity cost for using MLMC is $\mathcal{O}(((\log \epsilon)/\epsilon)^2)$. The squared of the sampling error can be written as the summation of the squared of the sampling error from each level. This means that MLMC splits the errors into smaller errors for each level.

5.3.1 The History of MLMC

MLMC has been used to study different SPDEs. MLMC was proposed by Heinrich in his work (Heinrich, 2001, 2006) and involves estimating $\mathbb{E}(f(x, \lambda))$, where λ is a parameter and x is a finite dimensional random variable. Heinrich discussed how he estimated $\mathbb{E}(f(x, \lambda))$ when $\lambda \in [0, 1]$. He used a geometric sequence of levels in his work.

Kebaier developed the approach of two-level for path simulation in (Kebaier, 2005). (Giles, 2008) also developed the approach of two-level for path simulation to multilevel based on the multigrid ideas from (Brandt et al., 1994). (Müller, 2014; Müller et al., 2013) studied single and two phase flow with a random heterogeneous porous media. He discussed how to solve an elliptic equation with a random permeability using MLMC. He used MLMC to accelerate the Monte Carlo using multi grid techniques. He focused on the error associated with MLMC when computing finite samples.

In 2014, Müller compares streamline-based solver and grid-based solver using

MLMC for two phase flow and transport in a random heterogeneous porous media (Müller et al., 2014). MLMC has been used to study elliptic PDEs with random coefficients (Cliffe et al., 2011; Teckentrup et al., 2013; Barth et al., 2011; Zollinger, 2010). In (Cliffe et al., 2011) the authors study the solution of elliptic PDEs with random coefficients. They quantify the uncertainty for groundwater flow using MLMC. They address the problem of the large cost of solving elliptic PDEs with random coefficients. Moreover, it is used to solve Feynman-Kac Formula for the Laplace equation (Pauli et al., 2015). Also Giles and others estimate distribution functions and densities using MLMC (Giles et al., 2015). (Efendiev et al., 2013) used MLMC with the finite element method to study two phase flow and transport simulation and (Alkhatib, 2014) used MLMC for quantifying the spatial uncertainty for the enhanced oil recovery process.

5.3.2 MLMC Implementation

We discuss how to implement MLMC in a more general way. To estimate the first term in (5.10a), we use MCI for generating the uncertain parameters m_i, $i = 1, \cdots, d$ (d is the number of parameters) using Latin Hypercube Sampling (LHS) or Sobol sequence sampling technique or random sampling [described in Chapter 7]. Therefore,

$$\mathbb{E}(U_{l_0}) = \frac{1}{M_{l_0}} \sum_{j=1}^{M_{l_0}} U_{l_0}(m_i^j),\ \forall i = 1, \cdots, d.$$

To estimate the correction terms, which is the second term in (5.10a), we use MCI for generating the uncertain parameters m_i, $i = 1, \cdots, d$ (d is the number of parameters) using LHS, Sobol sequence or random sampling [described in Chapter 7]. Therefore,

$$\mathbb{E}(U_l - U_{l-1}) = \sum_{l=l_0+1}^{L} \frac{1}{M_l} \sum_{j=1}^{M_l} U_l(m_i^j) - U_{l-1}(m_i^j),\ \forall i = 1, \cdots, d.$$

U_l is the solution of stochastic ODEs or stochastic PDEs using a finite difference scheme augmented with grids defined for level l, which is finer than the grid for level $l-1$.

In the following, we discuss in more details how to implement MLMC for estimating the expectation of a function that depends on time, $U(t)$ (Stochastic ODEs) and estimating the expectation of a function that depends on time and one-dimensional space, $U(x,t)$ (Stochastic PDEs).

5.3.2.1 MLMC for Stochastic ODEs

In the first problem (5.4.1), we will estimate the mean of the solution at fixed time T, when the solution of the Stochastic ODE is a function of time. We use a numerical scheme for finding U_l, $\forall l$. The set up for MLMC is using a ladder in the time steps $\triangle t_l$ rather than a fixed time $\triangle t$ such that $\triangle t_l = T/\eta^l$, where η is a natural number greater than 1 and $l = l_0, l_1. \cdots, L$ is the level. For each level, we calculate U_l, which is the approximation to $U(T)$ using $\triangle t_l$ time step. From our choice of $\triangle t_l$, we can see the the largest time step is $\triangle t_{l_0}$. We choose $\triangle t_{l_0}$ to be smaller than the time step, which is required for stability (Lord et al., 2014; Giles, 2008). To achieve the accuracy of the estimator $\mathcal{O}(\epsilon)$, we choose $\triangle t_L < \epsilon/2$, then $L = \left\lceil \frac{\log(2T/\epsilon)}{\log(\eta)} \right\rceil$.

We use the telescoping sum for estimating the expectation (5.10). The telescoping sum relation says that we can express the estimation of the expectation on the finest level L with the smallest time step $\triangle t_L$ in terms of the estimation on the coarsest level l_0 with the largest time step $\triangle t_{l_0}$ and correction terms.

5.3.2.2 MLMC for Stochastic PDEs

In case of studying Advection, Buckley-Leverett and pressure equations (see Section 5.4), we will estimate the mean of solution at fixed time T for all x in the space when the solution of Stochastic PDEs is $U(x,t)$. We use a numerical scheme for finding U_l, $\forall l$. The set up for MLMC is using a ladder for the time steps as in the case of the function of time only. However, we will also have a ladder in the x steps based on the Courant Friedrichs Lewy (CFL) condition. We solve the problems augmented with the coarsest grid using initial samples M_{up}, then estimate the mean and variance of the solution $U(x,T)$. To check the solution accuracy of ϵ, we have to calculate the left hand side of (5.11) and then calculate the extra samples required using $M_l - M_{up}$. Since the variance $\mathbb{V}_l$ is a function of x therefore, for each point of

x, we have a number of samples required to achieve the accuracy of the estimator $\mathcal{O}(\epsilon)$ (point wise). To illustrate, Figure 5.3 shows the number of samples required for 5 points in the space, let $m_1 = 10$, $m_2 = 100$, $m_3 = 50$, $m_4 = 75$ and $m_5 = 5$.

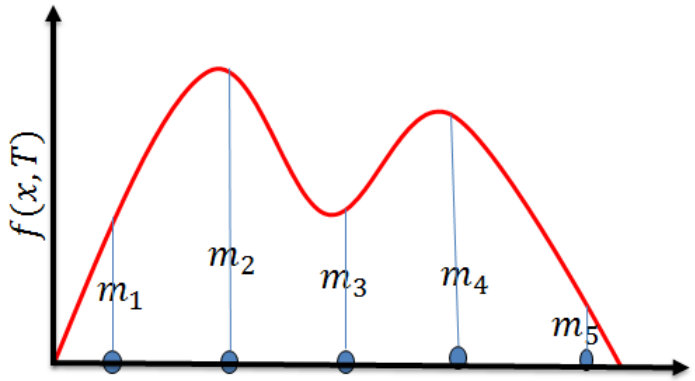


Figure 5.3: Number of samples required corresponding to each point in the space.

In practice, we have to choose how many samples we need to use for solving the problem.

$$\text{control}(\mathbb{V}(\mathbb{E}(U_L))) \leq \epsilon^2/2. \tag{5.13}$$

We can use different options for the control e.g., minimum, first quantile $1Q$, mean, median, third quantile $3Q$ and maximum of the number of samples with respect to x as shown in Table 5.1

Table 5.1: Different choices for the number of sample required to use for solving the SPDE.

Option	Number of samples (M)
minimum $\{m_1, m_2, m_3, m_4, m_5\}$	5
$1Q$ $\{m_1, m_2, m_3, m_4, m_5\}$	10
mean $\{m_1, m_2, m_3, m_4, m_5\}$	48
median $\{m_1, m_2, m_3, m_4, m_5\}$	50
$3Q$ $\{m_1, m_2, m_3, m_4, m_5\}$	75
maximum $\{m_1, m_2, m_3, m_4, m_5\}$	100

Therefore, we solve the problem for the number of samples required, which can be chosen to be the mean of the sample required, $M = 48$.

5.3.3 MLMC Algorithm

The following algorithm describes how MLMC works. Algorithm 4 step 1, we need to solve the problem for initial samples (warm-up samples M_{up}) because the idea

for MLMC is checking the condition (5.13) and running until the required samples satisfy this condition.

Algorithm 4: MLMC (Cliffe et al., 2011; Müller et al., 2013)

1: Fix a sequence of grid resolutions $l = l_0, \cdots, L$, fix a number of warm-up samples M_{up}, fix the variance of the error in the observed data V_0 and also the accuracy ϵ

2: Starting with $l = l_0$, compute $M_{l_0} = M_{up}$ and then check the convergence, if it is satisfied then, go to step 3. Otherwise add more samples.

3: Warm-up phase: Compute $M_l = M_{up}$ samples of $U_l - U_{l-1}$ on every level.

4: Update the mean estimator. Then, update the variance of the estimator and the cost for each level.

5: Solve the optimization problem and update the required number of samples M_l (5.11). In other words, evaluate extra samples at each level if required and then check the condition (5.13). For each level we can go back and add more samples to satisfy the condition.

6: Set $l = l + 1$ and go back to step 3.

The choice of warm-up samples, M_{up}, is somewhat delicate. It should be large enough to determine the variance of our estimator accurately. On the other hand, if M_{up} is larger than any M_l suggested by the optimization problem in step 5, unnecessary samples are calculated and computation time is wasted.

Figure 5.4 shows the flowchart of MLMC. Check the condition, which mentioned in the flowchart is (5.13). The initial samples for each level is M_{up} and then based on the variance of the mean of the M_{up} samples and the condition (5.13), we know how many samples requires to run to obtain a suitable accuracy for the mean of solution.

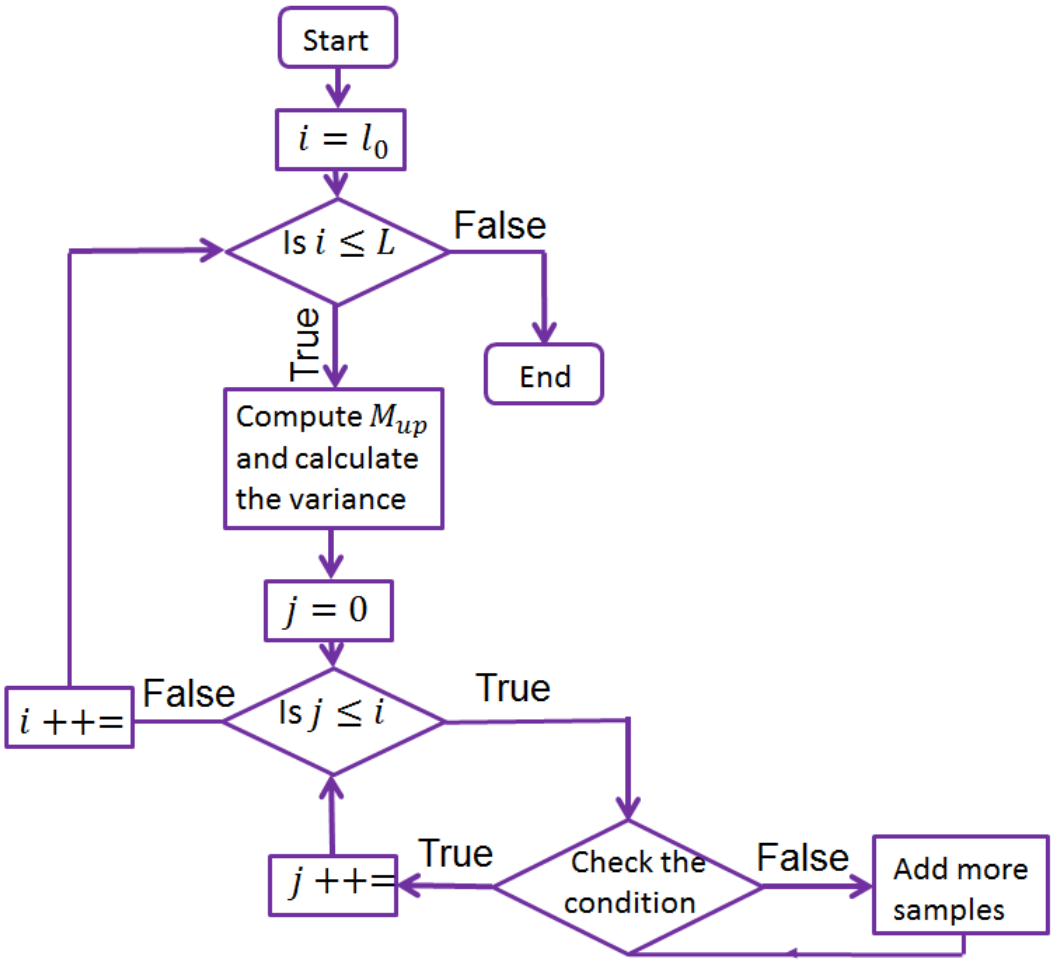


Figure 5.4: The flowchart of MLMC.

In the next section, we discuss 4 problems: the exponential growth and decay equation, the advection equation, the Buckley-Leveret equation and the pressure equation. We estimate the mean of the solution and the uncertainty range by using MLMC and MCI. The next section compares the performance of MLMC and MCI for solving the problems.

5.4 Applications

In this section, we study 4 problems, the first one as a toy problem. The second is a linear hyperbolic equation, the third is nonlinear hyperbolic equation and the last one is a parabolic equation (linear or semi-linear). The aim is to test the applicability of MLMC to solve hyperbolic and parabolic equations, which are the types of the PDE in the reservoir simulation.

For all the results presented in this section, they are averaged over five runs each

with many samples. The choice of 5 is based on the convergence of the solution and if we add more run does not add more information about the solution. For example, in case of estimating the mean of exponential growth and decay problem at $t = 1$. We know the exact mean at $t = 1$ in case of generating the initial condition from a uniform distribution, $\mathcal{U}(0, 1)$ is equal to $e^{0.5}$. Figure 5.5 shows that with different 5 runs the absolute error for fixed accuracy, has the same order of magnitude.

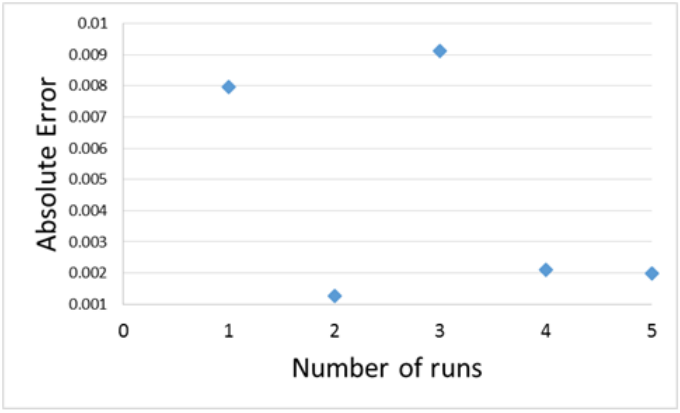


Figure 5.5: Absolute error of estimating the mean of exponential growth and decay problem for a fixed accuracy corresponding to 5 different runs.

5.4.1 Exponential Growth and Decay Equation (Toy example)

The decay ODE can be expressed as follows,

$$\frac{\mathrm{du}}{\mathrm{dt}} = -au, \qquad u(0) = u_0. \tag{5.14}$$

Equation (5.14) has many applications in the real world, for example, to describe cooling/warming law, population growth, radio-active decay, Carbon dating and draining a tank (Siddiqi and Manchanda, 2005). The analytical solution for (5.14) is $u = u_0 e^{-at}$. To solve the problem numerically, if $a > 0$, then we can use the explicit Euler scheme first order as follows,

$$u^{n+1} = u^n - \triangle t\, a\, u^n = (1 - a \triangle t)u^n.$$

In reality, it may be difficult to specify the initial state, however, we might know the range. Why do we have a random initial condition? The ODE will be converted to a random ODE because we do not have a deterministic initial condition any more. The stochastic form for the (5.14) is:

$$\frac{\mathrm{du}}{\mathrm{dt}} = -au,\ u(0) = u_0 \sim \mathcal{U}(0,1). \tag{5.15}$$

Figure 5.6 shows the absolute error between the exact solution and numerical solution (5.14) with initial condition $u_0 = 0.35$ and assuming $a = 1$. The figure shows the absolute error in range $10^{-3} - 10^{-2}$, which means the accuracy of the approximate solution is $\mathcal{O}(10^{-2})$.

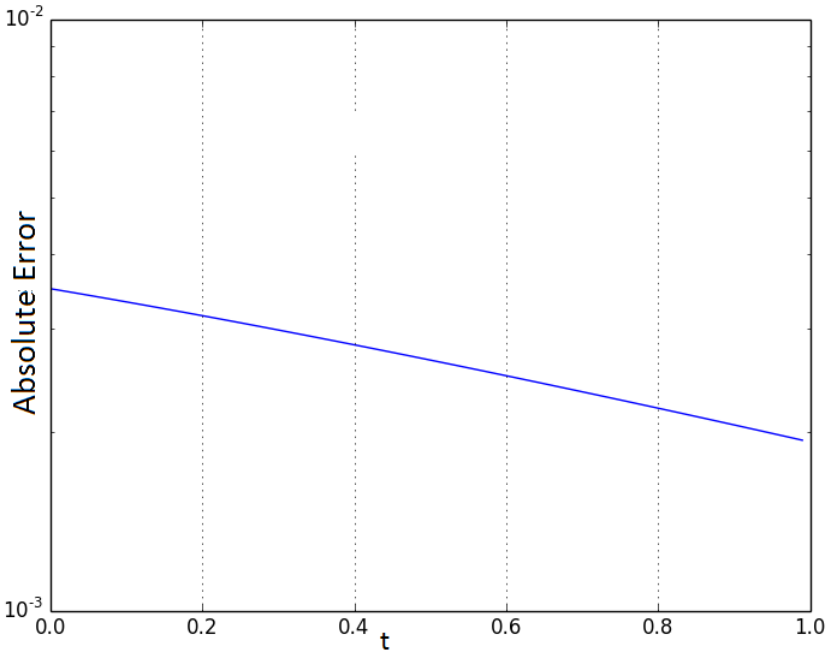


Figure 5.6: Absolute error of (5.14) with Euler explicit scheme.

We solve (5.15) by using MCI and MLMC with $\eta = 4$ ($\eta = \frac{\triangle t_{l-1}}{\triangle t_l} \forall l$) in Section 5.3.2.1. Figure 5.7 shows that MLMC is cheaper than MCI based on the computational cost (5.12, 5.16).

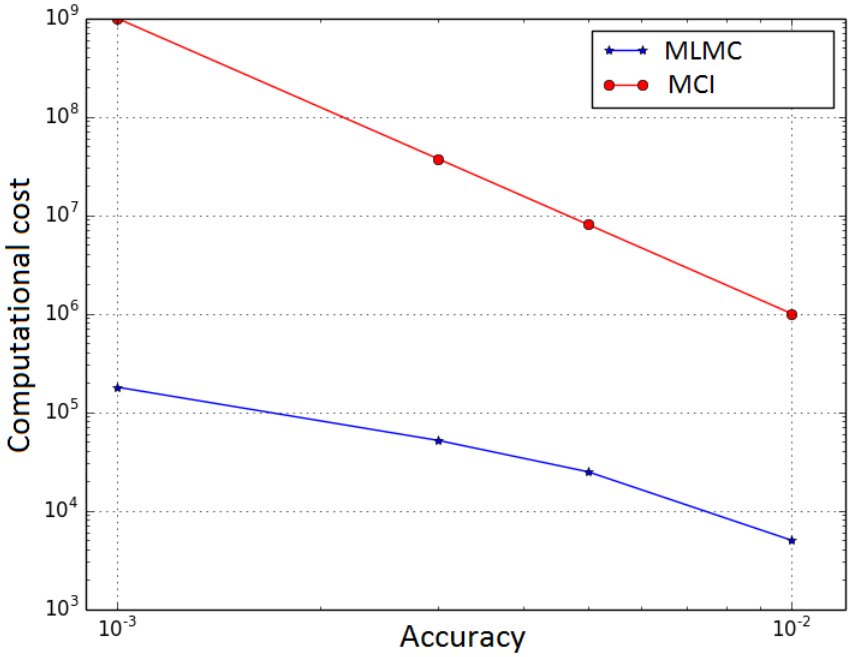


Figure 5.7: Comparison of computational cost from MCI and MLMC for the exponential growth and decay equation problem.

When using the root mean square error for solving the problem, using MCI with explicit Euler, we have two kinds of errors, one from the discretization and another one from Monte Carlo as follows,

$$\begin{aligned}\mathbb{E}(f(X(t))) - \mu_M &= \underbrace{(\mathbb{E}(f(X(t))) - \mathbb{E}(f(X_N)))}_{\text{discretization error}} + \underbrace{(\mathbb{E}(f(X_N)) - \mu_M)}_{\text{Monte Carlo error}} \\ &= \mathcal{O}(\triangle t) + \mathcal{O}(1/\sqrt{M}),\end{aligned}$$

where, M is the number of samples and f should satisfy the Lipschitz condition. If we are looking for an estimation with accuracy of ϵ, it requires $\triangle t = \mathcal{O}(\epsilon)$ and $M = \mathcal{O}(\epsilon^{-2})$. The complexity cost is the total number of steps. Therefore, the cost for using MCI is

$$C = M \times \frac{T}{\triangle t} = \mathcal{O}(\epsilon^{-3}). \tag{5.16}$$

Figure 5.8 shows the speed up factor, the ratio of the computational cost for MCI and the computational cost for MLMC for different accuracy, ϵ values with $\eta = 4$ in Section 5.3.2.1. We obtain the performance speed up in the range of $10^2 - 10^4$.

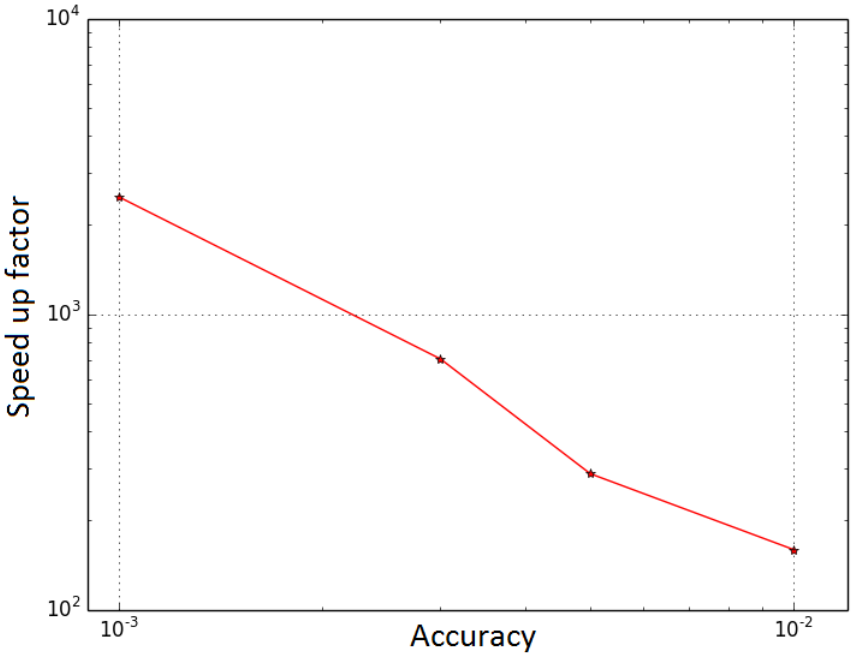


Figure 5.8: Speed up factor for MCI and MLMC for the exponential growth and decay equation problem.

Figure 5.9 compares between the speed up factor based on $\eta = 2$ and $\eta = 4$ in Section 5.3.2.1. It shows in case of using $\eta = 2$ is faster than $\eta = 4$.

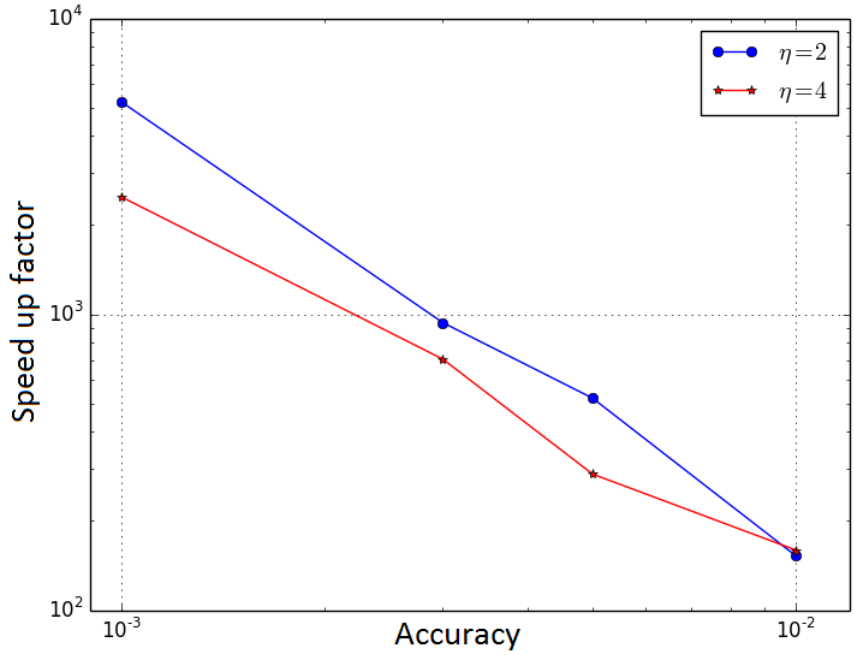


Figure 5.9: Comparison between the speed up factor for $\eta = 2$ and $\eta = 4$ in Section 5.3.2.1 for the exponential growth and decay equation problem.

Figure 5.10 shows the behaviour of the estimators for the correction term corresponding to each level. Also, it shows the line for $\log_4 |\mathbb{E}(U_l - U_{l-1})|$ has a slope of approximately -1, implying an $\mathcal{O}(\triangle t_l)$ weak convergence.

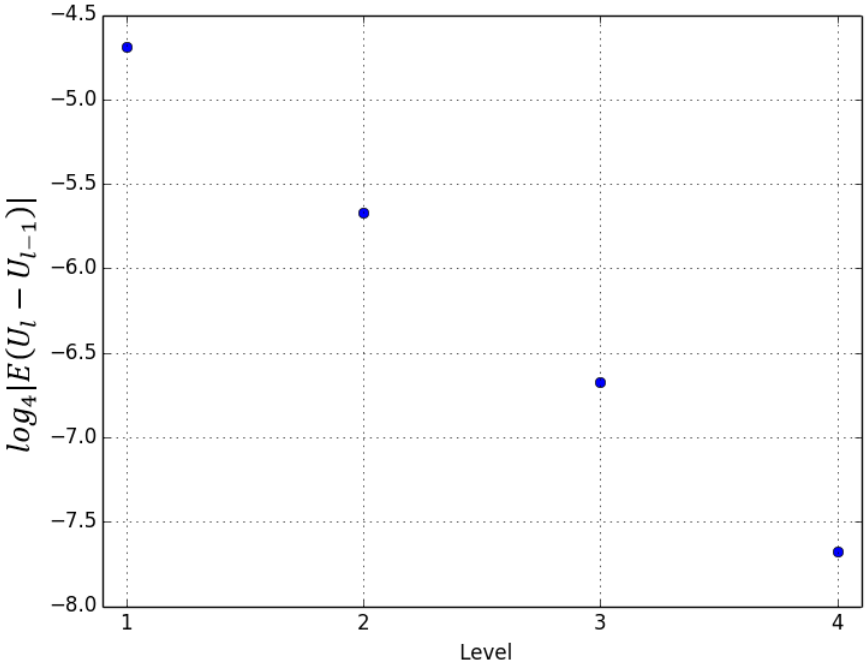


Figure 5.10: The behaviour of the estimators for the correction terms corresponding to every level for the exponential growth and decay equation problem.

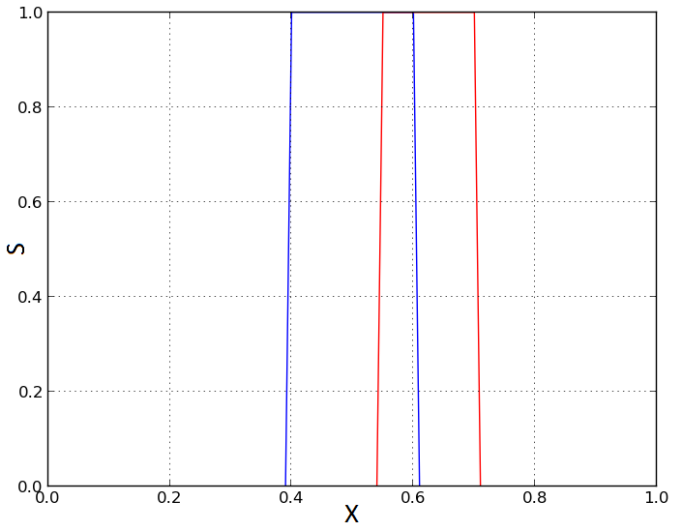
5.4.2 Advection Equation

The advection equation (4.3) has been described in Section 4.3. In Section 4.3, we discussed the analytical solution and in Section 4.3.1, the numerical solution was found using different numerical schemes with different initial conditions. Now, we estimate the mean of the solution of the advection equation with random initial conditions using MCI and MLMC.

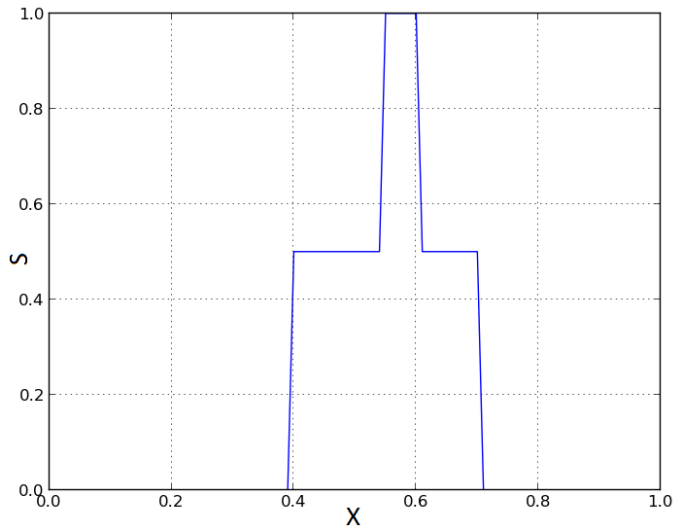
We take the initial condition to be:

$$s(x,0) = \begin{cases} 1 & \min\{a,\, b\} < x < \max\{a,\, b\},\ a,\, b \sim \mathcal{U}(0,1) \\ 0 & \text{otherwise} \end{cases} \tag{5.17}$$

Figure 5.11 shows two random initial conditions based on (5.17). Since the exact solution depends on the initial condition, estimating the mean of the solution is equivalent to averaging over the initial conditions. Figure 5.11(b) shows the average sum of two random initial conditions in Figure 5.11 (a).



(a)



(b)

Figure 5.11: (a) Two random initial conditions based on (5.17), (b) The average sum of two initial conditions in (a).

Figure 5.12 shows that there is not much difference in between using 1000 or 100 grid points in x, the absolute error for both is 'close' to each other.

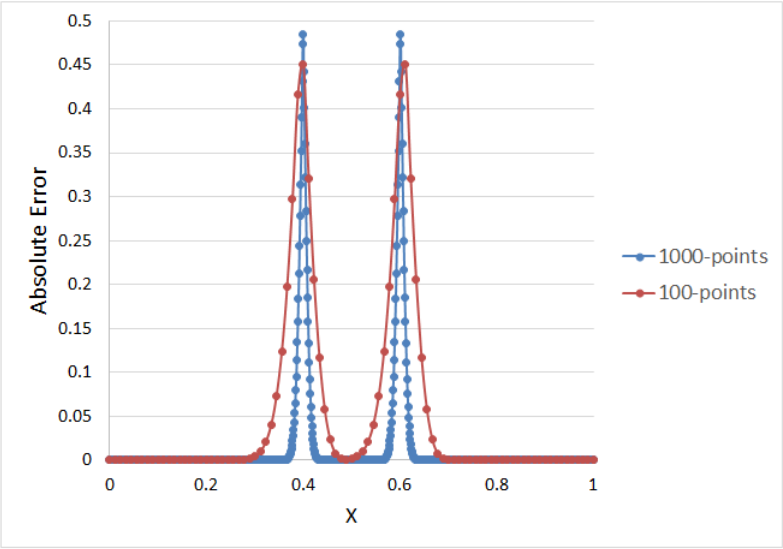


Figure 5.12: The absolute error for different approximate solutions, one with 100 grid points and the other with 1000 grid points.

In Figure 5.13, the top shows the exact solution and approximate solution (Coarse with $\triangle t = 0.01$ and Fine with $\triangle t = 0.005$) with CFL= 0.9. The middle shows the error between the fine solution and the interpolate solution for the coarse solution. The bottom shows the error between the projected fine solution and the coarse solution.

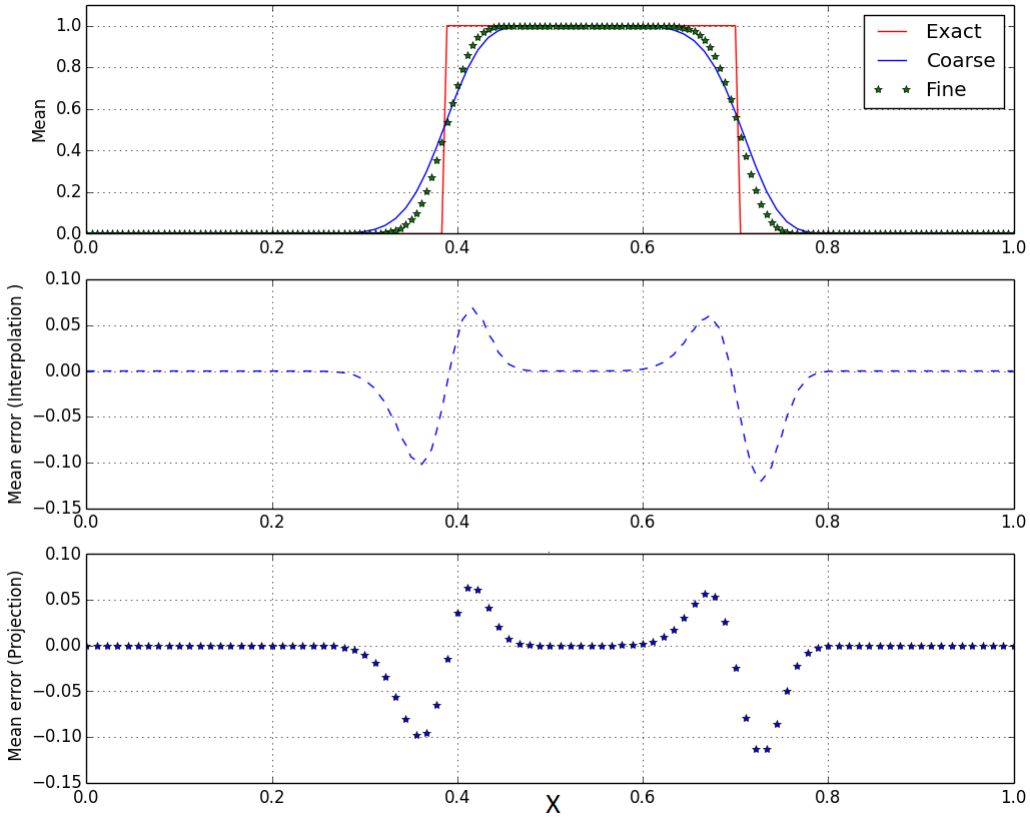


Figure 5.13: The interpolation and projection error using solution with 100 grid points and solution with 200 grid points with CFL= 0.9.

Studying the interpolated and projected error is important because of using MLMC, we have to estimate the correction term which is solved with two different mesh sizes and find the errors. Therefore, we should interpolate or project. All the result present later in this chapter are based on the projection of the solution on the coarse grid. Since the variance and the mean are functions of x, we constrain the number of samples required for every level by the following equation as explained in Section 5.3.2.2

$$mean(\mathbb{V}(\mathbb{E}(U_L)) \leq \epsilon^2/2. \tag{5.18}$$

We use the single point upstream scheme from subsection 4.3.1.1 to solve the advection equation (4.3) with a periodic boundary condition and an initial condition (5.17). We assume the velocity $v = 1$. We find the solution at $t = 1$ as in the Figure 5.14. Figure 5.14 shows the approximate mean solution is smearing close to the boundaries, but it is almost the same for the rest of the domain compared with the exact mean solution, using $CFL = v\lambda = 0.9$.

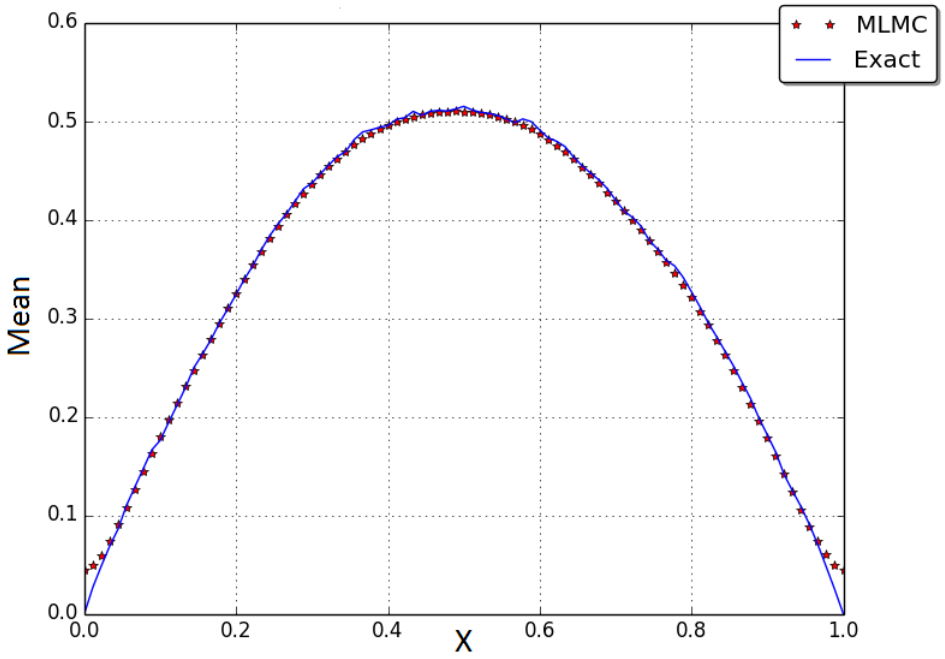


Figure 5.14: Mean of the advection solution with the single point upstream.

Figure 5.15 shows that MLMC is faster than MCI for estimating the mean of the solution for the advection equation. For example, if we need to achieve the accuracy of estimating the mean of the solution is equal to 0.005, it requires 10 minutes using MLMC, while it requires 100 minutes using MCI.

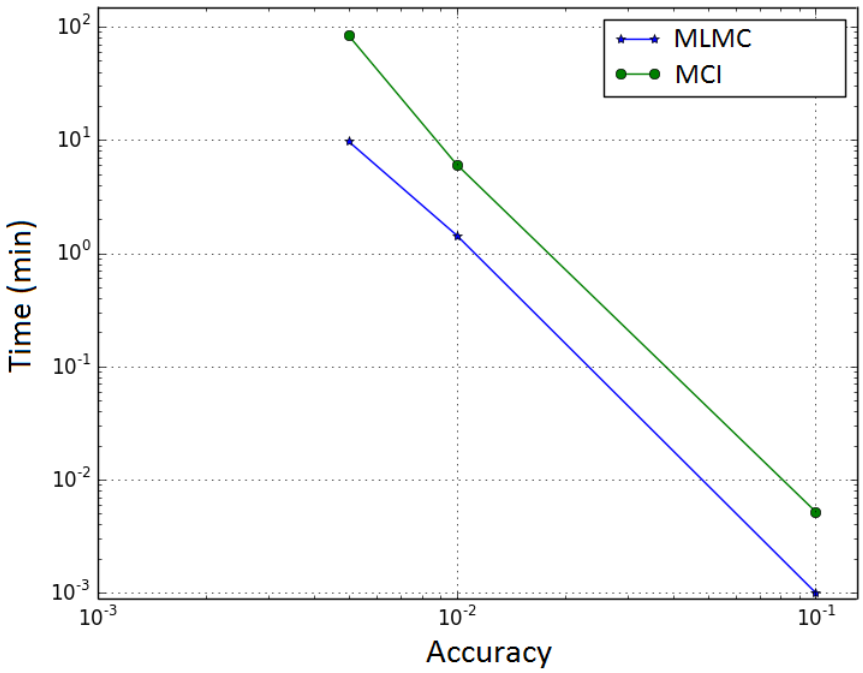


Figure 5.15: Comparison between MLMC and MCI for CPU time in minutes assco-ciated with different accuracy values for the advection equation.

