NUMERICAL TREATMENTS FOR VOLTERRA DELAY INTEGRO-DIFFERENTIAL EQUATIONS

F. A. RIHAN¹, E. H. DOHA², M. I. HASSAN³, AND N. M. KAMEL³

Abstract — This paper presents a new technique for numerical treatments of Volterra delay integro-differential equations that have many applications in biological and physical sciences. The technique is based on the mono-implicit Runge — Kutta method (described in [12]) for treating the differential part and the collocation method (using Boole’s quadrature rule) for treating the integral part. The efficiency and stability properties of this technique have been studied. Numerical results are presented to demonstrate the effectiveness of the methodology.

2000 Mathematics Subject Classification: 65R20, 65Q05, 68W40, 45L05, 65Y20, 65L2.

Keywords: mono-implicit RK method, Boole’s quadrature rule, Volterra delay integro-differential equation, stability, time-lag.

1. Introduction

The last few decades have seen a rising interest in using Retarded Functional Differential Equations (RFDEs) for modelling real-life phenomena. In this type of equations, the rate of change of the current state of the process depends not only on the current conditions but also on the previous ones. The dependence on the previous conditions is called the after-effect or lag. RFDEs have many applications in different areas such as biosciences, economics, material science, medicine, control problems and dynamical systems (see, e.g., [1,3,18,28,30]). Delay Differential Equations (DDEs) and Volterra Delay Integro-Differential Equations (VDIDEs) are special classes of RFDEs for which the analytical solution is too complex and for qualitative results one turns to reliable numerical methods. The qualitative and quantitative analyses of numerical methods for DDEs are now quite well understood, as reflected in [7], and many DDEs codes, based on different numerical methods, are also available such as DDE23 [29], Archi [25] and RADAR5 [15]. This is in remarkable contrast to the situation in the numerical analysis of VDIDEs, where there is a limited number of codes available for VDIDEs. The monograph by Brunner [8] introduces the basic principles and analysis of collocation methods for different types of Volterra integral, Volterra integro-differential, and Volterra delay integro-differential equations. In this paper we introduce

¹*Department of Mathematical Sciences, College of Science, United Arab Emirates University, Al-Ain, 17551, UAE. Permanent address: Department of Mathematics, Helwan University, Cairo, Egypt. E-mail: frihan@uaeu.ac.ae*

²*Department of Mathematics, Faculty of Science, Cairo University, Giza, Egypt. E-mail: eid-doha@frcu.eun.eg*

³*Department of Mathematics, Faculty of Engineering, Ain Shams University, Cairo, Egypt. E-mail: noha.medhat@mathasu.edu.eg*

a particular collocation scheme using *continuous mono-implicit Runge — Kutta* methods for different types of VDIDEs. In addition, we investigate the asymptotic stability of the numerical scheme.

The mono-implicit Runge — Kutta (MIRK) method represents a subclass of the well-known family of implicit Runge — Kutta methods [9] and has many applications in the efficient numerical solution of systems of initial and boundary value ordinary differential equations. These methods are suitable for stiff problems (where the global accuracy of the numerical solution is determined by the stability rather than by the local error and implicit methods are more appropriate for it (see, for example, [11])). In a previous work (see [12]), the authors adapted MIRK methods for stiff and nonstiff DDEs and studied the efficiency and stability properties of this class of methods. In this paper, we extend the technique described in [12] to solve Volterra delay integro-differential equations of the form

$$\begin{aligned} y'(t) &= f\left(t, y(t), y(\alpha(t, y(t))), \int_{a(t)}^t g(t, s, y(s)) ds\right) \quad \text{for } t \geq 0, \\ y(t) &= \phi(t), \quad t \leq 0, \end{aligned} \quad (1.1)$$

where $\alpha(t, y(t)) < t$. The integral part in (1.1) introduces the *continuously distributed delay*. If $a(t) = t - \tau$, with fixed $\tau > 0$, then (1.1) has a fixed time-lag and a bounded retardation since the difference between $t - \tau$ and t is fixed and bounded. If $a(t) = 0$, then (1.1) has an unbounded time-lag since the difference between 0 and t is unbounded, and if $a(t) \rightarrow -\infty$, then (1.1) has an infinite time-lag (see [4, 8]).

The absence of the integral term in Eq. (1.1) reduces it to the DDE

$$\begin{aligned} y'(t) &= f(t, y(t), y(\alpha(t, y(t)))) \quad \text{for } t \geq 0, \\ y(t) &= \phi(t), \quad t \leq 0. \end{aligned} \quad (1.2)$$

Here, if $\alpha(t, y(t)) \equiv t - \tau$, with fixed $\tau > 0$, then Eq. (1.2) has a *discretely distributed delay* and the time-lag or retardation is bounded while if $\alpha(t, y(t)) = \mu t$, $\mu \in [0, 1)$, then Eq. (1.2) has an unbounded time-lag.

For simplicity, consider a scalar VDIDE of the form

$$\begin{aligned} y'(t) &= f\left(t, y(t), y(t - \tau), \int_{a(t)}^t g(t, s, y(s)) ds\right) \quad \text{for } t \geq 0, \\ y(t) &= \phi(t), \quad t \leq 0, \end{aligned} \quad (1.3)$$

with fixed $\tau > 0$ and the integral term in (1.3) introduces either an unbounded time-lag *i.e.* $a(t) = 0$ or a bounded time-lag, *i.e.*, $a(t) = t - \tau$. The functions f, g are assumed to be sufficiently smooth with respect to their arguments, and $\phi(t)$ is an initial function which is assumed to be continuous. To solve (1.3) numerically, we have to not only approximate the solution at the proposed meshpoints but also create a solution at the non-mesh points to produce a dense output of the solution in order to approximate the delay term $y(t - \tau)$ and the integral term $\int_{a(t)}^t g(t, s, y(s)) ds$. The proposed technique is based on the continuous mono-implicit Runge — Kutta (CMIRK) method (see [12, 22]) to treat the differential part and Boole's quadrature rule (see [14, 21]) to treat the integral part.

