PHYSICS CONSTRAINED DATA-DRIVEN TECHNIQUE FOR RESERVOIR PROXY MODEL AND MODEL ORDER REDUCTION

A Dissertation

by

ANQI BAO

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Chair of Committee,	Eduardo Gildin
Committee Members,	Michael King
	John Killough
	Yalchin Efendiev
Head of Department,	Jeff Spath

August 2019

Major Subject: Petroleum Engineering

ABSTRACT

In reservoir engineering, data-driven methodologies have been applied successfully to infer interwell connections and flow patterns in the subsurface, model order reduction of reservoir simulations, and in assisting field development plans, including, history matching and performance prediction phases, of conventional and unconventional reservoirs. In this work, we propose to utilize data driven methods for achieving two main objectives: (1) enhance model order reduction (MOR) techniques accounting for sparsity in the data; and (2) reservoir simulation proxy model development based solely on data.

For the first objective, fast simulation algorithms based on reduced-order modeling have been developed in order to facilitate large-scale and complex computationally intensive reservoir simulation and optimization. Methods like proper orthogonal decomposition (POD) and Dynamic Mode Decomposition (DMD) have been successfully used to efficiently capture and predict the behavior of reservoir fluid flow. Non-intrusive techniques (e.g., DMD), are especially attractive as it is a data-driven approach that do not require code modifications (equation free). To achieve our first objective with the concept of sparsity in statistical learning, we further enhance the performance and reduce the dimension of standard DMD, by investigating sparse approximations of the snapshots.

The method to achieve the second objective can further be classified into two categories: (1) building proxy model by system identification method; and (2) end to end production prediction with machine learning techniques. Although real-time data

acquisition and analysis, are becoming routine in many workflows (such as in reservoir simulations), there is still a disconnect between raw data and the traditional theoretical first laws principles, whereby conservation laws and phenomenological behavior are used to derive the underlying spatio-temporal evolution equations. We propose to combine sparsity promoting methods and machine learning techniques to find the governing equation from the spatio-temporal data series from a reservoir simulator. The idea is to connect data with the physical interpretation of the dynamical system. We achieve this by identifying the nonlinear ODE system equations of our discretized reservoir system. In addition, as production prediction analysis has been the ultimate goal of many reservoir simulation/modeling, various types reservoir simulation has been developed to build efficient and accurate model to provide the most information about reserves and aid in decision making process. The other proxy model we developed is benefit from the evolution of machine learning technique and increasing availability of extensive amounts of historical data. A powerful technique called recurrent neural network (RNN) has been proved useful for modeling with sequence data. We apply RNN on analyzing control parameter data and synthetic historical production data for better reservoir characterization and prediction. All of the above mentioned MOR and proxy model development will be tested on single- and two-phase fluid flow reservoir simulation problems.

DEDICATION

To

my parents and friends

for their love and support throughout my Ph.D.

ACKNOWLEDGEMENTS

I would like to thank my committee chair, Dr. Eduardo Gildin, and my committee members, Dr. Michael King, Dr. John Killough, and Dr. Yalchin Efendiev, for their guidance and support throughout the course of this research.

I would also like to extend my gratitude to my advisor Dr. Huang in Statistics department for his effort and support for my interdisciplinary research.

Thanks also go to my friends and colleagues and the department faculty and staff for making my time at Texas A&M University a great experience.

Finally, thanks to my parents for their patience and love. None of these works is possible without them.

CONTRIBUTORS AND FUNDING SOURCES

Contributors

This work was supervised by a dissertation committee consisting of Professor Eduardo Gildin [advisor], Professor John Killough and Professor Michael King of the Department of Petroleum Engineering, as well as Professor Yalchin Efendiev of the Department of Mathematics.

Work in Chapter 5 is supervised by both Professor Eduardo Gildin and Professor Jianhua Huang from Statistics Department.

All the other work conducted for the dissertation was completed by the student independently.

Funding Sources

This work was supported in part by Energi Simulation (Former Foundation CMG) Research Chair Grant at Texas A&M University.

NOMENCLATURE

Abbreviations

BHP	Bottom Hole Pressure
DMD	Dynamic Mode Decomposition
POD	Proper Orthogonal Decomposition
MOR	Model Order Reduction
DEIM	Discrete Empirical Interpolation Method
SVD	Singular Value Decomposition
SINDy	Sparse Identification of Nonlinear Dynamic system
RNN	Recurrent Neural Network
CNN	Convolutional Neural Network
LSTM	Long Short Term Memory

Variables

f	fractional flow
J	Jacobian
K	permeability tensor, ft^2
k	permeability, md
P	pressure, psi
$\mathbb{P}$	snapshot matrix
q	volumetric flow rate, bbl/day
R	residual

s	water saturation
t	time, days
v	phase velocity, ft/sec
$\boldsymbol{v}$	well control parameter
ρ	density, lb/ft^3
η	viscosity, cp
φ	porosity
λ	mobility factor
ψ	basis of finite dimensional space
α	the amplitude diagonal matrix in sparse DMD method
γ	the penalty factor on number of nonzero element of α
θ, τ	Lagrange multiplier
ϕ	POD or DMD basis

Subscripts

χ	phase water or oil
W	water
O	oil
r	reduced order
ro	relative to oil
rq	relative to water

Superscripts

ns	total number of timestep

nc	total number of grid block
*n*v	number of velocity variables
np	number of pressure variables

TABLE OF CONTENTS

Page

LIST OF FIGURES

Page

LIST OF TABLES

Page

1. INTRODUCTION

Reservoir simulation processes have long been used for production prediction, uncertainty assessment, inverse problems, sensitivity analysis, and optimizing well controls [1, 2]. In addition, it has been combined with the idea of closed loop reservoir management including the real-time data assimilation processes into the reservoir simulation workflow to improve accuracy. The complexity of the dynamic system lies in mathematical representation of the mass conservation equations, where coupled first order or second order differential equations are present, which may reach tens or hundreds of partial differential equations (PDEs), if multiphase and compositional models are used [3]. The discretization of these PDEs, especially in 3D modeling, can easily lead to more than 1 million of equations to be solved every single time step and Newton iteration. Thus, although the high-fidelity model is preferred for high accuracy, solving this the large dimension problem has become the bottom neck for post-process such as reservoir optimization/management and uncertainty quantification [4-7]. Considering the computational time and storage capability, the reduced order modeling becomes the choice to solve the problem in a timely manner while preserves accuracy. Other techniques for reducing simulation time, such as parallelization in the scope of high-performance computation, efficient sampling in MCMC (Markov Chain Monte Carlo), also exist but they will not be discussed in this work, as we view them as complementary to our developments

1.1. Current MOR and Its Limitation

The motivation of model order reduction is to efficiently compute relatively accurate low dimensional realization of such large-scale systems. Usually, in a real-world reservoir where highly heterogeneous structure such as shale, river channels and faults exist, the detailed simulation of fluid flow will be computationally expensive owning to the very fine mesh required to capture the spatial variability of the static properties. For example, in shale reservoir, there exist both high conductivity or permeability regions due to the fracture and low conductivity regions of shale formation. To alleviate the computation burden, many Model Order Reduction (MOR) methods have been developed such as the multi-scale finite element [8], upscaling method and model order reduction (MOR)[9, 10] etc. The family of MOR can be categorized into two types: the non-intrusive (do not require revision of reservoir simulation code) and intrusive methods. In the realm of non-intrusive method, data-driven methods, such as sparsity-promoting techniques and neural networks, are the most popular choice for material balance type modeling [11]. For the intrusive methods, which requires modifying code in the reservoir simulator, model order reduction with Galerkin projection has been used in the dynamic system, such as balanced truncation, POD (proper orthogonal decomposition), TPWL (trajectory piecewise linear method), bilinear Krylov subspace methods and quadratic bilinear model order reduction [12]. For example, POD as one of the traditional MOR method, can generate modes purely from data, but the application of the modes has to be combined with Galerkin projection, which means the POD method cannot be used without a reservoir simulator. On the other hand, the DMD method, as one of the data-driven

method, is able to perform forward modeling with only calculated basis and corresponding eigenvalues. This is one of the advantages of DMD over POD. Many of the MOR method including POD and DMD are based on snapshots from numerical simulation results, which is the so-called snapshot matrix. Although the snapshot matrix contains coherent structure of the dynamic system, it is not ideal to use as basis or modes for constructing subspace because of its possible linear dependence. To compensate that, the POD performs a singular value decomposition and selects high energy eigenvectors as basis, whereas the DMD performs eigen-decomposition on the linear operator that relates two sequential snapshot matrices.

For any MOR method, to further reduce the number of basis, techniques beyond decomposition needs to be considered. For POD mode, the so called DEIM (discrete empirical interpolation method) [13] was developed to use a greedy algorithm to select few grid blocks for nonlinear function evaluation and thus project the system onto a even smaller subspace. While to reduce the DMD modes, the sparsity promoting technique has been applied where a l_1 penalty term is introduced for creating a convex problem that has sparse solution. The sparsity promoting DMD is one of the topics in this work, and we will discuss it later.

Although MOR has ways of further reducing its modes, we are still interested in a simulation-free method to perform forward modeling. Discovering the underling governing equation with the aid of raw data provides a possibility of mitigating the overall computational cost of MOR steps. To this end, the methodologies we will investigated in this work, will be able to determine the conservation equations directly from data, that is,

the nonlinear ODE is transformed into a large linear system problem by building a library with all possible nonlinear terms precomputed and the only unknown becomes the coefficients associated with the nonlinear terms. Finally, the coefficients can be solved by various sparse regression methods, which we will show in this work. With the evolution of machine learning and the more and more problems in oil and gas industry proved solvable using this approach, another alternative to MOR is the end to end neural network forward modeling for production prediction

1.2. Dynamic Mode Decomposition

Usually nonlinear fluid flow in heterogeneous porous media is governed by infinite dimensional PDE, which are spatially discretized into a large dimensional ODE system of equations by methods such as finite elements, finite differences and finite volumes. Despite the fact that the flow is of nonlinear large dimension, the system can often be approximated with models of low complexity because of the coherent structure in the flow. To identify the coherent structure, model order reduction (MOR) method such as POD, POD-DEIM, balanced truncation and DMD method has been developed. The DMD method has gain popularity in fluid modeling industry field given its practical use and solid theoretical background connected with other methodology such as Arnoldi method [14], POD, spectral analysis of Koopman operator [15], and Fourier analysis [16]. The Koopman operator, as the heart of DMD method, is a linear operator whose modes and eigenvalues could be used to completely describe the nonlinear dynamic system. The modes will identify the coherent structure of the dynamics, while the eigenvalues will

specify the decay rate and frequency of each corresponding mode. In fact, it's just because DMD is a numerical approximation to the Koopman spectral analysis, it could be applied to nonlinear systems. As the recent evolution of artificial intelligence, Koopman operator, as a crucial but hard to approximate infinite-dimensional operator, has been re-investigated by several researchers using deep learning neural network, dictionary learning and autoencoders [17-19]. Also, the family of DMD methods has been growing to accommodate different situations, for example, the extended DMD [19] was developed to approximate the Koopman operator with dictionary learning, DMD with control (DMDc) [20] for including the effect of control and disambiguate the underlying dynamic and the effect of actuation.

For DMD, the essential idea is to describe the flow with superposition of a set of vectors, also called modes. This way the numerical simulator is not necessarily to be used when approximating the solution. Another way of utilizing the modes is to combine with Galerkin projection, which will project the large dimensional ODE system onto a smaller subspace spanned by the modes to ease the complexity [3, 21] .

Both of POD and DMD method are snapshot based, post processing method that can extract coherent structure from snapshot data，which means they rely on a two-step process called off- and online. In the offline step, one deals with gathering snapshots and computing the projection basis or modes, which are computed through an SVD decomposition. In the online step, one performs dimensionality-reduction of the states of the system, based on the highly ranked modes, and implement different boundary conditions as used in the offline step. However, Unlike the POD method, it is hard for

DMD method to choose a subset of the modes since the modes are not ranked by energy and it is even harder considering the modes are non-orthogonal. We cannot simply select modes with the largest amplitude or largest singular value. Several methods including optimal DMD and gradient based DMD [16, 22, 23], have been proposed to solve the issue. Traditional DMD select modes based on the rank of the singular value matrix. However, the rank is usually approximately equal to the smaller dimension of the snapshot matrix, which is not an impressive reduction in dimension. To filter for a less number of DMD modes, [24] proposed the sparsity promoting DMD method which imposes a sparse structure on the DMD modes.

In this work, we focus on the sparsity-promoting DMD method for choosing appropriate DMD modes. Sparsity-promoting DMD uses a least square deviation norm plus an extra l_1 norm that penalizes on the number of non-zero basis vectors as a convex optimization function, to which the solution consists sparsity structure of DMD modes. Note that l_1 regularization is widely used and proved efficient in other fields of study such as statistics [25] and geophysical inversion [26].The purpose is to seek a tradeoff between the number of non-zero DMD basis and the accuracy of the approximation. l_1 norm instead of l_0 norm (cardinality function) is used in order to keep the objective function convex [27].Then, the ADMM algorithm, which is efficient for solving large scale dynamic system, is applied to solve the convex problem.

Although the DMD and sparsity-promoting DMD are data driven methods that enables forward calculation based on selected modes, it is still insufficient to determine

the governing equation behind the data. We next propose to a novel approach to draw the connection between the output data and governing ODE system.

1.3. System Identification

Data driven methods for discovering the dynamics of a system has long been studied, including model order reduction, prediction of the controls of complex systems, and the newly developed technique to link the underlying natural laws with data, which we will explore in detail in this work. The underlying physical equations for modeling fluid flow is usually a system of ODEs that involves state variables changing in both time and space. Based on the assumption that there are only a few terms in the ODEs that governs the dynamic [28], we can identify them with sparse-promoting techniques and measurement data. Such assumption could be useful in the petroleum industry especially when it comes to proxy modeling for complicated physical based equations such as geomechanics coupled flow for unconventional reservoir [29, 30] and compositional simulation for multiphase flow reservoir. An important progress of the innovative approach for discovering nonlinear dynamic system structure is raised by [31]. They use a symbolic regression with generic programming [32] for finding the specific variables, constants and mathematical operators. This an innovative breakthrough, although this method could be expensive for multi-dimensional large system. Later on, this idea of distilling analytic equation with data is combined with the statistical concept of sparsity and compressed sensing by many researchers [28, 33].

Similar to compressed sensing [34], which is frequently used in image and signal processing to extract several basis to recover the full signal with very few measurements, the sparse regression method selects basis from a pre-defined library to approximate the dynamic system. The method does not require prior knowledge on the structure of the equation or information of any properties of the reservoir, and the only assumption made is that the ODE system can be represented with only a few basis functions.

Several methods have been proposed to solve for a sparse structure, one of the popular method is Lasso [35, 36]. Although Lasso has many forms of variations, such as elastic net and grouped lasso, the basic form is a least square error term plus a regularization term that penalizes on the number of nonzero coefficients of the linear regression. Compared with l_2 regularization that gives an all nonzero coefficient and l_0 norm that makes the problem non-convex, the l_1 regularization formulates a solvable convex and sparse solution. Many algorithms have been developed to solve the system of equations coming from Lasso, such as gradient decent, coordinate decent, and alternating direction of multipliers (ADMM), which will be used in this paper. Another sparse regression method we will explore here is the SINDy (Sparse identification of nonlinear dynamical systems), first proposed by Brunton et al. in 2015. Compared with Lasso, their method can be more suitable for large dimensional system due to its simplicity in calculation. The method uses a direct solver for linear regression and impose a hard threshold on the solution for sparse structure. Inspired by Rudy et al. [37], in this work, we modify the SINDy with a ridge regression and hard threshold for stability purpose.

Both SINDy and Lasso, as a sparse promoting technique, relies on crucial parameters for model selection. The SINDy hard threshold is a crucial parameter that will be obtained through a cross validation algorithm from machine learning. For Lasso, the parameter before the penalty term needs to be determined for obtaining a solution with fewest nonzero expansion coefficients with reasonable accuracy.

In the sense of equation-free and selection of few active basis, the sparse regression method is similar with the previously mentioned sparsity promoting DMD method. They both seek a sparse structure from a large library of basis by a convex optimization problem of a least square deviation plus a l_1 norm penalty terms. But, bearing in mind the variety of model reduction methods, we hope to find an entry point to apply MOR to the discovering ODE technique in the future. In addition, the prediction of complex control (e.g. the injection/ production profile) can be another problem to be handled by the sparse regression method with some alterations in the nonlinear term library.

So far, we have discussed the two data-driven techniques that could be used as a proxy in reservoir modeling. The final topic in this work is another machine learning based end to end proxy model that directly provide production prediction based on different control parameters.

1.4. Proxy Model by Recurrent Neural Network

Forecasting oil production is a pertinent task in petroleum industry. Many methods have been developed and to estimate the production rate in an accurate and fast manner, including decline curve analysis [38], well test analysis [39], (proxy model for) reservoir

simulation for conventional and unconventional reservoir [40-47]. For real field production prediction, the challenge lies in the noise and missing data and anomalies that could happen in the field. For synthetic data, it may involve large dimensions for the reservoir simulation, and highly nonlinearity and heterogeneity that involved in the simulation process [48, 49]. Even with proxy model or model reduction technique for reservoir simulation, there is chance that as the control parameter changes, its hard or requires special handling to approximate the production rate for the new dynamic system.

As the production rate is a time series process, there are plenty of techniques that could be used for prediction, which mainly can be categorized in two areas: statistical approaches and soft computing approaches [50]. The statistical approaches includes traditional linear models such as AR, MA and ARIMA [51] and some nonlinear time series models such as ARCH (autoregressive conditional heteroscedastic)[52] and threshold autoregressive model [53]. However, those methods usually fail to incorporate the physics behind the oil producing process. Another well-known parametric statistical approach is called decline curve analysis (DCA), which is essentially based on the Arp's equation. DCA has been widely used in the industry to model production depletion in the reservoir due to its model simplicity. However, the same problem persists, the model is questionable to describe real world, noisy, complex process in highly heterogeneous reservoir.

The other area is the soft computing approach, which involves data mining and machine learning technique. Liu and Horne have proposed to use the convolutional kernel and feature based approach to interpret pressure and flow rate with downhole pressure

gauge data [39]. Similarly, Ma and Liu presented an extended Arp's decline curve analysis that uses kernel trick to build a nonlinear multivariate prediction model [54]. Going further in machine learning, the neural network approach including ANN (artificial neural network) and RNN (recurrent neural network) has been used in the past two decades. The ANN algorithm, as a feed forward neural network, the information travel through the network in only one direction. RNN on the other hand, is ANN with recurrent connection and mainly used for modeling a recursive process. Sequences of inputs are processed and stored as the internal state of the each of the connected memory cells. The updates of the new output or internal state will be relied on both current and previous states through delicate gates. In addition, RNN is shown to have a stronger model capability that can extract time series data pattern without features provided, thus under the circumstances that feature design is difficult, RNN is preferred over the feature-based machine learning [55].

1.4.1. Long short term memory

As pointed out before, the standard RNN has the major weakness of preserving the long-time dependencies. Modifications and extensions based on the original RNN has been made including GRU, bi-directional RNN, LSTM to overcome these disadvantages. Among those extensions, the LSTM (long short term memory) algorithm, first proposed by Hochreiter and Schmidhuber in 1995 [56], surpasses others and shows promising results for long-range time dependencies by providing improvement on the structure of memory gates.

However, even with LSTM, shallow neural network is usually not sufficient for modeling complex systems. For that reason, deep learning neural network is preferred to deal with highly nonlinear, long term time series data with complicated features [57]. Two popular deep neural network that has demonstrate their capability is CNN (convolutional neural network) and LSTM, or the combination of them. Deep convolutional neural network is extensively used for computer vision areas such as self-driving cards, traffic monitoring, face recognitions, etc. In oil industry, it is also seen to be used in seismic reservoir characterization, facies recognition[58], well schematic digitalization [59]. Deep LSTM is commonly used in the area of natural language processing such as speech recognition, machine translation, sentiment classification etc. In the petroleum industry, it is used for various types of time series problem: drilling and pumping surface pressure response[60], well testing [55], well logging [61], and gas/oil production prediction [62].

1.4.2. Long Short Term Memory with Ensemble Kalman Filter

Further study has proved that the LSTM combining with Bayesian method is capable of improving the network performance [63]. Method such as the Ensemble Kalman Filter (EnKF) or Extended Kalman Filter (EKF), can be used together with LSTM. In the past, ensemble method has been successfully applied to areas such as history matching for reservoir characterization [64, 65], drilling anomaly activity detection, petrophysical characteristic [66]. In this work, we will focus on the cascaded deep LSTM application on oil production prediction for multiphase fluid flow reservoir and further

improve the model with EnKF. The model will also integrate the essential physical fact in a typical waterflooding reservoir.

1.5. Research Objective and Scope

As stated before, the objective of this research is to enhance the efficiency of reservoir simulation by taking into account (1) sparsity into the reduced-order projection modes; (2) ODE discovering methods for porous media flow; and (3) multivariate time series method for production prediction. In particular,

1. We seek a way to reduce the DMD modes to create a smaller subspace for Galerkin projection.
2. We develop a simulation-free method to alleviate the computational cost by building a connection between raw experiment/simulation data with its underlying ODE
3. To accurately predict the complex control problem, we develop a simulation free method (RNN).

To accomplish these objectives, we complete the following tasks in each of the chapter

Chapter 1: Model reduction based on Sparsity-Promoting DMD

The aim of this task is to apply the sparsity DMD method to our single phase and two phase fluid flow simulator and, in the meantime, compare the relative error and efficiency with the two other reduced order modeling method, namely POD and standard DMD.

Chapter 2: Discovering the governing ODE

In this task, we attempt to obtain the governing ODE by solving a large-scale linear system with sparse regression. The two sparse regression algorithms used are SINDy (sparse identification of nonlinear dynamics) and Lasso. The linear system composed of suspected nonlinear ODE terms and its associated coefficients

Chapter 3: Dealing with complex controls time series problem with Recurrent Neural Network

We developed an end to end production prediction model by identifying the connection between control parameter and production response and developed a forward model for oil production prediction using RNN.

2. RESERVOIR SIMULATION*

2.1. General Multiphase Flow Reservoir Simulation

Flow and transport models in porous media can be described by a set of partial differential equations representing conservation of mass, momentum and energy as functions of pressure, saturations and temperature, and reconciled by the equation of state. In what follows, we describe briefly the single- and two-phase flows models used in this paper.

As discussed in [67, 68], to simplify the problem we neglect the inertial effects and assume the flow to be isothermal. Hence, one can assume the black-oil formulation, where there are two components (oil-water) and there are two phases of the hydrocarbon substance (oil and gas) present in the reservoir.

For the case of two-phase oil-water system in a reservoir domain (denoted byΩ), the mass balance equation for each phase is

$$\nabla \cdot (\rho_\chi \mathrm{v}_\chi) + \frac{\partial(\rho_\chi \varphi s_\chi)}{\partial t} - \rho_\chi q_\chi = 0 \quad in \quad \Omega, \quad \chi \in \{o, w\}, \tag{2.1}$$

where ρ_χ is the fluid phase density, v_χ is the fluid phase superficial velocity, $t.$ is time, $\nabla \cdot$ denotes the divergence operator, $\varphi.$ is the porosity of the rock, $s_\chi.$ denotes fluid phase

*Part of this section is reprinted with permission from "Data-driven Model Reduction Based on Sparsity-promoting Methods for Multiphase Flow in Porous Media" by A. Bao, E. Gildin. SPE Latin America and Caribbean Petroleum Engineering Conference, Copyright [2017] by Society of Petroleum Engineers

saturation, q_χ is the volumetric source/sink term and subscript $\chi \in \{o, w\}$ indicates the oil and water phases, respectively. We apply Darcy's law,

$$\mathrm{v}_\chi = \frac{k_{r\chi}}{\eta_\chi} K \cdot (\nabla P_\chi - \rho_\chi g \nabla h) \quad in \ \ \Omega, \tag{2.2}$$

where ∇ is the gradient operator, K denotes the permeability tensor, η_χ is the fluid phase viscosity, $k_{r\chi}$ denotes the relative permeability of each phase (which is a function of water saturation), P_χ is the phase pressure, g is the constant of gravity acceleration and h denotes depth. Combining Eq. 2.1 and Eq. 2.2 yields,

$$\nabla \cdot \left[-\frac{\rho_\chi k_{r\chi}}{\eta_\chi} K \cdot (\nabla P_\chi - \rho_\chi g \nabla h) \right] + \frac{\partial (\rho_\chi \varphi s_\chi)}{\partial t} - \rho_\chi q_\chi = 0. \tag{2.3}$$

The general oil-water model is completed by enforcing the saturation constraint $s_o + s_w = 1$ and by specifying a capillary pressure relationship $P_c(s_w) = P_o - P_w$. As can be seen, Eq. 2.3 is nonlinear. We take P_o and s_w to be the primary unknown variables, from which P_w and s_o can be easily computed. Eq. 2.3 can be solved numerically using a fully-implicit finite volume procedure. More details can be found in Aziz and Settari [69].

With the fully-implicit procedure, each grid block is characterized by two primary state variables, oil pressure P_o and water saturation s_w. Defining $x = [P_o, s_w]$ as the state vector for all reservoir cells and $\boldsymbol{v}$ as the well control parameters, one can write Eq. 2.3 in the fully implicit form as

$$R(x, x^n, c^{n+1}) = F(x) + A(x, x^n) + Q(x, c^{n+1}) = 0, \tag{2.4}$$

where R is the residual vector, superscripts n and $n+1$ indicate time levels, $F(x) = T(x)x$ represents the transmissibility term, with $T(x)$ being the transmissibility matrix, $A(x, x^n) = -D(x)(x - x^n)$ is the accumulation term, and $Q(x,\ c^{n+1})$ is the source/sink

term. The source/sink term corresponding to injection/production from a well is modeled by using a well equation as described by Peaceman in 1983 [70]

$$-q_\chi = WI\lambda_\chi(P_\chi - P_{wb}), \tag{2.5}$$

where q_χ is the well flow rate of phase χ, $\lambda_\chi = \rho_\chi k_{r\chi}/\eta_\chi$ is the mobility of phase, P_χ is the computed pressure of phase χ in the reservoir cell, P_{wb} is the wellbore pressure, and WI is the well index which in this paper is based on the Peaceman model. In this paper, bottom hole pressure (BHP) is chosen as the well control parameter c.

The goal of a simulation time step is to find the next state x^{n+1} in Eq. 2.4 such that

$$R(x^{n+1}, x^n, c^{n+1}) = 0. \tag{2.6}$$

Eq. 2.6 is nonlinear. Typically, Newton-Raphson iteration method is used to solve Eq. 2.6, where x^{n+1} is obtained iteratively as follows,

$$x^{n+1,k+1} = x^{n+1,k} - J(x^{n+1,k})^{-1}R(x^{n+1,k}, x^n, c^{n+1}). \tag{2.7}$$

In Eq. 2.7, J is the Jacobian matrix and is defined as $J(x) = \partial R/\partial x$. Iteration for solving Eq. 2.7 will be stopped when the relative norm of residual is close to zero. Given the size of the Jacobian matrix and the computational cost associated with solving the linear system in the Newton Raphson algorithm, model order reduction described in the next sections can alleviate some of these bottlenecks by reducing the size of these matrices.

2.2. Single-Phase Flow

In the case of single-phase flow, the equations are simplified and the mass balance equation can be given by a single equation in pressure. The governing parabolic partial

differential equation for single-phase flow in porous media arising from Eq. 2.3 is well familiar as

$$\frac{\partial(\varphi\rho)}{\partial t} = \nabla \cdot \left(\frac{\rho}{\mu} k(\nabla P - \rho g \nabla \mathrm{z})\right) - \rho \tilde{\mathrm{q}}, \tag{2.8}$$

where $\int_{\Omega} \rho\tilde{\mathrm{q}} d\Omega = q$ is the source or sink term.

The left hand side represents the accumulation term, while on the right hand side, the first term is representing flux from Darcy's equation and the second term is source/sink term.

After finite difference discretization in space, the PDE reduces to an ODE system

$$B_p \frac{dP(t)}{dt} = T_p\big(P(t)\big)P(t) - q\big(P(t)\big) - G\big(P(t)\big) = T_p\big(P(t)\big)P(t) - F_P\big(P(t)\big). \tag{2.9}$$

Here t denotes time, $P(t) = [P_1(t), \dots, P_{nc}(t)]^T \in \mathbb{R}^{nc}$ is a vector of pressure solutions in each grid block, $B_p = \frac{\partial(\phi)}{\partial P} + \frac{\partial(\rho)}{\partial P} \in \mathbb{R}^{nc \times nc}$ is a coefficient matrix , F_P accounts for the source/sink term and gravity term and is a function of pressure, i.e., $F_P = [F_P(P_1(t)), \dots, F_P(P_{nc}(t))]^T$. $T_p \in \mathbb{R}^{nc \times nc}$ is transmissibility matrix, which is seven banded diagonal matrix for 3D (five banded for 2D, 3 banded for 1D), e.g. $(T_o)_{i\pm1/2,j,k} = (\frac{kk_{ro}A}{\mu_o B_o \Delta x})_{i\pm1/2,j,k}, (T_o)_{i,j\pm1/2,k} = (\frac{kk_{ro}A}{\mu_o B_o \Delta y})_{i,j\pm1/2,k}, (T_o)_{i,j,k\pm1/2} = (\frac{kk_{ro}A}{\mu_o B_o \Delta z})_{i,j,k\pm1/2}$. B_p T_p, F_P are all function of pressure.

