Research Article

A New Numerical Technique for Solving Volterra Integral Equations Using Chebyshev Spectral Method

Ahmed A. Khidir

1Faculty of Technology of Mathematical Sciences and Statistics, Alneelain University, P.O. Box 12702, Khartoum, Sudan
2Department of Mathematics, Faculty of Sciences, University of Tabuk, P.O. Box 741, Tabuk, Saudi Arabia

Correspondence should be addressed to Ahmed A. Khidir; ahmed.khidir@yahoo.com

Received 16 April 2021; Accepted 25 August 2021; Published 3 September 2021

1. Introduction

Many mathematical formulations of physical phenomena contain integral equations; these equations arise in many fields including biology, economics, engineering, and medicine; for more details, see [1–4]. Finding solutions of linear or nonlinear Volterra integral equations is highly interesting for researchers and scientists in this field and there are available studies to find analytical or numerical solutions for Volterra integral equations (see [5–8]). In fact, Volterra integral equations are usually difficult to solve analytically, so we resort to finding approximate solutions to the problems using numerical or analytical approximation methods. The main purpose of this work is to introduce a new method for solving Volterra integral equations that uses Chebyshev spectral collocation method. Chebyshev spectral collocation methods have been applied successfully in different areas of sciences and engineering because of their ability to give highly accurate solutions of single or system of boundary value problems [9–13]. Chebyshev spectral methods are defined everywhere in the computational domain. Therefore, it is easy to compute high accurate values of a considered function at any point of the domain.

In the proposed method, we introduce a new integral transformation that is the replacement of an integral term in the Volterra integral equation by Chebyshev spectral differential matrix of known elements. The advantages of this approach over the standard same approaches are as follows: (i) the current technique suggests a standard way of choosing the auxiliary linear operator of the integral equation which is the main motivation behind the selection of the current algorithm whereas the other methods are choosing a linear operator to be simple in order to ensure that the integral or integrodifferential equations can be easily integrated and (ii) this algorithm transforms the integral equation(s) into a system of linear algebraic equations which is easier and faster to solve when compared to a system of integral equations. One of the disadvantages of this method is that it cannot be applied directly to the nonlinear integral equations; therefore, we need first to linearized the nonlinear integral equation using any available method such as the successive linearization method [14–17]. The applicability, accuracy, and reliability of the proposed method are confirmed by solving the various Volterra integral equations.

This paper is organized as follows: in Section 2, we introduce a description of the proposed method. In Section 3, the proposed method is implemented in some cases. The numerical results are discussed and investigated in Section 4. Finally, the paper is concluded in Section 5.
2. Description of the Method

Let us consider the Volterra integral equation of the second kind defined as

\[ u(x) + \int_0^x K(x,t)u(t)dt = f(x), \quad (1) \]

with

\[ u(0) = f(0), \quad (2) \]

where \( u(x) \) is an unknown function and the kernel \( K(x,t) \) and the source term function \( f(x) \) are given functions.

In the proposed method, we are seeking for \( u(x) \) by the Chebyshev interpolating polynomials at the L–Lobatto collocation points defined on the domain \([0,x_N]\) by [18–20]

\[ x_j = \frac{1}{2}x_N \left(1 - \cos \frac{j\pi}{N}\right), \quad j = 0, 1, 2, \ldots, N. \quad (3) \]

The unknown function \( u(x) \) is approximated as a truncated series of Chebyshev polynomials of the form

\[ u(x) = u(x_j) = \sum_{k=0}^{N} u_k T_k(x_j), \quad (4) \]

where \( T_k \) is the \( k \)th Chebyshev polynomial and \( u_k \) are the Chebyshev coefficients. The derivatives of \( u(x) \) at the collocation points are represented as

\[ \frac{d^r u}{dx^r} = \sum_{k=0}^{N} \mathcal{D}_{jk} u_k(x_j) = \mathcal{D}^r u, \quad (5) \]

where \( r \) is the order of differentiation, \( \mathcal{D} \) is the Chebyshev differentiation matrix, and \( N + 1 \) is the number of collocation points. We express the entries of the differentiation matrix \( \mathcal{D} \) as [18–20]

\[ \mathcal{D}_{00} = \frac{2N^2 + 1}{6}, \]

\[ \mathcal{D}_{jk} = \frac{c_j}{c_k} \left(1 \right)^{j-k}, \quad j \neq k, j, k = 0, 1, \ldots, N, \]

\[ \mathcal{D}_{kk} = \frac{\xi_k}{2(1 - \xi_k)}, \quad k = 1, 2, \ldots, N - 1, \]

\[ \mathcal{D}_{NN} = \frac{2N^2 + 1}{6}. \]

We introduce the following simple integral equation:

\[ \phi(x) = \int_0^x v(t)u(t)dt, \quad (7) \]

where \( v(t) \) is a known function. Clearly, the existence of the function \( \phi(x) \) depends on the existence of the integral. One approach is to note that the integral equation (7) is a special case of an initial value problem expressed as

\[ \frac{d}{dx} \phi(x) - v(x)u(x) = 0, \quad \text{with } \phi(0) = 0. \quad (8) \]

This differential equation can be solved using the Chebyshev spectral collocation method by replacing the differential operator \( d/dx \) with the transformed Chebyshev differentiation matrix \( \mathbf{D} \) defined on the domain \( x \in [0,x_N] \) (see [21]) and also the functions \( v(x) \) and \( u(x) \) are written in vectors forms of sizes \( (N + 1) \times 1 \); this yields

\[ \mathbf{D} \mathbf{\phi} = \mathbf{v}^T \mathbf{u}, \quad \text{with } \phi(0) = 0, \quad (9) \]

where \( \phi = [\phi_0, \phi_1, \ldots, \phi_N]^T \), \( u = [u_0, u_1, \ldots, u_N]^T \), and \( v = v(x) \) and the superscript \( T \) denotes the transpose.

