Computational Block-Pulse Functions Method for Solving Volterra Integral Equations with Delay

Hameed Kadhim Dawood
*University of Diyala, C*oll. of Basic Education , *Dep. of Mathematics*
Hkd73@mail.ru

ARTICLE INFO
Submission date: 18/11/2018
Acceptance date: 11 /12/2018
Publication date: 10/3/2019

Abstract

The aim of this work is to present method of the Block-pulse function approach to numerical solution of Volterra integral equations with delay. This method is used to obtain numerical solution. Moreover, programs for his method is written in MATLAB language. An error analysis is worked out and applications demonstrated through illustrative examples.

Keywords: Block-Pulse Functions, Operational matrix, Numerical method, Volterra integral equation, time delay

1- Introduction

Delay Volterra integral differential equations arise in many areas of mathematical modelling: for example, population dynamics (taking into account the gestation times), infectious diseases (accounting for the incubation periods), physiological and pharmaceutical kinetics (modelling, for example, the body's reaction to CO2, etc. in circulating blood) and chemical kinetics (such as mixing reactants), the navigational control of ships and aircraft (with respectively large and short lags), and more general control problems [1]. The analysis to the Volterra integral equations with delays dates back to the works in [2] and [3]. Some more recent results on this subject can be found in [4] and [5].They must be solved successfully with efficient numerical approaches.

The numerical solutions of integral equations with delays have also been discussed by several authors such as [6] and [7].Block-Pulse functions (BPFs) have been used by many authors and applied for solving various problems, for example see Steffens [8] and [9].

The main objective of the current study is to implement the BPFs has been used to solve Volterra integral equations with constant time delay of the form

$$g(x) = f(x) + \int_0^x k(x,y) f(y-\tau) dy \qquad \dots\dots\dots (1.1)$$

The layout of this paper is as follows: In the next section, we discuss BPFs, their properties and function approximation by them. In Section 3, we give the description and development of the method for solving Volterra integral equations with time delay. Error estimation and rate of convergence of the method is discussed in Section 4. In Section 5 for showing the efficiency of this method, numerical example is presented. Finally, in Section 6 is devoted to the conclusion of this paper.

2- Review of Block Pulse Functions

The purpose of this section is to interpose definition and properties of BPFs. Function approximation using BPFs and operational matrix associated with BPFs have been discussed briefly.

2.1 Definition of BPFs and their properties

An *M*-set of Block-Pulse function is defined over the interval [0, *T*) as interval [0, T) is defined as with a positive integer value for k. In this paper, it is assumed that T = 1, so BPFs are defined over [0,1).

$$b_i^m(x) = \begin{cases} 1 & (i-1)h \le x < ih \\ 0 & otherwise \end{cases} \qquad \dots\dots\dots (2.1)$$

where, $i = 1, \dots, m$ i with m as a positive integer, $h = \frac{T}{m} = \frac{1}{m}$ also, b_i is called the ith BPF.

There are some main properties for BPFs, the most important of these properties disjointness, orthogonality and completeness that can be expressed as follows [10] and [11]:

Disjointness. The BPFs are disjoint with each other, i.e.,

$$b_i^m(x) b_j^m(x) = \begin{cases} \delta_{ij}\, b_i^m(x) & i = j \\ 0 & i \ne j \end{cases} \qquad \dots\dots\dots (2.2)$$

where $i, j = 1, \dots, m$ and δ_{ij} is Kronecker delta.

Orthogonality. The BPFs are orthogonal with each other, i.e., it is clear that

$$\int_0^1 b_i^m(x) b_j^m(x)\, dx = h\, \delta_{ij}\ , \qquad \text{i; j} = 1, \dots, \text{m} \qquad \dots\dots\dots (2.3)$$

