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

September 2016

Author . . . .
Department of Aeronautics and Astronautics
Apr 28, 2016

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

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

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

Accepted by . . . .
Paulo Lozano
Chairman, Department Committee on Graduate Theses

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 Apr 28, 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 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.

We consider the case where the flux function is unknown in the gray-box conservation law. The twin model method is presented to estimated the gradient by inferring the flux function from the space-time solution. The method enables the estimation of the gradient by solving the adjoint equation associated with the inferred conservation law. Building upon previous research, a Bayesian optimization framework is presented that admits the estimated gradient. The effectiveness of the proposed optimization method is compared to a conventional Bayesian optimization method where the gradient is unavailable. The performance of the conventional method is found to deteriorate as the optimization dimensionality increases. The twin model enhances the Bayesian optimization performance given a limited number of gray-box simulations.

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 must xiexie my baby, Evin Chen, nicknamed Abu, for adding an extra dimension to the optimization of my life.

Contents

1 Background 15

1.1 Motivation 15

1.2 Problem Formulation 21

1.3 Literature Review 22

1.3.1 Review of Optimization Methods 23

1.3.2 The Adjoint Method 30

1.3.3 Adaptive Basis Construction 36

1.4 Notations 38

1.5 Thesis Objectives 40

1.6 Outline 40

2 Estimate the Gradient by Using the Space-time Solution 43

2.1 Approach 44

2.1.1 Parameterization 48

2.1.2 Elements for Adaptive Basis Construction 53

2.1.3 Algorithm 59

2.1.4 Minimizing the Truncation Error 61

2.2 Numerical Results 64

2.2.1 Buckley-Leverett Equation 65

2.2.2 Navier-Stokes Flow 66

2.2.3 Polymer Injection in Petroleum Reservoir 69

2.3 Chapter Summary 76

3 Leveraging the Twin Model for Bayesian Optimization 77

3.1 Approach 78

3.1.1 Modeling the Objective and Gradient by Gaussian Processes . 78

3.1.2 Algorithm 81

3.1.3 Convergence Properties Using True Hyper Parameters 82

3.2 Numerical Results 86

3.2.1 Buckley-Leverett Equation 86

3.2.2 Navier-Stokes Flow 87

3.2.3 Polymer Injection in Petroleum Reservoir 91

3.3 Chapter Summary 93

4 Conclusions 97

4.1 Thesis Summary 97

4.2 Contributions 99

4.3 Future Work 100

A Proof of Theorems 101

A.1 Theorem 1 101

A.2 Theorem 2 103

A.3 Theorem 3 106

List of Figures

1-1 The computational graph for (1.27). The yellow nodes indicate the input variables, the blue node indicates the output variable, and the white nodes indicate the intermediate variables. The arrows indicate elementary operations. The begining and end nodes of each arrow indicate the independent and dependent variables for each operation. 33

1-2 Computational graphs for the PDE simulation and objective evaluation. 35

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. 47

2-2 An example mother wavelet, the Meyer wavelet. 49

2-3 Red line: the integral (2.10) of the Meyer wavelet. Black line: the logistic sigmoid function. 50

2-4 An ad hoc set of bases 52

2-5 Space time solutions. 52

2-6 The objective function ξ evaluated by either the gray-box model or the trained twin model. 53

2-7 The first row shows the three different initial conditions used to generate the gray-box space-time solution. The second row compares the trained $\tilde{F}$ (blue) and the Buckley-Leverett F (red). The third row 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 exists. 54

2-8 An illustration of the tuple representation and the corresponding univariate sigmoid. 56

2-9 Neighborhood for univariate bases. (a) shows the neighborhood (blue) of a single basis (red). (b) shows the neighborhood (blue) of several bases (red). The left column represents the basis on the $\left(j, \frac{\eta}{2^j}\right)$ plane, and the right column shows the actual basis $\phi_{j,\eta}$. 57

2-10 The discretized gray-box solution is shuffled into 3 sets, each indicated by a color. 58

2-11 The basis dictionary for the three solutions in Figure 2-7. 65

2-12 The errors of estimated gradients for the three solutions. 66

2-13 The return bend geometry and the mesh for the simulation. 67

2-14 Left column: an example gray-box solution for a given geometry. Right column: the solution mismatch after training a twin model. 69

2-15 The gray-box state equation (right column) and the trained state equation (left column). The gray-box model uses either the ideal gas equation (first row) or the Reclich-Kwong equation (second row). The convex hull of the gray-box solution is shown by the dashed red line. 70

2-16 A comparison of the estimated gradient and the true gradient. 71

2-17 Water flooding in petroleum reservoir engineering (courtesy from PetroWiki). 72

2-18 The geometry of the petroleum reservoir. 73

2-19 The isosurfaces of $S_w = 0.25$ and $S_w = 0.7$ at $t = 30$ days. 74

2-20 The 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. 75

3-1 The flowchart of Algorithm 3. 83

3-2 Optimized results for the Buckley-Leverett equation. 87

3-3 A comparison of the optimized $u(t = 1, x)$ 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 indicates the $u(t = 1, x)$ obtained by setting the source term to zero. 88

3-4 The current best objective at each iterate. The red line is obtained by the vanilla Bayesian optimization and the green line by the twin-model Bayesian optimization. The black horizontal line indicates the true optimal. 88

3-5 The left plot shows the initial guess of control points (blue dots), the initial guess of the geometry (blue line), the optimized control points (red dots), and the optimized geometry (red line). The purple squares indicate the bound constraints for each control point. The right plot shows the pressure along the interior and the exterior boundaries for the initial (blue) and the optimized (red) geometry. 89

3-6 The current best objective at each iterate for the ideal gas and the Redlich-Kwong gas. The green lines are obtained by the twin-model Bayesian optimization. The red lines are obtained by the vanilla Bayesian optimization. The black horizontal lines indicate the true optimal. . . 90

3-7 The cumulative and per-iterate wall clock time, in minutes. 90

3-8 The permeability of the reservoir, in 100 milli Darcy. The 5 injectors are indicated by the black dots, and the producer is indicated by the green dot. 91

3-9 The current best objective evaluation against the number of iterates. 92

3-10 $\xi(t)$ for the initial and the optimized injection rates. 92

3-11 The optimized time-dependent injection rates. 94

3-12 The current best objective evaluation using the backtracking-Armijo gradient descent method, where the gradient is provided by the twin model. 94

A-1 The state-space trajectories of the gray-box model and the twin model. $\mathcal{M}$ measures the difference of the twin model trajectory (blue) with the gray-box trajectory (red). $\mathcal{T}$ measures the difference of the twin model trajectory with restarts (green) and the gray-box trajectory (red). . . 104

List of Tables

2.1 The integrated errors of the estimated gradients for the three solutions. 66

2.2 The error of the gradient estimation, in percentage. 69

2.3 The error of estimated gradient at day 2, 16, 30, and 44, in percentage. 75

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 of u, and q is the source for 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 [9]. To prevent material failure due to overheating, channels can be drilled inside the rotor blades to circulate coolant air whose dynamics are governed by the Navier-Stokes equation [6]. The pressure used to drive the coolant flow is provided by the compressor, resulting in a penalty on the turbine's thermo-dynamic efficiency

[7]. 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 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, where the reservoir is controlled by the pressure at each wells, by the the injection rate of polymer, and by the temperature of the steam [5]. The pressure, injection rate, and temperature can vary in each well and at 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 is being revolutionized by the numerical simulation and optimization algorithms. On 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 towards the optimal with reduced number of simulation [39, 40, 41, 47, 51, 52, 53, 69]. However, optimization based on conservation law simulation can still be overwhelmingly costly. The cost is two-folded: Firstly, each simulation for a given control may run for hours or days even on a high-end computer. This is mainly because of the high-fidelity physical models, the complex numerical schemes, and the large scale space-time discretization employed in the simulation. Secondly, 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's degree of freedom [57]. 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 [57]. A gradient-based algorithm requires significantly less iterations than a derivative-free algorithm for problems with many control variables [18, 57, 40]. Gradient-based algorithms require the gradient of the optimization objective to the control variables, which is efficiently computable through the adjoint method [10]. The adjoint method propogates the gradient from the objective backward to the control variables through the path of time integration [10] or through the chain of numerical operations [17]. To keep track of the back propogation, the simulator source code needs to be available. In real-world industrial simulators, adjoint is scarcely implemented because most source codes are proprietary and/or legacy. 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 unpreferable because it can take 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 superceded by adjoint-enabled simulators, but it is difficult to be replaced due to its legacy use and cost concerns. The proprietary and 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 proprietary and legacy nature, most simulators for unsteady conservation laws are able to 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 are able to provide the spatial solution. the discussion will focus on the unsteady case, since a steady state simulator can be viewed as a special case of the unsteady one where the solution remains the same over many time steps.

I argue that 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 which simulates

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

with proper intial 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 $0 \le t_1 < t_2 < \cdots < t_M \le 1$ and $0 \le x_1 < x_2 < \cdots < x_N \le 1$ 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

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

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

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

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 [10]. The gradient is

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

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) [10]. Similarly, the gradient of $\tilde{\xi}(c) \equiv \xi(\tilde{u}(c), c)$ is

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

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**, since 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.

My thesis focuses on a class of simulators that I call **gray-box**. A simulator is defined to be gray-box if the following two conditions are met:

1. the adjoint is unavailable, and is impractical to implement into the source code.

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

Many industrial simulators, such as *PSim*, satisfy both conditions. In contrast, a simulator is named **open-box** if condition 1 is violated. For example, *OpenFOAM* [58] 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 benifit of efficient gradient computation brought by adjoint, thereby are not within the research scope of my thesis. If condition 1 is met but 2 is violated, a simulator is named **black-box**. For example, *Aspen* [59], 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 [103] and CFX [104] 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. In addition, my thesis will investigate how the estimated gradient can faciliate 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 multi-phase multi-component 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 the Darcy's law. The Darcy's law 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

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

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

[1] The permeability quantifies the easiness of liquids to pass through the rock.

[2] The viscosity quantifies the internal friction of the liquid flow.

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 equation. The initial and boundary conditions are known. D is a known differential operator that may depend on u, and F is an unknown function that depends on u. q is a known source term that depends on u and c. Notice (1.10) degenerates to (1.1) when D equals 1. The simulator 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.

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 optimization algorithm is relatively small. Chapter 2 develops a mathematical procedure, called the twin model method, that enables adjoint gradient computation by leveraging the full space-time solution. Based upon previous research [63, 64, 68, 69, 71, 73, 74], 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 investigated particularly. In addition, I review the adjoint method since 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 [40], depending on whether the gradient information is used. In the sequel, 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 [40], thus 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 [40]. Local methods seek a local optimum which is also the global optimum for convex problems. An important local method is the trust-region method [44]. Trust-region method introduces a surrogate model that is cheap to evaluate and presumably accurate within a trust region: an adaptive neighborhood around the current iterate [44]. At each iteration, the surrogate is optimized in a domain bounded by the trust region to generate candidte steps for additional objective evaluations [44]. The surrogates can be constructed either by interpolating the objective evaluations [45, 48], or by running a low-fidelity simulation [46, 54]. Convergence to the objective function's optimum is guaranteed by ensuring that the surrogate have the same value and gradient as the objective function when the size of the trust region shrinks to zero [47, 48].

Global methods seek the global optimum. Example methods include the branch-and-bound search [49], evolution methods [50], and Bayesian methods [68, 70, 90]. The branch-and-bound search sequentially partitions the entire control space into a tree structure, and determines lower and upper bounds for the optimum [49].

Partitions that are inferior are eliminated in the course of the search [49]. The bounds are usually obtained through the assumption of the Lipschitz continuity or statistical bounds for the objective function [49]. Evolution methods maintain a population of candidate controls, which adapts and mutates in a way that resembles natural phenomenons such as the natural selection [51, 53] and the swarm intelligence [52]. 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 [68, 69]. 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 [70, 79, 66]. 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 [40]. Besides, DFO methods usually require a large number of function evaluations to converge, especially when the dimension of control is large [40]. This issue can be alleviated by incorporating the gradient information [63, 71, 86, 87]. The details are discussed in the next subsection.

Gradient-based Optimization

Gradient-based optimization (GBO) requires the availability of the gradient values [57, 80]. A gradient value, if exists, provides the optimal infinitesimal change of control variables at each iterate, thus is useful in searching for a better control. Similar to DFO, GBO can also be categorized into local methods and global methods [57]. Examples of local GBO methods include the gradient descent methods [81, 101], the conjugate gradient methods [82, 83], and the quasi-Newton methods [39, 41]. The gradient descent methods and the conjugate gradient methods choose the search step in the direction of either the gradient [81, 101] or a conjugate gradient

[82, 83]. Quasi-Newton methods, such as the Broyden-Fletcher-Goldfarb-Shannon (BFGS) method [39], 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 [39]. In addition, some local DFO methods can be enhanced to use gradient information [55, 56]. For instance, in trust-region methods, the construction of local surrogates can incorporate gradient values if available [55, 56]. The usage of gradient usually improves the surrogate's accuracy thus enhances the quality of the search step, thereby reducing the required number of iterations [55, 56].

Global GBO methods search for the global optimum using gradient values [57, 80]. Many global GBO methods can trace their development to corresponding DFO methods [84, 85, 86, 87, 71]. For example, the stochastic gradient-based global optimization method (StoGo) [84, 85] works by partitioning the control space and bounding the optimum in the same way as the branch-and-bound method [49]. But the search in each partition is performed by gradient-based algorithms such as BFGS [39]. Similarly, some gradient-based evolution methods, such as the gradient-based particle swarm method [86] and the gradient-based cuckoo search method [87], can be viewed as gradient variations of corresponding derivative-free counterparts [52, 53]. For example, the gradient-based particle swarm method combines particle swarm algorithm with the stochastic gradient descent method [86]. The movement of each particle is dictated not only by the function evaluations of all particles, but also by its local gradient [86].

My thesis is particularly interested in the gradient-based Bayesian optimization method [72]. In this method, the posterior of the objective function assimilates both the gradient and function values in a CoKriging framework [63, 72]. The details of my treatment is discussed in Section 1.3.1 and Chapter 3. I expect that the inclusion of gradient values results in 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 [68, 79]. In addition, 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 [68, 79]. The advantage of such properties can be justified when the objective and gradient evaluations are dominantly more expensive than the overhead of optimization algorithm [68]. Besides, my thesis proves that the Baysian optimization method is convergent even if the gradient values are estimated inexactly, which is discussed in Section 3.1.3. The conclusion of Section 3.1.3 is: 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 a desired objective value, GBO methods generally require much less iterations than DFO methods for problems with many control variables [57, 80]. GBO methods can be efficiently applied to optimization constrained by open-box simulators, because the gradient is efficiently computable by the adjoint method [10, 57], 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 $\mathcal{C} \subset \mathbb{R}^d$ [68, 69, 79]. However, Bayesian optimization distinguishes from other methods by maintaining a probabilistic model for ξ [68, 69, 79]. The probabilistic model is exploited to make decisions about where to invest the next function evaluation in $\mathcal{C}$ [68, 69, 79]. In addition, it uses all information of available evaluations, not just local evaluations, to direct the search step [68, 69, 79].

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 [68, 69, 79]. A stochastic process is a function

$$\begin{aligned} f \ :& \mathcal{C} \times \Omega \to \mathbb{R} \\ (c, \omega) &\to f(c, \omega) \end{aligned}, \tag{1.11}$$

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 sample 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.

Stationary Gaussian process is a type of stochastic process that is used ubiquitously in Bayesian optimization [88]. 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 $K(c, c')$ [88]

$$\begin{aligned} m(c) &= \mathbb{E}_\omega \big[f(c, \omega) \big] \\ K(c, c') &= \mathbb{E}_\omega \Big[\big(f(c, \omega) - m(c) \big) \big(f(c', \omega) - m(c') \big) \Big] \, , \end{aligned} \tag{1.12}$$

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 [88]

$$\begin{aligned} \tilde{m}(c) &= m(c) + K(c, \underline{c}_n) K(\underline{c}_n, \underline{c}_n)^{-1} \left(\xi(\underline{c}_n) - m(\underline{c}_n) \right) \\ \tilde{K}(c, c') &= K(c, c') - K(c, \underline{c}_n) K(\underline{c}_n, \underline{c}_n)^{-1} K(\underline{c}_n, c') \end{aligned}, \tag{1.13}$$

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 [88]. In many cases, the covariance are assumed isotropic, indicating that $K(c, c')$ only depends on the L_2 norm $\|c - c'\|$ [88]. 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 [88]. Among such choices, the Matérn 5/2 kernel [89]

$$K(c, c') = \sigma^2 \left(1 + \frac{\sqrt{5}\|c - c'\|}{L} + \frac{5\|c - c'\|^2}{3L^2}\right) \exp\left(-\frac{\sqrt{5}\|c - c'\|}{L}\right) , \qquad (1.14)$$

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 [68]. My thesis will focus on using the Matérn 5/2 kernel. Notice the parameters L and σ, known as the hyperparametes, are yet to be determined. They can be determined by the posterior maximum likelihood estimation (MLE) or by a fully-Bayesian approach [68, 79]. I refer to the reference [68] 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 $c_{\text{best}} = \text{argmax}_{c \in \underline{c}_n} \xi(c)$, Bayesian optimization introduces an acquisition function, $a : \mathcal{C} \to \mathbb{R}^+$, that evaluates the expected utility of investing the next sample at $c \in \mathcal{C}$ [66, 68, 69, 79, 90]. The location of the next sample is determined by an optimization $c_{N+1} = \text{argmax}_{c \in \mathcal{C}} a(c)$ [66, 68, 69, 79, 90]. In most cases, a greedy acquisition function is used, which evaluates the one-step-lookahead utility [66, 68, 69, 79, 90]. There are several choices for the acquisition function, such as

- the probability of improvement (PI) [90],

$$a_{\mathrm{PI}}(c) = \Phi(\gamma(c))\,, \tag{1.15}$$

- the expected improvement (EI) [69, 70],

$$a_{\mathrm{EI}}(c) = \sigma(c)\big(\gamma(c)\Phi(\gamma(c)) + \mathcal{N}(\gamma(c))\big)\,, \tag{1.16}$$

- and the upper confidence bound (UCB) [66],

$$a_{\mathrm{UCB}}(c) = \mu(c) + \kappa\sigma(c)\,, \tag{1.17}$$

  with a tunable parameter $\kappa > 0$,

where μ, σ are the posterior mean and variance, $\gamma(c) = \sigma^{-1}(c)\big(\mu(c) - \xi(c_{\mathrm{best}})\big)$, 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 [68]. Because (1.16) has a closed-form gradient, the acquisition function can be maximized by a global GBO method, e.g. StoGo [85], to obtain its global maximum.