From the solution of the initial value problem (9), the integral equation (7) can be written in matrices form of unknown function \( u(x) \) as follows:

\[ \int_0^x v(t)u(t)dt = \begin{bmatrix} 1 & 0 & \cdots & 0 & 0 \\ \mathbf{D}_{1,0} & \mathbf{D}_{1,N} & \cdots & \mathbf{D}_{1,N} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{D}_{N,0} & \mathbf{D}_{N,1} & \cdots & \mathbf{D}_{N,N} \end{bmatrix}^{-1} \begin{bmatrix} v_0 \\ v_1 \\ \vdots \\ v_N \end{bmatrix}, \quad (10) \]

where \( u_0, v_0, u_N, v_N \) are known values; here, the first row of transformed Chebyshev differential matrix \( \mathbf{D} \) in equation (10) is interchanging by the row \([1 \ 0 \ \cdots \ 0]\). This is caused by the condition \( \phi(0) = 0 \).

Now, according to equations (7) and (10), we introduce the following transformation.

2.1. Definition. Let \( u(x) \) and \( v(x) \) be continuous functions on \([0,x_N]\); then, the integral \( \int_0^x v(t)u(t)dt \) is transformed into matrices form as follows:

\[ \int_0^x v(t)u(t)dt = \mathbf{L}_u u - u_0 v_0, \quad (11) \]

where \( \mathbf{L} \) is a square matrix of size \((N + 1) \times (N + 1)\) defined as

\[ \mathbf{L} = \mathbf{D}^{-1} \begin{bmatrix} 1 & 0 & \cdots & 0 \\ \mathbf{D}_{1,0} & \mathbf{D}_{1,N} & \cdots & \mathbf{D}_{1,N} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{D}_{N,0} & \mathbf{D}_{N,1} & \cdots & \mathbf{D}_{N,N} \end{bmatrix}^{-1}, \quad (12) \]

and \( \mathbf{L}_u \) is a multiplication of matrix \( \mathbf{L} \) by the vector \( v \) defined by
\[ \mathbf{L}_{v} = \begin{bmatrix} 1 & 0 & \cdots & 0 & 0 \\ D_{1,0} & \cdots & D_{1,N} \\ \vdots & \ddots & \vdots \\ D_{N,0} & \cdots & D_{N,N} \end{bmatrix} \begin{bmatrix} v_0 \\ v_1 \\ \vdots \\ v_N \end{bmatrix}, \quad (13) \]

where \( \mathbf{D} \) is the modified matrix \( \mathbf{D} \) after implementing initial condition. It is clear that this transformation is linear and it is easy to show this property.

The existence and uniqueness of the linear transformation \( \mathbf{L} \) defined in (12) depend on the existence and uniqueness of the inverse of matrix \( \mathbf{D} \). In linear algebra, if \( A \) is a square matrix of size \( N \times N \), then (i) \( A \) is invertible if and only if rank \( (A) = N \) and (ii) if \( A \) is invertible, then its inverse is unique [22]. We applied this theorem to all square matrices \( \mathbf{D} \) to show that these matrices are invertible. In Figure 1, we display all the sizes of the matrices \( \mathbf{D} \) of sizes \( (N+1) \times (N+1) \) against the corresponding ranks for \( 1 \leq N \leq 200 \). We notice that, from Figure 1, the rank of the matrix \( \mathbf{D} \) is equal to its size for all those values of \( N \); that is, the matrix \( \mathbf{D} \) is invertible and consequently the inverse is unique.

For the kernel \( K(x,t) \) of the Volterra integral equation (1), we assume that the function \( K(x,t) \) can be separated into two multiplied functions \( w(x) \) and \( v(t) \) as follows:

\[ \int_{0}^{x} K(x,t)u(t)dt = \int_{0}^{x} w(x)v(t)u(t)dt = w(x) \int_{0}^{x} v(x)u(x)dx. \quad (14) \]

Substituting (5), (11), and (14) in equation (1) gives the following system of linear equations:

\[ \mathbf{Au} = \mathbf{F}, \quad (15) \]

in which \( \mathbf{A} \) is an \( (N+1) \times (N+1) \) square matrix and \( \mathbf{F} \) is \( (N+1) \times 1 \) column vector obtained by

\[ \mathbf{A} = \mathbf{I} + \text{diag}[w] \text{diag}[\mathbf{L}_{x}], \quad \mathbf{F} = [u(x_0), u(x_1), \ldots, u(x_N)]^T. \quad (16) \]

where \( \text{diag} [\ldots] \) is the diagonal matrix of size \( (N+1) \times (N+1) \) that puts the vector \( [\ldots] \) of size \( (N+1) \times 1 \) on the main diagonal and \( \mathbf{I} \) is an identity matrix of size \( (N+1) \times (N+1) \). Finally, the solution of the Volterra integral equation (1) is obtained by solving the linear system (15) as

\[ u = \mathbf{A}^{-1}\mathbf{F}. \quad (17) \]

The general idea underpinning the use of the proposed method is the replacement of a linear Volterra integral equation by a system of linear algebraic equations that replace the integral part in the integral equation by the introduced integral transformation. The obtained linear algebraic equations can easily be solved with the help of symbolic computation software such as Maple, Mathematica, MATLAB, and other symbolic computer packages.

