Adjoint-based gradient estimation for gray-box solutions of unknown conservation laws

Han Chen, Qiqi Wang

1 Abstract

Many engineering applications can be formulated as optimizations constrained by conservation laws. Such optimizations can be efficiently solved by the adjoint method which computes the gradient of the objective to the design variables. However, traditionally, adjoint can not be implemented in many "gray-box" conservation law simulators. In gray-box simulators, the adjoint is not implemented, and the analytical and numerical form of the conservation law is unknown; but the full solution of relevant flow quantities are available. However, adjoint is not available and is not easily implementable in many conservation law simulators. This article introduces a method to estimate the gradient by inferring the governing conservation law from the solution, and then solving the adjoint equation of the inferred conservation law. The method is demonstrated in the sensitivity analysis of a 1-D flow problem.

2 Background

Optimization problems is of great interest in the engineering community. We consider an optimization problem constrained by conservation laws.

$$\begin{aligned} &\min_{c\in\mathbb{R}^d} \left\{ J(u,c) = \int_0^T \int_\Omega j(u,c)\, \mathrm{d}t\mathrm{d}\mathbf{x} \right\} \\ &\text{where } u \text{ satisfies } \frac{\partial u}{\partial t} + \nabla \cdot F(u) = q(u,c)\,, \end{aligned} \tag{1}$$

where c is the design variable to be optimized. F can be a differential operator of u. For example, oil reservoir simulations may employ PDEs of the black-oil model, in which gas, water, and oil phases satisfy a set of conservation laws [6]. We may optimize the well pressure to maximize the total oil production in a time window. Another example is from the turbine airfoil cooling. We are interested in optimizing the interior flow path of turbine airfoil cooling to minimize pressure loss [15, 16].

In many cases, such simulations can be computational costly, potentially due to the complex computational models involved, and large scale time and space discretization. Besides, the design space can be high-dimensional. For example, in oil reservoir simulations the well pressure can be controlled at each well individually, and they can vary in time. To parameterize the well pressure, we require Nm number of design variables, where N is the number of wells, and m is the number of parameters to describe the variation in time for each well. Similarly, in turbine airfoil cooling, the geometry of the internal flow path can also be parameterized by many variables. Optimizing a high-dimensional design can be challenging. A tool to enable efficient high-dimensional optimization is adjoint sensitivity analysis [4] , which computes the gradient of the objective to the design variables efficiently. Continuous adjoint method solves a continuous adjoint equation derived from the conservation law, which requires the PDE of the conservation law. Discrete adjoint method solves a discrete adjoint equation derived from the numerical implementation of the conservation law, which requires the simulator's numerical implementation. Adjoint automatic differentiation applies the chain rule to every elementary arithmetic operation of the simulator, which requires accessing and modifying the simulator's source code.

We are interested in *gray-box* simulations. By gray-box, we mean a conservation law simulation without adjoint implemented. Besides, we are not able to implement adjoint when the governing PDE for the conservation law and its numerical implementation is unavailable, for example when the source code is proprietary or legacy. Another defining property of gray-box simulation is that it can provide the space-time solution of the conservation law. If the simulation solves for time-independent problems, a gray-box simulation should be able to provide the steady state solution. In contrast, we define a simulator to be a black-box, if neither the adjoint nor the solution is available. The only output of such simulations is the value of the objective function to be optimized. If adjoint is implemented or is able to be implemented, we call such simulations *open-box*. We summarize their differences in Table 1.

Table 1: Comparison of black-box, gray-box, and open-box simulations

	PDE and Implementation	Space (or space-time) solution	Adjoint
Black-box	✘	✘	✘
Gray-box	✘	✔	✘
Open-box	✔	✔	✔

Depending on the type of simulations involved, we may choose different optimization methods. If the simulation is black-box, we may use gradient-free optimization. Gradient-free optimization methods require only the availability of objective function values but not derivative information[2]. Gradient-free optimization methods are popular because they are easy to use. However, when the dimension of the design space increases, these methods generally suffer from the *curse of dimensionality*. The term *curse of dimensionality* refers to problems caused by the rapid increase in the search volume associated with adding extra dimension in the search space. The resulting increase of search volume increases the number of objective evaluation required. It's not uncommon to encounter tens or hundreds of dimensions in real life engineering problems, making gradient-free optimization very expensive.

If the simulation is open-box, we may use gradient-based optimization. Gradient-based optimizations use gradient information to locate a local optimum. A well-known example is the quasi Newton's methods [3]. Generally gradient-based methods require less number of simulations to converge than gradient-free methods, and are more efficient at finding local optimum for high-dimensional problems. In addition of requiring $J(c)$ from the simulation, these methods also require $\frac{\partial J}{\partial c}$, the sensitivity. Adjoint methods are efficient methods for sensitivity analysis[4] for open-box models. Continuous adjoint method develops the continuous adjoint equations from the continuous PDE of the simulation through Lagrange multiplier; and it requires the PDE of the simulation. Discrete adjoint method applies variational analysis directly to the discretized PDE; and it requires the discretized PDE (i.e. the numerical implementation) of the simulation. Another popular method to compute sensitivity is automatic differentiation (AD)[5]. AD exploits the fact that every computer program can be broken down into a sequence of elementary arithmetic operations and elementary functions. By applying the chain rule repeatedly to these operations, derivatives can be computed automatically. Because adjoint methods require the accessibility of the PDE, and/or its implementation details, it can not be used to compute the sensitivity of a gray-box simulation either.

If the simulation is gray-box, the gradient is not able to be computed by adjoint. Most current research practices treat the simulation as black-box, and perform gradient free optimization. The gray-box simulation is viewed as a calculator for the objective, while its space-time solution is neglected.

In the remainder of the paper, section 3 introduces the prototype partial differential equations of conservation laws; section 4 discusses the feasibility inferring the unknown conservation laws by using the space-time solution of flow quantities; section 5 discusses the formulation of the inference; section 6 introduces the concept of excited domain and discuss a basis selection scheme for the inference; section 7 demonstrates the method on a 1D flow problem.

