Journal of Basrah Researches (Sciences) **52**(1), 31 (2026)
DOI: https://doi.org/10.56714/bjrs.52.1.3

An Approach to Volterra and Fredholm Integro-Differential Equations via Six Non-Polynomial Cubic Spline functions

Rahel J. Qadir[1], Shabaz Jalil Mohammedfaeq[2,*], Faraidun K. Hamasalh[3]

[1]Department of Mathematical Science, College of Basic Education, University of Sulaimani, Sulaymaniyah 46001, Iraq.
[2]Department of Mathematics, College of Science, University of Sulaimani, Sulaymaniyah 46001, Iraq.
[3]Department of Mathematics, College of Education, University of Sulaimani, Sulaymaniyah 46001, Iraq.

ARTICLE INFO

Received 08 November 2025
Revised 24 January 2026
Accepted 25 January 2026
Published 30 June 2026

Keywords:

Non-Polynomial Cubic Splines, Collocation Method, Convergence Analysis, Numerical Approximation, Integro-Differential Equations.

Citation: R. J. Qadir et al., J. Basrah Res. (Sci.) **52**(1), 31 (2026).
DOI:https://doi.org/10.56714/bjrs.52.1.3

ABSTRACT

This study introduces an effective numerical method for approximating solutions to Volterra and Fredholm integro-differential equations through a novel general formulation of non-polynomial cubic spline functions (NCSF). The proposed method enhances spline techniques by integrating trigonometric and exponential components with six non-polynomial cubic spline function formulations. A comprehensive convergence analysis occurs, establishing sufficient conditions for the stability and error bounds of the method. Multiple test problems are addressed, including Volterra and Fredholm integro-differential equations. The resulting numerical outcomes are compared with exact solutions and methodologies from the literature, presented in tables and figures. Comparisons illustrate the efficacy and robustness of the proposed six non-polynomial cubic spline method in addressing Volterra and Fredholm problems while obtaining higher precision with fewer grid points, as demonstrated by computational reports generated using Python 3.12.4.

1. Introduction

Integral and integro-differential equations are a cornerstone of mathematical physics, used to express a wide range of topics. While it is impossible to list all their applications, this paper will focus on specific examples to showcase their importance. These equations are so central to nearly every field of applied mathematics and mathematical physics that a vast body of literature has been dedicated to them. They are typically classified as either Fredholm, Volterra, or Fredholm-Volterra Integro-differential equations. Although many physical phenomena are described by these equations, they are often very difficult to solve analytically.

Numerical methods are crucial for analyzing complex problems across various fields because many of these problems lack analytical solutions. As a result, researchers have developed and refined a wide range of numerical techniques. Some notable methods include: Taylor-series expansion [3], the hybrid

*Corresponding author email: shabaz.mohammedfaeq@univsul.edu.iq

ISSN: 1817-2695 (Print); 2411-524X (Online)
Online at: https://jou.jobrs.edu.iq

function method [4], the Modified Laplace Adomian decomposition [5], the Legendre wavelets method [7], the Tau method [8], the finite difference method [9, 10], the Haar function methods [11, 12], the Homotopy perturbation [13], Fractional-Differential Equations [14], the differential transform [6]. Many of these numerical techniques have been successfully applied to solve linear Fredholm and Volterra integrodifferential equations [6, 16, 17]. Additionally, researchers continue to refine these methods for greater accuracy and efficiency. For example, Hilmi et al. [22] developed a method that provides precise numerical solutions for fractional differential equations. Other studies have focused on using non-polynomial cubic spline functions and cubic splines to create numerical methods for solving both first- and second-order Fredholm integro-differential equations [1, 6, 23]. This work aims to use new six non-polynomial cubic spline functions in matrix form to solve multi-term linear integro-differential equations, in the closed interval $[a, b(z)]$, as follows:

$$\sum_{\alpha=0}^{n} \eta_\alpha u^\alpha(z) = y(z) + \int_a^{b(z)} k(z,x)u(x)dx \tag{1}$$

with general initial condition.

$$\sum_{j=0}^{n-1} \zeta_j \frac{\partial^j u(a)}{\partial z^j} = \vartheta_j \tag{2}$$

Where η, y and k are known functions, $\alpha \in N, u \in C^2[J, R]$, and $J = [a, b(z)]$. where $b(z) = z$ and $b(z) = b$, then equation (1) is called linear Volterra Integro-Differential Equations (V-IDEs) and linear Fredholm IntegroDifferential Equations (F-IDEs) of the high orders with constant coefficients, respectively. $u(z)$ needs to be determined, which is an unknown solution function, while $\alpha, \eta_\alpha, \zeta_j, \vartheta_j \in Z^+, y(z) \in C([a, b(z)], R)$, and $K(z,x) \in C(D, R), D = (z,x): a \le z \le x \le b(z)$, are known, with ζ and ϑ are real appropriate constants $\forall j = 0,1,2, \dots, n-1$, and n is of arbitrary integer order ($n \in Z^+$). Equation (2) delineates a general initial condition established to guarantee the well-posedness of the multi-term integro-differential problem. This condition is assumed and is standard in the investigation of higher-order integro-differential equations.

The outline of the paper is as follows. First, in Section 2 and Section 3, the researchers proposed a new type of non-polynomial cubic spline method and describe the basic formulation of spline approximation required for our subsequent development. The convergence analysis of the suggested approach to problem solving is described in Section 4. In Section 5, numerical examples are provided to demonstrate the effectiveness of the suggested approach. Lastly, a conclusion of the numerical results is provided in Section 6.

In this study, a novel six non-polynomial spline approach for multi-term linear integro-differential equations is proposed, its convergence conditions are established, and numerical experiments show that it works more accurately and efficiently than current methods.

2. Six non-polynomial cubic spline function method

Spline interpolation refers to the process of generating a spline function that approximates a given data set or function, given a knot vector over the interval $[a, b(z)]$ splines of degree M constitute a vector space. For example, the set of all exponential cubic splines forms a subspace within the space of all cubic $C^2[a, b(z)]$ splines, a cubic spline function $S(z)$ that interpolates a function $u(z)$ over the interval $[a, b(z)]$ is defined by two main properties, within each subinterval $[z_k, z_{k+1}]$ is a polynomial of degree no greater than three, and $S'(z)$, and $S''(z)$ are continuous in the interval $[a, b(z)]$. The $M-$ th order nonpolynomial spline function defined in [2].

$$P_M \in \text{span}\{1, z, \dots, z^{M-1}, e^{-\tau z}, e^{\tau z}, \cos(\tau z), \sin(\tau z), \cosh(\tau z), \sinh(\tau z)\}$$

In this section, a special case of the above linear span has been considered, and a uniform mesh Δ with nodal points z_k will be taken into consideration, on $[a, b(z)]$ such that

$$\Delta: a = z_0 < z_1 < \cdots < z_M = b(z),$$

where $z_k = z_0 + jh$, for $j = 0,1, \dots, M$, and $h = \frac{b(z)-a}{M}$. The interpolating six non-polynomial cubic spline function (NCSF) is $S_k(z)$ which interpolate u at z_k,

$$S_\Delta(z) = a_k\left(k_1 e^{\gamma\tau(z-z_k)} + k_2 e^{-\gamma\tau(z-z_k)}\right) + b_j(z-z_k)^2 + c_k(z-z_k) + d_k, k = 0,1, \dots, M \quad (3)$$

It has been generated in the following table:

Table 1. Six Non-polynomial cubic Spline Functions.

k_1	k_2	γ	Spline function	Formula's name
1	0	1	$a_k e^{\tau(z-z_k)} + b_k(z-z_k)^2 + c_k(z-z_k) + d_k$	NCSF. 1
0	1	1	$a_k e^{-\tau(z-z_k)} + b_k(z-z_k)^2 + c_k(z-z_k) + d_k$	NCSF. 2
0.5	0.5	i	$a_k \cos\left(\tau(z-z_k)\right) + b_k(z-z_k)^2 + c_k(z-z_k) + d_k$	NCSF. 3
-0.5i	0.5 i	i	$a_k \sin\left(\tau(z-z_k)\right) + b_k(z-z_k)^2 + c_k(z-z_k) + d_k$	NCSF. 4
0.5	-0.5	1	$a_k \sinh\left(\tau(z-z_k)\right) + b_k(z-z_k)^2 + c_k(z-z_k) + d_k$	NCSF. 5
0.5	0.5	1	$a_k \cosh\left(\tau(z-z_k)\right) + b_k(z-z_k)^2 + c_k(z-z_k) + d_k$	NCSF. 6

Where γ and τ are arbitrary constants, the following boundary conditions in equation (4) have been used to derive the approximate solution of equation (3)

$$S_k(z_k) = u_k, S_k(z_{k+1}) = u_{k+1}, S_k''(z_k) = F_k, S_k''(z_{k+1}) = F_{k+1}, \quad (4)$$

It can easily confirm that the spline scheme approximation $S(z)$ is successfully and uniquely determined with the recurrence formula equation (4) for all h by using algebraic manipulation of equation (4) and equation (5) the following have been obtained:

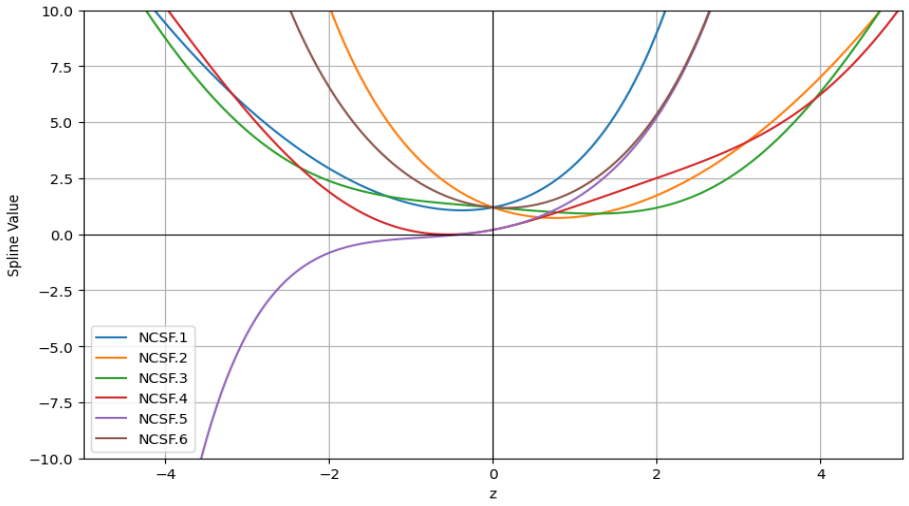

Fig. 1. Six Non-polynomial Cubic Spline Functions generated in Table 1.

$$
\begin{aligned}
a_k &= \frac{\beta_0(F_k - F_{k+1})}{\gamma^2\tau^2[(\beta_0-1)k_2+(\beta_0-\beta_1)k_1]},\\
b_k &= \frac{(k_1+k_2)\beta_0 F_{k+1}-(k_1\beta_1+k_2)F_k}{(2\beta_1-2)k_2+(2\beta_0-2\beta_1)k_1},\\
c_k &= \frac{\gamma^2\tau^2\big((1-\beta_0)k_2+(\beta_1-\beta_0)k_1\big)[u_k-2u_{k+1}]}{2h\gamma^2\tau^2[(\beta_0-1)k_2+(\beta_0-\beta_1)k_1]} + \frac{[(\gamma^2\tau^2h^2\beta_1-2\beta_1+2\beta_0)k_1+(\gamma^2\tau^2h^2-2+2\beta_0)k_2]F_k}{2h\gamma^2\tau^2[(\beta_0-1)k_2+(\beta_0-\beta_1)k_1]}\\
&\quad + \frac{[(-\gamma^2\tau^2h^2\beta_0-2\beta_0+2\beta_1)k_1+(-\gamma^2\tau^2h^2\beta_0+2-2\beta_0)k_2]F_{k+1}}{2h\gamma^2\tau^2[(\beta_0-1)k_2+(\beta_0-\beta_1)k_1]},\\
d_k &= \frac{[\beta_0-1)k_2+(\beta_0-\beta_1)k_1]u_k+(k_1+k_2)(F_{k+1}-F_k)\beta_0}{\gamma^2\tau^2[(\beta_0-1)k_2+(\beta_0-\beta_1)k_1]}.
\end{aligned}
\tag{5}
$$

The coefficients a_k, b_k, c_k, and d_k are established by enforcing the interpolation and smoothness constraints specified in equation (4), leading to a linear algebraic system. The resolution of this system produces the explicit expressions delineated in equation (5). These coefficients are well-defined provided that $\gamma \neq 0, \tau \neq 0$, $(\beta_0 - 1)k_2 - (\beta_0 - \beta_1)k_1 \neq 0$, and $(\beta_1 - 1)k_2 - (\beta_0 - \beta_1)k_1 \neq 0$. with $k_1 > 0$ and $k_2 > 0$.
where $\beta_0 = e^{\gamma\tau h}$ and $\beta_1 = e^{2\gamma\tau h}$, From the continuity condition $S'_k(z_k) = S'_{k-1}(z_k)$ we get:

$$\sigma_0 F_{k-1} + \sigma_1 F_k + \sigma_2 F_{k+1} = \gamma_0[u_{k-1} - 2u_k + u_{k+1}]. \tag{6}$$

where

$$
\begin{aligned}
\sigma_0 &= 2\gamma\tau h\beta_0(k_2 - k_1) - (2\beta_0 + \gamma^2\tau^2h^2)(k_1 + k_2) + 2(\beta_1 k_1 + k_2),\\
\sigma_1 &= 4\beta_0 k_1(\gamma\tau h + 1) + 4\beta_0 k_2(1 - \gamma\tau h) + \gamma^2\tau^2h^2k_1(\beta_0 + \beta_1) + \gamma^2\tau^2(h^2k_2 + h^2\beta_0k_2 + 2\beta_1k_1) - 2\beta_1k_1 - 4k_2,\\
\sigma_2 &= 2\gamma\tau\beta_0 h(k_2 - k_1) - (\gamma^2\tau^2h^2\beta_0 + 2\beta_0)(k_1 + k_2) + 2(k_2 - \beta_0k_1),\\
\gamma_0 &= 2\gamma^2\tau^2(k_2 + \beta_1k_1) - 2\gamma^2\tau^2\beta_0(k_2 + k_1).
\end{aligned}
$$

By expanding equation (3) by Taylor series expansion, the form of the truncation error has been expanded about z_k :

$$
\begin{aligned}
T_k = (\sigma_0 + \sigma_1 + \sigma_2 - \gamma_0h^2)u''_k + h(\sigma_0 - \sigma_2)u_k^{(3)} + h^2\left(\frac{\sigma_0 + \sigma_2}{2} - \frac{\gamma_0h^2}{12}\right)u_k^{(4)}\\
+ \frac{h^3}{6}(\sigma_0 - \sigma_2)u_k^{(5)} + O(h^4).
\end{aligned}
\tag{7}
$$

If $\sigma_0 + \sigma_1 + \sigma_2 = \gamma_0h^2$ then there is no $O(1)$ error, to vanish the $O(h)$ term we can use $\sigma_0 = \sigma_2$, if these two conditions have been fulfilled, we obtain $T_k = h^2\left(\frac{\sigma_0+\sigma_2}{2} - \frac{\gamma_0h^2}{12}\right)u_k^{(4)} + O(h^4)$, and moreover if $\frac{\sigma_0+\sigma_2}{2} = \frac{\gamma_0h^2}{12}$, the approach provides a higher order, so $T_k = O(h^4)$, the natural spline condition $F_0 = 0 = F_M$, and above these conditions have been used to convert equation (3) to a tridiagonal linear relation, in addition the matrix form is:

$$
\underbrace{\begin{bmatrix}
1 & 0 & 0 & \cdots & 0\\
1 & 10 & 1 & \ddots & \vdots\\
0 & 1 & 10 & \ddots & 0\\
\vdots & \ddots & \ddots & \ddots & 1\\
0 & \cdots & 0 & 0 & 0\\
 & & & & 1
\end{bmatrix}}_{Z}
\underbrace{\begin{bmatrix} F_0\\ F_1\\ F_2\\ \vdots\\ F_{M-1}\\ F_M \end{bmatrix}}_{M}
= \frac{12}{h^2}
\begin{bmatrix}
0 & 0 & 0 & \cdots & & 0\\
1 & -2 & 1 & \ddots & & \vdots\\
0 & 1 & -2 & \ddots & & 0\\
\vdots & \ddots & \ddots & \ddots & & 1\\
0 & \cdots & 0 & 0 & 0 & 0
\end{bmatrix}
\underbrace{\begin{bmatrix} u_0\\ u_1\\ u_2\\ \vdots\\ u_{M-1}\\ u_M \end{bmatrix}}_{U}.
$$

Hence,

$$M = \frac{12}{h^2}QU, \text{ where } Q = Z^{-1}R. \tag{8}$$

The following equation has been obtained by substituting equation (5) in equation (2).

$$S_k(z) = \left(\frac{\beta_0(F_k - F_{k+1})}{\gamma^2\tau^2[(\beta_0-1)k_2 + (\beta_0-\beta_1)k_1]}\right)\left(k_1 e^{\gamma\tau(z-z_k)} + k_2 e^{-\gamma\tau(z-z_k)}\right)$$
$$+\left(\frac{(k_1+k_2)\beta_0 F_{k+1} - (k_1\beta_1 + k_2)F_k}{(2\beta_1-2)k_2 + (2\beta_0 - 2\beta_1)k_1}\right)(z-z_k)^2 + \left[\frac{\gamma^2\tau^2\left((1-\beta_0)k_2 + (\beta_1-\beta_0)k_1\right)[u_k - 2u_{k+1}]}{2h\gamma^2\tau^2[(\beta_0-1)k_2 + (\beta_0-\beta_1)k_1]} + \right.$$
$$\frac{[(\gamma^2\tau^2h^2\beta_1 - 2\beta_1 + 2\beta_0)k_1 + (\gamma^2\tau^2h^2 - 2 + 2\beta_0)k_2]F_k}{2h\gamma^2\tau^2[(\beta_0-1)k_2 + (\beta_0-\beta_1)k_1]}$$
$$+\frac{[(-\gamma^2\tau^2h^2\beta_0 - 2\beta_0 + 2\beta_1)k_1 + (-\gamma^2\tau^2h^2\beta_0 + 2 - 2\beta_0)k_2]F_{k+1}}{2h\gamma^2\tau^2[(\beta_0-1)k_2 + (\beta_0-\beta_1)k_1]}(z-z_k) +$$
$$\left.\frac{[\beta_0-1)k_2 + (\beta_0-\beta_1)k_1]u_k + (k_1+k_2)(F_{k+1}-F_k)\beta_0}{\gamma^2\tau^2[(\beta_0-1)k_2 + (\beta_0-\beta_1)k_1]} + O(h^4), k = 0,1,\dots,n.\right.$$

3. Non-polynomial cubic spline function method

Non-polynomial cubic spline functions used in a variety of scientific investigations to analyze integro-differential equations, Volterra, Fredholm, and mixed Volterra-Fredholm integral equations, such as [6,15,19,20,21,22]. These investigations have produced highly favorable results, in this section the proposed methods have been utilized to approximate the proposed volterra, fredholm integro-differential equations.

