Comp. Appl. Math.
DOI 10.1007/s40314-015-0296-2

A new approach for numerical solution of a linear system with distributed delays, Volterra delay-integro-differential equations, and nonlinear Volterra-Fredholm integral equation by Bezier curves

F. Ghomanjani[1] · M. H. Farahi[2] · N. Pariz[3]

Received: 13 November 2013 / Revised: 23 August 2015 / Accepted: 2 November 2015

Abstract In this paper, we present Bezier curves method to solve Volterra delay-integro-differential equations. Also, this paper is concerned with a linear system with distributed input delay and input saturation. The approximation process by Bezier curves method is done in two steps. First we divide the time interval into $2k$ subintervals, second approximate the trajectory and control functions in each subinterval by Bezier curves. We have chosen the Bezier curves as piecewise polynomials of degree n, and determine Bezier curves on any subinterval by $n + 1$ control points. Also, we have used Bezier curves method to solve linear and nonlinear Volterra-Fredholm integral equations, numerically. The proposed method is simple and computationally advantageous. Some numerical examples demonstrate the validity and applicability of the technique.

Keywords Numerical solution · Time delay · Distributed input delay · Input saturation · Bezier curves · Dynamic system

Mathematics Subject Classification 49J30 · 49J21

1 Introduction

Delay differential equations with continuous argument deviation are defined as distributed delay systems. These equations are encountered in many practical engineering systems. For

Communicated by Luz de Teresa.

✉ F. Ghomanjani
fatemeghomanjani@gmail.com

1 Kashmar Higher Education Institute, Kashmar, Iran

2 Department of Applied Mathematics, Faculty of Mathematical Sciences, Ferdowsi University of Mashhad, Mashhad, Iran

3 Department of Control, Faculty of Engineering, Ferdowsi University of Mashhad, Mashhad, Iran

Published online: 20 November 2015

instance, distributed delay appears in the modeling of feeding system and combustion chamber in a liquid monopropellant rocket motor with pressure feeding [see Chen and Zheng (2007) and Fridman and Shaked (2002)]. Also distributed delay controllers are proposed for discrete delay plants which leads to distributed delay differential equations [see Manitius and Olbrot (1979) and Watanabe et al. (2006)]. A common way to deal with distributed delay systems is to replace the distributed delay by the sum of a series of discrete (often commensurate) delays (Zhong 2004). Furthermore, it is possible to implement distributed delay using rational transfer function (Zhong 2005). Another approach to stabilize distributed delay systems is the reduction technique with which the system is reduced to a delay-free system by employing a transformation (Zheng et al. 1994). It is obvious that approximations made in the above-mentioned methods reduce their results' accuracy. Wu et al. (2006) proposed a sufficient condition for robust stability of linear time invariant distributed delay systems with polytopictype uncertainties using Lyapunov-Krasovskii functional. Then, Wu et al. (2006) developed the sufficient condition of the robust stabilization controller and the existence condition for sliding mode. The results are given in terms of linear matrix inequalities (LMIs). This design method is very complicated and can be done only for a special class of systems. The model predictive control (MPC) has become an attractive feedback stabilization strategy. It is formulated as solving online a finite horizon open loop optimal control problem. However, there are major computational problems in the design of optimal control signal for delay systems. Orthogonal functions in Nazarzadeh (1998) were used to derive optimal control for time delay systems. Also Chebyshev polynomials were utilized to establish a computational procedure for deriving predictive control for linear time-varying systems with distributed time delay. The proposed method in Esfanjani and Nikravesh (2010) is based on expanding all time functions in the system equation in terms of Chebyshev functions. The operational matrices of product and integration together with the operational matrix of delay are used to transform the solution of distributed differential delay systems to the solution of algebraic equations (Horng and Chou 1985). Therefore, piecewise polynomial functions are often used to represent the approximate solution in the numerical solution of differential equations, [see (Winkel 2001; Zheng et al. 2004; Heinkenschloss 2005; Juddu 2002)]. Due to numerical stability and arbitrary order of accuracy, B-splines have become popular tools for solving differential equations (where Bezier form is a special case of B-splines). There are many papers and books deal with the Bezier curves or surface techniques.

Harada and Nakamae (1982), N$\ddot{u}$rnberger and Zeilfelder (2000) have used the Bezier control points in approximate data and functions. Zheng et al. (2004) proposed the use of control points of the Bernstein-Bezier form for solving differential equations numerically and also Evrenosoglu and Somali (2008) have used this approach for solving singular perturbed two points boundary value problems. The Bezier curves are used in solving partial differential equations, as well, Wave and Heat equations are solved in Bezier form, [see (Beltran and Monterde 2004; Cholewa et al. 2002; Lang 2004; Layton and Van de Panne 2002)], Bezier curves are used for solving dynamical systems, (see Gachpazan 2011). Also the Bezier control points method is used for solving delay differential equation and switched systems (see Ghomanjani and Farahi 2012a, b; Ghomanjani et al. 2012, 2013a, b). Some other applications of the Bezier functions and control points are found in (Chu et al. 2008; Farin et al. 1988; Shi and Sun 2000), which are used in computer-aided geometric design and image compression.

We suggest a technique similar that is used in Zheng et al. (2004), Evrenosoglu and Somali (2008), and Ghomanjani and Farahi (2012a) for solving a linear system with distributed input delay.

2 Statement of the problem

Consider the following optimal control problem with distributed time delays:

$$
\begin{aligned}
\min I &= \frac{1}{2}\int_{t_0}^{t_f} (\mathbf{x}^T(s)Q(s)\mathbf{x}(s) + \mathbf{u}^T(s)R(s)\mathbf{u}(s))\,\mathrm{d}s \\
s.t.\ \dot{\mathbf{x}}(t) &= A_0(t)\mathbf{x}(t) + A_1(t)(x_1(t-h_1^1)\dots x_p(t-h_1^p))^T + B(t)\mathbf{u}(t) \\
&\quad + \int_0^{\tau} G(t)(x_1(t-\sigma)\dots x_p(t-\sigma))^T \mathrm{d}\sigma, \\
\mathbf{x}(t) &= \boldsymbol{\phi}(t),\ t\in[t_0-\tau, t_0],
\end{aligned}
\tag{1}
$$

where the state $\mathbf{x}(t)$ is a p vector function; $\mathbf{u}(t)$ is a m vector control function; h_1^i $(i = 1, 2, \dots, p)$ and σ are non negative constant time delays; τ is non negative constant. $A_0(t) = [a_{ij}^0(t)]_{p\times p}$, $A_1(t) = [a_{ij}^1(t)]_{p\times p}$, $B(t) = [b_{ij}(t)]_{p\times m}$ and $G(t) = [g_{ij}(t)]_{p\times p}$ are matrix functions, and the vector function $\boldsymbol{\phi}(t)$ is defined appropriately and is given. We assume the matrix $Q(s) = [q_{ij}(s)]_{p\times p}$ is semi-positive definite, and $R(s) = [r_{ij}(s)]_{m\times m}$ is positive definite matrix.