### 3. Numerical Examples

To illustrate the effectiveness and accuracy of the proposed technique in the present work, several examples are carried out in this section.

**Example 1.** Consider the following Volterra integral equation:

\[ u(x) + xe^{-x^2}(1 + x) + \int_{0}^{x} xe^{-x^2}(e^t - e^{-x}u(t))dt = 0. \quad (18) \]

The exact solution is \( u(x) = -xe^x \). According to (14), we have

\[ u(x) + xe^{-x^2}(1 + x) + xe^x\int_{0}^{x} e^{-x^2}(e^t - e^{-x}u(t))dt = 0. \quad (19) \]

To apply the proposed method to this test problem, we rewrite equation (19) in the following system of linear equations:

\[ \mathbf{Au} = \mathbf{F}, \quad (20) \]

where \( \mathbf{A} = \mathbf{I} - \text{diag}[x] \mathbf{L}_{x^2}, \quad \mathbf{F} = [-xe^{-x^2}(1 + x) - xe^x \mathbf{L}_{x^2}]^T, \quad \text{and} \quad u = [u(x_0), u(x_1), \ldots, u(x_N)]^T. \]

Thus, the final solution is obtained as

\[ u = \mathbf{A}^{-1}\mathbf{F}. \quad (21) \]

**Example 2.** Consider the following Volterra–Fredholm integral equation [23]:

\[ u(x) - \int_{0}^{x} xt u(t)dt + \int_{0}^{1} (t - x)u(t)dt = f(x), \quad (22) \]

where \( f(x) = -(2/5)x^{7/2} - (2/3)x + \sqrt{x} + (2/5) \). The exact solution is \( u(x) = x \sqrt{x} \). According to (14), we have

\[ u(x) - x \int_{0}^{x} tu(t)dt + \int_{0}^{1} tu(t)dt - x \int_{0}^{1} u(t)dt = f(x). \quad (23) \]

Applying the proposed method to this problem, equation (23), we obtain the system of linear equations:

\[ \mathbf{Au} = \mathbf{F}, \quad (24) \]

where \( \mathbf{A} = \mathbf{I} - \text{diag}[x] \mathbf{L}_{x^2} + R\text{diag}[x] - \text{diag}[x] \mathbf{R}, \quad \mathbf{F} = [-xe^{-x^2}(1 + x) - xe^x \mathbf{L}_{x^2}]^T, \quad \text{and} \quad u = [u(x_0), u(x_1), \ldots, u(x_N)]^T; \) the rows of the matrix \( \mathbf{R} \) are obtained by subtracting the first row from the last row of the matrix:

\[ \mathbf{L}_{v} = \begin{bmatrix} 1 & 0 & \cdots & 0 & 0 \\ D_{1,0} & \cdots & D_{1,N} \\ \vdots & \ddots & \vdots \\ D_{N,0} & \cdots & D_{N,N} \end{bmatrix}^{-1}. \quad (25) \]

Thus, the final solution of the Volterra–Fredholm integral equation (22) is obtained as
Example 3. Consider the following system of linear Volterra integral equations [24]:

\[
\begin{align*}
(3x - 8)u(x) + (5 - 2x)g(x) - \int_0^x (x + t)u(t)\,dt - \int_0^x xt\,g(t)\,dt &= f_1(x), \\
4xu(x) + (x - 5)g(x) - \int_0^x (2x - t)u(t)\,dt - \int_0^x (xt + t)\,g(t)\,dt &= f_2(x),
\end{align*}
\]

with

\[
\begin{align*}
f_1(x) &= 5e^x - 2x - 9\sin x - x^2e^x + 2x\cos x - xe^x + 3x\sin x, \\
f_2(x) &= -3x - 4e^x + \sin x - x^2e^x + x\cos x + xe^x + 4x\sin x - 1,
\end{align*}
\]

where the exact solution is \(u(x) = \sin x\) and \(g(x) = e^x\). To illustrate the use of the proposed method to approximate the solution of this integral equations system, we rewrite the system as

\[
\begin{align*}
(3x - 8)u(x) + (5 - 2x)g(x) - x\int_0^x u(x)\,dx - \int_0^x xu(x)\,dx - x\int_0^x g(x)\,dx &= f_1(x), \\
4xu(x) + (x - 5)g(x) - 2x\int_0^x u(x)\,dx + \int_0^x xu(x)\,dx - x\int_0^x g(x)\,dx &= f_2(x).
\end{align*}
\]

Now, converting the system above into a system of linear algebraic equations using the proposed method gives

\[
\begin{align*}
\begin{bmatrix}
(3x - 8) - [x]L_{[t]} - L_{[x]} & u + (5 - 2x) - [x]L_{[x]} \\
4x & u + (x - 5) - [x]L_{[x]} - L_{[x]}
\end{bmatrix}
\begin{bmatrix}
u \\
g
\end{bmatrix} = \begin{bmatrix}
f_1 \\
f_2
\end{bmatrix}.
\end{align*}
\]
where $F_2 = [f_1 - xu(0)]^T$ and $F_2 = [f_2 - 2xu(0)]^T$. Finally, the solution of the system of integral equations (30) is obtained as

$$u = A^{-1}F,$$  \hspace{1cm} (31)

where $u, F$, and $A$ are defined as

$$u = [u(x_0), u(x_1), \ldots, u(x_N), g(x_0), g(x_1), \ldots, g(x_N)]^T,$$

$$F = [F_1; F_2],$$

$$A = \begin{bmatrix}
3x - 8 - [x] L_{[1]} - L_{[x]} & 5 - 2x - [x] L_{[x]} \\
4x - [2x] L_{[1]} + L_{[x]} & [x] - 5 - [x] L_{[x]} - L_{[1]}
\end{bmatrix}. \hspace{1cm} (32)$$

**Example 4.** Consider the following quadratic nonlinear Volterra integral equation [25]:

$$u(x) - \frac{x}{10} \sqrt{u(x)} \int_0^1 \frac{1}{1 + u(t)} \ dt = f(x), \quad 0 \leq x \leq 1,$$ \hspace{1cm} (33)

with $f(x) = e^x + (x/10) \sqrt{2} \left( \log(1 + e^{-x}) - \log(2) \right)$. The exact solution is $u(x) = e^x$.