$S_k(z)$ has been substituted in equation (1) to obtain:

$$\sum_{\alpha=0}^{n}\eta_\alpha u^\alpha(z_k) = y(z_k) + \int_a^{b(z)} k(z_k,x)u(x)dx$$
$$\approx y(z_k) + \sum_{i=0}^{M-1}\int_{z_i}^{z_{i+1}} k(z_k,x)S_i(x)dx + O(h^4)$$
$$= y(z_k) + \sum_{i=0}^{M-1}\int_{z_i}^{z_{i+1}} k(z_k,x)\left[\left(\frac{\beta_0(F_i - F_{i+1})}{\gamma^2\tau^2[(\beta_0-1)k_2 + (\beta_0-\beta_1)k_1]}\right)\left(k_1 e^{\gamma\tau(x-x_i)} + k_2 e^{-\gamma\tau(x-x_i)}\right)\right.$$
$$+\left(\frac{(k_1+k_2)\beta_0 F_{i+1} - (k_1\beta_1 + k_2)F_i}{(2\beta_1-2)k_2 + (2\beta_0 - 2\beta_1)k_1}\right)(x-x_i)^2 + \left[\frac{\gamma^2\tau^2\left((1-\beta_0)k_2 + (\beta_1-\beta_0)k_1\right)[u_i - 2u_{i+1}]}{2h\gamma^2\tau^2[(\beta_0-1)k_2 + (\beta_0-\beta_1)k_1]}\right.$$
$$+\frac{[(\gamma^2\tau^2h^2\beta_1 - 2\beta_1 + 2\beta_0)k_1 + (\gamma^2\tau^2h^2 - 2 + 2\beta_0)k_2]F_i}{2h\gamma^2\tau^2[(\beta_0-1)k_2 + (\beta_0-\beta_1)k_1]}$$
$$\left.+\frac{[(-\gamma^2\tau^2h^2\beta_0 - 2\beta_0 + 2\beta_1)k_1 + (-\gamma^2\tau^2h^2\beta_0 + 2 - 2\beta_0)k_2]F_{i+1}}{2h\gamma^2\tau^2[(\beta_0-1)k_2 + (\beta_0-\beta_1)k_1]}\right](x-x_i)$$
$$\left.+\frac{[\beta_0-1)k_2 + (\beta_0-\beta_1)k_1]u_i + (k_1+k_2)(F_{i+1}-F_i)\beta_0}{\gamma^2\tau^2[(\beta_0-1)k_2 + (\beta_0-\beta_1)k_1]}\right]dx + O(h^4)$$

Assume that

$a_{k,i} = b_{k,i+1} = \int_{z_i}^{z_i+1} k(z_k,x)\left(k_1 e^{\gamma\tau(x-x_k)} + k_2 e^{-\gamma\tau(x-x_k)}\right)dx,$
$c_{k,i} = d_{k,i+1} = \int_{z_i}^{z_{i+1}} k(z_k,x)(x-x_k)^2dx,$
$e_{k,i} = l_{k,i+1} = \int_{z_i}^{z_{i+1}} k(z_k,x)(x-x_k)dx,$
$g_{k,i} = h_{k,i+1} = \int_{z_i}^{z_{i+1}} k(z_k,x)dx.$
And $a_{k,M} = b_{k,0} = c_{k,M} = d_{k,0} = e_{k,M} = l_{k,0} = g_{k,M} = h_{k,0} = 0$, then

$$\sum_{\alpha=0}^{n} \eta_\alpha u^\alpha(z_k) = y(z_k) + \sum_{i=0}^{M} \left[\left(\frac{\beta_0 F_i}{\gamma^2\tau^2[(\beta_0-1)k_2+(\beta_0-\beta_1)k_1]}\right) a_{k,i} - \left(\frac{\beta_0 F_i}{\gamma^2\tau^2[(\beta_0-1)k_2+(\beta_0-\beta_1)k_1]}\right) b_{k,i}\right.$$
$$- \left(\frac{(k_1\beta_1+k_2)F_i}{(2\beta_1-2)k_2+(2\beta_0-2\beta_1)k_1}\right) c_{k,i} + \left(\frac{(k_1+k_2)\beta_0 F_i}{(2\beta_1-2)k_2+(2\beta_0-2\beta_1)k_1}\right) d_{k,i}$$
$$+ \left[\frac{\gamma^2\tau^2[(1-\beta_0)k_2+(\beta_1-\beta_0)k_1]u_i+[(\gamma^2\tau^2h^2\beta_1-2\beta_1+2\beta_0)k_1+(\gamma^2\tau^2h^2-2+2\beta_0)k_2]F_i}{2h\gamma^2\tau^2[(\beta_0-1)k_2+(\beta_0-\beta_1)k_1]} e_{k,i}\right.$$
$$+ \left.\left[\frac{-2\gamma^2\tau^2[(1-\beta_0)k_2+(\beta_1-\beta_0)k_1]u_i+[(-\gamma^2\tau^2h^2\beta_0-2\beta_0+2\beta_1)k_1+(-\gamma^2\tau^2h^2\beta_0+2-2\beta_0)k_2]F_i}{2h\gamma^2\tau^2[(\beta_0-1)k_2+(\beta_0-\beta_1)k_1]}\right] l_{k,i}\right.$$
$$\left. - \frac{[\beta_0-1)k_2+(\beta_0-\beta_1)k_1]u_i+(k_1+k_2)\beta_0 F_i}{\gamma^2\tau^2[(\beta_0-1)k_2+(\beta_0-\beta_1)k_1]} g_{k,i} + \frac{[\beta_0-1)k_2+(\beta_0-\beta_1)k_1]u_i+(k_1+k_2)\beta_0 F_i}{\gamma^2\tau^2[(\beta_0-1)k_2+(\beta_0-\beta_1)k_1]} h_{k,i}\right] + O(h^4).$$

(9)

Let $A = a_{k,i}, B = b_{k,i}, D = d_{k,i}, E = e_{k,i}, L = l_{k,i}, G = g_{k,i}$, and $H = h_{k,i}$.

$$F = \begin{bmatrix} F_0 \\ F_1 \\ \vdots \\ F_M \end{bmatrix}, U = \begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ u_M \end{bmatrix}, Y = \begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_M \end{bmatrix}, \text{ and } \boldsymbol{\eta} U = \begin{bmatrix} \sum_{\alpha=1}^{M} \eta_\alpha u^\alpha(z_0) \\ \sum_{\alpha=1}^{M} \eta_\alpha u^\alpha(z_1) \\ \vdots \\ \sum_{\alpha=1}^{M} \eta_\alpha u^\alpha(z_M) \end{bmatrix}$$
$$= \begin{bmatrix} u^0(z_0) & u^1(z_0) & \cdots & u^M(z_0) \\ u^0(z_1) & u^1(z_1) & \cdots & u^M(z_1) \\ \vdots & \vdots & \ddots & \vdots \\ u^0(z_M) & u^1(z_M) & \cdots & u^M(z_M) \end{bmatrix} \begin{bmatrix} \eta_0 \\ \eta_1 \\ \vdots \\ \eta_M \end{bmatrix}.$$

Then we get:

$$U = Y + 12[\beta_2 A - \beta_2 B - \beta_3 C + \beta_4 D + \beta_5 E - \beta_5 L + \beta_6 G + \beta_6 H]F + [\beta_7 E - 2\beta_7 L + \beta_7 G - \beta_7 H - \eta]U \quad (10)$$

where

$$\beta_2 = \frac{\beta_0}{\gamma^2\tau^2[(\beta_0-1)k_2+(\beta_0-\beta_1)k_1]}, \beta_3 = \frac{(k_1\beta_1+k_2)}{(2\beta_1-2)k_2+(2\beta_0-2\beta_1)k_1}, \beta_4 = \frac{(k_1+k_2)\beta_0}{(2\beta_1-2)k_2+(2\beta_0-2\beta_1)k_1}$$
$$\beta_5 = \frac{(\gamma^2\tau^2h^2\beta_1-2\beta_1+2\beta_0)k_1+(\gamma^2\tau^2h^2-2+2\beta_0)k_2}{2h\gamma^2\tau^2[(\beta_0-1)k_2+(\beta_0-\beta_1)k_1]}, \beta_6 = \frac{(k_1+k_2)\beta_0}{\gamma^2\tau^2[(\beta_0-1)k_2+(\beta_0-\beta_1)k_1]}$$
$$\beta_7 = \frac{(\beta_0-1)k_2+(\beta_0-\beta_1)k_1}{\gamma^2\tau^2[(\beta_0-1)k_2+(\beta_0-\beta_1)k_1]}$$

Then by substituting equation (8) into equation (10) we obtained:

$$U = Y + \frac{12}{h^2}[\beta_2 A - \beta_2 B - \beta_3 C + \beta_4 D + \beta_5 E - \beta_5 L + \beta_6 G + \beta_6 H]QU + [\beta_7 E - 2\beta_7 L + \beta_7 G - \beta_7 H - \boldsymbol{\eta}]U$$

