MATHEMATICS

https://doi.org/10.31489/2025M4/5-20 *Research article*

The quadratic $\mathcal{B}$-spline method for approximating systems of Volterra integro fractional-differential equations involving both classical and fractional derivatives

D.Kh. Abdullah, K.H.F. Jwamer*, Sh.Sh. Ahmed

College of Science, University of Sulaimani, Sulaymaniyah, Iraq
(E-mail: Diar.khalid85@gmail.com, karwan.jwamer@univsul.edu.iq, Shazad.ahmed@univsul.edu.iq)

The quadratic $\mathcal{B}$-spline method is a widely recognized numerical technique for solving systems of Volterra integro-differential equations that involve both classical and fractional derivatives (SVIDE's-CF). This study presents an improved application of the quadratic $\mathcal{B}$-spline approach to achieve highly accurate and computationally efficient solutions. In the method developed in this paper, control points are treated as unknown variables within the framework of the approximate solution. The fractional derivative ${}_a^C\mathcal{D}_x^\sigma$ is considered in the Caputo sense. First, we divide the domain into subintervals, then construct quadratic $\mathcal{B}$-spline basis functions over each subinterval. The approximate solution is presented as a quadratic combination of these $\mathcal{B}$-spline functions over each subinterval, where the control points act as variables. To simplify the system of (VIDE's-CF) into a solvable set of algebraic equations, the collocation method is applied by discretizing the equations at chosen points within each subinterval. The Jacobian matrix method is employed to perform computations efficiently. In addition, a careful, step-by-step algorithm for employing the proposed method is presented to simplify its use, we implemented the method in a Python program and optimized it for efficiency. Experimental example demonstrates effectiveness and accuracy of the proposed technique and its comparison with present techniques in terms of accuracy and computational efficiency.

Keywords: system of Volterra integro-fractional differential equation (SVIDE's), quadratic $\mathcal{B}$-spline functions, Caputo fractional derivative, collocation method, Jacobian matrix algorithm, Clenshaw-Curtis quadrature rule.

2020 Mathematics Subject Classification: 34K33, 45D05, 45J05.

Introduction

Mathematicians have extended the classical concepts of differentiation and integration to fractional (non-integer) orders over the centuries [1]. This kind of generalization, which is referred to as fractional calculus (FC), is a more general mathematical framework for investigating complex systems [2]. Compared with the ordinary calculus that deals with essentially local and instantaneous changes, the fractional calculus incorporates memory and hereditary properties and therefore is particularly suitable to model those processes where the present state depends not only on the present status but also on the past history [3, 4]. Several real-life phenomena demonstrate such dynamics [5]. For example, diffusion

*Corresponding author. *E-mail: karwan.jwamer@univsul.edu.iq*
Received: 26 June 2025; *Accepted:* 4 September 2025.

in porous media, viscoelasticity, and biological systems with memory function regularly display dynamics that are not possible to describe through classical models. In all these cases, fractional derivatives give more accurate and flexible descriptions, accounting for long-range temporal and spatial dependencies [6–8]. After that, researchers developed integro-fractional differential equations that combine fractional derivatives with integral terms. Such equations extend traditional differential and integral equations to include both instantaneous rates of change and accumulative past effects at the same time. This makes them powerful tools for modelling dynamic processes where past states exert strong impacts on current and future dynamics [8,9]. Furthermore, fractional integro-differential and integro-differential equations of fractional order have garnered significant interest in the literature, leading to the development of several unique methodologies. Benzahi et al. demonstrated a least squares method [10]. Ghosh et al. presented an iterative difference scheme for solving an arbitrary order nonlinear Volterra integro-differential population growth model [11]. Rahimkhani et al. illustrated nonlinear fractional integro-differential equations using fractional alternative Legendre functions [12]. Akbar et al. presented an analysis of delay [13]. Miran et al. presented Laplace transform multi-time delay [14]. Akhlaghi et al. addressed fractional order integro-differential equations via Muntz orthogonal functions [15]. Yuzbai et al. presented a fractional Bell collocation method [16]. In practice, most linear Volterra integro-fractional differential systems with variable coefficients are too complex to solve exactly using analytical methods. Because of this, researchers often turn to approximation techniques and numerical methods. One of the most common and effective tools for this purpose is the use of spline and $\mathcal{B}$-spline functions [17, 18]. These functions play a crucial role in solving both linear and nonlinear functional equations. Many researchers use $\mathcal{B}$-spline functions to solve various mathematical problems because of their flexibility and accuracy [19–21].

This study presents an approximate method for solving the linear system associated with Volterra integro-differential equations, encompassing classical and fractional orders (LSVIDE's-CF). For the derivation, it deals with quadratic $\mathcal{B}$-spline interpolation functions. Which takes the following general forms:

$$\mathcal{P}_i(x)\mathcal{U}_i''(x) + a_{i0}(x)\,{}_a^C\mathcal{D}_x^{\sigma_{i0}}\mathcal{U}_i(x) + a_{i1}(x)\,{}_a^C\mathcal{D}_x^{\sigma_{i1}}\mathcal{U}_i(x) + a_{i2}(x)\mathcal{U}_i(x) = \mathcal{F}_i(x) + \sum_{j=0}^{m}\omega_{ij}\int_a^x \mathcal{K}_{ij}(x,s)\,{}_a^C\mathcal{D}_s^{\beta_{ij}}\mathcal{U}_j(s)\,ds. \quad (1)$$

Under the following conditions:

$$\left[\mathcal{D}_x^{k_i}\mathcal{U}_i(x)\right]_{x=a} = \vartheta_{ik_i}, \forall k_i = 0,1,\ldots,\mu_i - 1, \text{ and } i = 0,1,\ldots,m. \quad (2)$$

The variable coefficients $\mathcal{P}_i(x)(\not\equiv 0), a_{i0}(x),\ a_{i1}(x) \in C([a,b],\mathbb{R})$ and $\mathcal{K}_{ij} \in C(\Theta,\mathbb{R}),\ \Theta = \{(x,s) : a \le s < x \le b\}$, with fractional orders: $\sigma_{i1} > \sigma_{i0} > 0$ and $\beta_{im} > \beta_{i(m-1)} > \cdots > \beta_{i1} > \beta_{i0} = 0$. Furthermore, the $\mu_i = \max\left\{2, m_{im}^{\beta}\right\}$, where $m_{ij}^{\beta} - 1 < \beta_{ij} \le m_{ij}^{\beta}, m_{ij}^{\beta} = \lceil \beta_{ij} \rceil,\ \omega_{ij} \in \mathbb{R}$, for all $i,j = 0,1,\ldots,m$. In the manuscript, we examined and assessed the systems of Volterra integro-differential equations for classical and fractional orders (SVIDE's-CF); according to the conditions, fractional orders between 0 and 1. We approximate these integrals using the Clenshaw-Curtis quadrature rule [17, 22] in conjunction with quadratic $\mathcal{B}$-spline functions. Four algorithms summarized the information, and we later produced Python software to implement each algorithm. These algorithms resolved a few test instances. The paper is structured as follows: in Section 1, we introduce some notions of fractional calculus necessary for the description of our model, and then we define the fundamental concepts of $\mathcal{B}$-spline functions. In Section 2, we introduce the numerical approximation that we use throughout our work. The experimental outcomes are presented in Section 3. Lastly, the concluding remarks on the proposed method are presented in Section 4.

1 Basic definitions and notation

In this section, we will introduce and study the concepts.

Definition 1. [1,2] Let $n-1<\alpha\leq n$ $(\in\mathbb{Z}^+)$, $\alpha\in\mathbb{R}^+$. The operators $({}_a^R\mathcal{D}_x^\alpha\mathcal{V}(x))$ and $({}_a^C\mathcal{D}_x^\alpha\mathcal{V}(x))$ of fractional order α are defined as:

$$ {}_a^R\mathcal{D}_x^\alpha\mathcal{V}(x)=\mathcal{D}_x^n\left({}_a\mathcal{J}_x^{n-\alpha}\mathcal{V}(x)\right)=\frac{1}{\Gamma(n-\alpha)}\frac{d^n}{dx^n}\int_a^x\frac{\mathcal{V}(s)}{(x-s)^{\alpha+1-n}}ds,\quad x>a, \tag{3}$$

$$ {}_a^C\mathcal{D}_x^\alpha\mathcal{V}(x)={}_a\mathcal{J}_x^{n-\alpha}\mathcal{D}_x^n\mathcal{V}(x)=\frac{1}{\Gamma(n-\alpha)}\int_a^x\frac{\mathcal{V}^{(n)}(s)}{(x-s)^{\alpha+1-n}}ds,\quad x>a. \tag{4}$$

Equation (3) represents the *Riemann–Liouville fractional differential operator*. Additionally, the operator ${}_a\mathcal{J}_x^\alpha$, known as the *Riemann–Liouville fractional integral* of order α, is defined as

$${}_a\mathcal{J}_x^\alpha\mathcal{V}(x)=\frac{1}{\Gamma(\alpha)}\int_a^x(x-s)^{\alpha-1}\mathcal{V}(s)ds,\quad {}_a\mathcal{J}_x^0\mathcal{V}(x)=\mathcal{V}(x),\quad x>a.$$

Equation (4) defines the *Caputo fractional differential operator*. Similar to integer-order differentiation, the Caputo fractional differentiation is a linear operation:

$${}_a^C\mathcal{D}_x^\alpha\left[\rho_1\mathcal{V}_1(x)+\rho_2\mathcal{V}_2(x)\right]=\rho_1{}_a^C\mathcal{D}_x^\alpha\mathcal{V}_1(x)+\rho_2{}_a^C\mathcal{D}_x^\alpha\mathcal{V}_2(x).$$

Furthermore, the Caputo derivative of any constant function (say $\mathcal{A}\in\mathbb{R}$) vanishes: ${}_a^C\mathcal{D}_x^\alpha\mathcal{A}=0$.

Lemma 1. [1,9] The function $\mathcal{V}(x)=(x-a)^n$ for $n\geq 0$ has a β-Caputo derivative ($\beta\geq 0$), which is given as follows: for $n\in\{0,1,2,\ldots,\lceil\beta\rceil-1\}$, the β-Caputo derivative vanishes, i.e., ${}_a^C\mathcal{D}_x^\beta\mathcal{V}(x)=0$. While for $n\in\mathbb{N}$ and $n\geq\lceil\beta\rceil$ or $n\notin\mathbb{N}$ and $n>\lceil\beta\rceil-1$, where $\lceil\beta\rceil$ represents the least integer that is not less than β, it is given by:

$${}_a^C\mathcal{D}_x^\beta\mathcal{V}(x)=\frac{\Gamma(n+1)}{\Gamma(n-\beta+1)}(x-a)^{n-\beta}.$$

Definition 2. [17,22] In 1960, Clenshaw and Curtis established a method for evaluating a definite integral by representing the integrand through a finite Chebyshev series. This involves sequentially summing the individual terms the series. This approach proves to be highly effective, especially when applied to integral equations.

$$\int_{-1}^{1}\mathcal{V}(x)\,dx\approx\sum_{\substack{r=0\\ \text{even}}}^{\mathcal{N}}\left(\frac{2}{\mathcal{N}}\sum_{K=0}^{\mathcal{N}}\cos\left(\frac{rK\pi}{\mathcal{N}}\right)\mathcal{V}\left(\cos\left(\frac{K\pi}{\mathcal{N}}\right)\right)\right),\quad K=0,1,\ldots,\mathcal{N}.$$

Remark:

(I) The symbol $\sum$ indicates that the initial and final terms should be divided by two before summation.