Remark 2.1 We also consider system (1) in the presence of saturation,

$$
\begin{aligned}
\dot{\mathbf{x}}(t) &= A_0(t)\mathbf{x}(t) + A_1(t)(x_1(t-h_1^1)\dots x_p(t-h_1^p))^T + B(t)\mathbf{u}(t) \\
&\quad + \int_0^{\tau} G(t)(sat(x_1(t-\sigma))\dots sat(x_p(t-\sigma)))^T \mathrm{d}\sigma,
\end{aligned}
$$

where $sat(.)$ is the standard vector-valued saturation functions defined as

$$
sat(x_i) = sign(x_i)\max\{1, |x_i|\}, \quad 1\le i\le p.
$$

Remark 2.2 It has been shown that the integral (distributed time delay) in the Eq. (1) can be approximated in the time domain using the forward rectangular rule as follows (Zhong 2004):

$$
\begin{aligned}
&\int_0^{\tau} G(t)(x_1(t-\sigma)\dots x_p(t-\sigma))^T \mathrm{d}\sigma \\
&\quad = \frac{\tau}{N}\sum_{g=0}^{N-1} G(t)\left(x_1\left(t-\frac{\tau}{N}g\right)\dots x_p\left(t-\frac{\tau}{N}g\right)\right)^T.
\end{aligned}
\tag{2}
$$

where the interval $[0, \tau]$ over which the function is to be integrated is divided into N equal subintervals of length $\frac{\tau}{N}$. By substituting (2) in (1), we have

$$
\begin{aligned}
\min I &= \frac{1}{2}\int_{t_0}^{t_f} (\mathbf{x}^T(s)Q(s)\mathbf{x}(s) + \mathbf{u}^T(s)R(s)\mathbf{u}(s))\,\mathrm{d}s \\
s.t.\ \dot{\mathbf{x}}(t) &= A_0(t)\mathbf{x}(t) + A_1(t)(x_1(t-h_1^1)\dots x_p(t-h_1^p))^T + B(t)\mathbf{u}(t) \\
&\quad + \frac{\tau}{N}\sum_{g=0}^{N-1} G(t)\left(x_1\left(t-\frac{\tau}{N}g\right)\dots x_p\left(t-\frac{\tau}{N}g\right)\right)^T, \\
\mathbf{x}(t) &= \boldsymbol{\phi}(t),\ t\in[t_0-\tau, t_0];
\end{aligned}
\tag{3}
$$

thus, the distributed delay is replaced by a weighted sum of punctual delays.

3 System analysis

For solving problem (3), one can divide the interval $[t_0, t_f]$ into a set of grid points such that
$$t_i = t_0 + ih, \quad i = 0,\ 1, \ldots, 2k,$$
where $h = \frac{t_f - t_0}{2k}$, and k is a positive integer. Let $S_j = [t_{j-1}, t_j]$ for $j = 1,\ 2, \ldots, 2k$. Then, for $t \in S_j$, the problem (3) can be decomposed to the following problems:
$$\begin{aligned}
\min I_j &= \frac{1}{2}\int_{t_{j-1}}^{t_j} (\mathbf{x}_j^T(s)Q(s)\mathbf{x}_j(s) + \mathbf{u}_j^T(s)R(s)\mathbf{u}_j(s))\,\mathrm{d}s \\
s.t.\ \dot{\mathbf{x}}_j(t) &= A_0(t)\mathbf{x}_j(t) + A_1(t)\left(x_1^{-k_1^1+j}\left(t-h_1^1\right)\ldots x_p^{-k_1^p+j}\left(t-h_1^p\right)\right)^T \\
&\quad + B(t)\mathbf{u}(t) \\
&\quad + \frac{\tau}{N}\sum_{g=0}^{N-1} G(t)\left(x_1^{-k_{2,g}^j+j}\left(t-\frac{\tau}{N}g\right)\ldots x_p^{-k_{2,g}^p+j}\left(t-\frac{\tau}{N}g\right)\right)^T, \\
&\quad j = 1, 2, \ldots, 2k, \\
\mathbf{x}(t) &= \boldsymbol{\phi}(t),\ t \in [t_0 - \tau, t_0],
\end{aligned} \tag{4}$$
where $\mathbf{x}_j(t) = (x_1^j(t)\ldots x_p^j(t))^T$, and $\mathbf{u}_j(t) = (u_1^j(t)\ldots u_m^j(t))^T$ are, respectively, vectors of $\mathbf{x}(t)$ and $\mathbf{u}(t)$ which are considered in $t \in S_j$; we mention that $x_i^{-k_1^i+j}(t-h_1^i)$; $1 \le i \le p$, is the ith component of $(x_1^{-k_1^1+j}(t-h_1^1)\ldots x_p^{-k_1^p+j}(t-h_1^p))^T$ where $(t-h_1^i) \in [t_{-k_1^i+j-1}, t_{-k_1^i+j}]$. Also
$$k_1^i = \begin{cases} \frac{h_1^i}{h} & \frac{h_1^i}{h} \in \mathbb{N} \\ \left([\frac{h_1^i}{h}] + 1\right) & \frac{h_1^i}{h} \notin \mathbb{N}, \end{cases}$$
$$k_{2,g}^i = \begin{cases} \frac{\tau}{hN}g & \frac{\tau}{hN}g \in \mathbb{N} \\ \left([\frac{\tau}{hN}g] + 1\right) & \frac{\tau}{hN}g \notin \mathbb{N}, \end{cases}$$
$$1 \le i \le p, \quad 0 \le g \le N-1,$$
where $\left[\frac{h_1^i}{h}\right]$ and $\left[\frac{\tau}{hN}g\right]$ denote the integer part of $\frac{h_1^i}{h}$ and $\frac{\tau}{hN}g$, respectively.

Our strategy is to using Bezier curves to approximate the solutions $\mathbf{x}_j(t)$ and $\mathbf{u}_j(t)$ by $\mathbf{v}_j(t)$ and $\mathbf{w}_j(t)$, respectively, where $\mathbf{v}_j(t)$ and $\mathbf{w}_j(t)$ are given below. Individual Bezier curves that are defined over the subintervals are joined together to form the Bezier spline curves. For $j = 1, 2, \ldots, 2k$, define the Bezier polynomials of degree n that approximate, respectively, the actions of $\mathbf{x}_j(t)$ and $\mathbf{u}_j(t)$ over the interval $[t_{j-1}, t_j]$ as follows:
$$\begin{aligned}
\mathbf{v}_j(t) &= \sum_{r=0}^{n} \mathbf{a}_r^j B_{r,n}\left(\frac{t-t_{j-1}}{h}\right), \\
\mathbf{w}_j(t) &= \sum_{r=0}^{n} \mathbf{b}_r^j B_{r,n}\left(\frac{t-t_{j-1}}{h}\right),
\end{aligned} \tag{5}$$
where
$$B_{r,n}\left(\frac{t-t_{j-1}}{h}\right) = \binom{n}{r}\frac{1}{h^n}(t_j - t)^{n-r}(t - t_{j-1})^r$$

is the Bernstein polynomial of degree n over the interval $[t_{j-1}, t_j]$, $\mathbf{a}_r^j$ and $\mathbf{b}_r^j$ are, respectively, p and m ordered vectors from the control points (see Zheng et al. (2004)). By substituting $\mathbf{v}_j(t)$ and $\mathbf{w}_j(t)$ in (5), respectively, for $\mathbf{x}_j(t)$ and $\mathbf{u}_j(t)$ in (4), one may define $R_{1,j}(t)$ for $t \in [t_{j-1}, t_j]$ as follows:

$$R_{1,j}(t) = \dot{\mathbf{v}}_j(t) - A_0(t)\mathbf{v}_j(t) - A_1(t)\left(v_1^{-k_1^1+j}\left(t - h_1^1\right)\ldots v_p^{-k_1^p+j}\left(t - h_1^p\right)\right)^T$$
$$-B(t)\mathbf{w}(t) - \frac{\tau}{N}\sum_{g=0}^{N-1} G(t)\left(v_1^{-k_{2,g}^j+j}\left(t - \frac{\tau}{N}g\right)\ldots v_p^{-k_{2,g}^p+j}\left(t - \frac{\tau}{N}g\right)\right)^T,$$
$$R_{2,j}(t) = \mathbf{v}_j^T(t)Q(t)\mathbf{v}_j(t) + \mathbf{w}_j^T(t)R(t)\mathbf{w}_j(t).$$