For solving the nonlinear Volterra integral equation (33), we first use the successive linearization method (SLM) [14–17] to linearize the nonlinear integral equation (33). The SLM was recently introduced as an efficient linearization algorithm and robust method for solving boundary value problems. The procedure of the SLM is based on transforming the governing nonlinear equation into an iterative scheme made up of linear equations which is subsequently solved using the proposed method. In applying the SLM, we assume that the solution of (33) is obtained by

$$u(x) = u_m(x) + u_{m-1}(x), \quad \text{with } u_m(0) = 0, m = 1, 2, 3, \ldots,$$ \hspace{1cm} (34)

where $m$ is the iteration of SLM and $u_m(x)$ are unknown functions that are obtained by solving the linearized integral equation assuming that $u_{m-1}(x)$ are known functions from previous iterations. The procedure starts with an initial guess $u_0(x)$ which is chosen to satisfy the condition $u_0(0) = f(0)$. A suitable initial guess that satisfies this condition of equation (33) is

$$u_0(x) = f(x).$$ \hspace{1cm} (35)

The linearized integral equation to be solved is obtained by substituting equation (34) in integral equation (33) and deleting the nonlinear terms. Thus, the linearized equation is expressed as

$$u_m(x) + \frac{x}{10} \sqrt{u_{m-1}(x)} \int_0^x \frac{1}{1 + u_{m-1}(t)} u_m(t) \ dt - \frac{u_m(x)}{2u_{m-1}} \int_0^x \frac{1}{1 + u_{m}(t)} \ dt = f_1(x),$$ \hspace{1cm} (36)

where

$$f_1(x) = f(x) - u_{m-1}(x) + (x/10) \sqrt{u_{m-1}(x)} \int_0^x \frac{1}{1 + u_{m-1}(t))} dt.$$ \hspace{1cm}

Applying the proposed method on (36) gives the following matrices system:

$$A = I + \text{diag} \left[ \frac{x}{10} \sqrt{u_{m-1}} \right] L_{v_1} - \text{diag} \left[ \frac{x}{20 \sqrt{u_{m-1}}} u_{m} \right] L_{v_2} + \text{diag} \left[ \frac{x}{20 \sqrt{u_{m-1}}} v_2 \right],$$ \hspace{1cm} (38)

where $v_1 = 1/(1 + u_{m})^2$ and $v_2 = 1/(1 + u_{m})$. Finally, the solution of the integral equation (33) is given by

$$u = u_m = A^{-1}F.$$ \hspace{1cm} (39)

**Example 5.** Consider the following system of nonlinear Volterra integral equations [26, 27]:

where

$$u(x) - \int_0^x \left( u^2(t) + g^2(t) \right) dt = \sin x - x,$$

$$g(x) - \int_0^x u(t) g(t) dt = \cos x - \frac{\sin^2 x}{2},$$ \hspace{1cm} (40)

with the exact solution $(u(x), g(x)) = (\sin x, \cos x)$. To apply the proposed method to this test problem, we first
linearize the equations using the SLM; we start by assuming that the solution of system (40) may be obtained by

\[ u(x) = u_m(x) + u_{m-1}(x), \]
\[ g(x) = g_m(x) + g_{m-1}(x), \quad m = 1, 2, 3, \ldots \]  

(41)

Substituting (41) into (40) gives the linearized form

\[
\begin{align*}
&u_m(x) - 2\int_0^x u_{m-1}(t)u_m(t)dt - 2\int_0^x g_{m-1}(t)g_m(t)dt \\
&= \sin x - x - u_{m-1}(x) + \int_0^x u_m^2(t)dt + \int_0^x g_m^2(t)dt, \\
&g(x) - \int_0^x g_{m-1}(t)u_m(t)dt - \int_0^x u_m(t)g_m(t)dt - \int_0^x u_{m-1}(t)g_m(t)dt \\
&= \cos x - \frac{\sin^2 x}{2} - g_{m-1} + \int_0^x g_m(t)u_m(t)dt.
\end{align*}
\]

(42)

The suitable initial guesses for \( u_0(x) \) and \( g_0(x) \) are \( u_0(x) = \sin x - x \) and \( g_0(x) = \cos x - (\sin^2 x)/2 \). Applying the proposed method on (42) gives the following \( m \) th approximations of the linear system of equations:

\[
\begin{align*}
(I - 2L_{n_{m,1}})u_m - 2L_{g_{m,1}}g_m &= F_{1,m-1}, \\
-L_{g_{m,1}}u_m + (I - L_{n_{m,1}})g_m &= F_{2,m-1},
\end{align*}
\]

(43)

with

\[
F_{1,m-1} = \sin x - x - u_{m-1} + L_{n_{m-1}}u_{m-1} - u_{m-1}(0) - g_{m-1}(0),
\]
\[
F_{2,m-1} = \cos x - \frac{\sin^2 x}{2} - g_{m-1} + L_{g_{m-1}}u_{m-1} - u_{m-1}(0)g_{m-1}(0),
\]

