SIAM J. SCI. COMPUT.
Vol. 43, No. 4, pp. A2981–A3008

RECOVERY OF A TIME-DEPENDENT BOTTOM TOPOGRAPHY FUNCTION FROM THE SHALLOW WATER EQUATIONS VIA AN ADJOINT APPROACH[*]

JOLENE BRITTON[†], YAT TIN CHOW[†], WEITAO CHEN[†], AND YULONG XING[‡]

Abstract. We develop an adjoint approach for recovering the topographical function included in the source term of one-dimensional hyperbolic balance laws. We focus on a specific system, namely, the shallow water equations, in an effort to recover the riverbed topography. The novelty of this work is the ability to robustly recover the bottom topography using only noisy boundary data from one measurement event and the inclusion of two regularization terms in the iterative update scheme. The adjoint scheme is determined from a linearization of the forward system and is used to compute the gradient of a cost function. The bottom topography function is recovered through an iterative process given by a three-operator splitting method which allows the feasibility of including two regularization terms. Numerous numerical tests demonstrate the robustness of the method regardless of the choice of initial guess and in the presence of discontinuities in the solution of the forward problem.

Key words. inverse problems, adjoint methods, discontinuous Galerkin methods, hyberbolic balance laws, shallow water equations, L^1 and H^1 regularization

AMS subject classifications. 35L60, 35L65, 35Q35, 65M30, 65M32, 65M60, 65K10

DOI. 10.1137/20M1341787

1. Introduction.

Various phenomena arising frequently in natural, engineering, and socio-economical applications can be modeled by hyperbolic conservation and balance laws. Examples of conservation laws include models for traffic flow [15], fluid dynamics [12], and supply chains [3]. Moreover, conservation laws with source terms, also known as balance laws, are used in different models, e.g., the gas pipeline flow [24], shallow water flow [63, 42], gas dynamics under a gravitational field [64], and blood flow through arteries [66]. A variety of theoretical studies have been conducted to understand the underlying mathematical structure and a wide range of numerical approaches dedicated to solving for the state variables have been developed in the past few decades. On the other hand, optimization, control, and recovery of the system parameters is a problem of great interest due to its high practical value. In this work, we derive an inverse problem algorithm for a specific system of hyperbolic balance laws, in which a time-dependent topographical profile is recovered based on the measurement from the boundary.

Various optimization and control methods have been developed for conservation and balance laws such as backstepping [61, 20], Lyapunov-based [61], derivative-free [43], and optimal control methods [35]. In this work we employ an adjoint approach, which is often praised for its efficiency. Its computational cost (of each iteration) is comparable to that of solving a partial differential equation (PDE) once, instead

[*]Submitted to the journal's Methods and Algorithms for Scientific Computing section May 29, 2020; accepted for publication (in revised form) April 29, 2021; published electronically August 24, 2021.

https://doi.org/10.1137/20M1341787

Funding: The work of the third author was partially supported by the NSF grant DMS-1853701 and DMS-1762063. The work of the fourth author was partially supported by the NSF grant DMS-1753581.

[†]Department of Mathematics, University of California Riverside, Riverside, CA 92521 USA (jhout001@ucr.edu, yattinc@ucr.edu, weitaoc@ucr.edu).

[‡]Department of Mathematics, The Ohio State University, Columbus, OH 43210 USA (xing.205@osu.edu).

of depending on the number of control variables or design parameters as in other optimization methods.

Adjoint methods were introduced for optimal control problems in 1971 by Lions [46] in the context of shape optimization of aerodynamic bodies. A few years later the methods were extended to fluid dynamics by Pironneau [53]. Jameson popularized the techniques for potential flow and the Euler equations [37, 38]. The methods have also been applied to biological systems in the search of parameter identification [55]. Estimation of open water states [5] and traffic states on the freeways [34] have also been accomplished via the adjoint optimization method.

The adjoint method for computing the gradient of a cost function can be accomplished in a variety of manners. The discrete adjoint approach [25, 41, 19] begins with discretizing the forward system of PDEs while the continuous adjoint approach [25, 36, 47, 57] begins with the continuous forward system. Both cases result in discrete adjoint equations. The continuous approach allows one to obtain an explicit gradient formulation [56] while automatic differentiation techniques can be applied to ease discrete approach solvers [50, 17]. With regards to scalar conservation laws, Holden, Priuli, and Risebro [31] developed a reconstruction procedure for the coefficient inverse problem in which a spatially dependent coefficient of the flux term is recovered. Bürger, Coronel, and Sepúlveda [4] solved the inverse scalar conservation law modeling sedimentation numerically by assuming a variational form of the problem. The flux function of a scalar conservation law was reconstructed using the information from the shock that forms in the work by Kang and Tanuma [39]. In a more general setting for balance laws, Montecinos et al. [49] derived a unified scheme for solving the forward and adjoint problems simultaneously. Methodology for the scalar Burgers' equation was presented by Lellouche, Devenon, and Dekeyser [44] in which the authors aimed to find the best approximation for the measured data by means of boundary control and an adjoint approach. Ferlauto [13] obtained optimal geometric shapes for aerodynamic bodies by solving an inverse problem for the three-dimensional incompressible Euler equations.

Numerical computation of optimization problems for conservation laws have been studied extensively due to the theoretical and numerical challenges that arise. As the exact solution of conservation laws often contains discontinuities, one challenge in the related optimization problem is that nonnegligible numerical errors may occur in capturing the discontinuities. Some of these difficulties are mitigated via introduction of Lax–Friedrichs schemes [18] or relaxation methods [1], for instance. Convergence analyses have been provided for optimization problems in the aforementioned works. The fact that many conservation laws are nonlinear presents another challenge because this can lead to nonconvex formulations of the optimization problem. One method to tackle this difficulty is to use linear programming methods once the discretization scheme is "relaxed" [21, 67]. This allows for a global optimum to be found and reduces computational cost, but the linearization may not be a good physical representation of the original system [56]. To maintain the nonlinearity of the system, a more expensive method, such as gradient descent, can be used, but may not ensure a global optimum is achieved.

In this paper, we focus on an inverse problem that arises in first order nonlinear hyperbolic balance laws. Many difficulties arise in this research field as a result of uncertainties in data, measurements, and the use of complex data. It is very important to develop highly accurate, easy to implement, and cost-efficient methods with high resolution to study fluvial environments numerically. Although the method we employ can be quite general, for the sake of simplicity and better illustrative purposes, we focus on a prototype example of hyperbolic balance laws in this work, namely, the nonlinear shallow water equations (SWEs), also referred to as the Saint-Vernant equations. This system models fluvial environments such as flood plane dynamics, coastal and tidal flows, and flow and sediment transport. It has wide applications in ocean, environmental, and hydraulic engineering, as well as atmospheric modeling. The model parameter we aim to recover in this context is the riverbed topography, however, other terms such as friction may also be of interest to reconstruct.

Ground surveys of riverbeds or direct topographical data collection are not always effective for determining underwater topography because the operations are costly and time consuming. Numerical techniques of constructing riverbed topography can offer many benefits over aerial and ground techniques in terms of cost reduction, efficiency, as well as flexibility. Heining and Aksel [27] used a direct approach to reconstruct the bottom topography of steady-state thin-film flow. Castaings et al. [5] presented an automatic differentiation technique and free surface information to reconstruct river bed topology. Honnorat et al. [32] derived a method for recovering channel topology from a steady-state solution of the forward problem using an optimization technique called variational data assimilation. A direct approach from the one-dimensional SWEs was used by Gessese et al. [16] to reconstruct the river bed from free surface data. Last, a stable finite volume scheme in the presence of wetting-drying fronts and inverse computational algorithms (based on a variational approach) were presented in [48]. All methods developed in these studies dealt with time-independent bottom topography functions.

The main objective of this paper is to develop a robust algorithm that requires less data to reconstruct a dynamic bottom function, thereby allowing the construction of a more accurate and inexpensive model. We assume the measurements, possibly with noise, are taken only at the two boundaries of the spatial domain in a given time period. Traditionally, the bottom topography in the SWEs is a function of space alone within the framework of the inverse problem construction. Here, we consider the time-dependent bottom topography function, which allows for the recovery of the bottom topography with less data considering the fact that movement in the forward problem solution coming from the change of the bottom topography allows more information propagation to the boundary measurements. Usually inverse problems are more difficult when we need to recover both the temporal and spatial profile. In this work, as a first step, we assume a special form of the topographical profile which represents two known spatial profiles and an unknown temporal interaction. This time-dependent bottom topography can practically describe a physical phenomenon when two platonic plates with known topography are moving against one another, e.g., an earthquake, underwater volcanic buildup, or a moving sand bottom. We aim to recover only the temporal profile representing, e.g., the pulse of the earthquake. After constructing the adjoint formulation, we present the cost function with two regularization terms added to suppress noise and handle the ill-posedness of this problem. An iterative update scheme based on a three-operator splitting scheme is employed to update the targeting function. This splitting scheme requires each operator to be computed only once per iteration and is straightforward to implement.

This paper is organized as follows. In section 2 we introduce the primal equations used throughout the paper. The discontinuous Galerkin (DG) numerical scheme is also presented as the method used for solving the forward problem. Section 3 includes a discussion on the formulation of the inverse problem. The adjoint equations are derived from a linearized system and used to derive the gradient formulation of the cost function. This section also contains the description of the iterative updating procedure for determining the desired source term, as well as a discussion on the choice of regularization terms. Numerical examples are presented in section 4, and conclusions are discussed in section 5.

2. Forward problem.

In this section, we present the hyperbolic PDE system used to define the forward (or primal) problem. The forward system will be used in section 3 to determine the adjoint formulation, which in turn is employed to derive the gradient of a cost function and iteratively update the time-dependent bottom topography function. We will discuss the PDE system as well as the numerical scheme used to solve the forward problem.

2.1. Forward problem formulation.

Hyperbolic balance laws are conservation laws with a source term. A few examples of such systems include the nonlinear SWEs, the arterial blood flow model, the Euler equations under gravity, and the telegrapher's equations. A source term might arise as a result of many factors, such as a friction term or a topographical term.

In this work, we only consider the one-dimensional systems of m hyperbolic balance laws which take the form

$$\begin{cases} \partial_t \boldsymbol{U} + \partial_x \boldsymbol{F}(\boldsymbol{U}) = \hat{\boldsymbol{S}}(\boldsymbol{U}, \boldsymbol{B}), & (x,t) \in (x_0, x_L) \times (0, T], \\ \boldsymbol{U}(x,0) = \boldsymbol{U}_0(x), & x \in (x_0, x_L), \end{cases} \tag{2.1}$$

where $\boldsymbol{U}$ are the state variables, $\boldsymbol{F}(\boldsymbol{U})$ are the fluxes, and $\boldsymbol{U}_0(x)$ are the initial conditions. The vector $\boldsymbol{B}$ represents the model parameters we aim to recover in the inverse problem which are only present in the source term, $\hat{\boldsymbol{S}}(\boldsymbol{U}, \boldsymbol{B})$. The source term can be rewritten in the form $\hat{\boldsymbol{S}}(\boldsymbol{U}, \boldsymbol{B}) = \boldsymbol{S}(\boldsymbol{U}, \boldsymbol{B})\boldsymbol{U}$ in which $\boldsymbol{S}(\boldsymbol{U}, \boldsymbol{B}) \in \mathbb{R}^m \times \mathbb{R}^m$ is a matrix.

Alternatively, the system can be written in quasi-linear form using the Jacobian matrix $\boldsymbol{A}(\boldsymbol{U}) = \frac{\partial \boldsymbol{F}}{\partial \boldsymbol{U}}$,

$$\begin{cases} \partial_t \boldsymbol{U} + \boldsymbol{A}(\boldsymbol{U}) \partial_x \boldsymbol{U} = \boldsymbol{S}(\boldsymbol{U}, \boldsymbol{B})\boldsymbol{U}, & (x,t) \in (x_0, x_L) \times (0, T], \\ \boldsymbol{U}(x,0) = \boldsymbol{U}_0(x), & x \in (x_0, x_L). \end{cases} \tag{2.2}$$

To determine the solution of a forward problem, one seeks to determine the state variables $\boldsymbol{U}$ with the model parameters $\boldsymbol{B}$ given. In this work, we will only consider the case when we have a single topographical function, denoted by B. While the information in B is traditionally a function of space alone, here we consider B as a function depending on both space and time as follows:

$$B(x,t) = B_0(x) + p(t) B_1(x), \tag{2.3}$$

where we assume $B_0(x)$ and $B_1(x)$ to be known and $p(t)$ to be the component we wish to recover.

In this paper we will focus on the SWEs with a nonflat bottom topography, one of the most well-known systems of hyperbolic balance laws. In particular, we aim to recover the riverbed topography function, denoted by b. This term occurs only in the source term of the momentum equation in the form of its derivative, $\partial_x b$, so we define

$$B = \partial_x b(x,t) = \partial_x b_0(x) + p(t) \partial_x b_1(x). \tag{2.4}$$

The state variables, flux terms, and source term for the SWEs are given by:

$$\boldsymbol{U} = \begin{bmatrix} h \\ hu \end{bmatrix}, \quad \boldsymbol{F}(\boldsymbol{U}) = \begin{bmatrix} hu \\ hu^2 + \frac{1}{2} g h^2 \end{bmatrix}, \quad \hat{\boldsymbol{S}}(\boldsymbol{U}, B) = \begin{bmatrix} 0 \\ -ghB \end{bmatrix} = \begin{bmatrix} 0 \\ -gh\partial_x b \end{bmatrix}, \tag{2.5}$$

by following the form (2.1). Here, $h(x,t) \geq 0$ is the water height, $(hu)(x,t)$ is the water discharge with $u(x,t)$ being the depth averaged velocity, and $g = 9.812$ is the gravitational constant. On the other hand, under the quasi-linear formulation, we write:

$$\boldsymbol{A}(\boldsymbol{U}) = \begin{bmatrix} 0 & 1 \\ gh - u^2 & 2u \end{bmatrix}, \qquad \boldsymbol{S}(\boldsymbol{U}, B) = \begin{bmatrix} 0 & 0 \\ -g\partial_x b & 0 \end{bmatrix}. \tag{2.6}$$

Time-dependent bottom topography functions have been considered in the literature. In a more complicated model, e.g., [59, 29, 33, 45, 42], the change of the bottom function in the SWEs may depend on other state variables. For instance, the bottom function may be determined by erosion, sediment transport, dam breaks, or landslides due to floods. In such cases, additional equations to model the evolution of bottom topography may arise in the system in order to better describe this dependence. However, these models are different from the one we consider in this work.

