NUMERICAL SOLUTION OF NONLINEAR INTEGRAL AND INTEGRO-DIFFERENTIAL EQUATIONS USING BIORTHOGONAL SPLINE WAVELET FULL-APPROXIMATION TRANSFORM METHOD

[1]R. A. Mundewadi, [2]B. A. Mundewadi
[1]P. A. College of Engineering, Mangalore, Karnataka, India.
[2]Govt. First Grade College for Women's, Bagalkot, Karnataka, India.

Abstract:

Biorthogonal spline wavelet full-approximation transform method is proposed for the numerical solution of nonlinear integral and integro-differential equations. It contains biorthogonal spline wavelet filter coefficients in the prolongation and restriction operators. The performance of the proposed method is better than the existing ones in terms of super convergence with low computational time. Some of the test problems are demonstrated for the applicability and efficiency of the scheme.

Keywords: Biorthogonal spline wavelets; Filter coefficients; Full-approximation scheme; Integral equations; Integro-differential equations.

1. INTRODUCTION

Linear and nonlinear phenomenon appearing in many applications in scientific fields can be modelled by integral and integro-differential equations. Specific applications of integral and integro-differential equations can be found in the mathematical modelling of spatiotemporal developments, epidemic modelling [1] and various biological and physical problems. Analytical solutions of integral and integro-differential equations, however, either do not exist or it is often hard to find. It is precisely due to this fact that several numerical methods have been developed for finding approximate solutions of integral and integro-differential equations [2-4].

In the historical three decades the development of effective iterative solvers for nonlinear systems of algebraic equations has been a significant research topic in numerical analysis, computational science and engineering. Brandt [5] was one of the first to introduce nonlinear multigrid method, which seeks to use concepts from the linear multigrid iteration and apply them directly in the nonlinear setting. Since the early application to elliptic partial differential equations, multigrid method have been applied successfully to a large and growing class of problems. Classical multigrid begins with a two-grid process. First, iterative relaxation is applied, whose effect is to smooth the error. Applying multigrid method directly to the nonlinear problems by employing the method so-called Full Approximation Scheme (FAS). In FAS, a nonlinear iteration, such as the nonlinear Gauss-Seidel method is applied to smooth the error and the residual is passed from the fine grids to the coarser grids. Vectors from fine grids are transferred to coarser grids with Restriction operator (R), while vectors are transferred from coarse grids to the finer grids with a Prolongation operator (P) respectively. For a detailed treatment of FAS is given in Briggs et al. [6]. An introduction of FAS is found in Hackbusch and Trottenberg [7], Wesseling [8] and Trottenberg et al. [9]. Many authors applied the FAS for some class of differential equations. The full-approximation scheme (FAS) is largely applicable in increasing the efficiency of the iterative methods used to solve nonlinear system of algebraic equations. FAS are a well-founded numerical method for solving nonlinear system of equations for approximating given differential equation. Subsequently, the development of multiresolution analysis and the fast wavelet transforms by Avudainayagam and Vani [10] and Bujurke et al. [11-13] led to extensive research in wavelet multigrid schemes to solve certain

differential equations arising in fluid dynamics. Beylkin et al. [14] observed that wavelet decomposition can be used to approximate the system of highly sparse matrices. Lee [15] has introduced a multigrid method for solving the nonlinear Urysohn integral equations. Ramane et al. [27] have applied a new Hosoya polynomial of path graphs for the numerical solution of Fredholm integral equations.

Wavelet analysis is a new branch of mathematics and widely applied in signal analysis, image processing and numerical analysis etc. The wavelet methods have proved to be very effective and efficient tool for solving problems of mathematical calculus. In recent years, these methods have attracted the interest of researchers of structural mechanics and many papers in this field are published. In most of the papers the Daubechies wavelets are applied. These wavelets are orthogonal, sufficiently smooth and have a compact support. Their shortcoming is that an explicit expression is lacking. This obstacle makes the differentiation and integration of these wavelets very complicated. For evaluation of such integrals the connection coefficients are introduced, but this complicates the course of the solution to a great extent [16]. Shiralashetti et al. [17, 18] applied the Haar wavelet collocation method for the numerical solution of multi-term fractional differential equations and singular initial value problems. Shilralashetti et al. [19] has proposed the wavelet based decoupled method for the investigation of surface roughness effects in elastohydrodynamic lubrication problems using couple stress fluid. Also, Shilralashetti et al. [20] have introduced a new wavelet based full-approximation scheme for the numerical solution of nonlinear elliptic partial differential equations.

Biorthogonal wavelet basis were introduced by Cohen-Daubechies-Feauveau in order to obtain wavelet pairs that are symmetric, regular and compactly supported. Unfortunately, this is incompatible with the orthogonality requirement that has to be dropped altogether. Biorthogonal wavelets build with splines are especially attractive because of their short support and regularity. So it is called a "Biorthogonal Spline Wavelets" [21]. In the biorthogonal case, rather than having one scaling and wavelet function, there are two scaling functions ϕ , $\tilde{\phi}$, that may generate different multiresolution analysis, and accordingly two different wavelet functions ψ , $\tilde{\psi}$. But biorthogonal wavelet based multigrid schemes are found to be effective [22]. Biorthogonal wavelet based multigrid schemes provide some remedy in such challenging cases. Sweldens [23] highlights effectively the construction of biorthogonal wavelet filters for the solution of large class of ill-conditioned system. In this paper, we developed the biorthogonal spline wavelet full-approximation transform method (BSWFATM) for the numerical solution of nonlinear integral and integro-differential equations using discrete biorthogonal spline wavelet transform (DBSWT) matrix. This matrix designed and implemented by Ruch and Fleet [24, 25] for decomposition and reconstruction of the given signals and images. Using these decomposition and reconstruction matrices we introduced restriction and prolongation operators respectively in the implementation of biorthogonal spline wavelet full-approximation transform method (BSWFATM).

The organization of this paper is as follows. In section 2, properties of biorthogonal spline wavelets are discussed. In Section 3, method of solution is presented. In section 4, method of implementation, numerical experiments and results are given. Finally, conclusion of the proposed work is given in section 5.

PROPERTIES OF BIORTHOGONAL WAVELETS

Discrete Biorthogonal Spline wavelet transform (DBSWT) matrix:

Let us consider the (5, 3) biorthogonal spline wavelet filter pair,

We have

$$\tilde{c} = (\tilde{c}_{-1}, \tilde{c}_0, \tilde{c}_1) = \left(\frac{\sqrt{2}}{4}, \frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{4} \right)$$

and

$$c = (c_{-2}, c_{-1}, c_0, c_1, c_2) = \left(\frac{-\sqrt{2}}{8}, \frac{\sqrt{2}}{4}, \frac{3\sqrt{2}}{4}, \frac{\sqrt{2}}{4}, \frac{-\sqrt{2}}{8} \right)$$

To form the highpass filters, We have

$$d_k = (-1)^k \tilde{c}_{1-k} \text{ and } \tilde{d}_k = (-1)^k c_{1-k}$$

The highpass filter pair d and $\tilde{d}$ for the (5, 3) biorthogonal spline filter pair.

$$d_0 = \frac{\sqrt{2}}{4}, d_1 = \frac{-\sqrt{2}}{2}, d_2 = \frac{\sqrt{2}}{4} \text{ and } \tilde{d}_{-1} = \frac{\sqrt{2}}{8}, \tilde{d}_0 = \frac{\sqrt{2}}{4}, \tilde{d}_1 = \frac{-3\sqrt{2}}{4}, \tilde{d}_2 = \frac{\sqrt{2}}{4}, \tilde{d}_3 = \frac{\sqrt{2}}{8}$$

In this paper, we use the filter coefficients which are,

Low pass filter coefficients: c_{-2}, c_{-1}, c_0, c_1, c_2 and High pass filter coefficients: d_0, d_1, d_2 for decomposition matrix.

Low pass filter coefficients: $\tilde{c}_{-1} = d_2$, $\tilde{c}_0 = -d_1$, $\tilde{c}_1 = d_0$ and High pass filter coefficients: $\tilde{d}_{-1} = -c_2, \tilde{d}_0 = c_1, \tilde{d}_1 = -c_0, \tilde{d}_2 = c_{-1}, \tilde{d}_3 = -c_{-2}$ for reconstruction matrix.