Then

$$U = [I - Q_0 Q - Q_1]^{-1} Y \quad (11)$$

where $Q_0 = \frac{12}{h^2}[\beta_2 A - \beta_2 B - \beta_3 C + \beta_4 D + \beta_5 E - \beta_5 L + \beta_6 G + \beta_6 H]$ and $Q_1 = [\beta_7 E - 2\beta_7 L + \beta_7 G - \beta_7 H - \eta]$. An approximation solution of equation (1) found by solving equation (11), the following equation is an approximate solution of u_k

$$\begin{aligned} S_k(z) &= \left(\frac{\beta_0(\hat{F}_k - \hat{F}_{k+1})}{\gamma^2\tau^2[(\beta_0-1)k_2+(\beta_0-\beta_1)k_1]}\right)\left(k_1 e^{\gamma\tau(z-z_k)} + k_2 e^{-\gamma\tau(z-z_k)}\right) \\ &+ \left(\frac{(k_1+k_2)\beta_0\hat{F}_{k+1} - (k_1\beta_1+k_2)\hat{F}_k}{(2\beta_1-2)k_2+(2\beta_0-2\beta_1)k_1}\right)(z-z_k)^2 + \left[\frac{\gamma^2\tau^2\left((1-\beta_0)k_2+(\beta_1-\beta_0)k_1\right)[\hat{u}_k - 2\hat{u}_{k+1}]}{2h\gamma^2\tau^2[(\beta_0-1)k_2+(\beta_0-\beta_1)k_1]} + \right. \\ &\frac{[(\gamma^2\tau^2h^2\beta_1 - 2\beta_1 + 2\beta_0)k_1 + (\gamma^2\tau^2h^2 - 2 + 2\beta_0)k_2]\hat{F}_k}{2h\gamma^2\tau^2[(\beta_0-1)k_2+(\beta_0-\beta_1)k_1]} \\ &\left. + \frac{[(-\gamma^2\tau^2h^2\beta_0 - 2\beta_0 + 2\beta_1)k_1 + (-\gamma^2\tau^2h^2\beta_0 + 2 - 2\beta_0)k_2]\hat{F}_{k+1}}{2h\gamma^2\tau^2[(\beta_0-1)k_2+(\beta_0-\beta_1)k_1]}\right](z-z_k) + \\ &\frac{[(\beta_0-1)k_2+(\beta_0-\beta_1)k_1]\hat{u}_k + (k_1+k_2)(\hat{F}_{k+1}-\hat{F}_k)\beta_0}{\gamma^2\tau^2[(\beta_0-1)k_2+(\beta_0-\beta_1)k_1]} + O(h^4), k = 0,1,\dots,n \end{aligned} \tag{12}$$

4. Convergence and Error Estimation

In this section, the convergence of the NCSF method for solving equation (1) discussed by using the following lemma and theorems.

Lemma 4.1. [20, 21] Assume A be a square matrix with $\|A\| < 1$ then the matrix $(I - A)$ is invertible, in addition to $\|(I-A)^{-1}\|_\infty \le \frac{1}{1-\|A\|_\infty}$.

Lemma 4.2. The matrix $[I - Q_0Q - Q_1]$ is invertible if

$$\|Q_0\|_\infty \|Z^{-1}\|_\infty \|R\|_\infty + \|Q_1\|_\infty \le 1,$$

Proof. Let $A := Q_0Q + Q_1$, by Lemma (4.1) $I - A$ be invertible and

$$\|(I - Q_0Q - Q_1)^{-1}\|_\infty \le \frac{1}{1 - \|Q_0Q + Q_1\|_\infty}, \text{ if } \|A\|_\infty < 1$$

In the following the condition that $\|A\|_\infty < 1$ has been clarified: Assume that $k(z,x)$ is bounded on $[a, b(z)]^2$, and the parameters γ, τ, k_1, and k_2 are fixed; then the integral blocks A, B, C, D, E, L, G, and H are $O(h^2), O(h)$, and $O(1)$ as given by their definitions, and the β_i in equation (10) are smooth functions of h, therefore $h_0 > 0$ such that $0 < h < h_0$, so

$$\|Q_0Q + Q_1\|_\infty \le \|Q_0\|_\infty \|Q\|_\infty + \|Q_1\|_\infty = \|Q_0\|_\infty \|z^{-1}\|_\infty \|R\|_\infty + \|Q_1\|_\infty,$$

Thus it is sufficient to enforce

$$\|Q_0\|_\infty \|Z^{-1}\|_\infty \|R\|_\infty + \|Q_1\|_\infty \le 1.$$

Hence $[I - Q_0Q - Q_1]$ is invertible and equation (11) admits a unique solution.

Theorem 4.1. we assume that u is the exact solution and S be an approximate solution, then $\left\|u - S_\Delta^h\right\|_\infty \le C_{\text{tot}}\, h^4$.

Proof: Let $U^* = \begin{bmatrix} u(z_0) \\ u(z_1) \\ \vdots \\ u(z_j) \end{bmatrix}$ be the exact nodal vector and U^h the numerical solution of equation (11), by using equations (7)-(9), and the affected system is obtained by entering the exact value of u into equations (9)-(11).

$$\big(I - (Q_0Q + Q_1)\big)U^\star = Y + r_h, \|r_h\|_\infty \le Ch^4 \tag{13}$$

where C depends on $\max_j \left\|u^{(j)}\right\|_\infty$ and bounds on k, η. Let $e := U^\star - U^h$. Subtracting $(I - Q_0Q - Q_1)U^h = Y$ from (13) gives

$$(I - Q_0Q - Q_1)e = r_h \;\Rightarrow\; e = (I - Q_0Q - Q_1)^{-1}r_h$$

From Lemma (4.1) and $\|r_h\|_\infty \le Ch^4$,

$$\left\|U^{\star}-U^{h}\right\|_{\infty} \leq \frac{C}{1-\|Q_0 Q+Q_1\|_{\infty}} h^4 \tag{14}$$

Let S_{Δ}^{h} be the global spline built from $\left(U^{h}, F^{h}\right)$ with $F^{h}=\frac{12}{h^2} Q U^{h}$ (from (8)). For the exact data $(U^{\star}, F^{\star})$, the interpolation formula (12) gives

$$u(z)-\mathcal{S}(U^{\star}, F^{\star})(z)=O(h^4) \text{ for all } z \in [a, b(z)] \tag{15}$$

Decompose the error:

$$u-S_{\Delta}^{h}=\underbrace{u-\mathcal{S}(U^{\star}, F^{\star})}_{\text{interpolation remainder}}+\underbrace{\mathcal{S}(U^{\star}, F^{\star})-\mathcal{S}\left(U^{h}, F^{h}\right)}_{\text{data-to-spline- stability}}. \tag{16}$$

The first term is $O(h^4)$ by equation (15).

For the second term, the spline map $\mathcal{S}$ is linear and bounded:

$$\left\|\mathcal{S}(U^{\star}, F^{\star})-\mathcal{S}\left(U^{h}, F^{h}\right)\right\|_{\infty} \leq C_S\left(\left\|U^{\star}-U^{h}\right\|_{\infty}+h^2\left\|F^{\star}-F^{h}\right\|_{\infty}\right) \tag{17}$$

From the continuity relation with truncation $t_h=O(h^2)$,

$$F^{\star}-F^{h}=\frac{12}{h^2} Q\left(U^{\star}-U^{h}\right)+Q t_h. \tag{18}$$

So

$$\left\|F^{\star}-F^{h}\right\|_{\infty} \leq \frac{12}{h^2}\|Q\|_{\infty}\left\|U^{\star}-U^{h}\right\|_{\infty}+\|Q\|_{\infty}\|t_h\|_{\infty}. \tag{19}$$

Combining equations (17)-(19) with equation (14) and $\|t_h\|_{\infty}=O(h^2)$ obtained:

$$\left\|\mathcal{S}(U^{\star}, F^{\star})-\mathcal{S}\left(U^{h}, F^{h}\right)\right\|_{\infty} \leq \frac{C_S(1+12\|Q\|_{\infty})C}{1-\|Q_0 Q+Q_1\|_{\infty}} h^4+O(h^4)$$

From equation (16),

$$\left\|u-S_{\Delta}^{h}\right\|_{\infty} \leq C_1 h^4+\frac{C_2}{1-\|Q_0 Q+Q_1\|_{\infty}} h^4=C_{\text{tot}}\, h^4$$

Therefore, under $\|Q_0 Q+Q_1\|_{\infty}<1$,

$$\left\|U^{\star}-U^{h}\right\|_{\infty} \leq \frac{C_0}{1-\|Q_0 Q+Q_1\|_{\infty}} h^4, \left\|u-S_{\Delta}^{h}\right\|_{\infty} \leq C_{\text{tot}}\, h^4$$

Here C_0, C_{tot} depend only on the bounds of $u^{(j)}, k, \eta$, and the fixed method parameters (γ, τ, k_1, k_2), therefore, it follows $\|e\| \to 0$ as $h \to 0$, then the convergence of the fourth-order NCSF method has been explained.

5. Numerical examples and applications

To demonstrate the effectiveness and precision of the proposed six non-polynomial cubic spline function method, some numerical examples are presented, these examples are chosen to cover a range of problem types, to illustrate the method's applicability in practical scenarios, all computational reports obtained by Python 3.12.4.

Example 5.1[15, 6, 18, 1]

Let us consider first-order Fredholm integro-differential equation

$$u'(z)=1-\frac{z}{3}+\int_0^1 zxu(x)dx$$

$$u(0)=0$$

where $n=1, \eta_0=0, \eta_1=1, y(z)=1-\frac{z}{3}, k(z,x)=zx, a=0, b(z)=1$, and $u(z)=z$ is the exact solution.

Reports the numerical results of this example in Table 2. presents a comparison between the exact solution and the results obtained using the NCSF methods for $M=10$, at each collocation point z_k, and Table 3. reports the absolute errors between the numerical approximations obtained by NCSF methods and the best results reported in earlier works for $M=10$, also Table 10. consisted of least square errors for different M.