2.2. DG method for the forward problem. The DG method will be used to solve the forward problem (2.6). It is a high order accurate scheme that has gained significant attention in the last decade. The method is advantageous for hyperbolic conservation laws because it is both stable, similar to the finite volume method, and flexible, like the finite element method. The arbitrary order feature of the DG method can provide accurate results on a coarse mesh. In particular, in an inverse problem algorithm, a forward solver is usually employed during each iteration, hence the use of a coarser mesh is ideal in an effort to reduce computation cost in the iterative process. Last, the DG scheme is able to capture the discontinuous solutions well and help us locate the interfaces accurately.

The computational domain, $I = [x_0, x_L]$, is first discretized into N cells. The jth cell is denoted by $I_j = [x_{j-\frac{1}{2}}, x_{j+\frac{1}{2}}]$ with x_j being the center of the cell. The size of the jth cell is denoted by $\Delta x_j = x_{j+\frac{1}{2}} - x_{j-\frac{1}{2}}$ and we let $h = \max_j \Delta x_j$. We seek an approximation $\boldsymbol{U}_h$ of the solution $\boldsymbol{U}$ in which $\boldsymbol{U}_h^{(i)}$ for $i = 1, \dots, m$ belongs to the finite-dimensional piecewise polynomial space

$$\mathbb{V}_h^k = \{v : v|_{I_j} \in P^k(I_j), j = 1, \dots, J\}, \tag{2.7}$$

where $P^k(I_j)$ denotes the space of polynomials of degree up to k on I_j. The topographical source term variable, B, is also projected into $\mathbb{V}_h^k$, and is written as B_h. The scheme does not require $\boldsymbol{U}_h$ to be continuous at the cell interface $x_{j+\frac{1}{2}}$, so we introduce the notation $\boldsymbol{U}^+_{h,j+\frac{1}{2}}$ as the limit of the solution $\boldsymbol{U}_h$ at $x_{j+\frac{1}{2}}$ from the right cell I_{j+1}, and $\boldsymbol{U}^-_{h,j+\frac{1}{2}}$ as the limit from the left cell I_j.

The DG scheme in each cell I_j is based on a modification of the weak formulation of the PDE,

$$\int_{I_j} \partial_t \boldsymbol{U}_h \boldsymbol{v}\, dx - \int_{I_j} \boldsymbol{F}(\boldsymbol{U}_h)\partial_x \boldsymbol{v}\, dx + \hat{\boldsymbol{F}}_{j+\frac{1}{2}} \boldsymbol{v}^-_{j+\frac{1}{2}} - \hat{\boldsymbol{F}}_{j-\frac{1}{2}} \boldsymbol{v}^+_{j-\frac{1}{2}} = \int_{I_j} \hat{\boldsymbol{S}}(\boldsymbol{U}_h, B_h)\boldsymbol{v}\, dx, \tag{2.8}$$

where $\boldsymbol{v}(x)$ is a vector of test functions from the test space $\mathbb{V}_h^k$, and the numerical flux $\hat{\boldsymbol{F}}_{j+\frac{1}{2}} = f(\boldsymbol{U}^-_{h,j+\frac{1}{2}}, \boldsymbol{U}^+_{h,j+\frac{1}{2}})$ is a function that takes information from both the left and right side of the cell interface. We implement the simple Lax–Friedrichs flux

$$f(a,b) = \frac{1}{2}\Big(\boldsymbol{F}(a) + \boldsymbol{F}(b) - \alpha(b-a)\Big), \tag{2.9}$$

where α is chosen to be the maximum of the largest eigenvalue of $\boldsymbol{A}(\boldsymbol{U})$ over the entire computational domain or locally in each cell. In the case of the SWEs, let $\alpha = \max(|u| + \sqrt{gh})$.

The semidiscrete method (2.8) can be rewritten in the ODE form as

$$\partial_t \boldsymbol{U}_h = \mathcal{F}(\boldsymbol{U}_h)$$

after representing $\boldsymbol{U}_h$ as a linear combination of the basis functions of $\mathbb{V}_h^k$. In order to advance the scheme in time, we use the high order strong-stability preserving Runge–Kutta (SSP-RK) temporal discretization [23]. Throughout this paper, the third order SSP-RK method of the form

$$\begin{aligned}\boldsymbol{U}_h^{(1)} &= \boldsymbol{U}_h^n + \Delta t \mathcal{F}(\boldsymbol{U}_h^n),\\ \boldsymbol{U}_h^{(2)} &= \frac{3}{4}\boldsymbol{U}_h^n + \frac{1}{4}\Big(\boldsymbol{U}_h^{(1)} + \Delta t \mathcal{F}\big(\boldsymbol{U}_h^{(1)}\big)\Big),\\ \boldsymbol{U}_h^{n+1} &= \frac{1}{3}\boldsymbol{U}_h^n + \frac{2}{3}\Big(\boldsymbol{U}_h^{(2)} + \Delta t \mathcal{F}\big(\boldsymbol{U}_h^{(2)}\big)\Big),\end{aligned} \tag{2.10}$$

is used. Furthermore, the equation for the space- and time-dependent bottom topography function (2.3) must also involve a temporal discretization. As $\mathcal{F}(\boldsymbol{U}_h^n)$ is evaluated at time t^n, $\mathcal{F}\big(\boldsymbol{U}_h^{(1)}\big)$ is evaluated at time $t^n + \Delta t$, and $\mathcal{F}\big(\boldsymbol{U}_h^{(2)}\big)$ is evaluated at time $t^n + \frac{1}{2}\Delta t$, we evaluate $B(x,t)$ at the same time values for each Runge–Kutta step. When the function values at $t^n + \frac{1}{2}\Delta t$ are unavailable, they are determined via quadratic interpolation.

3. Inverse problem.

In this section, we describe the inverse problem of our focus, which is the reconstruction of the topographical source function B from boundary data of the hyperbolic conservation law (2.1) from a single measurement event. We reduce the inverse problem to an optimization problem of a residual functional coming from boundary measurements, with an addition of two regularization terms, which will be described more concretely later in this section.

In our work, we adopt the adjoint method to numerically obtain a gradient of our functional. We will describe the cost function we wish to minimize, the derivation of the adjoint formulation for the gradient calculation, and the iterative update scheme for determining the time component, $p(t)$, of the source function B. The numerical solution to the adjoint problem will be calculated using the DG method.

We assume that noisy measurements of $\boldsymbol{U}$ are taken only at both boundaries of the spatial domain, given a period of time $[0, T]$ in one single measurement event. During the numerical reconstruction process, we assume that only these noisy solutions at the boundary of the spatial domain are known to us. For notational sake, we denote these noisy measurements as $\hat{\Lambda}^{\text{noisy}} = \hat{\Lambda}\mu$, where the multiplicative noise is uniformly distributed, $\mu \sim \mathcal{U}\left[1 - \frac{1}{2}\eta_{meas}, 1 + \frac{1}{2}\eta_{meas}\right]$, with a given noise level η_{meas}. The goal of the inverse scheme is to find the function, $p(t)$, that provides the best approximation $\Lambda(B(p)) \approx \hat{\Lambda}^{\text{noisy}}$. The map $\Lambda(B(p)) = \boldsymbol{U}|_{\{x_0, x_L\}\times[0,T]}$ represents the forward map with the input, $B(p)$, as the topographical function and the output as the solutions, $\boldsymbol{U}$, at the boundary points, $\{x_0, x_L\}$, over the time interval $[0, T]$. We thus declare the control to be $p(t)$ and the number of control variables to be equivalent to the number of time steps in the numerical scheme. On the other hand, the observation values are $\Lambda(B(p)) = \boldsymbol{U}|_{\{x_0, x_L\}\times[0,T]}$. The number of observation values is determined by taking the product of the number of state variables (m), the number of boundary points (two in 1 dimension), and the number of time steps in the numerical scheme.

Finding the best approximation reduces to minimizing the error or residue of the predicted and measured data for all time at the boundary of the computational domain. This corresponds to minimizing the functional

$$\boldsymbol{J}(p) := \boldsymbol{J}_0(p) + \boldsymbol{R}(p) := \int_0^T \frac{1}{2} \left|\left[\mathcal{E}(p)\right](x_0,t)\right|^2 + \frac{1}{2}\left|\left[\mathcal{E}(p)\right](x_L,t)\right|^2 dt + \boldsymbol{R}(p), \tag{3.1}$$

where the error function, $\mathcal{E}$, for a given p is defined as

$$\left[\mathcal{E}(p)\right](x,t) = \Lambda(B(p))(x,t) - \hat{\Lambda}^{\text{noisy}}(x,t). \tag{3.2}$$

The term $\boldsymbol{R}(p)$ is a regularization term that will be discussed in section 3.3.2.

The optimization problem becomes

$$\text{minimize } \boldsymbol{J}(p) := \boldsymbol{J}_0(p) + \boldsymbol{R}(p) \quad \text{subject to (2.2)}. \tag{3.3}$$

We employ a descent method to minimize the above functional, in which the (formal) gradient $\nabla \boldsymbol{J}$ will be obtained via the adjoint method following a linearization process of (2.2).

3.1. Gradient derivation. In an effort to determine $\nabla \boldsymbol{J}_0$, we begin by calculating the variational derivative (in the sense of the Gateaux differential) of $\boldsymbol{J}_0$ and dualize it using L^2-pivoting. In what follows, we would like to denote, for a functional $\mathcal{F}$, the variational derivative of $\mathcal{F}$ at p along $\tilde{p}$ as

$$\delta\mathcal{F}(p;\tilde{p}) := \lim_{\epsilon\to 0} \frac{\mathcal{F}(p+\epsilon\tilde{p}) - \mathcal{F}(p)}{\epsilon} \tag{3.4}$$

whenever it exists. Furthermore, whenever $\delta\mathcal{F}(p;\tilde{p})$ is linear with respect to $\tilde{p}$, we (formally) dualize the variational derivative $\delta\mathcal{F}(p;\tilde{p})$ using L^2-pivoting and define the gradient, $\nabla\mathcal{F}(p)$, such that it satisfies the relation

$$\delta\mathcal{F}(p;\tilde{p}) := \int_0^T \left[\nabla\mathcal{F}(p)\right](t)\,\tilde{p}(t)\; dt. \tag{3.5}$$

With these notions at hand, we readily compute that

$$\begin{aligned} \boldsymbol{\delta J}_0(p;\tilde{p}) &= \lim_{\epsilon\to 0} \frac{\boldsymbol{J}_0(p+\epsilon\tilde{p}) - \boldsymbol{J}_0(p)}{\epsilon} \\ &= \int_0^T \Big(\left[\delta\mathcal{E}^T(p;\tilde{p})\,\mathcal{E}(p)\right](x_0,t) + \left[\delta\mathcal{E}^T(p;\tilde{p})\,\mathcal{E}(p)\,\right](x_L,t)\Big)\; dt\,, \end{aligned} \tag{3.6}$$

where the superscript ^{T} now represents the transpose of a matrix (and not the adjoint operator). From the definition of $\mathcal{E}$, we quickly realize that $\delta\mathcal{E}(p;\tilde{p})(x,t) = \delta\Lambda(B(p);B(\tilde{p}))$. Hence, (3.6) reduces to

$$\boldsymbol{\delta J}_0(p;\tilde{p}) = \int_0^T \left(\left[\delta\Lambda^T(B(p);B(\tilde{p}))\,\mathcal{E}(p)\,\right](x_0,t) + \left[\delta\Lambda^T(B(p);B(\tilde{p}))\,\mathcal{E}(p)\right](x_L,t)\right)\; dt\,. \tag{3.7}$$

We now see the necessity of evaluating the term $\delta\Lambda(B(p);B(\tilde{p}))$ explicitly. Albeit seemingly complicated, the difficulty of the evaluation will be mitigated via the solving of a related adjoint equation, which will be described in the next subsection.

3.2. Linearization and adjoint formulation.

The adjoint formulation can be understood from multiple perspectives. One way is the Lagrange framework in which the adjoint variables are Lagrange multipliers. This method, commonly used in the aeronautical community, was popularized by Jameson [37], because it provided a solid connection to theories of constrained optimal control and optimization. Another type of approach, the duality framework, requires one to linearize the system in order to derive the adjoint equations. We will use the duality framework in this paper, however, the Lagrange framework provides the exact same adjoint formulation.

