An Adjoint-Based Optimization Method Using the Solution of Gray-Box Conservation Laws

by

Han Chen

Submitted to the Department of Aeronautics and Astronautics
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy in Aerospace Computational Engineering
at the
MASSACHUSETTS INSTITUTE OF TECHNOLOGY
Feburary 2017

Author ..... Signature redacted .....
Department of Aeronautics and Astronautics
Dec 4, 2016

Certified by ..... Signature redacted .....
Qiqi Wang
Associate Professor of Aeronautics and Astronautics
Thesis Supervisor

Certified by ..... Signature redacted .....
Karen Willcox
Professor of Aeronautics and Astronautics
Committee Member

Certified by ..... Signature redacted .....
Youssef Marzouk
Associate Professor of Aeronautics and Astronautics
Committee Member

Accepted by ..... Signature redacted .....
Youssef Marzouk
Associate Professor of Aeronautics and Astronautics
Chair, Graduate Program Committee

An Adjoint-Based Optimization Method Using the Solution of Gray-Box Conservation Laws

by

Han Chen

Submitted to the Department of Aeronautics and Astronautics
on Dec 4, 2016, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in Aerospace Computational Engineering

Abstract

Many design applications can be formulated as an optimization constrained by conservation laws. Such optimization can be efficiently solved by the adjoint method, which computes the gradient of the objective to the design variables. Traditionally, the adjoint method has not been able to be implemented in "gray-box" conservation law simulations. In gray-box simulations, the analytical and numerical form of the conservation law is unknown, but the full solution of relevant flow quantities is available. Optimization constrained by gray-box simulations can be challenging for high-dimensional design because the adjoint method is not directly applicable.

My thesis considers the gray-box models whose flux functions contain unknown algebraic dependence on the state variables. I develop a twin-model method that estimates the adjoint gradient from the gray-box space-time solution. My method utilizes the gray-box space-time solution in order to infer the unknown components of the flux. The solution is used to train a parameterized, adjoint-enabled conservation law simulator such that a metric of solution mismatch is minimized. After the training, the twin model can estimate the gradient of the objective function by the adjoint method, at a cost independent of the dimensionality of the gradient. Also, an adaptive basis construction procedure is presented for the training to fully exploit the information contained in the gray-box solution. The availability of the estimated gradient enables more efficient optimization. My thesis considers a Bayesian optimization framework, in which the objective, the true gradient, and the error in the estimated gradient are modeled by Gaussian processes. Building upon previous research, a twin-model-enhanced Bayesian optimization algorithm is developed. I show that the algorithm can find the optimum of the objective function regardless of the gradient accuracy if the true hyperparameters of the Gaussian models are given.

The twin-model method and the twin-model-enhanced optimization are demonstrated in several gray-box models: a Buckley-Leverett equation whose flux function is unknown, a steady-state Navier-Stokes equation whose state equation is unknown, and a porous

media flow equation governing a petroleum reservoir whose componentwise mobility factors are unknown. In these examples, the twin model is shown to accurately estimate the gradients. Besides, the twin-model-enhanced Bayesian optimization can achieve near-optimality within fewer iterations than without using the twin model. Finally, I explore the applicability of the twin-model method in an example with 1000-dimensional control by using a gradient descent approach. The last example implies that the twin model may be adopted by other optimization frameworks to improve convergence, which indicates a direction of future research.

Thesis Supervisor: Qiqi Wang
Title: Associate Professor of Aeronautics and Astronautics

Committee Member: Karen Willcox
Title: Professor of Aeronautics and Astronautics

Committee Member: Youssef Marzouk
Title: Associate Professor of Aeronautics and Astronautics

Acknowledgments

I must firstly thank Professor Qiqi Wang, my academic advisor. He shows me to the door of applied mathematics. I can not accomplish my PhD without his support. I also thank Professor Karen Willcox. Her kindness helps me a lot during the hardest time of my PhD. Her insist on the mathematical rigor profoundly affects my thesis, my view of academic research, and my style of thinking. In addition, I thank Professor Youssef Marzouk. I get insightful suggestions every time I talk with him. Indeed, the 3rd chapter of my thesis is inspired by a meeting with him. Besides, I thank Hector Klie. The topic of my thesis was motivated by my internship in ConocoPhillips in 2011, when I realized my collegues could wait for days for gradient-free optimizations just because the code didn't have adjoint. The time I spent in ConocoPhillips working with Hector was one of my happiest time in the US. I also thank Professor David Darmofal and Professor Paul Constantine for being my thesis readers.

Finally, I am sincerely grateful to have the constant support from my wife Ran Huo and my mother Hailing Xiong. Last but not least, I became a father here. It's a wonderful thing to have Evin Chen, nicknamed Abu, adding an extra dimension to the optimization of my life.

Contents

List of Figures

List of Tables

Nomenclature

- $t \in [0, T]$: the time,
- $\{t_i\}_{i=1}^{M}$: the time steps,
- $x \in \Omega$: the space,
- $\{x_j\}_{j=1}^{N}$: the spatial grid points,
- u: the space-time solution of gray-box conservation law,
- $\tilde{u}$: the space-time solution of twin-model conservation law,
- $\boldsymbol{u}$: the discretized space-time solution of gray-box simulator,
- $\tilde{\boldsymbol{u}}$: the discretized space-time solution of twin-model simulator,
- k: the number of equations of the conservation law; or the number of folds in cross validation.
- D: a known differential operator,
- F: the unknown function of the gray-box model,
- $\tilde{F}$: the inferred F,
- q: the source term,
- c: the control variables,
- $\underline{c}_n = (c_1, \cdots, c_n)$: a sequence of n control variables,
- w: the quadrature weights in the numerical space-time integration,
- ξ: the objective function,
- $c_{\min}$, $c_{\max}$: bound constraints,

- $\xi_{\tilde{\nabla}}$: the estimated gradient of ξ with respect to c,
- d: the number of control variables,
- $\mathcal{C} \subset \mathbb{R}^d$: the control space,
- K, G: the covariance functions,
- a: the acquisition function,
- $\mathcal{M}_u$: the solution mismatch,
- $\mathcal{M}_\tau$: the integrated truncation error,
- $\overline{\mathcal{M}}$: the mean solution mismatch in cross validation,
- ϕ: a basis function for $\tilde{F}$,
- $j, \boldsymbol{j}$: the dilation parameters of the basis for uni- and multi-variate functions,
- $\eta, \boldsymbol{\eta}$: the translation parameters of the basis for uni- and multi-variate functions,
- $\mathcal{A}$: the basis dictionary,
- α: the coefficients for ϕ,
- T: twin model,
- τ: residual,
- $\boldsymbol{\tau}$: discretized residual.

Chapter 1

Background

1.1 Motivation

A conservation law states that a particular property of a physical system does not appear or vanish as the system evolves over time, such as the conservation of mass, momentum, and energy. Mathematically, a conservation law can be expressed locally as a continuity equation (1.1),

$$\frac{\partial u}{\partial t} + \nabla \cdot F = q\,, \tag{1.1}$$

where u is the conserved physical quantity, t is time, F is the flux that depends on u, and q is the source term that also depends on u. Many equations fundamental to the physical world, such as the Navier-Stokes equation, the Maxwell equation, and the porous medium transport equation, can be described by (1.1).

Optimization constrained by conservation laws is present in many engineering applications. For example, in gas turbines, the rotor blades can operate at a temperature close to 2000K [10]. To prevent material failure due to overheating, channels can be forged inside the rotor blades to circulate coolant air whose dynamics are governed by the Navier-Stokes equation [7]. The pressure used to drive the coolant flow is provided by the compressor, resulting in a penalty on the turbine's thermo-dynamic efficiency

[8]. Engineers are thereby interested in optimizing the coolant channel geometry in order to suppress the pressure loss. In this optimization problem, the control variables are the parameters that describe the channel geometry. The dimensionality of the optimization is the number of control variables, i.e., the control's degree of freedom. Another example is the field control of petroleum reservoir. In the petroleum reservoir, the fluid flow of various phases and chemical components is dictated by porous medium transport equations [4]. The flow can be passively and actively controlled by a variety of techniques [1], such as the wellbore pressure control, the polymer injection, and the steam heating [5]. The pressure, injection rate, and temperature can vary in each well and during every day over decades of continuous operations. The dimensionality of the optimization is the total number of these control variables. Driven by economic interests, petroleum producers are devoted to optimizing the controls for enhanced recovery and reduced cost.

Such optimization has been revolutionized by the numerical simulation and optimization algorithms. On the one hand, conservation law simulation can provide an evaluation of a candidate control that is cheaper, faster, and more scalable than conducting physical experiments. On the other hand, advanced optimization algorithms can guide the control toward the optimal with reduced number of simulation [40, 41, 42, 50, 54, 55, 56, 72]. However, optimization based on conservation law simulation can still be overwhelmingly costly. The cost is two-fold: First, each simulation at a given control may run for hours or days even on a high-end computer. Such expense in time is usually a result of using high-fidelity physical models, complex numerical schemes, and large-scale space-time discretization schemes. Second, optimization algorithms generally take many iterations of simulation on various controls. The number of iterations required to achieve near-optimality usually increases with the control variables' degree of freedom [60]. The two costs are multiplicative. The multiplicative effect compromises the impact of computational efforts among field engineers.

Fortunately, the cost due to iteration can be alleviated by adopting gradient-based optimization algorithms [60]. A gradient-based algorithm requires significantly fewer iterations than a derivative-free algorithm for problems with many control variables [19, 60, 41]. Gradient-based algorithms require the gradient of the optimization objective to the control variables, which is efficiently computable through the adjoint method [11]. The adjoint method propagates the gradient from the objective backward to the control variables through the path of time integration [11] or the chain of numerical operations [18]. To keep track of the back propagation, the simulator source code needs to be available. However, many industrial simulators do not have the adjoint capability because their codes were developed without the adjoint technique in mind. The implementation of the adjoint method in these legacy simulators can be a major undertaking. For example, *PSim*, a reservoir simulator developed and owned by *ConocoPhillips*, is a multi-million-line Fortran-77 code that traces its birth back to the 1980's. Implementing adjoint directly into the source code is not preferable because it can take a tremendous amount of brain hours. Besides, the source code and its physical models are only accessible and modifiable by the computational team inside the company. For the sake of gradient computation, *PSim* has been superseded by adjoint-enabled simulators, but it is hard to be replaced due to its legacy usage. The legacy nature of many industrial simulators hinders the prevalence of the adjoint method and gradient-based algorithms in many real-world problems with high-dimensional control.

Despite their legacy nature, most simulators for unsteady conservation laws can provide the discretized space-time solution of relevant flow quantities. For example, *PSim* provides the space-time solution of pressure, saturation, and concentration for multi-phase flow. Similarly, most steady-state simulators can provide the spatial solution. Thus, the discussion will focus on the unsteady case because a steady-state simulator can be viewed as a special case of unsteady-state simulators where the solutions remain the same over many time steps.

My thesis considers the conservation laws whose flux functions have an unknown algebraic dependence on the state variables. The adjoint gradient computation may be enabled by leveraging the space-time solution. The discretized space-time solution provides invaluable information about the conservation law hardwired in the simulator. For illustration, consider a code that simulates

$$\frac{\partial u}{\partial t} + \frac{\partial F(u)}{\partial x} = c, \quad x \in [0,1], \quad t \in [0,1] \tag{1.2}$$

with proper initial and boundary conditions and F being differentiable. c indicates the control that acts as a source for u. If the expression of $F(u)$ in the simulator is not accessible by the user, adjoint can not be implemented directly. However, F may be partially inferred from a discretized space-time solution of u for a given c. To see this, let the discretized solution be $\boldsymbol{u} \equiv \{u(t_i, x_j)\}_{i=1,\cdots,M,\, j=1,\cdots,N}$, where $0 \leq t_1 < t_2 < \cdots < t_M \leq 1$ and $0 \leq x_1 < x_2 < \cdots < x_N \leq 1$ indicate the time and space discretization. Given $\boldsymbol{u}$, the $\frac{\partial u}{\partial t}$ and $\frac{\partial u}{\partial x}$ can be sampled by finite difference. Because (1.2) can be written as

$$\frac{\partial u}{\partial t} + \frac{dF}{du}\frac{\partial u}{\partial x} = c, \quad x \in [0,1], \quad t \in [0,1] \tag{1.3}$$

away from the shock wave, the samples of $\frac{\partial u}{\partial t}$ and $\frac{\partial u}{\partial x}$ can be plugged into (1.3) to obtain samples of $\frac{dF}{du}$. The reasoning remains intact at the shock wave, where $\frac{dF}{du}$ in (1.3) is replaced by the finite difference form $\frac{\Delta F}{\Delta u}$ according to the Rankine-Hugoniot condition. Based upon the sampled $\frac{dF}{du}$ and $\frac{\Delta F}{\Delta u}$, the unknown flux function F can be approximated up to a constant for values of u that appeared in the solution, by using indefinite integral. Let $\tilde{F}$ be the approximation for F. An alternative conservation law can be proposed

$$\frac{\partial \tilde{u}}{\partial t} + \frac{\partial \tilde{F}(\tilde{u})}{\partial x} = c, \quad x \in [0,1], \quad t \in [0,1], \tag{1.4}$$

that approximates the true but unknown conservation law (1.2), where $\tilde{u}$ is the solution associated with $\tilde{F}$, in the following sense: If $\tilde{F}$ and F are off by a constant

a, i.e. $\tilde{F} = F + a$, then $\frac{dF(u)}{du} = \frac{d(F(u)+a)}{du} = \frac{d\tilde{F}(u)}{du}$; therefore, the solutions of (1.2) and (1.4) to any initial value problem will be the same. The gradient of any objective function $\xi(c) \equiv \xi(u(c), c)$ can be obtained by the adjoint method [11]. The gradient is

$$\frac{d\xi}{dc} = \int_0^1 \int_0^1 \left(\frac{\partial \xi}{\partial c} + \lambda \right) dxdt\,, \tag{1.5}$$

where λ, the adjoint solution, satisfies

$$\frac{\partial \lambda}{\partial t} + \frac{\partial}{\partial x}\left(\lambda \frac{dF}{du} \right) = -\frac{\partial \xi}{\partial u}\,. \tag{1.6}$$

In (1.6), $\frac{dF}{du}$ and $\frac{\partial \xi}{\partial u}$ are defined on the solution u of (1.3) [11]. Similarly, the gradient of $\tilde{\xi}(c) \equiv \xi(\tilde{u}(c), c)$ is

$$\frac{d\tilde{\xi}}{dc} = \int_0^1 \int_0^1 \left(\frac{\partial \xi}{\partial c} + \tilde{\lambda} \right) dxdt\,, \tag{1.7}$$

where $\tilde{\lambda}$, the adjoint solution, satisfies

$$\frac{\partial \tilde{\lambda}}{\partial t} + \frac{\partial}{\partial x}\left(\tilde{\lambda} \frac{d\tilde{F}}{du} \right) = -\frac{\partial \xi}{\partial u}\,. \tag{1.8}$$

In (1.8), $\frac{d\tilde{F}}{du}$ and $\frac{\partial \xi}{\partial u}$ are defined on the solution $\tilde{u}$ of (1.4). If the two solutions, u and $\tilde{u}$, are the same, and if $\frac{dF}{du} = \frac{d\tilde{F}}{du}$ on the solution, then the adjoint solutions, λ and $\tilde{\lambda}$ will be the same. As a result, the gradients, (1.5) and (1.7), will be the same. Therefore $\frac{d\tilde{\xi}}{dc}$ can drive the optimization constrained by (1.2). A simulator for the approximated conservation law is named "twin model" because it behaves as an adjoint-enabled twin of the original simulator. If a conservation law has a system of equations and/or has a greater-than-one spatial dimension, the above simple method to recover the flux function from a solution will no longer work. Nonetheless, much information about the flux function can be extracted from the solution. Given some additional information of the conservation law, one may be able to recover the unknown aspects of the flux function. The details of this topic are discussed in Chapter 2.

I call a simulator **gray-box** if it meets two conditions:

1. The adjoint is unavailable, and the adjoint is impractical to be implemented into the source code.

2. The full space-time solution of relevant flow quantities is available.

My thesis considers gray-box simulators whose flux functions contain unknown algebraic dependence on the state, and whose boundary conditions, initial conditions, source terms, and differential operators in the flux functions are known. For example, a Navier-Stokes flow simulation can have an unknown state equation; in other words, the pressure term contains an unknown algebraic dependence on the state. Another example is a reservoir simulation whose phase velocities are governed by Darcy's law [4], which can contain unknown algebraic dependence on the phases' saturations. In contrast, a simulator is named **open-box** if condition one is violated. For example, *OpenFOAM* [61] is an open-source fluid simulator where adjoint can be implemented directly into its source code, so it is open-box by definition. Open-box simulators enjoy the benefit of efficient gradient computation brought by adjoint and thereby are not within the research scope of my thesis. If condition one is met, but two is violated, a simulator is named "black-box". For example, *Aspen* [62], an industrial chemical reactor simulator, provides neither the adjoint nor the full space-time solution. Black-box simulators are simply calculators for the objective function. Due to the lack of space-time solution, adjoint can not be enabled using the twin model. Gray-box simulators are ubiquitous in many engineering applications. Examples are Fluent [106] and CFX [107] for computational fluid dynamics, and ECLIPSE (Schlumberger), PSim (ConocoPhillips), and MORES (Shell) for petroleum reservoir simulations. My thesis will only investigate gray-box simulators.

My thesis aims at reducing the number of expensive iterations in the optimization constrained by gray-box simulators. Motivated by the adjoint gradient computation, a mathematical procedure will be developed to estimate the adjoint gradient by leveraging the full space-time solution. Also, my thesis will investigate how the estimated gradient can facilitate a suitable optimization algorithm to reduce the

number of iterations. Finally, the iteration reduction achieved by my approach will be assessed, especially for problems with many control parameters.

Instead of discussing gray-box simulators in general, my thesis only focuses on simulators with partially unknown flux function, while their boundary condition, initial condition, and the source term are known. For example, one may know that the flux depends on certain variables, but the specific function form of such dependence is unknown. This assumption is valid for some applications, such as simulating a petroleum reservoir with polymer injection. The flow in such reservoir is governed by multiphase multicomponent porous medium transport equations [4]. The initial condition is usually given at the equilibrium state, the boundary is usually described by a no-flux condition, and the source term can be modeled as controls with given flow rate or wellbore pressure. Usually, the flux function is given by Darcy's law which involves physical models like the permeability[1] and the viscosity[2]. The mechanism through which the injected polymer modifies the rock permeability and flow viscosity can be unavailable. Thereby the flux is partially unknown. The specific form of PDE considered in my thesis is given in Section 1.2. It is a future work to extend my research to more general gray-box settings where the initial condition, boundary condition, source term, and the flux are jointly unknown.

1.2 Problem Formulation

Consider the optimization problem

$$\begin{aligned} c^* &= \underset{c_{\min} \le c \le c_{\max}}{\operatorname{argmax}} \quad \xi(\boldsymbol{u}, c) \\ \xi(c) &= \sum_{i=1}^{M} \sum_{j=1}^{N} w_{ij} f(\boldsymbol{u}_{ij}, c; t_i, x_j) \approx \int_0^T \int_\Omega f(u, c; t, x) d\boldsymbol{x} dt \end{aligned} \tag{1.9}$$

[1]The permeability quantifies the easiness of liquids to pass through the rock.
[2]The viscosity quantifies the internal friction of the liquid flow.

where $\boldsymbol{u}$ and u are the discretized and continuous space-time solutions of a gray-box conservation law simulator. $\boldsymbol{u}$ and u depend on the control variables c. We assume ξ to be a twice differentiable function. The spatial coordinate is $x \in \Omega$ and the time is $t \in [0, T]$. $i = 1, \cdots, M$ and $j = 1, \cdots, N$ are the indices for the time and space grid points. f is a given function that depends on u, c, t, and x. w_{ij}'s are the quadrature weights for the integration. $c \in \mathbb{R}^d$ indicates the control variable. $c_{\min}$ and $c_{\max}$ are elementwise bound constraints.

The gray-box simulator solves the partial differential equation (PDE)

$$\frac{\partial u}{\partial t} + \nabla \cdot \big(DF(u)\big) = q(u, c)\,, \tag{1.10}$$

which is a system of k equations. The initial and boundary conditions are known. D is a known differential operator that may depend on u, and F is an unknown function. Although the state variables that F depend on are known, the algebraic form of such dependence is unknown. q is a known source term that depends on u and c. The simulator for (1.10) does not have the adjoint capability, and it is infeasible to implement the adjoint method into its source code. But the full space-time solution $\boldsymbol{u}$ is provided. The steady-state conservation law is a special case of the unsteady one, so it will not be discussed separately.

For example, consider a 1-D scalar-state convective equation

$$\frac{\partial u}{\partial t} + \frac{\partial}{\partial x} F(u) = c\,, \tag{1.11}$$

can be described by (1.10). The flux function F is known to depend on the local value of the state variable u, but the algebraic form of the dependence is unknown. In this case, F represents the entire unknown flux function while D equals 1. If $F(u) = \frac{1}{2}u^2$, (1.11) is the Burger's equation; If $F(u) = \frac{u^2}{1+(1-u)^2}$, (1.11) is a Buckley-Leverett equation [3].

Another example can be the compressible, viscous, adiabatic Navier-Stokes equation

$$\frac{\partial}{\partial t}\begin{pmatrix}\rho\\ \rho u\\ \rho v\\ \rho E\end{pmatrix}\frac{\partial}{\partial x}\begin{pmatrix}\rho u\\ \rho u^2+p-\sigma_{xx}\\ \rho uv-\sigma_{xy}\\ u(E\rho+p)-\sigma_{xx}u-\sigma_{xy}v\end{pmatrix}+\frac{\partial}{\partial y}\begin{pmatrix}\rho v\\ \rho uv-\sigma_{xy}\\ \rho v^2+p-\sigma_{yy}\\ v(E\rho+p)-\sigma_{xy}u-\sigma_{yy}v\end{pmatrix}=\mathbf{0}\,, \tag{1.12}$$

where

$$\begin{aligned}\sigma_{xx}&=\mu\left(2\frac{\partial u}{\partial x}-\frac{2}{3}\left(\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}\right)\right)\\ \sigma_{yy}&=\mu\left(2\frac{\partial v}{\partial y}-\frac{2}{3}\left(\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}\right)\right)\\ \sigma_{xy}&=\mu\left(\frac{\partial u}{\partial y}+\frac{\partial v}{\partial x}\right)\end{aligned}\,. \tag{1.13}$$

Let ρ, u, v, E, U, and p be the density, Cartesian velocity components, total energy, internal energy, and pressure. The pressure depends on the density and energy, but the algebraic form of the dependence is unknown. In this example, $F(u)$ corresponds to the pressure equation $p = p(\rho, U)$, and D corresponds to the known components of the flux functions.

My thesis does not accomodate the PDEs that contain unknown source terms or flux functions with unknown differential operators, thus limiting the applicability of my thesis. It is a future research topic to extend the methods developed in this thesis for such PDEs.

My thesis focuses on reducing the number of gray-box simulations in the optimization, especially for problems where d, the dimensionality of the control variable, is large. I assume that the computational cost is dominated by the repeated gray-box simulation, while the cost of the optimization algorithm is relatively small. Chapter 2 develops a mathematical procedure, called the twin-model method, which enables adjoint

gradient computation by leveraging the full space-time solution. Based on previous research [66, 67, 71, 72, 74, 76, 77], Chapter 3 develops an optimization algorithm that takes advantage of the estimated gradient to achieve iteration reduction. The utility of the estimated gradient for optimization is analyzed both numerically and theoretically.

1.3 Literature Review

Given the background, I review the literature on derivative-free optimization and gradient-based optimization, in which the Bayesian optimization method is particularly investigated. In addition, I review the adjoint method because it is an essential ingredient for Chapter 2. Finally, I review methods for adaptive basis construction, which is useful for the adaptive parameterization of a twin model.

1.3.1 Review of Optimization Methods

Optimization methods can be categorized into derivative-free and gradient-based methods [41], depending on whether the gradient information is used. In this section, I review the two types of methods.

Derivative-Free Optimization

Derivative-free optimization (DFO) requires only the availability of objective function values but no gradient information [41]; thus, it is useful when the gradient is unavailable, unreliable, or too expensive to obtain. Such methods are suitable for problems constrained by black-box simulators.

Depending on whether a local or global optimum is desired, DFO methods can be categorized into local methods and global methods [41]. Local methods seek a local optimum which is also the global optimum for convex problems. A local method is the derivative-free trust-region method [47]. The derivative-free trust-region method

introduces a surrogate model that interpolates the objective evaluations [48, 51]. The surrogate model is cheap to evaluate and presumably accurate within a trust region: an adaptive neighborhood around the current iteration [46, 47]. At each iteration, the surrogate is optimized in a domain bounded by the trust region to generate candidate steps for additional objective evaluations [46, 47].

Global methods seek the global optimum. Example methods include the branch-and-bound search [52], evolution methods [53], and Bayesian methods [71, 73, 91]. The branch-and-bound search sequentially partitions the entire control space into a tree structure and determines lower and upper bounds for the optimum [52]. Partitions that are inferior are eliminated in the course of the search [52]. The bounds are usually obtained through the assumption of the Lipschitz continuity or statistical bounds for the objective function [52]. Evolution methods maintain a population of candidate controls, which adapt and mutate in a way that resembles natural phenomenons such as natural selection [54, 56] and swarm intelligence [55]. Bayesian methods model the objective function as a random member function from a stochastic process. At each iteration, the statistics of the stochastic process are calculated and the posterior, a probability measure, of the objective, is updated using Bayesian metrics [71, 72]. The posterior is used to pick the next candidate step that best balances the exploration of unsampled regions and the exploitation around the sampled optimum [73, 82, 69]. Details of Bayesian optimization methods are discussed in Section 1.3.1.

Because many real-world problems are non-convex, global methods are usually preferred to local methods if the global optimum is desired [41]. Besides, DFO methods usually require a large number of function evaluations to converge, especially when the dimension of control is large [41]. This issue can be alleviated by incorporating the gradient information [66, 74]. The details are discussed in the next subsection.

Gradient-Based Optimization

Gradient-based optimization (GBO) requires the availability of the gradient values [60, 83]. A gradient value, if it exists, provides the optimal infinitesimal change of control variables at each iterate and thus is useful in searching for better control. Similar to DFO, GBO also can be categorized into local methods and global methods [60]. Examples of local GBO methods include the gradient descent methods [84, 104], the conjugate gradient methods [85, 86], and the quasi-Newton methods [40, 42]. The gradient descent methods and the conjugate gradient methods choose the search step in the direction of either the gradient [84, 104] or a conjugate gradient [85, 86]. Quasi-Newton methods, such as the Broyden-Fletcher-Goldfarb-Shannon (BFGS) method [40], approximate the Hessian matrix using a series of gradient values. The approximated Hessian allows a local quadratic approximation to the objective function, which determines the search direction and stepsize by the Newton's method [40]. In addition, some local DFO methods can be enhanced to use gradient information [58, 59]. For instance, in trust-region methods, the construction of local surrogates can incorporate gradient values if available [58, 59]. The usage of a gradient usually improves the surrogate's accuracy, thus enhancing the quality of the search step and thereby reducing the required number of iterations [58, 59].

Global GBO methods search for the global optimum using gradient values [60, 83]. Some global GBO methods can trace their development to corresponding DFO methods. For example, the stochastic gradient-based global optimization method (StoGo) [87, 88] works by partitioning the control space and bounding the optimum in the same way as the branch-and-bound method [52]. But the search in each partition is performed by gradient-based algorithms such as BFGS [40].

My thesis is particularly interested in the gradient-based Bayesian optimization method [75]. In this method, the posterior of the objective function assimilates both the gradient and function values in a CoKriging framework [66, 75]. The details of

my treatment are discussed in Section 1.3.1 and Chapter 3. It is reasonable to expect that the inclusion of gradient values results in a more accurate posterior mean and reduced posterior uncertainty, which in turn reduces the number of iterations required to achieve near-optimality. The effect of iteration reduction is analyzed numerically in Chapter 3.

A property of the Bayesian method is that the search step can be determined using all available objective and gradient values [71, 82]. Also, given the current knowledge of the objective function which is represented in Bayesian probability, the search step is optimal under a particular metric such as the expected improvement metric [71, 82]. The advantage of such properties can be justified when the objective and gradient evaluations are dominantly more expensive than the overhead of optimization algorithm [71]. Besides, my thesis proves that the Bayesian optimization method is convergent even if the gradient values are estimated inexactly, which is discussed in Section 3.3. The conclusion of Section 3.3 reveals that, under some assumptions of the objective and the inexact gradient, a Bayesian optimization algorithm can find the optimum regardless of the accuracy of the gradient estimation.

To achieve the desired objective valuation, GBO methods generally require fewer iterations than DFO methods for problems with many control variables [60, 83]. GBO methods can be efficiently applied to optimization constrained by open-box simulators because the gradient is efficiently computable by the adjoint method [11, 60], which is introduced in the next subsection. My thesis extends GBO to optimization constrained by gray-box simulation by estimating the gradient using the full space-time solution.

Bayesian Optimization

Similar to other kinds of optimization, Bayesian optimization aims at finding the maximum of a function $\xi(\cdot)$ in a bounded set $\mathcal{C} \subset \mathbb{R}^d$ [71, 72, 82]. However, Bayesian optimization distinguishes from other methods by maintaining a probabilistic model

for ξ [71, 72, 82]. The probabilistic model is exploited to make decisions about where to invest the next function evaluation in $\mathcal{C}$ [71, 72, 82]. In addition, it uses all information of available evaluations, not just local evaluations, to direct the search step [71, 72, 82].

Consider the case when the objective function evaluation is available. Bayesian optimization begins by assuming that the objective function is sampled from a stochastic process [71, 72, 82]. A stochastic process is a function

$$\begin{aligned} f \ &: \mathcal{C} \times \Omega \to \mathbb{R} \\ &(c, \omega) \to f(c, \omega) \end{aligned}, \tag{1.14}$$

where for any $c \in \mathcal{C}$. w is a random variable that models the stochastic dependence of f. $f(c, \cdot)$ is a random variable defined on the probability space $(\Omega, \Sigma, \mathbb{P})$. The objective function ξ is assumed to be a sampled function from the stochastic process $\xi(\cdot) = f(\cdot, \omega^*)$, where $\omega^* \in \Omega$ is deterministic but unknown. My thesis will use the notations $\xi(\cdot)$, $f(\cdot, \omega)$, and $f(\cdot, \omega^*)$ interchangeably when the context is clear.

A stationary Gaussian process is a type of stochastic process that is used ubiquitously in Bayesian optimization [89]. For any given ω and any finite set of N points $\{c_i \in \mathcal{C}\}_{i=1}^N$, a stationary Gaussian process $f(\cdot, \cdot)$ has the property that $\{f(c_i, \cdot)\}_{i=1}^N$ are multivariate Gaussian distributed; in addition, the distribution remains unchanged if c_i's are all added by the same constant in $\mathcal{C}$. The Gaussian process is solely determined by its mean function $m(c)$ and its covariance function $K(c, c')$ [89]

$$\begin{aligned} m(c) &= \mathbb{E}_\omega\big[f(c, \omega)\big] \\ K(c, c') &= \mathbb{E}_\omega\Big[\big(f(c, \omega) - m(c)\big)\big(f(c', \omega) - m(c')\big)\Big], \end{aligned} \tag{1.15}$$

for any $c, c' \in \mathcal{C}$, which is denoted by $f \sim \mathcal{N}(m, K)$. Conditioned on a set of samples $\{\xi(c_1), \cdots, \xi(c_N)\}$, the posterior is also a Gaussian process with the mean

and covariance [89]

$$\begin{aligned} \tilde{m}(c) &= m(c) + K(c, \underline{c}_n) K(\underline{c}_n, \underline{c}_n)^{-1} \left(\xi(\underline{c}_n) - m(\underline{c}_n)\right) \\ \tilde{K}(c, c') &= K(c, c') - K(c, \underline{c}_n) K(\underline{c}_n, \underline{c}_n)^{-1} K(\underline{c}_n, c') \end{aligned}, \tag{1.16}$$

where $\underline{c}_n = (c_1, \cdots, c_N)$, $\xi(\underline{c}_n) = (\xi(c_1), \cdots, \xi(c_N))^T$, $m(\underline{c}_n) = (m(c_1), \cdots, m(c_N))^T$, $K(c, \underline{c}_n) = K(\underline{c}_n, c)^T = (K(c, c_1), \cdots, K(c, c_N))$, and

$$K(\underline{c}_n, \underline{c}_n) = \begin{pmatrix} K(c_1, c_1) & \cdots & K(c_1, c_N) \\ \vdots & \ddots & \vdots \\ K(c_N, c_1) & \cdots & K(c_N, c_N) \end{pmatrix}.$$

Without prior knowledge about the underlying function, $m(\cdot)$ is usually modeled as a constant independent of c [89]. In many cases, the covariance are assumed isotropic, indicating that $K(c, c')$ only depends on the L_2 norm $\|c - c'\|$ [89]. There are many choices for K, such as the exponential kernel, the squared exponential kernel, and the Matérn kernels, each embeds different degrees of smoothness (differentiability) for the underlying function. For a survey of various covariance functions, I refer to the Chapter 4 in [89]. Among such choices, the Matérn 5/2 kernel [90]

$$K(c, c') = \sigma^2 \left(1 + \frac{\sqrt{5}\|c - c'\|}{L} + \frac{5\|c - c'\|^2}{3L^2}\right) \exp\left(-\frac{\sqrt{5}\|c - c'\|}{L}\right), \tag{1.17}$$

has been recommended because it results in functions that are twice differentiable, an assumption made by, e.g. quasi-Newton methods, but without further smoothness [71]. My thesis will focus on using the Matérn 5/2 kernel. Notice the parameters L and σ, known as the hyperparameters, are yet to be determined. They can be determined by the posterior maximum likelihood estimation (MLE) or by a fully-Bayesian approach [71, 82]. I refer to the reference [71] for the details and a comparison of these treatments. My thesis will focus on MLE due to its simpler numerical implementation.