Table 2. Absolute Errors with M = 10 for Example 5.1**.**

z_k	NCSF. 1	NCSF. 2	NCSF. 3	NCSF. 4	NCSF. 5	NCSF. 6	Best in [1]	Best in [15], [6] and [18].
0	0.0	0.0	0.0	0.0	0.0	0.0	-	1.0×10^{-9}
0.1	0.0	1.2×10^{-15}	1.4×10^{-17}	0.0	0.0	1.4×10^{-17}	1.4×10^{-17}	1.0×10^{-9}
0.2	0.0	2.8×10^{-17}	2.8×10^{-17}	0.0	0.0	2.8×10^{-17}	0.0	1.0×10^{-9}
0.3	0.0	0.0	1.1×10^{-16}	0.0	0.0	1.1×10^{-16}	5.6×10^{-17}	1.0×10^{-9}
0.4	0.0	0.0	0.0	0.0	0.0	0.0	1.1×10^{-16}	1.0×10^{-9}
0.5	5.6×10^{-17}	0.0	2.2×10^{-16}	0.0	5.5×10^{-17}	2.2×10^{-16}	1.1×10^{-16}	1.0×10^{-9}
0.6	0.0	1.1×10^{-16}	2.2×10^{-16}	0.0	0.0	2.2×10^{-16}	1.1×10^{-16}	1.0×10^{-9}
0.7	0.0	0.0	2.2×10^{-16}	0.0	0.0	2.2×10^{-16}	1.1×10^{-16}	2.0×10^{-9}
0.8	1.2×10^{-16}	1.1×10^{-16}	3.3×10^{-16}	0.0	1.1×10^{-16}	4.4×10^{-16}	1.1×10^{-16}	3.0×10^{-9}
0.9	1.1×10^{-16}	4.4×10^{-16}	4.9×10^{-15}	0.0	1.1×10^{-16}	4.9×10^{-15}	1.1×10^{-16}	2.0×10^{-9}
1	1.1×10^{-16}	1.1×10^{-16}	1.1×10^{-16}	1.1×10^{-16}	1.1×10^{-16}	1.1×10^{-16}	2.2×10^{-16}	1.3×10^{-1}

Table 3. Comparison of Exact solution and NCSF method for Example 5.1.

z_k	τ	Exact	NCSF. 1	NCSF. 2	NCSF. 3	NCSF. 4	NCSF. 5	NCSF. 6
0	1	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.2	1	0.2	0.2	0.2	0.2	0.2	0.2	0.2
0.4	1	0.4	0.4	0.4	0.4	0.4	0.4	0.4
0.6	1	0.6	0.6	0.6	0.6	0.6	0.6	0.6
0.8	1	0.8	0.8	0.8	0.8	0.8	0.8	0.8
1	1	1	1	1	1	1	1	1
L.S.E.	1	-	2.8×10^{-32}	2.3×10^{-28}	2.5×10^{-29}	1.2×10^{-32}	2.8×10^{-32}	2.5×10^{-29}

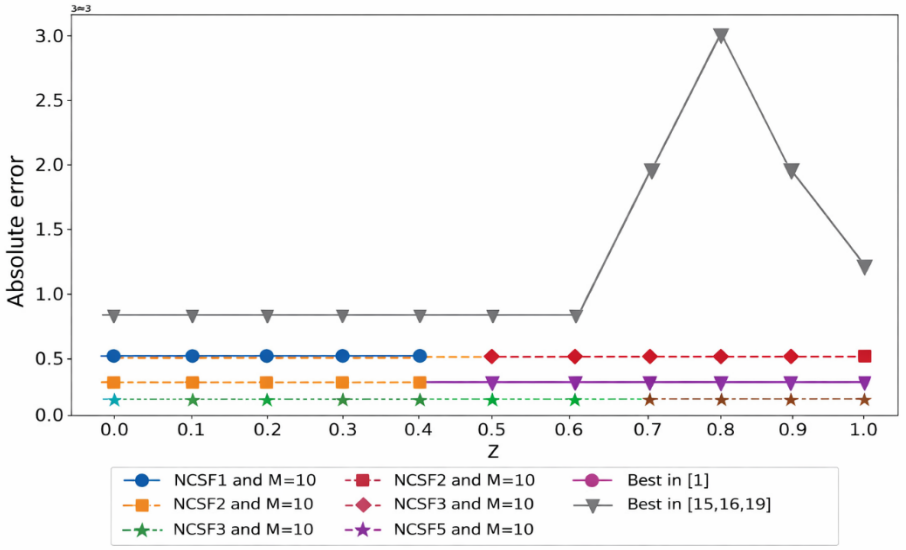

Fig. 2. Comparison of absolute errors between NCSF methods and previous methods for Example 5.1.

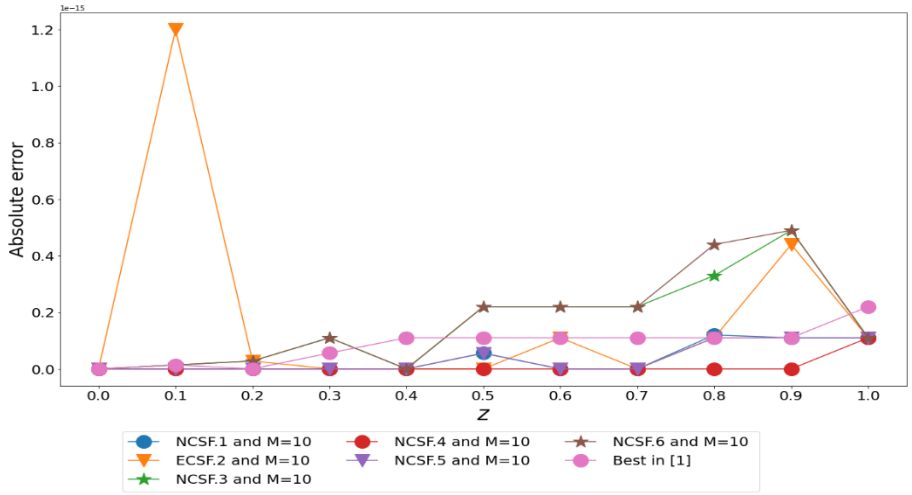

Fig. 3. Comparison of absolute errors between NCSF methods and Method in [1] for Example 5.1.

Example 5.2[15, 6]
Let us consider first-order fredholm integro-differential equation

$$u'(z) = ze^z + e^z - z + \int_0^1 zu(x)dx$$

$$u(0) = 0$$

where $n = 1, \eta_0 = 0, \eta_1 = 1, y(z) = ze^z + e^z - z, k(z,x) = z, a = 0, b(z) = 1$, and $u(z) = ze^z$ is the exact solution. Reports the numerical results of this example in Table 4. presents a comparison between the exact solution and the results obtained using the NCSF methods for $M = 10$, at each collocation point z_k, and Table 5. reports the absolute errors between the numerical approximations obtained by NCSF methods and the best results reported in earlier works for $M = 10$, also Table 10. consisted of least square errors for different M.

Table 4. Comparison of Exact solution and NCSF method for Example 5.2.

z_k	τ	Exact	NCSF. 1	NCSF. 2	NCSF. 3	NCSF. 4	NCSF. 5	NCSF. 6
0	5×10^2	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.2	5×10^2	0.24	0.25	0.28	0.24	0.24	0.25	0.24
0.4	5×10^2	0.60	0.61	0.66	0.60	0.58	0.58	0.57
0.6	5×10^2	1.09	1.12	1.05	1.08	0.98	0.98	1.01
0.8	5×10^2	1.78	1.69	1.47	1.64	1.47	1.47	1.56
1	5×10^2	2.7	2.06	2.06	2.1	2.12	2.14	2.14
L.S.E.	5×10^2	-	9.2×10^{-2}	3.7×10^{-1}	2.4×10^{-1}	3.5×10^{-1}	3.4×10^{-1}	2.7×10^{-1}

Table 5. Absolute Errors with M=10 for Example 5.2.

z_k	NCSF. 1	NCSF. 2	NCSF. 3	NCSF. 4	NCSF. 5	NCSF. 6	Best in [15]	Best in [6]
0	0.0	0.0	0.0	0.0	0.0	0.0	-	-
0.1	1.6×10^{-4}	1.3×10^{-2}	6.3×10^{-4}	1.3×10^{-3}	1.3×10^{-3}	2.4×10^{-3}	1.4×10^{-3}	1.0×10^{-2}
0.2	1.1×10^{-3}	4.0×10^{-2}	2.1×10^{-3}	1.1×10^{-3}	2.2×10^{-3}	1.2×10^{-3}	1.2×10^{-3}	2.8×10^{-2}
0.3	8.2×10^{-3}	6.1×10^{-2}	6.9×10^{-4}	2.1×10^{-3}	2.4×10^{-3}	1.8×10^{-2}	5.7×10^{-3}	5.1×10^{-2}
0.4	1.9×10^{-2}	6.1×10^{-2}	1.9×10^{-4}	1.9×10^{-2}	1.9×10^{-2}	3.3×10^{-2}	5.9×10^{-2}	7.6×10^{-2}
0.5	3.0×10^{-2}	3.0×10^{-2}	1.5×10^{-3}	5.2×10^{-2}	5.2×10^{-2}	5.2×10^{-2}	1.3×10^{-2}	9.7×10^{-2}
0.6	3.0×10^{-2}	4.3×10^{-2}	1.3×10^{-2}	1.1×10^{-1}	1.1×10^{-1}	8.2×10^{-2}	4.4×10^{-2}	1.1×10^{-1}
0.7	6.8×10^{-4}	1.6×10^{-1}	1.3×10^{-2}	1.9×10^{-1}	1.9×10^{-1}	1.5×10^{-1}	1.4×10^{-2}	1.0×10^{-1}
0.8	2.3×10^{-2}	3.1×10^{-2}	1.4×10^{-2}	3.1×10^{-2}	3.1×10^{-2}	2.2×10^{-2}	1.4×10^{-2}	7.0×10^{-2}
0.9	2.9×10^{-2}	4.9×10^{-2}	4.7×10^{-2}	4.5×10^{-2}	4.4×10^{-2}	4.4×10^{-2}	1.3×10^{-2}	1.0×10^{-2}
1	1.3×10^{-1}	2.1×10^{-1}	1.2×10^{-1}	1.2×10^{-1}	5.8×10^{-1}	5.8×10^{-1}	-	1.6×10^{-1}

