A Solution For Volterra Integral Equations of the First Kind Based on Bernstein Polynomials

M. Mohamadi *, E. Babolian ††, S. A. Yousefi §

Received Date: 2017-03-03 Revised Date: 2017-07-08 Accepted Date: 2017-10-22

Abstract

In this paper, we present a new computational method to solve Volterra integral equations of the first kind based on Bernstein polynomials. In this method, using operational matrices turn the integral equation into a system of equations. The computed operational matrices are exact and new. The comparisons show this method is acceptable. Moreover, the stability of the proposed method is studied.

Keywords : Volterra integral equation; Bernstein polynomials; Operational matrices; Transformation matrices.

1 Introduction

Integral equations are an important topic in mathematics, physics and engineering sciences. Many researchers spent their time to find a solution for these equations. Their efforts led to many numerical and analytical methods like Neumann series, Nystrom method, expansion method, collocation methods, residual methods, Galerkin methods, homotopy methods, perturbation method, the variational iteration method, the Laplace transform method, the Adomian decomposition method, the series solution method, and the direct computation methods [9, 10, 11, 1, 8, 27, 33]. Recently polynomials play a fundamental role in some valid numerical methods. In some of these approaches integral equations convert to a linear or nonlinear system and by solving the system the approximate solution of the integral equation will be found. Maleknejad et al. [17] and Mandal and Bhattacharya used Bernstein polynomials in approximation techniques [19], Shahsavaran employed Block Pulse functions and Taylor Expansion method [29]. Taylor polynomials were also used by Bellour and Rawashdeh [7] and Wang [32] with computer algebra. These polynomials have been also used for solving Fredholm integral equations of the second kind by Shirin and Islam [30]. Babolian and Delves have described an augmented Galerkin technique for the numerical solution of the first kind Fredholm integral equations [2]. In [12], a numerical solution of Fredholm integral equations of the first kind via piecewise interpolation is proposed. Lewis studied a computational method to solve first kind integral equations[16], also, for more researches see [28, 13, 24, 31, 25, 3, 4, 5, 34, 35, 18, 36]
In the next section, we review Bernstein polynomials and some basic theorems and concepts. In section 3, transformation matrices are defined. In subsections of section 4, operational matrices are computed. These matrices are new and exact. In section 5, the integral equations are changed to a linear or nonlinear system. Some illustrative examples, in section 6, show accuracy and exactness of method. Then, a comparison between our method with a direct method and an expansion-iterative method is presented in section 7. Finally, in section 8, effect of a random noise on data function is investigated.

2 Preliminaries

Definition 2.1 Suppose $m$ is a positive integer number, BPs of degree $m$ on interval $[a, b]$ are defined as follows:

$B_{i,m}(x) = \binom{m}{i} \frac{(x-a)^i (b-x)^{m-i}}{(b-a)^m}, \quad 0 \leq i \leq m.$

Also, $B_{i,m}(x) = 0$ if $i < 0$ or $i > m$. For convenience we consider $[a, b] = [0, 1]$ , namely $B_{i,m}(x) = \binom{m}{i} x^i (1-x)^{m-i}, \quad 0 \leq i \leq m.$

We denote $\Phi_m$, an $m$-column vector as follows:

$\Phi_m(x) = [\phi_0(x) \quad \phi_1(x) \quad \cdots \quad \phi_m(x)]^T$, where $\phi_i(x) = B_{i,m}(x), \quad 0 \leq i \leq m.$

The BPs have many interesting properties [14, 26, 20, 22]. But, here some of them that are useful in our work are stated.

$P1)$ $B_{i,m}(x)B_{j,m}(x) = \frac{\binom{m}{i} \binom{m}{j}}{\binom{2m}{i+j}}B_{i+j,2m}(x), 0 \leq i, j \leq m.$

$P2)$ $B_{i,m}(\frac{m}{i+1}) = B_{i,m+1}(\frac{m+1}{i+1}) \quad \frac{B_{i+1,m+1}(\frac{m+1}{i+1})}{i = 0, ..., m}.$

The following theorems are a fundamental tool that justifies the use of polynomials.

Theorem 2.1 [15]. Suppose $H = L^2([a, b])$ is a Hilbert space with the inner product defined by $(f, g) = \int_a^b f(t)g(t)dt$ and also, $Y = \text{Span} \{B_{0,m}(x), B_{1,m}(x), ..., B_{m,m}(x)\}$ be the span space by Bernstein polynomials of degree $m$. Let $f$ be an arbitrary element in $H$. Since $Y$ is a finite dimensional and closed subspace, it is a complete subset of $H$. So, $f$ has the unique best approximation out of $Y$ such that $\exists y_0 \in Y ; \forall y \in Y : \| f - y_0 \|_2 \leq \| f - y \|_2$.