(44)

where \( u_m, g_m, F_{1,m-1}, \) and \( F_{1,m-1} \) are \((N+1) \times 1\) column vectors. System (43) leads to the matrix equation given as

\[
A_{m-1}U_m = F_{m-1},
\]

(45)

where \( U_m, F_{m-1}, \) and \( A_{m-1} \) are defined as

\[
U = [u_m \, g_m]^T, \\
F = [F_{1,m-1} \, F_{2,m-1}]^T, \\
A_{m-1} = \begin{bmatrix} I - 2L_{n_{m-1}} & -2L_{g_{m-1}} \\ -L_{g_{m-1}} & I - L_{n_{m-1}} \end{bmatrix}.
\]

(46)

Thus, the final solution of the integral equations (40) for \( u(x) \) and \( g(x) \) is obtained as

\[
[u \, g]^T = U_m = A_{m-1}^{-1}F_{m-1}.
\]

(47)

4. Results and Discussion

In this section, we present the numerical results obtained using the proposed method for solving the Volterra integral equations. Implementation of the numerical schemes was performed using the personal computer of 2.5 GHz CPU speed including MATLAB 2017 package to perform the simulation results. The accuracy of the method is demonstrated by presenting infinity error norms \( u_{E}(x) \) between exact and approximate solutions computed as

\[
u_{E}(x) = \max|u(x) - \bar{u}(x)|,
\]

(48)

where \( u(x) \) and \( \bar{u}(x) \) are the exact and the numerical solutions, respectively. The accuracy of the current results was checked by comparing them with other methods reported in the literature. We also generate the computational times of the results to show the computational efficiency of the proposed method. The results of our solutions for all introduced cases in this work are displayed in the tables and graphs and discussed below.

Table 1 gives a comparison between the absolute errors of the present results and the Adomian Decomposition Method (ADM) at different iterations at selected nodes for Example 1. A striking feature of the proposed method is that a high level of accuracy is achieved at the first order of approximation than the ADM, for example, whereas the proposed method gives the exact value at only one iteration for the 14th digits. This is because the proposed method selects all the terms as linear operator.

Table 2 shows a comparison between the results obtained by the current method and Shifted Chebyshev Collocation (SCC) method reported by Yousri and Hafez [23]; both methods have been applied to the Volterra–Fredholm integral equation (Example 2). It is noticed that accurate results with errors of order up to eight digits are observed.
than the BPE, ANN, and TQR methods. Table 4 shows the selected nodes obtained by the proposed method, the improvement in the accuracy of the solution using the proposed method, and the Central Processing Unit (CPU) time (sec) for Example 1.

| x     | 1st iteration | 2nd iteration | 3rd iteration | 4th iteration | Current results |
|-------|---------------|---------------|---------------|---------------|-----------------|
| 0     | 0             | 0             | 2.960e − 009  | 1.323e − 011  | 1.665e − 015    |
| 0.25  | 7.214e − 005  | 5.318e − 007  | 5.291e − 007  | 4.719e − 007  | 6.123e − 009    |
| 0.50  | 1.385e − 003  | 2.931e − 005  | 5.924e − 006  | 5.924e − 006  | 1.276e − 007    |
| 0.75  | 6.417e − 003  | 2.226e − 004  | 2.770e − 005  | 7.781e − 007  | 7.105e − 015    |
| 1.00  | 1.679e − 002  | 7.669e − 004  | 7.312e − 005  | 2.503e − 006  | 2.655e − 015    |
| 1.25  | 3.241e − 002  | 1.743e − 003  | 1.443e − 004  | 5.520e − 006  | 7.994e − 015    |
| 1.50  | 5.201e − 002  | 3.087e − 003  | 2.334e − 004  | 9.594e − 006  | 3.730e − 014    |
| 1.75  | 7.398e − 002  | 4.664e − 003  | 3.312e − 004  | 1.427e − 005  | 7.461e − 014    |
| 2.00  | 9.688e − 002  | 6.334e − 003  |               |               |                 |

Table 2: Comparison of the absolute errors with various choices of \( N \) and \( x \) for Example 2.

| x     | Yousri and Hafez [23] | Current method |
|-------|------------------------|----------------|
| 0.1   | 2.96e − 003            | 7.30e − 009    |
| 0.2   | 8.79e − 004            | 2.30e − 009    |
| 0.3   | 7.10e − 004            | 2.70e − 009    |
| 0.4   | 9.59e − 004            | 7.81e − 009    |
| 0.5   | 1.75e − 005            | 1.31e − 008    |
| 0.6   | 7.12e − 004            | 1.87e − 008    |
| 0.7   | 2.48e − 004            | 2.49e − 008    |
| 0.8   | 1.87e − 004            | 3.19e − 008    |
| 0.9   | 4.93e − 004            | 4.02e − 008    |
| 1.0   | 5.12e − 004            | 5.03e − 008    |

using the proposed method while the SCC method is accurate up to six digits when applied to this example. Also, we notice that increasing the values of \( N \) leads to improvement in the accuracy of the solution using the SCC method whereas increasing the values of \( N \) does not affect the improvement in the accuracy of the solution using the proposed method.

