symmetry

MDPI

Article

Numerical Solution of Direct and Inverse Problems for Time-Dependent Volterra Integro-Differential Equation Using Finite Integration Method with Shifted Chebyshev Polynomials

Ratinan Boonklurb *, Ampol Duangpan and Phansphitcha Gugaew

Department of Mathematics and Computer Science, Faculty of Science, Chulalongkorn University, Bangkok 10330, Thailand; ty_math@hotmail.com (A.D.); poohdd28@hotmail.com (P.G.)

* Correspondence: ratinan.b@chula.ac.th

Received: 3 February 2020; Accepted: 5 March 2020; Published: 30 March 2020

Abstract: In this article, the direct and inverse problems for the one-dimensional time-dependent Volterra integro-differential equation involving two integration terms of the unknown function (i.e., with respect to time and space) are considered. In order to acquire accurate numerical results, we apply the finite integration method based on shifted Chebyshev polynomials (FIM-SCP) to handle the spatial variable. These shifted Chebyshev polynomials are symmetric (either with respect to the point $x = \frac{L}{2}$ or the vertical line $x = \frac{L}{2}$ depending on their degree) over $[0, L]$, and their zeros in the interval are distributed symmetrically. We use these zeros to construct the main tool of FIM-SCP: the Chebyshev integration matrix. The forward difference quotient is used to deal with the temporal variable. Then, we obtain efficient numerical algorithms for solving both the direct and inverse problems. However, the ill-posedness of the inverse problem causes instability in the solution and, so, the Tikhonov regularization method is utilized to stabilize the solution. Furthermore, several direct and inverse numerical experiments are illustrated. Evidently, our proposed algorithms for both the direct and inverse problems give a highly accurate result with low computational cost, due to the small number of iterations and discretization.

Keywords: finite integration method; shifted Chebyshev polynomial; direct and inverse problems; Volterra integro-differential equation; Tikhonov regularization method

MSC: 65R20; 65R32

1. Introduction

An integro-differential equation (IDE) is an equation which contains both derivatives and integrals of an unknown function. Several situations in the branches of science and engineering can be demonstrated by developing mathematical models which are often in the form of IDEs, such as in RLC circuit analysis, the activity of interacting inhibitory and excitatory neurons, the Wilson–Cowan model, and so on; see Reference [1] for more applications. In fact, many of these problems cannot be directly solved, because we may not know all necessary information or an incomplete system may be provided. This has led to the study of both direct and inverse problems for a certain type of one-dimensional IDE involving time, which is called the one-dimensional time-dependent Volterra IDE (TVIDE). Hence, in this study, we investigate the TVIDE of the following form

$$v_t(x,t) + \mathcal{L}v(x,t) = \int_0^t \kappa_1(x,\eta)v(x,\eta)d\eta + \int_0^x \kappa_2(\xi,t)v(\xi,t)d\xi + F(x,t), \quad (1)$$

for all $(x,t) \in (0,L) \times (0,T]$, where x and t represent space and time variables, respectively; $\mathcal{L}$ is the spatial linear differential operator of order n; $\kappa_1(x,t)$ and $\kappa_2(x,t)$ are the given continuously integrable kernel functions; and $v(x,t)$ is an unknown function, which is to be determined subject to prescribed initial and boundary conditions. We remark that, if a forcing term $F(x,t)$ of (1) is given, then this problem has only one unknown $v(x,t) \in C^{n,1}([0,L] \times [0,T])$ to be solved, and it is called a direct problem. In contrast, if the forcing term $F(x,t)$ is missing, then this problem has two unknowns $F(x,t) \in C([0,L] \times [0,T])$ and $v(x,t) \in C^{n,1}([0,L] \times [0,T])$ to be solved, and it is called an inverse problem. However, for the inverse problem in this paper, we specifically define the forcing term $F(x,t) := \beta(t)f(x,t)$, where $\beta(t)$ is a missing source function to be retrieved and $f(x,t)$ is the given function. We note that (1) has both $\int_0^t \kappa_1(x,\eta)v(x,\eta)d\eta$ and $\int_0^x \kappa_2(\xi,t)v(\xi,t)d\xi$, while several studies in the literature have considered similar problems containing only one of these two terms.

The Volterra IDE containing only an integration term with respect to time arises in many applications, including the compression of poro-viscoelastic media, blow-up problems, analysis of space–time-dependent nuclear reactor dynamics, and so on; see Reference [2]. The existence, uniqueness, and asymptotic behavior of its solution have been discussed in Reference [3]. There are many authors who have studied the numerical solution of this type of problem by using techniques such as the finite element method [2], finite difference method (FDM) [4], collocation methods in polynomial spline [5], the implicit Runge–Kutta–Nyström method [6], the Legendre collocation method [7], and so on.

On the other hand, the Volterra IDE containing only an integration term with respect to space has also been studied in various areas, such as for the one-dimensional viscoelastic problem and one-dimensional heat flow in materials with memory [8], modeling heat/mass diffusion processes, biological species coexisting together with increasing and decreasing rates of growth, electromagnetism, and ocean circulation, among others [9]. Moreover, the existence and uniqueness for this type of Volterra IDE were shown in Reference [8]. Consequently, abundant numerical methods have appeared for finding solutions to this type of Volterra IDE using, for example, spline collocation method [10], collocation method with implicit Runge–Kutta method [11], decomposition method [12], and so on.

However, our problem deals with a Volterra IDE involving both temporal and spatial integrations. There have been no results in the literature regarding the existence and uniqueness of solutions to this type of problem. In this paper, we concentrate on providing a decent numerical procedure to find approximate solutions for both the direct and inverse problems of the proposed TVIDE (1).

Generally, it is well-known that the classification of problems involving differential equations was defined by Hadamard [13] in 1902. Mathematical problems involving differential equations are well-posed if the following conditions hold: existence, uniqueness, and stability. Otherwise, the problem is called ill-posed; this normally occurs in the inverse problem. Even though the initial and boundary conditions are prescribed, it is not sufficient to guarantee that our inverse problem (1) has unique solutions $\beta(t)$ and $v(x,t)$. Hence, additional conditions (e.g., the observation or measurement of data) need to be involved. In practice, there are many kinds of additional conditions; for example, a fixed point of the system, an average time of the system, or an integral of the system. After the additional conditions has been added as an auxiliary condition in our inverse problem (1), we can obtain the existence and uniqueness of $\beta(t)$ and $v(x,t)$. However, the additional condition may contains measurement or observation errors, which may cause the instability in the solutions; namely, a small perturbation in the input data can produce a considerable error, especially for $\beta(t)$. Thus, some regularization techniques are required to overcome the ill-posedness and stabilize the solution.

There exist many schemes which are generally used to solve both direct and inverse problems of Volterra IDEs, such as the above-mentioned methods. However, those methods utilize the process of approximating differentiation. It is well-known that numerical differentiation is very sensitive to rounding errors, as its manipulation task involves division by a small step-size. On the other hand, the process of numerical integration involves multiplication by a small step-size and, so, it is very insensitive to rounding errors. In recent years, the finite integration method (FIM) has been developed to find approximate solutions of linear boundary value problems for partial differential

equations (PDEs). The concept of FIM is to transform a given PDE into an equivalent integral equation, following which a numerical integration method, such as the trapezoid, Simpson, or Newton–Cotes methods (see References [14–16]), are applied. In 2018, Boonklurb et al. [17] modified the traditional FIM by using Chebyshev polynomials to solve one- and two-dimensional linear PDEs and obtained a more accurate result compared to the traditional FIMs and FDM. However, their technique [17] has not yet been utilized to overcome the direct and inverse problems of TVIDE, which are the major focuses of this work.

In this paper, we formulate numerical algorithms for solving the direct and inverse problems of TVIDE (1). We manipulate the idea of FIM in Reference [17] by using shifted Chebyshev polynomials, which we call the FIM with shifted Chebyshev polynomials (FIM-SCP), to deal with the spatial variable and use the forward difference quotient to estimate the time derivative. We further apply the Tikhonov regularization method to stabilize our ill-posed problem (1). The rest of the paper is organized as follows. In Section 2, the definition and some basic properties concerning the shifted Chebyshev polynomial are given to construct the shifted Chebyshev integration matrices. The Tikhonov regularization method is also presented in Section 2. In Section 3, we use the FIM-SCP and the forward difference quotient to devise efficient numerical algorithms to find approximate solutions to the direct and inverse problems of (1). Then, we implement our proposed algorithms through several examples, in order to demonstrate their efficiency compared with their analytical solutions. Furthermore, we also display the time convergence rate and CPU time (s) in Section 4. Finally, the conclusion and some directions for future work are given in Section 5.

2. Preliminaries

In this section, we introduce some necessary tools for solving the direct and inverse problems of TVIDE (1): the FIM-SCP and the Tikhonov regularization method.

2.1. Shifted Chebyshev Integration Matrices

We first introduce the definition and some basic properties of shifted Chebyshev polynomials [18], which are used to establish the first- and higher-order shifted Chebyshev integration matrices based on the idea of constructing integration matrices in Reference [17]. However, we slightly modify this idea by instead using a shifted Chebyshev expansion suitable for solving our problem (1) without domain transformation. We give the definition and properties as follows.

Definition 1. *The shifted Chebyshev polynomial of degree $n \geq 0$ is defined by*

$$S_n(x) = \cos\left(n \arccos\left(\frac{2x}{L} - 1\right)\right) \text{ for } x \in [0, L].$$

Note that this shifted Chebyshev polynomial is symmetric, either with respect to the point $x = \frac{L}{2}$ or the vertical line $x = \frac{L}{2}$ over $[0, L]$, depending on its degree. Next, we provide some important properties of the shifted Chebyshev polynomial, which we use to constructing the shifted Chebyshev integration matrix, as follows.

Lemma 1. (i) *For $n \in \mathbb{N}$, the zeros of $S_n(x)$ are symmetrically distributed over $[0, L]$ and given by*

$$x_k = \frac{L}{2}\left(\cos\left(\frac{2k-1}{2n}\pi\right) + 1\right), \; k \in \{1, 2, 3, ..., n\}. \tag{2}$$

(ii) *For $r \in \mathbb{N}$, the r^{th}-order derivatives of $S_n(x)$ at the endpoint $b \in \{0, L\}$ are*

$$\frac{d^r}{dx^r} S_n(x)\Big|_{x=b} = \prod_{k=0}^{r-1}\left(\frac{n^2 - k^2}{2k+1}\right)\left(\frac{2b}{L} - 1\right)^{n+r}. \tag{3}$$

(iii) *For $x \in [0, L]$, the single-layer integrations of shifted Chebyshev polynomial $S_n(x)$ are*

$$\begin{aligned}
\bar{S}_0(x) &= \int_0^x S_0(\xi)\, d\xi = x, \\
\bar{S}_1(x) &= \int_0^x S_1(\xi)\, d\xi = \frac{x^2}{L} - x, \\
\bar{S}_n(x) &= \int_0^x S_n(\xi)\, d\xi = \frac{L}{4}\left(\frac{S_{n+1}(x)}{n+1} - \frac{S_{n-1}(x)}{n-1} - \frac{2(-1)^n}{n^2-1}\right),\ n \in \{2,3,4,...\}.
\end{aligned}$$

(iv) *Let $\{x_k\}_{k=1}^n$ be a set of zeros of $S_n(x)$, the shifted Chebyshev matrix $\mathbf{S}$ is defined by*

$$\mathbf{S} = \begin{bmatrix} S_0(x_1) & S_1(x_1) & \cdots & S_{n-1}(x_1) \\ S_0(x_2) & S_1(x_2) & \cdots & S_{n-1}(x_2) \\ \vdots & \vdots & \ddots & \vdots \\ S_0(x_n) & S_1(x_n) & \cdots & S_{n-1}(x_n) \end{bmatrix}.$$

Then, it has the multiplicative inverse $\mathbf{S}^{-1} = \frac{1}{n}\mathrm{diag}(1,2,2,...,2)\mathbf{S}^\top$.