The matrix formulation of the discrete biorthogonal spline wavelet transforms (DBSWT) plays an important role in both biorthogonal spline wavelet transforms method (BSWTM) and biorthogonal Spline wavelet full-approximation transform method (BSWFATM) for the numerical computations. As we already know about the DBSWT matrix and its applications in the wavelet method and is given in [24] as,

Decomposition matrix:

$$D_W = \begin{pmatrix} c_{-1} & c_0 & c_1 & c_2 & 0 & 0 & \dots & 0 & 0 & c_{-2} \\ d_1 & d_2 & 0 & 0 & 0 & 0 & \dots & 0 & 0 & d_0 \\ 0 & c_{-2} & c_{-1} & c_0 & c_1 & c_2 & \dots & 0 & 0 & 0 \\ 0 & d_0 & d_1 & d_2 & 0 & 0 & \dots & 0 & 0 & 0 \\ \vdots & & \ddots & & & \dots & & & 0 & 0 \\ c_1 & c_2 & 0 & 0 & \dots & & \dots & 0 & c_{-2} & c_{-1} \\ 0 & 0 & 0 & 0 & \dots & & \dots & 0 & d_0 & d_1 \end{pmatrix}_{N \times N}$$

Reconstruction matrix:

$$R_W = \begin{pmatrix} \tilde{c}_0 & \tilde{c}_1 & 0 & 0 & 0 & 0 & \dots & 0 & 0 & \tilde{c}_{-1} \\ \tilde{d}_0 & \tilde{d}_1 & \tilde{d}_2 & \tilde{d}_3 & 0 & 0 & \dots & 0 & 0 & \tilde{d}_{-1} \\ 0 & \tilde{c}_{-1} & \tilde{c}_0 & \tilde{c}_1 & 0 & 0 & \dots & 0 & 0 & 0 \\ 0 & \tilde{d}_{-1} & \tilde{d}_0 & \tilde{d}_1 & \tilde{d}_2 & \tilde{d}_3 & \dots & 0 & 0 & 0 \\ \vdots & & \ddots & & & \dots & & & 0 & 0 \\ 0 & 0 & 0 & 0 & \dots & & \dots & 0 & \tilde{c}_{-1} & \tilde{c}_0 \\ \tilde{d}_2 & \tilde{d}_3 & 0 & 0 & \dots & & \dots & 0 & \tilde{d}_{-1} & \tilde{d}_0 \end{pmatrix}_{N \times N}$$

Biorthogonal Spline wavelet Full approximation transform operators:

Using the above matrices, we introduced biorthogonal spline wavelet restriction and biorthogonal spline wavelet prolongation operators respectively. i.e.,

Biorthogonal spline wavelet restriction operator:

$$BSWT_R = \begin{pmatrix} c_{-1} & c_0 & c_1 & c_2 & 0 & 0 & \dots & 0 & 0 & c_{-2} \\ d_1 & d_2 & 0 & 0 & 0 & 0 & \dots & 0 & 0 & d_0 \\ 0 & c_{-2} & c_{-1} & c_0 & c_1 & c_2 & \dots & 0 & 0 & 0 \\ 0 & d_0 & d_1 & d_2 & 0 & 0 & \dots & 0 & 0 & 0 \\ \vdots & & \ddots & & & \dots & & & 0 & 0 \\ 0 & 0 & \dots & 0 & d_0 & d_1 & d_2 & 0 & \dots & 0 & 0 \end{pmatrix}_{\frac{N}{2} \times N}$$

Biorthogonal spline wavelet prolongation operator:

$$BSWT_P = \begin{pmatrix} \tilde{c}_0 & \tilde{c}_1 & 0 & 0 & 0 & 0 & \dots & 0 & 0 & \tilde{c}_{-1} \\ \tilde{d}_0 & \tilde{d}_1 & \tilde{d}_2 & \tilde{d}_3 & 0 & 0 & 0 & \dots & 0 & \tilde{d}_{-1} \\ 0 & \tilde{c}_{-1} & \tilde{c}_0 & \tilde{c}_1 & 0 & 0 & \dots & 0 & 0 & 0 \\ 0 & \tilde{d}_{-1} & \tilde{d}_0 & \tilde{d}_1 & \tilde{d}_2 & \tilde{d}_3 & 0 & \dots & 0 & 0 \\ \vdots & & \ddots & & \ddots & & & \ddots & & \vdots \\ 0 & \dots & 0 & \tilde{c}_{-1} & \tilde{c}_0 & \tilde{c}_1 & 0 & 0 & \dots & 0 & 0 \\ 0 & \dots & 0 & \tilde{d}_{-1} & \tilde{d}_0 & \tilde{d}_1 & \tilde{d}_2 & \tilde{d}_3 & 0 & \dots & 0 \end{pmatrix}^T_{\frac{N}{2}\times N}.$$

Modified Discrete Biorthogonal Spline wavelet transform (MDBSWT) matrix:

Here, we developed MDBSWT matrix from DBSWT matrix in which by adding rows and columns consecutively with diagonal element as 1, which is built as,

New decomposition matrix:

$$MD_W = \begin{pmatrix} c_{-1} & 0 & c_0 & 0 & c_1 & 0 & c_2 & 0 & \dots & 0 & 0 & 0 & c_{-2} & 0 \\ 0 & 1 & 0 & 0 & \dots & & \dots & & \dots & 0 & 0 & 0 & 0 & 0 \\ d_1 & 0 & d_2 & 0 & \dots & & \dots & & \dots & 0 & 0 & 0 & d_0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & \dots & & \dots & 0 & 0 & 0 & 0 & 0 \\ \vdots & & & \ddots & & & \dots & & \dots & & \ddots & & \vdots & \\ c_1 & 0 & c_2 & 0 & \dots & & 0 & 0 & 0 & c_{-2} & 0 & c_{-1} & 0 & c_0 & 0 \\ 0 & 0 & & \dots & & & \dots & & \dots & & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & & \dots & & \dots & & 0 & d_0 & 0 & d_1 & 0 & d_2 & 0 \\ 0 & 0 & & \dots & & & \dots & & \dots & & 0 & 0 & 0 & 0 & 1 \end{pmatrix}_{N\times N}$$

New reconstruction matrix:

$$MR_W = \begin{pmatrix} \tilde{c}_0 & 0 & \tilde{c}_1 & 0 & 0 & 0 & 0 & \dots & 0 & 0 & \tilde{c}_{-1} & 0 \\ 0 & 1 & 0 & 0 & \dots & & \dots & & \dots & 0 & 0 & 0 & 0 \\ \tilde{d}_0 & 0 & \tilde{d}_1 & 0 & \tilde{d}_2 & 0 & \tilde{d}_3 & \dots & 0 & 0 & 0 & \tilde{d}_{-1} & 0 \\ 0 & 0 & 0 & 1 & 0 & & \dots & & \dots & 0 & 0 & 0 & 0 \\ \vdots & & & \ddots & & & \dots & & \dots & & \ddots & & \vdots & \\ 0 & 0 & & \dots & & & 0 & \tilde{c}_{-1} & 0 & \tilde{c}_0 & 0 & \tilde{c}_1 & 0 \\ 0 & 0 & 0 & \dots & & & \dots & & \dots & 0 & 1 & 0 & 0 \\ \tilde{d}_2 & 0 & \tilde{d}_3 & \dots & 0 & 0 & 0 & \tilde{d}_{-1} & 0 & \tilde{d}_0 & 0 & \tilde{d}_1 & 0 \\ 0 & 0 & 0 & \dots & & & \dots & & \dots & 0 & 0 & 0 & 1 \end{pmatrix}_{N\times N}$$

Modified Biorthogonal Spline Wavelet Full Approximation Transform operators:

Using the above matrices, we introduced a new biorthogonal spline wavelet restriction and prolongation operators respectively as,

New biorthogonal spline wavelet restriction operator:

$$MBSWT_R = \begin{pmatrix} c_{-1} & 0 & c_0 & 0 & c_1 & 0 & c_2 & 0 & 0 & 0 & \dots & 0 & c_{-2} & 0 \\ 0 & 1 & 0 & 0 & \dots & 0 & 0 & \dots & \dots & & & 0 & 0 & \\ d_1 & 0 & d_2 & 0 & \dots & 0 & 0 & \dots & \dots & 0 & 0 & d_0 & 0 & \\ 0 & 0 & 0 & 1 & 0 & \dots & 0 & 0 & \dots & \dots & & & 0 & 0 \\ 0 & 0 & c_{-2} & 0 & c_{-1} & 0 & c_0 & 0 & c_1 & 0 & c_2 & 0 & \dots & 0 & 0 \\ 0 & 0 & d_0 & 0 & d_1 & 0 & d_2 & 0 & 0 & \dots & \dots & 0 & 0 & 0 \\ \vdots & & \ddots & & \ddots & & \ddots & & \ddots & & & \ddots & & \\ 0 & 0 & \dots & 0 & d_0 & 0 & d_1 & 0 & d_2 & 0 & 0 & \dots & 0 & 0 \end{pmatrix}_{\frac{N}{2}\times N}$$