(II) The transformation $x=\frac{b-a}{2}t+\frac{b+a}{2}$, or $t=2\left(\frac{x-a}{b-a}\right)-1$, where $x\in[a,b]$ and $t\in[-1,1]$.

Definition 3. [23] Let $\mathcal{T}_\mathcal{N}=\{x_0,x_1,\ldots,x_\mathcal{N}\}$ be a uniform or non-uniform partition of the interval $[a,b]$. The $\mathcal{K}$-degree $\mathcal{B}$-spline basis function $\mathcal{B}_r^\mathcal{K}(x)$, $r\geq 0$, is defined as follows:

$$\mathcal{B}_r^\mathcal{K}(x)=\frac{x-x_r}{x_{r+\mathcal{K}}-x_r}\mathcal{B}_r^{\mathcal{K}-1}(x)+\frac{x_{r+\mathcal{K}+1}-x}{x_{r+\mathcal{K}+1}-x_{r+1}}\mathcal{B}_{r+1}^{\mathcal{K}-1}(x),$$

$$\mathcal{B}_r^0(x) = \begin{cases} 1, & x \in [x_r, x_{r+1}), \\ 0, & \text{otherwise.} \end{cases} \tag{5}$$

$$\mathcal{B}_r^1(x) = \begin{cases} \frac{x-x_r}{x_{r+1}-x_r}, & \text{if } x \in [x_r, x_{r+1}), \\ \frac{x_{r+2}-x}{x_{r+2}-x_{r+1}}, & \text{if } x \in [x_{r+1}, x_{r+2}), \\ 0, & \text{otherwise.} \end{cases} \tag{6}$$

Equations (5) and (6) represent zero-degree and one-degree $\mathcal{B}$-splines, respectively [24, 25].

Note that the local support property is $\mathcal{B}_r^{\mathcal{K}}(x) = 0$ for all $x \notin [x_r, x_{r+\mathcal{K}+1})$ and the nonnegativity property is $\mathcal{B}_r^{\mathcal{K}}(x) \geq 0$ for all $x \in [x_r, x_{r+\mathcal{K}+1})$.

2 Methods analysis

In this section, we utilize the quadratic $\mathcal{B}$-spline collocation method to compute the approximate solution (SVIDE's-CF) of equation (1) subject to equation (2), where the mesh points $a = x_0 < x_1 < x_2 < \cdots < x_{\mathcal{N}-1} < x_{\mathcal{N}} = b$ form a uniform partition of the solution domain $[a, b]$ defined by the knots x_r with $h = \frac{x_{r+1}-x_r}{\mathcal{N}} = \frac{b-a}{\mathcal{N}}$, $r = 0, 1, \ldots, \mathcal{N}-1$. The numerical solution for treating equations (1) and (2) for all $x \in [a, b]$, $\mathcal{U}_i(x) \approx \mathbb{P}_i^{\mathcal{Q},2}(x)$ for each $i = 0, 1, \ldots, m$, using collocation techniques with quadratic $\mathcal{B}$-spline to find an approximate solution $\mathbb{P}_i^{\mathcal{Q},2}(x)$ given by:

$$\mathbb{P}_i^{\mathcal{Q},2}(x) \approx \sum_{l=0}^{n} \mathcal{C}_i^l \mathcal{B}_{il}^k(x), \quad i = 0, 1, \ldots, m. \tag{7}$$

Here, the general expression of a quadratic $\mathcal{B}$-spline curve defined on the interval $[a, b]$ is,

$$\mathbb{P}_i^{\mathcal{Q},2}(x) = \mathcal{C}^0 \left(\frac{b-x}{b-a}\right)^2 + 2\mathcal{C}^1 \left(\frac{x-a}{b-a}\right)\left(\frac{b-x}{b-a}\right) + \mathcal{C}^2 \left(\frac{x-a}{b-a}\right)^2, \quad x \in [a, b]. \tag{8}$$

Now, the Caputo fractional derivative of order $\alpha \in (0, 1]$, and the recursive derivative formula for quadratic $\mathcal{B}$-spline curves for equation (8) are given, respectively:

$$\begin{aligned} &{}_a^C\mathcal{D}_x^\alpha \mathbb{P}_i^{\mathcal{Q},2}(x) \\ &\quad = \frac{2(x-a)^{1-\alpha}}{(\mathcal{N}h)^2\Gamma(3-\alpha)} \left\{\mathcal{C}^0[(x-a) - (\mathcal{N}h)(2-\alpha)] + \mathcal{C}^1[(\mathcal{N}h)(2-\alpha) - (x-a)] + \mathcal{C}^2(x-a)\right\}, \end{aligned} \tag{9}$$

$$\frac{d^2}{dx^2}\mathbb{P}_i^{\mathcal{Q},2}(x) = \frac{2\left(\mathcal{C}^0 - 2\mathcal{C}^1 + \mathcal{C}^2\right)}{(\mathcal{N}h)^2}, \tag{10}$$

where h is a step size and $\mathcal{N}$ is the number of iterations. For the numerical approximate solutions of (SVIDE's-CF) based on equation (1) the control points $\mathcal{C}_i^l$ are unknowns. Also, for all $i = 0, 1, \ldots, m$, the control points $\mathcal{C}_i^0$ are determined by initial conditions specified in equation (1), and the control points $\mathcal{C}_i^1 = \frac{(\mathcal{N}h)}{2}\frac{(d\mathcal{B}_i^2(a))}{dt} + \mathcal{C}_i^0$ for the quadratic $\mathcal{B}$-spline curve can be determined for each $i = 0, 1, \ldots, m$, to find $\mathcal{C}_i^2$, from the (VIDE's-CF) in equation (1). The unknown function $\mathbb{P}_i^{\mathcal{Q},2}(x)$ is approximated by $\mathcal{B}$-spline interpolation of degree 2 as in equation (7), so the equation (1) becomes:

$$\begin{aligned} &\mathcal{P}_i(x)\frac{d^2}{dx^2}\mathbb{P}_i^{\mathcal{Q},2}(x) + a_{i0}(x)\,{}_a^C\mathcal{D}_x^{\sigma_{i0}}[\mathbb{P}_i^{\mathcal{Q},2}(x)] + a_{i1}(x)\,{}_a^C\mathcal{D}_x^{\sigma_{i1}}[\mathbb{P}_i^{\mathcal{Q},2}(x)] + a_{i2}(x)\mathbb{P}_i^{\mathcal{Q},2}(x) \\ &\qquad = \mathcal{F}_i(x) + \sum_{j=0}^{m} \omega_{ij} \int_a^x K_{ij}(x, s)\,{}_a^C\mathcal{D}_s^{\beta_{ij}}[\mathbb{P}_j^{\mathcal{Q},2}(s)]\,ds. \end{aligned} \tag{11}$$

The fractional orders $\sigma_{i0,i1}, \beta_{ij} \in (0,1]$, $\forall i, j = 0, 1 \ldots, m$. Consequently, using equation (11), applying the linearity property of fractional Caputo derivatives, and using equation (8), by defining $[x_r, x_{r+1}]$ and analyzing the collocation points, we use a quadratic $\mathcal{B}$-spline function ($\mathcal{K} = 2$, $n = 2$) in the interval $[x_r, x_{r+1}]$ as established, from equations (9), and (10), yielding results for each $r = 0, 1, \ldots, \mathcal{N}-1$ and $i = 0, 1, \ldots, m$, derive the following:

$$
\begin{aligned}
&\mathcal{P}_i(x_{r+1})\left[\tfrac{2\mathcal{C}_i^0}{(\mathcal{N}h)^2} - \tfrac{4\mathcal{C}_i^1}{(\mathcal{N}h)^2} - \tfrac{2\mathcal{C}_i^2}{(\mathcal{N}h)^2}\right] + a_{i0}(x_{r+1})\frac{(x_{r+1}-a)^{1-\sigma_{i0}}}{(\mathcal{N}h)^2\Gamma(3-\sigma_{i0})}\begin{bmatrix}\mathcal{C}_i^0(-2(\mathcal{N}h)(2-\sigma_{i0})+(x_{r+1}-a))\\ +2\mathcal{C}_i^1((\mathcal{N}h)(2-\sigma_{i0})-2(x_{r+1}-a))\\ +2\mathcal{C}_i^2(x_{r+1}-a)\end{bmatrix}\\
&+ a_{i1}(x_{r+1})\frac{(x_{r+1}-a)^{1-\sigma_{i1}}}{(\mathcal{N}h)^2\Gamma(3-\sigma_{i1})}\begin{bmatrix}\mathcal{C}_i^0(-2(\mathcal{N}h)(2-\sigma_{i1})+(x_{r+1}-a))\\ +2\mathcal{C}_i^1((\mathcal{N}h)(2-\sigma_{i1})-2(x_{r+1}-a))\\ +2\mathcal{C}_i^2(x_{r+1}-a)\end{bmatrix} + a_{i2}(x_{r+1})\begin{bmatrix}\frac{\mathcal{C}_i^0(b-x_{r+1})^2}{(\mathcal{N}h)^2}+\\ \frac{2\mathcal{C}_i^1(x_{r+1}-a)(b-x_{r+1})}{(\mathcal{N}h)^2}\\ +\frac{\mathcal{C}_i^2(x_{r+1}-a)^2}{(\mathcal{N}h)^2}\end{bmatrix}\\
&= \mathcal{F}_i(x_{r+1}) + \sum_{j=1}^{m}\omega_{ij}\left\{\sum_{q=0}^{r-1}\int_{x_q}^{x_{q+1}}\mathcal{K}_{ij}(x_{r+1},s)\frac{(s-a)^{1-\beta_{ij}}}{(\mathcal{N}h)^2\Gamma(3-\beta_{ij})}\begin{bmatrix}\mathcal{C}_j^0(-2(\mathcal{N}h)(2-\beta_{ij})+2(s-a))\\ +2\mathcal{C}_j^1((\mathcal{N}h)(2-\beta_{ij})-2(s-a))\\ +2\mathcal{C}_j^2(s-a)\end{bmatrix}ds\right\}\\
&+ \int_{x_r}^{x_{r+1}}\mathcal{K}_{ij}(x_{r+1},s)\frac{(s-a)^{1-\beta_{ij}}}{(\mathcal{N}h)^2\Gamma(3-\beta_{ij})}\begin{bmatrix}\mathcal{C}_j^0(-2(\mathcal{N}h)(2-\beta_{ij})+2(s-a))+2\mathcal{C}_j^1\\ ((\mathcal{N}h)(2-\beta_{ij})-2(s-a))+2\mathcal{C}_j^2(s-a)\end{bmatrix}ds\\
&+ \omega_{i0}\left\{\sum_{q=0}^{r-1}\int_{x_q}^{x_{q+1}}\mathcal{K}_{i0}(x_{r+1},s)\left[\tfrac{\mathcal{C}_0^0}{(\mathcal{N}h)^2}(b-s)+\tfrac{2\mathcal{C}_0^1}{(\mathcal{N}h)^2}(s-a)(b-s)+\tfrac{\mathcal{C}_0^2}{(\mathcal{N}h)^2}(s-a)^2\right]ds\right.\\
&\quad\left.+\int_{x_r}^{x_{r+1}}\mathcal{K}_{i0}(x_{r+1},s)\left[\tfrac{\mathcal{C}_0^0}{(\mathcal{N}h)^2}(b-s)+\tfrac{2\mathcal{C}_0^1}{(\mathcal{N}h)^2}(s-a)(b-s)+\tfrac{\mathcal{C}_0^2}{(\mathcal{N}h)^2}(s-a)^2\right]ds\right\}.
\end{aligned}
\tag{12}
$$