To solve this equation, we perform backward Euler scheme for time discretization. Since F_P could be nonlinear function, we solve for P^{k+1} with newton iterations at each time step with the residual and Jacobian defined in Eq. 2.10 and 2.11.

$$R = \frac{B_p\left(P^{k+1}(t) - P^k(t)\right)}{\Delta t} + F_P\left(P^{k+1}(t)\right) - T_p(P^{k+1}(t))\, P^{k+1}(t) \tag{2.10}$$

$$J = \frac{J_B\left(P^{k+1}(t) - P^k(t)\right)}{\Delta t} + \frac{B_p}{\Delta t} + J_F - J_T\, P^{k+1}(t) + T_p(P^{k+1}(t)) \tag{2.11}$$

Here J_B and $J_T \in \mathbb{R}^{nc \times nc}$ are matrix derivative in terms of pressure, J_F is diagonal matrix given by $J_F = diag\{F_P{}'(P_1(t)), \dots, F_P{}'(P_{nc}(t))\} \in \mathbb{R}^{nc \times nc}$, where $F_P{}'$ is derivative of function F_P in terms of pressure.

2.3. Mixed Formulation for the Two-Phase Flow

In this work, we use the two-phase flow system a mixed formulation of the equations [71]. This is incompressible two-phase flow model combining Darcy's law with mass conservation law as shown below. Here we neglect the gravity effect and capillary pressure. Simplified from Eq. 2.1 and 2.2, we have

$$\mathrm{v}_\chi = -\frac{k_{r\chi}}{\eta_\chi} K \nabla P, \tag{2.12}$$

$$\text{and } \varphi \frac{\partial s_\chi}{\partial t} + \nabla \cdot \mathrm{v}_\chi = q_\chi\,. \tag{2.13}$$

In Eq. 2.12, the subscript χ refer to phase water or oil, v_χ is phase velocity, $k_{r\chi}$ is relative permeability, η_χ is phase viscosity, K is permeability tensor matrix and P is pressure. In Eq. 2.13, φ is porosity, s_χ and q_β are phase saturation and phase low rate respectively.

With the condition $s_w + s_o = 1$ and manipulate on Eq. 2.12 and 2.13 we have the following governing equation for pressure and saturation, assuming no flow boundary

condition and constant initial condition for saturation ($s_w(t=0)=s_0$). For pressure and velocity, the governing equation appears

$$\nabla \cdot \mathrm{v} = q_w + q_o = q \tag{2.14}$$

$$\text{and } -\lambda K \nabla P = \mathrm{v}, \tag{2.15}$$

with the boundary condition $\mathrm{v} \cdot n = 0$, n is the outward norm on the boundary. $\lambda = \lambda_w + \lambda_o = \frac{k_{rw}}{\mu_w} + \frac{k_{ro}}{\mu_o}$ is the total mobility. We define two function spaces $Q = \{l \in L^2(\Omega), \int_\Omega l \, \mathrm{dx} = 0\}$ and $W = \{w \in \mathrm{H(div}, \Omega), \mathrm{w} \cdot \mathrm{n} = 0 \text{ on } \partial\Omega\}$, where $H(div, \Omega) = \{\mathrm{w} = (w_1, \ldots, w_d) \in (L^2(\Omega))^d, \nabla \cdot \mathrm{w} \in L^2(\Omega)\}$.

Governing equation for saturation is of the form

$$\varphi \frac{\partial s_w}{\partial t} + \nabla \cdot (f_w \mathrm{v}) = q_w, \tag{2.16}$$

where $f_w(s) = \frac{\lambda_w}{\lambda} = \frac{k_{rw}(s)}{k_{rw}(s) + \frac{\mu_w}{\mu_o} k_{ro}(s)}$ is the fractional flow function.

For pressure and velocity, mixed finite element method [72, 73] is used for discretization. We seek a pair of (P, v) that belongs to $Q \times W$, where w and l are test functions, such that

$$\int_\Omega (\lambda K)^{-1} \mathrm{v} \cdot w \; dx - \int_\Omega P \nabla \cdot w = 0, \; w \in W \tag{2.17}$$

$$\int_\Omega l \, \nabla \cdot \mathrm{v}, = \int_\Omega l \, q \, dx, \; l \in Q. \tag{2.18}$$

For detailed definition of functional spaces for the mixed FEM, refer to [68]. Now the problem becomes finding $P_h \in Q_h$ and $\mathrm{v}_h \in W_h$ such that the following weak form is true,

$$\int_\Omega (\lambda K)^{-1}\mathrm{v}_h \cdot w_h \, dx - \int_\Omega P_h \nabla \cdot w_h = 0, \; w_h \in W_h \tag{2.19}$$

$$\int_\Omega l_h \nabla \cdot \mathrm{v}_h dx = \int_\Omega l_h \, q \, dx, \; l_h \in Q_h, \tag{2.20}$$

where Q_h and W_h are mixed finite element space that contains lower order piecewise polynomials.

For discretization, let $\{\psi_{\mathrm{v}i}\}$ and $\{\psi_{Pi}\}$ be basis for W_h and Q_h, thus v and P are approximated as $\mathrm{v}_h = \sum \psi_{\mathrm{v}i}\mathrm{v}_i$ and $P_h = \sum \psi_{Pi}P_i$. Then the system of discretized equation is of the form,

$$\begin{bmatrix} B & -C^T \\ C & 0 \end{bmatrix} \begin{bmatrix} \mathrm{v} \\ P \end{bmatrix} = \begin{bmatrix} 0 \\ D \end{bmatrix}, \tag{2.21}$$

where $B = \{b_{ij}\}$, $C = \{c_{ik}\}$, $D = \{d_k\}$, $b_{ij} = \int_\Omega \psi_{\mathrm{v}i}(\lambda K)^{-1} \psi_{\mathrm{v}i} dx$, $c_{ik} = \int_\Omega \psi_{Pi} \nabla \cdot \psi_{\mathrm{v}i} \, dx$ and $d_k = \int_\Omega \psi_{Pi} q \, dx$. For more discussion on multiscale method with different local basis functions, refer to [72].

We note that the whole system is solved sequentially, which means to solve pressure first and then use the results to solve for saturation. For saturation, finite volume is used for space discretization and backward Euler is used for time derivative part.

For grid block *i*, we perform discretization in time for Eq. 2.16, and thus, we have the following expression (use to represent s_w)

$$s_i^{n+1} = s_i^n + \frac{\Delta t}{\varphi_i} q_w - \frac{\Delta t}{\varphi_i} \nabla \cdot (f_w(s_i) \mathrm{v}_i). \tag{2.22}$$

Applying finite volume scheme and divergence theorem (Here Ω_i is the domain of cell *i*) yield,

$$\nabla \cdot (f_w(s_i)\mathrm{v}) = \frac{1}{|\Omega_i|} \int_{\Omega_i} \nabla \cdot (f_w(s_i)\mathrm{v}_i)\mathrm{dx} = \frac{1}{|\Omega_i|} \int_{\partial\Omega_i} f_w(s_i)\mathrm{v}_i \cdot \mathbf{n}\, \mathrm{dS}$$

$$= \frac{1}{|\Omega_i|} \sum_j \int_{e_{ij}} [f_{ij}(s)\mathrm{v}_{ij}] \cdot n_{ij} dS = \sum_j F_{ij}(s^{n+1}), \tag{2.23}$$

where $F_{ij} \approx \int_{e_{ij}} [f_{ij}\mathrm{v}_{ij}] \cdot n_{ij} dx$ is the phase flux integration on edge e_{ij} between two adjacent cell domains Ω_i and Ω_j. The integral is approximated with a first order approximation, the upstream weighting scheme, which is

$$f_{ij}(s) = \begin{cases} f_{ij}(s_i), & if\ \mathrm{v}_{ij} \cdot n_{ij} \geq 0 \\ f_{ij}(s_j), & if\ \mathrm{v}_{ij} \cdot n_{ij} < 0 \end{cases}.$$

Rewriting the flux term with two parts,

$$(q_w)_i = q_i^+ - f_w(s_i^{n+1}) q_i^-, \tag{2.24}$$

where $q_i^+ = \max(q_i, 0)$ indicating the injection rate and $q_i^- = \min(q_i, 0)$ indicating the production rate. We have

$$s_i^{n+1} = s_i^n + \frac{\Delta t}{\varphi_i |\Omega_i|} \left[q_i^+ - \sum_j F_{ij}(s^{n+1}) + f_w(s_i^{n+1}) q_i^- \right]. \tag{2.25}$$

In the following chapters, we will utilize the single-phase fluid model (Eq. 2.8-2.11) and two-phase fluid model (Eq. 2.12-2.25) described here to perform model order reduction and proxy model development.

3. SPARSITY PROMOTING MODEL ORDER REDUCTION*

The sparsity promoting method developed here can be used in both MOR and proxy model development as both process involves data mining techniques to either figure out the optimum set of modes (for MOR) or mining for the best terms in the governing equation for the dynamic system, which we will discuss in next chapter.

We first introduce the problem of PDE governing equations and discretization for both single phase and two-phase flow. Then provide the two standard MOR algorithms (POD, standard DMD) and one newly developed sparsity DMD algorithm for constructing the required basis. Finally, the results of application on sparsity DMD is presented followed by a comparison and discussion between the three above mentioned model order reduction methods.

3.1. Galerkin Projection

For single phase flow, the nonlinear ODE differential equation (Eq. 2.9) could be solved by Newton-Raphson iteration at each time step involving *nc* by *nc* system of equations. We use Galerkin projection theory to reduce the dimensionality of the system as $P(t) = \Phi P_r(t)$, where $\Phi^* \Phi = \mathrm{I}_k,\ \Phi = \{\phi_i\}_{i=1}^{r} \in \mathbb{R}^{nc \times r}$ is the matrix contains basis vectors. Here the operator $*$ on a matrix means conjugate transpose of the matrix, which

*Part of this section is reprinted with permission from "Data-driven Model Reduction Based on Sparsity-promoting Methods for Multiphase Flow in Porous Media" by A. Bao, E. Gildin. SPE Latin America and Caribbean Petroleum Engineering Conference, Copyright [2017] by Society of Petroleum Engineers

means take the transpose of the matrix and then take the complex conjugate of each entry. The basis vectors are also called modes for POD and DMD method. The linear combination of modes can be used either to construct subspaces or perform prediction. Note that for POD method, the column vectors are orthogonal to each other. Thus, the reduced system is of the form,

$$\Phi^* B_p \Phi \frac{d}{dt} P_r(t) + \Phi^* F_P(\Phi P_r(t)) = \Phi^* T_p \Phi P_r(t) \tag{3.1}$$

$$\Phi^* \Phi \left({P_r}^{k+1}(t) - {P_r}^{k-1}(t) \right) + \Delta t \Phi^* F_P \left(\Phi \, {P_r}^{k+1}(t) \right) = \Delta t \Phi^* \Phi {P_r}^{k+1}(t) \tag{3.2}$$

$$R_r = \Phi^* R \tag{3.3}$$

$$J_r = \frac{\partial R_r}{\partial P_r} = \Phi^* \frac{\partial \mathrm{R}}{\partial \mathrm{P}} \frac{\partial \mathrm{P}}{\partial P_r} = \Phi^* J \Phi \, . \tag{3.4}$$

In two-phase flow model, we will be using Eq. 2.22 as our target equation and apply Galerkin projection with basis $\Phi_{\mathrm{v},P} \in \mathbb{C}^{(n\mathrm{v}+nc)\times r}$ for velocity and pressures. Thus, the Galerkin projection on to the subspace spanned by the basis $\Phi_{\mathrm{v},P}$ is given by

$$\Phi_{\mathrm{v},p}^T \begin{bmatrix} B & -C^T \\ C & 0 \end{bmatrix} \Phi_{\mathrm{v},P} \begin{bmatrix} \mathrm{v}_r \\ P_r \end{bmatrix} = \Phi_{\mathrm{v},p}^T \begin{bmatrix} 0 \\ D \end{bmatrix} . \tag{3.5}$$

3.2. Standard Technique for Model Order Reduction

3.2.1. Proper Orthogonal Decomposition

The POD method is derived from constructing a lower dimensional approximation in Hilbert space [21]. Basically, it can be viewed as an application of singular value decomposition in a finite dimensional space (usually Euclidean space). The idea was first

proposed by Volkwein [74] and followed by a number of scholars researching on modifications and improvement [13, 75]. We will briefly review the method here.

Consider given a set of pressure snapshots at different time steps $\mathbb{P} = [P^1, \dots, P^{ns}]^T \in \mathbb{R}^{nc \times ns}$, $P^j = P(t_j), d = \dim(span\{P^1, \dots, P^{ns}\})$, where *ns* is the number of time steps and *nc* is the number of grid blocks. Let $\{\phi_i\}_{i=1}^d$ denotes the orthonormal basis, then each of the snapshot column can be expressed as

$$P(t_j) = \sum_{i=1}^{d} \langle P^j, \phi_i \rangle_X \phi_i. \tag{3.6}$$

The goal is to find the corresponding basis $\{\phi_i\}_{i=1}^l$ with l<<d to describe the snapshot data by solving the minimization problem

$$\min_{\{\phi_i\}_{i=1}^r} \sum_{j=0}^{n_s} \xi_j \left\| P^j - \sum_{i=1}^{l} \langle P^j, \phi_i \rangle_X \phi_i \right\|_H, \tag{3.7}$$

$$\text{Subject to } \langle \phi_j, \phi_i \rangle_X = \delta_{ij},$$

where $\{\xi_j\}_{j=0}^{ns}$ are positive coefficient and *H* denotes a proper Hilbert space.

The solution of the above minimization problem is obtained from singular value decomposition of the snapshot matrix [76]. For simplification purpose, we perform economy SVD in Euclidean space as

$$\mathbb{P} = V \Sigma W^*, \tag{3.8}$$

where $V = \{v_i\}_{i=1}^d \in \mathbb{R}^{nc \times d}$ and $W = \{w_i\}_{i=1}^d \in \mathbb{R}^{ns \times d}$ are unitary matrix, $\Sigma = \text{diag}\{\sigma_1, \dots, \sigma_d\} \in \mathbb{R}^{d \times d}$ contains singular values of snapshot matrix. The solution for optimal POD basis is simply the first l columns in V, which is$\{v_i\}_{i=1}^l$.

The error in corresponding space is

$$\min_{\{\phi_i\}_{i=1}^{r}} \sum_{j=0}^{ns} \xi_j \left\| P^j - \sum_{i=1}^{l} ((P^j)^T \phi_i)\phi_i \right\|_2^2 = \sum_{i=l+1}^{d} \sigma_i^2. \quad (3.9)$$

This error formula indicates a factor called energy fraction (Eq. 3.10) that can be used to improve the performance of POD basis by choosing the truncation parameter l so that Energy Factor (EF), defined below (Eq. 3.10), is close to one.

$$EF = \frac{\sum_{i=1}^{l} \sigma_i^2}{\sum_{i=1}^{d} \sigma_i^2} \quad (3.10)$$

3.2.2. Dynamic Mode Decomposition

Dynamic mode decomposition (DMD) has been proposed as a non-intrusive method for discovering system dynamics [14]. It can be a non-intrusive method from the simulator-run point of view. In this case, we call it an offline process. The major assumption of the method is that there exists a linear mapping matrix A that connects the temporal evolution of the state of the system. Without elaborating further in the theory of DMD, we provide the steps used in the DMD algorithm. The reader can refer to [14] for more details.

1. Suppose we have two snapshot matrices comprised of the states of the system. They can be pressures for the single-phase flow case, or pressures, velocities and saturations for the two-phase flow. The columns of the matrix data are captured equal-spaced in time, with a time step Δt.

$$\mathbb{P}_{\mathbf{1}} = \begin{bmatrix} | & | & & | \\ P^1 & P^2 & \dots & P^{ns-1} \\ | & | & & | \end{bmatrix}$$

$$\mathbb{P}_{\mathbf{2}} = \begin{bmatrix} | & | & & | \\ P^2 & P^3 & \dots & P^{ns} \\ | & | & & | \end{bmatrix}$$

Each $P^i = P(i\Delta t)$ is a vector with *nc* components (*nc* gridlocks), and thus $\mathbb{P}_1, \mathbb{P}_2 \in \mathbb{R}^{nc\times(ns-1)}$. Ideally, we assume there is a time independent matrix *A* that satisfy

$$P^{i+1} = AP^i. \tag{3.11}$$

And thus

$$\mathbb{P}_2 = A\mathbb{P}_1. \tag{3.12}$$

Also notice that this assumption potentially gives us a Krylov subsequence since

$$\mathbb{P}_{\mathbf{1}} = \{P^1, AP^1, A^2P^1 \dots A^{ns-2}P^1\}.$$

Next, to get an approximation of the matrix *A*, we perform an economy-sized singular value decomposition on snapshot matrix $\mathbb{P}_{\mathbf{1}}$.

$$\mathbb{P}_{\mathbf{1}} = YS_\mu Z^* \tag{3.13}$$

Here $Y = \{y_i\}_{i=1}^r \in \mathbb{R}^{nc\times r}$ and $Z = \{z_i\}_{i=1}^r \in \mathbb{R}^{ns\times r}$ are unitary matrix, and $S_\mu = \text{diag}\{\mu_1, \dots, \mu_d\} \in \mathbb{R}^{r\times r}$, $r \leq \min\{n, n_s - 1\}$ is the rank of matrix $\mathbb{P}_{\mathbf{1}}$.

2. Define matrix $\tilde{A}$ as

$$\tilde{A} = Y^*AY = Y^*\mathbb{P}_{\mathbf{2}}ZS_\mu{}^{-1}. \tag{3.14}$$

3. Perform eigenvalue decomposition of matrix $\widetilde{A}$,

$$\widetilde{A}\,X = X\Lambda, \tag{3.15}$$

where $X = [x_1 \dots x_r] \in \mathbb{R}^{r \times r}$ is eigenvector matrix and $\Lambda = \text{diag}\{\lambda_1 \dots \lambda_r\} \in \mathbb{R}^{r \times r}$ is eigenvalue diagnal matrix.

4. The dynamic mode can be calculated as

$$\phi_j = Ux_j, \quad \Phi = \{\phi_i\}_{i=1}^{r}. \tag{3.16}$$

It is usually preferred to normalize the DMD mode ϕ_j as $\overline{\phi_j}$ according to the first snapshot so that it preserves the correct scale [77]. Essentially, $d_{DMD} = \Phi^* P^1$, $\overline{\phi_j} = \phi_j d_{DMD}$.

5. Finally, the pressure value at a particular time step k can be approximated as

$$P^k \approx \sum_{\boldsymbol{j=1}}^{\boldsymbol{r}} (\lambda_j)^{\boldsymbol{k}} \overline{\phi_j} \, . \tag{3.17}$$

3.3. Sparsity Promoting Dynamic Mode Decomposition

Unlike the POD method that selects the subset of the mode according to the rank of the energy level from singular values, there is no natural ways for DMD method to select modes. Furthermore, the DMD modes are not orthogonal, which makes the projection to behave in different way as the POD modes. One of the approaches proposed is the sparsity-promoting DMD method that includes the sparsity structure by augmenting an objective function with and additional term that penalize the cardinality of non-zero elements. For detailed proof and procedure, we direct to the paper [24].

We return to the classical DMD method and introduce a vector of amplitude α to Eq. 3.17,

$$P^k \approx \sum_{j=1}^{r} (\lambda_j)^k \, \phi_j \, \alpha_j \, , \tag{3.18}$$

where $\alpha = [\alpha_1 \dots \alpha_r]$.

In the matrix form, the forward prediction is performed as Eq. 3.19,

$$\mathbb{P}_{\mathbf{1}} = [P^1 \; P^2 \; \dots \; P^{n_s-1}] \approx [\phi_1 \; \phi_2 \; \dots \; \phi_r] \begin{bmatrix} \alpha_1 & & & \\ & \alpha_2 & & \\ & & \ddots & \\ & & & \alpha_r \end{bmatrix} \begin{bmatrix} 1 & \lambda_1 & \cdots & (\lambda_1)^{ns-1} \\ 1 & \lambda_2 & \cdots & (\lambda_2)^{ns-1} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & \lambda_r & \cdots & (\lambda_r)^{ns-1} \end{bmatrix} =$$

$$\Phi \alpha \mathrm{V}_{and}. \tag{3.19}$$

The object now is to determine the initial value of α and furthermore seek a sparsity structure with optimal value of α. The initial value of α is determined through the following optimization problem

$$\underset{\alpha}{\text{minimize}} \; I(\alpha) = \|\mathbb{P}_1 - \Phi \alpha \mathrm{V}_{and}\|_F^2. \tag{3.20}$$

To determine the sparsity structure of α, an extra penalty term of l_1 norm instead of the l_0 norm (cardinality function) is used to preserve the convex structure of the optimization problem.

$$\min_{\alpha} I(\alpha) + \gamma \sum_{i=1}^{r} |\alpha_i| \tag{3.21}$$

The process could be illustrated with the steps depicted in Figure 3.1. After the sparsity structure is fixed for α, the problem reduces to determining only the nonzero amplitude element from the optimization problem in Eq. 3.21, with the constraint $E^T \alpha = 0$, where the matrix E contains the sparsity structure of α. For example, if we prefer a

structure of alpha as $\alpha = [0 \ \ \alpha_2 \ \ 0 \ \ \alpha_4]^T$, the matrix E could be $E = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}^T$. Finally, we get the optimal amplitude by using the method of Lagrange multiplier again:

$$L_p(\alpha, \tau) = I(\alpha) + \tau^* E^T \alpha + (E^T \alpha)^* \tau, \tag{3.22}$$

where τ is the vector constant of Lagrange multiplier.

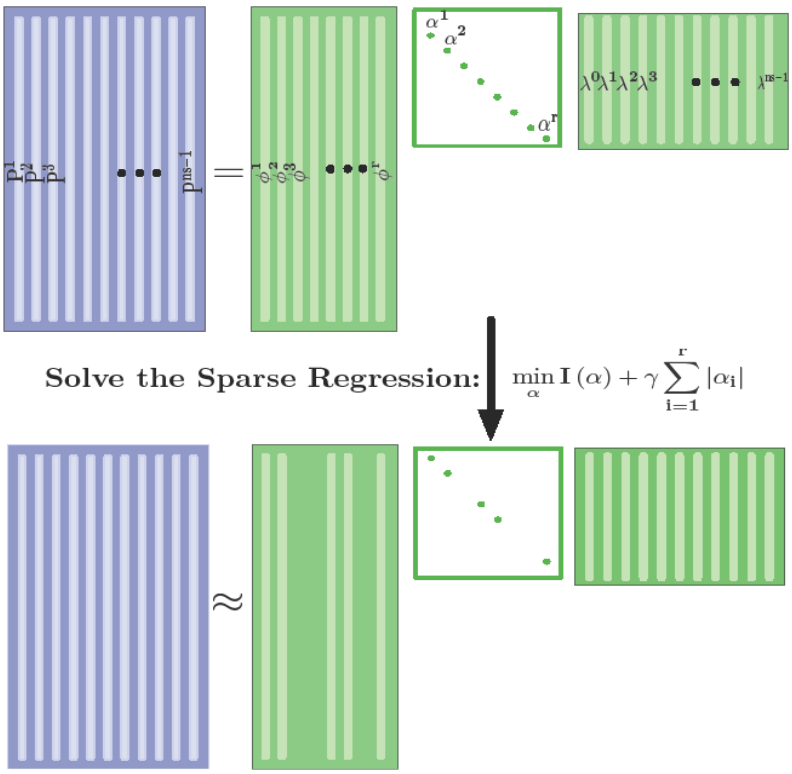


Figure 3.1 Schematic results of sparsity generating algorithm. This is a graphical representation of Eq. 3.19. Starting with Eq. 3.18 with full dimension of α, after applying the algorithm as in Eq. 3.21, the solution will have sparsity structure of amplitude α, that will decide which DMD modes to pick.

3.4. Results and Discussion

3.4.1. Single Phase Flow Example

In this example, we will demonstrate how the sparsity DMD modes are applied to the single-phase flow model. Consider a reservoir discretized into 15x15x3 Cartesian grids with each size 30ft ×30ft×30ft. The field is equipped with one producer at the center of the reservoir with constant bottom-hole pressure of 2900 psia. We assume the well is only perforated at the bottom layer (layer1). The permeability and porosity of the three layers are shown in Figure 3.2. The initial pressure is set to be 3100 psia. We simulate the reservoir for 365 days to get the pressure snapshot matrix of dimension 675 x 365. We will first apply the method on training dataset and get an initial idea of how the algorithm performs. Then, we get the testing data by perturbing the control parameter (production bottom hole pressure) and use the DMD modes generated from training data to test the actual performance.

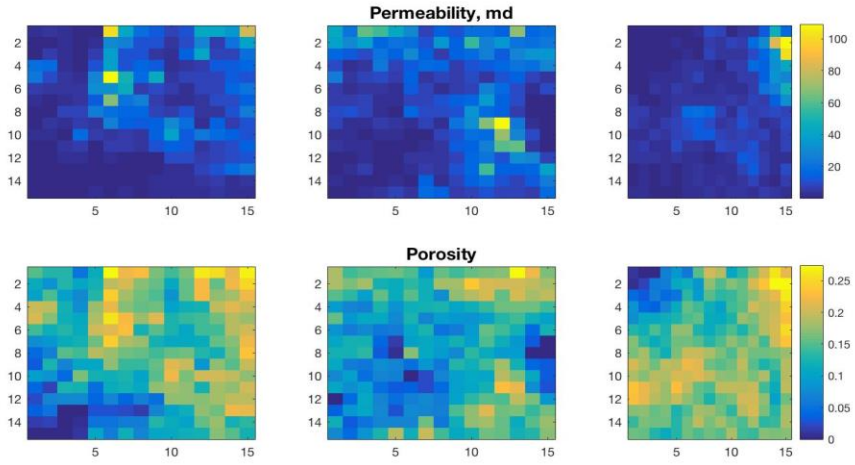


Figure 3.2 Permeability and porosity of single-phase reservoir model (reprinted from [78] with permission)

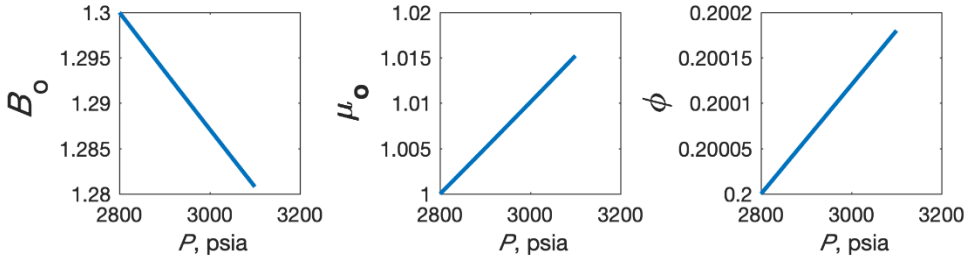


Figure 3.3 Rock and fluid properties of single-phase reservoir model

Through the sparsity DMD algorithm, we obtain the basis with sparsity structure. The level of sparsity of structure will depend on the value of the penalty factor gamma (γ) in Eq. 3.21. The larger the gamma value, the higher the sparsity level will be (Figure 3.4).The sparsity pattern can be seen further from Figure 3.5. Out of total 26 DMD modes (the numerical rank of the singular value matrix is 26), the sparsity DMD is able to select several of the important modes with low level of frequency (smaller value for the imaginary part of the eigenvalue).

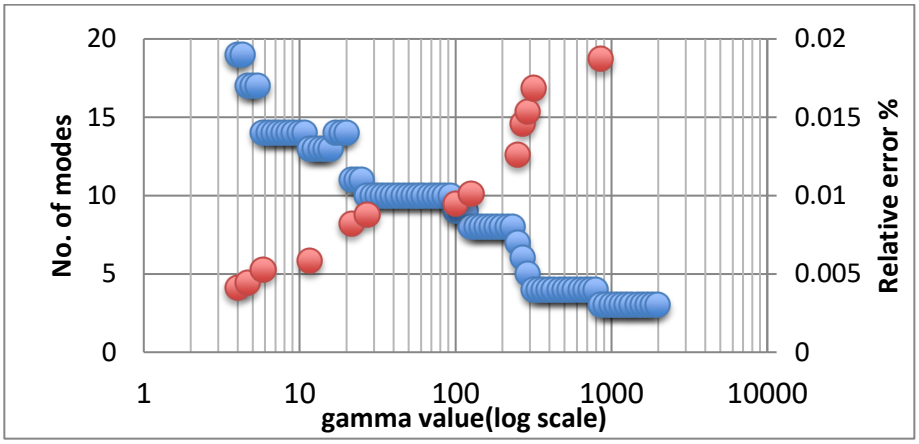


Figure 3.4 Number of sparsity promoting DMD modes changes with penalty factor gamma (reprinted from [78] with permission).