Next, we use the above definition and properties of shifted Chebyshev polynomials to construct the shifted Chebyshev integration matrices. First, let N be a positive integer and L be a positive real number. Define an approximate solution $u(x)$ of a certain differential equation by a linear combination of shifted Chebyshev polynomials $S_n(x)$; that is,

$$u(x) = \sum_{n=0}^{N-1} c_n S_n(x) \text{ for } x \in [0, L]. \tag{4}$$

Let x_k for $k \in \{1,2,3,...,N\}$ be the interpolated points which are meshed by the zeros of $S_N(x)$ defined in (2). Substituting each x_k into (4), it can be expressed (in matrix form) as

$$\begin{bmatrix} u(x_1) \\ u(x_2) \\ \vdots \\ u(x_N) \end{bmatrix} = \begin{bmatrix} S_0(x_1) & S_1(x_1) & \cdots & S_{N-1}(x_1) \\ S_0(x_2) & S_1(x_2) & \cdots & S_{N-1}(x_2) \\ \vdots & \vdots & \ddots & \vdots \\ S_0(x_N) & S_1(x_N) & \cdots & S_{N-1}(x_N) \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ \vdots \\ c_{N-1}, \end{bmatrix},$$

which is denoted by $\mathbf{u} = \mathbf{S}\mathbf{c}$. The unknown coefficient vector can be performed by $\mathbf{c} = \mathbf{S}^{-1}\mathbf{u}$. Let us consider the single-layer integration of $u(x)$ from 0 to x_k, which is denoted by $U^{(1)}(x_k)$; we obtain

$$U^{(1)}(x_k) = \int_0^{x_k} u(\xi)\, d\xi = \sum_{n=0}^{N-1} c_n \int_0^{x_k} S_n(\xi)\, d\xi = \sum_{n=0}^{N-1} c_n \bar{S}_n(x_k)$$

for $k \in \{1,2,3,...,N\}$ or, in matrix form,

$$\begin{bmatrix} U^{(1)}(x_1) \\ U^{(1)}(x_2) \\ \vdots \\ U^{(1)}(x_N) \end{bmatrix} = \begin{bmatrix} \bar{S}_0(x_1) & \bar{S}_1(x_1) & \cdots & \bar{S}_{N-1}(x_1) \\ \bar{S}_0(x_2) & \bar{S}_1(x_2) & \cdots & \bar{S}_{N-1}(x_2) \\ \vdots & \vdots & \ddots & \vdots \\ \bar{S}_0(x_N) & \bar{S}_1(x_N) & \cdots & \bar{S}_{N-1}(x_N) \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ \vdots \\ c_{N-1} \end{bmatrix}.$$

We denote the above matrix by $\mathbf{U}^{(1)} = \bar{\mathbf{S}}\mathbf{c} = \bar{\mathbf{S}}\mathbf{S}^{-1}\mathbf{u} := \mathbf{A}\mathbf{u}$, where $\mathbf{A} = \bar{\mathbf{S}}\mathbf{S}^{-1} := [a_{ki}]_{N\times N}$ is called the first-order shifted Chebyshev integration matrix for the FIM-SCP; that is,

$$U^{(1)}(x_k) = \int_0^{x_k} u(\xi)\, d\xi = \sum_{i=1}^{N} a_{ki} u(x_i).$$

Next, consider the double-layer integration of $u(x)$ from 0 to x_k, which denoted by $U^{(2)}(x_k)$. We have

$$U^{(2)}(x_k) = \int_0^{x_k} \int_0^{\xi_2} u(\xi_1)\, d\xi_1 d\xi_2 = \sum_{i=1}^{N} a_{ki} \int_0^{x_i} u(\xi_1)\, d\xi_1 = \sum_{i=1}^{N} \sum_{j=1}^{N} a_{ki} a_{ij} u(x_j)$$

for $k \in \{1, 2, 3, ..., N\}$. It can be written, in matrix form, as $\mathbf{U}^{(2)} = \mathbf{A}^2\mathbf{u}$. Similarly, we can calculate the n-layer integration of $u(x)$ from 0 to x_k, which is denoted by $U^{(n)}(x_k)$. Then, we have

$$U^{(n)}(x_k) = \int_0^{x_k} \cdots \int_0^{\xi_2} u(\xi_1)\, d\xi_1 \cdots d\xi_n = \sum_{i_n=1}^{N} \cdots \sum_{j=1}^{N} a_{ki_n} \cdots a_{i_1 j} u(x_j)$$

for $k \in \{1, 2, 3, ..., N\}$, which can be expressed, in matrix form, as $\mathbf{U}^{(n)} = \mathbf{A}^n\mathbf{u}$.

2.2. Tikhonov Regularization Method

In this section, we briefly present the idea of the Tikhonov regularization method [19], which is usually applied to stabilize ill-posed problems, such as our inverse problem. Normally, the considered inverse problem can be represented by the system of m linear equations with n unknowns, as

$$\mathbf{A}\mathbf{x} = \mathbf{b}^{\epsilon}, \tag{5}$$

where $\mathbf{b}^{\epsilon}$ is the vector in the right-hand side, which is perturbed by some noise ϵ, and $\mathbf{x}$ is the solution of the system (5) after perturbation. Tikhonov regularization replaces the inverse problem (5) by a minimization problem to obtain an efficiently approximate solution, which can be described as

$$\arg\min_{\mathbf{x} \in \mathbb{R}^n} \left\{ \|\mathbf{A}\mathbf{x} - \mathbf{b}^{\epsilon}\|^2 + \lambda \|\mathbf{x}\|^2 \right\}, \tag{6}$$

where $\lambda > 0$ is a regularization parameter balancing the weighting between the two terms of the function and $\|\cdot\|$ is the standard Euclidean norm. To reformulate the above minimization problem (6), we obtain

$$\arg\min_{\mathbf{x} \in \mathbb{R}^n} \left\{ \left\| \begin{bmatrix} \mathbf{A} \\ \sqrt{\lambda}\mathbf{I} \end{bmatrix} \mathbf{x} - \begin{bmatrix} \mathbf{b}^{\epsilon} \\ \mathbf{0} \end{bmatrix} \right\|^2 \right\}.$$

Clearly, this is a linear least-square problem in $\mathbf{x}$. Then, the above problem turns out to be the normal equation of the form

$$\begin{bmatrix} \mathbf{A} \\ \sqrt{\lambda}\mathbf{I} \end{bmatrix}^{\top} \begin{bmatrix} \mathbf{A} \\ \sqrt{\lambda}\mathbf{I} \end{bmatrix} \mathbf{x} = \begin{bmatrix} \mathbf{A} \\ \sqrt{\lambda}\mathbf{I} \end{bmatrix}^{\top} \begin{bmatrix} \mathbf{b}^{\epsilon} \\ \mathbf{0} \end{bmatrix}.$$

To simplify the above equation, the solution $\mathbf{x}$ under the regularization parameter λ (denoted by $\mathbf{x}_\lambda$) can be computed by

$$\mathbf{x}_\lambda = (\mathbf{A}^{\top}\mathbf{A} + \lambda\mathbf{I})^{-1}\mathbf{A}^{\top}\mathbf{b}^{\epsilon}. \tag{7}$$

We can see that the accuracy of $\mathbf{x}_\lambda$ in (7) depends on the regularization parameter λ, which plays an important role in the calculation: A large regularization parameter may over-smoothen the solution, while a small regularization parameter may lose the ability to stabilize the solution. Therefore, a suitable choice of the regularization parameter λ is very significant for finding a stable approximate solution. There are many approaches for choosing a value of the parameter λ, such as the discrepancy principle criterion, the generalized cross-validation, the L-curve method, and so on. Nevertheless, the regularization parameter λ in this paper is chosen according to Morozov's discrepancy principle combined with Newton's method, which has been proposed in Reference [20]. We provide the procedure for calculating the optimal regularization parameter λ below, which can be carried out by the following steps:

Step 1: Set $n = 0$ and give an initial regularization parameter $\lambda_0 > 0$.

Step 2: Compute $\mathbf{x}_{\lambda_n} = (\mathbf{A}^\top \mathbf{A} + \lambda_n \mathbf{I})^{-1} \mathbf{A}^\top \mathbf{b}^\epsilon$.

Step 3: Compute $\nabla \mathbf{x}_{\lambda_n} = -(\mathbf{A}^\top \mathbf{A} + \lambda_n \mathbf{I})^{-1} \mathbf{x}_{\lambda_n}$.

Step 4: Compute $G(\lambda_n) = \|\mathbf{A}\mathbf{x}_{\lambda_n} - \mathbf{b}^\epsilon\|^2 - \epsilon^2$.

Step 5: Compute $G'(\lambda_n) = 2\lambda_n \|\mathbf{A}\nabla \mathbf{x}_{\lambda_n}\|^2 + 2\lambda_n^2 \|\nabla \mathbf{x}_{\lambda_n}\|^2$.

Step 6: Compute $\lambda_{n+1} = \lambda_n - \frac{G(\lambda_n)}{G'(\lambda_n)}$.

Step 7: If $\|\lambda_{n+1} - \lambda_n\| < \delta$ for a tolerance δ, end. Else, set $n = n + 1$ and return to Step 2.

Therefore, we receive the optimal regularization parameter λ, which is the terminal value λ_n obtained from the above procedure. When the regularization parameter λ is fixed as the mentioned optimal value, we can directly obtain the corresponding regularized solution by (7).

3. Numerical Algorithms for Direct and Inverse Problems of TVIDE

In this section, we apply the FIM-SCP described in Section 2.1 to devise the numerical algorithms for solving both the direct and inverse TVIDE problems (1), in order to obtain accurate approximate results. Let u be an approximate solution of v in (1). Then, we have the following linear TVIDE over the domain $\Omega = (0, L) \times (0, T]$:

$$u_t(x,t) + \mathcal{L}u(x,t) = \int_0^t \kappa_1(x,\eta)u(x,\eta)d\eta + \int_0^x \kappa_2(\xi,t)u(\xi,t)d\xi + F(x,t), \tag{8}$$

subject to the initial condition

$$u(x,0) = \phi(x), \quad x \in [0,L], \tag{9}$$

and the boundary conditions

$$u^{(r)}(b,t) = \psi_r(t), \quad t \in [0,T], \tag{10}$$

for $b \in \{0, L\}$ and $r \in \{0, 1, 2, ..., n-1\}$, where t and x represent time and space variables, respectively. Additionally, κ_1, κ_2, F, ϕ, and ψ_r are given continuous functions and $\mathcal{L}$ is the spatial linear differential operator of order n defined by $\mathcal{L} := \sum_{i=0}^{n} p_i(x,t) \frac{d^i}{dx^i}$, where $p_i(x,t)$ is given and sufficiently smooth.