The quadratic $\mathcal{B}$-spline function throughout the interval $[x_r, x_{r+1}]$ is optimized to simplify its representation and promote efficient solution methods; it is also determined, and in practice, these integrals must be approximated using the Clenshaw-Curtis quadrature rule; hence, the equation (12) is applicable for $r = 0, 1, \ldots, \mathcal{N}-1$, $i = 0, 1, \ldots, m$.

$$
\begin{aligned}
\mathcal{H}_i^r(\sigma_{i0})\mathcal{C}_i^2 + \mathcal{O}_i^r(\sigma_{i0})\mathcal{C}_i^1 + \mathcal{T}_i^r(\sigma_{i0})\mathcal{C}_i^0 &= \mathcal{F}_i(x_{r+1}) + \sum_{j=1}^{m}\omega_{ij}\left(\mathcal{A}_{K_{ij}}^{\beta_{ij}}\right)\mathcal{C}_j^2 + \sum_{j=1}^{m}\omega_{ij}\left(S_{K_{ij}}^{\beta_{ij}}\right)\mathcal{C}_j^1\\
&\quad + \sum_{j=1}^{m}\omega_{ij}\left(\mathcal{X}_{K_{ij}}^{\beta_{ij}}\right)\mathcal{C}_j^0 + \left(\omega_{i0}^{(r)}\mathcal{Y}_{K_{i0}}\right)\mathcal{C}_0^0 + \left(\omega_{i0}^{(r)}\bar{\mathcal{Y}}_{K_{i0}}\right)\mathcal{C}_0^1 + \left(\omega_{i0}^{(r)}\breve{\mathcal{Y}}_{K_{i0}}\right)\mathcal{C}_0^2,
\end{aligned}
$$

where

$$
\begin{aligned}
\mathcal{H}_i^r(\sigma_{i0}) = \frac{2\mathcal{P}_i(x_{r+1})}{(\mathcal{N}h)^2} &+ \frac{2a_{i0}(x_{r+1})\left((r+1)h\right)^{1-\sigma_{i0}}(x_{r+1}-a)}{(\mathcal{N}h)^2\Gamma(3-\sigma_{i0})}\\
&+ \frac{2a_{i1}(x_{r+1})\left((r+1)h\right)^{1-\sigma_{i1}}(x_{r+1}-a)}{(\mathcal{N}h)^2\Gamma(3-\sigma_{i1})} + \frac{a_{i2}(x_{r+1})(x_{r+1}-a)^2}{(\mathcal{N}h)^2},
\end{aligned}
$$

$$\mathcal{O}_i^r = \frac{-4\mathcal{P}_i(x_{r+1})}{(\mathcal{N}h)^2} + \frac{2a_{i0}(x_{r+1})\left((r+1)h\right)^{1-\sigma_{i0}}}{(\mathcal{N}h)^2\Gamma(3-\sigma_{i0})}\left(-2(\mathcal{N}h)^{2-\sigma_{i0}} + 2(x_{r+1}-a)\right)$$
$$+ \frac{2a_{i1}(x_{r+1})\left((r+1)h\right)^{1-\sigma_{i1}}}{(\mathcal{N}h)^2\Gamma(3-\sigma_{i1})}\left(-2(\mathcal{N}h)^{2-\sigma_{i1}} + 2(x_{r+1}-a)\right) + \frac{2a_{i2}(x_{r+1})(x_{r+1}-a)}{(\mathcal{N}h)^2(b-x_{r+1})},$$

$$\mathcal{T}_i^r = \frac{2\mathcal{P}_i\left(x_{r+1}\right)}{(\mathcal{N}h)^2} + \frac{a_{i0}\left(t_{r+1}\right)\left((r+1)h\right)^{1-\sigma_{i0}}}{(\mathcal{N}h)^2\Gamma(3-\sigma_{i0})}\left\{-2(\mathcal{N}h)\left(2-\sigma_{i0}\right) + 2\left(x_{r+1}-a\right)\right\}$$
$$+ \frac{a_{i1}\left(x_{r+1}\right)\left((r+1)h\right)^{1-\sigma_{i1}}}{(\mathcal{N}h)^2\Gamma(3-\sigma_{i1})}\left\{-2(\mathcal{N}h)\left(2-\sigma_{i1}\right) + 2\left(x_{r+1}-a\right)\right\} + \frac{a_{i2}\left(x_{r+1}\right)}{(\mathcal{N}h)^2}\left(b-x_{r+1}\right)^2,$$

$${}_r\mathcal{A}_{\mathcal{K}_{ij}}^{\beta_{ij}} = \sum_{q=0}^{r-1}\sum_{k=0}^{\mathcal{N}}\left[\frac{x_{q+1}-x_q}{2}\omega_k\mathcal{K}_{ij}(x_{r+1},S_k)\frac{\left(2(S_k-a)\right)^{1-\beta_{ij}}\left(S_k-a\right)}{(\mathcal{N}h)^2\Gamma(3-\beta_{ij})}\right]$$
$$+\sum_{k=0}^{\mathcal{N}}\left[\frac{x_{r+1}-x_r}{2}\omega_k\mathcal{K}_{ij}(x_{r+1},S_k)\frac{\left(2(S_k-a)\right)^{1-\beta_{ij}}\left(S_k-a\right)}{(\mathcal{N}h)^2\Gamma(3-\beta_{ij})}\right], \quad (13)$$

$${}_r\mathcal{S}_{\mathcal{K}_{i0}}^{\beta_{ij}} = \sum_{q=0}^{r-1}\sum_{k=0}^{\mathcal{N}}\left[\frac{x_{q+1}-x_q}{2}\omega_k\mathcal{K}_{ij}(x_{r+1},S_k)\frac{2(S_k-a)^{1-\beta_{ij}}}{(\mathcal{N}h)^2\Gamma(3-\beta_{ij})}\right]\left[(\mathcal{N}h)(2-\beta_{ij}) - 2(S_k-a)\right]$$
$$+\sum_{k=0}^{\mathcal{N}}\left[\frac{x_{r+1}-x_r}{2}\omega_k\mathcal{K}_{ij}(x_{r+1},S_k)\frac{2(S_k-a)^{1-\beta_{ij}}}{(\mathcal{N}h)^2\Gamma(3-\beta_{ij})}\right]\left[(\mathcal{N}h)(2-\beta_{ij}) - 2(S_k-a)\right], \quad (14)$$

$${}_r\mathcal{X}_{\mathcal{K}_{ij}}^{\beta_{ij}} = \sum_{q=0}^{r-1}\sum_{k=0}^{\mathcal{N}}\left[\frac{x_{q+1}-x_q}{2}\omega_k\mathcal{K}_{ij}(x_{r+1},S_k)\frac{(S_k-a)^{1-\beta_{ij}}}{(\mathcal{N}h)^2\Gamma(3-\beta_{ij})}\right]\left[-2(\mathcal{N}h)(2-\beta_{ij}) + 2(S_k-a)\right]$$
$$+\sum_{k=0}^{\mathcal{N}}\left[\frac{x_{r+1}-x_r}{2}\omega_k K_{ij}(x_{r+1},S_k)\frac{(S_k-a)^{1-\beta_{ij}}}{(\mathcal{N}h)^2\Gamma(3-\beta_{ij})}\right]\left[-2(\mathcal{N}h)(2-\beta_{ij})\right], \quad (15)$$

$${}_r^{\omega_{i0}}\mathcal{Y}_{\mathcal{K}_{i0}} = \omega_{i0}\left\{\sum_{q=0}^{r-1}\sum_{k=0}^{\mathcal{N}}\left[\frac{x_{q+1}-x_q}{2}\omega_k\mathcal{K}_{i0}(x_{r+1},S_k)\frac{(b-S_k)^2}{(\mathcal{N}h)^2}\right]\right.$$
$$\left.+\sum_{k=0}^{\mathcal{N}}\left[\frac{x_{r+1}-x_r}{2}\omega_k\mathcal{K}_{i0}(x_{r+1},S_k)\frac{(b-S_k)^2}{(\mathcal{N}h)^2}\right]\right\}, \quad (16)$$

$${}_r^{\omega_{i0}}\overline{\mathcal{Y}}_{\mathcal{K}_{i0}} = \omega_{i0}\left\{\sum_{q=0}^{r-1}\sum_{k=0}^{\mathcal{N}}\left[\frac{x_{q+1}-x_q}{2}\omega_k\mathcal{K}_{i0}(x_{r+1},S_k)\frac{2(S_k-a)(b-S_k)}{(\mathcal{N}h)^2}\right]\right.$$
$$\left.+\sum_{k=0}^{\mathcal{N}}\left[\frac{x_{r+1}-x_r}{2}\omega_k\mathcal{K}_{i0}(x_{r+1},S_k)\frac{2(S_k-a)(b-S_k)}{(\mathcal{N}h)^2}\right]\right\}, \quad (17)$$

$$
{}_{r}^{\omega_{i0}}\breve{\mathcal{Y}}_{\mathcal{K}_{i0}} = \omega_{i0}\left\{\sum_{q=0}^{r-1}\sum_{k=0}^{\mathcal{N}}\left[\frac{x_{q+1}-x_q}{2}\omega_k\mathcal{K}_{i0}(x_{r+1},S_k)\frac{(S_k-a)^2}{(\mathcal{N}h)^2}\right]\right.
$$

$$
\left.+\sum_{k=0}^{\mathcal{N}}\left[\frac{x_{r+1}-x_r}{2}\omega_k\mathcal{K}_{i0}(x_{r+1},S_k)\frac{(S_k-a)^2}{(\mathcal{N}h)^2}\right]\right\}. \quad (18)
$$