This paper is organized as follows: in section 2, we briefly discuss the analytical stability of VDIDEs, and review MIRK methods for initial value problems in section 3. In section 4, we present the numerical technique, and the numerical algorithm for VDIDEs that are based on CMIRK methods of fourth order along with Boole's quadrature rule. Numerical stability regions of the proposed methods are presented in section 5. Numerical results are presented in section 6, and some concluding remarks are given in section 7.

2. The analytical stability

Consider the test problem of the scalar VDIDE of the form

$$y'(t) = \lambda y(t) + \mu y(t - \tau) + \gamma \int_{t-\tau}^t y(\nu) d\nu, \quad t > 0,$$

$$y(t) = \phi(t), \quad -\tau \leq t \leq 0, \quad (2.1)$$

where λ, μ and γ are complex and $\tau > 0$ is a constant delay. Looking for solutions of (2.1) of the form $y(t) = \exp(st)$ we are led to the characteristic equation

$$H(s) \equiv s - \lambda - \mu \exp(-\tau s) - \frac{\gamma}{s}(1 - \exp(-\tau s)) = 0. \quad (2.2)$$

The asymptotic stability of the zero solution to (2.1) is equivalent to the condition that all roots of the characteristic equation (2.2) have negative real parts and bounded away from the imaginary axis [5, 16].

The analytical stability region S_a of (2.1) is determined by the set of (λ, μ, γ) values for which all roots, s , of (2.2) have negative real parts

$$S_a = \{(\lambda, \mu, \gamma) : \operatorname{Re}(s) < 0 \quad \forall s\}.$$

To find the boundary of the analytical stability region for which $\operatorname{Re}(s) = 0$, we assume that $\lambda, \beta, \gamma \in \mathbb{R}$ and $s \in \mathbb{C}$ is pure complex, i.e., $s = i\theta$, then equation (2.2) takes the form

$$H(i\theta) = i\theta - \lambda - \mu \exp(-i\theta\tau) - \frac{\gamma}{i\theta}(1 - \exp(-i\theta\tau)) = 0. \quad (2.3)$$

Separating the real and complex parts of (2.3) and writing μ and γ in terms of λ and θ , we get

$$C(\lambda, \theta) = \left\{ \begin{array}{l} \mu(\lambda, \theta) = \lambda - \frac{\theta \sin \theta \tau}{(1 - \cos \theta \tau)} \\ \gamma(\lambda, \theta) = \frac{\theta(\theta \cos \theta \tau - \lambda \sin \theta \tau)}{(1 - \cos \theta \tau)} \end{array} \right\} \theta \neq 0, \quad (2.4)$$

for any $\tau > 0$ and $\theta \in (0, 2\pi)$. If $\theta = 0$, then relations in (2.4) reduce to the straight line

$$\mu + \tau \gamma = -\lambda. \quad (2.5)$$

Note that when $\gamma = 0$, VDIDE (2.1) coincides with the test DDE

$$y'(t) = \lambda y(t) + \mu y(t - \tau), \quad t > 0,$$

$$y(t) = \phi(t), \quad -\tau \leq t \leq 0, \quad (2.6)$$

and the boundary (2.4) of the analytical stability region reduces to

$$C(\lambda, \theta) = \left\{ \begin{array}{l} \mu(\lambda, \theta) = \lambda - \frac{\theta \sin \theta \tau}{(1 - \cos \theta \tau)} \\ 0 = (\theta \cos \theta \tau - \lambda \sin \theta \tau) \end{array} \right\} \theta \neq 0, \quad (2.7)$$

that can be simplified to

$$C(\theta) = \left\{ \begin{array}{l} \mu(\theta) = -\theta / \sin \theta \tau \\ \lambda(\theta) = \theta \cot \theta \tau, \end{array} \right\} \theta \neq 0. \quad (2.8)$$

This equation gives the same bounding below curve of the analytical stability region for the DDE (2.6). Equation (2.5) for $\gamma = 0$ reduces immediately to the line $\mu = -\lambda$ which is the same boundary line of (2.6) (see Fig. 2.1).

Fig. 2.1. Analytical stability region for Eq. (2.6)

The graphs of Fig. 2.2 show the analytical stability regions of (2.1) determined by the parametrized curves in (2.4) and the line of (2.5). The asymptotic stability region becomes wider as λ becomes more negative. For further discussions, we may refer to Baker and Ford in [5], Huange and Vandewalle in [16] and Wu and Gan in [32].

The necessary conditions imposed on the coefficients to guarantee the asymptotic stability of (2.1) are given by the following theorem.

Theorem 2.1. *The zero solution of (2.1) is delay-independent asymptotically stable if and only if the following three conditions are satisfied:*

$$\lambda + \mu + \gamma \tau \neq 0 \quad \text{for any } \tau \geq 0,$$

$$\operatorname{Re}(\lambda) < 0 \quad \text{and} \quad (\operatorname{Re}(\lambda \bar{\gamma}) + (\operatorname{Im}(\gamma))^2) < 0 \quad \text{or} \quad \gamma = 0$$

$\text{Im}((\lambda + \mu)\bar{\gamma}) = 0$ and $\{|\mu|^2 < (\text{Re}(\lambda))^2 + 2\text{Re}(\gamma) \text{ or } \text{Im}(\lambda) = 0, |\mu|^2 = (\text{Re}(\lambda))^2 + 2\text{Re}(\gamma)\}$.

When $\gamma = 0$, the above conditions are reduced to $|\mu| < -\text{Re}(\lambda)$. However, when λ , μ and γ are all real and $\gamma \neq 0$, these conditions are reduced to $\lambda < 0$, $\gamma < 0$, and $\mu^2 \leq \lambda^2 + 2\gamma$ (see [19, 20, 32] for the proof and more discussions).