We use second order scheme Lax-Wendroff from the subsection 4.3.1.2, we showed

in Chapter 4 the solution for a deterministic advection equation using Lax-Wendroff has oscillations as shown in Figure 4.3. However, the oscillations disappear in the case of estimating the mean of solution for Stochastic advection as shown in Figure 5.16. We see that as the number of samples is increased, the estimate becomes smoother. In the case of using 100 samples is less smooth than using 10000 samples.

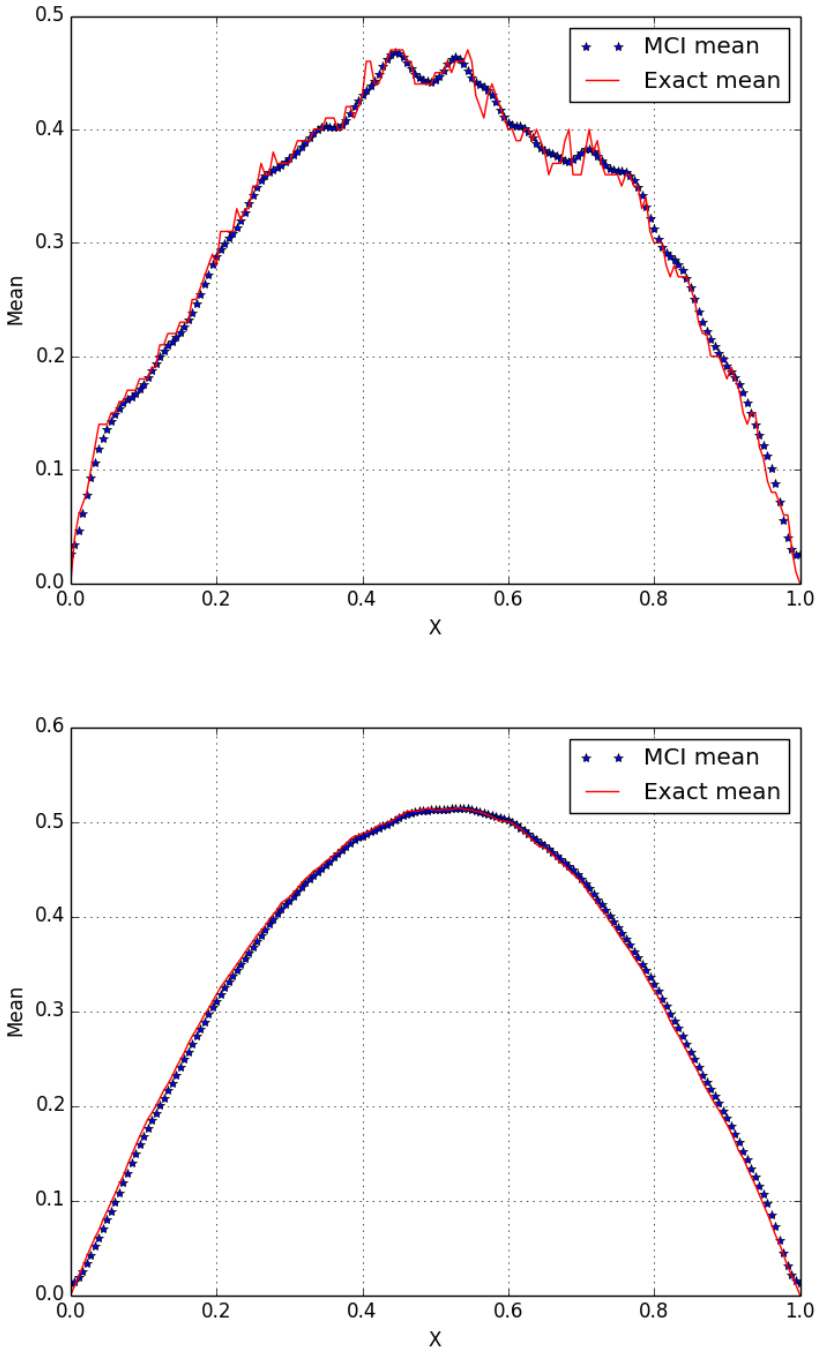


Figure 5.16: Mean of the advection solution with the Lax-Wendroff using (top) 100 samples (bottom) 10000 samples with CFL= 0.9.

Table 5.2 compares the MCI and MLMC based on, CPU time, computational

cost (5.12, 5.16), error and number of samples. We used the Lax-Wendroff scheme from the subsection 4.3.1.2 with accuracy, $\epsilon = 0.01$ under the constraint (5.13), using maximum as control for solving the advection equation. The table shows MLMC is faster than MCI using Lax-Wendroff scheme and both of them have 'almost' the same error.

Table 5.2: Comparison between MCI and MLMC with Lax-Wendroff scheme for solving the advection equation.

		MCI	MLMC
CPU time(min)		21	3
Computational cost		36×10^7	64×10^6
Error		0.012	0.014
Number of samples	Coarse	0	6241
	Fine	10^4	700

Figure 5.17 shows that the error is almost the same when using maximum, median and mean constraint in equation (5.13). However, the mean constraint is the fastest.

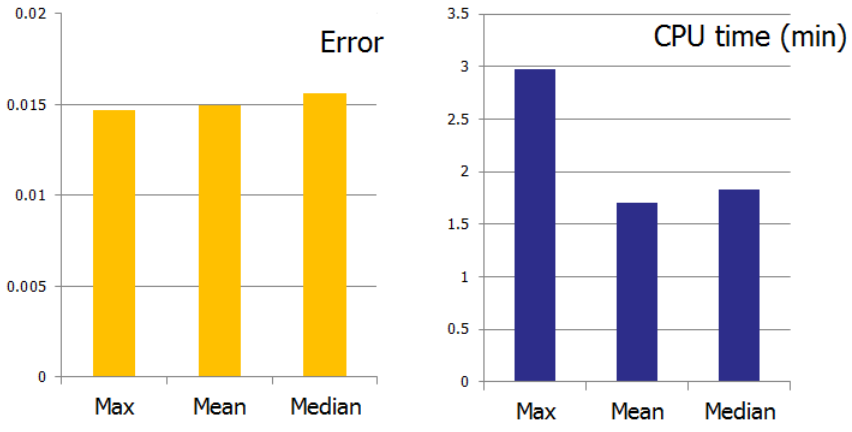


Figure 5.17: Comparison between different constraints for MLMC in terms of error and CPU time for solving advection equation.

In conclusion, MLMC decreases the computational cost compared with MCI in different scenarios for the advection equation, discontinuous random initial condition

and different numerical schemes. There are different ways to control the number of samples required for minimizing the computational cost.

5.4.3 Buckley-Leverett Equation

The Buckley-Leverett equation has been described in Section 4.4. Figure 5.18 shows ten Buckley-Leverett profiles for different flux functions. The flux function that has been used to create this figure is

$$F(s) = \frac{s^p}{s^p + A\,(1-s)^q}, \qquad p,\, q \sim \mathcal{U}(1.5,\, 4) \,\text{and}\, A \sim \mathcal{U}(1,\, 1.5). \tag{5.19}$$

We use the single point upstream scheme to solve the Buckley-Leverett equation

$$\frac{\partial s}{\partial t} + \frac{\partial s}{\partial x}\,\frac{\mathrm{dF}}{\mathrm{ds}} = 0,\; s(x,0) = s_0 \sim \mathcal{U}(0, 0.1) \tag{5.20}$$

with a deterministic boundary condition $s(0,1) = 1$ and a random initial condition as in equation (5.20) and a random flux function as in equation (5.19). We find the approximate distribution for the solution at $t = 1$ as in Figure 5.19a.

Figure 5.19a shows the approximate distribution for the solution using MLMC and MCI using $\lambda = \dfrac{\triangle t}{\triangle x} = 0.3$ (Wang et al., 2013). We applied different constraints for controlling the variance of the mean estimator as in equation (5.13) like maximum, mean and median. We compared between all of them and MCI for fixed accuracy, $\epsilon = 0.01$.

Figure 5.19b shows the 'median' constraint is the cheapest way to get the distribution for the solution. We assume the truth solution if we use MCI with accuracy 0.01. Based on that assumption, we find the L^2 error between MCI and 'Max, Mean, Median ' constraints as in Figure 5.19c. Also, we compared in terms of the number of samples. For example, if we use MCI, then we have to solve on fine grid $10,000$ samples, but for 'Max' constraint, we solve about 6000 samples on coarse grid and 5000 samples on fine grid which is cheaper than MCI with almost the same error.

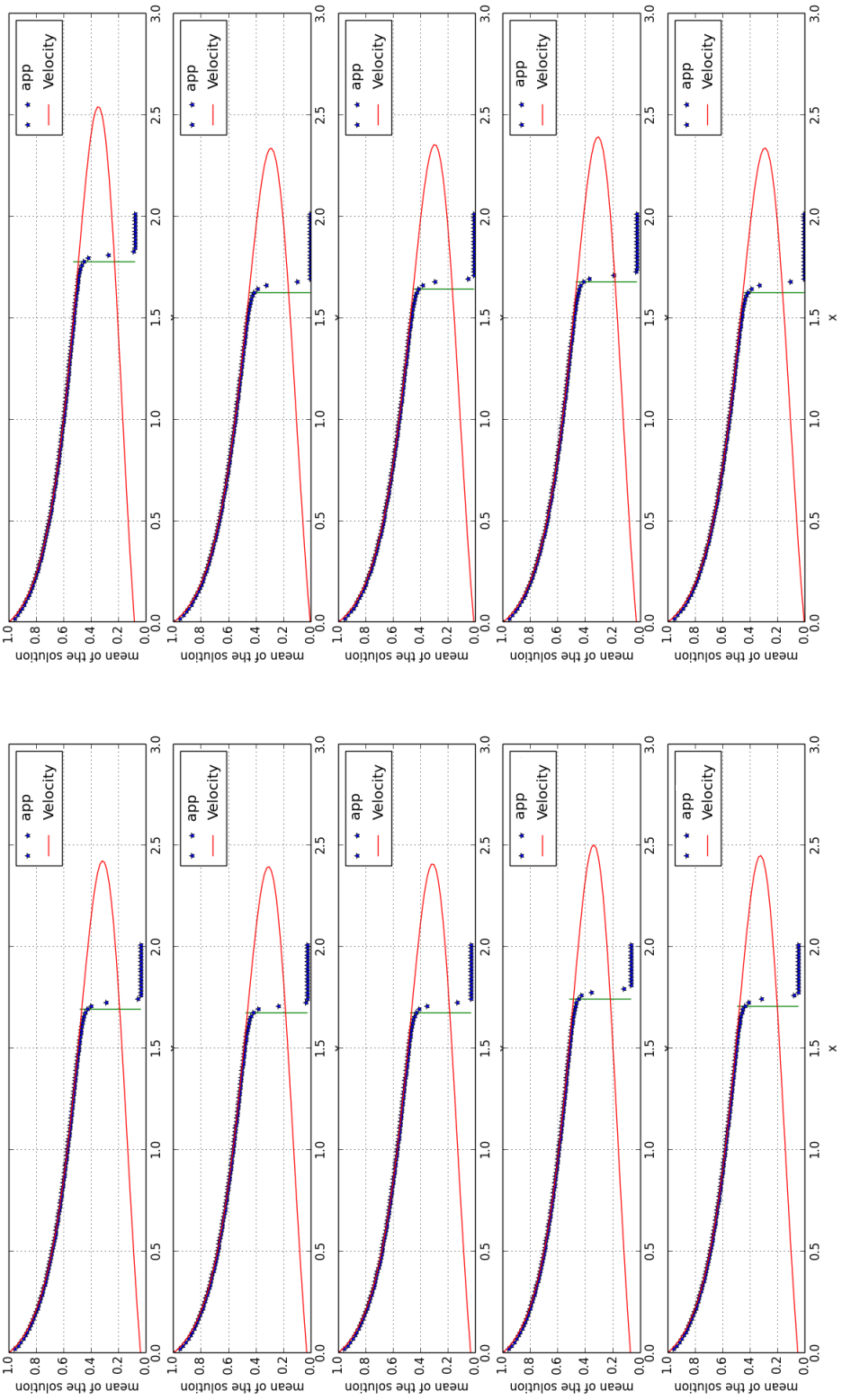
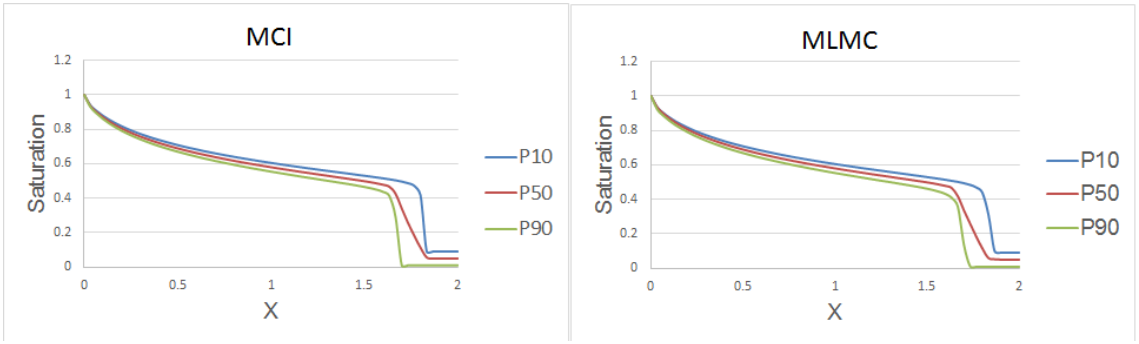
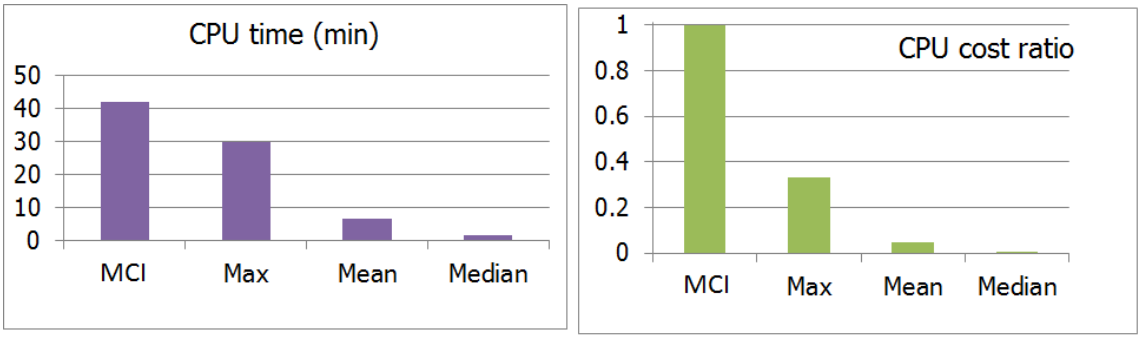


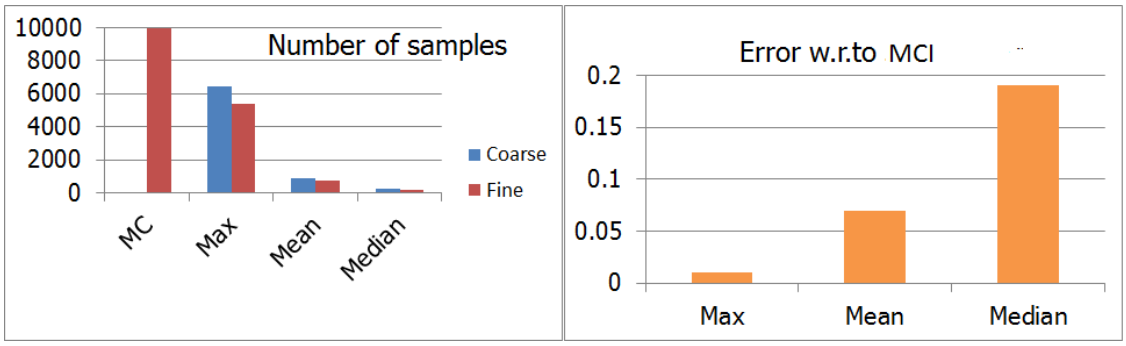
Figure 5.18: Ten Buckley-Leverett profiles associated with different flux function.



(a)



(b)



(c)

Figure 5.19: (a) Estimate of Buckley-Leverett profile distribution using MCI and MLMC. (b) CPU time and cost for MCI with using different constraints for MLMC. (c) Number of samples and error for MLMC with using different constraints compared with MCI.

Figure 5.20 shows that MCI and MLMC under the 'Mean' constraint (5.13) have almost the same distribution, i.e. $P10$, $P50$ and $P90$ with high correlation. The figure shows MLMC is as efficient as MCI.

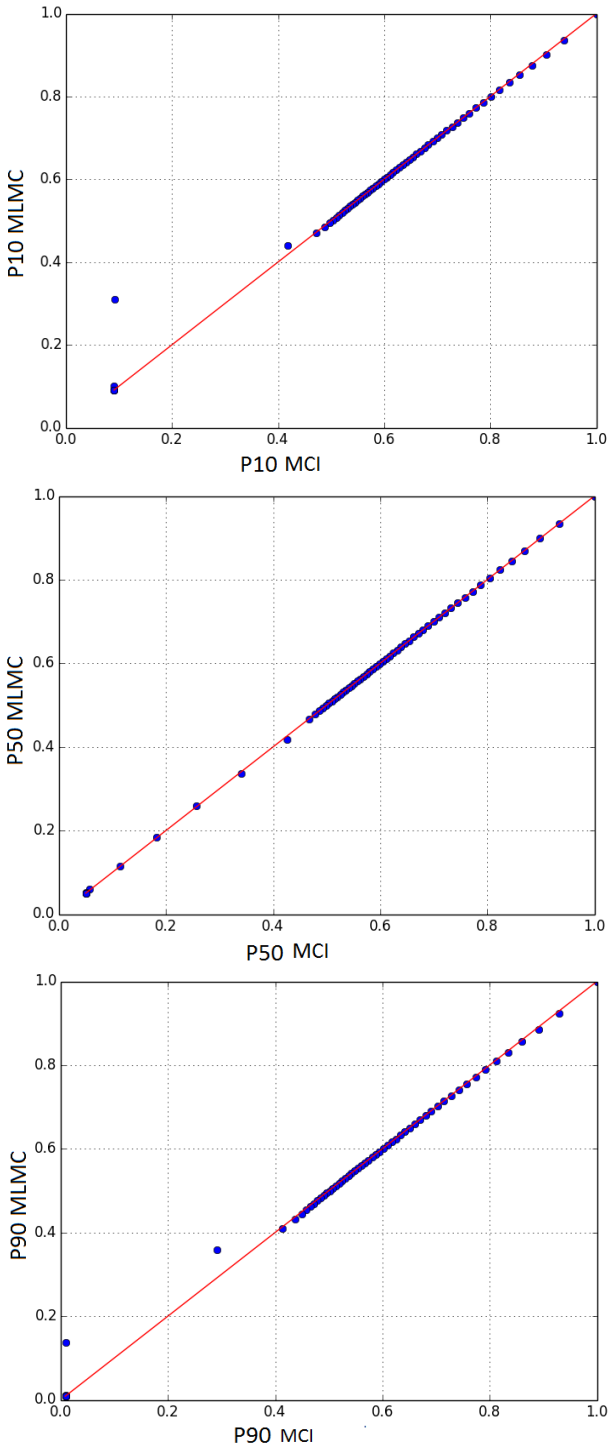


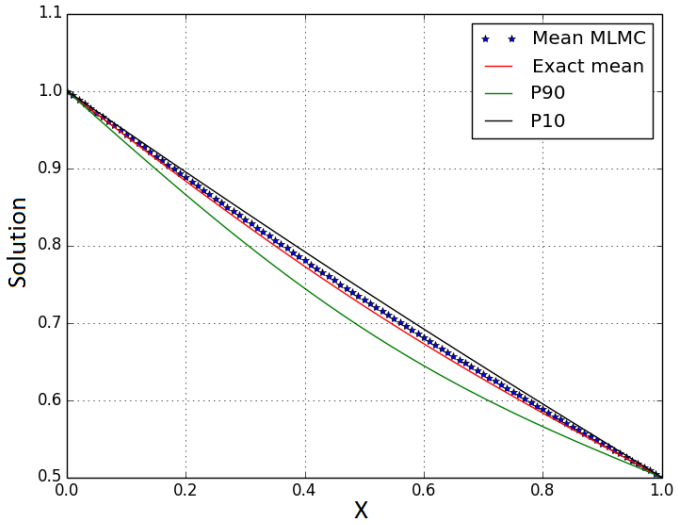
Figure 5.20: Cross plot between MLMC and MCI for solving Buckley-Leverett equation for (top) P10 (middle) P50 (bottom) P90.

In conclusion, MLMC decreases the computational cost compared with MCI.

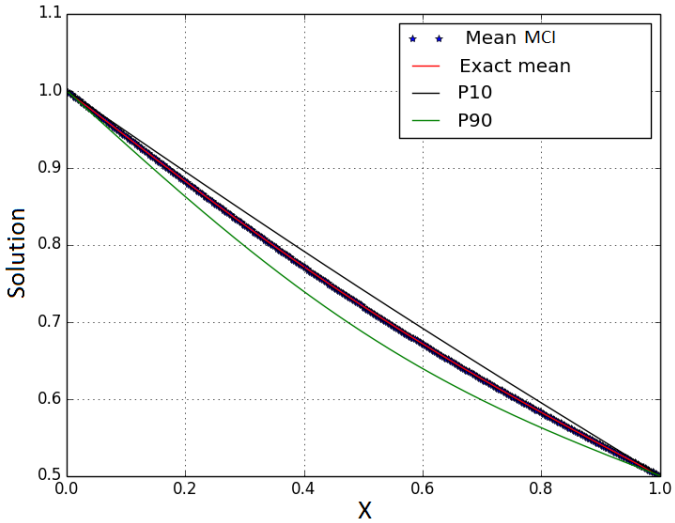
5.4.4 Pressure Equation

The linear equation for the pressure has been described in Section 4.5. In Section 4.5.3, the numerical solution has been found using an implicit scheme with a

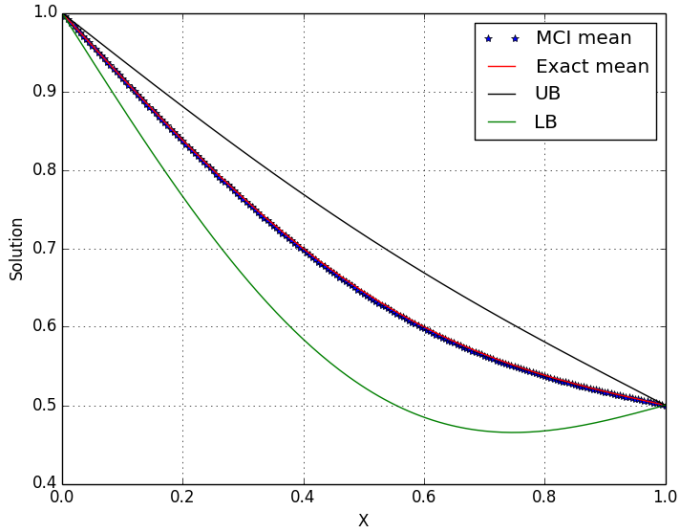
deterministic initial, boundary conditions and a deterministic diffusion coefficient. Now, we estimate the distribution of the solution and compare it with the exact mean solution when the diffusion coefficient is constant and uniformly random distributed, i.e. $D \sim \mathcal{U}(0.25, 0.75)$. We use an implicit scheme to solve numerically with MCI and MLMC. Due to the scheme being unconditional stable therefore, we choose $\triangle x = \triangle t$. For MCI $\triangle t = \mathcal{O}(\epsilon)$ and $M = \mathcal{O}(\epsilon^{-2})$ and for MLMC the finest level $\triangle t_L = \mathcal{O}(\epsilon)$. Figure 5.21 shows the distribution for the solution and shows the exact and approximate mean are almost the same when using (a) MCI and (b) MLMC. The figure shows how the mean changes within the time as shown in (b) and (c).



(a)



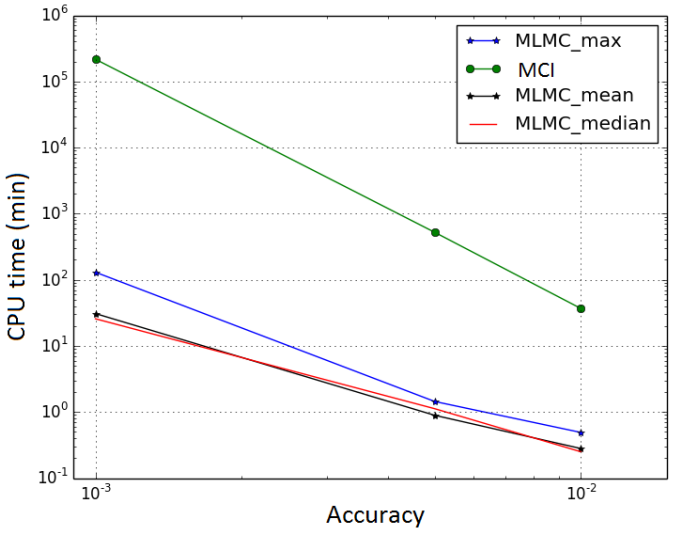
(b)



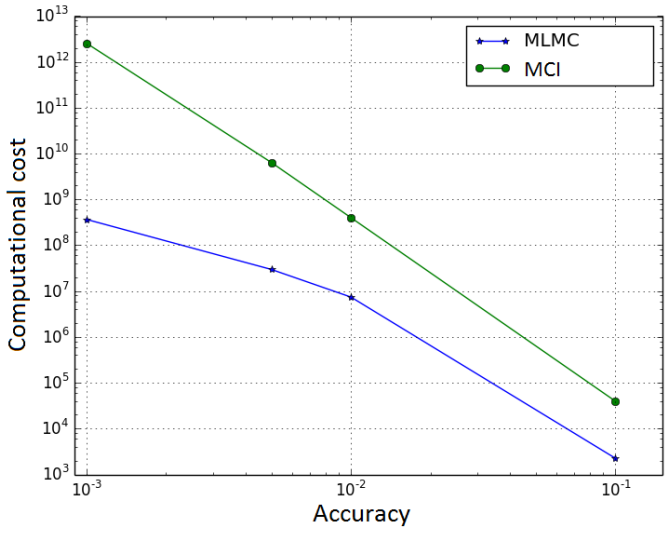
(c)

Figure 5.21: Estimate of the distribution of the linear pressure solution with (a) MLMC at $t = 1$ (b) MCI at $t = 1$ (c) MCI at $t = 0.5$ with $\epsilon = 0.005$.

Figure 5.22 (a) shows that, MLMC with different constraints (5.13) 'max, mean, median' for the variance of the estimator is faster than MCI for estimating the mean of the solution for the linear pressure equation. (b) shows that MCI has a high computational cost compared with MLMC. For example, if we need to find a solution with an accuracy of 0.001, we need about 69 days to have the solution, but for MLMC we need just a few hours.



(a)



(b)

Figure 5.22: For linear pressure equation (a): CPU time for MCI and different constraints for MLMC (b): Computational cost for MCI and MLMC with 'max' constraint.

The semi-linear pressure equation can be written

$$\frac{\partial P}{\partial t} = \frac{\partial}{\partial x}(K(x)\frac{\partial P}{\partial x}). \tag{5.21}$$

The finite difference scheme for (5.21) is

$$\begin{aligned} P_i^{n+1} - P_i^n &= \frac{\triangle t}{\triangle x}((K\,\frac{\partial P}{\partial x})_{i+1/2} - (K\,\frac{\partial P}{\partial x})_{i-1/2}) \\ &= \frac{\triangle t}{\triangle^2 x}(\bar{K}_{i+1/2}(P_{i+1}^{n+1} - P_i^{n+1}) - \bar{K}_{i-1/2}(P_i^{n+1} - P_{i-1}^{n+1})). \end{aligned}$$

We use the harmonic average for the permeability $K(x)$, i.e.

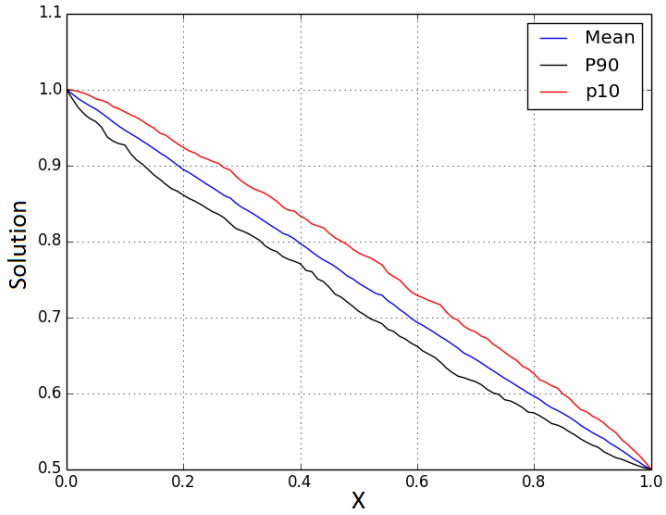
$$\bar{K}_{i+1/2} = \frac{2\,K_i\,K_{i+1}}{K_i + K_{i+1}}.$$

Substituting with the harmonic average for permeability into the finite difference, we get the following scheme.

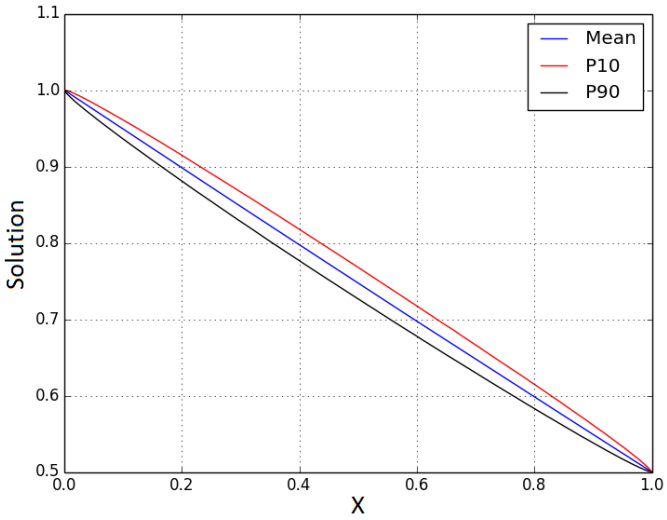
$$\begin{aligned} &P_{i+1}^{n+1}(\frac{\triangle t}{\triangle^2 x}\frac{2\,K_i\,K_{i+1}}{K_i + K_{i+1}}) - P_i^{n+1}(1 + \frac{\triangle t}{\triangle^2 x}\frac{2\,K_i\,K_{i+1}}{K_i + K_{i+1}} + \frac{\triangle t}{\triangle^2 x} + \frac{2\,K_i\,K_{i-1}}{K_i + K_{i-1}}) \\ &+ P_{i-1}^{n+1}(\frac{\triangle t}{\triangle^2 x}\frac{2\,K_i\,K_{i-1}}{K_i + K_{i-1}}) = -P_i^n. \end{aligned}$$

Now, we estimate the mean of the solution of the semi-linear pressure equation with random permeability using MCI and MLMC. We assume the distribution for the logarithm of permeability is a normal distribution with mean and standard deviation equal to zero and one respectively.

Figure 5.23 shows the distribution of the solution of the semi-linear pressure equation, assuming the accuracy is equal to 0.005, (a) MLMC and (b) MCI. The figure shows MLMC is able to solve problem does not have an exact solution.



(a)



(b)

Figure 5.23: Estimate of the distribution of the semi-linear pressure solution with (a) MLMC (b) MCI with $\epsilon = 0.005$.

In conclusion, MLMC estimate the distribution of the solution of the linear and semi-linear pressure equations cheaper than MCI and are almost as efficient as MCI. MLMC is applicable for solving the parabolic equation and we can address problems that do not have an exact solution.

5.5 Summary

The goal of this chapter is applying MLMC to a sequence of problems with increasing the difficulty and relevance to the equations govern the flow motion. The chapter compares the performance of MLMC and MCI for solving stochastic ODE: the

exponential growth and decay equation and stochastic PDEs: Advection, Buckley-Leverett and Pressure equations using different scenarios.

- For the exponential growth and decay equation, MLMC obtains a performance speed up over MCI in the range of $10^2 - 10^4$ as shown in Figure 5.8.
- For the advection equation, MLMC is faster than MCI for estimating the mean of the solution as shown in Figure 5.15. Also, MLMC is cheaper than MCI using Lax-Wendroff scheme. MLMC is more efficient algorithm than MCI as shown in Table 5.2.
- The choice of the control condition for the number of samples required to minimize the computational cost does have an effect on the CPU time, although they have the same absolute error for solving the advection equation as shown in Figure 5.17.
- For the Buckley-Leverett equation, MLMC is cheaper than MCI using three different control conditions as shown in Figure 5.19b.
- For the Buckley-Leverett equation, MLMC is successful in estimating the uncertainty range of the solution as in Figure 5.19a and almost as efficient as MCI, see Figure 5.20.
- For the Pressure equation, MLMC estimates the distribution of the solution in case of linear and semi-linear pressure problem as in Figures 5.21 and 5.23. Also, MLMC is cheaper than MCI for solving the pressure equation as observed from Figure 5.22.

In conclusion, MLMC is applicable for solving hyperbolic and parabolic differential equations, linear, semi-linear and nonlinear equations. MLMC is as efficient or more efficient than MCI. MLMC is faster than MCI.

In the next chapter, we will discuss how to use MLMC for quantifying the uncertainty in a reservoir simulation.

Chapter 6

Multilevel Markov Chain Monte Carlo Applied to Uncertainty Quantification

In this chapter, we demonstrate an application of a new technique, Multilevel Markov Chain Monte Carlo (MLMCMC), for quantifying uncertainty in reservoir simulations. The purpose of this technique is to gain speed by decomposing the desired results into a component calculated with a coarse model, with corrections obtained on a sequence of finer models (Dodwell et al., 2015).

We start by reviewing Random Walk Metropolis (RWM). Then, introduce the use of the MLMCMC method in reservoir simulation. The chapter uses MLMCMC to solve multi-phase flow using reservoir simulations and shows results for two fields. The first is Teal South in the Gulf of Mexico and the second is Scapa in the UK North Sea. The chapter reviews how to use the sensitivity analysis for reservoir model.

One of the main interests in the field of oil reservoir engineering is the accurate prediction of subsurface flow. An example output is the Field Oil Production Rate (FOPR), which is a discrete time series. This time series is statistical because the forecasting variables can be described using probability distributions. Accurate predictions are obtained by quantifying the uncertainty.

Uncertainty quantification in subsurface flows is an important task in reservoir

simulation studies, which may be used to inform reservoir management decisions. Based on a Bayesian framework, we sample from a posterior distribution (updating the model based on observed data) and produce Bayesian credible intervals for quantities of interest.

The major difficulty in applying MCMC methods is the high computational cost (Robert and Casella, 2010). In many situations, using MCMC techniques requires 10^5 realizations to run to obtain reasonable error (Oliver et al., 2008). For reservoir simulation, each realization or function evaluation can take 15 minutes to 6 hours. 10^5 samples would be impractical even on a large cluster.

6.1 Random Walk Metropolis (RWM)

At its simplest, MCMC consists of a proposal mechanism to suggest a new set of unknown parameters for the reservoir model, along with an accept-reject mechanism to ensure that the stationary distribution of the Markov chain has the desired distribution. Forecasts are then generated by running the reservoir model for each of the samples in the Markov Chain (or a thinned subset of the samples) and computing mean behaviour or appropriate p-quantiles.

A simple and often-used MCMC algorithm is the Random Walk Metropolis (RWM) algorithm (Robert and Casella, 2010), where the new proposed location (i.e. parameter value in parameter space) is given by a random walk from the current location. The size of the step length is a parameter of the algorithm, and various theoretical studies show how to choose the step length for optimal efficiency in the MCMC algorithm (MacKay, 2002; Neal et al., 2012).

In the case of a strictly positive density function, the random walk Metropolis kernel is irreducible and aperiodic (Tierney, 1994). The most efficient rate of RWM is $30\% - 55\%$ (Gilks et al., 1996).

RWM is easy to implement, although it can often take a long time to explore all the space and it may converge slowly. This means large numbers of forward simulations may be required to obtain sensible, accurate solution (Liu and Oliver, 2003).

The partial differential equations which govern the flow in the reservoir are solved for each sample. The discretization of the model and the structure of the model and how many unknown parameters often result in a prohibitive computational cost. Thus, an important research area is to develop and construct a new method that has the same accuracy as a long MCMC chain, but is less expensive.

MCMC was used to study history matching of noisy transient pressure in the reservoir simulation by (Oliver et al., 1997). They concluded that the uncorrelated MCMC samples from the posterior distribution were inadequate to quantify the uncertainty and make decisions about the reservoir. Later, Oliver used Hybrid Monte Carlo (described in Chapter 8) to study the history matching problem resulting in a higher acceptance rate, but unfortunately, it is computationally expensive (Bonet-Cunha et al., 1996).

Algorithm 5: Random Walk Metropolis (Robert and Casella, 2010).

Data: Initialize the set of parameters $\boldsymbol{m}_0$ and $\boldsymbol{\sigma}$.
Result: Vector contains all the acceptance states $m = (\boldsymbol{m}_0, \boldsymbol{m}_1, \cdots)$.
for $i = 1$ **to** $N_{samples}$ **do**
  Draw $\triangle_{i-1} \sim \mathcal{N}(0, \boldsymbol{\sigma^2})$;
  $\boldsymbol{m'} = \boldsymbol{m}_{i-1} + \triangle_{i-1}$ %proposal
  Draw $\alpha \sim \mathcal{U}(0, 1)$
  if $\alpha < \min\{1, \frac{\pi(\boldsymbol{m'}|\mathcal{D})}{\pi(\boldsymbol{m}_{i-1}|\mathcal{D})}\}$ **then**
    $\boldsymbol{m}_i = \boldsymbol{m'}$
  else
    $\boldsymbol{m}_i = \boldsymbol{m}_{i-1}$;
  end
end

In the algorithm 5, π is the posterior distribution and $\boldsymbol{\sigma}$ is the standard deviation vector which determines the step size for the random walk. The RWM approach consists of generating a new sample depending on the previous one according to the proposed distribution and comparing the likelihood for the new sample to the likelihood for the previous sample. The misfit function can be defined as discussed in Section 2.6

$$\text{Misfit} = \sum_{j=1}^{N} (q_{t_j}^{\text{sim}} - q_{t_j}^{\text{obs}})^2 / 2\sigma_{t_j}^2$$

Based on assuming the data measurement error is independent and Gaussian dis-

tributed at any time t, then the likelihood can be defined as discussed in Section 2.6.0.2

$$\mathcal{L}(\mathcal{D}|m) = \frac{1}{(2\pi)^{N/2}\prod_{i=1}^{N}\sigma_{t_i}} e^{-\text{Misfit}}$$

The acceptance or the rejection of the new sample can be done using this comparison. If the new sample is accepted, it is used for inference and if it is rejected, we use the previous sample. The major problem is calculating the likelihood function is the cost involved in running the simulation and then potentially rejecting the result and wasting the CPU time. In large-scale applications, using MCMC, the cost of the likelihood calculation is expensive as we are solving non-linear PDEs in high dimensions on a fine spatial grid.