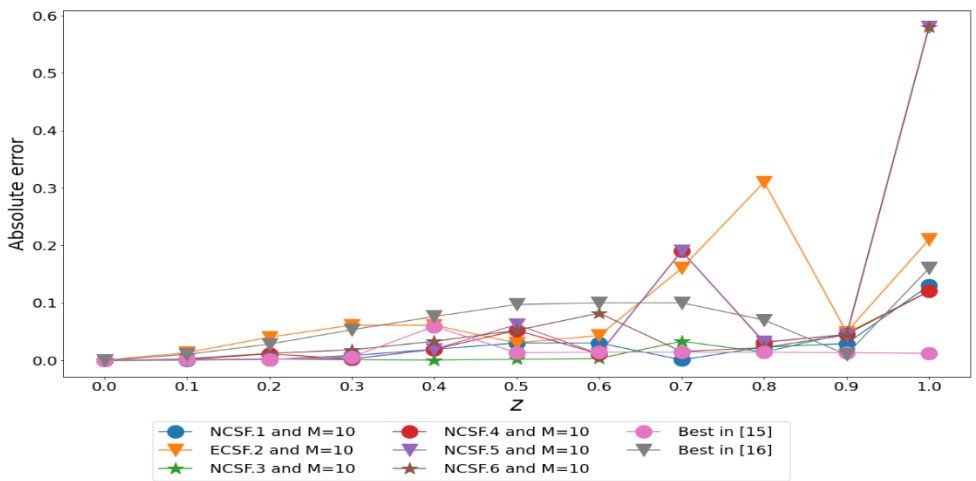

Fig. 4. Comparison of absolute errors between NCSF methods and previous methods for Example

5.2.

Looking at Tables 4 and 5, we can see that Example 2 has bigger absolute errors than Example 1. This is mostly because the test problem is getting harder and more sensitive, which causes bigger truncation and discretisation errors for all number schemes. Still, the numbers show that the suggested method makes a lot less error than other methods that have been published before when the same discretisation parameters are used. In this case, it shows that the proposed method is more accurate and reliable, even for harder assessments.

Example 5.3

Let us consider first-order volterra integro-differential equation

$$u(z) + u'(z) = \sin(z) - 1 + 2\cos(z) + \int_0^z u(x)dx$$

$$u(0) = 0$$

where $n = 1, \eta_{0,1} = 1, y(z) = \sin(z) - 1 + 2\cos(z), k(z,x) = 1, a = 0, b(z) = z$, and $u(z) = \sin(z)$ is the exact solution.

Reports the numerical results of this example in Table 6. presents a comparison between the exact solution and the results obtained using the NCSF methods for $M = 10$, at each collocation point z_k, and Table 7. reports the absolute errors obtained by the NCSF methods for $M = 20$, also Table 10. consisted of least square errors for different M.

Table 6. Comparison of Exact solution and NCSF method for Example 5.3.

z_k	τ	Exact	NCSF. 1	NCSF. 3	NCSF. 4	NCSF. 5	NCSF. 6
0	1×10^2	0.0	4.3×10^{-8}	2.8×10^{-4}	0.0	0.97	3.6×10^{-4}
0.2	1×10^2	0.20	0.20	0.20	0.21	0.20	0.20
0.4	1×10^2	0.39	0.39	0.4	0.42	0.41	0.39
0.6	1×10^2	0.57	0.59	0.59	0.63	0.63	0.59
0.8	1×10^2	0.72	0.78	0.78	0.83	0.82	0.77
1	1×10^2	0.84	0.97	0.98	0.98	0.98	0.97
L.S.E.	1×10^2	-	1.4×10^{-2}	2.2×10^{-2}	2.4×10^{-1}	9.9×10^{-1}	2.4×10^{-2}

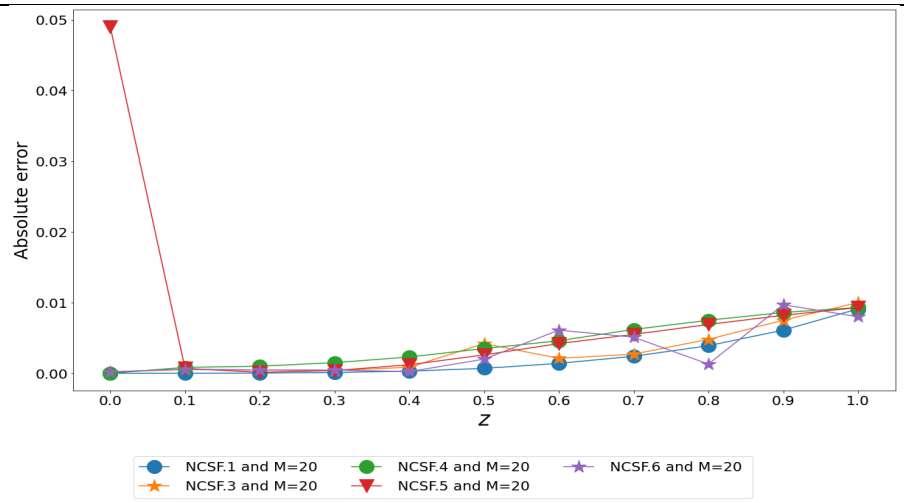

Fig. 5. Comparison of absolute errors between NCSF methods for Example 5.3.

Table 7. Absolute Errors with M=10 for Example 5.3.

z_k	NCSF. 1	NCSF. 3	NCSF. 4	NCSF. 5	NCSF. 6
0	8.1×10^{-7}	1.9×10^{-4}	0.0	4.9×10^{-2}	2.1×10^{-4}
0.1	1.2×10^{-6}	5.1×10^{-4}	8.3×10^{-4}	6.9×10^{-4}	5.7×10^{-4}
0.2	2.5×10^{-5}	4.7×10^{-4}	1.0×10^{-4}	3.1×10^{-4}	4.7×10^{-4}
0.3	1.1×10^{-4}	3.8×10^{-4}	1.5×10^{-3}	3.8×10^{-4}	9.1×10^{-4}
0.4	3.0×10^{-4}	8.5×10^{-4}	2.3×10^{-3}	1.2×10^{-3}	3.2×10^{-4}
0.5	6.9×10^{-4}	4.2×10^{-4}	3.5×10^{-3}	2.6×10^{-3}	2.0×10^{-3}
0.6	1.4×10^{-3}	2.1×10^{-3}	4.8×10^{-3}	4.0×10^{-3}	6.0×10^{-4}
0.7	2.4×10^{-3}	2.7×10^{-3}	6.2×10^{-3}	5.5×10^{-3}	4.5×10^{-3}
0.8	3.9×10^{-3}	4.8×10^{-3}	7.5×10^{-3}	6.9×10^{-3}	3.1×10^{-3}
0.9	6.1×10^{-3}	7.5×10^{-3}	8.6×10^{-3}	8.2×10^{-3}	9.7×10^{-3}
1	9.1×10^{-3}	1.0×10^{-2}	9.3×10^{-3}	9.3×10^{-3}	8.0×10^{-3}

Example 5.4
Let us consider third-order volterra integro-differential equation

$$u(z) + u'''(z) = -\frac{z^4}{4} + z^3 - \frac{z^2}{2} + z + 6 + \int_0^z u(x)dx$$
$$u(0) = 0, u'(0) = 1, u''(0) = 0$$

where $n = 3, \eta_{0,3} = 1, \eta_{1,2} = 0, y(z) = -\frac{z^4}{4} + z^3 - \frac{z^2}{2} + z + 6, k(z,x) = 1,, a = 0, b(z) = z$, and $u(z) = z^3 + z$ is the exact solution.
Reports the numerical results of this example in Table 8. presents a comparison between the exact solution and the results obtained using the NCSF methods for $M = 10$, at each collocation point z_k, and Table 9. reports the absolute errors obtained by the NCSF methods for $M = 20$, also Table 10. contained least square errors for different M.