Fig. 2.2. Analytical stability region for Eq. (2.1) with $\lambda = -30$ (solid curve) and $\lambda = -40$ (dashed curve) on the left. On the right: a 3D plot of the analytical stability region with $\lambda = -10, -20, -30, -40, -50$

3. MIRK method for initial value problems

In this section, we briefly review the mono-implicit Runge — Kutta (MIRK) method for the initial value problems. It is well known that implicit Runge — Kutta methods have a higher accuracy and higher computational costs compared to the of explicit Runge — Kutta methods. The MIRK method represents a special subclass of implicit Runge — Kutta methods but is designed to reduce the computational cost of the fully implicit method (see [9, 22, 31]).

The MIRK method can be applied to the initial value ODEs (see [31])

$$y'(t) = f(t, y(t)), \quad 0 < t \leq T,$$

$$y(0) = y_0. \quad (3.1)$$

On a defined mesh $\Delta = \{0 < t_1 < \dots < t_N = T\}$ with a stepsize $h_n = t_{n+1} - t_n$, we have

$$y_{n+1} = y_n + h_n \sum_{r=1}^s b_r K_{n+1}^r, \quad (3.2)$$

where $y_n = y(nh_n)$ and the stages are defined by

$$K_{n+1}^r = f\left(t_n + c_r h_n, (1 - v_r) y_n + v_r y_{n+1} + h_n \sum_{j=1}^{r-1} x_{rj} K_{n+1}^j\right). \quad (3.3)$$

Here K_n^r is the r th stage, b_r, v_r, c_r and x_{rj} are the parameters of the MIRK method represented by the following tableau (with $1 \leq r, j \leq s$):

$$\begin{array}{c|c|cccccc}
 c_1 & v_1 & 0 & 0 & \dots & 0 & 0 \\
 c_2 & v_2 & x_{2,1} & 0 & \dots & 0 & 0 \\
 \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
 c_s & v_s & x_{s,1} & x_{s,2} & \dots & x_{s,s-1} & 0 \\
 \hline
 & & b_1 & b_2 & b_3 & \dots & b_s
 \end{array} \equiv \begin{array}{c|c|c}
 \mathbf{C} & \mathbf{V} & \mathbf{X} \\
 \hline
 & & \mathbf{b}^\top
 \end{array}.$$

The values c_i (usually $\in [0, 1]$), for $i = 1, 2, \dots, s$, are called the *abscissae*. $\mathbf{b}^\top = [b_1, b_2, \dots, b_s]$, $\sum_{i=1}^s b_i = 1$, $\mathbf{V} = [v_1, v_2, \dots, v_s]^\top$, $\mathbf{C} = [c_1, c_2, \dots, c_s]^\top$ and $\mathbf{X} = \{x_{ij}\}_{i,j=1}^{s,s-1}$. The *abscissae* c_r relate b_r and v_r by the form $c_r = v_r + \sum_{i=1}^s x_{ri}$.

The stability function, $R(z)$, of the MIRK satisfies (see [22, 31])

$$R(z) = \frac{P(z, \mathbf{e} - \mathbf{V})}{P(z, -\mathbf{V})}, \quad P(z, \mathbf{w}) = 1 + z\mathbf{b}^\top (\mathbf{I} - z\mathbf{X})^{-1}\mathbf{w},$$

with $\mathbf{w} \in \mathbb{R}^s$ and $\mathbf{e} = [1, 1, \dots, 1]^\top$ is $s \times 1$ vector. The MIRK scheme is of order p (i.e., it has a local error of order $p + 1$) if for the local problem, $y'(t_i) = f(t_i, y(t_i))$, $y(t_{i-1}) = y_{i-1}$, the numerical solution, y_i , given by solving (3.2) and (3.3) satisfies $|y(t_i) - y_i| = O(h_i^{p+1})$.

In [12], we adapted the above technique of the MIRK method to solve the DDE of the form

$$\begin{aligned}
 y'(t) &= f(t, y(t), y(t - \tau)) \quad \text{for } t \geq 0, \\
 y(t) &= \phi(t), \quad -\tau \leq t \leq 0,
 \end{aligned} \tag{3.4}$$

where $\tau > 0$ is the delay in time. The function f is assumed to be sufficiently smooth with respect to its arguments, and $\phi(t)$ is an initial function which is assumed to be continuous. Applying the s -stages of the MIRK scheme to (3.4) results in

$$y_{n+1} = y_n + h_n \sum_{r=1}^s b_r K_{n+1}^r, \tag{3.5}$$

where the stages are defined by

$$K_{n+1}^r = f\left(t_n + c_r h_n, (1 - v_r)y_n + v_r y_{n+1} + h_n \sum_{j=1}^{r-1} x_{rj} K_{n+1}^j, y(t_n + c_r h_n - \tau)\right). \tag{3.6}$$

The complexity of this scheme depends on the number of stages s , the coefficients $\{v_r\}_{r=1}^s$, $\{x_{ij}\}_{i,j=1}^{s,i-1}$ and the weights $\{b_r\}_{r=1}^s$. Due to the nature of DDEs, the numerical solution obtained by the discrete MIRK scheme at mesh points is not as sufficient as in the earlier described ODE case. The presence of the delay term $y(t_n + c_r h_n - \tau)$ in Eq. (3.6) requires the availability of an approximation to the solution at any point $t_n + c_r h_n - \tau$ which is not always a mesh point. The continuous mono-implicit Runge — Kutta (CMIRK) method provides a continuous approximate solution $u(t)$ at any point between the two mesh points t_n and t_{n+1} by the equation

$$u(t_n + \theta h_n) = y_n + h_n \sum_{r=1}^{s^*} b_r(\theta) K_{n+1}^r, \quad 0 \leq \theta \leq 1, \tag{3.7}$$

where $s^* \geq s$ is the number of stages for the CMIRK scheme, $K_{n+1}^r, r = 1, \dots, s^*$ are defined by the same equation (3.6) as in the discrete scheme to avoid extra computations.