Although my thesis only focuses on bound constraints as shown in (3), Bayesian optimization can accommodate more general inequality and equality constraints [96]. The constraints can be enforced by modifying the objective, such as the penalty method [91], the augmented Lagrangian method [92], and the barrier function method [93]. They can also be enforced by modifying the acquisition function, such as the recently developed expected improvement with constraints (EIC) method [94], and the integrated expected conditional improvement (IECI) method [95]. See Chapter 2 of [96] for a detailed review of constrained Bayesian optimization.

In addition to function evaluations $\xi(\underline{c}_n)$, Bayesian optimization admits gradient information [63, 71]. In Chapter 3, I investigate the scenario where the gradient evaluations are inexact [74]. The Bayesian optimization method developed in my thesis allows both the exact function evaluation and the inexact gradient evaluation. Details of this topic will be discussed in Section 3.1.

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, that uniquely defines 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

$$\left(\frac{d\xi}{dc}\right)_i \approx \frac{1}{\delta}\big(\xi(u+\Delta u_i, c+\delta e_i) - \xi(u,c)\big)\,, \tag{1.18}$$

where

$$\mathcal{F}(u,c) = 0\,, \quad \mathcal{F}(u+\Delta u_i, c+\delta e_i) = 0\,. \tag{1.19}$$

e_i indicates the ith unit Cartesian basis vector in $\mathbb{R}^d$, and $\delta > 0$ indicates a small perturbation. Because (1.19) needs to be solved for every δe_i, so that the corresponding Δu_i can be used in (1.18), $d{+}1$ 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 [10]. To see this, linearize $\mathcal{F}(u,c) = 0$ into a variational form

$$\delta\mathcal{F} = \frac{\partial\mathcal{F}}{\partial u}\delta u + \frac{\partial\mathcal{F}}{\partial c}\delta c = 0\,, \tag{1.20}$$

which gives

$$\frac{du}{dc} = -\left(\frac{\partial\mathcal{F}}{\partial u}\right)^{-1}\frac{\partial\mathcal{F}}{\partial c} \tag{1.21}$$

Using (1.21), $\frac{d\xi}{dc}$ can be expressed by

$$\begin{aligned}\frac{d\xi}{dc} &= \frac{\partial\xi}{\partial u}\frac{du}{dc} + \frac{\partial\xi}{\partial c}\\ &= -\frac{\partial\xi}{\partial u}\left(\frac{\partial\mathcal{F}}{\partial u}\right)^{-1}\frac{\partial\mathcal{F}}{\partial c} + \frac{\partial\xi}{\partial c}\,,\\ &= -\lambda^T\frac{\partial\mathcal{F}}{\partial c} + \frac{\partial\xi}{\partial c}\end{aligned} \tag{1.22}$$

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

$$\left(\frac{\partial\mathcal{F}}{\partial u}\right)^T\lambda = \left(\frac{\partial\xi}{\partial u}\right)^T \tag{1.23}$$

Therefore, the gradient can be evaluated by (1.22) using one simulation of $\mathcal{F}(u,c) = 0$ and one simulation of (1.23) 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 excuted first [14]. The above procedure, (1.20) thru. (1.23), is the continuous adjoint, where $\mathcal{F}$ is a differential operator. The continous adjoint method linearizes the continuous PDE $\mathcal{F}(u,c) = 0$ first, then discretizes the adjoint equation (1.23) [10]. In (1.23), $\left(\frac{\partial\mathcal{F}}{\partial u}\right)^T$ can be derived as another differential operator. With proper boundary and/or initial conditions, it uniquely determines the adjoint solution λ. See [18] for a detailed derivation of the continuous adjoint equation.

The discrete adjoint method [16] 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 (1.20) thru. (1.23), the discrete adjoint equation can be obtained

$$\left(\frac{\partial \mathcal{F}_d}{\partial \boldsymbol{u}}\right)^T \boldsymbol{\lambda} = \left(\frac{\partial \xi}{\partial \boldsymbol{u}}\right)^T , \tag{1.24}$$

which is a linear system of equations. $\left(\frac{\partial \mathcal{F}_d}{\partial \boldsymbol{u}}\right)^T$ is derived as another difference operator, a function whose output 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

$$\frac{d\xi}{d\boldsymbol{c}} = -\boldsymbol{\lambda}^T \frac{\partial \mathcal{F}_d}{\partial \boldsymbol{c}} + \frac{\partial \xi}{\partial \boldsymbol{c}} . \tag{1.25}$$

See Chapter 1 of [19] for a detailed derivation of the discrete adjoint.

The discrete adjoint method can be implemented by automatic differentiation (AD) [17]. AD exploits the fact that a PDE simulation, no matter how complicated, executes a sequence of elementary arithmetic operations (e.g. addition, multiplication) and elementary functions (e.g. exp, sin) [17]. For example, consider the function

$$\xi = f(c_1, c_2) = c_1 c_2 + \sin(c_1) . \tag{1.26}$$

The function can be broken down into a series of elementary arithmetic operations and elementary functions.

$$\begin{aligned} w_1 &= c_1 \\ w_2 &= c_2 \\ w_3 &= w_1 w_2 \\ w_4 &= \sin(w_1) \\ \xi &= w_3 + w_4 \ . \end{aligned} \tag{1.27}$$

(1.27) can be represented by a computational graph in Figure 1-1. In the graph, the

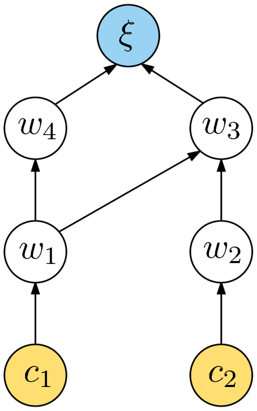


Figure 1-1: The computational graph for (1.27). The yellow nodes indicate the input variables, the blue node indicates the output variable, and the white nodes indicate the intermediate variables. The arrows indicate elementary operations. The begining and end nodes of each arrow indicate the independent and dependent variables for each operation.

gradient of the output with respect to the input variables can be computed using the chain rule [17]. Let $\bar{z}$ denote the gradient of ξ with respect to z, for any independent or intermediate variable z in (1.27). To compute the derivatives $\bar{c}_1 = \frac{\partial \xi}{\partial c_1}$ and $\bar{c}_2 = \frac{\partial \xi}{\partial c_2}$, one can propogate the derivatives backward in the computational graph as follows

$$\begin{aligned}
\bar{w}_4 &= 1 \\
\bar{w}_3 &= 1 \\
\bar{w}_2 &= \bar{w}_3 \frac{\partial w_3}{\partial w_2} = 1 \cdot w_1 \\
\bar{w}_1 &= \bar{w}_4 \frac{\partial w_4}{\partial w_1} + \bar{w}_3 \frac{\partial w_3}{\partial w_1} = 1 \cdot \cos(w_1) + 1 \cdot w_2 \\
\bar{c}_2 &= \bar{w}_2 = c_1 \\
\bar{c}_1 &= \bar{w}_1 = \cos(c_1) + c_2
\end{aligned} \tag{1.28}$$

The derivatives in (1.28) are straightforward to compute. This is because every forward operation in (1.27) is among a small library of elementary operations, and their derivatives can be hardwired in AD softwares. Notice each arrow in Figure 1-1 is traversed once and only once in the backward propogation (1.28). Therefore, the backward gradient computation has a similar cost as the forward output computation, regardless of the number of input variables. See [17] for a thorough review of AD.

Because a PDE simulation can be viewed as performing a sequence of elementary operations, AD can be used to evaluate the discrete adjoint. Consider the PDE $\mathcal{F}(u, c) = 0$ in (1.19), where u is space-time dependent. After space-time discretization, one obtain a set of timestepwise equations

$$\mathcal{F}_{t+1} = \mathcal{F}(\boldsymbol{u}_t, \boldsymbol{u}_{t+1}, \boldsymbol{c}_{t+1}) = 0\,, \tag{1.29}$$

for $t = 0, \cdots, T-1$, where $\boldsymbol{u}_t$ and $\boldsymbol{c}_t$ are the state and control variables at the tth timestep. Here $\mathcal{F}$ is redefined as a function whose output has the same dimension as $\boldsymbol{u}_0$ through $\boldsymbol{u}_T$. The equation uniquely determines $\boldsymbol{u}_1$ and $\boldsymbol{u}_T$ given $\boldsymbol{u}_0$. AD can be used to compute the gradient of an objective function

$$\xi = \xi(\boldsymbol{u}_0, \cdots, \boldsymbol{u}_T; \boldsymbol{c}_1, \cdots \boldsymbol{c}_T)$$

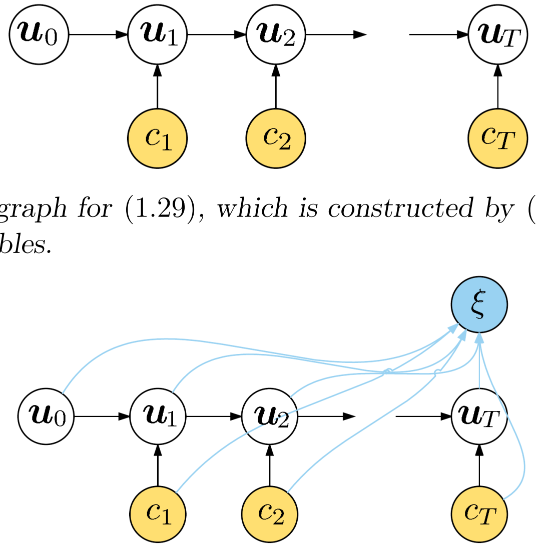
to the control variables. To see this, consider the evaluation of (1.29) using an AD software. The gradients $\frac{\partial \mathcal{F}_{t+1}}{\partial \boldsymbol{u}_t}$, $\frac{\partial \mathcal{F}_{t+1}}{\partial \boldsymbol{u}_{t+1}}$, and $\frac{\partial \mathcal{F}_{t+1}}{\partial \boldsymbol{c}_{t+1}}$, for $t = 0, \cdots, T-1$, can be computed from the functional form of $\mathcal{F}$. Therefore, one can obtain

$$\begin{aligned}
\frac{\partial \boldsymbol{u}_{t+1}}{\partial \boldsymbol{u}_t} &= -\left(\frac{\partial \mathcal{F}_{t+1}}{\partial \boldsymbol{u}_{t+1}}\right)^{-1} \left(\frac{\partial \mathcal{F}_{t+1}}{\partial \boldsymbol{u}_t}\right) \\
\frac{\partial \boldsymbol{u}_{t+1}}{\partial \boldsymbol{c}_{t+1}} &= -\left(\frac{\partial \mathcal{F}_{t+1}}{\partial \boldsymbol{u}_{t+1}}\right)^{-1} \left(\frac{\partial \mathcal{F}_{t+1}}{\partial \boldsymbol{c}_{t+1}}\right)\,.
\end{aligned} \tag{1.30}$$

Therefore a computational graph, Figure 1-2a, can be constructed using the chain rule. The graph enables the evaluation of all $\frac{\partial \boldsymbol{u}_t}{\partial \boldsymbol{c}_{t-i}}$, for $t = 1, \cdots, T$ and $i = 0, \cdots, t-1$, because

$$\frac{\partial \boldsymbol{u}_t}{\partial \boldsymbol{c}_{t-i}} = \left(\frac{\partial \boldsymbol{u}_t}{\partial \boldsymbol{u}_{t-1}}\right) \cdots \left(\frac{\partial \boldsymbol{u}_{t-i+1}}{\partial \boldsymbol{u}_{t-i}}\right) \left(\frac{\partial \boldsymbol{u}_{t-i}}{\partial \boldsymbol{c}_{t-i}}\right) \tag{1.31}$$

Given the solutions $\boldsymbol{u}_t$'s and the controls $\boldsymbol{c}_t$'s, the evaluation of ξ is nothing but overlaying the graph by an additional layer of computations, shown in Figure 1-2b.



(a) The computational graph for (1.29)*, which is constructed by* (1.30)*. The yellow nodes indicate the input variables.*

(b) The computational graph for evaluating the objective function ξ. The blue node indicates the output variable.

Figure 1-2: Computational graphs for the PDE simulation and objective evaluation.

Because $\frac{\partial \xi}{\partial \boldsymbol{u}_t}$'s and $\frac{\partial \xi}{\partial \boldsymbol{c}_t}$'s can be obtained by AD, the gradient

$$\frac{d\xi}{d\boldsymbol{c}_t} = \frac{\partial \xi}{\partial \boldsymbol{c}_t} + \frac{\partial \xi}{\partial \boldsymbol{u}_t}\frac{\partial \boldsymbol{u}_t}{\partial \boldsymbol{c}_t} + \frac{\partial \xi}{\partial \boldsymbol{u}_{t+1}}\frac{\partial \boldsymbol{u}_{t+1}}{\partial \boldsymbol{c}_t} + \cdots + \frac{\partial \xi}{\partial \boldsymbol{u}_T}\frac{\partial \boldsymbol{u}_T}{\partial \boldsymbol{c}_t} \tag{1.32}$$

can be computed, for all $t = 1, \cdots, T$.

The adjoint method has seen wide applications in optimization problems constrained by conservation law simulations, such as in airfoil design [11, 12, 13], adaptive mesh refinement [19], injection policy optimization in petroleum reservoirs [2], history matching in reservoir geophysics [14], and optimal well placement in reservoir management [15]. Besides, there are many free AD softwares available for various languages, such as *ADOL-C* (C, C++) [20], *Adiff* (Matlab) [21], and *Theano* (Python) [22]. Unfortunately, the adjoint method is not directly applicable to gray-box simulations, as explained in Section 1.1. To break this limitation, Chapter 2 develops the twin model method that enables the adjoint gradient computation for gray-box simulations.

1.3.3 Adaptive Basis Construction

The unknown function F in (1.10) can be approximated by a linear combination of basis functions [23]. An over-complete or incomplete set of bases can negatively affect the approximation due to overfitting or underfitting [24]. 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 [23], $\{\phi\}_{i\in\mathbb{N}}$, such as the polynomial basis, Fourier basis, and the wavelet basis [99].

$$F(\cdot) = \sum_{i\in\mathbb{N}} \alpha_i \phi_i(\cdot) \,, \tag{1.33}$$

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 [23].

For example, a bivariate function can be represented by monomials (Weierstrass approximation theorem [102])

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

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

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

$$F(\cdot) \approx \sum_{i\in\mathcal{A}} \alpha_i \phi_i(\cdot) \,, \tag{1.34}$$

where $\{\phi_i\}_{i\in\mathcal{A}}$ is called a basis dictionary [30]. 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 $p \in \mathbb{N}$ [25]. Given a dictionary, the coefficients for $\tilde{F}$ can be determined by the minimization [25]

$$\boldsymbol{\alpha}^* = \underset{\boldsymbol{\alpha} \in \mathbb{R}^{|\mathcal{A}|}}{\operatorname{argmin}} \left\| \tilde{F} - \sum_{i \in \mathcal{A}} \alpha_i \phi_i \right\|_{L_p}, \tag{1.35}$$

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. Details are discussed in Section 2.1.1.

If the dictionary is pre-determined, its cardinality can increase as the number of variables increases, and as the basis complexity increases [25]. For example, for 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 [25].

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 [25, 27, 30, 42]. 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 [42], matching pursuit [30], and basis pursuit [27]. Lasso regularization adds a penalty $\lambda \sum_{i \in \mathcal{A}} |\alpha_i|$ to the approximation error, where $\lambda > 0$ is a tunable parameter [42]. 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 [42]. Matching pursuit adopts a greedy, stepwise approach [30]. It either selects a significant basis one-at-a-time (forward selection) from a dictionary [31], or prunes an insignificant basis one-at-a-time (backward pruning) from the dictionary [32]. Basis pursuit minimizes $\|\boldsymbol{\alpha}\|_{L_1}$ subject to (1.33), which is equivalently reformulated and efficiently solved as a linear programming problem [27].