Therefore, there are the unique coefficients $\alpha_j, 0 \leq j \leq m$, such that $f(t) \approx y_0(t) = \sum_{j=0}^{m} \alpha_j B_{j,m}(t) = \alpha^T \Phi_m$

where, $\alpha = [\alpha_0 \quad \alpha_1 \cdots \quad \alpha_m]^T$, can be obtained by $\alpha = \langle f(t), \Phi_m(t) \rangle \langle \Phi_m(t), \Phi_m(t) \rangle$ such that $\langle f, \Phi_m(t) \rangle = \int_a^b f(t)\Phi_m(t)dt$. In the above theorem we denote $Q = \langle \Phi_m(t), \Phi_m(t) \rangle$ as dual matrix. Furthermore, it is easy to see $Q_{i,j} = \frac{\left(m\right)_{i-1} \left(m\right)_{j-1}}{(2m+1)\left(i+j-2\right)}$, $i, j = 1, ..., m + 1$.

Next theorem indicates dual matrix is symmetric and invertible.

Theorem 2.2 [15]. Elements $y_0, y_1, ..., y_n$ of a Hilbert space $H$ constitute a linearly independent set in $H$ if and only if $G(y_0, y_1, ..., y_n) \neq 0$.

Where $G(y_0, y_1, ..., y_n)$ is the Gram determinant of $y_0, y_1, ..., y_n$ defined by $G(y_0, y_1, ..., y_n) = \begin{vmatrix} \langle y_0, y_0 \rangle & \langle y_0, y_1 \rangle & \cdots & \langle y_0, y_n \rangle \\ \langle y_1, y_0 \rangle & \langle y_1, y_1 \rangle & \cdots & \langle y_1, y_n \rangle \\ \cdots & \cdots & \cdots & \cdots \\ \langle y_n, y_0 \rangle & \langle y_n, y_1 \rangle & \cdots & \langle y_n, y_n \rangle \end{vmatrix}$.

For a 2-dimensional function $k(x, t) \in L^2([0, 1] \times [0, 1])$, it can be similarly expanded with respect to BPs such as $k(x, t) \simeq \Phi^T(t)K\Phi(x)$, and $K$ is the BP coefficient matrix with $K_{ij} = Q^{-1}(Q^{-1} \int_0^1 (Q^{-1} \int_0^1 k(x, t)\phi_i(t)dt)\phi_j(x)dx), 0 \leq i, j \leq m$.

3 Transformation matrices

Transformation matrix is used to change the dimension of the problem. In other words, these matrix can convert $\Phi_m$ to $\Phi_n$ and vice versa. Suppose $m$ is less than $n$, $T_m^n$ is an $(m + 1) \times (n + 1)$ matrix, called increasing transformation matrix, that converts $\Phi_m$ to $\Phi_n$. In other words, $\Phi_m = T_m^n \Phi_n$. The increasing transformation matrix can be computed as follows:

$[T_m^n]_{i,j} = \begin{cases} 0 & \text{if } i < j \text{ or } j > i + k \\ \frac{\left(m\right)_{i-1} \left(k\right)_{j-1}}{\left(m+k\right)_{i+j-2}} & \text{otherwise} \end{cases}$
It is sufficient to use \( P2 \), \( k \) times where \( k = n - m \). Also, decreasing transformation matrix is an \((n+1) \times (m+1)\) matrix which is shown by \( T_m^n \) and converts \( \Phi_n \) to \( \Phi_m \) where \( n \) is greater than \( m \). In other words, \( \Phi_n = T_m^n \Phi_m \). The \( i \) th row of decreasing transformation matrix can be calculated as follows:

\[
\frac{1}{m+n+1} \begin{bmatrix}
\left(\frac{1}{m+1}\right) & \left(\frac{1}{m+2}\right) & \cdots & \left(\frac{1}{m+n}\right)
\end{bmatrix} Q^{-1},
\]

\( i = 0, \ldots, n \).

## 4 Operational Matrices

Operational matrix is a matrix that works on basis like an operator, in other words, if \( A \) is an operator an operational matrix is a matrix like \( P \) such that \( A(\Phi) \approx P \Phi \).