$b_r(\theta), r = 1, \dots, s^*$, are polynomials in θ , $0 \leq \theta \leq 1$, of degree $d \leq p$ where p is the order of accuracy of the Runge — Kutta method [23]. The additional $s^* - s$ stages in the continuous extension formula (3.7) are assumed to raise its uniform order to the order of the discrete Runge — Kutta methods.

The parameters of the CMIRK scheme are represented by a similar tableau

$$\begin{array}{c|c|c|c|c|c|c}
 c_1 & v_1 & 0 & 0 & \dots & 0 & 0 \\
 c_2 & v_2 & x_{2,1} & 0 & \dots & 0 & 0 \\
 \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
 c_{s^*} & v_{s^*} & x_{s^*,1} & x_{s^*,2} & \dots & x_{s^*,s^*-1} & 0 \\
 \hline
 & & b_1(\theta) & b_2(\theta) & b_3(\theta) & \dots & b_{s^*}(\theta)
 \end{array} \equiv \begin{array}{c|c|c}
 \mathbf{C} & \mathbf{V} & \mathbf{X} \\
 \hline
 & & \mathbf{b}^\top(\theta)
 \end{array}.$$

The polynomials $b_r(\theta)$ and s^* are chosen to satisfy some continuity and order conditions (see [10, 13, 23] for more detail). The stability of the MIRK method, of different orders, for DDEs has been derived as a part of the work done in [12] for which we refer the reader for more detail. We extend this technique for VDIDEs.

4. Numerical treatment of VDIDEs

The adaptation of ODE numerical methods to solve VDIDEs must take into account their special features. The two main special features of VDIDEs, which coincide with those of DDEs under the assumption that the integrated function is smooth [2], are the propagation of derivative jump discontinuities and the dependence of the solution on previous conditions.

The propagation of derivative jump discontinuities feature means that a jump in the first derivative of the solution at $t = 0$ can cause a jump in the second derivative at $t = \tau$ and a jump in the third derivative at $t = 2\tau$ and so on till the n th derivative which will have a jump at $t = (n - 1)\tau$ (see [6, 27, 28]). The dependence of the solution on the previous conditions, through the delay term $y(t - \tau)$ and the integral term $\int_{a(t)}^t g(t, s, y(s))ds$, requires the availability of a dense output solution [8, 24].

4.1. MIRK scheme for VDIDEs. The mono-implicit Runge — Kutta scheme defined by Eqs. (3.5) and (3.6) is extended to solve VDIDE (1.3) on the defined mesh $\Delta = \{0 < t_1 < \dots < t_n < \dots < t_i = T\}$ as follows:

$$y_{n+1} = y_n + h_n \sum_{r=1}^s b_r K_{n+1}^r, \quad (4.1)$$

where the stages are defined by

$$K_{n+1}^r = f\left(t_n + c_r h_n, (1 - v_r)y_n + v_r y_{n+1} + h_n \sum_{j=1}^{r-1} x_{rj} K_{n+1}^j, y(t_n + c_r h_n - \tau), I_n\right). \quad (4.2)$$

Here $I_n = \int_{a(t_n + c_r h_n)}^{t_n + c_r h_n} g(t, s, y(s))ds$ is evaluated numerically by Boole's quadrature rule (see below Section 4.2).

The fourth order ($s = 5$) discrete MIRK scheme is defined by the tableau

0	0	0	0	0	0	0
1	1	0	0	0	0	0
$\frac{1}{20}$	$\frac{29}{4000}$	$\frac{361}{8000}$	$-\frac{19}{8000}$	0	0	0
$\frac{19}{20}$	$\frac{3971}{4000}$	$\frac{19}{8000}$	$-\frac{361}{8000}$	0	0	0
$\frac{1}{2}$	$\frac{11}{16}$	$\frac{1}{32}$	$\frac{267}{608}$	$\frac{25}{684}$	$-\frac{25}{36}$	0
		$-\frac{43}{228}$	$-\frac{43}{228}$	$\frac{25}{57}$	$\frac{25}{57}$	$\frac{1}{2}$

The discrete solution at the mesh points obtained using the discrete MIRK scheme is not enough to solve the VDIDEs. A dense output solution is needed at non-mesh points as explained earlier, using Eq. (3.7) of the fourth-order CMIRK scheme (see [12]) (with $s^* = 5$), such that

0	0	0	0	0	0	0
1	1	0	0	0	0	0
$\frac{1}{20}$	$\frac{29}{4000}$	$\frac{361}{8000}$	$-\frac{19}{8000}$	0	0	0
$\frac{19}{20}$	$\frac{3971}{4000}$	$\frac{19}{8000}$	$-\frac{361}{8000}$	0	0	0
$\frac{1}{2}$	$\frac{11}{16}$	$\frac{1}{32}$	$\frac{267}{608}$	$\frac{25}{684}$	$-\frac{25}{36}$	0
		$b_1(\theta)$	$b_2(\theta)$	$b_3(\theta)$	$b_4(\theta)$	$b_5(\theta)$

where

$$\begin{aligned}
 b_1(\theta) &= -\frac{1}{228}\theta(1200\theta^3 - 2714\theta^2 + 1785\theta - 228), & b_2(\theta) &= \frac{1}{228}\theta^2(1200\theta^2 - 2086\theta + 843), \\
 b_3(\theta) &= \frac{25}{171}\theta^2(40\theta^2 - 86\theta + 49), & b_4(\theta) &= -\frac{25}{171}\theta^2(40\theta^2 - 74\theta + 31), & b_5(\theta) &= -\frac{1}{2}\theta^2(2\theta - 3).
 \end{aligned}
 \tag{4.3}$$

4.2. Numerical integration formula and Boole's quadrature rule To find an approximation to the definite integral

$$\int_a^b z(x) dx \tag{4.4}$$

with known tabulated data of $z(x)$ at some mesh points $a = x_1 < x_2 < \dots < x_N = b$, a *quadrature rule* is needed. A quadrature rule approximates the integral by

$$\int_a^b z(x) dx \simeq \sum_{k=1}^N w_k f(x_k) \tag{4.5}$$

where w_k , $k = 1, 2, \dots, N$, and x_k , $k = 1, 2, \dots, N$, are called *weights* and *nodes* respectively. Of course, quadrature rules differ in the choice of *weights* and *nodes*.