Based on the posterior and the current best evaluation $c_{\text{best}} = \text{argmax}_{c \in \mathcal{C}_n} \xi(c)$, Bayesian optimization introduces an acquisition function, $a : \mathcal{C} \to \mathbb{R}^+$, which evaluates the expected utility of investing the next sample at $c \in \mathcal{C}$ [69, 71, 72, 82, 91]. The location of the next sample is determined by an optimization $c_{N+1} = \text{argmax}_{c \in \mathcal{C}} a(c)$ [69, 71, 72, 82, 91]. In most cases, a greedy acquisition function is used, which evaluates the one-step-lookahead utility [69, 71, 72, 82, 91]. There are several choices for the acquisition function, such as

- the probability of improvement (PI) [91],

$$a_{\text{PI}}(c) = \Phi(\gamma(c)) , \tag{1.18}$$

- the expected improvement (EI) [72, 73],

$$a_{\text{EI}}(c) = \sigma(c)\big(\gamma(c)\Phi(\gamma(c)) + \mathcal{N}(\gamma(c))\big) , \tag{1.19}$$

- and the upper confidence bound (UCB) [69],

$$a_{\text{UCB}}(c) = \mu(c) + \kappa\sigma(c) , \tag{1.20}$$

  with a tunable parameter $\kappa > 0$,

where μ, σ are the posterior mean and variance, $\gamma(c) = \sigma^{-1}(c)\big(\mu(c) - \xi(c_{\text{best}})\big)$, and $\Phi, \mathcal{N}$ indicate the cumulative and density functions for the standard normal distribution. My thesis will focus on the EI acquisition function, as it behaves better than the PI and requires no extra tunable parameters [71]. Because (1.19) has a closed-form gradient, the acquisition function can be maximized by a global GBO method, e.g., StoGo [88], to obtain its global maximum.

Although my thesis only focuses on bound constraints as shown in (1.9), Bayesian optimization can accommodate more general inequality and equality constraints [97]. The constraints can be enforced by modifying the objective, such as the penalty

method [92], the augmented Lagrangian method [93], and the barrier function method [94]. They also can be enforced by modifying the acquisition function, such as the recently developed expected improvement with constraints (EIC) method [95], and the integrated expected conditional improvement (IECI) method [96]. See Chapter 2 of [97] for a detailed review of constrained Bayesian optimization.

In addition to function evaluations $\xi(\underline{c}_n)$, Bayesian optimization admits gradient information [66, 74]. In Chapter 3, I investigate the scenario where the gradient evaluations are inexact [77]. The Bayesian optimization method developed in my thesis allows both the exact function evaluation and the inexact gradient evaluation.

1.3.2 The Adjoint Method

Consider a differentiable objective function constrained by a conservation law PDE (1.10). Let the objective function be $\xi(u, c)$, $c \in \mathbb{R}^d$, and let the PDE (1.10) be abstracted as $\mathcal{F}(u, c) = 0$. $\mathcal{F}$ is a parameterized differential operator, together with boundary conditions and/or initial conditions, which uniquely define a u for each c. The gradient $\frac{d\xi}{dc}$ can be estimated trivially by finite difference. The ith component of the gradient is given by

$$\left(\frac{d\xi}{dc}\right)_i \approx \frac{1}{\delta}\big(\xi(u + \Delta u_i, c + \delta e_i) - \xi(u, c)\big)\,, \tag{1.21}$$

where

$$\mathcal{F}(u, c) = 0\,, \quad \mathcal{F}(u + \Delta u_i, c + \delta e_i) = 0\,. \tag{1.22}$$

e_i indicates the ith unit Cartesian basis vector in $\mathbb{R}^d$, and $\delta > 0$ indicates a small perturbation. Because (1.22) needs to be solved for every δe_i, so that the corresponding Δu_i can be used in (1.21), $d+1$ PDE simulations are required to evaluate the gradient. As explained in Section 1.3.1, d can be large in many control optimization problems.

Therefore, it can be costly to evaluate the gradient by finite difference.

In contrast, the adjoint method evaluates the gradient using only one PDE simulation plus one adjoint simulation [11]. To see this, linearize $\mathcal{F}(u, c) = 0$ into a variational form

$$\delta\mathcal{F} = \frac{\partial\mathcal{F}}{\partial u}\delta u + \frac{\partial\mathcal{F}}{\partial c}\delta c = 0\,, \tag{1.23}$$

which gives

$$\frac{du}{dc} = -\left(\frac{\partial\mathcal{F}}{\partial u}\right)^{-1}\frac{\partial\mathcal{F}}{\partial c} \tag{1.24}$$

Using (1.24), $\frac{d\xi}{dc}$ can be expressed by

$$\begin{aligned}\frac{d\xi}{dc} &= \frac{\partial\xi}{\partial u}\frac{du}{dc} + \frac{\partial\xi}{\partial c} \\ &= -\frac{\partial\xi}{\partial u}\left(\frac{\partial\mathcal{F}}{\partial u}\right)^{-1}\frac{\partial\mathcal{F}}{\partial c} + \frac{\partial\xi}{\partial c}\,, \\ &= -\lambda^T\frac{\partial\mathcal{F}}{\partial c} + \frac{\partial\xi}{\partial c}\end{aligned} \tag{1.25}$$

where λ, the adjoint state, is given by the adjoint equation

$$\left(\frac{\partial\mathcal{F}}{\partial u}\right)^T\lambda = \left(\frac{\partial\xi}{\partial u}\right)^T \tag{1.26}$$

Therefore, the gradient can be evaluated by (1.25) using one simulation of $\mathcal{F}(u, c) = 0$ and one simulation of (1.26) that solves for λ.

Adjoint methods can be categorized into continuous adjoint and discrete adjoint methods, depending on whether the linearization or the discretization is executed first [15]. The above procedure, (1.23) through (1.26), is the continuous adjoint, where $\mathcal{F}$ is a differential operator. The continuous adjoint method linearizes the continuous PDE $\mathcal{F}(u, c) = 0$ first and then discretizes the adjoint equation (1.26) [11]. In (1.26), $\left(\frac{\partial\mathcal{F}}{\partial u}\right)^T$ can be derived as another differential operator. With proper boundary and/or initial conditions, it uniquely determines the adjoint solution λ. See [19] for a detailed derivation of the continuous adjoint equation.

The discrete adjoint method [17] discretizes $\mathcal{F}(u, c) = 0$ first. After the discretization, u and c become vectors $\boldsymbol{u}$ and $\boldsymbol{c}$. $\boldsymbol{u}$ is defined implicitly by the system $\mathcal{F}_d(\boldsymbol{u}, \boldsymbol{c}) = 0$, where $\mathcal{F}_d$ indicates the discretized difference operator, a nonlinear function whose output is of the same dimension as its first input $\boldsymbol{u}$. Using the same derivation as (1.23) through (1.26), the discrete adjoint equation can be obtained

$$\left(\frac{\partial \mathcal{F}_d}{\partial \boldsymbol{u}}\right)^T \boldsymbol{\lambda} = \left(\frac{\partial \xi}{\partial \boldsymbol{u}}\right)^T, \tag{1.27}$$

which is a linear system of equations. $\left(\frac{\partial \mathcal{F}_d}{\partial \boldsymbol{u}}\right)^T$ is derived as another difference operator which is a square matrix. It contains the discretized boundary and initial conditions, and uniquely determines the discrete adjoint vector $\boldsymbol{\lambda}$, which subsequently determines the gradient

$$\frac{d\xi}{d\boldsymbol{c}} = -\boldsymbol{\lambda}^T \frac{\partial \mathcal{F}_d}{\partial \boldsymbol{c}} + \frac{\partial \xi}{\partial \boldsymbol{c}}. \tag{1.28}$$

See Chapter 1 of [20] for a detailed derivation of the discrete adjoint.

The adjoint method has seen wide applications in optimization problems constrained by conservation law simulations, such as in airfoil design [12, 13, 14], adaptive mesh refinement [20], injection policy optimization in petroleum reservoirs [2], history matching in reservoir geophysics [15], and optimal well placement in reservoir management [16].

1.3.3 Adaptive Basis Construction

The unknown function F in (1.10) can be approximated by a linear combination of basis functions [24]. An over-complete or incomplete set of bases can negatively affect the approximation due to overfitting or underfitting [25]. Therefore, adaptive basis construction is needed.

Consider the problem of function approximation in a bounded domain. Square-integrable functions can be represented by the linear combination of a set of basis functions [24], $\{\phi_i\}_{i\in\mathbb{N}}$, such as the polynomial basis, Fourier basis, and the wavelet basis [102].

$$F(\cdot) = \sum_{i\in\mathbb{N}} \alpha_i \phi_i(\cdot)\,, \tag{1.29}$$

where ϕ_i's are linearly independent basis functions, α_i's are the coefficients, and i indices the basis. For a rigorous development of function approximation and basis functions, I refer to the book [24].

For example, a bivariate function can be represented by monomials (Weierstrass approximation theorem [105])

$$1,\ u_1,\ u_1^2,\ u_2,\ u_1u_2,\ u_1^2u_2,\ u_2^2,\ u_1u_2^2,\ u_1^2u_2^2,\ \cdots\,.$$

on any real interval $[a, b]$.

Let $\mathcal{A}$ be a finite non-empty subset of $\mathbb{N}$, F can be approximated using a subset of bases,

$$F(\cdot) \approx \sum_{i\in\mathcal{A}} \alpha_i \phi_i(\cdot)\,, \tag{1.30}$$

where $\{\phi_i\}_{i\in\mathcal{A}}$ is called a basis dictionary [31]. The approximation is solely determined by the choices of the dictionary and the coefficients. For example, in polynomial approximation, the basis dictionary can consist of the basis whose total polynomial degree does not exceed $p \in \mathbb{N}$ [26]. Given a dictionary, the coefficients for $\tilde{F}$ can be determined by the minimization [26]

$$\boldsymbol{\alpha}^* = \underset{\boldsymbol{\alpha}\in\mathbb{R}^{|\mathcal{A}|}}{\operatorname{argmin}} \left\| \tilde{F} - \sum_{i\in\mathcal{A}} \alpha_i \phi_i \right\|_{L_p}, \tag{1.31}$$

where $\|\cdot\|_{L_p}$ indicates the L_p norm[3]. My thesis parameterizes the twin-model flux $\tilde{F}$ and optimizes the coefficients, so the twin model serves as a proxy of the gray-box model. The details are discussed in Section 2.2.

If the dictionary is pre-determined, its cardinality can increase as the number of variables increases, and as the basis complexity increases [26]. For example, for the d-variate polynomial basis, the total number of bases is d^p if one bounds the polynomial degree of each variable by p; and is $\binom{p+d}{d}$ if one bounds the total degree by p [26].

In many applications, one may deliver a similarly accurate approximation by using a much smaller subset of the dictionary as the bases than using all the basis functions in the dictionary [26, 28, 31, 44]. To exploit the sparse structure, only significant bases shall be selected, and the selection process shall be adaptive depending on the values of function evaluations. There are several methods that adaptively determine the sparsity, such as Lasso regularization [44], matching pursuit [31], and basis pursuit [28]. Lasso regularization adds a penalty $\lambda \sum_{i \in \mathcal{A}} |\alpha_i|$ to the approximation error, where $\lambda > 0$ is a tunable parameter [44]. The larger λ is, the sparser the basis functions will be. In this way, Lasso balances the approximation error and the number of non-zero coefficients [44]. Matching pursuit adopts a greedy, stepwise approach [31]. It either selects a significant basis one at a time (forward selection) from a dictionary [32], or prunes an insignificant basis one at a time (backward pruning) from the dictionary [33]. Basis pursuit minimizes $\|\boldsymbol{\alpha}\|_{L_1}$ subject to (1.29), which is equivalently reformulated and efficiently solved as a linear programming problem [28].

Conventionally, the dictionary for the sparse approximation needs to be predetermined, with the belief that the dictionary is a superset of the required bases for an accurate approximation [35]. This can be problematic because the required bases are unknown a priori. To address this issue, methods have been devised that construct an adaptive dictionary [34, 35, 36]. Although different in details, such methods share the same

[3]Usually $p = 1$ [28] or 2 [29, 31].

approach: In the beginning, some trivial bases are given as inputs. For example, the starting basis can be 1 for a polynomial basis [34]. The starting bases serve as seeds from which more complex bases grow. I refer to [34, 35, 36] for more details of the heuristics. Then a dictionary is built up progressively by iterating over a forward step and a backward step [34, 35, 36]. The forward step searches over a candidate set of bases and appends the significant ones to the dictionary [34, 35, 36]. The backward step searches over the current dictionary and removes the insignificant ones from the dictionary [34, 35, 36]. The iteration stops only when no alternation is made to the dictionary or when a targeted accuracy is achieved, without bounding the basis complexity a prior [34, 35, 36]. Such approach is adopted in my thesis to build up the bases for $\tilde{F}$. The details are discussed in Section 2.3.

1.4 Thesis Objectives

Based on the motivation and literature review, we find it useful to enable adjoint gradient computation for gray-box conservation law simulations whose flux functions have an unknown algebraic dependence on the state variables. We also need to exploit the estimated gradient to optimize more efficiently, especially for problems with many control variables. To summarize, the objectives of my thesis are

1. to develop an adjoint approach that estimates the gradient of objective functions constrained by gray-box conservation law simulations with unknown algebraic dependence in the flux functions, by leveraging the space-time solution;

2. to assess the utility of the estimated gradient in a suitable gradient-based optimization method; and

3. to demonstrate the effectiveness of the developed procedure in several numerical examples, given a limited computational budget.

1.5 Outline

My thesis is organized as follows. Chapter 2 describes a method to estimate the gradient of an objective function constrained by a gray-box simulation, at a cost independent of the dimensionality of the gradient. This is achieved through first training a twin model and then applying the adjoint method to the trained twin model. To train a twin model, two metrics, the solution mismatch and the integrated truncation error, are presented, and the relationship between the two metrics is studied. Then we present a method to parameterize the unknown component of the twin model. Using the two metrics and the parameterization, a twin-model algorithm is developed to approximate the unknown components in the twin model. Finally, the twin-model algorithm is demonstrated by several numerical examples. Chapter 3 develops a global optimization method by using the estimated gradient obtained from the twin-model algorithm in Chapter 2. The gray-box objective function and the estimated gradient are modeled as unknown realizations of Gaussian processes. Based on the Gaussian process model, a Bayesian optimization algorithm is developed that leverages the estimated gradient for more efficient optimization. Its convergence properties are studied. Finally, the twin-model Bayesian optimization algorithm is demonstrated by several numerical examples. Chapter 4 summarizes the thesis and my contributions and proposes several directions of future works.

Chapter 2

Estimate the Gradient by Using the Space-Time Solution

This chapter develops a method to estimate the gradient by using the space-time solution of gray-box conservation law simulations.

Chapter 1 considered a code that simulates a conservation law (1.2) with an unknown F,

$$\frac{\partial u}{\partial t} + \frac{\partial F(u)}{\partial x} = c\,, \quad x \in [0,1]\,, \;\; t \in [0,1]\,,$$

with proper initial and boundary conditions, for a control variable c. Such simulator is named gray-box, and its discretized space-time solution is named gray-box solution. It is explained that F can be approximated up to a constant for values of u that appeared in the gray-box solution, by utilizing the gray-box solution. Therefore, a twin model that simulates (1.4),

$$\frac{\partial \tilde{u}}{\partial t} + \frac{\partial \tilde{F}(\tilde{u})}{\partial x} = c\,, \quad x \in [0,1]\,, \;\; t \in [0,1]\,,$$

can be obtained, where $\tilde{F}$ is the approximated flux. It is also explained that the adjoint method can be applied to the twin model to estimate the gradient of any

objective function with respect to c. Finally, it is envisioned that the adjoint gradient of the twin model can drive the optimization of the objective function constrained by the gray-box model.

The above example involves only one equation and one-dimensional space. This chapter develops a more general procedure suitable for systems of equations and for problems with a spatial dimension greater than one.

2.1 Approach

This section discusses the general approach for training a twin model. In particular, the metric of solution mismatch is presented. We then study what aspects of F can be inferred by using the metric for a special case of conservation laws. Besides, another metric, the integrated truncation error, is proposed. The latter metric has less theoretical backup but can be cheaper to evaluate and useful in practice, which will be demonstrated in Section 2.4. The relationship of the two metrics is studied. Finally, we discuss the method to minimize the two metrics.

Consider a gray-box simulator that solves the PDE (1.10),

$$\frac{\partial u}{\partial t} + \nabla \cdot \big(DF(u)\big) = q(u, c)\,,$$

a system of k equations, for $u(t, x)$ with $t \in [0, T]$ and $x \in \Omega$. The PDE has an unknown flux F, but known source term q, and known initial and boundary conditions. Let its discretized space-time solution be $\boldsymbol{u}$. My thesis introduces an open-box simulator solving another PDE, namely the twin model,

$$\frac{\partial \tilde{u}}{\partial t} + \nabla \cdot \big(D\tilde{F}(\tilde{u})\big) = q(\tilde{u}, c)\,, \tag{2.1}$$

which is also a system of k equations with the same source term and the same initial

and boundary conditions. Equation (2.1) differs from (1.10) in its flux. For simplicity, let the solution of the open-box simulator, $\tilde{\boldsymbol{u}}$, be defined on the same spatial grid points and time steps of the gray-box simulator.

The metric used to measure the difference of the twin model and the gray-box model is the solution mismatch. The solution mismatch is defined to be

$$\mathcal{M}_u(\tilde{F}) = \sum_{i=1}^{M}\sum_{j=1}^{N} w_{ij}\left\|\tilde{\boldsymbol{u}}_{ij} - \boldsymbol{u}_{ij}\right\|^2, \tag{2.2}$$

which approximates

$$\int_0^T \int_\Omega \left\|\tilde{u}(t,x) - u(t,x)\right\|^2 \mathrm{d}x\,\mathrm{d}t\,. \tag{2.3}$$

In (2.2), $i = 1, \cdots, M$ are the indices for time grid, and $j = 1, \cdots, N$ are the indices for the space grid. $\|\cdot\|$ is the norm of the state vector. w_{ij}'s are the quadrature weights defined for (2.3). For example, if a uniform Cartesian space-time grid is used, the quadrature weights equal a constant. Notice that $\mathcal{M}_u$ depends solely on $\tilde{F}$ through the twin-model solution $\tilde{\boldsymbol{u}}$ if the quadrature weights and the gray-box solution are given.

Given a function space $\mathcal{S}_F$, I propose to infer a flux $\tilde{F}$ such that $\mathcal{M}_u$ is minimized,

$$\tilde{F}^* = \underset{\tilde{F}\in\mathcal{S}_F}{\operatorname{argmin}}\, \mathcal{M}_u\,. \tag{2.4}$$

The choice for $\mathcal{S}_F$ will be discussed later in Section 2.2. By setting the F in (2.1) to be $\tilde{F}^*$, one obtains a trained twin-model equation

$$\frac{\partial \tilde{u}}{\partial t} + \nabla\cdot\left(D\tilde{F}^*(\tilde{u})\right) = q(\tilde{u}, c)\,, \tag{2.5}$$

Let $\tilde{\boldsymbol{u}}^*$ be the space-time solution of the twin model governed by (2.5). Given $\tilde{F}^*$, $\tilde{\boldsymbol{u}}^*$ depends on c. The gradient of any objective function $\xi(\tilde{\boldsymbol{u}}^*, c)$ with respect to c can be obtained by applying the adjoint method to the trained twin model. The gradient

$\frac{d\xi(\tilde{\boldsymbol{u}}^*,c)}{dc}$ can drive the gradient-based optimization of $\xi(\boldsymbol{u},c)$, where $\boldsymbol{u}$ is the gray-box space-time solution.

The key to inferring F is to leverage the gray-box space-time solution. Compared with the surrogate modeling of the output ξ [98], the advantage of the twin-model approach lies in the usage of the big data, the space-time solution, generated from gray-box PDE solvers. The usage of the big data may lead to the more accurate modeling of F and more accurate predictions of ξ with fewer runs of the gray-box simulation.

For example, the following theorem illustrates what aspect of F can be inferred from the gray-box solution for a special form of (1.2),

$$\frac{\partial u}{\partial t} + \nabla \cdot \big(F(u)\big) = q(u,c)\,,$$

where the spatial dimension is one and the source term is zero.

Theorem 1. *Consider two PDEs*

$$\frac{\partial u}{\partial t} + \frac{\partial F(u)}{\partial x} = 0\,, \text{ and} \tag{2.6}$$

$$\frac{\partial \tilde{u}}{\partial t} + \frac{\partial \tilde{F}(\tilde{u})}{\partial x} = 0\,, \tag{2.7}$$

with the same initial condition $u(0,x) = u_0(x)$. The spatial domain is $(-\infty,\infty)$. The function u_0 is bounded, differentiable, Lipschitz continuous with constant L_u, and has a finite support. F and $\tilde{F}$ are both twice-differentiable and Lipschitz continuous with constant L_F. Let

$$B_u \equiv \left\{ u \,\middle|\, u = u_0(x) \text{ for } x \in \mathbb{R} \text{ that satisfies } \left|\frac{du_0}{dx}\right| \geq \gamma > 0\,, \right\} \subseteq \mathbb{R}\,.$$

be a non-empty and measurable set. We have:
For any $\epsilon > 0$, there exist $\delta > 0$ and $T > 0$ such that

- *if* $|\tilde{u}(t,x) - u(t,x)| < \delta$ *for all* $x \in \mathbb{R}$ *and* $t \in [0,T]$, *then* $\left|\frac{d\tilde{F}}{du} - \frac{dF}{du}\right| < \epsilon$ *for all* $u \in B_u$.

Proof:

We prove false the contradiction of the theorem, which reads:
For any $\delta > 0$ *and* $T > 0$, *there exist* $\epsilon > 0$, *and* $F, \tilde{F}$ *satisfying the conditions stated in theorem 1, such that* $\|\tilde{u} - u\|_\infty < \delta$ *and* $\left\|\frac{d\tilde{F}}{du} - \frac{dF}{du}\right\|_\infty > \epsilon$ *on* B_u.

We show the following exception to the contradiction in order to prove it false. *For any* $\epsilon > 0$ *and any* $F, \tilde{F}$ *satisfying* $\left\|\frac{d\tilde{F}}{du} - \frac{dF}{du}\right\|_\infty > \epsilon$ *on* B_u, *we can find* $\delta > 0$ *and* $T > 0$ *such that* $\|\tilde{u} - u\|_\infty > \delta$. The idea is to construct such an exception by the method of lines [100].

Firstly, assume there is no shock wave for (2.6) and (2.7) for $t \in [0,T]$. Choose a segment in space, $[x_0 - \Delta, x_0]$ with $0 < \Delta < \frac{\epsilon}{L_F L_u}$, that satisfies

- $u_0(x) \in B_u$ for any $x \in [x_0 - \Delta, x_0]$;
- $\left|\frac{d\tilde{F}}{du}\big(u_0(x_0)\big) - \frac{dF}{du}\big(u_0(x_0)\big)\right| > \epsilon$;
- $x_0 - \Delta + \frac{dF}{du}\big(u_0(x_0 - \Delta)\big)T = x_0 + \frac{d\tilde{F}}{du}\big(u_0\big)T \equiv x^*$.

Without loss of generality, we assume $\frac{dF}{du} > 0$ and $\frac{d\tilde{F}}{du} > 0$ for $\big\{u\big|u = u_0(x)\,,\ x \in [x_0 - \Delta, x_0]\big\}$. Using the method of lines, we have

$$u\left(T\,,\ x_0 - \Delta + \frac{dF}{du}\big(u_0(x_0 - \Delta)\big)T\right) = u_0(x_0 - \Delta)\,,$$

and

$$\tilde{u}\left(T\,,\ x_0 + \frac{d\tilde{F}}{du}\big(u_0(x_0)\big)T\right) = u_0(x_0)\,.$$

Therefore

$$|\tilde{u}(x^*,T) - u(x^*,T)| = |u_0(x_0) - u_0(x_0 - \Delta)| \geq \gamma\Delta \equiv \delta\,,$$

by using the definition of B_u.

Set $T = \frac{\Delta}{\left|\frac{d\tilde{F}}{du}\left(u_0(x_0-\Delta)\right)-\frac{dF}{du}\left(u_0(x_0)\right)\right|}$, we have

$$\begin{aligned}
&\left|\frac{d\tilde{F}}{du}\left(u_0(x_0-\Delta)\right)-\frac{dF}{du}\left(u_0(x_0)\right)\right| \\
=&\left|\frac{dF}{du}\left(u_0(x_0)\right)-\frac{d\tilde{F}}{du}\left(u_0(x_0)\right)+\frac{dF}{du}\left(u_0(x_0-\Delta)\right)-\frac{dF}{du}\left(u_0(x_0)\right)\right| \\
=&\left|\frac{dF}{du}\left(u_0(x_0)\right)-\frac{d\tilde{F}}{du}\left(u_0(x_0)\right)+\overline{\frac{d^2F}{du^2}}\left(u_0(x_0-\Delta)-u_0(x_0)\right)\right| \\
\geq&\left|\frac{dF}{du}\left(u_0(x_0)\right)-\frac{d\tilde{F}}{du}\left(u_0(x_0)\right)\right|-L_uL_F\Delta \\
\geq&\epsilon-L_uL_F\Delta\equiv\epsilon_F>0
\end{aligned}$$

by using the mean value theorem. Therefore $T\leq\frac{\Delta}{\epsilon_F}<\infty$. So we find a $\delta=\gamma\Delta$ and a $T<\infty$ that provides an exception to the contradiction of the theorem.

Secondly, if there is shock wave within $[0,T]$ for either (2.6) or (2.7), we let T^* be the time of the shock occurrence. Without loss of generality, assume the shock occurs for (2.6) first. The shock implies the intersection of two characteristic lines. Choose a $\Delta>0$ such that $\left|\frac{dF}{du}\left(u_0(x)\right)-\frac{dF}{du}\left(u_0(x-\Delta)\right)\right|T^*=\Delta$. Using the mean value theorem, we have

$$T^*=\frac{\Delta}{\overline{\frac{d^2F}{du^2}}\left(u_0(x)-u_0(x-\Delta)\right)}\geq\frac{1}{L_uL_F}$$

Thus, if we choose

$$T=\min\left\{\frac{1}{L_uL_K},\frac{\Delta}{\epsilon_\Delta}\right\},$$

no shock occurs in $t\in[0,T]$. Since the theorem is already proven for the no-shock scenario, the proof completes. ■

In this theorem, B_u consists of the value of u that appear in the initial condition $u_0(x)$. Also, on such value of u, the initial condition must satisfy $\left|\frac{du_0}{dx}\right|\geq\gamma>0$. An

example of B_u is given in Figure 2-1. The initial condition u_0 and its derivative $\frac{du_0}{dx}$ are indicated by the solid blue and the green dashed lines. Given the value of γ, B_u is shown on the right vertical axis which consists of values of u that appear in u_0 and satisfy $\left|\frac{du_0}{dx}\right| \geq \gamma > 0$.

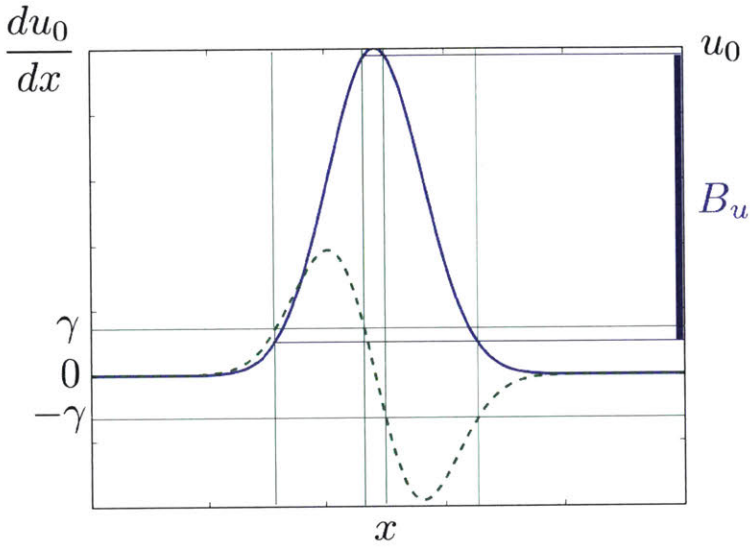


Figure 2-1: An illustration of B_u defined in Theorem 1. The blue line is u_0 and the green dashed line is $\frac{du_0}{dx}$. B_u is the set of u_0 where the derivative $\frac{du_0}{dx}$ has an absolute value larger than γ. The left y-axis is $\frac{du_0}{dx}$, and the right y-axis is u_0. B_u, represented by the bold blue line on the right y-axis, is the domain of u in which the error of the inferred flux can be bounded by the solution mismatch.

Several observations can be made from Theorem 1. First, if the solutions of (2.6) and (2.7) match closely, i.e. $|\tilde{u}(t,x) - u(t,x)| < \delta$, then the derivatives of their flux functions must match closely in B_u, i.e. $\left|\frac{d\tilde{F}}{du} - \frac{dF}{du}\right| < \epsilon$. Second, only the derivatives of the fluxes are guaranteed to match, i.e., $\left|\frac{d\tilde{F}}{du} - \frac{dF}{du}\right| < \epsilon$, rather than the fluxes themselves. If F or $\tilde{F}$ is added by a constant, the solution of the gray-box or the twin-model will not change. Third, the conclusion can only be drawn for values of u which appeared in the initial condition ($u \in \{u_0(x) \text{ for all } x \in \mathbb{R}\}$), and where the initial condition has large enough slope $\left(\left|\frac{du_0}{dx}\right| \geq \gamma > 0\right)$. Generally speaking, it is reasonable to expect that F is only inferable (up to a constant) in the domain of

u covered by the gray-box solution, which will be demonstrated in the numerical examples in this chapter.

If the twin model uses implicit time marching schemes, the minimization of $\mathcal{M}_u$ can be expensive because the computation of $\mathcal{M}_u$ requires solving a system of equations at every time step [101]. To reduce the computational cost, we introduce another metric: the integrated truncation error. Define

$$\tau = \frac{\partial u}{\partial t} + \nabla \cdot \left(D\tilde{F}(u)\right) - q(u, c)\,, \tag{2.8}$$

to be the residual of (2.1) by replacing $\tilde{u}$ with u. Let $\boldsymbol{\tau}$ be the discretized residual obtained by plugging the discretized gray-box solution into the twin-model simulator. The integrated truncation error is defined to be

$$\mathcal{M}_\tau(\tilde{F}) = \sum_{i=1}^{M}\sum_{j=1}^{N} w_{ij}\left\|\boldsymbol{\tau}_{ij}\right\|^2, \tag{2.9}$$

which approximates

$$\int_0^T \int_\Omega \|\tau\|^2 \,\mathrm{d}x\mathrm{d}t\,. \tag{2.10}$$

In (2.9), i, j, w_{ij} are defined in the same way as in (2.2).

We study the relationship between $\mathcal{M}_\tau$ and $\mathcal{M}_u$. A sufficient condition is studied under which $\mathcal{M}_u$ can be bounded by $\mathcal{M}_\tau$.

Theorem 2. *Consider a scalar-state gray-box model whose one-step time marching is*

$$\mathcal{H}_i : \mathbb{R}^N \mapsto \mathbb{R}^N,\ \boldsymbol{u}_{i\cdot} \to \boldsymbol{u}_{i+1\cdot} = \mathcal{H}_i \boldsymbol{u}_{i\cdot}\,, \quad i = 1, \cdots, M-1\,, \tag{2.11}$$

and a twin model whose one-step time marching is

$$\mathcal{G}_i : \mathbb{R}^N \mapsto \mathbb{R}^N,\ \tilde{\boldsymbol{u}}_{i\cdot} \to \tilde{\boldsymbol{u}}_{i+1\cdot} = \mathcal{G}_i \tilde{\boldsymbol{u}}_{i\cdot}\,, \quad i = 1, \cdots, M-1\,. \tag{2.12}$$

The solution mismatch is defined by (2.2)

$$\mathcal{M}_u(\tilde{F}) = \sum_{i=1}^{M}\sum_{j=1}^{N} w_{ij}\left(\tilde{\boldsymbol{u}}_{ij} - \boldsymbol{u}_{ij}\right)^2 .$$

Assume $\Delta t = t_2 - t_1 = \cdots = t_M - t_{M-1}$ *and* $w_{ij} = w_j$ *for all* i, j. *The integrated truncation error is defined by* (2.9),

$$\mathcal{M}_\tau(\tilde{F}) = \sum_{i=1}^{M}\sum_{j=1}^{N} w_{ij}\boldsymbol{\tau}_{ij}^2 ,$$

where

$$\boldsymbol{\tau}_{ij} = \frac{1}{\Delta t}\left(\boldsymbol{u}_{i+1\,j} - (\mathcal{G}\boldsymbol{u}_{i\cdot})_j\right) . \tag{2.13}$$

If $\mathcal{G}_i$ *satisfies*

$$\|\mathcal{G}_i a - \mathcal{G}_i b\|_W^2 \le \beta\|a - b\|_W^2 , \tag{2.14}$$

for any $a, b \in \mathbb{R}^N$ *and for all* i, *then*

$$\mathcal{M}_u \le \left(1 + \beta + \cdots + \beta^{M-1}\right)\Delta t^2 \mathcal{M}_\tau , \tag{2.15}$$

where

$$\|v\|_W^2 \equiv v^T \begin{pmatrix} w_1 & & \\ & \ddots & \\ & & w_N \end{pmatrix} v \tag{2.16}$$

for any $v \in \mathbb{R}^N$.

Proof:

The integrated truncation error can be written as

$$\begin{aligned}
\mathcal{M}_\tau(\tilde{F}) &= \sum_{i=1}^{M}\sum_{j=1}^{N} w_j \frac{1}{\Delta t^2}\left(\boldsymbol{u}_{i+1\,j} - (\mathcal{G}\boldsymbol{u}_{i\cdot})_j\right)^2 \\
&= \sum_{i=1}^{M} \frac{1}{\Delta t^2}(u_{i+1\cdot} - \mathcal{G}u_{i\cdot})^T W (u_{i+1\cdot} - \mathcal{G}u_{i\cdot}) \\
&= \sum_{i=1}^{M} \frac{1}{\Delta t^2}\left\|u_{i+1\cdot} - \mathcal{G}u_{i\cdot}\right\|_W^2 \\
&= \sum_{i=1}^{M} \frac{1}{\Delta t^2}\left\|\mathcal{H}u_{i\cdot} - \mathcal{G}u_{i\cdot}\right\|_W^2 \\
&= \sum_{i=1}^{M} \frac{1}{\Delta t^2}\left\|\left(\mathcal{H}^i - \mathcal{G}\mathcal{H}^{i-1}\right)u_{0\cdot}\right\|_W^2 .
\end{aligned}$$

Similarly, the solution mismatch can be written as

$$\mathcal{M}_u(\tilde{F}) = \sum_{i=1}^{M}\left\|\left(\mathcal{H}^i - \mathcal{G}^i\right)u_{0\cdot}\right\|_W^2$$

Fig 2-2 gives an explanation of $\mathcal{M}_u$ and $\mathcal{M}_\tau$ by viewing the simulators as discrete time dynamical systems.