3.1. Procedure for Solving the Direct TVIDE Problem

First, we linearize (8) by uniformly discretizing the temporal domain into M subintervals with time step τ. Then, we specify (8) at a time $t_m = m\tau$ for $m \in \mathbb{N}$ and use the first-order forward difference quotient to estimate the time derivative term u_t. Next, we replace each x by x_k for $k \in \{1, 2, 3, ..., N\}$ as generated by the zeros of the shifted Chebyshev polynomial $S_N(x)$ defined in (2). Thus, we have

$$\frac{u^{\langle m \rangle} - u^{\langle m-1 \rangle}}{\tau} + \mathcal{L}u^{\langle m \rangle} = \int_0^{t_m} \kappa_1(x_k,\eta)u(x_k,\eta)d\eta + \int_0^{x_k} \kappa_2(\xi,t_m)u(\xi,t_m)d\xi + F^{\langle m \rangle}, \tag{11}$$

where $u^{\langle m \rangle} = u^{\langle m \rangle}(x_k) = u(x_k, t_m)$ and $F^{\langle m \rangle} = F^{\langle m \rangle}(x_k) = F(x_k, t_m)$. Next, consider the first integral term with respect to time by letting it be $J_1^{\langle m \rangle}(x_k)$, we approximate $J_1^{\langle m \rangle}(x_k)$ by using the trapezoidal rule. Thus, we approximate $J_1^{\langle m \rangle}(x_k)$ as

$$\begin{aligned} J_1^{\langle m\rangle}(x_k) &:= \int_0^{t_m} \kappa_1(x_k,\eta)u(x_k,\eta)d\eta \\ &= \sum_{i=0}^{m-1}\int_{t_i}^{t_{i+1}} \kappa_1(x_k,\eta)u(x_k,\eta)d\eta \\ &\approx \sum_{i=0}^{m-1}\frac{\tau}{2}\left(\kappa_1^{\langle i\rangle}(x_k)u^{\langle i\rangle}(x_k)+\kappa_1^{\langle i+1\rangle}(x_k)u^{\langle i+1\rangle}(x_k)\right) \\ &= \frac{\tau}{2}\kappa_1^{\langle 0\rangle}(x_k)u^{\langle 0\rangle}(x_k)+\tau\sum_{i=1}^{m-1}\kappa_1^{\langle i\rangle}(x_k)u^{\langle i\rangle}(x_k)+\frac{\tau}{2}\kappa_1^{\langle m\rangle}(x_k)u^{\langle m\rangle}(x_k) \end{aligned}$$

for each $x_k \in \{x_1, x_2, x_3, ..., x_N\}$. The above equation can be written, in matrix form, as

$$\mathbf{J}_1^{\langle m\rangle} = \frac{\tau}{2}\mathbf{K}_1^{\langle 0\rangle}\mathbf{u}^{\langle 0\rangle} + \tau\sum_{i=1}^{m-1}\mathbf{K}_1^{\langle i\rangle}\mathbf{u}^{\langle i\rangle} + \frac{\tau}{2}\mathbf{K}_1^{\langle m\rangle}\mathbf{u}^{\langle m\rangle}, \tag{12}$$

where each parameter in (12) can be defined as follows:

$$\begin{aligned} \mathbf{J}_1^{\langle m\rangle} &= \left[J_1^{\langle m\rangle}(x_1), J_1^{\langle m\rangle}(x_2), J_1^{\langle m\rangle}(x_3), ..., J_1^{\langle m\rangle}(x_N)\right]^\top, \\ \mathbf{u}^{\langle i\rangle} &= \left[u^{\langle i\rangle}(x_1), u^{\langle i\rangle}(x_2), u^{\langle i\rangle}(x_3), ..., u^{\langle i\rangle}(x_N)\right]^\top, \\ \mathbf{K}_1^{\langle i\rangle} &= \text{diag}\left(\kappa_1^{\langle i\rangle}(x_1), \kappa_1^{\langle i\rangle}(x_2), \kappa_1^{\langle i\rangle}(x_3), ..., \kappa_1^{\langle i\rangle}(x_N)\right). \end{aligned}$$

Then, we consider the second integral term with respect to space by letting it be $J_2^{\langle m\rangle}(x_k)$ and using the idea of FIM-SCP (as described in Section 2.1) to approximate it. Then, we obtain

$$J_2^{\langle m\rangle}(x_k) := \int_0^{x_k}\kappa_2(\xi,t_m)u(\xi,t_m)d\xi = \int_0^{x_k}\kappa_2^{\langle m\rangle}(\xi)u^{\langle m\rangle}(\xi)d\xi \approx \sum_{i=1}^{N}a_{ki}\kappa_2^{\langle m\rangle}(x_i)u^{\langle m\rangle}(x_i)$$

for each $x_k \in \{x_1, x_2, x_3, ..., x_N\}$. The above equation can be written, in matrix form, as

$$\mathbf{J}_2^{\langle m\rangle} = \mathbf{A}\mathbf{K}_2^{\langle m\rangle}\mathbf{u}^{\langle m\rangle}, \tag{13}$$

where $\mathbf{A} = \bar{\mathbf{S}}\mathbf{S}^{-1}$ is the shifted Chebyshev integration matrix defined in Section 2.1,

$$\begin{aligned} \mathbf{J}_2^{\langle m\rangle} &= \left[J_2^{\langle m\rangle}(x_1), J_2^{\langle m\rangle}(x_2), J_2^{\langle m\rangle}(x_3), ..., J_2^{\langle m\rangle}(x_N)\right]^\top, \\ \mathbf{u}^{\langle m\rangle} &= \left[u^{\langle m\rangle}(x_1), u^{\langle m\rangle}(x_2), u^{\langle m\rangle}(x_3), ..., u^{\langle m\rangle}(x_N)\right]^\top, \\ \mathbf{K}_2^{\langle m\rangle} &= \text{diag}\left(\kappa_2^{\langle m\rangle}(x_1), \kappa_2^{\langle m\rangle}(x_2), \kappa_2^{\langle m\rangle}(x_3), ..., \kappa_2^{\langle m\rangle}(x_N)\right). \end{aligned}$$

Then, we apply the FIM-SCP (described in Section 2.1) to eliminate all spatial derivatives from (11) by taking the n-layer integral on both sides of (11), to obtain the following equation at the shifted Chebyshev node x_k, as defined in (2), as

$$\int_0^{x_k}...\int_0^{\xi_2}\left(\frac{u^{\langle m\rangle}-u^{\langle m-1\rangle}}{\tau}+\mathcal{L}u^{\langle m\rangle}\right)d\xi_1...d\xi_n = \int_0^{x_k}...\int_0^{\xi_2}\left(J_1^{\langle m\rangle}+J_2^{\langle m\rangle}+F^{\langle m\rangle}\right)d\xi_1...d\xi_n. \tag{14}$$

To simplify the n-layer integration of the spatial derivative terms of $\mathcal{L}u^{\langle m\rangle}$, by letting it be $Q^{\langle m\rangle}(x_k)$ and using the technique of integration by parts, we have

$$\begin{aligned}
Q^{\langle m\rangle}(x_k) &:= \int_0^{x_k}\cdots\int_0^{\xi_2}\mathcal{L}u^{\langle m\rangle}(\xi_1)\,d\xi_1\ldots d\xi_n\\
&= \int_0^{x_k}\cdots\int_0^{\xi_2}\sum_{i=0}^{n}p_i(\xi_1,t_m)\frac{d^i}{dx^i}u^{\langle m\rangle}(\xi_1)\,d\xi_1\ldots d\xi_n\\
&= \sum_{i=0}^{n}(-1)^i\binom{n}{i}\int_0^{x_k}\cdots\int_0^{\eta_2}p_n^{(i)}(\eta_1,t_m)u^{\langle m\rangle}(\eta_1)d\eta_1\ldots d\eta_i\\
&+\int_0^{x_k}\left(\sum_{i=0}^{n-1}(-1)^i\binom{n-1}{i}\int_0^{\xi_n}\cdots\int_0^{\eta_2}p_{n-1}^{(i)}(\eta_1,t_m)u^{\langle m\rangle}(\eta_1)d\eta_1\ldots d\eta_i\right)d\xi_n\\
&+\int_0^{x_k}\int_0^{\xi_n}\left(\sum_{i=0}^{n-2}(-1)^i\binom{n-2}{i}\int_0^{\xi_{n-1}}\cdots\int_0^{\eta_2}p_{n-2}^{(i)}(\eta_1,t_m)u^{\langle m\rangle}(\eta_1)d\eta_1\ldots d\eta_i\right)d\xi_{n-1}d\xi_n\\
&\;\;\vdots\\
&+\int_0^{x_k}\cdots\int_0^{\xi_2}p_0(\xi_1,t_m)u^{\langle m\rangle}(\xi_1)d\xi_1\ldots d\xi_n+d_1\frac{x_k^{n-1}}{(n-1)!}+d_2\frac{x_k^{n-2}}{(n-2)!}+d_3\frac{x_k^{n-3}}{(n-3)!}+\ldots+d_n,
\end{aligned}$$

where $d_1, d_2, d_3, \ldots, d_n$ are the arbitrary constants which emerge from the process of integration by parts. Then, we substitute each $x_k \in \{x_1, x_2, x_3, \ldots, x_N\}$ into the above equation and utilize the idea of FIM-SCP. Thus, we can express it, in matrix form, by

$$\begin{aligned}
\mathbf{Q}^{\langle m\rangle} &= \sum_{i=0}^{n}(-1)^i\binom{n}{i}\mathbf{A}^i\mathbf{P}_n^{(i)}\mathbf{u}^{\langle m\rangle}+\sum_{i=0}^{n-1}(-1)^i\binom{n-1}{i}\mathbf{A}^{i+1}\mathbf{P}_{n-1}^{(i)}\mathbf{u}^{\langle m\rangle}\\
&+\sum_{i=0}^{n-2}(-1)^i\binom{n-2}{i}\mathbf{A}^{i+2}\mathbf{P}_{n-2}^{(i)}\mathbf{u}^{\langle m\rangle}+\cdots+\mathbf{A}^n\mathbf{P}_0^{(0)}\mathbf{u}^{\langle m\rangle}+\mathbf{X}_n\mathbf{d}\\
&= \sum_{j=0}^{n}\sum_{i=0}^{n-j}(-1)^i\binom{n-j}{i}\mathbf{A}^{i+j}\mathbf{P}_{n-j}^{(i)}\mathbf{u}^{\langle m\rangle}+\mathbf{X}_n\mathbf{d},
\end{aligned}\tag{15}$$

where $\mathbf{A}=\bar{\mathbf{S}}\mathbf{S}^{-1}$ is the shifted Chebyshev integration matrix, $\mathbf{d}=[d_1,d_2,d_3,\ldots,d_N]^\top$,

$$\begin{aligned}
\mathbf{Q}^{\langle m\rangle} &= \left[Q^{\langle m\rangle}(x_1),Q^{\langle m\rangle}(x_2),Q^{\langle m\rangle}(x_3),\ldots,Q^{\langle m\rangle}(x_N)\right]^\top,\\
\mathbf{X}_n &= \left[\mathbf{x}_{n-1},\mathbf{x}_{n-2},\mathbf{x}_{n-3},\ldots,\mathbf{x}_0\right]\ \text{for each}\ \mathbf{x}_i=\tfrac{1}{i!}\left[x_1^i,x_2^i,x_3^i,\ldots,x_N^i\right]^\top,\\
\mathbf{P}_{n-j}^{(i)} &= \operatorname{diag}\left(p_{n-j}^{(i)}(x_1,t_m),p_{n-j}^{(i)}(x_2,t_m),p_{n-j}^{(i)}(x_3,t_m),\ldots,p_{n-j}^{(i)}(x_N,t_m)\right).
\end{aligned}$$