Newton — Cotes quadrature (NCQ) rules are very popular numerical integration techniques. In NCQ rules the nodes are equally spaced. The N –node NCQ technique finds a Lagrange polynomial P_N that passes through N nodes and then the integral of $z(x)$ is

approximated by the integral of P_N . NCQ are classified into closed NCQ, if x_1, x_N are used, and open NCQ, if x_1, x_N are not used [21].

Many well-known quadrature rules are NCQ rules such as the trapezoidal rule which is a 2-node closed NCQ, Simpson's 1/3 rule which is a 3-node closed NCQ, Simpson's 3/8 rule which is a 4-node closed NCQ, and Boole's rule which is a 5-node closed NCQ.

Boole's rule is a 5-node closed NCQ which approximates the integral by the following formula:

$$\int_{x_1}^{x_5} z(s)ds \simeq \frac{2}{45}h(7z_1 + 32z_2 + 12z_3 + 32z_4 + 7z_5), \quad (4.6)$$

where $h = x_i - x_{i-1}$, $i = 2, \dots, 5$ and with an error of order h^7 . If the number of points used in the evaluation of the integral is greater than 5, then Boole's rule is applied repeatedly which results in the composite Boole's rule. For example, if we have 9-points, then the composite Boole's rule takes the form

$$\int_{x_1}^{x_9} z(s)ds \simeq \frac{2}{45}h(7z_1 + 32z_2 + 12z_3 + 32z_4 + 14z_5 + 32z_6 + 12z_7 + 32z_8 + 7z_9). \quad (4.7)$$

To find an approximation to the integral I_n in (4.2), we subdivide its integration interval as follows: if $a(t) = t - \tau$,

$$I_n = \int_{t_n - \tau}^{t_n} g(t, s, y(s))ds + \int_{t_n}^{t_n + c_r h_n} g(t, s, y(s))ds - \int_{t_n - \tau}^{t_n - \tau + c_r h_n} g(t, s, y(s))ds;$$

when $a(t) = 0$,

$$I_n = \int_0^{t_n} g(t, s, y(s))ds + \int_{t_n}^{t_n + c_r h_n} g(t, s, y(s))ds.$$

Each of the above integrals is approximated by applying Boole's rule(4.6). For previous intervals, the CMIRK scheme provides the solution at any non-mesh points needed by Boole's rule. But for a current mesh, cubic spline interpolation is used to get the solution at any non-mesh point needed by Boole's rule.

4.3. MIDDE software aspects. The numerical technique proposed in this work is implemented in a new Matlab code, MIDDE which is based on the CMIRK method to treat the differential part and Boole's quadrature rule to treat the integral part. The code is designed to solve systems of both DDEs and VDIDEs. Of course, other professional codes such as RADAR5 [15] and Archi [25] are based on different numerical techniques.

The MIDDE code has the following aspects:

- The points of jump discontinuities are located and taken as mesh points.
- The dense output solution is obtained inside the MIDDE code using the continuous mono-implicit Runge — Kutta method.
- A step-size control strategy is implemented inside the code to maintain the defect in the numerical solution within a specified tolerance.

In the following, we briefly introduce the numerical algorithm and stepsize control in the MIDDE code.

Numerical algorithm and step-size control. The general steps of the numerical algorithm implementing the MIDDE code for solving VDIDEs (1.3) can be briefly summarized as follows.

1. Derivative jump discontinuity points are located and taken as mesh points. For each interval $[t_n, t_{n+1}]$ the next steps are repeated.

2. The interval is divided into equally spaced points $t_n = t_1, t_2, \dots, t_m, t_{m+1} = t_{n+1}$ with stepsize $h_n = t_{n+1} - t_n$ where $\tau = mh_n$. Delay terms $y(t_n + c_r h_n - \tau)$ are calculated by $\phi(t)$ in (1.3) (in the first interval), or by the CMIRK formula (3.7).

3. Approximations to the solution $\mathbf{Y} = \{y_i\}_{i=1}^{m+1}$ and the continuous record of the previous mesh are used by Boole's rule to evaluate I_n (see section 4.2). Then the delay terms $y(t_n + c_r h_n - \tau)$, and the integral terms I_n and $\mathbf{Y}$ are used to calculate the stages $\{K_{n+1}^r\}_{r=1}^s$ by (4.2).

4. The residual function $\Psi(\mathbf{Y})$ whose n th component $\Psi_n = y_{n+1} - y_n - h_n \sum_{r=1}^s b_r K_{n+1}^r$ is evaluated. Then the system of equations $\Psi(\mathbf{Y}) = 0$ is solved using Newton's iteration.

5. Newton's iteration is terminated when the numerical solution $\mathbf{Y}$ causes $\Psi(\mathbf{Y})$ to be within the specified tolerance otherwise $\mathbf{Y}$ is updated and steps 3 – 5 are repeated.

6. The CMIRK scheme is then used to obtain the continuous solution $u(t)$ (3.7). Then an estimate for the defect [13] is calculated by

$$\text{Defect}_i = \frac{|u'(t_i) - f(t_i, u(t_i), u(t_i - \tau), I_i)|}{1 + |f(t_i, u(t_i), u(t_i - \tau), I_i)|}. \quad (4.8)$$

7. An adaptive algorithm is used to control the stepsize in order to keep the defect (4.8) within a specified tolerance. If the defect is not within the tolerance, then the step size is halved and iteration steps 2 – 7 are repeated.