### 4.1 Operational matrix of integration

**Lemma 4.1** Let \( M \) be operational matrix of integration and \( \Phi_m(x) = [\phi_0(x) \phi_1(x) \cdots \phi_m(x)]^T \), then

\[
\int_0^x \Phi_m(x)dx = M \Phi_m(x). \tag{4.1}
\]

**Proof.** With a simple calculation can be seen

\[
B_{i,m}(x) = \int_0^x m(B_{i-1,m-1}(t) - B_{i,m-1}(t))dt
\]

Assume \( 0 \leq k \leq m \),

\[
\sum_{i=k}^m B_{i,m}(x) = \sum_{i=k}^m \int_0^x m(B_{i-1,m-1}(t) - B_{i,m-1}(t))dt = m \int_0^x B_{k,m-1}(t)dt.
\]

Therefore,

\[
\int_0^x B_{k,m}(t)dt = \frac{1}{m+1} \sum_{i=k+1}^{m+1} B_{i,m-1}(x) = M_k^T \Phi_{m+1}
\]

where

\[
M_k = \frac{1}{m+1} \begin{bmatrix}
k+1 \\
0, \cdots, 0, 1, \cdots, 1 \\
\end{bmatrix}^T,
\]

it is obvious, \( im = \begin{bmatrix} M_0^T \\ M_1^T \\ \vdots \\ M_m^T \end{bmatrix} \) is an \((m+2) \times (m+1)\) matrix. Accordingly \( M = imT_{m+1}^n \).

### 4.2 Operational matrix of product

**Lemma 4.2** Let \( C \) be an \((m+1) \times (m+1)\) matrix then,

\[
\Phi_m^T C \Phi_m = \hat{C}^T \Phi_{2m} \tag{4.2}
\]

where \( \hat{C}_k = \sum_{j=0}^{k} \frac{(m+j)}{(2m)} C_{k-j,j} \) \( k = 0, \ldots, 2m \).

**Proof.** Let \( \phi_i^*(x) = B_{i,2m}(x) \), for \( i = 0, \ldots, 2m \),

\[
\Phi_m^T C \Phi_m = \sum_{i=0}^m \sum_{j=0}^m c_{i,j} \phi_i \phi_j
\]

using \( P1 \) gives

\[
\Phi_m^T C \Phi_m = \sum_{i=0}^m \sum_{j=0}^m \frac{(m)}{(2m)} C_{i-j,j} \phi_i \phi_j
\]

\[
= \begin{bmatrix}
\frac{1}{2} & \frac{(m)}{(2m))} & C_{1-j,1} \\
\frac{(m)}{(2m)} & \frac{(m)}{(2m)} & C_{2-j,2} \\
\vdots & \vdots & \ddots \\
\frac{(m)}{(2m)} & \frac{(m)}{(2m)} & C_{m,m}
\end{bmatrix}
\]

\[
\phi^* = \hat{C}^T \Phi_{2m}(x).
\]

**Lemma 4.3** Let \( u \) be an arbitrary \((m+1)\)-vector then,

\[
\Phi_m \Phi_m^T u = \tilde{u} \Phi_2m, \tag{4.3}
\]

where \( \tilde{u} \) is an \((m+1) \times (2m+1)\) matrix with elements
Table 1: Results of example 6.1 in some special points.

<table>
<thead>
<tr>
<th>$x$</th>
<th>$m = 8$</th>
<th>$e_8(x)$</th>
<th>$m = 10$</th>
<th>$e^{10}(x)$</th>
<th>exact solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.99274777</td>
<td>7.25 x 10^{-3}</td>
<td>1.0049204</td>
<td>4.92 x 10^{-3}</td>
<td>1.00</td>
</tr>
<tr>
<td>0.1</td>
<td>0.90237509</td>
<td>2.46 x 10^{-3}</td>
<td>0.90703960</td>
<td>3.10 x 10^{-3}</td>
<td>0.904837418</td>
</tr>
<tr>
<td>0.2</td>
<td>0.82088162</td>
<td>2.15 x 10^{-3}</td>
<td>0.81635890</td>
<td>2.37 x 10^{-3}</td>
<td>0.818730753</td>
</tr>
<tr>
<td>0.3</td>
<td>0.73878286</td>
<td>2.03 x 10^{-3}</td>
<td>0.74336086</td>
<td>2.54 x 10^{-3}</td>
<td>0.740818220</td>
</tr>
<tr>
<td>0.4</td>
<td>0.67145627</td>
<td>1.13 x 10^{-3}</td>
<td>0.66769187</td>
<td>2.63 x 10^{-3}</td>
<td>0.670320046</td>
</tr>
<tr>
<td>0.5</td>
<td>0.60695828</td>
<td>4.54 x 10^{-4}</td>
<td>0.60840722</td>
<td>1.87 x 10^{-3}</td>
<td>0.606530659</td>
</tr>
<tr>
<td>0.6</td>
<td>0.54718461</td>
<td>1.62 x 10^{-3}</td>
<td>0.54837489</td>
<td>4.37 x 10^{-4}</td>
<td>0.548811636</td>
</tr>
<tr>
<td>0.7</td>
<td>0.49821291</td>
<td>1.62 x 10^{-3}</td>
<td>0.49561978</td>
<td>9.65 x 10^{-4}</td>
<td>0.496585303</td>
</tr>
<tr>
<td>0.8</td>
<td>0.448407286</td>
<td>9.21 x 10^{-4}</td>
<td>0.45118619</td>
<td>1.85 x 10^{-3}</td>
<td>0.449328964</td>
</tr>
<tr>
<td>0.9</td>
<td>0.40743778</td>
<td>8.68 x 10^{-4}</td>
<td>0.40466694</td>
<td>1.90 x 10^{-3}</td>
<td>0.406569659</td>
</tr>
</tbody>
</table>

Table 2: Results of example 6.2 in some special points.