3.2.1. Linearization of the forward system.

In this subsection, we aim to linearize the forward system (2.2) as follows. We consider an ϵ-perturbation of B, $B^{\varepsilon} := B + \epsilon\tilde{B}$ along the direction $\tilde{B}$, and we see how the resulting $\boldsymbol{U}$ that satisfies (2.2) is perturbed. We denote $\boldsymbol{U}^{\epsilon}$ as the solution to (2.2) given B^{ε} and define

$$\tilde{\boldsymbol{U}} := \lim_{\epsilon\to 0} \frac{\boldsymbol{U}^{\epsilon} - \boldsymbol{U}}{\epsilon}, \tag{3.8}$$

whenever it exists. Now we quickly realize that

$$\tilde{\boldsymbol{U}}(x,0) = \lim_{\epsilon\to 0} \frac{\boldsymbol{U}^{\epsilon}(x,0) - \boldsymbol{U}(x,0)}{\epsilon} = 0, \tag{3.9}$$

as the initial conditions of $\boldsymbol{U}^{\epsilon}$ and $\boldsymbol{U}$ shall coincide. Moreover, taking the differences of the respective equations coming from (2.2) for $\boldsymbol{U}^{\epsilon}$ and $\boldsymbol{U}$ directly gives

$$0 = \lim_{\epsilon\to 0} \frac{1}{\epsilon} \left(\partial_t[\boldsymbol{U}^{\epsilon} - \boldsymbol{U}] + [\boldsymbol{A}(\boldsymbol{U}^{\epsilon})\partial_x \boldsymbol{U}^{\epsilon} - \boldsymbol{A}(\boldsymbol{U})\partial_x \boldsymbol{U}] - [\boldsymbol{S}(\boldsymbol{U}^{\epsilon}, B^{\epsilon})\boldsymbol{U}^{\epsilon} - \boldsymbol{S}(\boldsymbol{U}, B)\boldsymbol{U}] \right), \tag{3.10}$$

and each term in the bracket can be simplified whenever they exist. For instance, we directly have

$$\lim_{\epsilon\to 0} \frac{1}{\epsilon} \partial_t \left(\boldsymbol{U}^{\epsilon} - \boldsymbol{U}\right) = \partial_t \tilde{\boldsymbol{U}}. \tag{3.11}$$

Meanwhile, we may simplify the flux term as

$$\begin{aligned}
&\lim_{\epsilon\to 0} \frac{1}{\epsilon} \left[\boldsymbol{A}(\boldsymbol{U}^{\epsilon})\partial_x \boldsymbol{U}^{\epsilon} - \boldsymbol{A}(\boldsymbol{U})\partial_x \boldsymbol{U}\right] \\
&\quad = \lim_{\epsilon\to 0} \frac{1}{\epsilon} \left[\boldsymbol{A}(\boldsymbol{U}) \left(\partial_x \left(\boldsymbol{U} + \epsilon\tilde{\boldsymbol{U}} \right) - \partial_x(\boldsymbol{U}) \right) + \left(\boldsymbol{A}(\boldsymbol{U} + \epsilon\tilde{\boldsymbol{U}}) - \boldsymbol{A}(\boldsymbol{U}) \right) \partial_x(\boldsymbol{U} + \epsilon\tilde{\boldsymbol{U}}) \right] \\
&\quad = \boldsymbol{A}(\boldsymbol{U})\partial_x \tilde{\boldsymbol{U}} + \left(\sum_{k=1}^{m} \partial_{\boldsymbol{U}_k} A(\boldsymbol{U})\, \tilde{\boldsymbol{U}}_k \right) \partial_x \boldsymbol{U} \\
&\quad = \boldsymbol{A}(\boldsymbol{U})\partial_x \tilde{\boldsymbol{U}} + \left(\sum_{j=1}^{m} \partial_{\boldsymbol{U}_j} A(\boldsymbol{U}) \partial_x \boldsymbol{U} \right) \tilde{\boldsymbol{U}} \\
&\quad = \boldsymbol{A}(\boldsymbol{U}) \left[\partial_x \tilde{\boldsymbol{U}} \right] + \left[\partial_x \boldsymbol{A}(\boldsymbol{U}) \right] \tilde{\boldsymbol{U}} \\
&\quad = \partial_x \left[\boldsymbol{A}(\boldsymbol{U}) \tilde{\boldsymbol{U}} \right].
\end{aligned} \tag{3.12}$$

Here, the first equality is derived from rewriting the flux term after adding and subtracting $\boldsymbol{A}(\boldsymbol{U})\partial_x \boldsymbol{U}^{\epsilon}$ and applying the definition (3.8). The second equality follows from applying the limit in which the second term is the result of partial derivatives involving the chain rule. The third equality is determined from the symmetry relation $\partial_{U_k}\boldsymbol{A}_{ij}(U) = \partial_{U_k}\partial_{U_j}\boldsymbol{F}_i(U) = \partial_{U_j}\boldsymbol{A}_{ik}(U)$. The last equality follows from the product rule.

Likewise, we can simplify the source term and obtain

$$
\begin{aligned}
\text{(3.13)}\quad & \lim_{\epsilon\to 0}\frac{1}{\epsilon}\left(\boldsymbol{S}(\boldsymbol{U}^{\epsilon}, B^{\epsilon})\boldsymbol{U}^{\epsilon} - \boldsymbol{S}(\boldsymbol{U}, B)\boldsymbol{U}\right) \\
&= \lim_{\epsilon\to 0}\frac{1}{\epsilon}\left(\boldsymbol{S}(\boldsymbol{U}, B)\left(\boldsymbol{U}^{\epsilon} - \boldsymbol{U}\right) + \left(\boldsymbol{S}(\boldsymbol{U}^{\epsilon}, B^{\epsilon}) - \boldsymbol{S}(\boldsymbol{U}, B)\right)\boldsymbol{U}^{\epsilon}\right) \\
&= \boldsymbol{S}(\boldsymbol{U}, B)\tilde{\boldsymbol{U}} + \lim_{\epsilon\to 0}\frac{1}{\epsilon}\left(\boldsymbol{S}(\boldsymbol{U} + \epsilon\tilde{\boldsymbol{U}}, B + \epsilon\tilde{B}) - \boldsymbol{S}(\boldsymbol{U}, B)\right)\left(\boldsymbol{U} + \epsilon\tilde{\boldsymbol{U}}\right) \\
&= \boldsymbol{S}(\boldsymbol{U}, B)\tilde{\boldsymbol{U}} + \left(\sum_{i=1}^{m}\partial_{\boldsymbol{U}_i}\boldsymbol{S}(\boldsymbol{U}, B)\tilde{\boldsymbol{U}}_i + \partial_B\boldsymbol{S}(\boldsymbol{U}, B)\tilde{B}\right)\boldsymbol{U} \\
&:= \left(\boldsymbol{S}(\boldsymbol{U}, B) + \boldsymbol{C}(\boldsymbol{U}, B)\right)\tilde{\boldsymbol{U}} + \partial_B\boldsymbol{S}(\boldsymbol{U}, B)\boldsymbol{U}\tilde{B},
\end{aligned}
$$

where $\boldsymbol{C}$ denotes the matrix $\boldsymbol{C}_{ij} = \sum_{k=1}^{m}\frac{\partial\boldsymbol{S}_{ik}}{\partial\boldsymbol{U}_j}\boldsymbol{U}_k$. The first equality is derived from rewriting the original equality after adding and subtracting the term $\boldsymbol{S}(\boldsymbol{U}, B)\boldsymbol{U}^{\epsilon}$. The second equality follows from applying the limit to the first term and from applying the definitions in (3.8) and of B^{ϵ} to the second term. The third equality follows from taking partial derivatives which involves the chain rule and applying the limit. Last, the fourth equality is the result of a symmetry relation.

Substituting (3.11), (3.12), and (3.13) into (3.10), and combining that with the initial condition (3.9), we therefore obtain the following linear system for $\tilde{\boldsymbol{U}}$,

$$
\text{(3.14)}\quad
\begin{cases}
\left(\partial_t - \boldsymbol{S} - \boldsymbol{C}\right)\tilde{\boldsymbol{U}} + \partial_x\left(\boldsymbol{A}\tilde{\boldsymbol{U}}\right) = \left(\partial_B\boldsymbol{S}(\boldsymbol{U}, B)\boldsymbol{U}\right)\tilde{B}, & (x,t)\in(x_0, x_L)\times(0,T], \\
\tilde{\boldsymbol{U}}(x,0) = 0, & x\in\left[x_0, x_L\right],
\end{cases}
$$

which serves as the linearization of the forward system (2.2).

3.2.2. The adjoint system.

With the linearization process given in the previous subsection, we may proceed to obtain $\delta\Lambda(B(p); B(\tilde{p}))$ at the boundary points, and thereby evaluate $\boldsymbol{\delta J}_0(p;\tilde{p})$ appropriately.

We start by considering $\boldsymbol{\sigma}$ which satisfies the following adjoint system with final time condition and boundary conditions:

$$
\text{(3.15)}\quad
\begin{cases}
(\partial_t + \boldsymbol{A}^T\partial_x + \boldsymbol{S}^T + \boldsymbol{C}^T)\boldsymbol{\sigma} = 0, & x\in(x_0, x_L)\times(0,T], \\
\boldsymbol{\sigma}(x,T) = 0, & x\in(x_0, x_L), \\
\boldsymbol{\sigma}(x_0,t) = -(\boldsymbol{A}^T)^{-1}(x_0,t)\left[\mathcal{E}(p)\right](x_0,t), & t\in(0,T], \\
\boldsymbol{\sigma}(x_L,t) = (\boldsymbol{A}^T)^{-1}(x_L,t)\left[\mathcal{E}(p)\right](x_L,t) & t\in(0,T].
\end{cases}
$$

In the particular case of the SWEs, the matrices appearing in (3.15) are given by

$$
\text{(3.16)}\quad
\boldsymbol{A}^T = \begin{bmatrix} 0 & gh - u^2 \\ 1 & 2u \end{bmatrix}, \qquad \boldsymbol{S}^T = \begin{bmatrix} 0 & -g\partial_x b \\ 0 & 0 \end{bmatrix}, \qquad \boldsymbol{C}^T = 0.
$$

Taking the inner product of the solution $\boldsymbol{\sigma}$ from (3.15) and the weak formulation of the linearized system in (3.14), we get

$$\begin{aligned}&\int_0^T\int_{x_0}^{x_L}\boldsymbol{\sigma}^T\left(\partial_B\boldsymbol{S}(\boldsymbol{U},B)\boldsymbol{U}\right)\tilde{B}\ dxdt\\ &\qquad=\int_0^T\int_{x_0}^{x_L}\boldsymbol{\sigma}^T\left(\partial_t+\partial_x\boldsymbol{A}-\boldsymbol{S}-\boldsymbol{C}\right)\tilde{\boldsymbol{U}}\ dxdt\\ &\qquad=-\int_0^T\int_{x_0}^{x_L}\tilde{\boldsymbol{U}}^T\left(\partial_t+\boldsymbol{A}^T\partial_x+\boldsymbol{S}^T+\boldsymbol{C}^T\right)\boldsymbol{\sigma}\ dxdt\\ &\qquad\quad+\int_{x_0}^{x_L}\tilde{\boldsymbol{U}}^T\boldsymbol{\sigma}\Big|_{t=0}^{t=T}\ dx+\int_0^T\tilde{\boldsymbol{U}}^T\boldsymbol{A}^T\boldsymbol{\sigma}\Big|_{x=x_0}^{x=x_L}\ dt,\end{aligned}\tag{3.17}$$

where we simplify further, with (3.15), to obtain

$$\begin{aligned}&\int_0^T\int_{x_0}^{x_L}\boldsymbol{\sigma}^T(x,t)\left(\partial_B\boldsymbol{S}(\boldsymbol{U},B)\boldsymbol{U}\right)(x,t)\tilde{B}(x,t)\ dxdt\\ &\qquad=\int_0^T\left[\delta\Lambda^T(B(p);\tilde{B}(p))\,\mathcal{E}(p)\right](x_L,t)\ dt\\ &\qquad\quad+\int_0^T\left[\delta\Lambda^T(B(p);\tilde{B}(p))\,\mathcal{E}(p)\right](x_0,t)\ dt.\end{aligned}\tag{3.18}$$

Here the last equality follows from the choice of boundary conditions described in (3.15) and the fact that $\boldsymbol{U}=\Lambda(B(p))$ implies $\tilde{\boldsymbol{U}}=\delta\Lambda(B(p);\tilde{B}(p))$. We may now readily substitute (3.18) into the expression (3.7) to obtain

$$\boldsymbol{\delta J}_0(p;\tilde{p})=\int_0^T\int_{x_0}^{x_L}\boldsymbol{\sigma}^T(x,t)\left(\partial_B\boldsymbol{S}(\boldsymbol{U},B)\boldsymbol{U}\right)(x,t)\tilde{B}(x,t)\ dxdt.\tag{3.19}$$

By utilizing the fact that

$$\left[\tilde{B}(p)\right](x,t)=\delta B(p;\tilde{p})=[\delta(B_0+pB_1)](p;\tilde{p})=B_1(x)\,\tilde{p}(t),\tag{3.20}$$

we further simplify (3.19) to

$$\boldsymbol{\delta J}_0(p;\tilde{p})=\int_0^T\left(\int_{x_0}^{x_L}\boldsymbol{\sigma}^T(x,t)\left(\partial_B\boldsymbol{S}(\boldsymbol{U},B)\boldsymbol{U}\right)(x,t)B_1(x)\ dx\right)\tilde{p}(t)\ dt.\tag{3.21}$$

Therefore from definition (3.5), we obtain the following (formal) gradient from (3.21):

$$\nabla\boldsymbol{J}_0(p)(t)=\int_{x_0}^{x_L}\boldsymbol{\sigma}^T(x,t)\left(\partial_B\boldsymbol{S}(\boldsymbol{U},B)\boldsymbol{U}\right)(x,t)B_1(x)\ dx\,.\tag{3.22}$$

We again remark that, in the case of the SWEs, we have

$$\partial_B\boldsymbol{S}(\boldsymbol{U},B)\boldsymbol{U}=\begin{bmatrix}0&0\\-g&0\end{bmatrix}\begin{bmatrix}h\\hu\end{bmatrix}=\begin{bmatrix}0\\-gh\end{bmatrix},\qquad B_1=\partial_x b_1,\tag{3.23}$$

and therefore the gradient is simplified to the form

$$\begin{aligned}\nabla\boldsymbol{J}_0(p)&=\int_{x_0}^{x_L}\left(\begin{bmatrix}\sigma_1\\\sigma_2\end{bmatrix}^T\begin{bmatrix}0\\-gh\end{bmatrix}\right)(x,t)\partial_x b_1(x)\ dx\\ &=\int_{x_0}^{x_L}-g\sigma_2(x,t)h(x,t)\partial_x b_1(x)\ dx,\end{aligned}\tag{3.24}$$

where $\boldsymbol{\sigma}=\left[\begin{smallmatrix}\sigma_1\\\sigma_2\end{smallmatrix}\right]$ is the solution of the adjoint equation (3.15).

3.3. Numerical scheme for the inverse problem. In this subsection we will discuss the numerical algorithms for the inverse problem. The DG scheme will be employed to solve the adjoint problem (3.15) and an iterative method will be presented to update the function p with the suitably chosen regularization terms.