[3] Usually $p = 1$ [27] or 2 [28, 30].

Conventionally, the dictionary for the sparse approximation needs to be predetermined, with the belief that the dictionary is a superset of the significant bases [34]. This can be problematic because the maximum complexity[4] of the significant bases are unknown a prior. To address this issue, methods have been devised that construct an adaptive dictionary [33, 34, 35]. Although different in details, such methods share the same approach: In the beginning, some trivial bases are given as inputs. For example, the starting basis can be 1 for polynomial basis [33]. The starting bases serve as seeds from which more complex bases grow. I refer to [33, 34, 35] for more details of the heuristics. Then a dictionary is built up progressively by iterating over a forward step and a backward step [33, 34, 35]. The forward step searches over a candidate set of bases, and appends the significant ones to the dictionary [33, 34, 35]. The backward step searches over the current dictionary, and removes the insignificant ones from the dictionary [33, 34, 35]. 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 [33, 34, 35]. Such approach is adopted in my thesis to build up the bases for $\tilde{F}$. Details are discussed in Section 2.1.2.

1.4 Notations

The general notations are declared here.

- $t \in [0, T]$: the time,
- $\{t_i\}_{i=1}^{M}$: the time discretization,
- $x \in \Omega$: the space,
- $\{x_j\}_{j=1}^{N}$: the space discretization,
- u: the space-time solution of gray-box conservation law,

[4] The definition of complexity is basis-dependent. For example, the complexity for polynomial basis can be its total polynomial degree; and the complexity for wavelet basis can be its finest resolution [99]. Here the "complexity" is discussed in a general sense.

- $\tilde{u}$: 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: 1) the number of equations of the conservation law; or 2) the number of folds in cross validation.
- D: a 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,
- $\mathcal{M}$: the solution mismatch,
- $\overline{\mathcal{M}}$: the mean solution mismatch in cross validation,

- ϕ: the basis functions for $\tilde{F}$,

- α: the coefficients for ϕ,

- $\mathcal{A}$: the index set for a basis dictionary,

- T: twin model,

- τ: residual,

- $\boldsymbol{\tau}$: discretized residual,

- $\mathcal{T}$: integrated truncation error.

1.5 Thesis Objectives

Based the motiviation and literature review, we find it important to to enable adjoint gradient computation for gray-box conservation law simulations. 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 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.

1.6 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 firstly training a twin model, then applying the adjoint method to the trained twin model. To train the twin model, a solution mismatch metric is presented. The metric is used for the training of twin models. In addition, an adaptive basis construction scheme is developed to approximate the unknown components in the twin model. Based on the developments, the algorithm for constructing the twin model is summarized. Furthermore, methods for reducing the computational cost for training the twin model is discussed. Finally, the twin model algorithm is demonstrated in several numerical examples. Chapter 3 develops an efficient global optimization method by using the twin-model gradient obtained from Chapter 2. In Chapter 3, the twin-model gradient is modeled stochastically by the Gaussian process. Based on the Gaussian process model, a Bayesian optimization algorithm is devised that leverages the twin-model gradient. Its convergence properties are studied. Finally, the twin-model Bayesian optimization algorithm is demonstrated in 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 which 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 example above 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

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 space-time grid of the gray-box simulator. Define the solution mismatch

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

where $i = 1, \cdots, M$ are the indices for time grid, and $j = 1, \cdots, N$ are the indices for the space grid. w_{ij}'s are the quadrature weights for the space-time integration.

For example, if a uniform Cartesian space-time grid is used, the quadrature weights equal a constant. More generally, the quadrature weights are defined with respect to the space-time integration, so $\mathcal{M}$ approximates the space-time integration of the continous solutions' mismatch,

$$\mathcal{M} \approx \int_0^T \int_\Omega \big(\tilde{u}(t,x) - u(t,x)\big)^2 \,\mathrm{d}x\,\mathrm{d}t \tag{2.3}$$

Notice that $\mathcal{M}$ solely depends on $\tilde{F}$ through the twin model solution $\tilde{\boldsymbol{u}}$ given the quadrature weights and the gray-box solution. Given a function space $\mathcal{S}_F$, I propose to infer a flux $\tilde{F}$ such that the mismatch between $\boldsymbol{u}$ and $\tilde{\boldsymbol{u}}$ is minimized, i.e.

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

The choice for $\mathcal{S}_F$ will be discussed later in Section 2.1.1 and 2.1.2. By setting the F in (2.1) to be $\tilde{F}^*$, one obtain a "trained" twin-model equation

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

Let $\tilde{\boldsymbol{u}}^*$ be the space-time solution of the twin model governed by (2.5). 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

$$\frac{d\mathcal{M}}{d\tilde{F}} = \frac{d\mathcal{M}}{d\tilde{\boldsymbol{u}}}\frac{d\tilde{\boldsymbol{u}}}{d\tilde{F}} \tag{2.6}$$

Using the gradient (2.6), the optimization problem, (2.4), can be solved by gradient-based methods. Finally, 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. I can not prove the inferrability for the general form (1.10)

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

However, the inferrability can be partially justified by the following theorem if (1.10) has only one equation, has one dimensional space, $q = 0$, and $D = 1$.

Theorem 1. *Consider two PDEs*

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

$$\frac{\partial \tilde{u}}{\partial t} + \frac{\partial \tilde{F}(\tilde{u})}{\partial x} = 0\,, \tag{2.8}$$

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 Lipschtiz continuous with constant L_F. Let

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

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

The proof is given in Appendix A.1. An illustration of B_u is given in Figure 2-1. Several observations can be made from Theorem 1. Firstly, if the solutions of (2.7) and (2.8) match closely ($|\tilde{u}(t,x) - u(t,x)| < \delta$), then the derivatives of their flux functions must match closely in B_u $\left(\left|\frac{d\tilde{F}}{du} - \frac{dF}{du}\right| < \epsilon\right)$. Secondly, 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 $\left(\left|\frac{du_0}{dx}\right| \geq \gamma > 0\right)$. Thirdly, only the derivatives of the fluxes are guaranteed to match $\left(\left|\frac{d\tilde{F}}{du} - \frac{dF}{du}\right| < \epsilon\right)$, rather than the fluxes themselves.

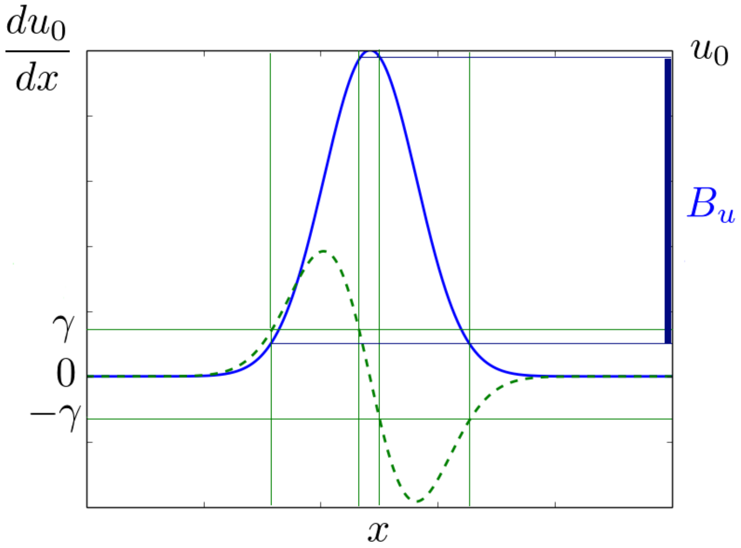


Figure 2-1: An illustration of B_u defined in Theorem 1. The blue line is u_0 and the green dashed line is $\frac{du_0}{dx}$. B_u is the set of u_0 where the derivative $\frac{du_0}{dx}$ has an absolute value larger than γ.

The remainder of this chapter is organized as follows.

- Section 2.1.1 discusses the choices of $\mathcal{S}_F$ occurred in (2.4).
- Section 2.1.2 develops a procedure that adaptively refines the parameterization.
- Section 2.1.3 summarizes the algorithm for training the twin model.
- Section 2.1.4 presents a numerical shortcut for (2.4) that is more computationally efficient.
- Section 2.2 demonstrates the algorithm in several numerical examples.
- Section 2.3 summarizes the chapter.

2.1.1 Parameterization

As discussed in Section 1.3.3, F can be parameterized by a linear combination of basis functions. Firstly, 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 [99]. 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 appeared and has large enough slope. Therefore, an ideal parameterization should admit local refinements so $\tilde{F}$ can match F better at some domain. 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.

A parameterization that allows local refinements is the wavelet parameterization [99]. The wavelet is a set of basis functions developed for multi-resolution analysis (MRA) [99]. MRA introduces an increasing sequence of closed function spaces $\{V_j\}_{j\in\mathbb{Z}}$,

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

[99]. For univariate MRA, V_j's satisfy the following properties known as self-similarity [99]:

$$\begin{aligned} f(u) \in V_j &\Leftrightarrow f(2u) \in V_{j+1},\ j \in \mathbb{Z} \\ f(u) \in V_j &\Leftrightarrow f(u - \frac{\eta}{2^j}) \in V_j,\ j \in \mathbb{Z}, \eta \in \mathbb{Z} \end{aligned}$$

The function space V_j is spanned by a set of orthonormal bases called the wavelet [99]

$$\hat{\phi}_{j,\eta}(u) = 2^{j/2}\hat{\phi}(2^j u - \eta)\,, \quad \eta \in \mathbb{Z} \tag{2.9}$$

where $\hat{\phi}$ is called the mother wavelet. The equation (2.9) 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

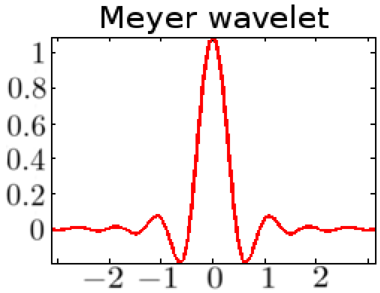


Figure 2-2: An example mother wavelet, the Meyer wavelet.

translation parameter. An example mother wavelet, the Meyer wavelet, is shown in Figure 2-2.

As discussed in 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.

$$\phi_{j,\eta}(u) = \int_{-\infty}^{u} \hat{\phi}_{j,\eta}(u')du' \,. \tag{2.10}$$

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

$$\frac{d\phi_{j,\eta}}{du} = \hat{\phi} \,, \tag{2.11}$$

and

$$\phi_{j,\eta}(u) = \begin{cases} 0, \, u \to -\infty \\ 1, \, u \to \infty \end{cases} \tag{2.12}$$

due to the normality of the wavelet.

Let

$$\phi(u) = \int_{-\infty}^{u} \hat{\phi}(u')du' \,, \tag{2.13}$$

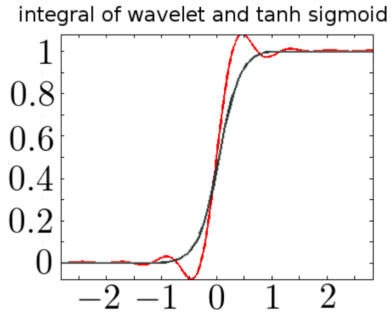


Figure 2-3: Red line: the integral (2.10) *of the Meyer wavelet. Black line: the logistic sigmoid function.*

then

$$\phi(2^j u - \eta) = \int_{-\infty}^{2^j u - \eta} \hat{\phi}(u') du' = \int_{-\infty}^{u} \hat{\phi}(2^j u' - \eta) du' = \int_{-\infty}^{u} \hat{\phi}_{j,\eta}(u') du' \tag{2.14}$$

(2.10) and (2.14) show that $\phi_{j,\eta}$ satisfies the self-similarity property

$$\phi_{j,\eta}(u) = \phi(2^j u - \eta)\,, \quad j \in \mathbb{Z}\,,\ \eta \in \mathbb{Z}\,, \tag{2.15}$$

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,

$$\phi(u) = \frac{1}{1 + e^{-u}}\,. \tag{2.16}$$

If $\tilde{F}$ is univariate, the logistic sigmoids $\phi_{j,\eta}$'s are used as the bases. If $\tilde{F}$ is multivariate, the basis can be formed by the tensor product of univariate sigmoids [102],

$$\phi_{\boldsymbol{j},\boldsymbol{\eta}}(u_1, \cdots, u_k) = \phi_{j_1,\eta_1}(u_1) \cdots \phi_{j_k,\eta_k}(u_k)\,, \tag{2.17}$$

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

$$\tilde{F} = \sum_{\boldsymbol{j}\in\mathbb{Z}^k,\boldsymbol{\eta}\in\mathbb{Z}^k} \alpha_{\boldsymbol{j},\boldsymbol{\eta}}\phi_{\boldsymbol{j},\boldsymbol{\eta}}\,, \tag{2.18}$$

where α's are the coefficients of the bases. There are infinite number of bases involved in this expression, making it infeasible to be implemented in the computer. To address this issue, a systematic procedure for choosing a suitable subset of the bases will be presented in Section 2.1.2.

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]

$$\frac{\partial u}{\partial t} + \frac{\partial}{\partial x}\Big(\underbrace{\frac{u^2}{1+2(1-u)^2}}_{F}\Big) = c\,, \tag{2.19}$$

with the initial condition $u(0,x) = u_0(x)$ and the periodic boundary condition $u(t,0) = u(t,1)$. The Buckley-Leverett equation models the two-phase porous media flow where u stands for the saturation of a phase, and the saturation is always positive and no larger than one. so $0 \le u_0(x) \le 1$ 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

$$\frac{\partial \tilde{u}}{\partial t} + \frac{\partial}{\partial x}\tilde{F}(\tilde{u}) = c\,, \tag{2.20}$$

with the same c and the same initial and boundary conditions. $\tilde{F}$ is parameterized by (2.18) where j's and η's are chosen ad hoc. Figure 2-4 gives an example of the bases used in this section. To ensure the well-posedness of (2.4), an ad hoc L_1 regularization on α is used in minimizing $\mathcal{M}$.

Figure 2-5a shows the discretized space-time solution of (2.19) for $x \in [0,1]$, $t \in [0,1]$ and $c = 0$. The solution is used to train a twin model according to (2.4). The discretized space-time solution of the trained twin model is shown in Figure 2-5b.

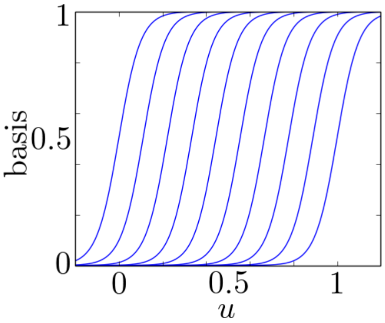
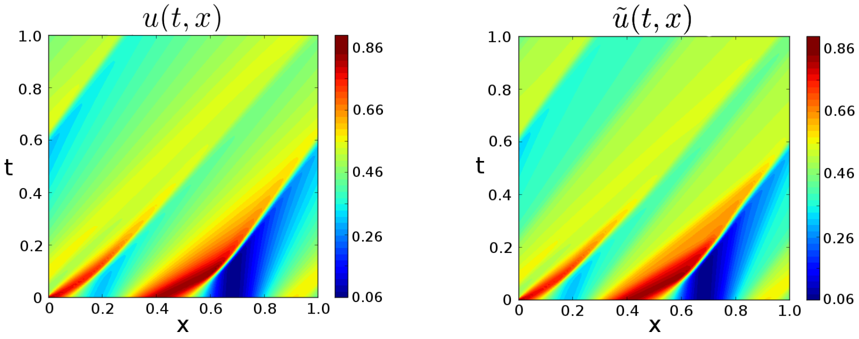


Figure 2-4: An ad hoc set of bases



(a) Gray-box model.

(b) Trained twin model.

Figure 2-5: Space time solutions.

Once a twin model is trained, its adjoint can be used for gradient estimation. Consider an objective function

$$\xi(c) \equiv \int_{x=0}^{1} \left(u(1,x;c) - \frac{1}{2}\right)^2 \mathrm{d}x\,. \tag{2.21}$$

Its gradient $\frac{d\xi}{dc}$ can be estimated by the trained twin model. Figure 2-6 shows the objective function, evaluated using the gray-box model and the trained twin model. It is observed that the gradients of ξ match closely at $c = 0$, i.e. the control where the twin model is trained.

In addition to the gradient estimation, the inferred $\tilde{F}$ is examined. If different solutions are used in the training, it is expected that the trained twin model will also

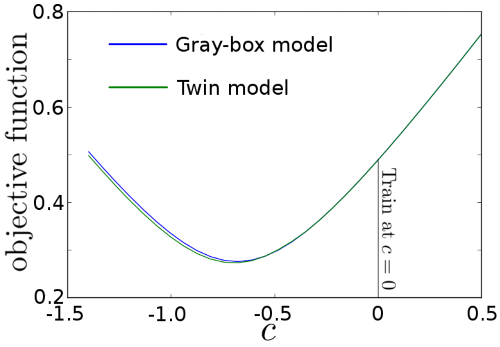


Figure 2-6: The objective function ξ evaluated by either the gray-box model or the trained twin model.

be different. Figure 2-7 shows the training results for three different initial conditions. Some observations can be made: 1) As expected, $\tilde{F}$ can differ from F by a constant without affecting $\mathcal{M}$; 2) $\frac{d\tilde{F}}{du}$ matches $\frac{dF}{du}$ only in a domain of u where the solution exists (indicated by the green area); 3) Sometimes the bases seem redundant thus can be safely dropped out. The issue seems particularly important in the third column, where most bases are suppressed; 4) Sometime the bases seem too coarse thus may be refined in order to reduce the minimal $\mathcal{M}$. The issue seems particularly important in the first column, where $\frac{d\tilde{F}}{du}$ exhibits a wavy deviation from $\frac{dF}{du}$. Addressing these issues systematically is crucial to the rigorous development of the twin model method.