3 Prototypes of PDEs

When a simulation is gray-box, we can not apply adjoint to the gray-box conservation law to compute the gradient. Instead, we estimate the gradient by using the space-time solution of the gray-box model.

We propose a two-step procedure to estimate the objective's gradient by using the gray-box simulation's space-time solution. In the first step, we infer the conservation law governing the simulation by using its space-time solution; in the second step, we apply adjoint method to the inferred conservation law to estimate the gradient.

We use the fact that the governing PDE of the gray-box model is a conservation law. For time-dependent problems, the PDE for conservation laws can be written as

$$\frac{\partial}{\partial t}u_i + \nabla \cdot F_i(\mathcal{D}u) = q_i(u,c)\,, \qquad i = 1, \cdots, n\,, \tag{2}$$

where $t \in [0,T]$ is time; $x \in \Omega \subseteq \mathbb{R}^n$ is the spatial coordinate. Ω may depend on the design variables c. q_i's are the source terms that may also depend on c. c is the design variable and may be space-time dependent. The boundary and initial conditions are known. F_i's are the flux functions. The flow variables are $u = \{u_1, \cdots, u_n\}$. $\mathcal{D}u = \left\{u_1, \nabla u_1, \cdots, \nabla^{i_1} u_1; \cdots; u_n, \nabla u_n, \cdots, \nabla^{i_n} u_n\right\}$, where ∇^j indicates the jth order spatial derivative tensor. We assume $i_1, \cdots, i_n$, i.e. the maximum order of derivatives, are known. The discretized space-time solution of Eqn(2) given by a gray-box simulation is written as $\hat{u}(t_i, \mathbf{x}_i; c)\,,\ i = 1, \cdots, N$, where $t = \{t_1, \cdots, t_N\}$ indicates the discretized time, and $\mathbf{x}_i$ indicates the spatial discretization at time t_i.

The objective function is defined by

$$J = \int_0^T \int_\Omega j(u,c)\mathrm{d}\mathbf{x}\mathrm{d}t \tag{3}$$

Notice j can depend explicitly on c, while u depends implicitly on c.

Similarly, for time-independent conservation laws, the prototype is

$$\nabla \cdot F_i(\mathcal{D}u) = q_i(u,c)\,, \qquad i = 1, \cdots, n\,, \tag{4}$$

and the objective function is defined by

$$J = \int_\Omega j(u,c)\mathrm{d}\mathbf{x} \tag{5}$$

Notice the time-independent prototype can be seens as a special case of the time-dependent prototype. In many cases, the solution of time-independent PDEs can be seens as the converged solution of time-dependent PDEs. If we can evolve Eqn(2) until $\frac{\partial u_i}{\partial t} \to 0$, the converged solution is a solution of Eqn(4). In numerical implementations, this corresponds to the pseudo time marching scheme for solve time-independent PDEs. Therefore, we will mostly consider the time-dependent prototype.

For illustration purposes, consider an example for the time-dependent prototype. Consider a simulator modeling two phase flow in porous media. One of the simplest yet classical model for two-phase porous media flow is the Buckley-Leverett model [7, 8]. It models the displacement process of two-phase flow due to capillary pressure and Darcy's law. The PDE, Buckley-Leverett equation, is

$$\frac{\partial u}{\partial t} + \frac{\partial}{\partial x}\left(\frac{u^2}{1 + A(1-u)^2}\right) = c\,, \tag{6}$$

where $x \in [0,1]$ is the space domain; $u = u(t,x)$, $0 \le u \le 1$, is the saturation of phase I (e.g. water), and $1-u$ is the saturation of phase II (e.g. oil); $A > 0$ is a parameter dependent on the physical property of the two phases; $c = c(t,x)$ is the design variable. $c > 0$ models the injection of phase I replacing phase

II; and $c < 0$ vice versa. Suppose A is an unknown, we can use the prototype Eqn(2) to model the PDE.

As an example, we may want to control the flow through c, such that the saturation at $t = T$ is close to a target saturation $u^*(x)$. We parameterize $c(t, x)$ by a set of variables c_i, $i = 1. \cdots, d$. We may introduce a Tikhonov regularization into the objective function to model the control cost. Hereby the objective function

$$\begin{aligned} J &= \int_x |u(T,x) - u^*(x)|^2 \, \mathrm{d}x + \eta \sum_i c_i^2 \\ &= \int_t \int_x \left\{ |u(t,x) - u^*(x)|^2 \, \delta_T(t) + \frac{\eta}{T} \sum_i c_i^2 \right\} \mathrm{d}x\mathrm{d}t \,, \end{aligned} \tag{7}$$

where $\eta > 0$ models the price of control, and $\delta.(\cdot)$ is the Direc delta function. Eqn(7) can be modelled by Eqn(3). The optimization problem is

$$c^* = \arg\min_{c \in \mathbb{R}^d} J \tag{8}$$

4 Infer conservation laws by the space-time solution

We first discuss the reason that infering the conservation law is feasible. Consider a general dynamical system

$$\dot{u} = \mathcal{L}(u) \,, \tag{9}$$

where $u = \{u_1, \cdots, u_n\}$, $u_i = u_i(t, x)$, $i = 1, \cdots, n$, $x \in \mathbb{R}^n$. $\mathcal{L}$ is a differential operator known as the Hamiltonian of the system. Inferring the differential operator can be difficult, however it is not necessary. The flow quantities satisfy a conservation law, and their PDEs can be written as prototypes Eqn(2) or Eqn(4). Therefore, the problem of adapting the physics of the physics-based surrogate reduces to the problem of adjusting a set of functions F_i or q_i, for $i = 1, \cdots, n$.

We use the space-time solution of the gray-box simulations to infer these functions. There are several benefits to use the space-time solution [1]. Firstly, in conservation law simulations, the flow quantities only depend on the flow quantities in an older time inside a *domain of dependence*. When the timestep is small, the domain of dependence can be small too. For example, for scalar conservation laws without exogenous control, we can view solving the conservation law for one timestep Δt as a mapping $\mathbb{R}^{\omega_{\Delta t}} \to \mathbb{R}$, where $\omega_{\Delta t} \in \Omega$ is the domain of dependence. Loosely speaking, by applying such mapping repeatedly to all $x \in \Omega$ and $t \in [0, T]$ (in addition to the boundary and initial conditions), we perform a space-time simulation of the conservation law. Generally, the size of $\omega_{\Delta t}$ is small. Therefore, in the discretized simulation of conservation laws, the number of discretized flow variables involved in $\omega_{\Delta t}$ is small too, making it feasible to infer the mapping.