<table>
<thead>
<tr>
<th>$x$</th>
<th>$m = 5$</th>
<th>$e_5(x)$</th>
<th>$m = 10$</th>
<th>$e^{10}(x)$</th>
<th>exact solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.99945089</td>
<td>5.91 x 10^{-4}</td>
<td>1.0000</td>
<td>0.000</td>
<td>1.00</td>
</tr>
<tr>
<td>0.1</td>
<td>0.90500947</td>
<td>1.72 x 10^{-4}</td>
<td>0.90483741</td>
<td>1 x 10^{-10}</td>
<td>0.904837418</td>
</tr>
<tr>
<td>0.2</td>
<td>0.81849734</td>
<td>2.33 x 10^{-4}</td>
<td>0.81873075</td>
<td>1 x 10^{-10}</td>
<td>0.818730753</td>
</tr>
<tr>
<td>0.3</td>
<td>0.74075709</td>
<td>6.11 x 10^{-5}</td>
<td>0.74081822</td>
<td>1.5 x 10^{-11}</td>
<td>0.740818220</td>
</tr>
<tr>
<td>0.4</td>
<td>0.67060120</td>
<td>2.81 x 10^{-4}</td>
<td>0.67060120</td>
<td>3.6 x 10^{-11}</td>
<td>0.670320046</td>
</tr>
<tr>
<td>0.5</td>
<td>0.60669652</td>
<td>1.65 x 10^{-4}</td>
<td>0.60653065</td>
<td>2 x 10^{-10}</td>
<td>0.606530659</td>
</tr>
<tr>
<td>0.6</td>
<td>0.54849402</td>
<td>3.17 x 10^{-4}</td>
<td>0.54881163</td>
<td>1 x 10^{-10}</td>
<td>0.548811636</td>
</tr>
<tr>
<td>0.7</td>
<td>0.49620362</td>
<td>3.81 x 10^{-4}</td>
<td>0.49658530</td>
<td>1 x 10^{-10}</td>
<td>0.496585303</td>
</tr>
<tr>
<td>0.8</td>
<td>0.4498142362</td>
<td>4.85 x 10^{-4}</td>
<td>0.44932896</td>
<td>1.9 x 10^{-11}</td>
<td>0.449328964</td>
</tr>
<tr>
<td>0.9</td>
<td>0.407190358</td>
<td>5.89 x 10^{-4}</td>
<td>0.40656965</td>
<td>1 x 10^{-10}</td>
<td>0.406569657</td>
</tr>
</tbody>
</table>

Table 3: Results of example 6.3 in some special points.

<table>
<thead>
<tr>
<th>$x$</th>
<th>$m = 8$</th>
<th>$e_8(x)$</th>
<th>$m = 12$</th>
<th>$e^{12}(x)$</th>
<th>Exact solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>-0.03207883</td>
<td>3.20 x 10^{-2}</td>
<td>0.00008978829</td>
<td>8.97 x 10^{-5}</td>
<td>0.000000000</td>
</tr>
<tr>
<td>0.1</td>
<td>0.191776504</td>
<td>7.89 x 10^{-3}</td>
<td>0.1996870087</td>
<td>2.01 x 10^{-5}</td>
<td>0.199666833</td>
</tr>
<tr>
<td>0.2</td>
<td>0.405304229</td>
<td>7.96 x 10^{-3}</td>
<td>0.397391521</td>
<td>1.95 x 10^{-5}</td>
<td>0.397386616</td>
</tr>
<tr>
<td>0.3</td>
<td>0.581192769</td>
<td>9.84 x 10^{-3}</td>
<td>0.5910549712</td>
<td>1.45 x 10^{-5}</td>
<td>0.5910404134</td>
</tr>
<tr>
<td>0.4</td>
<td>0.786819169</td>
<td>7.98 x 10^{-3}</td>
<td>0.7788219798</td>
<td>1.47 x 10^{-5}</td>
<td>0.7788366846</td>
</tr>
<tr>
<td>0.5</td>
<td>0.959541169</td>
<td>6.90 x 10^{-4}</td>
<td>0.9588688383</td>
<td>1.77 x 10^{-5}</td>
<td>0.9588510772</td>
</tr>
<tr>
<td>0.6</td>
<td>1.115211269</td>
<td>1.41 x 10^{-2}</td>
<td>1.129263004</td>
<td>2.19 x 10^{-5}</td>
<td>1.1292849470</td>
</tr>
<tr>
<td>0.7</td>
<td>1.316221169</td>
<td>2.77 x 10^{-2}</td>
<td>1.288461250</td>
<td>2.58 x 10^{-5}</td>
<td>1.2884357340</td>
</tr>
<tr>
<td>0.8</td>
<td>1.391721169</td>
<td>4.29 x 10^{-2}</td>
<td>1.434688846</td>
<td>2.33 x 10^{-5}</td>
<td>1.4347121820</td>
</tr>
<tr>
<td>0.9</td>
<td>1.637921169</td>
<td>7.12 x 10^{-2}</td>
<td>1.566617164</td>
<td>3.66 x 10^{-5}</td>
<td>1.5666538190</td>
</tr>
</tbody>
</table>