2.1.2 Elements for Adaptive Basis Construction

This section develops several key elements that lead to the adaptive basis construction for twin models. The heuristics for the adaptive basis construction, discussed in Section 1.3.3, are applied to build up a basis dictionary consisted of only the significant candidates. The section is organized as follows: Firstly, a formulation is provided to efficiently assess the significance of each candidate basis; Secondly, the neighborhood of a sigmoid basis is defined; Thirdly, a metric is devised that determines when to add or remove a candidate basis. The three elements are then employed to build the twin model algorithm in Section 2.1.3.

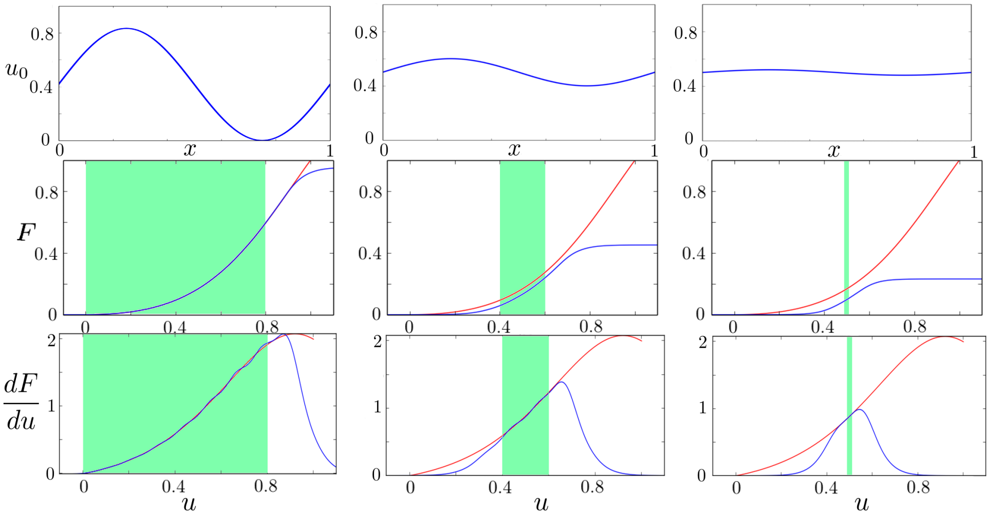


Figure 2-7: The first row shows the three different initial conditions used to generate the gray-box space-time solution. The second row compares the trained $\tilde{F}$ (blue) and the Buckley-Leverett F (red). The third row compares the trained $\frac{d\tilde{F}}{du}$ (blue) and the Buckley-Leverett $\frac{dF}{du}$ (red). The green background highlights the domain of u where the gray-box space-time solution exists.

Given a basis dictionary $\boldsymbol{\phi}_{\mathcal{A}} = \{\phi_i\}_{i\in\mathcal{A}}$, define the "minimal mismatch"

$$\mathcal{M}^*(\mathcal{A}) = \min_{\boldsymbol{\alpha}_{\mathcal{A}}\in\mathbb{R}^{|\mathcal{A}|}} \mathcal{M}\left(\sum_{i\in\mathcal{A}} \alpha_i\phi_i\right) , \tag{2.22}$$

to be the minimal solution mismatch (2.2) if $\tilde{F}$ were parameterized by $\boldsymbol{\phi}_{\mathcal{A}}$. $\mathcal{A}$ is a set containing $\{\boldsymbol{j}, \boldsymbol{\eta}\}$'s. $\boldsymbol{\alpha}_{\mathcal{A}} = \{\alpha_i\}_{i\in\mathcal{A}}$ is the coefficient for $\boldsymbol{\phi}_{\mathcal{A}}$. Let $\boldsymbol{\alpha}^*_{\mathcal{A}} = \{\alpha^*_i\}_{i\in\mathcal{A}}$ be the optimal coefficients, and let $\tilde{F}^*_{\mathcal{A}} = \sum_{i\in\mathcal{A}} \alpha^*_i\phi_i$. Consider appending $\boldsymbol{\phi}_{\mathcal{A}}$ by an additional basis ϕ_l, and let $\boldsymbol{\phi}_{\mathcal{A}'} = \{\boldsymbol{\phi}_{\mathcal{A}}, \phi_l\}$, $\mathcal{A}' = \{\mathcal{A}, l\}$. The minimal mismatch for the appended basis dictionary $\boldsymbol{\phi}_{\mathcal{A}'}$ is

$$\mathcal{M}^*(\mathcal{A}') = \min_{\boldsymbol{\alpha}_{\mathcal{A}'}\in\mathbb{R}^{|\mathcal{A}|+1}} \mathcal{M}\left(\sum_{i\in\mathcal{A}'} \alpha_i\phi_i\right) , \tag{2.23}$$

Clearly $\mathcal{M}^*(\mathcal{A}') \le \mathcal{M}^*(\mathcal{A})$. Define a "mismatch improvement" to be

$$\Delta\mathcal{M}^*\left(\mathcal{A}, l\right) = \mathcal{M}^*(\mathcal{A}) - \mathcal{M}^*(\mathcal{A}') \tag{2.24}$$

Approximate (2.24) by Taylor expansion, we get

$$\Delta\mathcal{M}^*\left(\mathcal{A}, l\right) \approx -\left(\int_{u\in\mathbb{R}^k} \frac{d\mathcal{M}}{d\tilde{F}}\bigg|_{\tilde{F}^*_{\mathcal{A}}} \phi_l \,\mathrm{d}u\right)\alpha_l\,, \tag{2.25}$$

where $\frac{d\mathcal{M}}{d\tilde{F}}$ is the derivative of $\mathcal{M}(\tilde{F})$ with respect to $\tilde{F}$, evaluated on $\tilde{F} = \tilde{F}^*_{\mathcal{A}}$. For a twin model consisted of a system of k equations, $\tilde{F}$ is a function of $u \in \mathbb{R}^k$, thus $\frac{d\mathcal{M}}{d\tilde{F}}$ is also a function of $u \in \mathbb{R}^k$. As discussed in the previous sections, $\frac{d\mathcal{M}}{d\tilde{F}}$ is non-zero only in a domain where there is solution. Thus (2.25) can be integrated by quadrature only over a bounded domain. For example, for a uniform Cartesian space-time grid, the quadrature weights equal to a constant. The absolute value of the coefficient for α_l,

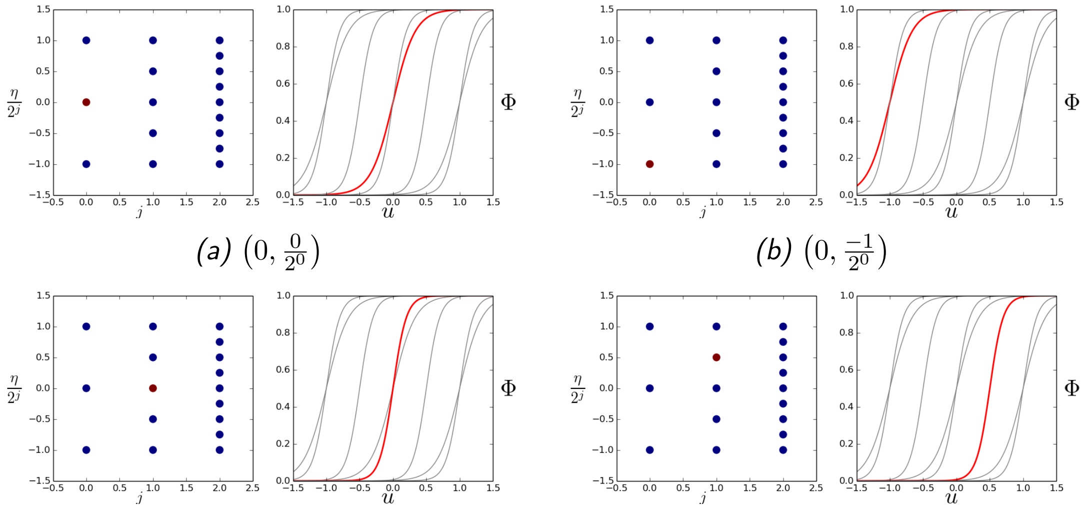
$$s_l(\mathcal{A}) \equiv \left|\int_{u\in\mathbb{R}^k} \frac{d\mathcal{M}}{d\tilde{F}}\bigg|_{\tilde{F}^*_{\mathcal{A}}} \phi_l \,\mathrm{d}u\right|\,, \tag{2.26}$$

estimates the significance of the basis ϕ_l [36]. If there are multiple candidate bases, (2.26) can be used to rank their significance.

In the sequel, a compact representation of the sigmoid bases is introduced. The univariate basis function, $\phi_{j,\eta}$ in (2.15), is represented by a tuple $(j, \frac{\eta}{2^j})$, where j can be viewed as the "resolution", and $\frac{\eta}{2^j}$ is the center of the basis. Similarly, the k-variate basis function, $\phi_{\boldsymbol{j},\boldsymbol{\eta}}$ in (2.17), is represented by a tuple $\left(\boldsymbol{j},\, \frac{\boldsymbol{\eta}}{2^{\boldsymbol{j}}}\right) = \left(\{j_1, \cdots, j_k\},\, \left\{\frac{\eta_1}{2^{j_1}}, \cdots, \frac{\eta_k}{2^{j_k}}\right\}\right)$. Thus, a sigmoid can be represented by a point in a $2k$-dimensional space. The representation is illustrated in Figure 2-8a thru. 2-8d for the univariate case.

Using this representation, define the "neighborhood" of a univariate sigmoid $(j, \frac{\eta}{2^j})$ to be

$$\mathcal{N}\left[\left(j, \frac{\eta}{2^j}\right)\right] = \left\{\left(j+1, \frac{\eta}{2^j}\right),\, \left(j, \frac{\eta \pm 1}{2^j}\right)\right\}\,. \tag{2.27}$$



(a) $\left(0, \frac{0}{2^0}\right)$ *(b)* $\left(0, \frac{-1}{2^0}\right)$

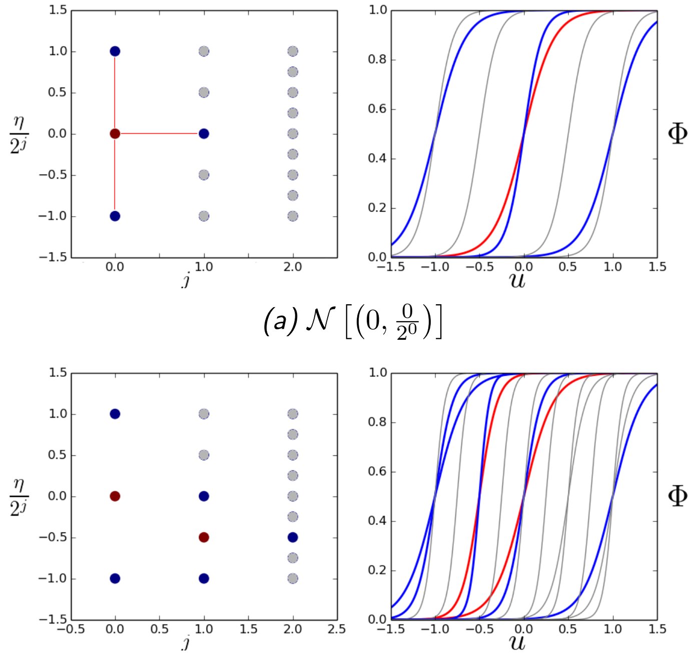
(c) $\left(1, \frac{1}{2^0}\right)$ *(d)* $\left(1, \frac{1}{2^1}\right)$

Figure 2-8: An illustration of the tuple representation and the corresponding univariate sigmoid.

The neighborhood contains 1) a basis $\left(j+1, \frac{\eta}{2^j}\right)$ with an increment of resolution; and 2) two basis $\left(j, \frac{\eta \pm 1}{2^j}\right)$ with the same resolution but a marginal shift of center. For illustration, the neighborhood of $\left(0, \frac{0}{2^0}\right)$ is shown in Figure 2-9a. Similarly, define the neighborhood of a multivariate sigmoid to be

$$\begin{aligned}
&\mathcal{N}\left[\left(\boldsymbol{j}, \frac{\boldsymbol{\eta}}{2^{\boldsymbol{j}}}\right)\right] = \mathcal{N}\left[\left(\{j_1, \cdots, j_k\}, \left\{\frac{\eta_1}{2^{j_1}}, \cdots, \frac{\eta_k}{2^{j_k}}\right\}\right)\right] \\
=&\Bigg\{ \left(\{j_1+1, \cdots, j_k\}, \left\{\frac{\eta_1}{2^{j_1+1}}, \cdots, \frac{\eta_k}{2^{j_k}}\right\}\right) \cdots, \left(\{j_1, \cdots, j_k+1\}, \left\{\frac{\eta_1}{2^{j_1}}, \cdots, \frac{\eta_k}{2^{j_k+1}}\right\}\right), \\
&\left(\{j_1, \cdots, j_k\}, \left\{\frac{\eta_1 \pm 1}{2^{j_1}}, \cdots, \frac{\eta_k}{2^{j_k}}\right\}\right) \cdots, \left(\{j_1, \cdots, j_k\}, \left\{\frac{\eta_1}{2^{j_1}}, \cdots, \frac{\eta_k \pm 1}{2^{j_k}}\right\}\right) \Bigg\},
\end{aligned} \tag{2.28}$$

which consists of k bases with incremental resolution, and $2k$ bases with center shifts. It is easy to see that a basis $(\boldsymbol{j}_0, \frac{\boldsymbol{\eta}_0}{2^{\boldsymbol{j}_0}})$ can be connected to any basis $(\boldsymbol{j}, \frac{\boldsymbol{\eta}}{2^{\boldsymbol{j}}})$ with $\boldsymbol{j} \geq \boldsymbol{j}_0$ through a chain of neighborhoods. In addition, define the neighborhood of multiple sigmoid functions to be the union of each individual's neighborhood, as illustrated by



(a) $\mathcal{N}\left[\left(0, \frac{0}{2^0}\right)\right]$

(b) $\mathcal{N}\left[\left(0, \frac{0}{2^0}\right), \left(1, \frac{-1}{2^1}\right)\right]$

Figure 2-9: Neighborhood for univariate bases. (a) shows the neighborhood (blue) of a single basis (red). (b) shows the neighborhood (blue) of several bases (red). The left column represents the basis on the $\left(j, \frac{\eta}{2^j}\right)$ plane, and the right column shows the actual basis $\phi_{j,\eta}$.

Figure 2-9b.

$$\mathcal{N}\left[(\boldsymbol{j}_1, \frac{\boldsymbol{\eta}_1}{2^{\boldsymbol{j}_1}}), \cdots, (\boldsymbol{j}_n, \frac{\boldsymbol{\eta}_n}{2^{\boldsymbol{j}_n}})\right] = \mathcal{N}\left[(\boldsymbol{j}_1, \frac{\boldsymbol{\eta}_1}{2^{\boldsymbol{j}_1}})\right] \bigcup \cdots \bigcup \mathcal{N}\left[(\boldsymbol{j}_n, \frac{\boldsymbol{\eta}_n}{2^{\boldsymbol{j}_n}})\right] . \tag{2.29}$$

Although the mismatch improvement, $\Delta\mathcal{M}^*(\mathcal{A}, l)$, is always non-negative, it is inadvisable to cram the basis dictionary with too many bases, otherwise a twin model can be overfitted. Therefore, a criterion is required to determine if a candidate basis shall be added to or removed from the basis dictionary. This can be achieved by cross validation, in particular, k-fold cross validation [37]. Given a basis dictionary, the k-fold cross validation proceeds in the following three steps: Firstly, 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 $k = 2$ is shown in Figure 2-10.

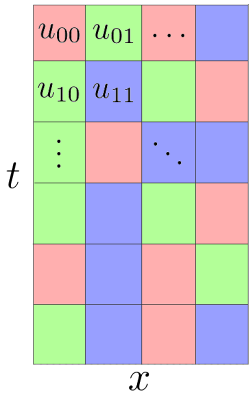


Figure 2-10: The discretized gray-box solution is shuffled into 3 sets, each indicated by a color.