Using the equality

$$\mathcal{G}^i - \mathcal{H}^i = (\mathcal{G}^i - \mathcal{G}^{i-1}\mathcal{H}) + (\mathcal{G}^{i-1}\mathcal{H} - \mathcal{G}^{i-2}\mathcal{H}^2) + \cdots + (\mathcal{G}\mathcal{H}^{i-1} - \mathcal{H}^i), \quad i \in \mathbb{N},$$

and triangular inequality, we have

$$\mathcal{M}_u \leq \left\{ \begin{array}{lll}
\|(\mathcal{G}^{M-1}\mathcal{G} - \mathcal{G}^{M-1}\mathcal{H})u_{0\cdot}\|_W^2 + \|(\mathcal{G}^{M-2}\mathcal{G}\mathcal{H} - \mathcal{G}^{M-2}\mathcal{H}^2)u_{0\cdot}\|_W^2 & + \cdots + \|(\mathcal{G}\mathcal{H}^{M-1} - \mathcal{H}^M)u_{0\cdot}\|_W^2 \\
+ \|(\mathcal{G}^{M-2}\mathcal{G} - \mathcal{G}^{M-2}\mathcal{H})u_{0\cdot}\|_W^2 & + \cdots + \|(\mathcal{G}\mathcal{H}^{M-2} - \mathcal{H}^{M-1})u_{0\cdot}\|_W^2 \\
\ddots & \vdots \\
 & + \|(\mathcal{G} - \mathcal{H})u_{0\cdot}\|_W^2
\end{array} \right\}.$$

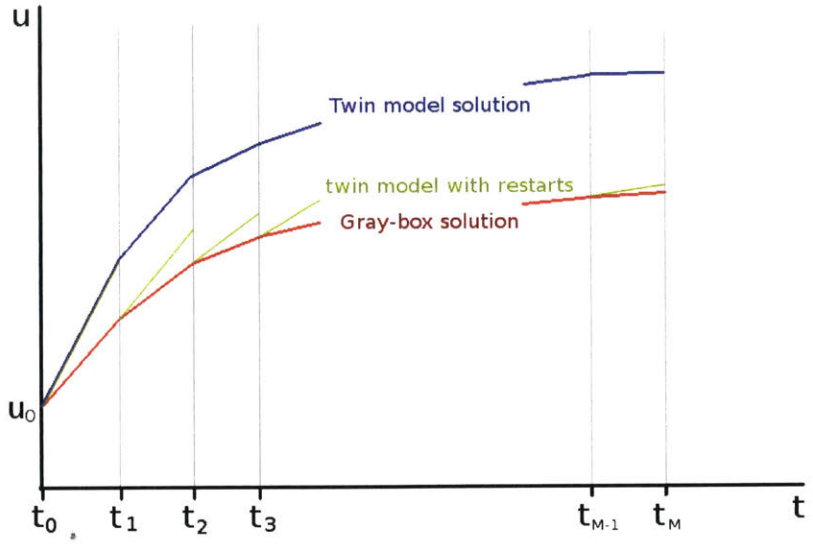


Figure 2-2: State-space trajectories of the gray-box model and the twin model. $\mathcal{M}_u$ measures the difference of the twin model trajectory (blue) with the gray-box trajectory (red). $\mathcal{M}_\tau$ measures the difference of the twin model trajectory with restarts (green) and the gray-box trajectory (red).

Therefore,

$$\mathcal{M}_u - \Delta t^2 \mathcal{M}_\tau \leq$$

$$\left\{ \begin{array}{lll} \|(\mathcal{G}^{M-1}\mathcal{G} - \mathcal{G}^{M-1}\mathcal{H})u_{0\cdot}\|_W^2 + \|(\mathcal{G}^{M-2}\mathcal{G}\mathcal{H} - \mathcal{G}^{M-2}\mathcal{H}^2)u_{0\cdot}\|_W^2 & + \cdots + \|(\mathcal{G}\mathcal{G}\mathcal{H}^{M-2} - \mathcal{G}\mathcal{H}^{M-1})u_{0\cdot}\|_W^2 \\ + \|(\mathcal{G}^{M-2}\mathcal{G} - \mathcal{G}^{M-2}\mathcal{H})u_{0\cdot}\|_W^2 & + \cdots + \|(\mathcal{G}\mathcal{G}\mathcal{H}^{M-3} - \mathcal{G}\mathcal{H}^{M-2})u_{0\cdot}\|_W^2 \\ \ddots & \vdots \\ & + \|(\mathcal{G}\mathcal{G} - \mathcal{G}\mathcal{H})u_{0\cdot}\|_W^2 \end{array} \right\}.$$

Under the assumption

$$\|\mathcal{G}a - \mathcal{G}b\|_W^2 \leq \beta \|a - b\|_W^2 ,$$

and its implication

$$\left\|\mathcal{G}^i a - \mathcal{G}^i b\right\|_W^2 \leq \beta^i \|a - b\|_W^2 , \quad i \in \mathbb{N} ,$$

we have

$$\mathcal{M}_u - \Delta t^2 \mathcal{M}_\tau \leq \left\{ \begin{array}{lll} \beta^{M-1}\|(\mathcal{G}-\mathcal{H})u_{0\cdot}\|_W^2 + \beta^{M-2}\|(\mathcal{G}\mathcal{H}-\mathcal{H}^2)u_{0\cdot}\|_W^2 & +\cdots+\beta\|(\mathcal{G}\mathcal{H}^{M-2}-\mathcal{H}^{M-1})u_{0\cdot}\|_W^2 \\ \qquad + \beta^{M-2}\|(\mathcal{G}-\mathcal{H})u_{0\cdot}\|_{M-1}^2 & +\cdots+\beta\|(\mathcal{G}\mathcal{H}^{n-3}-\mathcal{H}^{n-2})u_{0\cdot}\|_W^2 \\ \qquad \ddots & \vdots \\ & +\beta\|(\mathcal{G}-\mathcal{H})u_{0\cdot}\|_W^2 \end{array} \right\}.$$

Reorder the summation, we get

$$\mathcal{M}_u - \Delta t^2 \mathcal{M}_\tau \leq \left\{ \begin{array}{lll} \beta^{M-1}\|(\mathcal{G}-\mathcal{H})u_{0\cdot}\|_W^2 + \beta^{M-2}\|(\mathcal{G}-\mathcal{H})u_{0\cdot}\|_W^2 & +\cdots+\beta\|(\mathcal{G}-\mathcal{H})u_{0\cdot}\|_W^2 \\ \qquad + \beta^{M-2}\|(\mathcal{G}\mathcal{H}-\mathcal{H}^2)u_{0\cdot}\|_W^2 & +\cdots+\beta\|(\mathcal{G}\mathcal{H}-\mathcal{H}^2)u_{0\cdot}\|_W^2 \\ \qquad \ddots & \vdots \\ & +\beta\|(\mathcal{G}\mathcal{H}^{M-2}-\mathcal{H}^{M-1})u_{0\cdot}\|_W^2 \end{array} \right\}.$$

Therefore,

$$\mathcal{M}_u - \Delta t^2 \mathcal{M}_\tau \leq \left(\beta^{M-1} + \beta^{M-2} + \cdots + \beta\right) \Delta t^2 \mathcal{M}_\tau$$

If β is strictly less than 1, then

$$\mathcal{M}_u \leq \frac{\Delta t^2}{1-\beta} \mathcal{M}_\tau ,$$

thus completes the proof. ■

The theorem implies that, if the one-step time marching operator of the twin model is Lipschitz continuous, as given by (2.14), then the solution mismatch can be bounded by the integrated truncation error. Unfortunately, if the Lipschitz constant $\beta > 1$ and if the number of time steps $M \gg 1$, then $\left(1 + \beta + \cdots + \beta^{M-1}\right)$ can be large. Thus a small $\mathcal{M}_\tau$ does not always guarantee a small $\mathcal{M}_u$. Therefore, for twin models that have $\beta > 1$ and $M \gg 1$, if computational budget allows, we recommend minimizing $\mathcal{M}_u$ instead of $\mathcal{M}_u$ for training a twin model. Despite the theoretical flaw, $\mathcal{M}_\tau$ can be useful in practice when minimizing $\mathcal{M}_u$ is too computationally expensive.

Let $\mathcal{M}$ denote either $\mathcal{M}_u$ or $\mathcal{M}_\tau$. The minimization of $\mathcal{M}$ can be solved by

gradient-based methods. For $\mathcal{M} = \mathcal{M}_\tau$, the adjoint method can be applied to compute $\frac{d\mathcal{M}_\tau}{d\tilde{F}}$ to drive the optimization. For $\mathcal{M} = \mathcal{M}_u$, the adjoint method can be applied to compute the gradient of $\tilde{\boldsymbol{u}}$ with respect to $\tilde{F}$. Therefore, the gradient of $\mathcal{M}$ with respect to $\tilde{F}$ can be obtained through (2.2) according to

$$\frac{d\mathcal{M}}{d\tilde{F}} = \frac{d\mathcal{M}}{d\tilde{\boldsymbol{u}}} \frac{d\tilde{\boldsymbol{u}}}{d\tilde{F}}. \tag{2.17}$$

The remainder of this chapter is organized as follows. Section 2.2 discusses the choices of the function space $\mathcal{S}_F$ in (2.4). A choice of the basis functions, the sigmoid functions, is introduced to parameterize $\tilde{F}$. By using a fixed set of basis functions, the twin model is demonstrated in a numerical example. Section 2.3 develops an algorithm that adaptively constructs the basis functions. Section 2.4 demonstrates the algorithm in several numerical examples. Finally, Section 2.5 summarizes the chapter.

2.2 Choice of Basis Functions

As discussed in Section 1.3.3, F can be parameterized by a linear combination of basis functions. First, consider the case when $\tilde{F}$ is univariate. There are many types of basis functions to parameterize a univariate function, such as polynomial basis, Fourier basis, and wavelet basis [102]. Based on the observations from Theorem 1, $\tilde{F}$ and F are expected to match only on a domain of u where the gray-box space-time solution appears and has large enough slope. Therefore, an ideal parameterization should admit local refinements so $\tilde{F}$ can match F better at some domain locally. Another observation from Theorem 1 is that F can only be estimated up to a constant. This section presents a choice of the parameterization for $\tilde{F}$ that takes into account such considerations.

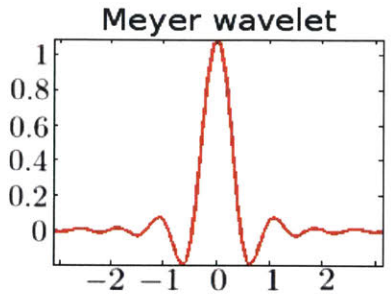


Figure 2-3: An example mother wavelet, the Meyer wavelet.

A parameterization that allows local refinements is the wavelet parameterization [102]. The wavelet is a set of basis functions developed for multiresolution analysis (MRA) [102]. MRA introduces an increasing sequence of closed function spaces $\{V_j\}_{j\in\mathbb{Z}}$,

$$\cdots \subset V_{-1} \subset V_0 \subset V_1 \subset \cdots ,$$

[102]. For univariate MRA, V_j's satisfy the following properties known as self-similarity [102]:

$$f(u) \in V_j \Leftrightarrow f(2u) \in V_{j+1},\ j \in \mathbb{Z}$$
$$f(u) \in V_j \Leftrightarrow f(u - \frac{\eta}{2^j}) \in V_j,\ j \in \mathbb{Z},\ \eta \in \{0, \pm 1, \pm 2, \cdots\} .$$

The function space V_j is spanned by a set of orthonormal bases called the wavelet [102]

$$\hat{\phi}_{j,\eta}(u) = 2^{j/2}\hat{\phi}(2^j u - \eta) , \quad \eta \in \{0, \pm 1, \pm 2, \cdots\} , \tag{2.18}$$

where $\hat{\phi}$ is called the mother wavelet. The equation (2.18) is called the self-similar property because any basis $\hat{\phi}_{j,\eta}$ can be obtained through a translation and a dilation of the mother wavelet $\hat{\phi}$, where j is called the dilation parameter and η is called the translation parameter. An example mother wavelet, the Meyer wavelet, is shown in Figure 2-3.

As discussed at the beginning of this chapter, only the derivative of F, rather

than F itself, can be inferred. If $\frac{d\tilde{F}}{du}$ is parameterized by the wavelet bases, $\tilde{F}$ shall be parameterized by the indefinite integrals of the wavelets, i.e.

$$\phi_{j,\eta}(u) = \int_{-\infty}^{u} \hat{\phi}_{j,\eta}(u')du' \,. \tag{2.19}$$

$\phi_{j,\eta}$'s are sigmoid functions, which satisfy

$$\frac{d\phi_{j,\eta}}{du} = \hat{\phi}\,, \tag{2.20}$$

and

$$\phi_{j,\eta}(u) = \begin{cases} 0, \, u \to -\infty \\ 1, \, u \to \infty \end{cases} \tag{2.21}$$

due to the normality of the wavelet.

Let

$$\phi(u) = \int_{-\infty}^{u} \hat{\phi}(u')du'\,, \tag{2.22}$$

then

$$\phi(2^j u - \eta) = \int_{-\infty}^{2^j u - \eta} \hat{\phi}(u')du' = \int_{-\infty}^{u} \hat{\phi}(2^j u' - \eta)du' = \int_{-\infty}^{u} \hat{\phi}_{j,\eta}(u')du' \tag{2.23}$$

(2.19) and (2.23) show that $\phi_{j,\eta}$ satisfies the self-similarity property

$$\phi_{j,\eta}(u) = \phi(2^j u - \eta)\,, \quad j \in \mathbb{Z}\,,\ \eta \in \mathbb{Z}\,, \tag{2.24}$$

where ϕ is called the mother sigmoid.

There are many choices of sigmoid functions for ϕ. My thesis will use the logistic sigmoid function as the mother sigmoid,

$$\phi(u) = \frac{1}{1 + e^{-u}}\,. \tag{2.25}$$

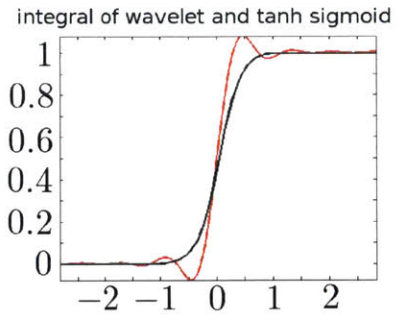


Figure 2-4: Red line: the integral (2.19) *of the Meyer wavelet. Black line: the logistic sigmoid function.*

If $\tilde{F}$ is univariate, the logistic sigmoids $\phi_{j,\eta}$'s, for $j \in \mathbb{Z}$ and $\eta \in \mathbb{Z}$, are used as the bases. If $\tilde{F}$ is k-variate, the basis can be formed by the tensor product of the univariate basis [105] ($\phi_{j_1,\eta_1}, \cdots, \phi_{j_k,\eta_k}$, for $j_1 \in \mathbb{Z}, \eta_1 \in \mathbb{Z}, \cdots, j_k \in \mathbb{Z}, \eta_k \in \mathbb{Z}$). In other words, the basis can be

$$\phi_{\boldsymbol{j},\boldsymbol{\eta}}(u_1, \cdots, u_k) = \phi_{j_1,\eta_1}(u_1) \cdots \phi_{j_k,\eta_k}(u_k)\,, \tag{2.26}$$

where $\boldsymbol{j} = (j_1, \cdots, j_k) \in \mathbb{Z}^k$, $\boldsymbol{\eta} = (\eta_1, \cdots, \eta_k) \in \mathbb{Z}^k$. To sum up, $\tilde{F}$ can be expressed by

$$\tilde{F} = \sum_{\boldsymbol{j} \in \mathbb{Z}^k, \boldsymbol{\eta} \in \mathbb{Z}^k} \alpha_{\boldsymbol{j},\boldsymbol{\eta}} \phi_{\boldsymbol{j},\boldsymbol{\eta}}\,, \tag{2.27}$$

where α's are the coefficients of the bases.

A compact representation of the sigmoid bases is introduced. A univariate basis function,

$$\phi_{j,\eta}(u) = \phi(2^j u - \eta)\,, \quad j \in \mathbb{Z}\,,\ \eta \in \mathbb{Z}\,,$$

can be represented by a tuple (j, η), where j is the dilation parameter, and $\frac{\eta}{2^j}$ is the center of the basis. Similarly, a k-variate basis function, $\phi_{\boldsymbol{j},\boldsymbol{\eta}}$ in (2.26), can be represented by a tuple $(\boldsymbol{j}, \boldsymbol{\eta}) = ((j_1, \cdots, j_k), (\eta_1, \cdots, \eta_k))$. Thus, a sigmoid function can be visualized by a point in a $2k$-dimensional space, which is illustrated in Figure 2-5a through 2-5d for the univariate case.

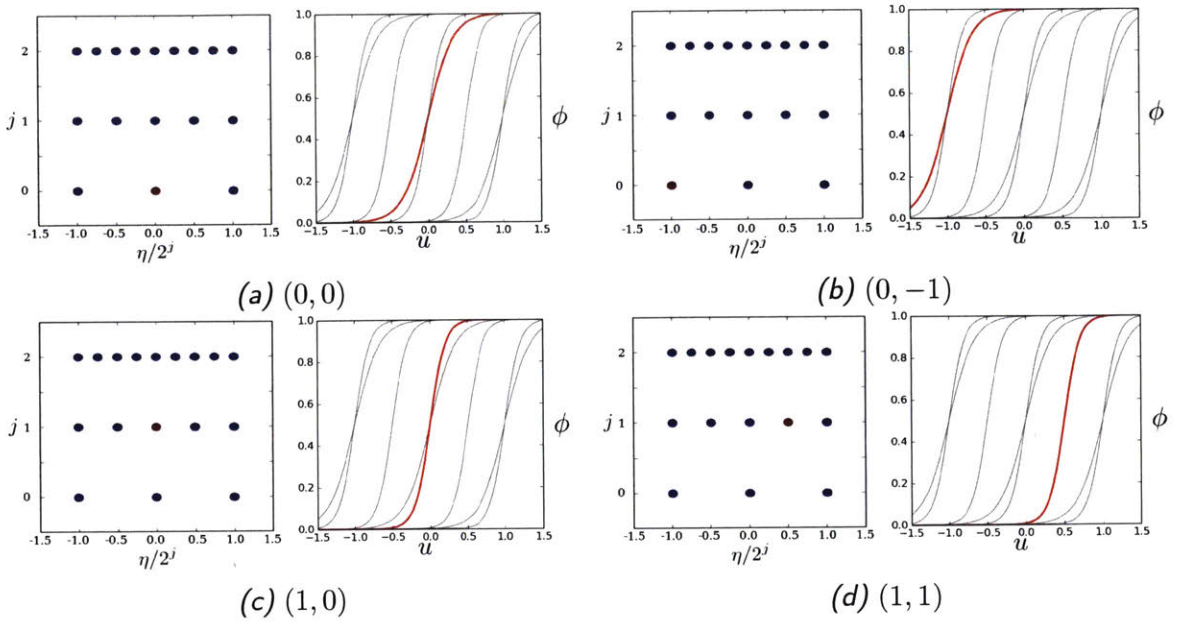


Figure 2-5: An illustration of the tuple representation of univariate sigmoid functions.

There are an infinite number of bases involved in this expression, making it infeasible to be implemented on the computer. To address this issue, a systematic procedure for choosing a suitable subset of the bases will be presented in Section 2.3.

In the remaining part of the section, a numerical example is given to illustrate the inference of F by using the sigmoid parameterization. Consider a gray-box model solving the 1-D Buckley-Leverett equation [3]

$$\frac{\partial u}{\partial t} + \frac{\partial}{\partial x}\Big(\underbrace{\frac{u^2}{1+2(1-u)^2}}_{F}\Big) = c\,, \tag{2.28}$$

with the initial condition $u(0,x) = u_0(x)$ and the periodic boundary condition $u(t,0) = u(t,1)$. c is a constant control variable. The Buckley-Leverett equation models the two-phase porous media flow where u stands for the saturation of one phase, and $1-u$ stands for the saturation of another phase. Therefore $0 \leq u_0(x) \leq 1$ for all $x \in [0,1]$. $c \in \mathbb{R}$ is a constant-valued control. F is assumed unknown and is inferred by a twin

model. The twin model solves

$$\frac{\partial \tilde{u}}{\partial t} + \frac{\partial}{\partial x}\tilde{F}(\tilde{u}) = c\,, \tag{2.29}$$

with the same c and the same initial and boundary conditions. A second-order finite volume scheme is used to simulate both the gray-box model (2.28) and the twin models (2.29). Parameterize $\tilde{F}$ by the sigmoid bases (2.2)

$$\tilde{F} = \sum_{(j,\eta)\in\mathcal{A}\subset\mathbb{Z}\times\mathbb{Z}} \alpha_{j,\eta}\phi_{j,\eta}\,, \tag{2.30}$$

where $\mathcal{A}$ is a finite set that contains the tuples representing the basis functions. (2.30) differs from (2.27) in that a finite number of basis functions are used so the parameterization can be implemented on the computer. In this example, the set $\mathcal{A}$ is chosen manually, such that the Buckley-Leverett flux can be well approximated. The chosen basis are (j,η) for $j = 3$, $\eta = 0, 1, \cdots, 8$, which are shown in Figure 2-6. The topic of how to algorithmically choose a suitable set of basis will be discussed in Section 2.3.

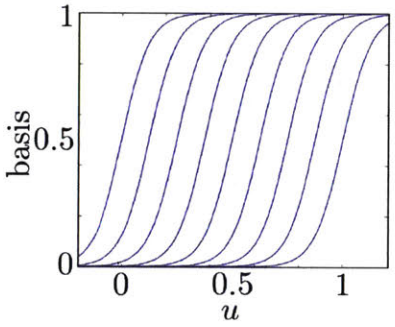


Figure 2-6: The bases chosen manually for the numerical example of Buckley-Leverett equation.

The twin model is trained to minimize $\mathcal{M}_u$. To avoid overfitting in (2.4), we consider applying an L_1 regularization on $\boldsymbol{\alpha}$. In other words, F is inferred by solving

the following minimization problem,

$$\boldsymbol{\alpha}^* = \underset{\alpha_{j,\eta}\in\mathbb{R}}{\operatorname{argmin}} \left(\sum_{i=1}^{M}\sum_{j=1}^{N} w_{ij}\left(\tilde{\boldsymbol{u}}_{ij} - \boldsymbol{u}_{ij}\right)^2 + \lambda \left\|\boldsymbol{\alpha}\right\|_{L_1} \right) , \tag{2.31}$$

where $\boldsymbol{\alpha} = \{\alpha_{j,k}\}_{(j,k)\in\mathcal{A}}$, $\|\cdot\|_{L_1}$ is the L_1 norm, and $\tilde{\boldsymbol{u}}$ is the twin-model space-time solution that depends on the value of $\boldsymbol{\alpha}$. $\lambda > 0$ is a tunable parameter for the L_1 regularization. As the value of λ increases, more entries in $\boldsymbol{\alpha}$ will be suppressed to zero [44].

The value of λ should be determined by maximizing the out-of-sample fit, such as the k-fold cross validation [38]. Given a basis dictionary, the k-fold cross validation proceeds in the following three steps: In the first step, the gray-box solution $\boldsymbol{u}$ is shuffled randomly into k disjoint sets $\{\boldsymbol{u}_1, \boldsymbol{u}_2, \cdots, \boldsymbol{u}_k\}$. An illustration for $k = 3$ is shown in Figure 2-7.

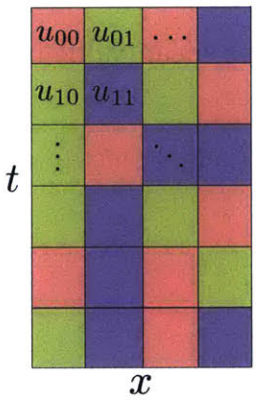


Figure 2-7: The discretized gray-box solution is shuffled into three sets, each indicated by a color. Each block stands for the state variable on a space-time grid point.

In the second step, k twin models are trained so that their space-time solutions match all but one sets of the gray-box solutions, as shown in (2.32), where T_i indicates

the ith twin model.

$$\begin{aligned} T_1 &= \texttt{TrainTwinModel}(\boldsymbol{u}_2, \boldsymbol{u}_3, \cdots, \boldsymbol{u}_k) \\ T_2 &= \texttt{TrainTwinModel}(\boldsymbol{u}_1, \boldsymbol{u}_3, \cdots, \boldsymbol{u}_k) \\ &\cdots \\ T_k &= \texttt{TrainTwinModel}(\boldsymbol{u}_1, \boldsymbol{u}_2, \cdots, \boldsymbol{u}_{k-1}) \end{aligned} , \tag{2.32}$$

where each equation requires solving (2.31).

In the third step, each trained twin model is validated by computing the solution mismatch on the remaining set of the gray-box solution, as shown in (2.33).

$$\begin{aligned} \mathcal{M}_u^1 &= \texttt{SolutionMismatch}\,(T_1, \boldsymbol{u}_1) \\ \mathcal{M}_u^2 &= \texttt{SolutionMismatch}\,(T_2, \boldsymbol{u}_2) \\ &\cdots \\ \mathcal{M}_u^k &= \texttt{SolutionMismatch}\,(T_k, \boldsymbol{u}_k) \end{aligned} \tag{2.33}$$

λ should be chosen to minimize the mean value of validation errors

$$\overline{\mathcal{M}}_u = \frac{1}{k}\left(\mathcal{M}_u^1 + \mathcal{M}_u^2 + \cdots + \mathcal{M}_u^k\right) . \tag{2.34}$$

(2.31) is solved by the L-BFGS method [42], using the NLopt package [43]. An example of training the twin model is shown in Figure 2-8. Figure 2-8 (a) shows the gray-box solution used to train the twin model. Figure 2-8 (b) shows the trained twin model solution by using the same initial condition. Figure 2-9 shows the gray-box flux F and the trained flux $\tilde{F}$, as well as $\frac{dF}{du}$ and $\frac{d\tilde{F}}{du}$.

In addition, the trained twin model is simulated using out-of-sample initial conditions which are different from the initial condition of the training solution. Two example

gray-box and twin-model solutions are shown in Figures 2-10 and 2-11. The solutions use the same second-order finite volume scheme and the Crank-Nicolson time marching scheme. In Figure 2-10, the domain of the gray-box solution is $[0.1, 0.3]$, which is contained in the domain of the training solution, $[0, 0.48]$. Therefore, it is reasonable to expect that the gray-box and the twin-model solutions match closely. In contrast, the domain of the gray-box solution in Figure 2-11 is $[0.05, 0.9]$. The domain is not contained in the domain of the training solution, and a larger solution mismatch is observed.

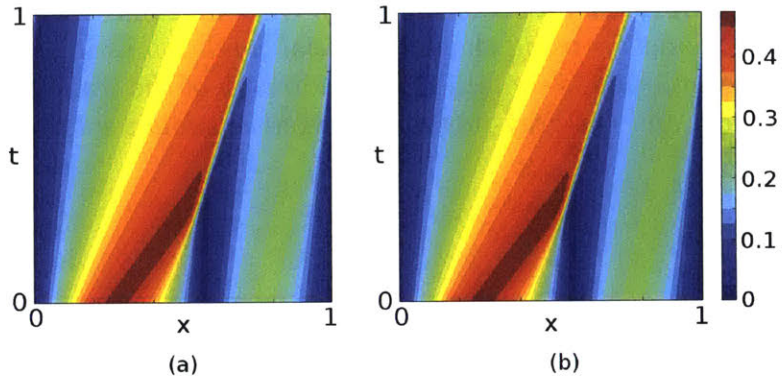


Figure 2-8: (a) Gray-box solution used to train the twin model. (b) Trained twin-model solution by using the same initial condition as in the gray-box solution.

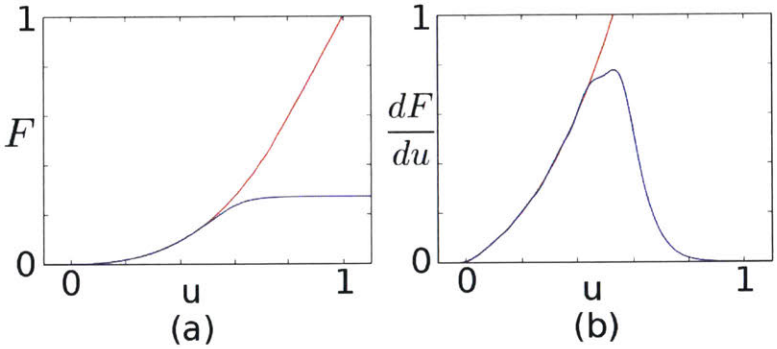


Figure 2-9: (a) Gray-box model's flux F (red) and the trained twin-model flux $\tilde{F}$ (blue). (b) $\frac{dF}{du}$ (red) and $\frac{d\tilde{F}}{du}$ (blue)

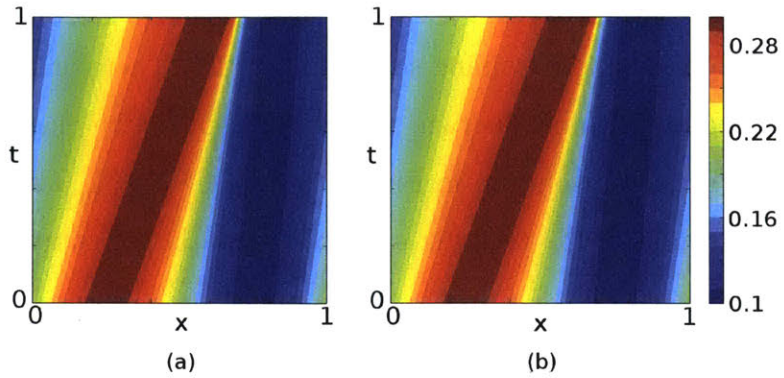


Figure 2-10: (a) Gray-box solution. (b) Out-of-sample solution of the trained twin model by using the same initial condition as in (a). Because the domain of solution is contained in the domain of the training solution, the twin model and the gray-box model produce similar solutions.

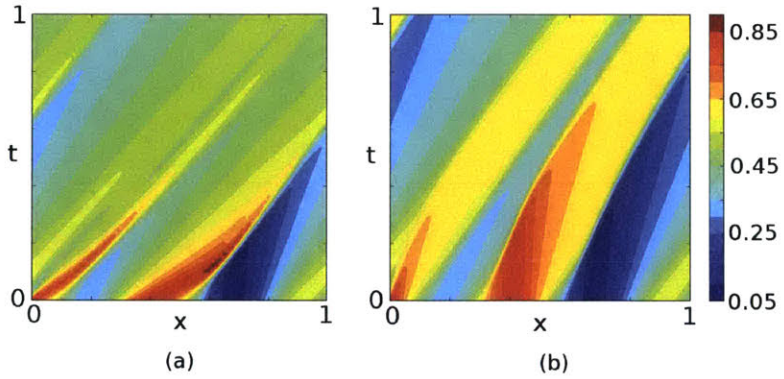


Figure 2-11: (a) Gray-box solution. (b) Out-of-sample solution of the trained twin model by using the same initial condition as in (a) . Because the domain of solution is beyond the domain of the training solution, a large deviation of the twin-model and gray-box solutions is observed.

After training the twin model, the adjoint method can be applied to the twin model to obtain the gradient of an objective function ξ to c. The gradient $\frac{d\xi(\tilde{\boldsymbol{u}},c)}{dc}$ approximates $\frac{d\xi(\boldsymbol{u},c)}{dc}$ for the value of c on which the twin model is trained. Consider the objective function

$$\xi(c) \equiv \int_{x=0}^{1} \left(u(1,x;c) - \frac{1}{2} \right)^2 \mathrm{d}x \,. \tag{2.35}$$

Figure 2-12 shows the objective function, evaluated using the gray-box model and the trained twin model, where the twin model is trained at $c = 0$ with the solution shown in Figure 2-8 (a). It is observed that the gradients of ξ match closely at $c = 0$ where the twin model is trained.

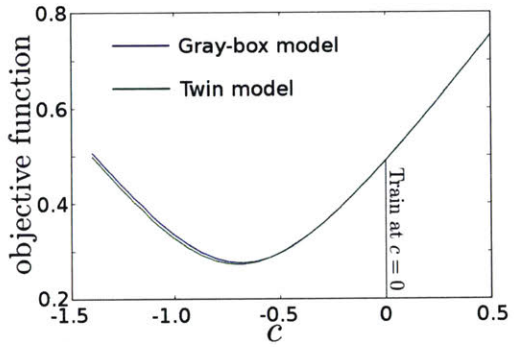


Figure 2-12: Objective function ξ evaluated by either the gray-box model and the trained twin model.

Because $\tilde{F}$ is trained by the gray-box space-time solution, and because the gray-box space-time solution depends on the initial condition $u_0(x)$, it is expected that the trained $\tilde{F}$ depends on $u_0(x)$. Figure 2-13 shows the training results using the gray-box solutions of three different initial conditions at $c = 0$. Some observations can be made: 1) As expected, the inferred $\tilde{F}$ can differ from F by a constant, which can be observed by in Figure 2-13 (d), (e), and (f); 2) $\frac{d\tilde{F}}{du}$ matches $\frac{dF}{du}$ only in a domain of u where the solution appears, as indicated by the green areas in Figure 2-13 (g), (h), and (i); 3) $\frac{d\tilde{F}}{du}$ does not necessarily match $\frac{dF}{du}$ outside the green area; the issue can be seen clearly in Figure 2-13 (c), (f), and (i); 4) In some regions of u, the bases are too

coarse. The issue appears in Figure 2-13 (g), where $\frac{d\tilde{F}}{du}$ exhibits a wavy deviation from $\frac{dF}{du}$. At such regions of u, the basis dictionary may be enriched by additional bases to enable a more accurate approximation of F. Addressing these issues in a systematic way is crucial to the rigorous development of the twin-model method. This topic is discussed in the next section.