3.3.1. DG method for the adjoint problem. Noting that the spatial derivative in the adjoint problem (3.15) is not in the conservative form, we start by reformulating the adjoint problem as a balance law of the form

$$\partial_t \boldsymbol{\sigma} + \partial_x \left(\boldsymbol{A}^T \boldsymbol{\sigma} \right) = (\partial_x \boldsymbol{A}^T - \boldsymbol{S}^T - \boldsymbol{C}^T)\boldsymbol{\sigma}, \tag{3.25}$$

where $\boldsymbol{A} = \boldsymbol{A}(\boldsymbol{U})$ does not depend on the unknown $\boldsymbol{\sigma}$. Following the same discretization strategy as presented in section 2.2, we seek an approximate solution $\boldsymbol{\sigma}_h$ in which $\boldsymbol{\sigma}_h^{(i)}$ for $i = 1, \ldots, m$ belong to $\mathbb{V}_h^k$. The DG method in cell I_j becomes

$$\begin{aligned} \int_{I_j} \partial_t \boldsymbol{\sigma}_h \boldsymbol{v} \, dx - \int_{I_j} \boldsymbol{A}(\boldsymbol{U}_h)^T \boldsymbol{\sigma}_h \partial_x \boldsymbol{v} \, dx + \hat{\boldsymbol{G}}_{j+\frac{1}{2}} \boldsymbol{v}^-_{j+\frac{1}{2}} - \hat{\boldsymbol{G}}_{j-\frac{1}{2}} \boldsymbol{v}^+_{j-\frac{1}{2}} \\ = \int_{I_j} \left(\partial_x \boldsymbol{A}^T(\boldsymbol{U}_h) - \boldsymbol{S}^T(B_h) - \boldsymbol{C}^T \right) \boldsymbol{\sigma}_h \boldsymbol{v} \, dx, \end{aligned} \tag{3.26}$$

where $\boldsymbol{v} \in \mathbb{V}_h^k$ is a vector of test functions and the Lax–Friedrichs numerical flux takes the form

$$\hat{\boldsymbol{G}}_{j+\frac{1}{2}} = \frac{1}{2} \left(\boldsymbol{A}(\boldsymbol{U}^-_{h,j+\frac{1}{2}})^T \boldsymbol{\sigma}^-_{h,j+\frac{1}{2}} + \boldsymbol{A}(\boldsymbol{U}^+_{h,j+\frac{1}{2}})^T \boldsymbol{\sigma}^+_{h,j+\frac{1}{2}} - \alpha \left(\boldsymbol{\sigma}^+_{h,j+\frac{1}{2}} - \boldsymbol{\sigma}^-_{h,j+\frac{1}{2}} \right) \right) \tag{3.27}$$

with the value of α being the same as in the forward DG scheme, described in section 2.2.

3.3.2. Regularization and update scheme. In this subsection, we describe the numerical method designed to recover the function $p(t)$ via an iterative scheme. Usually, either a descent type [6], Newton type [30], or a trust region algorithm [60] is employed. A Newton type algorithm usually provides a certain acceleration to the convergence, but as a trade-off, it is usually more computationally expensive. In this work, we employ a descent type algorithm, in light of the fact that our functional is highly nonlinear and highly nonconvex, applying a higher order method may result in getting stuck at a local optimum even more easily.

We employ an operator splitting algorithm to update the function $p(t)$. The scheme is initialized with a random initial guess for p, denoted by $p^{0,\text{noisy}}$. We use multiplicative noise following a uniform distribution, i.e., $p^{0,\text{noisy}} = p^0 \nu$, where $\nu \sim \mathcal{U}\left[1 - \frac{1}{2}\eta_p, 1 + \frac{1}{2}\eta_p\right]$, to define the random initial guess. Not only is the update of p dependent on $\nabla \boldsymbol{J}_0$, but it also relies on a regularization term. The regularization term ensures the optimization problem is locally convex and makes it possible to solve an ill-posed problem efficiently by incorporating a priori knowledge of the profile to be reconstructed. Various regularization terms have been constructed for different purposes. For instance, L^1 regularization [65, 22] results in a simpler sparse solution. On the other hand, TV regularization [58] favors piecewise constant functions of the coefficients to be recovered, whereas Sobolev regularization [14] favors smoothness of the coefficients to be reconstructed.

Before we focus on our choice of regularization, we first discuss the update algorithm. To better motivate our choice of algorithm, we start by simplifying our discussion and considering the situation when there is only one regularization term.

In this case, the proximal gradient descent method (or forward-backward splitting) [2, 52] is a common choice. The explicit term is usually assigned as the term coming from the gradient of a more complicated functional. The implicit term is typically chosen so that the proximal map is easy to evaluate and the stability of the algorithm is increased. Consequently, the kth iteration is given by

$$p^{k+1} = p^k - \ell_k \nabla \boldsymbol{J}_0\left(p^k\right) - \ell_k \partial \boldsymbol{R}\left(p^{k+1}\right), \tag{3.28}$$

where ℓ_k is the step size or learning rate and ∂ represents the subgradient when the proximal map of $\boldsymbol{R}$ can be computed. In our work, we will choose a constant step size, i.e., $\ell_k = \ell$ for all k. The scheme can be rewritten so that the update for iteration $k+1$ only depends on the information from iteration k,

$$p^{k+1} = \left(\mathcal{I} + \ell \partial \boldsymbol{R}\right)^{-1}\left(p^k - \ell \nabla \boldsymbol{J}_0\left(p^k\right)\right), \tag{3.29}$$

where $\mathcal{I}$ is the identity matrix and

$$\left(\mathcal{I} + \ell \partial \boldsymbol{R}\right)^{-1}(w) = \arg\min_y \left\{ \boldsymbol{R}(y) + \frac{1}{2\ell}\|w - y\|_2^2 \right\} = \operatorname{prox}_{R,\ell}(w). \tag{3.30}$$

This leads to the formulation

$$p^{k+1} = \arg\min_y \left\{ \boldsymbol{R}(y) + \frac{1}{2\ell}\|p^k - \ell \nabla \boldsymbol{J}_0(p^k) - y\|_2^2 \right\}. \tag{3.31}$$

A common choice for the regularization is L^1 regularization, where $\boldsymbol{R}(y) = \gamma\|y - p_0\|_1$ with p_0 being a chosen coefficient of homogeneous background and γ a scalar parameter, aiming to impose sparsity of the difference between the resulting optimum and p_0. The proximal gradient method coming from this choice of regularizer is

$$\begin{aligned} p^{k+1} &= \arg\min_y \left\{ \gamma\|y - p_0\|_1 + \frac{1}{2\ell}\|p^k - \ell \nabla \boldsymbol{J}_0\left(p^k\right) - y\|_2^2 \right\} \\ &= \mathcal{S}_{\gamma\ell}\left(p^k - \ell \nabla \boldsymbol{J}_0\left(p^k\right) - p_0\right) + p_0, \end{aligned} \tag{3.32}$$

where the shrinkage operator $\mathcal{S}_{\gamma\ell}$ [10, 9, 65] is given as follows:

$$\mathcal{S}_{\gamma\ell}\left(p\right) = \operatorname{sign}\left(p\right) \max\left\{|p| - \ell\gamma, 0\right\}. \tag{3.33}$$

After briefly describing the simple motivating example which carries only one regularization term, we now describe the combination of regularization terms that we use in our work, and how we perform the operator splitting in our algorithm. In this paper, the regularization term is taken as a sum of two regularizers

$$\boldsymbol{R}(p) = \boldsymbol{R}_{L^1}(p - p_0) + \boldsymbol{R}_{H^1}(p), \tag{3.34}$$

where $\boldsymbol{R}_{L^1}(p) = \|p\|_1$ represents L^1 regularization and $\boldsymbol{R}_{H^1}(p) = \|\nabla p\|_2^2$ represents H^1 regularization. The L^1 regularization term will aid in removing the noise by sparsifying it, while the H^1 regularization term will be beneficial for the purpose of smoothing out the noisy data, an advantage over total variation ($\|\nabla p\|_1$) regularization. H^1 regularization has been shown to be good for flow control problems [8, 28] as well as image reconstruction and deblurring [51, 40].

Now, we wish to minimize $\boldsymbol{J}_0(p) + \boldsymbol{R}_{L^1}(p) + \boldsymbol{R}_{H^1}(p)$, which reduces to finding p so that

$$0 \in \nabla \boldsymbol{J}_0(p) + \partial \boldsymbol{R}_{L^1}(p) + \nabla \boldsymbol{R}_{H^1}(p). \tag{3.35}$$

We must be careful in our approach and employ a more complicated splitting scheme than the proximal gradient descent since we now have an additional operator. In our work, we adapt the three-operator splitting algorithm [11], which we will describe in detail in Algorithm 3.1. For simplicity, we introduce the following notation corresponding to each regularization term,

$$\begin{aligned} \mathcal{J}_{\ell\gamma_L \boldsymbol{R}_{L^1}}(\omega) &= \left(\mathcal{I} + \ell\gamma_L \boldsymbol{R}_{L^1}\right)^{-1}(\omega) \\ &= \operatorname{sign}(\omega - p_0) \max\left\{|\omega - p_0| - \ell\gamma_L, 0\right\} + p_0 \end{aligned} \tag{3.36}$$

and

$$\mathcal{J}_{\ell\gamma_H \boldsymbol{R}_{H^1}}(\omega) = (\mathcal{I} + \ell\gamma_H \boldsymbol{R}_{H^1})^{-1}(\omega) = \left(\mathcal{I} - \ell\gamma_H \Delta\right)^{-1}(\omega), \tag{3.37}$$

where γ_L is the L^1 regularization parameter and γ_H is the H^1 regularization parameter. Furthermore, we would like to note that the gradient of the cost function is time dependent, i.e., $\nabla \boldsymbol{J}_0(p^k, t)$, but we denote it as $\nabla \boldsymbol{J}_0(p^k)$ for the sake of simplifying notation. The update becomes

$$p^{k+1} = \mathcal{J}_{\ell\gamma_L \boldsymbol{R}_{L^1}} \circ \left[z^k + \lambda_k \left(\mathcal{J}_{\ell\gamma_H \boldsymbol{R}_{H^1}} \circ \left[2p^k - z^k - \ell\nabla \boldsymbol{J}_0(p^k)\right] - p^k\right)\right], \tag{3.38}$$

where z^0 is originally initialized to be p^0 and λ_k is the relaxation parameter which can be used to help speed up the rate of convergence of the iterative solutions. We are now ready to introduce our algorithm.

Algorithm 3.1 Three-operator splitting algorithm.

```
initialize $p^0$ to be the random initial guess
initialize $z^0 = p^0$
set regularization parameters $\gamma_L, \gamma_H$
set relaxation parameter $(\lambda_k)_{k\geq 0}$
set learning rate $\ell$
for $k = 0, 1, \dots$ do
    compute $\Lambda(B(p^k))$ from $B(p^k)$ by solving the forward problem (2.2)
    compute $\boldsymbol{\sigma}$ from $(B(p^k), \Lambda(B(p^k)))$ by solving the adjoint problem (3.15)
    evaluate $\nabla \boldsymbol{J}_0(p^k) = \int_{x_0}^{x_L} \boldsymbol{\sigma}^T(x,t) \left(\partial_B \boldsymbol{S}(\boldsymbol{U}, B)\boldsymbol{U}\right)(x,t) B_1(x)\, dx$
    define $\kappa^k = 2p^k - z^k - \ell\nabla \boldsymbol{J}_0(p^k)$
    evaluate $\omega^k = \mathcal{J}_{\ell\gamma_H \boldsymbol{R}_{H^1}}(\kappa^k) = (\mathcal{I} - \ell\gamma_H\Delta)^{-1}(\kappa^k)$
    update $z^{k+1} = z^k + \lambda_k(\omega^k - p^k)$
    update $p^{k+1} = \mathcal{J}_{\ell\gamma \boldsymbol{R}_{L^1}}(z^{k+1}) = \operatorname{sign}(z^{k+1} - p_0)\max\{|z^{k+1} - p_0| - \ell\gamma_L, 0\} + p_0$
end
```

4. Numerical examples for the SWEs.

In this section, we will be considering the one-dimensional nonlinear SWEs (2.5). We aim to recover the temporal component $p(t)$ in the bottom topography function $b(x,t)$; see (2.3).

In all the numerical tests, we use a relaxation parameter of $\lambda_k = 1$, a noise parameter for the measured data of $\eta_{meas} = 0.1$ or 5% noise, a noise parameter for the initial guess p^0 of $\eta_p = 0.25$ or 12.5% noise. A coefficient of homogeneous background is assumed to be known and taken as $p_0 = 1$. Each test is run for 1,000 iterations. The iteration with the smallest residue, $\boldsymbol{J}_0(p^k)$, is selected as the best recovered representation for the true temporal component of the bottom function $p(t)$. The measured data are computed using a high order accurate DG method with a uniform mesh of 400 cells and P^3 piecewise polynomials, with noise added to the DG solutions to represent noisy measurement. A mesh of 50 uniform cells with P^2 piecewise polynomials is used to solve the forward problem and a uniform mesh of 25 cells with P^1 piecewise polynomials is used to solve the adjoint problem, unless stated otherwise. The measured data, forward, and adjoint solvers are designed with different meshes and polynomial degree approximations in an effort to avoid committing an "inverse crime" [62].

4.1. Tests for recovering different time profiles of $p(t)$.

In this subsection, we will perform numerical experiments aiming to recover several unknown time profiles, $p_{true}(t)$, from noisy boundary measurements.

We solve the forward problem (2.1) with the DG method described in section 2.2, where our computational domain is chosen to be $[x_0, x_L] = [0, 1]$, the initial conditions are given by

$$h(x,0) = 7 + \exp(\sin(2\pi x)), \qquad hu(x,0) = \cos(2\pi x), \tag{4.1}$$

and the spatial components of the bottom topography function are defined as

$$b_0(x) = \cos(\sin(2\pi x), \qquad b_1(x) = \sin^2(\pi x). \tag{4.2}$$

The final time is set as $T = 0.05$ and periodic boundary conditions are used.

We examine several choices for the true value of $p(t)$ and the corresponding initial guesses, which are outlined in Table 1. A constant learning rate of $\ell = 0.6$ is used in each test. The regularization parameters are fixed with $\gamma_L = 1 \times 10^{-6}$ in all examples and $\gamma_H = 5 \times 10^{-8}$ in cases (4.1a), (4.1b), (4.1d), (4.1e), $\gamma_H = 1 \times 10^{-8}$ in cases (4.1c), (4.1d) and $\gamma_H = 5 \times 10^{-9}$ in case (4.1f).

TABLE 1
The true function for $p(t)$ denoted as p_{true}, and the corresponding initial guess used, p^0, with $\beta = -10{,}000$. Multiplicative noise is applied to p^0 in the simulations.