Finally, we vary all points $x_k \in \{x_1, x_2, x_3, \ldots, x_N\}$ in (14) and rearrange them into matrix form by using the FIM-SCP with the derived matrix equations (12), (13), and (15); thus, we obtain

$$\frac{\mathbf{A}^n\mathbf{u}^{\langle m\rangle}-\mathbf{A}^n\mathbf{u}^{\langle m-1\rangle}}{\tau}+\mathbf{Q}^{\langle m\rangle}=\mathbf{A}^n\mathbf{J}_1^{\langle m\rangle}+\mathbf{A}^n\mathbf{J}_2^{\langle m\rangle}+\mathbf{A}^n\mathbf{F}^{\langle m\rangle}$$

or, factorizing the unknown solution $\mathbf{u}^{\langle m\rangle}$ explicitly, as

$$\begin{aligned}
&\left(\mathbf{A}^n+\tau\textstyle\sum_{j=0}^{n}\sum_{i=0}^{n-j}(-1)^i\binom{n-j}{i}\mathbf{A}^{i+j}\mathbf{P}_{n-j}^{(i)}-\frac{\tau^2}{2}\mathbf{A}^n\mathbf{K}_1^{\langle m\rangle}-\tau\mathbf{A}^{n+1}\mathbf{K}_2^{\langle m\rangle}\right)\mathbf{u}^{\langle m\rangle}\\
&+\mathbf{X}_n\mathbf{d}=\tfrac{\tau^2}{2}\mathbf{A}^n\mathbf{K}_1^{\langle 0\rangle}\mathbf{u}^{\langle 0\rangle}+\tau^2\textstyle\sum_{i=1}^{m-1}\mathbf{A}^n\mathbf{K}_1^{\langle i\rangle}\mathbf{u}^{\langle i\rangle}+\mathbf{A}^n\mathbf{u}^{\langle m-1\rangle}+\tau\mathbf{A}^n\mathbf{F}^{\langle m\rangle}.
\end{aligned}\tag{16}$$

Next, consider the given boundary conditions (10) at the endpoints $b \in \{0, L\}$. We can convert them into matrix form by using the linear combination of shifted Chebyshev polynomial (4) in term of the r^{th}-order derivative of u at the iteration time t_m and using (3). Then, we have

$$\frac{d^r}{dx^r}u^{\langle m\rangle}(x)\Big|_{x=b} = \sum_{n=0}^{N-1} c_n^{\langle m\rangle}\frac{d^r}{dx^r}S_n(x)\Big|_{x=b} = \psi_r(t_m)$$

for all $r \in \{0, 1, 2, ..., n-1\}$. We can express the above equation, in matrix form, as

$$\begin{bmatrix} S_0(b) & S_1(b) & \cdots & S_{N-1}(b) \\ S_0'(b) & S_1'(b) & \cdots & S_{N-1}'(b) \\ \vdots & \vdots & \ddots & \vdots \\ S_0^{(n-1)}(b) & S_1^{(n-1)}(b) & \cdots & S_{N-1}^{(n-1)}(b) \end{bmatrix} \begin{bmatrix} c_0^{\langle m\rangle} \\ c_1^{\langle m\rangle} \\ \vdots \\ c_{N-1}^{\langle m\rangle} \end{bmatrix} = \begin{bmatrix} \psi_0(t_m) \\ \psi_1(t_m) \\ \vdots \\ \psi_{n-1}(t_m) \end{bmatrix}, \tag{17}$$

which can be denoted by $\mathbf{B}\mathbf{c}^{\langle m\rangle} = \mathbf{\Psi}^{\langle m\rangle}$ or $\mathbf{B}\mathbf{S}^{-1}\mathbf{u}^{\langle m\rangle} = \mathbf{\Psi}^{\langle m\rangle}$. Finally, we can construct the system of m^{th} iterative linear equations from (16) and (17), which has $N+n$ unknowns containing $\mathbf{u}^{\langle m\rangle}$ and $\mathbf{d}$, as follows:

$$\begin{bmatrix} \mathbf{H}^{\langle m\rangle} & \mathbf{X}_n \\ \mathbf{B}\mathbf{S}^{-1} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{u}^{\langle m\rangle} \\ \mathbf{d} \end{bmatrix} = \begin{bmatrix} \mathbf{E}_1^{\langle m\rangle} \\ \mathbf{\Psi}^{\langle m\rangle} \end{bmatrix}, \tag{18}$$

where $\mathbf{H}^{\langle m\rangle}$ is the coefficient matrix of $\mathbf{u}^{\langle m\rangle}$ in (16) and $\mathbf{E}_1^{\langle m\rangle}$ is the right-hand side column vector of (16). Consequently, the solution $\mathbf{u}^{\langle m\rangle}$ can be approximated by solving the system (18) starting from the given initial condition (9); that is, $\mathbf{u}^{\langle 0\rangle} = [\phi(x_1), \phi(x_2), \phi(x_3), ..., \phi(x_N)]^\top$. Note that, when we would like to find a numerical solution $u(x,t)$ at any point $x \in [0, L]$ for the terminal time T, we can calculate it by the following formula:

$$u(x,T) = \sum_{n=0}^{N-1} c_n^{\langle m\rangle} S_n(x) = \mathbf{s}(x)\mathbf{c}^{\langle m\rangle} = \mathbf{s}(x)\mathbf{S}^{-1}\mathbf{u}^{\langle m\rangle},$$

where $\mathbf{s}(x) = [S_0(x), S_1(x), S_2(x), ..., S_{N-1}(x)]$ and $\mathbf{u}^{\langle m\rangle}$ is the final m^{th} iterative solution of (18).

3.2. Procedure for Solving Inverse Problem of TVIDE

For the inverse problem in this paper, we specifically define the forcing term $F(x,t) := \beta(t)f(x,t)$, where $\beta(t)$ is a missing source function to be retrieved and $f(x,t)$ is the given function. Thus, our considered time-dependent inverse TVIDE problem (1) becomes

$$u_t(x,t) + \mathcal{L}u(x,t) = \int_0^t \kappa_1(x,\eta)u(x,\eta)d\eta + \int_0^x \kappa_2(\xi,t)u(\xi,t)d\xi + \beta(t)f(x,t), \tag{19}$$

where u is an approximate solution of v and the other parameters are defined as in (8). The initial and boundary conditions of (19) are (9) and (10), which satisfy the compatibility conditions. Now, we remove all spatial derivatives from (19) and use the shifted Chebyshev integration matrix (as explained in Section 2.1). Then, we obtain the following matrix equation, based on the same process as in (16), as

$$\begin{aligned} &\left(\mathbf{A}^n + \tau\textstyle\sum_{j=0}^{n}\sum_{i=0}^{n-j}(-1)^i\binom{n-j}{i}\mathbf{A}^{i+j}\mathbf{P}_{n-j}^{(i)} - \frac{\tau^2}{2}\mathbf{A}^n\mathbf{K}_1^{\langle m\rangle} - \tau\mathbf{A}^{n+1}\mathbf{K}_2^{\langle m\rangle}\right)\mathbf{u}^{\langle m\rangle} \\ &+\mathbf{X}_n\mathbf{d} - \tau\mathbf{A}^n\mathbf{f}^{\langle m\rangle}\beta^{\langle m\rangle} = \tfrac{\tau^2}{2}\mathbf{A}^n\mathbf{K}_1^{\langle 0\rangle}\mathbf{u}^{\langle 0\rangle} + \tau^2\textstyle\sum_{i=1}^{m-1}\mathbf{A}^n\mathbf{K}_1^{\langle i\rangle}\mathbf{u}^{\langle i\rangle} + \mathbf{A}^n\mathbf{u}^{\langle m-1\rangle}, \end{aligned} \tag{20}$$

where $\beta^{\langle m\rangle} = \beta(t_m)$, $\mathbf{f}^{\langle m\rangle} = [f(x_1,t_m), f(x_2,t_m), f(x_3,t_m), ..., f(x_N,t_m)]^\top$ and the other parameters in (20) are as defined in Section 3.1. However, the occurrence of missing data is caused by the given

conditions being insufficient to ensure a unique solution to our inverse problem. Hence, an additional condition or observed data needs to be involved. Thus, we use an additional condition, regarding the aggregated solution of the system, in the following form:

$$\int_0^L u(\xi,t)\,d\xi = g(t), \quad t \in [0,T], \tag{21}$$

where $g(t)$ is the measured data at time t, which probably contains measurement errors. In order to illustrate the realistic phenomena of this problem, we assume that the measurement data of the aggregated solution $g(t)$ involves some noise ϵ, which is denoted by $g^\epsilon(t)$ (where $\|g^\epsilon(t) - g(t)\| \leq \epsilon$) and define the noisy value ϵ by a random variable generated by the Gaussian normal distribution with mean $\mu = 0$ and standard deviation $\sigma = p|g(t)|$, where p is the percentage of the noise to be input. Then, the additional condition (21) becomes

$$\int_0^L u(\xi,t)\,d\xi = g^\epsilon(t), \quad t \in [0,T]. \tag{22}$$

Using the concept of FIM-SCP, the additional condition (22) at time t_m can be written, in vector form, as

$$\int_0^L u^{\langle m\rangle}(\xi)\,d\xi = \sum_{n=0}^{N-1} c_n^{\langle m\rangle} \int_0^L S_n(\xi)\,d\xi = \sum_{n=0}^{N-1} c_n^{\langle m\rangle} \bar{S}_n(L)\,d\xi := \mathbf{z}\mathbf{c}^{\langle m\rangle} = \mathbf{z}\mathbf{S}^{-1}\mathbf{u}^{\langle m\rangle} = g^\epsilon(t_m), \tag{23}$$

where $\mathbf{z} = [\bar{S}_0(L), \bar{S}_1(L), \bar{S}_2(L), ..., \bar{S}_{N-1}(L)]$ and each $\bar{S}_n(L)$ is as defined in Lemma 1(iii). Finally, we can establish the following system of m^{th} iterative linear equations for the inverse TVIDE problem (19) by utilizing (20) and (23), which has $N + n + 1$ unknown variables including $\mathbf{u}^{\langle m\rangle}$, $\mathbf{d}$, and $\beta^{\langle m\rangle}$, as