Secondly, k twin models who share the same basis dictionary are trained so their space-time solutions match all but one sets, shown in (2.30). T_i indicates the ith twin model.

$$\begin{aligned} T_1 &= \texttt{TrainTwinModel}(\boldsymbol{u}_2, \boldsymbol{u}_3, \cdots, \boldsymbol{u}_k) \\ T_2 &= \texttt{TrainTwinModel}(\boldsymbol{u}_1, \boldsymbol{u}_3, \cdots, \boldsymbol{u}_k) \\ &\cdots \\ T_k &= \texttt{TrainTwinModel}(\boldsymbol{u}_1, \boldsymbol{u}_2, \cdots, \boldsymbol{u}_{k-1}) \end{aligned} \tag{2.30}$$

Thirdly, each trained twin model is validated on the remaining set. In particular, the solution mismatch for the validation set is computed, as shown in (2.31).

$$\begin{aligned} \mathcal{M}_1 &= \texttt{MismatchValidation}\,(T_1, \boldsymbol{u}_1) \\ \mathcal{M}_2 &= \texttt{MismatchValidation}\,(T_2, \boldsymbol{u}_2) \\ &\cdots \\ \mathcal{M}_k &= \texttt{MismatchValidation}\,(T_k, \boldsymbol{u}_k) \end{aligned} \tag{2.31}$$

The mean value of validation errors,

$$\overline{\mathcal{M}} = \frac{1}{k} \left(\mathcal{M}_1 + \mathcal{M}_2 + \cdots + \mathcal{M}_k \right) \tag{2.32}$$

measures the performance of the basis dictionary. A basis shall be added to or removed from the dictionary only if such action reduces $\overline{\mathcal{M}}$. In practice, cross validation proliferates the computational cost. Therefore, a small k is preferrable if cost is a concern. All the numerical examples in the thesis use $k = 2$.

2.1.3 Algorithm

Based upon the developments in the previous sections, a twin model algorithm with adaptive basis construction is devised.

```
Input: Initial basis dictionary $\boldsymbol{\phi}_{\mathcal{A}}$, coefficients $\alpha_{\mathcal{A}} = \mathbf{0}$, Validation error $\overline{\mathcal{M}}_0 = \infty$,
    Gray-box solution $\boldsymbol{u}$.
Minimize solution mismatch $\alpha_{\mathcal{A}} \leftarrow \mathrm{argmin}_{\alpha} \mathcal{M}\left(\sum_{i\in\mathcal{A}} \alpha_i \phi_i\right)$
loop
    Find $\phi_l \in \mathcal{N}(\boldsymbol{\phi}_{\mathcal{A}})\backslash\boldsymbol{\phi}_{\mathcal{A}}$ with the maximal $s_l(\mathcal{A})$
    $\mathcal{A} \leftarrow \mathcal{A}\bigcup\{l\}$, $\boldsymbol{\phi}_{\mathcal{A}} \leftarrow \boldsymbol{\phi}_{\mathcal{A}}\bigcup\{\phi_l\}$, $\alpha_l = 0$, $\alpha_{\mathcal{A}} \leftarrow \{\alpha_{\mathcal{A}}, \alpha_l\}$
    Compute $\overline{\mathcal{M}}$ by $k$-fold cross validation.
    if $\overline{\mathcal{M}} < \overline{\mathcal{M}}_0$ then
        $\overline{\mathcal{M}}_0 \leftarrow \overline{\mathcal{M}}$
        $\alpha_{\mathcal{A}} \leftarrow \mathrm{argmin}_{\alpha} \mathcal{M}\left(\sum_{i\in\mathcal{A}} \alpha_i \phi_i\right)$
    else
        $\mathcal{A} \leftarrow \mathcal{A}\backslash\{l\}$, $\boldsymbol{\phi}_{\mathcal{A}} \leftarrow \boldsymbol{\phi}_{\mathcal{A}}\backslash\{\phi_l\}$, $\alpha_{\mathcal{A}} \leftarrow \alpha_{\mathcal{A}}\backslash\{\alpha_l\}$ break
    end if
    Find $\phi_{l'} \in \boldsymbol{\phi}_{\mathcal{A}}$ with the least $s_{l'}(\mathcal{A})$
    if $l' \neq l$ then
        $\mathcal{A} \leftarrow \mathcal{A}\backslash\{l'\}$, $\boldsymbol{\phi}_{\mathcal{A}} \leftarrow \boldsymbol{\phi}_{\mathcal{A}}\backslash\{\phi_{l'}\}$, $\alpha_{\mathcal{A}} \leftarrow \alpha_{\mathcal{A}}\backslash\{\alpha_{l'}\}$
        Compute $\overline{\mathcal{M}}$ by $k$-fold cross validation.
        if $\overline{\mathcal{M}} < \overline{\mathcal{M}}_0$ then
            $\overline{\mathcal{M}}_0 \leftarrow \overline{\mathcal{M}}$
            $\alpha_{\mathcal{A}} \leftarrow \mathrm{argmin}_{\alpha} \mathcal{M}\left(\sum_{i\in\mathcal{A}} \alpha_i \phi_i\right)$
        else
            $\mathcal{A} \leftarrow \mathcal{A}\bigcup\{l'\}$, $\phi_{\mathcal{A}} \leftarrow \phi_{\mathcal{A}}\bigcup\{\phi_{l'}\}$, $\alpha_{\mathcal{A}} \leftarrow \alpha_{\mathcal{A}}\bigcup\{\alpha_{l'}\}$
        end if
    end if
end loop
Output: $\mathcal{A}$, $\phi_{\mathcal{A}}$, $\alpha_{\mathcal{A}}$.
```

Algorithm 1: Training twin model with adaptive basis construction.

Algorithm 1 adopts the heuristics of the forward-backward iteration discussed in Section 1.3.3. The algorithm starts from training a twin model using a simple basis dictionary. Usually the starting dictionary contains one basis for each dimension with very low resolution. Details of the choice are given in Section 2.2 along with numerical examples. The main part of the algorithm iterates over a forward step (line 3-9) and a backward step (line 10-19). The forward step firstly finds the most promising candidate in the neighborhood of the current dictionary for addition, according to (2.26). If the addition indeed reduces the cross validation error, the candidate is appended to the dictionary; otherwise it is rejected. If the basis is appended, the coefficients are updated by minimizing the solution mismatch, which can be implemented by the Broyden-Fletcher-Goldfarb-Shannon (BFGS) algorithm [39]. The backward step finds the most promising 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 deleted, the coefficients are updated by BFGS again. The iteration exits when the most promising addition no longer reduces the validation error. In the end, the algorithm provides the basis dictionary and its coefficients as the output.

The algorithm requires to train multiple twin models at each iteration. For $k = 2$, 6 twin models are trained if both the forward and the backward step are acceptive. In practice, the trained coefficients at the last iteration usually provide good initial guess for the next iteration. Nonetheless, the algorithm can be costly if the dictionary turns out to have a high cardinality which results in a large number of iterations before the dictionary construction completes. Therefore, a numerical shortcut is provided in Section 2.1.4 that significantly reduces the cost.

2.1.4 Minimizing the Truncation Error

In the previous sections, a twin model is trained to minimize the solution mismatch. The training can be expensive. Because the minimization of the solution mismatch, coupled with the adaptive basis construction, can require a large number of solution mismatch evaluations, and each evaluation involves one twin model simulation. To reduce the computational cost, a "pre-training" step is proposed where an "integrated truncation error" is minimized. A pre-trained twin model is then "fine tuned" to minimize the solution mismatch. The applicability of the pre-training is studied; in particular, I study under what condition can the solution mismatch be bounded by the integrated truncation error. Finally, a stochastic gradient descent approach is adopted that efficiently minimizes the integrated truncation error.

Define

$$\tau = \frac{\partial u}{\partial t} + \nabla \cdot \big(D\tilde{F}(u)\big) - q(u,c)\,, \tag{2.33}$$

which is the residual if the gray-box PDE's solution is plugged in the twin-model PDE (2.1). Let its discretization be $\boldsymbol{\tau}$. For simplicity, assume the gray-box simulator and its twin model use the same space-time discretization. $\boldsymbol{\tau}$ can be obtained by plugging the discretized gray-box solution in the twin model simulator. Define the integrated truncation error to be

$$\mathcal{T}(\tilde{F}) = \sum_{i=1}^{M}\sum_{j=1}^{N} w_{ij}\boldsymbol{\tau}_{ij}^2\,, \tag{2.34}$$

where w_{ij} are the same quadrature weights as in (2.2). i, j are the indices for time and space discretization as in the previous sections. I propose to pre-train a twin model using Algorithm 1 with $\mathcal{M}$ replaced by $\mathcal{T}$. In other words, in the pre-training step, the coefficients are determined by

$$\alpha_{\mathcal{A}} \leftarrow \operatorname*{argmin}_{\alpha} \mathcal{T}\left(\sum_{i\in\mathcal{A}} \alpha_i\phi_i\right)\,. \tag{2.35}$$

Besides, the estimator for the significance of a candidate basis, $s_l(\mathcal{A})$, is replaced by

$$s_l^t(\mathcal{A}) \equiv \left| \int_{u\in\mathbb{R}^k} \frac{d\mathcal{T}}{d\tilde{F}}\bigg|_{\tilde{F}^*_{\mathcal{A}}} \phi_l \,\mathrm{d}u \right|\,. \tag{2.36}$$

Finally, the validation error, $\overline{\mathcal{M}}$, is replaced by

$$\overline{\mathcal{T}} = \frac{1}{k}\left(\mathcal{T}_1 + \mathcal{T}_2 + \cdots + \mathcal{T}_k\right)\,, \tag{2.37}$$

where

$$\mathcal{T}_i = \texttt{IntegratedTruncationError}(T_i, \boldsymbol{u}_i)\,. \tag{2.38}$$

for $i = 1, \cdots, k$. Using the pre-trained basis dictionary $\phi^t_{\mathcal{A}}$, the twin model is then fine tuned by minimizing the solution mismatch, where $\alpha^t_{\mathcal{A}}$ is used as the initial guess and is adjusted according to (2.4). For a simulation with implicit schemes, the residual and the integrated truncation error are cheaper to evaluate than the solution mismatch, thereby the benefit of the pre-train.

However, $\mathcal{M}$ may not be bounded by $\mathcal{T}$. A sufficient condition under which the bound exists is provided by Theorem 2.

Theorem 2. *Consider a twin model simulator whose one-step time marching is*

$$\mathcal{G}_i : \mathbb{R}^N \mapsto \mathbb{R}^N,\ \tilde{\boldsymbol{u}}_{i\cdot} \to \tilde{\boldsymbol{u}}_{i+1\cdot} = \mathcal{G}_i \tilde{\boldsymbol{u}}_{i\cdot}\,, \quad i = 1, \cdots, M-1\,. \tag{2.39}$$

Assume the quadrature weights are time-independent, i.e. $w_{ij} = w_j$ for all i, j. If $\mathcal{G}_i$ satisfies

$$\|\mathcal{G}_i a - \mathcal{G}_i b\|_W^2 \le \beta \|a - b\|_W^2\,, \tag{2.40}$$

with $\beta < 1$, for any $a, b \in \mathbb{R}^N$ and for all i, then

$$\mathcal{M} \le \frac{1}{1-\beta}\mathcal{T}\,, \tag{2.41}$$

where

$$\|v\|_W^2 \equiv v^T \begin{pmatrix} w_1 & & \\ & \ddots & \\ & & w_N \end{pmatrix} v \tag{2.42}$$

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

The proof is given in Appendix A.2. If the twin model is a contractive dynamical system [38], as given by (2.40), then the solution mismatch can be bounded by the integrated truncation error. In contrast, the bound may not exist for non-contractive dynamical systems, for example for systems that exhibit bifurcation. It is a future work to further investigate the applicability of the pre-training theoretically, in particular, to investigate the necessary and sufficient condition for the bound. My thesis will explore the usefulness of the pre-training by several numerical test cases.

Because the residual $\boldsymbol{\tau}$ can be evaluated explicitly given the gray-box solution, the evaluation can be decoupled for different space-time grid points $\{i, j\}$. By viewing the truncation error at each $\{i, j\}$ as a stochastic sample, (2.35) can be solved by stochastic gradient descent, Algorithm 2.

```
Input: $\alpha = \alpha_0$
for $(i, j) = (1, 1)$ to $(M, N)$ do
    if not converged then
        $\alpha \leftarrow \alpha - \lambda w_{ij} \frac{\partial}{\partial \alpha} \boldsymbol{\tau_{ij}}$
    else
        break
    end if
end for
Output: $\alpha$
```

Algorithm 2: Minimizing the integrated truncation error by stochastic gradient descent.

$\lambda > 0$ is a tunable step size. λ can tuned manually to increase convergence speed while avoiding divergence [81]. In practice, it is beneficial to compute the gradient against more than one grid points (called a "mini-batch") at each iteration. This is because the code can take advantage of vectorization libraries rather than computing the residual at each grid point separately.

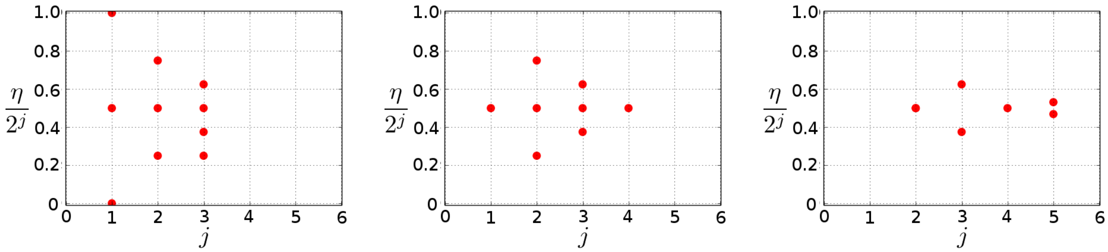
2.2 Numerical Results

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

2.2.1 Buckley-Leverett Equation

Section 2.1.1 has applied a sigmoid parameterization to the gray-box model governed by the Buckley-Leverett equation (2.19). In this section, the same problem is studied but using the adaptive basis construction developed in Section 2.1.3 and 2.1.4. The initial dictionary, $\phi_{\mathcal{A}}$, is selected to contain a single basis $\left(0, \frac{0}{2^0}\right)$. Clearly the choice is not unique. As long as the initial basis has a low resolution and is centered around $[u_{\min}, u_{\max}]$, Algorithm 1 shall build the dictionary adaptively.

Figure 2-11 shows the selected bases for the three solutions in Figure 2-7, respectively, by using the pre-train step. As $[u_{\min}, u_{\max}]$ shrinks, the dictionary's cardinality reduces and the resolution increases.



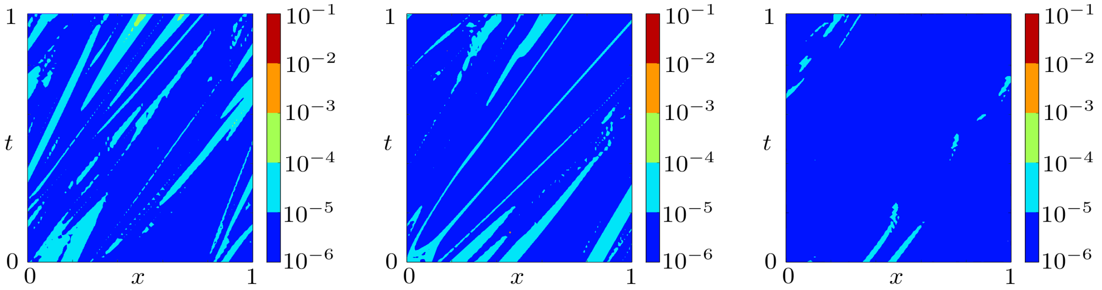
(a) Solution 1 *(b) Solution 2* *(c) Solution 3*

Figure 2-11: The basis dictionary for the three solutions in Figure 2-7.

Consider a time-space-dependent control $c = c(t, x)$ in (2.19) and (2.20). The gradient of ξ, (2.21), is estimated using 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 2-12.

The adaptive basis construction improves the accuracy of the gradient estimation. Table 2.1 shows the integrated gradient error [1] by using either the ad hoc bases in Figure 2-4 or by using the bases constructed adaptively.

[1] The gradient error is integrated using the same quadrature rule as in (2.2).



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

Figure 2-12: The errors of estimated gradients for the three solutions.

	Solution 1	Solution 2	Solution 3
Ad hoc 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: The integrated errors of the estimated gradients for the three solutions.

2.2.2 Navier-Stokes Flow

Consider a compressible internal flow in a 2-D return bend channel driven by the pressure difference between the inlet and the outlet. The return bend is bounded by no-slip walls. The inlet static pressure and the outlet pressure are fixed. The geometry of the return bend is given in Figure 2-13. The inner and outer boundaries of the bending section are each generated by 6 control points using quadratic B-spline.