Table 4: Results of example 6.4 in some special points.

<table>
<thead>
<tr>
<th>$x$</th>
<th>$m = 4$</th>
<th>$e_4(x)$</th>
<th>$m = 8$</th>
<th>$e_8(x)$</th>
<th>Exact solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.00</td>
<td>0.0084887020</td>
<td>8.4 x 10^{-3}</td>
<td>0.000000046</td>
<td>4.6 x 10^{-9}</td>
<td>0.000</td>
</tr>
<tr>
<td>0.15</td>
<td>0.1478336203</td>
<td>2.1 x 10^{-3}</td>
<td>0.1500000083</td>
<td>8.3 x 10^{-9}</td>
<td>0.150</td>
</tr>
<tr>
<td>0.30</td>
<td>0.3058088085</td>
<td>5.8 x 10^{-3}</td>
<td>0.300000032</td>
<td>3.2 x 10^{-9}</td>
<td>0.300</td>
</tr>
<tr>
<td>0.45</td>
<td>0.4463272725</td>
<td>3.6 x 10^{-3}</td>
<td>0.4499999969</td>
<td>3.1 x 10^{-9}</td>
<td>0.450</td>
</tr>
<tr>
<td>0.60</td>
<td>0.5904142554</td>
<td>9.5 x 10^{-3}</td>
<td>0.5999999962</td>
<td>3.8 x 10^{-9}</td>
<td>0.600</td>
</tr>
<tr>
<td>0.75</td>
<td>0.7674230210</td>
<td>1.6 x 10^{-3}</td>
<td>0.7500000045</td>
<td>4.5 x 10^{-9}</td>
<td>0.750</td>
</tr>
<tr>
<td>0.90</td>
<td>0.8934282466</td>
<td>6.5 x 10^{-3}</td>
<td>0.9000001447</td>
<td>1.4 x 10^{-7}</td>
<td>0.900</td>
</tr>
</tbody>
</table>
Table 5: Comparison between BPs method and block-pulse methods in example 6.1.

<table>
<thead>
<tr>
<th>method</th>
<th>Mid-points, ( k = 32 )</th>
<th>Mid-points, ( k = 64 )</th>
<th>Ten points, ( k = 32 )</th>
<th>Ten points, ( k = 64 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Direct method</td>
<td>( 3.3 \times 10^{-3} )</td>
<td>( 1.6 \times 10^{-3} )</td>
<td>( 5.9 \times 10^{-3} )</td>
<td>( 2.9 \times 10^{-3} )</td>
</tr>
<tr>
<td>expansion-iterative</td>
<td>( 6.6 \times 10^{-4} )</td>
<td>( 1.9 \times 10^{-4} )</td>
<td>( 5.2 \times 10^{-3} )</td>
<td>( 2.6 \times 10^{-3} )</td>
</tr>
<tr>
<td>BPs method</td>
<td>( 7.70 \times 10^{-4} )</td>
<td>( 1.78 \times 10^{-3} )</td>
<td>( 2.62 \times 10^{-3} )</td>
<td></td>
</tr>
</tbody>
</table>

Table 6: Comparison between BPs method and block-pulse methods in example 6.3.