There are alternative approaches such as the Langevin method (Dostert et al., 2006) or Hamiltonian Monte Carlo which we will discuss in Chapter 8, but these are more complex to programme and tune for optimal performance. One way to reduce the computational cost is to upscale the ordinary reservoir model. However, the solution obtained using a coarse grid can produce a poor estimation of the parameters. Moreover, poor estimations feed through forecasting (O'Sullivan, 2004). Quantifying the numerical errors due to upscaling was studied by (Glimm et al., 2001a). The computational cost can also be reduced using a variance reduction technique such as Multilevel Monte Carlo (MLMC), which was discussed in Chapter 5, but in this chapter we combine MLMC with RWM. For more details about MCMC methods, see Chapter 3.

6.1.0.1 Effect of Step Size on RWM

To choose the step size we should be aware of the shape of the target distribution. For example, if the target distribution is bivariate distribution, the step size has to small enough to explore the dimension with the smaller standard deviation.

The step size should not be too small because it leads to a high acceptance rate, but the samples are all highly correlated. This is leads to multiple iterations to explore small regions. If a large step size is used, the acceptance rate will be low,

especially in the tail distribution (Bhuripanyo, 2014).

6.2 Multilevel Markov Chain Monte Carlo (MLMCMC)

Multilevel Markov Chain Monte Carlo (MLMCMC) was first proposed by (Ketelsen et al., 2013) to quantify uncertainty much more rapidly than RWM. (Dodwell et al., 2015) improved the MLMCMC method, which discussed in (Ketelsen et al., 2013) by avoiding the bias error in the estimator. (Efendiev et al., 2014) used a generalized multiscale finite element method with MLMCMC for quantifying the uncertainty of quantities of interest for high contrast single-phase flow problems.

The contribution of this thesis is to use MLMCMC for solving multi-phase flows based on hyperbolic (saturation equation) and parabolic (pressure equation), which does not appear in literature.

The MLMCMC sampling strategy is based on comparing the quantities of interest at one level (e.g., at a finer level) to that at another level (e.g., at a coarser level). Just like the MLMC method, discussed in Chapter 5, the key is to avoid estimating quantities of interest directly on the fine grid, but instead to estimate the correction with respect to the next lower level. In this manner, we obtain samples from hierarchical posteriors corresponding to multilevel approximations which can be used for rapid computations within a MCMC framework.

6.2.1 Two-level MCMC

We demonstrate the simplest case first, which is the case of two-level MCMC, Equation (5.8) can be written as

$$f_{\text{fine}} = f_{\text{coarse}} + (f_{\text{fine}} - f_{\text{coarse}}), \tag{6.1}$$

where, f is the quantity of interest (e.g. FOPR). Two level MCMC replaces estimation of the quantity of interest on the fine grid, which is extremely expensive, by

estimation of this quantity on the coarse grid, f_{coarse}, plus estimation of a correction term, $(f_{\text{fine}} - f_{\text{coarse}})$. Independent estimators are used to estimate each term. If estimation of the first term, f_{coarse}, requires M_1 samples, and the correction term, $(f_{\text{fine}} - f_{\text{coarse}})$, requires M_2 samples, the formulae for M_1 and M_2 can be written as follows (Giles, 2015; Lord et al., 2014),

$$M_1 = \frac{2}{\epsilon^2 V_0}\sqrt{\frac{v_{\text{coarse}}}{N_{\text{coarse}}}}\Big(\sqrt{v_{\text{coarse}}N_{\text{coarse}}} + \sqrt{v_{\text{CF}}(N_{\text{fine}} + N_{\text{coarse}})}\Big) \tag{6.2a}$$

$$M_2 = \frac{2}{\epsilon^2 V_0}\sqrt{\frac{v_{\text{CF}}}{N_{\text{coarse}} + N_{\text{fine}}}}\Big(\sqrt{v_{\text{coarse}}N_{\text{coarse}}} + \sqrt{v_{\text{CF}}(N_{\text{fine}} + N_{\text{coarse}})}\Big), \tag{6.2b}$$

where N_{coarse} is the number of coarse grid blocks, N_{fine} is the number of fine grid blocks, v_{coarse} is the variance of f_{coarse}, v_{CF} is the variance of $(f_{\text{fine}} - f_{\text{coarse}})$, the variance of the error in the observed data V_0, which is scaling the equations and ϵ is the percentage error compared with the initial variance V_0 (the variance of the estimator is of order $\epsilon^2 V_0$). The variability of the estimator of $(f_{\text{fine}} - f_{\text{coarse}})$ is less than the variability of the estimator of f_{coarse} (Lord et al., 2014). Therefore, by using (6.2), we can obtain

$$\frac{M_2}{M_1} = \sqrt{\frac{N_{\text{coarse}}}{N_{\text{coarse}} + N_{\text{fine}}}}\sqrt{\frac{v_{\text{CF}}}{v_{\text{coarse}}}} < 1.$$

Algorithm 6: Two-level MCMC (Ketelsen et al., 2013)

Input : Fix coarse and fine grids, fix a number of warm-up samples M_{up}, fix the standard deviation $\boldsymbol{\sigma}$, fix the variance of the error in the observed data V_0 and fix the accuracy ϵ.

if $l = 1$ **then**
- Use RWM Algorithm 5, with $M_{samples} = M_{up}$
- Check condition (6.3). If it is satisfied, go to the second level. Otherwise, add more samples.

else
- Initialized $\boldsymbol{m}_0^{\text{coarse}}$ and $\boldsymbol{m}_0^{\text{fine}} = \boldsymbol{m}_0^{\text{coarse}}$
- **while** $(M_{up} > 0)$ **do**
  - **for** $i = 1$ **to** M_{up} **do**
    - Draw $\triangle_{i-1} \sim \mathcal{N}(0, \boldsymbol{\sigma}^2)$
    - $\boldsymbol{m}' = \boldsymbol{m}_{i-1}^{\text{coarse}} + \triangle_{i-1}$
    - Draw $\alpha \sim \mathcal{U}(0,1)$
    - **if** $\alpha < \min\{1, \frac{\pi^{coarse}(\boldsymbol{m}'|\mathcal{D})}{\pi^{coarse}(\boldsymbol{m}_{i-1}^{coarse}|\mathcal{D})}\}$ **then**
      - $\boldsymbol{m}_i^{\text{coarse}} = \boldsymbol{m}'$
      - Draw $\alpha_1 \sim \mathcal{U}(0,1)$
      - **if** $\alpha_1 < \min\{1, \frac{\pi^{fine}(\boldsymbol{m}'|\mathcal{D})}{\pi^{fine}(\boldsymbol{m}_{i-1}^{fine}|\mathcal{D})} \frac{\pi^{coarse}(\boldsymbol{m}_{i-1}^{coarse}|\mathcal{D})}{\pi^{coarse}(\boldsymbol{m}'|\mathcal{D})}\}$ **then**
        - $\boldsymbol{m}_i^{\text{fine}} = \boldsymbol{m}'$
      - **else**
        - $\boldsymbol{m}_i^{\text{fine}} = \boldsymbol{m}_{i-1}^{\text{fine}}$
      - **end**
    - **else**
      - $\boldsymbol{m}_i^{\text{coarse}} = \boldsymbol{m}_{i-1}^{\text{coarse}}$
      - $\boldsymbol{m}_i^{\text{fine}} = \boldsymbol{m}_{i-1}^{\text{fine}}$,
    - **end**
  - **end**
  - Check condition (6.3) and update the required number of samples, M_l
  - $M_{up} = M_l - M_{up}$
- **end**

end

The algorithm easily generalises to l levels. For more details about MLMCMC method, see (Efendiev et al., 2014; Dodwell et al., 2015). When the output is a time series, such as the FOPR, then we have to enforce condition (6.3) to control the number of samples required.

$$\text{control}(\mathbb{V}(\mathbb{E}(f_L))) \leq \epsilon^2 V_0/2 \tag{6.3}$$

where f_L is the FOPR with finest grid. We can change the control condition (6.3)

to the mean, median, the first quantile, $Q1$ or the third quantile, $Q3$. The choice depends on the case study.

The MLMCMC algorithm outputs are: all the input parameters and corresponding FOPR for every level and the number of samples at each level. Then, we find the effective number of samples based on the autocorrelation for each level. Because the samples at each level are independent, therefore, by finding all the possible sums from all the levels to obtain the distribution of the quantity of interest (adopted from the sum of independent variables).

The following section shows how to analyse the output results. We will use it for this chapter and in Chapters 7 and 8.

6.3 Output Analysis

Having obtained the MCMC output, we have to ask the following questions. How long is the burn-in period? Are the samples independent? Should we thin the chain? What is the effective sample size? Is the chain convergent? What is the speed of convergence?

To evaluate the sampling performance of MCMC chains, diagnostic tools are used to thin the chain, check the autocorrelation, check that the burn-in period is long enough, check the convergence and compute how many effective samples there are (Box and Jenkins, 1976).

6.3.1 Autocorrelation

The correlation between the values of a time series at different times can be described using **Autocorrelation**. Autocorrelation is a function of the two times t_1 and t_2 or a function of the time difference.

For the case where autocorrelation is a function of the two times t_1 and t_2, let θ be some repeatable process and i be some point in time. Then, θ_i is the value produced by a given run of the process at time i. Suppose we know the mean μ_i and the variance σ_i^2 for all times i. Then the definition of the autocorrelation between

times t_1 and t_2 is

$$\rho(t_1, t_2) = \frac{\mathbb{E}[(\theta_{t_1} - \mu_{t_1})(\theta_{t_2} - \mu_{t_2})]}{\sigma_{t_1}\sigma_{t_2}}. \tag{6.4}$$

If the variance is either zero (for a constant process) or infinite, then the previous expression is not well defined. If the function ρ is well defined, its value must lie in the range $[-1, 1]$ with 1 corresponding to perfect correlation and -1 indicating perfect anti-correlation (Walsh, 2004).

In the case of the autocorrelation is a function only of the time difference, the mean μ and the variance σ^2 are independent in time, and further, the autocorrelation depends only on the difference between t_1 and t_2. The correlation depends only on the time-distance $t_1 - t_2$. This implies that the autocorrelation can be expressed as a function of the time-lag, and that it would be an even function of the $lag\, \tau = t_2 - t_1$. This gives the more familiar form as follows (Jensen et al., 2000),

$$\rho(\tau) = \frac{\mathbb{E}[(\theta_t - \mu)(\theta_{t+\tau} - \mu)]}{\sigma^2}.$$

Autocorrelation describes the dependence between the samples at different times. To determine the sign of the correlation, we can plot X_t against X_{t-1}. If there is correlation, the slope of the line is the sign of the correlation (Box and Jenkins, 1976). The autocorrelation function is symmetric with respect to the sign of the lag. Table 6.1 shows the significance of the autocorrelation values.

Table 6.1: Significance of the autocorrelation values

Autocorrelation	Significance
$0 \rightarrow 0.19$	Very weak
$0.2 \rightarrow 0.39$	Weak
$0.4 \rightarrow 0.59$	Moderate
$0.6 \rightarrow 0.79$	Strong
$0.8 \rightarrow 1$	Very strong

There are tools for assessing the autocorrelation, for example, a time series plot, which plots time on the x-axis against MCMC output on the y-axis. A lagged scatter plot plots, for example, X_t against X_{t-1} which means plotting $X_2, \cdots, X_t$ against $X_1, \cdots, X_{t-1}$. The autocorrelation function plot plots the lag against the autocorrelation. Practically, for getting a good estimate of the autocorrelation function, we

should have at least $N = 50$ observations and the maximum lag should be smaller than $N/4$ (Box and Jenkins, 1976).

High autocorrelation shows that there is slow mixing and slow convergence for the chain. This means that we need a long run to obtain convergence to the stationary distribution, but at a high computational cost (William and Eaton, 2013; Cowles and Carlin., 1996).

Autocorrelation Length

$$L(\tau) = 1 + 2\sum_{\tau=1}^{\tau_{\max}} \rho(\tau)$$

where, the lag is τ, the autocorrelation is ρ and the maximum lag is $\tau_{\max}$. If $L = 1$, the sampler is ideal, otherwise, there is a dependence between the data.

The sampling efficiency of the chain in terms of the autocorrelation length is:

$$E = NL^{-1}.$$

where L is the autocorrelation length and N is the number of samples.

6.3.2 Effective Samples

For the effective number of samples, E, there are different techniques to evaluate it, e.g., autocorrelation length L.

$$E = \frac{N}{L(\tau)}.$$

In this thesis, we use the R-command to find the effective sample size. In standard software R, the effective sample size is

$$E = N\frac{1 - r_1}{1 + r_1},$$

where r_1 is the first order autocorrelation when the lag is 1.

6.3.3 Chain Thinning

Chain thinning can be used when the successive samples are highly correlated. If we wish to have an independent chain, we can do this by thinning the chain (Bishop, 2006). Thinning produces independent samples uses them for inference from the original chain, which has a strong autocorrelation (biased chain).

Thinning the chain means constructing a subset from the original chain. For example, if you want 1000 samples, run the chain for 10000 samples and thin by selecting every 10^{th} samples. Thinning of chains has been considered inefficient, when the aim is to have precise estimates of the samples (William and Eaton, 2013).

6.3.4 Burning-in Period

If the choice of the initial state or the proposal distribution is poor it may be necessary to run a long chain to obtain convergence. The burn-in period of the chain will slow down convergence (Roberts and Rosenthal., 2001). It is common to discarded a number of iterations at the beginning of the simulation and not take them into account for inference. The iterations which are cut are called the burn-in samples. The idea of discarding the burn-in samples is to be sure that the resulting distribution is independent of the initial state (Walsh, 2004).

The ability to accurately identify the burn-in period can decrease computational time by avoiding over estimation of the burn-in period. Underestimating burn-in time results in high autocorrelation, slowing down convergence. However, overestimating the burn-in throws away samples that we could have used to obtain a more accurate estimate of the distribution, thus also slowing down convergence (Sahlin, 2011).

6.3.5 Convergence Diagnostic

The Fundamental theorem of Markov Chains guarantees that an aperiodic, irreducible finite chain converges to a stationary distribution, but it does not tell us how fast convergence happens. One of the important questions is, does MCMC converge to the target distribution?

Firstly, can we predict how long a MCMC simulation will take to reach equilibrium. How many realizations are required to converge to the stationary distribution? In the case of Random Walk Metropolis (Section 6.1), we can obtain a lower bound for the time required for convergence. Predicting this time more precisely is a difficult problem. Secondly, can we detect convergence in a running simulation? This is also a difficult problem. There are a few practical tools available, but none of them are perfect, as described in (Cowles and Carlin., 1996).

How do we test the convergence of the chain? There are several tests to check where the chain converges. We will discuss two of them which have been used to obtain the resulting Gewek test and Raftery-Lewis test.

6.3.5.1 Gewek Test

This test compares the means of two non-overlapping time intervals, say the first 10% of the chain, and the last 50% of the chain. If the means of both of them are close, the two samples come from the same distribution. If the values are different, discard the first 10% and use the second 10% and keep repeating this until the values are close. In the case of discarding all the values in the first half, the chain is failing to converge. The output is a value Z_{score}, which describes the confidence that the two sets have the same mean (Geweke, 1992).

6.3.5.2 Raftery-Lewis Test

The test is designed to estimate the number of burn-in samples and whether the number of samples in the chain is long enough. It is a way of estimating a for $\mathbb{P}(x \leq a) = q$. Therefore, we have to choose q at an accuracy of ϵ. These convergence tests are built into R under the Coda package.

Finally, we have to ask, can we speed up the convergence time and the time between independent samples in a MCMC method? There are several ways of doing that one of them is the Hamiltonian Monte Carlo method described in Section 8.1.

6.4 Teal South

The Teal South reservoir is located in the Gulf of Mexico, approximately 144 km southwest of Morgan City, Louisiana. Data for this reservoir were made available by the Energy Resources Clearing House (ERCH) in Houston. Fluids are produced from a horizontal well at the crest of the reservoir (Pickup et al., 2008). Production started in November 1996 and monthly data for oil, water and gas rates are available. There are two measurements of reservoir pressure: initial pressure (3096 psi) and reservoir pressure after 540 days (2458 psi) (Christie et al., 2002). Figure 6.1 shows the structure map of Teal South with an $11 \times 11 \times 5$ simulation grid (Christie et al., 2002).

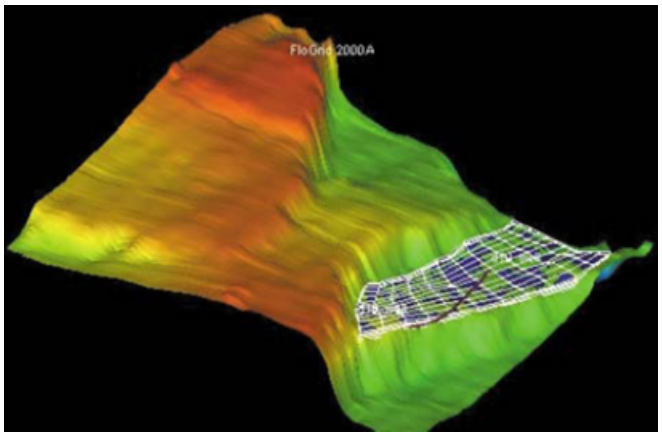
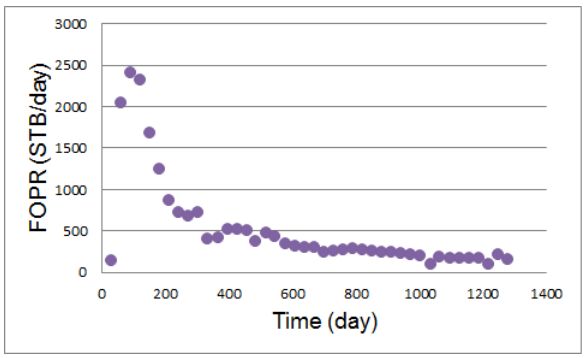


Figure 6.1: The structure map of Teal South.

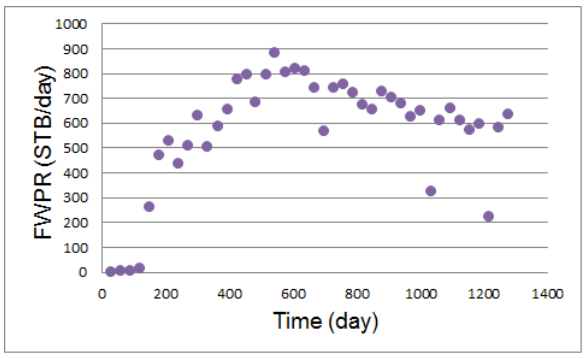
The observed data for oil, water and gas rates as a function of time, are shown in Figure 6.2 and we see that, after 80 days of production, the oil rate peaked and then rapidly declined as the water production started (Christie et al., 2002).

The grid for the reservoir simulation model in (Christie et al., 2011) is $11 \times 11 \times 5$, which we used as the coarse grid. Since MLMCMC requires both coarse and fine grids, for simplicity, we refined the model uniformly in the Z-direction by factors of 3 and 5 to give a grid resolution of $11 \times 11 \times 15$ and $11 \times 11 \times 25$. Then to obtain more accurate solution, we refine all directions (x, y, z) by a factor of 5 to give a grid resolution of $55 \times 55 \times 25$ for the fine grid.

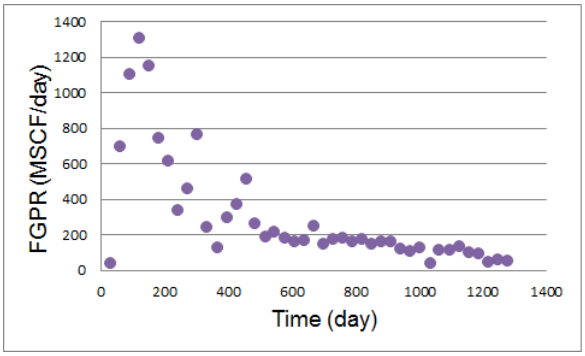
We used the first 181 days of production for history matching and kept back the remaining data to assess the quality of the forecasts. We measure how well a



(a)



(b)



(c)

Figure 6.2: Teal South observed data for (a) oil (b) water (c) gas rates.

specific set of reservoir model parameters fit the observed data using a least squares misfit (Mohamed et al., 2010a). The porosity is assumed to be fixed at 0.28 through the reservoir. The permeability field is only known at a few locations, but due to the lack of information, it is described by using randomness. We used a total of 8 unknown parameters for history matching: rock compressibility (Roc), aquifer strength (aqu), horizontal permeability for five layers (Kh), and vertical to horizontal permeability (Kvh). The prior distribution for each parameter was a uniform

distribution in the ranges shown in Table 6.2. The misfit was calculated using

$$\text{misfit} = \sum_{j=1}^{N} \frac{(q_{t_j}^{\text{Obs}} - q_{t_j}^{\text{sim}})^2}{2\sigma_{t_j}^2} \tag{6.5}$$

with a standard deviation, σ_{t_j} of 100 STB/day. The result will only focus on FOPR.

Table 6.2: Parameterisation and prior ranges for Teal South (Hajizadeh et al., 2011)

Parameters	Units	Minimum	Maximum
Roc	psi^{-1}	5×10^{-6}	10^{-4}
aqu	STB	10^7	10^9.
Kh	mD	10	1000
Kvh	-	0.0001	0.1

The choice for the prior distributions are different from those previously used in the literature because we switch from uniform in the logarithm of permeability to uniform in permeability.

6.4.1 Sensitivity Analysis

Modelling the real data require having uncertain parameters, however, increasing the number of uncertain parameters means increasing the complexity of the model. Moreover, there is a danger of over parameterization in the model.

The idea of sensitivity analysis is to scan the whole range of each uncertain parameter, to assess the effectiveness of the uncertain parameters on the model. Let N be a number of parameters then we run the flow simulation $2 \times N$ times. In the case of Teal South, we fix 7 parameters and tune the last one between $max - min$ in the prior range as in Table 6.2. Thus, to check the sensitivity of the model with respect to a parameter, we need to run one model with the maximum value for that parameter and one model with the minimum value. There are many ways to analyse the sensitivity of the model, one way is based on checking the differences between the quantity of interest at specific time steps (Schaaf et al., 2008), as in Figure 6.3. The figure shows Kvh and $Kh2$ have more influence on the model.

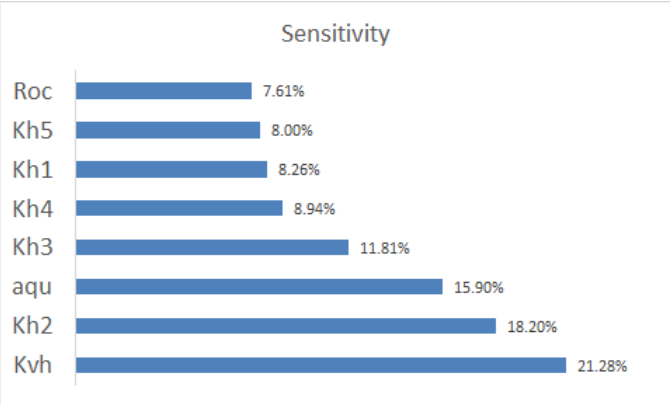


Figure 6.3: The sensitivity of Teal South Model at time step $t = 92$ days.

In the case of having a large number of unknown parameters, it would be good to use sensitivity analysis to select the most important parameters. Some of the parameters may be insensitive in the history matching period, but sensitive in the forecasting period (Floris et al., 2001).

6.4.2 Teal South Results

Since MCMC is a stochastic procedure, we must look at average behaviour to analyse the results. All the results in this section are the average of 3 runs.

Figure 6.4 shows the histogram for the unknown parameters using MLMCMC under the constraint (6.3) with the following set-up, $V_0 = 10000$ based on the variance of the error in the observed data, $\epsilon = 0.01$, grid $11 \times 11 \times 5$, $11 \times 11 \times 15$ and $11 \times 11 \times 25$, we used a burn-in period of 100 samples for each level and used the mean of the samples as the control condition.

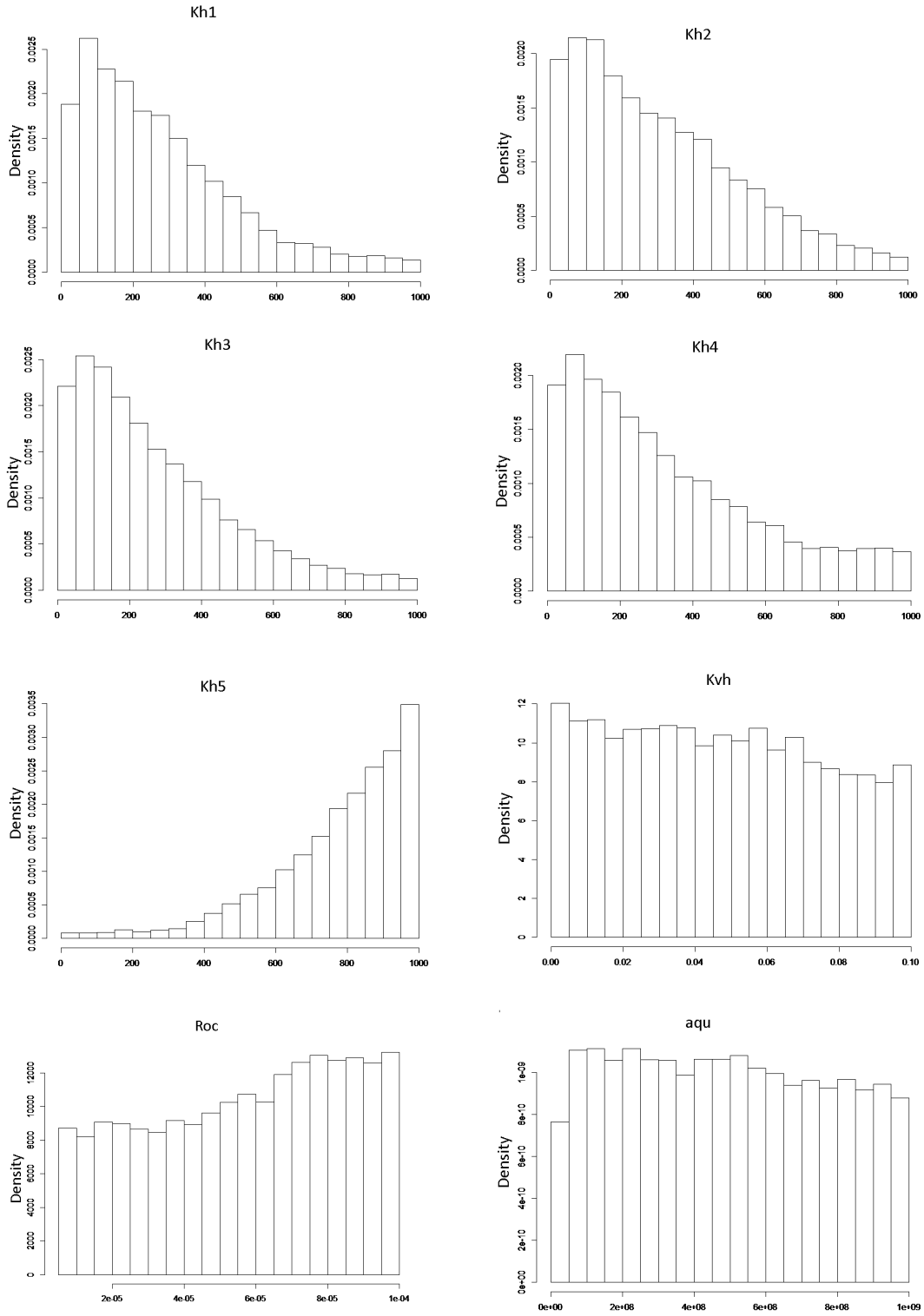


Figure 6.4: Histogram of the unknown parameters of Teal South model using MLMCMC.

Figure 6.5 shows the Bayesian credible intervals computed using both RWM and MLMCMC. The vertical line in Figure 6.5 represents the end of the history matching period. For RWM, we choose $\boldsymbol{\sigma}$ in the proposal equal to 20% of the parameter range. We obtained an acceptance rate of 50%, with the number of samples is 10000, and

a burn-in 500 samples. RWM used a grid is $11 \times 11 \times 25$ which corresponding to the finest level for MLMCMC. The figure shows that RWM requires to run longer because some of the observed data is outside the Bayesian credible interval.

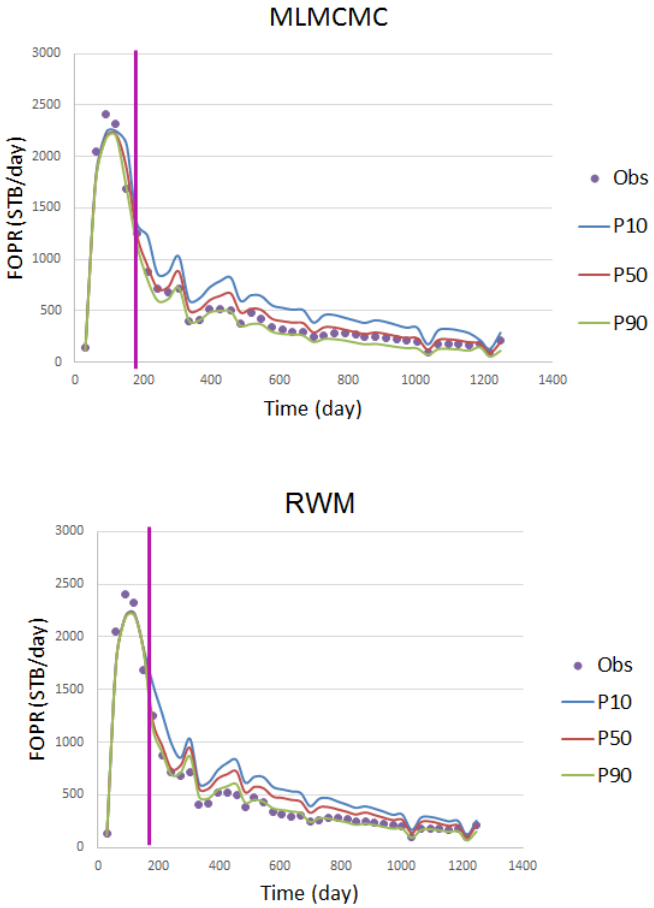


Figure 6.5: Bayesian credible intervals $P10 - P50 - P90$ of Teal South model (top) MLMCMC (bottom) RWM.

Figure 6.6 shows the autocorrelation (6.4) of MLMCMC per parameter. The figure shows the samples are highly correlated.

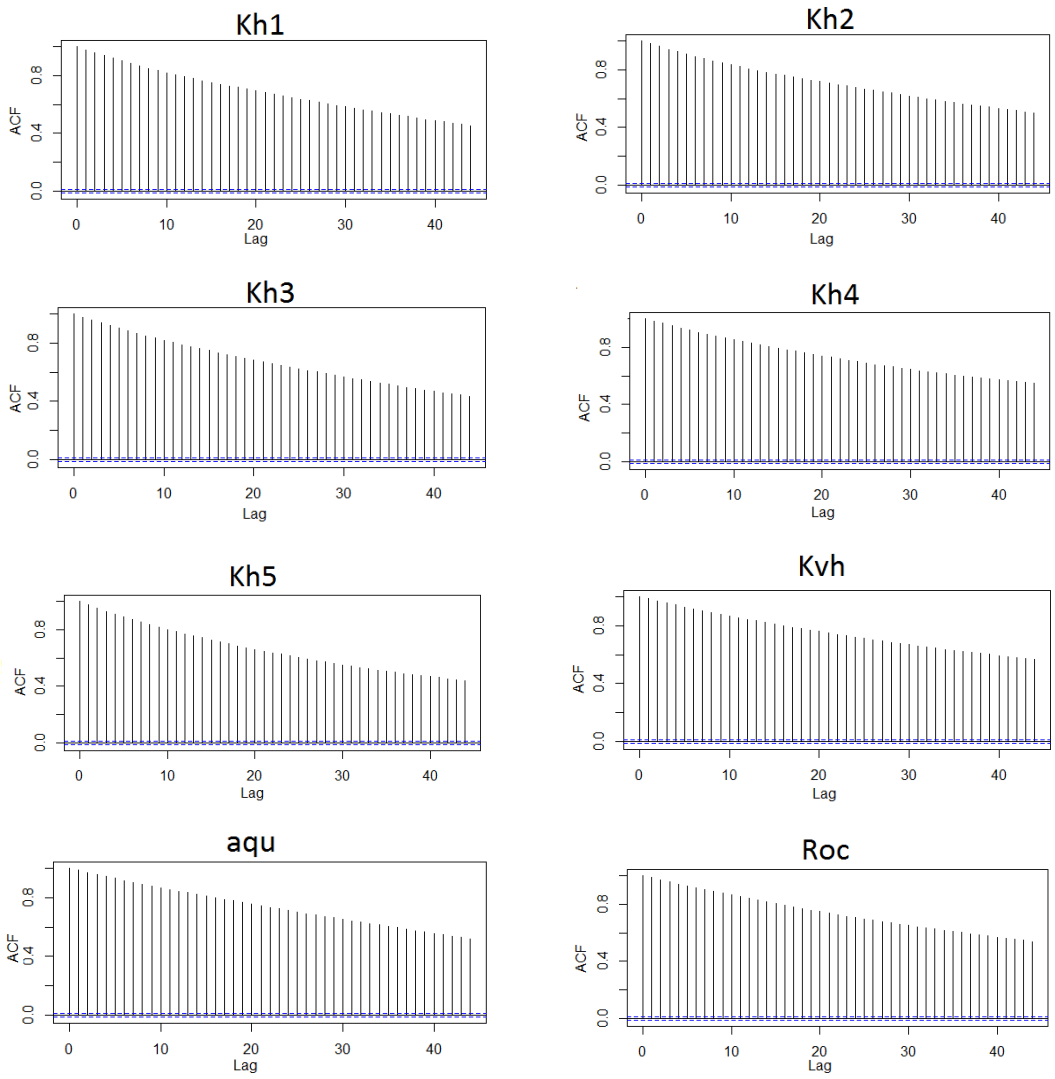


Figure 6.6: Autocorrelation of MLMCMC for each of the 8 unknown parameters of Teal South model.

Figure 6.7 shows the autocorrelation (6.4) of MLMCMC per parameter, based on the effective samples. The figure shows the samples are independent.

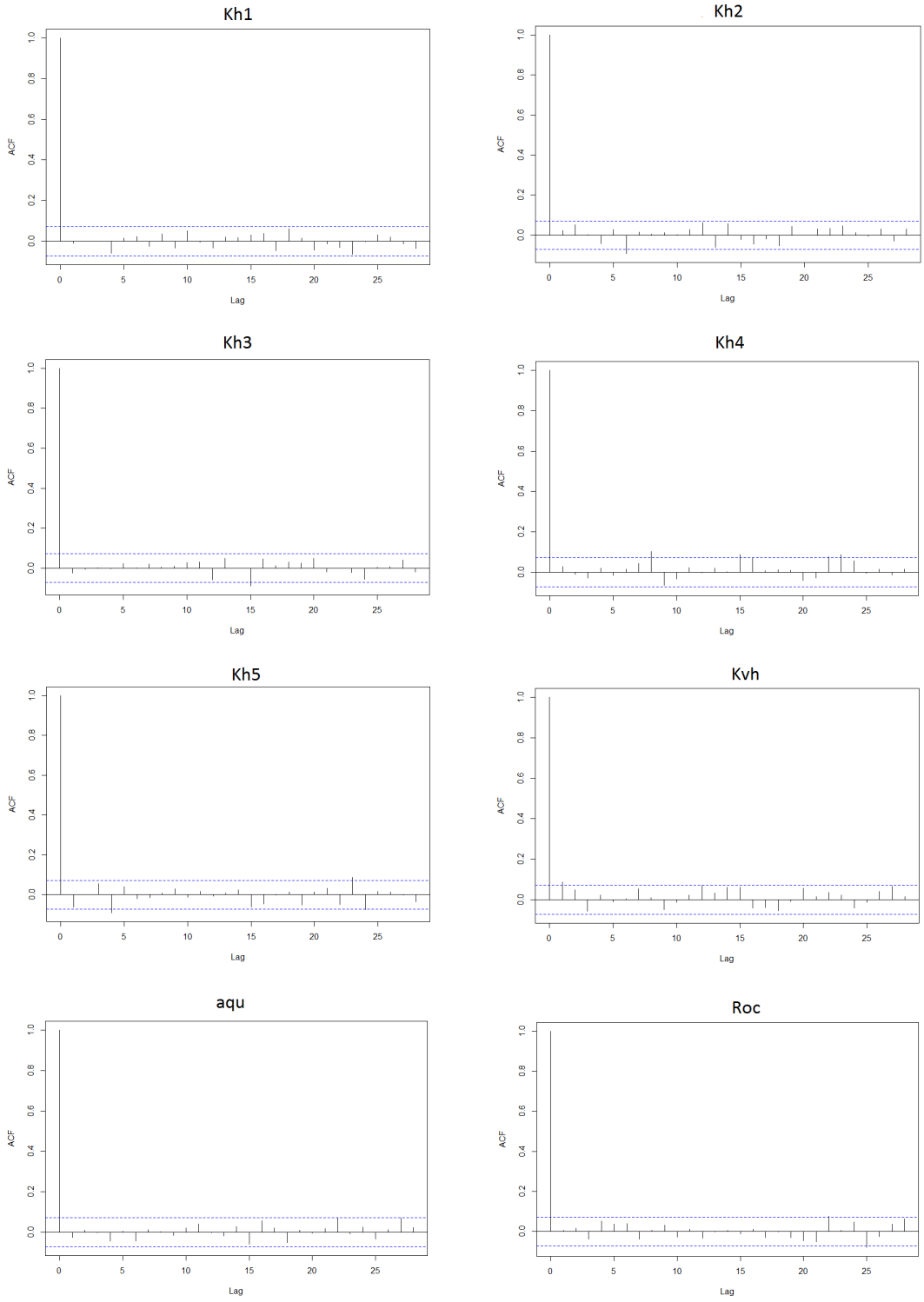


Figure 6.7: Autocorrelation of MLMCMC for each of the 8 unknown parameters for the effective samples of Teal South model.

Figure 6.8 compares the CDF of MLMCMC with RWM at the end of the history matching period, day 181 and at day 1187 in the forecast period. At day 181 the RWM distribution for the effective samples of RWM. It is not a smooth curve, but for long chain for RWM the distribution is a smooth curve. For day 1187 the

distribution for MLMCMC close to the distribution for long chain RWM. MLMCMC obtains almost the same CDF as MCMC on the finest grid within sampling error.

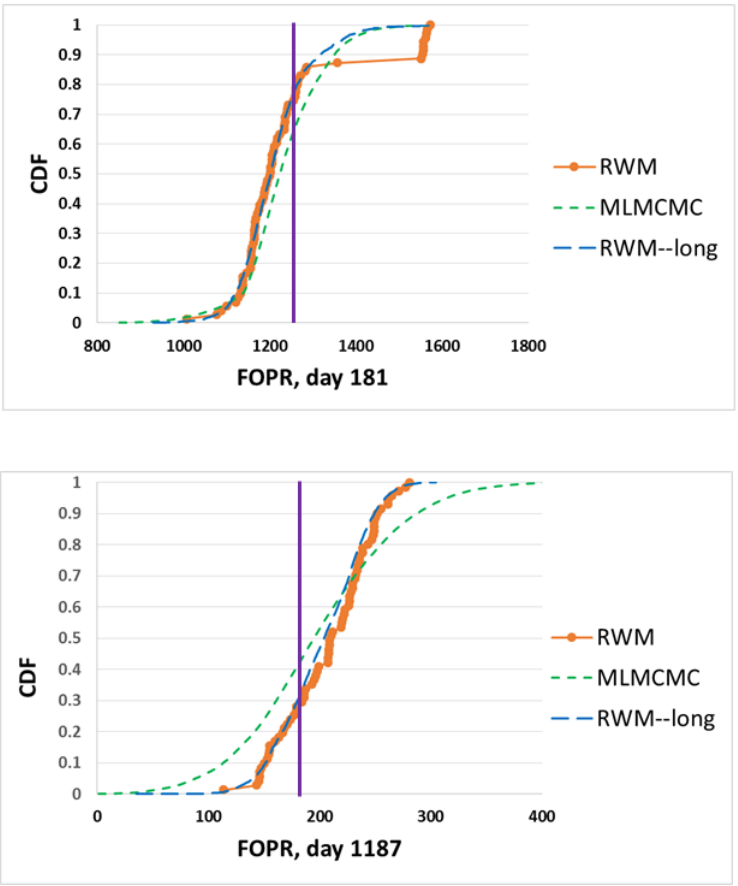


Figure 6.8: Cumulative distributions of Teal South model from MLMCMC, RWM and long chain of RWM at (top) day 181 (bottom) day 1187, the vertical line is the observed data.