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

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

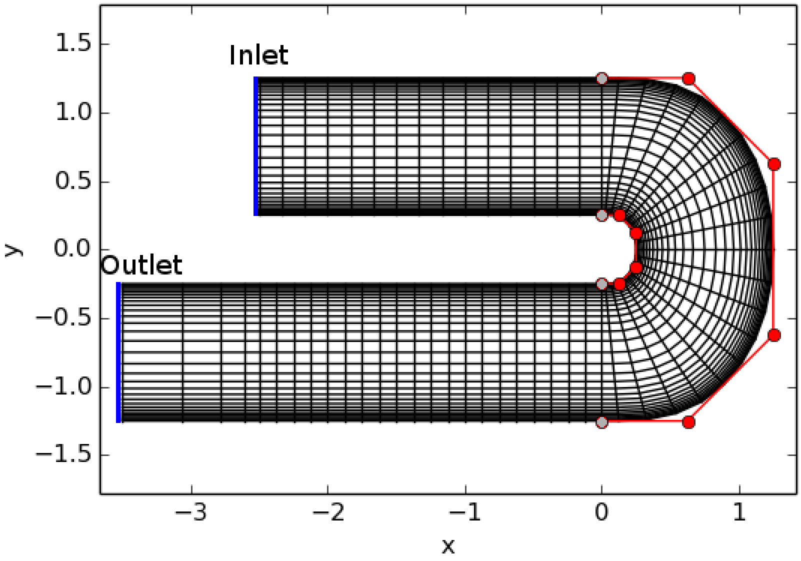


Figure 2-13: The return bend geometry and the mesh for the simulation.

where

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

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

$$p = p(U, \rho)\,, \tag{2.45}$$

where U denotes the internal energy per unit volume,

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

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 [100]. In the sequel, the true state equation in the gray-box simulator is assumed unknown and 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 Cartesian velocity, and E_∞ be the steady state energy density. The steady state mass flux is

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

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

Two state equations are tested: the ideal gas equation and the Redlich-Kwong equation, given by

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

where $a_{rk} = 10^7$ and $b_{rk} = 0.1$.

The solution mismatch, (2.2), is given by

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

where w_ρ, w_u, w_v, and w_E are non-dimensionalization constants. Figure 2-14 shows the gray-box solution and the solution mismatch after training the twin model [2]. Figure 2-15 compares the true state equation and the corresponding trained state equation, where the convex hull of (U_∞, ρ_∞), the internal energy and the density of the gray-box solution, is shown by the dashed red line. Because the state equation is expected to be inferrable only inside the domain of the gray-box solution, large deviation is expected outside the convex hull.

[2] Both the solution and the mismatch are normalized.

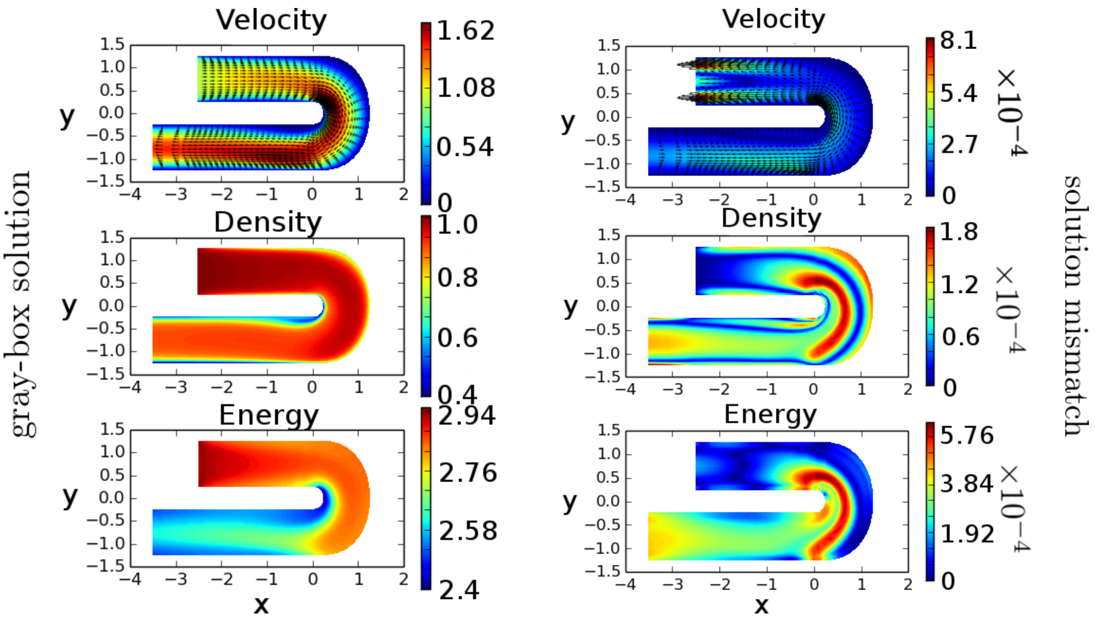


Figure 2-14: Left column: an example gray-box solution for a given geometry. Right column: the solution mismatch after training a twin model.

The trained twin model enables the adjoint gradient estimation. Figure 2-16 shows the estimated gradient of ξ with respect to the control points coordinates. It also compares the estimated gradient with the true gradient. The two gradients are indistinguishable, and the error is given in Table 2.2.

Gas	Interior control points				Exterior control points			
Ideal	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 2.2: The error of the gradient estimation, in percentage.

2.2.3 Polymer Injection in Petroleum Reservoir

Water flooding is a technique to enhance the secondary recovery in petroleum reservoirs, as illustrated in Figure 2-17. Injecting pure water can be cost-inefficient due to low water viscosity and high water cut. Therefore, water-solvent polymer can be utilized

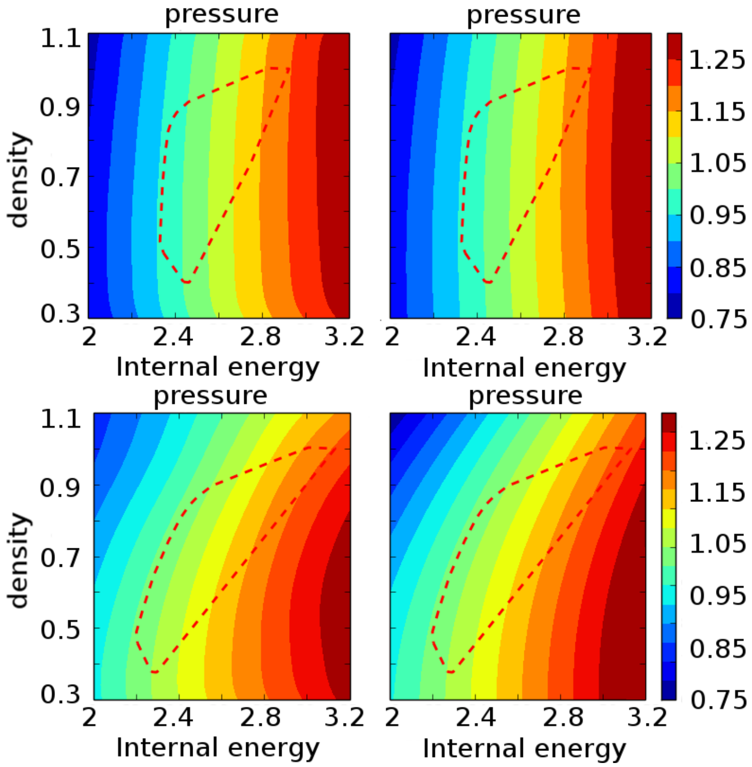
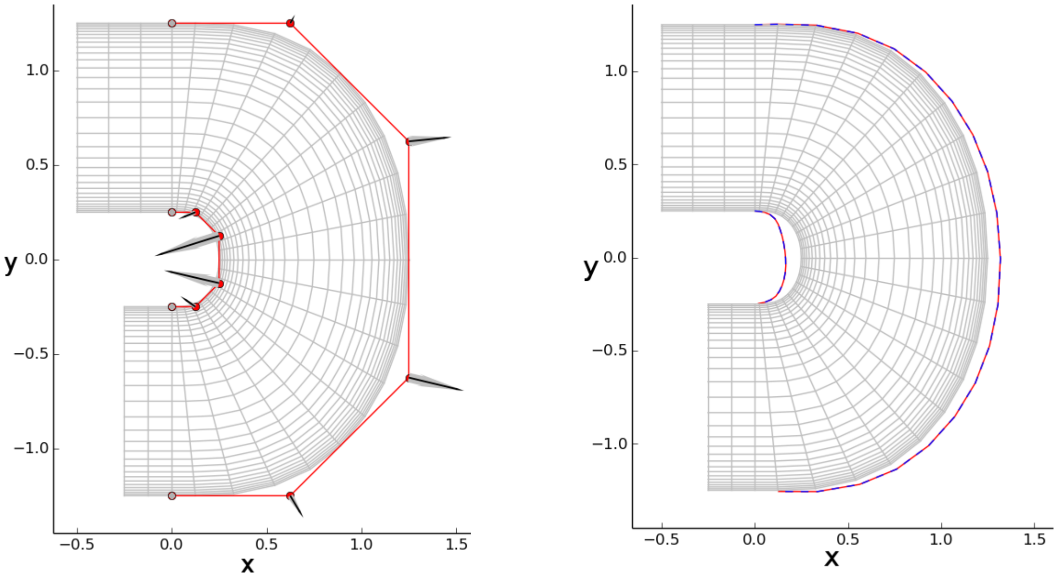


Figure 2-15: The gray-box state equation (right column) and the trained state equation (left column). The gray-box model uses either the ideal gas equation (first row) or the Reclich-Kwong equation (second row). The convex hull of the gray-box solution is shown by the dashed red line.



(a) The 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 gradient evaluated by the twin model using finite difference.

(b) The boundary perturbed according to the gradient. The blue dashed line is computed by finite difference of the gray-box model, while the red dashed line is computed by the twin model's gradient.

Figure 2-16: A comparison of the estimated gradient and the true gradient.

to increase the water-phase viscosity and to reduce the residual oil.

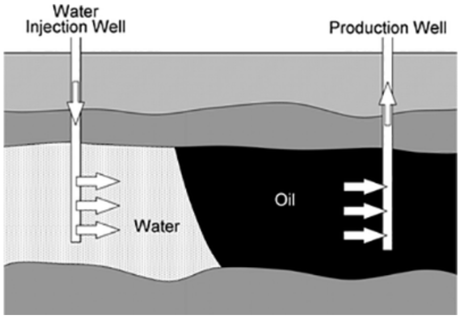


Figure 2-17: Water flooding in petroleum reservoir engineering (courtesy from PetroWiki).

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. In the sequel, the models for the mobility factors are unknown. The only knowledge about the mobility factors is that they depend on S_w, p, and c.

PSim, the simulator aforementioned in Section 1.1, is used as the gray-box simulator, which uses the IMPES time marching, i.e. implicit in pressure and explicit in saturation, as well as the upwind scheme. Its solution, S_w, c, and p can be used to train the twin model. The twin model uses fully implicit time marching and the upwind scheme. The solution mismatch is defined by

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

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

Consider a reservoir setup shown in Figure 2-18, 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 $10^6 \texttt{ft}^3/\texttt{day}$ is used at both the injectors. The reservoir is simulated for $t \in [0, 50]\texttt{day}$. The solution of S_w is illustrated in Figure 2-19 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.

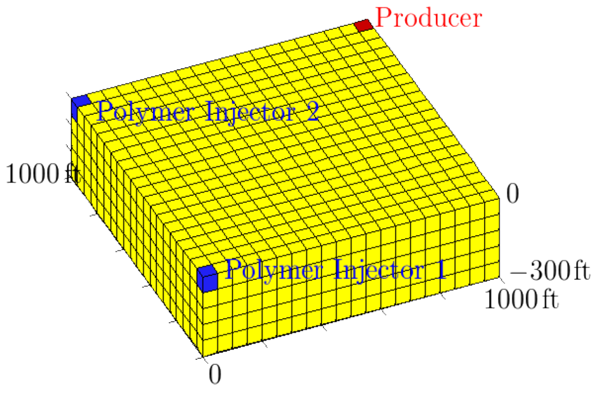
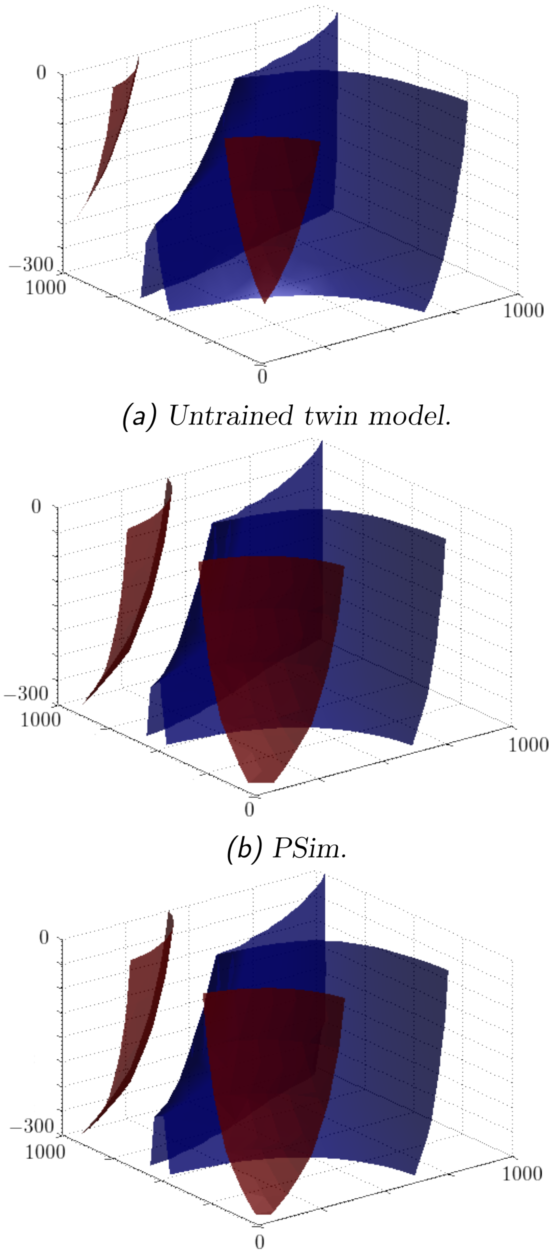


Figure 2-18: The geometry of the petroleum reservoir.

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

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



(a) Untrained twin model.

(b) PSim.

(c) Trained twin model.

Figure 2-19: The isosurfaces of $S_w = 0.25$ and $S_w = 0.7$ at $t = 30$ days.

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 2-20, 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 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, although the error slightly increases for smaller t, possibly because of the different numerical schemes used in the twin and gray-box models. The error is given in Table 2.3.

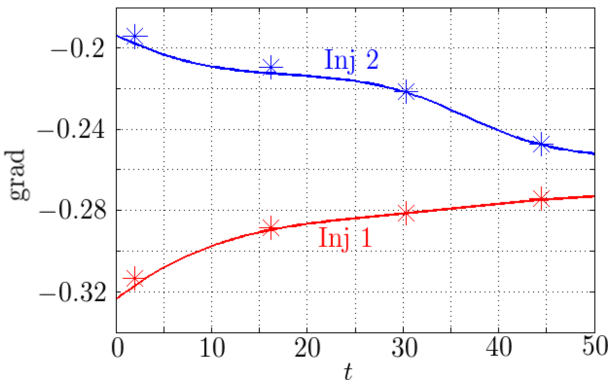


Figure 2-20: The gradient of ξ with respect to rates at the two injectors. The lines indicate the gradients estimated by the twin model, while the stars indicate the true gradient evaluated by finite difference.

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 2.3: The error of estimated gradient at day 2, 16, 30, and 44, in percentage.

2.3 Chapter Summary

This chapter develops a method for gradient estimation by using the space-time solution of gray-box conservation law simulations. In particular, an adjoint-enabled twin model is trained to minimize the solution mismatch metric. The inferrability of the twin model is studied theoretically for a simple PDE with only one equation and one dimensional space. To enable the training computationally, a sigmoid parameterization is presented. However, an ad hoc choice for the bases does not fully exploit the information contained in the gray-box solution. To address this issue, an adaptive basis construction procedure is presented. The adaptive procedure builds upon three key elements: the approximated basis significance, the basis neighborhood, and the cross validation. The algorithm for training the twin model is summarized. To alleviate the training cost, a pre-train step is suggested that minimizes the integrated truncation error instead of the solution mismatch.

The proposed twin model algorithm has a wide applicability, which 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 (3),

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

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 a desired objective evaluation, as well as to reduce the overall computational cost. The chapter is organized as follows. Section 3.1.1 develops the probabilistic ingredients for the Bayesian optimization. These ingredients are applied to build an algorithm in Section 3.1.2. Its convergence properties are investigated in Section 3.1.3. Finally, the algorithm is demonstrated in Section 3.2 through several numerical examples.

3.1 Approach

3.1.1 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 is not exactly the true gradient for several reasons. For example, the discretized gray-box solution can be under-resolved, 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 $\tilde{F}$ that yields the minimal solution mismatch may not be exactly F. It is difficult to identify the various sources of errors and unrealistic to quantify all the errors separately. Instead, my thesis models the gradient error as its entirety without distinguishing the sources of errors.

Given a gray-box solution, the trained twin model is deterministic. Assume the numerical schemes of the gray-box simulator to be deterministic too. Then the gradient error is deterministic. Besides, the gray-box solutions are generally correlated for different controls. Thus the twin models and their gradient estimation errors are also correlated. The deterministic and correlated error can be modeled as a realization of Gaussian process [60, 61, 62]. Let $\nabla\xi$ be the true gradient, $\xi_{\tilde{\nabla}}$ be the estimated gradient [1], and $\xi_{\tilde{\nabla}i}$ be its ith component. The relationship between $\nabla\xi$ and $\xi_{\tilde{\nabla}}$ can be modeled by [60, 61, 62]

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

for $i = 1, \cdots, d$. (3.1) has the idiosyncratic noise term removed because $\nabla\xi$ and $\xi_{\tilde{\nabla}}$ are deterministic.