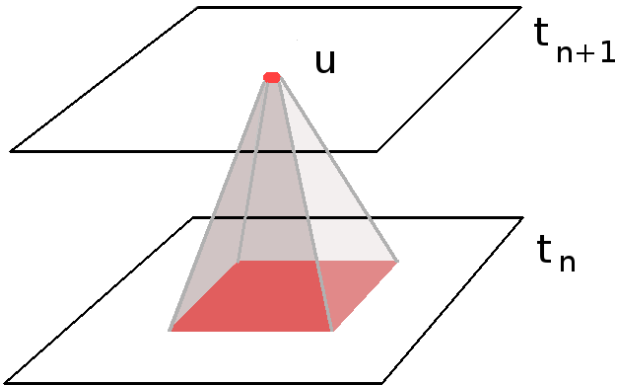


Figure 1: Domain of dependence: in conservation law simulations, the flow quantities at a given location depends on the flow quantities at an older time only within its domain of dependence ω. The domain of dependence can be much smaller then the overall spatial domain Ω when the timestep is reasonably small.

Secondly, the space-time solution at almost *every* space-time grid point can be viewed as a sample for the mapping $\mathbb{R}^\omega \to \mathbb{R}$. Because the number of space-time grid points in gray-box simulations is generally large, we have a large number of samples to infer the mapping. In the prototype PDEs Eqn(2) and Eqn(4), such a mapping is determined by the functions F_i's or q_i's. Therefore, we will have a large number of samples to infer the functions, making the inference potentially accurrate.

Thirdly, in many optimization problems, the design space is high only because the design is space and/or time dependent. In order to parameterize the space-time dependent design, a large number of design variables will be employed. However, the flow quantities only depend on the design variables in the domain of dependence. Therefore, even if the overall number of design variables is high, the number of design variables involved in the mapping is limited, making the inference problem potentially immune to the design space dimensionality.

Therefore, we propose to infer a new type of surrogate using the prototype equations Eqn(2) or Eqn(4) and the space-time solution of the gray-box simulation. The proposed surrogate is called *twin model*.

5 Twin model as an optimization problem

Conventionally, we have a given PDE, and want to compute its space-time solution. However, in twin model we want to infer the governing PDE to match a given space-time solution. Such a problem can boil down to an optimization problem. Finding a suitable prototype equation can be viewed as an inverse problem, which can be solved by optimization. We define a metric for the mismatch of the space-time solutions. Given the same inputs (design variables, initial conditions, boundary conditions, and exogenous inputs), an ideal twin model should give a space-time solution $\tilde{u}$ such that

$$\frac{1}{T}\int_{t=0}^{T}\int_{\mathbf{x}\in\Omega} w^2(t,\mathbf{x},u,\tilde{u})(\tilde{u}-u)^2\,dtd\mathbf{x}\,, \tag{10}$$

is minimized, where $w^2 > 0$ is a weight possibly depending on t, $\mathbf{x}$, u, and $\tilde{u}$. We assume $w \equiv 1$ for simplicity. Notice the square on $\tilde{u} - u$: it is a differentiable expression whose derivative is smooth.

The space-time solution is discretized rather than continuous. If the twin model and the primal model use the same space-time grid, then Eqn(10) can be approximated by

$$\frac{1}{T}\sum_{i=1}^{N}\sum_{k=1}^{T}\left(\tilde{u}_{ik}-u_{ik}\right)^2\Delta t_k\left|\Delta\mathbf{x}_i\right|\,, \tag{11}$$

where $|\Delta \mathbf{x}_i|$ indicates the size of the grid. If the grids are different, then a mapping P from u to $\tilde{u}$ is required. In this case, Eqn(11) would translate to

$$\frac{1}{T}\sum_{i=1}^{N}\sum_{k=1}^{T}\left(\tilde{u}_{ik}-P(u)_{ik}\right)^2\Delta t_k\left|\Delta \mathbf{x}_i\right|\,, \tag{12}$$

Right now, we assume the grids are the same for simplicity.

In the following we assume F_i's are unknown in the prototype equations (the extension to problems with unknown q_i's is straightforward). By parameterizing the functions using a set of basis, the problem of constructing a twin model is converted to the following problem. Notice we added a Tikhonov regularization to avoid ill-posedness.

Solve

$$\xi^* = \arg\min_{\xi} L(\tilde{u}(\xi)) = \arg\min_{\xi}\left\{\frac{1}{T}\sum_{i=1}^{N}\sum_{k=1}^{T}\left(\tilde{u}_{ik}-u_{ik}\right)^2\Delta t_k\left|\Delta \mathbf{x}_i\right|+\lambda\|\xi\|^2\right\}\,, \tag{13}$$

where u is the discretized space-time solution of the primal model, and $\tilde{u}$ is the discretized space-time solution of

$$\frac{\partial \eta_s \tilde{u}_s(t,x)}{\partial t}+\nabla\cdot\left\{\sum_{k=1}^{m_s}\xi_{sk}g_{sk}(\mathcal{D}\tilde{u},\kappa)\right\}=q_s(\tilde{u},c(t,x))\,,\qquad s=1,\cdots,n\,, \tag{14}$$

g_{sk}, $k=1\cdots m_s$ are the library of basis functions for F_s. ξ's are the coefficients for the basis functions. $\lambda\|\xi\|^2$ is the regularization with $\lambda>0$. $\|\cdot\|$ is the L_2 vector norm. The grids of the primal model solver and the twin model solver are assumed the same. Also, the primal model and the twin model use the same design, initial condition, boundary condition, and exogeneous parameters η and κ.