From equations (13)–(18) the nodes t_k, where the integrand is evaluated at points corresponding to the extrema of the Chebyshev polynomials on the interval $[-1,1]$ and are defined as $t_k=\cos\left(\frac{k\pi}{\mathcal{N}}\right)$ for $k=0,1,\ldots,\mathcal{N}$, for each subinterval $[x_q,x_{q+1}]$, the mapped node S_k is calculated by $S_k=\frac{x_{q+1}-x_q}{2}t_k+\frac{x_{q+1}+x_q}{2}$. The weights ω_k are coefficients that multiply the function values at the nodes to approximate the integral given by $\omega_k=\frac{2}{\mathcal{N}}\sum_{\substack{r=0\\ \text{even}}}^{\mathcal{N}}\cos\left(\frac{rk\pi}{\mathcal{N}}\right)$. These weights help the weighted sum of function evaluations accurately represent the integral over the chosen interval, and are determined following a linear system $(m+1)\times(m+1)$ of algebraic equations, which is provided.

$$
\mathbb{A}\cdot\mathbb{B}=\mathbb{C}, \quad (19)
$$

where

$$
\mathbb{A}=\begin{bmatrix}
\mathcal{H}_0^r(\sigma_{00})-{}_{r}^{\omega_{00}}\breve{\mathcal{Y}}_{\mathcal{K}_{00}} & -\omega_{01} & {}_{r}\mathcal{A}_{\mathcal{K}_{01}}^{\beta_{01}} & -\omega_{02} & {}_{r}\mathcal{A}_{\mathcal{K}_{02}}^{\beta_{02}} & \cdots & -\omega_{0m} & {}_{r}\mathcal{A}_{\mathcal{K}_{0m}}^{\beta_{0m}}\\
-{}_{r}^{\omega_{10}}\breve{\mathcal{Y}}_{\mathcal{K}_{10}} & -\omega_{11} & {}_{r}\mathcal{A}_{\mathcal{K}_{11}}^{\beta_{11}} & -\omega_{12} & {}_{r}\mathcal{A}_{\mathcal{K}_{12}}^{\beta_{12}} & \cdots & -\omega_{1m} & {}_{r}\mathcal{A}_{\mathcal{K}_{1m}}^{\beta_{1m}}\\
-{}_{r}^{\omega_{20}}\breve{\mathcal{Y}}_{\mathcal{K}_{20}} & -\omega_{21} & {}_{r}\mathcal{A}_{\mathcal{K}_{21}}^{\beta_{21}} & -\omega_{22} & {}_{r}\mathcal{A}_{\mathcal{K}_{22}}^{\beta_{22}} & \cdots & -\omega_{2m} & {}_{r}\mathcal{A}_{\mathcal{K}_{2m}}^{\beta_{2m}}\\
\vdots & & \vdots & & \vdots & \ddots & \vdots & \\
-{}_{r}^{\omega_{m0}}\breve{\mathcal{Y}}_{\mathcal{K}_{m0}} & -\omega_{m1} & {}_{r}\mathcal{A}_{\mathcal{K}_{m1}}^{\beta_{m1}} & -\omega_{m2} & {}_{r}\mathcal{A}_{\mathcal{K}_{m2}}^{\beta_{m2}} & \cdots & \mathcal{H}_m^r(\sigma_{m0})-\omega_{mm} & {}_{r}\mathcal{A}_{\mathcal{K}_{mm}}^{\beta_{mm}}
\end{bmatrix}, \quad (20)
$$

$$
\mathbb{B}=\begin{bmatrix}\mathcal{C}_0^2 & \mathcal{C}_1^2 & \mathcal{C}_2^2 & \ldots & \mathcal{C}_m^2\end{bmatrix}^T, \quad (21)
$$

$$
\mathbb{C}=\begin{bmatrix}
\mathcal{F}_0(x_{r+1})-\mathcal{O}_0^r\mathcal{C}_0^1-\mathcal{T}_0^r\mathcal{C}_0^0+\sum_{j=1}^m\omega_{0j}\left(\mathcal{S}_{\mathcal{K}_{0j}}^{\beta_{0j}}\right)\mathcal{C}_j^1+\sum_{j=1}^m\omega_{0j}\left(\mathcal{X}_{\mathcal{K}_{0j}}^{\beta_{0j}}\right)\mathcal{C}_j^0+(\omega_{00}^r\mathcal{Y}_{\mathcal{K}_{00}})\mathcal{C}_0^0+(\omega_{00}^r\bar{\mathcal{Y}}_{\mathcal{K}_{00}})\mathcal{C}_0^1\\
\mathcal{F}_1(x_{r+1})-\mathcal{O}_1^r\mathcal{C}_1^1-\mathcal{T}_1^r\mathcal{C}_1^0+\sum_{j=1}^m\omega_{1j}\left(\mathcal{S}_{\mathcal{K}_{1j}}^{\beta_{1j}}\right)\mathcal{C}_j^1+\sum_{j=1}^m\omega_{1j}\left(\mathcal{X}_{\mathcal{K}_{1j}}^{\beta_{1j}}\right)\mathcal{C}_j^0+(\omega_{10}^r\mathcal{Y}_{\mathcal{K}_{10}})\mathcal{C}_0^0+(\omega_{10}^r\bar{\mathcal{Y}}_{\mathcal{K}_{10}})\mathcal{C}_0^1\\
\mathcal{F}_2(x_{r+1})-\mathcal{O}_2^r\mathcal{C}_2^1-\mathcal{T}_2^r\mathcal{C}_2^0+\sum_{j=1}^m\omega_{2j}\left(\mathcal{S}_{\mathcal{K}_{2j}}^{\beta_{2j}}\right)\mathcal{C}_j^1+\sum_{j=1}^m\omega_{2j}\left(\mathcal{X}_{\mathcal{K}_{2j}}^{\beta_{2j}}\right)\mathcal{C}_j^0+(\omega_{20}^r\mathcal{Y}_{\mathcal{K}_{20}})\mathcal{C}_0^0+(\omega_{20}^r\bar{\mathcal{Y}}_{\mathcal{K}_{20}})\mathcal{C}_0^1\\
\vdots\\
\mathcal{F}_m(x_{r+1})-\mathcal{O}_m^r\mathcal{C}_m^1-\mathcal{T}_m^r\mathcal{C}_m^0+\sum_{j=1}^m\omega_{mj}\left(\mathcal{S}_{\mathcal{K}_{mj}}^{\beta_{mj}}\right)\mathcal{C}_j^1+\sum_{j=1}^m\omega_{mj}\left(\mathcal{X}_{\mathcal{K}_{mj}}^{\beta_{mj}}\right)\mathcal{C}_j^0+(\omega_{m0}^r\mathcal{Y}_{\mathcal{K}_{m0}})\mathcal{C}_0^0+(\omega_{m0}^r\bar{\mathcal{Y}}_{\mathcal{K}_{m0}})\mathcal{C}_0^1
\end{bmatrix}. \quad (22)
$$

An algebraic linear system consisting of $(m+1)$ equations is derived, containing $(m+1)$ unknown control points $\mathcal{C}_i^2, i=0,1,\ldots,m$. To solve for these control points, the linear system $(m+1)$ equations, as shown in equation (11), is efficiently solved using a Jacobian matrix method. Once the control points $\mathcal{C}_i^2$, $i=0,1,\ldots,m$, are determined, they are substituted into equation (7). Using initial conditions (2), the control points $\mathcal{C}_i^1=\frac{\mathcal{N}h}{2}\frac{d\mathcal{B}_i^2(a)}{dx}+\mathcal{C}_i^0$ for the quadratic $\mathcal{B}$-spline curve can be determined, and their derivative parts can be determined by $\mathcal{C}_i^1$ using $\mathcal{U}_i'(x)\Big|_{x=a}\approx\frac{d\mathcal{B}_i^2(a)}{dx}$, $\forall\, i=0,1,\ldots,m$, and in practice, these integrals must be approximated using the Clenshaw-Curtis quadrature rule. The following algorithms are considered to solve (LSVIDE's-CF) using quadratic $\mathcal{B}$-spline functions:

Algorithm describing the approximate solution (LSVIDE's-CF) using quadratic $\mathcal{B}$-spline.

INPUT:

(I) a, b, and $\mathcal{N}$ is the number of iterations, $(m+1)$ is the number of equations.

(II) $\mathcal{P}_i(x)$, $a_{i0}(x)$, $a_{i1}(x)$, $a_{i2}(x)$, $\mathcal{F}_i(x)$, ω_{ij}, $\mathcal{K}_{ij}(x,s)$, $\mathcal{C}_i^0$, σ_{i0}, and β_{ij} for each $i,j=0,1,\ldots,m$.

OUTPUT: Solution vector $\mathbb{B}$ containing the control points $\mathcal{C}_i^2$, $i=0,1,\ldots,m$.

Steps:

(i) Construct arrays $\mathbb{B}$, $\mathbb{C}$ of size $(m+1)$ and matrix $\mathbb{A}$ from equation (20) of size $(m+1)\times(m+1)$.
(ii) Compute the step size: $h = \frac{b-a}{\mathcal{N}}$, $\mathcal{N} \in \mathbb{N}$ and partition points: $x_{r+1} = a + (r+1)h$, $r = 0, 1, \ldots, \mathcal{N}-1$.
(iii) Compute the approximation: $\mathcal{U}_i'(x)\Big|_{x=a} \approx \frac{d\mathcal{B}_i^2(a)}{dx}$, using the initial conditions.
(iv) Compute the elements of $\mathbb{C}$ from equation (22): $\mathcal{C}_i^1 = \frac{\mathcal{N}h}{2}\frac{d\mathcal{B}_i^2(a)}{dx} + \mathcal{C}_i^0,\ i = 0, 1, \ldots, m$.
(v) Compute the elements of matrix $\mathbb{A}$ and vector $\mathbb{C}$ using the Jacobi iteration method.
(vi) Apply the initial conditions $\mathcal{C}_i^0$ to modify $\mathbb{A}$ and $\mathbb{C}$.
(vii) Solve the system: $\mathbb{A}\cdot\mathbb{B} = \mathbb{C}$ from equation (19), using numerical integration techniques such as the Clenshaw-Curtis quadrature rule.