New biorthogonal spline wavelet prolongation operator:

$$MBSWT_P = \begin{pmatrix} \tilde{c}_0 & 0 & \tilde{c}_1 & 0 & 0 & 0 & \dots & 0 & 0 & \tilde{c}_{-1} & 0 \\ 0 & 1 & 0 & 0 & \dots & 0 & 0 & \dots & \dots & 0 & 0 \\ \tilde{d}_0 & 0 & \tilde{d}_1 & 0 & \tilde{d}_2 & 0 & \tilde{d}_3 & 0 & \dots & 0 & \tilde{d}_{-1} & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & \dots & 0 & 0 & \dots & 0 \\ 0 & 0 & \tilde{c}_{-1} & 0 & \tilde{c}_0 & 0 & \tilde{c}_1 & 0 & 0 & \dots & 0 & 0 \\ \vdots & & \ddots & & & \ddots & & & \ddots & & \vdots & \\ 0 & \dots & 0 & \tilde{d}_{-1} & 0 & \tilde{d}_0 & 0 & \tilde{d}_1 & 0 & \tilde{d}_2 & 0 & \tilde{d}_3 & 0 \dots & 0 \end{pmatrix}^T_{\frac{N}{2}\times N}$$

3. Biorthogonal Spline Wavelet Full-Approximation Transform Method (BSWFATM) of solution

Consider the Nonlinear Fredholm integral equation of the second kind,

$$u(t) = f(t) + \int_0^1 K(t,s,u(s))\,ds, \qquad 0 \le t,s \le 1, \tag{3.1}$$

Consider the Nonlinear Volterra integral equation of the second kind,

$$u(t) = f(t) + \int_0^t K(t,s,u(s))\,ds, \qquad 0 \le t,s \le 1, \tag{3.2}$$

where $K(t, s, u(s))$ is a nonlinear function defined on [0, 1]×[0, 1]. The known function $K(t, s, u(s))$ is called the kernel of the integral equation, while the unknown function $u(t)$ represents the solution of the integral equation. After discretizing the integral equation through the trapezoidal discretization method (TDM) [26], we get the system of nonlinear equations of the form,

$$\text{i.e.,} \quad F(u) = f \tag{3.3}$$

where F is $M \times M$ coefficient matrix, I is the identity matrix, f is $M \times 1$ matrix and u is $M \times 1$ matrix to be determined. This has M equations with M unknowns. Solving the system of Eq. (3.3) through the iterative method that is Gauss Seidel (GS), we get approximate solution v of u . i.e., $u = e + v \Rightarrow v = u - e$, where e is ($M \times 1$ matrix) error to be determined.

Now, we are deliberating about the Biorthogonal Spline Wavelet Full-Approximation Transform Method (BSWFATM) of solutions given by Briggs et. al [6] is as follows the procedure,

From the system Eq. (3.3), we get the approximate solution v for u . Now we find the residual as

$$r_{M\times 1} = f_{M\times 1} - F(v)_{M\times 1}. \tag{3.4}$$

Reduce the matrices in the finer level to coarsest level using Biorthogonal Spline Wavelet Restriction operator and then construct the matrices back to finer level from the coarsest level using Biorthogonal Spline Wavelet Prolongation operator as given in section 2.

Next,

$$r_{\frac{M}{2}\times 1} = [BSWT_R]_{\frac{M}{2}\times M}[r]_{M\times 1}.$$

Similarly, $$v_{\frac{M}{2}\times 1} = [BSWT_R]_{\frac{M}{2}\times M}[v]_{M\times 1}$$

and $$F(v_{\frac{M}{2}\times 1} + e_{\frac{M}{2}\times 1}) + F(v_{\frac{M}{2}\times 1}) = r_{\frac{M}{2}\times 1}. \quad (3.5)$$

Solve Eq. (3.5) with initial guess '0', we get $e_{\frac{M}{2}\times 1}$.

Next,

$$r_{\frac{M}{4}\times 1} = [BSWT_R]_{\frac{M}{4}\times\frac{M}{2}}[r]_{\frac{M}{2}\times 1}.$$

Similarly, $$v_{\frac{M}{4}\times 1} = [BSWT_R]_{\frac{M}{4}\times\frac{M}{2}}[v]_{\frac{M}{2}\times 1}$$

and $$F(v_{\frac{M}{4}\times 1} + e_{\frac{M}{4}\times 1}) + F(v_{\frac{M}{4}\times 1}) = r_{\frac{M}{4}\times 1}. \quad (3.6)$$

Solve Eq. (3.6) with initial guess '0', we get $e_{\frac{M}{4}\times 1}$.

Next, The procedure is continue up to the coarsest level, we have,

$$r_{1\times 1} = [BSWT_R]_{1\times 2}[r]_{2\times 1}.$$

Similarly, $$v_{1\times 1} = [BSWT_R]_{1\times 2}[v]_{2\times 1}$$

and $$F(v_{1\times 1} + e_{1\times 1}) + F(v_{1\times 1}) = r_{1\times 1}. \quad (3.7)$$

Solve Eq. (3.7) we get, $e_{1\times 1}$.

Next, Interpolate error up to the finer level, i.e.

$$e_{2\times 1} = [BSWT_P]_{2\times 1}[e]_{1\times 1},$$

$$e_{4\times 1} = [BSWT_P]_{4\times 2}[e]_{2\times 1}$$

and so on we have, $$e_{M\times 1} = [BSWT_P]_{M\times\frac{M}{2}}[e]_{\frac{M}{2}\times 1}.$$

Lastly, Correct the solution with error, $u_{M\times 1} = [v]_{M\times 1} + [e]_{M\times 1}$.

This is the required solution of the given integral equation.

Similarly, we solve the Modified Biorthogonal Spline Wavelet Full Approximation Transform Method using the new wavelet intergrid operators.

4. METHOD OF IMPLEMENTATION

In this section, we implemented FAS, BSWFATM and MBSWFATM for the numerical solution of nonlinear integral and integro-differential equations and subsequently presented in tables and figures, here error analysis is considered as $E_{max} = \max|u_e - u_a|$, where u_e and u_a are exact and approximate solutions respectively.

Test problem 4.1 Firstly, consider Nonlinear Fredholm integral equations [27]

$$u(t) = -\sin(4t) - t^3\left(\frac{-367}{4096}\cos(4)\sin(4) + \frac{11357}{98304} - \frac{2095}{32768}\cos^2(4)\right) + \int_0^1 t^3 s^5 u^2(s)\,ds,\ 0 \le t \le 1 \quad (4.1)$$

which has the exact solution $u(t) = \sin(-4t)$. After discretizing the Eq. (4.1) through the trapezoidal discretization method (TDM), we get system of nonlinear algebraic equations of the form (for $M = 8$),

$$[F]_{8\times 8}[u]_{8\times 1} = [f]_{8\times 1} \quad (4.2)$$

Solving Eq. (4.2) through the iterative method, we get the approximate solution v of u . i.e., $u = e + v \Rightarrow v = u - e$, where 'e' is (8×1 matrix) error to be determined. The implementation of MBSWFATM is discussed in section 3 is as follows,

From Eq. (4.2), we find the residual as

$$r_{8\times1} = [b]_{8\times1} - [A]_{8\times8}[v]_{8\times1} \tag{4.3}$$

We get, $r_{8\times1}$ = [0 0 -3.98e-09 -3.69e-07 -5.75e-06 -1.50e-05 8.31e-05 6.44e-04]

We reduce the matrices in the finer level to coarsest level using Restriction operator $MBSWT_R$ and then construct the matrices back to finer level from the coarsest level using Prolongation operator $MBSWT_P^T$.

From Eq. (4.3),

$$r_{4\times1} = [MBSWT_R]_{4\times8}[r]_{8\times1} \tag{4.4}$$

Similarly, $v_{4\times1} = [MBSWT_R]_{4\times8}[v]_{8\times1}$

and $$A(v_{4\times1} + e_{4\times1}) + A(v_{4\times1}) = r_{4\times1}. \tag{4.5}$$

Solve Eq. (4.5) with initial guess '0', we get $e_{4\times1}$.

From Eq. (4.4),

$$r_{2\times1} = [MBSWT_R]_{2\times4}[r]_{4\times1} \tag{4.6}$$

Similarly, $v_{2\times1} = [MBSWT_R]_{2\times4}[v]_{4\times1}$

and $$A(v_{2\times1} + e_{2\times1}) + A(v_{2\times1}) = r_{2\times1}. \tag{4.7}$$

Solve Eq. (4.7) with initial guess '0', we get $e_{2\times1}$.