Let $\mathbf{v}(t) = \sum_{j=1}^k \chi_j^1(t)\mathbf{v}_j(t)$ and $\mathbf{w}(t) = \sum_{j=1}^k \chi_j^2(t)\mathbf{w}_j(t)$ where $\chi_j^1(t)$ and $\chi_j^2(t)$ are, respectively, characteristic function of $\mathbf{v}_j(t)$ and $\mathbf{w}_j(t)$ for $t \in [t_{j-1}, t_j]$. Beside the boundary conditions on $\mathbf{v}(t)$, at each node we need to impose continuity condition on each successive pair of $\mathbf{v}_j(t)$ to guarantee the smoothness. Since the differential equation is of first order, the continuity of $\mathbf{x}$ (or $\mathbf{v}$) and its first derivative gives

$$\mathbf{v}_j^{(s)}(t_j) = \mathbf{v}_{j+1}^{(s)}(t_j), \quad s = 0, 1, \quad j = 1, 2, \ldots, 2k-1. \tag{6}$$

where $\mathbf{v}_j^{(s)}(t_j)$ is the sth derivative $\mathbf{v}_j(t)$ with respect to t at $t = t_j$.

Thus, the vector of control points $\mathbf{a}_r^j$ ($r = 0, 1, n-1, n$) must satisfy (see Appendix A)

$$\begin{aligned} \mathbf{a}_n^j(t_j - t_{j-1})^n &= \mathbf{a}_0^{j+1}(t_{j+1} - t_j)^n, \\ (\mathbf{a}_n^j - \mathbf{a}_{n-1}^j)(t_j - t_{j-1})^{n-1} &= (\mathbf{a}_1^{j+1} - \mathbf{a}_0^{j+1})(t_{j+1} - t_j)^{n-1}. \end{aligned} \tag{7}$$

One may recall that $\mathbf{a}_r^j$ is a p ordered vector. This approach is called the subdivision scheme (or h-refinement in the finite element literature).

Remark 3.1 If we consider the C^1 continuity of $\mathbf{w}$, the following constraints will be added to constraints in (7):

$$\begin{aligned} \mathbf{b}_n^j(t_j - t_{j-1})^n &= \mathbf{b}_0^{j+1}(t_{j+1} - t_j)^n, \\ (\mathbf{b}_n^j - \mathbf{b}_{n-1}^j)(t_j - t_{j-1})^{n-1} &= (\mathbf{b}_1^{j+1} - \mathbf{b}_0^{j+1})(t_{j+1} - t_j)^{n-1}, \end{aligned}$$

where the so-called $\mathbf{b}_r^j$ ($r = 0, 1, n-1, n$) is a m ordered vector.

Now, we define the residual function in S_j as follows:

$$R_j = \int_{t_{j-1}}^{t_j} (M\|R_{1,j}(t)\|^2 + (R_{2,j}(t))^2)\mathrm{d}t, \tag{8}$$

where $\|.\|$ is the L_2 norm and M is a sufficiently large penalty parameter. Our aim is to solve the following problem over $S = \bigcup_{j=1}^{2k} S_j$:

$$\begin{aligned} \min \quad & \sum_{j=1}^{2k} R_j \\ s.t. \quad & \mathbf{a}_n^j(t_j - t_{j-1})^n = \mathbf{a}_0^{j+1}(t_{j+1} - t_j)^n, \end{aligned}$$

$$\begin{aligned}&(\mathbf{a}_n^j - \mathbf{a}_{n-1}^j)(t_j - t_{j-1})^{n-1} = (\mathbf{a}_1^{j+1} - \mathbf{a}_0^{j+1})(t_{j+1} - t_j)^{n-1},\\ &j = 1, 2, \ldots, 2k-1,\\ \mathbf{v}(t) &= \phi(t),\ t \in [t_0 - \tau, t_0],\end{aligned} \tag{9}$$

The mathematical programming problem (9) can be solved by many subroutine algorithms; we used Maple 12 to solve this optimization problem.

Ghomanjani et al. (2012) proved the convergence of this method where $n \to \infty$.

4 Numerical examples

In applying the method, in Example 1, we choose the Bezier curves as piecewise polynomials of degree 3.

Example 1 As a typical example consider the delay system described by (Esfanjani and Nikravesh 2010),

$$\begin{aligned}\min\ I &= \frac{1}{2}\int_0^{t_f} (x^T(s)Qx(s) + u^T(s)Ru(s))\,\mathrm{d}s\\ &\quad + \frac{1}{2}x^T(t_f)Fx(t_f)\\ s.t.\ \dot{x}(t) &= (t+1)x(t) + (2t+1)x(t-0.5)\\ &\quad + \int_0^{0.2} tx(t-s)\,\mathrm{d}s - u(t),\\ x(t) &= 1, \quad t \le 0,\end{aligned}$$

where $t_f = 2$, $Q = 500$, $R = 10$ and $F = 50$.

Let $k = 5$ and $n = 3$. the proposed method of the paper is used to solve the mentioned regulation problem. The results, $x(t)$ and $u(t)$, are shown in Figs. 1 and 2, respectively. The existing results are shown in Fig. 3. The objective function is $I = 0.00008235$ for presented method.