Case	$p_{true}(t)$	$p^0(t)$
(4.1a)	$e^{\beta\left(t-\frac{1}{3}T\right)^2}+1$	$e^{\beta\left(t-\frac{2}{3}T\right)^2}+1$
(4.1b)	$e^{\beta\left(t-\frac{2}{3}T\right)^2}+1$	$e^{\beta\left(t-\frac{1}{3}T\right)^2}+1$
(4.1c)	$e^{2\beta\left(t-\frac{1}{4}T\right)^2}+e^{2\beta\left(t-\frac{3}{4}T\right)^2}+1$	$\frac{3}{2}e^{\beta\left(t-\frac{1}{2}T\right)^2}+1$
(4.1d)	$e^{\beta(t-0.3T)^2}+\frac{3}{2}e^{2\beta(t-0.7T)^2}+1$	$3\cos^2\left(\frac{10\pi}{T}t\right)+\frac{3}{4}$
(4.1e)	$\frac{3}{2}e^{\beta(t-0.3T)^2}+e^{2\beta(t-0.7T)^2}+1$	$3\cos^2\left(\frac{10\pi}{T}t\right)+\frac{3}{4}$
(4.1f)	$e^{4\beta\left(t-\frac{1}{4}T\right)^2}+\frac{3}{2}e^{4\beta\left(t-\frac{1}{2}T\right)^2}-\frac{1}{2}e^{4\beta\left(t-\frac{3}{4}T\right)^2}+1$	$3\cos^2\left(\frac{10\pi}{T}t\right)+\frac{3}{4}$

Cases (4.1a) and (4.1b) represent the situation in which the true value of $p(t)$ is a bump function that is noncentered with respect to the time interval and the corresponding initial guess is a noisy horizontal shift of $p_{true}(t)$. The numerical results are shown in Figures 1 and 2. In both cases, the amplitude and shape of the true function and recovered numerical approximation are very close. The figures demonstrate that the scheme is robust even in the presence of multiplicative noise and the ill-posedness of the problem.

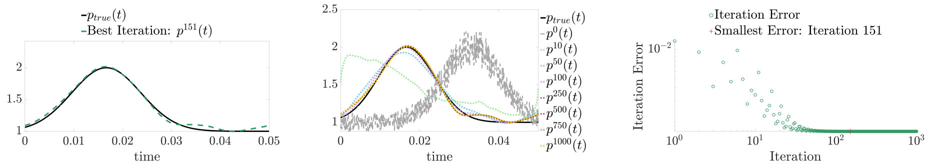


FIG. 1. *Results for case* (4.1a). *Left: plots of the true p and the numerically recovered p at iteration* 151, *corresponding to the smallest residue; Middle: plots of the true p, the noisy initial guess, and various iteration values for p; Right: iteration errors on a log-log scale.*

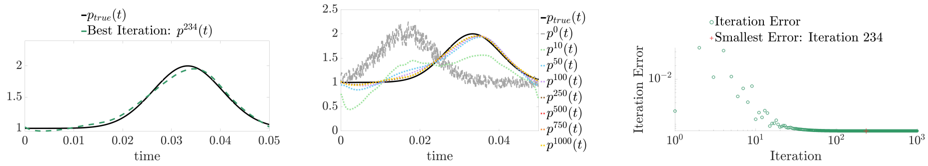


FIG. 2. *Results for case* (4.1b). *Left: plots of the true p and the numerically recovered p at iteration* 234, *corresponding to the smallest residue; Middle: plots of the true p, the noisy initial guess, and various iteration values for p; Right: iteration errors on a log-log scale.*

In case (4.1c), we examine a true function p that has two bumps of equal amplitude with an initial guess consisting of one bump with a larger amplitude. The results can be found in Figure 3. Cases (4.1d) and (4.1e) also include a true p function of two bumps, however, they have different amplitudes and the corresponding initial guesses are highly oscillatory trigonometric functions. The corresponding results are shown in Figures 4 and 5. In the examples with two bumps, the reconstructed function was also able to identify the two crests. The effect of the parameter γ_H is explored in these cases. The value $\gamma_H = 1 \times 10^{-8}$ is used for cases (4.1c) and (4.1d). We can see that this smaller choice of γ_H results in a less smooth solution in comparison to the results from cases (4.1d) and (4.1e) when $\gamma_H = 5 \times 10^{-8}$ is used. However, the plots corresponding to $\gamma_H = 5 \times 10^{-8}$ while more smooth, are more flattened.

The case (4.1f) contains two crests of different amplitudes and a trough for the true function with a highly oscillatory trigonometric function as the initial guess. Plots of the results corresponding to this case can be found in Figure 6. Additionally, for case (4.1f) we show the solutions of the forward problem in Figure 7 at different times ($t = \frac{T}{4}, \frac{T}{2}, \frac{3T}{4}$, and T). The water surface height, bottom topography, and water discharge of the measured data and the numerical solution at the iteration with the smallest residue are compared. The recovered bottom topography along with the recovered state variables match the true functions well, even for some more complicated choices of $p(t)$.

Figures 1–6 each contains a plot of the residues $\boldsymbol{J}_0$, defined by (3.1) and (3.2), at the endpoints of the spatial domain for each iteration on a log-log scale. We see in each case a similar behavior occurs in which an "elbow"-like shape appears. The portion of this residue curve with a steeper slope corresponds to the situation where the term $\boldsymbol{J}_0$ has a greater impact on the update of the function p, which happens for the beginning iterations. The flat portion of the residue curve corresponds to the situation when the iteration starts to enter a small neighborhood where the regularization term $\boldsymbol{R}(p)$ convexifies the optimization problem and dominates the update.

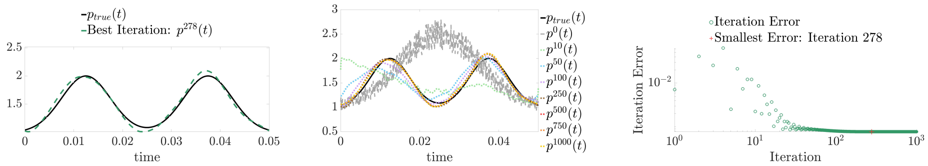
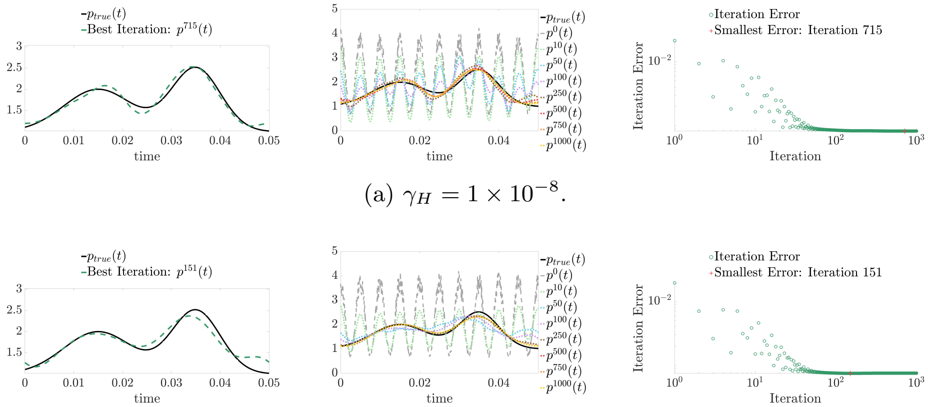


FIG. 3. *Results for case* (4.1c). *Left: plots of the true p and the numerically recovered p at iteration* 278, *corresponding to the smallest residue; Middle: plots of the true p, the noisy initial guess, and various iteration values for p; Right: iteration errors on a log-log scale.*



(a) $\gamma_H = 1 \times 10^{-8}$.

(b) $\gamma_H = 5 \times 10^{-8}$.

FIG. 4. *Results for case* (4.1d). *Left column: plots of the true p and the numerically recovered p at iteration* 715 *(top row) and* 151 *(bottom row), corresponding to the smallest residue; Middle column: plots of the true p, the noisy initial guess, and various iteration values for p; Right column: iteration errors on a log-log scale.*

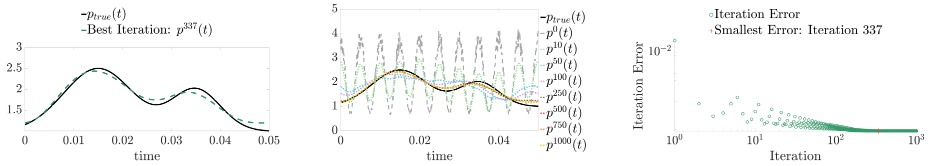


FIG. 5. *Results for case* (4.1e). *Left: plots of the true p and the numerically recovered p at iteration* 337, *corresponding to the smallest residue; Middle: plots of the true p, the noisy initial guess, and various iteration values for p; Right: iteration errors on a log-log scale.*

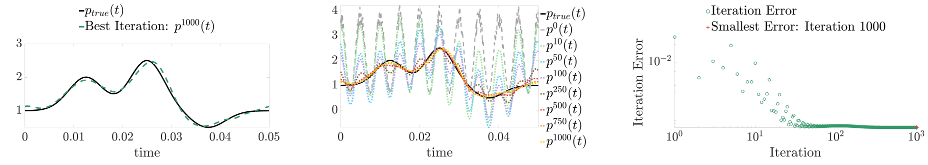


FIG. 6. *Results for case* (4.1f). *Left: plots of the true p and the numerically recovered p at iteration* 1,000, *corresponding to the smallest residue; Middle: plots of the true p, the noisy initial guess, and various iteration values for p; Right: iteration errors on a log-log scale.*

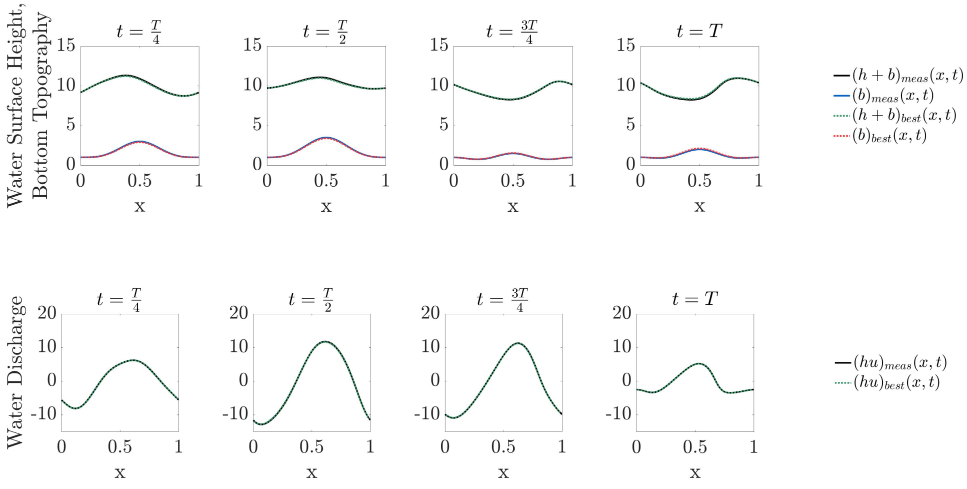


Fig. 7. *Comparison between the measured forward solutions and the results from the best iteration for SWEs case* (4.1f) *at times* $t = \frac{T}{4}, \frac{T}{2}, \frac{3T}{4},$ *and* T. *In the top row the bottom topography function,* b, *and the water surface heights,* $h + b$, *are compared. In the bottom row, the water discharge,* hu, *is compared. The measured data and results from the iterative scheme are well matched for all functions in each of the selected time snapshots.*

Table 2

The corresponding initial guesses used, p^0 *with* $T = 0.05$. *Multiplicative noise is applied to* p^0 *in the simulations.*

Case	$p^0(t)$
(4.2a)	1
(4.2b)	$4\sin^2\left(\frac{\pi}{T}t\right)$
(4.2c)	$-2\sin^2\left(\frac{\pi}{T}t\right) + 2$
(4.2d)	$3\cos^2\left(\frac{10\pi}{T}t\right) + 0.75$

4.2. Tests for recovering $p(t)$ from different initial guesses.

In this subsection, we run simulations with different initial guesses of $p^0(t)$ to recover the same $p_{true}(t)$. The goal is to demonstrate that the ability of our algorithm in recovering $p_{true}(t)$ does not depend on the initial guess.

We consider the forward problem with the initial conditions in (4.1) and the spatial bottom topography functions described in (4.2). The true time component of the bottom topography function is fixed to be

$$p(t) = \exp\left(\beta(t - 0.3T)^2\right) + \frac{3}{2}\exp\left(2\beta(t - 0.7T)^2\right) + 1 \tag{4.3}$$

with $\beta = -10,000$. Four different representative initial guesses, $p^0(t)$, listed in Table 2, will be tested. In all cases the final time is $T = 0.05$ (while the solution is still smooth) and periodic boundary conditions are used. The remaining hyperparameters include a learning rate of $\ell = 0.6$, $\gamma_L = 1 \times 10^{-6}$, and $\gamma_H = 1 \times 10^{-8}$.