$$\begin{bmatrix} \mathbf{H}^{\langle m\rangle} & \mathbf{X}_n & -\tau\mathbf{A}^n\mathbf{f}^{\langle m\rangle} \\ \mathbf{B}\mathbf{S}^{-1} & \mathbf{0} & \mathbf{0} \\ \mathbf{z}\mathbf{S}^{-1} & \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{u}^{\langle m\rangle} \\ \mathbf{d} \\ \beta^{\langle m\rangle} \end{bmatrix} = \begin{bmatrix} \mathbf{E}_2^{\langle m\rangle} \\ \mathbf{\Psi}^{\langle m\rangle} \\ g^\epsilon(t_m) \end{bmatrix}, \tag{24}$$

where $\mathbf{H}^{\langle m\rangle}$ is the coefficient matrix of $\mathbf{u}^{\langle m\rangle}$ defined in (20) and $\mathbf{E}_2^{\langle m\rangle}$ is the right-hand side column vector of (20). Before seeking an approximate solution $\mathbf{u}^{\langle m\rangle}$ and source term $\beta^{\langle m\rangle}$, as we have mentioned, we must address that our inverse problem is ill-posed. When a noisy value is input into the system, it may cause a significant error. Hence, we need to stabilize the solution of (24) by employing the Tikhonov regularization method. We denote the linear system (24) by the simplified matrix equation as

$$\mathbf{R}\mathbf{y} = \mathbf{b}^\epsilon. \tag{25}$$

Applying the Tikhonov regularization method (6) in order to filter out the noise in the corresponding perturbed data, we can stabilize the numerical solution (25) by using (7). Thus, we have

$$\mathbf{y}_\lambda = (\mathbf{R}^\top\mathbf{R} + \lambda\mathbf{I})^{-1}\mathbf{R}^\top\mathbf{b}^\epsilon. \tag{26}$$

Finally, we can receive the optimal regularization parameter λ by using Morozov's discrepancy principle combined with Newton's method, as described in Section 2.2. Thus, we can directly obtain the corresponding regularized solution by (26).

3.3. Algorithms for Solving the Direct and Inverse TVIDE Problems

For computational convenience, we summarize the aforementioned procedures for finding approximate solutions to the direct (8) and inverse (19) TVIDE problems in Sections 3.1 and 3.2, respectively, as the numerical Algorithms 1 and 2, which are in the form of pseudocode.

Algorithm 1 Numerical algorithm for solving the direct TVIDE problem via FIM-SCP

Input: x, τ, L, T, N, $\phi(x)$, $\psi_r(t)$, $p_i(x,t)$, $\kappa_1(x,t)$, $\kappa_2(x,t)$, and $F(x,t)$.

Output: An approximate solution $u(x,T)$.

1: Set $x_k = \frac{L}{2}\left(\cos\left(\frac{2k-1}{2N}\pi\right)+1\right)$ for $k \in \{1,2,3,...,N\}$ in descending order.
2: Compute $\mathbf{A}$, $\mathbf{B}$, $\mathbf{S}$, $\bar{\mathbf{S}}$, $\mathbf{S}^{-1}$, $\mathbf{X}_n$, and $\mathbf{u}^{\langle 0\rangle}$.
3: Set $m = 1$ and $t_1 = \tau$.
4: **while** $t_m \leq T$ **do**
5: Compute $\mathbf{K}_1^{\langle m\rangle}$, $\mathbf{K}_2^{\langle m\rangle}$, $\mathbf{F}^{\langle m\rangle}$, $\mathbf{H}^{\langle m\rangle}$, $\mathbf{\Psi}^{\langle m\rangle}$, and $\mathbf{E}_1^{\langle m\rangle}$.
6: Find $\mathbf{u}^{\langle m\rangle}$ by solving the linear system (18).
7: Update $m = m+1$.
8: Compute $t_m = m\tau$.
9: **end while**
10: **return** Find $u(x,T) = \mathbf{s}(x)\mathbf{S}^{-1}\mathbf{u}^{\langle m\rangle}$.

Algorithm 2 Numerical algorithm for solving the inverse TVIDE problem via FIM-SCP

Input: x, p, τ, δ, L, T, N, λ_0, $\phi(x)$, $g(t)$, $\psi_r(t)$, $p_i(x,t)$, $\kappa_1(x,t)$, $\kappa_2(x,t)$, and $f(x,t)$.

Output: An approximate solution $u(x,T)$ and the source terms $\beta(t_m)$ at all discretized times.

1: Set $x_k = \frac{L}{2}\left(\cos\left(\frac{2k-1}{2N}\pi\right)+1\right)$ for $k \in \{1,2,3,...,N\}$ in descending order.
2: Compute $\mathbf{A}$, $\mathbf{B}$, $\mathbf{S}$, $\bar{\mathbf{S}}$, $\mathbf{S}^{-1}$, $\mathbf{X}_n$, $\mathbf{A}_N$, and $\mathbf{u}^{\langle 0\rangle}$.
3: Set $m = 1$ and $t_1 = \tau$.
4: **while** $t_m \leq T$ **do**
5: Set the measurement data $g^\epsilon(t_m) = g(t_m) + \epsilon$, where $\epsilon \sim \mathcal{N}(0, p^2|g(t_m)|^2)$.
6: Compute $\mathbf{K}_1^{\langle m\rangle}$, $\mathbf{K}_2^{\langle m\rangle}$, $\mathbf{f}^{\langle m\rangle}$, $\mathbf{H}^{\langle m\rangle}$, $\mathbf{\Psi}^{\langle m\rangle}$, $\mathbf{E}_2^{\langle m\rangle}$, $\mathbf{R}$, and $\mathbf{b}^\epsilon$.
7: Set $n = 0$.
8: **do**
9: Compute $\mathbf{y}_{\lambda_n} = (\mathbf{R}^\top\mathbf{R} + \lambda_n\mathbf{I})^{-1}\mathbf{R}^\top\mathbf{b}^\epsilon$.
10: Compute $\nabla\mathbf{y}_{\lambda_n} = -(\mathbf{R}^\top\mathbf{R} + \lambda_n\mathbf{I})^{-1}\mathbf{y}_{\lambda_n}$.
11: Compute $G(\lambda_n) = \|\mathbf{R}\mathbf{y}_{\lambda_n} - \mathbf{b}^\epsilon\|^2 - \epsilon^2$.
12: Compute $G'(\lambda_n) = 2\lambda_n\|\mathbf{R}\nabla\mathbf{y}_{\lambda_n}\|^2 + 2\lambda_n^2\|\nabla\mathbf{y}_{\lambda_n}\|^2$.
13: Compute $\lambda_{n+1} = \lambda_n - \frac{G(\lambda_n)}{G'(\lambda_n)}$.
14: Update $n = n+1$.
15: **while** $\|\lambda_n - \lambda_{n-1}\| \geq \delta$
16: Set the optimal regularization parameter $\lambda = \lambda_n$.
17: Find $\mathbf{u}^{\langle m\rangle}$ and $\beta^{\langle m\rangle}$ by explicitly solving $\mathbf{y}_\lambda$ using the matrix equation (7).
18: Update $m = m+1$.
19: Compute $t_m = m\tau$.
20: **end while**
21: **return** Find $u(x,T) = \mathbf{s}(x)\mathbf{S}^{-1}\mathbf{u}^{\langle m\rangle}$.

4. Numerical Experiments

In this section, we implement our devised numerical algorithms for solving the direct and inverse TVIDE problems through several examples, in order to demonstrate the efficiency and accuracy of the solutions obtained by proposed methods. Examples 1 and 2 are used to examine Algorithm 1 for the direct TVIDE problems (8). Examples 3 and 4 are inverse TVIDE problems (19), as solved by Algorithm 2. Additionally, time convergence rates and CPU times(s) for each example are presented to indicate the computational cost and time. The time convergence rate is defined by $\text{Rate} = \lim_{t_m \to T} \frac{\|\mathbf{u}^*(t_{m+1}) - \mathbf{u}(t_{m+1})\|_\infty}{\|\mathbf{u}^*(t_m) - \mathbf{u}(t_m)\|_\infty}$, where T is the terminal time, t_m is a partitioned time contained in $[0, T]$, $\mathbf{u}^*(t_m)$ is the exact solution at time t_m, $\mathbf{u}(t_m)$ is the numerical solution at time t_m, and $\|\cdot\|_\infty$ is the l^∞ norm. Graphical solutions of each example are also depicted. Our numerical algorithms were implemented using the MatLab R2016a software, run on a Intel(R) Core(TM) i7-6700 CPU @ 3.40 GHz computer system.

Example 1. *Consider the following direct TVIDE problem, which consists of a second-order derivative with constant coefficient for* $x \in (0, 1)$ *and* $t \in (0, T]$*:*

$$u_t + u_{xx} + u = \int_0^t 2e^{-x} u(x, \eta) d\eta + \int_0^x (\xi + t) u(\xi, t) d\xi + F(x, t), \tag{27}$$

where

$$F(x, t) = -t - e^x(t^2 - 3t + tx - 1),$$

subject to the homogeneous initial condition $u(x, 0) = 0$ *for* $x \in [0, 1]$ *and the Dirichlet boundary conditions* $u(0, t) = t$ *and* $u(1, t) = te$ *for* $t \in [0, T]$*. The analytical solution of this problem is* $u^*(x, t) = te^x$*.*

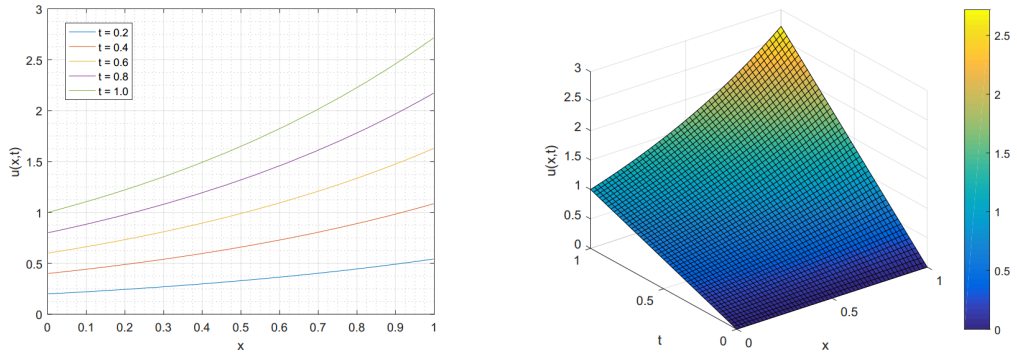
In the numerical testing based on Algorithm 1, we first took the double-layer integral of both sides of (27) and transformed it into matrix form (16). Then, we obtained the approximate solutions $u(x, T)$ for this problem (27) by applying the numerical Algorithm 1. The accuracy of our obtained approximate results was measured by the mean absolute error, which compared it to the analytical solution at different values of $x \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$ and the terminal time $T = 1$, as shown in Table 1. From Table 1, we observe that, when the partitioning number of the temporal domain M was fixed and nodal numbers N were increasingly varied, then the accuracy was significantly improved. Similarly, for a fixed nodal number N but various time partitioning numbers M, the accuracy results were also significantly improved. Moreover, the convergence rates with respect to the time in Algorithm 1 were estimated for various numbers of the time partition ($M \in \{5, 10, 15, 20, 25\}$) for the spatial points $N = 10$, as shown in Table 2. We can notice, from Table 2, that these time convergence rates for the ℓ^∞ norm indeed approached linear convergence for $T \in \{5, 10, 15\}$. The computational cost, in terms of CPU times (s), is also displayed in Table 2. Finally, a graph of our approximate solutions $u(x, t)$ for different times t and the surface plot of the solution under the parameters $N = 20$, $M = 20$, and $T = 1$ are depicted in Figure 1.