From Eq. (4.6),

$$r_{1\times1} = [MBSWT_R]_{1\times2}[r]_{2\times1} \tag{4.8}$$

Similarly, $v_{1\times1} = [MBSWT_R]_{1\times2}[v]_{2\times1}$

and $$A(v_{1\times1} + e_{1\times1}) + A(v_{1\times1}) = r_{1\times1}. \tag{4.9}$$

Solve Eq. (4.9) we get, $e_{1\times1}$.

From $e_{1\times1}$, Interpolate error up to the finer level, i.e.

$$e_{2\times1} = [MBSWT_P^T]_{2\times1}[e]_{1\times1},$$

$$e_{4\times1} = [MBSWT_P^T]_{4\times2}[e]_{2\times1}$$

and lastly we have, $$e_{8\times1} = [MBSWT_P^T]_{8\times4}[e]_{4\times1}. \tag{4.10}$$

We get $e_{8\times1}$ = [1.43e-06 7.32e-11 3.56e-05 -3.67e-07 1.03e-05 0 -7.15e-07 0]

From Eq. (4.10) Correct the solution with error $u_{8\times1} = v_{8\times1} + e_{8\times1}$.

Lastly, we get $u_{8\times1}$ is the required solution of Eq. (4.1). The numerical solutions of the given equation is obtained through the present method as explained in section 3 and are presented in comparison with the exact solution are shown in the table 1 and the figure 1, for M = 64. Maximum error analysis and CPU time are shown in table 2.

Table 1. Numerical results of test problem 4.1, for $M = 16$.

t	FAS	BSWFATM	MBSWFATM	EXACT
0	0.0000	0.0000	0.0000	0
0.0666	-0.2635	-0.2631	-0.2635	-0.2635
0.1333	-0.5084	-0.5083	-0.5081	-0.5084
0.2000	-0.7173	-0.7173	-0.7173	-0.7173
0.2666	-0.8755	-0.8755	-0.8754	-0.8756
0.3333	-0.9718	-0.9718	-0.9718	-0.9719
0.4000	-0.9994	-0.9994	-0.9994	-0.9995
0.4666	-0.9562	-0.9562	-0.9562	-0.9565
0.5333	-0.8454	-0.8454	-0.8454	-0.8459
0.6000	-0.6748	-0.6748	-0.6748	-0.6754
0.6666	-0.4564	-0.4564	-0.4564	-0.4572
0.7333	-0.2056	-0.2056	-0.2056	-0.2067
0.8000	0.0598	0.0597	0.0597	0.0583
0.8666	0.3212	0.3210	0.3210	0.3193
0.9333	0.5598	0.5595	0.5595	0.5578
1	0.7592	0.7587	0.7587	0.7568

Table 2. Maximum error and CPU time (in seconds) of the methods of test problem 4.1.

M	Method	E_{max}	Setup time	Running time	Total time
16	FAS	2.45e-03	0.0161	0.0915	0.1075
	BSWFATM	1.98e-03	0.0245	0.0753	0.0999
	MBSWFATM	1.98e-03	0.0102	0.0701	0.0803
32	FAS	5.54e-04	0.0285	0.4081	0.4366
	BSWFATM	3.17e-04	0.0148	0.3372	0.3520
	MBSWFATM	3.17e-04	0.0080	0.3222	0.3302
64	FAS	1.28e-04	0.2280	3.0487	3.2767
	BSWFATM	8.45e-05	0.0148	2.7879	2.8027
	MBSWFATM	8.45e-05	0.0075	1.8799	1.8873

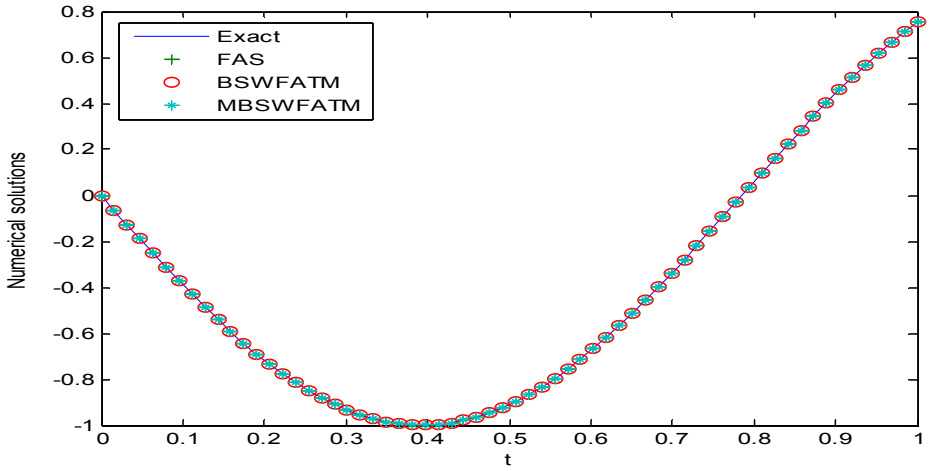

Fig. 1. Comparison of numerical solutions with exact solution of test problem 4.1, for M=64.

Test problem 4.2 Secondly, consider the nonlinear Fredholm-Hammerstein integral equation [28],

$$u(t) = t\ln(t+1) - \frac{55}{108}t + \frac{1}{3}\ln 2\left(\frac{8}{3}t + 2 - t\ln 2\right) - \frac{241}{576} + \frac{1}{2}\int_0^1 (t-s)u^2(s)\,ds, \quad 0 \le t \le 1 \tag{4.11}$$

which has the exact solution $u(t) = t\ln(t+1)$. The numerical solutions of Eq. (4.11) is obtained through the present method as explained in section 3 and are presented in comparison with the exact solution are shown in the table 3 and the figure 2, for M = 64. Maximum error analysis and CPU time are shown in table 4.

Table 3. Numerical results of test problem 4.2, for M = 16.

t	FAS	BSWFATM	MBSWFATM	EXACT
0	-0.0003	-0.0003	-0.0003	0
0.0666	0.0039	0.0039	0.0039	0.0043
0.1333	0.0163	0.0163	0.0163	0.0166
0.2000	0.0361	0.0361	0.0361	0.0364
0.2666	0.0627	0.0627	0.0627	0.0630
0.3333	0.0956	0.0956	0.0956	0.0958
0.4000	0.1343	0.1343	0.1343	0.1345
0.4666	0.1784	0.1784	0.1784	0.1787
0.5333	0.2277	0.2277	0.2277	0.2279
0.6000	0.2817	0.2817	0.2817	0.2820
0.6666	0.3403	0.3403	0.3403	0.3405
0.7333	0.4031	0.4031	0.4031	0.4033
0.8000	0.4700	0.4700	0.4700	0.4702
0.8666	0.5407	0.5407	0.5407	0.5409
0.9333	0.6151	0.6151	0.6151	0.6152
1	0.6930	0.6930	0.6930	0.6931

Table 4. Maximum error and CPU time (in seconds) of the methods of test problem 4.2.

M	Method	E_{max}	Setup time	Running time	Total time
16	FAS	3.66e-04	0.0274	0.0462	0.0736
	BSWFATM	3.66e-04	0.0178	0.0314	0.0492
	MBSWFATM	3.66e-04	0.0100	0.0289	0.0389
32	FAS	8.57e-05	0.0252	0.0483	0.0735
	BSWFATM	8.57e-05	0.0148	0.0447	0.0594
	MBSWFATM	8.57e-05	0.0100	0.0399	0.0500
64	FAS	2.07e-05	0.0806	0.1031	0.1836
	BSWFATM	2.07e-05	0.0148	0.0993	0.1141
	MBSWFATM	2.07e-05	0.0103	0.0954	0.1057
128	FAS	5.10e-06	0.1789	0.2082	0.3871
	BSWFATM	5.10e-06	0.0093	0.2195	0.2288
	MBSWFATM	5.10e-06	0.0063	0.1904	0.1966