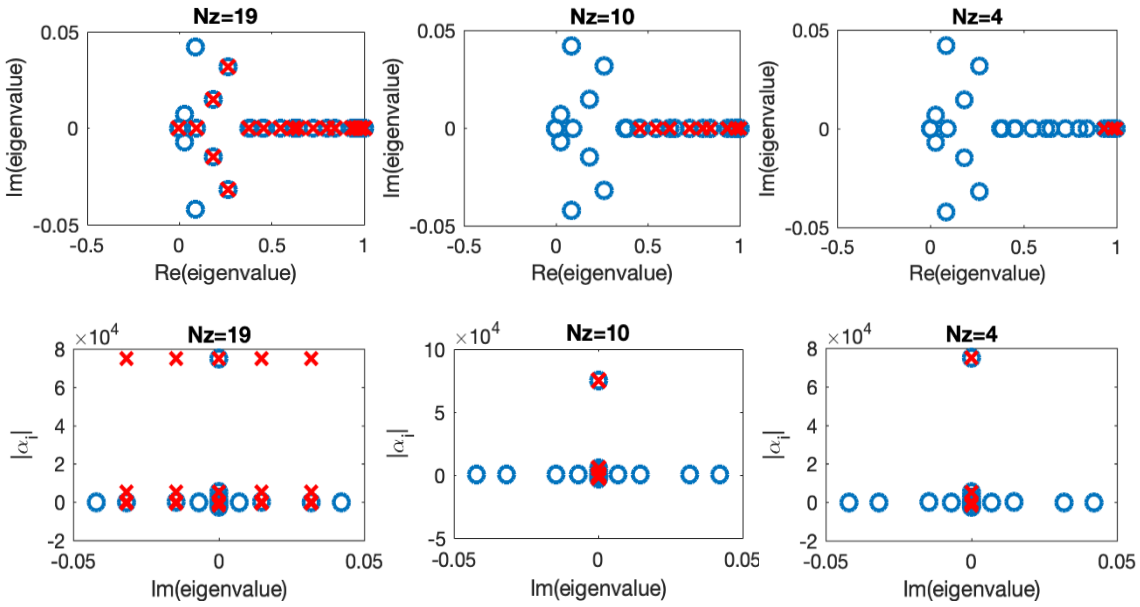


Figure 3.5 Eigenvalues of $\widetilde{A}$ (the matrix that contains coherent structure for DMD method as in Eq. 3.14) and the absolute values of the DMD amplitude α. The crosses are from sparsity promoting DMD algorithm with Nz DMD modes. The circles are for the standard DMD algorithm .

The basis can be used either in the offline process or the online process. Note that the offline process does not require the use of a simulator, instead, it only involves a superposition of the basis (Eq. 3.19). Thus, with affordable effort to compute the sparse DMD modes (see Figure 3.7), we could obtain a computationally more efficient prediction, especially with the offline process. The online process utilizes Galerkin projection to project the system onto a subspaces spanned by the basis. The final results shown in Figure 3.6 imply good agreement between fine scale model and reduced model (with sparse DMD). In all examples, the overall relative error is computed with Frobenius norm,

$$\text{error} = \frac{\|\mathbb{P}_{ref} - \mathbb{P}_{app}\|_F}{\|\mathbb{P}_{ref}\|_F} \times 100\%,$$

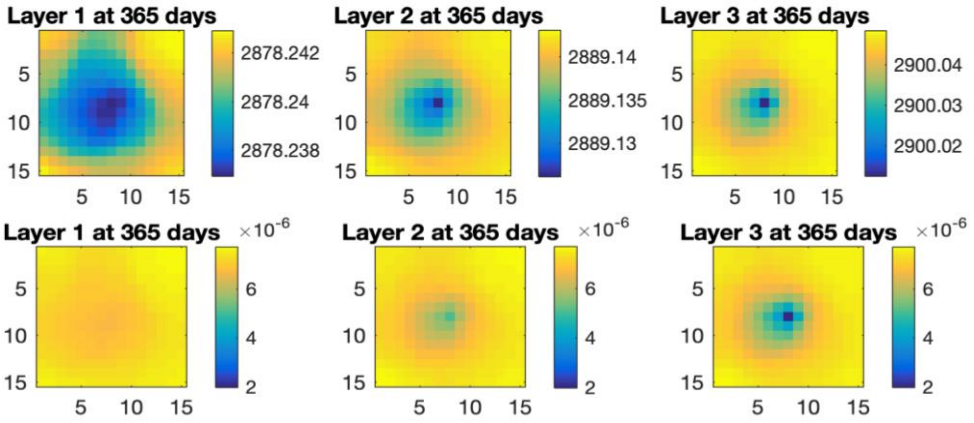
where $\mathbb{P}_{ref}$ is the snapshot matrix from simulator, $\mathbb{P}_{app}$ is approximated solution from model reduction.

The relative error in terms of days is computed with l_2 norm,

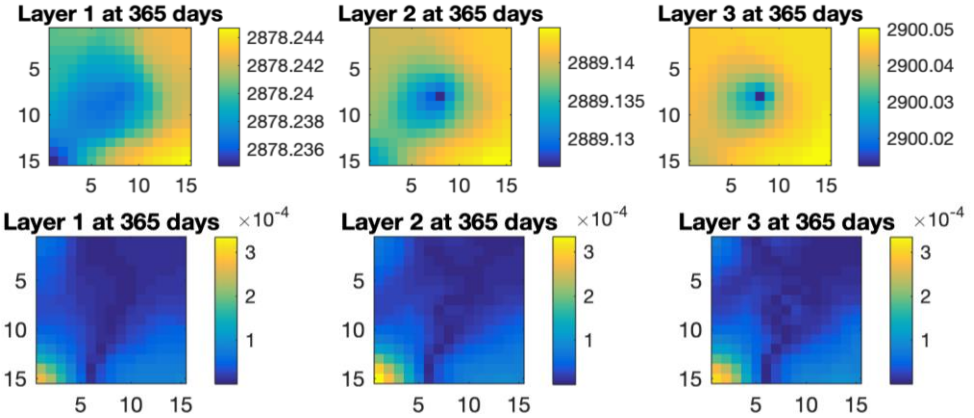
$$\mathrm{P}_{err}(t_i) = \frac{\|\mathrm{P}_{ref}(t_i) - \mathrm{P}_{app}(t_i)\|_{l^2}}{\|\mathrm{P}_{ref}(t_i)\|_{l^2}} \times 100\%,$$

where $\mathrm{P}_{ref}(t_i)$ is a vector of pressure results from simulator at time t_i, while $\mathrm{P}_{app}(t_i)$ is the approximated solution from model reduction.

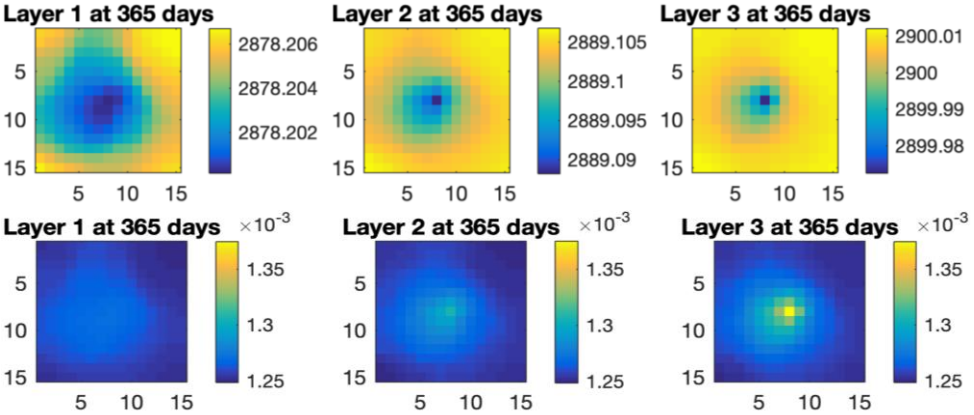
As we already noticed before [78], the overall error of Sparse DMD compared with POD method suggests unstable behavior of POD, the error is not always decreasing as the number of modes decreased. When the number of modes reduced to a certain amount (in this case 5) for POD method, as seen in Figure 3.6 the pressure map shows incorrect, non-physical behavior, lower left corner of layer 1 has pressure even lower than the actual production well. We further investigate the computational efficiency of sparsity DMD by showing the reduced solver time with fewer basis used, as seen in Table 3.1. The computational time for selecting optimum basis is only around 0.1%~0.2% of the original solver time. But the solver time could be reduced more than 100 times for sparsity-promoting DMD method.



(a) Online sparsity DMD with 5 basis (first row = pressure maps; second row = error maps)



(b) Online POD with 5 basis (first row = pressure maps; second row = error maps)



(c) Offline sparsity DMD with 24 basis (first row = pressure maps; second row = error maps)

Figure 3.6 Results of reduced order model(training case) with different methods, within each group of (a)(b)(c) the upper one is pressure map and the lower one is error map.

Table 3.1—Compare online computation time of solver with different MOR method (training dataset).

	Jacobian size	Computation time (sec)	Mode selection time (sec)	Relative error (%)
Fine scale	675x675	9.89 (solver time)	-	-
Standard DMD, online	50x50	1.51 (solver time)	-	1.77e-4
Sparse DMD, online	10x10	0.09 (solver time)	0.0042	0.0056
Sparse DMD, online	5x5	0.0748 (solver time)	0.0099	0.011
Sparse DMD, offline	24x24	0.042 (total time)	0.0174	0.010

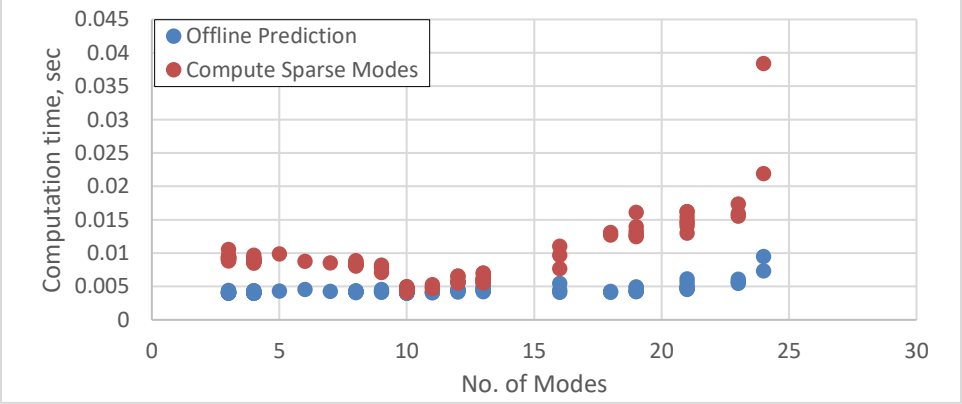


Figure 3.7 Computation time for sparse DMD modes and offline prediction

Next, we repeat the process for a different testing dataset, which we will use perturbed control parameters (bottom hole pressure for producer), as seen in Figure 3.8. For the online process, we use the DMD modes generated from the training dataset and perform Galerkin projection with the simulator. To illustrate the significance of offline process, we use the first 70% of the testing data to generate DMD modes and predict the results for the rest 30% of the testing data. With a different control parameter to formulate the testing case, the online DMD results is shown in Figure 3.9.

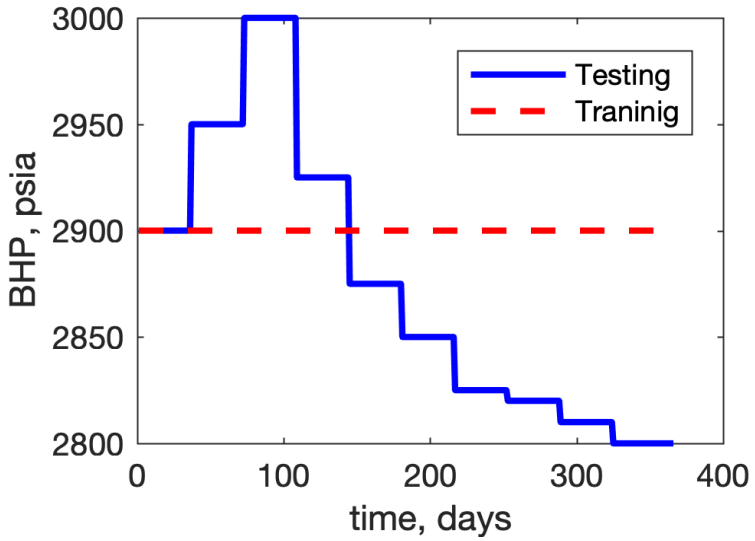


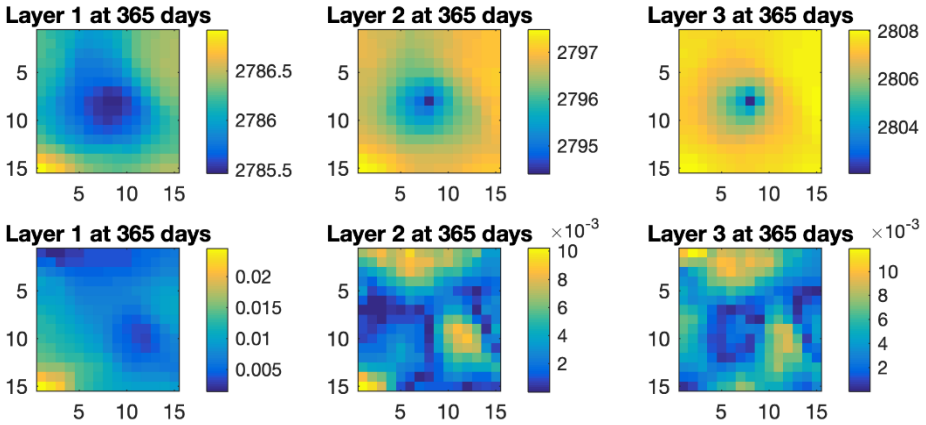
Figure 3.8 The Training and Testing (perturbed) bottom hole pressure for producer.

Table 3.2—Compare online computation time of solver with different MOR method (testing dataset).

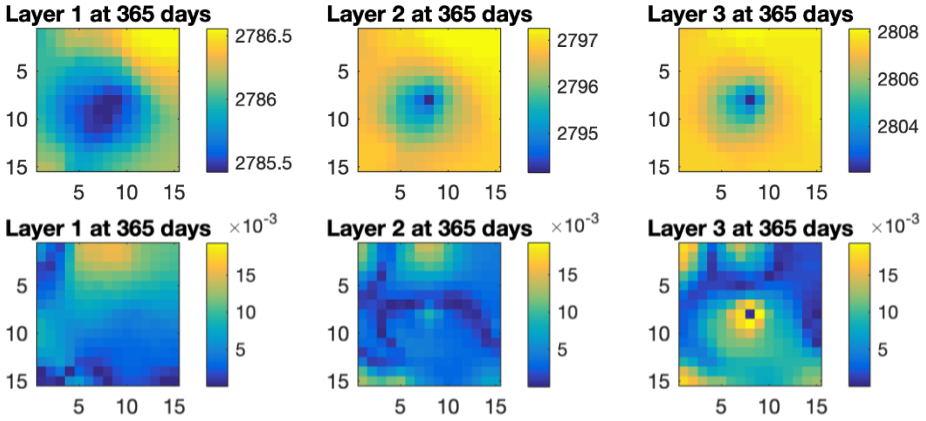
	Jacobian size	**Computation time (sec)**	**Relative error (%)**
Fine scale	675x675	9.90 (solver time)	-
Sparse DMD, online	24x24	0.5347 (solver time)	0.0056
Sparse DMD, online	9x9	0.0884 (solver time)	0.0079
Sparse DMD, online	5x5	0.0820 (solver time)	0.0129
Sparse DMD, online	3x3	0.079 (solver time)	0.0210

A comparison of POD and sparsity-promoting DMD with limited number of modes (9 modes) to capture the dynamics is shown in Figure 3.9. The POD and DMD has similar performance, however DMD seems outperform POD in mass conservation results (Figure 3.10) especially in layer 3 (where the well is perforated) near the well location. In addition, the DMDc is specifically designed for the complex control problem such as the case we used in our testing case. In the future, we will test on the possibility of sparsity-

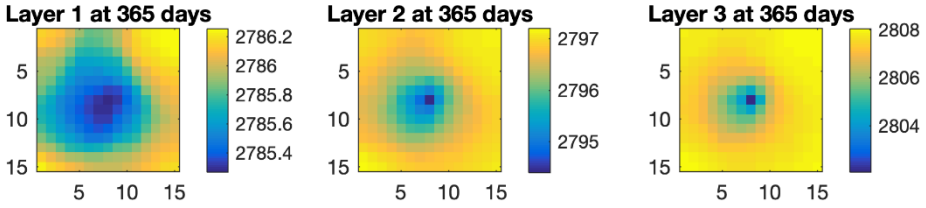
promoting DMDc leading to a better result in the complex control case study. Finally, we noted here that the DMD method's main advantage is to perform offline prediction, which we will show next.



(a) Online sparsity DMD with 9 basis



(b) Online POD with 9 basis



(c) High fidelity model.

Figure 3.9 Results of reduced order model (testing case) with different algorithm, within each group of (a)(b), the upper row is pressure map, the lower row is relative error map (%).

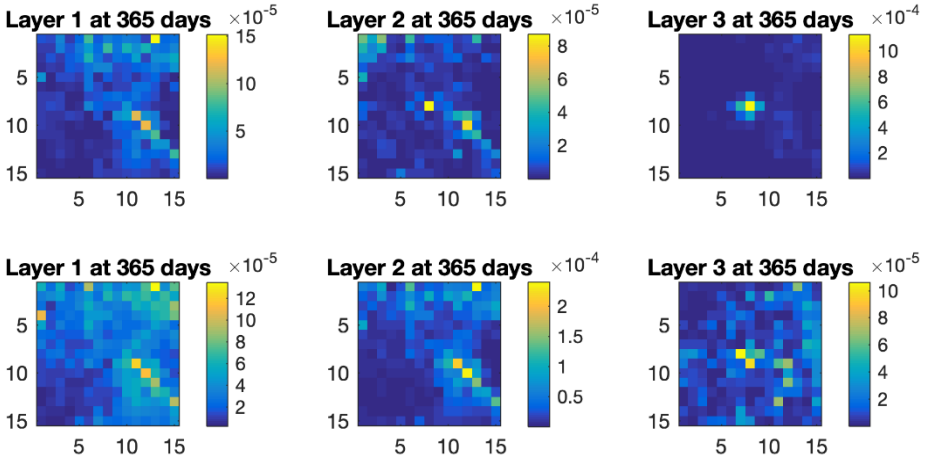


Figure 3.10 Mass conservation violation scaled as average pore volume. The upper row is from POD method and lower row is from DMD method.

For the offline process, we use the first 60% (220 timestep) of the testing data to generate DMD modes and predict the results for the rest 40% (145 timestep) of the testing data. The offline results for using 41 modes are shown below (Figure 3.11). The initial numerical rank is 48 for from the calculation based on first 60% dataset. The total prediction time is only 0.2267 second with 41 sparse DMD modes. The relative error is high compare with the online DMD (Figure 3.12). But we note here that there is special technique, such as DMDc (DMD with control) to deal with complex control situation and that will be the future work of this research.

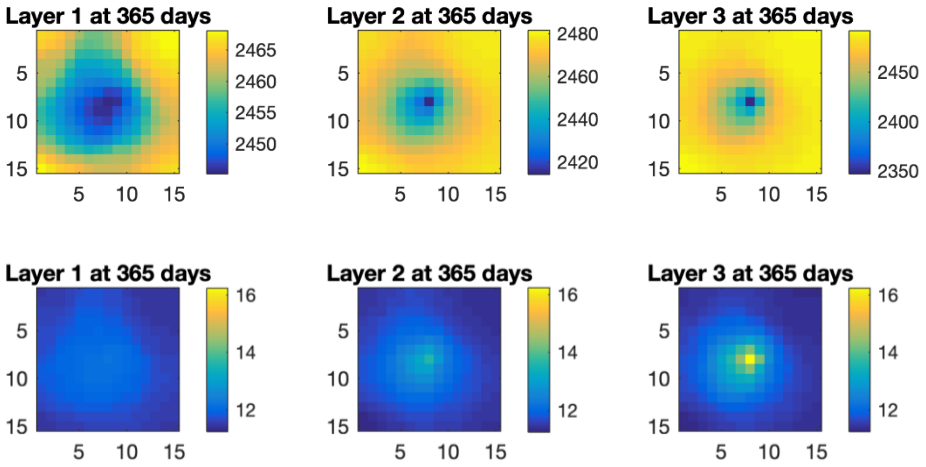


Figure 3.11 Results of reduced order model used offline. The upper row are pressure maps and the bottom row are relative error maps (%).

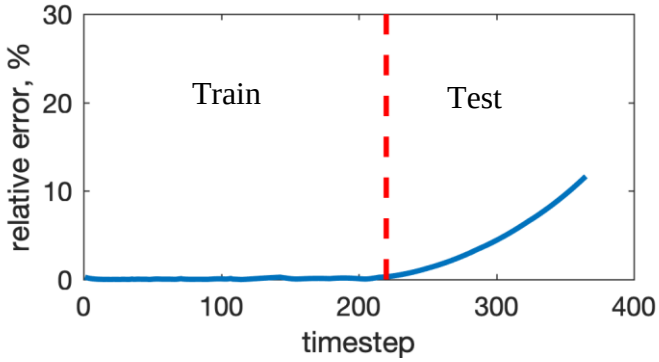


Figure 3.12 Relative error vs. timestep.

3.4.2. Two Phase Flow Example

In this example, we will show that sparsity-promoting DMD method is applicable to larger heterogeneous reservoir with two-phase flow dynamics. The reservoir model is under water flooding process with five spot structure: four producers at corner and one injector at the center. The model is based on the SPE10 comparative benchmark [79] with one layer only. Thus, the reservoir is discretized into 60 x220x1 Cartesian grids with each

grid size of 20ft x 10ft x 2ft. The absolute permeability is shown in Figure 3.13. The relative permeability is approximated with the Corey correlation, which is essentially a power law function of saturation (with power 2 and 3 for water and oil respectively). Porosity is assumed a constant value of 0.2. Viscosity for water and oil are 0.3 and 3 respectively. We assume a constant injection rate of 188 STB/time step and constant bottom hole pressure of 2500 psia for the four production wells. Note that the injection rate is chosen to ensure at the end of simulation time (1000 days), four pore volume of water is injected. The initial condition is set as 0.0 for saturation and 2500 psia for pressure.

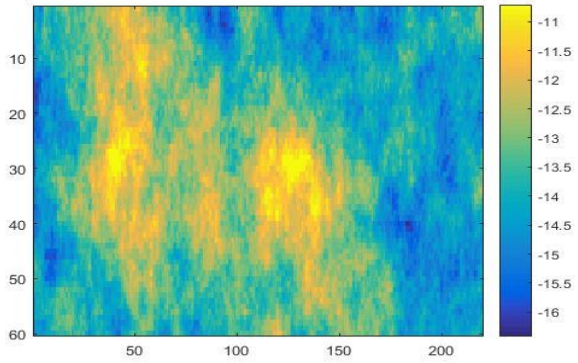
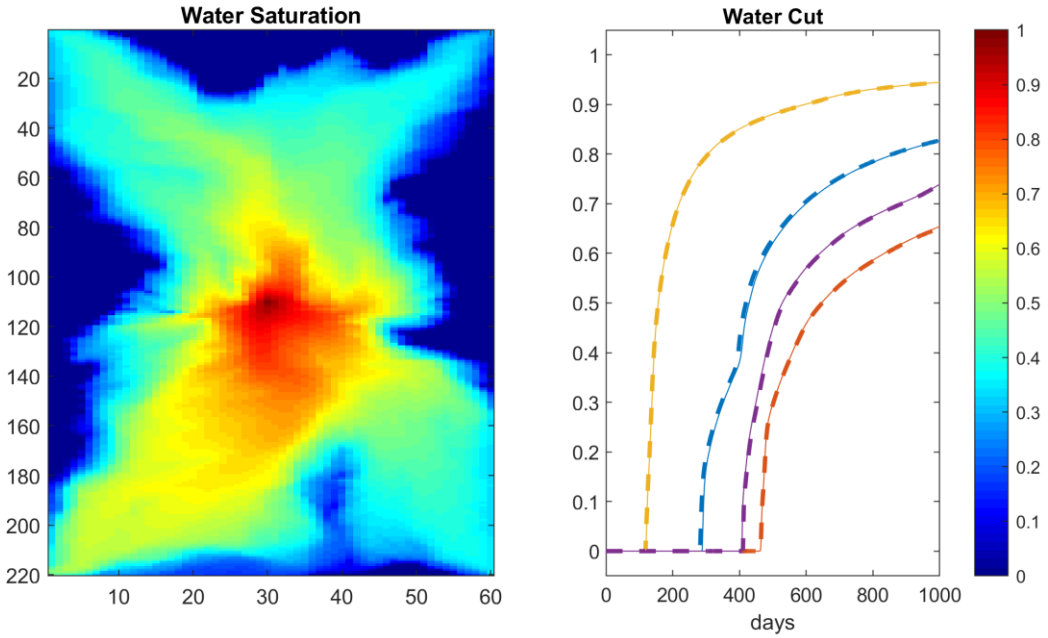


Figure 3.13 Logarithmic of permeability field in the 2D two phase flow reservoir model (reprinted from [78] with permission).

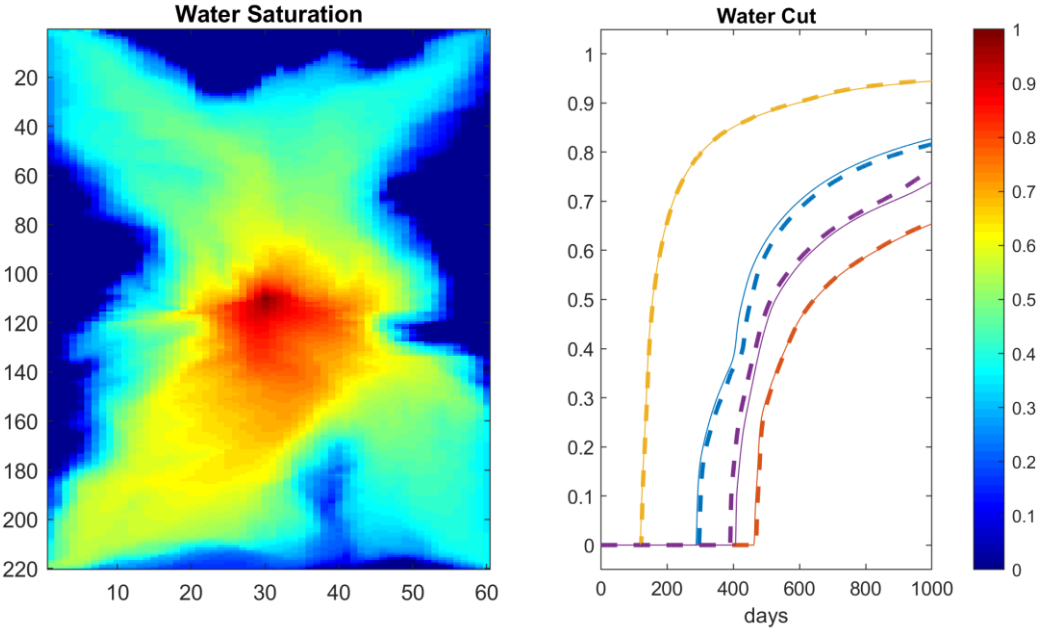
To use the Sparsity DMD method, first we collect 1000 days of simulation results of pressure and velocity to build the snapshot matrix with dimension of 13200 x 4000 (time interval is 0.25 days). We then apply the sparsity-promoting DMD algorithm mentioned before to obtain the DMD basis with sparse structure. The basis could be used to reproduce pressure and velocity using Eq. 3.19. Total time for this offline process is

often several seconds since it only involves one step matrix multiplication (Eq. 3.19). This is rather efficient compared with the high-fidelity simulator run time. In Table 3.3, we present the overall relative error between the offline approximated solution and the results from fine scale model. The error is rather small, with magnitude of 3e-3 for sparsity DMD. However, the error becomes non-negligible for POD when number of modes reduced to 40 (Table 3.3, Figure 3.16).