Completeness. For every $f \in \mathcal{L}^2([0,1))$ approaches to the infinity, Parseval's identity holds, that is

$$\int_0^1 f^2(x)\,dx = \sum_{i=1}^{\infty} f_i^2 \|b_i^m(x)\|^2 \ , \qquad \dots\dots\dots (2.4)$$

where $f_i = \frac{1}{h}\int_0^1 f(x)\, b_i^m(x)\,dx$ $\dots\dots\dots (2.5)$

2.2 Vector Form and BPFs Expans*ion*

The set of BPFs is written as

$$B(x) = (b_1(x), b_2(x), \dots, b_m(x))^T.$$

Also, F is a m-vector given by

$$F = (f_1, f_2, \dots, f_m)^T$$

where f_i is the block pulse coefficient with respect to the ith BPF $b_i^m(x)$.
A function $f(x) \in \mathcal{L}^2([0,1))$ may be expanded by the block pulse series as [6] :

$$f(x) \simeq \sum_{i=1}^{m} f_i b_i^m(x) = F^T B(x) = B^T(x)F \ . \qquad \dots\dots\dots (2.6)$$

Now assume that $k(y,x)$be arbitrary $\mathcal{L}^2$ function of two variables on $[0,1) \times [0,1)$. It can be expanded as:

$$k(y,x) = \beta^T(y)KB(x) = B^T(x)K^T\beta(y) \dots\dots\dots (2.7)$$

where β & $B(x)$ *are* m_1 & m_2 dimensional BPFs vectors respectively, and K is the $m_1 \times m_2$ block pulse coefficient matrix with
k_{ij} , $i = 1,2,\dots,m_1$, $j = 1,2,\dots,m_2$ as follows:

$$k_{ij} = m_1\, m_2 \int_0^1 \int_0^1 k(y,x)\ \beta_i^{(m_1)}(y) B_i^{(m_2)}(x)\,dx\,dy.$$

In this paper for convenience, we put $m_1 = m_2 = m$.

*2.3 O*perational matrix of integration

The integral $\int_0^x b_i^m(y)\,dy$ is follows

$$\int_0^x b_i^m(y)\,dy = \begin{cases} 0 & 0 \le x < (i-1)h, \\ x-(i-1)h & (i-1)h \le x < ih, \\ h & ih \le x < 1. \end{cases} \qquad \dots\dots\dots (2.8)$$

As represented in [11]:

$$\int_0^x B(y)\,dy \simeq \Upsilon\ B(x) \quad \dots\dots\dots (\mathbf{2.9})$$

where Υ *is* called operational matrix of integration which can represented by

$$\Upsilon = \frac{h}{2}\begin{pmatrix} 122\dots2 \\ 012\dots2 \\ 001\dots2 \\ \vdots\,\vdots\,\vdots\,\ddots\,\vdots \\ 000\dots1 \end{pmatrix}. \quad \dots\dots\dots(2.10)$$

So, the integral of every function $f(x)$ can be approximated as follows:

$$\int_0^x f(y)\,dy \simeq F^T \Upsilon\ B(x) \quad \dots\dots\dots(2.11)$$

For approximate a function with time delay**,** we consider a block pulse function containing time delay $\tau = (p+\lambda)h$ with a nonnegative integer p and $0 \le \lambda < 1$ that can be expressed as a vector form:

$$b_i^{(m)}(x-\tau) = \Delta_i^T H^p\ B(x) - \Delta_i^T H^p B_\lambda(x) + \Delta_i^T H^{p+1} B_\lambda(x), \quad \dots\dots\dots(2.12)$$

to avoid the expression $B_\lambda(x)$ in the above equation, we expand the function $b_i^{(m)}(x-\tau)$ into its block pulse series :

$$b_i^{(m)}(x-\tau) = (c_{i1}, c_{i2}, \dots, c_{im})B(x),$$

where c_{ij} $, i,j = 1,2,\dots,m$ are block pulse coefficients given by

$$c_{ij} = \Delta_i^T\big((1-\lambda)H^p + \lambda\, H^{p+1}\big)\Delta_j \quad . \quad \dots\dots\dots(2.13)$$