Figure 6.9 shows that the CPU time is decreased significantly by using MLMCMC compared with RWM. The speed up obtained from MLMCMC is in the range of 10 to 100.

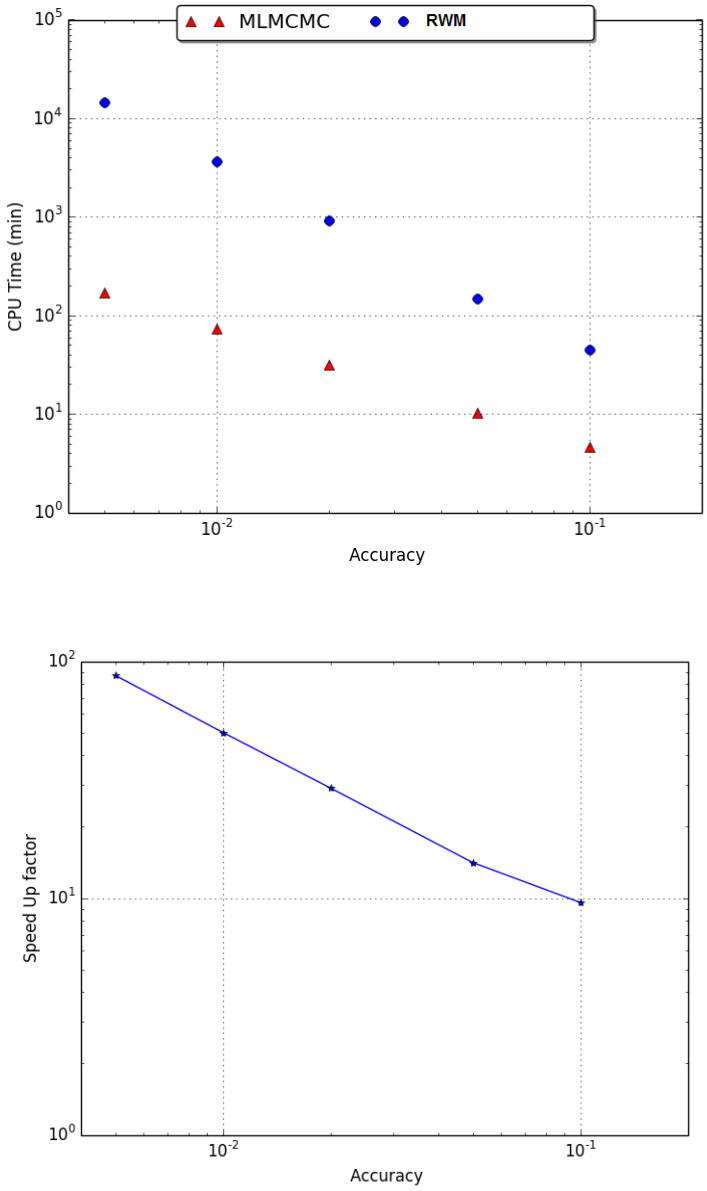
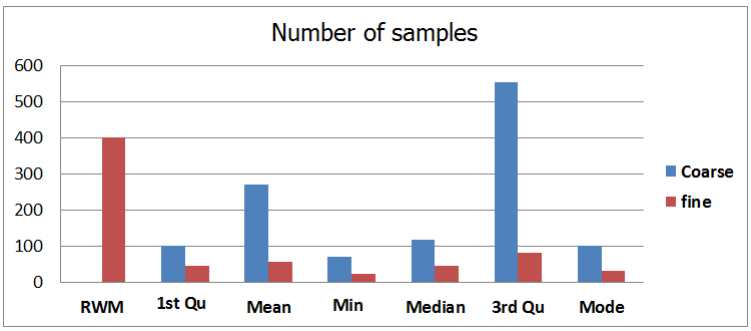


Figure 6.9: For Teal South model (top) Comparison of CPU time from RWM and MLMCMC (bottom) speed up factor for RWM and MLMCMC.

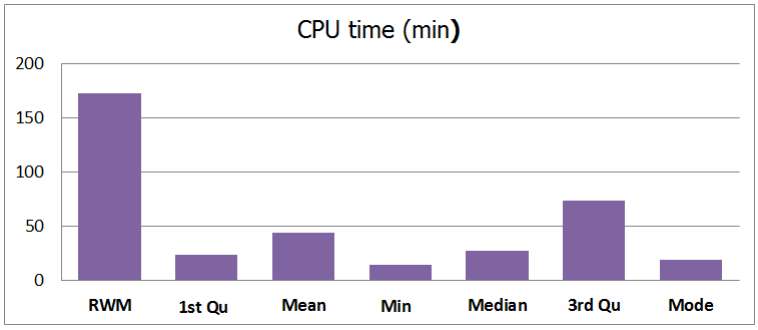
Figure 6.10 compares different control conditions (6.3) for the number of samples required for MLMCMC and RWM. For MLMCMC, we choose $\epsilon = 0.05$, grid $11 \times 11 \times 5$ and $11 \times 11 \times 25$, burning in period of 100 samples for each level. We compare different constraints for the number of samples required, control $\in \{Minimum, Mean, Q1, Median, Q3, Mode\}$.

Figure 6.10(a) compares the methods in terms of the number of samples required for coarse and fine grids, while Figure 6.10(b) compares in terms of the CPU time required. As shown in the figure, RWM is the slowest technique. Figure 6.10(c-d) show the uncertainty of the predictions made by RWM and the different constraints

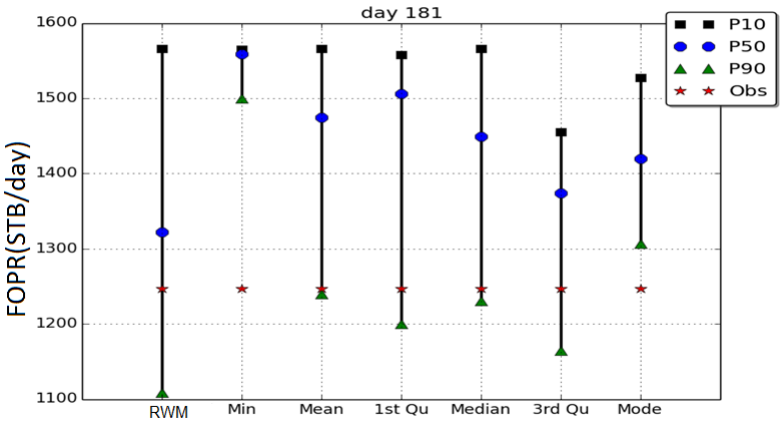
for MLMCMC. As shown in Figure 6.10(c-d), Mode and Minimum constraints are poor choices for Teal South case study because the observed data is outside the Bayesian credible interval. However, 1st Quantile, mean and Median constraints are good choices in terms of CPU time and the efficiency.



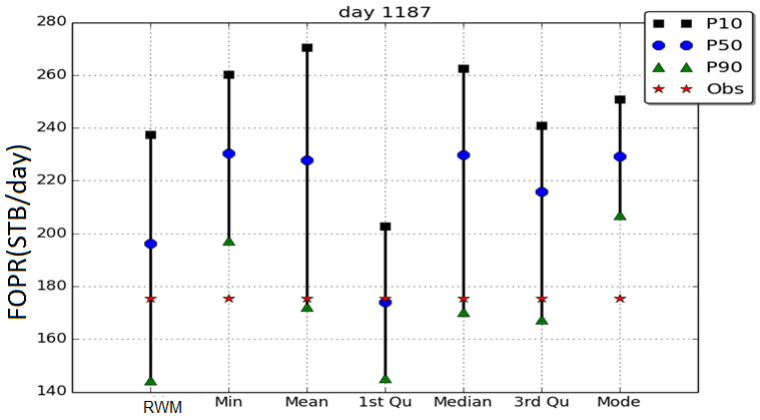
(a)



(b)



(c)

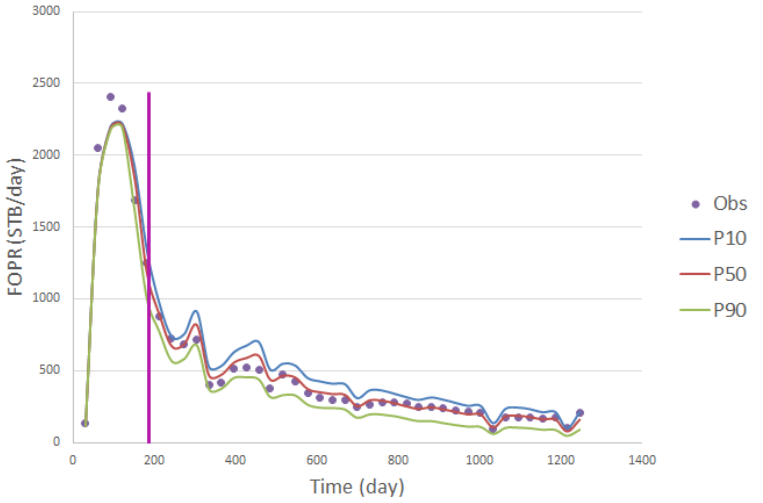


(d)

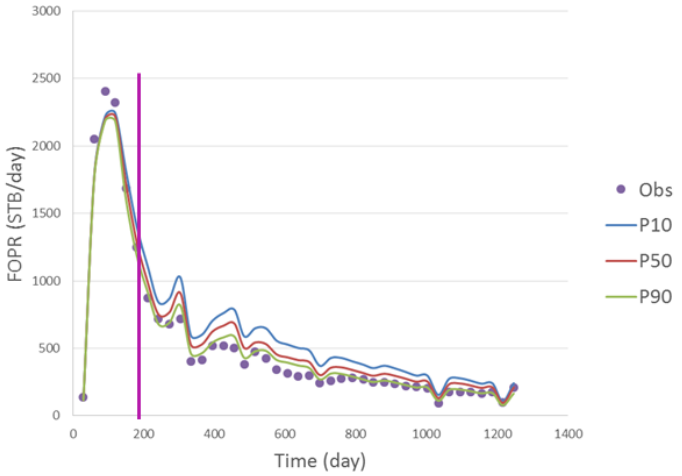
Figure 6.10: Comparison between RWM and different controls for the number of samples required for MLMCMC for Teal South model.

Figure 6.11(a-b) compare MLMCMC with the Particle Swarm Optimization (PSO) with Neighbourhood Algorithm Bayes (NAB) technique, described in Chapter 3. Firstly, we run MLMCMC based on the following set-up 3 times, $V_0 = 10000$, $\epsilon = 0.02$, grid $11 \times 11 \times 5$ and $55 \times 55 \times 25$, burning in period of 100 samples for each level and the constraint for the number of samples (6.3) is the first quantile.

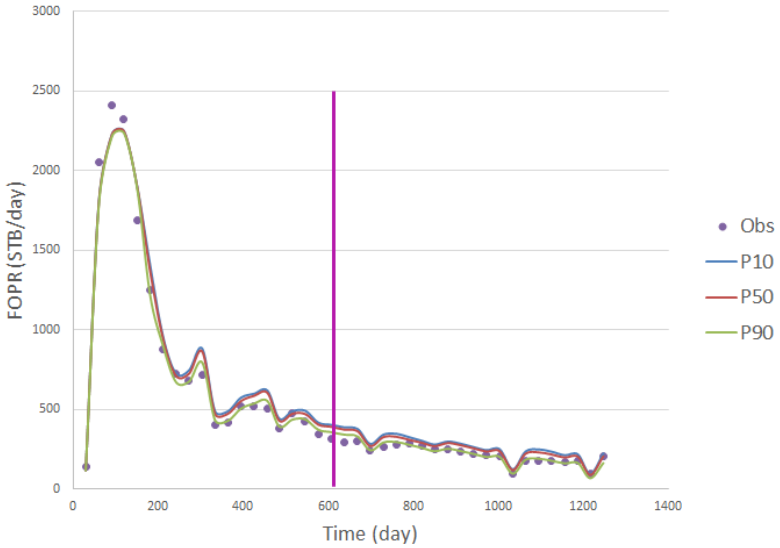
In Figure 6.11, we use the computational cost corresponding to each run of MLMCMC, to run PSO with NAB using Raven software (Epistemy Ltd). Figure 6.11(a-b) shows that, MLMCMC is more efficient than PSO with NAB may be because PSO needs more samples to obtain a better distribution. Figure 6.11(c) shows that if we increase the data taken as historical, the uncertainty range decreases and hence we have more confidence.



(a)



(b)



(c)

Figure 6.11: Bayesian credible intervals $P10 - P50 - P90$ (a) MLMCMC (b) PSO with NAB (c) MLMCMC with increased the historical data.

6.5 Scapa Field Description

The second field example is the Scapa field. Scapa is a field in the UK North Sea that came onstream in 1984. The field data for Scapa has been made available to Heriot-Watt University for use in teaching and research, and the model used here was originally developed in an MSc student project in 2011 (Farooq, 2011). The model contains $90 \times 53 \times 20$ grid blocks with a grid size of $300\,ft \times 300\,ft$ in the x and y directions. Figure 6.12 shows the initial reservoir state.

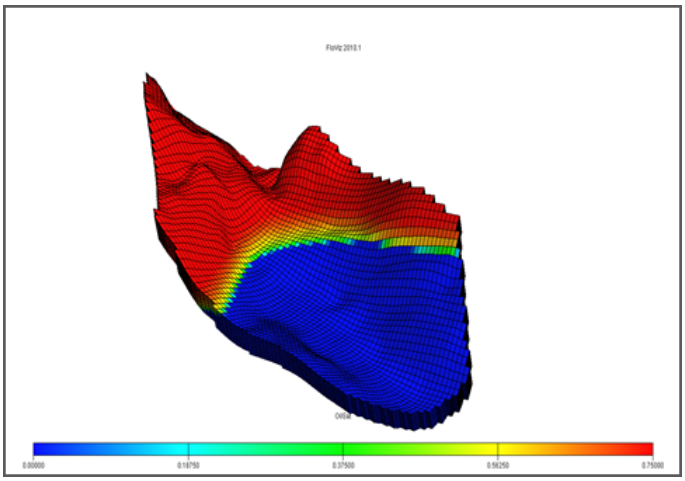
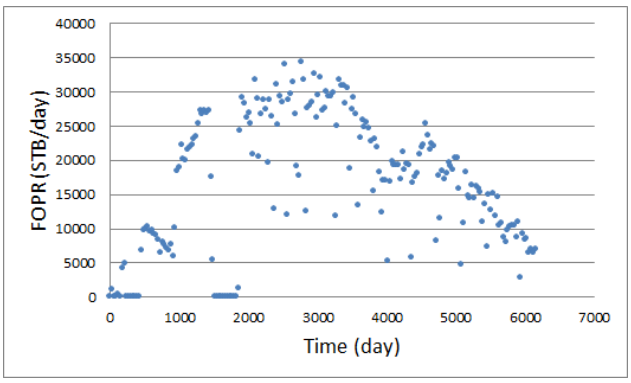


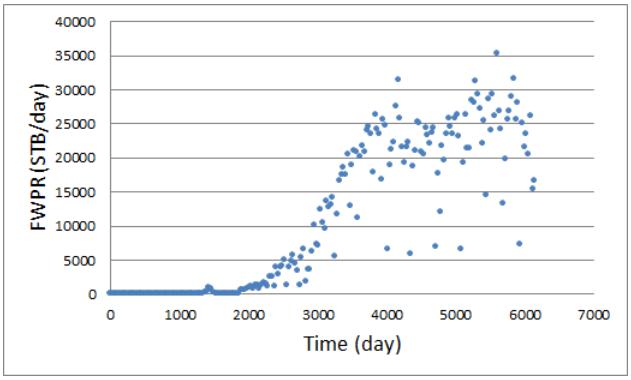
Figure 6.12: The structure map of Scapa.

Production data for Scapa are available from the UK Department Of Energy And Climate Change (DECC) website (DEC, 2015). The observed field oil and water rate as a function of time is shown in Figure 6.13. The first 3137 days of the production are used for history matching and the remaining data are used for forecasting. The model is controlled by the total liquid rate and the misfit is calculated for the oil rate using standard least squares (6.5), with a standard deviation of $1000 \times \sqrt{2}$ STB/day based on bias estimator in Chapter 2.

The number of unknown parameters for history matching is twenty three: porosity ϕ, transmissibility trans, horizontal permeability for twenty layers Kh, and vertical to horizontal permeability Kvh. The prior distribution for each parameter is a uniform distribution in the ranges shown in Table 6.3.



(a)



(b)

Figure 6.13: Scapa observed data for (a) oil (b) water.

Table 6.3: Parameterisation and prior ranges for Scapa (Farooq, 2011).

Parameters	Units	Minimum	Maximum
ϕ		0.16	0.21
trans		0	0.2
Kh	mD	10	3000
Kvh		0.001	0.3

The coarse model in the Multilevel MCMC algorithm was the original $90 \times 53 \times 20$ model, and the fine model was $90 \times 53 \times 60$, obtained by uniform refinement in the Z-direction. More details about the Scapa field can be found in (Chen, 1988; Ellison et al., 1992; McGann et al., 1991).

6.5.1 Scapa Results

Figure 6.14 shows MLMCMC can explore the whole range for 8 unknown parameters as example out of 23 parameters, which means improving the RWM exploration. The figure is based on the following set-up, $V_0 = 10000$, $\epsilon = 0.05$, $\sigma = 1000\sqrt{2}$, grid $90 \times 53 \times 20$ and $90 \times 53 \times 60$, burning in period of 100 samples for each level and

the constraint for the number of samples required is (6.3) using first quantile is the control.

Figure 6.14 shows that MLMCMC explores all the parameter space in 200 iterations, which means accelerating the random walk behaviour for RWM.

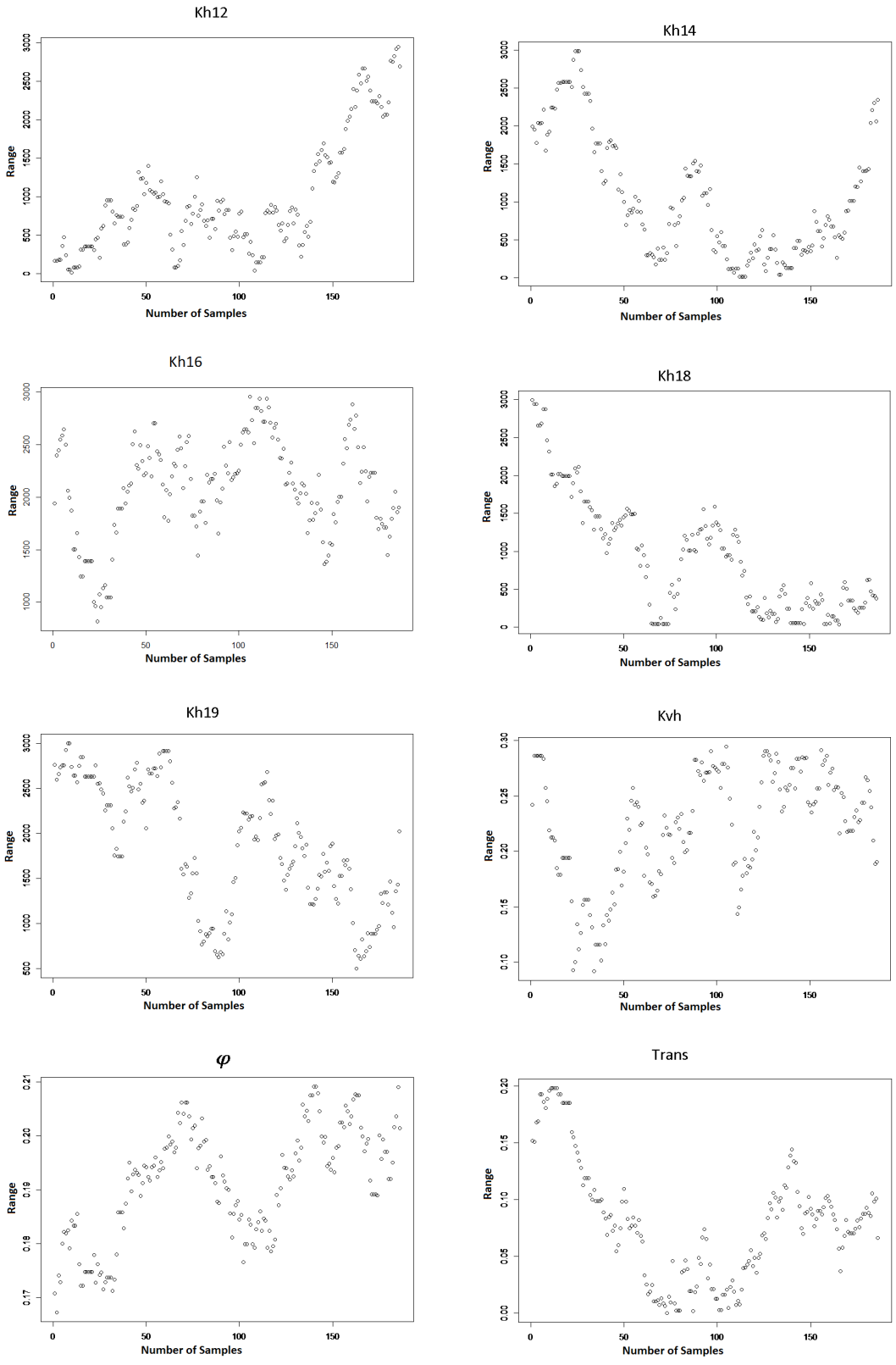
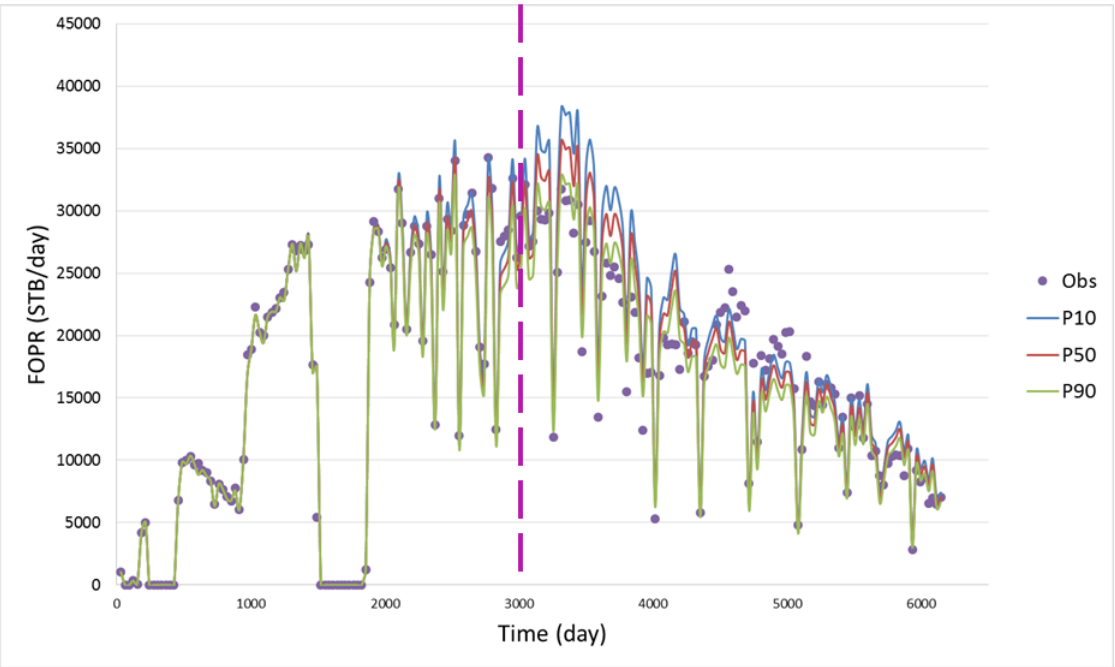


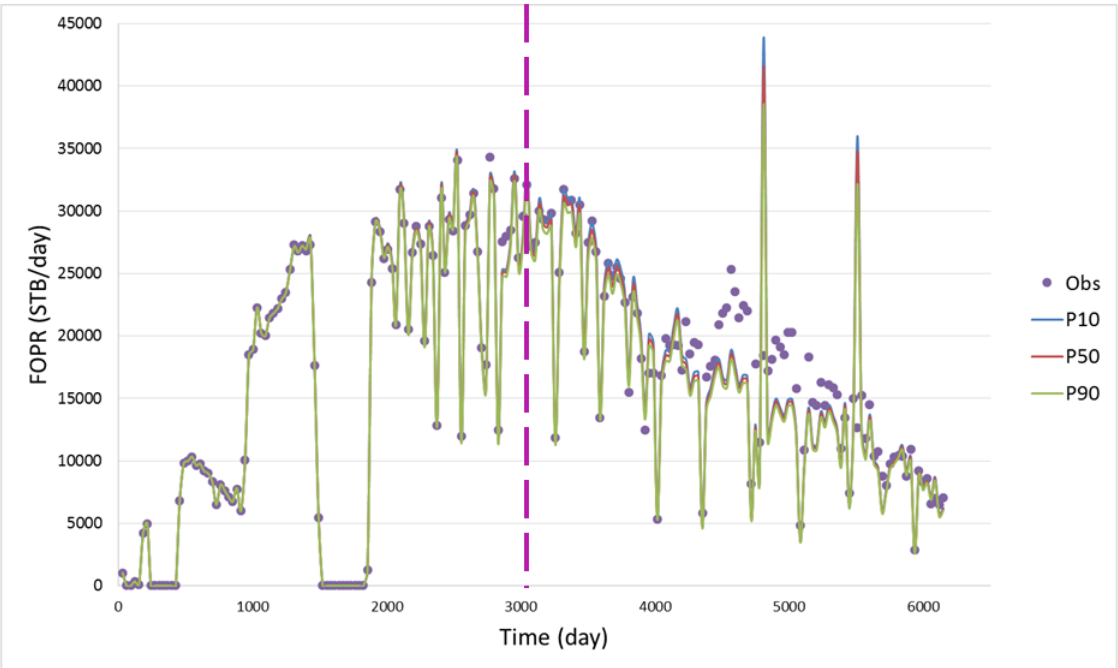
Figure 6.14: 175 realizations using MLMCMC for 8 unknown parameters of Scapa field.

We choose 3137 days for the historical period and 2975 days for the forecasting period. Figure 6.15(a), obtained by the same set-up as in Figure 6.14, shows

Bayesian credible intervals $P10-P50-P90$ for Scapa with MLMCMC. Figure 6.15(b), obtained by the same set-up as Figure 6.15(a) except σ, is dependent on time follows the biased estimator in Chapter 2, shows Bayesian credible intervals $P10-P50-P90$ for Scapa with MLMCMC. As shown, when σ in (6.5) is variable, we obtain more confident Bayesian credible intervals than when σ is constant as shown in the figure, more of observed data points captured by the distribution in case of σ is variable (dependent on time). Sometimes, it is hard to choose σ at every time step.



(a)



(b)

Figure 6.15: Bayesian credible intervals $P10-P50-P90$ for Scapa with MLMCMC, the dashed vertical line is the end of history period, σ in (2.11) is (a) constant (b) variable as in Subsection 2.6.0.2.

6.6 Summary

The aims of this chapter were to explore the feasibility of using MLMCMC to quantify the uncertainty in different scenarios, compare MLMCMC and RWM, with the same accuracy, and finally with the same computational cost compare MLMCMC with (PSO combined with NAB). We demonstrated the performance of the MLMCMC algorithm on two case studies Teal South field and Scapa field.

- For Teal South field, MLMCMC success fully obtained the histogram for the uncertain parameters as shown in Figure 6.4.
- For Teal South field, RWM with 10^4 realizations is not enough to obtain the uncertainty range to capture all the observed data, unlike MLMCMC with less computational cost as shown in Figure 6.5.
- For Teal South field, MLMCMC is faster than RWM and the speed up over RWM is in the range 10 to 100 as shown in Figure 6.9.
- For Teal South, using PSO and NAB with the same computational cost as MLMCMC. MLMCMC is more efficient than PSO with NAB because for the same computational cost MLMCMC capture the most of the observed data than PSO with NAB as shown in Figure 6.11.
- For Teal South field, RWM is the slowest technique compared with MLMCMCM augmented with different constraints for the number of samples required for MLMCMC as shown in Figure 6.10.
- For Teal South field, MLMCMC was able to generate forecasts significantly faster than RWM with no significant loss in accuracy.
- For Scapa field, MLMCMC estimates the Bayesian uncertainty range for the solution. The uncertainty distribution is more efficient when using σ as variable in (6.5) than when it is constant as shown in Figure 6.15.
- For Scapa, MLMCMC performed in a similar way to Teal South, with the difference that we could not afford to run the full RWM on Scapa to get a cost

comparison.

Overall, MLMCMC is applicable for quantifying the uncertainty in reservoir simulation. It is faster than RWM and as efficient as long chain RWM.

In the following chapter, we discuss another sampling technique for decreasing the computational cost that is also based on the multilevel concept.

Chapter 7

Multilevel Proxy for Quantifying Uncertainty

This chapter proposes a new approach, based on the multilevel concept, for improving and using a proxy to increase the confidence in the solution if we use it for inference, with less computational cost.

This chapter is structured as follows; first of all we review some experimental design techniques. Following this, we discuss Radial Basis Function (RBF), then how to build a proxy, using experimental design, with RBF. After that, we discuss how to use this proxy to construct an error model. Finally, we propose a new approach for improving the proxy based on the multilevel concept.

In reservoir simulation, one of the important tasks is quantifying uncertainties as high-cost investment decisions are based on these simulation predictions. This task requires running the computationally expensive reservoir simulation code. Therefore, it is desirable to find a way of decreasing this cost. One way of decreasing the computational cost is by constructing a proxy.

The goal of the proxy is to replace the reservoir simulator by a map from the parameter space to the quantity of interest. The aim is to reduce the number of expensive fine scale simulations and decrease the number of rejected samples. The cost of constructing the proxy is about $100 - 200$ function evaluations. After the proxy has been constructed, the computational cost for the proxy can be neglected.

Inference from the proxy response only is risky because it is based on a sim-

plified physical description and neglects important physical features (Sefat et al., 2012). Moreover, (Goodwin, 2015) shows how to combine Hamiltonian Monte Carlo (describe in Chapter 8) with proxy to obtain a reliable probabilistic uncertainty quantification for the forecasting.

7.1 Experimental Design

The goal of using experimental design is to find a suitable configuration (structure) for the data sets (Yu et al., 2008). Each simulation might require about 15 min–6 hours to run on average, which is extremely expensive. One way to decrease the computational cost is by constructing an accurate proxy (emulator) to predict the response of every data point. We hope to build an emulator with the minimum number of realizations, which we can then use to estimate the distribution of the quantity of interest.

There are different techniques for experimental design, for example, Random sampling, Latin Hypercube sampling, screening designs and Sobol sequence (Alpak and Kats, 2009). To obtain an efficient emulator, requires designing a sampling strategy to explore the parameter space and analysing the experimental results (Yeten et al., 2005). This section compares the sampling performance of different experimental designs.

7.1.1 Random Sampling

Random sampling generates randomly distributed samples based on a probability distribution. Figure 7.1 shows that how the random samples are distributed. We can see gaps where there are no samples. This will affect the sensitivity analysis and quantifying the uncertainty (Burhenne et al., 2011). Covering the parameter space is an important issue when constructing a reliable proxy that is able to interpolate all the points in the parameter space (Yu et al., 2008).

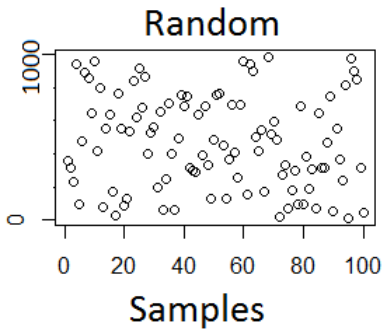


Figure 7.1: The sampling performance of random sampling, using 100 samples generated from $\mathcal{U}(10, 1000)$.

7.1.2 Stratified sampling

A stratified sampling technique is designed as follows. Divide the parameter domain into equal subintervals, so that each subinterval contains the same number of samples. This is to try to avoid gaps and clusters. The required number of samples, M, for this design is

$$M = s^d,$$

where, d is the number of parameters (dimension) and s, is the number of strata. How to choose a large enough number of strata to avoid gaps and clusters is a critical issue (Burhenne et al., 2011).

7.1.3 Latin Hypercube Sampling (LHS)

Latin Hypercube Sampling (LHS) is a special case of stratified sampling related to the quasi Monte Carlo method (Caflisch, 1998). It was introduced by (McKay et al., 1979) in a computer experiment. LHS was used to select parameters for Monte Carlo simulations (McKay et al., 1979; Iman and Conovera, 1980). It has been applied in different areas, namely soil science (Minasny and McBratney, 2002) and geostatistics (Pebesma and Heuvelink, 1999).

LHS guarantees covering the range of each parameter. This technique is based on dividing the initial domain into non-overlapping subsets. Then we generate samples such that, from each subset, we have only one sample as shown in Figure 7.2. Figure 7.2 shows how LHS with 5 samples from a uniform random variable $\mathcal{U}(0, 1)$ can be distributed in one dimension.

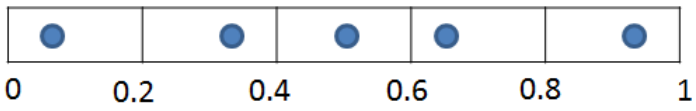


Figure 7.2: LHS with 5 samples from a uniform random variable $\mathcal{U}(0,1)$.

Generally, in the case of having d parameters, we need N samples. If we scale the parameter domain to $[0,1]$,

$$I_i = \{[\frac{i-1}{N-1}, \frac{i}{N-1}],\ 1 \leq i \leq N-1\}$$

where, I_i is the subinterval in the domain. LHS procedures can be summarised as follows, divide the cumulative distribution function domain for each parameter into N intervals. Pick one random number from each interval. Using the inverse map of the distribution, we obtain the exact values of the parameters in each interval. Then, for each parameter, the N values are combined after a random shuffle (Minasny and McBratney, 2006).

Figure 7.3 shows two possibilities of configuration for LHS in two dimensions (a) shows a correlation between the parameters and (b) shows that there are gaps in two dimensional problems and the parameter space is badly explored. Therefore, in high dimensional space, we have a poor exploration for the space.

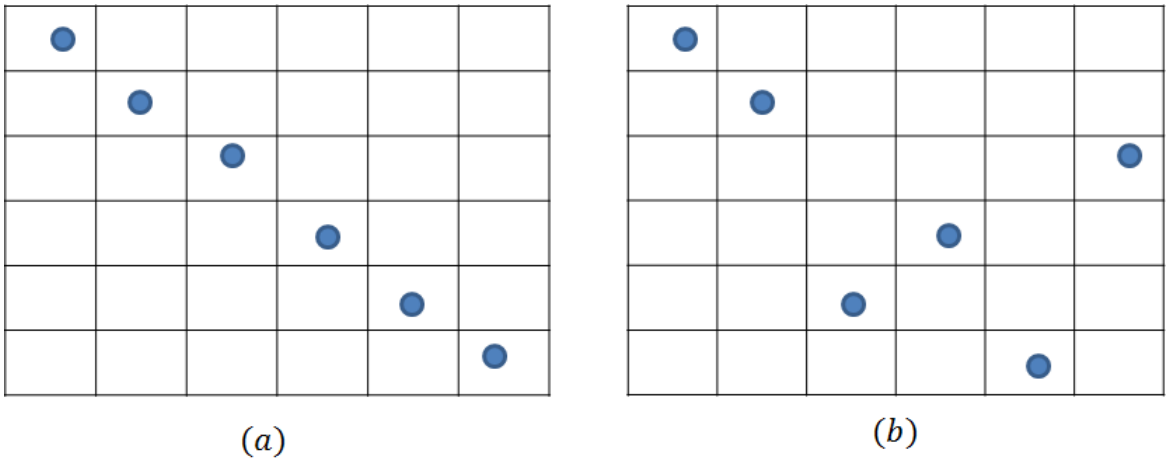


Figure 7.3: Two possible configurations for LHS with 6 samples.

Figure 7.4 shows there are gaps and clusters in the case of using LHS as random sampling. The reason for these is that the number of samples is not large enough to generate a sample with the same density (Burhenne et al., 2011; Saltelli et al.,

2008).

7.1.4 Sobol Sequence

Sobol sequence is a kind of quasi random sampling, designed to generate samples distributed uniformly over a unit hypercube with d dimensions (Burhenne et al., 2011; Saltelli et al., 2010). The generated samples are based on binary functions related to irreducible polynomials over the field $\{0, 1\}$. This design explores the parameter space of the quantity of interest (Sobol' and Levitan, 1999). If the number of samples is large enough, the parameter space will be explored easily. For more details related to how to construct Sobol Sequence see (Bratley and Fox, 1988).

Figure 7.4 shows a comparison of the sampling performance of LHS and the Sobol sequence. The figure shows that Sobol sequence explores the space more efficiently than LHS. In terms of Sobol sequence decreases the gaps between the data points compared to LHS.

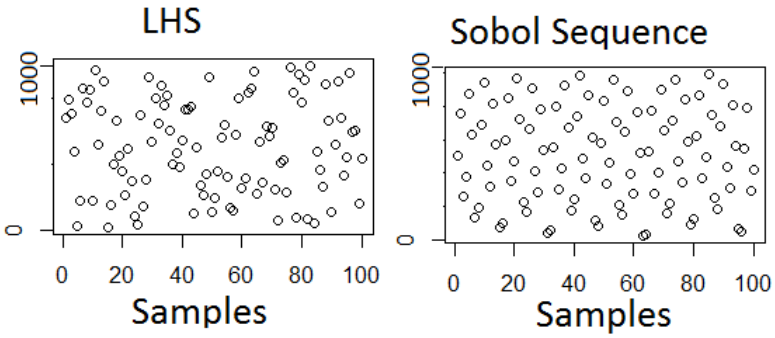


Figure 7.4: Compare the sampling performance between LHS and Sobol sequence, using 100 samples generated from $\mathcal{U}(10, 1000)$.

(Burhenne et al., 2011) recommended using Sobol sequence or LHS when using Monte Carlo techniques to be sure we explore all the space. The number of samples affects the performance of the sampling technique.

7.1.5 Radial Basis Function (RBF)

RBF was introduced by Rolland Hardy (Hardy, 1971). He presented a multiquadric basis function, one type of RBF. RBF is a powerful method for interpolating scattered data. It maps the input data onto a continuous function. It depends on the

radial distance from a point.

The RBF interpolation formula is

$$F(\mathbf{X}) = \sum_{i=1}^{N} \omega_i \, \phi(\|\mathbf{X} - \mathbf{X}_i\|),$$

where $\mathbf{X}$ is the input data, $\|.\|$ is Euclidean norm, ω_i are the weights, $F(\mathbf{X})$ is the output, $\mathbf{X}_i$ are the centre points, $\phi : [0, \infty) \to \mathbb{R}$ is the radial function and N is the number of basis functions. A RBF approximates a multivariate function by $F : \mathbb{R}^d \to \mathbb{R}^m$. This is can be written in matrix form as

$$\mathbf{F} = \boldsymbol{\phi}\, \mathbf{w} \Rightarrow \mathbf{w} = \boldsymbol{\phi}^{-1}\mathbf{F},$$

in the case of $\boldsymbol{\phi}$ being a non-singular matrix.

One feature of a RBF is that it is a monotonic function with respect to the distance from the centre point. The centre point, the function shape and the weights are unknown parameters (Orr, 1996). The centres are chosen randomly from the training data. There are different types of radial basis functions, e.g. linear and thin plate spline. The thin plate spline is two-dimensional and is one of the fundamental solutions to the biharmonic equation $\bigtriangledown^4 \phi = 0$, with the kernel form $\phi(r) = r^2 \ln r$. We use the linear and thin plate type of RBF in 8 dimensions as we look at on 8 dimensional parameter space below. Due to the computational cost of calculating the inverse matrix and then fitting all the data, it is inefficient.