The online process using sparse DMD modes produces good match in the water cut as compared with high fidelity model (Figure 3.14). The online case gives a better match than the offline process (Figure 3.15), and requires less computational time of solver compared with high fidelity model. Compared with POD method, in general, the sparse DMD method could results in smaller relative error. With as less as 40 modes, sparse DMD method keeps high consistency with reference solution and the error increases steadily, while POD method results in jump in relative error as the number of mode reduces to 40 (Figure 3.16).



(a)



(b)

Figure 3.14 Results comparison in 2D reservoir: Final Saturation and water cut for reduced model with (a) 40 sparse DMD modes (b) 40 POD modes. (dashed line: reduced model. Solid line: high-fidelity model).

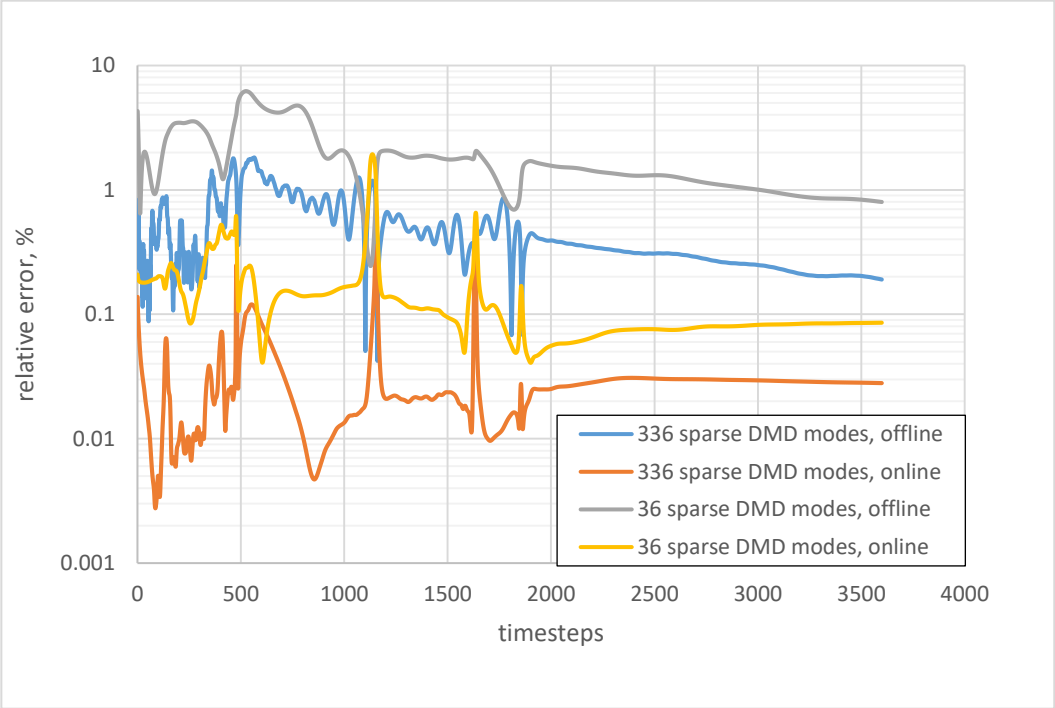


Figure 3.15 Relative error of pressure and velocity as the change of time in 2D reservoir

Table 3.3 Relative error of pressure and velocity between high fidelity and reduced model.

error, %	268 modes			196 modes			116 modes		
	Po	Sw	V	Po	Sw	V	Po	Sw	V
POD	0.0101	0.104	0.3838	0.0309	0.9949	0.6843	0.1045	1.0051	3.3654
DMD	0.053	0.4846	0.6396	0.0637	0.6097	0.7132	0.1	0.9858	0.9193
error, %	50 modes			40 modes			36		
	Po	Sw	V	Po	Sw	V	Po	Sw	V
POD	0.1457	1.6719	4.7611	0.3628	19.6819	9.7816	0.3664	19.9453	9.8448
DMD	0.1809	2.2609	1.2486	0.2013	2.4665	1.318	0.2482	2.4367	1.4164

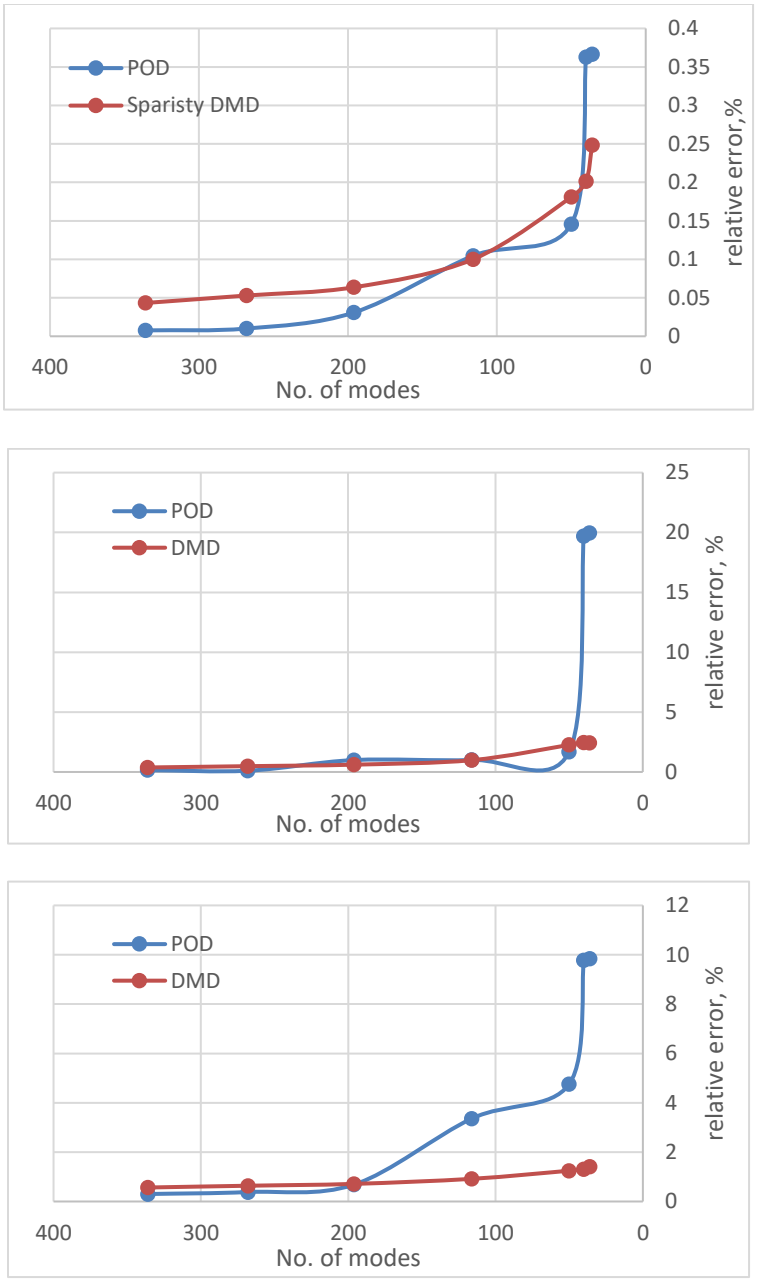


Figure 3.16 plot of relative error of pressure, velocity and saturation vs. number of mode used.

In this chapter, we presented applications of sparsity-promoting DMD method to single- and two-phase fluid flow modeling. Compared with standard DMD, the sparsity-promoting DMD can further reduce the number of modes required with a sparse structure solved from Lagrange optimization problem. The relative error is steadily reduced corresponding to fewer modes used. Within reasonable range of time period, the offline stage with DMD or sparse DMD is faster than full-order or reduced order online modeling without compromise on the accuracy. In the online process, the most time-consuming solver part will be speed up with the aid of Galerkin projection and calculated basis from model reduction algorithms. Furthermore, from the example, sparse DMD mode preserves more accuracy than POD modes especially when number of modes is small. In the future, we will investigate on the snapshot selection in the DMD method in order to improve the efficiency and robustness of the sparse regression algorithm.

4. PHYSICS CONSTRAINED DATA MINING TECHNIQUE FOR SYSTEM IDENTIFICATION*

The goal of our system identification is to discover the governing conservation equations (ODE's in this case) for the dynamic reservoir system. To illustrate the problem, we will start with a simple example, as stated in section 4.1, whereby we identify a set of ODEs with three variables. However, the governing equation of the reservoir simulation is essentially originated from mass conservation and Darcy's flow PDE equation. To apply the system identification technique, the PDE needs to be discretized into the ODE system with each equation depicting one state variable (pressure or saturation of each grid block). In this chapter, we develop a framework for identifying the reservoir simulation equation.

The outline of this chapter is as follows: First, we describe the governing equations for single- phase and two-phase reservoir dynamic system. Then describe the work flow and algorithm of SINDy and Lasso that we customized to apply to the dynamic system. Finally show the numerical results for both sparse regression method and discuss the error and model selection process.

4.1. Simple Example

Here is an example of a three-variable ODE system to motivate the problem.

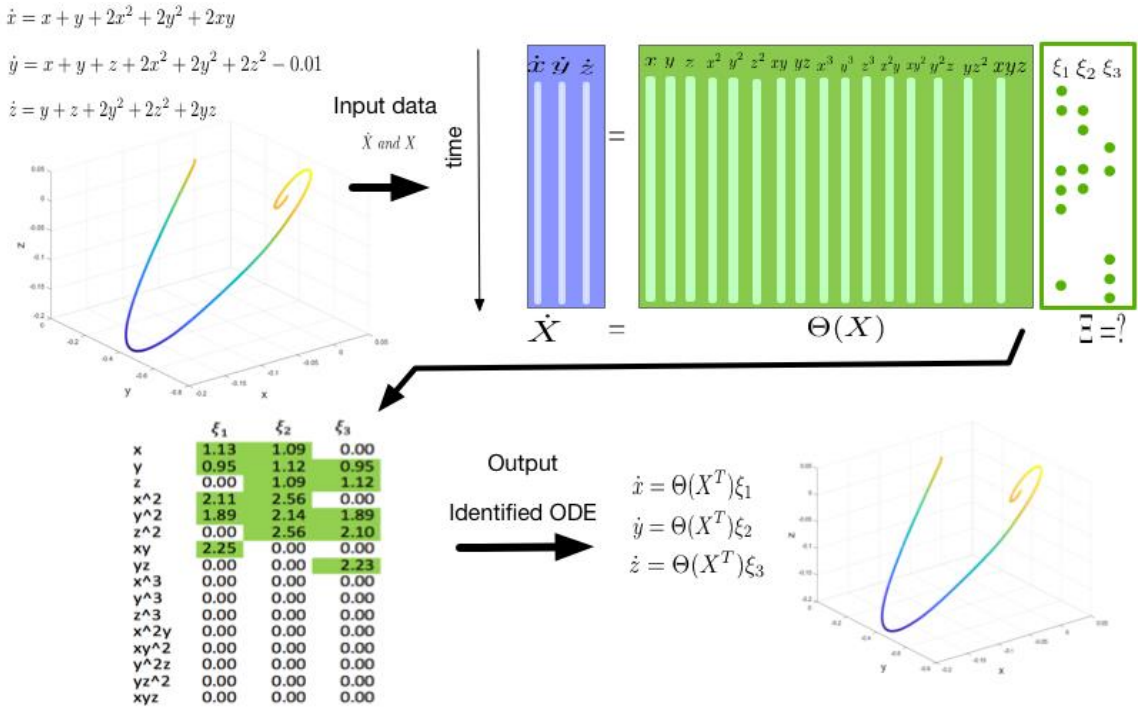
*Part of this section is reprinted with permission from "Development of Proxy Models For Reservoir Simulation By Sparsity Promoting Methods And Machine Learning Techniques" by Bao et al, ECMOR XVI - 16th European Conference on the Mathematics of Oil Recovery , Copyright [2018] by European Association of Geoscientists and Engineers

$$\dot{x} = x + y + 2x^2 + 2y^2 + 2xy \tag{4.1}$$

$$\dot{y} = x + y + z + 2x^2 + 2y^2 + 2z^2 - 0.01 \tag{4.2}$$

$$\dot{z} = z + y + 2z^2 + 2y^2 + 2yz \tag{4.3}$$

As seen in Figure 4.1, to determine the actual ODE system from data, we first collect data $\mathrm{X} = [x \quad y \quad z]^T$ and calculate its derivative in terms of time $\dot{X}$ in all time step. Then we construct a library $\Theta(X)$ that consist all the candidate basis functions in the ODE. Finally, each of the ODE is identified with a sparse vector of coefficient ξ_i . The coefficient vector determines which candidate terms to pick for the particular equation. Once the coefficient matrix $\Xi = [\xi_1 \quad \xi_2 \quad \xi_3]$ is determined, the each of the governing ODE can be expressed as (e.g. the first equation) $\dot{x} = \Theta(X^T)\xi_1$. For example the final results for the first equation is ξ_1= [1.13, 0.95, 0, 2.11, 1.89, 0, 2.25, 0, 0, 0, 0, 0, 0, 0, 0, 0], which indicate for the first ODE, the algorithm identifies 5 non-zero terms $[1.13x, 0.95y, 2.11x^2, 1.89y^2, 2.25xy]$. The results are very close the true equations with terms $[x, y, 2x^2, 2y^2, 2xy]$.



	ξ_1	ξ_2	ξ_3
x	1.13	1.09	0.00
y	0.95	1.12	0.95
z	0.00	1.09	1.12
x^2	2.11	2.56	0.00
y^2	1.89	2.14	1.89
z^2	0.00	2.56	2.10
xy	2.25	0.00	0.00
yz	0.00	0.00	2.23
x^3	0.00	0.00	0.00
y^3	0.00	0.00	0.00
z^3	0.00	0.00	0.00
x^2y	0.00	0.00	0.00
xy^2	0.00	0.00	0.00
y^2z	0.00	0.00	0.00
yz^2	0.00	0.00	0.00
xyz	0.00	0.00	0.00

Figure 4.1 schematic plot of the system identification technique with a simple example. The three basic steps are creating input data, solving for coefficient matrix Ξ and finally outputting identified ODE system.

4.2. Governing Equation

In the following section, we start with the discretized ODE for both single and two-phase flow to show the application of identification governing equation technique.

4.2.1. Governing Equation for Single-Phase Flow

From the governing equation in Eq. 2.8, after discretization in space, the partial differential equation reduces to,

$$B\frac{dP(t)}{dt} = T_p\big(P(t)\big)P(t) - Q\big(P(t)\big) - G\big(P(t)\big) = T_p\big(P(t)\big)P(t) - F_P\big(P(t)\big). \quad (4.4)$$

Here, t denotes time, $P(t) = [P_1(t), \dots, P_{nc}(t)]^T \in \mathbb{R}^{nc}$ is a vector of pressures, $T_p \in \mathbb{R}^{nc \times nc}$ is transmissibility matrix, $B_p = \frac{\partial(\phi)}{\partial P} + \frac{\partial(\rho)}{\partial P} \in \mathbb{R}^{nc \times nc}$ is a coefficient matrix, F_P accounts for the source/sink and gravity terms and is a function of pressure, i.e., $F_P = [F_P(P_1(t)), \dots, F_P(P_{nc}(t))]^T$.

Our data-driven methodology requires that we decompose the coefficients of Eq. 4.4 into several nonlinear terms, as it will be shown below. As for the flux term Q, it can be dealt as a constant term added to the left hand side if it is assumed a constant flow rate well or could be merged with the other nonlinear terms in the library if we impose a constant bottom hole pressure well.

After decomposing T_p and F_P into nonlinear terms, at a specific time t_k, the set of nonlinear ordinary differential equations (ODE) yields,

$$B\dot{P}(t_k) = \Xi^T \Theta_{t_k}^T(P(t_k)). \tag{4.5}$$

where $\Theta_{t_k}^T$ is the vector of nonlinear terms in the ODE systems at time t_k. The coefficient matrix Ξ will not change with time and can be used to simulate the dynamic system. Ξ can be obtained by solving the following the equation (4)

$$B\dot{P} = \Theta(P)\Xi, \tag{4.6}$$

where

$$\Theta(P)=\begin{bmatrix} 1 & P_1(t_1) & P_2(t_1) & \cdots & P_n(t_1) & P_1^2(t_1) & \cdots & P_n^2(t_1) & P_1P_2(t_1) & \cdots & P_{n-1}P_n(t_1) & P_1^3(t_1) & \cdots \\ 1 & P_1(t_2) & P_2(t_2) & \cdots & P_n(t_2) & P_1^2(t_2) & \cdots & P_n^2(t_2) & P_1P_2(t_2) & \cdots & P_{n-1}P_n(t_2) & P_1^3(t_2) & \cdots \\ \vdots & \vdots & \vdots & \cdots & \vdots & \vdots & \cdots & \vdots & \vdots & \cdots & \vdots & \vdots & \cdots \\ 1 & P_1(t_n) & P_2(t_n) & \cdots & P_n(t_n) & P_1^2(t_n) & \cdots & P_n^2(t_n) & P_1P_2(t_n) & \cdots & P_{n-1}P_n(t_n) & P_1^3(t_n) & \cdots \end{bmatrix}$$

$$\begin{bmatrix} P_n^3(t_1) & P_1^2P_2(t_1) & \cdots & P_{n-1}P_n^2(t_1) \\ P_n^3(t_2) & P_1^2P_2(t_2) & \cdots & P_{n-1}P_n^2(t_1) \\ \vdots & \vdots & \cdots & \vdots \\ P_n^3(t_n) & P_1^2P_2(t_n) & \cdots & P_{n-1}P_n^2(t_1) \end{bmatrix}$$

is the library matrix with columns containing all possible nonlinear terms and rows for every time step. Essentially, every column is a time series.

The left hand side term

$$B\dot{P}=\begin{bmatrix} b_{11}\dot{P}_1(t_1) & b_{22}\dot{P}_2(t_1) & \cdots & b_{nn}\dot{P}_n(t_1) \\ b_{11}\dot{P}_1(t_2) & b_{22}\dot{P}_2(t_2) & \cdots & b_{nn}\dot{P}_n(t_2) \\ \vdots & \vdots & \ddots & \vdots \\ b_{11}\dot{P}_1(t_n) & b_{22}\dot{P}_2(t_n) & \cdots & b_{nn}\dot{P}_n(t_n) \end{bmatrix}$$

is a pressure derivative matrix with columns distinguished by space variable, rows separated by time steps (See Figure 4.2 for a schematic view).

4.2.2. Governing Equation for Two-Phase Flow

As already disscuss the in Chapter 2, we recap the mix formulation two phase flow simulator. For the sake of completeness, we assume an incompressible two-phase flow model and neglect gravity effects and capillary pressure. As previously mentioned, the formulation of two-phase flow equations is written as

$$\mathrm{v}_\chi = -\frac{k_{r\chi}}{\eta_\chi} K\nabla P \tag{4.7}$$

$$\varphi \frac{\partial s_\chi}{\partial t} + \nabla v_\chi = q_\chi. \tag{4.8}$$

We can write the saturation equaton 2.25 in a matrix form

$$\frac{ds}{dt} = -Af(s) + q_{inj}, \tag{4.9}$$

where

$A_{i,j} =$

$$\begin{cases} \sum_k v_{ik} \cdot n_{ik} - q_i^- & i = j,\ \ k\ is\ the\ edge\ satisfies\ v_{ik} \cdot n_{ik} \geq 0 \\ v_{ik} \cdot n_{ik} & if\ j\ is\ the\ upwind\ index\ for\ kth\ edge\ of\ cell\ i,\ \ k = 1,2,3,4 \\ 0 & otherwise \end{cases}$$

$q_{inj} = \{\frac{q_1^+}{\varphi_1|\Omega_1|}, \dots, \frac{q_i^+}{\varphi_i|\Omega_i|}, \dots \frac{q_n^+}{\varphi_n|\Omega_n|}\}^T$ and $f(s) = \{f(s_1), \dots f(s_n)\}^T$. As seen in the equation, the matrix *A* is a function of the velocity and production rate

Our objective in this work is to identify the governing ODEs for saturation (Eq. 2.25) assuming pressure and velocity are known. In the future, we will explore the possibility of identifying pressure and velocity equations and to achive a fully funtional system identification technique for two phase flow. But first, to reveal only the satraution euqations, we approximate the fractional flow with polynomial function of saturation. To avoid the linear dependence in the basis library, the intercept/constant term is not included in the polynomial function. To select the proper order for modeling the function between satratuion and water cut, we use the AIC ((Akaike Information Criterion) number. The AIC number is a model selection critiria that will balance between model complexity and error in order to select for the optimum model and commonly defned as

$$AIC = 2k + nlog(\frac{RSS}{n}), \tag{4.10}$$

where *n* is the number of samples in the linear regression. RSS is the residual sum of squre of the linear regression. *n* is the number of samples used to perform linear fit. *k* is the model complex number that equals to the number of parameters of the model plus 2, for example, $k = 5 - 2 = 3$ in Eq. 4.11. But we also noted here that the AIC number in different literatures can vary by adding a aconstant or multiply by a factor.

According to the statistics significance test shown in Table 4.1, the forth order polynomial has the lowest AIC number (4.10) with all components significant. So the polynomial function we will use is as follows,

$$f_w(s_w) = c_4 s_w^4 + c_3 s_w^3 + c_2 s_w^2 + c_1 s_w \,. \tag{4.11}$$

More advanced approaches have been raised to approximate the original equation with simpler functions, such as the quadrative bilinear formulation (QBDAE) [12] and Carleman bilinearization [80]. However, here we only use a polynomial function of single variable (s_w) for the purpose of not introducing extra unknow states as in the QBDAE, and the unknown derivative terms as in the Taylor expansion in the Carleman bilinearization.

Table 4.1 Selection for order of polynomial function (reprinted from [48] with permission).

Model	Significance of Each Polynomial Component	Mean Square Error	AIC
$f_w = c_5 s_w^5 + c_4 s_w^4 + c_3 s_w^3 + c_2 s_w^2 + c_1 s_w$	s_w^5 not significant	0.001	-164.73
$f_w = c_4 s_w^4 + c_3 s_w^3 + c_2 s_w^2 + c_1 s_w$	all significant	0.001	-164.73
$f_w = c_3 s_w^3 + c_2 s_w^2 + c_1 s_w$	all significant	0.006	-149.31

4.3. Sparse Identification of Nonlinear Dynamical system (SINDy)

We start by introducing the SINDy method developed by Brunton, Proctor and Kutz in 2016 to identify nonlinear terms in governing ODEs. This method solves the linear system in Eq. 4.6 with a direct solver or ridge regression. As the solution is not sparse, they then apply a hard threshold to screen out the small coefficient and thus create a sparse structure. We customized this method to solve for our discretized reservoir ODE system (Eq. 4.4). Several modifications will be made, which include constructing a library, physical constraints for fluid flow in reservoir, data screening, and different solvers for linear system. The schematic workflow for our customized SINDy method is in shown in Figure 4.2

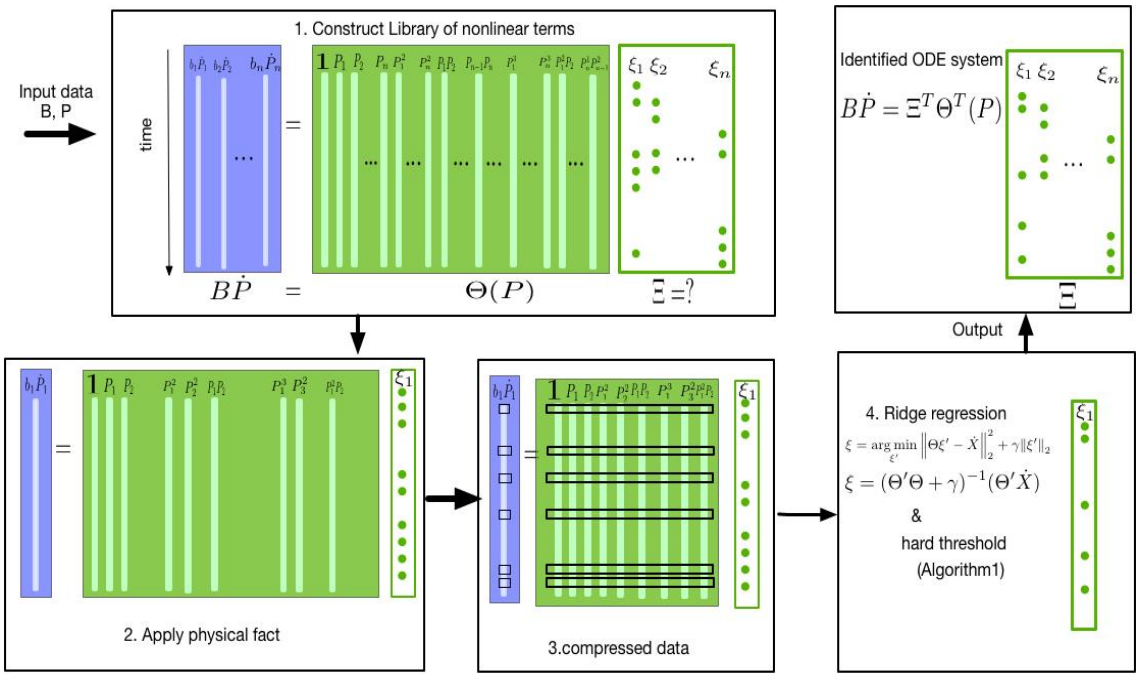


Figure 4.2 Work flow for discovering ODE with SINDy. The process consists of four major steps, 1. Construct library of possible nonlinear terms; 2. Apply physical constraints to reduce columns of data; 3. Reduce the rows according to magnitude of change of dynamics (time derivative terms); 4. Apply ridge regression and hard threshold to obtain sparse structure function (reprinted from [48] with permission).

First step is to build the nonlinear library. The T_pP term can be approximated as a polynomial function up to third degree as seen in following figure. The sink/source term Q, if is a constant, can be accounted for the in the library as a column full of ones or can be directly deducted from the accumulation part (left hand side). The physical constraints that one should be aware of are inherited from the fact that we are assuming a discretization using a 5-point stencil, that is, each state variable is only related with its adjacent states and itself. For example, to calculate P_1 in a one-dimensional system, only P_2 and P_1 are needed to construct nonlinear terms. See Figure 4.3 as an example.

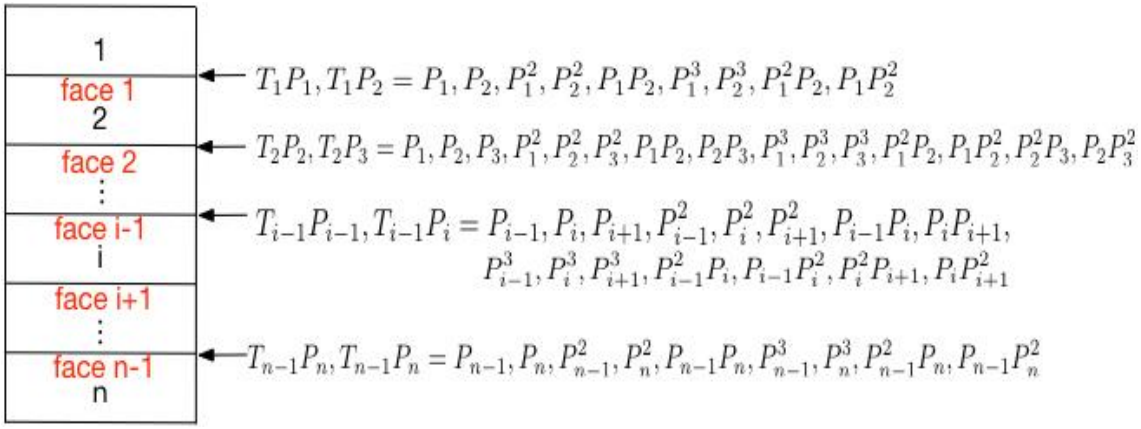


Figure 4.3 Scheme for selection of nonlinear times based on one dimensional model. Terms in each grid block is only related with parameters in its neighbor grid block function (reprinted from [48] with permission).

We make use of another physical fact (constraint) that allows data screening. At the reservoir boundary where pressure wave took time to reach, the $\frac{\partial P}{\partial t}$ remains zero at the early time. It is hard for the program to capture the dynamics in the boundary cells at the

early stage. Keeping those close to zero terms in left hand side will jeopardize the accuracy of solver. So we eliminate the almost zero entries in accumulation part and the corresponding row entries in the library. In addition, we need to decide how much data is needed for training the model, that is the length of the time series data. This is obtained by balancing the relative error and the less data used as possible, because we do not want excessive training data to overfit the model and cost unnecessary extra time. Figure 4.4 illustrates how the prediction error decreases as the length of training data increases for a 3x3 reservoir system. Although small offline error, using the original library with calculated coefficients for prediction, does not necessarily mean a good fit of model, as we may overfit it by pursuing a very small relative error, it could be an initial guideline to tune the parameters, such as the length of training dataset. We typically choose a relatively small amount of training data to achieve a relatively small error. Thus, using reasonable amount of training time series data with enough resolution (fine Δt), we can identify the feasible ODEs.