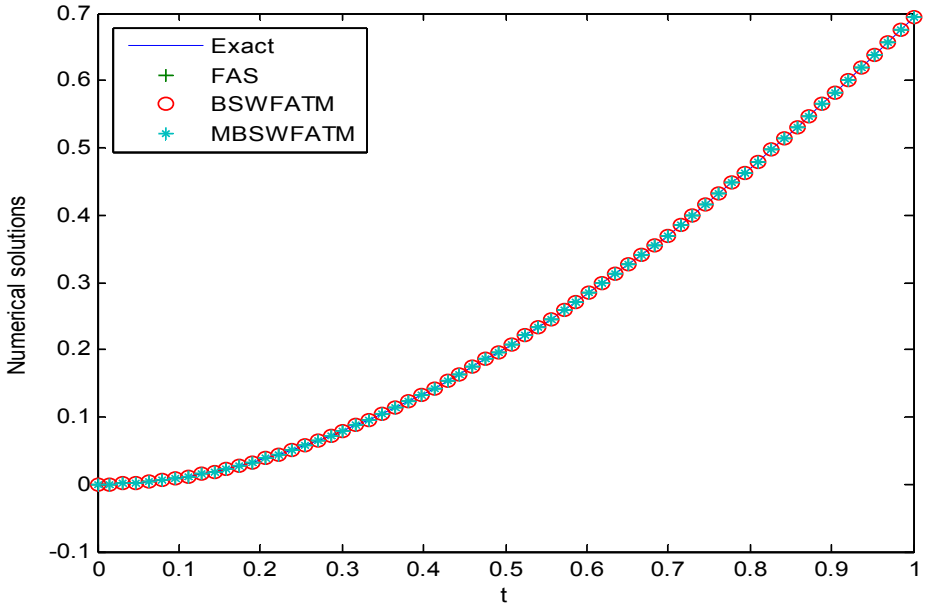

Fig. 2. Comparison of numerical solutions with exact solution of test problem 4.2, for M=64.

Test problem 4.3 Next, consider Nonlinear Fredholm-Hammerstein integro-differential equation [29],

$$u'(t) = 1 - \frac{1}{3}t + \int_0^1 t\,u^2(s)\,ds,\ u(0) = 0, \quad 0 \le t \le 1 \tag{4.12}$$

which has the exact solution $u(t) = t$.

Integrating the Eq. (4.12) w.r.to t and using the initial condition, we get

$$u(t) = t - \frac{t^2}{6} + \frac{t^2}{2}\int_0^1 u^2(s)\,ds,$$

Solving this equation, we obtain the numerical solution through the present method as explained in section 3 and are presented in comparison with the exact solution are shown in the table 5 and the figure 3, for M = 64. Maximum error analysis and CPU time are shown in table 6.

Table 5. Numerical results of test problem 4.3, for $M = 16$.

t	FAS	BSWFATM	MBSWFATM	EXACT
0	0.0000	0.0000	0.0000	0
0.0666	0.0666	0.0666	0.0666	0.0666
0.1333	0.1333	0.1333	0.1333	0.1333
0.2000	0.2000	0.2000	0.2000	0.2000
0.2666	0.2666	0.2666	0.2666	0.2666
0.3333	0.3333	0.3333	0.3333	0.3333
0.4000	0.4000	0.4000	0.4000	0.4000
0.4666	0.4666	0.4666	0.4666	0.4666
0.5333	0.5333	0.5333	0.5333	0.5333
0.6000	0.6000	0.6000	0.6000	0.6000
0.6666	0.6666	0.6666	0.6666	0.6666
0.7333	0.7333	0.7333	0.7333	0.7333
0.8000	0.7998	0.7998	0.7998	0.8000
0.8666	0.8665	0.8665	0.8665	0.8666
0.9333	0.9332	0.9332	0.9332	0.9333
1	0.9998	0.9998	0.9998	1

Table 6. Maximum error and CPU time (in seconds) of the methods of test problem 4.3.

M	Method	E_{max}	Setup time	Running time	Total time
16	FAS	1.70e-03	0.0157	0.1053	0.1210
	BSWFATM	1.70e-03	0.0205	0.0797	0.1002
	MBSWFATM	1.70e-03	0.0104	0.0668	0.0772
32	FAS	9.71e-04	0.0286	0.4099	0.4385
	BSWFATM	9.71e-04	0.0148	0.4074	0.4222
	MBSWFATM	9.71e-04	0.0100	0.4010	0.4110
64	FAS	5.13e-04	0.0633	2.8380	2.9012
	BSWFATM	5.13e-04	0.0150	2.4331	2.4481
	MBSWFATM	5.13e-04	0.0062	1.5218	1.5281

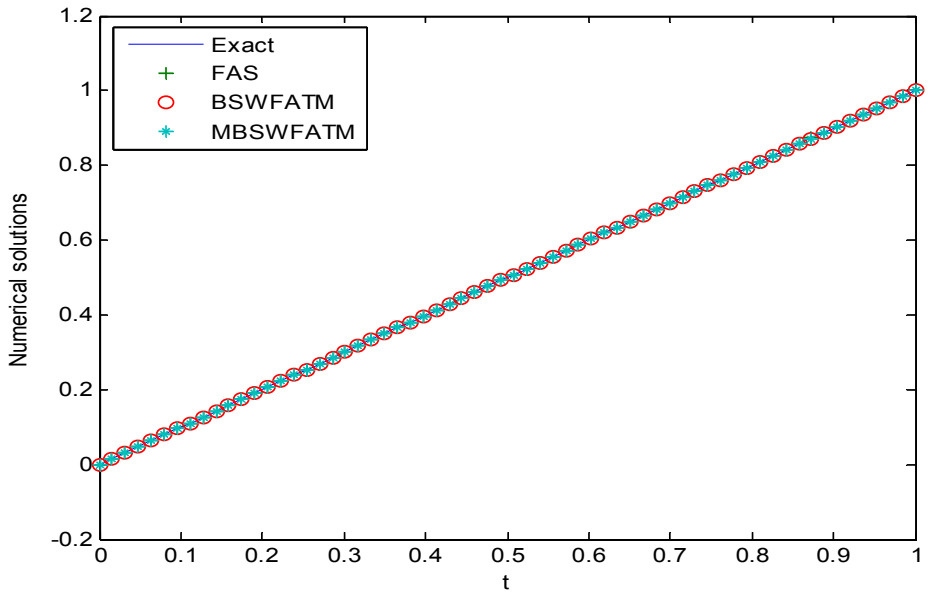

Fig. 3. Comparison of numerical solutions with exact solution of test problem 4.3, for M=64.

Test problem 4.4 Next, consider the Nonlinear Volterra integral equation [30],

$$u(t) = f(t) + \int_0^t t\, s^2\, u^2(s)\, ds, \;, \quad 0 \le t \le 1 \tag{4.13}$$

where $f(t) = \left(1 - \frac{11}{9}t + \frac{2}{3}t^2 - \frac{1}{3}t^3 + \frac{2}{9}t^4\right)\ln(t+1) - \frac{1}{3}(t+t^3)(\ln(t+1))^2 - \frac{11}{9}t^2 + \frac{5}{18}t^3 - \frac{2}{27}t^4.$

which has the exact solution $u(t) = \ln(t+1)$. After discretizing the Eq. (4.13) through the trapezoidal discretization method (TDM), we get system of nonlinear algebraic equations of the form (for $M = 8$),

$$[A]_{8\times 8}[u]_{8\times 1} = [b]_{8\times 1} \tag{4.14}$$

Solving Eq. (4.14) through the iterative method, we get the approximate solution v of u. i.e., $u = e + v \Rightarrow v = u - e$, where '$e$' is ($8\times 1$ matrix) error to be determined. The implementation of MBSWFATM is discussed in section 3 is as follows,

From Eq. (4.14), we find the residual as

$$r_{8\times 1} = [b]_{8\times 1} - [A]_{8\times 8}[v]_{8\times 1} \tag{4.15}$$

We get, $r_{8\times 1}$ = [0 0 3.52e-07 1.14e-05 1.29e-04 8.29e-04 3.66e-03 4.29e-03]

We reduce the matrices in the finer level to coarsest level using Restriction operator $MBSWT_R$ and then construct the matrices back to finer level from the coarsest level using Prolongation operator $MBSWT_P^T$.

From Eq. (4.15),

$$r_{4\times 1} = [MBSWT_R]_{4\times 8}[r]_{8\times 1} \tag{4.16}$$

Similarly, $v_{4\times 1} = [MBSWT_R]_{4\times 8}[v]_{8\times 1}$

and $$A(v_{4\times 1} + e_{4\times 1}) + A(v_{4\times 1}) = r_{4\times 1}. \tag{4.17}$$

Solve Eq. (4.17) with initial guess '0', we get $e_{4\times 1}$.

From Eq. (4.16),

$$r_{2\times 1} = [MBSWT_R]_{2\times 4}[r]_{4\times 1} \tag{4.18}$$

Similarly, $v_{2\times 1} = [MBSWT_R]_{2\times 4}[v]_{4\times 1}$

and $$A(v_{2\times 1} + e_{2\times 1}) + A(v_{2\times 1}) = r_{2\times 1}. \tag{4.19}$$

Solve Eq. (4.19) with initial guess '0', we get $e_{2\times 1}$.