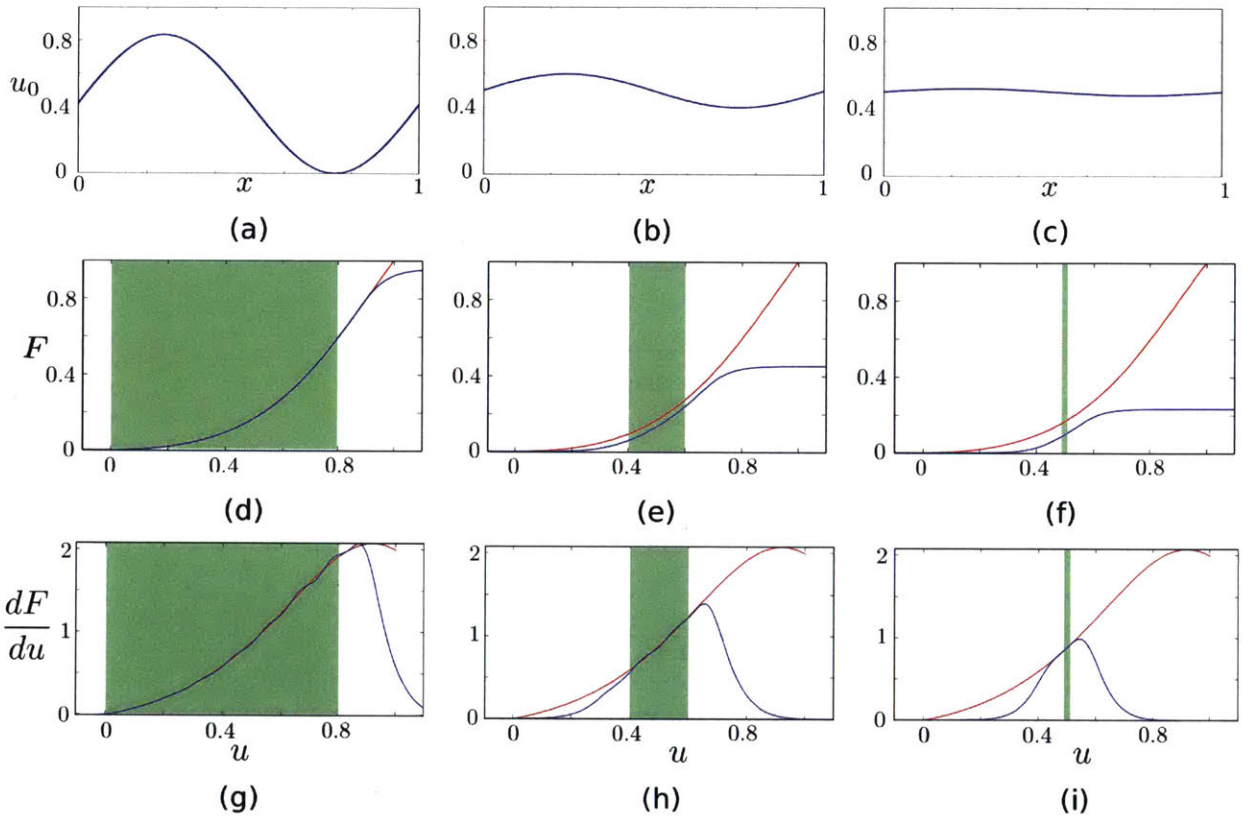


Figure 2-13: (a,b,c) Three different initial conditions used to generate the gray-box space-time solution. (d,e,f) compares the trained $\tilde{F}$ (blue) and the Buckley-Leverett F (red). (g,h,i) compares the trained $\frac{d\tilde{F}}{du}$ (blue) and the Buckley-Leverett $\frac{dF}{du}$ (red). The green background highlights the domain of u where the gray-box space-time solution appears.

2.3 Adaptive Basis Construction

This section addresses the problem of adaptively choosing a finite set of basis functions for the parameterization of $\tilde{F}$. Assume all candidate basis functions form a countable

set, Algorithm 1 outlines an iterative approach to constructing the basis dictionary. Starting from an initial set of basis functions, the basis dictionary is built up progressively by iterating over a forward step and a backward step [34, 35, 36]. The forward step searches over a candidate set of bases, and appends the most useful bases to the dictionary. The backward step searches over the current dictionary, and removes the unnecessary bases from the dictionary. The iteration stops only when no alteration is made to the dictionary or when a criterion, such as a targeted approximation accuracy, is achieved. My thesis applies this approach to the adaptive construction of the bases for the parameterization of $\tilde{F}$.

Input: Solution $\boldsymbol{u}$, Basis selection $\mathcal{A}$
1: **loop**
2: $\quad\mathcal{A}' \leftarrow$ Enrich $\mathcal{A}$ by the "best" basis.
3: $\quad$**if** $Q(\mathcal{A}') \succ Q(\mathcal{A})$ **then**
4: $\quad\quad\mathcal{A} \leftarrow \mathcal{A}'$
5: $\quad$**else**
6: $\quad\quad$**break**
7: $\quad$**end if**
8: $\quad\mathcal{A}' \leftarrow$ Delete the "worst" basis from $\mathcal{A}$
9: $\quad$**if** $Q(\mathcal{A}') \succ Q(\mathcal{A})$ **then**
10: $\quad\quad\mathcal{A} \leftarrow \mathcal{A}'$
11: $\quad$**end if**
12: **end loop**
13: $\boldsymbol{\alpha} \leftarrow \text{argmin}_{\boldsymbol{\alpha}}\, \mathcal{M}\left(\tilde{F}(\mathcal{A}, \boldsymbol{\alpha}), \boldsymbol{u}\right)$
Output: $\mathcal{A}$, $\boldsymbol{\alpha}$.

Algorithm 1: The outline of the algorithm for training a twin model with an adaptive basis. $\mathcal{A}$ indicates the basis dictionary. $\boldsymbol{\alpha}$ indicates the bases' coefficients. Starting from an initial dictionary, the algorithm iterates over the forward and the backward step to adaptively construct the dictionary and find the optimal coefficients. As explained in the previous section, the solution mismatch is a function that depends on u and $\tilde{F}$, where $\tilde{F}$ depends on the dictionary $\mathcal{A}$ and its coefficients α.

Some components of the algorithm require measuring how significant a basis is in training a twin model. Two criteria are needed. The first criterion determines **which** basis shall be chosen as the candidate basis to add to or delete from the dictionary, based on a metric of the significance of the basis. The second criterion

determines **whether** the current dictionary contains too few or too many bases, based on whether the approximation is sufficiently accurate. The two criteria are developed in this section.

To define the first criterion, a formulation is developed to efficiently assess the significance of a candidate basis. Given a basis dictionary $\mathcal{A}$, define the minimal mismatch

$$\mathcal{M}^*(\mathcal{A}) = \min_{\boldsymbol{\alpha}_\mathcal{A} \in \mathbb{R}^{|\mathcal{A}|}} \mathcal{M}\left(\sum_{(\boldsymbol{j},\boldsymbol{\eta})\in\mathcal{A}} \alpha_{\boldsymbol{j},\boldsymbol{\eta}}\phi_{\boldsymbol{j},\boldsymbol{\eta}}\right), \tag{2.36}$$

where $\mathcal{M}$ can be either the solution mismatch $\mathcal{M}_u$ or the integrated truncation error $\mathcal{M}_\tau$. Given the gray-box solution, the minimal mismatch is a function of $\mathcal{A}$. Let $\boldsymbol{\alpha}^*_\mathcal{A} = \{\alpha^*_{\boldsymbol{j},\boldsymbol{\eta}}\}_{(\boldsymbol{j},\boldsymbol{\eta})\in\mathcal{A}}$ be the optimal coefficients that solves (2.36), and let $\tilde{F}^*_\mathcal{A} = \sum_{(\boldsymbol{j},\boldsymbol{\eta})\in\mathcal{A}} \alpha^*_{\boldsymbol{j},\boldsymbol{\eta}}\phi_{\boldsymbol{j},\boldsymbol{\eta}}$. Consider appending $\mathcal{A}$ by an additional basis $l \equiv (\boldsymbol{j}_l, \boldsymbol{\eta}_l)$, and let $\mathcal{A}'$ be the appended basis dictionary. The minimal mismatch for the appended basis dictionary $\mathcal{A}'$ is

$$\mathcal{M}^*(\mathcal{A}') = \min_{\boldsymbol{\alpha}_{\mathcal{A}'} \in \mathbb{R}^{|\mathcal{A}|+1}} \mathcal{M}\left(\sum_{(\boldsymbol{j},\boldsymbol{\eta})\in\mathcal{A}'} \alpha_{\boldsymbol{j},\boldsymbol{\eta}}\phi_{\boldsymbol{j},\boldsymbol{\eta}}\right), \tag{2.37}$$

If the coefficients for the bases $\mathcal{A}'\backslash(\boldsymbol{j}_l, \boldsymbol{\eta}_l)$ are set to be $\boldsymbol{\alpha}^*_\mathcal{A}$ while the coefficient for the basis $(\boldsymbol{j}_l, \boldsymbol{\eta}_l)$ is set to be 0, then $\mathcal{M}(\mathcal{A}') = \mathcal{M}^*(\mathcal{A})$. Therefore, $\mathcal{M}^*(\mathcal{A}') \leq \mathcal{M}^*(\mathcal{A})$. The appension of an additional basis never increases the minimal mismatch.

Consider setting the coefficients of $\mathcal{A}'$ to be $\{\boldsymbol{\alpha}^*_\mathcal{A}, \epsilon\}$. For $\epsilon \to 0$, apply first-order approximation, we have

$$\begin{aligned} &\mathcal{M}\left(\sum_{(\boldsymbol{j},\boldsymbol{\eta})\in\mathcal{A}} \alpha^*_{\boldsymbol{j},\boldsymbol{\eta}}\phi_{\boldsymbol{j},\boldsymbol{\eta}}\right) - \mathcal{M}\left(\sum_{(\boldsymbol{j},\boldsymbol{\eta})\in\mathcal{A}} \alpha^*_{\boldsymbol{j},\boldsymbol{\eta}}\phi_{\boldsymbol{j},\boldsymbol{\eta}} + \epsilon\phi_l\right) \\ &\approx -\left(\int_{u\in\mathbb{R}^k} \left.\frac{d\mathcal{M}}{d\tilde{F}}\right|_{\tilde{F}^*_\mathcal{A}} \phi_l \, \mathrm{d}u\right)\epsilon\,. \end{aligned} \tag{2.38}$$

The absolute value of the coefficient for ϵ,

$$s_l(\mathcal{A}) \equiv \left| \int_{u \in \mathbb{R}^k} \left. \frac{d\mathcal{M}}{d\tilde{F}} \right|_{\tilde{F}^*_{\mathcal{A}}} \phi_l \, \mathrm{d}u \right| \tag{2.39}$$

is the rate of change of $\mathcal{M}$ by perturbing the coefficient of ϕ_l, which estimates the significance of the appended basis [37]. If there are multiple candidate bases, their significance can be sorted by (2.39).

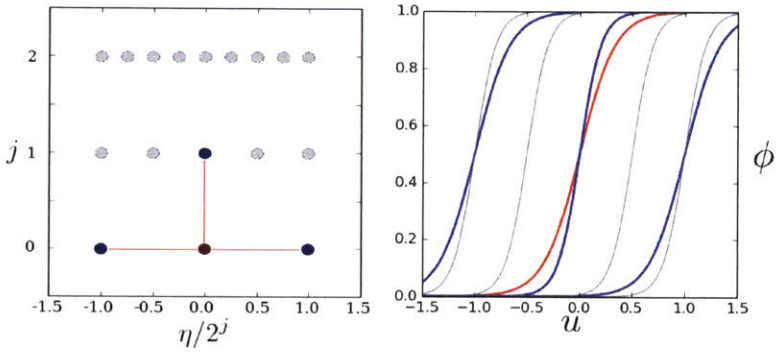
In practice, (2.39) is not computable for **all** the candidate bases $(\boldsymbol{j}, \boldsymbol{\eta})$ for $\boldsymbol{j} \in \mathbb{Z}^k$ and $\boldsymbol{\eta} \in \mathbb{Z}^k$, because the number of bases is infinite. Therefore, at every iteration in Algorithm 1, (2.39) shall only be evaluated on a finite number of bases. To address this issue, we define the neighborhood of a sigmoid basis. For univariate basis, the neighborhood of (j, η) is defined to be a set of the sigmoid bases

$$\mathcal{N}\left[(j, \eta)\right] = \left\{ (j+1, \eta) \, , \, (j, \eta \pm 1) \right\} \, . \tag{2.40}$$

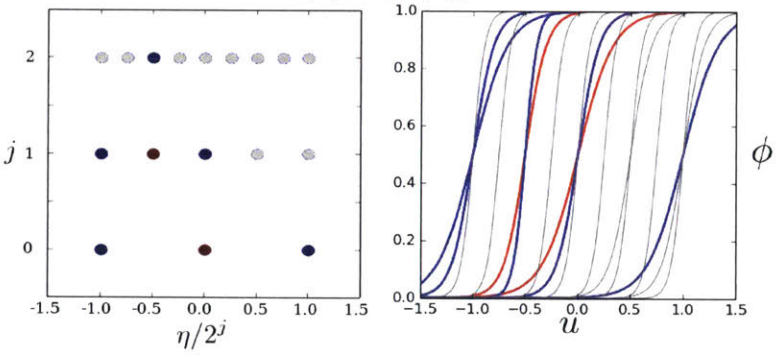
The neighborhood contains three basis functions: one basis $(j+1, \eta)$ whose dilation parameter is incremented by one; and two basis $(j, \eta \pm 1)$ whose dilation parameter keeps the same but the translation parameter is shifted by ± 1. For illustration, the neighborhood of $(0, 0)$ is shown in Figure 2-14a. The definition can be extended to the multivariate sigmoid. The neighborhood of a multivariate sigmoid is defined to be

$$\begin{aligned} &\mathcal{N}\left[(\boldsymbol{j}, \boldsymbol{\eta})\right] = \mathcal{N}\left[\left((j_1, \cdots, j_k), (\eta_1, \cdots, \eta_k)\right)\right] \\ =&\Big\{ \left((j_1+1, \cdots, j_k), (\eta_1, \cdots, \eta_k) \right), \cdots, \left((j_1, \cdots, j_k+1), (\eta_1, \cdots, \eta_k) \right), \\ &\left((j_1, \cdots, j_k), (\eta_1 \pm 1, \cdots, \eta_k) \right), \cdots, \left((j_1, \cdots, j_k), (\eta_1, \cdots, \eta_k \pm 1) \right) \Big\}, \end{aligned} \tag{2.41}$$

which consists of k bases whose dilation parameters are shifted by 1, and $2k$ bases whose translation parameters are shifted by ± 1. In addition, define the neighborhood of a set of sigmoid functions to be the union of the neighborhoods of all member



(a) $\mathcal{N}\left[(0,0)\right]$



(b) $\mathcal{N}\left[(0,0),(1,-1)\right]$

Figure 2-14: Neighborhood for univariate bases. (a) *Neighborhood (blue) of a single basis (red).* (b) *Neighborhood (blue) of several bases (red).*

bases minus the set itself, (2.42). The neighborhood of a set of sigmoid functions is illustrated in Figure 2-14b.

$$\begin{aligned}&\mathcal{N}\left[(\boldsymbol{j}_1,\boldsymbol{\eta}_1),\cdots,(\boldsymbol{j}_n,\boldsymbol{\eta}_n)\right]\\&=\left(\mathcal{N}\left[(\boldsymbol{j}_1,\boldsymbol{\eta}_1)\right]\bigcup\cdots\bigcup\mathcal{N}\left[(\boldsymbol{j}_n,\boldsymbol{\eta}_n)\right]\right)\Big\backslash\left\{(\boldsymbol{j}_1,\boldsymbol{\eta}_1),\cdots,(\boldsymbol{j}_n,\boldsymbol{\eta}_n)\right\}.\end{aligned}\tag{2.42}$$

Using the definition of the neighborhood and the significance metric, we can determine which basis to add and delete in the Algorithm 1. To add a basis, the basis significance, (2.39), is computed for all the bases in the neighborhood of the current dictionary. At each iteration, the basis with the largest significance will be considered for addition. Similarly, to delete a basis, the basis significance is computed for all the bases in the current dictionary. At each iteration, the basis with the smallest significance will be considered for deletion.

Another criterion is needed to determine whether a basis shall indeed be added or removed. To develop this criterion, the technique of k-fold cross validation can be applied. The k-fold cross validation is discussed in Section 2.2, where k twin models are trained and validated on randomly shuffled disjoint sets of the gray-box solution. The mean value of validation errors (2.34),

$$\overline{\mathcal{M}}=\frac{1}{k}\left(\mathcal{M}_1+\mathcal{M}_2+\cdots+\mathcal{M}_k\right)$$

can be used to measure the performance of the basis dictionary. A basis shall be added to or removed from the dictionary only if such action reduces $\overline{\mathcal{M}}$.

Based on the above developments, Algorithm 2 gives the details needed in Algorithm 1 to adaptively construct the basis dictionary. The main part of the algorithm is the forward-backward iteration that determines which and whether a basis is added or deleted in the dictionary, by using the metric $\mathcal{M}$ and the significance s. The

metric $\mathcal{M}$ and the significance s can be defined either according to the solution mismatch $\mathcal{M}_u$, or according to the integrated truncation error $\mathcal{M}_\tau$. We suggest $\mathcal{M}_\tau$ due to less computational cost when the twin model uses implicit numerical schemes. The algorithm starts from training a twin model using an arbitrary basis dictionary $\mathcal{A}$. The main part of the algorithm iterates over a forward step (line 3-9) and a backward step (line 11-19). The forward step first finds the most significant candidate in the neighborhood of the current dictionary according to (2.39). If the addition indeed reduces the cross-validation error, the candidate is added to the dictionary; otherwise, it is rejected. If the basis is added, the coefficients are updated by minimizing the solution mismatch, which can be implemented by the Broyden-Fletcher-Goldfarb-Shannon (BFGS) algorithm [40]. The backward step finds the most significant candidate in the current dictionary for deletion. If the deletion reduces the cross-validation error, the candidate is removed from the dictionary. If the basis is indeed deleted, the coefficients are updated by BFGS again. The iteration exits when the most significant addition no longer reduces the validation error. In the end, the coefficients are tuned to minimize the solution mismatch $\mathcal{M}_u$, which ensures that $\mathcal{M}_u$ is minimized. The output of the algorithm is the basis dictionary $\mathcal{A}$ and the coefficients $\boldsymbol{\alpha}_{\mathcal{A}}$.

Input: Initial basis dictionary $\mathcal{A}$, Validation error $\overline{\mathcal{M}}_0 = \infty$, Gray-box solution $\boldsymbol{u}$.

Minimize solution mismatch $\boldsymbol{\alpha}_{\mathcal{A}} \leftarrow \text{argmin}_{\boldsymbol{\alpha}} \mathcal{M}\left(\sum_{(\boldsymbol{j},\boldsymbol{\eta})\in\mathcal{A}} \alpha_{\boldsymbol{j},\boldsymbol{\eta}}\phi_{\boldsymbol{j},\boldsymbol{\eta}}\right)$
loop

$$l^* \leftarrow \underset{l\in\mathcal{N}(\mathcal{A})}{\text{argmax}}\, s_l(\mathcal{A}) \ , \quad \mathcal{A} \leftarrow \mathcal{A}\bigcup\{l^*\}$$

    Compute $\overline{\mathcal{M}}$ by k-fold cross validation.
    if $\overline{\mathcal{M}} < \overline{\mathcal{M}}_0$ **then**

$$\overline{\mathcal{M}}_0 \leftarrow \overline{\mathcal{M}} \ , \ \boldsymbol{\alpha}_{\mathcal{A}} \leftarrow \underset{\alpha}{\text{argmin}}\, \mathcal{M}\left(\sum_{(\boldsymbol{j},\boldsymbol{\eta})\in\mathcal{A}} \alpha_{\boldsymbol{j},\boldsymbol{\eta}}\phi_{\boldsymbol{j},\boldsymbol{\eta}}\right)$$

    else

$$\mathcal{A} \leftarrow \mathcal{A}\backslash\{l^*\}$$

        break
    end if

$$g^* \leftarrow \underset{g\in\mathcal{A}}{\text{argmin}}\, s_g(\mathcal{A})$$

    if $g^* \neq l^*$ **then**

$$\mathcal{A} \leftarrow \mathcal{A}\backslash\{g^*\}$$

        Compute $\overline{\mathcal{M}}$ by k-fold cross validation.
        if $\overline{\mathcal{M}} < \overline{\mathcal{M}}_0$ **then**

$$\overline{\mathcal{M}}_0 \leftarrow \overline{\mathcal{M}} \ , \ \boldsymbol{\alpha}_{\mathcal{A}} \leftarrow \underset{\alpha}{\text{argmin}}\, \mathcal{M}\left(\sum_{(\boldsymbol{j},\boldsymbol{\eta})\in\mathcal{A}} \alpha_{\boldsymbol{j},\boldsymbol{\eta}}\phi_{\boldsymbol{j},\boldsymbol{\eta}}\right)$$

        else

$$\mathcal{A} \leftarrow \mathcal{A}\bigcup\{g^*\}$$

        end if
    end if
end loop

$$\boldsymbol{\alpha}_{\mathcal{A}} \leftarrow \underset{\alpha}{\text{argmin}}\, \mathcal{M}_u\left(\sum_{(\boldsymbol{j},\boldsymbol{\eta})\in\mathcal{A}} \alpha_{\boldsymbol{j},\boldsymbol{\eta}}\phi_{\boldsymbol{j},\boldsymbol{\eta}}\right)$$

Output: $\mathcal{A}$, $\alpha_{\mathcal{A}}$.

Algorithm 2: Training twin model with adaptive basis construction.

2.4 Numerical Results

This section demonstrates the twin model on the estimation of the gradients for several numerical examples.

2.4.1 Buckley-Leverett Equation

Section 2.2 applies a sigmoid parameterization to the gray-box model governed by the Buckley-Leverett equation (2.28)

$$\frac{\partial u}{\partial t} + \frac{\partial}{\partial x}\left(\frac{u^2}{1+2(1-u)^2}\right) = c\,,$$

In this section, the same problem is studied but using the adaptive basis construction developed in Section 2.3. The flux function $F(u) = \frac{u^2}{1+2(1-u)^2}$ is assumed unknown and is going to be inferred. The gray-box and the twin models use the same second-order finite volume scheme and Crank-Nicolson time marching scheme. The initial dictionary, $\mathcal{A}$, is selected to contain a single basis $(1,0)$. The choice of the initial dictionary is not unique. We choose $(1,0)$ because it has a low resolution and is centered inside $[u_{\min}, u_{\max}]$ of the gray-box solution.

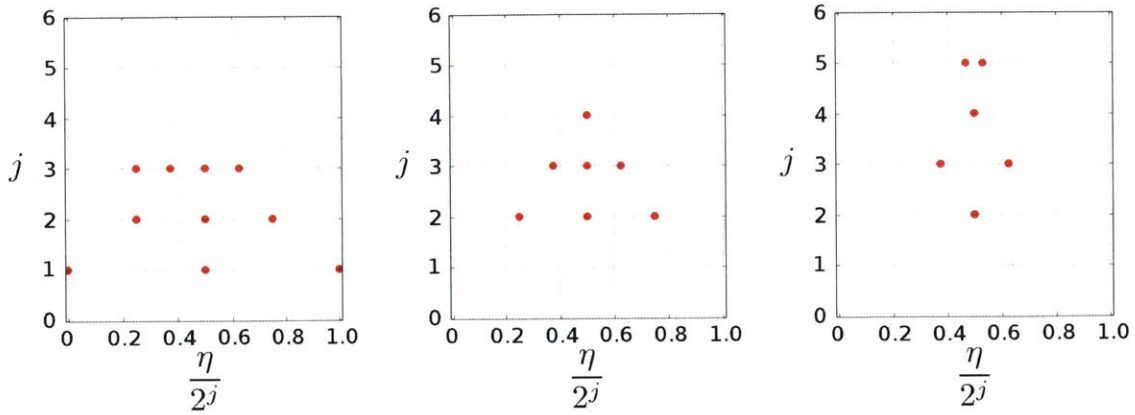
Figure 2-15 shows the selected bases for the three solutions in Figure 2-13, respectively, obtained by algorithm 2. As $[u_{\min}, u_{\max}]$ shrinks, the resolution of the bases increases. Figure 2-16 shows the dictionary and $\frac{d\tilde{F}}{du}$ at each forward-backward iteration for solution 3 (Figure 2-15c).

Consider a time-space-dependent control $c = c(t,x)$ in (2.28)

$$\frac{\partial u}{\partial t} + \frac{\partial}{\partial x}\left(\frac{u^2}{1+2(1-u)^2}\right) = c\,,$$

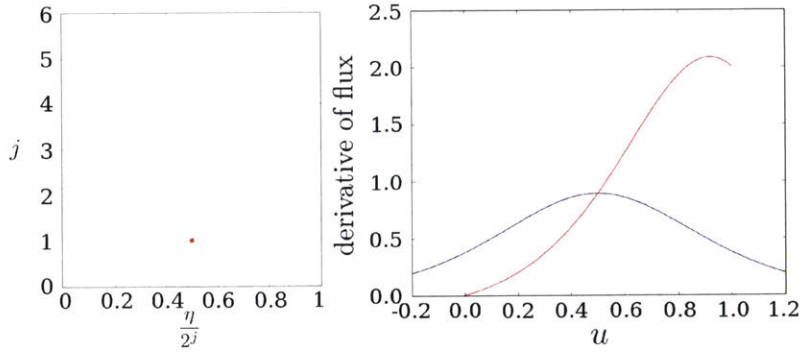
and (2.29)

$$\frac{\partial \tilde{u}}{\partial t} + \frac{\partial}{\partial x}\tilde{F}(\tilde{u}) = c\,.$$

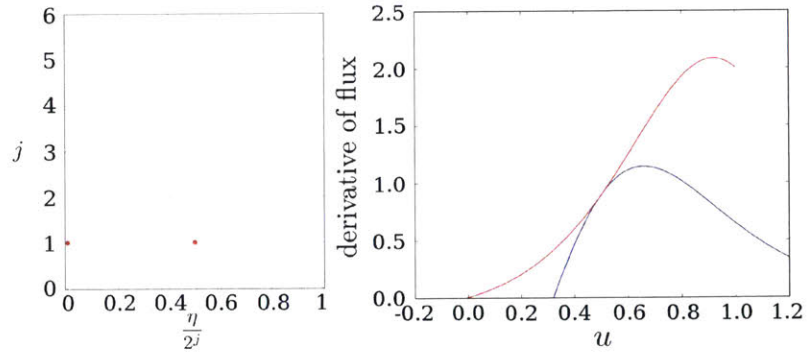


(a) Solution 1 (b) Solution 2 (c) Solution 3

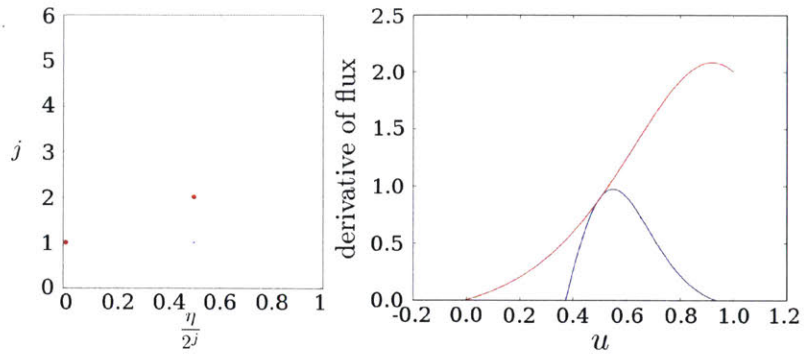
Figure 2-15: The basis dictionary for the three solutions in Figure 2-13. The iteration starts from the initial basis $(1, 0)$.



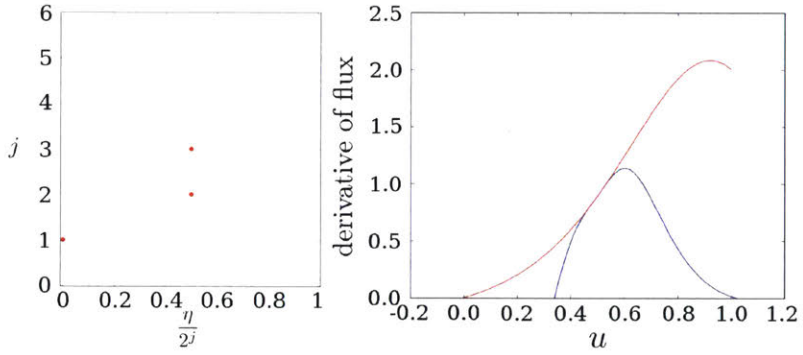
(a) Iteration 1



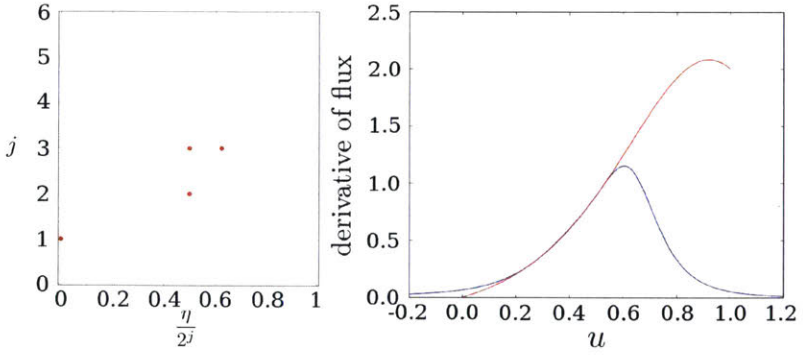
(b) Iteration 2



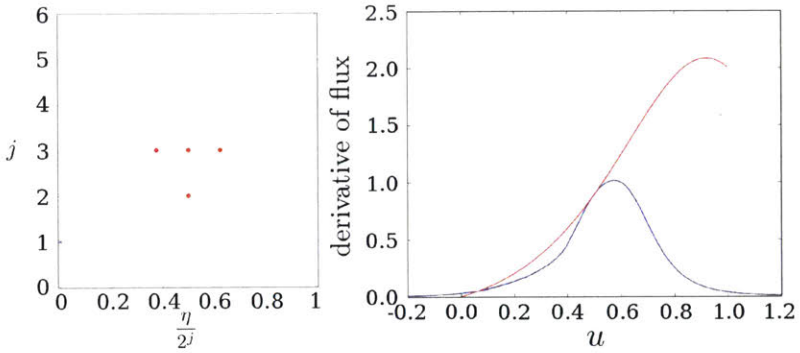
(c) Iteration 3



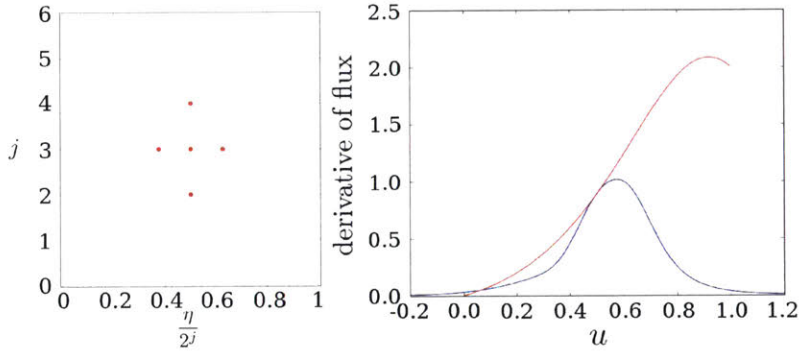
(d) Iteration 4



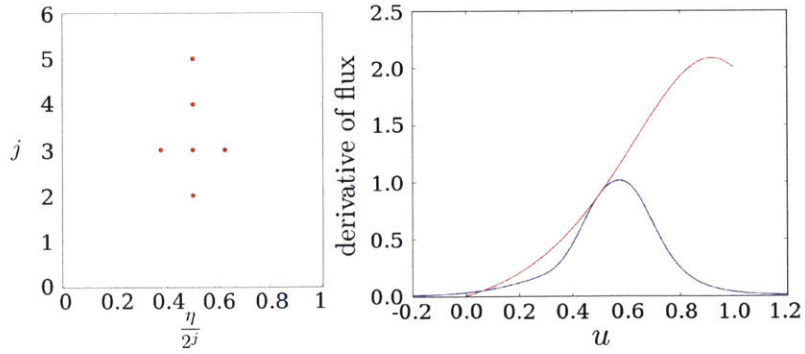
(e) Iteration 5



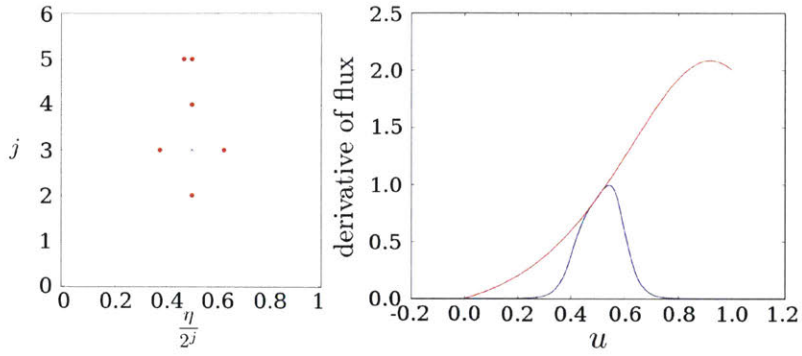
(f) Iteration 6



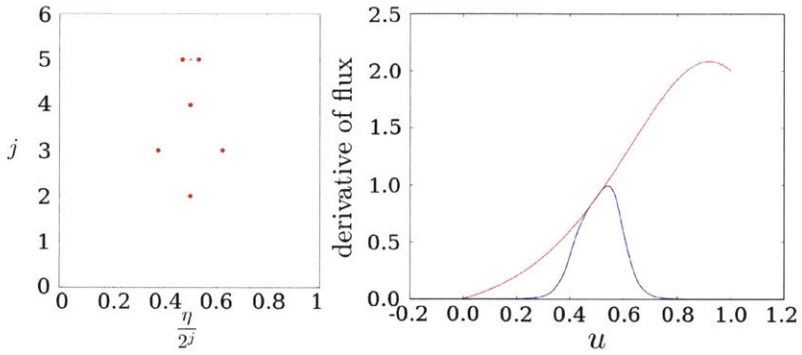
(g) Iteration 7



(h) Iteration 8



(i) Iteration 9



(j) Iteration 10

Figure 2-16: The basis dictionary at each forward-backward iteration when the initial condition is chosen as Figure 2-13c. In the left figure, the red dots indicate the bases in the dictionary, while the blue crosses indicate the deleted basis. In the right figure, the red line indicates the derivative of the true flux, while the blue line indicates the derivative of the trained flux.

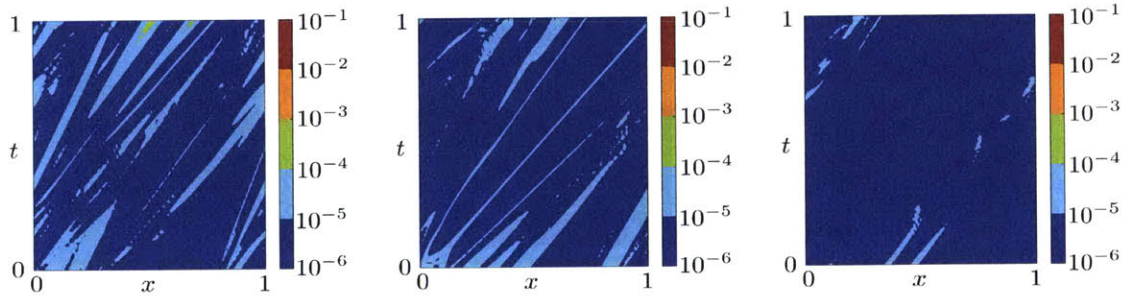
The gradient of ξ, (2.35),

$$\xi(c) \equiv \int_{x=0}^{1} \left(u(1,x;c) - \frac{1}{2} \right)^2 \mathrm{d}x \,,$$

can be estimated by the trained twin model. The estimated gradients are compared with the true adjoint gradients of the gray-box model, and the errors are shown in Figure 2-17. The adaptive basis construction improves the accuracy of the gradient estimation. Table 2.1 shows the error in the estimated gradient for the three solutions. The error in the estimated gradient is given by

$$\left(\int_0^1 \int_0^1 \left| \frac{d\tilde{\xi}}{dc} - \frac{d\xi}{dc} \right|^2 dx dt \right)^{1/2} ,$$

where ξ is the objective function evaluated by the gray-box model, and $\tilde{\xi}$ is the objective function evaluated by the twin model. We compare the result by using the manually chosen bases in Figure 2-6 and by using the bases constructed adaptively.