Therefore, the block pulse series in a vector form :

$$B(x-\tau) = \big((1-\lambda)H^p + \lambda\, H^{p+1}\big)\, B(x) \quad \dots\dots\dots(2.14)$$

the matrix $(1-\lambda)H^p + \lambda\, H^{p+1}$ is called the block pulse operational matrix for time delay as:

$$(1-\lambda)H^p + \lambda\, H^{p+1} = \begin{pmatrix} 0\dots0\ 1-\lambda & \lambda & 0\cdots & 0 \\ 0\cdots\ 0 \quad 0 & 1-\lambda & \lambda\ \cdots & 0 \\ \vdots\ \cdots\ \vdots\ \vdots\ \vdots & \vdots & \ddots & \vdots \\ 0\cdots0\ 0\ 0 & 0 & \cdots & \lambda \\ 0\cdots0\ 0\ 0 & 0 & \cdots\ 1- & \lambda \\ 0\cdots0\ 0\ 0 & 0 & \cdots & 0 \\ \vdots\ \cdots\ \vdots\ \vdots\ \vdots & \vdots & \ddots & \vdots \\ 0\cdots0\ 0\ 0 & 0 & \cdots & 0 \end{pmatrix}_{m\times m} \quad \dots\dots\dots(2.15)$$

Thus, the block pulse series of a function containing time delay τ can be expressed as follows:

$$f(x-\tau) \simeq F^T B(x-\tau) = F^T\big((1-\lambda)H^p + \lambda\, H^{p+1}\big)\, B(x) \quad \dots\dots\dots(2.16)$$

3. Application of the Method

In this section, we applied the BPFs method to solve (1.1).

Now, we approximate $g(x), f(x), k(x,y)$ with respect to BPFs as follows :

$$g(x) \simeq G^T B(x) = B^T(x)G,$$
$$f(x) \simeq F^T B(x) = B^T(x)F,$$
$$k(x,y) \simeq B^T(x)K\beta(y) = \beta^T(y)K^T B(x),$$

we approximate $g(y-\tau)$ by relation (2.16) as follows,

$$g(y-\tau) \simeq G^T\beta(y-\tau) \simeq G^T\left((1-\lambda)H^p + \lambda H^{p+1}\right)\beta(y),$$

and putting, $N = (1-\lambda)H^p + \lambda H^{p+1}$, we obtain, $g(y-\tau) \simeq G^T N\beta(y)$.

and substituting above approximation in equation (1.1), we have

$$G^T B(x) \simeq F^T B(x) + \int_0^x G^T N\beta(y)\beta^T(y)K^T B(x)dy$$

$$\simeq F^T B(x) + G^T N\left(\int_0^x \beta(y)\beta^T(y)dy\right)K^T B(x) \quad \dots\dots\dots(3.1)$$

Now, assume that K_i be the *ith* row of the constant matrix K^T , V_i be the *ith* row of the matrix Υ, and D_{K_i} be a diagonal matrix with K_i as its diagonal entries, we will have,

$$\left(\int_0^x \beta(y)\beta^T(y)dy\right)K^T B(x) = \left(\int_0^x B(x)B^T(x)dx\right)K^T B(x)$$

$$= \begin{pmatrix} V_1\,B(x) & 0 & \cdots & 0 \\ 0 & V_2\,B(x) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & V_m\,B(x) \end{pmatrix}\begin{pmatrix} K_1 \\ K_2 \\ \vdots \\ K_m \end{pmatrix}B(x)$$

$$= \begin{pmatrix} V_1\,B(x)\,K_1\,B(x) \\ V_2\,B(x)\,K_2\,B(x) \\ \vdots \\ V_m\,B(x)\,K_m\,B(x) \end{pmatrix} = \begin{pmatrix} V_1\,B(x)B^T(x)K_1^T \\ V_2\,B(x)B^T(x)K_2^T \\ \vdots \\ V_m\,B(x)B^T(x)K_m^T \end{pmatrix}$$

$$= \begin{pmatrix} V_1 D_{K_1} \\ V_2 D_{K_2} \\ \vdots \\ V_m D_{K_m} \end{pmatrix}B(x) = RB(x), \dots\dots\dots(3.2)$$

where

$$R = \frac{h}{2}\begin{pmatrix} k_{11} & 2k_{21} & \cdots & 2k_{m1} \\ 0 & k_{22} & \cdots & 2k_{m2} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 2k_{m1} \end{pmatrix}_{m\times m} \quad \dots\dots\dots(3.3)$$

substituting relation (3.2) in (3.1), we get

$$G^T B(x) \simeq F^T B(x) + G^T B(x) NR\, B(x),$$

hence,

$$G^T (I - NR) \simeq F^T \quad \ldots\ldots\ldots (3.4)$$

We put $C = I - NR$ and replacing $\simeq$ with $=$ yields,

$$C^T G - F \quad \ldots\ldots\ldots (3.5)$$

Therefore, the equation (3.5) that gives the approximate block pulse coefficient of the unknown function $g(x)$.