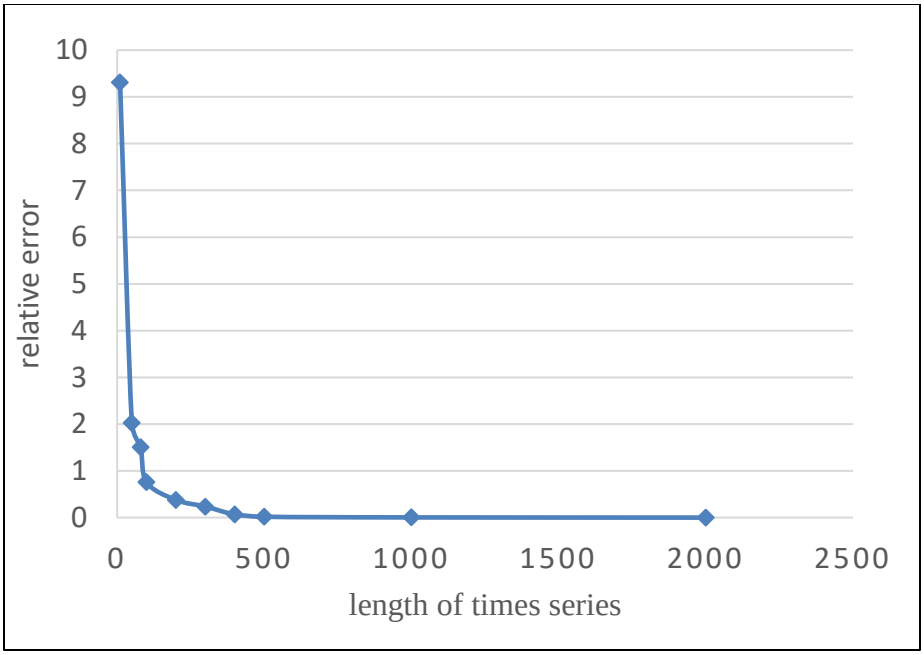


Figure 4.4 Offline relative error vs. length of training data function (reprinted from [48] with permission).

The model selection procedure as depicted in Figure 4.2 step 4 is based on selection of the optimum hard threshold for the algorithm. There are many techniques for model selection in machine learning, among them, the popular ones are Akaike Information Criteria (AIC) [81], Bayesian Information Criteria BIC [82], Pareto front analysis [83] and cross validation [84]. Pareto front analysis finds the threshold that balances between the level of sparse and the accuracy. The information criteria will select models that balance the information loss and model complexity. Alternatively, we could possibly use a sampling technique such as the Markov chain Monte Carlo [64, 85-87] to explore the whole distribution of the error and decide the optimum tolerance that minimize the error.

Here we will be using an algorithm that applies the cross-validation technique. The algorithm is developed by Rudy et al [37] and we are showing it briefly in the flow chart

of Figure 4.5. Notice in step 3 (in Figure 4.5), the algorithm uses ridge regression, which gives a shrinkage estimate, meaning a reduced variance but a biased solution. The ridge regression problem consists of a least square error plus a l_2 penalty term as described in Eq. 4.12.

$$min\left\|\Theta^{train}\xi_i - \dot{X}_i^{train}\right\|_2^2 + \lambda\|\xi_i\|_2^2 \tag{4.12}$$

The solution can be computed directly as

$$\widehat{\xi_i} = \left(\left(\Theta^{train}\right)^T\Theta^{train} + \lambda I\right)^{-1}\left(\Theta^{train}\right)^T\dot{X}_i^{train}. \tag{4.13}$$

Although the ridge regression will introduce bias, by adding an extra λ on the diagonal of the matrix, we make the matrix inversion more stable. This biased solution will not have any effect in applying the hard threshold in what comes next. After the sparsity pattern is determined, the algorithm provides an unbiased estimate by solving an exact least square problem with gaussian elimination or conjugate gradient solver.

Finally, we perform the validation step. After identifying the governing ODEs, we need to verify if the coefficients identified are correct. Due to the large scale and complex nature of our reservoir simulator, instead of calculating all the coefficient of the actual ODE equation, we apply an alternate way. One way is to use the original library matrix we have and multiply by the calculated coefficient matrix to get an offline prediction of $\frac{\partial X}{\partial t}$. The error between the offline $\frac{\partial X}{\partial t}$ and the real one is what we call model building error (or offline error). The other way is to use Newton Raphson method to actually solve the identified ODE system with an initial value. The difference between the results that comes

from simulating the identified ODE system and original simulator is what we call simulation error (or online error).

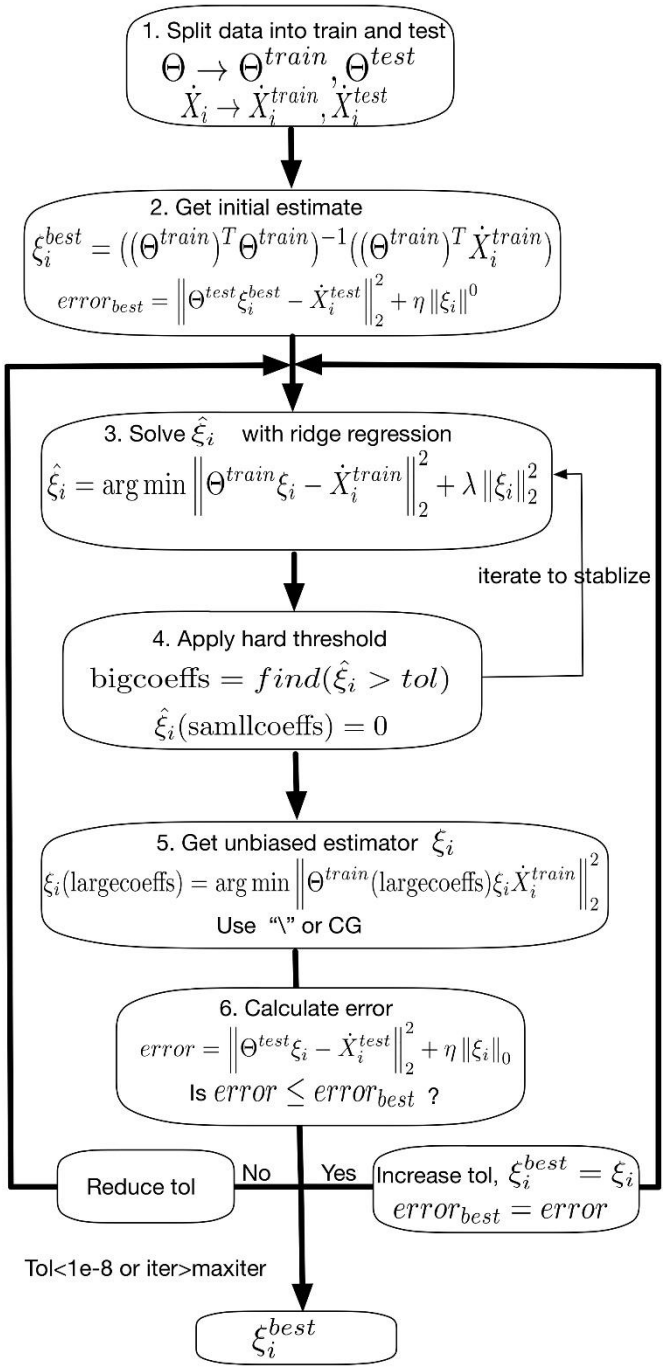


Figure 4.5 Flowchart for optimum threshold algorithm function (reprinted from [48] with permission).

4.4. Lasso

Lasso is another method widely known in the area of statistical learning to identify sparse system. The problem is usually formulated as to optimize the sum of a least square loss and l_1 norm of the coefficient (Eq. 4.14). This optimization problem, to its nature, is an application of soft threshold. In our specific reservoir simulator example, we found the ADMM [36] works best among other Lasso method in terms of accuracy, which is also the one used in our previous work [78]. We will briefly use the Lasso method in the single phase flow model and compare it with SINDy as in Table 4.2. The objective function with classical Lasso with l_1 Penalty is

$$\min_{\xi} \frac{1}{2} \left\| \Theta\xi - \dot{X} \right\|_2^2 + \gamma \|\xi\|_1. \tag{4.14}$$

Converting Eq. 4.13 to to the ADMM framework, it becomes

$$\min_{\xi_i} \frac{1}{2} \left\| \Theta\xi - \dot{X} \right\|_2^2 + \gamma \|z\|_1 , \tag{4.15}$$

$$\text{subject to} \quad \xi - z = 0.$$

Finally, the optimization problem is solved through augmented Lagrangian equation, as in Eq. 4,15, and depicted below,

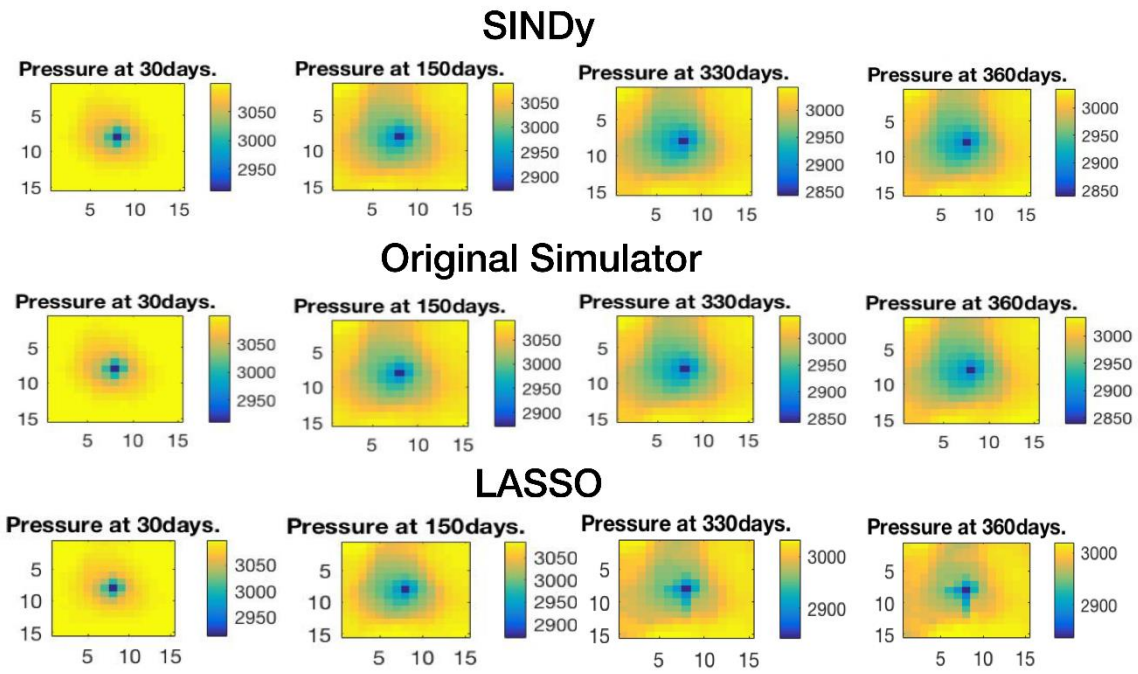
$$L_\rho(\xi,\, y,\, z) = \frac{1}{2} \left\| \Theta\xi - \dot{X} \right\|_2^2 + \gamma \|z\|_1 + \frac{\rho}{2} \|\xi - z\|_2^2 + \rho\, y^T(\xi - z). \tag{4.16}$$

4.5. Results and Discussion

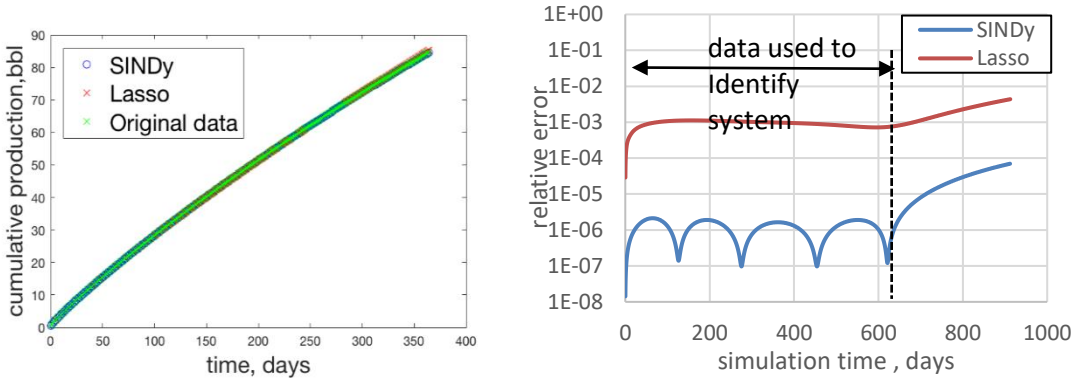
4.5.1. Two-Dimensional Single-Phase Pressure Prediction

In the case of single-phase flow models, we formulate the pressure in each grid block as the unknown state, and thus a system of ODE can be identified with sparse

regression using hard threshold method (SINDy) or soft threshold (LASSO). Here we have a 15x15 heterogeneous single-phase flow reservoir with one production well in the center, that is a typical depletion problem. As seen from Figure 4.6 and Table 4.2, the increased amount of grid block will affect the speed and accuracy of the model. As the equation number increases (Table 4.2), the model may suffer loss of accuracy (some figure distortion and added error in Figure 4.6), but it still has control on the overall results and maintain a correct production profile (Figure 4.6b). From our numerical experiments, SINDy works better than Lasso in terms of relative error (Figure 4.6c) and takes less computational time (minutes vs. hours) especially for large dimensional ODE system. However, the disadvantage of SINDy is that it takes several numerical experiments to decide which is the best initial hard threshold (or the range of the initial hard threshold for cross validation part) to run the simulation. For the Lasso with ADMM algorithm, we barely need to tune any initial parameters to achieve a feasible result (a set of identified ODE that could be solved by Newton iteration) since the parameter for penalty term γ (Eq. 4.15) is not significantly crucial as compared with the hard threshold λ (in Figure 4.2).



(a)



(b) (c)

Figure 4.6 Simulation results from three different method for single phase 225 grid block heterogeneous reservoir, (a) Pressure evolvement through 360 days. (b) Comparison of cumulative production for 225 grid block reservoir (c) relative online error compared with original simulator (semi-log scale). Reprinted from [48] with permission.

Table 4.2 Comparisons of relative error between identified system and original simulator (reprinted from [48] with permission).

Grid Block	Algorithm	Simulation Error (Online)	Model Building Error (Offline)	Figure Distortion
3x3, Initial Tol=0.005	Sindy	2.595e-7	2.069e-05	No
15x15, Initial Tol=0.002	Sindy	1.1836e-5	0.019	Almost None
15x15, γ=1e-4	Lasso	0.0013	0.195	Some

We could further investigate the coefficients of the single-phase flow equations to see if the discovered equation is reasonable or if it is indeed a physical equation. Here, we present some of the selected equations to observe. Cell 113 is the one we put the production well. As we see this is the only one that has non-zero values for the constant term that comes from the constant term in the well term $WI \cdot P_{wf}$. In fact, this is a constraint we manually enforced when solving the linear system for the coefficient. In addition, it is not surprising to notice that the dominant terms in each equations are the first order linear terms (Table 4.3), this is because the rock and fluid properties are strong linear function of pressure (Figure 4.7). This result holds true for most of the cells in the single-phase system. The large coefficients are selected based on the hard threshold that we obtained from the cross-validation algorithm described in Figure 4.5. Any coefficient that has absolute value less than the hard threshold is forced to be considered zero in the next iteration. Within each iteration, the hard threshold will be adjusted to achieve a smaller error. The final hard threshold as we can tell from table 4.3 is around $1e-12, 1e-11$ and $1e-11$ for cell 1, cell 7 and cell 113 respectively.

Table 4.3 Coefficients of three selected ODEs in a 15x15 system. The coefficients are three columns vectors in matrix Ξ in Eq. 4.6.

	Cell 1			Cell 7			Cell 113	
	terms	coeff		terms	coeff		terms	coeff
const term	1.00E+00	0	const term	1.00E+00	0	const term	1.00E+00	9.82
single terms	P_1	-2.35E-03	single terms	P_7	-3.82E-03	single terms	P_{113}	-9.09E-03
	P_2	-1.30E-03		P_6	8.65E-04		P_{98}	-8.39E-03
	P_{16}	3.45E-03		P_8	-8.06E-04		P_{112}	1.95E-02
	P_1^2	-4.44E-07		P_{22}	4.22E-03		P_{114}	2.24E-03
	P_2^2	-3.44E-07		P_7^2	3.07E-06		P_{128}	5.79E-04
	P_{16}^2	5.70E-07		P_6^2	7.42E-07		P_{113}^2	-2.92E-06
	P_1^3	1.84E-10		P_8^2	1.07E-06		P_{98}^2	4.50E-06
	P_2^3	-8.61E-11		P_{22}^2	-1.49E-06		P_{112}^2	-3.48E-06
	P_{16}^3	-9.02E-11		P_7^3	2.75E-10		P_{114}^2	4.19E-06
mixed terms with cell on the right	P_1P_2	3.47E-07		P_6^3	-4.07E-10		P_{128}^2	9.82E-07
	$P_1^2P_2$	0.00E+00		P_8^3	1.04E-10		P_{113}^3	2.70E-10
	$P_1P_2^2$	-1.81E-12		P_{22}^3	5.28E-10		P_{98}^3	-3.51E-10
	P_1P_{16}	-3.19E-10	mixed terms with cell on the right	P_7P_8	-2.97E-06		P_{112}^3	-9.00E-11
	$P_1^2P_{16}$	3.03E-10		$P_7^2P_8$	7.65E-07		P_{114}^3	-5.04E-10
	$P_1P_{16}^2$	-1.07E-11		$P_7P_8^2$	-2.35E-11		P_{128}^3	3.00E-10
				P_7P_{22}	-9.86E-10	mixed terms with cell on the right	$P_{113}P_{114}$	-4.14E-06
				$P_7^2P_{22}$	6.42E-11		$P_{113}^2P_{114}$	-1.46E-06
				$P_7P_{22}^2$	-7.77E-11		$P_{113}P_{114}^2$	6.87E-10
			mixed terms with left cells	P_7P_6	-1.50E-06		$P_{113}P_{128}$	-8.15E-10
				$P_7^2P_6$	8.82E-10		$P_{113}^2P_{128}$	-8.49E-11
				$P_7P_6^2$	-3.07E-10		$P_{113}P_{128}^2$	1.90E-11
						mixed terms with cells on the left	$P_{113}P_{112}$	2.88E-06
							$P_{113}^2P_{112}$	-4.56E-06
							$P_{113}P_{112}^2$	9.67E-10
							$P_{113}P_{98}$	-2.98E-10
							$P_{113}^2P_{98}$	-1.61E-09
							$P_{113}P_{98}^2$	1.83E-09

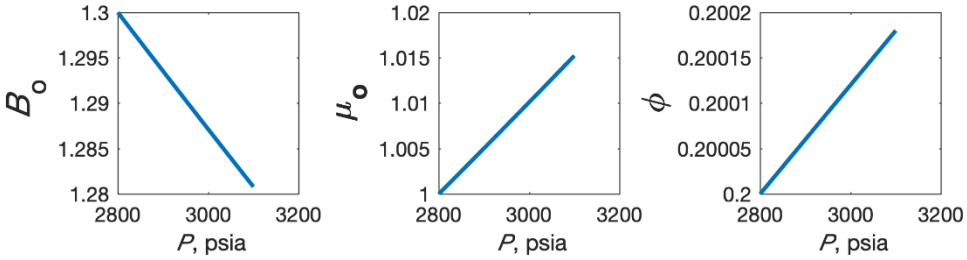


Figure 4.7 Rock and fluid properties of single-phase reservoir model

4.5.2. Two-Dimensional Two-Phase Pressure/Saturation Prediction

As mentioned before, we are going to clearly specify the nonlinear terms and calculate their associated coefficients in the saturation equation with sparse regression method. The example here is from a mix formulation of two-phase flow model with a homogeneous reservoir and inverted five spot production profile, four production well in the corner and one injection well in the center.

We notice that the identified ODE system may suffer increasing relative error after running for a long time (after 580, which is the amount of data we used to training). But the overall error is constrained below 6% even at the end of simulation. Notice that the two-phase example yielded a larger error compared with the single-phase. In order to enhance the performance of the sparse regression algorithm, we attempt to add constraints for the polynomial approximation of the fractional low function (Eq. 4.11). From the fact that $0 < f_w < 1$, we derived the constraints as follows,

$$\text{Inequality constraints: } 0 < A_{cons}\xi_i < 1. \tag{4.17}$$

We illustrate this with one dimensional system. For the $i\,th$ gridblock, the equality constraints mentioned above becomes Eq. 4.18. Notice this is only for a certain timestep. One can add more constraints by selecting data from multiple timesteps. However adding

constraints needs to be done cautiously since the more constraints exist, the less the degree of freedom is left which may introduce increasing bias into the approximation.

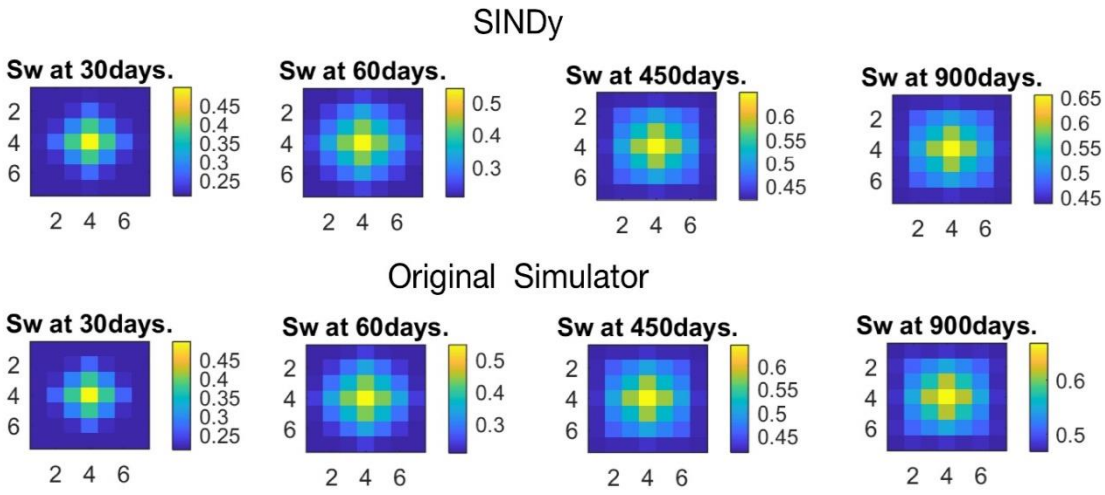
$$\begin{bmatrix} 1 & 0 & \cdots & S_{w,i-1}^4 & S_{w,i-1}^3 & S_{w,i-1}^2 & S_{w,i-1} & 0 & \cdots & & & & & & & 0 \\ 1 & 0 & \cdots & & & & & 0 & S_{w,i}^4 & S_{w,i}^3 & S_{w,i}^2 & S_{w,i} & 0 & \cdots & & 0 \\ 1 & 0 & \cdots & & & & & & & & & & 0 & S_{w,i+1}^4 & S_{w,i+1}^3 & S_{w,i+1}^2 & S_{w,i+1} & \cdots & 0 \end{bmatrix} \xi_i = \begin{bmatrix} f_{w,i-1} \\ f_{w,i} \\ f_{w,i+1} \end{bmatrix}$$

(4.18)

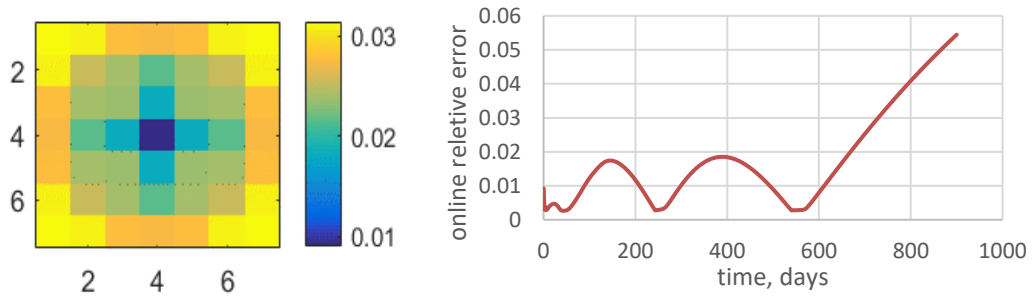
where

$$\xi_i = \begin{bmatrix} \xi_0 & \xi_{1,1} & \cdots & \xi_{i-1,1} & \xi_{i-1,2} & \xi_{i-1,3} & \xi_{i-1,4} & \xi_{i,1} & \xi_{i,2} & \xi_{i,3} & \xi_{i,4} & \xi_{i+1,1} & \xi_{i+1,2} & \xi_{i+1,3} & \xi_{i+1,4} & \cdots & \xi_{n,4} \end{bmatrix}^T.$$

This constrained problem will be solved along with the ridge regression problem as in step 3 (in the algorithm in Figure 4.5) and the least square problem in step 4 (in the algorithm in Figure 4.5). The tool used for solving constrained optimization is the *fmincon* function in Matlab [88], which by default applies the algorithm BFGS for Hessian approximation. As seen in Table 4.4, with initial hard threshold equals 1e-7 and considering a constant term in the library, the model performs better and gives smaller error during online simulation. As shown in Figure 4.10, the benifit of using constraint SINDy is that the fractional flow curve has a better match and lies in a reasonable range (between 0 and 1). Finally, notice that the constraints here are within each cell/ODE equation. In the future, we may try constraints that draw connections between cells. As states by Loiseau and Brunton [89], those constraints may come from physical fact such as mass/energy conservation.



(a)



(b) (c)

Figure 4.8 Simulation results from identified ODE (by SINDy) and reservoir simulator for two-phase 49 grid block homogeneous reservoir with initial hard threshold=0.0001, (a) Saturation evolvement through 900 days (b) The elementwise relative error map throughout entire timesteps (c) relative online error compared with original simulator (semi-log scale). Reprinted from [48] with permission.

Table 4.4 Results comparisons from unconstraint and constraint problem for two-phase flow reservoir.

Initial hard threshold=1e-7, with constant term		offline error	online error
unconstrained	SINDy	0.0291	0.0691
Inequality constraints	Ridge regression+fmincon	0.0247	0.1120
Initial hard threshold=1e-7, without constant term			
Unconstrained	SINDy	0.0174	0.1907
Inequality constraints	Ridge regression+fmincon	0.0166	0.1955

As with the single-phase flow, we assess the results to check if it indeed allows for discovery of the conservation equations. We will check on the fractional flow function generated from the discovered coefficients. As seen in the following two figures, the fractional flow curve roughly follows the true solution for each of the grid blocks in the reservoir simulator. The constraint SINDy gives a better match than the unconstraint SINDy. In addition, the constraint SINDy managed to keep all values of fractional flow between 0 and 1.

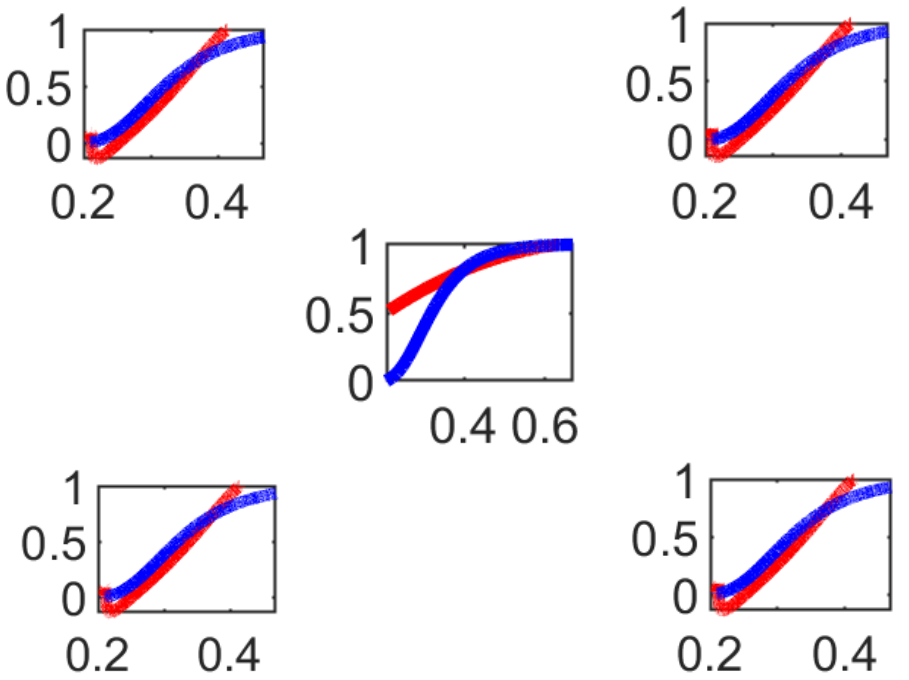


Figure 4.9 The comparison of fractional flow curve from unconstraint customized SINDy (Red line is from reconstructed f_w, blue line is the true value). x axis is saturation, y axis is fractional flow function value. Four plots on the corner are for production well grid block. One plot in the middle is for injection well grid block.