(a) Solution 1 *(b) Solution 2* *(c) Solution 3*

Figure 2-17: Error of the estimated gradient, $\left| \frac{d\tilde{\xi}}{dc} - \frac{d\xi}{dc} \right|$, for the three solutions. The basis dictionary is constructed adaptively.

	Solution 1	Solution 2	Solution 3
Manual basis	2.5×10^{-3}	6.6×10^{-4}	7.3×10^{-5}
Adaptive basis	4.2×10^{-6}	1.5×10^{-6}	8.9×10^{-7}

Table 2.1: Error of the estimated gradients for the three solutions. The adaptively constructed bases reduce the estimation error.

2.4.2 Navier-Stokes Flow

Consider a steady-state, compressible, viscous, adiabatic flow in a 2-D return bend channel driven by the pressure difference between the inlet and the outlet. The geometry of the return bend is given in Figure 2-18. The return bend is bounded by no-slip walls. The inlet static pressure and the outlet pressure are fixed. The inner and outer boundaries of the bending section are each generated by six control points using quadratic B-spline.

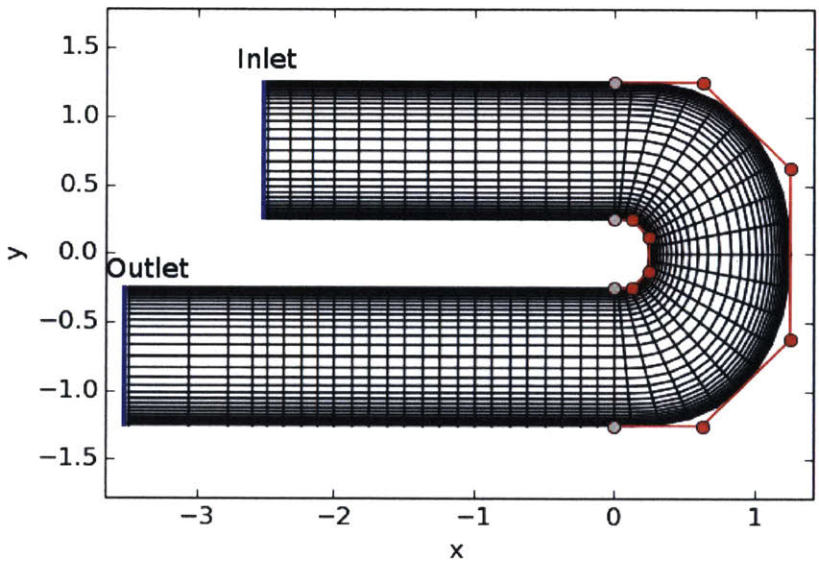


Figure 2-18: Return bend geometry and the mesh for the simulation. The control points for the inner and outer boundaries are indicated by the red dots.

The flow is governed by the Navier-Stokes equations. Let ρ, u, v, E, and p be the density, Cartesian velocity components, total energy, and pressure. The steady-state

Navier-Stokes equation is [6]

$$\frac{\partial}{\partial x}\begin{pmatrix} \rho u \\ \rho u^2 + p - \sigma_{xx} \\ \rho uv - \sigma_{xy} \\ u(E\rho + p) - \sigma_{xx}u - \sigma_{xy}v \end{pmatrix} + \frac{\partial}{\partial y}\begin{pmatrix} \rho v \\ \rho uv - \sigma_{xy} \\ \rho v^2 + p - \sigma_{yy} \\ v(E\rho + p) - \sigma_{xy}u - \sigma_{yy}v \end{pmatrix} = \mathbf{0}\,, \quad (2.43)$$

where

$$\begin{aligned} \sigma_{xx} &= \mu\left(2\frac{\partial u}{\partial x} - \frac{2}{3}\left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y}\right)\right) \\ \sigma_{yy} &= \mu\left(2\frac{\partial v}{\partial y} - \frac{2}{3}\left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y}\right)\right). \\ \sigma_{xy} &= \mu\left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right) \end{aligned} \quad (2.44)$$

The Navier-Stokes equation requires an additional equation, the state equation, for closure [6]. The state equation has the form

$$p = p(U, \rho)\,, \quad (2.45)$$

where U denotes the internal energy per unit volume [6],

$$U = \rho\left(E - \frac{1}{2}(u^2 + v^2)\right). \quad (2.46)$$

Many models have been developed for the state equation, such as the ideal gas equation, the van der Waals equation, and the Redlich-Kwong equation [103]. We assume the true state equation in the gray-box simulator is unknown. The state equation will be inferred from the gray-box solution. Let ρ_∞ be the steady-state density, $\boldsymbol{u}_\infty = (u_\infty, v_\infty)$ be the steady-state velocity, and E_∞ be the steady-state energy density. The steady-state mass flux is

$$\xi = -\int_{\text{outlet}} \rho_\infty u_\infty\big|_{\text{outlet}}\, dy = \int_{\text{inlet}} \rho_\infty u_\infty\big|_{\text{inlet}}\, dy \quad (2.47)$$

The goal is to estimate the gradient of ξ to the red control points' coordinates.

Two state equations are tested: the van der Waals equation and the Redlich-Kwong equation [6, 9],

$$\begin{aligned} p_{vdw} &= \frac{(\gamma-1)U}{1-b_{vdw}\rho} - a_{vdw}\rho^2 \\ p_{rk} &= \frac{(\gamma-1)U}{1-b_{rk}\rho} - \frac{a_{rk}\rho^{5/2}}{((\gamma-1)U)^{1/2}(1+b_{rk}\rho)} \end{aligned}, \tag{2.48}$$

where we set $a_{vdw} = 10^4$, $b_{vdw} = 0.1$, $a_{rk} = 10^7$ and $b_{rk} = 0.1$.

In both the gray-box and the twin models, we use the same second-order finite volume scheme for discretization and a pseudo-time marching scheme to solve for the steady-state solution. The solution mismatch, (2.2), is given by

$$\begin{aligned} \mathcal{M} =& w_\rho \int_\Omega |\tilde{\rho}_\infty - \rho_\infty|^2 \, d\boldsymbol{x} + w_u \int_\Omega |\tilde{u}_\infty - u_\infty|^2 \, d\boldsymbol{x} \\ &+ w_v \int_\Omega |\tilde{v}_\infty - v_\infty|^2 \, d\boldsymbol{x} + w_E \int_\Omega \left|\tilde{E}_\infty - E_\infty\right|^2 d\boldsymbol{x}, \end{aligned}$$

where w_ρ, w_u, w_v, and w_E are non-dimensionalization constants. Figure 2-19 shows the gray-box solution and the solution mismatch after training the twin model. The selected bases in the dictionary, represented by $(j_U, j_\rho, \eta_U, \eta_\rho)$, are listed in Table 2.2 and 2.3. They are also shown in Figure 2-20 on a $(\frac{\eta_U}{2^{j_U}}, \frac{\eta_\rho}{2^{j_\rho}})$ plane. Figure 2-21 shows the cross-validation error $\mathcal{M}_\tau$ at each forward-backward iteration. Figure 2-22 shows the trained state equation and its error. The convex hull of (U_∞, ρ_∞) is shown by the red dashed line. Because the state equation is expected to be inferable only inside the domain of the gray-box solution, a large deviation is expected outside the convex hull.

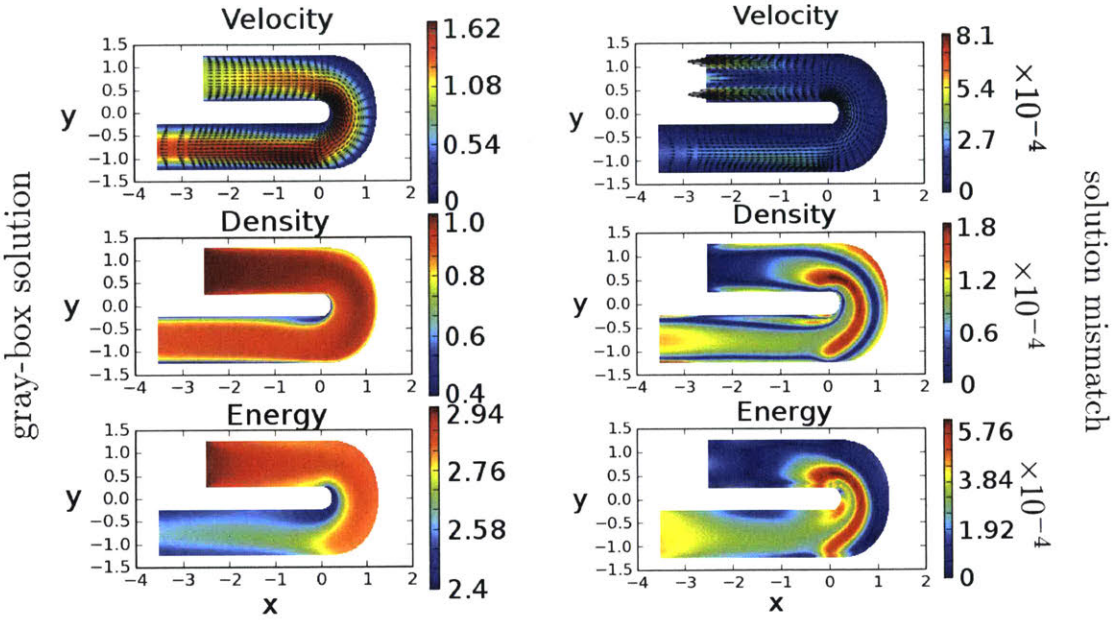


Figure 2-19: Left column: Example gray-box solution for a given geometry. Right column: Solution mismatch after training a twin model.

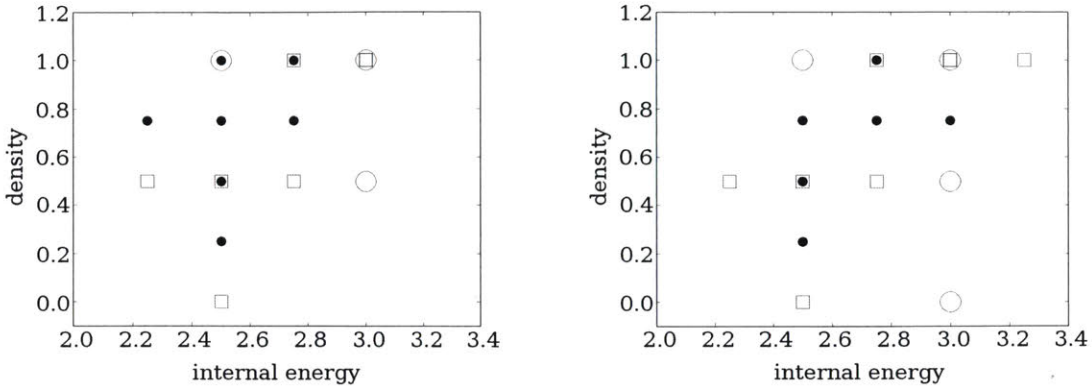
(1,1,5,2)	(1,1,6,2)	(1,1,6,1)	(2,1,9,1)	(2,1,10,1)
(2,1,11,2)	(2,1,11,1)	(2,1,10,0)	(2,1,12,2)	(2,2,9,3)
(2,2,10,2)	(2,2,10,3)	(2,2,11,3)	(2,2,10,1)	(2,2,11,4)
(2,2,10,4)				

Table 2.2: List of the dictionary for the van der Waals gas, $(j_U, j_\rho, \eta_U, \eta_\rho)$.

(1,1,5,2)	(1,1,6,2)	(1,1,6,1)	(1,1,6,0)	(2,1,9,1)
(2,1,10,1)	(2,1,11,2)	(2,1,11,1)	(2,1,10,0)	(2,1,12,2)
(2,1,13,2)	(2,2,10,2)	(2,2,10,3)	(2,2,11,3)	(2,2,12,3)
(2,2,10,1)	(2,2,11,4)			

Table 2.3: List of the dictionary for the Redlich-Kwong gas, $(j_U, j_\rho, \eta_U, \eta_\rho)$.

The trained twin model enables the adjoint gradient estimation. Figure 2-23 shows the estimated gradient of ξ to the control points' coordinates. It also compares the estimated gradient with the true gradient. The error of the gradient estimation is given in Table 2.4.



(a) Van der Waals state equation. *(b) Redlich-Kwong state equation.*

Figure 2-20: The basis dictionary for the state equations, plotted on the $(\frac{\eta_U}{2^{j_U}}, \frac{\eta_\rho}{2^{j_\rho}})$ *plane. The circles represent the bases that have* $j_U = 1, j_\eta = 1$. *The squares represent the bases that have* $j_U = 2, j_\eta = 1$. *The dots represent the bases that have* $j_U = 2, j_\eta = 2$.

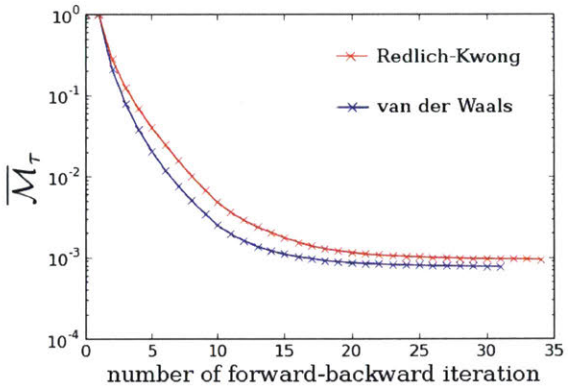
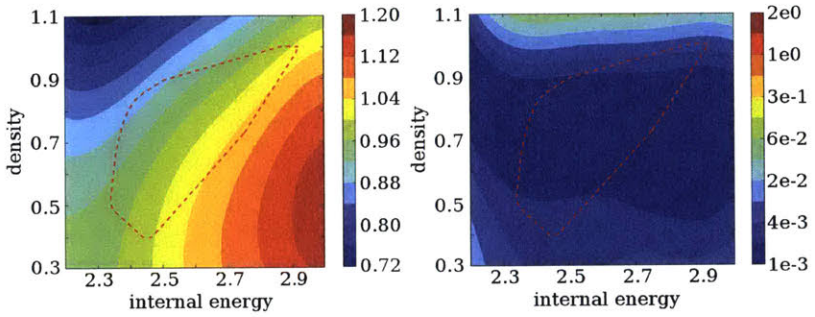
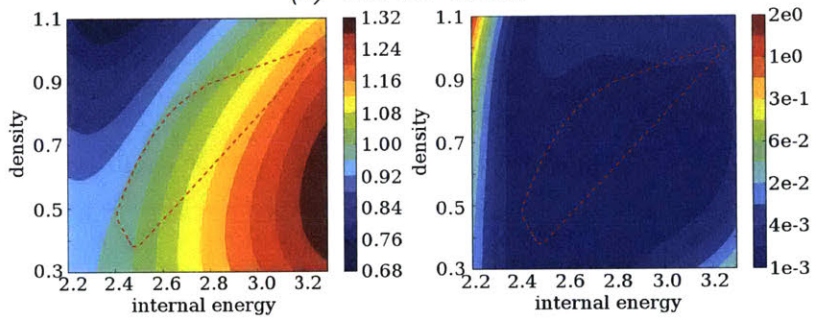


Figure 2-21: Cross-validation error $\overline{\mathcal{M}}_\tau$ *at each forward-backward iteration. The y-axis is scaled by a constant, so that* $\overline{\mathcal{M}}_\tau$ *at the first forward-backward iteration equals* 1.

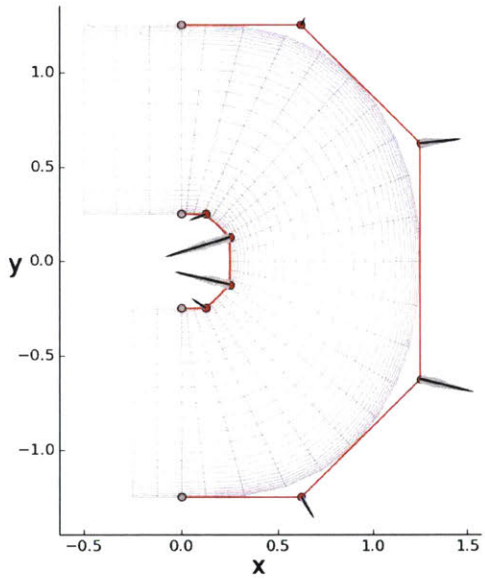


(a) Van der Waals

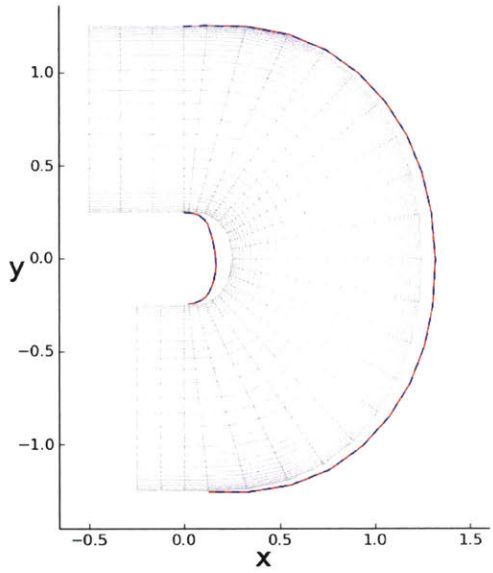


(b) Redlich-Kwong

Figure 2-22: State equation for the van der Waals gas, and for the Redlich-Kwong gas. Left column shows the trained state equation; right column shows the error of the state equation. The trained state equation is added by a constant, so the pressure matches the pressure of the gray-box equation at $U = 2.6$ and $\rho = 0.7$. The dashed red line shows the convex hull of the gray-box solution.



(a) Gradient of ξ to the control points for the Redlich-Kwong gas. The wide gray arrow is the gradient evaluated by the gray-box model, while the thin black arrow is the true gradient.



(b) Boundary perturbed according to the gradient. The blue dashed line is obtained by the true gradient, while the red dashed line is obtained by the twin-model gradient.

Figure 2-23: Comparison of the estimated gradient and the true gradient for the Redlich-Kwong gas. The result for the van der Waals gas is visually indistinguishable to this plot.

Gas	Interior control points				Exterior control points			
van der Waals	0.13	0.04	0.05	0.32	0.16	0.15	0.07	0.02
Redlich-Kwong	0.32	0.03	0.07	0.50	0.40	0.12	0.06	0.05

Table 2.4: Error of the gradient estimation, in percentage.

2.4.3 Polymer Injection in Petroleum Reservoir

Water flooding is a technique to enhance the secondary recovery in petroleum reservoirs, as illustrated in Figure 2-24. Injecting pure water can be cost-inefficient due to low water viscosity and high water cut. Therefore, the water-solvent polymer can be utilized to increase the water-phase viscosity and to reduce the residual oil.

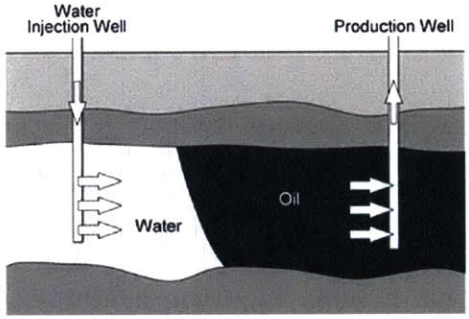


Figure 2-24: Water flooding in petroleum reservoir engineering (from PetroWiki). Polymer solved in the water phase can be injected into the reservoir to enhance the production of oil.

Consider a reservoir governed by the two-phase porous media flow equations

$$\begin{aligned} &\frac{\partial}{\partial t}\left(\rho_\alpha \phi S_\alpha\right) + \nabla \cdot \left(\rho_\alpha \vec{v}_\alpha\right) = 0\,, \quad \alpha \in \{w, o\} \\ &\frac{\partial}{\partial t}\left(\rho_w \phi S_w c\right) + \nabla \cdot \left(c\rho \vec{v}_{wp}\right) = 0 \end{aligned}\,, \tag{2.49}$$

for $x \in \Omega$ and $t \in [0, T]$, where the phase velocities are given by the Darcy's law

$$\begin{aligned} \vec{v}_\alpha &= -M_\alpha k_{r\alpha} \boldsymbol{K} \cdot (\nabla p - \rho_w g \nabla z), \quad \alpha \in \{w, o\} \\ \vec{v}_{wp} &= -M_{wp} k_{rw} \boldsymbol{K} \cdot (\nabla p - \rho_w g \nabla z) \end{aligned} . \tag{2.50}$$

w, o indicate the water and oil phases. ρ is the phase density. ϕ is the porosity. S is the phase saturation where $S_w + S_o = 1$. c is the polymer concentration in the water phase. v_w, v_o, v_{wp} are the componentwise velocities of water, oil, and polymer. $\boldsymbol{K}$ is the permeability tensor. k_r is the relative permeability. p is the pressure. z is the depth. g is the gravity constant. The mobility factors, M_o, M_w, M_{wp}, model the modification of the componentwise mobility due to the presence of polymer (oil, water, polymer).

The mobility factors depend on S_w, p, and c, but the algebraic form of the dependence can be proprietary and unknown. In this example, we are going to infer such dependence.

PSim, the industrial simulator aforementioned in Section 1.1, is used as the gray-box simulator, where we use the upwinding scheme and the IMPES time marching [4], i.e. implicit in pressure and explicit in saturation. Its solution, S_w, c, and p can be used to train the twin model. The twin model is implemented in MRST, an open-source reservoir simulation toolbox [109]. We use the upwinding scheme and the fully implicit time marching to simulate the twin model. The solution mismatch is defined by

$$\mathcal{M}_u = w_{S_w} \int_0^T \int_\Omega |S_w - \tilde{S}_w|^2 d\boldsymbol{x}dt + w_c \int_0^T \int_\Omega |c - \tilde{c}|^2 d\boldsymbol{x}dt + w_p \int_0^T \int_\Omega |p - \tilde{p}|^2 d\boldsymbol{x}dt \,, \tag{2.51}$$

where w_{S_w}, w_c, and w_p are non-dimensionalization constants.

Consider a reservoir setup shown in Figure 2-25, which is a 3D block with two

injectors and one producer. The permeability is 100 milli Darcy, and the porosity is 0.3. A constant injection rate of $10^6 \mathtt{ft}^3/\mathtt{day}$ is used at both the injectors. The reservoir is simulated for $t \in [0, 50]\mathtt{day}$. To select the basis dictionary for the twin model, the truncation errors for the three equations in (2.49) are minimized separately by using the forward-backward iteration. Figure 2-26 shows the trained mobilities for oil, polymer, and water. Tables 2.5 through 2.7 list the basis dictionary for the three mobilities. Figure 2-27 shows the truncation error at each forward-backward iteration for the three mobilities. The solution of S_w is illustrated in Figure 2-28 for the untrained twin model, the gray-box model, and the trained twin model, respectively. After the training, the twin-model solution matches the gray-box solution closely.

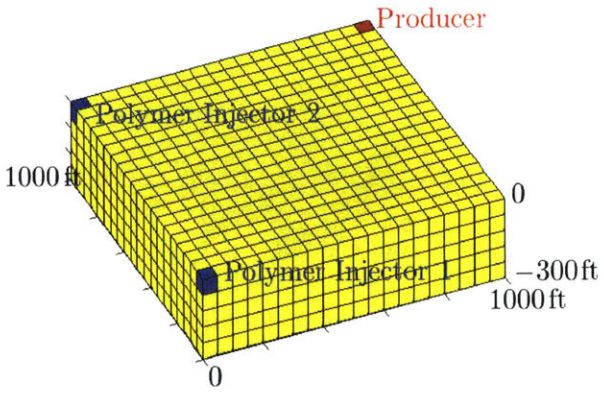
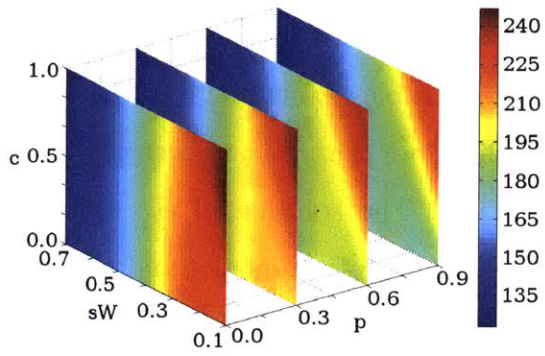


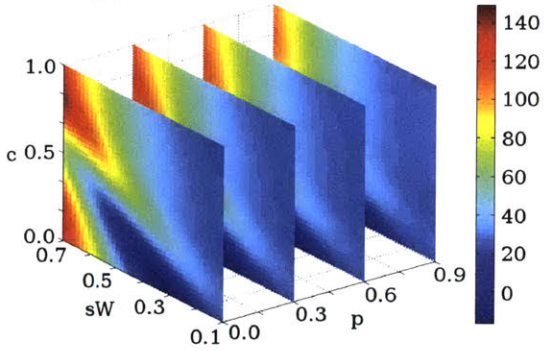
Figure 2-25: The geometry of the petroleum reservoir.

(-2, -2, -2, -1, 0, 0)	(-2, -2, -2, 0, 0, 1)	(-2, -2, -2, 0, 1, 2)
(-2, -2, -2, 0, 1, 3)	(-2, -1, -2, 0, 0, 0)	(-2, -1, -2, -1, 1, 1)
(-2, -1, -2, 0, 0, 1)	(-2, -1, -2, 0, 1, 1)	(-2, -1, -2, 0, 1, 2)
(-2, -1, -2, 0, 1, 3)	(-2, -1, -1, 0, 0, 0)	(-2, -1, -1, 0, 0, 1)
(-2, -1, -1, 0, 1, 1)	(-2, -1, -1, -1, 1, 2)	(-2, -1, -1, 0, 0, 2)
(-2, -1, -1, 0, 1, 2)	(-2, -1, -1, 0, 1, 3)	(-2, -1, -1, 0, 2, 3)

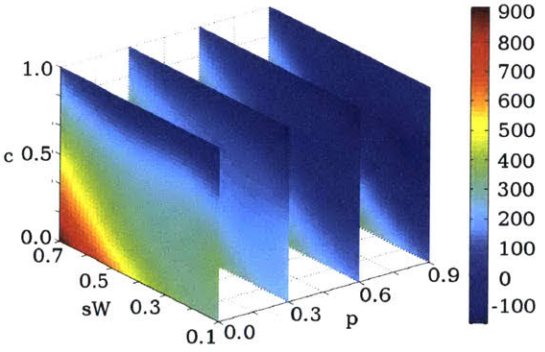
Table 2.5: $(j_p, j_{sW}, j_c, \eta_p, \eta_{sW}, \eta_c)$ for M_o.



(a) Oil mobility.



(b) Polymer mobility.



(c) Water mobility.

Figure 2-26: The trained mobilities M_o, M_w, M_{wp}.

(-2, -2, -2, -1, 0, 0)	(-2, -2, -2, 0, 0, 0)	(-2, -2, -2, 0, 0, 1)
(-2, -1, -2, 0, 0, 0)	(-2, -1, -2, -1, 1, 1)	(-2, -1, -2, 0, 0, 1)
(-2, -1, -2, 0, 1, 1)	(-2, -1, -2, 0, 1, 2)	(-2, -1, -2, 0, 1, 3)
(-2, -1, -2, 0, 2, 3)	(-2, -1, -1, 0, 0, 0)	(-2, -1, -1, -1, 0, 1)
(-2, -1, -1, 0, 1, 1)	(-2, -1, -1, -1, 1, 2)	(-2, -1, -1, 0, 0, 2)
(-2, -1, -1, 0, 1, 2)	(-2, -1, -1, 0, 1, 3)	(-2, -1, -1, 0, 2, 3)
(-2, -1, -1, 0, 1, 4)	(-2, -1, -1, 0, 2, 4)	(-2, -1, -1, 0, 2, 5)
(-2, 0, -1, -1, 1, 0)	(-2, 0, -1, 0, 0, 0)	(-2, 0, -1, 0, 1, 0)
(-2, 0, -1, -1, 0, 1)	(-2, 0, -1, 0, 1, 1)	(-2, 0, -1, 0, 3, 1)
(-2, 0, -1, 0, 0, 2)	(-2, 0, -1, 0, 1, 2)	(-2, 0, -1, 0, 1, 3)
(-2, 0, -1, 0, 2, 4)	(-2, 0, 0, -1, 0, 2)	

Table 2.6: $(j_p, j_{sW}, j_c, \eta_p, \eta_{sW}, \eta_c)$ *for* M_{wp}.

(-2, -2, -2, -1, 0, 0)	(-2, -2, -2, 0, 0, 0)	(-2, -2, -2, 0, 0, 1)
(-2, -2, -2, 0, 1, 2)	(-2, -2, -2, 0, 1, 3)	(-2, -1, -2, -1, 0, 0)
(-2, -1, -2, 0, 0, 0)	(-2, -1, -2, -1, 1, 1)	(-2, -1, -2, 0, 0, 1)
(-2, -1, -2, 0, 1, 1)	(-2, -1, -2, 0, 1, 2)	(-2, -1, -2, 0, 2, 2)
(-2, -1, -2, 0, 1, 3)	(-2, -1, -1, -1, 0, 0)	(-2, -1, -1, 0, 0, 0)
(-2, -1, -1, -1, 0, 1)	(-2, -1, -1, 0, 0, 1)	(-2, -1, -1, 0, 1, 1)
(-2, -1, -1, -1, 1, 2)	(-2, -1, -1, 0, 0, 2)	(-2, -1, -1, 0, 1, 2)
(-2, -1, -1, 0, 1, 3)	(-2, -1, -1, 0, 2, 3)	

Table 2.7: $(j_p, j_{sW}, j_c, \eta_p, \eta_{sW}, \eta_c)$ *for* M_w.

Let the objective function be the residual oil at $T = 50\,\texttt{days}$,

$$\xi = \int_\Omega \rho_o(T)\phi S_o(T)\,\mathrm{d}\boldsymbol{x}\,. \tag{2.52}$$

The gradient of ξ with respect to the time-dependent injection rate is computed. The gradient estimated by the twin model is shown in Figure 2-29, where the red and blue lines indicate the gradient for the two injectors. In comparison, the star markers show the true gradient at day 2, 16, 30, and 44, evaluated by finite difference. Clearly, a

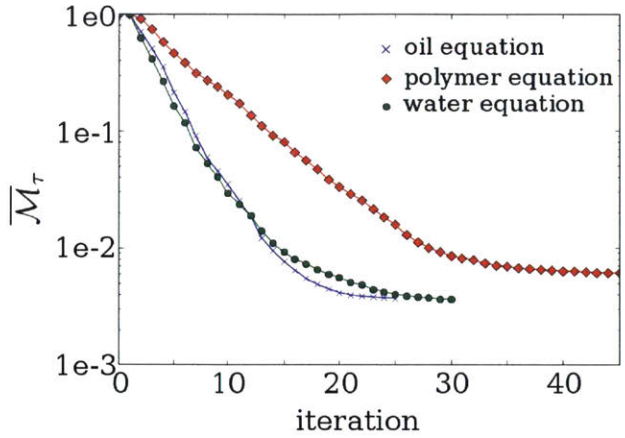


Figure 2-27: Relative cross-validation error at each forward-backward iteration. The x axis is the number of iteration, and the y axis is the integrated truncation error for $\overline{M}_\tau$ of the three equations in (2.49).

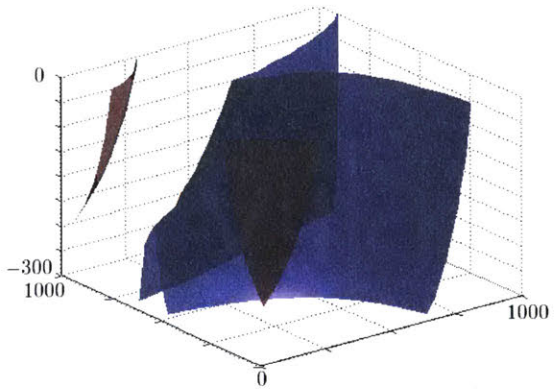
rate increase at the injector 1 leads to more residual oil reduction than the injector 2. This is because the injector 2 is closer to the producer, where a larger rate accelerates the water breakthrough that impedes further oil production. It is observed that the estimated gradient closely matches the true gradient which is computed by finite difference. The error is given in Table 2.8.

Error	$t = 0.04$	$t = 0.32$	$t = 0.6$	$t = 0.88$
Inj 1	1.7	1.0	0.6	0.2
Inj 2	2.2	1.9	0.7	0.2

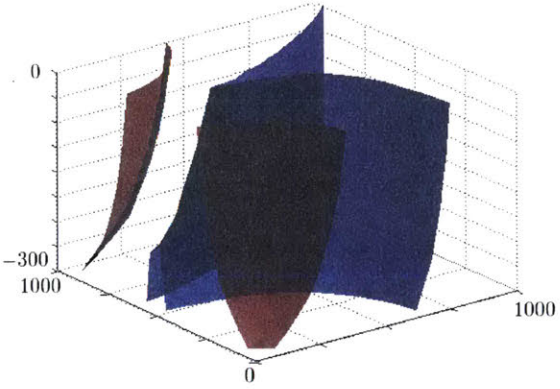
Table 2.8: Error of estimated gradient at day 2, 16, 30, *and* 44, *in percentage.*

2.5 Chapter Summary

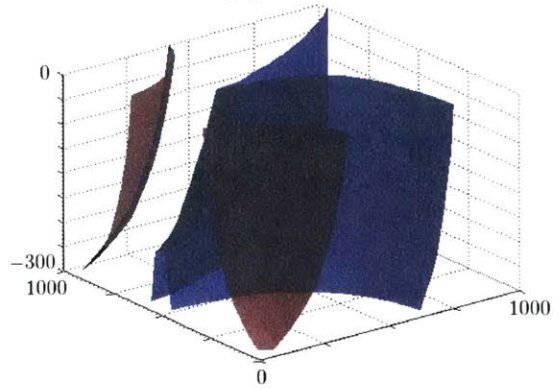
This chapter develops a method for gradient estimation by using the space-time solution of gray-box conservation law simulations, at a cost independent of the dimensionality of the gradient. The key to inferring F is to leverage the gray-box space-time solution. My method uses the big data, the gray-box space-time solution, to estimate



(a) Untrained twin model.



(b) PSim.



(c) Trained twin model.

Figure 2-28: Isosurfaces of $S_w = 0.25$ and $S_w = 0.7$ at $t = 30$ days. After the training, the twin-model solution matches the gray-box solution.