4. The Rate of Convergence

In this section, we obtain an error estimate for the numerical method in previous section.

Theorem 1. [7] *Suppose that* $f(x)$ *is an arbitrary real bounded function, which is square integrable in the interval* $[0, 1)$, *and* $e(x) = f(x) - \hat{f}_m(x)$, $x \in I = [0,1)$, *which* $\hat{f}_m(x) = \sum_{i=1}^{m} f_i B_i^{(m)}(x)$ *is block pulse series of* $f(x)$. *Then,*

$$\|e(x)\| \le \frac{h}{2\sqrt{3}} \sup_{x \in I} |f'(x)|. \qquad \ldots\ldots\ldots (4.1)$$

Theorem 2. [7] *Suppose that* $f(y,x) \in \mathcal{L}^2([0,1) \times [0,1))$ *and* $e(y,x) = f(y,x) - \hat{f}_m(y,x)$, $(y,x) \in D = ([0,1) \times [0,1))$, *which is* $\hat{f}_m(y,x) = \sum_{i=1}^{m} \sum_{j=1}^{m} f_{ij}\, \beta_i^{(m)}(y)\; B_j^{(m)}(x)$ *is block pulse series of* $f(y,x)$. *Then,*

$$\|e(y,x)\| \le \frac{h}{2\sqrt{3}} \sqrt{\sup_{(s,t) \in D} \left|f_y'(s,t)\right|^2 + \sup_{(s,t) \in D} |f_x'(s,t)|^2}. \qquad \ldots\ldots\ldots (4.2)$$

From Theorem 1 and Theorem 2 we have, $\|e(x)\| = O(h)$ *and* $\|e(y,x)\| = O(h)$ and because of it we can obtain good degree of accuracy.

5. Numerical Implementation

In this section, to achieve the validity, the accuracy and support our theoretical discussion of the proposed method, we give some computational results. The computations, associated with the example, are performed by MATLAB 7. Let G_i denote the Block pulse coefficient of exact solution of the given example, and let g_i be the Block pulse coefficient of computed solutions by the presented method.

The error is defined as

$$\|e\|_\infty = \max_{1\le i\le m} |G_i - g_i|$$

Illustrative example: Consider the following Volterra integral equation with (constant) time Delay $\tau > 0$,

$$g(x) = x^2\left(1 - {\tau^2}/{2}\right) + \left({x^3}/{3}\right)\tau - {x^4}/{4} + \int_0^x y\, g(y-\tau)dy$$

The exact solutions are $g(x) = x^2 \;\; for \;\; 0 \le x \le 1$ .The numerical results obtained with BPFs are presented in Tables 1-2 and Figures 1-4.

Table 1: Numerical results.

X	Exact Solutions (G_i)	computed Solutions (g_i)	
		m=32	m=64
0.0	0.0000	0.0005	0.0001
0.1	0.0100	0.0129	0.0108
0.2	0.0400	0.0422	0.0391
0.3	0.0900	0.0896	0.0914
0.4	0.1600	0.1511	0.1577
0.5	0.2500	0.2650	0.2588
0.6	0.3600	0.3705	0.3634
0.7	0.4900	0.4955	0.4807
0.8	0.6400	0.6368	0.6493
0.9	0.8100	0.7912	0.8061