OUTPUT: Solution vector $\mathbb{B}$ from equation (21), containing the control points, $i = 0, 1, \ldots, m$.

$$\mathbb{B} = \begin{bmatrix}\mathcal{C}_0^2 & \mathcal{C}_1^2 & \mathcal{C}_2^2 & \ldots & \mathcal{C}_m^2\end{bmatrix}^T.$$

I. Algorithm (NCP2DBS): Normal Control Points Second Degree $\mathcal{B}$-Spline.
We perform all steps in the previous main algorithm and follow the additional steps below:
(viii) For $r = 0, 1, \ldots, \mathcal{N}-1$, set: $x = x_{r+1} = a + (r+1)h, \quad n = 2, \quad k = 2$.
(ix) Compute: $\mathbb{P}_i^{\mathcal{Q},2}(x_{r+1}) \approx \sum_{l=0}^{n} \mathcal{C}_i^l \mathcal{B}_{il}^k(x_{r+1}), \quad i = 0, 1, \ldots, m$.
Output: $\mathbb{P}_0^{\mathcal{Q},2}(x_{r+1}), \mathbb{P}_1^{\mathcal{Q},2}(x_{r+1}), \ldots, \mathbb{P}_m^{\mathcal{Q},2}(x_{r+1})$ are the approximate solutions for each function.

II. Algorithm (FCP2BS): First Control Point Second Degree $\mathcal{B}$-Spline.
We perform all steps in the previous main algorithm and follow the following two steps:
(viii) Use: $\mathcal{C}_i^2 = \mathcal{C}_i^2(x_1), \quad i = 0, 1, \ldots, m$.
(ix) Compute: $\mathbb{P}_i^{\mathcal{Q},2}(x_{r+1}) \approx \sum_{l=0}^{n} \mathcal{C}_i^l \mathcal{B}_{il}^k(x_{r+1}), \quad i = 0, 1, \ldots, m$.
Output: $\mathbb{P}_0^{\mathcal{Q},2}(x_{r+1}), \mathbb{P}_1^{\mathcal{Q},2}(x_{r+1}), \ldots, \mathbb{P}_m^{\mathcal{Q},2}(x_{r+1})$ are the approximate solutions for each function.

III. Algorithm (MCP2BS): Mean Control Point Second Degree $\mathcal{B}$-Spline.
We perform all steps in the previous main algorithm and follow the following two steps:
(viii) Use: $\mathcal{C}_i^2 = \frac{1}{\mathcal{N}}\sum_{r=0}^{\mathcal{N}-1} \mathcal{C}_i^2(x_{r+1}), \quad i = 0, 1, \ldots, m$.
(ix) Compute: $\mathbb{P}_i^{\mathcal{Q},2}(x_{r+1}) \approx \sum_{l=0}^{n} \mathcal{C}_i^l \mathcal{B}_{il}^k(x_{r+1}), \quad i = 0, 1, \ldots, m$.
Output: $\mathbb{P}_0^{\mathcal{Q},2}(x_{r+1}), \mathbb{P}_1^{\mathcal{Q},2}(x_{r+1}), \ldots, \mathbb{P}_m^{\mathcal{Q},2}(x_{r+1})$ are the approximate solutions for each function.

IV. Algorithm (FFCP2BS): Average First and Final Control Point Second Degree $\mathcal{B}$-Spline.
We perform all steps in the previous main algorithm and follow the following two steps:
(viii) Use: $\mathcal{C}_i^2 = \frac{1}{2}\left(\mathcal{C}_i^2(x_1) + \mathcal{C}_i^2(x_{\mathcal{N}-1})\right), \quad i = 0, 1, \ldots, m$.
(ix) Compute: $\mathbb{P}_i^{\mathcal{Q},2}(x_{r+1}) \approx \sum_{l=0}^{n} \mathcal{C}_i^l \mathcal{B}_{il}^k(x_{r+1}), \quad i = 0, 1, \ldots, m$.
Output: $\mathbb{P}_0^{\mathcal{Q},2}(x_{r+1}), \mathbb{P}_1^{\mathcal{Q},2}(x_{r+1}), \ldots, \mathbb{P}_m^{\mathcal{Q},2}(x_{r+1})$ are the approximate solutions for each function.

3 Numerical results

In this section, the validity and efficiency of the proposed systems are verified by using the Least squares error. Numerical results are developed in Python 3.9, and those derived by the proposed techniques are compared.

Example. Consider the following classical and fractional-order systems of Volterra integro-differential equations (CF-VIDE's) with variable coefficients on $[0, 1]$:

$$\begin{aligned}&\cos(x)\mathcal{U}_0''(x) + x\,{}_a^C\mathcal{D}_x^{0.3}\mathcal{U}_0(x) + e^x\,{}_a^C\mathcal{D}_x^{0.5}\mathcal{U}_0(x) + x^2\mathcal{U}_0(x)\\ &\qquad = \mathcal{F}_0(x) + \omega_{00}\int_0^x sx^3\mathcal{U}_0(s)ds + \omega_{01}\int_0^x (1+sx^2)\,{}_a^c\mathcal{D}_s^{0.7}\mathcal{U}_1(s)ds + \omega_{02}\int_0^x e^x\,{}_a^c\mathcal{D}_s^{0.8}\mathcal{U}_2(s)ds,\end{aligned}$$

$$e^x\mathcal{U}_1''(x)+\sin(x){}_a^C\mathcal{D}_x^{0.6}\mathcal{U}_1(x)+x^3{}_a^C\mathcal{D}_x^{0.8}\mathcal{U}_1(x)+\ln(x+1)\mathcal{U}_1(x)$$
$$=\mathcal{F}_1(x)+\omega_{10}\int_0^x(x^3-s+1)\mathcal{U}_0(s)ds+\omega_{11}\int_0^x(x+s^2){}_a^c\mathcal{D}_s^{0.45}\mathcal{U}_1(s)ds+\omega_{12}\int_0^x(x^2s){}_a^c\mathcal{D}_S^{0.5}\mathcal{U}_2(s)ds,$$

$$\sin(x)\mathcal{U}_2''(x)+x^3{}_a^C\mathcal{D}_x^{0.4}\mathcal{U}_2(x)+\cos(x){}_a^C\mathcal{D}_x^{0.7}\mathcal{U}_2(x)+\tan(x)\mathcal{U}_2(x)$$
$$=\mathcal{F}_2(x)+\omega_{20}\int_0^x xs\mathcal{U}_0(s)ds+\omega_{21}\int_0^x(\sin(x)-1){}_a^c\mathcal{D}_s^{0.3}\mathcal{U}_1(s)ds+\omega_{22}\int_0^x(2-sx^2){}_a^c\mathcal{D}_s^{0.6}\mathcal{U}_2(s)ds.$$

The given functions $\mathcal{F}_0(x)$, $\mathcal{F}_1(x)$, and $\mathcal{F}_2(x)$ are defined as follows:

$$\mathcal{F}_0(x)=\frac{3x^{1.7}}{\Gamma(1.7)}+\frac{3x^{0.5}e^x}{\Gamma(1.5)}+x^2(3x+2)-\omega_{00}(x^6+x^5)$$
$$-\frac{2\omega_{01}}{\Gamma(2.3)}\left(\frac{x^{2.3}}{2.3}+\frac{x^{5.3}}{3.3}\right)-\frac{\omega_{02}}{\Gamma(2.2)}\left(\frac{e^xx^{2.2}}{2.2}\right),$$

$$\mathcal{F}_1(x)=2e^x+\frac{2\sin(x)x^{1.4}}{\Gamma(2.4)}+\frac{2x^{4.2}}{\Gamma(2.2)}+\ln(x+1)(x^2+1)-\omega_{10}\left(\frac{3x^5}{2}-x^3+\frac{1}{2}x^2+2x^4+2x\right)$$
$$-\frac{2\omega_{11}}{\Gamma(2.55)}\left(\frac{x^{3.55}}{2.55}+\frac{x^{4.55}}{4.55}\right)-\frac{\omega_{12}}{\Gamma(2.5)}\left(\frac{x^{5.5}}{3.5}\right),$$

$$\mathcal{F}_2(x)=\sin(x)+\frac{x^{4.6}}{\Gamma(2.6)}+\frac{x^{1.3}\cos(x)}{\Gamma(2.3)}+\tan(x)\left(\frac{1}{2}x^2-1\right)-\omega_{20}(x^4+x^3)-\frac{2\omega_{21}}{\Gamma(2.7)}\left(\frac{x^{2.7}\sin(x)}{2.7}-\frac{x^{2.7}}{2.7}\right)$$
$$-\frac{\omega_{22}}{\Gamma(2.4)}\left(\frac{2x^{2.4}}{2.4}-\frac{x^{5.4}}{3.4}\right).$$

Together with the initial conditions: $\mathcal{U}_0(0)=2,\ \mathcal{U}_1(0)=1,\ \mathcal{U}_2(0)=-1$, while the exact solutions by: $\mathcal{U}_0(x)=3x+2,\ \mathcal{U}_1(x)=x^2+1,\ \mathcal{U}_2(x)=\frac{1}{2}x^2-1$.

The coefficients are defined as:

$$\omega_{00}=\frac{\sin(0.3)}{\Gamma(13)},\quad \omega_{01}=\frac{\sinh(0.7)}{\Gamma^4(16)},\quad \omega_{02}=\frac{\cosh(30)}{\Gamma^5(14)},$$
$$\omega_{10}=\frac{\cos(89)}{\Gamma(12)},\quad \omega_{11}=\frac{\ln(5)}{\Gamma(12)},\quad \omega_{12}=\frac{\sinh(0.3)}{\Gamma(11)},$$
$$\omega_{20}=\frac{\cos(89)}{\Gamma^2(9)},\quad \omega_{21}=\frac{\sin(179)}{\Gamma(11)},\quad \omega_{22}=\frac{\sin(30)}{\Gamma(12)}.$$

We set the parameters as: $\mathcal{N}=10,\ h=0.1,\ x_r=a+rh,$ for $r=0,1,\ldots,\mathcal{N}-1$. We aim to approximate the solutions $\mathbb{P}_i^{\mathcal{Q},2}(x)$ for $i=0,1,2$, as defined in equation (7). The programs NCP2BS, MCP2BS, FFCP2BS, and FCP2BS are executed to compute the unknown control points $\mathcal{C}_i^0$, $\mathcal{C}_i^1$, and $\mathcal{C}_i^2$ for $i=0,1,2$, we then use these control points to construct the approximate solutions for the given system. The first table presents the values of all control points for $\mathbb{P}_0^{\mathcal{Q},2}(x)$, $\mathbb{P}_1^{\mathcal{Q},2}(x)$, and $\mathbb{P}_2^{\mathcal{Q},2}(x)$ according to the four methods, respectively.