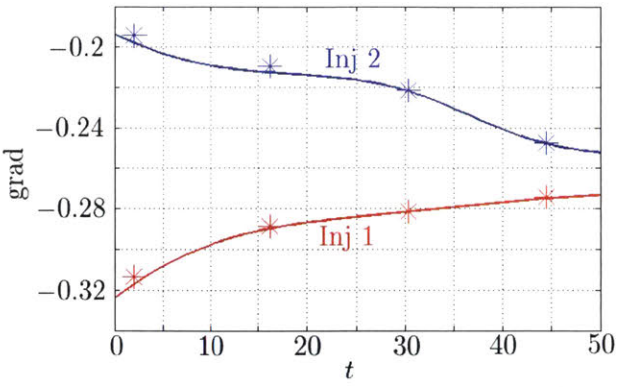


Figure 2-29: Gradient of ξ with respect to rates at the two injectors. The lines indicate the gradients estimated by the twin model, while the stars indicate the true gradient evaluated by finite difference.

the unknown component in the gray-box model and to estimate the gradient. The twin model is an adjoint-enabled conservation law simulator, and can be trained to minimize a metric measuring its difference against the gray-box simulator. Two metrics, the solution mismatch and the integration truncation error, are proposed. To enable the training computationally, a sigmoid parameterization is presented. Then the twin model method is demonstrated in the Buckley-Leverett equation by using a set of manually chosen bases. To further exploit the information contained in the gray-box solution, an adaptive basis construction procedure is presented. The adaptive procedure iterates over a forward step and a backward step to append and delete basis in the basis dictionary.

The proposed twin-model algorithm is demonstrated on a variety of numerical examples. The first example is the Buckley-Leverett equation, whose flux function is inferred. The trained twin model accurately estimates the gradient of an objective to the source term. The second example is the steady-state Navier-Stokes equation in a return bend, whose state equation is inferred. The inferred state equation allows estimating the gradient of mass flux to the control surface geometry. The third example is the petroleum reservoir with polymer injection, where the mobility factors are inferred. The gradient of the residual oil to the injection rate is estimated. With

the aid of the estimated gradient, the objective can be optimized more efficiently, which will be discussed in the next chapter.

Chapter 3

Leveraging the Twin Model for Bayesian Optimization

This chapter develops a Bayesian optimization framework to solve (1.9),

$$c^* = \underset{c_{\min} \le c \le c_{\max}}{\operatorname{argmax}} \quad \xi(\boldsymbol{u}, c)$$
$$\xi(c) = \sum_{i=1}^{M}\sum_{j=1}^{N} w_{ij} f(\boldsymbol{u}_{ij}, c; t_i, x_j) \approx \int_0^T \int_\Omega f(u, c; t, x) d\boldsymbol{x} dt$$

where ξ is the objective function that is twice differentiable, $\boldsymbol{u}$ is the gray-box solution, c is the control variables, $i = 1, \cdots, M$, $j = 1, \cdots, N$ are the indices for the time and space grid, and w_{ij}'s are the quadrature weights. In the following context, we assume ξ to be twice differentiable. As discussed in Section 1.3.1, the advantage of Bayesian optimization is that it uses **all** the information available from previous evaluations. This advantage can be valuable in our context where the gray-box simulation is expensive. The estimated gradient, provided by the twin model, is utilized to improve the optimization performance. The goal is to reduce the number of gray-box simulations required to achieve the desired objective evaluation, as well as to reduce the overall computational cost. The chapter is organized as follows. Section 3.1 discusses the Bayesian modeling of the objective function and its gradient. The modeling is used to develop a Bayesian optimization algorithm in Section 3.2. The

convergence properties of the algorithm are investigated in Section 3.3. Finally, the algorithm is demonstrated in Section 3.4 in several numerical examples.

3.1 Modeling the Objective and Gradient by Gaussian Processes

Assume the gray-box simulator evaluates the objective function ξ accurately. The adjoint gradient estimated by the twin model may not equal the true gradient for several reasons. For example, the gray-box solution can be under-resolved if the space-time grid is too coarse, thus limiting the accuracy of the inference of F. In addition, the simulators for the twin and gray-box models may use different numerical schemes, so the twin-model solution may not equal the gray-box solution even if $F = \tilde{F}$. Similarly, the $\tilde{F}$ that minimizes the solution mismatch may not equal F. Because of the errors in estimating F, an error is introduced in estimating the gradient. It is difficult to identify and separately quantify the various sources of errors in the estimated gradient. Instead, I model the gradient error as a whole without distinguishing the sources of errors.

Let $\nabla\xi$ be the true gradient of ξ, $\xi_{\tilde{\nabla}}$ be the twin-model estimated gradient, and $\xi_{\tilde{\nabla}i}$ be its ith component. We model the relationship between $\nabla\xi$ and $\xi_{\tilde{\nabla}}$ by [63, 64, 65]

$$\xi_{\tilde{\nabla}i} = \nabla\xi_i + \epsilon_i\,, \tag{3.1}$$

for $i = 1, \cdots, d$, where $\boldsymbol{\epsilon} = (\epsilon_1, \cdots, \epsilon_d)$ models the error in the estimated gradient, where ϵ_i's are functions that depend on the control variable.

Gaussian processes are adopted to model the terms in (3.1). In particular, I made the following assumptions.

1. ξ is a realization of a stationary Gaussian process with mean μ, and covariance

kernel $K(\cdot,\cdot)$;

2. $\epsilon_1, \cdots, \epsilon_d$ are realizations of zero-mean stationary Gaussian processes with covariances $G_1(\cdot,\cdot), \cdots, G_d(\cdot,\cdot)$, respectively;

3. The gradient errors, ϵ_i's, are independent with the objective,

$$\operatorname{cov}\left[\xi(c_1), \epsilon_i(c_2)\right] = 0\,, \tag{3.2}$$

for all $c_1, c_2 \in \mathbb{R}^d$, $i = 1, \cdots, d$;

4. The components of the gradient error are pairwise independent,

$$\operatorname{cov}\left[\epsilon_i(c_1), \epsilon_j(c_2)\right] = 0\,,$$

for all $c_1, c_2 \in \mathbb{R}^d$ and $i \neq j$;

5. The covariances are isotropic, i.e., $K(c_1, c_2)$, $G_1(c_1, c_2)$, $\cdots$ $G_d(c_1, c_2)$ only depend on $\left\|c_1 - c_2\right\|_{L_2}$.

Suppose ξ and $\xi_{\tilde{\nabla}}$ have been evaluated on $\underline{c}_n$ [1]. Based upon the assumptions above, the joint distribution of $\xi(c)$, $\xi(\underline{c}_n)$, and $\xi_{\tilde{\nabla}}(\underline{c}_n)$ is multivariate normal and is given by

$$\begin{pmatrix} \xi(c) \\ \xi(\underline{c}_n) \\ \xi_{\tilde{\nabla}}(\underline{c}_n) \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} \mu \\ \boldsymbol{\mu} \\ \mathbf{0} \end{pmatrix}, \begin{pmatrix} K(c,c) & \boldsymbol{v} & \boldsymbol{w} \\ \boldsymbol{v}^T & \boldsymbol{D} & \boldsymbol{H} \\ \boldsymbol{w}^T & \boldsymbol{H}^T & \boldsymbol{E} + \overline{\boldsymbol{G}} \end{pmatrix} \right), \tag{3.3}$$

where

$$\boldsymbol{v} = \left(K(c, c_1), \cdots, K(c, c_N)\right)\,, \tag{3.4}$$

$$\boldsymbol{w} = \left(\nabla_{c_1} K(c, c_1), \cdots, \nabla_{c_N} K(c, c_N)\right)\,, \tag{3.5}$$

$$\boldsymbol{D} = \begin{pmatrix} K(c_1, c_1) & \cdots & K(c_1, c_N) \\ \vdots & \ddots & \vdots \\ K(c_N, c_1) & \cdots & K(c_N, c_N) \end{pmatrix}, \tag{3.6}$$

[1] The notations are consistent with Section 1.3.1. The objective and estimated gradient evaluations are assumed to be collocated, which will be shown in Section 3.2.

$$\boldsymbol{H} = \begin{pmatrix} \nabla_{c_1} K(c_1, c_1) & \cdots & \nabla_{c_N} K(c_1, c_N) \\ \vdots & \ddots & \vdots \\ \nabla_{c_1} K(c_N, c_1) & \cdots & \nabla_{c_N} K(c_N, c_N) \end{pmatrix}, \tag{3.7}$$

$$\boldsymbol{E} = \begin{pmatrix} \nabla_{c_1} \nabla_{c_1'} K(c_1, c_1') & \cdots & \nabla_{c_1} \nabla_{c_N'} K(c_1, c_N') \\ \vdots & \ddots & \vdots \\ \nabla_{c_1} \nabla_{c_N'} K(c_N, c_1') & \cdots & \nabla_{c_N} \nabla_{c_N'} K(c_N, c_N') \end{pmatrix}, \tag{3.8}$$

$$\overline{\boldsymbol{G}} = \begin{pmatrix} \boldsymbol{G}(c_1, c_1) & \cdots & \boldsymbol{G}(c_1, c_N) \\ \vdots & \ddots & \vdots \\ \boldsymbol{G}(c_N, c_1) & \cdots & \boldsymbol{G}(c_N, c_N) \end{pmatrix}, \tag{3.9}$$

where $\boldsymbol{G}(c_i, c_j)$ is the covariance matrix of $\boldsymbol{\epsilon}(c_i)$ and $\boldsymbol{\epsilon}(c_j)$ given by

$$\boldsymbol{G}(c_i, c_j) = \operatorname{diag}\big(G_1(c_i, c_j), \cdots, G_d(c_i, c_j)\big), \ i, j = 1, \cdots, d. \tag{3.10}$$

The derivation of (3.7) and (3.8) can be found in [74].

As discussed in Section 1.3.1, there are many choices for the covariance kernels K and G, such as the exponential kernel, the squared exponential kernel, and the Matérn kernel. In the following context, the Matérn 5/2 kernel is used. This is because the functions simulated by this kernel are twice differentiable but without further smoothness [71, 89, 90]. We have

$$K(c_1, c_2) = \sigma_\xi^2 \left(1 + \frac{\sqrt{5}\|c_1 - c_2\|_{L_2}}{L_\xi} + \frac{5\|c_1 - c_2\|_{L_2}^2}{3{L_\xi}^2}\right) \exp\left(-\frac{\sqrt{5}\|c_1 - c_2\|_{L_2}}{L_\xi}\right), \tag{3.11}$$

$$G_i(c_1, c_2) = \sigma_{G_i}^2 \left(1 + \frac{\sqrt{5}\|c_1 - c_2\|_{L_2}}{L_{G_i}} + \frac{5\|c_1 - c_2\|_{L_2}^2}{3{L_{G_i}}^2}\right) \exp\left(-\frac{\sqrt{5}\|c_1 - c_2\|_{L_2}}{L_{G_i}}\right), \tag{3.12}$$

where σ_ξ, σ_{G_i}'s are the standard deviation parameters, and L_ξ, L_{G_i}'s are the correlation length parameters.

Let θ denote the hyper parameters L_ξ, σ_ξ, L_{G_i}'s, σ_{G_i}'s, and μ. Given the samples of ξ and $\xi_{\tilde{\nabla}}$ on a set of c's, θ can be estimated by log maximum likelihood. As discussed in Section 1.3.1, we use $\underline{c}_n = (c_1, \cdots, c_n)$ to represent a sequence of control variables on which ξ and $\xi_{\tilde{\nabla}}$ have been evaluated. The likelihood of observing $\xi(\underline{c}_n)$ and $\xi_{\tilde{\nabla}}(\underline{c}_n)$ is given by

$$\begin{aligned} p\left(\xi(\underline{c}_n), \xi_{\tilde{\nabla}}(\underline{c}_n)|\theta\right) &= \int p\big(\xi(\underline{c}_n), \xi_{\tilde{\nabla}}(\underline{c}_n), \nabla\xi(\underline{c}_n)|\theta\big) d\big(\nabla\xi(\underline{c}_n)\big) \\ &= \int p\big(\xi(\underline{c}_n), \nabla\xi(\underline{c}_n)|\theta\big) p\big(\xi_{\tilde{\nabla}}(\underline{c}_n)|\xi(\underline{c}_n), \nabla\xi(\underline{c}_n); \theta\big) d\big(\nabla\xi(\underline{c}_n)\big)\,. \end{aligned} \tag{3.13}$$

Because

$$\xi(\underline{c}_n), \nabla\xi(\underline{c}_n)\big|\theta \sim \mathcal{N}\left(\begin{pmatrix}\boldsymbol{\mu}\\ \mathbf{0}\end{pmatrix}, \begin{pmatrix}\boldsymbol{D} & \boldsymbol{H}\\ \boldsymbol{H}^T & \boldsymbol{E}\end{pmatrix}\right), \tag{3.14}$$

and

$$\xi_{\tilde{\nabla}}(\underline{c}_n)|\xi(\underline{c}_n), \nabla\xi(\underline{c}_n); \theta \sim \mathcal{N}\left(\nabla\xi(\underline{c}_n), \overline{\boldsymbol{G}}\right), \tag{3.15}$$

the log marginal likelihood can be derived. It has the closed form

$$\begin{aligned} &\log p(\xi(\underline{c}_n), \xi_{\tilde{\nabla}}(\underline{c}_n)|\theta) \\ &= -\frac{1}{2}\begin{pmatrix}\xi(\underline{c}_n) - \boldsymbol{\mu}\\ \xi_{\tilde{\nabla}}(\underline{c}_n)\end{pmatrix}^T \begin{pmatrix}\boldsymbol{D} & \boldsymbol{H}\\ \boldsymbol{H}^T & \boldsymbol{E} + \overline{\boldsymbol{G}}\end{pmatrix}^{-1} \begin{pmatrix}\xi(\underline{c}_n) - \boldsymbol{\mu}\\ \xi_{\tilde{\nabla}}(\underline{c}_n)\end{pmatrix} - \frac{1}{2}\log\left(\det\begin{pmatrix}\boldsymbol{D} & \boldsymbol{H}\\ \boldsymbol{H}^T & \boldsymbol{E} + \overline{\boldsymbol{G}}\end{pmatrix}\right) \\ &\quad - \frac{N(d+1)}{2}\log(2\pi)\,. \end{aligned} \tag{3.16}$$

The log marginal likelihood can be optimized efficiently using GBO methods. In my thesis, the optimization is done by the BFGS algorithm in the NLopt package [43].

Given the joint distribution (3.3) and the estimated hyperparameter θ, the posterior

of $\xi(c)$, for any $c \in \mathbb{R}^d$, can be obtained by (1.16),

$$\tilde{m}(c) = m(c) + K(c, \underline{c}_n)K(\underline{c}_n, \underline{c}_n)^{-1}\left(\xi(\underline{c}_n) - m(\underline{c}_n)\right)$$
$$\tilde{K}(c, c') = K(c, c') - K(c, \underline{c}_n)K(\underline{c}_n, \underline{c}_n)^{-1}K(\underline{c}_n, c')$$

As discussed in Section 1.3.1, the expected improvement (EI) acquisition function, $\rho(c)$, can be evaluated by using the posterior. The acquisition function can be optimized to find the next control variable to evaluate the objective function. In my thesis, the optimization is done by the StoGo algorithm [87, 88], a gradient-based branch-and-bound algorithm implemented in the NLopt package [43].

3.2 Optimization Algorithm

Based upon the developments in Section 3.1, I present the Bayesian optimization algorithm 3. The flowchart of the algorithm is sketched in Figure 3-1.

Input: Initial guess c. Current best control c_0^*. Current best objective ξ_0^*. Max iteration $n_{\max}$.
Expected improvement threshold $\texttt{EI}_{\min}$. $D_c = [\,]$, $D_\xi = [\,]$, $D_{\xi_{\tilde{\nabla}}} = [\,]$.
1: **for** $i = 1$ **to** $n_{\max}$ **do**
2: Simulate the gray-box model on c, obtain $\xi(c)$ and $\boldsymbol{u}(c)$.
3: Train a twin model using $\boldsymbol{u}(c)$, obtain $\xi_{\tilde{\nabla}}(c)$.
4: $D_c = [D_c, c]$, $D_\xi = [D_\xi, \xi(c)]$, $D_{\xi_{\tilde{\nabla}}} = [D_{\xi_{\tilde{\nabla}}}, \xi_{\tilde{\nabla}}(c)]$.
5: **if** $\xi(c) > \xi_0^*$ **then**
6: $c_0^* \leftarrow c$
7: **end if**
8: Update hyper parameters by MLE.
9: $c \leftarrow \text{argmax}_{c_{\min} \le c \le c_{\max}} \log(\rho_{\texttt{EI}}(c))$.
10: **if** $\rho_{\texttt{EI}}(c) < \texttt{EI}_{\min}$ **then**
11: **break**
12: **end if**
13: **end for**
Output: c_0^*, ξ_0^*

Algorithm 3: Bayesian optimization enhanced by the gradient estimated by the twin model method.

The algorithm starts from an initial value of the control variable c, then iterates

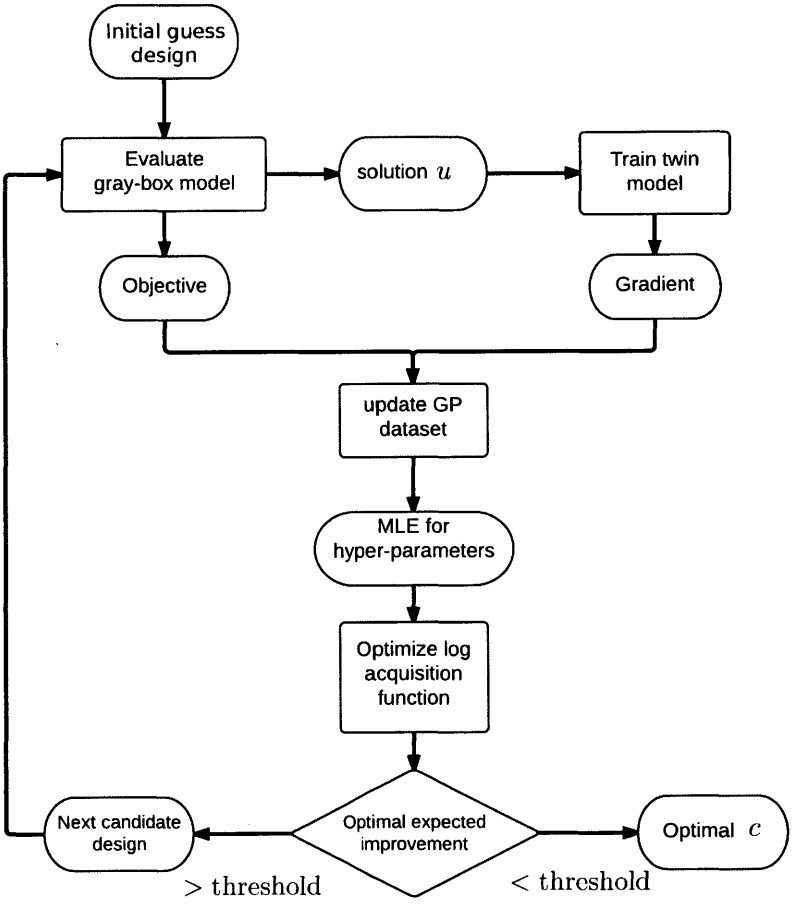


Figure 3-1: Flowchart of Algorithm 3.

over line 2-12 to find the next candidate control. At each iteration, the gray-box model is run at the current control variable, which provides the current objective function $\xi(c)$ and the gray-box solution $\boldsymbol{u}(c)$. The resulting gray-box solution is used to train a twin model according to the twin-model algorithm, Algorithm 2, which provides the estimated gradient $\xi_{\tilde{\nabla}}(c)$. Using the new evaluations of ξ and $\xi_{\tilde{\nabla}}$ at c, the hyperparameters are updated by the maximum likelihood. Then the next candidate control variable is determined according to the expected improvement acquisition function. If the expected improvement at the candidate control is smaller than a threshold value, the optimization exits and reports the best control variable; otherwise the iteration continues until the maximum number of iterations is reached.

In line 3 of Algorithm 3, a new twin model is trained at each iteration for the current control variable. The basis dictionary for the new twin model does not need to be constructed from scratch. The dictionary of the last iteration can be used to provide an initial guess of the dictionary for the current twin model. The bases in the old dictionary may be insignificant for the current twin model; therefore, they should be pruned to give the initial guess. We present a greedy approach to pruning the bases in Algorithm 4. The pruned dictionary is then used as the initial basis dictionary for training the current twin model.

3.3 Convergence Properties Using True Hyperparameters

This section investigates the convergence properties of Algorithm 3. For Bayesian optimization with only the objective function evaluation, the convergence properties have been explored in the literature. Locatelli [73] proved that Bayesian optimization with EI acquisition generates a dense search sequence for the 1-D optimization problem $c^* = \text{argmax}_{c\in[0,1]}\,\xi(c)$, if ξ is a realization of the Wiener process. Vazquez [67] generalized the results by showing that the sequence is still dense for higher dimensional space and for more general classes of stochastic processes. Recently, Bull [68] showed that Bayesian optimization with EI has a convergence rate at $\mathcal{O}(n^{-\nu/d})$, where $\nu > 0$ is

Input: $\mathcal{A}, \alpha_{\mathcal{A}}$ of the trained twin model at the last iteration.

Compute $\overline{\mathcal{M}}$ by k-fold cross validation. $\overline{\mathcal{M}}_0 \leftarrow \overline{\mathcal{M}}$.
while $|\mathcal{A}| > 1$ **do**

$$l^* \leftarrow \underset{l \in \mathcal{N}(\mathcal{A})}{\operatorname{argmin}} s_l(\mathcal{A}) \ , \quad \mathcal{A} \leftarrow \mathcal{A} \backslash \{l^*\}$$

    Compute $\overline{\mathcal{M}}$ by cross validation.
    if $\overline{\mathcal{M}} < \overline{\mathcal{M}}_0$ **then**
        $\overline{\mathcal{M}}_0 \leftarrow \overline{\mathcal{M}}$
    else
        $\mathcal{A} \bigcup \{l^*\}$, **break**
    end if
end while

Output: $\mathcal{A}$

Algorithm 4: Prune the basis dictionary of previously trained twin model. To be consistent, the metric $\mathcal{M}$ is set as the same metric, either $\mathcal{M}_\tau$ or $\mathcal{M}_u$, as in Algorithm 2.

a constant parameter controlling the kernel smoothness, and d is the dimensionality of the control variable. Similar results have been given for UCB acquisition. N. Srinivas [69] shows that the convergence rate for UCB is $\mathcal{O}(n^{-\frac{\nu}{2\nu+d(d+1)}})$. However, to the best of my knowledge, the convergence analysis found in the literature only considers objective function evaluations but not estimated gradient evaluations. My contribution is to extend the convergence analysis to incorporate estimated gradient evaluations. In this section, I analyze the convergence properties of Algorithm 3. I assume that the objective function is a realization of a zero-mean stationary Gaussian process and assume the gradient error is a realizations of another zero-mean Gaussian process. In addition, I assume the kernel functions and the hyperparameters of the Gaussian processes are known. Under the assumptions in Section 3.1, I prove that the search sequence of the control variable is dense in the search space. The conclusion implies that the algorithm is able to find the optimal control as $n_{\max} \to \infty$, regardless of the magnitude of error in the gradient estimation.

The assumptions in Section 3.1 are revisited as follows: ξ belongs to the reproducing kernel Hilbert space (RKHS) $\mathcal{H}_K$ generated by a semi-positive definite kernel K :

$\mathcal{C} \times \mathcal{C} \to [0, \infty)$. Let K be differentiable; then the gradients of all functions in $\mathcal{H}_K$ form a RKHS $\mathcal{H}_{K_\nabla}$ defined by the kernel $K_\nabla(c_1, c_2) \equiv \nabla_{c_1} \nabla_{c_2} K(c_1, c_2)$ for all $c_1, c_2 \in \mathcal{C}$ (Theorem 1 in [66]). Besides, ϵ_i, for $i = 1, \cdots, d$, belongs to the RKHS $\mathcal{H}_G^i$ generated by a semi-positive definite kernel $G_i : \mathcal{C} \times \mathcal{C} \to [0, \infty)$. ϵ_i's are pairwise independent. Denote the tensor product of the RKHSs by $\mathcal{H}_G \equiv \mathcal{H}_G^1 \otimes \cdots \otimes \mathcal{H}_G^d$.

Represent the stochastic dependence of ξ by ω_ξ, and represent the stochastic dependence of ϵ_i by ω_ϵ^i. Let $(\Omega_\xi, \Sigma_\xi, \mathbb{P}_\xi)$ be the probability space for ω_ξ, and let $(\Omega_\epsilon^i, \Sigma_\epsilon^i, \mathbb{P}_\epsilon^i)$ be the probability space for ω_ϵ^i. We have

$$\begin{aligned} \xi \ &: \mathcal{C} \times \Omega_\xi \to \mathbb{R} \\ &(c, \omega_\xi) \to \xi(c; \omega_\xi) \,, \end{aligned} \tag{3.17}$$

and

$$\begin{aligned} \epsilon \ &: \mathcal{C} \times \Omega_\epsilon^i \to \mathbb{R}^d \\ &(c, \omega_\epsilon^i) \to \epsilon(c; \omega_\epsilon^i) \end{aligned} \,, \tag{3.18}$$

for $i = 1, \cdots, d$. Let $\omega_\epsilon = (\omega_\epsilon^1, \cdots, \omega_\epsilon^d)$ and $\Omega_\epsilon = \Omega_\epsilon^1 \otimes \cdots \otimes \Omega_\epsilon^d$. The true objective function is $\xi(c; \omega_\xi^*)$ for $\omega_\xi^* \in \Omega_\xi$. The true estimated gradient error is $\epsilon(c; \omega_\epsilon^*)$ for $\omega_\epsilon^* \in \Omega_\epsilon$. In other words, $\xi(c; \omega_\xi^*) = \xi(c)$ and $\epsilon(c; \omega_\epsilon^*) = \epsilon(c)$ for all $c \in \mathcal{C}$. Conditioned on $\xi(\underline{c}_n)$ and $\xi_{\tilde{\nabla}}(\underline{c}_n)$, Bayesian optimization generates the next search point deterministically. Given the initial control c_{init}, the search sequence can be seen as a mapping

$$\underline{C}(\omega_\xi, \omega_\epsilon) = (C_1(\omega_\xi, \omega_\epsilon), C_2(\omega_\xi, \omega_\epsilon), \cdots) \,, \tag{3.19}$$

The search strategy $\underline{C}$ generates a search sequence $C_1, C_2, \cdots$ in $\mathcal{C}$, with the property that C_{n+1} is $\mathcal{F}_n$-measurable, where $\mathcal{F}_n$ is the σ-algebra generated by $\xi(\underline{c}_n)$ and $\xi_{\tilde{\nabla}}(\underline{c}_n)$. At the n-th search step, the posterior mean and variance of $\xi(c)$ conditioned on $\xi(\underline{c}_n)$ and $\xi_{\tilde{\nabla}}(\underline{c}_n)$ can be written as

$$\hat{\xi}_n(c; \underline{c}_n) = \mathbb{E}_{\omega_\xi, \omega_\epsilon} \left[\xi(c, \omega_\xi) \middle| \underline{c}_n, \xi(\underline{c}_n), \xi_{\tilde{\nabla}}(\underline{c}_n) \right] \,, \tag{3.20}$$

and

$$\sigma_n^2(c;\underline{c}_n) = \mathbb{E}_{\omega_\xi,\omega_\epsilon}\left[\left(\xi(c)-\hat{\xi}_n(c)\right)^2 \left|\underline{c}_n, \xi(\underline{c}_n), \xi_{\bar{\nabla}}(\underline{c}_n)\right.\right] . \tag{3.21}$$

Notice $\sigma_n^2(c;\underline{c}_n)$ only depends on $\underline{c}_n$, and is independent of $\xi(\underline{c}_n), \xi_{\bar{\nabla}}(\underline{c}_n)$ because of the Gaussian process assumption.

The following theorem holds.

Theorem 3. *Let $\Phi(c) \equiv K(c,0)$ for all $c \in \mathcal{C}$, and let $\hat{\Phi}$ be its Fourier transform. If there exist $C \geq 0$ and $k \in \mathbb{N}^+$, such that $(1+|\eta|^2)^k\left|\hat{\Phi}(\eta)\right| \geq C$ for all $\eta \in \mathbb{R}^d$, and if $n_{\max} \to \infty$ and $EI_{\min} = 0$, then $\underline{c}_n$ is dense in $\mathcal{C}$ for all $c_{init} \in \mathcal{C}$, all $\xi \in \mathcal{H}_K$ and all $\epsilon \in \mathcal{H}_G$.*

Proof:

First, we have the following lemma (Chapter 1, Theorem 4.1, [70]).

lemma 1. *Let K_1, K_2 be the reproducing kernels of functions on $\mathcal{C}$ with norms $\|\cdot\|_{\mathcal{H}_1}$ and $\|\cdot\|_{\mathcal{H}_2}$ respectively. Then $K = K_1 + K_2$ is the reproducing kernel of the space*

$$\mathcal{H} = \mathcal{H}_1 \oplus \mathcal{H}_2 = \{f = f_1 + f_2,\ f_1 \in \mathcal{H}_1,\ f_2 \in \mathcal{H}_2\}$$

with norm $\|\cdot\|_{\mathcal{H}}$ defined by

$$\forall f \in \mathcal{H} \quad \|f\|_{\mathcal{H}}^2 = \min_{f=f_1+f_2,\ f_1\in\mathcal{H}_1, f_2\in\mathcal{H}_2} \left(\|f_1\|_{\mathcal{H}_1}^2 + \|f_2\|_{\mathcal{H}_2}^2\right)$$

Using Lemma 1, we prove the following Cauchy-Schwarz inequality,

lemma 2.

$$\left|\xi(c,\omega_\xi) - \hat{\xi}(c;\underline{c}_n)\right|^2 \leq$$
$$\left(\left(1+\frac{4d}{3}\right)\|\xi(c;\omega_\xi)\|_{\mathcal{H}_K} + \frac{4d}{3}\|\nabla_c\xi(c;\omega_\xi)\|_{\mathcal{H}_{K_\nabla}} + \frac{4}{3}\sum_{i=1}^{d}\left\|\epsilon_i(c;\omega_\epsilon^i)\right\|_{\mathcal{H}_G^i}\right)\sigma^2(c;\underline{c}_n)$$

To prove Lemma 2, we define a vector

$$u = (u_1, \cdots, u_d)^T \in \mathcal{U},$$

where $\mathcal{U} = [0,1]^d$. Define an auxiliary function

$$\mathcal{Y}(c, u; \omega_\xi, \omega_\epsilon) = \left(1 - \sum_{i=1}^{d} u_i\right) \xi(c, \omega_\xi) + u^T \left[\nabla_c \xi(c, \omega_\xi) + \epsilon(c; \omega_\epsilon)\right] .$$

$u_1, \cdots, u_d$ are functions from the Sobolev space $W^{1,2}$ defined on $\mathcal{U}$, equipped with the inner product

$$\langle \phi, \psi \rangle = \int_{\mathcal{U}} \phi\psi + (\nabla\phi)^T (\nabla\psi)\, du,$$

The Sobolev space is a RKHS with the kernel

$$K_u(\phi, \psi) = \frac{1}{2} \exp\left(-|\phi - \psi|\right)$$

on $\mathcal{U} = [0,1]$. Given ω_ξ and ω_ϵ, $\mathcal{Y}(\cdot, \cdot; \omega_\xi, \omega_\epsilon)$ can be viewed as a realization from a RKHS $\mathcal{H}_\mathcal{Y}$, defined on $\mathcal{C} \times \mathcal{U}$. Let the kernel function of $\mathcal{H}_\mathcal{Y}$ be

$$\begin{aligned} K_\mathcal{Y} &: \mathcal{C} \times \mathcal{U}, \mathcal{C} \times \mathcal{U} \to \mathbb{R} \\ &(c_1, u_1), (c_2, u_2) \to K_\mathcal{Y}((c_1, u_1), (c_2, u_2)) \end{aligned}$$

Notice

$$\mathcal{Y}(c, \mathbf{0}; \omega_\xi, \omega_\epsilon) = \xi(c, \omega_\xi)$$

is the objective function, and

$$\left(\mathcal{Y}(c, e_1; \omega_\xi, \omega_\epsilon), \cdots, \mathcal{Y}(c, e_d; \omega_\xi, \omega_\epsilon)\right) = \nabla_c \xi(c; \omega_\xi) + \epsilon(c; \omega_\epsilon)$$

is the estimated gradient, where $e_i, i = 1, \cdots, d$ indicates the ith unit Cartesian basis vector in $\mathbb{R}^d$. Conditioned on the samplings $\xi(\underline{c}_n)$ and $\xi_{\tilde{\nabla}}(\underline{c}_n)$, we can bound the error

of the estimation of $\mathcal{Y}(c, \mathbf{0}; \omega_\xi, \omega_\epsilon)$ by the Cauchy-Schwarz inequality [70] in $\mathcal{H}_\mathcal{Y}$,

$$\left|\mathcal{Y}(c, \mathbf{0}; \omega_\xi, \omega_\epsilon) - \hat{\mathcal{Y}}(c, \mathbf{0}; \underline{c}_n)\right| = \left|\xi(c; \omega_\xi) - \hat{\xi}_n(c; \underline{c}_n)\right| \leq \sigma(c; \underline{c}_n)\|\mathcal{Y}\|_{\mathcal{H}_\mathcal{Y}}$$

Besides,

$$\begin{aligned}
\|\mathcal{Y}\|_{\mathcal{H}_\mathcal{Y}} &= \left\| \left(1 - \sum_{i=1}^{d} u_i\right) \xi(c; \omega_\xi) + u^T \left[\nabla_c \xi(c; \omega_\xi) + \epsilon(c; \omega_\epsilon)\right] \right\|_{\mathcal{H}_\mathcal{Y}} \\
&\leq \|\xi(c; \omega_\xi)\|_{\mathcal{H}_K} + \left(\sum_{i=d}^{d} \|u_i\|_{\mathcal{H}_u}\right) \|\xi(c; \omega_\xi)\|_{\mathcal{H}_K} + \left(\sum_{i=d}^{d} \|u_i\|_{\mathcal{H}_u}\right) \|\nabla_c \xi(c; \omega_\xi)\|_{\mathcal{H}_{K_\nabla}} \\
&\quad + \sum_{i=1}^{d} \left\|u_i \epsilon_i(c; \omega_\epsilon^i)\right\|_{\mathcal{H}_u \otimes \mathcal{H}_G^i} \\
&= \|\xi(c, \omega)\|_{\mathcal{H}_K} + \frac{4d}{3} \|\xi(c, \omega)\|_{\mathcal{H}_K} + \frac{4d}{3} \|\nabla_c \xi(c; \omega_\xi)\|_{\mathcal{H}_{K_\nabla}} + \frac{4}{3} \sum_{i=1}^{d} \left\|\epsilon_i(c; \omega_\epsilon^i)\right\|_{\mathcal{H}_G^i} ,
\end{aligned}$$

where the inequality obtained by Lemma 1. The proof for Lemma 2 completes.