In this thesis, we modify the RBF function, which is built into PYTHON and use it to interpolate the time series FOPR. Figure 7.5 shows that interpolation using the RBF is accurate. For more details about RBF, see (Buhmann, 2003; Iske, 2004; Busby et al., 2007b).

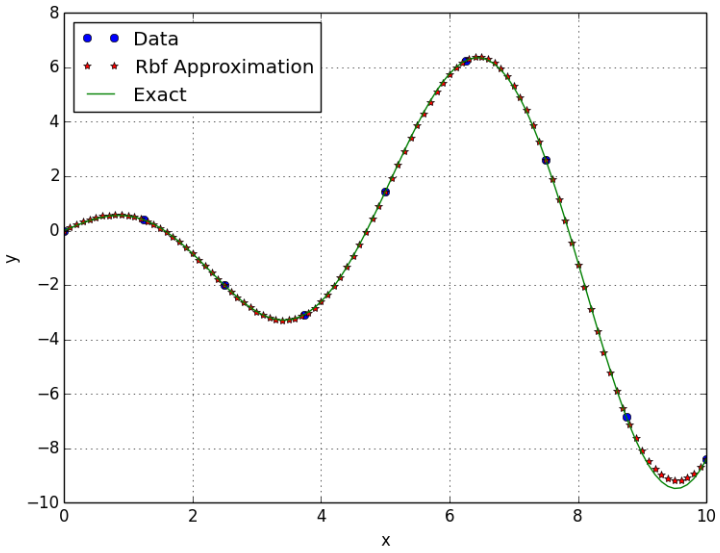


Figure 7.5: Interpolation of data generated by $x\cos(x)$ using RBF with a multi-quadrics kernel type.

7.2 Proxy Model

The proxy (surrogate model) is an approach to mimic the actual behaviour of the reservoir simulation. The main idea is to find a relationship between the input parameters (scattered data points) and simulation output using an interpolation technique. The interpolation can be done for the simulation output (e.g. FOPR) or the misfit function. Based on this, we can find the quantity of interest at every input parameter vector (Busby et al., 2007b). The advantage of using the proxy is we can decrease the number of reservoir simulation runs. This is especially important when the simulation is too expensive to run, taking many hours for one realization (Josset et al., 2015).

The goal is to use a few data points to construct an accurate function to decrease the computational cost. Constructing the proxy (emulator) requires the following steps (Busby et al., 2007b)

- Checking the sensitivity of our models in terms of reducing the number of unknown parameters.
- Experimental design for constructing the input data configuration.
- Scattering data points in the parameter space for input data and the corre-

sponding output.

- Choosing the interpolated technique, e.g., RBF.

The proxy can be used with any MCMC algorithm to quantify the uncertainty. (Mohaghegh, 2006) used the Monte Carlo method and surrogate reservoir model to quantify the uncertainty. (Christie et al., 2006) used a neural network to build the proxy model. They conclude that, based on the comparison between the simulated and proxy models, if we get a bad match in some regions, then we can generate extra samples to improve the proxy model. Also, the initialization and training stages of the proxy model are very important for obtaining a good proxy model. (Zubarev, 2009) presented a comparative study for assisted history matching between different proxy models and full numerical simulations. Zubarev concludes that the efficiency of all the proxy models depends on the complexity of the problem. The hierarchical nonlinear approximation, proposed by (Busby et al., 2007b), is in the parameter space and fills the gaps. They modify the proxy by starting with the initial proxy and then generate more samples to be sure to explore the domain.

Algorithm 7 step 4, we use the Random Walk Metropolis (RWM) for sampling the testing data because the aim is to use the proxy with RWM for estimating the uncertainty distribution. In the case of using random samples to test the quality of proxy, these random samples are wasted and will not feed into the uncertainty distribution.

Algorithm 7: Proxy

1: Fix the number of initial samples, choose the experimental design (LHS or Sobol sequence), fix the step size for RWM and the R-squared value required (stopping criterion).

2: Generate initial datasets using the experimental design technique, then run the flow simulation to create the training data.

3: Construct the map between input parameters and the output (quantity of interest). Estimate the surface of the quantity of interest using RBF, for example (we have to choose the kernel type).

4: After finding the weights, create test data using RWM. Then, test the quality of the proxy based on the R-squared value between interpolated and simulated data.

5: Check the criterion (R-squared close to one), if it is satisfied, use RWM with the proxy to quantify the uncertainty, otherwise, add more samples to the initial samples and find the new weights and check again until the test data satisfies the criterion.

The input parameters for the proxy are unknown parameters. Therefore, further development is required for the reservoir description (Demyanov et al., 2010). We compare the proxy result and the simulated result for the same set of parameters and decided how good the proxy is. This can be done by plotting the testing simulated results profile and the interpolated result. Also, construct the cross plot of the simulated results against the interpolated results, then check the regression coefficient, if it is close to 1, we have a good proxy (Schaaf et al., 2008). In the following section, we use the proxy to estimate the error between the coarse and fine solutions.

7.3 Error Model

Decision making is based on accurate predictions. Obtaining an accurate prediction is very hard because it is impossible to quantify the simulation error (Carter, 2004). In the past, the simulation error has been ignored and the coarse grid model

preferable to reduce the CPU time.

The error model can be used to improve the prediction. It has been designed to decrease the bias error. It is based on using the result from the coarse model with the error data to decrease the computational cost without loosing the efficiency obtained by the finest model (O'Sullivan and Christie, 2006a; Christie et al., 2008). Solution error modelling is based on a limited number of realizations. We can define the solution error as,

$$\text{Error}(t, \mathbf{m}) = \text{Fine}(t, \mathbf{m}) - \text{Coarse}(t, \mathbf{m}),$$

where, Fine and Coarse refer to the solution using fine and coarse grids respectively, t refers to time and $\mathbf{m}$ is the parameter vector.

The first stage in solution error modelling is to decide which models to use to perform the set of fine and coarse simulations. The next stage is to interpolate the errors throughout the parameter space. If a small number of parameters are being used a simple method such as linear interpolation can be used. However, with higher dimensions, a more sophisticated method is advisable.

(Glimm et al., 2003, 2004) introduced the solution error model. They compared the solution of two-phase flow on a coarse and a fine grid. They concluded that there is a linear relationship between the parameters and that the error corresponds to this. (O'Sullivan, 2004; O'Sullivan and Christie, 2005, 2006b) improved the result from Todd and Longstaff's approximation for a heterogeneous medium with unstable miscible flow. (Glimm et al., 2001b) formulated and analysed the probability model for the numerical coarse grid solution error.

7.4 Multilevel Proxy (MLproxy)

Imagine you want to estimate the posterior distribution. You can apply RWM to do this, but it requires a huge computational cost. (Schulze-Riegert et al., 2013; Helbert et al., 2009; Alpak et al., 2013) combine the proxy with the RWM technique to estimate the distribution with less computational cost compared to RWM.

We now show a different way of decreasing the computational cost without loss of efficiency. The technique combines the multilevel concept with the proxy:

$$\text{sim} = \text{proxy} + (\text{sim} - \text{proxy.}) \tag{7.1}$$

Instead of using the extremely expensive RWM (sim), to estimate the distribution, it uses the proxy with RWM (proxy) and the correction term between the simulated results and interpolated results, (sim − proxy). A proxy is cheaper to run than the full simulation with RWM. The correction imperfection by running RWM is the difference (sim − proxy). Moreover, the computational cost decreases because the distribution of the differences obtained is narrower than the distribution of the simulation response.

Constructing the proxy can be done using Algorithm 7. Then, to estimate the first term in (7.1), we can use RWM Algorithm 5, and to estimate the correction term, we use Algorithm 8.

Algorithm 8 has two levels, with level $l = 1$ given by the proxy and level $l = 2$ by the difference between sim and proxy, $\mathbf{x}_i^p$, $\mathbf{x}_i^s$ refer to parameter vectors for the proxy and simulated respectively. $\mathbb{P}(x)$ refers to the prior distribution for the parameters and $\pi(x|\mathcal{D})$ refers to the posterior density. Recall from Chapter 6 that we seek to control the variance so that,

$$\text{control}(\mathbb{V}(\mathbb{E}(U_L))) \leq \epsilon^2 V_0/2 \tag{7.2}$$

where U_L is the solution of the simulated FOPR, V_0, is scaling the equations and ϵ is the percentage error compared with the initial variance V_0 (the variance of the estimator is of order $\epsilon^2 V_0$). The accept-reject criterion for the correction term (sim − proxy) is similar to the accept-reject criterion for two–stage MCMC (Efendiev et al., 2005; Christen and Fox, 2005).

Algorithm 8: Two-level proxy

Fix a number of warm-up samples M_{up}, the variance of the error in the observed data V_0, fix the standard deviation for step size $\boldsymbol{\sigma}$ and the accuracy ϵ.

if $l = 1$ **then**

Use RWM Algorithm 5, with $M_{samples} = M_{up}$

Check condition (7.2). If it is satisfied, go to the second level. Otherwise, add more samples.

else

Initialized $\boldsymbol{x}_0^p \sim \mathbb{P}(x)$ and $\boldsymbol{x}_0^s = \boldsymbol{x}_0^p$

while $(M_{up} > 0)$ **do**

for $i = 1$,**to** M_{up} **do**

$\triangle_{i-1} \sim \mathcal{N}(0, \boldsymbol{\sigma}^2)$

$\boldsymbol{x}' = \boldsymbol{x}_{i-1}^p + \triangle_{i-1}$

Draw $\alpha \sim \mathcal{U}(0,1)$

if $\alpha < \min\left\{1, \dfrac{\pi^p(\boldsymbol{x}'|\mathcal{D})}{\pi^p(\boldsymbol{x}_{i-1}^p|\mathcal{D})}\right\}$ **then**

$\alpha_1 \sim \mathcal{U}(0,1)$

if $\alpha_1 < \min\left\{1, \dfrac{\pi^s(\boldsymbol{x}'|\mathcal{D})}{\pi^s(\boldsymbol{x}_{i-1}^s|\mathcal{D})} \dfrac{\pi^p(\boldsymbol{x}_{i-1}^p|\mathcal{D})}{\pi^p(\boldsymbol{x}'|\mathcal{D})}\right\}$ **then**

$\boldsymbol{x}_i^s = \boldsymbol{x}^s$

else

$\boldsymbol{x}_i^s = \boldsymbol{x}_{i-1}^s$

end

$\boldsymbol{x}_i^p = \boldsymbol{x}'$

else

$\boldsymbol{x}_i^p = \boldsymbol{x}_{i-1}^p$

$\boldsymbol{x}_i^s = \boldsymbol{x}_{i-1}^s$

end

end

Check condition (7.2) and update the required number of samples, M_l

$M_{up} = M_l - M_{up}$

end

end

Figure 7.6 shows the flowchart for producing the proxy and test it. Figure 7.7 shows

the flowchart for how to estimate the difference between the proxy and the simulated result (correction term). In Figures 7.6 and 7.7, check the condition refers to (7.2) with (7.4).

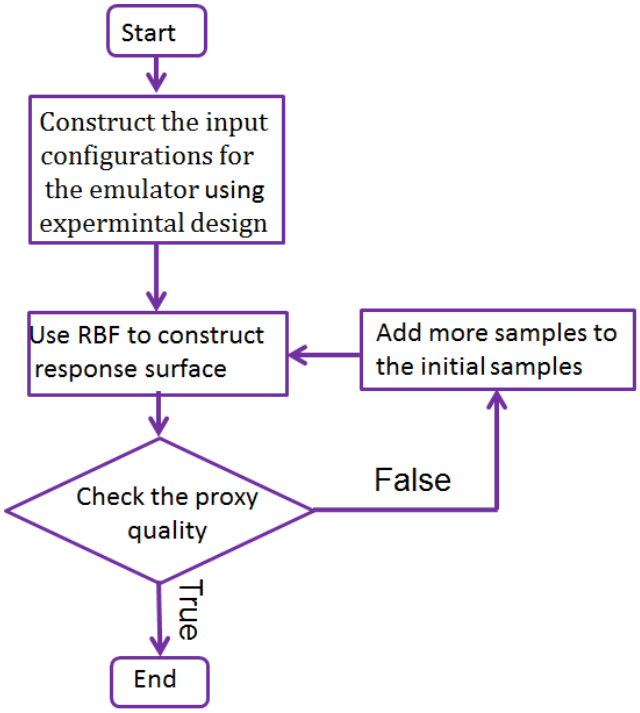


Figure 7.6: Flowchart for estimating the proxy.

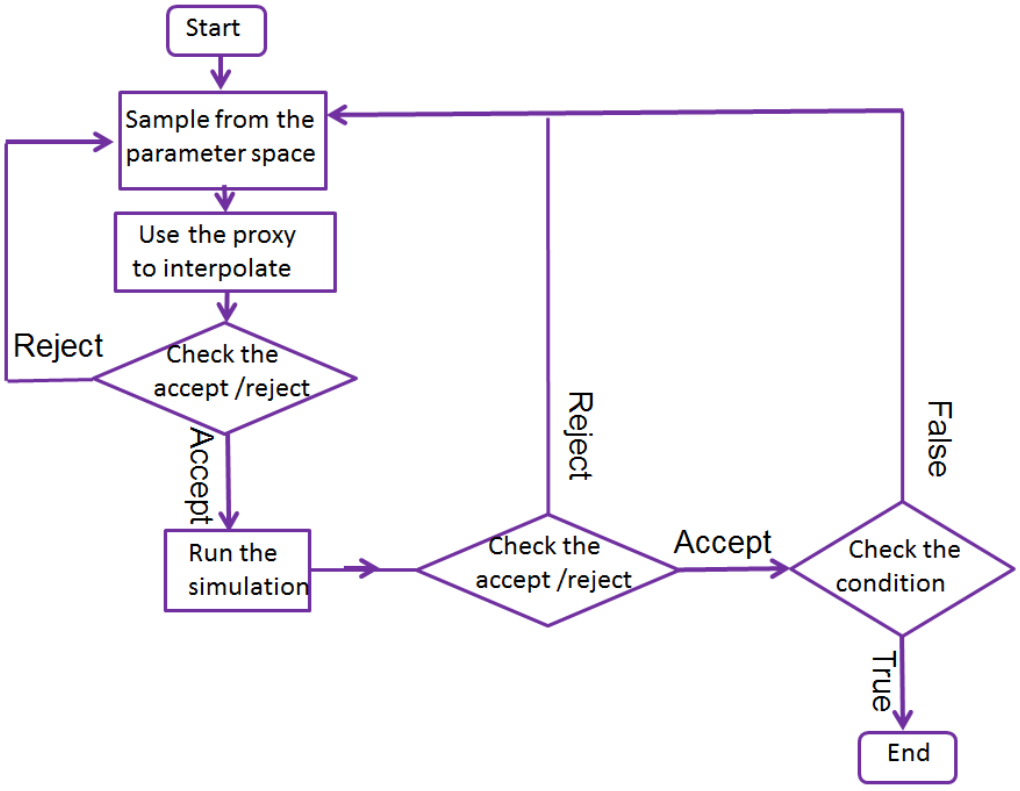


Figure 7.7: Flowchart for estimating $(sim - proxy)$.

It is straightforward to generalise the algorithm to $l > 2$ terms. The simulated result using a fine grid can be replaced by the proxy result constructed using a coarse grid, added to the correction between the simulated results using coarse grid and the proxy and added to the correction term between the simulated results using fine and coarse grids (error model idea).

7.4.1 Optimising the Number of Samples

The total computational cost for two level proxy is the sum of the computational cost for constructing the proxy, estimating the proxy distribution and the computational cost for estimating the (sim − proxy) distribution. This technique is a variance reduction technique. As the distribution of the second term is narrow, it requires few samples. Minimising the variance of the estimator (e.g., mean) with respect to the number of samples, M_l, required for every level gives,

$$\frac{\partial \mathbb{V}(\mathbb{E}(U_L))}{\partial M_l} = 0,\ l = 1, 2 \tag{7.3}$$

where, l refer to a first and second terms in (7.1) and U_L is the simulated oil rate. Obtaining the formula for the number of samples required at each level (M_1 and M_2) is based on solving Equation (7.3) under a constraint (that the total computational cost is constant). This gives

$$M_1 = \frac{2}{V_0\,\epsilon^2}\sqrt{\mathbb{V}_{l_1}}\left(\sqrt{\mathbb{V}_{l_1}} + \sqrt{\mathbb{V}_{l_2} \times \text{no of grid blocks}}\right) \tag{7.4a}$$

$$M_2 = \frac{2}{V_0\,\epsilon^2}\sqrt{\frac{\mathbb{V}_{l_2}}{\text{no of grid blocks}}}\left(\sqrt{\mathbb{V}_{l_1}} + \sqrt{\mathbb{V}_{l_2} \times \text{no of grid blocks}}\right) \tag{7.4b}$$

where, the variance of all the realizations from the two terms are $\mathbb{V}_{l_1}$ and $\mathbb{V}_{l_2}$, V_0 is the variance of the error in the observed data to scale the equation (7.4) and ϵ is the percentage error compared to V_0 (the variance of the estimator is of order $\epsilon^2\,V_0$). Therefore, by using (7.4), we can obtain

$$\frac{M_{l_2}}{M_{l_1}} = \sqrt{\frac{1}{\text{no of grid blocks}}}\sqrt{\frac{\mathbb{V}_{l_2}}{\mathbb{V}_{l_1}}} < 1.$$

For multiple levels, we apply the same techniques to obtain the number of samples required.

7.5 Results

We use the same synthetic reservoir model studied in Chapter 6, Teal South, to demonstrate the effectiveness of proxy, error model and MLproxy.

In Figures 7.8, 7.9 and 7.10, we use a coarse grid, which is $11 \times 11 \times 5$. Different colours in the plots corresponding to different realizations.

Figure 7.8 shows that the interpolated data and the simulated data for FOPR using the Sobol sequence to construct the initial samples. As shown in the figure, the area enclosed by the black ellipse is not estimated well because there is a difference between the interpolated and simulated result. This indicates that the proxy needs to be improved. The area enclosed by the black ellipse is important because later, we use the first 181 days for history matching when we estimate the distribution of the uncertainty.

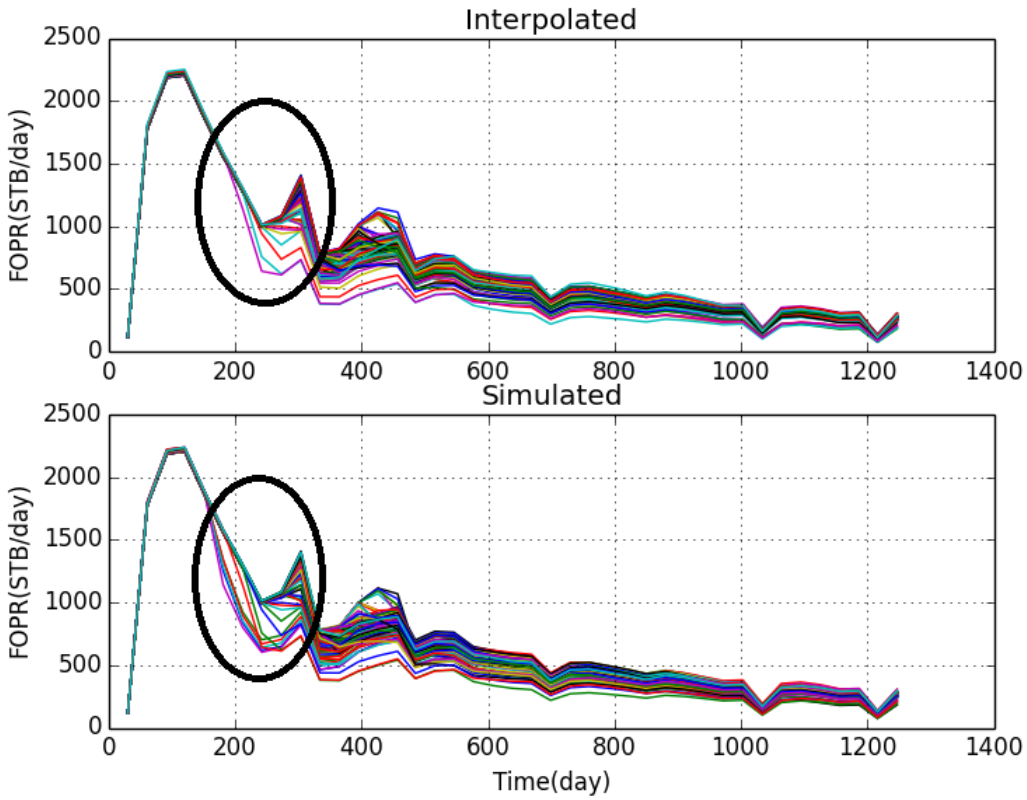


Figure 7.8: Testing data for 100 samples are constructed by (top) the proxy (bottom) the exact simulation using Sobol sequence with the thin plate kernel for RBF.

Figure 7.9 shows the cross plot for 30 samples between simulated FOPR and interpolated FOPR using random sampling, LHS and Sobol sequence for constructing 100 initial samples for the proxy and using RBF with different kernel types: linear and thin plate. The figure shows the performance of the proxy for each type. We can see that using the Sobol sequence with a linear kernel is the best. However, the thin plate is a bad choice of the kernel. Also for the LHS and random sampling the proxy needs to be improved more than the one created by Sobol sequence.

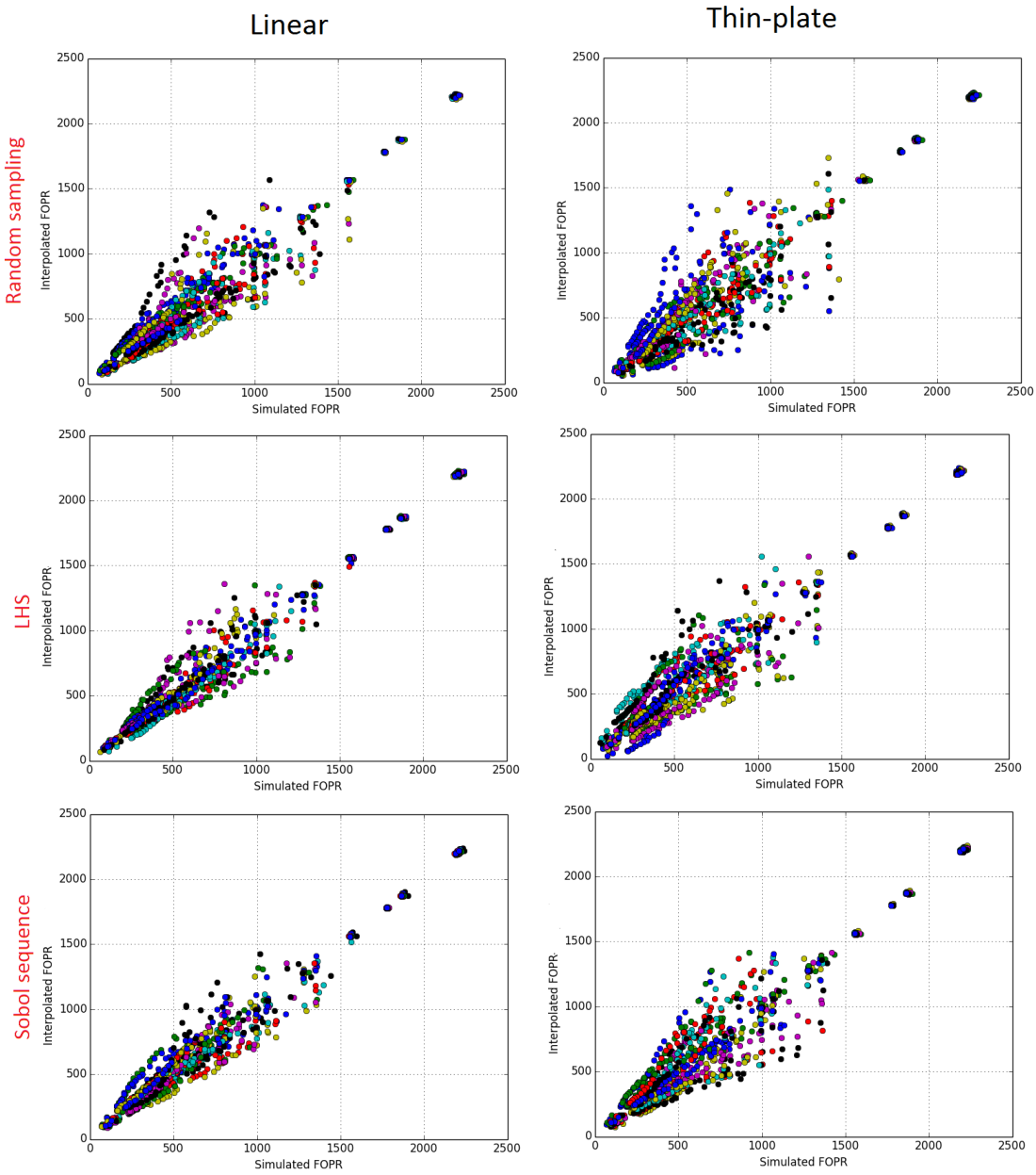


Figure 7.9: (top) Cross plot between simulated FOPR and interpolated FOPR using random sampling and RBF with (left) Linear kernel (right) Thin Plate kernel. (mid) Cross plot between simulated FOPR and interpolated FOPR using LHS and RBF with (left) Linear kernel (right) Thin Plate kernel. (bottom) Cross plot between simulated FOPR and interpolated FOPR using Sobol sequence and RBF with (left) Linear kernel (right) Thin Plate kernel.

In Figure 7.10, the initial samples were constructed using the Sobol sequence. The figure shows that the interpolated data and the simulated data for FOPR are 'similar' to each other, which means Two-level proxy improves the quality of the proxy in Figure 7.8.

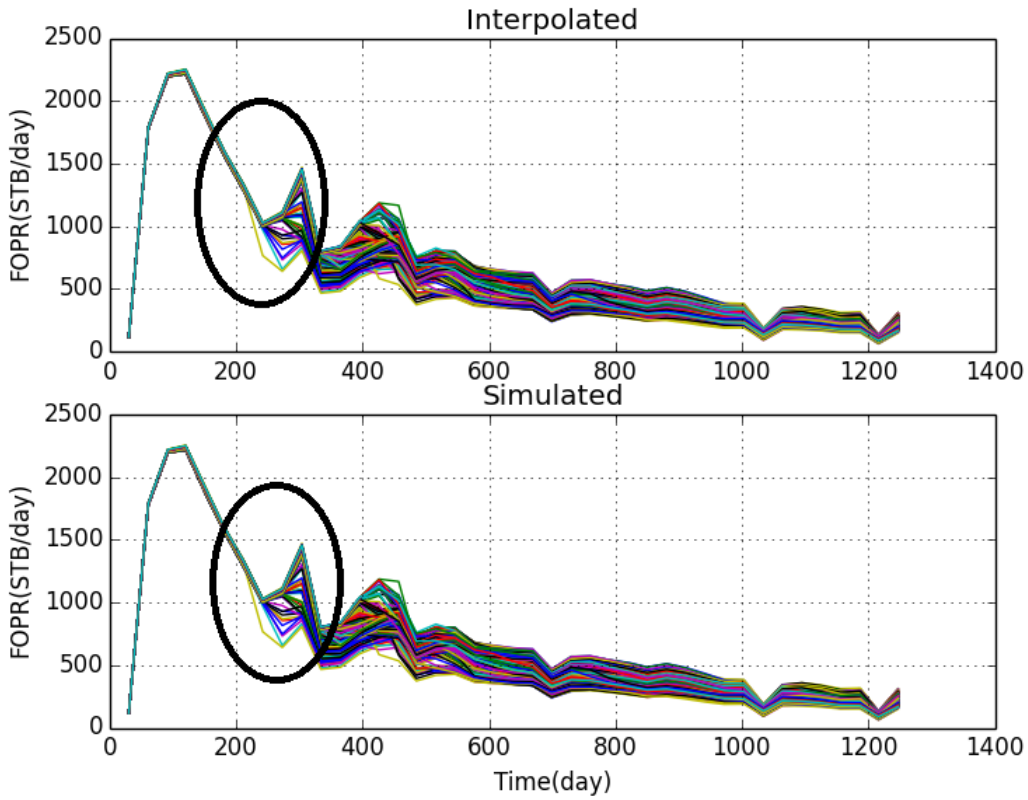


Figure 7.10: Testing data for 100 samples constructed by (top) the Two-level proxy (bottom) the exact simulation using Sobol sequence with a thin plate kernel for RBF.

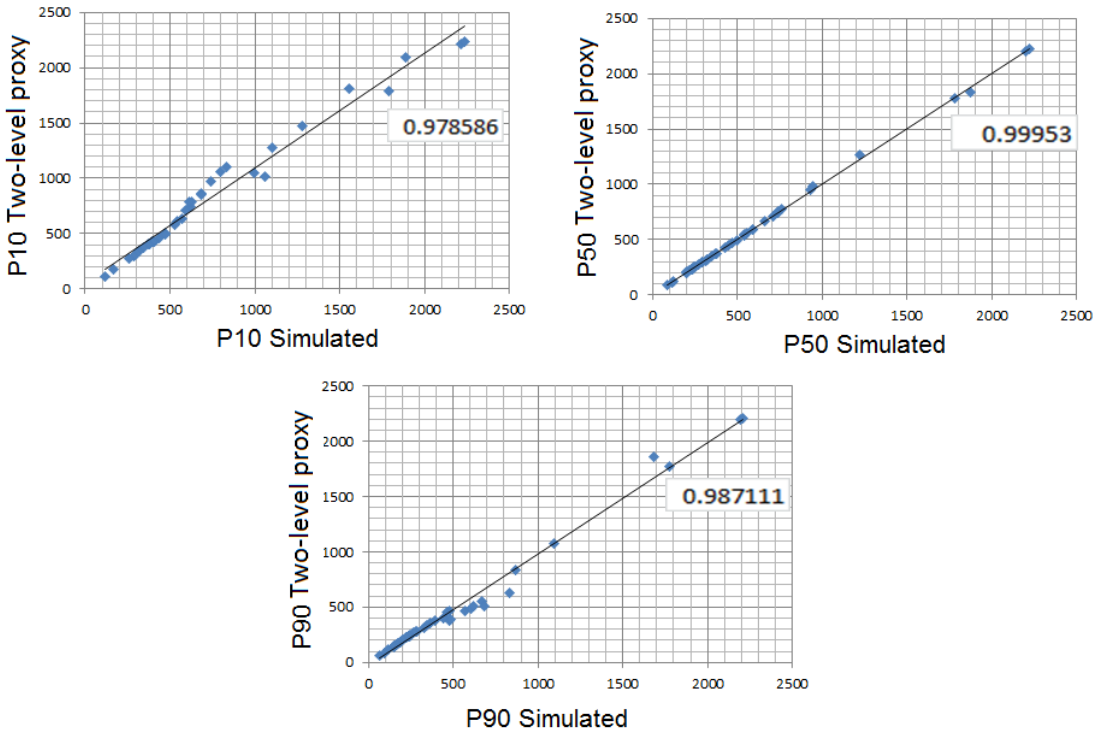


Figure 7.11: Cross plot between the credible Bayesian interval, $P10 - P50 - P90$ for the simulated and Two-level proxy result.

Figure 7.11 shows that the simulated result and Two-level proxy result have almost the same distribution, i.e. $P10$, $P50$ and $P90$ are highly correlated. In the Teal South example, there are 8 unknown parameters, which we have imposed maximum and minimum limits on. If we ran simulations for all combinations of the maximum and minimum parameter values, we would have to perform 2^8 sets, which is time consuming. Instead, we chose 100 of these randomly. Then we performed the fine and coarse simulations and calculated the errors. The coarse grid is $11 \times 11 \times 5$ and the fine grid is $11 \times 11 \times 15$. Figure 7.12 shows that the proxy for estimating the error between the solutions using fine and coarse grids. The proxy is accurate in terms of the error between the interpolated error and the simulated error, as shown in Figure 7.13. Different colours in the plots correspond to different realizations.

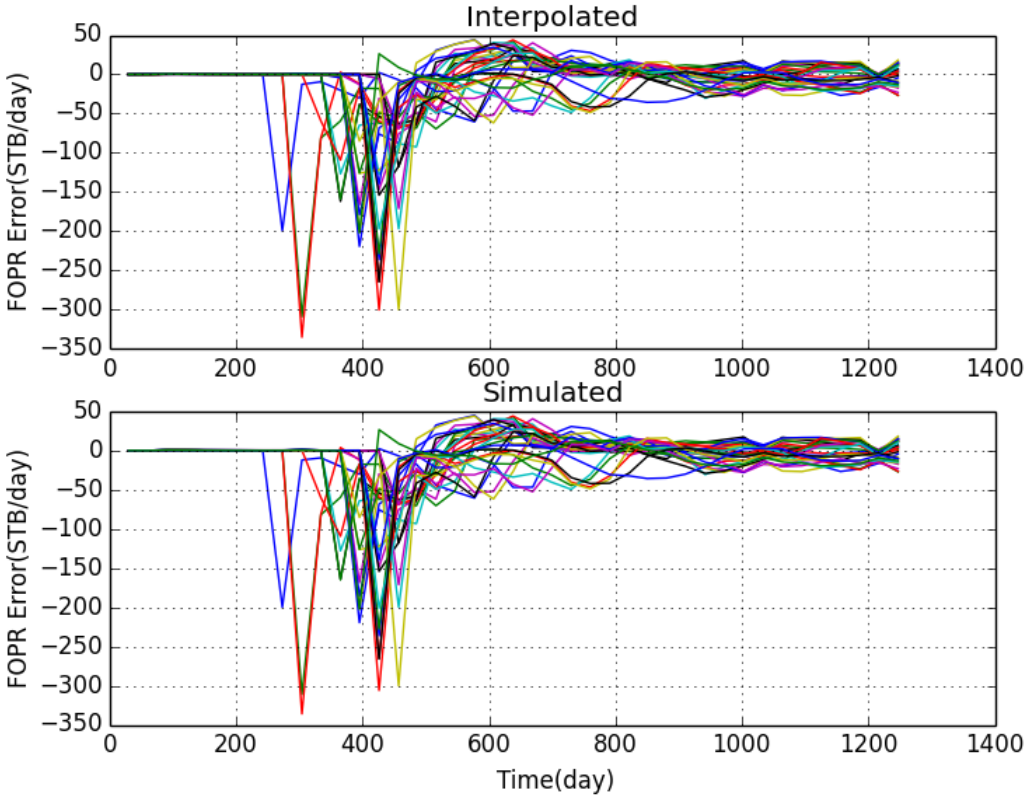


Figure 7.12: Learning (testing) data for 30 samples to show (top) the interpolated error between the fine and the coarse solution (bottom) the simulated error between the fine and the coarse solution.

The idea of Two-level proxy can easily be generalized. In the case of estimating

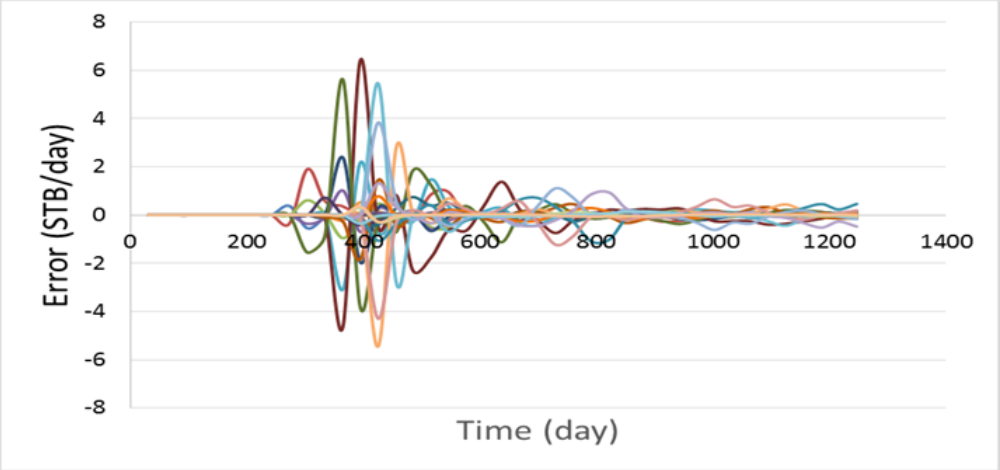


Figure 7.13: Error between the interpolated and the simulated result from Figure 7.12.

the distribution of FOPR using the fine grid $11 \times 11 \times 15$,

$$\text{sim}_f = \text{Proxy}_c + (\text{sim}_c - \text{Proxy}_c) + (\text{sim}_f - \text{sim}_c) \tag{7.5}$$

In Equation (7.5), the first two terms are the Two-level proxy and the last part is the solution errors on coarse and fine grids. Figure 7.14 shows the extremely computationally expensive, simulated distribution created by RWM, with 50000 realizations. The simulated solution is accurate. The figure shows that the estimated distribution created by RWM based on the proxy for the coarse solution is cheaper. The proxy is created using a Sobol sequence for configuring the initial 100 samples, then checking the criterion for accepting the proxy and if it is not satisfied, adding more samples using RWM. The approximated solution using the proxy is not more accurate. However, by adding the correction terms, we obtain a better distribution as shown in the figure. This means the distribution of FOPR can be improved using MLproxy, with less computational cost. The CPU-time for the simulated result is about 7 days but for MLproxy it is about a day.

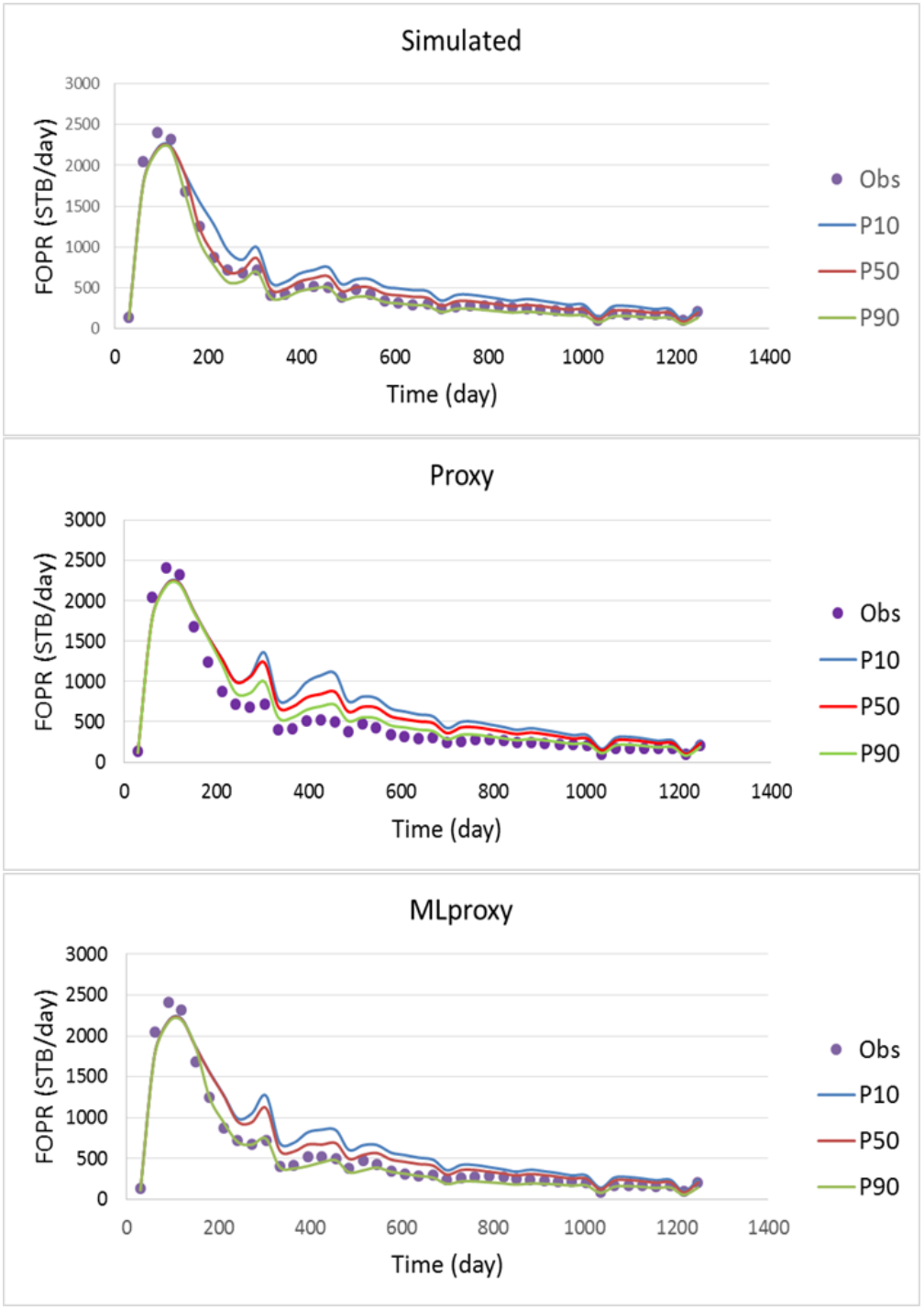


Figure 7.14: Bayesian credible intervals $P10 - P50 - P90$ (top) simulated for the fine grid (middle) proxy for the coarse (bottom) MLproxy.