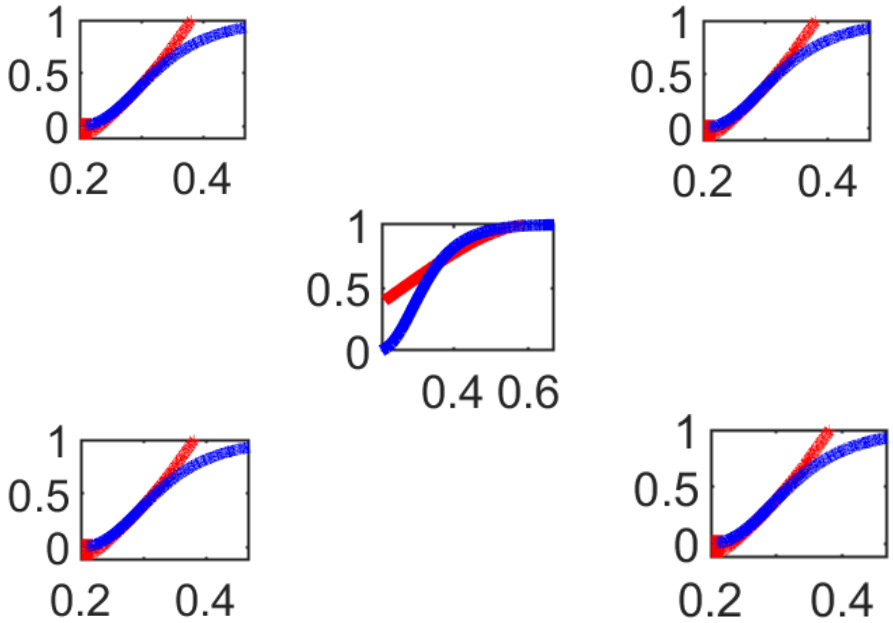


Figure 4.10 The comparison of fractional flow curve from constraint customized SINDy (Red line is from reconstructed f_w, blue line is the true value). x axis is saturation, y axis is fractional flow function value. Four plots on the corner are for production well grid block. One plot in the middle is for injection well grid block.

The application of the system identification technique can be extended to a more realistic reservoir system, such as the well-known SPE-10 model. However, as reservoir model gains complexity, the proxy model we use should be adjusted accordingly. For example, we could apply more sophisticated polynomials such as B-spline or Chebyshev polynomials [90] to gain extra power. Or we could apply extra constraints to regularize the problem and incorporates deeper constrains on mass or energy conservation [91]. In addition, the change of control instead of constant control can be taken into consideration as another factor for realistic oil reservoir. Those are potential work that can be explored in the future work.

In this chapter, we have shown that our customized sparse identification of the nonlinear dynamic (SINDy) is a viable algorithm for identifying both single-phase and two-phase flow dynamic system. Comparing with Lasso, SINDy tends to reveal the dynamic system more accurately, which might because SINDy provides an unbiased solution compared with Lasso. In addition, it is of importance to choose the parameters such as the length and resolution of the training data and the initial hard threshold (or the range of the hard threshold we are doing cross validation on). The structure of the reservoir grid block (1D/2D/3D) should also be taken into consideration to further regularize the problem. Finally, adding constraints to the sparse regression will aid in improving accuracy of the solution and reducing overfitting. We will further investigate on the constraint problem in terms of robust optimization algorithm and optimum parameters and develop various types constraints that could incorporates crucial physical fact such as the

mass or energy conservations. Other future work may include combining the sparse regression problem with complex control, large dimension heterogeneous reservoirs.

5. RECURRENT NEURAL NETWORK FOR RESERVOIR PROXY MODEL

5.1. Introduction and Motivation

As shown before in chapter 2, reservoir simulation aims to solve highly nonlinear ODEs which involves both time and spatial variables. In addition, it is of large dimension spanning millions to billions of equations- for example the SPE10 benchmark [79] results in a system of 2.244 million ODEs (from the discretization of PDE on 1.122 million grid blocks), which requires gradient calculation and numerical iterations to solve.

A general overview of possible solutions to deal with the large dimension is shown below.

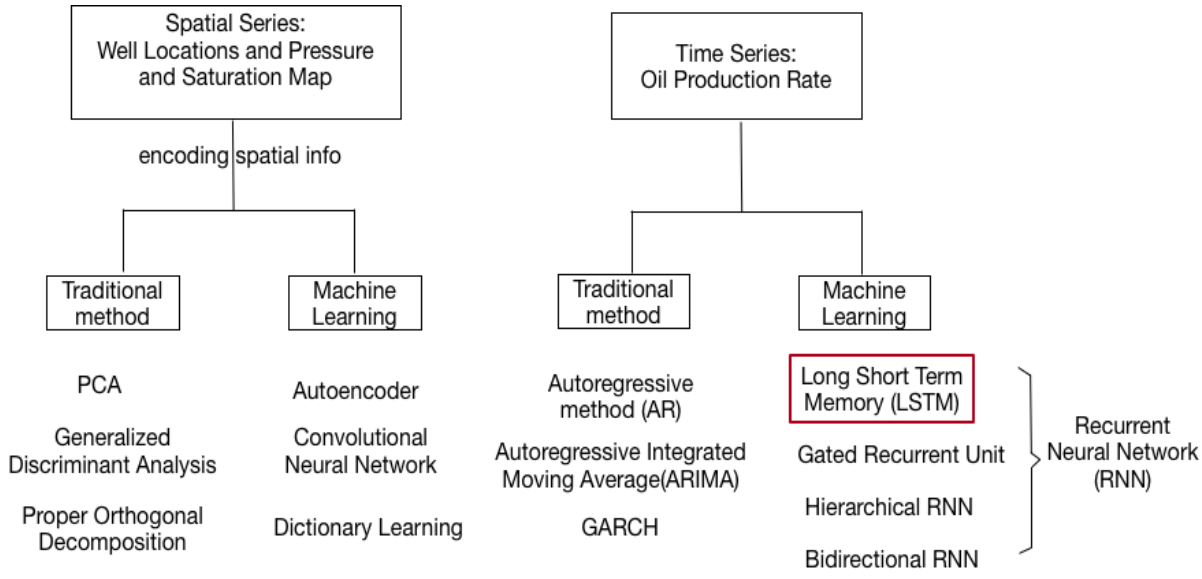


Figure 5.1 Overview of complexity reduction methods with its possible solutions

In this chapter, we explore the applications of LSTM network on time series oil production data in order to

- Avoid sophisticated calculation of physical connections between each of the variables
- Reduce complexity by avoiding space discretization into millions of grid blocks.
- Enhance stability of the solution by avoiding gradient based numerical iteration

5.2. Problem Formulation

5.2.1. Overview of Reservoir Simulation and Production Prediction

We consider developing proxy model of two-phase waterflooding reservoir for production prediction. The core content is how to use the multivariate time series to perform prediction with the deep LSTM model. Recall the multiphase governing equation 2.1, we have Eq. 5.1-5.2 for the two-phase flow equation. We noted here that either solving the system with fully implicit or sequential simulator, the process is recursive in nature, as seen in Eq. 5.3 that the current timestep is highly dependent on the previous timestep. The production response will be computed as in Eq. 5.4. In this work, we will solve the system using the mix formulation sequential simulator as stated in chapter 2 for computation efficiency (Eq. 2.12-2.25).

$$\frac{\partial}{\partial t}(\varphi \rho_w s_w) - q_w = \nabla \cdot \left(\frac{k_{rw}\rho_w}{\mu_w} K(\nabla P) \right) \tag{5.1}$$

$$\frac{\partial}{\partial t}(\varphi \rho_o s_o) - q_o = \nabla \cdot \left(\frac{k_{ro}}{\mu_o} K(\nabla P) \right) \tag{5.2}$$

$$(P^t, s_w^t) = f(P^{t-1}, s_w^{t-1}, q_w^t, BHP^t) \tag{5.3}$$

$$q_o^t = (1 - f_w(s_w)) * q_{total}^t = g_o(P^t, s_w^t, BHP^t) \tag{5.4}$$

The main goal of this end to end process is to avoid the computation of the state variables explicitly, and thus we will only focus on the final output, which is the total rate and water cut. We replace the highly nonlinear and implicit recursive function with the deep LSTM network, where the input are control parameters (injection rate q_w and production bottom hole pressure BHP), previous water cut and production rate. This formulation (Eq. 5.5 and 5.6) is motivated with recursive equation as indicated in Eq. 5.3 and 5.4. As seen in Eq. 5.3, the current the state variable (pressure and saturation) is a function of the previous step state variables and the current step control parameters. Furthermore, the current production rate q_{total}^t and water cut f_w^t is a function of the current state variables. Those information leads to the flowing Eq. 5.5 and 5.6.

$$f_w^t, q_{total}^t = f_{LSTM}\left(q_{total}^{t-1,\ldots,t-n}, f_w^{t-1,\ldots t-n}, q_w^{t,\ldots,t-n+1}, BHP^{t\ldots t-n+1}\right) \tag{5.5}$$

$$q_o^t = (1 - f_w^t) * q_{total}^t \tag{5.6}$$

5.2.2. The LSTM Network

Figure 5.2 shows the schematics of a sample LSTM cell. The h_t and c_t is the current cell output and the current cell states. x_t is the current input multivariate time series. There are in total three gates that controls the forget, update and output behavior of the LSTM model. The three gates are associated with four activation functions, which are either sigmoid or hyperbolic tangent functions. As seen from Figure 5.2, the cell state c_t is the variable that connect information from all cells/timesteps, which is the essential tool of LSTM to preserve long-range time dependencies (for more details refer to [56]).

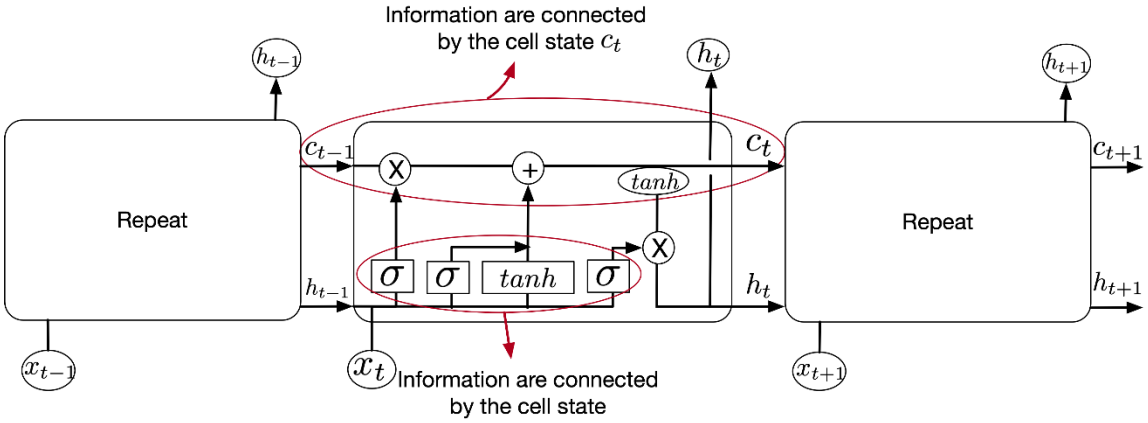


Figure 5.2 Structure of LSTM cell. Adapted from [56].

Looking into details of the four gates and activation function, we have the following equations for illustration purpose. Eq. 5.7 is the forget gate with sigmoid σ function that will decide which components to forget. Eq. 5.8 and 5.9 together assembled the update gate. i_t decide which component to be updated and $\tilde{c}_t$ provides the changed cell states. $[w_f, w_i, w_c, w_o]$ and $[b_f, b_i, b_c, b_o]$ are the weights of the layers that related with each activation function. Eq. 5.10 will calculate the updated cell states based on forget gate and update gates. Finally Eq. 5.11 and 5.12 are the output gate that will decide what part of the cell states to output.

$$f_t = \sigma(w_f \cdot [h_{t-1}, x_t] + b_f) \tag{5.7}$$

$$i_t = \sigma(w_c \cdot [h_{t-1}, x_t] + b_i \tag{5.8}$$

$$\tilde{c}_t = tanh(w_c \cdot [h_{t-1}, x_t] + b_c) \tag{5.9}$$

$$c_t = f_t * c_{t-1} + i_t * \tilde{c}_t \tag{5.10}$$

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o) \tag{5.11}$$

$$h_t = o_t \tanh(c_t) \tag{5.12}$$

The LSTM networks we will use here are the deep LSTM (Figure 5.3) and Cascaded deep LSTM (Figure 5.4). The cascaded deep LSTM will have an extra D-LSTM that model the breakthrough time and feed the output to the next D-LSTM as an extra feature. As we will demonstrate in the results, when feeding the breakthrough time into the second network, we have to recast it as a nominal variable that will have either a value 1, if the water front reaches the well, or a value of 0 before the water breakthrough. The motivation for using Cascaded D-LSTM is from the observation in Figure 5.12, where standard D-LSTM fails to accurately predict the water breakthrough time especially for the second well. We will explore this fact further in this chapter.

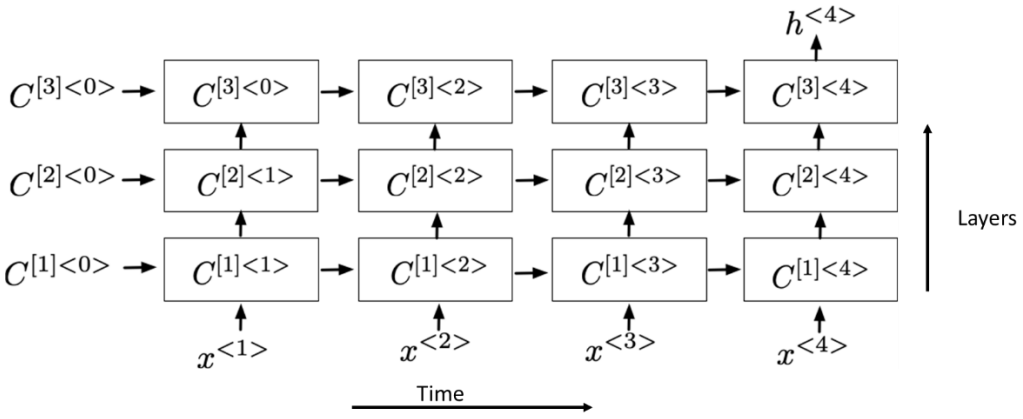


Figure 5.3 Deep LSTM Network. Adapted from [92].

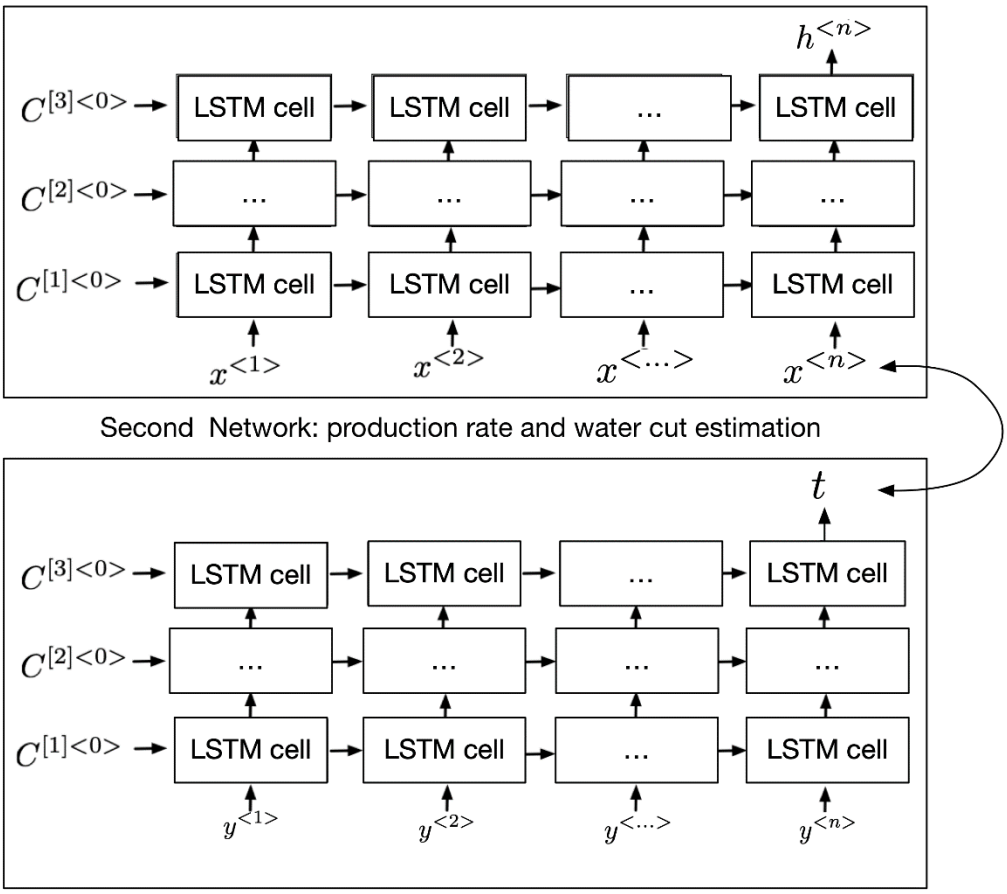


Figure 5.4 Cascaded deep LSTM network. The output t of the first networks represents the predicted breakthrough time.

5.2.3. Training Strategy

We deploy a typical training schemes of LSTM called sliding window technique, where the input variable will be a sequence of time series data with length *n*, and *n*+1 is our window size. As shown in the Figure 5.5, window size is 50. The first 49 timestep will be the length of input data and the last one will be our output results.

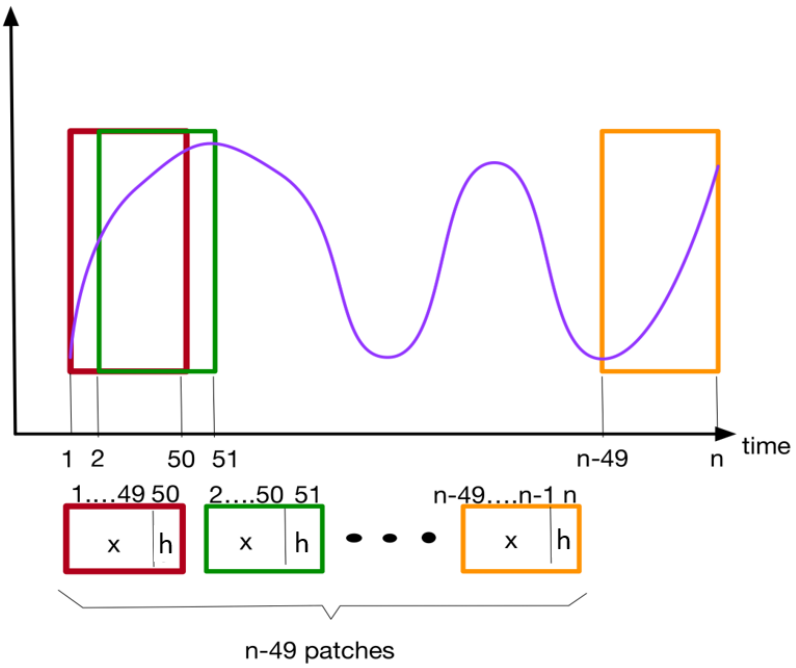


Figure 5.5 Sliding window scheme for training LSTM network.

In our specific production forecasting problem, x is the input multivariate time series composed of previous production rate, water cut and control parameters (bottom hole pressure *BHP* and injection rate q_w) as

$$x = [q_{total}^{t-1,\ \ldots,t-n}, f_w^{t-1,\ldots t-n},\ q_w^{t,\ldots,t-n+1}, BHP^{t\ldots t-n+1}],$$

and *h* is the output total production ate and water cut at each well, that is, $h = [q_{total}^{t}, f_w^{t}]$.

5.2.4. Bayesian Optimization

As noticed in the LSTM model, there are many hyperparameters required to create an optimum model, such as window size, number of neutrals, number of layers number of iterations, activation functions etc. Tuning the hyperparameters is critical to the problem but the process is time consuming. Several hyperparametric tuning algorithm have been

proposed and among them, the Bayesian optimization is one of the most popular algorithms. We implemented the Bayesian optimization algorithm with the python package Sklearn [93]. The procedure is basically composed of two parts: the initialization and update loop. The function to be optimized is the average loss function from a three folds cross validation of the model. In the initialization part, we define the prior for the objective function as a gaussian process, with Matern type kernel for covariance matrix estimation and initialize the prior with several observation points. Later in the update loops, we update the posterior with new sampling point, which is obtained through a certain type pf acquisition function.

Import sklearn.Gaussian_process as gp

Initialization

- Place a Gaussian process prior on f
  - kernel = gp.kernels.Matern()
  - model = gp.GaussianProcessRegressor(kernel=kernel)
- Observe true value of f at $n0$ points from true loss function, set n=n0
  - (y_i=cross_val_loss(x_i), for i=1…n0)

While $n \leq N$ **do:**

- Update the posterior probability distribution on f using all available sample (model.fit(x_n,y_n))
- Identify the maximizer $\boldsymbol{x'}$ of the **acquisition function** over parameter domain $\mathcal{X}$, where the acquisition function is calculated using the current posterior distribution
- Observe $y' = f(x')$
- y_n =[y_n,y'], x_n =[x_n , x']
- Increment n

There are several types of acquisition functions that has been widely used, including probability of improvement, expected improvement, GP-UCB [94] etc. The acquisition function we used in this work is expected improvement.

5.2.5. Ensemble Kalman Filter

Since we are tackling the production prediction problem, in the real oil field, as the production process continues, the physical condition might change. The current model may not be sufficient for a new case, or an updated model change. We hypothesize here that it is better to have our model updated according to the real time observations streaming out of any production record. This is the so-called data assimilation process [95], and one of the popular techniques is the Ensemble Kalman Filter (EnKF) that incorporates new data as it becomes available to improve the existing model. As shown below (Figure 5.6), we will adopt the same methodology as the EnKF to define the new weights of the system required in STM network based on the true observations. In Figure 5.6, y_j^n is the vector that consists of LSTM weights w_j ($N_m \times 1$) and response variables d^n ($N_n \times 1$). d^n consists of the total production rates and water cuts and is the same as *h* as described in last section 5.2.3. Superscript *n* is the *nth* time step, and subscript *j* is the *jth* the ensemble. Superscript *f* represents the updates from mathematical model and superscript *a* denotes the updates from observations. $d_{uc,j}^n$ is the perturbed observation point based on true observation d_{obs}^n and covariance C_{Dn}. H_n is the matrix that will map the vector *y* into response vector *d*. *O* is null matrix with dimension $N_n \times N_m$ and I_{N_n} is identity matrix with dimension $N_n \times N_n$.

$$y_j^n = \begin{bmatrix} w_j \\ d^n \end{bmatrix} \tag{5.13}$$

$$d_{obs}^n + \varepsilon_j = d_{uc,j}^n \sim N(d_{obs}^n, C_{Dn}) \tag{5.14}$$

$$d_j^{n,f} = H_n y_j^{n,f} \tag{5.15}$$

$$H_n = [O\ I_{N_n}] \tag{5.16}$$

Generating ensembles of the parameters is critical to the successes of EnKF. In this work, since the weights of the LSTM network are between 0 and 1, we will simple choose the covariance to be 0.1 for generating 100 ensembles from a normal distribution $W \sim Normal(w_0, 0.1)$, where w_0 is the initial weights generated by the trained cascaded LSTM network. The number 100 is used as suggested in [96].

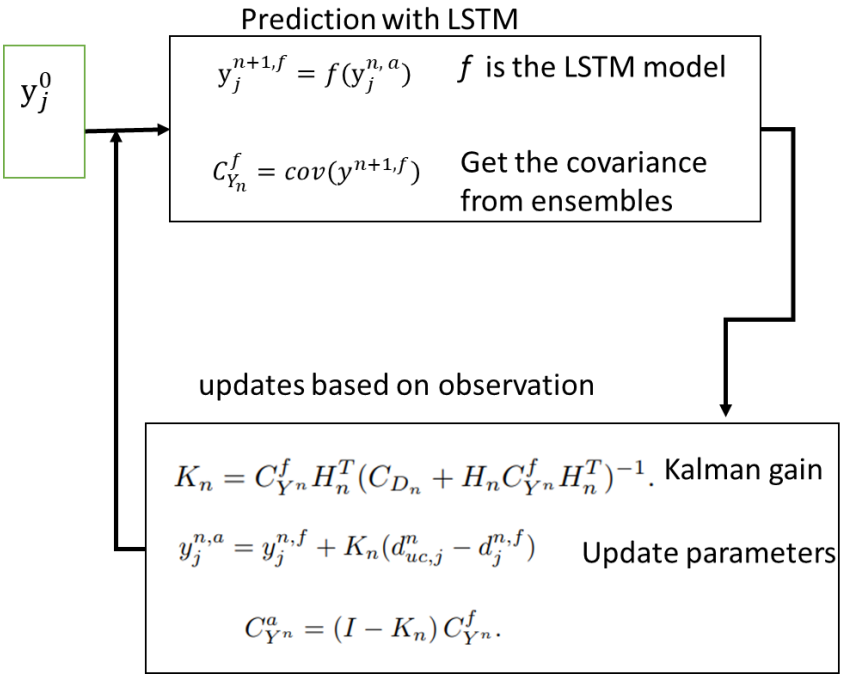


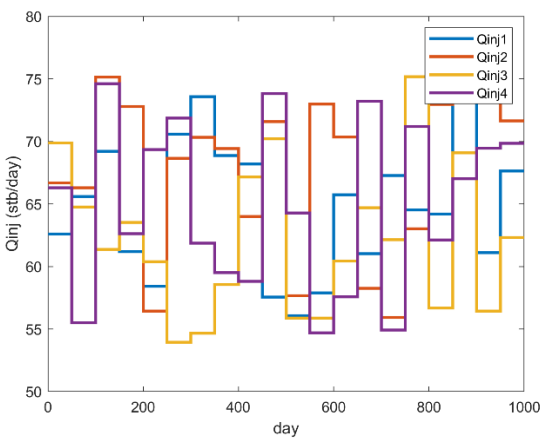
Figure 5.6 Workflow of EnKF model update scheme. The observation is production rate and water cut values. The unknow parameter are weights of the LSTM network. The loop consists of prediction with LSTM and updates with EnKF algorithm. Within each iteration, the uncertainly of the unknown parameters will be reduced using observation data.

5.3. Experiments and Results

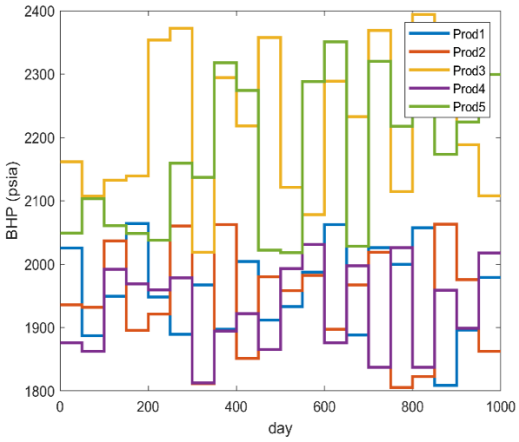
5.3.1. The Training Dataset

Features we used in the neural network are the control parameters: *BHP* (bottom hole pressure) and q_w (injection water rate profile). The reservoirs simulation model we use to generate data is the SPE 10 model (one layer) with a nine spot production scenario (four injector in the corner and five producer in the middle).