Using Lemma 2, we prove

lemma 3. *Let $(\underline{c}_n)_{n\geq 1}$ and $(\underline{a}_n)_{n\geq 1}$ be two sequences in $\mathcal{C}$. Assume that the sequence (a_n) is convergent, and denote by a^* its limit. Then each of the following conditions implies the next one:*

1. *a^* is an adherent point of $\underline{c}_n$ (there exists a subsequence in $\underline{c}_n$ that converges to a^*),*
2. *$\sigma^2(a_n; \underline{c}_n) \to 0$ when $n \to \infty$,*
3. *$\hat{\xi}(a_n; \underline{c}_n) \to \xi(a^*, \omega)$ when $n \to \infty$, for all $\xi \in \mathcal{H}_K$, $\epsilon \in \mathcal{H}_G$.*

The proof of Lemma 3 is the similar as the proposition 8 in [67], except that the Cauchy-Schwarz inequality used in the paper is replaced by Lemma 2. We do not repeat the proof but refer to [67] for the details.

Next, we show the three conditions are equivalent in Lemma 3. Using the assumption:

There exist $C \geq 0$ *and* $k \in \mathbb{N}^+$, *such that* $(1+|\eta|^2)^k \left|\hat{\Phi}(\eta)\right| \geq C$ *for all* $\eta \in \mathbb{R}^d$, we have, for any $\xi \in \mathcal{H}_K$ and its Fourier transform $\hat{\xi}$,

$$\|\xi\|_{W^{k,2}} = \int (1+|\eta|^2)^k |\hat{\xi}|^2 \, d\eta \geq C \int \left|\hat{\Phi}(\eta)\right|^{-1} |\hat{\xi}(\eta)|^2 \, d\eta = C\sqrt{(2\pi)^d} \|\xi\|_{\mathcal{H}_K},$$

where $W^{k,2}$ is the Sobolev space whose weak derivatives up to order k have a finite L^2 norm [68]. Therefore, $W^{k,2} \subseteq \mathcal{H}_K$. The result can be extended to $\xi \in \mathcal{H}_K(\mathcal{C})$ defined on the domain $\mathcal{C} \in \mathbb{R}^d$, because $\mathcal{H}_K(\mathcal{C})$ embeds isometrically into $\mathcal{H}_K(\mathbb{R}^d)$ [80]. Besides, we have that C_c^∞ is dense in $W^{k,2}$ (Chapter 2, Lemma 5.1 [81]), where C_c^∞ is the C^∞ functions with compact support on $\mathcal{C}$. As a consequence, $\mathcal{C}_c^\infty \subseteq \mathcal{H}_K$ [67]. If the condition 1 is false, then there exist a neighborhood U of a^* that does not intersect $\underline{c}_n$. There exist $\xi \in \mathcal{H}_K$ that is compactly supported in U, and $\epsilon = \mathbf{0}$, such that $\hat{\xi}(a^*; \underline{c}_n) = 0$ whereas $\xi(a^*) \neq 0$, which violates the condition 3. Therefore, the three conditions in Lemma 3 are equivalent.

Finally, we have:

lemma 4. (E. Vazquez, Theorem 5 [67]) *If the three conditions in Lemma 3 are equivalent,* $n_{\max} \to \infty$, *and* $EI_{\min} = 0$, *then for all* $c_{init} \in \mathcal{C}$ *and all* $\omega \in \mathcal{H}$, *the sequence* $\underline{c}_n$ *generated by the Bayesian optimization with expected improvement acquisition is dense in* $\mathcal{C}$.

We refer to [67] for the details of the proof and do not repeat it here. To summarize, under the conditions in Theorem 3, $\underline{c}_n$ is dense in the search space. ■

Under the condition that the true hyperparameters are known, the theorem implies that the Bayesian optimization algorithm (Algorithm 3) can find the maximum of the objective function in the limiting case of $n_{\max} \to \infty$ and $EI_{\min} = 0$, regardless of the error of the gradient estimation. Even if the gradient is poorly estimated, the current best control is always close to the true optimal and the optimization won't get stuck at a non-optimal control. The assumption of $(1+|\eta|^2)^k \left|\hat{\Phi}(\eta)\right| \geq C$ in the theorem indicates that the Fourier transform of $K(\cdot, c)$ has at least polynomial growth, which is a condition satisfied by the Matern 5/2 kernel [67]. It is a future work to extend

the theory to unknown and estimated hyper parameters.

3.4 Numerical Results

This section demonstrates the optimization algorithm on several numerical examples.

3.4.1 Buckley-Leverett Equation

Consider the same gray-box model of Buckley-Leverett equation in Section 2.4.1

$$\frac{\partial u}{\partial t} + \frac{\partial}{\partial x}\left(\frac{u^2}{1+2(1-u)^2}\right) = c\,,$$

with the periodic boundary condition and the same initial condition shown in Figure 2-11. We parameterize the control c by

$$\begin{aligned} c(t,x) &= \sum_{i=1}^{5}\sum_{j=1}^{5} c_{ij}\, B_{ij}(t,x) \\ B_{ij} &= \exp\left(-\frac{(t-t_i)^2}{L_t^2}\right)\exp\left(-\frac{(x-x_j)^2}{L_x^2}\right) \end{aligned}\,, \tag{3.22}$$

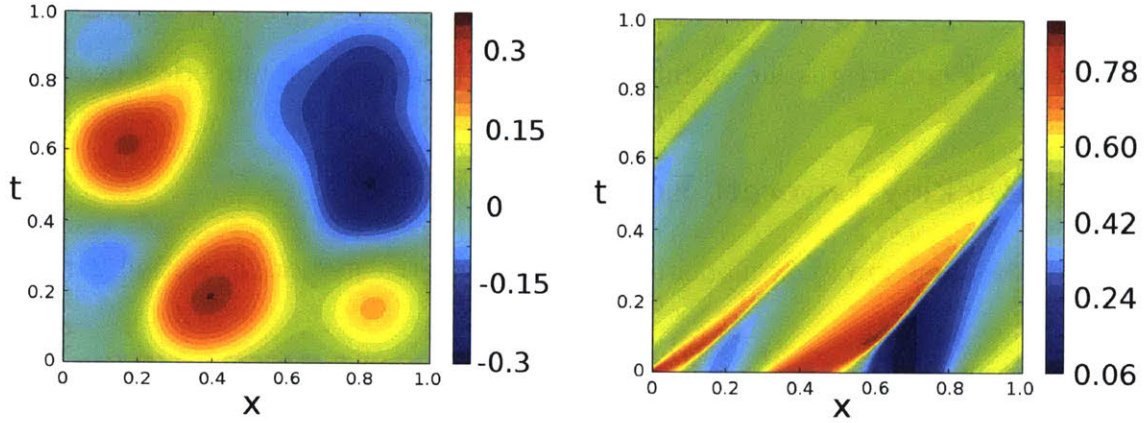
where $L_t = L_x = 0.15$, and $(t_1, \cdots, t_5) = (x_1, \cdots, x_5) =$ `linspace(0,1,5)`. Consider minimizing the objective

$$\xi(c) = \int_{x=0}^{1} \left| u(t=1,x) - \frac{1}{2} \right|^2 + \frac{1}{100}\sum_{ij} c_{ij}^2\,, \tag{3.23}$$

with the bound constraints $-1 \le c_{ij} \le 1$ for $i, j = 1, \cdots, 5$.

The optimization is done by using Algorithm 3. Figure 3-2a shows the optimized source term. Figure 3-2b shows the corresponding gray-box solution. Constrained by a limited number of gray-box simulations, the optimized solution and objective are examined. Figure 3-3 compares the optimized $u(t = 1, x)$ obtained by using the Bayesian optimization with and without the estimated gradient, after 20 gray-

box simulations. Figure 3-4 shows the current best (minimal) objective evaluation at each iteration. The usage of twin-model gradient makes the objective evaluation decrease faster, especially when the number of iteration is small.



(a) Optimized source term. *(b) Optimized gray-box solution.*

Figure 3-2: Optimized results for the Buckley-Leverett equation.

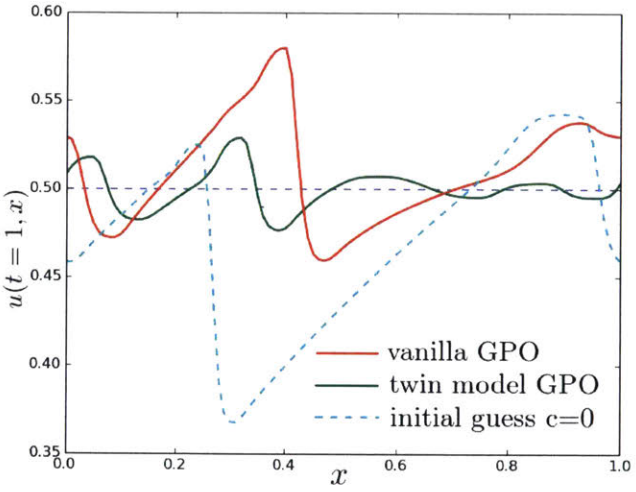


Figure 3-3: A comparison of the current best $u(t = 1, x)$ after 20 gray-box simulations. The red line is obtained by the vanilla Bayesian optimization and the green line by the twin-model Bayesian optimization. The cyan dashed line is the $u(t = 1, x)$ obtained by setting the source term to zero.

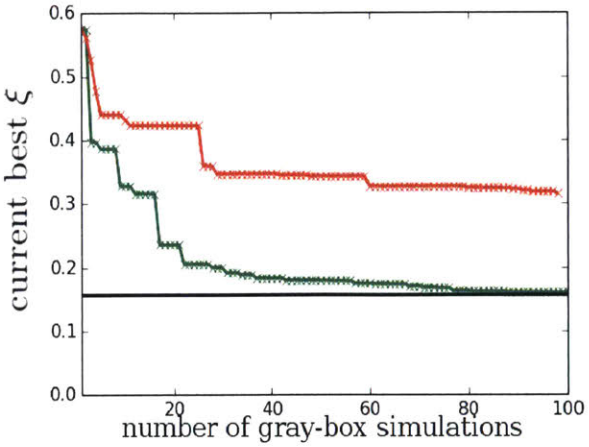


Figure 3-4: The current best objective evaluation at each iteration. The red line is obtained by the Bayesian optimization without using the estimated gradient. The green line is obtained by using the estimated gradient. The black horizontal line indicates the true optimal, which is obtained by BFGS using the true adjoint of the Buckley-Leverett equation.

3.4.2 Navier-Stokes Flow

Consider the same Navier-Stokes flow in Section 2.4.2. Let $S(c)$ be the area of the return bend. $S(c)$ is a function of the control points' coordinates represented by c. Let $S_0 = \frac{1}{2}\pi(1.25^2 - 0.25^2)$. The objective function is the steady-state mass flux with a penalty term representing the difference of S and S_0,

$$\xi(c) = \int_{\text{outlet}} \rho_\infty u_\infty \big|_{\text{outlet}} \, dy - \lambda(S - S_0)^2 \,, \tag{3.24}$$

where $\lambda > 0$. The goal is to maximize $\xi(c)$ in a bounded domain $c_{\min} \leq c \leq c_{\max}$ shown by the blue boxes in Figure 3-5. There are four variable control points at each boundary, where each control point has the x- and y-coordinates. Thus, the control is 16 dimensional. Figures 3-5 (a) and (c) show the initial and the optimized geometries for the van der Waals and the Redlich-Kwong gasses. Figures 3-5 (b) and (d) show the corresponding pressure profiles at the interior and the exterior boundaries along the streamwise direction. The optimized geometry reduces the adverse pressure gradient at the flow separation and thus decreases the drag and increases the mass flux.

Figure 3-6 shows the geometry, the trained state equation, and the basis dictionary for the first five iterations and the optimized result of the Bayesian optimization. In this numerical example, the trained state equation and the basis dictionary are similar at each Bayesian optimization iteration. Figure 3-7 shows the current best objective evaluation at each iteration. The twin-model estimated gradient enables faster improvement of the objective evaluation. Figure 3-8 shows the wall-clock time of the optimization against the number of iterations. Although the twin model increases the computational cost per iteration, the increased cost is offset by the faster improvement of the objective function.

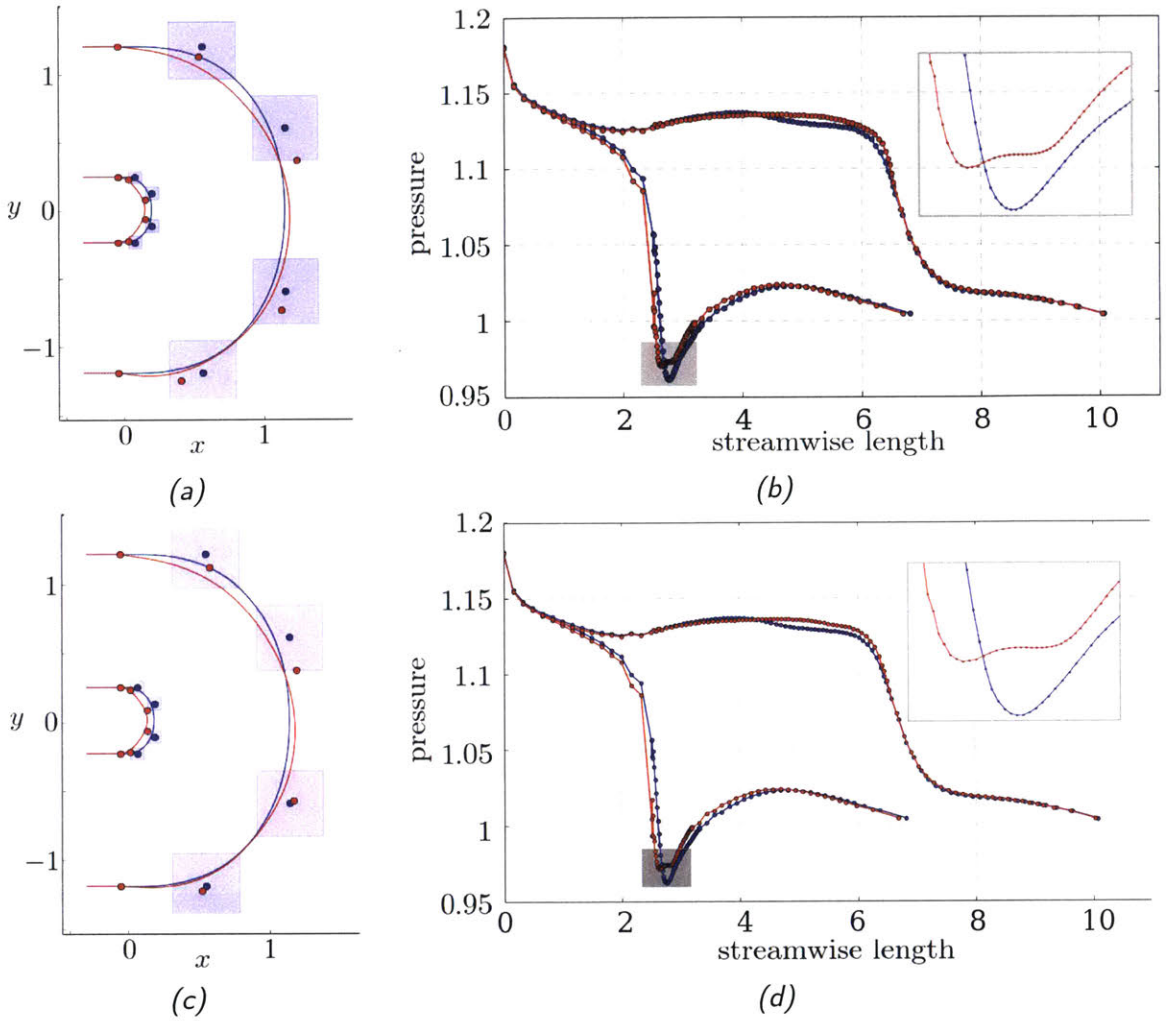
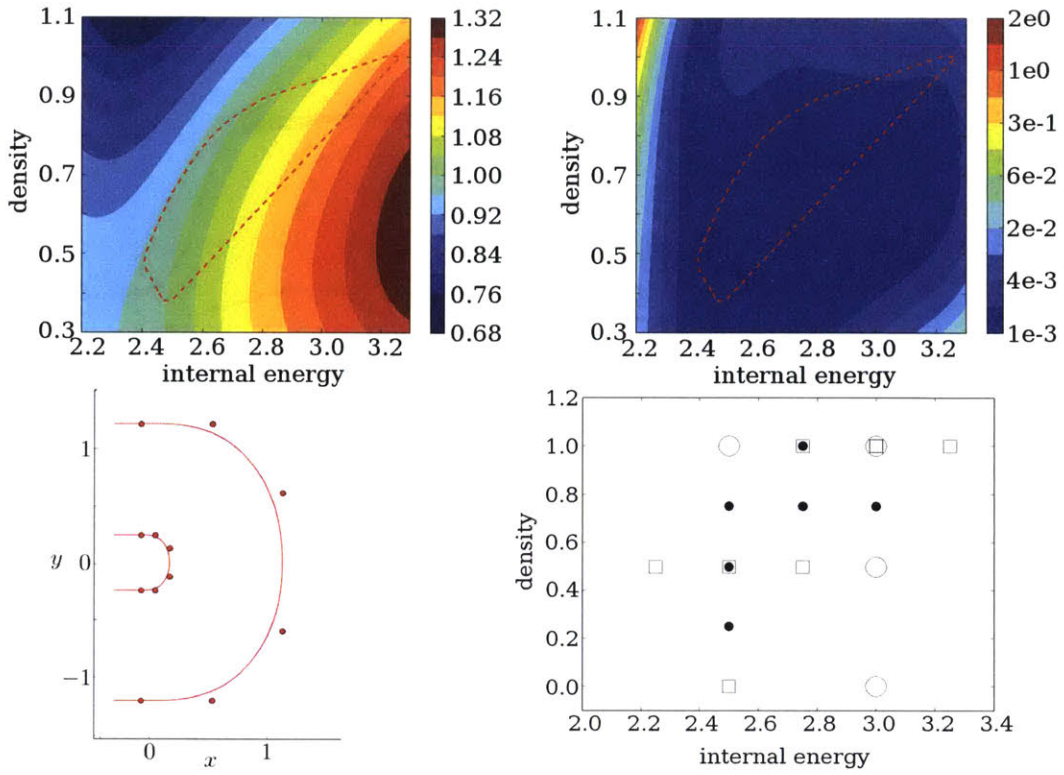
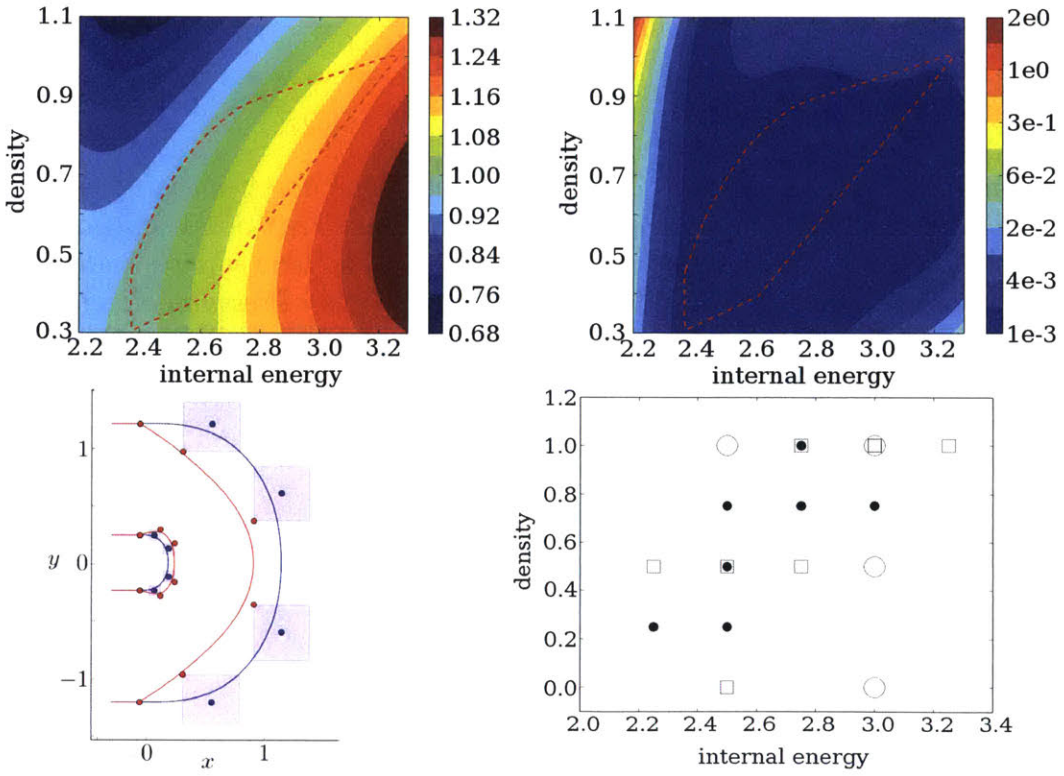


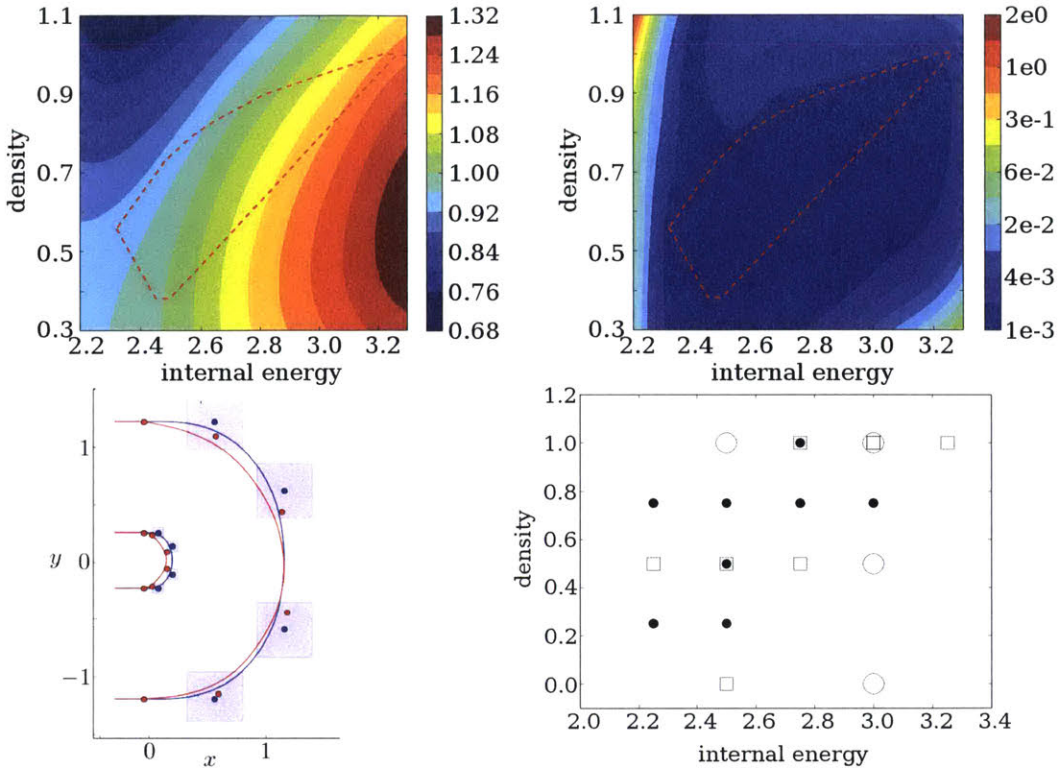
Figure 3-5: (a) Initial guess of control points (blue dots); initial guess of the geometry (blue line); optimized control points (red dots); optimized geometry (red line) for the van der Waals gas. The blue squares indicate the bound constraints for each control point. (b) Pressure along the interior and the exterior boundaries for the initial (blue) and the optimized (red) geometry. (c) and (d) Results for the Redlich-Kwong gas.



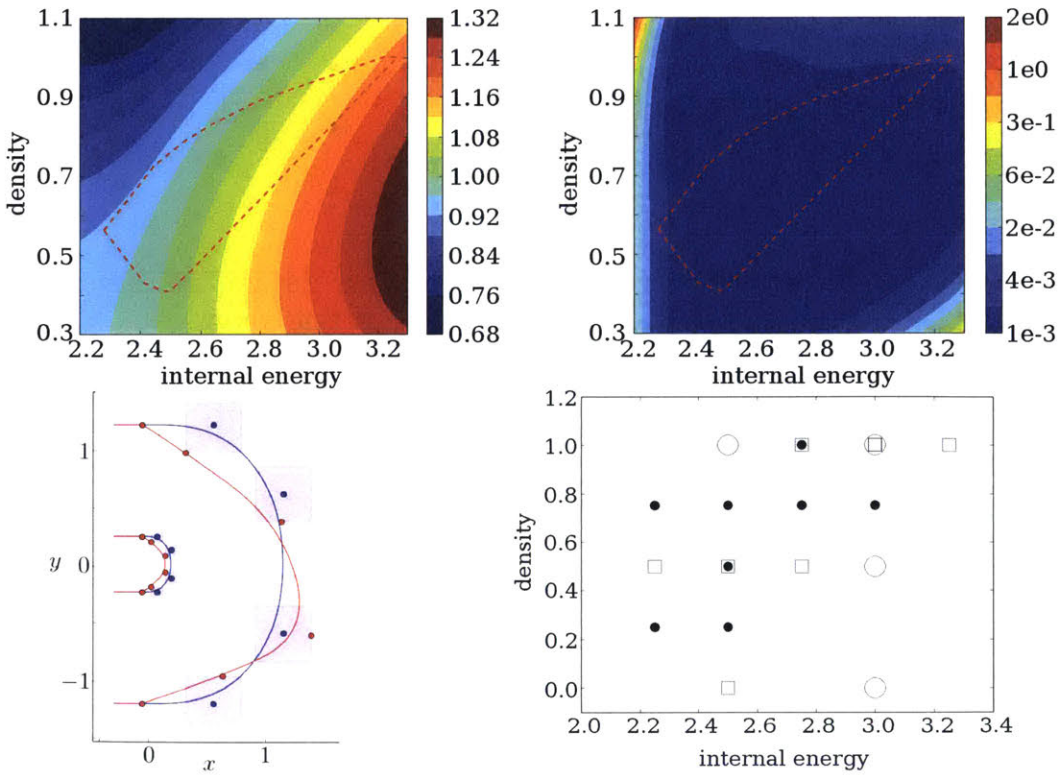
(a) Iteration 0



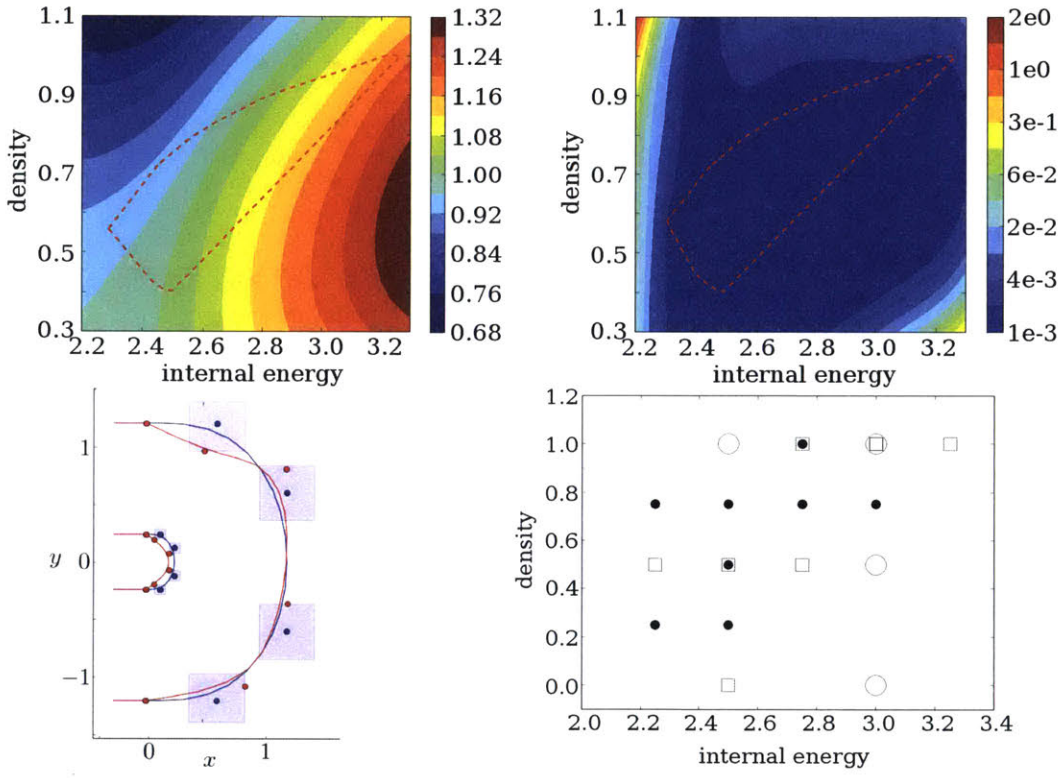
(b) Iteration 1



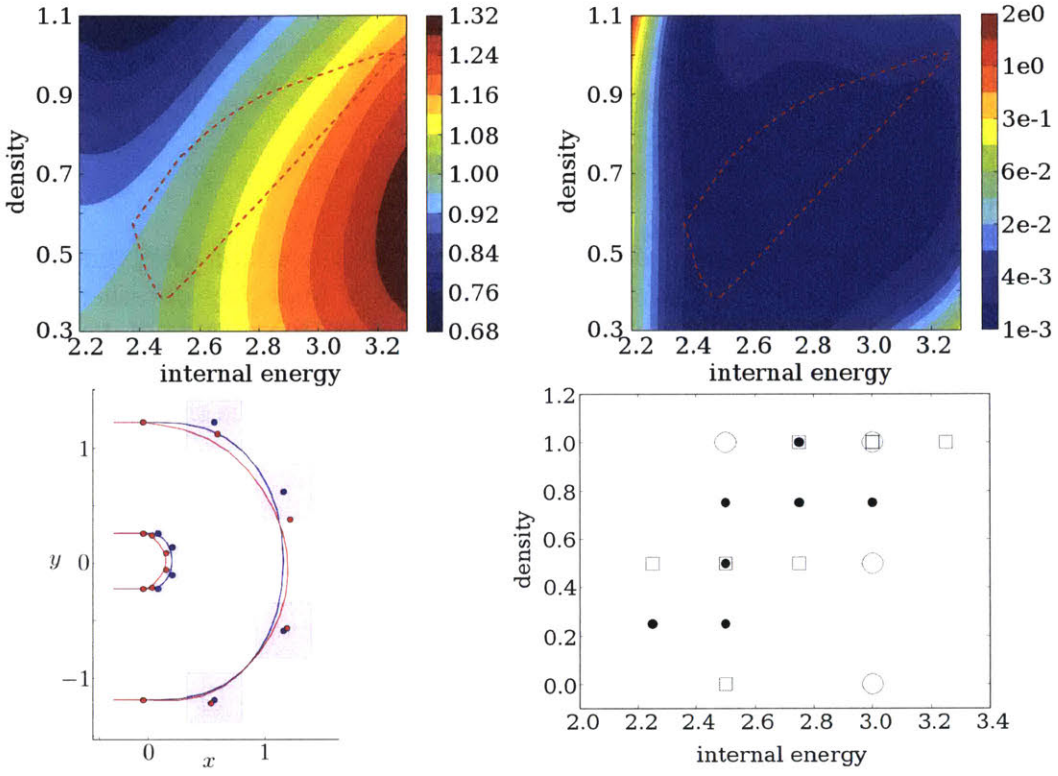
(c) Iteration 2



(d) Iteration 3



(e) Iteration 4



(f) Optimized

Figure 3-6: Trained state equation, the error of the trained state equation, the return bend geometry, and the basis dictionary at some iterations of the Bayesian optimization. The gray-box model uses the Redlich-Kwong state equation. The resolution of the bases is represented using the same symbols as in Figure 2-20.

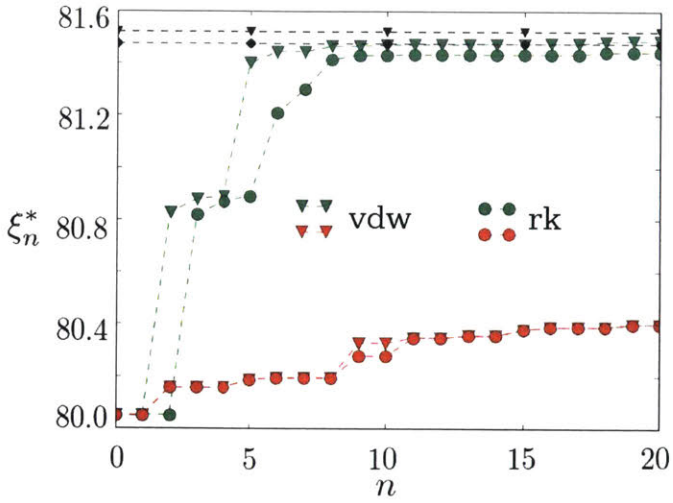


Figure 3-7: Current best objective at each iteration for the van der Waals gas and the Redlich-Kwong gas. Green lines are obtained by the twin-model Bayesian optimization. Red lines are obtained by the Bayesian optimization without the estimated gradient. Black horizontal lines indicate the true optimal, obtained by BFGS using the true gradient.

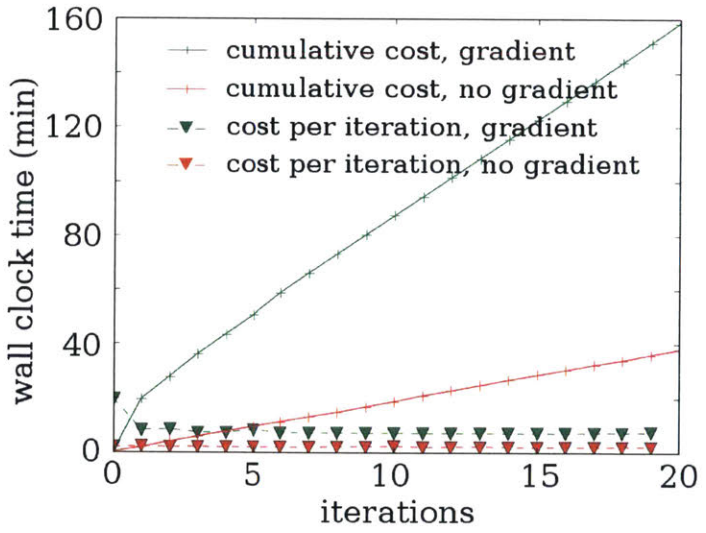


Figure 3-8: Cumulative and per-iteration wall clock time, in minutes. Although the twin-model Bayesian optimization is costlier per-iteration due to the training of twin model, it achieves near-optimality with less overall computational time. The gray-box model uses the Redlich-Kwong state equation.