7.6 Summary

The target of this chapter is decreasing the computational cost of uncertainty quantification without losing the efficiency of the solution by using the multilevel concept for improving the quality of the proxy (emulator). This proxy is based on a few realizations, which is cheaper than running the full flow simulation. We show results

for Teal South field.

The chapter showed the following:

- Different experimental design for constructing the initial samples to build the proxy. We showed that the Sobol sequence is the best choice compared to random sampling and LHS for exploring the parameter space as shown in Figure 7.4.
- RBF is a good interpolation method, as shown in Figure 7.5, because the interpolated result is similar to the exact result.
- I constructed PYTHON code to interpolate a time series using RBF.
- A linear kernel is better than a thin plate kernel in the RBF for interpolating the misfit surface of Teal South model as shown in Figure 7.9 because the simulated misfit is highly linearly correlated with the interpolated misfit.
- In most of the situations we faced the proxy should be improved as shown in Figure 7.8 because it missed some physical elements of the problem.
- Two-level proxy improves the quality of the proxy (see Figure 7.10) because it obtains the same uncertainty distribution as the simulated results as shown in Figure 7.11. The computational cost decrease by 85% in case of using RWM with 50000 samples.
- We can interpolate the error model efficiently, as shown in Figure 7.12, because the error between the simulated and interpolated error model is very small (as in Figure 7.13).
- MLproxy decreases the computational cost compared with running the full simulation and modifies the distribution based on combining the proxy with RWM as shown in Figure 7.14.

In conclusion, the chapter proposed a new technique for modifying the proxy (emulator) distribution, using the multilevel concept. The new technique is faster and 'efficient' as running the full flow simulation.

In the next chapter, we propose a sampling technique for accelerating RWM based on Hamiltonian dynamics and Multilevel Monte Carlo method.

Chapter 8

Multilevel Hamiltonian Monte Carlo for Quantifying Uncertainty in Reservoir Simulation

In this chapter, we present a new way to accelerate the convergence of MCMC called Multilevel Hamiltonian Monte Carlo (MLHMC). MLHMC is a combination of the Multilevel Monte Carlo method discussed in Chapter 5 and the Hamiltonian Monte Carlo technique (HMC). The principal difference between Multilevel Markov Chain Monte Carlo (MLMCMC) and MLHMC is the proposal step. The proposal step in HMC avoids the random walk behaviour in MLMCMC and significantly reduces the computational cost by increasing the effective sample size.

This chapter compares the performance of Random Walk Metropolis (RWM) (see Chapter 6) and HMC. It also compares Multilevel Markov Chain Monte Carlo (see Chapter 6) and HMC. MLHMC has been implemented on a real field called Teal South to assess the uncertainty in the Field Oil Production Rate (FOPR). The following example explains the difficulty of using the RWM.

Example 8.0.0.1 *The target distribution is a multivariate Gaussian distribution (blue ellipse) with different standard deviations in each direction,* σ_0 *and* σ_1*. The proposal distribution is an isotropic Gaussian distribution (green circle) with Standard deviation* ϵ *as shown in Figure 8.1. The task is to sample from the target*

distribution using the proposal distribution, minimizing the rejection rate. We must keep $\epsilon = \mathcal{O}(\sigma_0)$ *and to explore the target distribution requires* $(\sigma_1/\sigma_0)^2$ *steps (Bishop, 2006)*

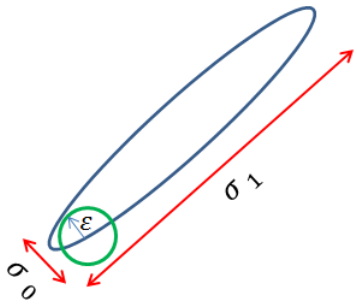


Figure 8.1: Example showing the difficulty of using RWM.

The movement through the parameter space using RWM is proportional to the square root of the number of steps (Gilks et al., 1996). For complex probability distributions, requires increasing numbers of steps, often to 10^6 or more samples (Oliver et al., 2008). In the next section, we explain one solution to the problem based on Hamiltonian dynamics, called Hamiltonian Monte Carlo (HMC). HMC uses gradient information to reduce the CPU time required to obtain convergence to the stationary distribution.

8.1 Hamiltonian Monte Carlo (HMC) Algorithm

Hybrid Monte Carlo (HMC), was developed by Duane in 1987 (Duane et al., 1987). Later, the name was changed to Hamiltonian Monte Carlo because it is based on Hamiltonian mechanics. HMC adds auxiliary momentum variables, which allow the use of Hamiltonian dynamics to propose a new location a long way from the current location.

HMC combines the benefits from Hamiltonian mechanics and the Metropolis-Hasting algorithm to be able to sample from complex distributions by generating a sequence of random samples from the posterior distribution to approximate the distribution.

HMC accelerates convergence and reduces the correlation between successive states compared with RWM (MacKay, 2002; Neal, 2011). There are several applica-

tions for this method in different disciplines, such as neural network models (Choo, 2000) and history matching for reservoir models (Mohamed et al., 2010a).

8.1.1 Hamiltonian Dynamics

HMC movement looks like rolling a ball along a surface. The main idea is to define the potential energy, physically "potential energy is the stored energy of position possessed by an object" $U(x)$, $x \in \mathbb{R}^d$ and the kinetic energy $K(u)$, $u \in \mathbb{R}^d$ as follows

$$U(x) = -\ln \pi(x|\mathcal{D}) \tag{8.1a}$$

$$K(u) = \frac{1}{2} u^T \mathbf{M}^{-1} u, \tag{8.1b}$$

where x are the spatial coordinates referring to the unknown parameters in the reservoir simulations, u is the momentum vector, $\pi(x|\mathcal{D})$ is the posterior distribution and $\mathbf{M}$ is a mass matrix, which is symmetric and positive definite. In most cases, $\mathbf{M}$ is a multiple of the identity matrix or a diagonal matrix.

The sum of the kinetic energy and the potential energy is the Hamiltonian (total energy) $H(x, u)$,

$$H(x, u) = U(x) + K(u),\ H(x, u) \in \mathbb{R}^{2d}. \tag{8.2}$$

Instead of sampling from $\pi(x|\mathcal{D})$ directly, we sample from the joint distribution $p(x, u)$,

$$\begin{aligned} p(x, u) &\propto \exp(-H(x, u)) \\ &= \exp(-U(x)) \exp(-K(u)) \\ &= \pi(x|\mathcal{D}) \exp(-u^T u/2) \\ &\propto \pi(x|\mathcal{D}) \mathcal{N}(0, \mathbf{M}). \end{aligned}$$

Since the distribution is separable, samples from p are samples from π. We take

$\mathbf{M} = \mathbf{I}$ in this thesis, therefore,

$$p(x, u) \propto \pi(x|\mathcal{D})\mathcal{N}(0, \mathbf{I}).$$

Each step in HMC consists of drawing a new pair of samples, (x, u), according to the joint distribution, $p(x, u)$. It starts from the current state x and randomizes the momenta u, from a Gaussian distribution, then simulates the dynamics based on the randomized momenta. The time evolution of this system is then governed by Hamilton's equations of motion,

$$\dot{x}_i = \frac{\partial H}{\partial u} = u_i, \qquad \dot{u_i} = -\frac{\partial H}{\partial x_i}, \qquad i = 1, 2, \cdots, d. \tag{8.3}$$

Equation (8.3) is Newton's second law of motion because $F = \dot{u}$ and as the force is conservative, $F = -\nabla U(x)$. If the dynamics are done accurately, the total energy is conserved (MacKay, 2002).

8.1.2 Leapfrog Method

Hamiltonian dynamics preserves the volume and the total energy and it is time reversible. If the dynamics are simulated exactly, the joint distribution $p(x, u)$ is invariant (Bishop, 2006). That is, if we start from $(x_0, u_0) \sim p$, then after the system evolves in time t, the new configuration at time t, (x_t, u_t) also follows the distribution p.

In reality nothing can be done perfectly – as we discretise Hamilton's equations with step-size δ, we observe that H is not conserved. Therefore, in practice the Hamiltonian dynamics are simulated by the leapfrog algorithm with a small step size δ to conserve the volume, conserves energy to $\triangle(\delta^2)$ (Neal, 1993) and preserve time reversibility. The leapfrog technique for discretizing Hamilton's equations (8.3)

with step size δ can be written as follows

$$u(t+\frac{\delta}{2}) = u(t) + \frac{\delta}{2}\dot{u} = u(t) - \frac{\delta}{2}\nabla U(x(t))$$
$$x(t+\delta) = x(t) + \delta\ \dot{x} = x(t) + \delta\, u(t+\frac{\delta}{2})$$
$$u(t+\delta) = u(t+\frac{\delta}{2}) + \frac{\delta}{2}\dot{u} = u(t+\frac{\delta}{2}) - \frac{\delta}{2}\nabla U(x(t+\delta)).$$

Figure 8.2 shows the leapfrog movement updates momentum every half step, then one step for position and a half step for the momentum.

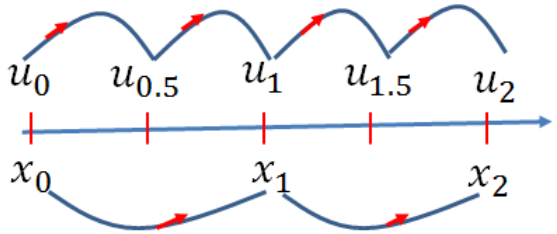


Figure 8.2: Leapfrog movement scheme.

The Euler method for discretizing Hamilton's equations (8.3) with time step δ can be written as follows

$$u(t+\delta) = u(t) + \delta\dot{u} = u(t) - \delta\,\nabla U(x(t))$$
$$x(t+\delta) = x(t) + \delta\ \dot{x} = x(t) + \delta\, u(t)$$

Figure 8.3 shows the difference between using the Euler method and the Leapfrog method for updating the position and the momentum and why we use the leapfrog method. The figure shows the Euler method produces a solution, which diverges to infinity, however the true solution is a circle. The leapfrog solution is very close to the true solution, which means leapfrog preserves volume exactly [recalculated from (Neal, 2011)].

Two parameters affect the performance of the HMC techniques: step size δ and the number of leap frog steps, τ. The choice of these parameters is related to the acceptance rate and the efficiency of the chain (Beskos et al., 2010, 2013). The choice of τ should be large enough to end up far from the initial state. If τ is too large, we increase the computational cost, while if it is too small, we have a high correlation

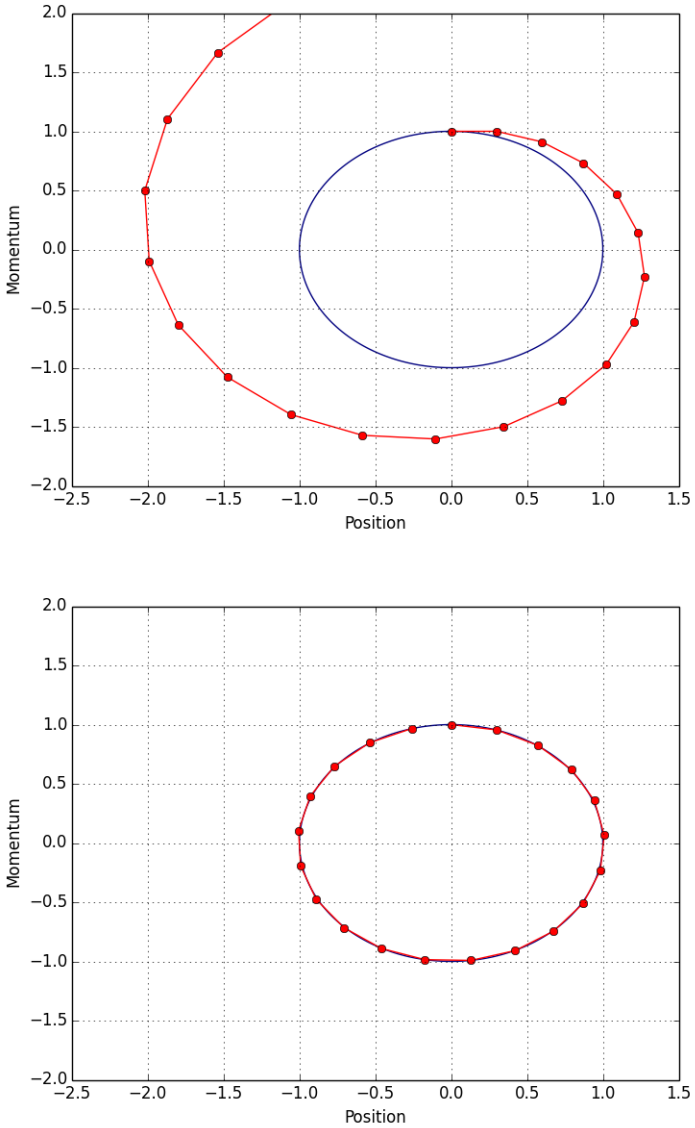


Figure 8.3: Hamiltonian dynamics when $H(x, u) = (x^2 + u^2)/2$ with initial state $(0, 1)$ and step-size $\delta = 0.3$ for 20 steps are shown for (top) the Euler method (bottom) the Leapfrog method along with the true path (blue curve).

between the states, which is undesirable. Balancing these effects by the choice of τ is a current topic of research problems. In the case of high dimensions, (Beskos et al., 2010, 2013) suggest the choice of the leapfrog step size to be $\delta = \mathcal{O}(d^{-0.25})$ where d is the dimension. Thus, the leapfrog length would be $\mathcal{O}(d^{0.25})$ to obtain an acceptance probability close to 0.651 (Beskos et al., 2013).

There are two kinds of errors obtained by applying the leapfrog technique–the local error obtained after one step δ, is of $\mathcal{O}(\delta)^3$ and the other, obtained after τ leapfrog steps, the global error, is of $\mathcal{O}(\delta)^2$ (Neal, 2011). The effect of using the leapfrog on the global error can be eliminated by using the Metropolis-Hasting

accept-reject step. The Euler method has local error of $\mathcal{O}(\delta)^2$ and global error of $\mathcal{O}(\delta)$. If the local error for (x, u) is no more than $\mathcal{O}(\delta^2)$, then the error for $H(x, u)$ will also be no more than $\mathcal{O}(\delta^2)$ (Leimkuhler and Reich, 2005).

The acceptance and rejection rule is based on the Metropolis-Hastings criterion for acceptance and rejection:

$$\alpha = \min\{1, \exp(H(x_{old}, u_{old}) - H(x_{new}, u_{new})\}, \tag{8.4}$$

HMC is asymptotically convergent to an invariant distribution if the HMC chain is ergodic. This depends on the choice of step-size and the number of leapfrog steps. For example, if we have complex, non-linear problems, we obtain periodicity in most cases. To eliminate this periodicity, we can use different step-sizes (Choo, 2000).

Overall, the importance of using the leapfrog method is to preserve the volume and maintain time reversibility, which is important for satisfying the detailed balance. To prove that HMC satisfies the detailed balance, see Appendix B.

Algorithm 9 describes HMC, we use a Gibbs sampler to generate the momentum randomly and then, update the position using leapfrog. This updating is based on the gradient information for the potential. The acceptance and rejection rule is based on the Metropolis-Hastings criterion for acceptance and rejection.

Algorithm 9: HMC Hajian (2007)

Initialize x_0, δ, τ and M_{samples}
for $i = 1$,**to** $M_{samples}$ **do**
  $u_0 \sim \mathcal{N}(0, 1)$
  $x_0 = x_{i-1}$;
  for $j = 1$, **to** τ **do**
    $u_{j-1/2} = u_{j-1} - \delta/2\, \nabla U(x_{j-1})$
    $x_j = x_{j-1} + \delta\, u_{j-1/2}$
    $u_j = u_{j-1/2} - \delta/2\, \nabla U(x_j)$
  end
  $x^* = x_\tau, u^* = u_\tau$
  $\alpha \sim \mathcal{U}(0, 1)$;
  if $(\alpha < \min\{1, e^{-(H(x^*,u^*)-H(x_0,u_0))}\})$ **then**
    $x_i = x^*$
  else
    $x_i = x_{i-1}$
  end
end

8.1.3 Relation between the Potential Energy and the Misfit Function

The misfit function represents the mismatch between the observed data and simulated data.

$$\text{Misfit} = \sum_{i=1}^{N} (q_{t_i}^{\text{sim}} - q_{t_i}^{\text{obs}})^2 / 2\sigma_i^2$$

Under the assumption the data measurement error is independent and Gaussian distributed at any time t. Therefore, the likelihood $\mathcal{L}(\mathcal{D}|x)$ can be defined by

$$\begin{aligned}\mathcal{L}(\mathcal{D}|x) &= \frac{1}{(2\pi)^{N/2} \prod_{i=1}^{N} \sigma_i} e^{-\sum_{i=1}^{N} (q_{t_i}^{\text{sim}} - q_{t_i}^{\text{obs}})^2 / 2\sigma_i^2} \\ &= \frac{1}{(2\pi)^{N/2} \prod_{i=1}^{N} \sigma_i} e^{\text{Misfit}}\end{aligned}$$

Using Bayes' theorem, which links posterior distribution $\pi(x|\mathcal{D})$ with the prior density and the likelihood.

$$\pi(x|\mathcal{D}) = \frac{\mathcal{L}(\mathcal{D}|x)\,\text{prior}}{\text{Normalization}}$$

Equation (8.1a) can be written as follows

$$U(x) = \log(\text{Normalization constant}) - \log(\mathcal{L}(\mathcal{D}|x)) - \log(\text{prior}) \tag{8.5}$$

$$= \log(\text{Normalization constant}) + Misfit + \frac{N}{2}\log(2\pi) + \sum_{i=1}^{N} \log(\sigma_i) - \log(\text{prior}) \tag{8.6}$$

The gradient for the potential energy

$$\nabla U(x) = \nabla(Misfit) - \nabla(\log(\text{prior})). \tag{8.7}$$

8.1.4 Advantages and Disadvantages of HMC over MCMC

The advantages and disadvantages of HMC (Duane et al., 1987; Beskos et al., 2013, 2010; Hajian, 2007; Choo, 2000) can be summarised as follows

- HMC resolves some inefficiencies of the traditional MCMC methods by avoiding the random walk. It proposes moving across the sample space in larger steps, therefore the samples are less correlated.
- HMC is applicable for continuous distributions and requires calculation of the gradient for the logarithm of the probability.
- HMC is well suited for sampling from non-Gaussian and curved distributions.
- HMC accelerates convergence and increases the acceptance rate.
- In a high dimensional problem, HMC is a good method to use.
- The method is very simple to code, as we can see in Algorithm 9. However, it requires tuning the parameters, τ, δ and calculating the gradient $\bigtriangledown U$, $M\tau$ times, where M is the number of samples, which increases the computational cost.

Example 8.1.4.1 *Consider a* 6*-dimensional Gaussian distribution [the example is recomputed from (Hajian, 2007)]*

$$H(x) = \frac{1}{2}\mathbf{x}^T\mathbf{x}$$

Estimate the distribution $H(x)$ *using RWM and HMC.*

The RWM algorithm uses a Gaussian proposal distribution with a step-size $\triangle \approx 2.4/\sqrt{6}$*. However, HMC requires the gradient which equals to* $\mathbf{x}$*.*

In Figure 8.4, we compare RWM and HMC Algorithms 5 and 9. Figure 8.4 shows that, the chain created by using HMC method is convergent, but for the RWM algorithm, it requires more samples to reach convergence. Also, the acceptance rate for the RWM algorithm is 21% *and for the HMC is* 85%*.*

Example 8.1.4.2 *Estimating the target distribution (8.8) using RWM and HMC algorithms [the example is recomputed from (Kaipio and Somersalo, 2005)].*

$$\pi(x, y) \propto e^{-10(x^2-y^2)^2-(y-0.25)^4} \tag{8.8}$$

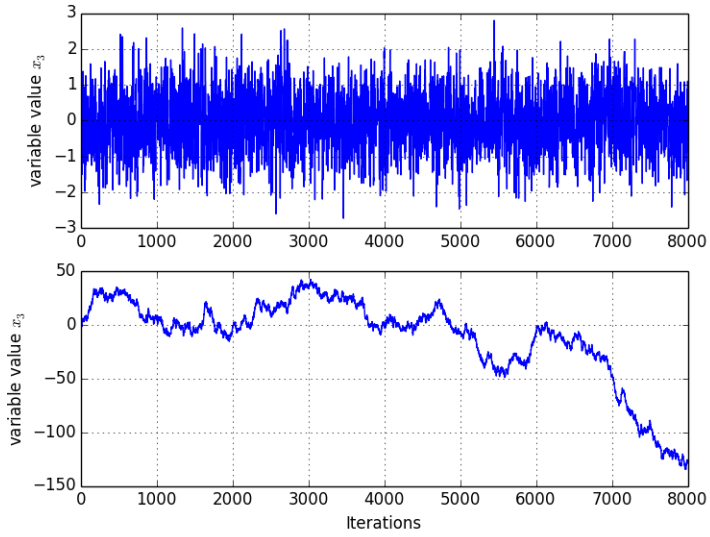


Figure 8.4: 8000 samples drawn from an isotropic six-dimensional Gaussian distribution using (top) the HMC method with $\tau = 100$ and $\delta = 0.01$ (bottom) the RWM.

Figure 8.5 shows that the HMC is better than the RWM for exploring the whole distribution. The acceptance probability for RWM is 16% *and for HMC is* 85%.

In order to use the HMC to quantify the uncertainty of history matching and forecasting, we must calculate the gradient of the potential function. This means calculating the gradient of the misfit function. If the gradient of the solution with respect to the uncertain parameters is not available, one way of calculating the gradient is to use a proxy model to estimate the misfit surface and then find the gradient. However, this way is less accurate. In other words, in most practical cases, we cannot derive the gradient in a closed form because we do not have access to internal code to find the gradients via adjoint computations (in particular when using a black-box to compute the forward problem). Therefore, we have to estimate the misfit surface using experimental design within an emulator (Mohamed et al., 2010a). In the following section, we discuss how to build the proxy using interpolation techniques.

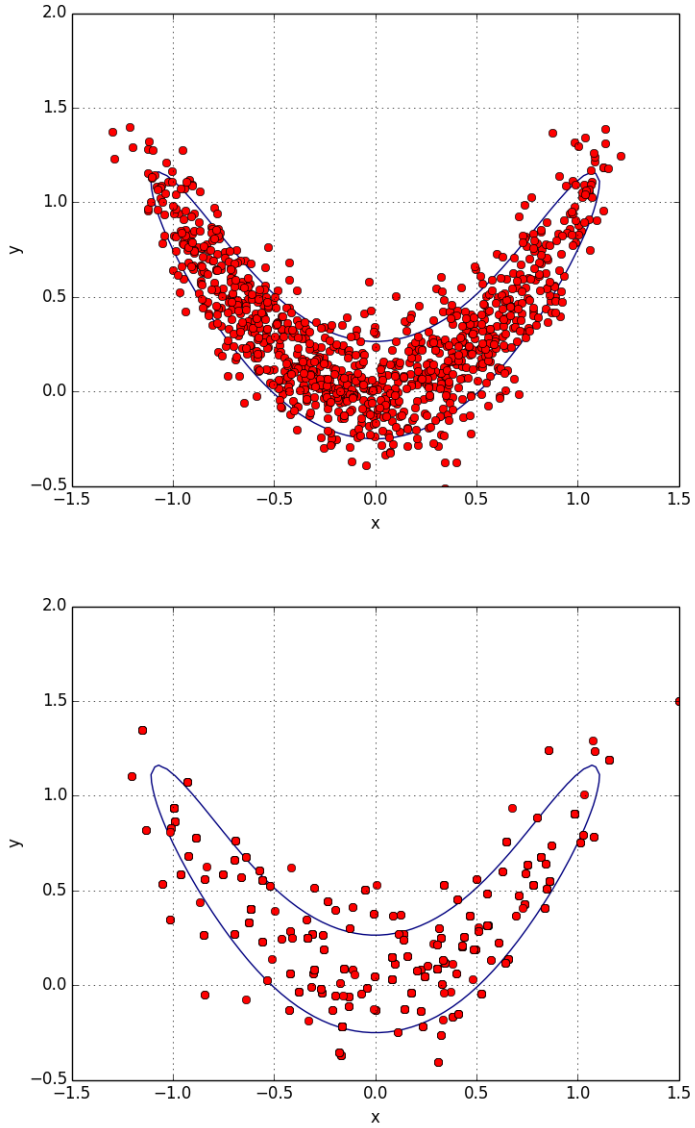


Figure 8.5: 1000 samples drawn from (8.8) using (top) the HMC method with $\tau = 40$ and $\delta = 0.01$ (bottom) the RWM along with the true contour ($\pi(x, y) = constant$, blue curve).

8.2 Nadaraya-Watson Kernel Regression

The Nadaraya-Watson kernel was proposed by (Nadaraya, 1964; Watson, 1964). These kernels are usually probability density functions. We can define a map from the input parameters to the quantity of interest (output) as follows

$$M = f(m) + \epsilon, \tag{8.9}$$

where m is the input parameters, f is a smooth function, ϵ is the noise and M is the output. The aim is to construct the function that describes the surface of the

output M. One way is using Nadaraya-Watson kernel regression as follows,

$$\widehat{M}(m) = \sum_{i=1}^{n} M_i \frac{K(m, m_i)}{\sum_{i=1}^{n} K(m, m_i)} = \frac{R}{S}, \tag{8.10}$$

where M_i is the weights and $\widehat{M}$ is the estimator. There are different types of kernel, e.g. Gaussian and polynomial.

8.2.1 Gaussian Kernel

The Gaussian kernel is a special case of Radial Basis Function (RBF) (discussed in Section 7.1.5) and can be defined as follows

$$K(m, m_i) = e^{-(m-m_i)^2/2\sigma^2}, \tag{8.11}$$

where σ is the bandwidth, which describes the kernel width. The choice of σ is difficult as, it controls the smoothness of the estimator. If σ is very small, the kernel estimator is very noisy as we have the sum of delta functions around each data point. Also, if σ is very big, then we lose the useful structure for the kernel as we have the sum of constant functions. There are different ways to obtain σ, e.g. cross validation. Procedures for obtaining σ are as follows,

- Divide the data into three parts: training 60%; cross validation 20% and testing 20%.
- Train with different bandwidths σ_j and for every σ_j calculate $\widehat{M}_{\text{CVj}}$ (8.10), evaluate the choice by calculating the cross validation error

$$\text{CV}_{\text{error}} = \min_{\forall j} \left\{ \sqrt{\frac{1}{n} \sum_{i=1}^{n} (\widehat{M}_{\text{CVj}} - M_i)^2} \right\}. \tag{8.12}$$

- Choose σ_{opt} based on the minimum cross validation error (8.12).

The drawback of these kernel methods is that they suffer from the curse of dimensionality (Demyanov et al., 2010).

8.2.1.1 Calculate Gradient of Misfit

To find the gradient of (8.10), we do the following

$$\bigtriangledown(\widehat{M}(m)) = \frac{S \bigtriangledown R - R \bigtriangledown S}{S^2}$$

By differentiating the numerator and denominator of (8.10),

$$\bigtriangledown R = \sum_{i=1}^{n} M_i \frac{m_i - m}{\sigma^2} e^{-(m-m_i)^2/2\sigma^2}$$
$$\bigtriangledown S = \sum_{i=1}^{n} \frac{m_i - m}{\sigma^2} e^{-(m-m_i)^2/2\sigma^2}.$$

By using the calculation for the gradient of R and S we obtain

$$\bigtriangledown(\widehat{M}(m)) = \frac{\widehat{M}(m)}{\sigma^2} \sum_{i=1}^{n} (m_i - m) e^{-(m-m_i)^2/2\sigma^2} (\frac{M_i}{R} - \frac{1}{S}).$$

8.2.2 Polynomial Kernel

The polynomial kernel can defined as follows

$$K(m, m_i) = (m\, m_i + 1)^d \tag{8.13}$$

where d is the polynomial degree. To find the gradient, we apply the same steps as in the Gaussian case, to obtain

$$\bigtriangledown(\widehat{M}(m)) = \widehat{M}(m) \sum_{i=1}^{n} m_i (m_i\, m + 1)^{d-1} (\frac{M_i}{R} - \frac{1}{S}).$$

We use Latin Hypercube Sampling (LHS) or a Sobol sequence (described in Chapter 7) to generate the initial samples for estimating the misfit surface in our studies.

Figure 8.6 shows that the emulator is good enough to use for estimating the misfit using RBF with different kernel types. The figure shows that the Gaussian kernel is better than a polynomial kernel of degree 3.

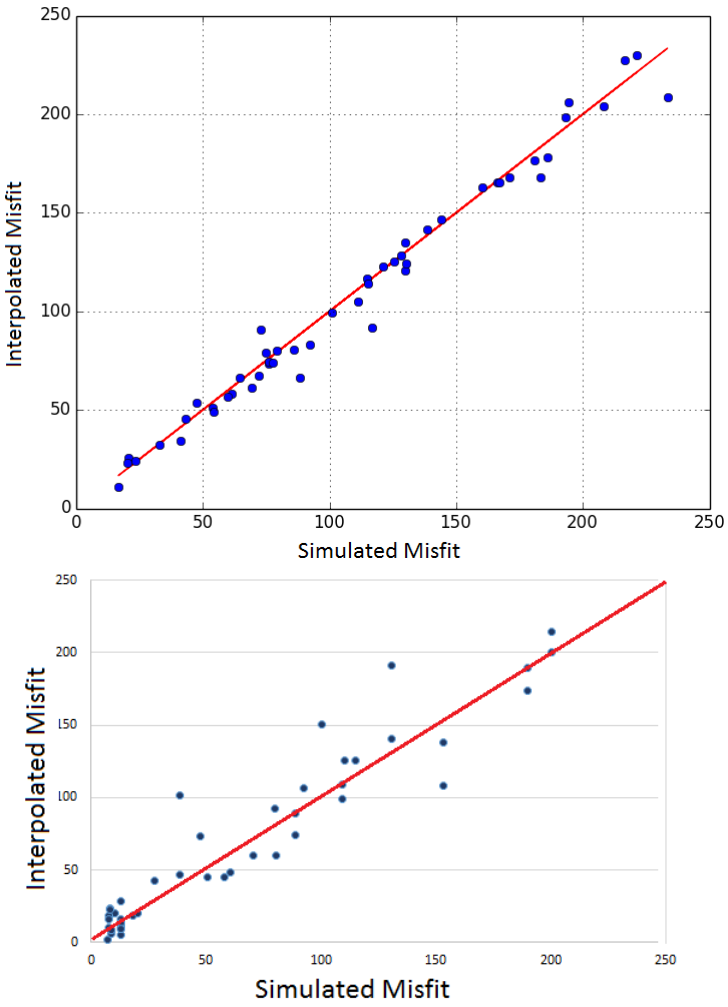


Figure 8.6: Simulated misfit versus interpolated misfit for 50 samples using, (top) Gaussian Kernel (bottom) Polynomial kernel.

Figure 8.7 shows the solution of the Teal South example, described in Chapter 6, using HMC combined with the Gaussian and the polynomial kernels of degree 3 for estimating the misfit surface. It shows how the choice of kernels affects the efficiency of the solution. The choice of kernels may change from one problem to another. This figure shows that the Gaussian kernel is better than the polynomial kernel in the Teal South case study. In Figure 8.7, the Bayesian credible interval obtained using Gaussian kernel includes more of the data than the polynomial kernel.

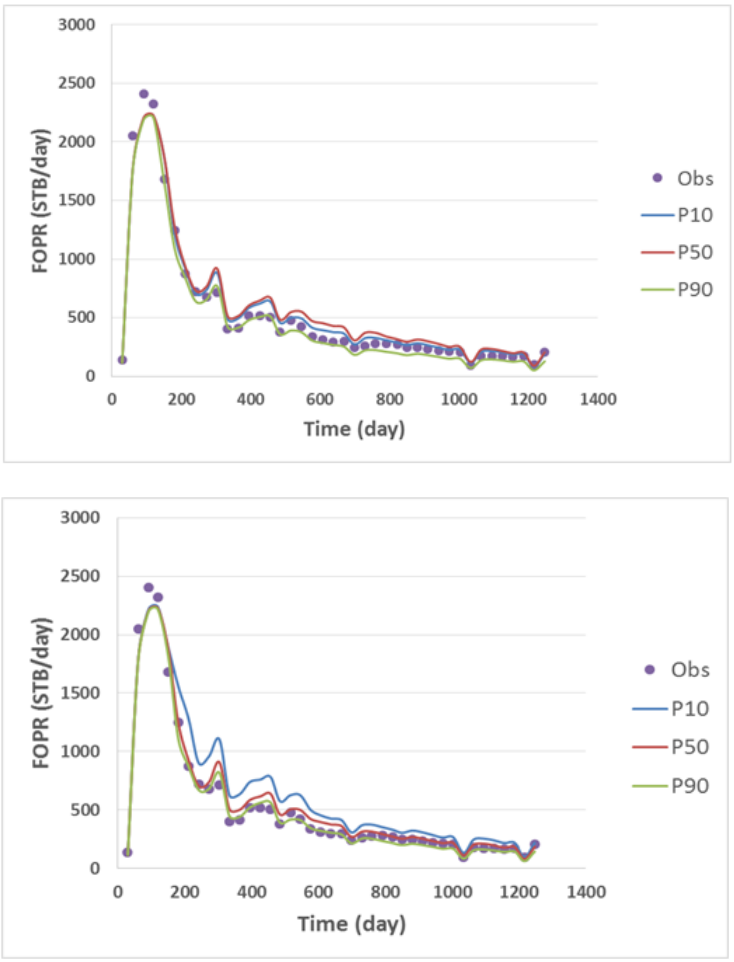


Figure 8.7: Bayesian credible intervals $P10 - P50 - P90$ using HMC with (top) the Gaussian kernel (bottom) the Polynomial kernel with degree 3, the initial number of samples to construct the kernel is 100, $\tau = 25$, the number of samples is 1250 and we use the first 181 days for the history period.

8.3 Comparison between MLMCMC and HMC

In this section, we compare Bayesian credible intervals $P10-P50-P90$ and the CDF between the HMC and MLMCMC method for a fixed computational cost created by running MLMCMC and HMC 3 times and finding the average result for each of them.

Figure 8.8 shows that most of the observed data for HMC are outside the uncertainty range; however, for MLMCMC, the observed data are inside the range as shown in Chapter 6, Figure 6.5. This means that the HMC requires more samples to achieve a distribution as good as that produced by the MLMCMC method.

Figure 8.9 shows that the CDF for the MLMCMC is better than for the HMC

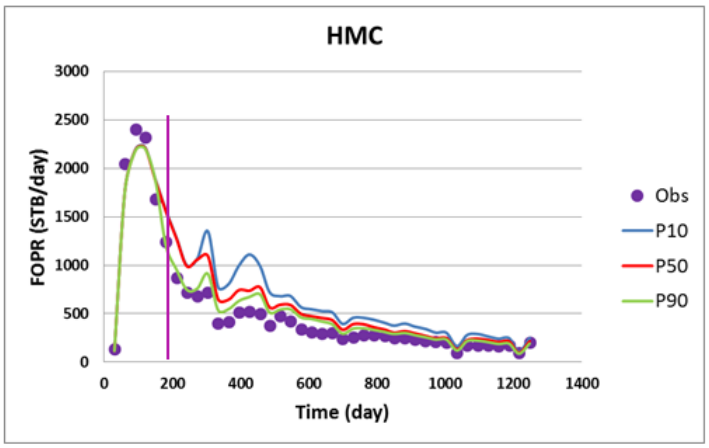


Figure 8.8: Bayesian credible intervals $P10 - P50 - P90$ using the HMC with the same setup as Chapter 6, Figure 6.5. The vertical line is the end of the history period.

for 1187 days. At day 181, the distributions are close to each others based on Kolmogorov–Smirnov test (Jensen et al., 2000).

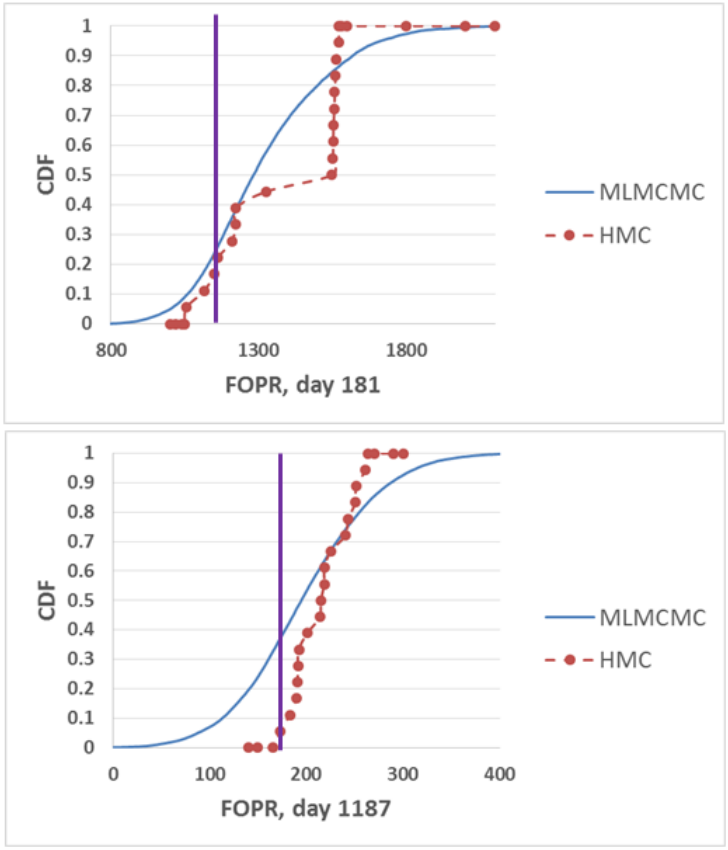


Figure 8.9: Cumulative distributions from the HMC and the MLMCMC methods with Sobol sequence at (top) day 181, (bottom) day 1187. The vertical line is the observed data. Using the number of the initial samples is 30 for LHS, $\tau = 15$ and 867 as the number of samples.

8.4 Multilevel Hamiltonian Monte Carlo (MLHMC)

We proposed an efficient solution to accelerate convergence for MCMC based on a combination of Hamiltonian Monte Carlo and Multilevel Monte Carlo, called Multilevel Hamiltonian Monte Carlo. MLHMC has the advantage of HMC for accelerating the convergence rate by using Hamiltonian dynamics and the advantage of Multilevel Monte Carlo (MLMC) for avoiding the estimation of the quantities of interest directly on a fine grid, but instead estimates the correction with respect to the next lower level. Thus, we obtain samples from hierarchical posteriors corresponding to multilevel approximations.