Table 1

The values of control points of $\mathbb{P}_0^{\mathcal{Q},2}(x)$, $\mathbb{P}_1^{\mathcal{Q},2}(x)$, and $\mathbb{P}_2^{\mathcal{Q},2}(x)$ for four methods NCP2BS, MCP2BS, FFCP2BS, and FCP2BS

Methods	Interval	Control points for each function		
		$\mathbb{P}_0^{\mathcal{Q},2}(x)$	$\mathbb{P}_1^{\mathcal{Q},2}(x)$	$\mathbb{P}_2^{\mathcal{Q},2}(x)$
NCP2BS	$]0, 0.1]$	$\mathcal{C}_0^0 = 2.000000000000$ $\mathcal{C}_0^1 = 3.500000000000$ $\mathcal{C}_0^2 = 4.991448639638$	$\mathcal{C}_1^0 = 1.000000000000$ $\mathcal{C}_1^1 = 1.000000000000$ $\mathcal{C}_1^2 = 1.999999999997$	$\mathcal{C}_2^0 = -1.000000000000$ $\mathcal{C}_2^1 = -1.000000000000$ $\mathcal{C}_2^2 = -0.499999999434$
	$]0.1, 0.2]$	$\mathcal{C}_0^0 = 2.000000000000$ $\mathcal{C}_0^1 = 3.500000000000$ $\mathcal{C}_0^2 = 4.978635126166$	$\mathcal{C}_1^0 = 1.000000000000$ $\mathcal{C}_1^1 = 1.000000000000$ $\mathcal{C}_1^2 = 1.999999999965$	$\mathcal{C}_2^0 = -1.000000000000$ $\mathcal{C}_2^1 = -1.000000000000$ $\mathcal{C}_2^2 = -0.499999998681$
	$]0.2, 0.3]$	$\mathcal{C}_0^0 = 2.000000000000$ $\mathcal{C}_0^1 = 3.500000000000$ $\mathcal{C}_0^2 = 4.966525349698$	$\mathcal{C}_1^0 = 1.000000000000$ $\mathcal{C}_1^1 = 1.000000000000$ $\mathcal{C}_1^2 = 1.999999999855$	$\mathcal{C}_2^0 = -1.000000000000$ $\mathcal{C}_2^1 = -1.000000000000$ $\mathcal{C}_2^2 = -0.499999997856$
	$]0.3, 0.4]$	$\mathcal{C}_0^0 = 2.000000000000$ $\mathcal{C}_0^1 = 3.500000000000$ $\mathcal{C}_0^2 = 4.957359583196$	$\mathcal{C}_1^0 = 1.000000000000$ $\mathcal{C}_1^1 = 1.000000000000$ $\mathcal{C}_1^2 = 1.999999999609$	$\mathcal{C}_2^0 = -1.000000000000$ $\mathcal{C}_2^1 = -1.000000000000$ $\mathcal{C}_2^2 = -0.499999997028$
	$]0.4, 0.5]$	$\mathcal{C}_0^0 = 2.000000000000$ $\mathcal{C}_0^1 = 3.500000000000$ $\mathcal{C}_0^2 = 4.952023072809$	$\mathcal{C}_1^0 = 1.000000000000$ $\mathcal{C}_1^1 = 1.000000000000$ $\mathcal{C}_1^2 = 1.999999999169$	$\mathcal{C}_2^0 = -1.000000000000$ $\mathcal{C}_2^1 = -1.000000000000$ $\mathcal{C}_2^2 = -0.499999996274$
	$]0.5, 0.6]$	$\mathcal{C}_0^0 = 2.000000000000$ $\mathcal{C}_0^1 = 3.500000000000$ $\mathcal{C}_0^2 = 4.950374272935$	$\mathcal{C}_1^0 = 1.000000000000$ $\mathcal{C}_1^1 = 1.000000000000$ $\mathcal{C}_1^2 = 1.999999998489$	$\mathcal{C}_2^0 = -1.000000000000$ $\mathcal{C}_2^1 = -1.000000000000$ $\mathcal{C}_2^2 = -0.499999995668$
	$]0.6, 0.7]$	$\mathcal{C}_0^0 = 2.000000000000$ $\mathcal{C}_0^1 = 3.500000000000$ $\mathcal{C}_0^2 = 4.951655066708$	$\mathcal{C}_1^0 = 1.000000000000$ $\mathcal{C}_1^1 = 1.000000000000$ $\mathcal{C}_1^2 = 1.999999997543$	$\mathcal{C}_2^0 = -1.000000000000$ $\mathcal{C}_2^1 = -1.000000000000$ $\mathcal{C}_2^2 = -0.499999995271$
	$]0.7, 0.8]$	$\mathcal{C}_0^0 = 2.000000000000$ $\mathcal{C}_0^1 = 3.500000000000$ $\mathcal{C}_0^2 = 4.954905674858$	$\mathcal{C}_1^0 = 1.000000000000$ $\mathcal{C}_1^1 = 1.000000000000$ $\mathcal{C}_1^2 = 1.999999996333$	$\mathcal{C}_2^0 = -1.000000000000$ $\mathcal{C}_2^1 = -1.000000000000$ $\mathcal{C}_2^2 = -0.499999995114$
	$]0.8, 0.9]$	$\mathcal{C}_0^0 = 2.000000000000$ $\mathcal{C}_0^1 = 3.500000000000$ $\mathcal{C}_0^2 = 4.959250140980$	$\mathcal{C}_1^0 = 1.000000000000$ $\mathcal{C}_1^1 = 1.000000000000$ $\mathcal{C}_1^2 = 1.999999994883$	$\mathcal{C}_2^0 = -1.000000000000$ $\mathcal{C}_2^1 = -1.000000000000$ $\mathcal{C}_2^2 = -0.499999995189$
	$]0.9, 1]$	$\mathcal{C}_0^0 = 2.000000000000$ $\mathcal{C}_0^1 = 3.500000000000$ $\mathcal{C}_0^2 = 4.964022227702$	$\mathcal{C}_1^0 = 1.000000000000$ $\mathcal{C}_1^1 = 1.000000000000$ $\mathcal{C}_1^2 = 1.999999993235$	$\mathcal{C}_2^0 = -1.000000000000$ $\mathcal{C}_2^1 = -1.000000000000$ $\mathcal{C}_2^2 = -0.499999999434$
MCP2BS		$\mathcal{C}_0^0 = 2.000000000000$	$\mathcal{C}_1^0 = 1.000000000000$	$\mathcal{C}_2^0 = -1.000000000000$

Continued on next page

Continued from previous page

Methods	Interval	$\mathbb{P}_0^{\mathcal{Q},2}(x)$	$\mathbb{P}_1^{\mathcal{Q},2}(x)$	$\mathbb{P}_2^{\mathcal{Q},2}(x)$
	$]0,1]$	$\mathcal{C}_0^1 = 3.500000000000$	$\mathcal{C}_1^1 = 1.000000000000$	$\mathcal{C}_2^1 = -1.000000000000$
		$\mathcal{C}_0^2 = 4.962619915469$	$\mathcal{C}_1^2 = 1.999999997908$	$\mathcal{C}_2^2 = -0.499999996534$
FFCP2BS		$\mathcal{C}_0^0 = 2.000000000000$	$\mathcal{C}_1^0 = 1.000000000000$	$\mathcal{C}_2^0 = -1.000000000000$
	$]0,1]$	$\mathcal{C}_0^1 = 3.500000000000$	$\mathcal{C}_1^1 = 1.000000000000$	$\mathcal{C}_2^1 = -1.000000000000$
		$\mathcal{C}_0^2 = 4.977735433670$	$\mathcal{C}_1^2 = 1.999999996616$	$\mathcal{C}_2^2 = -0.499999997407$
FCP2BS		$\mathcal{C}_0^0 = 2.000000000000$	$\mathcal{C}_1^0 = 1.000000000000$	$\mathcal{C}_2^0 = -1.000000000000$
	$]0,1]$	$\mathcal{C}_0^1 = 3.500000000000$	$\mathcal{C}_1^1 = 1.000000000000$	$\mathcal{C}_2^1 = -1.000000000000$
		$\mathcal{C}_0^2 = 4.991448639638$	$\mathcal{C}_1^2 = 1.999999999997$	$\mathcal{C}_2^2 = -0.499999999419$

From the equation (7), we obtain the approximate solution for the classical and fractional-order linear systems of Volterra integro-differential equations (SVIDE's-CF) with variable coefficients, as shown below:

$$\frac{\mathbb{P}_0^{\mathcal{Q},2}(x)}{NCP2BS} = \begin{cases} 2.0000000000(1-x)^2 + 7.0000000000x(1-x) + 4.991448639638x^2, & 0 < x \leq \frac{1}{10}, \\ 2.0000000000(1-x)^2 + 7.0000000000x(1-x) + 4.978635126166x^2, & \frac{1}{10} < x \leq \frac{1}{5}, \\ 2.0000000000(1-x)^2 + 7.0000000000x(1-x) + 4.966525349698x^2, & \frac{1}{5} < x \leq \frac{3}{10}, \\ 2.0000000000(1-x)^2 + 7.0000000000x(1-x) + 4.957359583196x^2, & \frac{3}{10} < x \leq \frac{2}{5}, \\ 2.0000000000(1-x)^2 + 7.0000000000x(1-x) + 4.952023072809x^2, & \frac{2}{5} < x \leq \frac{1}{2}, \\ 2.0000000000(1-x)^2 + 7.0000000000x(1-x) + 4.950374272935x^2, & \frac{1}{2} < x \leq \frac{3}{5}, \\ 2.0000000000(1-x)^2 + 7.0000000000x(1-x) + 4.951655066708x^2, & \frac{3}{5} < x \leq \frac{7}{10}, \\ 2.0000000000(1-x)^2 + 7.0000000000x(1-x) + 4.954905674858x^2, & \frac{7}{10} < x \leq \frac{4}{5}, \\ 2.0000000000(1-x)^2 + 7.0000000000x(1-x) + 4.959250140980x^2, & \frac{4}{5} < x \leq \frac{9}{10}, \\ 2.0000000000(1-x)^2 + 7.0000000000x(1-x) + 4.964022227702x^2, & \frac{9}{10} < x \leq 1.0, \end{cases}$$