5. The numerical stability

In this section, we investigate the numerical stability for the test problem VDIDE (2.1). For simplicity, assume that $\tau = mh$ ($m \in \mathbb{I}$ and we do not need internal stages to estimate $y(t - \tau)$ (that is, $y(t - \tau) = y_{n-m}$), then we have

$$y_{n+1} = y_n + h \sum_{r=1}^s b_r K_{n+1}^r, \quad (5.1)$$

where

$$K_{n+1}^r = \lambda \left((1 - v_r) y_n + v_r y_{n+1} + h \sum_{j=1}^{r-1} x_{rj} K_{n+1}^j \right) + \mu y_{n-m} + \gamma h \sum_{l=0}^m w_l y_{n-l}. \quad (5.2)$$

Substituting y_{n+1} from (5.1) into (5.2) yields

$$y_{n+1} = (1 + \alpha \mathbf{b}^\top \mathbf{S} \mathbf{e}) y_n + \beta \mathbf{b}^\top \mathbf{S} \mathbf{e} y_{n-m} + G \mathbf{b}^\top \mathbf{S} \mathbf{e} \sum_{l=0}^m w_l y_{n-l}, \quad (5.3)$$

where $\alpha = h\lambda$, $\beta = h\mu$, $G = h^2\gamma$, $\mathbf{S} = [\mathbf{I} - \alpha(\mathbf{V}\mathbf{b}^\top + \mathbf{X})]^{-1}$ and $\mathbf{e} = (1, 1, \dots, 1)^\top$ is $s \times 1$ vector.

Letting $\xi^{-1} = y_n$, we get the characteristic equation

$$1 - (1 + \alpha \mathbf{b}^\top \mathbf{S} \mathbf{e}) \xi^{-1} - \beta \mathbf{b}^\top \mathbf{S} \mathbf{e} \xi^{-1-m} - G \mathbf{b}^\top \mathbf{S} \mathbf{e} \sum_{l=0}^m w_l \xi^{-1-l} = 0. \quad (5.4)$$

If $\xi^{-1} = \exp(i\theta)$, then (5.4) can be separated into real and imaginary parts of the form

$$\begin{aligned} 1 - \cos \theta &= \mathbf{b}^\top \mathbf{Se} \left(\alpha \cos \theta + \beta \cos(m+1)\theta + G \sum_{l=0}^m w_l \cos(l+1)\theta \right), \\ -\sin \theta &= \mathbf{b}^\top \mathbf{Se} \left(\alpha \sin \theta + \beta \sin(m+1)\theta + G \sum_{l=0}^m w_l \sin(l+1)\theta \right). \end{aligned} \quad (5.5)$$

For $\theta = 0$, these equations reduce to

$$\alpha + \beta + G \sum_{l=0}^m w_l = 0 \equiv \lambda + \mu + \gamma\tau = 0, \quad (5.6)$$

assuming that Boole's quadrature rule is exact for constants, i.e., $\int_{t-\tau}^t ds = h \sum_{l=0}^m w_l = \tau$. This is the same straight line that bounds the analytical stability region obtained earlier in Section 2.

For $\theta \neq 0$, the boundary of the stability region is defined by (5.5). To plot this curve, we write β , G in terms of α , and after some manipulations we have

$$\begin{aligned} G &= \left(1 / \sum_{l=0}^m w_l \sin(m-l)\theta \right) * \left\{ -\alpha \sin m\theta + \frac{\sin(m+1)\theta - \sin m\theta}{\mathbf{b}^\top \mathbf{Se}} \right\}, \\ \beta &= -(1/(\sin(m+1)\theta - \sin m\theta)) * \left\{ \alpha \sin \theta + G \sum_{l=0}^m w_l (\sin(l+1)\theta + \sin l\theta) \right\}. \end{aligned} \quad (5.7)$$

Since G and β are even functions of θ , we consider Eqs. (5.7) for $\theta \in (0, 2\pi)$. Figure 5.1 shows the numerical stability regions for three values of $m = 8, 64, 128$ and fixed $\lambda = -20$. It also shows a comparison of the numerical stability region with the analytical stability region for fixed $\lambda = -20$ and $m = 128$. The numerical stability region becomes bigger as the stepsize decreases.

Fig. 5.1. Numerical stability regions (when $\tau/h = m$) for different values of $m = 8; 64; 128$ and fixed $\lambda = -20$ (on the left). The region becomes wider as m increases. Numerical stability region (solid line) for fixed $\lambda = -20$ and $m = 128$ compared with the boundary of the analytical stability region (dashed line) for the same value of λ (on the right).

6. Numerical Results

In this section, we report the numerical results for various types of Volterra delay integro-differential equations ([26, 33, 34]), using the proposed technique.

Example 6.1. Consider first the scalar VDIDE of the form

$$y'(t) = \exp(-2)y(t) + 2 \int_{t-1}^t \exp(s-t)y(s)ds, \quad t \geq 0,$$

$$\phi(t) = \exp(t), \quad t \leq 0, \quad (6.1)$$

for which only a continuously distributed delay with bounded time-lag is presented. The exact solution of (6.1) is $\exp(t)$. The numerical solution is obtained by the MIDDE with defect (4.8) bounded within tolerance 10^{-10} . Figure 6.1 displays the numerical solution of (6.1) and the error function, $\|y(t_n) - y_n\|$. The error is of order 10^{-5} at $t_f = 10$.

Fig. 6.1. Numerical solution of (6.1) (on the left) and error function (on the right) at the mesh points

Example 6.2. In this example, we consider the VDIDE

$$y'(t) = y(t-1) + \int_{t-1}^t y(s)ds, \quad t \geq 0,$$

$$\phi(t) = \exp(t), \quad t \leq 0, \quad (6.2)$$

with both discretely and continuously distributed delays. The exact solution of (6.2) is also $\exp(t)$. With tolerance 10^{-10} , the obtained numerical solution, using the proposed method, and the error function are given in Fig. 6.2. The error is of order 10^{-5} at $t_f = 10$.