In Table 3, we present the numerical results of the absolute errors \( u_E(x) \) and \( g_E(x) \) for \( u(x) \) and \( g(x) \), respectively, at selected nodes obtained by the proposed method, the Bernstein Polynomials Expansion (BPE) method [24], the Artificial Neural Network (ANN) method [24], and the Trapezoidal Quadrature Rule (TQR) method [28] for Example 3. It is remarkable to note that accurate results with errors of order up to sixteen digits are obtained using the proposed method while the BPE, ANN, and TQR methods are accurate up to only two, five, and two digits, respectively. This indicates that the proposed method converges much faster than the BPE, ANN, and TQR methods.

| \( x = 1/2^r \) | BPE method | ANN method | TQR method | Presented results |
|----------------|------------|------------|------------|------------------|
| \( u_E(x) \)   |            |            |            |                  |
| \( r = 1 \)    | 2.66e − 005| 1.160e − 002| 8.320e − 003| 5.51e − 017      |
| \( r = 2 \)    | 5.85e − 005| 4.300e − 003| 2.140e − 002| 5.51e − 017      |
| \( r = 3 \)    | 8.55e − 005| 2.040e − 003| 3.010e − 002| 6.106e − 016     |
| \( r = 4 \)    | 6.14e − 005| 1.730e − 003| 3.910e − 002| 5.51e − 017      |
| \( r = 5 \)    | 3.89e − 005| 8.010e − 004| 1.120e − 001| 7.355e − 016     |
| \( r = 6 \)    | 2.12e − 005| 6.230e − 004| 3.620e − 001| 4.823e − 016     |
| \( r = 7 \)    | 1.11e − 005| 5.370e − 004| 8.540e − 002| 1.422e − 016     |
| \( r = 8 \)    | 5.70e − 006| 4.950e − 004| 6.370e − 002| 3.209e − 017     |
| \( r = 9 \)    | 2.90e − 006| 4.750e − 004| 8.640e − 002| 4.597e − 017     |

\( g_E(x) \)

Table 4: Errors of the approximate solutions \( u(x) \) and \( g(x) \) for increasing values of \( N \) for Example 3.

| \( N \) | \( u_E(x) \) | \( g_E(x) \) | CPU time (sec) |
|---------|-------------|-------------|---------------|
| 10      | 5.96e − 013 | 6.817e − 013| 0.044         |
| 15      | 3.109e − 015| 3.997e − 015| 0.004         |
| 20      | 3.886e − 015| 3.997e − 015| 0.001         |
| 30      | 2.776e − 015| 3.997e − 015| 0.002         |
| 60      | 9.104e − 015| 1.288e − 014| 0.008         |
| 100     | 9.215e − 015| 1.110e − 014| 0.021         |

Table 5: Comparison of the absolute errors of present and AMD results at different iterations of \( u(x) \) for Example 1.

| \( x \) | ADM iterations | CPU time (sec) |
|---------|----------------|---------------|
| 0       |                |               |
| 0.25    |                |               |
| 0.50    |                |               |
| 0.75    |                |               |
| 1.00    |                |               |
| 1.25    |                |               |
| 1.50    |                |               |
| 1.75    |                |               |
| 2.00    |                |               |

various values of nodes \( (N) \). It can be seen that the values of nodes start from \( N = 10 \) to converge fully. The accuracy of the method does not improve when increasing grid points \( N \).

In Table 5, we have compared between the errors of present results for Example 4 corresponding to different values of collocation points \( N \) and those obtained by Maleknejad’s method (MAL) reported in [29] and Mean Value Theorem (MVT) reported in [25]. It can be seen that the present method converges rapidly compared to the
MAL and MVT for this example. Full convergence to the 7 decimal places’ accurate results reported by MAL and MVT is achieved for the values $N = 50$ and $N = 100$ while high convergence is achieved to the 16 decimal places’ accurate results for the present method for all values of $N \geq 12$. Table 6 shows the maximum absolute errors of $u(x)$ for Example 4 at different orders of approximation at selected nodes. The maximum absolute errors are generally very small. A high level of accuracy is achieved at very low orders of approximation. We remark that increasing the number of nodes (i.e., increasing $N$) does not result in a significant improvement in the accuracy of the approximation. Figure 3 displays the errors against the iterations of the solution for Example 4 when $N$ is fixed at 20. From Figure 3, it can be seen that the current solution rapidly converges, but the accuracy does not improve after 7 iterations.

In Table 7, we have compared between the errors of present results for Example 5 corresponding to different values of $x$ and those obtained by Single-Term Walsh Series (STWS) reported in [26] and Homotopy Perturbation Method (HPM) reported in [27]. Table 7 indicates that 15 decimal places’ accurate results are obtained using the present technique for large values of $x$ while only 2 and 6 decimal places’ accurate results are obtained using HPM and STWS, respectively. It is interesting to note from Table 7 that when $x$ increases, this would lead to a decrease in the absolute error for HPM and STWS, while increasing $x$ does not result in a decrease in the accuracy of the current approximation. This shows that the present method is more efficient than the method of STWS and HPM. Table 8 shows the maximum absolute errors $u_E(x)$ and $g_E(x)$ of the present method at different orders of approximation for different values of $N$ for Example 5. It can be seen that the maximum absolute errors are very small and further decrease with an increase in the order of the solution. Also, a
Table 7: Comparison between absolute errors of HPM, STWS, and present results of $u(x)$ and $g(x)$ at different values of $x$ for Example 5.