Similarly, for time-independent prototype equations, we have

Solve

$$\xi^* = \arg\min_{\xi} L(\tilde{u}(\xi)) = \arg\min_{\xi}\left\{\sum_{i=1}^{N}\left(\tilde{u}_{ik}-u_{ik}\right)^2\left|\Delta \mathbf{x}_i\right|+\lambda\|\xi\|^2\right\}\,, \tag{15}$$

where u is the discretized spatial solution of the primal model, and $\tilde{u}$ is the discretized spatial solution of

$$\nabla\cdot\left\{\sum_{k=1}^{m_s}\xi_{sk}g_{sk}(\mathcal{D}\tilde{u},\kappa)\right\}=q_s(\tilde{u},c(x))\,,\qquad s=1,\cdots,n\,, \tag{16}$$

g_{sk}, $k=1\cdots m_s$ are the library of basis functions for F_s. ξ's are the coefficients for the basis functions. $\lambda\|\xi\|^2$ is the regularization with $\lambda>0$. $\|\cdot\|$ is the L_2 vector norm. The grids of the primal model solver and the twin model solver are assumed the same. Also, the primal model and the twin model use the same design, boundary condition, and exogeneous parameters κ.

Twin model is an open-box. Therefore, adjoint method can be implemented for the optimization of the parameters ξ.

Next, we consider choosing an appropriate basis library g. Many basis can be used, such as polynomial basis, Fourier basis, and wavelet basis. For the example of inferring the 1D conservation law, we are interested in inferring the flux. The addition of a constant to the flux function does not change the conservation law, therefore we are actually interested in inferring the derivative of the flux. We will use sigmoid functions as the basis, because the derivative of sigmoid functions is almost zero except at the region close to its center.

Although the objective is to minimize $\sum_{i=1}^{N}\sum_{k=1}^{T}\left(\tilde{u}_{ik}-u_{ik}\right)^2\Delta t_k\left|\Delta \mathbf{x}_i\right|$, we will not use it as the objective function directly and perform optimization in one shot. If $\tilde{F}(u)$ deviates from $F(u)$ a lot, then $\tilde{u}(x,t)$ can deviate from $u(x,t)$ significantly even at a small t. Therefore, solving the twin model and its adjoint in $t=[0,T]$ without an educated $\tilde{F}(u)$ can be a waste of computation resources. To improve

efficiency, we propose a progressive optimization procedure:

$\xi^* = \mathbf{0}$;
Set integers $2 = i_1 < \cdots < i_M = T$;
for $I = i_1, \cdots, i_M$ **do**
  Optimize

$$\xi^* \leftarrow \arg\min_{\xi} \sum_{i=1}^{N} \sum_{k=1}^{I} \left(\tilde{u}_{ik} - u_{ik}\right)^2 \Delta t_k \left|\Delta \mathbf{x}_i\right|$$

  with initial guess ξ^*.
end

Algorithm 1: Progressive optimization procedure

Notice $k = 1$ corresponds to the initial condition. Since the twin model uses the same initial condition as the black-box model, $\tilde{u}_{i1} - u_{i1}$ is always zero. Choosing the integer sequence $i_1, \cdots, i_M$ can be problem dependent. Our experience shows the sequence should be denser at small i, and sparser at larger i. The tolerance of each sub-optimization problem does not need to be tight, except for the last iteration where $I = T$. In our problem, we use $i_l = \min\left\{1 + 2^l, T\right\}$, a relative tolerance of 10%, and maximum 10 iterations for the sub-optimization problems.

6 Excited domain and basis selection

The basis for modelling the flux or the source term may be over-complete. In other words, the inference may be ill-posed and not has a unique solution. A twin model with a basis selection scheme should be able to adaptively refine its basis on-the-fly during fitting the basis coefficients ξ. For example, in the Buckley-Leverett equation example, the value of $u(t,x)$, $t \in [0,T]$, $x \in [0,1]$ is bounded, i.e. $0 < u_{\min} \leq u(t,x) \leq u_{\max} < 1$; therefore as long as $\nabla\tilde{F}(u) = \nabla F(u)$ for $u \in [u_{min}, u_{max}]$, we will have $\tilde{u}(t,x) = u(t,x)$ for $t \in [0,T]$ and $x \in [0,1]$. In other words, in the numerical example, $\nabla\tilde{F}(u) = \nabla F(u)$ for $u \in [0,1]$ is a sufficient but not necessary condition for $\tilde{u}(t,x) = u(t,x)$. Intuitively, for some domains u, $\nabla\tilde{F}(u)$ has to approximate $\nabla F(u)$ accurately in order to give a good space-time solution match. We will call such domains *excited domain*, written as $\mathcal{E}$. Notice the excited domain is not a domain of space or time. Clearly $\mathcal{E}$ depends on $u(t,x)$. For u not inside $\mathcal{E}$, $\nabla\tilde{F}(u)$ will have little or no effect on the solution match; therefore we may not certify $\nabla\tilde{F}(u)$'s accuracy outside $\mathcal{E}$ no matter how closely the solutions match. Basis selections should only be navigated to $\mathcal{E}$.

Consider a twin model solving

$$\frac{\partial \tilde{u}(t,x)}{\partial t} + \nabla \cdot \tilde{F}(\mathcal{D}\tilde{u}, \kappa) = q(\tilde{u}, c(t,x)), \quad \tilde{u}(t,x) \in \mathbb{R}^n, \tag{17}$$

We define the excited domain $\mathcal{E}$ by its complement $\bar{\mathcal{E}}$:

Definition 1. *Given a primal model, its discretized solution of $u(t,x)$, and a twin model Eqn(17), the excited domain $\mathcal{E}$ is a domain of $\tilde{F}(\cdot)$. Let the complement of $\mathcal{E}$ be $\bar{\mathcal{E}}$. Consider a perturbed twin model flux $\tilde{F}_\delta(\cdot) = F(\cdot) + \delta(\cdot)$. The perturbed twin model gives the discretized solution $\tilde{u}_\delta$.*

A set $e \subseteq \bar{\mathcal{E}}$ if and only if

$$\frac{1}{T} \sum_{i=1}^{N} \sum_{k=1}^{T} \left(\tilde{u}_{\delta,ik} - u_{ik}\right)^2 \Delta t_k \left|\Delta \mathbf{x}_i\right|$$

is a constant for any δ with support$[\delta] = e$.