Fig. 6.2. Numerical solution for (6.2) (on the left) and error function (on the right) at the mesh points

Example 6.3. Consider the VDIDE of partially variable coefficients

$$y'(t) = -(6 + \sin t)y(t) + y(t - \pi/4) - \int_{t-\pi/4}^t \sin s y(s) ds + 5 \exp(\cos t), \quad t \geq 0,$$

$$\phi(t) = \exp(\cos t), \quad t \leq 0. \quad (6.3)$$

The exact solution of (6.3) is $\exp(\cos t)$. The graphs in Fig. 6.3 show the obtained numerical solution with tolerance 10^{-10} , the error in the numerical solution decaying to zero, and the max error of order 10^{-10} .

Fig. 6.3. The numerical solution for (6.3) (on the left) and the error function (on the right) at the mesh points

Example 6.4. Consider the nonlinear system of VDIDEs:

$$\frac{d}{dt} \begin{pmatrix} y_1(t) \\ y_2(t) \end{pmatrix} = -3 \begin{pmatrix} y_1(t) \\ y_2(t) \end{pmatrix} + \begin{pmatrix} 0 & \sin t \\ \cos t & 0 \end{pmatrix} \begin{pmatrix} y_1(t - \pi/4) \\ y_2(t - \pi/4) \end{pmatrix} +$$

$$\begin{aligned}
 & \frac{1}{\sqrt{2}} \int_{t-\pi/4}^t \left(\frac{(1 + \sin^2 t)y_1^2(s)}{1 + y_1^2(s)} \right. \\
 & \quad \left. \frac{(1 + \cos^2 t)y_2^2(s)}{1 + y_2^2(s)} \right) ds + \\
 & \frac{\sqrt{2}}{16} \begin{pmatrix} 16 \sin(t + \pi/4) - 2 \sin 2t + 6 \cos 2t - \pi - 4 + 16\sqrt{2} \sin t \\ 16 \cos(t + \pi/4) - 2 \sin 2t + 6 \cos 2t + \pi + 4 + 16\sqrt{2} \cos t \end{pmatrix}, \quad t \geq 0, \\
 & \phi(t) = \begin{pmatrix} \sin t \\ \cos t \end{pmatrix}, \quad t \leq 0,
 \end{aligned} \tag{6.4}$$

with both discretely and continuously distributed delays and a bounded time-lag. This system has exact solutions $(\sin t, \cos t)$. Figure 6.4 shows the numerical solution of (6.4) with tolerance 10^{-10} and the error components e_1 and e_2 .

Fig. 6.4. The two components of the numerical solutions for (6.4) on the interval $[0, 2\pi]$ (on the left) and the error function (on the right) at the mesh points

Example 6.5. Consider the Lotka — Volterra system of the form

$$\begin{aligned}
 u'(t) &= u(t) \left(1 - 0.5 v(t) - \int_0^t \exp((1.5)(s - t)) v(s) ds \right), \quad t \geq 0, \\
 v'(t) &= -v(t) \left(0.75 - 0.25 u(t) - \int_0^t \exp((3.5)(s - t)) u(s) ds \right), \quad t \geq 0, \\
 u(0) &= v(0) = 1.
 \end{aligned} \tag{6.5}$$

The Lotka — Volterra system (6.5) is of great importance in population dynamics problems. This system is a mathematical model of the prey-predator population dynamics in which $u(t)$ represents the prey population size and $v(t)$ represents the predator population size. This system has a continuously distributed delay and the time-lag is unbounded. Figure 6.5 shows the numerical solution for (6.5) obtained by the MIDDE with tolerance 10^{-4} .

Now, if we change the lower bound of the integration from $0 \rightarrow (t - \tau)$ where $\tau > 0$ is fixed, then equations (6.5) will take the form

$$\begin{aligned} u'(t) &= u(t) \left(1 - 0.5 v(t) - \int_{t-\tau}^t \exp((1.5)(s-t)) v(s) ds \right), \quad t \geq 0, \\ v'(t) &= -v(t) \left(0.75 - 0.25 u(t) - \int_{t-\tau}^t \exp((3.5)(s-t)) u(s) ds \right), \quad t \geq 0, \\ u(t) &= v(t) = 1, \quad t \leq 0. \end{aligned} \quad (6.6)$$

The system in (6.6) is now with a bounded time-lag. The oscillatory behaviour of the numerical solution of the Lotka — Volterra system (6.6) is shown in Fig. 6.6 when $\tau = 1$ and defect (4.8) is bounded with the same tolerance 10^{-4} .

Fig. 6.5. The numerical solution for the Lotka — Volterra system (6.5) using the MIDDE code

Fig. 6.6. The numerical solution for the Lotka — Volterra system (6.6), with $\tau = 1$, using the MIDDE code

7. Conclusions

This paper presented a new technique for the solution of the Volterra delay integro-differential equation. The technique is based on the continuous mono-implicit Runge — Kutta method for the differential part and Boole's quadrature rule for the integral part. The analytical and numerical stability regions have been deduced. The methodology has been tested by different types of VDIDEs. The proposed method is suitable for stiff and nonstiff problems. The Matlab MIDDE code is available for noncommercial proposes.

References

1. C. T. H. Baker, *A perspective on the numerical treatment of Volterra equations*, J. Comput. Appl. Math., **125** (2000) pp. 217–249.
2. C. T. H. Baker, *Numerical Analysis of Volterra Functional and Integral Equations- State of the Art*, Numerical Analysis Report, **292**, 1996, Manchester Center for Comput. Mathematics, Univ. of Manchester, U. K.