From Eq. (4.18),

$$r_{1\times 1} = [MBSWT_R]_{1\times 2}[r]_{2\times 1} \tag{4.20}$$

Similarly, $v_{1\times 1} = [MBSWT_R]_{1\times 2}[v]_{2\times 1}$

and $$A(v_{1\times 1} + e_{1\times 1}) + A(v_{1\times 1}) = r_{1\times 1} \tag{4.21}$$

Solve Eq. (4.21) we get, $e_{1\times 1}$.

From $e_{1\times 1}$, Interpolate error up to the finer level, i.e.

$$e_{2\times 1} = [MBSWT_P^T]_{2\times 1}[e]_{1\times 1},$$

$$e_{4\times 1} = [MBSWT_P^T]_{4\times 2}[e]_{2\times 1}$$

and lastly we have, $$e_{8\times 1} = [MBSWT_P^T]_{8\times 4}[e]_{4\times 1}. \tag{4.22}$$

We get $e_{8\times 1}$ = [1.50e-05 0 1.75e-03 7.95e-06 5.03e-04 0 -7.53e-06 0]

From Eq. (4.22) Correct the solution with error $u_{8\times 1} = v_{8\times 1} + e_{8\times 1}$.

Lastly, we get $u_{8\times 1}$ is the required solution of Eq. (4.13). The numerical solutions of the given equation is obtained through the present method as explained in section 3 and are presented in comparison with the exact solution are shown in the table 7 and the figure 4 for M = 64. Maximum error analysis and CPU time is shown in table 8.

Table 7. Numerical results of test problem 4.4, for $M = 16$.

t	FAS	BSWFATM	MBSWFATM	EXACT
0	0.0000	0.0000	0.0000	0
0.0666	0.0645	0.0650	0.0645	0.0645
0.1333	0.1251	0.1253	0.1259	0.1251
0.2000	0.1823	0.1822	0.1823	0.1823
0.2666	0.2364	0.2364	0.2366	0.2363
0.3333	0.2877	0.2877	0.2877	0.2876
0.4000	0.3367	0.3367	0.3366	0.3364
0.4666	0.3835	0.3835	0.3835	0.3829
0.5333	0.4284	0.4284	0.4284	0.4274
0.6000	0.4717	0.4716	0.4716	0.4700
0.6666	0.5137	0.5135	0.5135	0.5108
0.7333	0.5544	0.5542	0.5542	0.5500
0.8000	0.5946	0.5940	0.5940	0.5877
0.8666	0.6341	0.6331	0.6331	0.6241
0.9333	0.6729	0.6718	0.6718	0.6592
1	0.6958	0.6947	0.6947	0.6931

Table 8. Maximum error and CPU time (in seconds) of the methods of test problem 4.4.

M	Method	$E_{\max}$	Setup time	Running time	Total time
16	FAS	1.36e-02	0.0158	0.0750	0.0908
	BSWFATM	1.26e-02	0.0147	0.0733	0.0880
	MBSWFATM	1.26e-02	0.0101	0.0676	0.0778
32	FAS	7.86e-03	0.0284	0.4068	0.4352
	BSWFATM	7.50e-03	0.0148	0.4064	0.4211
	MBSWFATM	7.50e-03	0.0100	0.3366	0.3467
64	FAS	4.19e-03	0.0833	3.2563	3.3396
	BSWFATM	4.09e-03	0.0149	3.0753	3.0903
	MBSWFATM	4.09e-03	0.0102	2.2629	2.2732

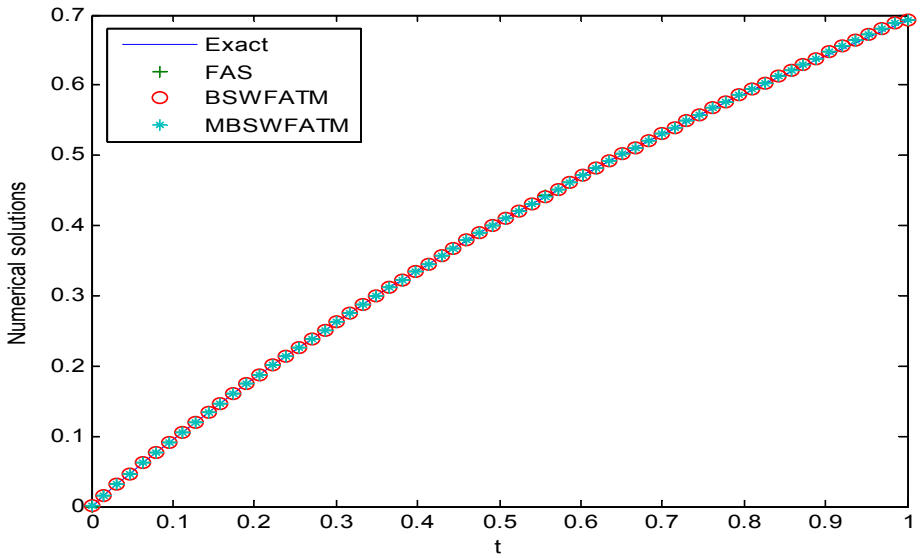

Fig. 4. Comparison of numerical solutions with exact solution of test problem 4.4, for M=64.

Test problem 4.5 Next, consider the Nonlinear Volterra-Hammerstein integral equation [31],

$$u(t) = \frac{-15}{56}t^8 + \frac{13}{14}t^7 - \frac{11}{10}t^6 + \frac{9}{20}t^5 + t^2 - t + \int_0^t (t+s)u^3(s)\,ds, \quad 0 \le t \le 1 \tag{4.23}$$

which has the exact solution $u(t) = t^2 - t$. The numerical solutions of Eq. (4.23) is obtained through the present method as explained in section 3 and are presented in comparison with the exact solution are shown in the table 9 and the figure 5, for M = 64. Maximum error analysis and CPU time are shown in table 10.

Table 9. Numerical results of test problem 4.5, for M = 16.

t	FAS	BSWFATM	MBSWFATM	EXACT
0	0.0000	0.0000	0.0000	0
0.0666	-0.0622	-0.0622	-0.0622	-0.0622
0.1333	-0.1155	-0.1155	-0.1155	-0.1155
0.2000	-0.1600	-0.1600	-0.1600	-0.1600
0.2666	-0.1957	-0.1957	-0.1957	-0.1955
0.3333	-0.2224	-0.2224	-0.2224	-0.2222
0.4000	-0.2403	-0.2404	-0.2403	-0.2400
0.4666	-0.2493	-0.2493	-0.2493	-0.2488
0.5333	-0.2494	-0.2494	-0.2494	-0.2488
0.6000	-0.2405	-0.2405	-0.2405	-0.2400
0.6666	-0.2227	-0.2227	-0.2227	-0.2222
0.7333	-0.1959	-0.1959	-0.1959	-0.1955
0.8000	-0.1602	-0.1602	-0.1602	-0.1600
0.8666	-0.1156	-0.1156	-0.1156	-0.1155
0.9333	-0.0622	-0.0622	-0.0622	-0.0622
1	-0.0000	-0.0000	-0.0000	0

Table 10. Maximum error and CPU time (in seconds) of the methods of test problem 4.5.

M	Method	E_{max}	Setup time	Running time	Total time
16	FAS	5.72e-04	0.0144	0.0949	0.1093
	BSWFATM	5.59e-04	0.0135	0.0915	0.1050
	MBSWFATM	5.57e-04	0.0100	0.0737	0.0837
32	FAS	2.81e-04	0.0208	0.4138	0.4346
	BSWFATM	2.77e-04	0.0083	0.3506	0.3588
	MBSWFATM	2.77e-04	0.0060	0.3348	0.3408
64	FAS	1.38e-04	0.0688	0.1026	0.1714
	BSWFATM	1.37e-04	0.0098	0.0948	0.1045
	MBSWFATM	1.37e-04	0.0072	0.0962	0.1034
128	FAS	6.87e-05	0.3143	0.4560	0.7703
	BSWFATM	6.85e-05	0.0100	0.4471	0.4571
	MBSWFATM	6.85e-05	0.0137	0.4430	0.4567