Table 1. Mean absolute errors between exact and numerical solutions of $u(x, 1)$ for Example 1.

x	$M = 20$			$N = 12$		
	$N = 8$	$N = 10$	$N = 12$	$M = 11$	$M = 13$	$M = 15$
0.1	1.6855×10^{-5}	1.1723×10^{-8}	1.0208×10^{-10}	2.3823×10^{-7}	5.0060×10^{-9}	3.3268×10^{-11}
0.3	4.3851×10^{-5}	3.0501×10^{-8}	2.6554×10^{-10}	6.1976×10^{-7}	1.3024×10^{-8}	8.6539×10^{-11}
0.5	5.3554×10^{-5}	3.7247×10^{-8}	3.2429×10^{-10}	7.5684×10^{-7}	1.5904×10^{-8}	1.0567×10^{-10}
0.7	4.2555×10^{-5}	2.9599×10^{-8}	2.5767×10^{-10}	6.0140×10^{-7}	1.2638×10^{-8}	8.3964×10^{-11}
0.9	1.5864×10^{-5}	1.1034×10^{-8}	9.6049×10^{-11}	2.2419×10^{-7}	4.7112×10^{-9}	3.1289×10^{-11}

Table 2. Time convergence rates and CPU times (s) for Example 1 by Algorithm 1 with $N = 10$.

M	$T = 5$			$T = 10$			$T = 15$		
	$\|u^* - u\|_\infty$	**Rate**	**Time(s)**	$\|u^* - u\|_\infty$	**Rate**	**Time(s)**	$\|u^* - u\|_\infty$	**Rate**	**Time(s)**
5	4.298×10^{-12}	1.4584	0.0465	8.565×10^{-12}	1.5076	0.0456	1.262×10^{-11}	1.5112	0.0469
10	4.318×10^{-12}	1.2499	0.0487	8.576×10^{-12}	1.2863	0.0469	1.276×10^{-11}	1.3040	0.0485
15	4.309×10^{-12}	1.1723	0.0495	8.547×10^{-12}	1.2008	0.0481	1.275×10^{-11}	1.2135	0.0501
20	4.311×10^{-12}	1.1327	0.0506	8.533×10^{-12}	1.1555	0.0516	1.277×10^{-11}	1.1657	0.0535
25	1.135×10^{-12}	1.1353	0.0521	8.540×10^{-12}	1.1272	0.0538	1.277×10^{-11}	1.1365	0.0553



(**a**) $u(x,t)$ at different times t (**b**) Surface plot of $u(x,t)$

Figure 1. The graphical results of Example 1 for $N = 20$, $M = 20$, and $T = 1$.

Example 2. *Consider the following direct TVIDE problem, which consists of a third-order derivative with variable coefficient for* $x \in (0,1)$ *and* $t \in (0,T]$*:*

$$u_t + tu_{xxx} + \cos(x)u_{xx} = \int_0^t \frac{2}{x-1}u(x,\eta)d\eta + \int_0^x \frac{6t}{\xi - 1}u(\xi,t)d\xi + F(x,t), \tag{28}$$

where

$$F(x,t) = x - x^2 + xt^2(3x+1) - 2t\cos(x),$$

subject to the initial condition $u(x,0) = 0$ *for* $x \in [0,1]$ *and the boundary conditions* $u(0,t) = 0$, $u(1,t) = 0$, *and* $u'(0,t) = t$ *for* $t \in [0,T]$. *The analytical solution of this problem is* $u^*(x,t) = (x - x^2)t$.

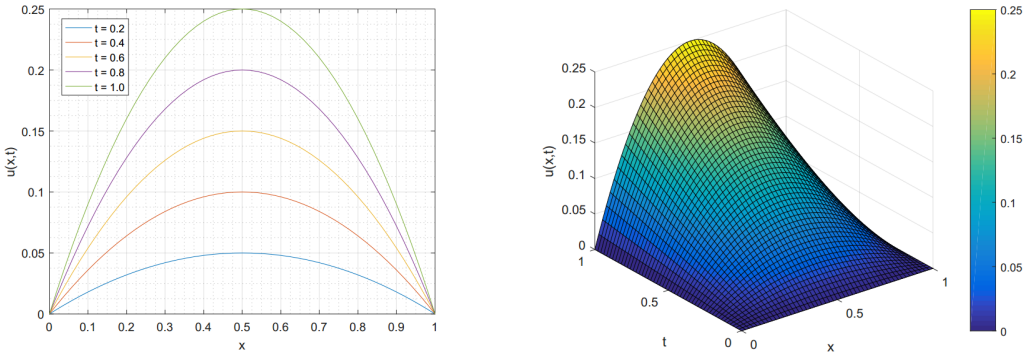
We test the efficiency and accuracy of the proposed Algorithm 1 via the problem (28). First, we took a triple-layer integral on both sides of (28) and utilized the shifted Chebyshev integration matrix to transform it into matrix form (16). Next, we implemented Algorithm 1 to obtain numerical solutions $u(x,T)$ for this problem (28). Table 3 shows the precision of our obtained approximate results at different values of $x \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$ and at the terminal time $T = 1$, through the mean absolute error. We can see that the accuracy was significantly improved according to an increase in the number of both the partitioning space and time domains. However, we observe that, in the case of fixed N, when M was increased, the mean absolute errors provide accurate results with a lower computational number M. Furthermore, the time convergence rates concerning the ℓ^∞ norm and CPU times (s) are demonstrated in Table 4, under various values of M ($M \in \{5, 10, 15, 20, 25\}$) and final times T ($T \in \{5, 10, 15\}$). The graphical solutions for $u(x,t)$ in both one and two dimensions are shown in Figure 2.

Table 3. Mean absolute errors between exact and numerical solutions of $u(x,1)$ for Example 2.

x	$M = 10$			$N = 10$		
	$N = 8$	$N = 10$	$N = 12$	$M = 5$	$M = 10$	$M = 15$
0.1	3.5098×10^{-10}	5.0498×10^{-13}	1.6695×10^{-14}	5.1849×10^{-13}	5.0498×10^{-13}	4.9435×10^{-13}
0.3	1.0060×10^{-9}	1.1285×10^{-12}	4.1411×10^{-14}	1.1544×10^{-12}	1.1285×10^{-12}	1.0850×10^{-12}
0.5	1.0780×10^{-9}	1.4543×10^{-12}	5.1958×10^{-14}	1.4672×10^{-12}	1.4543×10^{-12}	1.3845×10^{-12}
0.7	1.0237×10^{-9}	1.1625×10^{-12}	4.5908×10^{-14}	1.1572×10^{-12}	1.1625×10^{-12}	1.0923×10^{-12}
0.9	3.1567×10^{-10}	5.2400×10^{-13}	2.0983×10^{-14}	5.1567×10^{-13}	5.2400×10^{-13}	4.9050×10^{-13}

Table 4. Time convergence rates and CPU times (s) for Example 2 by Algorithm 1 with $N = 10$.

M	$T = 5$			$T = 10$			$T = 15$		
	$\|u^* - u\|_\infty$	Rate	Time(s)	$\|u^* - u\|_\infty$	Rate	Time(s)	$\|u^* - u\|_\infty$	Rate	Time(s)
5	1.426×10^{-12}	1.0241	0.0524	1.620×10^{-12}	1.0489	0.0531	3.549×10^{-12}	1.3720	0.0535
10	1.533×10^{-12}	1.0334	0.0577	1.635×10^{-12}	1.0278	0.0576	1.874×10^{-12}	1.1035	0.0576
15	1.426×10^{-12}	1.0208	0.0597	1.664×10^{-12}	1.0243	0.0585	1.806×10^{-12}	1.0294	0.0598
20	1.476×10^{-12}	1.0223	0.0609	1.609×10^{-12}	1.0210	0.0610	1.537×10^{-12}	1.0203	0.0620
25	1.496×10^{-12}	1.0165	0.0619	1.488×10^{-12}	1.0182	0.0638	1.276×10^{-12}	1.0079	0.0641



(**a**) $u(x,t)$ at different times t (**b**) Surface plot of $u(x,t)$

Figure 2. The graphical results of Example 2 for $N = 20$, $M = 20$, and $T = 1$.

Example 3. *Consider the following inverse TVIDE problem, which consists of a second-order derivative with constant coefficient and a continuous forcing function $f(x,t)$ for $x \in (0,1)$ and $t \in (0,T]$:*

$$u_t - u_{xx} + 2u = \int_0^t 2\ln(x)u(x,\eta)d\eta + \int_0^x e^{-\xi}u(\xi,t)d\xi + \beta(t)f(x,t), \tag{29}$$

where

$$f(x,t) = e^{2t}\left[1 + t - x + e^x + te^{-x} - (2e^x + t)t\ln x\right],$$

subject to the initial condition $u(x,0) = e^x$ for $x \in [0,1]$ and the boundary conditions $u(0,t) = t + 1$ and $u(1,t) = t + e$ for $t \in [0,T]$. The additional condition, in terms of the aggregated solution of the system, is $g(t) = t + e - 1$. The analytical solutions of this problem are $u^*(x,t) = t + e^x$ and $\beta^*(t) = e^{-2t}$.