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

[1] $\tilde{\xi}(c)$ is the objective evaluation provided by a twin model trained using the gray-box solution at c. $\xi_{\tilde{\nabla}}(c)$ is the adjoint gradient estimated by the twin model. If $\tilde{\xi}$ is differentiable, there is another gradient, $\nabla\tilde{\xi}$, which can be evaluated by finite difference. Usually $\nabla\tilde{\xi} \neq \xi_{\tilde{\nabla}}$. Notice $\xi_{\tilde{\nabla}}$ may not be a conservative vector field.

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 $\underline{c}_n$ [2]. 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

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

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

[2] The notations are consistent with Section 1.3.1. The objective and estimated gradient evaluations are assumed to be collocated, which will be revealed in Section 3.1.2.

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

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

$$\boldsymbol{E} = \begin{pmatrix} \nabla_{c_1} \nabla_{c'_1} K(c_1, c'_1) & \cdots & \nabla_{c_1} \nabla_{c'_N} K(c_1, c'_N) \\ \vdots & \ddots & \vdots \\ \nabla_{c_1} \nabla_{c'_N} K(c_N, c'_1) & \cdots & \nabla_{c_N} \nabla_{c'_N} K(c_N, c'_N) \end{pmatrix}, \tag{3.8}$$

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

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

The derivation of (3.7) and (3.8) can be found in [71].

The Matérn 5/2 kernel is used for K and G_i's, in particular,

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

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

Let θ denote the hyper parameters L_ξ, σ_ξ, L_{G_i}'s, σ_{G_i}'s, and μ. θ can be estimated by log maximum likelihood. The likelihood of observing $\xi(\underline{c}_n)$ and $\xi_{\tilde{\nabla}}(\underline{c}_n)$ is given by

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

which is marginalized over $\nabla\xi(\underline{c}_n)$. 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 has the closed form

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

which can be optimized efficiently using GBO methods such as StoGo, as discussed in Section 1.3.1. Given the joint distribution (3.3), the posterior of $\xi(c)$, for any $c \in \mathbb{R}^d$, can be obtained by (3.1.1),

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

Using the posterior, the acquisition function, $\rho(c)$, can be constructed and optimized to find the next evaluation point. My thesis will use the expected improvement acquisition function. See Section 1.3.1 for the details.