| $x$  | Errors in HPM [26] | Errors in STWS [27] | Errors in present method |
|------|---------------------|----------------------|--------------------------|
|      | $u(x)$ | $g(x)$ | $u(x)$ | $g(x)$ | $u(x)$ | $g(x)$ |
| 0.1  | 1.10e-007 | 2.40e-007 | 1.67e-006 | 8.31e-008 | 1.04e-015 | 6.66e-016 |
| 0.2  | 3.97e-006 | 1.09e-005 | 3.32e-006 | 3.30e-007 | 5.55e-016 | 5.55e-016 |
| 0.3  | 5.55e-005 | 1.19e-004 | 4.97e-006 | 7.33e-007 | 3.61e-015 | 3.33e-016 |
| 0.4  | 3.81e-004 | 6.27e-004 | 6.59e-006 | 1.28e-006 | 2.61e-015 | 2.22e-016 |
| 0.5  | 1.64e-003 | 2.20e-003 | 8.20e-006 | 1.95e-006 | 7.22e-016 | 2.22e-016 |
| 0.6  | 5.20e-003 | 5.96e-003 | 9.79e-006 | 2.73e-006 | 7.77e-016 | 4.44e-016 |
| 0.7  | 1.32e-002 | 1.33e-002 | 1.14e-005 | 3.58e-006 | 5.11e-015 | 1.11e-015 |
| 0.8  | 2.86e-002 | 2.59e-002 | 1.30e-005 | 4.47e-006 | 4.33e-015 | 1.55e-015 |
| 0.9  | 5.41e-002 | 4.49e-002 | 1.47e-005 | 5.36e-006 | 4.66e-015 | 1.67e-015 |
| 1.0  | 9.21e-002 | 7.07e-002 | 1.65e-005 | 6.22e-006 | 3.33e-015 | 1.78e-015 |

Table 8: Errors of $u(x)$ and $g(x)$ at different orders of approximation for different values of $N$ for Example 5.

| $N$  | $u_E(x)$ | $g_E(x)$ | $u_E(x)$ | $g_E(x)$ |
|------|----------|----------|----------|----------|
|      | 3rd iteration | 4th iteration | 5th iteration | 6th iteration |
| 10   | 6.024e-005 | 1.028e-010 | 1.145e-010 | 1.145e-010 |
| 12   | 6.027e-005 | 1.278e-010 | 1.841e-013 | 1.834e-013 |
| 14   | 6.026e-005 | 1.311e-010 | 4.663e-015 | 4.774e-015 |
| 18   | 6.026e-005 | 1.317e-010 | 4.885e-015 | 5.218e-015 |
| 22   | 6.026e-005 | 1.317e-010 | 5.967e-015 | 6.898e-015 |
| CPU time (sec) | 0.002304 | 0.002993 | 0.003679 | 0.004365 |
| 10   | 1.938e-005 | 6.811e-011 | 7.369e-011 | 7.369e-011 |
| 12   | 1.939e-005 | 3.733e-011 | 1.179e-013 | 1.177e-013 |
| 14   | 1.939e-005 | 3.901e-011 | 1.665e-015 | 1.776e-015 |
| 18   | 1.939e-005 | 3.929e-011 | 1.332e-015 | 1.332e-015 |
| 22   | 1.939e-005 | 3.929e-011 | 1.110e-015 | 1.110e-015 |
| CPU time (sec) | 0.002329 | 0.003601 | 0.004878 | 0.006196 |

Figure 4: Errors of $u_E(x)$ and $g_E(x)$ varied $N$ for Example 5.

A large increase in the number of nodes (increasing $N$) does not result in a significant improvement in the accuracy of the present solution. In Figure 4, we display the absolute errors of $u(x)$ and $g(x)$ against collocation points $N$ between the present results of Example 5 at the fifth iteration. From Figure 4, it is clear that the present method rapidly converges, and also, we notice that the accuracy does not improve for $N > 13$. 
5. Conclusion

In this paper, we have proposed a new method for solving linear and nonlinear Volterra integral equations. The technique has been constructed by introducing a new integral transformation together with Chebyshev pseudospectral method. The main difference between this collocation method and the other collocation methods is the replacement of the integral terms by a linear transformation using the Chebyshev differential matrix. The numerical results have been shown and compared with the exact solutions and other methods in the literature. These results are shown in the tables and figures for all cases considered in this work. To give a sense of the computational efficiency of the technique, the CPU computational time to generate the results is presented in the tables. It can be seen that high accuracy and less computational time (less than 0.02 seconds) substantiate our proposed method is a powerful numerical method for solving linear or nonlinear Volterra integral equations even for large values of grid points or iterations. The main conclusions emerging from this study may be summarized as follows:

(1) The method is more flexible than other numerical methods such as BPE, ANN, TQR, STWS, HPM, MAT, and MVT since it allows us to choose the integral operators in the integral equations in terms of the Chebyshev spectral collocation differentiation matrix.

(2) The proposed method is highly accurate and efficient, converges rapidly, and is sufficient to give good agreement with the exact solution.

(3) The suggested technique does not depend on the number of collocation points \( N \) and it requires a few numbers of collocation points to achieve high accuracy of results.

Finally, we note that the proposed method described above is a powerful method that is appropriate in solving Volterra integral equations and can be generalized for Volterra and Fredholm integral equations. In future, we intend to show that the proposed method can be extended to coupled Volterra integrodifferential equations.

Data Availability

No data were used to support this study.

Conflicts of Interest

The author declares that there are no conflicts of interest.

References

[1] P. K. Kythe and P. Puri, Computational Methods for Linear Integral Equations, University of New Orleans, New Orleans, LA, USA, 1992.

[2] A.-M. Wazwaz, “A comparison study between the modified decomposition method and the traditional methods for solving nonlinear integral equations,” Applied Mathematics and Computation, vol. 181, no. 2, pp. 1703–1712, 2006.

[3] A. M. Wazwaz, Linear and Nonlinear Integral Equations, Springer, Berlin, Germany, 2011.

[4] H. Brunner, “Volterra integral equations: an introduction to theory and applications,” in Cambridge Monographs on Applied and Computational Mathematics, vol. 30, Cambridge University Press, Cambridge, UK, 2017.