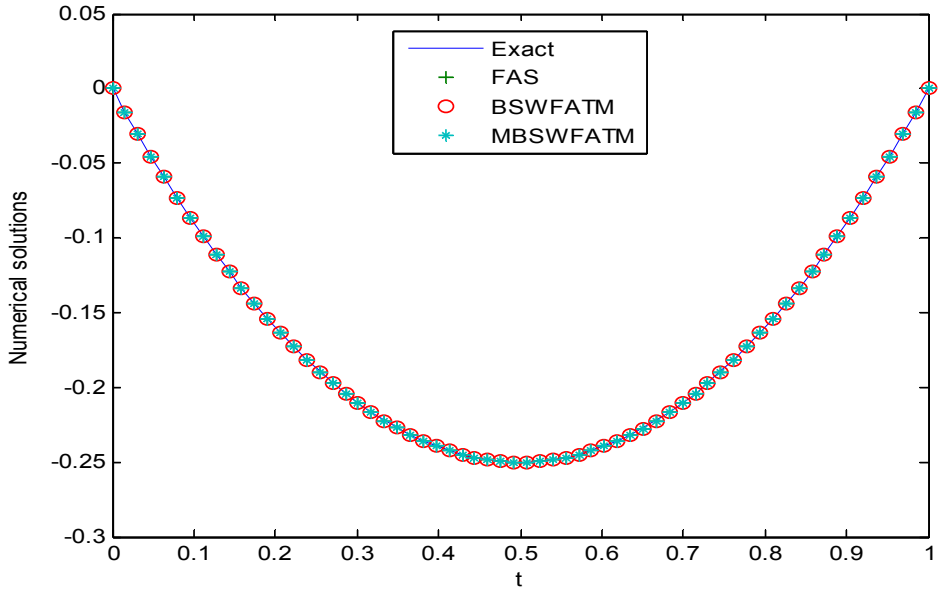

Fig. 5. Comparison of numerical solutions with exact solution of test problem 4.5, for M=64.

Test problem 4.6 Next, consider the Nonlinear Volterra integro-differential equation [32],

$$u'(t) = -1 + \int_0^t u^2(s)\,ds,\ u(0) = 0, \quad 0 \le t \le 1 \tag{4.24}$$

which has the exact solution $u(t) = -t + \frac{t^4}{12} - \frac{t^7}{252} + \frac{t^{10}}{6048} - \frac{t^{13}}{157248}$.

We convert the Volterra integro-differential equation to equivalent Volterra integral equation by using the well-known formula, which converts multiple integrals into a single integral.
i.e.,

$$\int_0^t \int_0^t \ldots\ldots \int_0^t u(t)dt^n = \frac{1}{(n-1)!}\int_0^t (t-s)^{n-1}u(s)ds$$

Integrating Eq. (4.24) on both sides from 0 to t and using the initial condition and also converting the double integral to the single integral, we obtain,

$$u(t) = f(t) + \int_0^t k(t,s)u^2(s)ds, \tag{4.25}$$

where $k(t, s)=(t-s)$ and $f(t)=-t$.

The numerical solutions of Eq. (4.25) is obtained through the present method as explained in section 3 and presented in table 11 for M = 16 and in figure 6 for M = 64. Maximum error analysis and CPU time is shown in table 12.

Table 11. Numerical results of test problem 4.6, for $M = 16$.

t	FAS	BSWFATM	MBSWFATM	EXACT
0	0.0000	0.0000	0.0000	0
0.0666	-0.0666	-0.0666	-0.0666	-0.0666
0.1333	-0.1333	-0.1333	-0.1333	-0.1333
0.2000	-0.1998	-0.1998	-0.1998	-0.1998
0.2666	-0.2662	-0.2662	-0.2662	-0.2662
0.3333	-0.3323	-0.3323	-0.3323	-0.3323
0.4000	-0.3979	-0.3979	-0.3979	-0.3978
0.4666	-0.4628	-0.4628	-0.4628	-0.4627
0.5333	-0.5267	-0.5267	-0.5267	-0.5266
0.6000	-0.5894	-0.5894	-0.5894	-0.5893
0.6666	-0.6505	-0.6505	-0.6505	-0.6504
0.7333	-0.7098	-0.7098	-0.7098	-0.7096
0.8000	-0.7668	-0.7668	-0.7668	-0.7666
0.8666	-0.8213	-0.8213	-0.8213	-0.8210
0.9333	-0.8727	-0.8727	-0.8727	-0.8724
1	-0.9207	-0.9207	-0.9207	-0.9204

Table 12. Maximum error and CPU time (in seconds) of the methods of test problem 4.6.

M	Method	E_{max}	Setup time	Running time	Total time
16	FAS	2.81e-04	0.0159	0.0349	0.0508
	BSWFATM	2.81e-04	0.0148	0.0317	0.0465
	MBSWFATM	2.81e-04	0.0130	0.0275	0.0404
32	FAS	6.56e-05	0.0233	0.0385	0.0618
	BSWFATM	6.56e-05	0.0117	0.0356	0.0474
	MBSWFATM	6.56e-05	0.0080	0.0318	0.0398
64	FAS	1.57e-05	0.0595	0.0763	0.1358
	BSWFATM	1.57e-05	0.0107	0.0719	0.0825
	MBSWFATM	1.57e-05	0.0072	0.0689	0.0761
128	FAS	3.69e-06	0.2930	0.3258	0.6188
	BSWFATM	3.69e-06	0.0149	0.3280	0.3430
	MBSWFATM	3.69e-06	0.0102	0.3207	0.3309

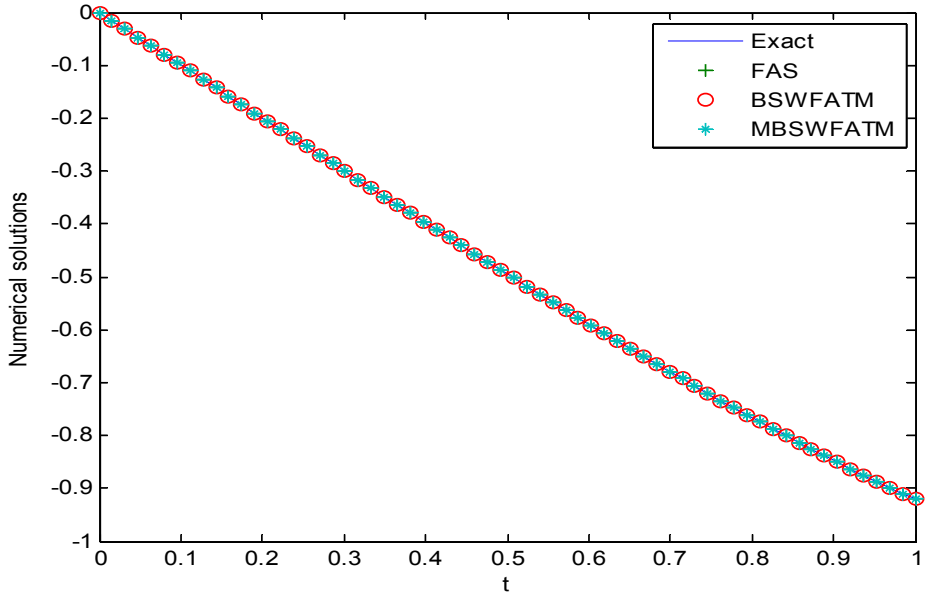

Fig. 6. Comparison of numerical solutions with exact solution of test problem 4.6, for M=64.

Test problem 4.7. Lastly, consider the nonlinear Volterra-Fredholm-Hammerstein integral equation [33],

$$u(t) = 1 + \sin^2(t) + \int_0^1 K(t,s)\,u^2(s)\,ds, \quad 0 \le t \le 1 \tag{4.26}$$

$$K(t,s) = \begin{cases} -3\sin(t-s), & 0 \le s \le t \\ 0, & t \le s \le 1 \end{cases}$$

which has the exact solution $u(t) = \cos t$. The numerical solutions of Eq. (4.26) is obtained through the present method as explained in section 3 and presented in comparison with the exact solution are shown in the table 13 and in the figure 7, for M = 64. Maximum error analysis and CPU time are shown in table 14.

Table 13. Numerical results of test problem 4.7, for M = 16.

t	FAS	BSWFATM	MBSWFATM	EXACT
0	1.0000	1.0000	1.0000	1
0.0666	0.9977	0.9977	0.9977	0.9977
0.1333	0.9911	0.9911	0.9911	0.9911
0.2000	0.9800	0.9800	0.9800	0.9800
0.2666	0.9646	0.9646	0.9646	0.9646
0.3333	0.9449	0.9449	0.9449	0.9449
0.4000	0.9209	0.9209	0.9209	0.9210
0.4666	0.8929	0.8929	0.8929	0.8930
0.5333	0.8610	0.8610	0.8610	0.8611
0.6000	0.8252	0.8252	0.8252	0.8253
0.6666	0.7857	0.7857	0.7857	0.7858
0.7333	0.7427	0.7427	0.7427	0.7429
0.8000	0.6965	0.6965	0.6965	0.6967
0.8666	0.6472	0.6472	0.6472	0.6473
0.9333	0.5950	0.5950	0.5950	0.5951
1	0.5401	0.5401	0.5401	0.5403

Table 14. Maximum error and CPU time (in seconds) of the methods of test problem 4.7.