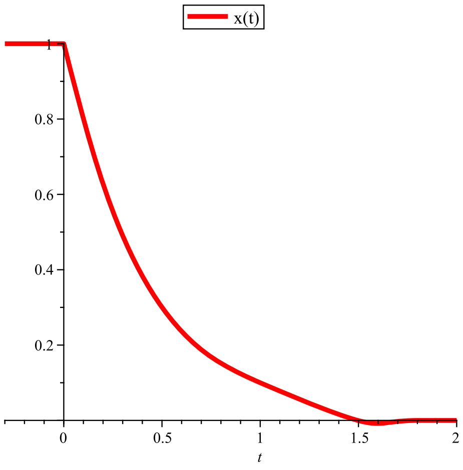

Fig. 1 The graph of approximated trajectory $x(t)$ for Example 1

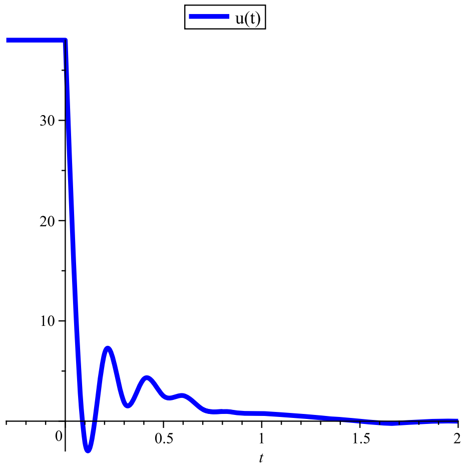

Fig. 2 The graph of approximated control $u(t)$ for Example 1

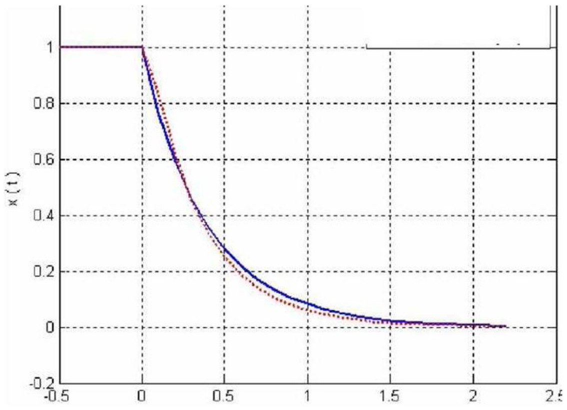

Fig. 3 State trajectory $x(t)$. *Solid line* for the method in Esfanjani and Nikravesh (2010) and *dotted line* for the method in Watanabe et al. (2006) (for Example 1)

Remark 4.1 Volterra delay-integro-differential equations (VDIDEs) arise widely in scientific fields such as ecology, medicine, biology, and physics (see El-Hawary and El-Shami 2013). This class of equations plays an important role in modeling diverse problems of natural science and engineering and hence have come to intrigue researchers in numerical computation and analysis.

In this Remark, we propose a numerical technique which is based on Bezier curves method to solve VDIDEs of the following form:

$$\begin{aligned} &\dot{x}(t) = \alpha x(t) + \beta x(t-\tau) + \gamma \int_{t-\tau}^{t} g(s)x(s)\,\mathrm{d}s + D(t), \quad 0 \le t \le 1, \\ &x(t) = \phi(t) \;\; t \in [-\tau, 0], \end{aligned} \tag{10}$$

where $\alpha,\ \beta,\ \gamma \in R$, and τ is non-negative constant time delay. The function D is assumed to be sufficiently smooth with respect to its argument, and $\phi(t)$ is an initial function which is assumed to be continuous.

Huang and Vandewalle (2004) proved that the repeated trapezium rule retains the asymptotic stability of (10). Wu and Gan (2008) further extended the above study to the case of neutral equations. Rihan et al. (2009) presented a new technique for numerical treatments of

Volterra delay integro-differential equations which was based on the mono-implicit Runge-Kutta method for the differential part and Boole's quadrature rule for the integral part. The analytical and numerical stability regions had been deduced (Rihan et al. 2009).

Now, set

$$z(t) = \int_{t-\tau}^{t} g(s)x(s)\,\mathrm{d}s, \tag{11}$$

Then, (10) can be transformed into a system of delay differential equations

$$\begin{bmatrix} \dot{x}(t) \\ \dot{z}(t) \end{bmatrix} = \begin{bmatrix} \alpha & \gamma \\ g(t) & 0 \end{bmatrix} \begin{bmatrix} x(t) \\ z(t) \end{bmatrix} + \begin{bmatrix} \beta & 0 \\ -g(t-\tau) & 0 \end{bmatrix} \begin{bmatrix} x(t-\tau) \\ z(t-\tau) \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix} D(t) \tag{12}$$

Now, the residual function in (8) is defined as follows:

$$R_j = \int_{t_{j-1}}^{t_j} \|R_{1,j}(t)\|^2\,\mathrm{d}t, \tag{13}$$

Example 2 Consider first the scalar VDIDE of the form

$$\begin{aligned} &\dot{x}(t) = x(t-1) + \int_{t-1}^{t} x(s)\,\mathrm{d}s, \quad 0 \le t \le 1, \\ &x(t) = e^t, \quad t \le 0, \end{aligned} \tag{14}$$

with both discretely and continuously distributed delays. The exact solution of (14) is $x(t) = e^t$. Let $k = 10$ and $n = 3$, then the time interval [0, 1] is divided into 10 subintervals. By Bezier curves method, the following approximated solutions can be found for the state $x(t)$:

$$x(t) = \begin{cases} 1 + 1.005514830t + .4632270000t^2 + .2412560000t^3, & 0 \le t \le 0.1, \\ 0.9999999970 + 1.005514920t + .4632261000t^2 + 0.2412590000t^3, & 0.1 \le t \le 0.2, \\ 1.000000035 + 1.005514410t + .4632285000t^2 + 0.2412550000t^3, & 0.2 \le t \le 0.3, \\ 1.000000089 + 1.005514050t + .4632291000t^2 + 0.2412550000t^3, & 0.3 \le t \le 0.4, \\ 1.000000121 + 1.005514050t + .4632285000t^2 + 0.2412560000t^3, & 0.4 \le t \le 0.5, \\ .9999998710 + 1.005515550t + 0.4632255000t^2 + 0.2412580000t^3, & 0.5 \le t \le 0.6, \\ .9654480790 + 1.178274510t + 0.1752939000t^2 + 0.4012200000t^3, & 0.6 \le t \le 0.7, \\ 1.004154747 + 1.012389000t + 0.4122729000t^2 + 0.2883730000t^3, & 0.7 \le t \le 0.8, \\ 0.9323501150 + 1.281656070t + 0.07568940000t^2 + 0.4286160000t^3, & 0.8 \le t \le 0.9, \\ 0.9540383780 + 1.209361650t + .1560168000t^2 + 0.3988650000t^3, & 0.9 \le t \le 1, \end{cases},$$

Figure 4 displays the numerical and the exact solution of (14), and the absolute error is shown in Table 1.

Example 3 In this example, we consider the VDIDE

$$\begin{aligned} &\dot{x}(t) = e^{-2}x(t-1) + 2\int_{t-1}^{t} e^{s-t}x(s)\,\mathrm{d}s, \quad 0 \le t \le 1, \\ &x(t) = e^t, \quad t \le 0, \end{aligned} \tag{15}$$

for which only a continuously distributed delay with bounded time-lag is presented. The exact solution of (15) is $x(t) = e^t$. By Bezier curves method, the following approximated solutions can be found for the state$x(t)$:

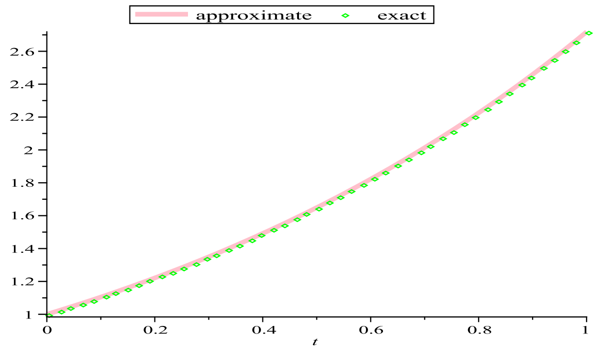

Fig. 4 The the numerical and the exact solution for Example 2

Table 1 The error of $x(t)$ for Example 2

t	Absolute error
0.1	0.0002540909244
0.2	0.0001593388398
0.3	$4.239968960\times10^{-10}$
0.4	0.00006201264127
0.5	$2.998718532\times10^{-10}$
0.6	0.00006330860949
0.7	$4.704765216\times10^{-10}$
0.8	0.00002665050753
0.9	0.000006944843050
1.0	$4.590452354\times10^{-10}$

$$x(t)=\begin{cases} 1+1.009357980t+.4427301000t^2+0.2668770000t^3, & 0\le t\le 0.1,\\ 0.9999999990+1.009357980t+0.4427301000t^2+0.2668780000t^3, & 0.1\le t\le 0.2,\\ 0.9999999990+1.009357980t+0.4427301000t^2+0.2668780000t^3, & 0.2\le t\le 0.3,\\ 1.000000053+1.009357530t+0.4427313000t^2+0.2668770000t^3, & 0.3\le t\le 0.4,\\ 1.000000101+1.009357290t+0.4427316000t^2+0.2668770000t^3, & 0.4\le t\le 0.5,\\ 1.000000051+1.009357740t+0.4427304000t^2+0.2668780000t^3, & 0.5\le t\le 0.6,\\ 1.275358813-0.3674360100t+2.737386600t^2-1.007931000t^3, & 0.6\le t\le 0.7,\\ 0.007030612000+5.068255860t-5.027886900t^2+2.689818000t^3, & 0.7\le t\le 0.8,\\ 1.250960340+0.4035189000t+0.8030349000t^2+0.2602670000t^3, & 0.8\le t\le 0.9,\\ 0.8859587880+1.620190740t-0.5488227000t^2+0.7609550000t^3, & 0.9\le t\le 1, \end{cases},$$

The absolute error is shown in Table 2.

Remark 4.2 For linear integral equations, it is usually hard to find exact solutions. Therefore, they have been of great interest to several authors. The books edited by Green (1989) and Kanwal (1996) contain many different methods to solve integral equations analytically. Numerical methods also take an important place in solving integral equations such as Galerkin method, Collocation method, Taylor series, Taylor polynomials, homotopy perturbation method, varitional iteration method, and expansion method (Abbasbandy 2006; Kauthen 1989; Yalsinbas 2002). In the papers such as (Maleknejad and Derili 2006; Rashed 2004), new methods for

Table 2 The error of $x(t)$ for Example 3

t	Absolute error
0.1	0.0004590579244
0.2	0.0003130648398
0.3	$4.239968960 \times 10^{-10}$
0.4	0.0001644966413
0.5	$2.998718532 \times 10^{-10}$
0.6	0.0005244866095
0.7	$4.704765216 \times 10^{-10}$
0.8	0.004566428492
0.9	0.005282849157
1.0	$4.590452354 \times 10^{-10}$

the numerical solution of integral and IDEs have been proposed. The presented methods are based on representation of the solutions in the linear combination of Lagrange's fundamental polynomials and on approximation of the integral terms by the Clenshaw-Curtis quadrature formula (see Davis 1975; Rashed 2004).

The discussion of the Fredholm-Volterra integral equations numerically and analytically can be found in Cerdik-Yaslan and Akyuz-Dascioglu (2006), Yusufoglu and Erbas (2008). Cui and Du (2006) obtained the representation of the exact solution for the nonlinear Volterra-Fredholm integral in the reproducing kernel space. The exact solution was given by the form of series. Its approximate solution was obtained by truncating the series. Bildik and Inc (2007) calculated the approximate solutions of the nonlinear Volterra-Fredholm integral equations using modified decomposition method. They demonstrated that the modified decomposition procedure is quite efficient to determine the solution in closed form by using initial condition.

Yusufoglu and Erbas (2008) transformed the Fredholm-Volterra type integral equation into a matrix equation by substituting the interpolation points in the equations. These were acquired on using the transformed matrix equation, which corresponds to a system of linear algebraic equations.

A Chebyshev collocation method was developed to find an approximate solution for nonlinear Fredholm-Volterra integro-differential equation. This method transformed the nonlinear Fredholm-Volterra integro-differential equation into the matrix equation with the help of Chebyshev collocation points. The matrix equation corresponded to a system of nonlinear algebraic equations with the unknown Chebyshev coefficients (Yalsinbas 2002).

Mixed Volterra-Fredholm integral equations arise in the theory of parabolic boundary value problems, the mathematical modeling of the spatio-temporal development of an epidemic, and various physical and biological problems. Several authors consider the mixed Volterra-Fredholm integral equation of the form

$$y(t) = f(t) + \lambda_1 \int_{t_0}^{t} K_1(x,t) F(y(x))\,\mathrm{d}x + \lambda_2 \int_{t_0}^{t_f} K_2(x,t) G(y(x))\,\mathrm{d}x, \quad t_0 \le x, t \le t_f, \tag{16}$$

where $t_0, t_f \in R$, $f(t)$ and the kernels $K_1(x,t)$ and $K_2(x,t)$ are assumed to be in $L^2(R)$ on the interval $t_0 \le x, t \le t_f$.

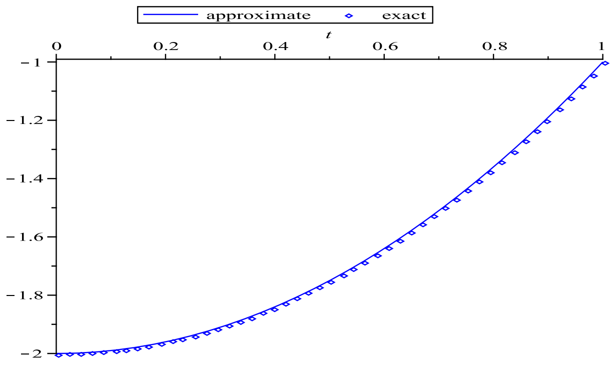

Fig. 5 The exact and approximated solution $y(t)$ for Example 4

In this work, the Bezier curves method is used to find the numerical solution of (16) (see Ghomanjani et al. 2012, 2013b).

By substituting (5) in (16), $R(t)$ can be defined for $t \in [t_0, t_f]$ as

$$R(t) = y(t) - \left(f(t) + \lambda_1 \int_{t_0}^{t} K_1(x,t)F(y(x))\,\mathrm{d}x + \lambda_2 \int_{t_0}^{t_f} K_2(x,t)G(y(x))\,\mathrm{d}x \right).$$

Example 4 Consider the nonlinear Volterra-Fredholm integral equation given in Lan (2007) by

$$y(t) = -\frac{1}{30}t^6 + \frac{1}{3}t^4 - t^2 + \frac{5}{3}t - \frac{5}{4} + \int_0^t (t-x)(y^2(x))\,\mathrm{d}x + \int_0^1 (x+t)y(x)\,\mathrm{d}x,$$

with $y(t) = t^2 - 2$, as the exact solution. Let $N = 3$. From (9), one can find the following approximated solution:

$$y(t) = -2 - 3 \times 10^{-9}t + t^2 - 5 \times 10^{-9}t^3.$$

The exact solution and the approximated solution are plotted in Fig. 5.