<table>
<thead>
<tr>
<th>method</th>
<th>Mid-points, ( k = 64 )</th>
<th>Mid-points, ( k = 128 )</th>
<th>Ten points, ( k = 64 )</th>
<th>Ten points, ( k = 128 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Direct method</td>
<td>( 5.2 \times 10^{-3} )</td>
<td>( 2.6 \times 10^{-3} )</td>
<td>( 8.2 \times 10^{-3} )</td>
<td>( 4.1 \times 10^{-3} )</td>
</tr>
<tr>
<td>expansion-iterative</td>
<td>( 4.9 \times 10^{-4} )</td>
<td>( 1.4 \times 10^{-4} )</td>
<td>( 6.5 \times 10^{-3} )</td>
<td>( 3.3 \times 10^{-3} )</td>
</tr>
<tr>
<td>BPs method</td>
<td>( 1.3 \times 10^{-3} )</td>
<td>( 2.5 \times 10^{-4} )</td>
<td>( 3.96 \times 10^{-5} )</td>
<td></td>
</tr>
</tbody>
</table>

Table 7: Effect of noise on example 7.1.

<table>
<thead>
<tr>
<th>( x )</th>
<th>( m = 4 )</th>
<th>( m = 4, \varepsilon = 0.01 )</th>
<th>( m = 4, \varepsilon = 0.02 )</th>
<th>( m = 4, \varepsilon = 0.03 )</th>
<th>Exact solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>0.00000</td>
<td>-0.00312608987</td>
<td>-0.00311339313</td>
<td>-0.00313694445</td>
<td>0.000000</td>
</tr>
<tr>
<td>0.1</td>
<td>0.01000</td>
<td>0.01100083307</td>
<td>0.01097076095</td>
<td>0.01103789223</td>
<td>0.010000</td>
</tr>
<tr>
<td>0.2</td>
<td>0.04000</td>
<td>0.0385449585</td>
<td>0.03877979162</td>
<td>0.0400000000</td>
<td>0.040000</td>
</tr>
<tr>
<td>0.3</td>
<td>0.09000</td>
<td>0.08881385614</td>
<td>0.08881276142</td>
<td>0.0900000000</td>
<td>0.090000</td>
</tr>
<tr>
<td>0.4</td>
<td>0.16000</td>
<td>0.1599768635</td>
<td>0.15997897612</td>
<td>0.1600000000</td>
<td>0.160000</td>
</tr>
<tr>
<td>0.5</td>
<td>0.25000</td>
<td>0.2494161014</td>
<td>0.2494161014</td>
<td>0.2500000000</td>
<td>0.250000</td>
</tr>
<tr>
<td>0.6</td>
<td>0.36000</td>
<td>0.3569125103</td>
<td>0.3569125103</td>
<td>0.3600000000</td>
<td>0.360000</td>
</tr>
<tr>
<td>0.7</td>
<td>0.49000</td>
<td>0.4853278320</td>
<td>0.4853278320</td>
<td>0.4900000000</td>
<td>0.490000</td>
</tr>
<tr>
<td>0.8</td>
<td>0.64000</td>
<td>0.637159494</td>
<td>0.637159494</td>
<td>0.6400000000</td>
<td>0.640000</td>
</tr>
<tr>
<td>0.9</td>
<td>0.81000</td>
<td>0.8073183571</td>
<td>0.8073183571</td>
<td>0.8100000000</td>
<td>0.810000</td>
</tr>
<tr>
<td>1.0</td>
<td>1.00000</td>
<td>0.971787716</td>
<td>0.971787716</td>
<td>1.0000000000</td>
<td>1.000000</td>
</tr>
</tbody>
</table>

Table 8: Effect of noise on example 7.2.

<table>
<thead>
<tr>
<th>( x )</th>
<th>( m = 8 )</th>
<th>( m = 8, \varepsilon = 0.01 )</th>
<th>( m = 8, \varepsilon = 0.02 )</th>
<th>( m = 8, \varepsilon = 0.03 )</th>
<th>Exact solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>4.60 \times 10^{-8}</td>
<td>0.005199858864</td>
<td>0.003101836424</td>
<td>0.00313694445</td>
<td>0.000000</td>
</tr>
<tr>
<td>0.1</td>
<td>0.100000018</td>
<td>0.1063214102</td>
<td>0.1042278036</td>
<td>0.1027686108</td>
<td>0.100000</td>
</tr>
<tr>
<td>0.2</td>
<td>0.199999982</td>
<td>0.2062665325</td>
<td>0.2041697813</td>
<td>0.2041278280</td>
<td>0.200000</td>
</tr>
<tr>
<td>0.3</td>
<td>0.300000003</td>
<td>0.3063655647</td>
<td>0.3042692155</td>
<td>0.302127705</td>
<td>0.300000</td>
</tr>
<tr>
<td>0.4</td>
<td>0.400000025</td>
<td>0.4061393977</td>
<td>0.4040429008</td>
<td>0.4012588448</td>
<td>0.400000</td>
</tr>
<tr>
<td>0.5</td>
<td>0.499999963</td>
<td>0.5063383726</td>
<td>0.5042451973</td>
<td>0.5047514740</td>
<td>0.500000</td>
</tr>
<tr>
<td>0.6</td>
<td>0.599999995</td>
<td>0.6046074962</td>
<td>0.6043110989</td>
<td>0.6021586937</td>
<td>0.600000</td>
</tr>
<tr>
<td>0.7</td>
<td>0.700000067</td>
<td>0.7058532921</td>
<td>0.7038894241</td>
<td>0.7123865047</td>
<td>0.700000</td>
</tr>
<tr>
<td>0.8</td>
<td>0.799999897</td>
<td>0.8065711616</td>
<td>0.8044714803</td>
<td>0.7994762500</td>
<td>0.800000</td>
</tr>
<tr>
<td>0.9</td>
<td>0.900000144</td>
<td>0.9058954184</td>
<td>0.9037997147</td>
<td>0.9145590776</td>
<td>0.900000</td>
</tr>
</tbody>
</table>