M	Method	E_{max}	Setup time	Running time	Total time
16	FAS	1.60e-04	0.0160	0.0350	0.0510
	BSWFATM	1.60e-04	0.0148	0.0312	0.0460
	MBSWFATM	1.60e-04	0.0100	0.0280	0.0380
32	FAS	3.75e-05	0.0286	0.0485	0.0771
	BSWFATM	3.75e-05	0.0149	0.0448	0.0597
	MBSWFATM	3.75e-05	0.0101	0.0400	0.0501
64	FAS	9.10e-06	0.0790	0.1030	0.1820
	BSWFATM	9.10e-06	0.0149	0.0994	0.1143
	MBSWFATM	9.10e-06	0.0101	0.0953	0.1054
128	FAS	2.23e-06	0.2856	0.3512	0.6368
	BSWFATM	2.23e-06	0.0167	0.3224	0.3391
	MBSWFATM	2.23e-06	0.0103	0.3268	0.3371

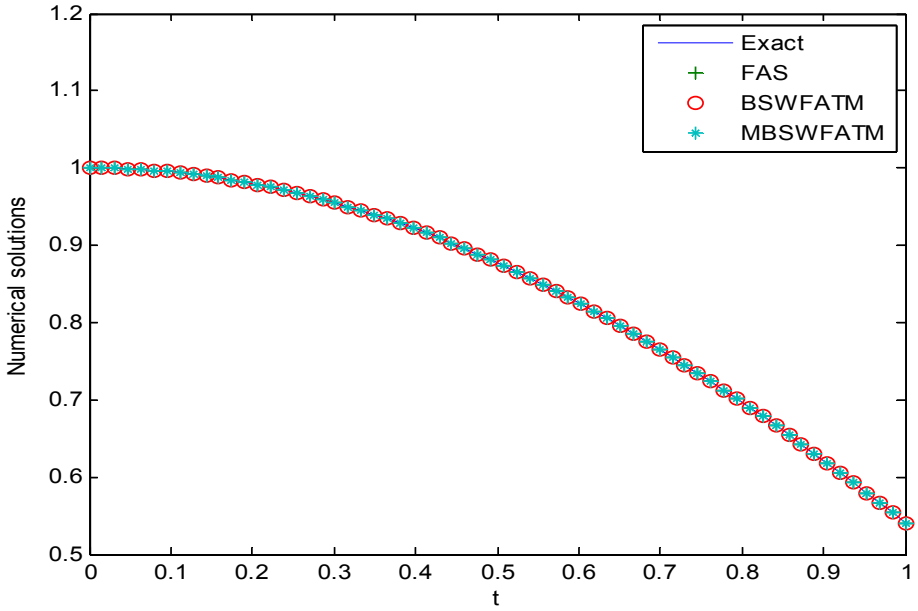

Fig. 7. Comparison of numerical solutions with exact solution of test problem 4.7, for M=64.

5. CONCLUSION

We proposed a biorthogonal spline wavelet transform method using wavelet intergrid operators based on biorthogonal spline wavelet filter coefficients for the numerical solution of integral and integro-differential equations. Wavelet intergrid operators of prolongation and restrictions are defined, a MBSWFATM, has been shown to be effective and versatile. Test problems are justified through the error analysis, as the level of resolution M increases for higher accuracy. The numerical solutions obtained agree well with the exact ones, as increasing the number M used. In this paper, the FAS, BSWFATM and MBSWFATM shows the error's are same, but the CPU time changes. The standard FAS and BSWFATM converges slowly with larger computational cost as compared to MBSWFATM ensure such slower convergence with lesser computational cost as shown in the tables. The numerical implementation from the tables demonstrates the accuracy of the approximations and super convergence phenomena with less CPU time. Hence, the proposed scheme is very convenient and efficient than the existing standard methods.

Acknowledgement
The authors thank for the financial support of UGC's UPE Fellowship vide sanction letter D. O. No. F. 14-2/2008(NS/PE), dated-19/06/2012 and F. No. 14-2/2012(NS/PE), dated 22/01/2013.

References

[1] Thiem, H.R.: A model for spatio spread of an epidemic. J. Math. Biol. **4**, 337–351 (1977).
[2] Wazwaz, A.M.: Linear and Nonlinear Integral Equations Methods and Applications. Springer, (2011)
[3] Jerri, A.: Introduction to integral equations with applications. A wiley-interscience publication, Wiley (1999)
[4] Atkinson, K.E.: The numerical solution of integral equations of the second kind. Cambridge university press, (1997)
[5] A. Brandt. Multi-level adaptive solutions to boundary-value problems. Math. Comp. 31 (1977) 333-390.
[6] W.L. Briggs, V.E. Henson, S.F. McCormick, A Multigrid Tutorial, SIAM Philadelphia, 2000.
[7] W. Hackbusch, U. Trottenberg, Multigrid Methods, Springer-Verlag, Berlin, 1982.
[8] P. Wesseling, An Introduction to Multigrid Methods, John Wiley, Chichester, 1992.
[9] U. Trottenberg, C. Oosterlee, A. Schuller, Multigrid, Academic Press, London, 2001.
[10] A. Avudainayagam, C. Vani, Wavelet based multigrid methods for linear and nonlinear elliptic partial differential equations, Appl. Math. Comp. 148 (2004) 307–320.
[11] G. Beylkin, R. Coifman, V. Rokhlin, Fast wavelet transforms and numerical algorithms-I, Commu. Pure. Appl. Maths. 44 (1991) 141-183.
[12] H. Lee, Multigrid method for nonlinear integral equations, Korean J. Comp. Appl. Math. 4 (1997) 427 – 440.
[13] G. Hariharan, K. Kannan, An Overview of Haar Wavelet Method for Solving Differential and Integral Equations, World Applied Sciences Journal 23(12) (2013) 01-14.
[14] M. Unser, Ten good reasons for using spline wavelets, Wavelets Applications in Signal and Image Processing V, proc. Spie. 3169 (1997) 422-431.
[15] A. Cohen, I. Daubechies, J. Feauvean, Biorthogonal based of compactly supported wavelets, Comm. Pure Appl. Math. 45 (1992) 485-560.
[16] W. Sweldens, The construction and application of wavelets in numerical analysis, PhD. Thesis, Department of Computer Science, Katholieke Universiteit Leuven, Belgium, 1994.
[17] D.K. Ruch, P.J.V. Fleet, Wavelet theory an elementary approach with applications, John Wiley and Sons, 2009.
[18] P.J.V. Fleet, Discrete wavelet transformations an elementary approach with applications, John Wiley and Sons, 2008.
[19] I.S. Kotsireas, A Survey on Solution Methods for Integral Equations, 2008.
[20] Y. Liu, Application of Chebyshev polynomial in solving Fredholm integral equations, Math. Comput. Model. 50 (2009) 465–469.
[21] J. Rashidinia, A. Parsa, Analytical-Numerical Solution for Nonlinear Integral Equations of Hammerstein Type, Inter. J. Math. Model. Comp. 02(01) (2012) 61 – 69.
[22] M. A. Abdou, M.M. El-Borai, M.M. El-Kojok, Toeplitz matrix method and nonlinear integral equation of Hammerstein type, J. Comput. Appl. Math. 223 (2009) 765–776.
[23] B. Sepehrian, M. Razzaghi, Solution of nonlinear volterra-hammerstein integral equations via single-term walsh series method, Math. Prob. Eng. 2005:5 (2005) 547–554. DOI: 10.1155/MPE.2005.547.
[24] J. Zhao, R.M. Corless, Compact finite difference method for integro-differential equations, Appl. Math. Comput. 177 (2006) 271–288.
[25] J.P. Kauthen, A survey of singularly perturbed Volterra equations, Appl. Numer. Math. 24 (1997) 95–114.
[26] Y. Mahmoudi, Wavelet Galerkin method for numerical solution of nonlinear integral equation, Appl. Math. Comp. 167 (2005) 1119–1129.
[27] Ramane H. S., Shiralashetti S. C., Mundewadi R. A., Jummannaver R. B., Numerical solution of Fredholm Integral Equations Using Hosoya Polynomial of Path Graphs, American Journal of Numerical Analysis, 5(1), 2017, 11-15.