Example 5 Consider the following linear Volterra-Fredholm integral equation:

$$y(t) = -2t^3 - \frac{9}{2} + 12t + \frac{1}{2} + \int_{-1}^{t} (2t - x)y(x)\,\mathrm{d}x + \int_{-1}^{1} (2t + 3t^2x)y(x)\,\mathrm{d}x,$$

where the exact solution is $y(t) = 3t - 1$. Let $N = 3$. From (9), one can find the following approximated solution:

$$y(t) = -0.999999999947223415 + 3.00000000005277658t - 5.277658485 \times 10^{-11}t^2 - 5.277658485 \times 10^{-11}t^3.$$

The exact solution, approximate solution by this method, presented method in Ezzati and Najafalizadeh (2011) by Chebyshev polynomials, and absolute error of $y(t)$ are shown in Table 3.

Table 3 Exact, Approximation solution, Chebyshev polynomials, and absolute error of $y(t)$ for Example 5

t	Exact	Approximation solution	Chebyshev polynomials	Absolute error of presented method
-1	-4	-4	-3.99954	0.0
-0.75	-3.25	-3.24999999999422756	-3.24952	5.77244×10^{-12}
-0.5	-2.5	-2.49999999998020878	-2.49950	1.979122×10^{-11}
-0.25	-1.75	-1.74999999996289146	-1.74948	3.710854×10^{-11}
0.0	-1	-0.999999999947223420	-0.99945	5.2776580×10^{-11}
0.25	-0.25	-0.249999999938152440	-0.24944	6.1847560×10^{-11}
0.5	0.5	0.500000000059373658	0.50006	5.9373658×10^{-11}
0.75	1.25	1.25000000004040707	1.25061	4.040707×10^{-11}
1.0	2	2	2.00063	0.0

Table 4 Exact, approximation solution, Chebyshev polynomials, and absolute error of $y(t)$ for Example 6

t	Exact	Approximation solution	Chebyshev polynomials	Absolute error of presented method
-1	2.71828	2.7182818272	2.56680	1.25904524×10^{-9}
-0.75	2.117	2.11536846702804688	1.98470	0.00163154958462779
-0.5	1.64872	1.645579629098750	1.60013	0.00314164160137815
-0.25	1.28403	1.28158627648429688	1.27500	0.00243914020344460
0.0	1	0.99999999998	1.00000	2×10^{-11}
0.25	0.77880	0.781373018104296875	0.77501	0.002572235032892007
0.5	0.60653	0.610198177098750	0.60068	0.003667517386116576
0.75	0.47237	0.474908950928046875	0.47456	0.002542398187032168
1.0	0.36788	0.36787944128	0.28520	1.08557678×10^{-10}

Example 6 As the third example consider the following integral equation:

$$y(t) = e^{-t} - e^{t} + \int_{-1}^{t} e^{x+t} y(x)\,\mathrm{d}x + \int_{-1}^{1} \frac{-t}{2} e^{x+t} y(x)\,\mathrm{d}x,$$

where the exact solution is $y(t) = e^{-t}$. Let $N = 4$. From (9), one can find the following approximated solution:

$$y(t) = 0.99999999998 - 0.98877487168t + 0.50104727188t^2 - 0.18642632128t^3 + 0.04203336238t^4.$$

The exact solution, approximate solution by this method, presented method in Ezzati and Najafalizadeh (2011) by Chebyshev polynomials, and absolute error of $y(t)$ are shown in Table 4.

Table 5 Exact, approximation solution, Chebyshev polynomials, and absolute error of $y(t)$ for Example 7

t	Exact	Approximation solution	Chebyshev polynomials	Absolute error of presented method
-1	-2	-2.000000000000000	-2.00093	0.0
-0.75	-1.5	-1.49999999996812012	-1.50076	$3.1879880282031250\times10^{-11}$
-0.5	-1	-0.999999999927740760	-1.00059	$7.22592400125\times10^{-11}$
-0.25	-0.5	-0.499999999933969937	-0.50042	$6.603006251953125\times10^{-11}$
0.0	0	$-4.8724622000000\times10^{-12}$	-0.00025	4.8724622×10^{-12}
0.25	0.5	0.499999999878530063	0.49992	$1.2146993748046875\times10^{-10}$
0.5	1	0.999999999772259240	1.00009	$2.277407599875\times10^{-10}$
0.75	1.5	1.49999999976937988	1.50026	$2.3062011971796875\times10^{-10}$
1.0	2	2.00000000000000000	2.00043	0.0

Example 7 For the following nonlinear Volterra-Fredholm integral equation

$$y(t) = -\frac{1}{3}(7t^4 - 2t - 7) + \int_{-1}^{t} (x+t)(y(x))^2 \, \mathrm{d}x + \int_{-1}^{1} (t-x)y(x) \, \mathrm{d}x,$$

where the exact solution is $y(t) = 2t$. Let $N = 4$. From (9), one can find the following approximated solution:

$$y(t) = -4.8724622 \times 10^{-12} + 1.9999999996t - 3.902550756 \times 10^{-10}t^2 + 4 \times 10^{-10}t^3 + 3.951275378 \times 10^{-10}t^4.$$

The exact solution, approximate solution by this method, presented method in Ezzati and Najafalizadeh (2011) by Chebyshev polynomials, and absolute error of $y(t)$ are shown in Table 5.

Example 8 Consider the following nonlinear Volterra-Fredholm integral equation:

$$y(t) = -\frac{2}{3}t^3 + \frac{11}{2}t^2 + \frac{20}{3}t - \frac{1}{6} + \int_{-1}^{t} (2t-x)y(x) \, \mathrm{d}x + \int_{-1}^{1} (2xt + 3t^2x)(y(x))^2 \, \mathrm{d}x,$$

where the exact solution is $y(t) = t - 1$. Let $N = 4$. From (9), one can find the following approximated solution:

$$y(t) = -0.9999999998 + 1.0000000004t - 4 \times 10^{-10}t^3 - 2 \times 10^{-10}t^4$$

The exact solution, approximate solution by this method, presented method in Ezzati and Najafalizadeh (2011) by Chebyshev polynomials, and absolute error of $y(t)$ are shown in Table 6.

Table 6 Exact, Approximation solution, Chebyshev polynomials, and absolute error of $y(t)$ for Example 8

t	Exact	Approximation solution	Chebyshev polynomials	absolute error of presented method
−1	−2	−2.000000000000000	−1.99995	0.0
−0.75	−1.75	−1.74999999999453125	−1.74994	5.46875×10^{-12}
−0.5	−1.5	−1.49999999996250	−1.50005	3.750×10^{-11}
−0.25	−1.25	−1.24999999989453125	−1.25008	1.0546875×10^{-10}
0.0	−1	−0.9999999998	−1.00008	2×10^{-10}
0.25	−0.75	−0.749999999707031250	−0.75007	2.92968750×10^{-10}
0.5	−0.5	−0.49999999966250	−0.50004	3.3750×10^{-10}
0.75	−0.25	−0.249999999732031250	−0.24996	2.67968750×10^{-10}
1.0	0.0	0.0	0.00009	0.0

5 Conclusions

The Bezier curves are used to derive predictive control for linear time-varying systems with distributed time delay and VDIDEs. The methodology has been tested by different types of VDIDEs. Also, linear and nonlinear Volterra-Fredholm integral equations are solved using Bezier curves method. Nonlinear integral equations are usually difficult to solve analytically. Numerical examples show that the proposed method is efficient and very easy to use.