3.4.3 Polymer Injection in Petroleum Reservoir

Consider a 2D horizontal reservoir governed by (2.49)

$$\begin{aligned} \frac{\partial}{\partial t}(\rho_\alpha \phi S_\alpha) + \nabla \cdot (\rho_\alpha \vec{v}_\alpha) = 0\,, \quad \alpha \in \{w, o\} \\ \frac{\partial}{\partial t}(\rho_w \phi S_w c) + \nabla \cdot (c\rho \vec{v}_{wp}) = 0 \end{aligned},$$

and (2.50)

$$\begin{aligned} \vec{v}_\alpha &= -M_\alpha k_{r\alpha} \boldsymbol{K} \cdot (\nabla p - \rho_w g \nabla z), \quad \alpha \in \{w, o\} \\ \vec{v}_{wp} &= -M_{wp} k_{rw} \boldsymbol{K} \cdot (\nabla p - \rho_w g \nabla z) \end{aligned}.$$

The permeability is heterogeneous and is shown in Figure 3-9. Five injectors are placed along the southern boundary, and one producer is placed in the northeastern corner. The reservoir is simulated for $t \in [0, T = 10]\,\mathtt{day}$.

Time-independent control

First, consider constant-in-time injection rates at the injectors. Let the price of unit mass oil be $e^{-\frac{t}{\tau}}$, which decays over time, where $\tau > 0$ is a constant. Let the price of unit mass water be $0 < \lambda < 1$. Define

$$\xi(t) = \left(\int_0^t \rho_{\mathrm{prod}\,o} S_o e^{-\frac{t}{\tau}} I_{\mathrm{prod}} \mathrm{d}t - \int_\Omega \rho_o(t=0) \phi S_o(t=0) \mathrm{d}x \right) - \lambda t \sum_{i=1}^{5} \rho_{\mathrm{inj}wi} I_{\mathtt{inj}i}\,, \tag{3.25}$$

which represents the price of the produced oil produced minus the price of all residual oil at $t = 0$ and the price of the total water injected. $\rho_{\mathrm{prod}o}$ is the oil phase density at the producer, $\rho_{\mathrm{inj}wi}$'s are the water phase densities at the injectors, I_{prod} is the production rate at the producer, and $I_{\mathtt{inj}i}$'s are the injection rates at the ith injector. The goal is to maximize $\xi(T)$ with bound constraints on the injection rates $0 \leq I_{\mathtt{inj}i} \leq I_{\max}$. Since there are five injectors, the optimization is five-dimensional.

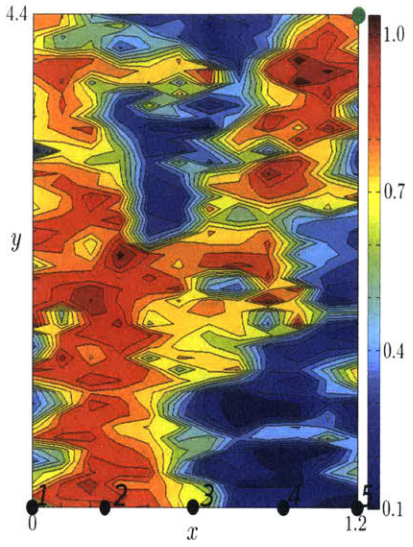


Figure 3-9: Permeability of the reservoir, in 100 milli Darcy. The five injectors are indicated by the black dots, and the producer is indicated by the green dot.

Figure 3-10 shows the current best objective evaluation against the number of iterates. The black line indicates the true optimal[2]. The twin-model Bayesian optimization achieves near-optimality faster than the vanilla Bayesian optimization without using the gradient information. Figure 3-11 shows $\xi(t)$ for the initial and the optimized injection rates. The initial rates are set at $I_{\mathtt{inj}i} = I_{\max}$ for all injectors, which results in early water breakthrough and high water cut. Although the profit is high at smaller t, it deteriorates for larger t due to the water being wasted.

Time-dependent control

Second, consider time-dependent injection rates. If $[0, T]$ is discretized uniformly into 200 segments, each $I_{\mathtt{inj}i}$ becomes a vector with a length of 200. Thus the optimization is 1000-dimensional. Clearly the Bayesian optimization algorithm developed in Section 3.2 is not long suitable because the large dimensionality leads to a huge covariance

[2]The true optimal is obtained by COBYLA [48] after running 192 gray-box simulations.

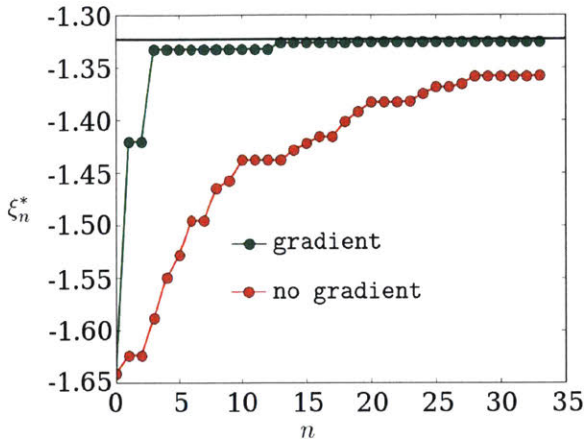


Figure 3-10: Current best objective evaluation against the number of iterates. The black line indicates the true optimal obtained by COBYLA optimization [48], a derivative-free optimization method.

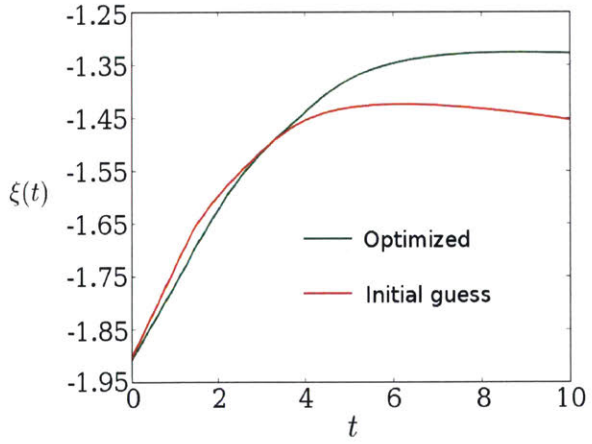


Figure 3-11: $\xi(t)$ for the initial and the optimized injection rates.

matrix[3]. Instead, the twin model is tested on a simple gradient descent method, the backtracking Armijo gradient descent method [104]. The method is a gradient descent method whose stepsize at the lth iteration is determined by Algorithm 5, the backtracking-Armijo line search [104]

Input: Initial stepsize α_0. $0 < \beta < 1$, $0 < \tau < 1$.
1: **for** $l = 1$ **to** $l_{\max}$ **do**
2: $\alpha_{l+1} = \gamma\alpha_0$, $l = l + 1$.
3: **if** $\xi(c + \alpha_{l+1}\nabla\xi) - \xi(c) \geq \beta\alpha_{l+1}\nabla\xi^T \cdot \nabla\xi$ **then**
4: **break**
5: **end if**
6: **end for**
Output: α_l

Algorithm 5: Determine the stepsize in the gradient descent optimization by the backtracking-Armijo line search [104].

Figure 3-12 shows the optimized injection rates. The first and fifth injectors, at the southeastern and southwestern corners, are turned on first. The rate at injector 5 is particularly large, possibly because the permeability is relatively low. Once water breaks through and a low-resistance water channel forms, the oil around the injector 5 will be more difficult to extract. Later, all injectors are turned on, and their rates gradually decrease when the water cut increases. Figure 3-13 shows the current best objective evaluation against the number of iterates. Using the time-dependent control, the objective evaluation gets more improvement than the constant-rate control.

3.5 Chapter Summary

Based up previous research, this chapter develops a Bayesian framework for the optimization problems constrained by gray-box conservation law simulations. Gaussian process models are presented for the objective function, the true gradient, the estimated gradient, and the gradient error. Using the Gaussian process models, the formulation of the joint and the posterior distributions is given, where the hyperparameters are

[3]As aforementioned, the covariance matrix for evaluating the posterior is $N(d+1)$-by-$N(d+1)$. For example, after 100 iterates, the matrix becomes 10^5-by-10^5. The optimization algorithm can dominate the computational cost instead of the conservation law simulation, which violates my assumptions in Chapter 1. The problem of scaling is generally suffered by non-parametric methods.

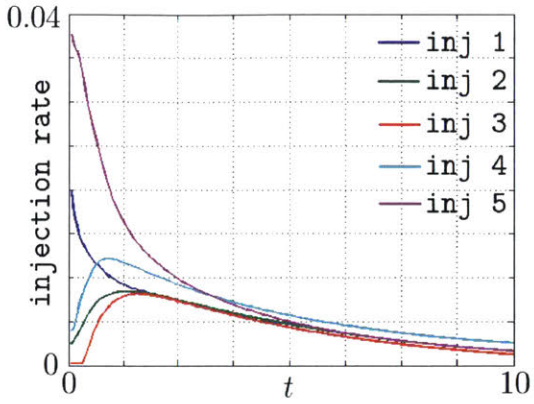


Figure 3-12: Optimized time-dependent injection rates.

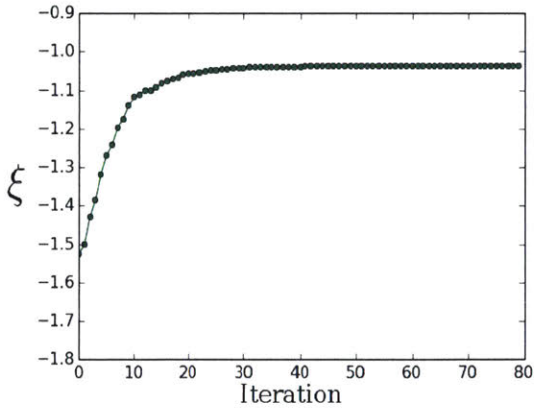


Figure 3-13: Current best objective evaluation using the backtracking-Armijo gradient descent method, where the gradient is provided by the twin model.

estimated by maximum likelihood. The developments are summarized in a Bayesian optimization algorithm which leverages the twin-model gradient estimation. In addition, the convergence property of the algorithm is theoretically studied. The algorithm is guaranteed to find the optimal regardless of the gradient estimation accuracy if the true hyperparameters are used. It is a future work to extend the theory to estimated hyper parameters.

The proposed optimization method is demonstrated by several numerical examples. The first example is the Buckley-Leverett equation whose flux is assumed unknown. The objective function is optimized by adjusting the source term represented by 25 control variables. The second example is a Navier-Stokes flow in a return bend, where the state equation is unknown. The mass flux with a penalty on the geometry is maximized by adjusting the flow boundaries which are controlled by 16 variables. The third example is a petroleum reservoir with polymer injections, where the mobility factors are unknown. The profit is maximized by adjusting the constant-time injection rates at five injectors. In all three examples, the twin-model optimization achieves near-optimality with fewer iterations than the vanilla Bayesian optimization. Finally, the time-dependent control is considered on the same petroleum reservoir example, which yields a 1000-dimensional problem. Conventionally, such high-dimensional optimization can be difficult without the adjoint gradient. The twin-model gradient is tested to work well using a gradient descent approach.

Chapter 4

Conclusions

In this thesis, I addressed the optimization constrained by gray-box simulations. I enabled the adjoint gradient computation for gray-box simulations by leveraging the space-time solution. In addition, I utilized the gradient information in a Bayesian framework to facilitate a more efficient optimization. To conclude, this chapter summarizes the developments and highlights the contributions of this work. I close with suggestions for continuing work on this topic.

4.1 Thesis Summary

Optimization constrained by conservation law simulations is prevalent in many engineering applications. In many cases, the simulator can be legacy and lack the adjoint capability. Chapter 1 categorizes such simulators as gray-box. The gray-box scenario limits the efficient application of gradient-based optimization methods. I motivate the need for the adjoint gradient and explain the feasibility of estimating the adjoint gradient in the gray-box scenario. The key is to leverage the gray-box space-time solution, which contains information of the gray-box simulator but is usually abandoned by conventional optimization methods. To restrict the scope of my thesis, a class of problems is formulated where the flux functions are partially unknown.

To address this issue, an adjoint-enabled twin model is proposed to match the

space-time solution. In Chapter 2, I develop a two-stage procedure to estimate the gradient. In the first stage, a twin model is trained to minimize the solution mismatch. In the second stage, the trained twin model computes an adjoint gradient which approximates the true gray-box gradient. For a simple conservation law with only one equation and one-dimensional space, I demonstrate theoretically that the twin model can indeed infer the gray-box conservation law on a domain that has large solution variation. To numerically implement the unknown part of the flux function is parameterized by a set of bases. I argue that the sigmoid bases are well suited for this problem because their gradients are local. The procedure is demonstrated on a Buckley-Leverett equation using a set of sigmoids chosen manually. Although the estimated gradient is accurate, several limitations are observed, which lead to the developments of adaptive basis construction. Several tools are introduced for the adaptive basis construction, including a metric of the basis significance, the basis neighborhood, and the cross validation. The adaptive basis construction fully exploits the information contained in the gray-box solution and avoids the problem of overfitting. Based upon these developments, a twin-model algorithm is presented. The algorithm selects the basis dictionary adaptively using a forward-backward iteration procedure, where either the solution mismatch or the integrated truncation error can be used as the metric for basis selection. The twin-model algorithm is demonstrated on several numerical examples: a 1D convection equation with unknown flux function, a 2D steady-state Navier-Stokes flow with unknown state equation, and a 3D petroleum reservoir flow with unknown mobility factors. In all the three examples, the twin model algorithm provides an accurate estimation of the true gradient, which represents a major contribution towards enabling the adjoint gradient computation for gray-box simulations.

Using the twin-model gradient, optimization can be done more efficiently. Chapter 3 incorporates the twin-model gradient into a Bayesian optimization framework, in which the objective function, the true gradient, the estimated gradient, and the gradient error are modeled by Gaussian processes. The model provides analytical

expressions for the posterior distributions and the acquisition function, while the hyperparameters are estimated by maximum likelihood. I present a Bayesian optimization algorithm that utilizes the twin-model gradient. In addition, I show that the algorithm can find the optimal control regardless of the gradient estimation accuracy if the true hyper parameters are used. The optimization algorithm is demonstrated on several similar problems discussed in Chapter 2: a Buckley-Leverett equation with source term controls, a Navier-Stokes flow in a return bend with boundary geometry controls, and a petroleum reservoir with polymer-water injection rate controls. In all the three examples, the twin-model optimization achieves near-optimality with fewer iterations than the vanilla Bayesian optimization without the gradient information, which represents another major contribution of my thesis. Finally, the twin-model gradient is tested on a high-dimensional control problem, by employing a simple gradient descent approach. The gradient efficiently enables the optimization of the high-dimensional problem.

4.2 Contributions

The main contributions of this work are:

1. a twin-model algorithm that enables the adjoint gradient computation for gray-box conservation law simulations;

2. an adaptive basis construction scheme that fully exploits the information of gray-box solutions and avoids overfitting;

3. a Gaussian process model of the twin-model gradient and a Bayesian optimization algorithm that employs the twin model; and

4. numerical demonstrations of the algorithms in several examples: the Buckley-Leverett equation, the Navier-Stokes equation, and the porous media flow equation.

4.3 Future Work

There are several potential directions of further research: A useful extension is to consider unknown source terms. My thesis is limited to flux functions that contains unknown algebraic dependence on the state variables. However, many industrial simulations have unknown source terms, which are not accommodated by my method. Being able to accommodate unknown source terms can extend the applicability of the twin-model method. Another useful extension is to investigate the inferability of twin models for various conservation laws. In particular, Theorem 1 may be extended to more general problems with a system of equations and higher spatial dimension. Besides, in the twin-model Bayesian optimization algorithm, it is of great practical value to reuse the twin model more efficiently. My current approach uses the basis dictionary of the twin model in the last iteration as an initial guess of the basis dictionary in the current iteration, then re-trains the twin model. In the future, a research direction is on how to utilize all previously trained twin models – for example, by employing the "trust region" technique in the optimization: the same twin model can be used multiple times at different controls inside a trust region of the control space[1], thus reducing the training cost.

[1]In my thesis, the twin model is re-trained at each new control. Generally, gradient-based trust region methods require the gradient to satisfy a property called full-linearity [50, 51]. Unfortunately, this property is not guaranteed by the twin-model gradient. The lack of full-linearity is a key factor that refrains me from exploring the trust-region methods in my thesis. It's an open question on how to introduce the trust-region framework into the optimization

Bibliography

[1] Ramirez, W.F. Application of Optimal Control Theory to Enhanced Oil Recovery. vol. 21, *Elsevier*, 1987.

[2] Ramirez, W. F., Fathi, Z., and Cagnol, J. L. Optimal Injection Policies for Enhanced Oil Recovery: Part 1 Theory and Computational Strategies. *Society of Petroleum Engineers Journal*, 24(03): 328-332, 1984.

[3] Buckley, S. E., and Leverett, M. Mechanism of Fluid Displacement in Sands. *Transactions of the AIME*, 146(01): 107-116, 1942.

[4] Peaceman, D. W. Fundamentals of Numerical Reservoir Simulation Vol. 6. *Elsevier*, 2000.

[5] Alvarado, V., and Manrique, E. Enhanced Oil Recovery: an Update Review. *Energies*, 3(9):1529-1575, 2010.

[6] Anderson Jr, J. D. Fundamentals of Aerodynamics. *Tata McGraw-Hill Education,* 1985.

[7] Verstraete, T., Coletti, F., Bulle, J., Vanderwielen, T., and Arts, T. Optimization of a U-Bend for Minimal Pressure Loss in Internal Cooling Channels - Part I: Numerical Method. *Journal of Turbomachinery*, 135(5):051015, 2013.

[8] Coletti, F., Verstraete, T., Bulle, J., Van der Wielen, T., Van den Berge, N., and Arts, T. Optimization of a U-Bend for Minimal Pressure Loss in Internal Cooling Channels - Part II: Experimental Validation. *Journal of Turbomachinery*, 135(5):051016, 2013.

[9] Redlich, O., and Kwong, J. N. On the Thermodynamics of Solutions. V. An Equation of State. Fugacities of Gaseous Solutions. *Chemical Reviews*, 44(1):233-244, 1949.

[10] Han, J. C., Dutta, S., and Ekkad, S. Gas Turbine Heat Transfer and Cooling Technology. *CRC Press*, 2012.

[11] Lions, J.L. Optimal Control of Systems Governed by Partial Differential Equations, vol. 170, *Springer-Verlag*, 1971.

[12] Jameson, A. Aerodynamic Design via Control Theory, *Journal of Scientific Computing*, 3(3):233-260, 1988.

[13] Anderson, W. K., and Venkatakrishnan, V. Aerodynamic Design Optimization on Unstructured Grids with a Continuous Adjoint Formulation. *Computers and Fluids*, 28(4):443-480, 1999.

[14] Renaud, J. E. Automatic Differentiation in Robust Optimization. *AIAA Journal*, 35(6):1072-1079, 1997.

[15] Plessix, R. E. A Review of the Adjoint-state Method for Computing the Gradient of a Functional with Geophysical Applications. *Geophysical Journal International*, 167(2):495-503, 2006.

[16] Zandvliet, M., Handels, M., van Essen, G., Brouwer, R., and Jansen, J. D. Adjoint-based Well-placement Optimization under Production Constraints. *SPE Journal*, 13(04):392-399, 2008.

[17] Giles, M. B., Duta, M. C., M-uacute, J. D., ller, and Pierce, N. A. Algorithm Developments for Discrete Adjoint Methods. *AIAA journal*, 41(2):198-205, 2003.

[18] Corliss, G. Automatic Differentiation of Algorithms: From Simulation to Optimization. *Springer Science and Business Media*, vol.1, 2002.

[19] Giles, M. B., and Pierce, N. A. An Introduction to the Adjoint Approach to Design. *Flow, Turbulence and Combustion*, 65(3-4):393-415, 2000.

[20] Schneider, R. Applications of the Discrete Adjoint Method in Computational Fluid Dynamics. *Doctoral Dissertation, The University of Leeds*, 2006.

[21] Walther, A. and Griewank, A. Getting Started with ADOL-C. *In U. Naumann und O. Schenk, Combinatorial Scientific Computing, Chapman-Hall CRC Computational Science*, pp. 181-202, 2012.

[22] McIlhagga, W. http://www.mathworks.com/matlabcentral/fileexchange/26807-automatic-differentiation-with-matlab-objects

[23] Bergstra, J., Breuleux, O., Bastien, F., Lamblin, P., Pascanu, R., Desjardins, G., Turian, J., Warde-Farley, D., and Bengio, Y. Theano: A CPU and GPU Math Expression Compiler, *Proceedings of the Python for Scientific Computing Conference*, Austin, TX , 2010.

[24] Royden, H. L., and Fitzpatrick, P. Real Analysis. *Vol. 198, No. 8. New York: Macmillan*, 1988.

[25] Draper, N. R., Smith, H., and Pownell, E. Applied Regression Analysis. *Vol. 3, New York: Wiley*, 1966.

[26] Ghanem, R. G., and Spanos, P. D. Stochastic Finite Elements: a Spectral Approach. *Courier Corporation*, 2003.

[27] Smolyak, S. A. Interpolation and Quadrature formulas for the classes W_s^a and E_s^a. *Dokl. Akad. Nauk SSSR*, vol.131:1028-1031, 1960.

[28] Chen, S. S., Donoho, D. L., and Saunders, M. A. Atomic Decomposition by Basis Pursuit. *SIAM Review*, 43(1):129-159, 2001.

[29] Daubechies, I. Time-frequency Localization Operators: a Geometric Phase Space Approach. *Information Theory, IEEE Transactions on*, 34(4):605-612, 1988.

[30] Saltelli, A., Chan, K., and Scott, E. M. Sensitivity Analysis vol. 1, New York, Wiley, 2000.

[31] Mallat, S. G., and Zhang, Z. Matching Pursuits with Time-Frequency Dictionaries. *Signal Processing, IEEE Transactions on*, 41(12):3397-3415, 1993.

[32] Friedman, J. H. An Overview of Predictive Learning and Function Approximation. *Springer Berlin Heidelberg*, pp. 1-61, 1994.

[33] Reed, R. Pruning Algorithms - a Survey. *Neural Networks, IEEE Transactions on*, 4(5):740-747, 1993.

[34] Jekabsons, G. Adaptive Basis Function Construction: an Approach for Adaptive Building of Sparse Polynomial Regression Models. *INTECH Open Access Publisher*, 2010.

[35] Blatman, G., and Sudret, B. Sparse Polynomial Chaos Expansions and Adaptive Stochastic Finite Elements Using a Regression Approach. *Comptes Rendus Mĕcanique*, 336(6):518-523, 2008.

[36] Blatman, G., and Sudret, B. An Adaptive Algorithm to Build up Sparse Polynomial Chaos Expansions for Stochastic Finite Element Analysis. *Probabilistic Engineering Mechanics*, 25(2):183-197, 2010.

[37] Miller, A. Subset Selection in Regression. *London: Chapmen and Hall Press*, 1990.

[38] Geisser, S. Predictive inference, *CRC press*, vol. 55, 1993.

[39] Lohmiller, W., and Slotine, J. J. E. On Contraction Analysis for Non-linear Systems. *Automatica*, 34(6):683-696, 1998.

[40] Dennis, Jr, John E., and Jorge J. Moré. Quasi-Newton Methods, Motivation and Theory. *SIAM Review*, 19(1):46-89, 1977.

[41] Rios, L. M., and Sahinidis, N. V. Derivative-free Optimization: A Review of Algorithms and Comparison of Software Implementations. *Journal of Global Optimization*, 56(3):1247-1293, 2013.

[42] Nocedal, J. Updating Quasi-Newton Matrices with Limited Storage. *Mathematics of computation*, 35(151):773-782, 1980.

[43] S. G. Johnson, The NLopt Nonlinear-optimization Package,. *http://ab-initio.mit.edu/nlopt*

[44] Tibshirani, R. Regression Shrinkage and Selection via the Lasso. *Journal of the Royal Statistical Society*, Series B (Methodological):267-288, 1996.

[45] Torczon, V. On the Convergence of Pattern Search Algorithms. *SIAM Journal on optimization*, 7(1):1-25, 1997.

[46] Conn, A. R., Gould, N. I., and Toint, P. L. Trust Region Methods. *SIAM*, vol. 1, 2000.

[47] Karasozen, B. Survey of Trust-region Derivative Free Optimization Methods. *Journal of Industrial and Management Optimization*, 3(2), 321, 2007.

[48] Powell, M.J. A Direct Search Optimization Method that Models the Objective and Constraint Functions by Linear Interpolation. *Advances in Optimization and Numerical Analysis*, pp. 51-67, Springer Netherlands, 1994.

[49] Alexandrov, N.M., Lewis, R.M., Gumbert, C.R., Green, L.L, and Newman, P.A. Approximation and Model Management in Aerodynamic Optimization with Variable Fidelity Models. *AIAA Journal of Aircraft*, 38(6):1093–1101, 2001.

[50] Conn, A. R., Scheinberg, K., and Vicente, L. N. Global Convergence of General Derivative-free Trust-region Algorithms to First-and Second-order Critical Points. *SIAM Journal on Optimization*, 20(1):387-415, 2009

[51] Wild, S. M., and Shoemaker, C. Global Convergence of Radial Basis Function Trust-region Algorithms for Derivative-free Optimization. *SIAM Review*, 55(2):349-371, 2013

[52] Pintér, J.D. Global Optimization in Action: Continuous and Lipschitz Optimization. Algorithms, Implementations and Applications. *Nonconvex Optimization and its Applications*, vol. 6, 1996.

[53] Schwefel, H. P. P. Evolution and Optimum Seeking: the Sixth Generation. *John Wiley & Sons, Inc.*, 1993.

[54] Holland, J.H. Adaptation in Natural and Artificial Systems. *The University of Michigan Press*, Ann Arbor, 1975.

[55] Banks, A., Vincent, J., and Anyakoha, C. A Review of Particle Swarm Optimization. Part I: Background and Development. *Natural Computing*, 6(4):467-484, 2007.

[56] Yang, X. S., and Deb, S. Engineering Optimisation by Cuckoo Search. *International Journal of Mathematical Modelling and Numerical Optimisation*, 1(4):330-343, 2010.

[57] Alexandrov, N. M., Dennis Jr, J. E., Lewis, R. M., and Torczon, V. A Trust-region Framework for Managing the Use of Approximation Models in Optimization. *Structural Optimization*, 15(1):16-23, 1998.

[58] Carter, R. G. On the Global Convergence of Trust Region Algorithms Using Inexact Gradient Information. *SIAM Journal on Numerical Analysis*, 28(1):251-265, 1991.

[59] Carter, R. G. Numerical Experience with a Class of Algorithms for Nonlinear Optimization Using Inexact Function and Gradient Information. *SIAM Journal on Scientific Computing*, 14(2):368-388, 1993.

[60] Fu, M. C. Optimization via Simulation: A Review. *Annals of Operations Research*, 53(1):199-247, 1994.

[61] OpenFOAM, *http://www.openfoam.org/*

[62] Aspen, *http://www.aspentech.com/products/aspenONE/*

[63] Chen, W., Xiong, Y., Tsui, K. L., and Wang, S. A Design-driven Validation Approach Using Bayesian Prediction Models. *Journal of Mechanical Design*, 130(2):021101, 2008.

[64] Wang, S., Tsui, K. L., and Chen, W., Bayesian Validation of Computer Models. *Technometrics*, 51(4):439–451, 2009.

[65] Qian, P. Z., and Wu, C. J. Bayesian Hierarchical Modeling for Integrating Low-accuracy and High-accuracy Experiments. *Technometrics*, 50(2):192-204, 2008.

[66] Zhou, D. X. Derivative Reproducing Properties for Kernel Methods in Learning Theory. *Journal of computational and Applied Mathematics*, 220(1):456-463, 2008

[67] Vazquez, E., and Bect, J. Convergence Properties of the Expected Improvement Algorithm with Fixed Mean and Covariance Functions. *Journal of Statistical Planning and inference*, 140(11):3088-3095, 2010.

[68] Bull, A. D. Convergence Rates of Efficient Global Optimization Algorithms. *The Journal of Machine Learning Research* 12:2879-2904, 2011.

[69] Srinivas, N., Krause, A., Kakade, S. M., and Seeger, M. Gaussian Process Optimization in the Bandit Setting: No Regret and Experimental Design. *arXiv preprint* arXiv:0912.3995, 2009

[70] Berlinet, A., and Thomas-Agnan, C. Reproducing Kernel Hilbert Spaces in Probability and Statistics. *Springer Science and Business Media*, 2011.

[71] Snoek, J., Larochelle, H., and Adams, R. P. Practical Bayesian Optimization of Machine Learning Algorithms. *Advances in Neural Information Processing Systems.* pp.2951-2959, 2012

[72] Močkus, J., Tiesis, V., and Zilinskas, A. The Application of Bayesian Methods for Seeking the Extremum. *Towards Global Optimization* 2(117-129), 1978.

[73] Locatelli, M. Bayesian Algorithms for One-dimensional Global Optimization. *Journal of Global Optimization*, 10(1):57-76, 1997.

[74] Chung, H. S., and Alonso, J. J. Using Gradients to Construct CoKriging Approximation Models for High-dimensional Design Optimization Problems. *American Institute of Aeronautics and Astronautics paper*, 992, 2001.

[75] Vauclin, M., Vieira, S. R., Vachaud, G., and Nielsen, D. R. The Use of CoKriging with Limited Field Soil Observations. *Soil Science Society of America Journal*, 47(2):175-184, 1983

[76] Kennedy, M. C., and O'Hagan, A. Predicting the Output from a Complex Computer Code when Fast Approximations are Available. *Biometrika*, 87(1):1-13, 2000.

[77] Kennedy, M. C., and O'Hagan, A. Bayesian Calibration of Computer Models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 63(3):425-464, 2001.

[78] Higdon, D., Kennedy, M., Cavendish, J. C., Cafeo, J. A., and Ryne, R. D. Combining Field Data and Computer Simulations for Calibration and Prediction. *SIAM Journal on Scientific Computing*, 26(2):448-466, 2004.

[79] O'Hagan, A. A Markov Property for Covariance Structures. *Statistics Research Report*, 98(13), 1998.

[80] Aronszajn, N. Theory of Reproducing Kernels. *Transactions of the American Mathematical Society*, 68(3):337-404, 1950.

[81] Showalter, R. E. Hilbert Space Methods for Partial Differential Equations. *Dover Publications*, Mineola, New York, 2010

[82] Jones, D. R., Schonlau, M., and Welch, W. J. Efficient Global Optimization of Expensive Black-box Functions. *Journal of Global Optimization*, 13(4):455-492, 1998.

[83] Bertsekas, D. P. Nonlinear Programming. *Athena Scientific*, Cambridge, MA, 1999.

[84] Spall, J. C. Introduction to Stochastic Search and Optimization: Estimation, Simulation, and Control. *John Wiley & Sons*, vol. 65, 2005.

[85] Fletcher, R., and Reeves, C. M. Function Minimization by Conjugate Gradients. *The Computer Journal*, 7(2):149-154, 1964.

[86] Dai, Y. H., and Yuan, Y. A Nonlinear Conjugate Gradient Method with a Strong Global Convergence Property. *SIAM Journal on Optimization*, 10(1):177-182, 1999.

[87] Gudmundsson, S. Parallel Global Optimization. *Master Thesis*, IMM, Technical University of Denmark, 1998.

[88] Žilinskas, J. Branch and Bound with Simplicial Partitions for Global Optimization. *Mathematical Modelling and Analysis*, 13(1):145-159, 2008.

[89] Rasmussen, C. E. Gaussian Processes in Machine Learning. in *Advanced Lectures on Machine Learning*, Springer Berlin Heidelberg, pp.63-71, 2004.

[90] Matérn, B. Spatial Variation, *Springer*, New York, 1960.

[91] Kushner, H. J. A New Method of Locating the Maximum Point of an Arbitrary Multipeak Curve in the Presence of Noise. *Journal of Basic Engineering*, 86(1):97-106, 1964.

[92] Homaifar, A., Qi, C. X., and Lai, S. H. Constrained Optimization via Genetic Algorithms. *Simulation*, 62(4):242-253, 1994.

[93] Conn, A. R., Gould, N. I., and Toint, P. A globally Convergent Augmented Lagrangian Algorithm for Optimization with General Constraints and Simple Bounds. *SIAM Journal on Numerical Analysis*, 28(2):545-572, 1991.

[94] Conn, A., Gould, N., and Toint, P. A Globally Convergent Lagrangian Barrier Algorithm for Optimization with General Inequality Constraints and Simple Bounds. *Mathematics of Computation of the American Mathematical Society*, 66(217):261-288, 1997.

[95] Gardner, J. R., Kusner, M. J., Xu, Z. E., Weinberger, K. Q., and Cunningham, J. Bayesian Optimization with Inequality Constraints. *International Conference on Machine Learning*, pp. 937-945, 2014.

[96] Gramacy, R. B., and Lee, H. K. Optimization Under Unknown Constraints. *arXiv preprint arXiv:1004.4027*, 2010.

[97] Gelbart, M. A. Constrained Bayesian Optimization and Applications. *Doctoral Dissertation*, Harvard University, 2015.

[98] Queipo, N. V., Haftka, R. T., Shyy, W., Goel, T., Vaidyanathan, R., and Tucker, P. K. Surrogate-based Analysis and Optimization. *Progress in Aerospace Sciences*, 41(1), 1-28, 2005.

[99] Chen, H. Black-box Stencil Interpolation Method for Model Reduction. *Master thesis*, Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, 2012.

[100] Schiesser, W. E. The Numerical Methods of Lines. *Academic Press*, 1991.

[101] Smith, G. D. Numerical Solution of Partial Differential Equations: Finite Difference Methods. *Oxford University Press*, pp. 67-68, 1985.

[102] Mallat, S. G. A Theory for Multiresolution Signal Decomposition: the Wavelet Representation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 11(7):674-693, 1989.

[103] Murdock, J. W. Fundamental Fluid Mechanics for the Practicing Engineer, *CRC Press*, 1993.

[104] Armijo, L. Minimization of Functions Having Lipschitz Continuous First Partial Derivatives. *Pacific Journal of Mathematics*, 16(1):1-3, 1966.

[105] Taylor, A. E., and Lay, D. C. Introduction to Functional Analysis. *vol. 2, Wiley, New York*, 1958.

[106] Ansys Fluent Theory Guide. *ANSYS Inc.*, USA, 2011.

[107] ANSYS CFX-solver Theory Guide. *ANSYS CFX Release*, 11, 69-118, 2012.

[108] Hirsch, M. W., Smale, S., and Devaney, R. L. Differential Equations, Dynamical Systems, and an Introduction to Chaos. *Academic Press*, 2012.

[109] K.-A. Lie. An Introduction to Reservoir Simulation Using MATLAB: User Guide for the Matlab Reservoir Simulation Toolbox (MRST). *SINTEF ICT*, Dec 2015.