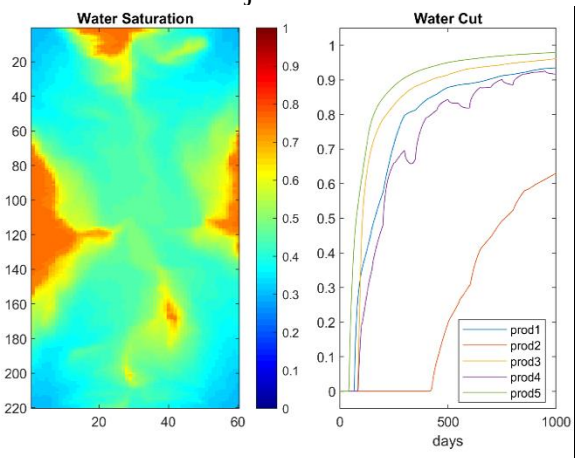
The bottom hole pressure data for well 1 and 2 are drawn from the distribution of Uniform (1800, 2070) psia, while the BHP for wells 3, 4, 5 are from Uniform (2000, 2400) psia. The injection rate for each of the four injectors is generated from different uniform distribution Uniform (q_w, 1.4q), where q is different for each case. For the ten case I generated, the $q_w \in (0.333, 0.285, 0.25, 0.222, 0.2, 0.182, 0.167, 0.154, 0.143, 0.125)$ pore volume/1000 days. The total injection volume of the four well over the 1000 days will follow uniform distribution Uniform (Q_w , $1.4Q_w$), with $Q_w \in (1.333, 1.143, 1, 0.889, 0.8, 0.727, 0.667, 0.615, 0.571, 0.5)$ pore volumes/1000 days. Below are two sample training datasets (Figure 5.7 and Figure 5.8).



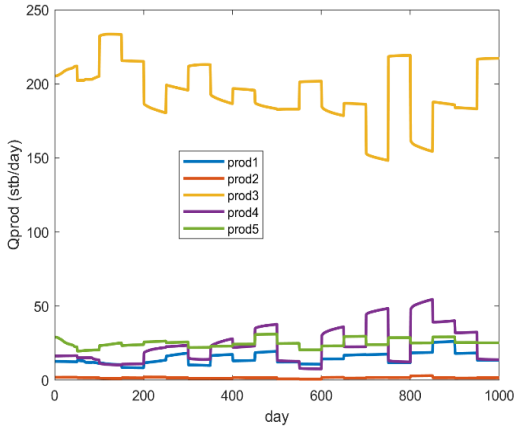
Injection rate



Production Bottom Hole Pressure

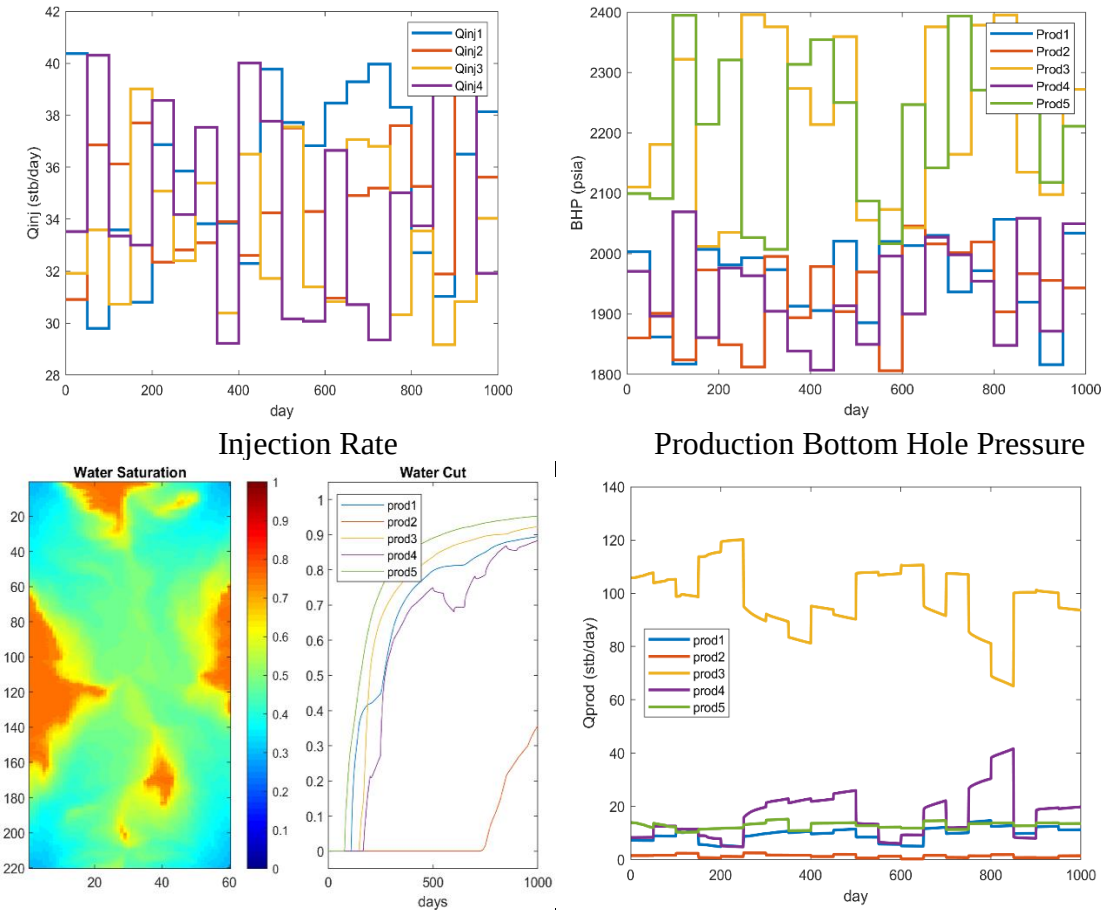


Water Cut



Prodution Rate

Figure 5.7 Sample training scenarios with injection rate follow Uniform (q_w, 1.4 q_w), where q_w =0.285 pore volume/day.



Injection Rate | Production Bottom Hole Pressure

Water Cut | Prodution Rate

Figure 5.8 Sample training scenarios with injection rate follow Uniform (q_w, 1.4 q_w), where q_w = 0.167 pore volume/day

5.3.2. Building Model and Fine Tune with Bayesian Optimization

We first build the model with regular LSTM network. Then consider the cascade LSTM with a new feature breakthrough indicator from an extra LSTM network built on the basic network. We will apply Bayesian optimization to both regular LSTM and cascaded LSTM network. Finally compare the results of all three cases.

We illustrate the usefulness of Bayesian optimization starting from tuning two parameters, the window size and number of neurons. As seen in the Figure 5.9, the

algorithm is working towards a global minimum of the objective function based on higher value of mean and higher uncertainty (high variance value).

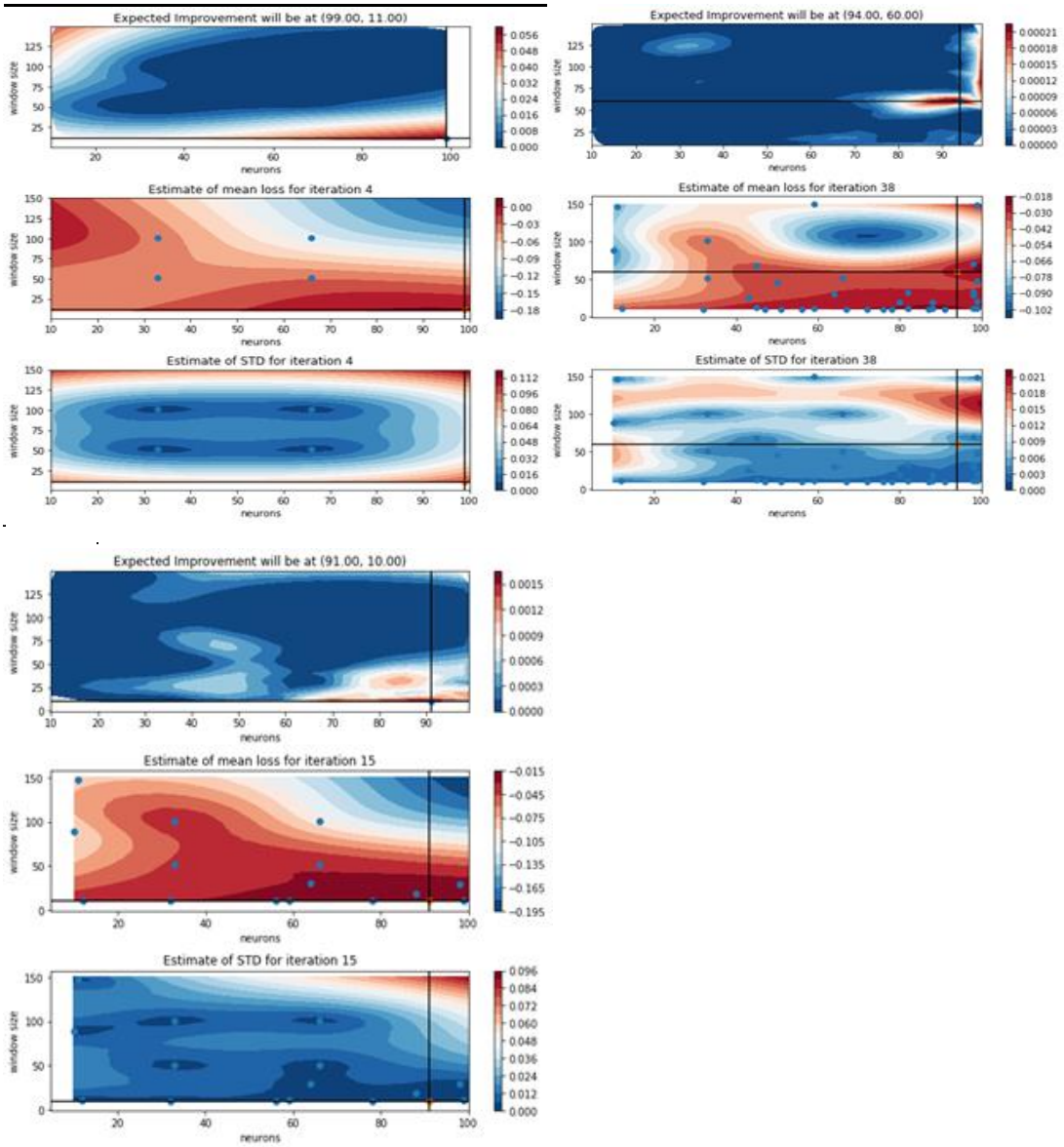


Figure 5.9 Bayesian optimization over iterations. Top row: expected improvement function. Middle row: estimated mean from gaussian process. Bottom row: estimated variance from gaussian process.

We further tune three parameters together: number of layers, window size, number of neurons to get optimum model. Figure 5.10 is a comparison of three hyperparameters

tuning from three method, Bayesian estimation, random search and grid search. Out of the three method, BO performs better in terms of accuracy and efficiency.

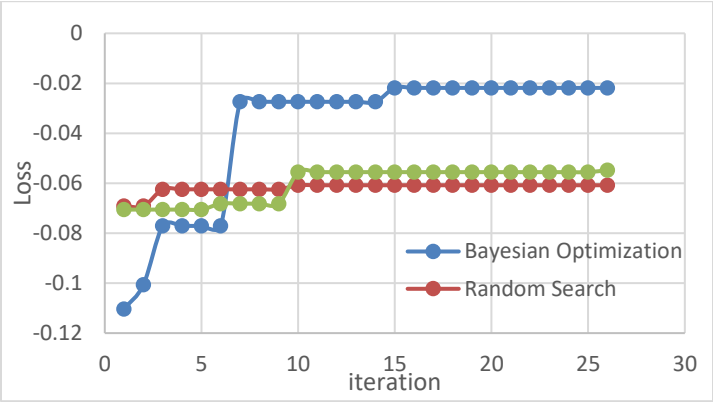


Figure 5.10 Tuning three hyperparameters.

5.3.3. Cascaded LSTM

However, even though Bayesian optimization gives a better model by tuning the hyper parameters, the model performs poorly in predicting the water breakthrough time and water cut values as we will see later in Figure 5.12 and Figure 5.11. In this case, we propose a cascaded LSTM model that added a simpler LSTM model to predict breakthrough time on top of the original model. In the future we also consider combining with streamline calculation and generalized travel time inversion (GTTI) [97-99] for more accurate breakthrough time estimation.

The breakthrough time is encoded as indicators 0 and 1 where before breakthrough the indicator is 0, while after breakthrough the indicator is 1. The following Figure 5.12 and Figure 5.11 shows the comparison results of including the indicator feature and not including the indicator feature. Notice that the performance improved when we used cascaded LSTM. And R^2 value improved from 0.578 to 0.966 (Table 5.1).

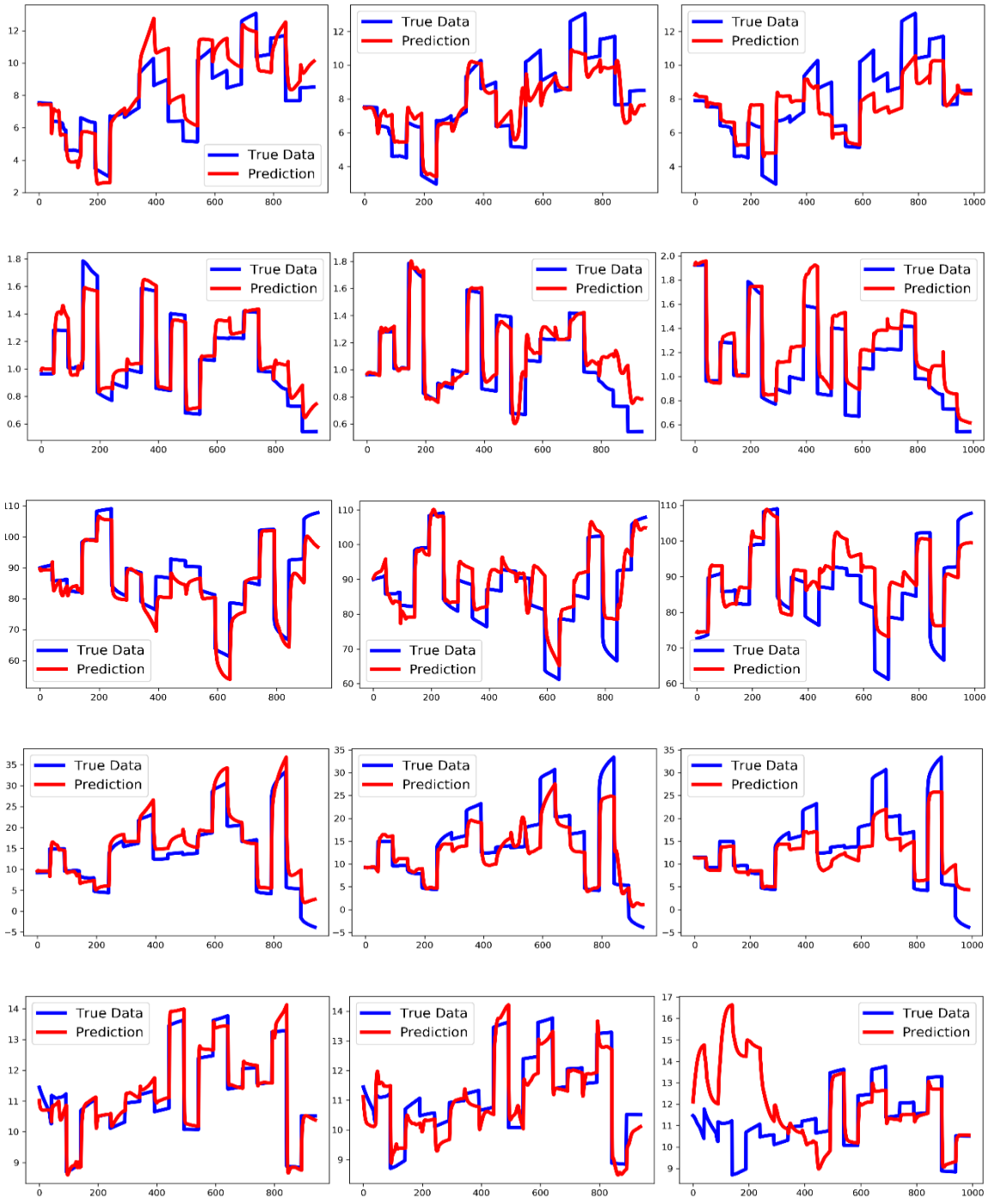


Figure 5.11 Performance evolution from left to right: standard LSTM, Standard LSTM with Bayesian optimization, cascaded LSTM with Bayesian optimization. The plots are production rate (bbl/day) vs. time (days).

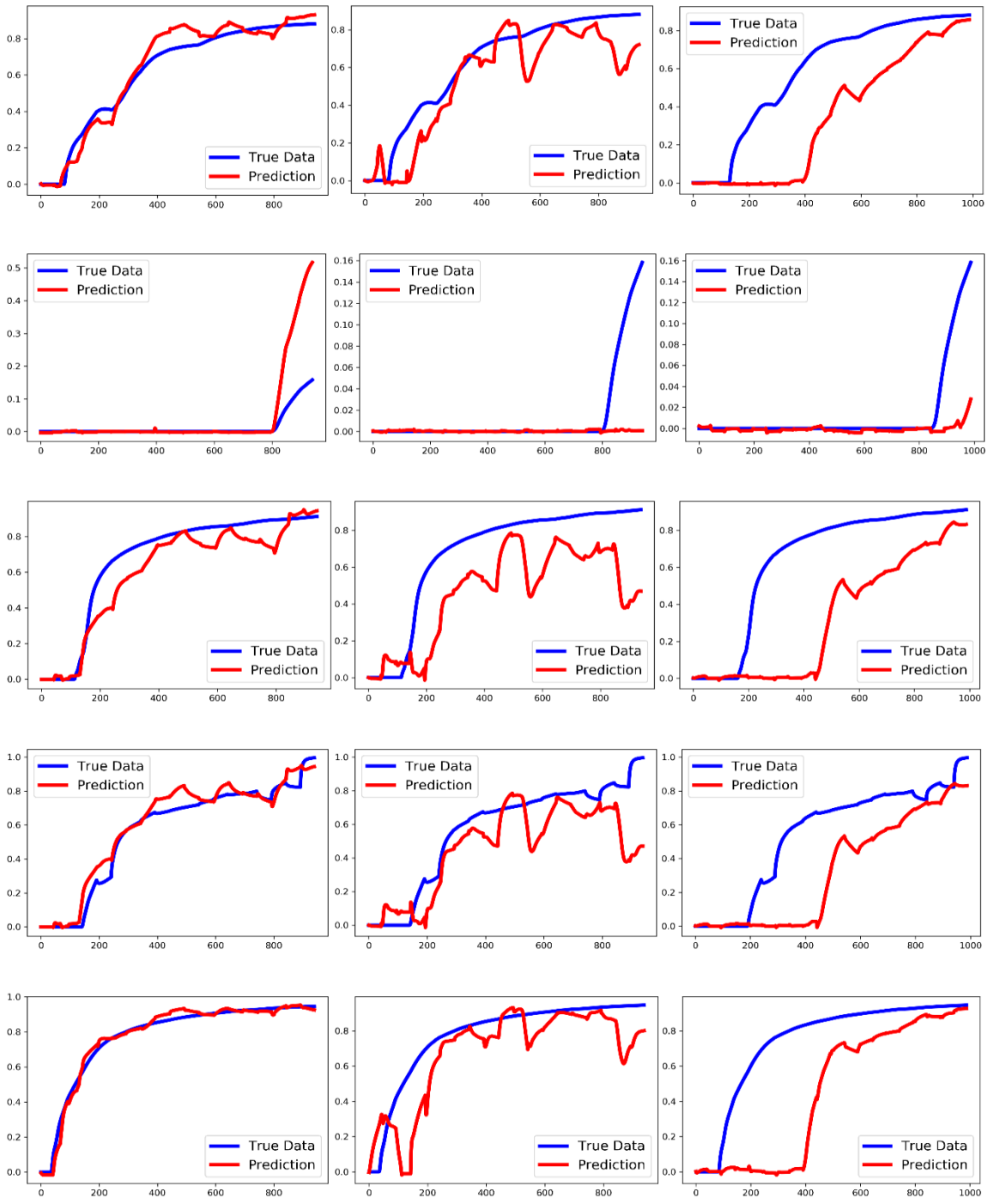
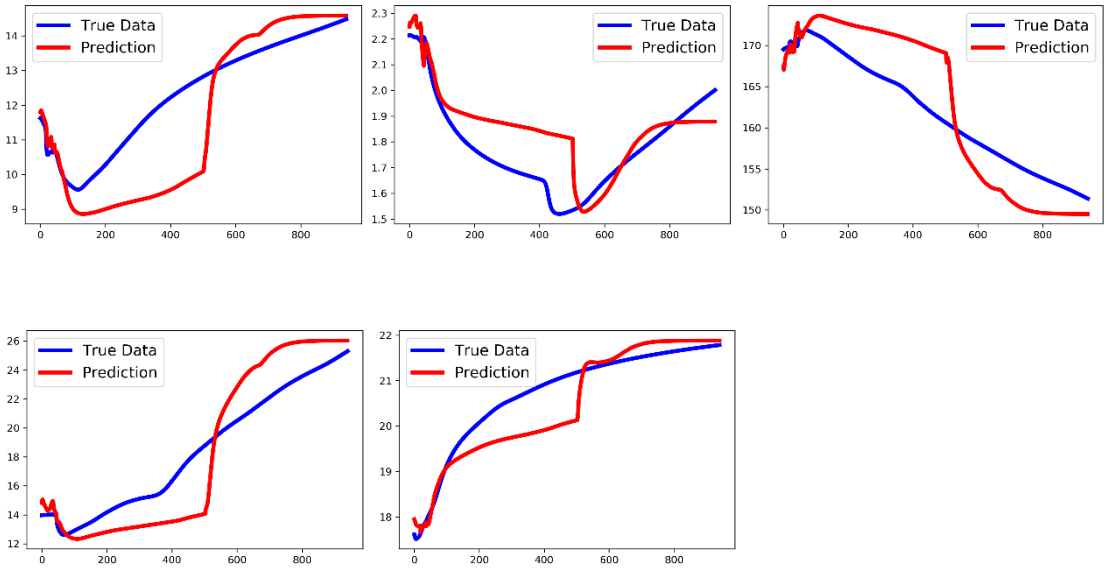


Figure 5.12 Performance evolution from left to right: standard LSTM, Standard LSTM with Bayesian optimization, cascaded LSTM with Bayesian optimization. The plots are for water cut vs. time (days).

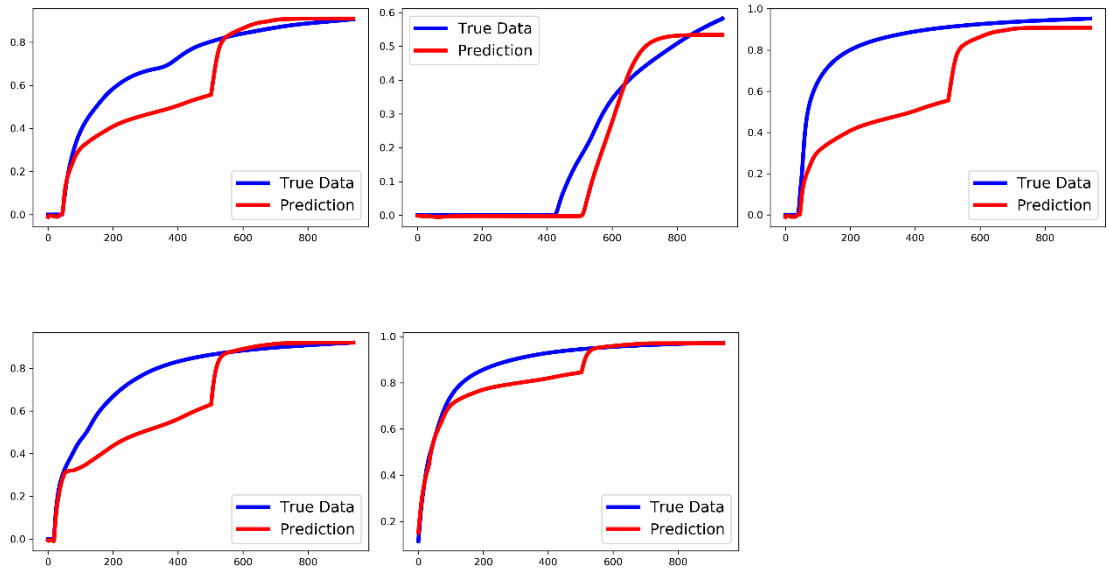
Table 5.1 Improvements over cascaded LSTM and Bayesian optimization

	Water Cut		Production Rate		Oil Production Rate	
	R^2	RMSE	R^2	RMSE	R^2	RMSE
Standard LSTM	0.294	0.317	0.987	3.753	0.578	16.117
Standard LSTM with BO	0.765	0.179	0.991	3.051	0.798	11.320
Cascaded LSTM with BO	0.959	0.075	0.993	2.695	0.958	5.154

To give a more comprehensive view of our model, we also test it on a simple case with constant injection rate (54 *bbl*/*day*) for each of the four injectors and constant producer bottom hole pressure (1950 *psia*, 1850 *psia*, 2200 *psia*, 1950 *psia*, 2200 *psia*) for each of the five production wells. The results show good match between predicted results and true observation (Figure 5.13). However, we do notice that all the output results has a unexpected jump around the time of the second well breakthrough, which might mean the second well's breakthrough time might has a unnecessary larger impact on the other wells in the model. With more training data on different injection/production scenarios might able to solve the problem.



(a) Production rate (bbl/day) vs. time (days)



(b) Water cut vs. time (days)

Figure 5.13 Performance test on a constant injection rate and constant production *BHP* scenario.

Table 5.2 Detailed results of the simplistic production scenarios.

	water cut		production rate		oil production rate	
	R^2	RMSE	R^2	RMSE	R^2	RMSE
Cascaded LSTM with BO	0.7613	0.1558	0.9984	2.3931	0.8608	19.1817

5.3.4. Apply Ensemble Kalman Filter

Finally, we apply the ensemble Kalman filter technique to our best model so far, the cascaded deep LSTM model. We generate 100 samples of weight samples from $w \sim N(w_{dense},\ \Sigma)$, where w_{dense} is $n \times 1$ vector of the weights from the last dense layer and $\Sigma = 0.1I_n$. The results of production rate prior and posterior is shown below (Figure 5.14). As seen in the figure, the 100 realization of total production rate and water cut (grey lines) generated using the 100 prior ensembles of weights covers the whole range of the true value (red line). However, the posterior results of total production rate and water cut fit too well with the true value, in other words, there is a risk of ensemble collapse associated with the model. In the future, we will further investigate the problem.

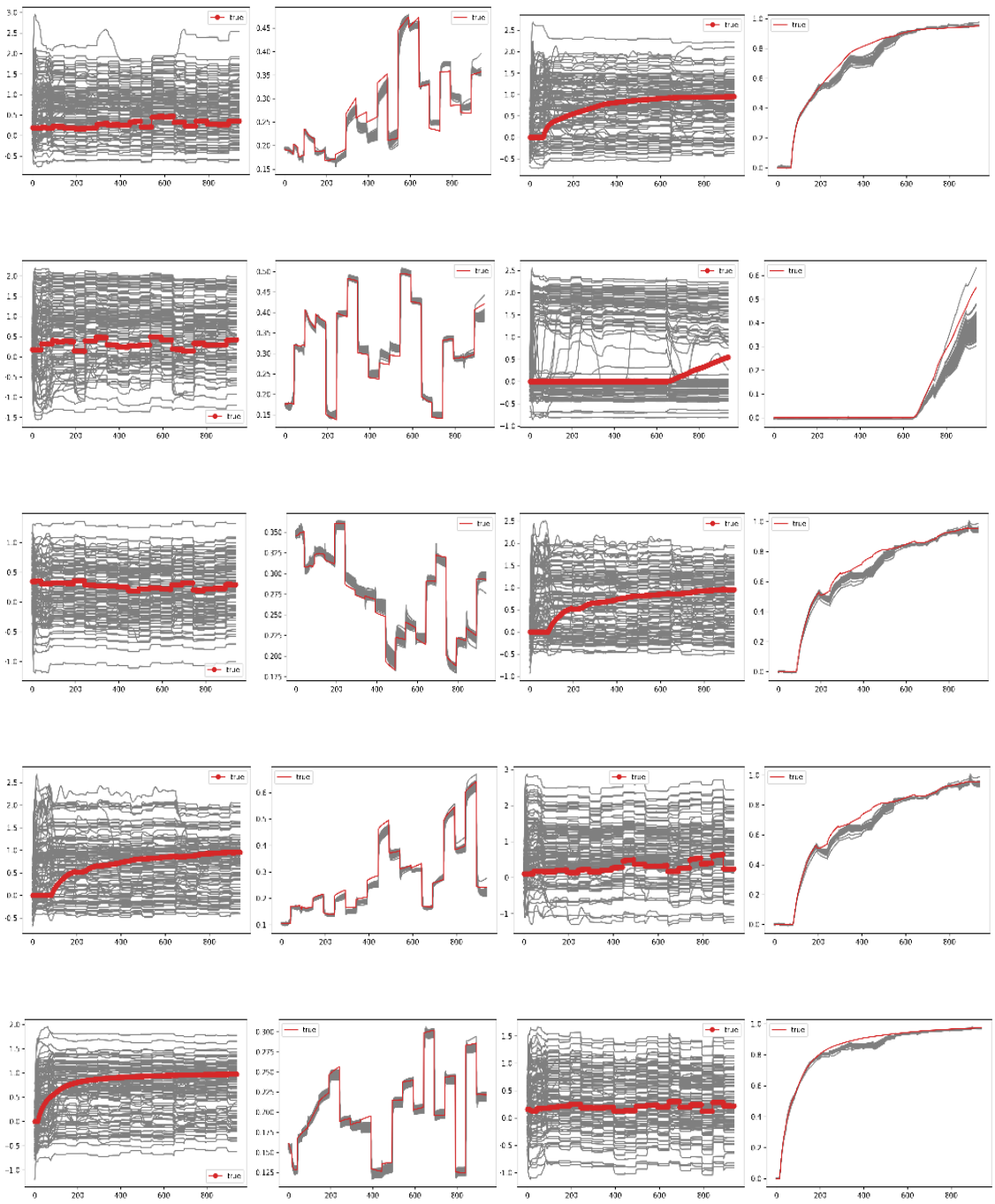


Figure 5.14 Training data with EnKF-LSTM. First column: Ensemble of prior for Production rate. Second column: Ensemble of posterior production rate. Third column: Ensemble of prior for water cut. Fourth column: Ensemble of posterior for water cut (All are normalized results).