The numerical results for cases (4.2a), (4.2b), and (4.2c) are shown in Figures 8–10, while the results for case (4.2d) can be found in Figure 4. In all four cases, the scheme is able to identify that $p_{true}(t)$ is a function consisting of two bumps, with the left bump (occurring earlier in time) having a smaller amplitude than the right

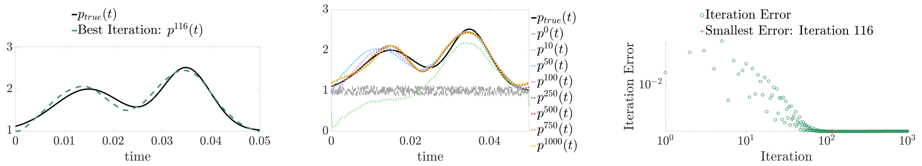


FIG. 8. *Results for case* (4.2a). *Left: plots of the true p and the p corresponding to the smallest residue error, at iteration* 116; *Middle: plots of the true p, the noisy initial guess, and various iteration values for p; Right: iteration errors on a log-log scale.*

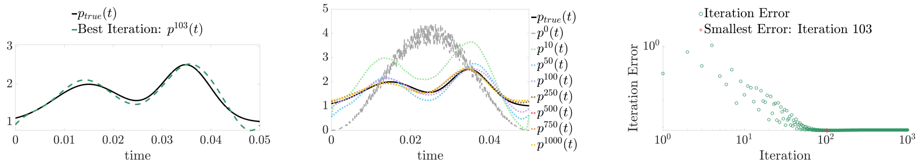


FIG. 9. *Results for case* (4.2b). *Left: plots of the true p and the p corresponding to the smallest residue error, at iteration* 103; *Middle: plots of the true p, the noisy initial guess, and various iteration values for p; Right: iteration errors on a log-log scale.*

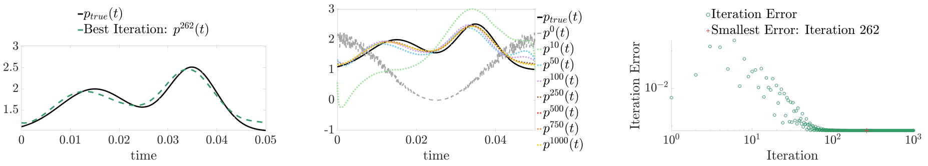


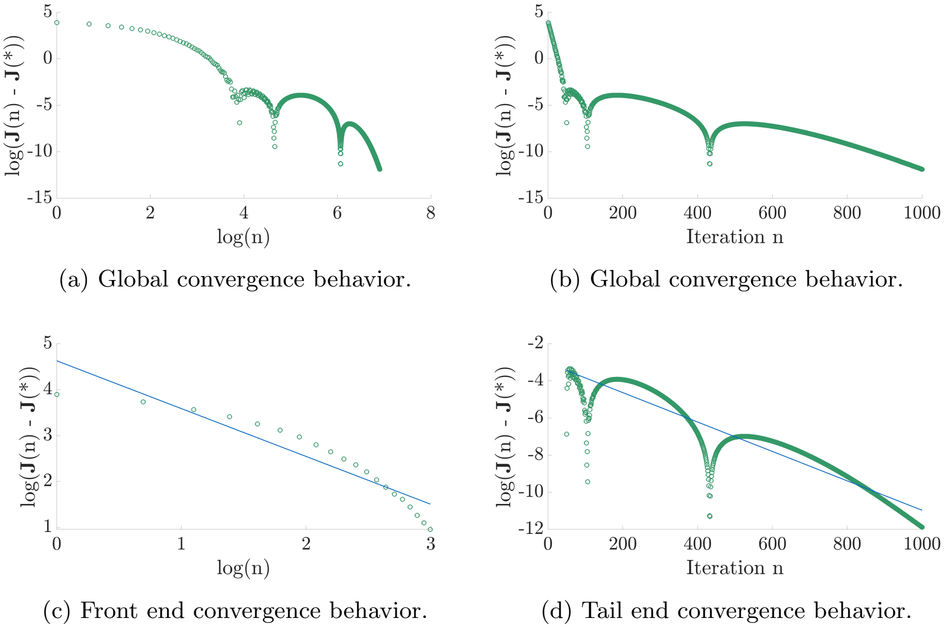
FIG. 10. *Results for case* (4.2c). *Left: plots of the true p and the p corresponding to the smallest residue error, at iteration* 262; *Middle: plots of the true p, the noisy initial guess, and various iteration values for p; Right: iteration errors on a log-log scale.*

bump (occurring later in time). This indicates the true function $p(t)$ can be recovered with the initial condition chosen from a wide range of functions. The numerical performances are similar in appearance and convergence rate. Cases (4.2a)–(4.2c) all achieve their best guess in less than 300 iterations. The true p and corresponding p with the smallest residue error tend to have the some discrepancy near the final time T. The exception is case (4.2a) in which the initial guess for p at time T is near to $p_{true}(T)$.

4.3. Convergence and accuracy. We discuss the convergence and accuracy of our numerical schemes in this section. All numerical simulations presented in this section consist of the same problem setup as was used in case (4.1d).

First, we discuss the local convergence of the optimization scheme. Since our problem is ill-posed, we structure our algorithm such that it optimizes the sum of the residual errors and the regularizers, that is $\boldsymbol{J} = \boldsymbol{J}_0(p) + \boldsymbol{R}_{L^1}(p) + \boldsymbol{R}_{H^1}(p)$. We apply our method to $\boldsymbol{J}$, which is locally convex and, therefore, the algorithm converges locally in first order to a local minimum of J.

For illustrative purpose, Figures 11 and 12 demonstrate the convergence behavior for two different choices of regularization. The results in Figure 11 correspond to regularization parameters $\gamma_L = 1 \times 10^{-6}$ and $\gamma_H = 5 \times 10^{-8}$ while the results in Figure 12 correspond to slightly higher regularization parameters $\gamma_L = 5 \times 10^{-6}$ and $\gamma_H = 1 \times 10^{-7}$. In both cases we examine the global and local convergence



(a) Global convergence behavior.

(b) Global convergence behavior.

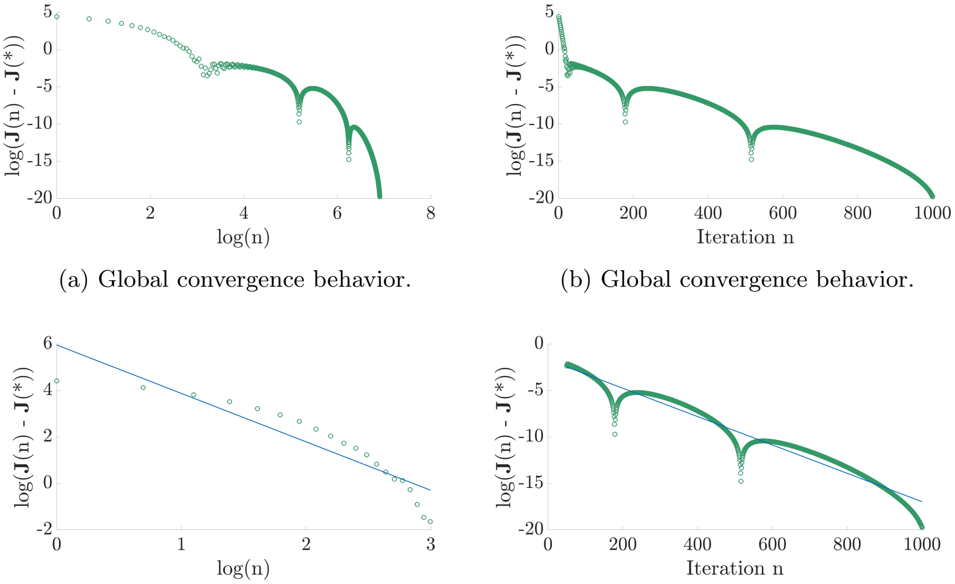
(c) Front end convergence behavior.

(d) Tail end convergence behavior.

Fig. 11. *Plots for convergence behavior corresponding to regularization parameters* $\gamma_L = 1 \times 10^{-6}$ *and* $\gamma_H = 5 \times 10^{-8}$*. Here* $\boldsymbol{J}(n)$ *is the error at iteration* n *including regularization and* $\boldsymbol{J}(*)$ *is the error at the local minimum. Plots in the top row demonstrate the global convergence behavior with* (a) *a log scale on the* x*-axis and* (b) *a standard scale on the* x*-axis. The bottom row of plots show local convergence behavior overlaid with the best fit linear line. Specifically,* (c) *demonstrates the first order convergence for the front end iterations* $(1$ *to* $20)$ *on a log-log scale with a fit line slope of* -1.0417 *and* (d) *demonstrates the first order convergence for the tail iterations* $(50$ *to* $1,000)$ *with a log scale on the* y*-axis and a fit line slope of* -0.0079.

behaviors. The difference between the iteration error and the local minimum error on a log scale is plotted against the log of the iteration number in Figures 11(a) and 12(a) and against the iteration number in Figures 11(b) and 12(b). These plots show different convergence behavior at the front end and the tail end of the iterative process. Hence, local convergence behavior is shown as well. For both cases we plot the errors against the first 20 iterations on a log-log scale with a best fit line to show the linear convergence. The results in Figure 11(c) are matched to a best fit line with a slope of -1.0417 showing first order convergence. The best fit line in Figure 12(c) has a slope of -2.0861 showing that an increased rate of convergence occurs with increased regulation rates. On the other hand, we also plot the errors corresponding to iterations 50 to 1,000 on a log scale against the iterations with a best fit line. Since the x-axis, in this case, is not on a log scale, the slope of the best fit line now corresponds to exponential rate of convergence. The results in Figure 11(d) correspond to a best fit line consisting of a slope of -0.0079 while the results for the case with higher regularization parameters demonstrated in Figure 12(d) have a slope of -0.0153, which is again twice as large as the case with smaller regularization parameters.

Second, we discuss the accuracy of the scheme. The solution of the forward problem is not analytically available, hence the order of accuracy is determined by comparing the state variables computed on a uniform mesh of cell width Δx that is



(a) Global convergence behavior.

(b) Global convergence behavior.

(c) Front end convergence behavior.

(d) Tail end convergence behavior.

FIG. 12. *Plots for convergence behavior corresponding to regularization parameters* $\gamma_L = 5 \times 10^{-6}$ *and* $\gamma_H = 1 \times 10^{-7}$*. Here* $\boldsymbol{J}(n)$ *is the error at iteration* n *including regularization and* $\boldsymbol{J}(*)$ *is the error at the local minimum. Plots in the top row demonstrate the global convergence behavior with* (a) *a log scale on the x-axis and* (b) *a standard scale on the x-axis. The bottom row of plots show local convergence behavior overlaid with the best fit linear line. Specifically,* (c) *demonstrates the first order convergence for the front end iterations (*1 *to* 20*) on a log-log scale with a fit line slope of* -2.0861 *and* (d) *demonstrates the first order convergence for the tail iterations (*50 *to* 1,000*) with a log scale on the y-axis and a fit line slope of* -0.0153*.*

repeatedly halved. We again consider the problem setup that was used in case (4.1d) with regularization parameters $\gamma_L = 1\times10^{-6}$ and $\gamma_H = 5\times10^{-8}$ and no noise. The L^1 errors and corresponding orders of accuracy for the forward problem computed with P^0, P^1, and P^2 piecewise polynomials are found in Table 3. We see that an order of $k+1$ accuracy is achieved for each polynomial degree k for both state variables h and hu.

4.4. Impact of inconsistent discretization schemes. In this section we further justify the decision to employ different discretization schemes for the forward and (linearized) adjoint schemes. The primary reason for this choice is to avoid any speculation of inverse crime. Here we compare the numerical results for case (4.1d) under two conditions. First, Figure 13 shows the numerical results under consistent conditions in which the forward and adjoint problems are both discretized using P^2 polynomials and a uniform mesh of 25 cells. Second, Figure 4(b) displays the numerical results under inconsistent conditions. The forward problem was discretized with P^2 piecewise polynomials and a mesh of 50 uniform cells, while the adjoint problem was solved with P^1 piecewise polynomials and a uniform mesh of 25 cells. The inconsistency of the discretized gradient, derived from the discrete adjoint solution, with the discretized forward model does not contaminate the behavior of the inverse problem algorithm as evidence by the results in Figure 4(b).

Table 3

L^1 errors and convergence orders of the accuracy test for the forward problem using P^0, P^1, and P^2 piecewise polynomials and a uniform mesh of N cells. In each case, $k+1$ order of accuracy is achieved.

		$k=0$		$k=1$		$k=2$	
	N	L^1 Error	Order	L^1 Error	Order	L^1 Error	Order
h	25	0.0490	-	2.9640×10^{-3}	-	2.4032×10^{-3}	-
	50	0.0260	0.9143	4.9877×10^{-4}	2.5711	1.1128×10^{-5}	7.7547
	100	0.0133	0.9725	1.2434×10^{-4}	2.0041	1.4679×10^{-6}	2.9223
	200	0.0067	0.9858	3.0999×10^{-5}	2.0040	1.7730×10^{-7}	3.0494
	400	0.0034	0.9927	7.7396×10^{-6}	2.0019	2.2152×10^{-8}	3.0008
hu	25	0.4101	-	5.3311×10^{-2}	-	5.1219×10^{-2}	-
	50	0.1980	1.0504	5.1953×10^{-3}	3.3592	8.5791×10^{-5}	9.2216
	100	0.1008	0.9745	1.4146×10^{-3}	1.8768	1.1960×10^{-5}	2.8425
	200	0.0509	0.9857	3.7772×10^{-4}	1.9050	1.7950×10^{-6}	2.7362
	400	0.0256	0.9928	9.6943×10^{-5}	1.9621	2.3099×10^{-7}	2.9581

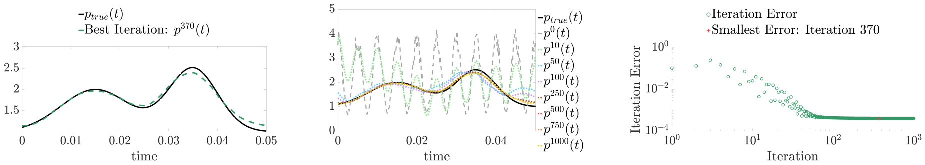


Fig. 13. *Results for case* (4.1d) *under consistent conditions. Left: plots of the true p and the p corresponding to the smallest residue error, at iteration* 370*; Middle: plots of the true p, the noisy initial guess, and various iteration values for p; Right: iteration errors on a log-log scale.*

Table 4

The regularization parameters tested for case (4.1d).

Case	γ_L	γ_H
(4.5a)	0	5×10^{-8}
(4.5b)	1×10^{-6}	0
(4.5c)	0	0

4.5. Impact of regularization terms. In this subsection, we demonstrate the necessity of the regularization terms discussed in section 3.3.2. First, we demonstrate what happens when one or more of the regularization terms are completely removed from the algorithm. Table 4 provides the regularization parameters used for the three test cases examined which are visualized in Figures 14–16.

Results for case (4.5a), representing the case with no L^1 regularization, are displayed in Figure 14. While in this case, two peaks of differing heights are recovered, the recovered peak heights are lower than the true peak locations and occur slightly earlier in time. Furthermore, the iteration with the smallest error (iteration $1,000$) occurs in case (4.5a) much later than in case (4.1d) (iteration 151). Case (4.5b) represents the case with no H^1 regularization. The corresponding results can be found in Figure 15. Not only does the lack of H^1 regularization result in a noisy solution, but also the two peaks are not even recovered after $1,000$ iterations, demonstrating slowed convergence of the iterative scheme. The situation in which neither L^1 nor H^1 regularization is implemented, case (4.5c), is shown in Figure 16. Results in this case are nearly identical to those found in case (4.5b) and Figure 15.