Table 8. Comparison of Exact solution and NCSF method for Example 5.4.

z_k	τ	Exact	NCSF. 1	NCSF. 3	NCSF. 4	NCSF. 5	NCSF. 6
0	1×10^{-2}	0.0	0.0	1.18×10^{-4}	0.0	0.0	1.3×10^{-4}
0.2	1×10^{-2}	0.208	0.233	0.233	0.224	0.229	0.240
0.4	1×10^{-2}	0.464	0.499	0.515	0.499	0.502	0.527
0.6	1×10^{-2}	0.816	0.858	0.852	0.852	0.860	0.865
0.8	1×10^{-2}	1.312	1.354	1.347	1.347	1.356	1.360
1	1×10^{-2}	2	2.044	2.036	2.035	2.046	2.051
L.S.E.	1×10^{-2}	-	6.85×10^{-3}	2.2×10^{-2}	4.6×10^{-4}	6.9×10^{-3}	1.2×10^{-2}

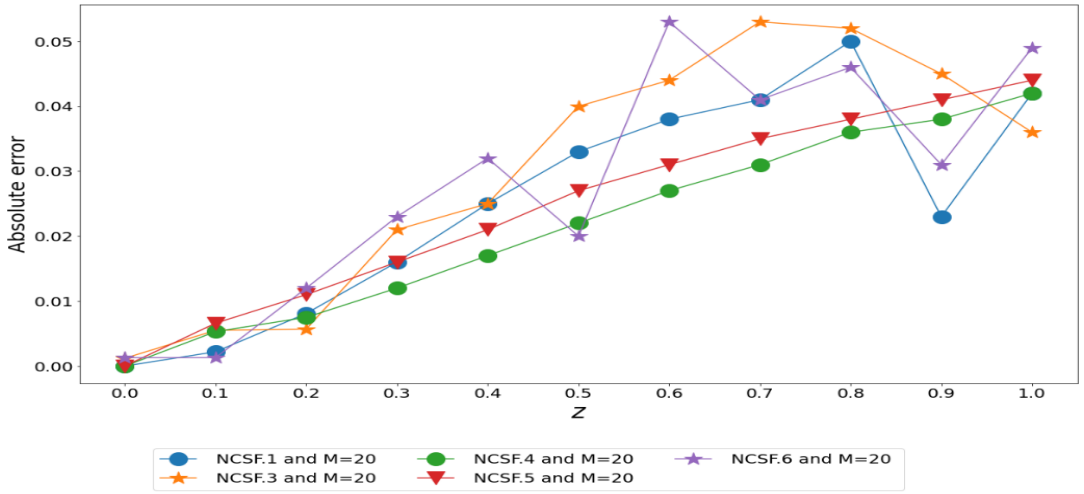

Fig. 6. Comparison of absolute errors between NCSF methods for Example 5.4.

Table 9. Absolute Errors with M=20 for Example 5.4.

z_k	NCSF. 1	NCSF. 3	NCSF. 4	NCSF. 5	NCSF. 6
0	0.0	1.2×10^{-3}	0.0	0.0	1.3×10^{-3}
0.1	2.2×10^{-3}	5.5×10^{-3}	5.3×10^{-3}	6.6×10^{-3}	1.3×10^{-3}
0.2	8.1×10^{-3}	5.7×10^{-3}	7.5×10^{-3}	1.1×10^{-2}	1.2×10^{-2}
0.3	1.6×10^{-2}	2.1×10^{-2}	1.2×10^{-2}	1.6×10^{-2}	1.3×10^{-2}
0.4	2.5×10^{-2}	2.5×10^{-2}	1.7×10^{-2}	2.1×10^{-2}	3.2×10^{-2}
0.5	3.3×10^{-2}	4.0×10^{-2}	2.2×10^{-2}	2.7×10^{-2}	2.0×10^{-3}
0.6	3.8×10^{-2}	4.4×10^{-2}	2.7×10^{-2}	3.1×10^{-2}	5.3×10^{-2}
0.7	4.1×10^{-2}	5.3×10^{-2}	3.1×10^{-2}	3.5×10^{-2}	4.1×10^{-2}
0.8	3.5×10^{-2}	5.1×10^{-2}	3.5×10^{-2}	3.8×10^{-2}	6.3×10^{-2}
0.9	2.3×10^{-2}	4.5×10^{-2}	3.8×10^{-2}	4.1×10^{-2}	3.1×10^{-2}
1	4.2×10^{-2}	3.6×10^{-2}	4.2×10^{-2}	4.4×10^{-2}	4.9×10^{-2}

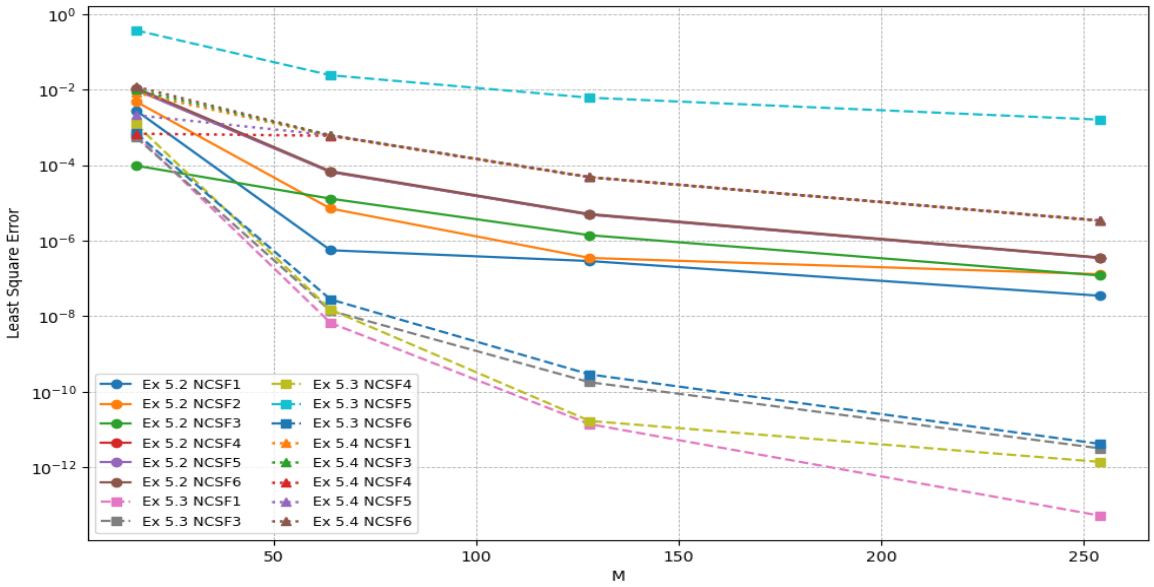

Fig. 7. Least Square Errors (Table 10)

Table 10. Least Square Errors.

Example	M	NCSF. 1	NCSF2	NCSF. 3	NCSF. 4	NCSF. 5	NCSF. 6
5.2	16	2.8×10^{-3}	4.8×10^{-3}	9.7×10^{-5}	9.9×10^{-3}	9.9×10^{-3}	1.1×10^{-2}
	64	5.6×10^{-7}	7.1×10^{-6}	1.3×10^{-5}	6.6×10^{-5}	6.6×10^{-5}	6.9×10^{-5}
	128	2.9×10^{-7}	3.5×10^{-7}	1.4×10^{-6}	4.9×10^{-6}	4.9×10^{-6}	5.1×10^{-6}
	254	3.5×10^{-8}	1.3×10^{-7}	1.2×10^{-7}	3.5×10^{-7}	3.5×10^{-7}	3.6×10^{-7}
5.3	16	6.3×10^{-4}	-	5.5×10^{-4}	1.2×10^{-3}	3.7×10^{-1}	7.0×10^{-4}
	64	6.7×10^{-9}	-	1.4×10^{-8}	1.5×10^{-8}	2.4×10^{-2}	2.8×10^{-8}
	128	1.4×10^{-11}	-	1.8×10^{-10}	1.7×10^{-11}	6.1×10^{-3}	2.9×10^{-10}
	254	5.3×10^{-14}	-	3.2×10^{-12}	1.4×10^{-12}	1.6×10^{-3}	4.2×10^{-12}
5.4	16	8.2×10^{-3}	-	9.9×10^{-3}	6.8×10^{-4}	2.1×10^{-3}	1.2×10^{-2}
	64	6.0×10^{-4}	-	6.2×10^{-4}	6.0×10^{-4}	6.2×10^{-4}	6.1×10^{-4}
	128	4.8×10^{-5}	-	4.8×10^{-5}	4.8×10^{-5}	4.8×10^{-5}	4.8×10^{-5}
	254	3.4×10^{-6}	-	3.4×10^{-6}	3.4×10^{-6}	3.4×10^{-6}	3.4×10^{-6}

In addition to the examples shown in Examples 1–4, Tables 2–10, and Figures 2–7, the suggested non-polynomial spline method is fully supported by these sources. The error values shown in the tables show that the numerical solutions are pretty close to the exact answers. This confirms up what we assumed would occur with convergence. In particular, the suggested method consistently gets better results than other methods that have been published with the same discretisation parameters, even though some examples have relatively larger absolute errors because the problems are more complicated. The table and graph results in the figures show that the suggested method works efficiently and consistently across a variety of test cases. Overall, these numerical experiments confirm the theoretical analysis and show that the proposed method works.

6. Conclusion

This study presents an effective numerical technique utilising six non-polynomial cubic spline functions to approximate solutions for integro–Volterra–Fredholm differential equations. The suggested method augments standard spline techniques by integrating new non-polynomial basis terms, therefore significantly enhancing the scheme's ability to represent oscillatory and rapidly shifting solution behaviours. The suggested approach offers significant advantages, including excellent numerical accuracy and computing efficiency, as demonstrated by the theoretical convergence analysis and accompanying numerical demonstrations. The method produces accurate approximations with fewer grid points relative to different existing numerical techniques, providing it both accurate and efficient. Furthermore, the consistent formulation enables the method to be applied effectively to both Volterra and Fredholm integro-differential cases.

A significant advantage of the proposed framework is its flexibility and extensibility, providing it accessible for academics aiming to construct novel numerical schemes grounded in comparable concepts. The construction procedure employed in this study can be systematically utilised to devise various non-polynomial spline methods by changing the basis functions, continuity criteria, or collocation techniques. The current method can be expanded to include adaptive spline formulations, integro-differential equation systems, and higher-dimensional challenges in future study. Subsequent study may concentrate on integrating the proposed spline technique with spectral or wavelet-based methodologies to enhance precision for intricate models encountered in applied sciences and engineering. These instructions offer a robust basis for researchers to develop and enhance the suggested numerical framework.