3. C. T. H. Baker, *Retarded differential equations*, J. Comput. Appl. Math., **125** (2000) pp. 309–335.
4. C. T. H. Baker, G. A. Bocharov, A. Filiz, N. J. Ford, C. A. H. Paul, F. A. Rihan, A. Tang, R. M. Thomas, H. Tian, and D. R. Willé, *Numerical modelling by delay and Volterra functional differential equations*, in E. A. Lipitakis, ed., *Topics in Computer Mathematics and its Applications*, LEA- Athen, Hellas, 2006, pp. 233–256.
5. C. T. H. Baker and N. J. Ford, *Stability properties of a scheme for the approximate solution of delay-integro-differential equation*, Appl. Numer. Math., **9** (1992), pp. 357–370.
6. C. T. H. Baker and C. A. H. Paul, *Issues in the numerical solution of evolutionary delay differential equations*, Adv. Comput. Math., **3** (1995), pp. 171–196.
7. A. Bellen and M. Zennaro, *Numerical Methods for Delay Differential Equations*, Oxford University Press, Oxford, 2003.
8. H. Brunner, *Collocation Methods for Volterra Integral and Related Functional Differential Equations*, Cambridge Monographs on Applied and Computational Mathematics, Vol. 15, Cambridge University Press, 2004.
9. J. C. Butcher, *Implicit Runge — Kutta processes*, Math. Comput., **18** (1964), pp. 50–64.
10. J. C. Butcher and P. Chartier, *A generalization of singly-implicit Runge — Kutta methods*, Appl. Numer. Math., **24** (1997), pp. 343–350.
11. J. R. Cash and A. Singhal, *Mono-implicit Runge — Kutta formula for the numerical integration of stiff differential systems*, IMA J. Numer. Anal., **2** (1982), pp. 211–227.
12. E. H. Doha, F. A. Rihan, M. I. Hassan, and N. M. Kamel, *Mono-implicit Runge — Kutta method for delay differential equations*, J. Egypt. Math. Soc. (to appear).
13. W. E. Enright and P. H. Muir, *A Runge — Kutta Boundary Value ODE Solver With Defect Control*, Technical Report **267/93**, June 1993, Department of Computer Science, University of Toronto.
14. G. E. Forsythe, M. A. Malcolm, and C. B. Moler, *Computer Methods for Mathematical Computations*, The NJ: Prentice-Hall Inc., Englewood Cliffs, 1977.
15. N. Guglielmi and E. Hairer, *Implementing Radau II-A methods for stiff delay differential equations*, J. Comput. Math., **67** (2001), pp. 1–12.
16. C. Huang and S. Vandewalle, *Stability of Runge — Kutta-Pouzet Methods for Volterra Integral and Integro-Differential Equations With Delays*, Report **TW 361**, June 2003, Department of Computer Science, Katholieke Universiteit Leuven.
17. K. J. In't Hout, *On the stability of adaptations of Runge — Kutta methods to systems of delay differential equations*, Appl. Numer. Math., **22** (1996), pp. 237–250.
18. V. Kolmanovskii and A. Myshkis, *Applied Theory of Functional Differential Equations*, The Kluwer Academic Publishers, 1992.
19. T. Koto, *Stability of Runge — Kutta methods for delay integro-differential equations*, Comput. Appl. Math., **145** (2002), pp. 483–492.
20. T. Koto, *Stability of θ -methods for delay integro-differential equations*, Comput. Appl. Math., **161** (2003), pp. 393–404.
21. J. H. Mathews and K. K. Fink, *Numerical Methods Using Matlab*, 4th edn., The Prentice-Hall Inc., Upper Saddle River, New Jersey, USA 2004.
22. P. H. Muir, *Optimal discrete and continuous mono-implicit Runge — Kutta schemes for BVODEs*, Adv. Comput. Math., **10** (1999), pp. 135–167.
23. P. H. Muir and B. Owren, *Order Barriers and Characterizations for Continuous Mono-Implicit Runge — Kutta Schemes*, Technical Report, **258/91**, December 1991, Department of Computer Science, University of Toronto.
24. H. J. Oberle and H. J. Pesch, *Numerical treatment of delay differential equations by Hermite interpolation*, J. Numer. Math., **37** (1981), pp. 235–255.
25. C. A. H. Paul, *A User-Guide to Archi- an Explicit Runge — Kutta Code for Solving Delay and Neutral Differential Equations and Parameter Estimation Problems*, Numerical Analysis Report, **283**, 1997, Manchester Center for Comput. Mathematics, Univ. of Manchester, U.K.
26. C. A. H. Paul, *A Test Set of Functional Differential Equations*, Numerical Analysis Report **243**, 1994, Manchester Center for Comput. Mathematics, Univ. of Manchester, U.K.
27. C. A. H. Paul, *The Numerical Treatment of Derivative Discontinuities in Differential Equations*, Numerical Analysis Report **337**, 1999, Manchester Center for Comput. Mathematics, Univ. of Manchester, U.K.
28. F. A. Rihan, *Numerical Treatment of Delay Differential Equations in Bioscience*, Ph.D. thesis, Department of Mathematics, University of Manchester, U.K., 2000.

29. L. F. Shampine and S. Thompson, *Solving DDEs in Matlab*, Appl. Numer. Math., **37** (2001), pp. 441–458.
30. P. L. Simon, *A retarded differential equation model of wave propagation in a thin ring*, SIAM J. Appl. Math., **61** (2001), pp. 1618–1627.
31. D. A. Voss and P. H. Muir, *Mono-implicit Runge — Kutta schemes for the parallel solution of initial value ODEs*, J. Comput. Appl. Math., **102** (1999), pp. 235–252.
32. S. Wu and S. Gan, *Analytical and numerical stability of neutral delay integro-differential equations and neutral delay partial differential equations*, Comput. Math. Appl., **55** (2008), pp. 2426–2443.
33. C. Zhang and S. Vandewalle, *Stability analysis of Volterra delay integro-differential equations and their backward differentiation time discretization*, Comput. Appl. Math., **164/165** (2004), pp. 797–814.
34. C. Zhang and S. Vandewalle, *General linear methods for Volterra integro-differential equations with memory*, SIAM J. Sci. Comput., **27** (2006), pp. 2010–2031.