Proof. Property P1 implies

\[
\tilde{u}_{i,j} = \begin{cases} 
0 & \text{if } j < i \text{ and } j > i + m \\
\left( \frac{m}{j+i} \right)_{(m)} \binom{m}{j} u_j & \text{otherwise}
\end{cases}
\]

for \( i,j = 0, \ldots, m \).

\[
\Phi_m(x)\Phi_m^T(x)u = \left[ \sum_{j=0}^{m} \binom{m}{j} \Phi_j^* \sum_{j=0}^{m} \binom{m}{j} \Phi_j^* \sum_{j=0}^{m} \binom{m}{j} \Phi_j^* \right] u_j.
\]
Now, the entry of the above matrix can be rewritten as follows:

\[
\sum_{j=0}^{m} \left( \frac{m!}{(m-j)!} \right) u_j \phi_{j+i-1}^* = \\
\begin{bmatrix}
\vdots \\
0 \cdots 0 \\
\vdots \\
0 \cdots 0 \\
\end{bmatrix}
\begin{bmatrix}
\phi_0 \\
\phi_1 \\
\vdots \\
\phi_{m+1} \\
\end{bmatrix}
\]

5 Solution of integral equation of the first kind

In this section, with respect to operational matrices and function approximation, integral equation converts to a system of equations.

5.1 Linear Volterra integral equation of the first kind

Consider the following Volterra integral equation of the first kind

\[
f(x) = \int_0^x k(x,t)u(t)dt
\]

where \(f\) and \(k\) are known but \(u\) is not. Moreover, \(k(x,t) \in L^2([0,1] \times [0,1])\) and \(f(t) \in L^2([0,1])\). Approximating functions \(f, u\) and \(k\) with respect to BPs gives

\[
f(x) = F^T \Phi_m(x) = \Phi_m^T(x)F
\]

\[
u(t) = U^T \Phi_m(t) = \Phi^T_m(t)U
\]

\[k(x,t) \simeq \Phi^T_m(x)K\Phi_m(t)
\]

where the vectors \(F, U\) and matrix \(K\) are BPs coefficients of \(f(x), u(t)\) and \(k(x,t)\) respectively. Now, replacing (5.5) into (5.4) gives:

\[
F^T \Phi_m(x) = \int_0^x \Phi^T_m(x)K\Phi_m(t)\Phi^T_m(t)U dt
\]

\[= \Phi_m^T(x)K \int_0^x \Phi_m(t)\Phi^T_m(t)U dt.
\]

Using (4.3) follows:

\[
F^T \Phi_m(x) = \Phi_m^T(x)K \int_0^x \tilde{U} \Phi_{2m}(t) dt
\]

\[= \Phi_m^T(x)K \tilde{U} \int_0^x \Phi_{2m}(t) dt.
\]

Using operational matrix of integration \(M\), in Eq. (5.6) gives:

\[
F^T \Phi_m(x) = \Phi_m^T(x)K \tilde{U} M \Phi_{2m}(t) dt.
\]

Let \(U^* = \tilde{K} \tilde{U} M \Phi_{2m}(t)\), where \(U^*\) is an \((m+1) \times (m+1)\) matrix. Eq. (5.7) changes to:

\[
F^T \Phi_m(x) = \Phi_m^T(x)U^* \Phi_m(x).
\]

Using Eq. (4.2) in (5.8) gives:

\[
F^T \Phi_m(x) = \Phi_m^T(x)U^* \Phi_m(x) = U^* T_{2m}\Phi_m(x).
\]

Using decreasing transformation matrix \(T_{2m}\), gives the final system:

\[
\tilde{U} = F,
\]

where \(U^T = U^* T_{2m}\).