This definition requires F a prior, therefore it is not directly implementable. The definition also requires enumeration of all possible δ to validate $\mathcal{E}$. In practice, we can only validate a finite set of δ, for example the basis function library g.

Basis selection may be performed by regularization [13, 14]. For example, it has been shown that basis selection can be performed by having a Lasso regularization term in the solution mismatch. In our problem, the metric of solution mismatch with Lasso regularization would be

$$\frac{1}{T}\sum_{i=1}^{N}\sum_{k=1}^{T}\left(\tilde{u}_{\epsilon\delta,ik}-u_{ik}\right)^2\Delta t_k\left|\Delta\mathbf{x}_i\right|+\lambda\|\xi\|_1\,, \tag{18}$$

where $\|\cdot\|_1$ is the L_1 norm.

7 Numerical example

In this section we demonstrate a numerical example for fitting a twin model with fixed structure. Consider a 1D Buckley-Leverett equation

$$\frac{\partial u}{\partial t}+\frac{\partial F(u)}{\partial x}=0\quad x\in[0,1]\,,\ t\in[0,1] \tag{19}$$

with periodic boundary condition

$$u(x=0)=u(x=1) \tag{20}$$

and initial condition

$$u(t=0)=u_0 \tag{21}$$

Buckley-Leverett equation is a simple model for 1D, two-phase, porous media flow driven by capillary pressure and Darcy's law [7]. u indicates one phase's saturation and $0\le u\le 1$. The flux function $F(u)$ depends on the phases and the porous media. A popular $F(u)$ is

$$F(u)=\frac{u^2}{1+A(1-u)^2}\,, \tag{22}$$

where A is a constant. In the following we assume the blackbox simulation solves Eqn(19) with the flux given by Eqn(22), and $A=2$.

Assume $F(u)$ is unknown, we will fit a twin model using the blackbox simulation. The twin model can be written as

$$\frac{\partial\tilde{u}}{\partial t}+\frac{\partial}{\partial x}\left(\sum_{k=1}^{m}\xi_k g_k(\tilde{u})\right)=0 \tag{23}$$

with the same initial and boundary conditions. We will use sigmoid functions to represent the flux. Both the black-box simulation and the twin model use a second order finite volume discretization and Crank-Nicolson time integration scheme.

The objective of fitting the twin model is given in Eqn(13), and is itself an optimization. We use an automatic differentiation module *numpad* [9] to compute $\frac{dJ}{d\xi_k}\,, k=1,\cdots,m$. Since the gradient information is available, we consider using a quasi-Newton method for the optimization. Quasi-Newton methods build the Hessian approximation iteratively using gradient, and can greatly accelerate convergence [11]. When the degree of freedom of the optimization is high, the memory required to store the Hessian matrix can be large. To reduce the memory requirement, we can use the *low-memory Broyden-Fletcher-Goldfarb-Shannon* (L-BFGS) algorithm [10, 12]. L-BFGS approximates the Hessian using only the gradients at newer previous iterations, and inverse the approximated Hessian efficiently using the Sherman-Morrison formula. For this work, we use the implementation of L-BFGS in Nlopt [10].

The initial conditions are shown in Fig 2.

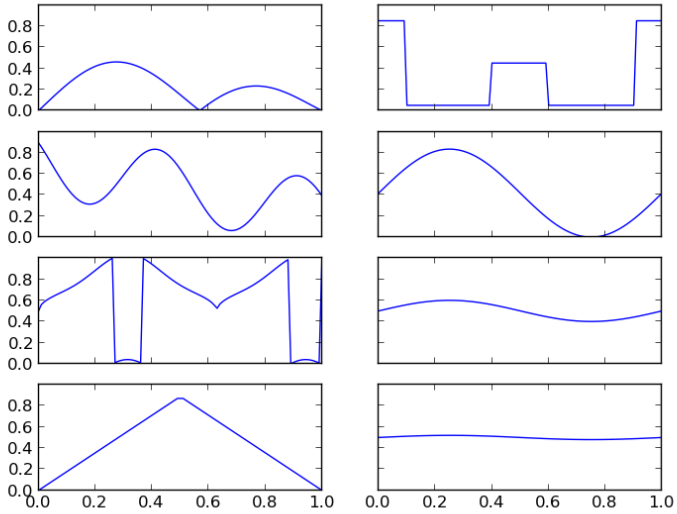
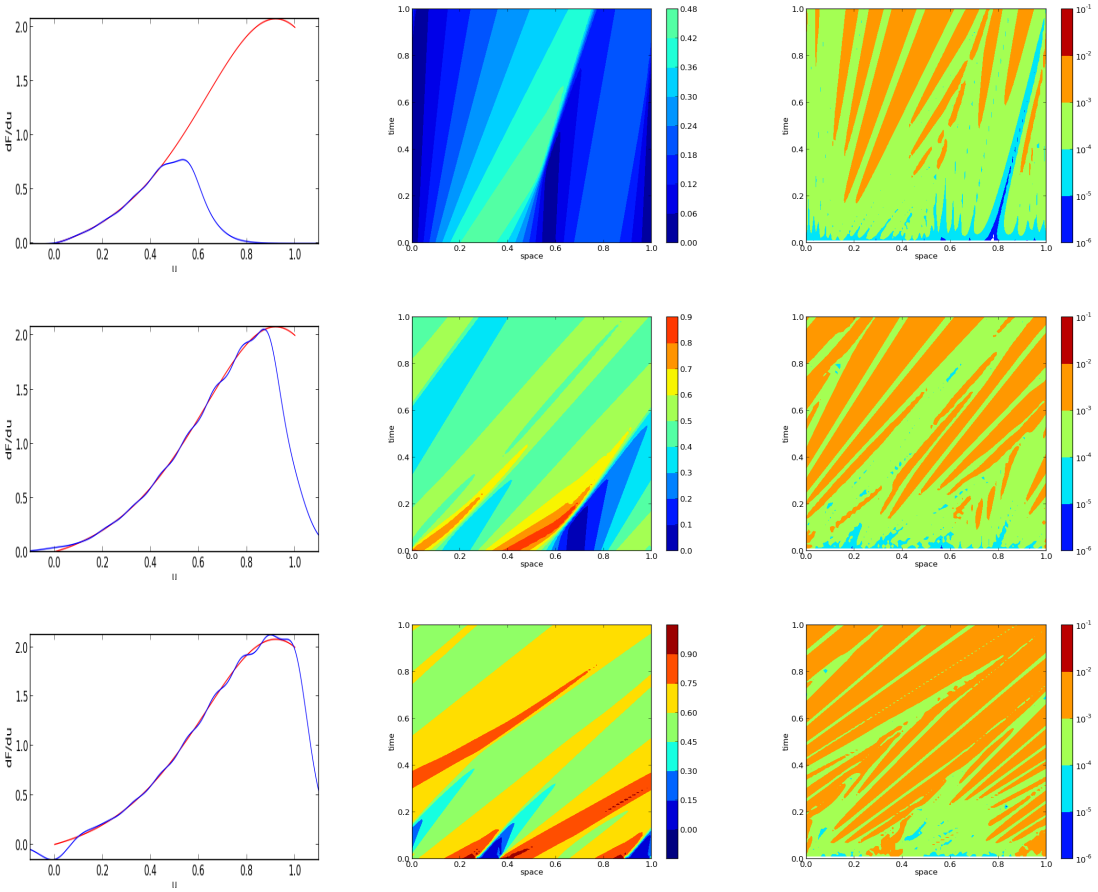