Implementing the numerical Algorithm 2 by taking the double-layer integral of both sides of (29) and transforming it into matrix form (24), we obtained the approximate solutions $u(x,1)$ and $\beta(t)$ for this problem (29). As the additional condition was measurement data, there may be an error in the measurement. Therefore, we perturbed the additional condition $g(t)$ with a percentage p of the noise ($p \in \{0\%, 1\%, 3\%, 5\%\}$). In Table 5, we show the accuracy of the solutions $u(x,1)$ and $\beta(t)$, in terms of the mean absolute error, respectively, denoted by $\mathcal{E}_u = \frac{1}{N}\sum_{i=1}^{N}|u_i^* - u_i|$ and $\mathcal{E}_\beta = \frac{1}{M}\sum_{j=1}^{M}|\beta_j^* - \beta_j|$, and the values of the optimal regularization parameters λ at time $t = 1$ with various $M = N \in$

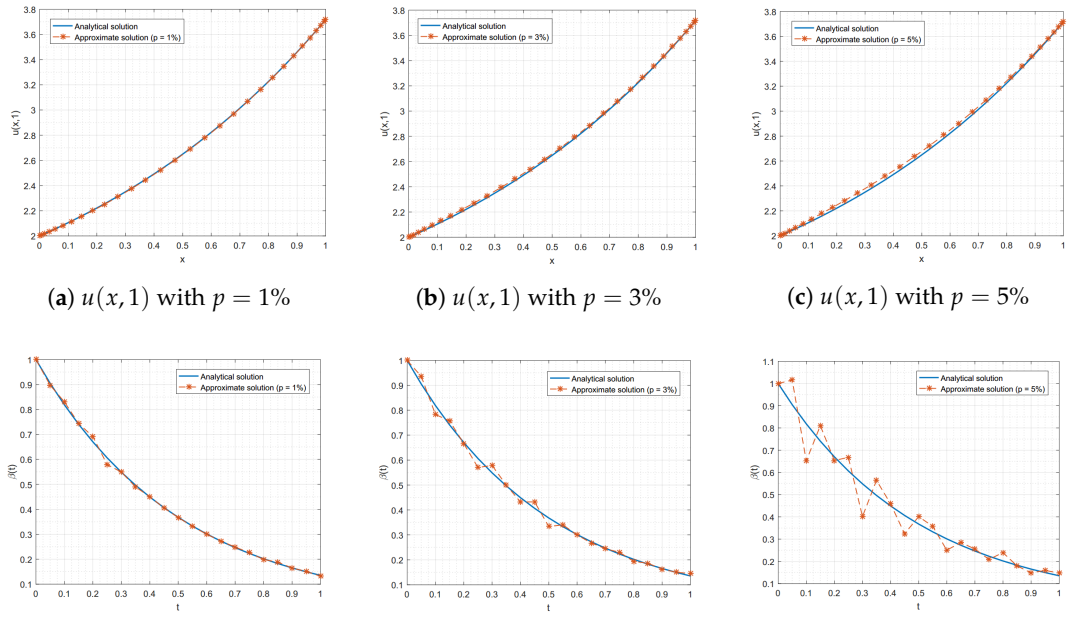
$\{5, 10, 15, 20\}$. From Table 5, we can observe that the optimal regularization parameters λ were close to zero and the mean absolute errors for both $\mathcal{E}_u$ and $\mathcal{E}_\beta$ significantly increased with an increasing percentage p of the perturbation. Furthermore, we used the regularization parameter $\lambda = 0$ to explore the rates of convergence with respect to the ℓ^∞ norm and CPU times (s) for $M = N \in \{5, 10, 15, 20\}$ with the final times $T \in \{1, 2, 3\}$ as shown in Table 6. The graphical solutions of the perturbed functions $u(x, 1)$ and $\beta(t)$ for $p \in \{1\%, 3\%, 5\%\}$ are depicted in Figure 3.

Table 5. Mean absolute errors of $u(x, 1)$ and $\beta(t)$ for optimal regularization parameter λ of Example 3.

$M = N$	$p = 0\%$			$p = 1\%$		
	λ	$\mathcal{E}_u$	$\mathcal{E}_\beta$	λ	$\mathcal{E}_u$	$\mathcal{E}_\beta$
5	6.22×10^{-14}	1.6609×10^{-5}	7.9997×10^{-7}	3.11×10^{-12}	1.1372×10^{-4}	4.1098×10^{-4}
10	2.38×10^{-18}	3.9844×10^{-13}	1.0459×10^{-12}	2.20×10^{-13}	3.4011×10^{-4}	8.8853×10^{-4}
15	1.02×10^{-17}	9.9950×10^{-14}	1.6384×10^{-13}	9.17×10^{-12}	7.8857×10^{-4}	7.0288×10^{-4}
20	2.14×10^{-18}	2.9774×10^{-13}	1.7125×10^{-13}	4.33×10^{-14}	1.9024×10^{-4}	1.3201×10^{-3}
$M = N$	$p = 3\%$			$p = 5\%$		
	λ	$\mathcal{E}_u$	$\mathcal{E}_\beta$	λ	$\mathcal{E}_u$	$\mathcal{E}_\beta$
5	8.80×10^{-11}	1.2533×10^{-3}	8.3870×10^{-3}	8.61×10^{-12}	2.2805×10^{-3}	1.0964×10^{-2}
10	6.64×10^{-11}	3.7067×10^{-3}	9.5414×10^{-3}	1.11×10^{-11}	5.0047×10^{-3}	2.7003×10^{-2}
15	1.09×10^{-12}	8.4361×10^{-3}	7.0094×10^{-3}	8.79×10^{-12}	3.2925×10^{-2}	3.2201×10^{-2}
20	8.61×10^{-12}	3.3382×10^{-3}	1.3582×10^{-2}	1.40×10^{-13}	1.0214×10^{-2}	3.8774×10^{-2}

Table 6. Time convergence rates and CPU times (s) for Example 3 by Algorithm 2 with $N = 10$.

M	$T = 1$			$T = 2$			$T = 3$		
	$\|u^* - u\|_\infty$	Rate	Time(s)	$\|u^* - u\|_\infty$	Rate	Time(s)	$\|u^* - u\|_\infty$	Rate	Time(s)
5	8.495×10^{-13}	1.0046	0.0667	9.130×10^{-13}	1.0203	0.0655	3.602×10^{-12}	1.6644	0.0662
10	8.131×10^{-13}	0.9970	0.0679	8.659×10^{-13}	1.0042	0.0667	2.456×10^{-12}	1.2409	0.0673
15	7.851×10^{-13}	0.9965	0.0684	8.362×10^{-13}	1.0001	0.0675	1.731×10^{-12}	1.1355	0.0693
20	8.344×10^{-13}	1.0003	0.0716	7.829×10^{-13}	0.9967	0.0722	2.928×10^{-12}	1.0956	0.0720
25	8.362×10^{-13}	1.0003	0.0776	8.686×10^{-13}	1.0022	0.0766	2.134×10^{-12}	1.0699	0.0751



(**a**) $u(x, 1)$ with $p = 1\%$ (**b**) $u(x, 1)$ with $p = 3\%$ (**c**) $u(x, 1)$ with $p = 5\%$

(**d**) $\beta(t)$ with $p = 1\%$ (**e**) $\beta(t)$ with $p = 3\%$ (**f**) $\beta(t)$ with $p = 5\%$

Figure 3. The graphical results of $u(x, 1)$ and $\beta(t)$ for Example 3 with $N = 30$ and $M = 20$.

Example 4. *Consider the following inverse TVIDE problem, which consists of a second-order derivative with variable coefficient and the piecewise forcing function $f(x,t)$ for $x \in (0,1)$ and $t \in (0,T]$:*

$$u_t + u_{xx} + u_x - \cos(xt)u = \int_0^t 2\sin(x)u(x,\eta)d\eta - \int_0^x 3t\cos(\xi)u(\xi,t)d\xi + \beta(t)f(x,t), \tag{30}$$

where

$$f(x,t) = \begin{cases} \frac{1}{2}\left[2t\cos(2x) + t\sin(2x) + (t\cos(xt)+1)\sin^2 x\right], & 0 < t \le \frac{T}{3}, \\ \frac{1}{3}\left[2t\cos(2x) + t\sin(2x) + (t\cos(xt)+1)\sin^2 x\right], & \frac{T}{3} < t \le \frac{2T}{3}, \\ \frac{1}{4}\left[2t\cos(2x) + t\sin(2x) + (t\cos(xt)+1)\sin^2 x\right], & \frac{2T}{3} < t \le T, \end{cases}$$

subject to the initial condition $u(x,0) = 0$ for $x \in [0,1]$ and the Dirichlet boundary conditions $u(0,t) = 0$ and $u(1,t) = t\sin^2(1)$ for $t \in [0,T]$. The additional condition, in terms of the aggregated solution of the system, is $g(t) = \frac{t}{4}(2 - sin(2)) + e^t$. The analytical solutions of this problem are $u^*(x,t) = t\sin^2(x)$ and

$$\beta^*(t) = \begin{cases} 2, & 0 < t \le \frac{T}{3}, \\ 3, & \frac{T}{3} < t \le \frac{2T}{3}, \\ 4, & \frac{2T}{3} < t \le T. \end{cases}$$

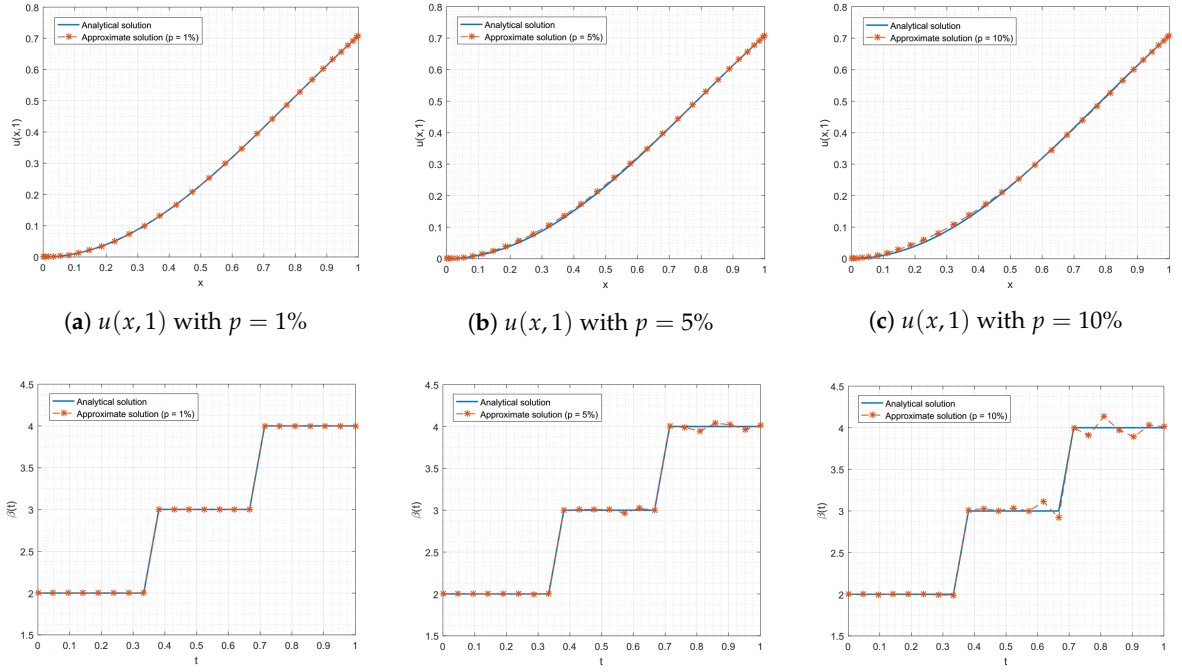
Based on the numerical Algorithm 2, we took the double-layer integral to both sides of (30) and transformed it into matrix form (24). We obtained the approximate solutions $u(x,1)$ and $\beta(t)$ for (29) by implementing Algorithm 2. Table 7 shows the accuracy of the solutions $u(x,1)$ and $\beta(t)$ obtained by our numerical algorithm, in terms of the mean absolute errors $\mathcal{E}_u$ and $\mathcal{E}_\beta$, as well as the values of the optimal regularization parameter λ at time $t = 1$ with the noisy percentage $p \in \{0\%, 1\%, 5\%, 10\%\}$ under various $M = N \in \{6, 9, 12, 15\}$. Although this problem had the piecewise forcing term $f(x,t)$, our Algorithm 2 perfectly performed in providing accurate results, as shown in Table 7. The time convergence rates concerning the ℓ^∞ norm and CPU times (s) are shown in Table 8, under various numbers of $M \in \{6, 9, 12, 15, 18\}$ with the final times $T \in \{1, 2, 3\}$. The graphical perturbed solutions $u(x,1)$ and $\beta(t)$ for $p \in \{1\%, 5\%, 10\%\}$ are shown in Figure 4.