5.2 Nonlinear Volterra integral equation of the first kind

Consider the following nonlinear Volterra integral equation

\[
f(x) = \int_0^x k(x,t)g(u(t))dt
\]

Put \(w(t) = g(u(t))\) and Subsequently \(w(t) = W^T \Phi_m(t)\). Where \(W\) is an unknown \((m+1)\)-vector. Following the same procedure, final system is as follows: \(W = F\). Finally, \(u(x) = g^{-1}(w(x))\) is the desired solution.

6 Numerical examples

To show the efficiency of the proposed numerical method, we implement it on some Volterra integral equations. For every example we use a table that shows exact solution, our approximation and absolute errors in some points. In the following examples, the absolute error is used to check the accuracy. The amount is far more than other computational errors like mean absolute error.

Example 6.1 \(u(x) = e^{-x}\) is the exact solution of the following Volterra integral equation of the first kind \(x e^x = \int_0^x e^{x+t} u(t)dt\). Numerical solution of this equation and its errors are shown in table 1.

Example 6.2 Consider \(x = \int_0^5 (x+t-1)u(y)dt\) with the exact solution \(u(x) = e^{-x}\). The table 2 shows approximation solutions, error and exact solution in some points.

Example 6.3 \(u(x) = 2 \sin x\) is the exact solution of the following Volterra integral equation of the first kind \(x \sin x = \int_0^x \cos (x-t)u(t)dt\). Result of example 6.3 are shown in table 3.
Example 6.4 \[ x = \int_0^x (x-t+1) e^{-u(t)} dt \] is a non-linear Volterra integral equation of the first kind with exact solution \( u(x) = x \). Table 4 shows the exact solution, approximation solutions and absolute errors at some points.

Now, we compare our method with a direct method to solve Volterra integral equation of the first kind using operational matrix with block-pulse functions [6] and an expansion-iterative method based on the block-pulse functions [21]. Consider example 6.1, Table 5 shows the mean absolute errors for direct method and expansion-iterative method for two values of \( k \), where \( k \) is the number of partitions of \([0,1]\) also we can see absolute errors for some values of \( m \), for the same example.

Table 6 shows the same errors for some different values of \( k \) and \( m \) for example 6.3. Tables 5 and 6 show our method is more accurate with respect to dimensions of the system. The final system in our method has smaller size than block-pulse methods also, as another advantage if \( k(x,x) = 0 \) then the block-pulse methods do not work and their final systems are incompatible but our method works correctly.

7 Stability

To demonstrate the stability of the method, we review effect of noise on data function. In other word, we replace \( f(x) \) by \((1 + \varepsilon p) f(x) \) in (5.4) or (5.9), where \( p \) is a real random number between -1 and 1, and \( \varepsilon \) is percent of noise.

Example 7.1 Consider the following Volterra integral equation of the first kind \[ \frac{7}{12} x^4 = \int_0^x (x+t) u(t) dt \] with the exact solution \( u(x) = x^2 \).

Suppose \( p \) is a random real number in \((0,1)\) and \( \varepsilon = 0.01, 0.02, 0.03 \). In table 7, we present exact solution, approximate and noisy solutions at some points.

Example 7.2 \( u(x) = x \) is the exact solution of the following Volterra integral equation of the first kind \[ x = \int_0^x (x+t) u(t) dt \].

Table 8 shows exact solution, approximate solution and noisy solutions.

As a result of the tables, errors are proportional to the amount of noise.

8 Conclusion

In this article, we applied Bernsteins approximation to approximate the solution of linear and nonlinear Volterra integral equations of the first kind. In this method, we obtained some new operational matrices based on Bernstein polynomials. Our achieve results in this paper, show that our approach for solving Volterra integral equations of the first kind is very effective, simple and stable. The answers are trusty and their accuracy are high and we this method can be can executed in a computer easily. The numerical examples support this claim. The method can be applied for integro-differential equations, integral equations of the second and control problems.

References


[29] A. Shamsavaran, Numerical approach to solve second kind Volterra integral equations of Abel type using Block-Pulse functions


Mohsen Mohamadi is Ph.D. Of applied mathematics and Faculty of Basic Science, Mathematics Department, Islamic Azad University, Ayatolah Amol Branch, Amol,Iran. Interested in Numerical Analysis, Integral Equations, and Differential equations.

Esmail Babolian - is Professor of applied mathematics and Faculty of Mathematical sciences and computer, Kharazmy University, Tehran, Iran. Interested in numerical solution of functional Equations, integral equations, differential equations, numerical linear algebra and mathematical education.

Sohrab Ali Yousefi- is Professor of applied mathematics and Faculty of Mathematical sciences and computer, Shahid Beheshti University, Tehran, Iran. Interested in Numerical Analysis, Inverse Problems, Integral Equations, Wavelets, Fractional Equations.