To show the MLHMC chain satisfies time reversibility, we should prove the detailed balance condition (8.14). From this, we can deduce that the target distribution $p^l(x,u)$ (x is the position, u is the momentum and l is level) is the stationary distribution. Based on the definitions of the acceptance rate and the transition probability, $\mathbb{P}^l(x \to y)$, the detailed balance condition is

$$\mathbb{P}^l(x \to y)p^l(x,u) = \mathbb{P}^l(y \to x)p^l(y,u') \tag{8.14}$$

We will prove that the detailed balance condition (8.14) satisfies the MLHMC chain. For the initial level l_0, it is as in HMC (Bishop, 2006). We focus to prove the case of $l > l_0$.

Lemma 8.4.0.1 *Assuming the acceptance probability from state x with momentum u to state y with momentum u' for level l is $\alpha^l(y|x) = \min\{1, \frac{e^{(H^l(x,u)-H^l(y,u'))}}{e^{(H^{l-1}(x,u)-H^{l-1}(y,u'))}}\}$ and the target distribution is $p^l(x,u) \propto e^{-H^l(x,u)}$ and the transition probability is $\mathbb{P}^l(x \to y) = \alpha^l(y|x)\, p^{l-1}(y,u')$, ($H^l(x,u)$ is the total energy w. r. t level l). Prove the detailed balance condition (8.14).*

Proof 8.4.0.2 *To prove the detailed balance condition, it is equivalent to prove*

$$\frac{\mathbb{P}^l(x \to y)p^l(x,u)}{\mathbb{P}^l(y \to x)p^l(y,u')} = 1.$$

$$
\begin{aligned}
LHS: \frac{\mathbb{P}^l(x \to y)p^l(x,u)}{\mathbb{P}^l(y \to x)p^l(y,u')} &= \frac{\min\{1, \frac{e^{(H^l(x,u)-H^l(y,u'))}}{e^{(H^{l-1}(x,u)-H^{l-1}(y,u'))}}\}e^{-H^{l-1}(y,u')-H^l(x,u)}}{\min\{1, \frac{e^{(H^l(y,u')-H^l(x,u))}}{e^{(H^{l-1}(y,u')-H^{l-1}(x,u))}}\}e^{-H^{l-1}(x,u)-H^l(y,u'))}} \\
&= \frac{\min\{1, e^{(-H^{l-1}(x,u)-H^l(y,u'))}\}}{\min\{1, e^{(-H^{l-1}(y,u')-H^l(x,u)}\}} = 1.
\end{aligned}
$$

□

The condition for controlling the number of samples required is based on minimizing the variance of the estimator:

$$\text{control}(\mathbb{V}(\mathbb{E}(U_L(x)))) \leq \epsilon^2 \, V_0/2, \tag{8.15}$$

where the quantity of interest is U, V_0 is a constant used to nondimensionalize the equation (the variance of the observed data), $x = (x_1, x_2, \cdots, x_d)$ are the unknown parameters and ϵ is the relative accuracy. Algorithm 10 describes a Two-level HMC. In Algorithm 10, $\mathbb{P}(x)$ refers to prior density and the superscripts f and c refer to fine and coarse grids.

Algorithm 10: Two-level HMC

Fix a sequence of grid resolutions $l = l_0, L$, N_0 initial samples for estimating the misfit surface, a number of warm-up samples M_{up}, fix the variance of the observed data V_0, the accuracy ϵ, the time step for leapfrog δ and the leapfrog length τ.
Generate N_0 samples based on LHS or Sobol sequence or any other experimental design.
Use the Nadaraya-Watson kernel to estimate the surface of the misfit.
if $l = l_0$ **then**
  Use HMC Algorithm 9, with $M_{samples} = M_{up}$
  Check condition (8.15). If it is satisfied, go to the second level. Otherwise, add more samples.
else
  Initialized $x_0^c \sim \mathbb{P}(x)$ and $x_0^f = x_0^c$
  while $(M_{up} > 0)$ **do**
    for $i = 1$,**to** M_{up} **do**
      $u_0 \sim \mathcal{N}(0, 1)$
      $x_0 = x_{i-1}^c$
      for $j = 1$, **to** τ **do**
        $u_{j-1/2} = u_{j-1} - \delta/2\, \nabla U(x_{j-1})$
        $x_j = x_{j-1} + \delta\, u_{j-1/2}$
        $u_j = u_{j-1/2} - \delta/2\, \nabla U(x_j)$
      end
      $x^f = x^c = x_\tau, u^* = u_\tau$
      $\alpha \sim \mathcal{U}(0, 1)$
      if $(\alpha < \min\{1, e^{-(H^c(x_\tau, u^*) - H^c(x_0, u_0))}\})$ **then**
        $\alpha_1 \sim \mathcal{U}(0, 1)$
        if $(\alpha_1 < \min\left\{1, \frac{e^{-(H^f(x_\tau, u^*) - H^f(x_0, u_0))}}{e^{-(H^c(x_\tau, u^*) - H^c(x_0, u_0))}}\right\}$ **then**
          $x_i^f = x^f$
        else
          $x_i^f = x_{i-1}^f$
        end
        $x_i^c = x^c$
      else
        $x_i^c = x_{i-1}^c$
      end
    end
    Check condition (8.15) and update the required number of samples, M_l
    $M_{up} = M_l - M_{up}$
  end
end

The algorithm is straightforward to generalise to l levels as shown by the flowchart, see Figure 8.10. However, this requires estimating the misfit surface for $l - 1$ levels. In Figure 8.10, check the condition refers to (8.12). In the Teal South reservoir field, we use two levels as a proof of concept.

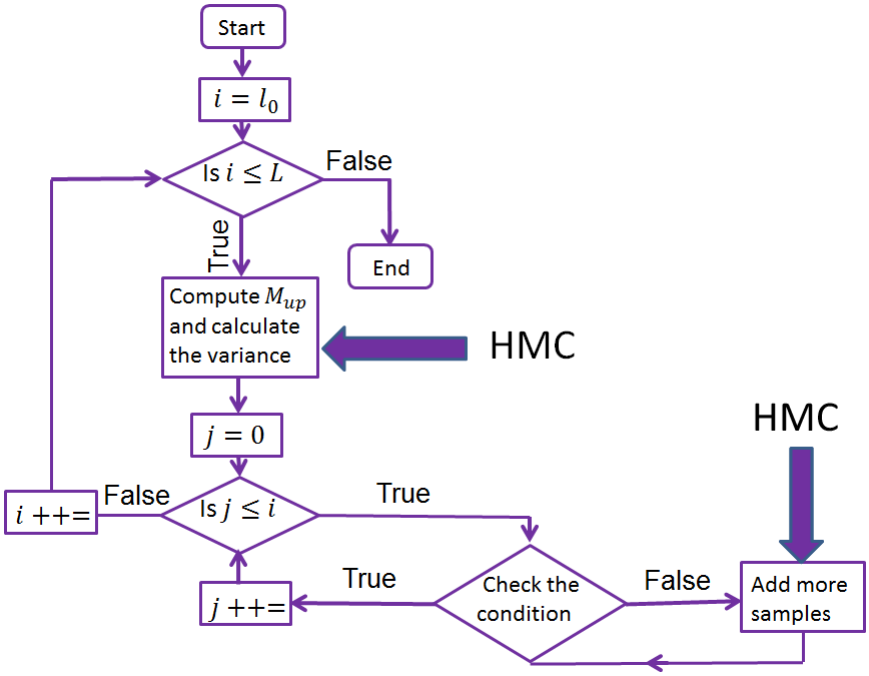


Figure 8.10: Flowchart for MLHMC Algorithm.

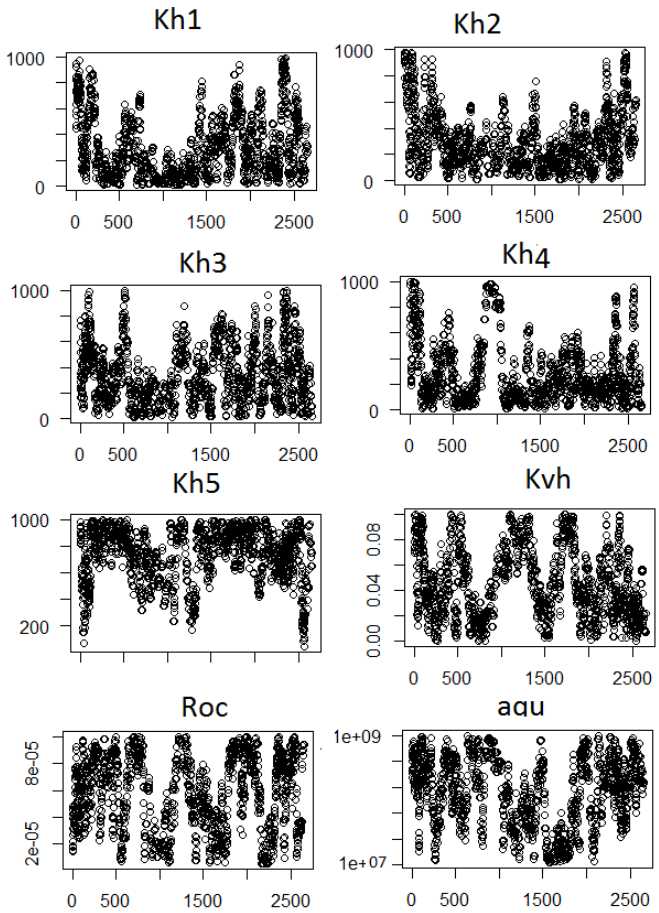


Figure 8.11: Teal South history match parameters using MLHMC with $V_0 = 10000$, $\epsilon = 0.01$, grid $11 \times 11 \times 5$, and $11 \times 11 \times 25$, burning in period of 100 samples for each level, $\tau \sim \mathcal{U}(10, 25)$.

Figure 8.11 shows the chain for each parameter using MLHMC. The step size δ is related to the standard deviation of the parameters and we use Sobol sequence and the initial number of samples used to construct the kernel is $N_0 = 30$. The figure shows that the chains are convergent.

Figure 8.12 shows the autocorrelation (6.4) of MLHMC per parameter, based on the effective samples, meaning the independent samples.

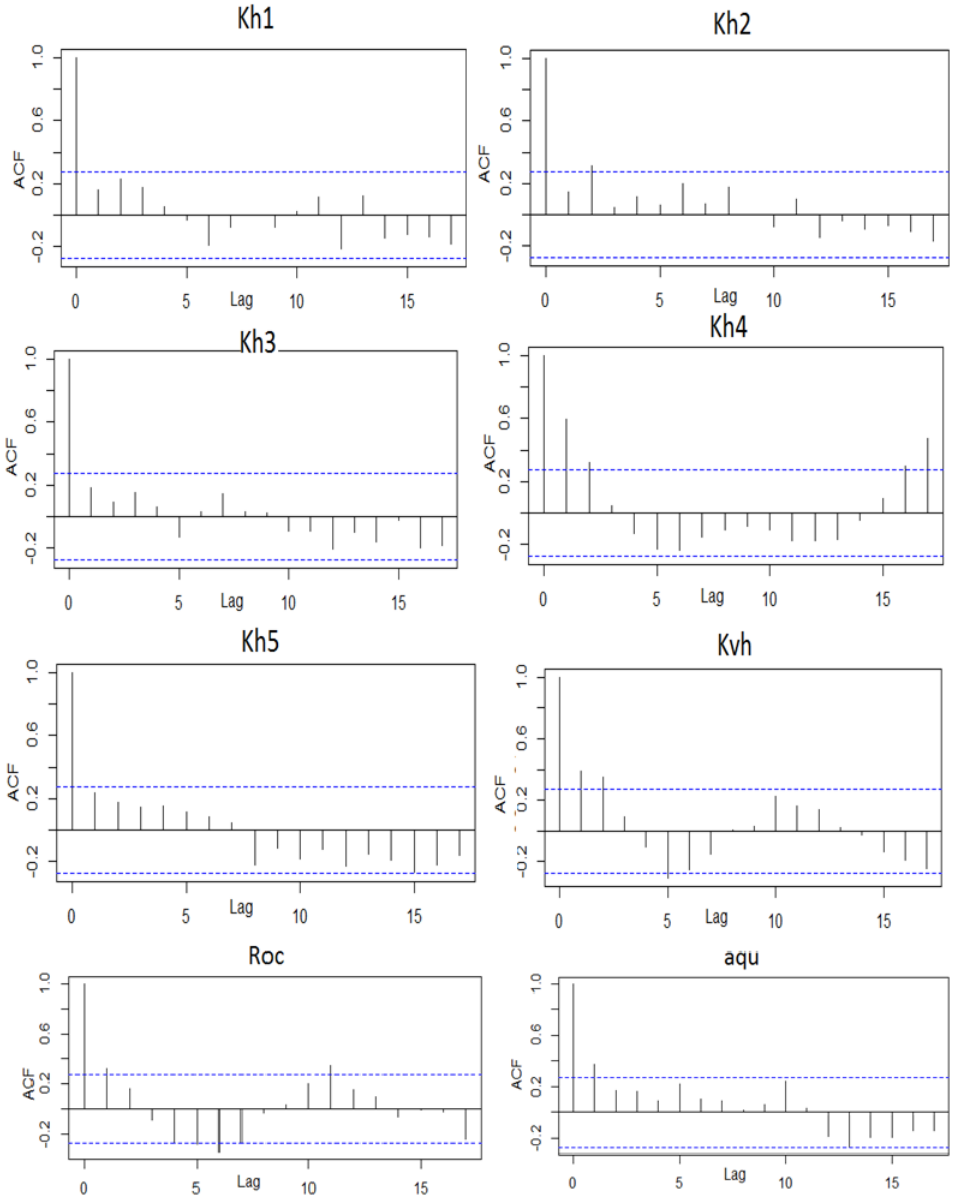


Figure 8.12: Autocorrelation of MLHMC for each of the 8 unknown parameters.

Figure 8.13 shows that the histogram for the unknown parameters using MLHMC is close to the histogram created by using MLMCMC Figure 6.4.

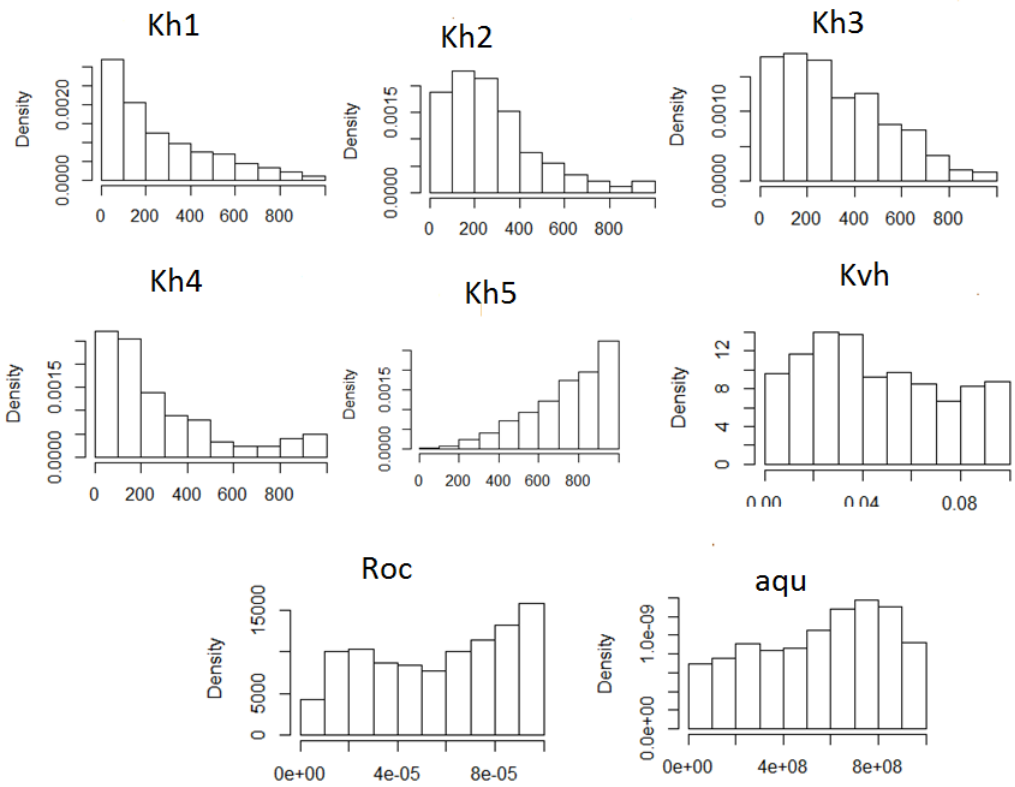


Figure 8.13: Histogram for the unknown parameters using MLHMC.

Figure 8.14 shows that all the observed data lies inside the Bayesian credible intervals $P10-P50-P90$ using MLHMC, however for MLMCMC Figure 6.5, some of the observed data outside the range.

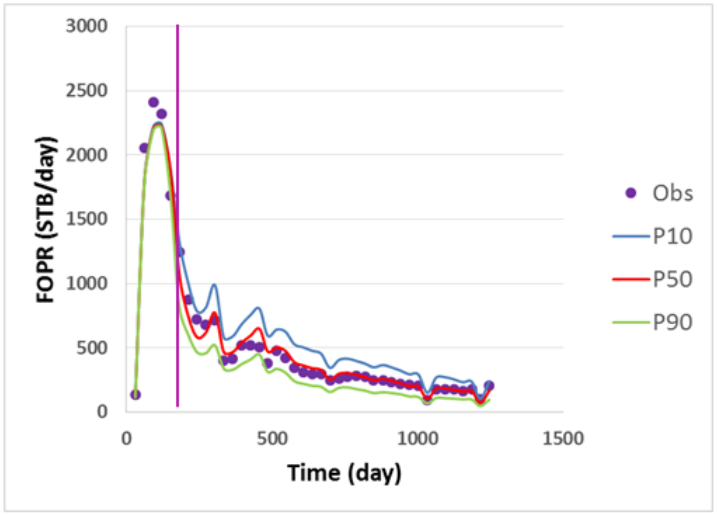


Figure 8.14: Bayesian credible intervals $P10-P50-P90$ using MLHMC with Sobol sequence, the vertical line represents the end of the history matching period.

Figures 8.15 and 8.16 compare 4 techniques: RWM, HMC, MLHMC with Sobol

sequence and MLHMC with LHS. The figures show that MLHMC with Sobol sequence is the fastest algorithm.

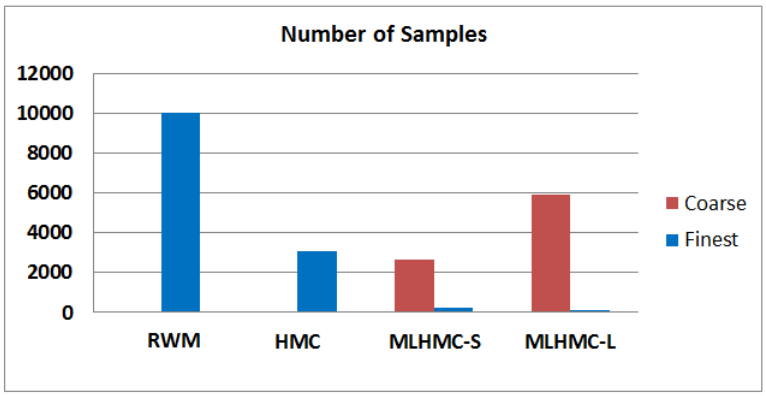


Figure 8.15: Comparison between RWM, HMC, MLHMC with Sobol sequence and LHS w.r.t number of samples with accuracy, $\epsilon = 0.01$ for all techniques and 100 is the initial samples for LHS and Sobol sequence.

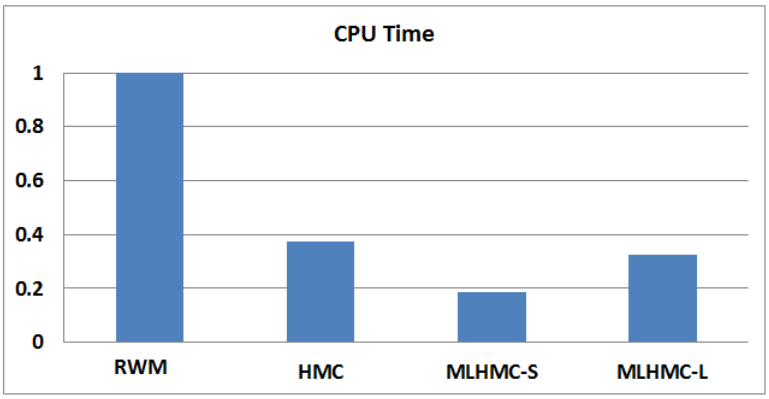


Figure 8.16: Comparison between RWM, HMC, MLHMC with Sobol sequence and LHS w.r.t normalized CPU time with accuracy, $\epsilon = 0.01$ for all techniques and 100 is the initial sample for LHS and Sobol sequence.

Figure 8.17 compares the CDF of HMC and MLHMC with Sobol sequence and LHS at the end of the history matching period, day 181, and at day 1187 in the forecast period. The result is the average of 3 runs. It is clear that Sobol sequence are better than LHS in history matching and forecasting periods. MLHMC with Sobol sequence obtains the same CDF as HMC on the finest grid within the sampling error. However, with LHS, more samples are probably required to explore the parameter space and we do not have the same agreement as with Sobol sequence.

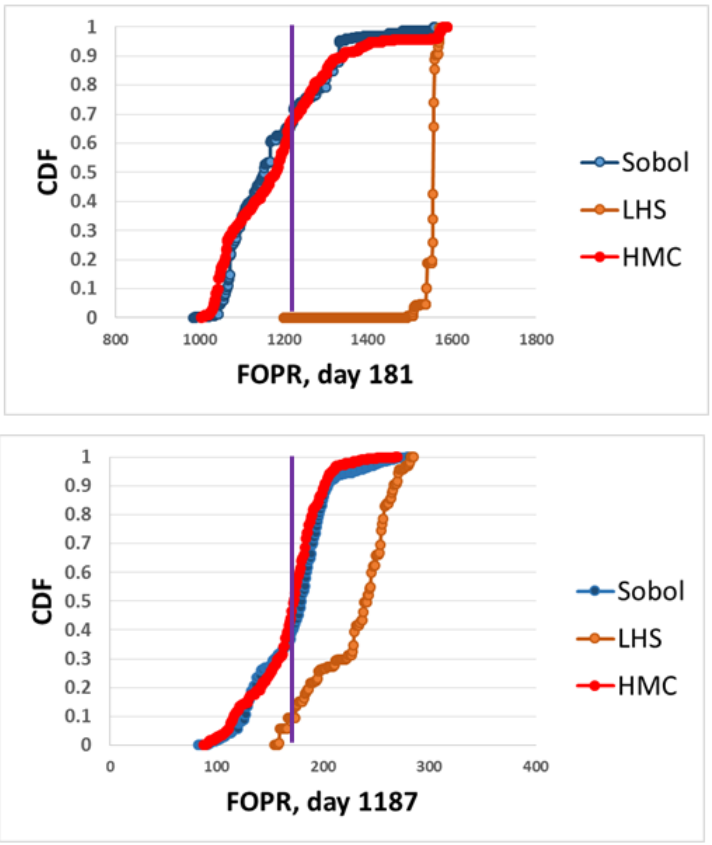


Figure 8.17: Cumulative distributions from HMC and MLHMC with Sobol sequence and LHS at (top) day 181, (bottom) day 1187. The vertical line is the observed data. Using $V_0 = 10000$, $\epsilon = 0.02$, grid $11 \times 11 \times 5$, and $11 \times 11 \times 25$, burning in period of 100 samples for each level, $\tau \sim \mathcal{U}(10, 25)$, the step size δ is related to the standard deviation of the parameters and we use Sobol sequence with the initial number of samples to construct the kernel is $N_0 = 30$. For HMC, $\tau \sim \mathcal{U}(10, 25)$, and the initial number of samples to construct the kernel is $N_0 = 30$ for Sobol sequence and the number of samples is 1350.

Figure 8.18 shows that if we increase the amount of historical data, then the relative uncertainty decreases.

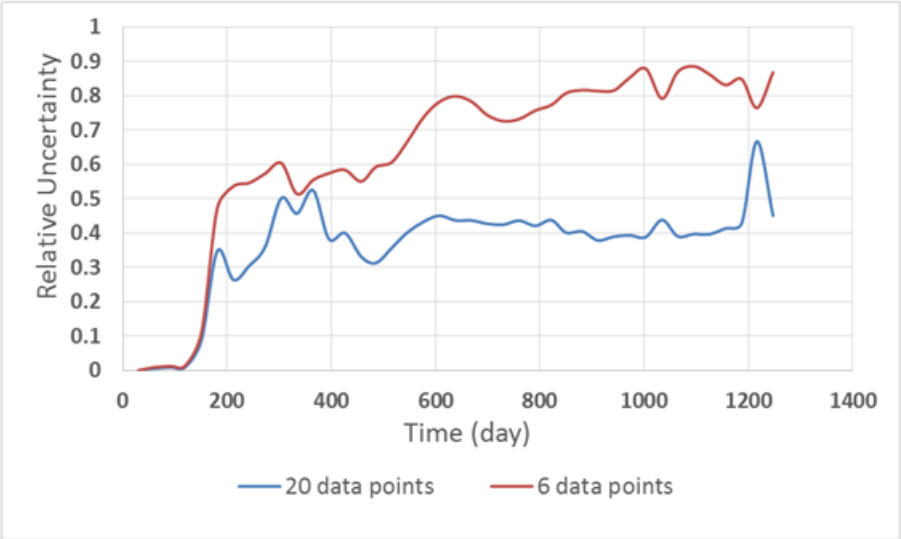


Figure 8.18: Relative uncertainty using MLHMC with 6 and 20 data points for historical data.

Figure 8.19 shows that MLHMC combined with Sobol sequence is faster than RWM for different accuracies.

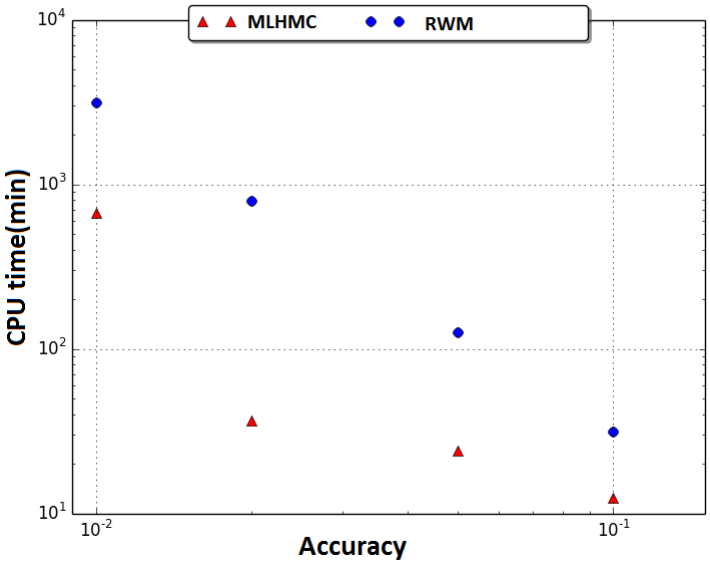


Figure 8.19: Comparison of CPU time of RWM and MLHMC with Sobol sequence.

8.5 Summary

The goals of this chapter were to propose a new sampling algorithm MLHMC for reservoir simulation history matching and uncertainty quantification, based on the combination of MLMC and HMC to accelerate the convergence of RWM. The chap-

ter demonstrated the performance of MLHMC on Teal South field.

The chapter showed the following:

- HMC can be better than RWM because the chain created using HMC method is convergent unlike the RWM one and required more samples to reach convergence as shown in Figure 8.4.
- The quality of interpolated misfit affected using different kernel types, Gaussian kernel is better than a polynomial kernel of degree 3 as shown in Figure 8.6.
- the choice of kernels effects on the efficiency of the Bayesian credible interval obtained, using Gaussian kernel is more confident than the polynomial kernel because the uncertainty distribution having all the observed data as shown in Figure 8.7.
- For the same computational cost, MLMCMC is better than HMC because MLMCMC is able to estimate the distribution of FOPR, but HMC requires more samples to achieve the same result as MLMCMC shown in Figure 8.8.
- The detailed balance condition is satisfied for MLHMC.
- MLHMC resolved some of the inefficiencies of the traditional MCMC methods by avoiding the random walk, which improves the acceptance rate. Therefore, shorter chains will be needed for a reliable parameter estimation, compared with a traditional MCMC chain, to yield the same performance.
- The histogram for the uncertain parameters using MLHMC as shown in Figure 8.13 is close to the histogram created by using MLMCMC Figure 6.4.
- For the same accuracy, Bayesian credible interval for estimating the uncertainty distribution using MLHMC has all the observed data inside as shown in Figure 8.14, however, for MLMCMC Figure 6.5, some of the observed data outside the range.
- MLHMC with Sobol sequence is the fastest algorithm as shown in Figure 8.16.

- For different accuracies, MLHMC combined with Sobol sequence is faster than RWM as shown in Figure 8.19.

- MLHMC with Sobol sequence obtained the same CDF as HMC on the finest grid within the sampling error as shown in Figure 8.17.

In conclusion, MLHMC obtains a good performance for studying Teal South field compared with RWM, HMC and MLMCMC. It accelerates the convergence of RWM, HMC and MLMCMC. However, the number of the parameters which require tuning increases because we have to tune both HMC parameters and MLMC parameters. Improved methods of tuning are an open area of research.

Chapter 9

Conclusion and Future Work

This chapter summarises the key results, main contributions and suggests recommendations for future research.

The main aim of the thesis is to improve the speed of uncertainty quantification using multilevel techniques. We want to have a confident prediction about the quantity of interest, such as oil rate, within a limited simulation time. To obtain a prediction with high confidence requires an estimate of the distribution of the quantity of interest. Estimating the distribution using a high resolution model, means a large computational time. There is a trade off between the solution accuracy and the speed.

One a widespread approach for estimating the posterior distribution is based on Markov Chain Monte Carlo (MCMC) method within a Bayesian framework. The most common MCMC technique is the Random Walk Metropolis (RWM). RWM has a high computational cost because it requires a huge number of simulations to run and about $45\% - 70\%$ of the samples will be rejected because of the acceptance rate for generating samples using RWM is between $30\% - 55\%$ (Gilks et al., 1996). Since the rejected samples do not contribute into the inference, the time for running the rejected samples is wasted.

In the thesis, I developed and investigated sampling techniques such as Multilevel Markov Chain Monte Carlo, Multilevel Hamiltonian Monte Carlo and Multilevel proxy to improve the prediction: to make them more reliable and faster to obtain.

9.1 Key Findings

The thesis succeeded in showing the Multilevel Monte Carlo (MLMC) can be applied for solving hyperbolic and parabolic, linear and nonlinear stochastic differential equations in porous media. The thesis explored the feasibility of the Multilevel Markov Chain Monte Carlo (MLMCMC) technique in reservoir simulation for uncertainty quantification.

The thesis proposes two new sampling techniques for estimating the uncertainty distribution based on the multilevel concept: Multilevel Hamiltonian Monte Carlo (MLHMC, Chapter 8) and Multilevel proxy (MLproxy, Chapter 7). MLHMC is a variance reduction technique based on the combination of the MLMC and the Hamiltonian Monte Carlo (HMC). It has the advantage of the HMC for accelerating the convergence rate using Hamiltonian dynamics and the advantage of the MLMC for avoiding the estimation of the quantities of interest directly on a fine grid, but instead estimates the correction with respect to the next lower level. Thus, we obtain samples from hierarchical posteriors corresponding to multilevel approximations. The MLHMC resolved some inefficiencies of the traditional MCMC methods by avoiding the random walk, which improved the acceptance rate. Therefore, shorter chains are needed for reliable parameter estimation, compared with a traditional MCMC chain, to yield the same computational cost. MLproxy is coupling the multilevel concept and proxy ideas. It is improving the quality of the proxy by adding the correction terms to the original proxy, it is a variance reduction technique as well.

Most of the results have been presented are based on a simple synthetic field, Teal South reservoir model, which is located in the Gulf of Mexico. Because of the model simplicity and measurement errors it has been used to assess the techniques and make robust conclusions. The Scapa field, which is located in the North Sea, has been used in Chapter 6 to show the MLMCMC technique is applicable to study more complicated reservoir model in terms of having more wells and more observed data compared with Teal South Model.

Codes used are written in Eclipse-100, R and Python. The approaches detailed

in the thesis are running on a single PC. Most of the results have been presented in the thesis are averages of several runs because the techniques used are stochastic techniques.

The key findings can be summarised as follows,

- Chapter 5, MLMC was able to solve stochastic ODE: the exponential growth and decay equation and PDEs with stochastic terms : Advection, Buckley-Leverett and (linear or semi-linear) pressure equations. For the exponential growth and decay equation, MLMC obtained performance speed up Monte Carlo Integration in the range of $10^2 - 10^4$. For the advection equation, MLMC was faster and cheaper (less computational cost) than Monte Carlo Integration without the loss of efficiency for estimating the mean of the solution. A number of numerical methods were applied. Also, the choice of the control condition (5.3) for the number of samples required to minimize the computational cost has an effect on the CPU time. MLMC was cheaper than Monte Carlo Integration for the pressure equation and for Buckley-Leverett equation. Also, Buckley-Leverett equation was almost as efficient as Monte Carlo Integration using three different control conditions for MLMC.

- Chapter 6, MLMCMC is applicable for uncertainty quantification in reservoir simulation. It is faster than Random Walk Metropolis (RWM) and as efficient as long chain RWM in terms of estimating the posterior distribution for oil rate. For Teal South field, MLMCMC was able to generate forecasts significantly faster than RWM and speed up RWM in the range 10 to 100 with no significant loss in accuracy. Moreover, MLMCMC succeeded in obtaining the histogram for the uncertain parameters and in addition, the uncertainty range captured most of the observed data with less computational cost compared with RMW. Furthermore, for fixed computational cost MLMCMC was more efficient than particle swarm optimization with neighbourhood algorithm Bayes because MLMCMC captured the most of the observed data.

- For Scapa field, MLMCMC estimated the Bayesian uncertainty range for the solution. The uncertainty distribution was more efficient using the standard

deviation σ, which control the mismatch between the observed data and simulated data, as the variable in (6.5) than when it was constant. MLMCMC performed in a similar way as on Teal South, with the difference that we could not afford to run the full RWM on Scapa to get a cost comparison because it required 15 minutes to run one simulation, which means 1000 samples would require 10 days to run.

- We saw in Chapter 7 that linear kernel was better than thin plate kernel for interpolating the oil rate surface of Teal South model because the simulated misfit was highly linearly correlated with the interpolated misfit. The quality of interpolated misfit was examined using different kernel types. A Gaussian kernel was better than a polynomial kernel of degree 3 for interpolating the misfit surface. The choice of kernels affects the efficiency of the Bayesian credible interval.

- Two-level proxy improved the quality of the proxy because it obtained the same uncertainty distribution as the simulated results. Furthermore, the computational cost decreased by 90% in case of using RWM with 10^4 samples.

- MLproxy decreased the computational cost compared to running the full simulation and modify the distribution created by the proxy with RWM.

- Chapter 8, for the same computational cost, MLMCMC was better than HMC because MLMCMC was able to estimate the distribution of oil rate, but HMC required more samples to achieve the same result.

- The detailed balance condition is proved for MLHMC. For the same accuracy, Bayesian credible interval for estimating the uncertainty distribution using MLHMC has all the observed data inside, however, for MLMCMC some of the observed data lie outside the range. MLHMC with Sobol sequence was the fastest algorithm compared with HMC, MLHMC with Latin hypercubic sampling and RWM. MLHMC with Sobol sequence obtained the same distribution as HMC on the finest grid within the sampling error, which means that MLHMC provided more robust inference than HMC.

9.2 Future Plan

In this section, I summarise my future research plans as follows

- Apply MLMC for solving polymer flood model, which has 3 components (water, oil and polymer) and two-phases (oil and water) [Chapter 5 discussed how to estimate the mean of the solution of one equation not a system of equations].
- Use MLMC for studying Buckley-Leverett equation augmented with a continuous initial condition as a function of x [Chapter 5 used B-L with constant initial condition].
- Parallelize the codes for MLMCMC, MLproxy and MLHMC to work faster for more complicated reservoir based on (Calderhead, 2014).
- Implement a new idea for MLMCMC (Dodwell et al., 2015) and compare between MLMCMC in the thesis and the new version using Teal South field as an example.
- Discuss the impact of assuming σ in the misfit definition (6.5) are unknown parameters corresponding to each time step. This means increasing the dimensionality of the problem.
- The misfit definition is based on assuming the errors are Gaussian distributed. To apply this assumption we have to check if the errors, $(q^s - q^o)$ are Gaussian distributed using Kolmogorov-Smirnov test (KS test). In case where the errors do not follow a Gaussian distribution, we have to find a suitable distribution to the errors. For example, Log-normal distribution, t-distribution or exponential distribution. I plane to study different distributions for the misfit.
- Apply MLMCMC on single phase model (Liu and Oliver, 2003) and compare between the performance of the solution and RWM solution.
- Use different experimental designs e.g., Box-Behnken and Central Composite. Then compare between them and the Sobol sequence.

- Use B-splines for constructing the emulator for the quantity of interest or the gradient of the misfit.

- After generating the initial samples using experimental design to build the proxy, we need to test the quality of the proxy. Based on the result of the test we decide whether to run more simulations or use the proxy. In case that more samples are needed based on the highest error between the simulated and interpolated result, we can select the region has that points and generate more samples from the bad region (Sefat et al., 2012). After that this proxy can be used to improve MLproxy.

- Study the error model of viscous fingering phenomena, in which instability occurs in the petroleum reservoir when oil is replaced by gas (O'Sullivan, 2004), using MLproxy.

- Discuss the optimal choice for MLHMC parameters by tuning the parameters. This includes HMC parameters and MLMC parameters as well.

- The parallel tempering method is used for optimisation and estimating the posterior distribution. It is able to explore a complex multimodal posterior distribution efficiently and those based on adding a parameter for controlling the behaviour of the sampler. When the parameter value is high, it makes the target distribution flatter, as a consequence it will be easier to sample from the target distribution. Parallel tempering can be combined with other simulation techniques, such as RWM (Earl and Deem, 2005; Carter and White, 2013) and HMC (Okur et al., 2007). For more details see (Malcolm, 2014; Machta and Ellis, 2011; Swendsen and Wang, 1986; Geyer, 1991; Atchad et al., 2011; Earl and Deem, 2005). One of my future research projects is to combine MLMC with parallel tempering to quantify the uncertainty.

- Improve MLMCMC code to quantify uncertainty when reservoir has irregular discretization as in (Ahmadi, 2012).

- There are different techniques for accelerating HMC, e.g., Riemannian mani-

fold HMC method. Riemannian Manifold HMC method accelerates the HMC based on the geometry of the parameter space (Girolami, 2011). One of my future research projects is comparing MLHMC and Riemannian Manifold HMC method.

- Reversible Jump Markov Chain Monte Carlo can select the model and estimate the distribution of the parameters. For more details see (Green, 1995). One of my research goals is to apply the Revisable Jump Markov Chain Monte Carlo for model selection in reservoir simulation.

Appendix A

We explain how to classify the first and the second order differential equations as hyperbolic, parabolic or elliptic.

Classification of PDE

In this section, we discuss how to classify a first and a second order PDE into elliptic, parabolic or hyperbolic types based on finding the eigenvalues and eigenvectors for the PDE (Strauss, 2007; Levandosky, 2002).

First Order PDE

The general formula for the first order PDE is,

$$au_t + bu_x + c = 0,$$

where $u(x,t)$, is the unknown function and (a, b, and c) are constants or functions of x and t. To calculate the eigenvalues for the PDE, we should solve $|a\lambda - b| = 0$ and find the eigenvalues λ. In more general, the general formula for the system of the first order PDEs is,

$$\mathbf{u}_t + \mathbf{A}.\mathbf{u}_x + \mathbf{d} = 0,$$

where $\mathbf{A}$, is a $(n \times n)$ matrix and $\mathbf{d}$ is a vector. To find the eigenvalues for the PDEs, we solve $\det(\mathbf{A} - \lambda\mathbf{I}) = 0$. Then, finding the eigenvalues λ, and corresponding eigenvectors. Based on both of them, we can decide the type of the PDE as follows,

(i) If we have no real eigenvalues, then the system is called elliptic.

(ii) If the eigenvalues are real and distinct, or if the eigenvalues are real and the system is not defective, then the system is called hyperbolic.