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

Appendix A

In this Appendix, we specify the derivative of Bezier curve. By (5), we have

$$v_j(t) = \sum_{i=0}^{n} a_i^j B_{i,n}(t), \quad t \in [0, 1],$$

where $B_{i,n}(t) = \frac{n!}{i!(n-i)!} t^i (1-t)^{n-i}$.

Now, we have

$$\frac{\mathrm{d}B_{i,n}(t)}{\mathrm{d}t} = n(B_{i-1,n-1}(t) - B_{i,n-1}(t)), \tag{17}$$

where $B_{-1,n-1}(t) = B_{n,n-1}(t) = 0$, and

$$B_{i-1,n-1}(t) = \frac{(n-1)!}{(i-1)!(n-i)!} t^{i-1}(1-t)^{n-i},$$
$$B_{i,n-1}(t) = \frac{(n-1)!}{i!(n-i-1)!} t^{i}(1-t)^{n-i-1}.$$

By using (17), the first derivative $\mathbf{v}_j(t)$ is shown as

$$\begin{aligned}\frac{\mathrm{d}\mathbf{v}_j(t)}{\mathrm{d}t} &= \sum_{i=1}^{n-1} n\mathbf{a}_i^j B_{i-1,n-1}(t) - \sum_{i=0}^{n-1} n\mathbf{a}_i^j B_{i,n-1}(t)\\ &= \sum_{i=0}^{n-1} n\mathbf{a}_{i+1}^j B_{i,n-1}(t) - \sum_{i=0}^{n-1} n\mathbf{a}_i^j B_{i,n-1}(t)\\ &= \sum_{i=0}^{n-1} B_{i,n-1}(t) n\{\mathbf{a}_{i+1}^j - \mathbf{a}_i^j\}. \end{aligned} \tag{18}$$

Now, we specify the procedure of derivation (7) from (6).
By (5), we have

$$\mathbf{v}_j(t) = \binom{n}{0}\mathbf{a}_0^j \frac{1}{h^n}(t_j - t)^n + \cdots + \binom{n}{n}\mathbf{a}_n^j \frac{1}{h^n}(t - t_{j-1})^n, \tag{19}$$

$$\mathbf{v}_{j+1}(t) = \binom{n}{0}\mathbf{a}_0^{j+1} \frac{1}{h^n}(t_{j+1} - t)^n + \cdots + \binom{n}{n}\mathbf{a}_n^{j+1} \frac{1}{h^n}(t - t_j)^n, \tag{20}$$

By substituting $t = t_j$ into (19) and (20), one has

$$\mathbf{v}_j(t_j) = \mathbf{a}_n^j \frac{1}{h^n}(t_j - t_{j-1})^n, \tag{21}$$

$$\mathbf{v}_{j+1}(t_j) = \mathbf{a}_0^{j+1} \frac{1}{h^n}(t_{j+1} - t_j)^n. \tag{22}$$

To preserve the continuity of Bezier curves at the nodes, one needs to impose the condition $\mathbf{v}_j(t_j) = \mathbf{v}_{j+1}(t_j)$, so from (21) and (22), we have

$$\mathbf{a}_n^j (t_j - t_{j-1})^n = \mathbf{a}_0^{j+1}(t_{j+1} - t_j)^n. \tag{23}$$

From (18), the first derivatives of $\mathbf{v}_j(t)$ and $\mathbf{v}_{j+1}(t)$ are respectively as:

$$\begin{aligned}\frac{\mathrm{d}\mathbf{v}_j(t)}{\mathrm{d}t} &= \sum_{i=0}^{n-1} B_{i,n-1}(t) n(\mathbf{a}_{i+1}^j - \mathbf{a}_i^j)\\ &= \sum_{i=0}^{n-1} \binom{n-1}{i}(t_j - t)^{n-1-i}(t - t_{j-1})^i \frac{1}{h^n}\left\{n(\mathbf{a}_{i+1}^j - \mathbf{a}_i^j)\right\}\\ &= \binom{n-1}{0}\{n(\mathbf{a}_1^j - \mathbf{a}_0^j)\}\frac{1}{h^n}(t_j - t)^{n-1} + \cdots + \binom{n-1}{n-1}\left\{n(\mathbf{a}_n^j - \mathbf{a}_{n-1}^j)\right\}\\ &\quad \times \frac{1}{h^n}(t - t_{j-1})^{n-1}, \end{aligned} \tag{24}$$

$$\begin{aligned}\frac{\mathrm{d}\mathbf{v}_{j+1}(t)}{\mathrm{d}t} &= \sum_{i=0}^{n-1} \binom{n-1}{i}(t_{j+1} - t)^{n-1-i}(t - t_j)^i \frac{1}{h^n}\left\{n(\mathbf{a}_{i+1}^{j+1} - \mathbf{a}_i^{j+1})\right\}\\ &= \binom{n-1}{0}\{n(\mathbf{a}_1^{j+1} - \mathbf{a}_0^{j+1})\}\frac{1}{h^n}(t_{j+1} - t)^{n-1} + \cdots + \binom{n-1}{n-1}\{n(\mathbf{a}_n^{j+1} - \mathbf{a}_{n-1}^{j+1})\}\\ &\quad \times \frac{1}{h^n}(t - t_j)^{n-1}. \end{aligned} \tag{25}$$

By substituting $t = t_j$ into (24) and (25), we have

$$\frac{\mathrm{d}\mathbf{v}_j(t_j)}{\mathrm{d}t} = n(\mathbf{a}_n^j - \mathbf{a}_{n-1}^j)\frac{1}{h^n}(t_j - t_{j-1})^{n-1}, \tag{26}$$

$$\frac{\mathrm{d}\mathbf{v}_{j+1}(t_j)}{\mathrm{d}t} = n(\mathbf{a}_1^{j+1} - \mathbf{a}_0^{j+1})\frac{1}{h^n}(t_{j+1} - t_j)^{n-1}, \tag{27}$$

and to preserve the continuity of the first derivative of the Bezier curves at nodes, by equalizing (26) and (27), we have

$$(\mathbf{a}_n^j - \mathbf{a}_{n-1}^j)(t_j - t_{j-1})^{n-1} = (\mathbf{a}_1^{j+1} - \mathbf{a}_0^{j+1})(t_{j+1} - t_j)^{n-1},$$

where it shows the equality (7).