[5] I. Aziz, I. Siraj-Ul, and F. Khan, “A new method based on haar wavelet for the numerical solution of two-dimensional nonlinear integral equations,” Journal of Computational and Applied Mathematics, vol. 272, pp. 70–80, 2014.

[6] A. Doucet, A. M. Johansen, and V. B. Tadić, “On solving integral equations using Markov chain Monte Carlo methods,” Applied Mathematics and Computation, vol. 216, no. 10, pp. 2869–2880, 2010.

[7] J. R. Loh and C. Phang, “A new numerical scheme for solving system of Volterra integro-differential equation,” Alexandria Engineering Journal, vol. 57, no. 2, pp. 1117–1124, 2018.

[8] M. Higazy, S. Aggarwal, and T. A. Nofal, “Sawi decomposition method for Volterra integral equation with application,” Journal of Mathematics, vol. 2020, Article ID 6687134, 13 pages, 2020.

[9] A. H. Khater, R. S. Temsah, and M. M. Hassan, “A Chebyshev spectral collocation method for solving Burgers’-type equations,” Journal of Computational and Applied Mathematics, vol. 222, no. 2, pp. 333–350, 2008.

[10] D. Olmos and B. D. Shizgal, “A pseudospectral method of solution of Fisher’s equation,” Journal of Computational and Applied Mathematics, vol. 193, no. 1, pp. 219–242, 2006.

[11] M. Javidi, “Spectral collocation method for the solution of the generalized Burger-Fisher equation,” Applied Mathematics and Computation, vol. 174, no. 1, pp. 345–352, 2006.

[12] M. Javidi, “A numerical solution of the generalized Burger’s-Huxley equation by spectral collocation method,” Applied Mathematics and Computation, vol. 178, no. 2, pp. 338–344, 2006.

[13] M. Dehghan and F. Fakhar-Izadi, “Pseudospectral methods for Nagumo equation,” International Journal for Numerical Methods in Biomedical Engineering, vol. 27, no. 4, pp. 553–561, 2011.

[14] S. S. Motsa, “New algorithm for solving non-linear BVPs in heat transfer,” International Journal of Modeling, Simulation & Scientific Computing, vol. 2, no. 3, pp. 355–373, 2011.

[15] F. G. Awad, P. Sibanda, S. S. Motsa, and O. D. Makinde, “Convection from an inverted cone in a porous medium with cross-diffusion effects,” Computers & Mathematics with Applications, vol. 61, no. 5, pp. 1431–1441, 2011.

[16] Z. G. Makukula, P. Sibanda, and S. S. Motsa, “A novel numerical technique for two-dimensional laminar flow between two moving porous walls,” Mathematical Problems in Engineering, vol. 2010, Article ID 528956, 15 pages, 2010.

[17] S. S. Motsa and S. Shateyi, “A new approach for the solution of three-dimensional magnetohydrodynamic rotating flow over a shrinking sheet,” Mathematical Problems in Engineering, vol. 2010, Article ID 586340, 2010.

[18] C. Canuto, M. Y. Hussaini, A. Quarteroni, and T. A. Zang, Spectral Methods in Fluid Dynamics, Springer-Verlag, Berlin, Germany, 1988.

[19] W. S. Don and A. Solomonoff, “Accuracy and speed in computing the Chebyshev collocation derivative,” SIAM Journal on Scientific Computing, vol. 16, no. 6, pp. 1253–1268, 1995.

[20] L. N. Trefethen, Spectral methods in MATLAB, SIAM, Philadelphia, PA, USA, 2000.
[21] O. A. Agbolade and T. A. Anake, “Solutions of first-order Volterra type linear integrodifferential equations by collocation method,” *Journal of Applied Mathematics*, vol. 2017, Article ID 1510267, 5 pages, 2017.

[22] A. Howard and R. Chris, *Elementary Linear Algebra: applications Version*, Wiley, Hoboken, NJ, USA, 11th edition, 2013.

[23] Y. H. Youssri and R. M. Hafez, “Chebyshev collocation treatment of Volterra-Fredholm integral equation with error analysis,” *Arabian Journal of Mathematics*, vol. 9, no. 2, pp. 471–480, 2020.

[24] A. Jafarian, S. Measoomy Nia, A. K. Golmankhanh, and D. Baleanu, “Numerical solution of linear integral equations system using the Bernstein collocation method,” *Advances in Difference Equations*, vol. 2013, pp. 1–15, Article ID 2013, 2013.

[25] P. De Angelis, R. De Marchis, and A. L. Martire, “A new numerical method for a class of Volterra and Fredholm integral equations,” *Journal of Computational and Applied Mathematics*, vol. 379, Article ID 112944, 2020.

[26] J. Biazar and H. Ghazvini, “He’s homotopy perturbation method for solving systems of Volterra integral equations of the second kind,” *Chaos, Solitons & Fractals*, vol. 39, no. 2, pp. 770–777, 2009.

[27] R. Chandra Guru Sekar and K. Murugesan, “STWS approach for Hammerstein system of nonlinear Volterra integral equations of the second kind,” *International Journal of Computer Mathematics*, vol. 94, no. 9, pp. 1867–1878, 2017.

[28] A. Jafarian and S. Measoomy Nia, “Utilizing feed-back neural network approach for solving linear Fredholm integral equations system,” *Applied Mathematical Modelling*, vol. 37, no. 7, pp. 5027–5038, 2013.

[29] K. Maleknejad, P. Torabi, and S. Sauter, “Numerical solution of a non-linear Volterra integral equation,” *Vietnam Journal of Mathematics*, vol. 44, no. 1, pp. 5–28, 2016.