(iii) If the eigenvalues are real, but the system is defective, then system is called parabolic.

Second order PDE

The classification of 2nd order linear PDEs of two independent variables can be classified into elliptic, parabolic or hyperbolic PDEs. The general formula for the second order PDE is:

$$a\,u_{xx} + b\,u_{xy} + c\,u_{yy} + d\,u_x + eu_y + f\,u + g = 0,$$

where a, b,, $\cdots$ are functions of x and y. To classify, we calculate the discriminant $b^2 - 4ac$.

- If $b^2 - 4ac < 0$, then the equation is elliptic.
- If $b^2 - 4ac = 0$, then the equation is parabolic.
- If $b^2 - 4ac > 0$, then the equation is hyperbolic.

Generally, its in case of having n independent variables, the second order PDE can be written as follows,

$$\sum_{i=1}^{n}\sum_{j=1}^{n} a_{ij}\frac{\partial^2 u}{\partial x_i \partial x_j} + \sum_{i=1}^{n} b_i \frac{\partial u}{\partial x_i} + cu = d.$$

Once we have a matrix $A = (a_{ij})$, we solve the eigenvalue problem $det(A - \lambda I) = 0$, and then we count two different counts based on the solution values. We count Z_1, the number of zero eigenvalue and P_1, the number of positive eigenvalues. Now, we can assign a category to the PDE.

- If $Z_1 > 0$, then the matrix A is singular i.e. $\det(A) = 0$, and the PDE is parabolic.

- If $Z_1 = 0$ and $(P_1 = 0 \vee P_1 = n)$, then the PDE is elliptic.
- If $Z_1 = 0$ and $(P_1 = 1 \vee P_1 = n - 1)$, then the PDE is hyperbolic.
- If $Z_1 = 0$ and $(1 < P_1 < n - 1)$, then the PDE is an ultra-hyperbolic.

Appendix B

We prove Metropolis- Hastings, Hamiltonian Monte Carlo and Multilevel Markov chain Monte Carlo satisfied detailed balance.

Proof Metropolis- Hastings Satisfies Detailed Balance

To show Metropolis-Hastings chain is satisfying time reversibility, we should proof the detailed balance condition (B.1). Which can deduce that the target distribution π is the stationary distribution (Robert and Casella, 2010).

$$\mathbb{P}(x \to y)\pi(x) = \mathbb{P}(y \to x)\pi(y) \tag{B.1}$$

Prove that the detailed balance condition satisfy for Metropolis-Hastings chain?

Proof B.0.0.3 *We assume the acceptance probability from state x to state y is $\alpha(y|x) = \min\{1, \frac{\pi(y)q(x|y)}{\pi(x)q(y|x}\}$ (Robert and Casella, 2010), the proposal distribution for Metropolis-Hastings is $q(y|x)$, the transition probability from state x to state y*

is $\mathbb{P}(x \to y) = \alpha(y|x)q(y|x)$ *and the target distribution is* π.

$$
\begin{aligned}
LHS: \ \mathbb{P}(x \to y)\pi(x) &= \alpha(y|x)\, q(y|x)\pi(x) \\
&= \min\{1, \frac{\pi(y)q(x|y)}{\pi(x)q(y|x)}\}\, q(y|x)\pi(x) \\
&= \min\{\pi(x)q(y|x), \pi(y)q(x|y)\} \\
&= \pi(y)q(x|y)\ \min\{1, \frac{\pi(x)q(y|x)}{\pi(y)q(x|y)}\} \\
&= \pi(y)\,\mathbb{P}(y \to x).
\end{aligned}
$$

Proof Hamiltonian Monte Carlo Satisfies Detailed Balance

To show Hamiltonian Monte Carlo chain is satisfying time reversibility, we should proof the detailed balance condition (B.2). Which can deduce that the target distribution $P(q, p)$(q is the position and p is the momentum) is the stationary distribution (Bishop, 2006).

$$\mathbb{P}(x \to y)P(x, u) = \mathbb{P}(y \to x)P(y, u') \tag{B.2}$$

Prove that the detailed balance condition satisfy for Hamiltonian Monte Carlo chain?

Proof B.0.0.4 *We assume the acceptance probability from state* x *with momentum* u *to state* y *with momentum* u' *is* $\alpha(y|x) = \min\{1, e^{(-H(y,u')+H(x,u))}\}$,*(*$H(x, u)$ *is the total energy), the transition probability from state* x *to state* y *is* $\mathbb{P}(x \to y) = \alpha(y|x)$ *and the target distribution is* $P(x, u) \propto e^{-H(x,u)}$. *We prove the following*

$$\frac{\mathbb{P}(x \to y)P(x, u)}{\mathbb{P}(y \to x)P(y, u')} = 1$$

$$LHS: \frac{\mathbb{P}(x \to y)P(x,u)}{\mathbb{P}(y \to x)P(y,u')} = \frac{\alpha(y|x)P(x,u)}{\alpha(x|y)P(y,u')} = \frac{\min\{1, e^{H(x,u)-H(y,u')}\}e^{-H(x,u)}}{\min\{1, e^{H(y,u')-H(x,u)}\}e^{-H(y,u')}} = \frac{\min\{e^{-H(x,u)}, e^{-H(y,u')}\}}{\min\{e^{-H(y,u')}, e^{-H(x,u)}\}} = 1$$

Proof Multilevel Markov chain Monte Carlo Satisfies Detailed Balance

To show MLMCMC chain is satisfying time reversibility, we should proof the detailed balance condition (B.3). Based on the definitions for the acceptance rate and $\mathbb{P}^l(x \to y) = \alpha^l(y|x)\,\pi^{l-1}(y)$, l is the level and $\alpha^l(y|x) = \min\{1, \frac{\pi^l(y)\pi^{l-1}(x)}{\pi^l(x)\pi^{l-1}(y)}\}$ (Dodwell et al., 2015; Efendiev et al., 2014).

$$\mathbb{P}^l(x \to y)\pi^l(x) = \mathbb{P}^l(y \to x)\pi^l(y) \tag{B.3}$$

Prove that the detailed balance condition satisfy for MLMCMC chain?

Proof B.0.0.5 *For the initial level l_0 it is as Metropolis-Hastings. We focus to prove in the case $l > l_0$. We prove the following*

$$\frac{\mathbb{P}^l(x \to y)\pi^l(x)}{\mathbb{P}^l(y \to x)\pi^l(y)} = 1$$

$$LHS: \frac{\mathbb{P}^l(x \to y)\pi^l(x)}{\mathbb{P}^l(y \to x)\pi^l(y)} = \frac{\min\{1, \frac{\pi^l(y)\pi^{l-1}(x)}{\pi^l(x)\pi^{l-1}(y)}\}\pi^{l-1}(y)\pi^l(x)}{\min\{1, \frac{\pi^l(x)\pi^{l-1}(y)}{\pi^l(y)\pi^{l-1}(x)}\}\pi^{l-1}(x)\pi^l(y)} = \frac{\min\{\pi^{l-1}(y)\pi^l(x), \pi^l(y)\pi^{l-1}(x)\}}{\min\{\pi^l(y)\pi^{l-1}(x), \pi^{l-1}(y)\pi^l(x)\}} = 1$$

Bibliography

(2015). Decc, department of energy and climate change. `https://www.og.decc.gov.uk/pprs/full_production.htm`.

Ahmadi, M. (2012). *Modelling and Quantification of Structural Uncertainties in Petroleum Reservoirs Assisted by a Hybrid Cartesian Cut Cell/Enriched Multi-point Flux Approximation Approach*. PhD thesis, Institute of Petroleum Engineering, Heriot Watt University.

Alkhatib, A. (2014). Applying the multi-level monte carlo method to quantify uncertainty for chemical eor processes. In *Second EAGE Integrated Reservoir Modelling Conference*.

Alpak, F. and Kats, F. (2009). Stochastic history matching of a deepwater turbidite reservoir. In *SPE Reservoir Simulation Symposium, 2-4 February, The Woodlands, Texas*.

Alpak, F. and Lake, L. (1999). Validation of a modified carmankozeny equation to model two-phase relative permeabilities. In *SPE Annual Technical Conference and Exhibition, 3-6 October, Houston, Texas*.

Alpak, F., Vink, J., Gao, G., and Mo, W. (2013). Techniques for effective simulation, optimization, and uncertainty quantification of the in-situ upgrading process. *Journal of Unconventional Oil and Gas Resources*, 3:1–14.

Anderson, D. and Higham, D. (2012). Multi-level monte carlo for continuous time markov chains with applications in biochemical kinetics. *SIAM Multiscale Modeling and Simulation*, 10(1):146–179.

Anderson, E. (1999). Monte carlo methods and importance sampling. University of California, Berkeley.

Andrieu, C., Freitas, N. D., Doucet, A., and Jordan., M. (2003). An introduction to mcmc for machine learning. *Machine Learning*, 50(1-2):5–43.

Asmussen, A. and Glynn, P. (2007). *Stochastic Simulation*. Springer, New York.

Atchad, Y., Roberts, G., and Rosenthal, J. (2011). Towards optimal scaling of metropolis-coupled markov chain monte carlo. *Statistics and Computing*, 21(4):555–568.

Barenblatt, G. I. (1996). *Scaling, self-similarity, and intermediate asymptotics*. Cambridge University press.

Barth, A., Schwab, C., and Zollinger, N. (2011). Multi-level monte carlo finite element method for elliptic pdes with stochastic coefficients. *Numerische Mathematik*, 119(1):123–161.

Begg, S., Bratvold, R., and Campbell, J. (2001). Improving investment decisions using a stochastic integrated asset model. In *SPE Annual Technical Conference and Exhibition, New Orleans, LA, 30 September3 October*. Society of Petroleum Engineers.

Behrenbruch, P., Turner, G., and Backhouse, A. (1985). Probabilistic hydrocarbon reserves estimation: a novel monte carlo approach. In *Offshore Europe Conference, Aberdeen, 10-13 September*.

Beichl, I. and Sullivan, F. (2000). The metropolis algorithm. *Computing in Science and Engineering*, 2(1):65–69.

Beskos, A., Pillai, N., Roberts, G., Sanz-Serna, J., and Stuart, A. (2010). The acceptance probability of the hybrid monte carlo method in high-dimensional problems. In *AIP Conference Proceedings*, volume 1281.

Beskos, A., Pillai, N., Roberts, G., Sanz-Serna, J., and Stuart, A. (2013). Optimal tuning of the hybrid monte carlo algorithm. *Bernoulli*, 19(5A):1501–1534.

Bhuripanyo, C. (2014). Quantifying the benefit of ensemble selection for optimization under uncertainty. Master's thesis, Texas A & M University.

Bishop, C. (2006). *Pattern Recognition and Machine Learning*. New York, USA: Springer.

Blunt, M. J. and Christie, M. (1989). High resolution schemes for the numerical solution of hyperbolic conservation laws. Technical report, BP Research Center.

Bonet-Cunha, L., Oliver, D., Redner, R., and Reynolds, A. (1996). A hybrid markov chain monte carlo method for generating permeability fields conditioned to multiwell pressure data and prior information. In *Society of Petroleum Engineers*.

Box, G. and Jenkins, G. (1976). *Time series analysis: Forecasting and control*. Holden-Day, San Francisco, CA.

Brandt, A., Galun, M., and Ron, D. (1994). Optimal multigrid algorithms for calculating thermodynamic limits. *Journal of Statistical Physics*, 74(1):313–348.

Bratley, P. and Fox, B. L. (1988). Algorithm 659: Implementing sobols quasirandom sequence generator. *ACM Trans. Math. Softw*, 14:88–100.

Bratton, D. and Kennedy, J. (2007). Defining a standard for particle swarm optimization. In *Swarm Intelligence Symposium, 2007. SIS 2007. IEEE*, pages 120–127.

Buckley, S. and Leverett, M. (1942). Mechanism of fluid displacement in sands. *Transactions ATME*, 146(115):107–116.

Buhmann, M. (2003). *Radial basis functions: theory and implementations*, volume 12. Cambridge university press.

Burhenne, S., Jacob, D., and Henze, G. (2011). Sampling based on sobolsequences for monte carlo techniques applied to building simulations. In *12th conference of international building performance simulation association*.

Busby, D., Farmer, C., and Iske, A. (2007a). Uncertainty evaluation in reservoir forecasting by bayes linear methodology. In *Algorithms for Approximation*, pages 187–196. Springer Berlin Heidelberg.

Busby, D., Farmer, C., and Iske, A. A. (2007b). Hierarchical nonlinear approximation for experimental design and statistical data fitting. *SIAM Journal on Scientific Computing*, 29(1):49–69.

Caflisch, R. (1998). Monte carlo and quasi-monte carlo methods. *Acta numerica*, 7:1–49.

Calderhead, B. (2014). A general constructing for parallelizing metropolis-hastings algorithm. *PNAS*, 111(49):17408–17413.

Carter, J. (2004). Using bayesian statistics to capture the effects of modelling errors in inverse problems. *Mathematical geology*, 36(2):187–216.

Carter, J. and Ballester, P. (2004). A real parameter genetic algorithm for cluster identification in history matching. In *ECMOR IX -9th European Conference on the Mathematics of Oil Recovery.*

Carter, J. and White, D. (2013). History matching on the imperial college fault model using parallel tempering. *Computational Geosciences*, 17(1):43–65.

Carter, S. (2010). *A stochastic Buckley Leverett model.* PhD thesis, Uninersity of Adelaide, Australia.

Chen, H. K. (1988). Field development of the scapa field: A marginal north sea field. In *European Petroleum Conference, 16-19 October, London, United Kingdom.* Society of Petroleum Engineers.

Chen, Y. and Oliver, D. (2012). Ensemble randomized maximum likelihood method as an iterative ensemble smoother. *Mathematical Geosciences*, 44(1):1–26.

Choo, K. (2000). Learning hyperparameters for neural network models using hamiltonian dynamics. Master's thesis, University of Toronto.

Christen, J. and Fox, C. (2005). Mcmc using an approximation. *Journal of Computational and Graphical Statistics*, 14(4):795–810.

Christie, M. and Blunt, M. (2001). Tenth spe comparative solution project: A comparison of upscaling techniques. *SPE*, pages 308–317.

Christie, M., Cliffe, A., Dawid, P., and Senn, S. (2011). *Simplicity,Complexity and Modeling*. John Wiley & Sons, Ltd, Chichester, UK.

Christie, M., Demyanov, V., and Erbas, D. (2006). Uncertainty quantification for porous media flows. *Journal of Computational Physics*, 217(1):143–158.

Christie, M., MacBeth, C., and Subbey, S. (2002). Multiple history-matched models for Teal South. *The Leading Edge*, 21(3):286–289.

Christie, M., Pickup, G., O'Sullivan, A., and Demyanov, V. (2008). Use of solution error models in history matching. In *ECMOR XI-.11th European Conference on the Mathematics of Oil Recovery*.

Christie, M. A. (1987). Analysis of numerical diffusion and dispersion in finite difference schemes for linear convection problems. Technical report, BP Research.

Chun, H. (2010). Hull form parameterization technique with local and global optimization algorithms. In *The International Conference of Marine Technology*.

Cliffe, K. A., Giles, M. B., Scheichl, R., and Teckentrup, A. L. (2011). Multi-level Monte Carlo methods and applications to elliptic PDEs with random coefficients. *Computing and Visualization in Science*, 14(1):3–15.

Cowles, M. and Carlin., B. (1996). Markov chain monte carlo convergence diagnostics: a comparative review. *American Statistical Association*, 91(434):883–904.

Dadashpour, M. (2009). *Reservoir Characterization Using Production Data and Time-Lapse Seismic Data*. PhD thesis, Norwegian University of Science and Technology Faculty of Engineering Science and Technology Department of Petroleum Engineering and Applied Geophysics.

Darcy, H. (1856). *Les Fontaines Publiques de la Ville de Dijon*. Dalmont, Paris.

Demyanov, V., Foresti, L., Kanevski, M., and Christie, M. (2010). Multiple kernel learning approach for reservoir modelling. In *12th European Conference on the Mathematics of Oil Recovery*.

Dodwell, T., Ketelsen, C., Scheichl, R., and Teckentrup, A. (2015). A hierarchical multilevel markov chain monte carlo algorithm with applications to uncertainty quantification in subsurface flow. *SIAM/ASA Journal on Uncertainty Quantification*, 3(1):1075–1108.

Dostert, P., Efendiev, Y., Hou, T., and Luo, W. (2006). Coarse-gradient langevin algorithms for dynamic data integration and uncertainty quantification. *Journal of computational physics*, 217(1):123–142.

Duane, S., Kennedy, A., Pendleton, B., and Roweth, D. (1987). Hybrid Monte Carlo. *Physics Letters B*, 195:216–222.

Duffy, P. (2004). Introduction to finite difference methods. University Lecture, Department of physics UCD.

Earl, D. and Deem, M. (2005). Parallel tempering: Theory, applications, and new perspectives. *Physical Chemistry Chemical Physics*, 7(23):3910–3916.

Efendiev, Y., Datta-Gupta, A., Ginting, V., Ma, X., and Mallick, B. (2005). An efficient two-stage markov chain monte carlo method for dynamic data integration. *WATER RESOURCES RESEARCH*, 41(12).

Efendiev, Y., Iliev, O., and Kronsbein, C. (2013). Multi-level monte carlo methods using ensemble level mixed msfem for two-phase flow and transport simulations. *Computational Geosciences*, 17(5):833–850.

Efendiev, Y., Jin, B., M. Presho, M., and Tan, X. (2014). Multilevel markov chain monte carlo method for high-contrast single-phase flow problems. *Communications in Computational Physics*, 17(1):259–286.

Ellison, R. I., Simlote, V. N., and Ko, R. S. (1992). Continued development, reservoir management and reservoir modelling have increased proven developed oil reserves by over 35scapa field. In *European Petroleum Conference, 16-18 November, Cannes, France*. Society of Petroleum Engineers.

Engelbrecht, A. (2005). *Fundamentals of computational Swarm Intelligence*. John Wiley & Sons, Ltd, Chichester, UK.

Erbas, D. (2006). *Sampling Strategies for Uncertainty Quantification in Oil Recovery Prediction*. PhD thesis, Institute of Petroleum Engineering, Heriot Watt University.

Erbas, D. and Christie, M. (2007). Effect of sampling strategies on prediction uncertainty estimation. In *SPE Reservoir Simulation Symposium.*

Evensen, G. (2007). *Data Assimilation: The Ensemble Kalman Filter*. Springer Verlag, Berlin.

Farmer, C. (2005). Geological modelling and reservoir simulation. In Iske, A. and Randen, T., editors, *Mathematical methods and modelling in hydrocarbon exploration and production*, pages 119–212. Springer-Verlag, Heidelberg.

Farooq, Y. (2011). Multiple history matched models for scapa field. Master's thesis, IPE., Heriot-Watt University, Edinburgh.

Floris, F., Bush, M., Cuypers, M., Roggero, F., and Syversveen, A. (2001). Methods for quantifying the uncertainty of production forecasts: a comparative study. *Petroleum Geoscience*, 7(S):S87–S96.

Floris, F. J. T. and Peersmann, M. R. H. E. (1998). Uncertainty estimation in volumetrics for supporting hydrocarbon exploration and production decision-making. *Petroleum Geoscience*, 4(1):33–40.

Gamerman, D. (1997). *Markov Chain Monte Carlo, Stochastic simulation for Bayesian inference*. Chapman and Hall CRC, USA.

Gelfand, A. and Smith, A. (1990). Sampling-based approaches to calculating marginal densities. *Journal of the American statistical association*, 85(410):398–409.

Geman, S. and Geman, D. (1984). Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *Pattern Analysis and Machine Intelligence, IEEE Transactions*, (6):721–741.

Geweke, J. (1992). Evaluating the accuracy of sampling based approaches to the calculation of posterior moments. *Bayesian Statistics*, 4:169–193.

Geyer, C. (1991). Computing science and statistics. In *23rd Symposium on the Interface, American Statistical Association, New York*.

Giles, M. (2015). Multilevel monte carlo methods. *Acta Numerica*, 24(1):259–328.

Giles, M., Nagapetyan, T., and Ritter, K. (2015). Multi-level monte carlo approximation of distribution functions and densities. *SIAM/ASA Journal on Uncertainty Quantification*, 3:267–295.

Giles, M. B. (2008). Multi-levelMonte Carlo path simulation. *Operation Research*, 56(3):607–617.

Gilks, W., Richardson, S., and Spiegelhalter, D. (1996). *Markov Chain Monte Carlo in Practice*. Chapman and Hall.

Girolami, M., . C. B. (2011). Riemann manifold langevin and hamiltonian monte carlo methods. *Statistical Methodology*, 73(2):123–214.

Glimm, J., Hou, S., ha Lee, Y., Sharp, D., and Ye, K. (2001a). Prediction of Oil Production With Confidence Intervals. In *SPE Reservoir Simulation Symposium, 11-14 February, Houston, Texas*, number 66350. Society of Petroleum Engineers.

Glimm, J., Hou, S., Kim, H., Lee, Y., Sharp, D., Ye, K., and Zou, Q. (2001b). Risk management for petroleum reservoir production: A simulation-based study of prediction. *Computational Geosciences*, 5(3):173–197.

Glimm, J., Hou, S., Lee, Y., Sharp, D., and Ye, K. (2003). Solution error models for uncertainty quantification. *Contemporary Mathematics*, 327:115–140.

Glimm, J., Hou, S., Lee, Y., Sharp, D., and Ye, K. (2004). Sources of uncertainty and error in the simulation of flow in porous media. *Computational & Applied Mathematics*, 23:109–120.

Glimm, J. and Sharp, D. (1999). Prediction and the quantification of uncertainty. *Physica D*, 133(1):152–170.

Goodwin, N. (2015). Bridging the gap between deterministic and probabilistic uncertainty quantification using advanced proxy based methods. In *SPE Reservoir Simulation Symposium, 23-25 February, Houston, Texas, USA*.

Goodwin, N. and Powell, M. (2012). Simulation and uncertainty: Lessons from other industries. *Society of Petroleum Engineers*, 8(3):20–24.

Green, P. (1995). Reversible jump markov chain monte carlo computation and bayesian model determination. *Biometrika*, 82(4):711–732.

H. Gould, H. and Tobochnik, J. (2010). *Statistical and thermal physics: with computer applications.* Princeton University Press.

Hadamard, J. (1902). Sur les problmes aux drives partielles et leur signification physique. *Princeton University Bulletin*, 13:49–52.

Hajian, A. (2007). Efficient Cosmological Parameter Estimation with Hamiltonian Monte Carlo. *Physical Review D*, 75(8):083525.

Hajizadeh, Y. (2011). *Population-Based Algorithms for Improved History Matching and Uncertainty Quantification of Petroleum Reservoirs.* PhD thesis, Institute of Petroleum Engineering, Heriot Watt University.

Hajizadeh, Y., Demyanov, V., Mohamed, L., and Christie, M. (2011). Comparison of evolutionary and swarm intelligence methods for history matching and uncertainty quantification in petroleum reservoir models. In *Intelligent Computational Optimization in Engineering*, pages 209–240. Springer.

Hardy, R. (1971). Multiquadric equations of topography and other irregular surfaces. *Journal of Geophysical Research*, 76(8):1905–1915.

Hastings, W. (1970). Monte carlo sampling methods using markov chains and their applications. *Biometrika*, 57(1):97–109.

Heinrich, S. (2001). Multi-level Monte Carlo methods. In *Large-scale scientific computing. 3rd international conference*.

Heinrich, S. (2006). Monte carlo approximation of weakly singular integral operators. *Journal of Complexity*, 22(2):192–219.

Helbert, C., Dupuy, D., and Carraro, L. (2009). Assessment of uncertainty in computer experiments from universal to bayesian kriging. *Applied Stochastic Models in Business and Industry*, 25(2):99–113.

Hubbert, M. (1957). Darcy's law and the field equations of the flow of underground fluids. *International Association of Scientific Hydrology. Bulletin*, 2(1):23–59.

Iman, R. and Conovera, W. (1980). Small sample sensitivity analysis techniques for computer models.with an application to risk assessment. *Communications in Statistics - Theory and Methods*, 9(7):1749–1842.

Iske, A. (2004). *Multiresolution methods in scattered data modelling.*, volume 37. Springer Science & Business Media.

Jensen, J., Lake, L., Corbett, P., and Goggin, D. (2000). *Statistics for petroleum engineers and geoscientists*. Elservier, New York.

Josset, L., Demyanov, V., Elsheikh, A., and Lunati, I. (2015). Accelerating monte carlo markov chains with proxy and error models. *Computers &Geosciences*, 85(2015):38–48.

Kabir, C. and Young, N. (2004). Handling production-data uncertainty in history matching: the meren reservoir case study. *SPE Reservoir Evaluation & Engineering*, 7(2):123 –131.

Kaipio, J. and Somersalo, E. (2005). *Computational and Statistical Methods for Inverse Problems. Monograph.* SpringerVerlag.

Kebaier, A. (2005). Statistical romberg extrapolation: a new variance reduction method and applications to options pricing. *Annals of Applied Probability*, 15(4):2681–2705.

Kennedy, J. and Eberhart, R. C. (1995). Particle swarm optimization . In *IEEE Int. Conf. Neural Networks, Perth, Australia*, pages 1942–1948.

Ketelsen, C., Scheichl, R., and Teckentrup, A. (2013). A hierarchical multilevel markov chain monte carlo algorithm with applications to uncertainty quantification in subsurface flow. *arXiv:1303.7343*.

Khouider, B. (2008). Monte carlo integration. University of Victoria.

Kuzmanovska, I. (2012). Markov chain monte carlo methods in biological mechanistic models. Master's thesis, ETH Zurich, ETH CSB, Computational Systems Biology.

Leimkuhler, B. and Reich, S. (2005). *Simulating Hamiltonian Dynamics*. Cambridge University Press.

Levandosky, J. (2002). Math 220a partial differential equations of applied Mathematics. http://www.stanford.edu/class/math220a/handouts/firstorder.pdf.

Leveque, R. J. (1992). *Numerical methods for conservation laws*. Birkhäuser.

Liu, N. and Oliver, D. (2003). Evaluation of Monte Carlo methods for assessing uncertainty. *Soc. Pet. Eng. J.*, 8(2):188–195. SPE 84936PA.

Lord, G. J., Powell, C. E., and Shardlow, T. (2014). *Introduction to computational stochastic PDEs*. Cambridge University Press.

Machta, J. and Ellis, R. (2011). Monte carlo methods for rough free energy landscapes: Population annealing and parallel tempering. *Journal of Statistical Physics*, 144(3):541–553.

MacKay, D. (2002). *Information Theory, Inference & Learning Algorithms*. Cambridge University Press.

Malcolm, S. (2014). A parallel tempering algorithm for probabilistic sampling and multimodal optimization. *Geophysical Journal International*, 196:357–374.

McGann, G. J., Green, S. C. H., Harker, S. D., and Romani, R. S. (1991). The scapa field, block 14/19, uk north sea. *Geological Society, London, Memoirs*, 14(1):369–376.

McKay, M., Beckman, R., and Conover, W. (1979). A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*, 21(2):239–245.

Metropolis, N., Rosenbluth, A., Rosenbluth, M., Teller, A., and Teller, E. (1953). Equation of state calculations by fast computing machines. ,. *The journal of chemical physics*, 21(6):1087–1092.

Metropolis, N. and Ulam, S. (1949). The monte carlo method. *Journal of the American statistical association*, 44(247):335–341.

Minasny, B. and McBratney, A. (2002). Uncertainty analysis for pedotransfer functions. *European Journal of Soil Science*, 53(3):417–430.

Minasny, B. and McBratney, A. (2006). A conditioned latin hypercube method for sampling in the presence of ancillary information. *.Computers and Geosciences*, 32(9):1378–1388.

Mohaghegh, S. (2006). Quantifying uncertainties associated with reservoir simulation studies using surrogate reservoir models. In *Annual Technical Conference and Exhibition, San Antonio, Texas, USA*.

Mohamed, L., Christie, M., and Demyanov, V. (2010a). Comparison of stochastic sampling algorithms for uncertainty quantification. *SPE*, 15(01):31–38.

Mohamed, L., Christie, M., Demyanov, V., Robert, E., and Kachuma, D. (2010b). Application of particle swarms for history matching in the brugge reservoir. In *SPE Annual Technical Conference and Exhibition*.

Mu, A., Cao, D., and Wang, X. (2009). A modified particle swarm optimization algorithm. *Natural Science*, 1(2):151–155.

Müller, F. (2014). *Stochastic methods for uncertainty quantification in subsurface flow and transport problems*. PhD thesis, ETH-Zürich.

Müller, F., Jenny, P., and Meyer, D. W. (2013). Multilevel Monte Carlo for two phase flow and transport in random heterogeneous porous media. *computional Physics*, 250:685–702.

Müller, F., Meyer, D., and Jenny, P. (2014). Solver-based vs. grid-based multilevel monte carlo for two phase flow and transport in random heterogeneous porous media. *Journal of Computational Physics*, 268:39–50.

Nadaraya, E. (1964). On estimating regression. *Theory of Probability and its Applications*, 9(1):157–159.

Neal, P., Roberts, G., and Yuen, J. (2012). Optimal scaling of random walk metropolis algorithms withdiscontinuous target densities. *The Annals of Applied Probability*, 22(5):1880–1927.

Neal, R. (1993). Probabilistic inference using markov chain monte carlo methods. Technical report, Department of Computer Science, University of Toronto.

Neal, R. (2011). *Handbook of Markov Chain Monte Carlo*. CRC Press.

Oksendal, B. K. (2002). *Stochastic Differential Equations: An Introduction with Applications*. Springer.

Okur, A., Roe, D., Cui, G., Hornak, V., and Simmerling, C. (2007). Improving convergence of replica-exchange simulations through coupling to a high-temperature structure reservoir. *Journal of Chemical Theory and Computation*, 3(2):557–568.

Oliver, D. (1996). Multiple realizations of the permeability field from well-test data. *SPE Journal,*, 1(2):145–154.

Oliver, D., Cunha, L., and Reynolds., A. (1997). Markov chain monte carlo methods for conditioning a permeability field to pressure data. *Mathematical Geology*, 29(1):61–91.

Oliver, D., He, N., and Reynolds, A. (1996). Conditioning permeability fields to pressure data. In *European Conference for the Mathematics of Oil Recovery (ECMOR), Leoben, Austria*.

Oliver, D., Reynolds, A., and Liu., N. (2008). *Inverse theory for petroleum reservoir characterization and history matching.* Cambridge University Press.

Orr, M. (1996). Introduction to radial basis function networks. Technical report, Technical Report, Center for Cognitive Science, University of Edinburgh.

O'Sullivan, A. and Christie, M. (2005). Error models for reducing history match bias. *Computational Geosciences*, 9(1):125–153.

O'Sullivan, A. and Christie, M. (2006a). Simulation error models for improved reservoir prediction. *Reliability Engineering and System Safety*, 91(10):1382–1389.

O'Sullivan, A. and Christie, M. (2006b). Simulation error models for improved reservoir prediction. *Reliability Engineering and System Safety*, 91(10-11):1382–1389.

O'Sullivan, A. E. (2004). *Modeling Simulation Error For Improved Reservoir Prediction*. PhD thesis, Institute of Petroleum Engineering, Heriot Watt University.

Pauli, S., Gantner, R., Arbenz, P., and Adelmann, A. (2015). Multilevel monte carlo for the feynman-kac formula for the laplace equation. *BIT Numerical Mathematics*, pages 1–19.

Pebesma, E. and Heuvelink, G. (1999). Latin hypercube sampling of gaussian random fields. *Technometrics*, 41(4):303–312.

Pettersen, Q. (2006). Basics of reservoir simulation with the eclipse reservoir simulator. Dept. of Mathematics, Univirsty of Bergen.

Pickup, G., Valjak, M., and Christie, M. (2008). Model complexity in reservoir simulation. In *11th European Conference on the Mathematics of Oil Recovery*. EAGE.

Rakhib, A. (2004). Numerical schemes applied to the burgers and buckleyleverett equation. Master's thesis, University of Reading.

Richardson, R. and Zandt, G. (2009). Inverse problems in geophysics. Department of Geosciences, University of Arizona , Tucson, Arizona 85721. `http://www.geo.arizona.edu/geo5xx/geos567/classnotes/Complete-fall09.pdf`.

Robert, C. and Casella, G. (1999). *Monte carlo statistical methods*. Springer-Verlag, New York, USA.

Robert, C. P. and Casella, G. (2010). *Introducing Monte Carlo Methods with R*. Springer, New York.

Roberts, G. and Rosenthal., J. (2001). Optimal scaling for various metropolis-hastings algorithms. *Statistical Science*, 16(4):351–367.

Rosin, M., Ricketson, L., Dimits, A., Caflisch, R., and Cohen, B. (2014). Multilevel monte carlo simulation of coulomb collisions. *Journal of Computational Physics*, 247:140–157.

Sahlin, K. (2011). Estimating convergence of markov chain monte carlo simulations. Master's thesis, Stockholms universitet.

Saltelli, A., Annoni, P., Azzini, I., Campolongo, F., Ratto, M., and Tarantola, S. (2010). Variance based sensitivity analysis of model output. design and estimator for the total sensitivity index. *Computer Physics Communications*, 181(2):259–270.

Saltelli, A., Ratto, M., Andres, T., Campolongo, F., Cariboni, J., Gatelli, D., Saisana, M., and Tarantola, S. (2008). *Global sensitivity analysis: The primer*. Wiley.

Sambridge, M. (1999a). Geophysical inversion with a neighbourhood algorithm−−i. searching a parameter space. *Geophysical Journal International*, 138(2):479–494.

Sambridge, M. (1999b). Geophysical inversion with a neighbourhood algorithm−−ii. appraising the ensemble. *Geophysical Journal International*, 138(3):727–746.

Schaaf, T., Labat, N., and Coureaud, B. (2008). Using experimental designs, assisted history matching tools and bayesian framework to get probabilistic production forecasts. In *70th EAGE Conference and Exhibition.*

Schulze-Riegert, R., Axmann, J., Haase, O., D. Rian, D., and You, Y. (2001). Optimisation methods for history matching of complex reservoirs. In *SPE Reservoir Simulation Symposium, SPE 66393, 1114 February, Houston, Texas, USA.*

Schulze-Riegert, R., Chataigner, F., Kueck, N., Pajonk, O., Baffoe, J., Ajala, I., Awofodu, D., Almuallim, H., et al. (2013). Strategic scope of alternative optimization methods in history matching and prediction workflows. In *SPE Middle East Oil and Gas Show and Conference. SPE-164337-MS Society of Petroleum Engineers.*

Sefat, M., Salahshoor, K., Jamialahmadi, M., and Moradi, B. (2012). A new approach for the development of fast-analysis proxies for petroleum reservoir simulation. *Petroleum Science and Technology*, 18(30):1920–1930.

Settles, M. (2005). An introduction to particle swarm optimization. Technical report, Department of Computer Science, University of Idaho, Moscow.

Shu, C. W. (2006). Numerical methods for hyperbolic conservation laws. Lecture notes (AM257).

Siddiqi, A. and Manchanda, P. (2005). A first course of differential equations with applications. King Fahd University of Petroleum & Minerals.

Sivia, D. (1996). *Data Analysis-A Bayesian Tutorial*. Claredon Press, Oxford.

Sobol', I. and Levitan, Y. (1999). A pseudo random number generator for personal computers. *Computers and Mathematics with Applications*, 37(4-5):33–40.

Sokal, A. (1997). *Monte Carlo methods in statistical mechanics: foundations and new algorithms*. Springer US.

Sorbie, K., Pickup, G., and Mackay, E. (2015). Reservoir simulation. Heriot-Watt University.

Stephen, K., Soldo, J., MacBeth, C., and Christie, M. (2005). Multiple model seismic and production history matching: A case study. In *SPE Europec / EAGE Annual Conference, Madrid, June, 13-16*. SPE.

Stoian, E. (1965). Fundamentals and applications of the monte carlo method. *Journal of Canadian Petroleum Technology*, 4(3):120–129.

Storn, R. and Price, K. (1995). Differential evolution a simple and efficient adaptive scheme for global optimization over continuous spaces. Technical report, International Computer Science Institute, Berkeley,.

Storn, R. and Price, K. (1997). Differential evolution - a simple and efficient heuristic for global optimization over continuous space. *Journal of Global Optimization*, 11:341–359.

Strauss, W. A. (2007). *Partial differential equations. An Introduction*. John Wiley and Sons, 2 edition.

Strikwerda, J. (1989). *Finite Difference Schemes and Partial Differential Equations*. Chapman & Hall.

Subbey, S., Christie, M., and Sambridge, M. (2004). Prediction under uncertainty in reservoir modeling. *Journal of Petroleum Science and Engineering*, 44(1-2):143–153.

Suzuki, S. and Caers, J. (2006). History matching with an uncertain geological scenario. In *SPE Annual Technical Conference and Exhibition, 24-27 September, San Antonio, Texas, USA*. Society of Petroleum Engineers.

Swendsen, R. and Wang, J. (1986). Replica monte carlo simulation of spin-glasses. *Physical Review Letters*, 57(21):2607–2609.

Tavassoli, Z., Carter, J., and King, P. (2005). An analysis of history matching errors. *Computational Geosciences*, 9(2-3):99–123.

Teckentrup, A., Scheichl, R., Giles, M., and Ullmann, E. (2013). Further analysis of multilevel monte carlo methods for elliptic pdes with random coefficients. *Numerische Mathematik*, 125(3):569–600.

Tierney, L. (1994). Markov chains for exploring posterior distributions (with discussion). *Annals of Statistics*, 22(4):1701–1762.

TNO (2003). Probabilistic History Matching: Production Forecasting with Uncertainty Quantification. Technical report, Netherlands Organisation for Applied Scientific Research.

Trangenstein, J. A. (1986). Multi-phase flow in porous media: Mechanics and Mathematics and Numerics. IBM Scientific center, Bergen Norway.

Trefethen, L. (1996). *Finite Difference and Spectral Methods for Ordinary and Partial Differential Equations*. Cornell University-Department of Computer Science and Center for Applied Mathematics.

Walsh, B. (2004). Markov Chain Monte Carlo and Gibbs Sampling.

Walstrom, J., Mueller, T., and McFarlane, R. (1967). Evaluating uncertainty in engineering calculations. In *SPE 1928, 42nd Annual Fall Meeting, Houston, Texas, USA, 1-4 October*.

Wang, P., Tartakovsky, D. M., Jarman, K. D., and Tartakovsky, A. M. (2013). Cdf solutions of buckley-leverett equation with uncertain parameters. *Multiscale Model. Simul*, 11(1):118–133.

Watson, G. (1964). Smooth regression analysis. *The Indian Journal of Statistics, Series A*, 26(4):359–372.

Weizhong Luo, B. (1986). Saturation profiles from petroleum reservoir simulation studies compared with buckley-leverett results. Master's thesis, Petroleum engineering, Texas Tech University.

William, A. and Eaton, M. (2013). On thinning of chains mcmc. *Methods in ecology and evolution*, 3:112–115.

Williams, G. J. J., Mansfield, M., MacDonald, D. G., and Bush, M. D. (2004). Top-Down Reservoir Modelling. In *SPE Annual Technical Conference and Exhibitions*, number 89974. Society of Petroleum Engineers.

Yeten, B., Castellini, A., Guyaguler, B., and Chen, W. (2005). A comparison study on experimental design and response surface methodologies. In *SPE Reservoir Simulation Symposium. Society of Petroleum Engineers*.

Yu, T., Wilkinson, D., and Castellini, A. (2008). Constructing reservoir flow simulator proxies using genetic programming for history matching and production forecast uncertainty analysis. *Journal of Artificial Evolution and Applications*, 2008:1–13.

Zollinger, N. (2010). Multi-level monte carlo finite element methods for elliptic partial differential equations with stochastic data. Master's thesis, ETH.

Zubarev, D. (2009). Pros and cons of applying proxy-models as a substitute for full reservoir simulations. In *Annual Technical Conference and Exhibition, New Orleans, Louisiana, USA*.

Zwhalen, E. and Patzek, T. W. (1997). Linear transient flow solution for primary oil recovery with infill and conversion to water injection. *In Situ*, 21(4):297–330.