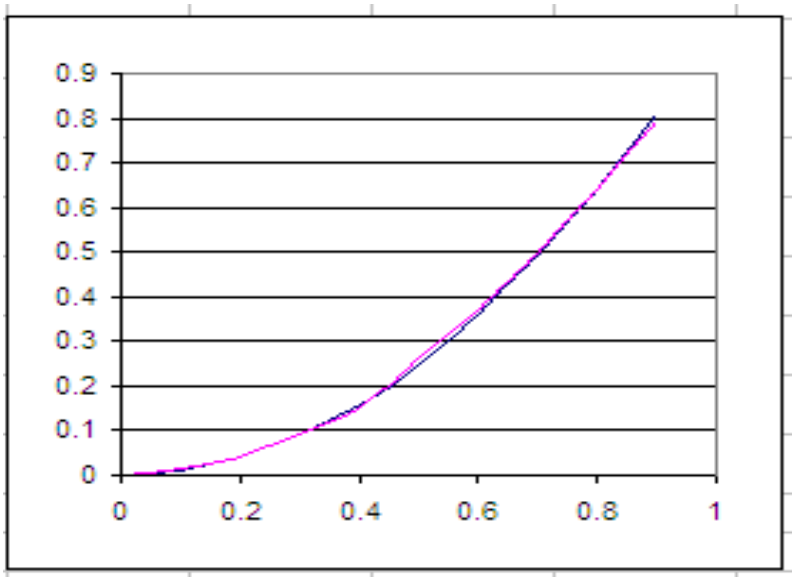

Figure 1. Plots of the exact solution against the computed solution when *m* = 32.

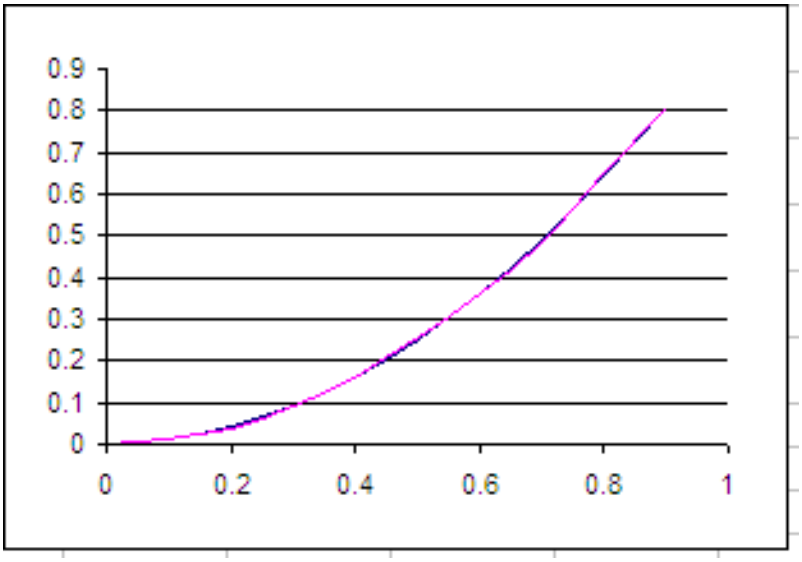

Figure 2. Plots of the exact solution against the computed solution when *m* = 64

Table 2: Maximum absolute errors

τ	$\|E\|_\infty$	
	m=32	m=64
0.01	2.8372810955E-7	1. 6419315611E-8
0.03	4.9337091632E-7	7.2100876314E-8
0.05	1.7135916728E-5	2.0641244876E-5
0.07	1.1827364911E-5	1.4937610927E-5
0.09	1.0371194285E-5	2.9103752773E-6
0.11	2.4218503252E-6	3.3307113458E-6
0.13	2.1006914883E-6	6.3179003419E-6
0.15	2.0472915782E-6	3.0014720815E-7
0.17	1.8729519054E-6	2.8260915427E-7
0.19	1.4178209331E-6	1.5151730168E-7