Figure 2: Initial condition $u_0(x)$

We compare $\frac{dF}{du}$ with the trained $\frac{d\tilde{F}}{du}$. We also show the space-time solution of the gray-box, as well as the solution mismatch, for the 8 initial conditions.



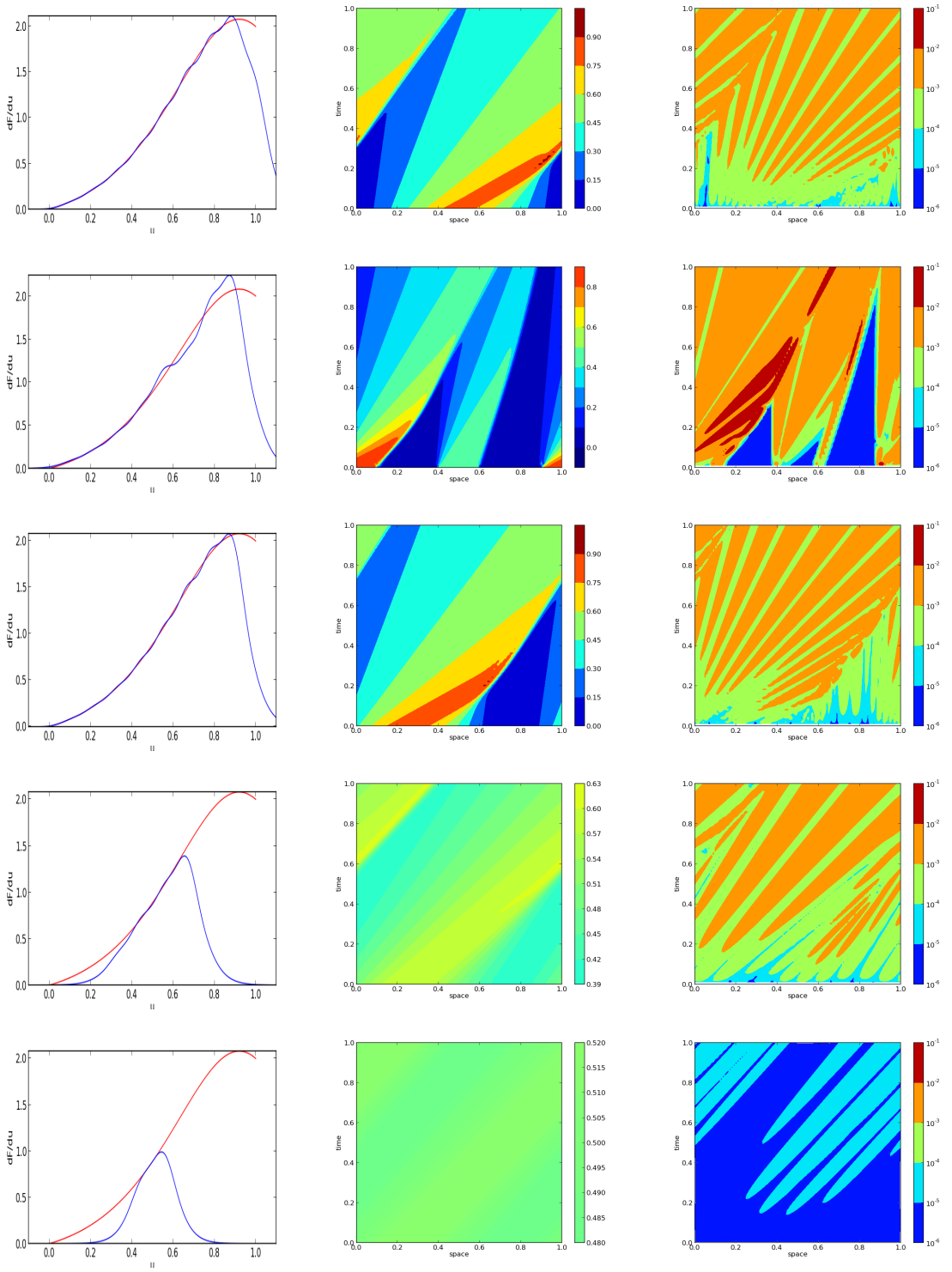


Figure 3: Left: ompare F' (red line) with $\tilde{F}'$ (blue line). Middle: the space-time solutions of the gray-box model. Right: $|u - \tilde{u}|$.