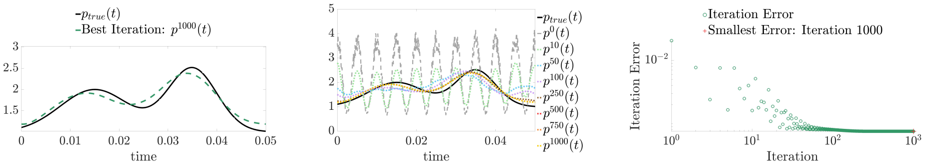


FIG. 14. *Results for case* (4.5a). *Left: plots of the true p and the numerically recovered p at iteration* 1,000 *corresponding to the smallest residue; Middle: plots of the true p, the noisy initial guess, and various iteration values for p; Right: iteration errors on a log-log scale.*

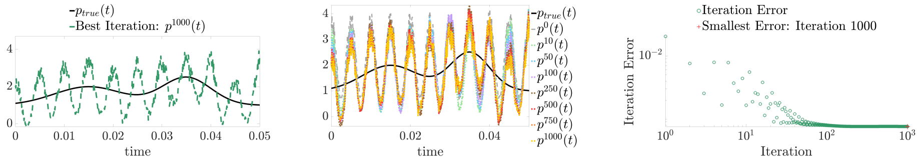


FIG. 15. *Results for case* (4.5b). *Left: plots of the true p and the numerically recovered p at iteration* 1,000 *corresponding to the smallest residue; Middle: plots of the true p, the noisy initial guess, and various iteration values for p; Right: iteration errors on a log-log scale.*

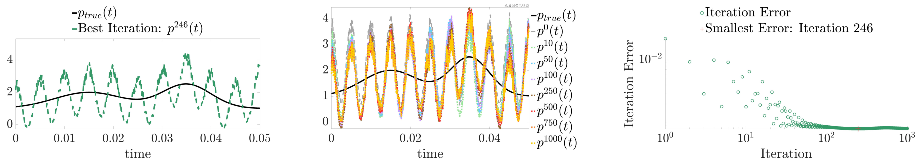


FIG. 16. *Results for case* (4.5c). *Left: plots of the true p and the numerically recovered p at iteration* 246 *corresponding to the smallest residue; Middle: plots of the true p, the noisy initial guess, and various iteration values for p; Right: iteration errors on a log-log scale.*

The second way we demonstrate the impact of regularization is by way of a modification to the minimization problem presented in (3.3). The modification involves an additional parameter and takes on the following form:

$$\text{minimize } \boldsymbol{J}(p) := \boldsymbol{J}_0(p) + \hat{\gamma}\boldsymbol{R}(p) \quad \text{subject to (2.2).} \tag{4.4}$$

This formulation allows us to perform an L-curve test [26] when running an example for multiple values of $\hat{\gamma}$. The L-curve test is a visualization tool used for examining the impact of regularization and for finding a balance with the residual errors. Here we consider the same problem setup as in case (4.1d) and set the regularization parameter to $\hat{\gamma} = 10^i$ for the integer i ranging between -5 and 5. Note that the choice for γ_L and γ_H stay fixed at 1×10^{-6} and 5×10^{-8}, respectively, for each choice of $\hat{\gamma}$. Figure 17 shows the smallest residual error over all iterations on the x-axis and the magnitude of the regularizer from the same iteration on the y-axis, both on a log-scale. The values of $\hat{\gamma}$ occurring near the elbow of the L-curve correspond to $\hat{\gamma} = 1, 10, 100, 1000$, and are reasonable choices of the regularization parameter to use.

4.6. Recovering $p(t)$ with spatial discontinuities in the solutions of the forward problem. In the previous two subsections, the final stopping time is chosen to ensure that the solutions are smooth over the entire computational region. One well-known fact of hyperbolic conservation laws is that discontinuities may appear even when the initial conditions are smooth. In this subsection, we explore the

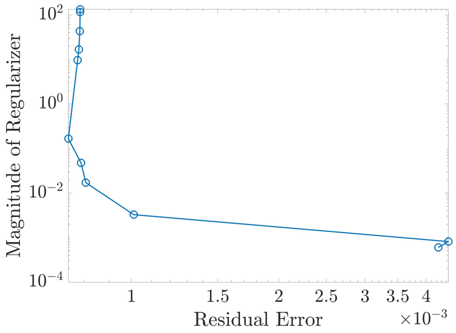


FIG. 17. *L-curve corresponding to case* (4.1d) *for regularization parameter* $\hat{\gamma} = 10^i$, $i \in [-5, 5]$.

TABLE 5

The true function for $p(t)$ *denoted as* p_{true} *and the corresponding initial guess used,* p^0 *with* $\beta = -700$. *Multiplicative noise is applied to* p^0 *in the simulations.*

Case	$p_{true}(t)$	$p^0(t)$
(4.6a)	$e^{\beta\left(t-\frac{1}{2}T\right)^2} + 1$	1
(4.6b)	$e^{4\beta\left(t-\frac{1}{4}T\right)^2} + \frac{3}{2}e^{4\beta\left(t-\frac{1}{2}T\right)^2} - \frac{1}{2}e^{4\beta\left(t-\frac{3}{4}T\right)^2} + 1$	1

capability of the proposed inverse algorithms when the solutions contain spatial discontinuities. We use the same initial conditions as in (4.1) and spatial components of the bottom topography functions as in (4.2). We run the forward problem for a longer time so that discontinuities form in the solutions. Here, we set the final time as $T = 0.2$. We consider two choices for $p_{true}(t)$ in this subsection, which are listed in Table 5 along with their corresponding initial guesses.

To capture the discontinuities well and remove the possible oscillations, a slope limiter is often employed in the DG method. We implement two different slope limiters for generating the measured data, as well as for solving the forward problem in the inverse scheme. The simple minmod limiter [7] is employed along the characteristic direction to generate the measured data. On the other hand, the WENO limiter, introduced by Qiu and Shu in [54] is used for the forward solver within the iterative inverse scheme. This limiter is known to be robust and it is able to capture the sharp transition of the discontinuities. The implementation of two different slope limiters is in an effort to avoid inverse crime.

A learning rate of $\ell = 0.02$ was implemented with the regularization parameters $\gamma_L = 1 \times 10^{-4}$ and $\gamma_H = 1 \times 10^{-6}$. Results for case (4.6a) are found in Figures 18 and 19 while Figures 20 and 21 contain the results for test (4.6b). Note that periodic boundary conditions are employed, therefore, the discontinuities can pass the right boundary and reenter the domain through the left boundary at some time between $3T/4$ and T, which means the measured data include the information of discontinuities. In both cases, we observe that our algorithm can recover the exact function $p_{true}(t)$ well, and the results are comparable with those containing smooth data only. We have included the comparison between the measured data of the water surface height, bottom topography function, water discharge, and the corresponding numerical solutions at the iteration with the smallest residue, at different times $t = \frac{T}{4}, \frac{T}{2}, \frac{3T}{4}$, and T, from which we can observe the numerical solutions match the measured data

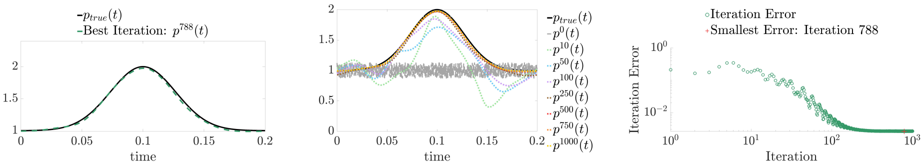


FIG. 18. *Results for case* (4.6a). *Left: plots of the true p and the p corresponding to the smallest residue error, at iteration* 788*; Middle: plots of the true p, the noisy initial guess, and various iteration values for p; Right: iteration errors on a log-log scale.*

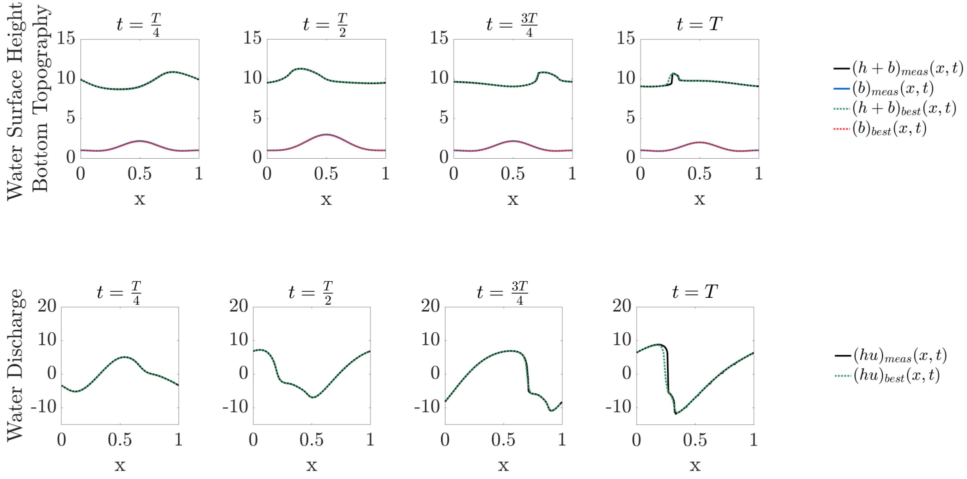


FIG. 19. *Comparison between the measured forward solutions and the numerical results from the iteration with the smallest residue error for case* (4.6a). *The results shown are for* 4 *different time snapshots. The bottom topography function, b, water surface height, h + b (top row), and the water discharge, hu (bottom row), are compared.*

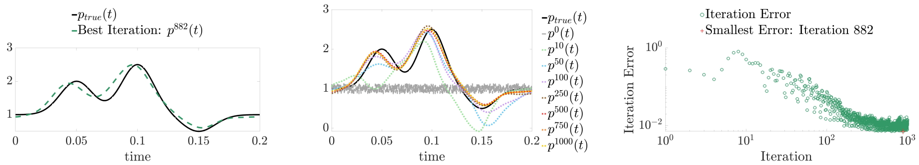


FIG. 20. *Results for case* (4.6b). *Left: plots of the true p and the p corresponding to the smallest residue error, at iteration* 882*; Middle: plots of the true p, the noisy initial guess, and various iteration values for p; Right: iteration errors on a log-log scale.*

well. This elucidates that the numerical scheme developed for the inverse problems can recover the true $p(t)$ well even when discontinuities develop in the solutions of the forward problem and hence in the measured data.

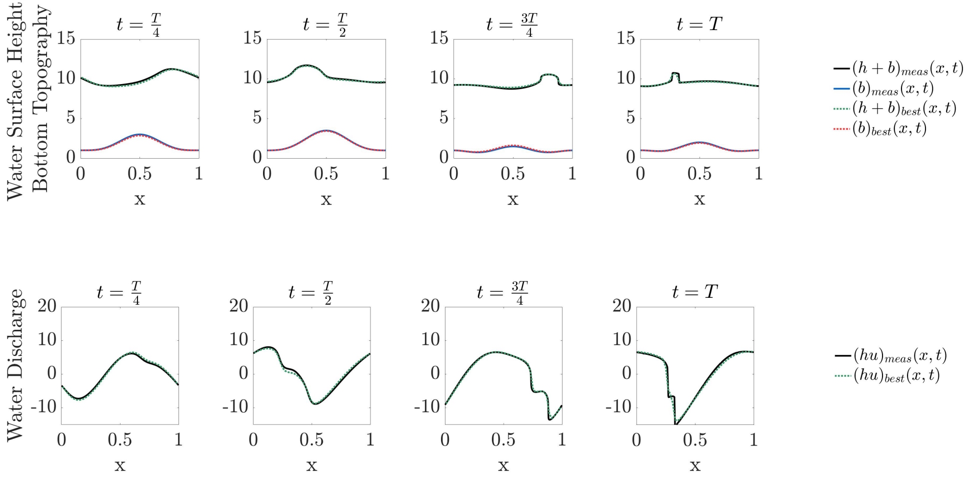


FIG. 21. *Comparison between the measured forward solutions and the numerical results from the iteration with the smallest residue error for case* (4.6b). *The results shown are for* 4 *different time snapshots. The bottom topography function, b, water surface height, $h+b$ (top row), and the water discharge, hu (bottom row), are compared.*

5. Conclusion. In this paper we constructed and validated an adjoint-based approach for recovering the bottom topography function in the source term of the one-dimensional SWEs, from the noisy measurement data at two boundaries of the domain. One novelty of this work is that the reconstruction of the bottom topography function is accomplished with only boundary data from a single measurement event. The adjoint scheme was determined by a linearization of the forward system, and has been derived for general hyperbolic balance laws. Another contribution of this work is the inclusion of two regularization terms. These extra regularization terms in the numerical approach aided in convexifying and handling the ill-posedness of the problem. The bottom topography function was recovered through an iterative process using a three-operator splitting descent method. Extensive numerical tests were carried out, which demonstrated that a variety of shapes for the true $p(t)$ function could be recovered regardless of the noisy initial guess. As a followup, we would like to develop an inverse algorithm to recover the general bottom topography $B(x,t)$ and its extension in higher dimensional problems. It is also worthwhile to extend this framework to study the inverse problems associated with other hyperbolic balance laws.

Source code. Source code available at
https://github.com/JoleneBritton/AdjointRecoveryOfBottomTopographyFromSWEs.

REFERENCES

[1] M. K. Banda and M. Herty, *Adjoint IMEX-based schemes for control problems governed by hyperbolic conservation laws*, Comput. Optim. Appl., 51 (2012), pp. 909–930, https://doi.org/10.1007/s10589-010-9362-2.

[2] A. Beck, *First-Order Methods in Optimization*, MOS-SIAM Ser. Optim. 25, SIAM, Philadelphia, 2017, https://doi.org/10.1137/1.9781611974997.

[3] S. B. Brunnermeier and S. A. Martin, *Interoperability costs in the US automotive supply chain*, Supply Chain Manag., 7 (2002), pp. 71–82, https://doi.org/10.1108/13598540210425821.

[4] R. Bürger, A. Coronel, and M. Sepúlveda, *Numerical solution of an inverse problem for a scalar conservation law modelling sedimentation*, in Hyerbolic Problems, American Mathematical Society, Providence, RI, 2009, pp. 445–454, https://doi.org/10.1090/psapm/067.2/2605240.

[5] W. Castaings, D. Dartus, M. Honnorat, F. X. Le Dimet, Y. Loukili, and J. Monnier, *Automatic differentiation: A tool for variational data assimilation and adjoint sensitivity analysis for flood modeling*, in Automtic Differentiation: Applications, Theory, and Implementations, Lect. Notes Comput. Sci. Eng., 50, Springer, Berlin, 2006, pp. 249–262, https://doi.org/10.1007/3-540-28438-9_22.

[6] A. Cauchy, *Methode generale pour la resolution des systemes d'equations simultanees*, C.R. Acad. Sci. Paris, 25 (1847), pp. 536–538.

[7] B. Cockburn and C.-W. Shu, *TVB Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws* II: *General framework*, Math. Comp., 52 (1989), pp. 411–435, https://doi.org/10.2307/2008474.