Table 7. Mean absolute errors of $u(x,1)$ and $\beta(t)$ for optimal regularization parameter λ of Example 4.

$M = N$	$p = 0\%$			$p = 1\%$		
	λ	$\mathcal{E}_u$	$\mathcal{E}_\beta$	λ	$\mathcal{E}_u$	$\mathcal{E}_\beta$
6	1.25×10^{-14}	6.4987×10^{-6}	18123×10^{-4}	2.60×10^{-12}	7.8604×10^{-6}	1.8959×10^{-4}
9	7.41×10^{-17}	2.8057×10^{-9}	7.1782×10^{-9}	3.45×10^{-10}	1.2932×10^{-7}	4.0319×10^{-5}
12	2.65×10^{-20}	1.4031×10^{-13}	4.5672×10^{-12}	6.02×10^{-11}	2.6701×10^{-7}	5.1531×10^{-5}
15	6.11×10^{-21}	5.4903×10^{-14}	2.7330×10^{-13}	8.41×10^{-12}	2.9871×10^{-6}	6.8381×10^{-5}
$M = N$	$p = 5\%$			$p = 10\%$		
	λ	$\mathcal{E}_u$	$\mathcal{E}_\beta$	λ	$\mathcal{E}_u$	$\mathcal{E}_\beta$
6	4.19×10^{-12}	8.8690×10^{-5}	5.1802×10^{-4}	5.51×10^{-11}	6.5680×10^{-4}	3.7335×10^{-3}
9	6.20×10^{-13}	1.8419×10^{-5}	7.0830×10^{-4}	7.96×10^{-11}	4.4035×10^{-4}	2.3292×10^{-3}
12	4.11×10^{-12}	7.0910×10^{-5}	1.6889×10^{-3}	8.65×10^{-12}	6.4815×10^{-4}	6.3981×10^{-3}
15	1.01×10^{-13}	2.7507×10^{-4}	1.7821×10^{-3}	5.64×10^{-12}	5.9709×10^{-4}	6.1579×10^{-3}

Table 8. Time convergence rates and CPU times (s) for Example 4 by Algorithm 2 with $N = 12$.

M	$T = 1$			$T = 2$			$T = 3$		
	$\|u^* - u\|_\infty$	**Rate**	**Time(s)**	$\|u^* - u\|_\infty$	**Rate**	**Time(s)**	$\|u^* - u\|_\infty$	**Rate**	**Time(s)**
6	2.681×10^{-13}	1.4574	0.0704	5.338×10^{-13}	1.4564	0.0726	8.024×10^{-13}	1.4563	0.0728
9	2.677×10^{-13}	1.3415	0.0727	5.353×10^{-13}	1.3406	0.0737	8.006×10^{-13}	1.3395	0.0746
12	2.681×10^{-13}	1.2758	0.0735	5.338×10^{-13}	1.2744	0.0753	8.011×10^{-13}	1.2743	0.0767
15	2.666×10^{-13}	1.2312	0.0749	5.360×10^{-13}	1.2332	0.0783	8.015×10^{-13}	1.2337	0.0807
18	2.682×10^{-13}	1.2028	0.0798	5.351×10^{-13}	1.2031	0.0799	7.989×10^{-13}	1.2022	0.0828



(**a**) $u(x,1)$ with $p = 1\%$ (**b**) $u(x,1)$ with $p = 5\%$ (**c**) $u(x,1)$ with $p = 10\%$

(**d**) $\beta(t)$ with $p = 1\%$ (**e**) $\beta(t)$ with $p = 5\%$ (**f**) $\beta(t)$ with $p = 10\%$

Figure 4. The graphical results $u(x,T)$ and $\beta(t)$ for Example 4 with $N = 30$ and $M = 21$.

5. Conclusions and Discussion

In this paper, we utilized FIM-SCP combined with the forward difference quotient to create efficient and accurate numerical algorithms for solving the considered direct and inverse TVIDE problems. According to the numerical examples in Section 4, we have demonstrated the performance of our proposed Algorithm 1 for seeking the approximate solutions of direct TVIDE problems in Examples 1 and 2. We can see that, for Example 1—which involved a second-order derivative with constant coefficients—Algorithm 1 provided an accurate result. Furthermore, for a problem involving a higher-order derivative with variable coefficients, it still provided high accuracy, in terms of solutions, as demonstrated in Example 2. Moreover, we handled inverse TVIDE problems using Algorithm 2, the effectiveness of which was illustrated in Examples 3 and 4. We used the Tikhonov regularization method to deal with the instability of the inverse problem; it can be seen that, in the examples, the regularization parameter λ was close to zero. Algorithm 2 could handle both continuous and piecewise-defined forcing terms with high accuracy, as demonstrated in Examples 3 and 4. Furthermore, when we perturbed the problems by adding noisy values, our Algorithm 2 still overcame the noise and provided approximate results that approached the analytical solutions. We further notice that our presented methods provide high accuracy, even when using only a small number of nodal points. Evidently, when we decrease the time step, they will furnish more accurate results. The rates of convergence with respect to time (based on the ℓ^∞ norm) of our methods were observed to be linear.

Finally, we also depicted the computational times for each example. However, we realize that there exist no theoretical error analysis results for the proposed numerical algorithms. Thus, our future research will study the error analysis, in order to find theories for order of accuracy and rate of convergence for our method. Another interesting direction for our future work is to extend our techniques to solve other types of IDEs and non-linear IDEs.

Author Contributions: Conceptualization, R.B., A.D., and P.G.; methodology, R.B. and A.D.; software, A.D. and P.G.; validation, R.B., A.D., and P.G.; formal analysis, R.B.; investigation, A.D. and P.G.; writing—original draft preparation, A.D. and P.G.; writing—review and editing, R.B.; visualization, A.D. and P.G.; supervision, R.B.; project administration, R.B.; funding acquisition, R.B. All authors have read and agreed to the published version of the manuscript.

Acknowledgments: The authors would like to thank the reviewers for their thoughtful comments and efforts towards improving our manuscript.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

FDM	finite difference method
FIM	finite integration method
FIM-SCP	finite integration method with shifted Chebyshev polynomial
IDE	integro-differential equation
PDE	partial differential equation
TVIDE	time-dependent Volterra integro-differential equation

References

1. Zill, D.G.; Wright, W.S.; Cullen, M.R. *Differential Equations with Boundary-Value Problem*, 8th ed.; Brooks/Cole, Cengang Learning: Boston, MA, USA, 2013.
2. Yanik, E.G.; Fairweather, G. Finite element methods for parabolic and hyperbolic partial integro–differential equations. *Nonlinear Anal.* **1988**, *12*, 785–809. [CrossRef] [CrossRef]
3. Engle, H. *On Some Parabolic Integro–Differential Equations: Existence and Asymptotics of Solution*; Lecture Notes in Mathematics, Springer: Berlin, Germany, 1983.
4. Tang, T. A finite difference scheme for partial integro–differential equations with a weakly singular kernel. *Appl. Numer. Math.* **1993**, *11*, 309–319. [CrossRef] [CrossRef]
5. Aguilar, M.; Brunner, H. Collocation methods for second–order Volterra integro–differential equations. *Appl. Numer. Math.* **1988**, *4*, 455–470. [CrossRef] [CrossRef]
6. Brunner, H. Implicit Runge–Kutta–Nyström methods for general second–order Volterra integro–differential equations. *Comput. Math. Appl.* **1987**, *14*, 549–559. [CrossRef] [CrossRef]
7. Jiang, Y.J. On spectral methods for Volterra-type integro–differential equations. *J. Comput. Appl. Math.* **2009**, *230*, 333–340. [CrossRef] [CrossRef]
8. Burton, T.A. *Volterra Integral and Differential Equations*; Academic Press: New York, NY, USA, 1983.
9. Rahman, M. *Integral Equations and Their Applications*; WIT Press: Southampton, UK, 2007.
10. Hu, Q. Stieltjes derivatives and beta–polynomial spline collocation for Volterra integro–differential equations with singularities. *SIAM J. Numer.* **1996**, *33*, 208–220. [CrossRef] [CrossRef]
11. Brunner, H. Superconvergence in collocation and implicit Runge–Kutta methods for Volterra–type integral equations of the second kind. *Internet Schriftenreihe Numer. Math.* **1980**, *53*, 54–72. [CrossRef]
12. El-Sayed, S.M.; Kaya, D.; Zarea, S. The decomposition method applied to solve high–order linear Volterra–Fredholm integro–differential equations. *Internet J. Nonlinear Sci. Numer. Simulat.* **2004**, *5*, 105–112. [CrossRef] [CrossRef]
13. Kabanikhin, S.I. Definitions and examples of inverse and ill–posed problems. *J. Inverse Ill-Pose Probl.* **2008**, *16*, 317–357. [CrossRef] [CrossRef]
14. Wen, P.H.; Hon, Y.C.; Li, M.; Korakianitis, T. Finite integration method for partial differential equations. *Appl. Math. Model.* **2013**, *37*, 10092–10106. [CrossRef] [CrossRef]

15. Li, M.; Chen, C.S.; Hon, Y.C.; Wen, P.H. Finite integration method for solving multi–dimensional partial differential equations. *Appl. Math. Model.* **2015**, *39*, 4979–4994. [CrossRef] [CrossRef]
16. Li, M.; Tian, Z.L.; Hon, Y.C.; Chen, C.S.; Wen, P.H. Improved finite integration method for partial differential equations. *Eng. Anal. Bound. Elem.* **2016**, *64*, 230–236. [CrossRef] [CrossRef]
17. Boonklurb, R.; Duangpan, A.; Treeyaprasert, T. Modified finite integration method using Chebyshev polynomial for solving linear differential equations. *J. Numer. Ind. Appl. Math.* **2018**, *12*, 1–19. [CrossRef]
18. Rivlin, T.J. *Chebyshev Polynomials, From Approximation Theory to Algebra and Number Theory*, 2nd ed.; John Wiley and Sons: New York, NY, USA, 1990.
19. Tikhonov, A.N.; Goncharsky, A.V.; Stepanov, V.V.; Yagola, A.G. *Numerical Methods for the Solution of Ill–Posed Problems*; Springer: Dordrecht, The Netherlands, 1995. [CrossRef]
20. Sun, Y. Indirect boundary integral equation method for the Cauchy problem of the Laplace equation. *J. Sci. Comput.* **2017**, *71*, 469–498. [CrossRef] [CrossRef]