References

Abbasbandy S (2006) Numerical solutions of the integral equations: Homotopy perturbation method and Adomian's decomposition method. Appl Math Comput 173:493–500

Beltran JV, Monterde J (2004) Bezier solutions of the wave equation. Lect Notes Comput Sci 2:631–640

Bildik N, Inc M (2007) Modified decomposition method for nonlinear Volterra-Fredholm integral equations. Chaos Solitons Fractals 33:308–313

Cerdik-Yaslan H, Akyuz-Dascioglu A (2006) Chebyshev polynomial solution of nonlinear Fredholm-Volterra integro-differential equations. J Arts Sci 5:89–101

Chen W, Zheng WX (2007) Delay-dependent robust stabilization for uncertain neutral systems with distributed delays. Automatica 43:95–104

Cholewa R, Nowak AJ, Bialecki RA, Wrobel LC (2002) Cubic Bezier splines for BEM heat transfer analysis of the 2-D continuous casting problems. Comput Mech 28:282–290

Chu CH, Wang CCL, Tsai CR (2008) Computer aided geometric design of strip using developable Bezier patches. Comput Ind 59(6):601–611

Cui M, Du H (2006) Representation of exact solution for the nonlinear Volterra-Fredholm integral equations. Appl Math Comput 182:1795–1802

Davis PJ (1975) Interpolation and approximation. Dover Publication, New York

El-Hawary HM, El-Shami KA (2013) Numerical solution of Volterra delay-integro-differential equations via spline/spectral methods. Int J Differ Equ Appl 12(3):149–157

Esfanjani RM, Nikravesh SKY (2010) Predictive control for a class of distributed delay systems using Chebyshev polynomials. Int J Comput Math 87(7):1591–1601

Evrenosoglu M, Somali S (2008) Least squares methods for solving singularity perturbed two-points boundary value problems using Bezier control point. Appl Math Lett 21(10):1029–1032

Ezzati R, Najafalizadeh S (2011) Numerical solution of nonlinear Volterra-Fredholm integral equation by using Chebyshev polynomials. Math Sci 5(1):1–12

Farin GE (1988) Curve and surfaces for computer aided geometric design, 1st edn. Academic Press, New York

Fridman E, Shaked U (2002) An improved stabilization method for linear time-delay systems. IEEE Trans Autom Control 47:1931–1937

Gachpazan M (2011) Solving of time varying quadratic optimal control problems by using Bezier control points. Comput Appl Math 30(2):367–379

Ghomanjani F, Farahi MH, Gachpazan M (2012) Bezier control points method to solve constrained quadratic optimal control of time varying linear systems. Comput Appl Math 31(3):433–456

Ghomanjani F, Farahi MH (2012a) Optimal control of switched systems based on bezier control points. Int J Intell Syst Appl 4(7):16-22

Ghomanjani F, Farahi MH (2012b) The Bezier control points method for solving delay differential equation. Intell Control Autom 3(2):188–196

Ghomanjani F, Farahi MH, Kamyad AV (2013a) Numerical solution of some linear optimal control systems with pantograph delays. IMA J Math Control Inf

Ghomanjani F, Kilichman A, Effati S (2013b) Numerical solution for IVP in Volterra type linear integro-differential equations system. Abstr Appl Anal, vol 2013, Article ID 490689, p 4. doi:10.1155/2013/490689

Green CD (1989) Integral equation methods. Nelson, New York

Harada K, Nakamae E (1982) Application of the Bezier curve to data interpolation, computer-aided design. Int J Comput Math 14(1):55–59

Heinkenschloss M (2005) A time-domain decomposition iterative method for the solution of distributed linear quadratic optimal control problems. Appl Math Comput 173:169–198

Horng IR, Chou JH (1985) Analysis, parameter estimation and optimal control of time-delay systems via Chebyshev series. Int J Control 41:1221–1234

Huang C, Vandewalle S (2004) An analysis of delay-dependent stability for ordinary and partial differential equations with fixed and distributed delays. SIAM J Sci Comput 25:1608–1632

Juddu H (2002) Spectral method for constrained linear-quaratic optimal control. Math Comput Simul 58:159–169

Kanwal RP (1996) Linear integral equations theory and technique. Birkhaeuser, Boston

Kauthen JP (1989) Continuous time collocation method for Volterra-Fredholm integral equations. Numer Math 56:409–424

Lan X (2007) Variational iteration method for solving integral equations. Comput Math Appl 54:1071–1078

Lang B (2004) The synthesis of wave forms using Bezier curves with control point modulation. 1st edn. US: Morgan kaufamann, The Second CEMS Research Student Conference

Layton AT, Van de Panne M (2002) A numerically evident and stable algorithm for animating water waves. Vis Comput 18:41–53

Maleknejad K, Derili H (2006) Numerical solution of integral equations by using combination of Spline-collocation method and Lagrange interpolation. Appl Math Comput 175:1235–1244

Manitius A, Olbrot AW (1979) Finite spectrum assignment problem for systems with delays. IEEE Trans Autom Control 24:541–553

N*ü*rnberger G, Zeilfelder F (2000) Developments in bivariate spline interpolation. Journal of Computational and Applied Mathematics. Comput Math Appl 121(1-2):125–152

Nazarzadeh J (1998) Finite time nonlinear optimal systems solution by spectral methods. Amirkabir University of Technology, Ph.D. diss

Rashed MT (2004) Numerical solution of functional differential, integral and integro-differential equations. Appl Math Comput 156:485–492

Rihan FA, Doha EH, Hassan MI, Kamel NM (2009) Numerical treatments for Volterra delay integro-differential equations. Comput Methods Appl Math 9(3):292–308

Watanabe K, Nobuyama E, Kojima K (2006) Recent advances in control of time-delay systems—a tutorial review. In: Paper presented at 35th IEEE Conference on Decision and Control, Kobe, Japan, in Decision and Control, 1996, Proceedings of the 35th IEEE, 2 (11-13 December 2006), pp 2083–2089

Winkel R (2001) Generalized bernstein polynomials and bezier curves: an application of umbral calculus to computer aided geometric design. Adv Appl Math 27(1):51–81

Wu S, Gan S (2008) Analytical and numerical stability of neutral delay integro-differential equations and neutral delay partial differential equations. Comput Math Appl 55:2426–2443

Wu L, Wang C, Gao H, Liu F (2006) Sliding mode control of uncertain systems with distributed delay parameter dependent Lyapunov functional approach. In: Paper presented at the First International Symposium on Systems and Control in Aerospace and Astronautics, Harbin, China, in System and Control in Aerospace and Astronautics, 2006, ISSCAA 2006, 1st International Symposium, pp 935–940

Yalsinbas S (2002) Taylor polynomial solution of nonlinear Volterra-Fredholm integral equations. Appl Math Comput 127:195–206

Yousefi SA, Lotfi A, Dehghan M (2009) He's variational iteration method for solving nonlinear mixed Volterra-Fredholm integral equations. Comput Math Appl 58:2172–2176

Yousefi S, Razzaghi M (2005) Legendre wavelets method for the nonlinear Volterra-Fredholm integral equations. Math Comput Simul 70:1–8

YQ Shi, Sun H (2000) Image and video compression for multimedia engineering. CRC press LLc

Yusufoglu E, Erbas E (2008) Numerical expansion methods for solving Fredholm-Volterra type linear integral equations by interpolation and quadrature rules. Emerald 37(6):768–785

Zheng F, Cheng M, Gao WB (1994) Feedback stabilization of linear systems with distributed delays in state and control variables. IEEE Trans Autom Control 39:1714–1718

Zheng J, Sederberg TW, Johnson RW (2004) Least squares methods for solving differential equations using Bezier control points. Appl Numer Math 48:237–252

Zhong CQ (2004) On distributed delay in linear control laws part I: discrete delay implementations. IEEE Trans Autom Control 49:2074–2080

Zhong CQ (2005) On distributed delay in linear control laws part II: rational implementations inspired from delta operator. IEEE Trans Autom Control 50:729–734