3.1.2 Algorithm

Based upon the review in Section 1.3.1 and the developments in Section 3.1.1, I present a Bayesian optimization algorithm with bound constraints $c_{\min} \leq c \leq c_{\max}$, 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 $\xi_0^*$. 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: $c_0^*$, $\xi_0^*$
```

Algorithm 3: Bayesian optimization with twin model.

In line 3 of Algorithm 3, it is beneficial to reuse twin models, because the twin models that are trained on previous controls may be good initial guesses for the current training. If the gray-box solutions are similar, it is expected that the trained twin models may be similar too. A rigorous investigation of the topic is a future work. Instead, I suggest a tentative procedure to reuse trained twin models as follows: 1) When running Algoithm 3, store $\boldsymbol{u}(c)$ and the trained twin models at all iterates; 2) At each iterate, find a previously trained twin model whose solution is closest to the current solution according to (2.2); 3) Loop over the backward step in Algorithm 1 to prune all possible redundant bases of the twin model; and 4) Apply Algorithm 1 to train the twin model as usual.

3.1.3 Convergence Properties Using True Hyper Parameters

This section investigates the convergence properties of Algorithm 3. For Bayesian optimization with only the objective evaluation, the convergence properties have

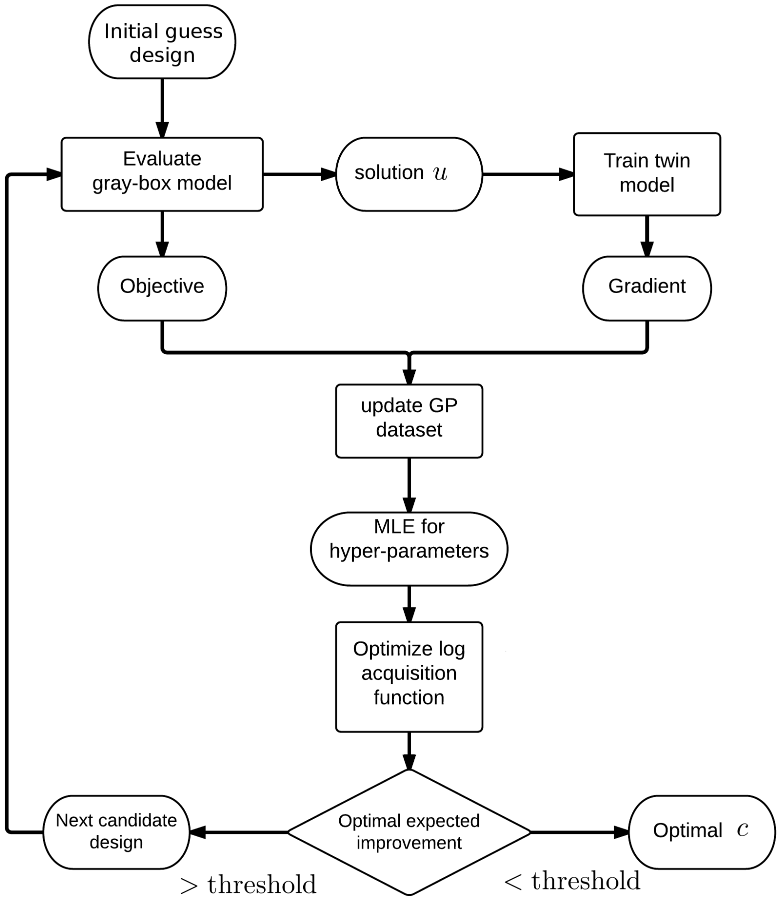


Figure 3-1: The flowchart of Algorithm 3.

been explored in the past. M. Locatelli [70] proved that Bayesian optimization with EI acquisition generates a dense search sequence for the 1-D optimization problem $c^* = \operatorname{argmax}_{c \in [0,1]} \xi(c)$, if ξ is a realization of the Wienner process. E. Vazquez [64] generalized the results by showing that the sequence is still dense for higher dimensional space and for more general classes of stochastic processes. Recently, A. Bull [65] showed that Bayesian optimization with EI has a convergence rate at $\mathcal{O}(n^{-\nu/d})$, where $\nu > 0$ is a constant parameter controlling the kernel smoothness. Similar results have been given for UCB acquisition. N. Srinivas [66] bounded the convergence rate from above at $\mathcal{O}(n^{-\frac{\nu}{2\nu+d(d+1)}})$ for UCB, and also established the relationship between the convergence rate and the information gain due to evaluating the objective. In this section, I analyze the convergence properties of Algorithm 3 when the true hyper parameters are used. My contribution is to extend the convergence analysis to incorporate estimated gradient evaluations. Under the assumptions in Section 3.1.1, it is proven that the search sequence is indeed dense. The conclusion implies that the algorithm is able to find the optimal as $n_{\max} \to \infty$, regardless of the gradient estimation quality.

The true hyper parameters values are taken as known constants throughout the section. Without loss of generality, assume the objective function to be a realization of zero-mean stationary Gaussian process. The assumptions aforementioned in Section 3.1.1 are reiterated more formally 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 reproducing kernel Hilbert space $\mathcal{H}_{K_\nabla}$ with 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 [63]). 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 $\mathcal{H}_G \equiv \mathcal{H}_G^1 \otimes \cdots \otimes \mathcal{H}_G^d$.

Let the stochastic dependence of ξ to be ω_ξ, and the stochastic dependence of ϵ_i to be 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. Then

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

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 $\omega_\xi^* \in \Omega_\xi$, and 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 random 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 $\xi_{\tilde{\nabla}}(\underline{c}_n)$ are 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

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

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

The following theorem holds, which is proven in Appendix A.3.

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* $(1+|\eta|^2)^k \big|\hat{\Phi}(\eta)\big| \geq C$ *for all* $\eta \in \mathbb{R}^d$*, and if* $n_{\max} \to \infty$ *and* $\mathtt{EI}_{\min} = 0$*, then* $\underline{c}_n$ *is dense in* $\mathcal{C}$ *for all* $c_{init} \in \mathcal{C}$*, all* $\xi \in \mathcal{H}_K$ *and all* $\epsilon_i \in \mathcal{H}_G^i$*, for* $i = 1, \cdots, d$*.*

In the limiting case of $n_{\max} \to \infty$ and $\mathtt{EI}_{\min} = 0$, the theorem implies that Algorithm 3 can find the maximum regardless of the accuracy of the gradient estimation if the true hyper parameters are known. It is a future work to extend the theory when the hyper parameters are estimated.

3.2 Numerical Results

This section demonstrates the optimization algorithm on several numerical examples.

3.2.1 Buckley-Leverett Equation

Consider the same problem setup as in Section 2.2.1. Represent the source term by 25 parameters

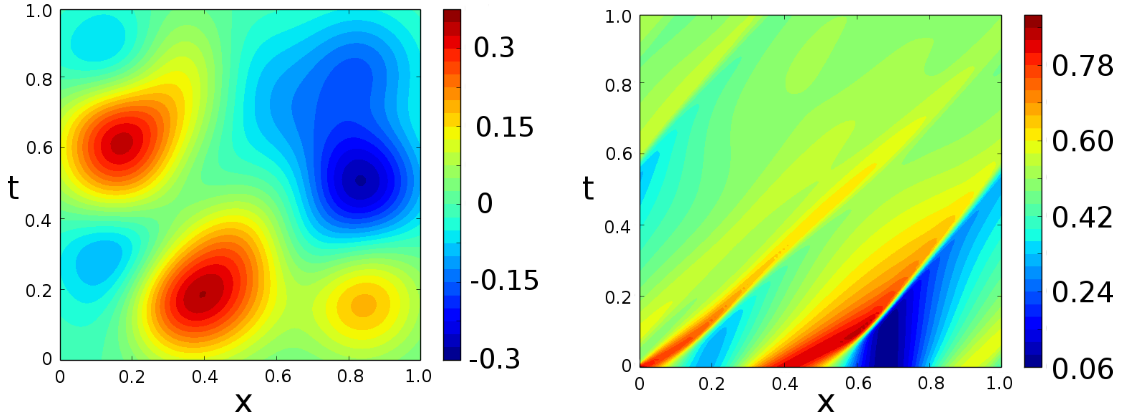
$$\begin{aligned} c(t,x) &= \sum_{i=1}^{5}\sum_{j=1}^{5} c_{ij}\, B_{ij}(t,x) \\ B_{ij} &= \exp\left(-\frac{(t-t_i)^2}{L_t^2}\right)\exp\left(-\frac{(x-x_j)^2}{L_x^2}\right) \end{aligned}, \tag{3.22}$$

where $L_t = L_x = 0.15$, and $(t_1, \cdots, t_5) = (x_1, \cdots, x_5) = \mathtt{linspace(0,1,5)}$. 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 $-1 \leq c_{ij} \leq 1$ for $i, j = 1, \cdots, 5$.

Figure 3-2a shows the optimized source term. Figure 3-2b shows the corresponding gray-box solution. Notice the solution at $t = 1$ is close to $\frac{1}{2}$ due to the optimized source.



(a) The optimized source term.

(b) The optimized gray-box solution.

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

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 either the twin-model Bayesian optimization and the vanilla Bayesian optimization[3], after 20 gray-box simulations. Figure 3-4 shows the current best (minimal) objective at each iterate. The use of twin model accelerates the optimization, especially when the number of iterate is small.

3.2.2 Navier-Stokes Flow

Consider the same Navier-Stokes flow as in Section 2.2.2. Let $S(c)$ be the area of the return bend, which is a function of the control points' coordinates, c. Let S_0 be the area that corresponds to the initial guess of the control points. 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 $c_{\min} \leq c \leq c_{\max}$. Because there are 4 adjustable control points at each boundary, the control is 16-dimensional. Figure 3-5 shows the initial and the optimized geometries as well as

[3] The vanilla Bayesian optimization uses only the objective evaluation.

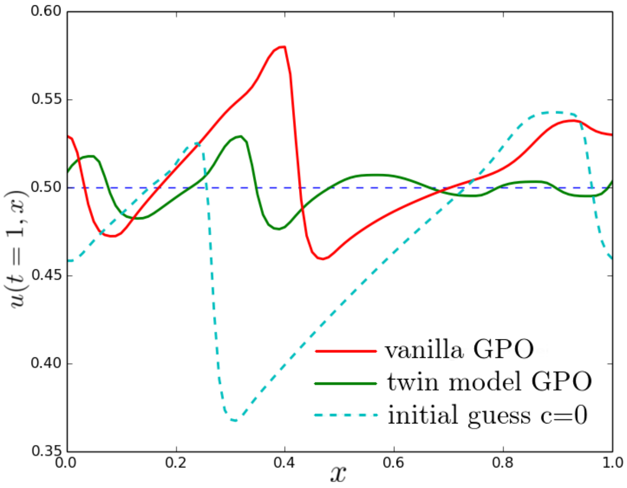


Figure 3-3: *A comparison of the optimized $u(t=1,x)$ after 20 gray-box simulations. The red line is obtained by the vanilla Bayesian optimization and the green line by the twin-model Bayesian optimization. The cyan dashed line indicates the $u(t=1,x)$ obtained by setting the source term to zero.*

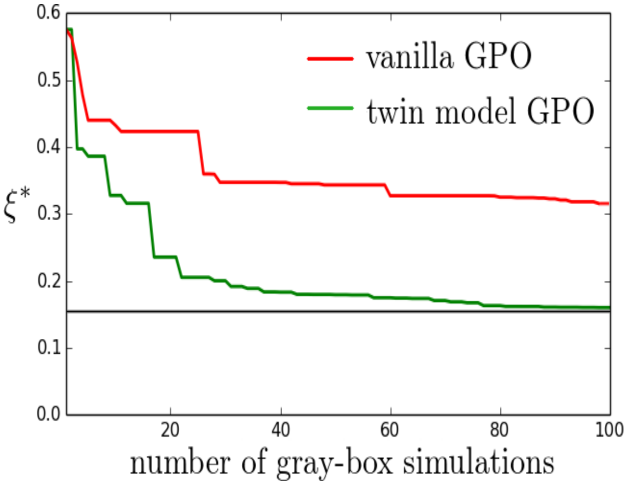


Figure 3-4: *The current best objective at each iterate. The red line is obtained by the vanilla Bayesian optimization and the green line by the twin-model Bayesian optimization. The black horizontal line indicates the true optimal.*

the bound constraints. It also shows the 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, thus decreases the drag and increases the mass flux.

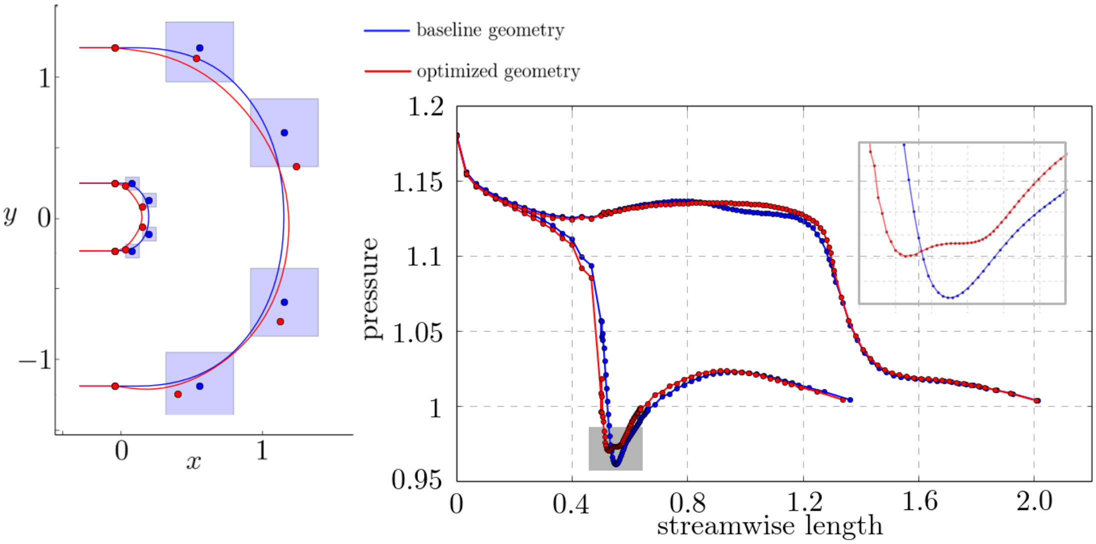


Figure 3-5: The left plot shows the initial guess of control points (blue dots), the initial guess of the geometry (blue line), the optimized control points (red dots), and the optimized geometry (red line). The purple squares indicate the bound constraints for each control point. The right plot shows the pressure along the interior and the exterior boundaries for the initial (blue) and the optimized (red) geometry.

Figure 3-6 shows the current best objective evaluation at each iterate. Twin model enables faster objective improvement than the vanilla Bayesian optimization. In particular, at the 8th iterate, twin-model Bayesian optimization already achieves near optimality. Figure 3-7 shows the wall clock time of the optimization against the number of iterates. Although twin model increases the per-iterate computational cost, the increased cost is offset by faster objective improvement. After around 80 minutes (8 twin-model optimization iterations), twin model achieves near optimality whereas the vanilla Bayesian optimization is still far from optimal.

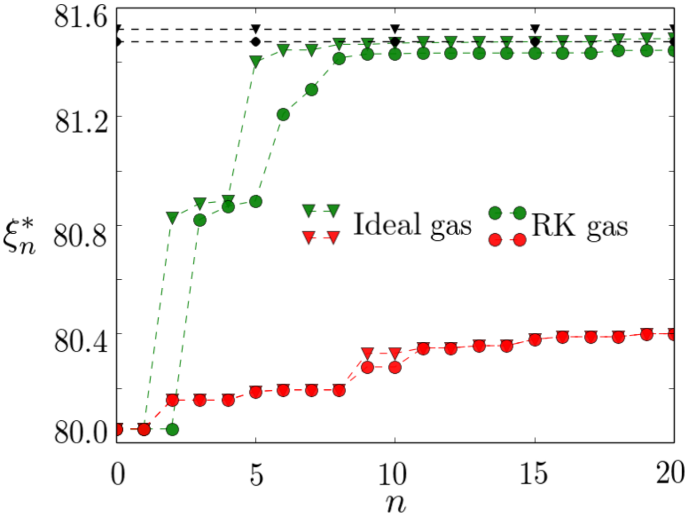


Figure 3-6: The current best objective at each iterate for the ideal gas and the Redlich-Kwong gas. The green lines are obtained by the twin-model Bayesian optimization. The red lines are obtained by the vanilla Bayesian optimization. The black horizontal lines indicate the true optimal.

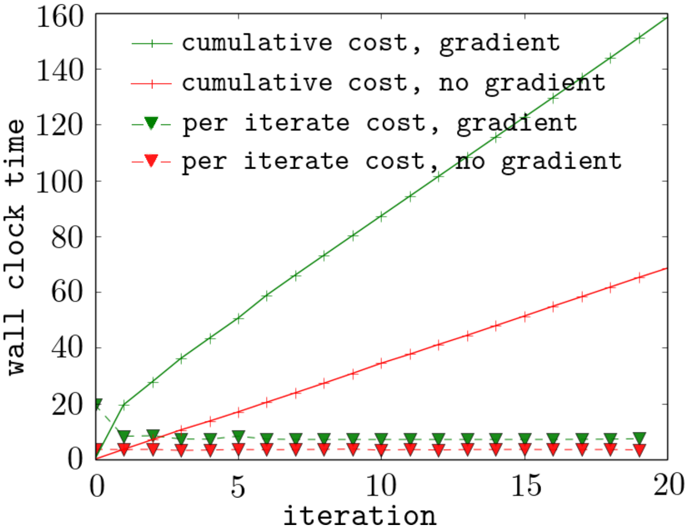


Figure 3-7: The cumulative and per-iterate wall clock time, in minutes.

3.2.3 Polymer Injection in Petroleum Reservoir

Consider a 2D horizontal reservoir governed by (2.49) and (2.50). The permeability is heterogeneous, and is shown in Figure 3-8. Five injectors are placed along the southern boundary, and one producer is placed in the northeastern corner. The reservoir is simulated for $t \in [0, T = 10]\,\mathtt{day}$.

Firstly, consider constant-in-time injection rates at the injectors. Define

$$\xi(t) = -\int_{\Omega} \rho_o(t)\phi S_o(t) d\boldsymbol{x} - \lambda t \sum_{i=1}^{5} I_{\mathtt{inj}i}\,, \tag{3.25}$$

which is the negative oil residual minus the water cost at time t. $\lambda = 0.4$ is the cost of water per unit volume. $I_{\mathtt{inj}i}$ is the injection rate 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 5-dimensional.

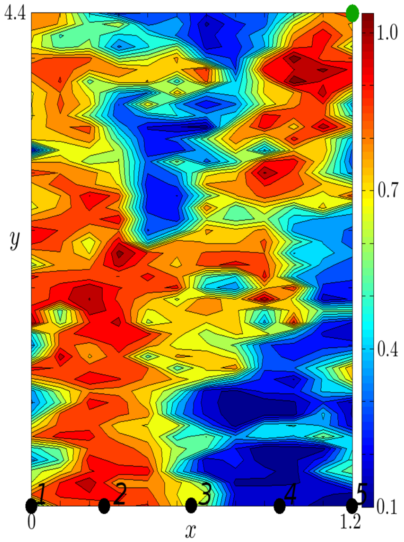


Figure 3-8: The permeability of the reservoir, in 100 milli Darcy. The 5 injectors are indicated by the black dots, and the producer is indicated by the green dot.

Figure 3-9 shows the current best objective evaluation against the number of iterates. The black line indicates the true optimal[4]. The twin model Bayesian optimization achieves near-optimality faster than the vanilla Bayesian optimization. Figure 3-10 shows $\xi(t)$ for the initial and the optimized injection rates. The initial rates are set at $I_{\texttt{inj}i} = I_{\max}$ 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.

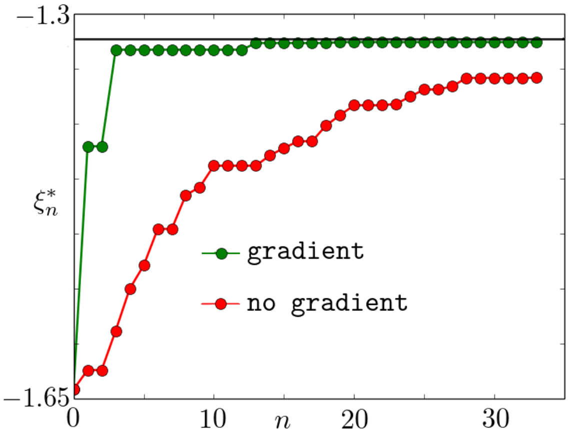


Figure 3-9: The current best objective evaluation against the number of iterates.

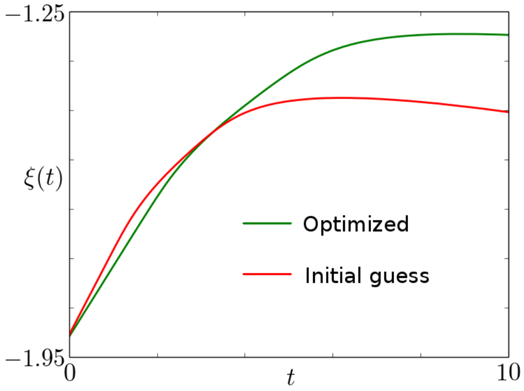


Figure 3-10: $\xi(t)$ for the initial and the optimized injection rates.

[4] The true optimal is obtained by simulated annealing after running 192 gray-box simulations.

Secondly, 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 one thousand dimensional. Clearly the Bayesian optimization algorithm developed in Section 3.1.2 is not long suitable, because the large dimensionality leads to a huge covariance matrix[5]. Instead, the twin model is tested on a simple gradient descent method, the backtracking-Armijo gradient descent method [101]. Clearly, such practice is theoretically flawed, because the gradient error can not be estimated or bounded, and the optimization may not converge at all. However, it is interesting to examine the practical utility of the twin model.

Figure 3-11 shows the optimized injection rates. The 1st and 5th 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 harder to extract. Later, all injectors are turned on, and their rates gradually decrease when the water cut increases. Figure 3-12 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.

3.3 Chapter Summary

Based upon 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 are given, where the hyper parameters are estimated by maximum likelihood. The developments are summarized in a Bayesian

[5] 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. Such scaling problem is suffered by most non-parametric methods.

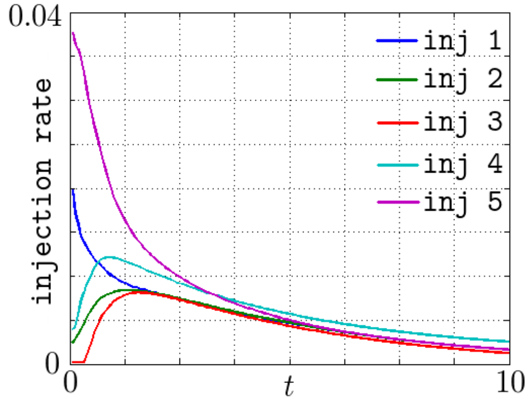


Figure 3-11: The optimized time-dependent injection rates.

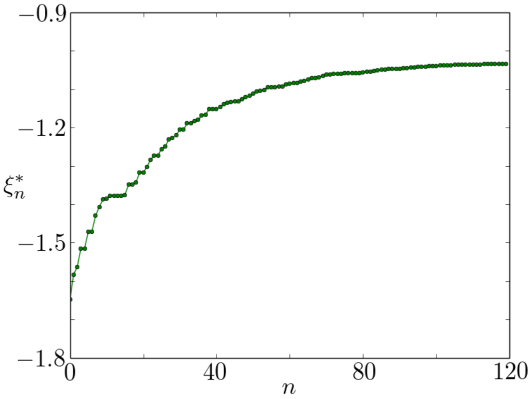


Figure 3-12: The current best objective evaluation using the backtracking-Armijo gradient descent method, where the gradient is provided by the twin model.

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 hyper parameters are used. It is a future work to extend the theory to estimated hyper parameters.

The proposed optimization method is demonstrated on 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 less 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 hard 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 faciliate 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 are prevelant in many engineering applications. In many cases, the code of the simulator is proprietary, legacy, and lacks 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 motivates the need for the adjoint gradient, and explains 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 implement the twin model numerically, the unknown part of the flux function is parameterized by a set of basis. I argue that the sigmoid bases are well suited for this problem since their gradients are local. The procedure is demonstrated on a Buckley-Leverett equation using an ad hoc set of sigmoids. 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. To reduce the computational cost of the algorithm, a pre-train step is suggested that minimizes the integrated truncation error instead of the solution mismatch. The twin model algorithm is demonstrated on a variety of numerical examples, including 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 accurate estimates 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 provide analytical expressions for the posterior distributions and the acquisition function, while the hyper parameters are estimated by maximum likelihood. I present a Bayesian optimization algorithm that utilizes the twin-model gradient. In addition, I show that the algorithm is able to find the optimal regardless of the gradient estimation accuracy, if the true hyper parameters are used. The optimization algorithm is demonstrated on several problems similar to Chapter 2, including 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 less iterations than the vanilla Bayesian optimization. Finally, the twin model gradient is tested on a 1000-dimensional control problem, by employing a simple gradient descent approach. The gradient efficiently enables the optimization of the high-dimensional problem, which represents another major contribution of my thesis.

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. a theoretical and numerical demonstration of the algorithms in a variety of problems.

4.3 Future Work

There are several potential thrusts of further research: A useful extension is to investigate the inferrability of twin models for various conservation laws. In particular, Theorem 1 may be extended for problems with a system of equations and higher spatial dimension. Another interesting extension is to study the applicability of the pre-train step, especially to obtain a necessary and sufficient condition for bounding $\mathcal{M}$ with $\mathcal{T}$. Finally, in the twin-model Bayesian optimization algorithm, it is of great practical value to reuse twin model more efficiently. My current approach uses the twin model with the closest solution as an initial guess, and re-trains the twin model at every iterate. In the future, an important research topic is on how to utilize all previously trained twin models. Another important topic is to employ "trust region" in the optimization: the same twin model can be used multiple times at different controls inside a trust region[1], thus reducing the training cost. Finally, it is interesting to generalize the formulation (1.10) to incorporate unknown source terms and boundary conditions.

[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. 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. See [47] and [48] for the details.

Appendix A

Proof of Theorems

A.1 Theorem 1

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 [98]. Firstly, assume there is no shock wave for (2.7) and (2.8) 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 $T=\frac{\Delta}{\left|\frac{d\tilde{F}}{du}\big(u_0(x_0-\Delta)\big)-\frac{dF}{du}\big(u_0(x_0)\big)\right|}$, we have

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

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 (2.7) or (2.8), we let T^* be the time of the shock occurrence. Without loss of generality, assume the shock occurs for (2.7) first. The shock implies the intersection of two characteristic lines. Choose a $\Delta>0$ such that $\big|\frac{dF}{du}\big(u_0(x)\big)-\frac{dF}{du}\big(u_0(x-\Delta)\big)\big|T^*=\Delta$. Using the mean value theorem, we have

$$T^* = \frac{\Delta}{\overline{\frac{d^2F}{du^2}}\big(u_0(x) - u_0(x-\Delta)\big)} \geq \frac{1}{L_u L_F}$$

Thus, if we choose

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

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

A.2 Theorem 2

Proof:

Let the one-step time marching of the gray-box simulator be

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

The integrated truncation error can be written as

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

Similarly, the solution mismatch can be written as

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

Fig A-1 gives an explanation of $\mathcal{M}$ and $\mathcal{T}$ by viewing the simulators as discrete-time dynamical systems.

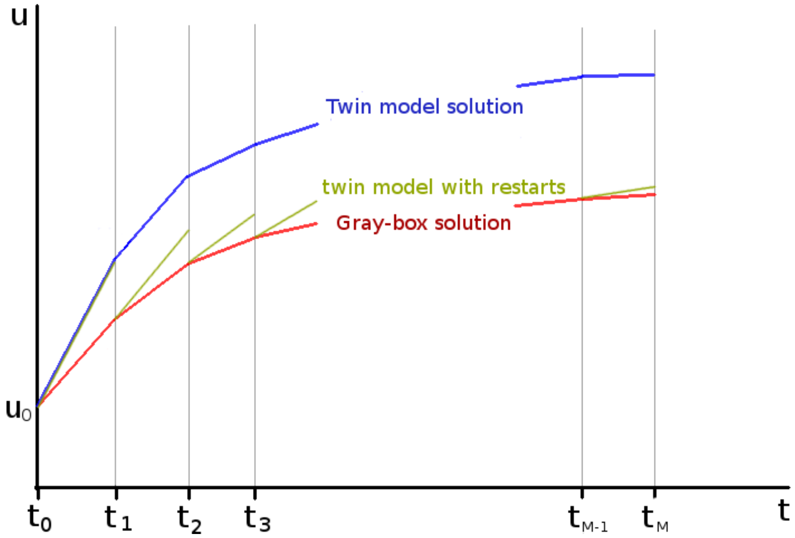


Figure A-1: The state-space trajectories of the gray-box model and the twin model. $\mathcal{M}$ measures the difference of the twin model trajectory (blue) with the gray-box trajectory (red). $\mathcal{T}$ measures the difference of the twin model trajectory with restarts (green) and the gray-box trajectory (red).

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

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

Therefore,

$$\mathcal{M} - \mathcal{T} \leq$$

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

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

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

Reorder the summation, we get

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

Therefore,

$$\mathcal{M} - \mathcal{T} \leq \left(\beta^{M-1} + \beta^{M-2} + \cdots + \beta\right)\mathcal{T}$$

If β is strictly less than 1, then

$$\mathcal{M} \leq \frac{1}{1-\beta}\mathcal{T} \, ,$$

thus completes the proof. ■

A.3 Theorem 3

Proof:

Firstly, we have the following lemma (Chapter 1, Theorem 4.1, [67]).

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

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

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

lemma 2.

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

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

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

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

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

Notice

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

is the objective function, and

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

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-Scharz inequality [67] 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,

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

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 [64], except that the Cauch-Schwarz inequality used in the paper is replaced by lemma 2. We do not repeat the proof but refer to [64] 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 $\hat{\xi}$,

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

where $W^{k,2}$ is the Sobolev space whose weak derivatives up to order k have a finite L^2 norm [65]. 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 $\mathcal{H}_K(\mathbb{R}^d)$ [77]. Besides, we have that C_c^∞ is dense in $W^{k,2}$ (Chapter 2, Lemma 5.1 [78]), where C_c^∞ is the C^∞ functions with compact support on $\mathcal{C}$. As a consequence, $\mathcal{C}_c^\infty \subseteq \mathcal{H}_K$ [64]. 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 [64]) *If the three conditions in lemma 3 are equivalent, $n_{\max} \to \infty$, and $\mathtt{EI}_{\min} = 0$, 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 do not repeat the proof. See [64] for the details. To sum up, under the conditions in Theorem 3, $\underline{c}_n$ is dense in the search space. ■

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] 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.

[7] 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.

[8] 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.

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

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

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

[12] 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.

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

[14] 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.

[15] 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.

[16] 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.

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

[18] 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.

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

[20] 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.

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

[22] 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.

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

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

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

[26] 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.

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

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

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

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

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

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

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

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

[35] 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.

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

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

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

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

[40] 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.

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

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

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

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

[45] 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.

[46] 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.

[47] 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

[48] 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

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

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

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

[52] 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.

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

[54] 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.

[55] 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.

[56] 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.

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

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

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

[60] 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.

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

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

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

[64] 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.

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

[66] 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

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

[68] 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

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

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

[71] 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.

[72] 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

[73] 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.

[74] 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.

[75] 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.

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

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

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

[79] 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.

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

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

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

[83] 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.

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

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

[86] Noel, M. M. A New Gradient Based Particle Swarm Optimization Algorithm for Accurate Computation of Global Minimum. *Applied Soft Computing*, 12(1):353-359, 2012.

[87] Yang, X. S., and Deb, S. Cuckoo Search: Recent Advances and Applications. Neural Computing and Applications, 24(1):169-174, 2014.

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

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

[90] 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.

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

[92] 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.

[93] 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.

[94] 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.

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

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

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

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

[99] 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.

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

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

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

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

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