$$\frac{\mathbb{P}_1^{\mathcal{Q},2}(x)}{NCP2BS} = \begin{cases} 1.000000000000(1-x)^2 + 2.000000000000x(1-x) + 1.999999999997x^2, & 0 < x \leq \frac{1}{10}, \\ 1.000000000000(1-x)^2 + 2.000000000000x(1-x) + 1.999999999965x^2, & \frac{1}{10} < x \leq \frac{1}{5} \\ 1.000000000000(1-x)^2 + 2.000000000000x(1-x) + 1.999999999855x^2, & \frac{1}{5} < x \leq \frac{3}{10}, \\ 1.000000000000(1-x)^2 + 2.000000000000x(1-x) + 1.99999999960x^2, & \frac{3}{10} < x \leq \frac{2}{5}, \\ 1.000000000000(1-x)^2 + 2.000000000000x(1-x) + 1.999999999169x^2, & \frac{2}{5} < x \leq \frac{1}{2}, \\ 1.000000000000(1-x)^2 + 2.000000000000x(1-x) + 1.999999998489x^2, & \frac{1}{2} < x \leq \frac{3}{5}, \\ 1.000000000000(1-x)^2 + 2.000000000000x(1-x) + 1.999999997543x^2, & \frac{3}{5} < x \leq \frac{7}{10}, \\ 1.000000000000(1-x)^2 + 2.000000000000x(1-x) + 1.999999996333x^2, & \frac{7}{10} < x \leq \frac{4}{5}, \\ 1.000000000000(1-x)^2 + 2.000000000000x(1-x) + 1.999999994883x^2, & \frac{4}{5} < x \leq \frac{9}{10}, \\ 1.000000000000(1-x)^2 + 2.000000000000x(1-x) + 1.999999993235x^2, & \frac{9}{10} < x \leq 1.0, \end{cases}$$

$$\frac{\mathbb{P}_2^{\mathcal{Q},2}(x)}{NCP2BS} = \begin{cases} -1.000000000000(1-x)^2 - 2.000000000000x(1-x) - 0.499999999434x^2, & 0 < x \leq \frac{1}{10}, \\ -1.000000000000(1-x)^2 - 2.000000000000x(1-x) - 0.499999998681x^2, & \frac{1}{10} < x \leq \frac{1}{5}, \\ -1.000000000000(1-x)^2 - 2.000000000000x(1-x) - 0.499999997856x^2, & \frac{1}{5} < x \leq \frac{3}{10}, \\ -1.000000000000(1-x)^2 - 2.000000000000x(1-x) - 0.499999997028x^2, & \frac{3}{10} < x \leq \frac{2}{5}, \\ -1.000000000000(1-x)^2 - 2.000000000000x(1-x) - 0.499999996274x^2, & \frac{2}{5} < x \leq \frac{1}{2}, \\ -1.000000000000(1-x)^2 - 2.000000000000x(1-x) - 0.499999995668x^2, & \frac{1}{2} < x \leq \frac{3}{5}, \\ -1.000000000000(1-x)^2 - 2.000000000000x(1-x) - 0.499999995271x^2, & \frac{3}{5} < x \leq \frac{7}{10}, \\ -1.000000000000(1-x)^2 - 2.000000000000x(1-x) - 0.499999995114x^2, & \frac{7}{10} < x \leq \frac{4}{5}, \\ -1.000000000000(1-x)^2 - 2.000000000000x(1-x) - 0.499999995189x^2, & \frac{4}{5} < x \leq \frac{9}{10}, \\ -1.000000000000(1-x)^2 - 2.000000000000x(1-x) - 0.499999995461x^2, & \frac{9}{10} < x \leq 1.0, \end{cases}$$

$$\frac{\mathbb{P}_0^{\mathcal{Q},2}(x)}{MCP2BS} = \big\{2.000000000000(1-x)^2 + 7.000000000000x(1-x) + 4.962619915469x^2, \quad 0 < x \leq 1,$$

$$\frac{\mathbb{P}_1^{\mathcal{Q},2}(x)}{MCP2BS} = \big\{1.000000000000(1-x)^2 + 2.000000000000x(1-x) + 1.999999997908x^2, \quad 0 < x \leq 1,$$

$$\frac{\mathbb{P}_2^{\mathcal{Q},2}(x)}{MCP2BS} = \big\{-1.000000000000(1-x)^2 - 2.000000000000x(1-x) - 0.499999996534x^2, \quad 0 < x \leq 1,$$

$$\frac{\mathbb{P}_0^{\mathcal{Q},2}(x)}{FFCP2BS} = \big\{2.000000000000(1-x)^2 + 7.000000000000x(1-x) + 4.977735433670x^2, \quad 0 < x \leq 1,$$

$$\frac{\mathbb{P}_1^{\mathcal{Q},2}(x)}{FFCP2BS} = \big\{1.000000000000(1-x)^2 + 2.000000000000x(1-x) + 1.999999996616x^2, \quad 0 < x \leq 1,$$

$$\frac{\mathbb{P}_2^{\mathcal{Q},2}(x)}{FFCP2BS} = \big\{-1.000000000000(1-x)^2 - 2.000000000000x(1-x) - 0.499999997407x^2, \quad 0 < x \leq 1,$$

$$\frac{\mathbb{P}_0^{\mathcal{Q},2}(x)}{FCP2BS} = \big\{2.000000000000(1-x)^2 + 7.000000000000x(1-x) + 4.991448639638x^2, \quad 0 < x \leq 1,$$

$$\frac{\mathbb{P}_1^{\mathcal{Q},2}(x)}{FCP2BS} = \big\{1.000000000000(1-x)^2 + 2.000000000000x(1-x) + 1.999999999997x^2, \quad 0 < x \leq 1,$$

$$\frac{\mathbb{P}_2^{\mathcal{Q},2}(x)}{FCP2BS} = \big\{-1.000000000000(1-x)^2 - 2.000000000000x(1-x) - 0.499999999419x^2, \quad 0 < x \leq 1.$$

Tables 2–4 demonstrate a comparison of the approximate solution with the exact solution of $\mathcal{U}_0(x)$, $\mathcal{U}_1(x)$, and $\mathcal{U}_2(x)$. By setting $\mathcal{N} = 10$, $h = 0.1$, and $x_r = a + rh$ for $r = 0, 1, \ldots, \mathcal{N} - 1$, we compare four methods of quadratic $\mathcal{B}$-spline curves: NCP2BS, MCP2BS, FFCP2BS, and FCP2BS, respectively. Finally Table (5) compares the least square errors for two algorithms (FCP2BS) and (MCP2BS) with various choices of step size.

Table 2

Compares the exact and approximate based on a least square error of $\mathcal{U}_0(x)$

		Approximate Solution $\mathbb{P}_0^{\mathcal{Q},2}(x)$ (10, 0.1)			
x_r	Exact $\mathcal{U}_0(x_r)$	NCP2BS	MCP2BS	FFCP2BS	FCP2BS
0	2.0	2.000000000000	2.000000000000	2.000000000000	2.000000000000
0.1	2.3	2.299914486396	2.299626199155	2.299777354337	2.299914486396
0.2	2.6	2.599145405046	2.598504796619	2.599109417347	2.599657945586
0.3	2.9	2.896987281472	2.896635792392	2.897996189030	2.899230377567
0.4	3.2	3.193177533311	3.194019186475	3.196437669387	3.198631782342
0.5	3.5	3.488005768202	3.490654978867	3.494433858418	3.497862159909
0.6	3.8	3.782134738256	3.786543169569	3.791984756121	3.796921510270
0.7	4.1	4.076310982687	4.081683758580	4.089090362498	4.095809833423
0.8	4.4	4.371139631909	4.376076745900	4.385750677549	4.394527129368
0.9	4.7	4.666992614193	4.669722131530	4.681965701273	4.693073398107
1	5.0	4.964022227701	4.962619915469	4.977735433670	4.991448639638
L.S.E		$4.29736734820\times10^{-3}$	$3.53970591382\times10^{-3}$	$1.25578445284\times10^{-3}$	$1.85249498044\times10^{-4}$
R.T./sec.		2.2315187454223	2.4110293388366	2.6110293388366	3.0368537902832

Table 3

Compares the exact and approximate based on a least square error of $\mathcal{U}_1(x)$

		Approximate Solution $\mathbb{P}_1^{\mathcal{Q},2}(x)$ (10, 0.1)			
x_r	Exact $\mathcal{U}_1(x_r)$	NCP2BS	MCP2BS	FFCP2BS	FCP2BS
0	1.00	1.000000000000	1.000000000000	1.000000000000	1.000000000000
0.1	1.01	1.009999999999	1.009999999979	1.009999999966	1.010000000000
0.2	1.04	1.039999999998	1.039999999916	1.039999999865	1.040000000000
0.3	1.09	1.089999999986	1.089999999812	1.089999999695	1.090000000000
0.4	1.16	1.159999999937	1.159999999665	1.159999999459	1.160000000000
0.5	1.25	1.249999999792	1.249999999477	1.249999999154	1.249999999999
0.6	1.36	1.359999999455	1.359999999247	1.359999998782	1.359999999999
0.7	1.49	1.489999998796	1.489999998975	1.489999998342	1.489999999999
0.8	1.64	1.639999997653	1.639999998661	1.639999997834	1.639999999998
0.9	1.81	1.809999995856	1.809999998305	1.809999997259	1.809999999998
1	2.00	1.999999993237	1.999999997908	1.999999996616	1.999999999997
L.S.E		$7.0199557924\times10^{-17}$	$1.1086895011\times10^{-17}$	$2.9009973891\times10^{-17}$	$2.2801089911\times10^{-23}$
R.T./sec.		2.2315187454223	2.4110293388366	2.6110293388366	3.0368537902832

Table 4

Compares the exact and approximate based on a least square error of $\mathcal{U}_2(x)$

		Approximate Solution $\mathbb{P}_2^{\mathcal{Q},2}(x)$ (10, 0.1)			
x_r	Exact $\mathcal{U}_2(x_r)$	NCP2BS	MCP2BS	FFCP2BS	FCP2BS
0	−1.000	−1.00000000000	−1.00000000000	−1.00000000000	−1.00000000000
0.1	−0.995	−0.99499999999	−0.99499999996	−0.99499999997	−0.994999999994
0.2	−0.980	−0.97999999994	−0.97999999986	−0.97999999989	−0.979999999977
0.3	−0.955	−0.95499999980	−0.95499999969	−0.95499999977	−0.954999999949
0.4	−0.920	−0.91999999952	−0.91999999945	−0.91999999959	−0.919999999909
0.5	−0.875	−0.87499999906	−0.87499999914	−0.87499999936	−0.874999999858
0.6	−0.820	−0.81999999843	−0.81999999877	−0.81999999908	−0.819999999796
0.7	−0.755	−0.75499999767	−0.75499999833	−0.75499999874	−0.754999999723
0.8	−0.680	−0.67999999686	−0.67999999782	−0.67999999836	−0.679999999638
0.9	−0.595	−0.59499999609	−0.59499999724	−0.59499999793	−0.594999999542
1	−0.500	−0.49999999545	−0.49999999659	−0.49999999744	−0.499999999434
L.S.E		$5.6479062171\times10^{-17}$	$2.9319410000\times10^{-17}$	$1.6511565195\times10^{-17}$	$8.1155790437\times10^{-19}$
R.T./sec.		2.2315187454223	2.4110293388366	2.6110293388366	3.0368537902832