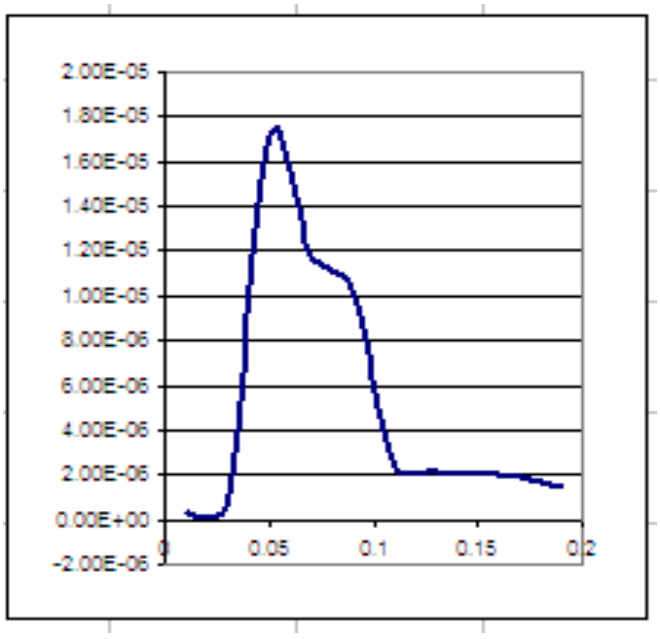

Figure 3. Plots of the absolute error functions $\|E\|_\infty$ when m = 32

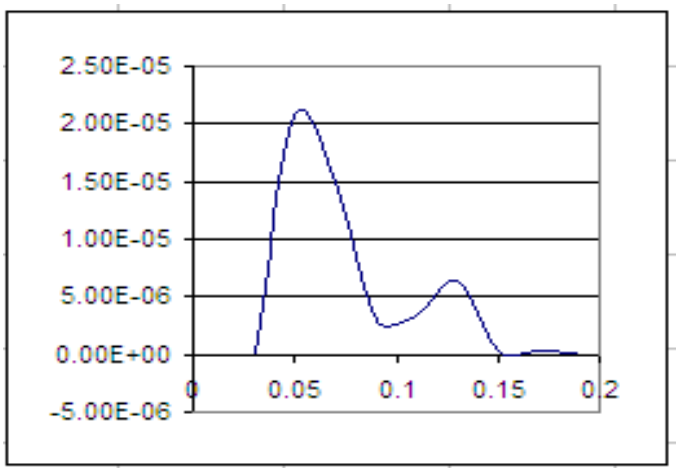

Figure 4. Plots of the absolute error functions $\|E\|_\infty$ when $m = 64$

6. Conclusion

This method is more efficient and more accurate than some other methods for solving this class of integral equations. On the other hand, the benefit of this method is low cost of computing operations. The applied method transforms the singular integral equation into triangular linear algebraic system that can be solved easily.

Acknowledgements

The author would like to the anonymous reviewers of this paper for their careful reading, constructive comments, and nice suggestions which have improved the paper very much

CONFLICT OF INTERESTS.

There are non-conflicts of interest.

References

[1] Roberts, C: Ordinary Differential Equations: Applications, Models, and Computing. CRC Press, Boca Raton (2011).

[2] J. Biazar, E. Babolian, R. Islam; Solution of a system of Volterra integral equations of the first kind by Adomian method, Appl. Math. Comput, 139 (2003) 249-258.

[3] J. Cerha, On some linear Volterra delay equations, Casopis Pest Mat., 101 (1976) ,111-123.

[4] F. Mirzaee, Numerical computational solution of the linear Volterra integral equations system via rationalized Haar functions, Journal of King Saud University-Science, 22 (4) (2010), 265-268.

[5] Markova E.V., Sidorov D.N. On One Integral Volterra Model of Developing Dynamical Systems. Automation and Remote Control, 2014, vol. 75, no. 3, pp. 413–421.
[6] H. Brunner, Iterated collocation methods for Volterra integral equations with delay arguments, Math. Comput. 62 (1994) 581-599.
[7] P. Linz, R. L. C. Wang, Error bounds for the solution of Volterra and delay equations, Appl. Numer. Math. 9 (1992) 201-207.
[8] Steffens, K. G. The history of approximation theory: From Euler to Brenstein. Boston: Birkhauser.(2006) ,ISBN 0817643532.
[9] G. P. Rao, Piecewise Constant Orthogonal Functions and Their Application to Systems and Control,Springer, Berlin, (1983).
[10] T. J. Rivlin, ”An introduction to the approximate of functions,” in New York, DovePublications, 1969.
[11] E. Babolian, K. Maleknejad, M. Mordad, and B. Rahimi, “A numerical method for solving Fredholm-Volterra integral equations in two dimensional spaces using block pulse functions and an operational matrix,” Journal of Computational and Applied Mathematics, vol. 235, no. 14, pp. 3965–3971, 2011.

الخلاصة

يتمثل الهدف من هذا العمل في اتباع نهج طريقة الضغط النبضي في الحل العددي لمعادلات فولتيرا التكاملية مع التأخير. تستخدم هذه الطريقة للحصول على حل رقمي. علاوة على ذلك، تتم كتابة البرنامج بلغة MATLAB. تم عمل تحليل للخطأ وتم توضيح التطبيقات من خلال المثال التوضيحي.