Using the twin model, we can estimate the objective's gradient by applying adjoint method to the twin model, i.e. we approximate $\frac{\partial J}{\partial u}$ with $\frac{\partial \tilde{J}}{\partial u}$. Suppose the gray-box model solves

$$\frac{\partial u}{\partial t} + \frac{\partial}{\partial x}\left(F(u)\right) = c\,, \tag{24}$$

for $c = 0$, with $F(u)$ given by Eqn(22). We trained a twin model Eqn(23) using the space time solution. We are interest in the approximation quality of the twin model's gradient at c adjacent to $c = 0$. The results are shown below

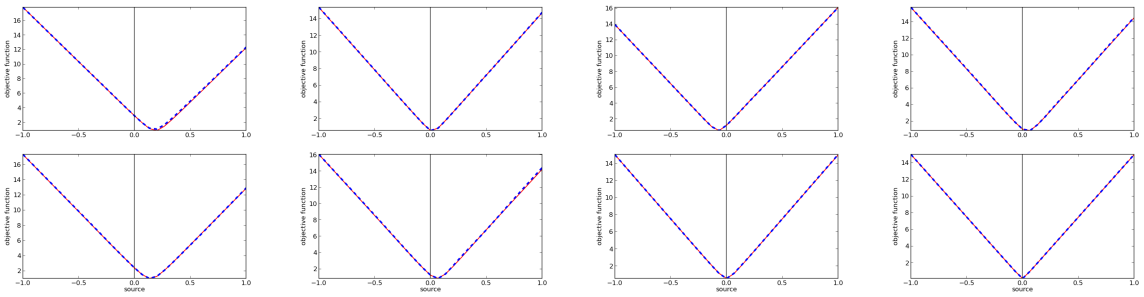
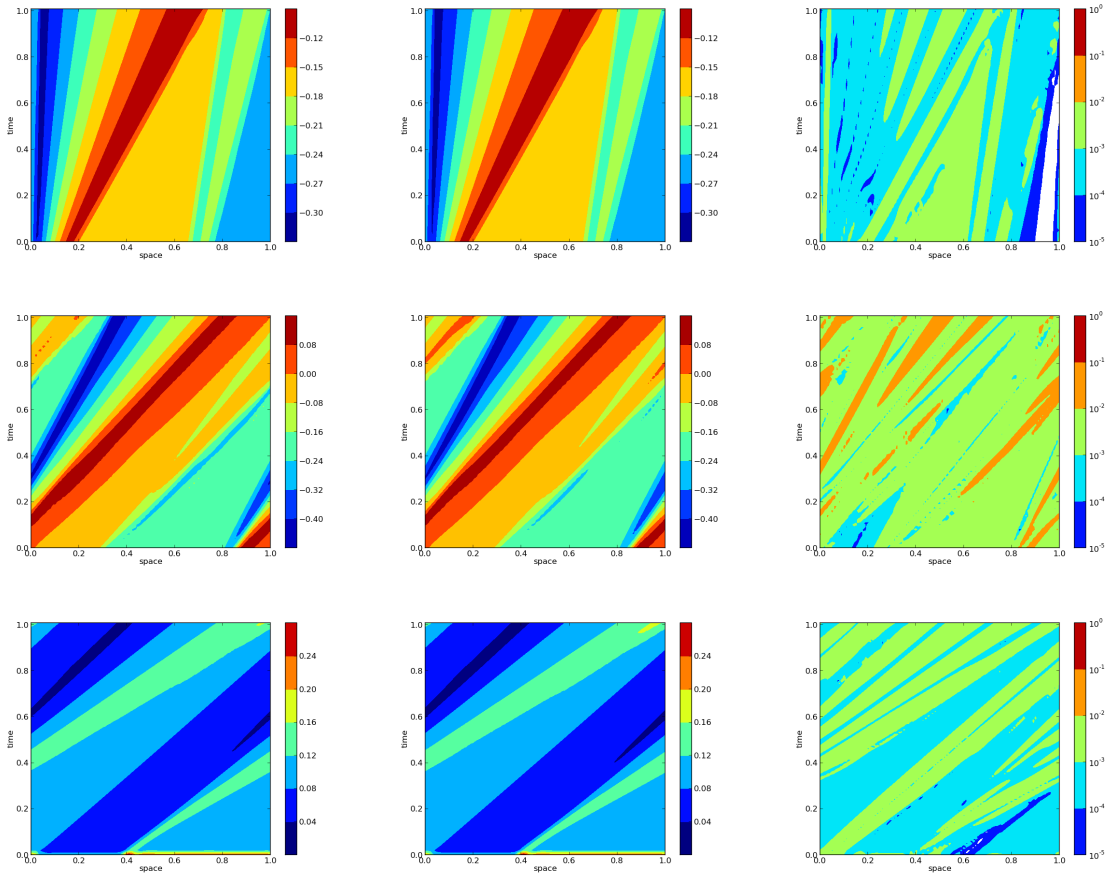


Figure 4: Compare $J(c)$ (red line) with $\tilde{J}(c)$ (blue dashed line) for the 8 cases. The black vertical line indicates $c = 0$ where the twin models are trained.

If the c is space-time dependent, $\frac{dJ}{dc}$ will be a space-time dependent field. We compare $\frac{dJ}{dc}$ with $\frac{d\tilde{J}}{dc}$.