[8] S. S. Collis, K. Ghayour, M. Heinkenschloss, M. Ulbrich, and S. Ulbrich, *Numerical solution of optimal control problems governed by the compressible Navier-Stokes equations*, in Optimal Control of Complex Structures, Birkhäuser Basel, Basel, Switzerland, 2001, pp. 43–55, https://doi.org/10.1007/978-3-0348-8148-7_4.

[9] J. Darbon, *On convex finite-dimensional variational methods in imaging sciences and Hamilton–Jacobi equations*, SIAM J. Imaging Sci., 8 (2015), pp. 2268–2293, https://doi.org/10.1137/130944163.

[10] J. Darbon and S. Osher, *Algorithms for overcoming the curse of dimensionality for certain Hamilton–Jacobi equations arising in control theory and elsewhere*, Res. Math. Sci., 3 (2016), 19, https://doi.org/10.1186/s40687-016-0068-7.

[11] D. Davis and W. Yin, *A three-operator splitting scheme and its optimization applications*, Set-Valued Var. Anal., 25 (2017), pp. 829–858, https://doi.org/10.1007/s11228-017-0421-z.

[12] P. R. Eiseman and A. P. Stone, *Conservation laws of fluid dynamics–A survey*, SIAM Rev., 22 (1980), pp. 12–27, https://doi.org/10.1137/1022002.

[13] M. Ferlauto, *A pseudo-compressibility method for solving inverse problems based on the* 3*D incompressible Euler equations*, Inverse Probl. Sci. Eng., 23 (2015), pp. 798–817, https://doi.org/10.1080/17415977.2014.939653.

[14] S. Fischer and I. Steinwart, *Sobolev norm learning rates for regularized least-squares algorithm*, J. Mach. Learn. Res., 21 (2020), pp. 1–38.

[15] M. Garavello and B. Piccoli, *Traffic Flow on Networks: Conservation Laws Model*, American Institute of Mathematical Sciences, Springfeild, MO, 2006.

[16] A. F. Gessese, M. Sellier, E. Van Houten, and G. Smart, *Reconstruction of river bed topography from free surface data using a direct numerical approach in one-dimensional shallow water flow*, Inverse Problems, 27 (2011), 025001, https://doi.org/10.1088/0266-5611/27/2/025001.

[17] R. Giering and T. Kaminski, *Recipes for adjoint code construction*, ACM Trans. Math. Software, 24 (1998), pp. 437–474, https://doi.org/10.1145/293686.293695.

[18] M. Giles and S. Ulbrich, *Convergence of linearized and adjoint approximations for discontinuous solutions of conservation laws. Part 2: Adjoint approximations and extensions*, SIAM J. Numer. Anal., 48 (2010), pp. 905–921, https://doi.org/10.1137/09078078X.

[19] M. B. Giles and N. A. Pierce, *An introduction to the adjoint approach to design*, Flow, Turbul. Combust., 65 (2000), pp. 393–415, https://doi.org/10.1023/A:1011430410075.

[20] O. Glass and S. Guerrero, *On the uniform controllability of the Burgers equation*, SIAM J. Control Optim., 46 (2007), pp. 1211–1238, https://doi.org/10.1137/060664677.

[21] G. Gomes and R. Horowitz, *Optimal freeway ramp metering using the asymmetric cell transmission model*, Transp. Res. Part C, 14 (2006), pp. 244–262, https://doi.org/10.1016/j.trc.2006.08.001.

[22] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, MIT Press, Cambridge, MA, 2016.

[23] S. Gottlieb, C.-W. Shu, and E. Tadmor, *Strong stability-preserving high-order time discretization methods*, SIAM Rev., 43 (2001), pp. 89–112, https://doi.org/10.1137/S003614450036757X.

[24] M. Gugat, M. Dick, and G. Leugering, *Gas flow in fan-shaped networks: Classical solutions and feedback stabilization*, SIAM J. Control Optim., 49 (2011), pp. 2101–2117, https://doi.org/10.1137/100799824.

[25] M. Gugat, M. Herty, A. Klar, and G. Leugering, *Optimal control for traffic flow networks*, J. Optim. Theory Appl., 126 (2005), pp. 589–616, https://doi.org/10.1007/s10957-005-5499-z.

[26] P. C. Hansen, *The l-curve and Its Use in the Numerical Treatment of Inverse Problems*, in Computational Inverse Problems in Electrocardiology, P. Johnston, ed., Adv. Comput. Bioeng., WIT Press, Southampton, England, 2000, pp. 119–142.

[27] C. Heining and N. Aksel, *Bottom reconstruction in thin-film flow over topography: Steady solution and linear stability*, Phys. Fluids, 21 (2009), https://doi.org/10.1063/1.3211289.

[28] M. Heinkenschloss, *Formulation and analysis of a sequential quadratic programming method for the optimal Dirichlet boundary control of Navier-Stokes flow*, in Optimal Control: Theory, Algorithms, and Applications, Springer, Boston, 1998, pp. 178–203, https://doi.org/10.1007/978-1-4757-6095-8_9.

[29] P. Heinrich, *Nonlinear water waves generated by submarine and aerial landslides*, J. Waterway, Port, Coastal Ocean Eng., 118 (1992), pp. 249–266, https://doi.org/10.1061/(ASCE)0733-950X(1992)118:3(249).

[30] M. Hintermüller, *Semismooth Newton Methods and Applications*, Technical report, Department of Mathematics, Humboldt-University of Berlin, 2010.

[31] H. Holden, F. S. Priuli, and N. H. Risebro, *On an inverse problem for scalar conservation laws*, Inverse Problems, 30 (2014), 035015, https://doi.org/10.1088/0266-5611/30/3/035015.

[32] M. Honnorat, J. Marin, J. Monnier, and X. Lai, *Dassflow v*1.0*: A Variational Data Assimilation Software for* 2*D River Flows*, Technical report, https://hal.inria.fr/inria-00137447v2 (2007).

[33] P. Hu, Z. Cao, G. Pender, and G. Tan, *Numerical modelling of turbidity currents in the Xiaolangdi reservoir, Yellow River, China*, J. Hydrol., 464-465 (2012), pp. 41–53, https://doi.org/10.1016/j.jhydrol.2012.06.032.

[34] D. Jacquet, C. C. De Wit, and D. Koenig, *Traffic control and monitoring with a macroscopic model in the presence of strong congestion waves*, in Proceedings of the 44th IEEE Conference on Decision and Control, and the European Control Conference, CDC-ECC '05, IEEE, Piscataway, NJ, 2005, pp. 2164–2169, https://doi.org/10.1109/CDC.2005.1582482.

[35] D. Jacquet, M. Krstic, and C. C. De Wit, *Optimal control of scalar one-dimensional conservation laws*, in Proceedings of the American Control Conference, IEEE, Piscataway, NJ, 2006, pp. 5213–5218, https://doi.org/10.1109/acc.2006.1657550.

[36] D. Jacquet, C. C. D. Wit, and D. Koenig, *Optimal ramp metering strategy with extended LWR model, analysis and computational methods*, IFAC Proc., 38 (2005), pp. 99–104, https://doi.org/10.3182/20050703-6-CZ-1902.00877.

[37] A. Jameson, *Aerodynamic design via control theory*, J. Sci. Comput., 3 (1988), pp. 233–260, https://doi.org/10.1007/BF01061285.

[38] A. Jameson, *Optimum aerodynamic design using CFD and control theory*, in Proceedings of the 12th Computational Fluid Dynamics Conference, American Institute of Aeronautics and Astronautics, Reston, VA, 1995, pp. 926–949, https://doi.org/10.2514/6.1995-1729.

[39] H. Kang and K. Tanuma, *Inverse problems for scalar conservation laws*, Inverse Probl., 21 (2005), pp. 1047–1060, https://doi.org/10.1088/0266-5611/21/3/015.

[40] S. Kindermann, S. Osher, and P. W. Jones, *Deblurring and denoising of images by nonlocal functionals*, Multiscale Model. Simul., 4 (2005), pp. 1091–1115, https://doi.org/10.1137/050622249.

[41] A. Kotsialos and M. Papageorgiou, *Nonlinear optimal control applied to coordinated ramp metering*, IEEE Trans. Control System Technol., 12 (2004), pp. 920–933, https://doi.org/10.1109/TCST.2004.833406.

[42] A. Kurganov, *Finite-volume schemes for shallow-water equations*, Acta Numerica, 27 (2018), pp. 289–351, https://doi.org/10.1017/S0962492918000028.

[43] J. Larson, M. Menickelly, and S. M. Wild, *Derivative-free optimization methods*, Acta Numer., 28 (2019), pp. 287–404, https://doi.org/10.1017/S0962492919000060.

[44] J. M. Lellouche, J. L. Devenon, and I. Dekeyser, *Boundary control of Burgers' equation-a numerical approach*, Comput. Math. Appl., 28 (1994), pp. 33–44, https://doi.org/10.1016/0898-1221(94)00138-3.

[45] S. Li and C. J. Duffy, *Fully coupled approach to modeling shallow water flow, sediment transport, and bed evolution in rivers*, Water Res. Res., 47 (2011), https://doi.org/10.1029/2010WR009751.

[46] J.-L. Lions, *Optimal Control of Systems Governed by Partial Differential Equations*, Springer, Berlin, 1971.

[47] P. Moin and T. Bewley, *Feedback control of turbulence*, Appl. Mech. Rev., 47 (1994), pp. S3–S13, https://doi.org/10.1115/1.3124438.

[48] J. Monnier, F. Couderc, D. Dartus, K. Larnier, R. Madec, and J. P. Vila, *Inverse algorithms for* 2*D shallow water equations in presence of wet dry fronts: Application to*

flood plain dynamics, Adv. Water Res., 97 (2016), pp. 11–24, https://doi.org/10.1016/j.advwatres.2016.07.005.

[49] G. I. Montecinos, J. C. López-Ríos, J. H. Ortega, and R. Lecarios, *A numerical procedure and unified formulation for the adjoint approach in hyperbolic PDE-constrained optimization problems*, IMA J. Appl. Math. 84 (2019), pp. 483–516.

[50] J.-D. Müller and P. Cusdin, *On the performance of discrete adjoint CFD codes using automatic differentiation*, Internat. J. Numer. Methods Fluids, 47 (2005), pp. 939–945, https://doi.org/10.1002/fld.885.

[51] M. K. Ng, R. H. Chan, T. F. Chan, and A. M. Yip, *Cosine transform preconditioners for high resolution image reconstruction*, Linear Algebra Appl., 316 (2000), pp. 89–104, https://doi.org/10.1016/S0024-3795(99)00274-8.

[52] N. Parikh, *Proximal Algorithms*, Found. Trends Optim., 1 (2014), pp. 127–239, https://doi.org/10.1561/2400000003.

[53] O. Pironneau, *On optimum design in fluid mechanics*, J. Fluid Mech., 64 (1974), pp. 97–110, https://doi.org/10.1017/S0022112074002023.

[54] J. Qiu and C.-W. Shu, *Runge-Kutta discontinuous Galerkin method using WENO limiters*, SIAM J. Sci. Comput., 26 (2005), pp. 907–929, https://doi.org/10.1137/S1064827503425298.

[55] R. L. Raffard, K. Amonlirdviman, J. D. Axelrod, and C. J. Tomlin, *An adjoint-based parameter identification algorithm applied to planar cell polarity signaling*, IEEE Trans. Automat. Control, 53 (2008), pp. 109–121, https://doi.org/10.1109/TAC.2007.911362.

[56] J. Reilly, S. Samaranayake, M. L. Delle Monache, W. Krichene, P. Goatin, and A. M. Bayen, *Adjoint-based optimization on a network of discretized scalar conservation laws with applications to coordinated ramp metering*, J. Optim. Theory Appl., 167 (2015), pp. 733–760, https://doi.org/10.1007/s10957-015-0749-1.

[57] J. Reuther, A. Jameson, J. Farmer, L. Marlinelli, and D. Saunders, *Aerodynamic Shape Optimization of Complex Aircraft Configurations Via an Adjoint Formulation*, in Proceedings of the 34th Aerospace Sciences Meeting and Exhibit, American Institute of Aeronautics and Astronautics, Reston, VA, 1996, 96-or94, https://doi.org/10.2514/6.1996-94.

[58] L. I. Rudin, S. Osher, and E. Fatemi, *Nonlinear total variation based noise removal algorithms*, Phys. D, 60 (1992), pp. 259–268, https://doi.org/10.1016/0167-2789(92)90242-F.

[59] J. F. Sleath, *Sediment transport by waves and currents*, J. Geophys. Res., Oceans, 100 (1995), pp. 10977–10986, https://doi.org/10.1029/95jc00140.

[60] W. Sun and Y.-X. Yuan, *Optimization Theory and Methods: Nonlinear Programming*, Springer, New York, 2006.

[61] R. Vazquez, J. M. Coron, M. Krstic, and G. Bastin, *Local exponential H^2 stabilization of a 2×2 quasilinear hyperbolic system using backstepping*, in Proceedings of the IEEE Conference on Decision and Control, IEEE, Piscataway, NJ, 2011, pp. 1329–1334, https://doi.org/10.1109/CDC.2011.6161075.

[62] A. Wirgin, *The Inverse Crime*, http://arxiv.org/abs/math-ph/0401050 (2004).

[63] Y. Xing, *Numerical methods for the nonlinear shallow water equations*, in Handb. Numer. Anal., Vol. 18, Elsevier, Amsterdam, 2017, pp. 361–384, https://doi.org/10.1016/bs.hna.2016.09.003.

[64] Y. Xing and C. W. Shu, *High order well-balanced WENO scheme for the gas dynamics equations under gravitational fields*, J. Sci. Comput., 54 (2013), pp. 645–662, https://doi.org/10.1007/s10915-012-9585-8.

[65] W. Yin, S. Osher, D. Goldfarb, and J. Darbon, *Bregman iterative algorithms for ℓ_1-minimization with applications to compressed sensing*, SIAM J. Imaging Sci., 1 (2008), pp. 143–168, https://doi.org/10.1137/070703983.

[66] T. Young, XIII*, Hydraulic investigations, subservient to an intended Croonian lecture on the motion of the blood*, Philos. Trans. Roy. Soc. Lond., 98 (1808), pp. 164–186, https://doi.org/10.1098/rstl.1808.0014.

[67] A. K. Ziliaskopoulos, *A linear programming model for the single destination system optimum dynamic traffic assignment problem*, Transp. Sci., 34 (2000), pp. 37–49, https://doi.org/10.1287/trsc.34.1.37.12281.