Table 5

The least square error with different step sizes for $\mathcal{U}_0(x)$, $\mathcal{U}_1(x)$, and $\mathcal{U}_2(x)$

		L.S.E		
		$\mathbb{P}_0^{\mathcal{Q},2}(x)$	$\mathbb{P}_1^{\mathcal{Q},2}(x)$	$\mathbb{P}_2^{\mathcal{Q},2}(x)$
MCP2BS	$\mathcal{N}=20$	$3.3804747524859968\times10^{-3}$	$9.261094453318532\times10^{-18}$	$2.853188809643298\times10^{-17}$
	$\mathcal{N}=50$	$3.2834709955904783\times10^{-3}$	$8.28101310189266\times10^{-18}$	$2.738736379765375\times10^{-17}$
	$\mathcal{N}=100$	$3.2510081333708340\times10^{-3}$	$7.972488621480216\times10^{-18}$	$2.7005547497925035\times10^{-17}$
	$\mathcal{N}=1000$	$3.251008133370834\times10^{-3}$	$7.696289581642592\times10^{-18}$	$2.667571186072652\times10^{-17}$
FCP2BS	$\mathcal{N}=20$	$2.499554593756400\times10^{-5}$	$9.86076131526260\times10^{-32}$	$1.596004257270100\times10^{-19}$
	$\mathcal{N}=50$	$1.564846997979940\times10^{-6}$	$8.86076131526260\times10^{-32}$	$1.745188520809840\times10^{-20}$
	$\mathcal{N}=100$	$1.821769238662400\times10^{-7}$	$7.86076131526260\times10^{-32}$	$3.468085471432110\times10^{-21}$
	$\mathcal{N}=1000$	$1.18952321777046\times10^{-10}$	$6.86076131526260\times10^{-32}$	$1.01330642945647\times10^{-23}$

4 Conclusion

In this paper, we constructed a numerical technique for solving systems of Volterra integro-differential equations that involve both classical and fractional derivatives (SVIDE's-CF) with variable coefficients based on quadratic $\mathcal{B}$-spline functions. Four algorithms, NCP2BS, MCP2BS, FFCP2BS, and FCP2BS, were successfully introduced. The control points were determined by converting the system of VIDEs-CF into a system of linear algebraic equations, which was then solved using the Jacobian method and the Clenshaw-Curtis quadrature rule. Numerical experiments demonstrated that all the proposed methods are novel and significant for our research. Furthermore, we show that FCP2BS outperforms the other algorithms in terms of accuracy and computational efficiency, simplifies the analysis and ensures that computations remain manageable using software such as Python. In general, Table 5 demonstrates that as the value of $\mathcal{N}$ increases, the approximation significantly improves. As a future direction, we aim to extend this framework by exploring more sophisticated spline functions, including modified quadratic, cubic, trigonometric, and exponential $\mathcal{B}$-splines.

Author Contributions

Most of this work was done by D.Kh. Abdullah. K.HF. Jwamer helped in auditing the results, providing critical revisions, and confirming the accuracy of the results. Sh.Sh. Ahmed helped in the review and assisted in the improvement of the analysis. All the authors revised the manuscript and approved the final manuscript.

Conflict of Interest

The authors declare no conflict of interest.

References

1 Podlubny, I. (1999). *Fractional Differential Equations* (Vol. 198). San Diego, CA: Academic Press.

2 Podlubny, I. (2002). Geometric and physical interpretation of fractional integration and fractional differentiation. *Fractional Calculus and Applied Analysis, 5*(4), 367–386.

3 Owolabi, M., & Abdon, A. (2019). *Numerical Methods for Fractional Differentiation.* Singapore: Springer. https://doi.org/10.1007/978-981-15-0098-5

4 Mariya, K. (2005). *Properties and Applications of the Caputo Fractional Operator.* Karlsruhe.

5 Bangti, J. (2021). *Fractional Differential Equations*. Cham: Springer. https://doi.org/10.1007/978-3-030-76043-4

6 Yeolekar, M., Dave, D., & Khirsariya, R. (2024). Solution of a cancer treatment model of a drug targeting treatment through nanotechnology using Adomian decomposition Laplace transform method. *Interactions, 245*, Article 278. https://doi.org/10.1007/s10751-024-02114-6

7 Chavada, A., Pathak, N., & Khirsariya, R. (2024). A fractional mathematical model for assessing cancer risk due to smoking habits. *Mathematical Modelling and Control, 4*(3), 246–259. https://doi.org/10.3934/mmc.2024020

8 Said, A., Mouffak, B., Jamal, E., Juan, J., & Yong, Zh. (2023). *Fractional Differential Equations and Inclusions: Classical and Advanced Topics*. London: World Scientific Publishing Co. https://doi.org/10.1142/12993

9 Weilbeer, M. (2005). *Efficient Numerical Methods for Fractional Differential Equations and Their Analytical Background* (Doctoral dissertation). Braunschweig: Technische Universität Braunschweig.

10 Benzahi, A., Arar, N., Abada, N., Rhaima, M., & Mchiri, L. (2023). Numerical investigation of Fredholm fractional integro-differential equations by least squares method and compact combination of shifted Chebyshev polynomials. *Journal of Nonlinear Mathematical Physics, 30*, 1392–1408. https://doi.org/10.1007/s44198-023-00128-2

11 Ghosh, B., & Mohapatra, J. (2023). An iterative difference scheme for solving arbitrary order nonlinear Volterra integro-differential population growth model. *The Journal of Analysis, 32*, 57–72. https://doi.org/10.1007/s41478-023-00593-4

12 Rahimkhani, P., & Ordokhani, Y. (2020). Approximate solution of nonlinear fractional integro-differential equations using fractional alternative Legendre functions. *Journal of Computational and Applied Mathematics, 365*, Article 112365. https://doi.org/10.1016/j.cam.2019.112365

13 Zada, A., Riaz, U., Jamshed, J., Alam, M., & Kallekh, A. (2024). Analysis of impulsive Caputo fractional integro-differential equations with delay. *Mathematical Methods in the Applied Sciences, 48*(2), 2102–2121. https://doi.org/10.1002/mma.10426

14 Amin, M.B.M., & Ahmad, S.S. (2022). Laplace transform for solving system of integro-fractional differential equations of Volterra type with variable coefficients and multi-time delay. *Symmetry, 14*(5), 984. https://doi.org/10.3390/sym14050984

15 Akhlaghi, S., Tavassoli Kajani, M., & Allame, M. (2023). Numerical solution of fractional order integro-differential equations via Müntz orthogonal functions. *Journal of Mathematics, 2023*, Article ID 6647128. https://doi.org/10.1155/2023/6647128

16 Yuzbasi, S. (2024). Fractional Bell collocation method for solving linear fractional integro-differential equations. *Mathematical Sciences, 18*(1), 29–40. https://doi.org/10.1007/s40096-022-00482-0

17 Jwamer, K.H.F., Ahmed, Sh.Sh., & Abdullah, D.Kh. (2021). Approximate solution of Volterra integro-fractional differential equations using quadratic spline function. *Bulletin of the Karaganda University. Mathematics Series,* 1(101), 50–64. https://doi.org/10.31489/2021M1/50-64

18 Karwan, H.F.J., Shazad, Sh.A., & Diar, Kh.A. (2020). Numerical treatment solution of Volterra integro-fractional differential equation by using linear spline function. *Journal of Zankoy Sulaimani — Part A, 22*(2), 327–338. https://doi.org/10.17656/jzs.10832

19 Masti, I., & Sayevand, K. (2024). On collocation–Galerkin method and fractional B-spline functions for a class of stochastic fractional integro-differential equations. *Mathematics and Computers in Simulation, 216*, 263–287. https://doi.org/10.1016/j.matcom.2023.09.013

20 Mirzaee, F., & Alipour, S. (2020). Cubic B-spline approximation for linear stochastic integro-

differential equation of fractional order. *Journal of Computational and Applied Mathematics, 336*, 112440. https://doi.org/10.1016/j.cam.2019.112440

21 Maleknejad, Kh., & Rostami, Y. (2019). B-spline method for solving Fredholm integral equations of the first kind. *International Journal of Industrial Mathematics, 11*(1), 63–70.

22 Rajani, B., & Debasish, D. (2011). *A Mixed Quadrature Rule by Blending Clenshaw–Curtis and Gauss–Legendre Quadrature Rules for Approximation of Real Definite Integrals in Adaptive Environment, I.* Hong Kong: Newswood Limited, International Association of Engineers.

23 Ronald, N., & Tom, L. (1993). *Knot Insertion and Deletion Algorithms for B-Spline Curves and Surfaces.* Washington, DC: Library of Congress Cataloging in Publication.

24 Abdullah, D.Kh., Ahmed, Sh.Sh., & Jwamer, K.H.F. (2025). An approximate solutions technique using quadratic $\mathcal{B}$-spline functions for a system of Volterra integro-fractional differential equations. *New Mathematics and Natural Computation*, 1–32. https://doi.org/10.1142/S1793005727500566

25 Abdullah, D.Kh., Ahmed, Sh.Sh., Jwamer, K.H.F. (2025). Employing Cubic B-Spline Functions to Solve Linear Systems of Volterra Integro-Fractional Differential Equations with Variable Coefficients. *European Journal of Pure and Applied Mathematics, 18*(4), 6763. https://doi.org/10.29020/nybg.ejpam.v18i4.6763

Author Information[*]

Diar Khalid Abdullah — PhD Student, Department of Mathematics, College of Science, University of Sulaimani, Sulaymaniyah, Kurdistan Region, Iraq; e-mail: *Diar.khalid85@gmail.com*; https://orcid.org/0009-0005-4245-0493

Karwan Hama Faraj Jwamer *(corresponding author)* — Doctor of Mathematics (Differential Equations), Professor, Department of Mathematics, College of Science, University of Sulaimani, Sulaymaniyah, Kurdistan Region, Iraq; e-mail: *karwan.jwamer@univsul.edu.iq*; https://orcid.org/0000-0003-4009-0357

Shazad Shawki Ahmed — Doctor of Mathematics (Numerical Analysis), Professor, Department of Mathematics, College of Science, University of Sulaimani, Sulaymaniyah, Kurdistan Region, Iraq; e-mail: *Shazad.ahmed@univsul.edu.iq*; https://orcid.org/0000-0002-9409-4743

[*]Authors' names are presented in the order: First name, Middle name, and Last name.