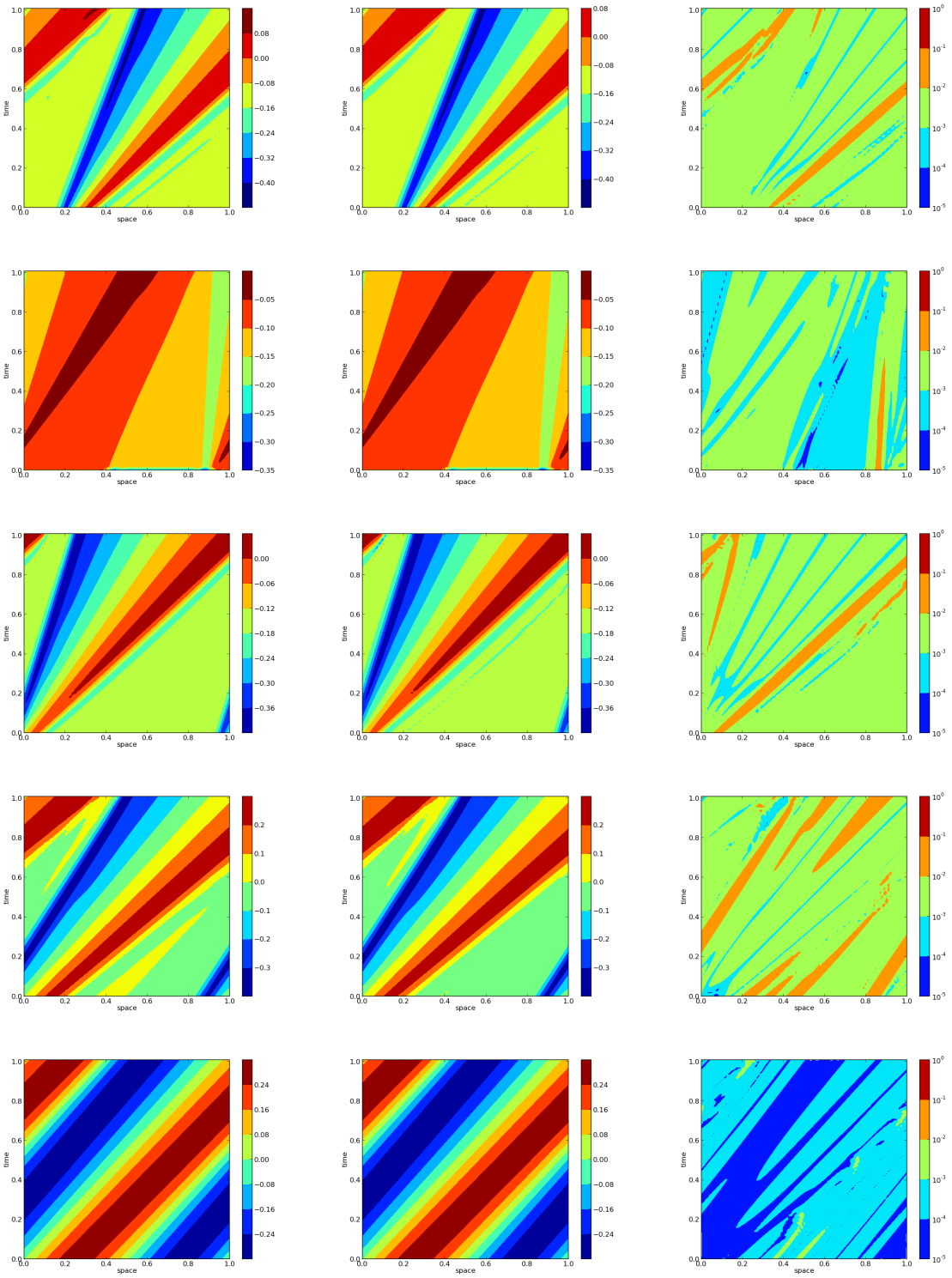


Figure 5: Left: $\frac{dJ}{dc}$*, evaluated by the gray-box model. Middle:* $\frac{d\tilde{J}}{dc}$*, evaluated by the trained twin model. Right:* $\left|\frac{dJ}{dc} - \frac{d\tilde{J}}{dc}\right|$.

The result is encouraging as the gradient computed by the twin model gives a good approximation of the gradient of the primal model. We reiterate that the good approximation quality benefits from the matching of space-time solution.

8 Conclusion

We propose a method to estimate the objective's gradient when the simulation does not implement adjoint. The proposed method enables adjoint computation for gray-box simulations, whose adjoint sensitivity were conventionally considered unavailable. The method infers the governing PDE by taking advantage of the space-time solution of the gray-box simulation, and applies adjoint to estimate the gradient. We demonstrate the proposed method on a 1D conservation law simulation.

9 References

[1] Han Chen. "Blackbox stencil interpolation method for model reduction" Master thesis, 2012

[2] Rios, Luis Miguel, and Nikolaos V. Sahinidis. "Derivative-free optimization: A review of algorithms and comparison of software implementations." Journal of Global Optimization 56.3 (2013): 1247-1293.

[3] Dennis, Jr, John E., and Jorge J. Mor. "Quasi-Newton methods, motivation and theory." SIAM review 19.1 (1977): 46-89.

[4] Nadarajah, Siva, and Antony Jameson. "A comparison of the continuous and discrete adjoint approach to automatic aerodynamic optimization." AIAA paper 667 (2000): 2000.

[5] A Griewank, GF Corliss Automatic differentiation of algorithms: theory, implementation, and application. Defense Technical Information Center, 1992.

[6] Zhangxin Chen "Reservoir Simulation: Mathematical Techniques in Oil Recovery" Society for Industrial and Applied Mathematics, ISBN 0898716403, 2007

[7] S.E. Buckley and M.C. Leverett "Mechanism of fluid displacement in sands." Transactions of the AIME 146 (1942): 107-116

[8] Chen, Zhangxin, Guanren Huan, and Yuanle Ma. "Computational methods for multiphase flows in porous media." Vol. 2. Siam, 2006.

[9] Qiqi Wang, Numpad package, https://github.com/qiqi/numpad.git

[10] Steven G. Johnson, The NLopt nonlinear-optimization package, http://ab-initio.mit.edu/nlopt

[11] John E. Dennis, Jorge J. Mor. "Quasi-Newton methods, motivation and theory." SIAM review 19.1 (1977): 46-89.

[12] J. Nocedal. "Updating quasi-Newton matrices with limited storage" Mathematics of Computation, 35 (1980): 773-782

[13] Tibshirani, Robert. "Regression shrinkage and selection via the lasso." Journal of the Royal Statistical Society. Series B (Methodological) (1996): 267-288.

[14] Dziak, John, Runze Li, and Linda Collins. "Critical review and comparison of variable selection procedures for linear regression (Technical report)." (2005).

[15] Verstraete, Tom, et al. "Optimization of a U-Bend for Minimal Pressure Loss in Internal Cooling ChannelsPart I: Numerical Method." Journal of Turbomachinery 135.5 (2013): 051015.

[16] Coletti, Filippo, et al. "Optimization of a U-Bend for Minimal Pressure Loss in Internal Cooling ChannelsPart II: Experimental Validation." Journal of Turbomachinery 135.5 (2013): 051016.