References

[1] O. M. Ogunlaran and M. O. Oke, “A numerical approach for solving first order integro-differential equations,” American Journal of Computational and Applied Mathematics, vol. 3, no. 4, pp. 214–219, 2013, DOI: 10.5923/j.ajcam.20130304.03.

[2] P. Singh et al., “Some studies on nonpolynomial interpolation and error analysis,” Applied Mathematics and Computation, vol. 244, pp. 809–821, 2014, DOI: 10.1016/j.amc.2014.07.049.

[3] S. Aggarwal, K. Bhatnagar, and A. Dua, “Method of Taylor’s series for the primitive of linear second kind non-homogeneous Volterra integral equations,” International Journal of Research and Innovation in Applied Science, vol. 5, no. 5, pp. 32–35, 2020. Available: https://rsisinternational.org/journals/ijrias

[4] A. Hosry, R. Nakad, and S. Bhalekar, “A hybrid function approach to solving a class of Fredholm and Volterra integro-differential equations,” Mathematical and Computational Applications, vol. 25, no. 2, p. 30, 2020, DOI: 10.3390/mca25020030.

[5] L. Dawood, A. Hamoud, and N. Mohammed, “Laplace discrete decomposition method for solving nonlinear Volterra-Fredholm integro-differential equations,” Journal of Mathematics and Computer Science, vol. 21, no. 2, pp. 158–163, 2020, DOI: 10.22436/jmcs.021.02.07.

[6] P. Darania and A. Ebadian, “A method for the numerical solution of the integro-differential equations,” Applied Mathematics and Computation, vol. 188, no. 1, pp. 657–668, 2007, DOI: 10.1016/j.amc.2006.10.046.

[7] M. Riahi Beni, “Legendre wavelet method combined with the Gauss quadrature rule for numerical solution of fractional integro-differential equations,” Iranian Journal of Numerical Analysis and

Optimization, vol. 12, no. 1, pp. 229–249, 2022. Available: https://ijnao.um.ac.ir/article_41459.html
[8] M. Masoud, "Numerical solution of systems of fractional order integro-differential equations with a Tau method based on monic Laguerre polynomials," Journal of Mathematical Analysis and Modeling, vol. 3, no. 3, pp. 1–13, 2022, DOI: 10.48185/jmam.v3i2.629.
[9] D. Xu, W. Qiu, and J. Guo, "A compact finite difference scheme for the fourth-order time-fractional integro-differential equation with a weakly singular kernel," Numerical Methods for Partial Differential Equations, vol. 36, no. 2, pp. 439–458, 2020, DOI: 10.1002/num.22436.
[10] R. Saber S., Y. Sabawi, and M. S. Hasso, "Numerical solution of the Fredholm integro-differential equations using high-order compact finite difference method," Journal of Educational Sciences, vol. 32, pp. 9–22, 2023, DOI: 10.33899/edusj.2023.138218.1332.
[11] M. H. Reihani and Z. Abadi, "Rationalized Haar functions method for solving Fredholm and Volterra integral equations," Journal of Computational and Applied Mathematics, vol. 200, no. 1, pp. 12–20, 2007, DOI: 10.1016/j.cam.2005.12.026.
[12] M. Erfanian and H. Zeidabadi, "Solving two-dimensional nonlinear Volterra integral equations using rationalized Haar functions," International Journal of Nonlinear Analysis and Applications, vol. 14, no. 8, pp. 95–105, 2023, DOI: 10.22075/ijnaa.2022.7226.
[13] M. Qayyum and I. Oscar, "Modification of homotopy perturbation algorithm through least square optimizer for higher order integro-differential equations," Punjab University Journal of Mathematics, pp. 45–57, 2023, DOI: 10.52280/pujm.2023.550201.
[14] S. J. Mohammedfaeq and R. J. Qadir, "An efficient fractional-order shifted Legendre function for solving multiterm high-order fractional-differential equations of Caputo-type with convergence analysis," Mathematical Methods in the Applied Sciences, 2025, DOI: 10.1002/mma.70408.
[15] H. Danfu and S. Xufeng, "Numerical solution of integro-differential equations by using CAS wavelet operational matrix of integration," Applied Mathematics and Computation, vol. 194, no. 2, pp. 460–466, 2007, DOI: 10.1016/j.amc.2007.04.048.
[16] A. O. Adesanya et al., "On approximate solution of high-order linear Fredholm integro-differential equations with variable coefficients using Legendre collocation method," Electronic Journal of Mathematical Analysis and Applications, vol. 11, no. 1, pp. 198–205, 2023. Available: http://math-frac.org/Journals/EJMAA/
[17] S. J. Mohammedfaeq and S. S. Ahmed, "Operational matrix of generalized block pulse functions for solving fractional Volterra-Fredholm integro-differential equations," Journal of Southwest Jiaotong University, vol. 57, no. 3, pp. 39–59, 2022, DOI: 10.35741/issn.0258-2724.57.3.4.
[18] B. Asady et al., "Direct method for solving integro differential equations using hybrid Fourier and block-pulse functions," International Journal of Computer Mathematics, vol. 82, no. 7, pp. 889–895, 2005, DOI: 10.1080/00207160412331336044.
[19] A. F. Fareed, A. G. Khattab, and M. S. Semary, "A novel stochastic ten non-polynomial cubic splines method for heat equations with noise term," Partial Differential Equations in Applied Mathematics, vol. 10, p. 100677, 2024, DOI: 10.1016/j.padiff.2024.100677.
[20] D. A. Hammad, M. S. Semary, and A. G. Khattab, "Ten non-polynomial cubic splines for some classes of Fredholm integral equations," Ain Shams Engineering Journal, vol. 13, no. 4, p. 101666, 2022, DOI: 10.1016/j.asej.2021.101666.
[21] F. K. Hamasalh and R. J. Qadir, "Non polynomial fractional spline method for solving Fredholm integral equations," Journal of Innovative Applied Mathematics and Computational Sciences, 2022. Available: https://n2t.net/ark:/49935/jiamcs.v2i3.51
[22] R. J. Qadir and F. Hamasalh, "Fractional spline model for computing Fredholm integral equations," in Proc. 2022 Int. Conf. Computer Science and Software Engineering (CSASE), IEEE, 2022, DOI: 10.1109/CSASE51777.2022.9759823.
[23] T. Tahernezhad and R. Jalilian, "Exponential spline for the numerical solutions of linear Fredholm integro-differential equations," Advances in Difference Equations, vol. 2020, no. 1, p. 141, 2020, DOI: 10.1186/s13662-020-02591-3.

Journal of Basrah Researches (Sciences) **52**(1), 31 (2026)
DOI: https://doi.org/10.56714/bjrs.52.1.3

مقاربة لمعادلات فولترّا وفريدهولم التكاملية-التفاضلية باستخدام ست دوال تكعيبية غير متعددة الحدود من نوع السبلاين

رەهيل جزا قادر[1] ، شاباز جليل محمدفائق[2,*] ، فريدون قادر حمەصالح[3]

[1]قسم العلوم الرياضية، كلية التربية الأساسية، جامعة السليمانية، السليمانية 46001، العراق.
[2]قسم الرياضيات، كلية العلوم، جامعة السليمانية، السليمانية 46001، العراق.
[3]قسم الرياضيات، كلية التربية، جامعة السليمانية، السليمانية 46001، العراق.

معلومات البحث

الاستلام 08 تشرين ثاني 2025
المراجعة 24 كانون ثاني 2026
القبول 25 كانون ثاني 2026
النشر 30 حزيران 2026

الكلمات المفتاحية

المنحنيات المكعبة غير متعددة، طريقة التجميع، تحليل التقارب، التقريب العددي، المعادلات التكاملية التفاضلية.

الملخص

تقدم هذه الدراسة طريقة عددية فعالة لتقريب حلول معادلات فولتيرا وفريدهولم التكاملية التفاضلية من خلال صياغة عامة جديدة لدوال التجزئة التكعيبية غير متعددة الحدود (NCSF). تُحسّن الطريقة المقترحة تقنيات التجزئة التكعيبية بدمج المكونات المثلثية والأسية باستخدام ست صيغ لدوال التجزئة التكعيبية غير متعددة الحدود. وقد أُجري تحليل شامل للتقارب، مما يُحدد الشروط الكافية لاستقرار الطريقة وحدود الخطأ. تمت معالجة مسائل اختبارية متعددة، بما في ذلك معادلات فولتيرا وفريدهولم التكاملية التفاضلية. وقورنت النتائج العددية بالحلول الدقيقة والمنهجيات المنشورة في الأدبيات، والمُقدمة في جداول وأشكال بيانية. تُوضح المقارنات فعالية ومتانة طريقة التجزئة التكعيبية غير متعددة الحدود الست المقترحة في معالجة مسائل فولتيرا وفريدهولم، مع الحصول على دقة أعلى باستخدام عدد أقل من نقاط الشبكة، كما يتضح من التقارير الحسابية المُولدة باستخدام بايثون 3.12.4.

Citation: R. J. Qadir et al., J. Basrah Res. (Sci.) **52**(1), 31 (2026).
DOI:https://doi.org/10.56714/bjrs.52.1.3

***Corresponding author email:** shabaz.mohammedfaeq@univsul.edu.iq

ISSN: 1817-2695 (Print); 2411-524X (Online)
Online at: https://jou.jobrs.edu.iq