Next, we evaluate the new model generated by EnKF-LSTM to a completely new data set. The result shows promising improvement when matching with true observation (Figure 5.15 and Table 5.3).

Figure 5.15 Test Results of EnKF-cascaded LSTM. First column: Production rate from best model in EnKF-cascaded LSTM (100 ensembles). Second column: Production rate prediction from cascaded LSTM model. Third column: water cut from best model in EnKF-cascaded LSTM (100 ensembles). Fourth column: water cut from Production rate prediction from cascaded LSTM model.

Table 5.3 Improvements over EnKF.

	water cut		production rate	
	R^2	RMSE	R^2	RMSE
cov=0.1I, No. ensembles=100	0.9622	0.0718	0.9961	2.019
before EnKF	0.9593	0.07446	0.9931	2.6948

5.4. Conclusion and Discussion

LSTM method provides a promising novel technique to predict future oil production. We see improvement from simple LSTM to Cascaded LSTM with the Bayesian hyperparameter tuning. Meanwhile it is noticed that Bayesian optimization is a reliable faster way to tune hyperparameters than simple grid search and random search algorithm. Finally, the Ensemble Kalman filter can help make the model prediction more robust. Coupled LSTM with EnKF can be useful when applied in real-time production optimization.

6. CONCLUSIONS

6.1. Conclusion

In this work, we have proposed to apply data-driven methods to accelerate traditional reservoir simulation considering model order reduction and proxy model developments. The reservoir simulator we are using include single phase fully implicit simulator and sequential mix formulation two phase simulator.

For the model order reduction part, we implemented the newly developed sparsity DMD algorithm and compared results with standard DMD and POD. From both single- and two-phase simulator, we notice that the POD method when considering limited number of modes, will either have jumps in the relative error or will fail to explain the correct physics process. On the other hand, sparsity DMD has more stable performance during the process of reducing modes.

For system identification with customized SINDy method, the essential idea is to make the problem of identify correct terms in ODE equivalent to seeking a sparsity structure from the basis library. The results show that our method outperforms the soft thresholding Lasso method when it comes to solving sparsity structure for single phase simulator ODE system.

Finally, the LSTM method is explored in this dissertation to further simplify the reservoir simulation to a multi-variate times series problem, with the fact that reservoir simulator is essentially a recursive process and dominated by control parameters such as injection rate and production bottom hole pressure. Results shown we have made valuable

improvement in accuracy from standard LSTM to cascaded LSTM with Bayesian optimization for network tuning. The extension to incorporates EnKF in our model also shows promising results and provides the possibility to use our mode for data assimilation.

We have built the foundation to the above-mentioned technique for its application on reservoir simulation. There is potential work to be done to improve the performance or extend the technique to larger and more complex problem. For the sparsity-promoting DMD problem, the algorithm is intended to deal with constant control problem and extra effort is required for complex control problem. To improve the algorithm, efficient sample selection procedure is to be considered and ways to tailor the algorithm for other simulators (e.g. fully implicit simulator) are to be explored. In the approach for discovering the governing equations for two phase flow, we only showed its capability to discover the saturation equation with constant control, extended the approach to reveal pressure and velocity equation in the context of complex control reservoir could be another area for this topic. For the last part of the work, the end to end proxy model for production prediction, its incorporation with EnKF is still at initial stage. To improve the performance, the network needs to be explored in depth to find the proper weights to update with corresponding observation.

6.2. Future Work

The future work can be categorized into three parts. First, according to what we achieved on the sparsity-promoting DMD method, we will attempt to apply the algorithm to fully implicit reservoir simulator and combine with DMDc algorithm to deal with

complex control problem. Another possible direction is to investigate efficient snapshot selection process to achieve more accurate algorithm for mode reduction.

Second, for proxy model development with discovering ODE technique, we will explore the possibility to Incorporate the complex control problem to the customized SINDy method and research on more physical constraints that could be utilized to make the model more robust for large dynamic system.

Finally, to improve the accuracy of EnKF enhanced LSTM network, we will further explore the structure of deep neural network, for example, updating partial network weights based on the connection of different groups of weights to different types of output results.

REFERENCES

1. Tang, J., Ehlig-Economides, C., Li, J., Guo, X., et al., *Investigation of Multiple Hydraulic Fracture Propagation for Low-Pressure Tight Sandstone Reservoirs in Northern Ordos Basin of Changqing Oilfield, China*, in *SPE International Hydraulic Fracturing Technology Conference and Exhibition*. 2018, Society of Petroleum Engineers: Muscat, Oman. p. 19.
2. Zhang, D., Li, L., and Tchelepi, H.A., *Stochastic Formulation for Uncertainty Analysis of Two-Phase Flow in Heterogeneous Reservoirs.* SPE Journal, 2000. **5**(01): p. 60-70 DOI: 10.2118/59802-PA.
3. Antoulas, A.C., *Approximation of large-scale dynamical systems*. Vol. 6. 2005: Siam.
4. Liu, Z. and Forouzanfar, F., *Ensemble clustering for efficient robust optimization of naturally fractured reservoirs.* Computational Geosciences, 2018. **22**(1): p. 283-296 DOI: 10.1007/s10596-017-9689-1.
5. Niu, G., Sun, J., Parsegov, S., and Schechter, D., *Integration of Core Analysis, Pumping Schedule and Microseismicity to Reduce Uncertainties of Production Performance of Complex Fracture Networks for Multi-Stage Hydraulically Fractured Reservoirs*, in *SPE Eastern Regional Meeting*. 2017, Society of Petroleum Engineers: Lexington, Kentucky, USA. p. 20.

6. Liu, Z., Forouzanfar, F., and Zhao, Y., *Comparison of SQP and AL algorithms for deterministic constrained production optimization of hydrocarbon reservoirs.* Journal of Petroleum Science and Engineering, 2018. **171**: p. 542-557.

7. Liu, Z. and Reynolds, A.C. *An SQP-Filter Algorithm with an Improved Stochastic Gradient for Robust Life-Cycle Optimization Problems with Nonlinear Constraints*. in *SPE Reservoir Simulation Conference*. 2019. Society of Petroleum Engineers DOI: 10.2118/193925-MS.

8. Hou, T. and Efendiev, Y., *Multiscale Finite Element Methods: Theory and Applications*. 2009.

9. Gao, S., Killough, J., He, J., Fadlelmula, M., et al., *A Different Approach for the Simulation of Fractured Vuggy Carbonate Reservoir with an Application to Upscaling*. 2017.

10. King, M.J., MacDonald, D.G., Todd, S.P., and Leung, H., *Application of Novel Upscaling Approaches to the Magnus and Andrew Reservoirs*, in *European Petroleum Conference*. 1998, Society of Petroleum Engineers: The Hague, Netherlands. p. 16.

11. Ghommem, M., Gildin, E., and Ghasemi, M., *Complexity Reduction of Multiphase Flows in Heterogeneous Porous Media.* SPE Journal, 2016. **21**(01): p. 144-151 DOI: 10.2118/167295-PA.

12. Ghasemi, M. and Gildin, E., *Model order reduction in porous media flow simulation using quadratic bilinear formulation.* Computational Geosciences, 2016. **20**(3): p. 723-735.

13. Chaturantabut, S. and Sorensen, D., *Nonlinear Model Reduction via Discrete Empirical Interpolation.* SIAM Journal on Scientific Computing, 2010. **32**(5): p. 2737-2764 DOI: 10.1137/090766498.

14. Schmid, P.J., *Dynamic mode decomposition of numerical and experimental data.* Journal of Fluid Mechanics, 2010. **656**: p. 5-28 DOI: 10.1017/S0022112010001217.

15. Rowley, C.W., Mezic, I., Bagheri, S., Schlatter, P., et al., *Spectral analysis of nonlinear flows.* Journal of Fluid Mechanics, 2009. **641**: p. 115-127 DOI: 10.1017/S0022112009992059.

16. Chen, K.K., Tu, J.H., and Rowley, C.W., *Variants of Dynamic Mode Decomposition: Boundary Condition, Koopman, and Fourier Analyses.* Journal of Nonlinear Science, 2012. **22**(6): p. 887-915 DOI: 10.1007/s00332-012-9130-9.

17. Lusch, B., Kutz, J.N., and Brunton, S.L., *Deep learning for universal linear embeddings of nonlinear dynamics.* Nature Communications, 2018. **9**(1): p. 4950 DOI: 10.1038/s41467-018-07210-0.

18. Takeishi, N., Kawahara, Y., and Yairi, T., *Learning Koopman invariant subspaces for dynamic mode decomposition*, in *Proceedings of the 31st International Conference on Neural Information Processing Systems*. 2017, Curran Associates Inc.: Long Beach, California, USA. p. 1130-1140.

19. Williams, M.O., Kevrekidis, I.G., and Rowley, C.W., *A Data–Driven Approximation of the Koopman Operator: Extending Dynamic Mode*

Decomposition. Journal of Nonlinear Science, 2015. **25**(6): p. 1307-1346 DOI: 10.1007/s00332-015-9258-5.

20. Proctor, J.L., Brunton, S.L., and Kutz, J.N., *Dynamic Mode Decomposition with Control.* SIAM Journal on Applied Dynamical Systems, 2016. **15**(1): p. 142-161 DOI: 10.1137/15m1013857.

21. Kunisch, K. and Volkwein, S., *Galerkin Proper Orthogonal Decomposition Methods for a General Equation in Fluid Dynamics.* SIAM Journal on Numerical Analysis, 2002. **40**(2): p. 492-515 DOI: 10.1137/S0036142900382612.

22. Goulart, P.J., Wynn, A., and Pearson, D. *Optimal mode decomposition for high dimensional systems*. in *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*. 2012. DOI: 10.1109/CDC.2012.6426995.

23. Tissot, G., Cordier, L., Benard, N., and Noack, B.R., *Model reduction using Dynamic Mode Decomposition.* Comptes Rendus Mécanique, 2014. **342**(6): p. 410-416 DOI: 10.1016/j.crme.2013.12.011.

24. Jovanović, M.R., Schmid, P.J., and Nichols, J.W., *Sparsity-promoting dynamic mode decomposition.* Physics of Fluids, 2014. **26**(2): p. 024103 DOI: 10.1063/1.4863670.

25. Chen, S., Donoho, D., and Saunders, M., *Atomic Decomposition by Basis Pursuit.* SIAM Journal on Scientific Computing, 1998. **20**(1): p. 33-61 DOI: 10.1137/S1064827596304010.

26. Claerbout, J.F. and Muir, F., *Robust Modeling With Erratic Data.* Geophysics, 1973. **38**(5): p. 826-844 DOI: 10.1190/1.1440378.

27. Boyd, S. and Vandenberghe, L., *Convex Optimization*. 2004: Cambridge University Press.

28. Brunton, S.L., Proctor, J.L., and Kutz, J.N., *Discovering governing equations from data by sparse identification of nonlinear dynamical systems.* Proceedings of the National Academy of Sciences, 2016. **113**(15): p. 3932 DOI: 10.1073/pnas.1517384113.

29. Tang, J., Wu, K., Li, Y., Hu, X., et al., *Numerical Investigation of the Interactions between Hydraulic Fracture and Bedding Planes with Non-orthogonal Approach Angle*. Vol. 200. 2018.

30. Tang, J., Wu, K., Zuo, L., Xiao, L., Sun, S., Ehlig-Economides, *Investigation of Rupture and Slip Mechanisms of Hydraulic Fracture in Multiple-layered Formations.* SPE Journal, 2019.

31. Schmidt, M. and Lipson, H., *Distilling Free-Form Natural Laws from Experimental Data.* Science, 2009. **324**(5923): p. 81 DOI: 10.1126/science.1165893.

32. Koza, J.R., *Genetic programming as a means for programming computers by natural selection.* Statistics and Computing, 1994. **4**(2): p. 87-112 DOI: 10.1007/BF00175355.

33. Kutz, J.N., Brunton, S.L., Brunton, B.W., and Proctor, J.L., *Dynamic mode decomposition: data-driven modeling of complex systems*. 2016: SIAM.

34. Candes, E., Romberg, J., and Tao, T. *Stable Signal Recovery from Incomplete and Inaccurate Measurements*. arXiv Mathematics e-prints, 2005.

35. Trevor, H., Robert, T., and JH, F., *The elements of statistical learning: data mining, inference, and prediction*. 2009, New York, NY: Springer.

36. Boyd, S., Parikh, N., Chu, E., Peleato, B., et al., *Distributed optimization and statistical learning via the alternating direction method of multipliers.* Foundations and Trends® in Machine learning, 2011. **3**(1): p. 1-122.

37. Rudy, S.H., Brunton, S.L., Proctor, J.L., and Kutz, J.N., *Data-driven discovery of partial differential equations.* Science Advances, 2017. **3**(4): p. e1602614 DOI: 10.1126/sciadv.1602614.

38. Zhou, P., Pan, Y., Sang, H., and Lee, W.J., *Criteria for Proper Production Decline Models and Algorithm for Decline Curve Parameter Inference*, in *Unconventional Resources Technology Conference, Houston, Texas, 23-25 July 2018*. 2018, Society of Exploration Geophysicists, American Association of Petroleum Geologists, Society of Petroleum Engineers. p. 3535-3551.

39. Liu, Y. and Horne, R.N., *Interpreting Pressure and Flow-Rate Data From Permanent Downhole Gauges by Use of Data-Mining Approaches.* SPE Journal, 2012. **18**(01): p. 69-82 DOI: 10.2118/147298-PA.

40. Bao, A., Hazlett, R.D., and Babu, D.K., *A Discrete, Arbitrarily Oriented 3D Plane-Source Analytical Solution to the Diffusivity Equation for Modeling Reservoir Fluid Flow.* SPE Journal, 2017. **22**(05): p. 1609-1623 DOI: 10.2118/185180-PA.

41. Parsegov, S.G., Niu, G., Schechter, D.S., and Laprea-Bigott, M., *Benefits of Engineering Fracture Design. Lessons Learned from Underperformers in the Midland Basin*, in *SPE Hydraulic Fracturing Technology Conference and*

Exhibition. 2018, Society of Petroleum Engineers: The Woodlands, Texas, USA. p. 21.

42. Zhang, F., Saputra, I.W.R., Niu, G., Adel, I.A., et al., *Upscaling Laboratory Result of Surfactant-Assisted Spontaneous Imbibition to the Field Scale through Scaling Group Analysis, Numerical Simulation, and Discrete Fracture Network Model*, in *SPE Improved Oil Recovery Conference*. 2018, Society of Petroleum Engineers: Tulsa, Oklahoma, USA. p. 24.

43. Tang, H., Yan, B., Chai, Z., Zuo, L., et al., *Analyzing the Well-Interference Phenomenon in the Eagle Ford Shale/Austin Chalk Production System With a Comprehensive Compositional Reservoir Model.* SPE Reservoir Evaluation & Engineering, 2018. **Preprint**(Preprint): p. 15 DOI: 10.2118/191381-PA.

44. Tang, H., Hasan, A.R., and Killough, J., *Development and Application of a Fully Implicitly Coupled Wellbore/Reservoir Simulator To Characterize the Transient Liquid Loading in Horizontal Gas Wells.* SPE Journal, 2018. **23**(05): p. 1615-1629 DOI: 10.2118/187354-PA.

45. Tang, J., Wu, K., Zeng, B., Huang, H., et al., *Investigate effects of weak bedding interfaces on fracture geometry in unconventional reservoirs.* Journal of Petroleum Science and Engineering, 2018. **165**: p. 992-1009 DOI: 10.1016/j.petrol.2017.11.037.

46. Tang, J., Ehlig-Economides, C., Fan, B., Cai, B., et al., *A microseismic-based fracture properties characterization and visualization model for the selection of*

infill wells in shale reservoirs. Journal of Natural Gas Science and Engineering, 2019. **67**: p. 147-159 DOI: 10.1016/j.jngse.2019.04.014.

47. Xiong, H., Huang, S., Devegowda, D., Liu, H., et al., *Simulation and Modeling of Pressure Difference on Steam Chamber and Production Performance*, in *SPE Western Regional Meeting*. 2018, Society of Petroleum Engineers: Garden Grove, California, USA. p. 20.

48. Bao, A., Gildin, E., and Zalavadia, H. *Development Of Proxy Models For Reservoir Simulation By Sparsity Promoting Methods And Machine Learning Techniques*. in *ECMOR XVI-16th European Conference on the Mathematics of Oil Recovery*. 2018. Barcelona, Span DOI: 10.3997/2214-4609.201802180

49. Sun, J., Niu, G., and Schechter, D., *Numerical Simulation of Stochastically-Generated Complex Fracture Networks by Utilizing Core and Microseismic Data for Hydraulically Fractured Horizontal Wells in Unconventional Reservoirs– A Field Case Study*, in *SPE Eastern Regional Meeting*. 2016, Society of Petroleum Engineers: Canton, Ohio, USA. p. 15.

50. Sagheer, A. and Kotb, M., *Time series forecasting of petroleum production using deep LSTM recurrent networks.* Neurocomputing, 2019. **323**: p. 203-213.

51. Ho, S.L. and Xie, M., *The use of ARIMA models for reliability forecasting and analysis.* Computers & Industrial Engineering, 1998. **35**(1): p. 213-216 DOI: 10.1016/S0360-8352(98)00066-7.

52. Engle, R.F., *Autoregressive Conditional Heteroscedasticity with Estimates of the Variance of United Kingdom Inflation.* Econometrica, 1982. **50**(4): p. 987-1007 DOI: 10.2307/1912773.

53. Tong, H., *Non-linear Time Series: A Dynamical System Approach*. 1993: Clarendon Press.

54. Ma, X. and Liu, Z., *Predicting the oil production using the novel multivariate nonlinear model based on Arps decline model and kernel method.* Neural Computing and Applications, 2018. **29**(2): p. 579-591 DOI: 10.1007/s00521-016-2721-x.

55. Tian, C. and Horne, R.N., *Recurrent Neural Networks for Permanent Downhole Gauge Data Analysis*, in *SPE Annual Technical Conference and Exhibition*. 2017, Society of Petroleum Engineers: San Antonio, Texas, USA. p. 12.

56. Hochreiter, S. and Schmidhuber, J., *Long Short-Term Memory.* Neural Comput., 1997. **9**(8): p. 1735-1780 DOI: 10.1162/neco.1997.9.8.1735.

57. Pascanu, R., Mikolov, T., and Bengio, Y. *On the difficulty of training Recurrent Neural Networks*. arXiv e-prints, 2012.

58. Canchumuni, S.A., Emerick, A.A., and Pacheco, M.A., *Integration of Ensemble Data Assimilation and Deep Learning for History Matching Facies Models*, in *OTC Brasil*. 2017, Offshore Technology Conference: Rio de Janeiro, Brazil. p. 13.

59. Kemajou, V.N., Bao, A., and Germain, O., *Wellbore Schematics to Structured Data Using Artificial Intelligence Tools*, in *Offshore Technology Conference*. 2019, Offshore Technology Conference: Houston, Texas. p. 18.

60. Madasu, S. and Rangarajan, K.P., *Deep Recurrent Neural Network DRNN Model for Real-Time Multistage Pumping Data*, in *OTC Arctic Technology Conference*. 2018, Offshore Technology Conference: Houston, Texas, USA. p. 9.

61. Zhang, D., Chen, Y., and Meng, J., *Synthetic well logs generation via Recurrent Neural Networks.* Petroleum Exploration and Development, 2018. **45**(4): p. 629-639 DOI: 10.1016/S1876-3804(18)30068-5.

62. Loh, K., Shoeibi Omrani, P., and van der Linden, R., *Deep Learning and Data Assimilation for Real-Time Production Prediction in Natural Gas Wells*. 2018.

63. Mirikitani, D.T. and Nikolaev, N. *Dynamic Modeling with Ensemble Kalman Filter Trained Recurrent Neural Networks*. in *2008 Seventh International Conference on Machine Learning and Applications*. 2008. DOI: 10.1109/ICMLA.2008.79.

64. Zhao, Y. and Forouzanfar, F., *A Simultaneous Bayesian Estimation of Channelized Facies and Reservoir Properties under Prior Uncertainty*, in *SPE Europec featured at 79th EAGE Conference and Exhibition*. 2017, Society of Petroleum Engineers: Paris, France. p. 19.

65. Zhao, Y., Forouzanfar, F., and Reynolds, A.C., *History matching of multi-facies channelized reservoirs using ES-MDA with common basis DCT*. Computational Geosciences, 2017. **21**(5): p. 1343-1364 DOI: 10.1007/s10596-016-9604-1.

66. Dhara, A., Trainor-Guitton, W., and Tura, A., *Machine-learning-based methods for estimation and stochastic simulation*, in *SEG Technical Program Expanded Abstracts 2018*. 2018, Society of Exploration Geophysicists. p. 2261-2265.

67. Aziz, K., *Petroleum Reservoir Simulation*. Applied Science Publishers. Vol. 476. 1979.

68. Chen, Z., Huan, G., and Ma, Y., *Computational Methods for Multiphase Flows in Porous Media*. 2006: Society for Industrial and Applied Mathematics.

69. Aziz, K. and Settari, A., *Petroleum reservoir simulation*. 1986, London: Elsevier Applied Science.

70. W Peaceman, D., *Interpretation of well-block pressures in numerical reservoir simulation with nonsquare grid blocks and anisotropic permeability*. Vol. 23. 1983. 531-543.

71. Yang, Y., Ghasemi, M., Gildin, E., Efendiev, Y., et al., *Fast Multiscale Reservoir Simulations With POD-DEIM Model Reduction.* SPE Journal, 2016. **21**(06): p. 2141-2154 DOI: 10.2118/173271-PA.

72. Aarnes, J., Krogstad, S., and Lie, K., *A Hierarchical Multiscale Method for Two-Phase Flow Based upon Mixed Finite Elements and Nonuniform Coarse Grids.* Multiscale Modeling & Simulation, 2006. **5**(2): p. 337-363 DOI: 10.1137/050634566.

73. Brezzi, F. and Fortin, M., *Mixed and hybrid finite element methods*. 1991: Springer-Verlag. 350.

74. Sirovich, L., *TURBULENCE AND THE DYNAMICS OF COHERENT STRUCTURES PART I: COHERENT STRUCTURES.* Quarterly of Applied Mathematics, 1987. **45**(3): p. 561-571.

75. Hinze, M. and Volkwein, S. *Proper Orthogonal Decomposition Surrogate Models for Nonlinear Dynamical Systems: Error Estimates and Suboptimal Control*. in *Dimension Reduction of Large-Scale Systems*. 2005. Berlin, Heidelberg: Springer Berlin Heidelberg.

76. Volkwein, S., *Model reduction using proper orthogonal decomposition.* Lecture Notes, Institute of Mathematics and Scientific Computing, University of Graz. see http://www.uni-graz.at/imawww/volkwein/POD.pdf, 2011. **1025**.

77. Tu, J.H., Rowley, C.W., Luchtenburg, D.M., Brunton, S.L., et al. *On Dynamic Mode Decomposition: Theory and Applications*. arXiv e-prints, 2013.

78. Bao, A. and Gildin, E., *Data-Driven Model Reduction Based on Sparsity-Promoting Methods for Multiphase Flow in Porous Media*, in *SPE Latin America and Caribbean Petroleum Engineering Conference*. 2017, Society of Petroleum Engineers: Buenos Aires, Argentina. p. 16.

79. Christie, M.A. and Blunt, M.J., *Tenth SPE Comparative Solution Project: A Comparison of Upscaling Techniques.* SPE Reservoir Evaluation & Engineering, 2001. **4**(04): p. 308-317 DOI: 10.2118/72469-PA.

80. Ghasemi, M., Ibrahim, A., and Gildin, E. *Reduced order modeling in reservoir simulation using the bilinear approximation techniques*. in *SPE Latin America and Caribbean Petroleum Engineering Conference*. 2014. Society of Petroleum Engineers.

81. Akaike, H., *Information theory and an extension of the maximum likelihood principle*, in *Selected papers of hirotugu akaike*. 1998, Springer. p. 199-213.

82. Schwarz, G., *Estimating the Dimension of a Model.* The Annals of Statistics, 1978. **6**(2): p. 461-464.

83. Mangan, N.M., Brunton, S.L., Proctor, J.L., and Kutz, J.N., *Inferring Biological Networks by Sparse Identification of Nonlinear Dynamics.* IEEE Transactions on Molecular, Biological and Multi-Scale Communications, 2016. **2**(1): p. 52-63 DOI: 10.1109/TMBMC.2016.2633265.

84. Bishop, C.M., *Pattern Recognition and Machine Learning (Information Science and Statistics).* 2006: Springer-Verlag.

85. HASTINGS, W.K., *Monte Carlo sampling methods using Markov chains and their applications.* Biometrika, 1970. **57**(1): p. 97-109.

86. Chen, B., Harp, D.R., Lin, Y., Keating, E.H., et al., *Geologic CO2 sequestration monitoring design: A machine learning and uncertainty quantification based approach.* Applied Energy, 2018. **225**: p. 332-345 DOI: 10.1016/j.apenergy.2018.05.044.

87. Chen, B., He, J., Wen, X.-H., Chen, W., et al., *Uncertainty quantification and value of information assessment using proxies and Markov chain Monte Carlo method for a pilot project.* Journal of Petroleum Science and Engineering, 2017. **157**: p. 328-339 DOI: 10.1016/j.petrol.2017.07.039.

88. MATLAB, *Statistics Toolbox Release* 2018b: The MathWorks Inc., Natick, MA, USA.

89. Loiseau, J.-C. and Brunton, S.L., *Constrained sparse Galerkin regression.* Journal of Fluid Mechanics, 2018. **838**: p. 42-67 DOI: 10.1017/jfm.2017.823.

90. Sorek, N., Gildin, E., Boukouvala, F., Beykal, B., et al., *Dimensionality reduction for production optimization using polynomial approximations.* Computational Geosciences, 2017. **21**(2): p. 247-266 DOI: 10.1007/s10596-016-9610-3.

91. Loiseau, J.-C. and Brunton, S.L. *Constrained Sparse Galerkin Regression*. arXiv e-prints, 2016.

92. Ng, A. *Sequence Models*. 2018; Available from: https://www.coursera.org/learn/nlp-sequence-models/home/welcome.

93. Pedregosa, F., Ga, #235, Varoquaux, l., et al., *Scikit-learn: Machine Learning in Python.* J. Mach. Learn. Res., 2011. **12**: p. 2825-2830.

94. Frazier, P.I. *A Tutorial on Bayesian Optimization*. arXiv e-prints, 2018.

95. Oliver, D.S., Reynolds, A.C., and Liu, N., *Inverse Theory for Petroleum Reservoir Characterization and History Matching*. 2008, Cambridge: Cambridge University Press.

96. Loh, K., Shoeibi Omrani, P., and van der Linden, R. *Deep Learning and Data Assimilation for Real-Time Production Prediction in Natural Gas Wells*. arXiv e-prints, 2018.

97. Chen, H., Onishi, T., Olalotiti-Lawal, F., and Datta-Gupta, A., *Streamline Tracing and Applications in Naturally Fractured Reservoirs Using Embedded Discrete Fracture Models*, in *SPE Annual Technical Conference and Exhibition*. 2018, Society of Petroleum Engineers: Dallas, Texas, USA. p. 22.

98. Cheng, H., Kharghoria, A., Zhong, H., and Datta-Gupta, A., *Fast History Matching of Finite-Difference Models Using Streamline-Derived Sensitivities*, in

SPE/DOE Symposium on Improved Oil Recovery. 2004, Society of Petroleum Engineers: Tulsa, Oklahoma. p. 14.

99. Datta-Gupta, A. and King, M.J., *Streamline simulation: theory and practice*. Vol. 11. 2007: Society of Petroleum Engineers Richardson.