A compact Fourth-Order Implicit-Explicit Runge-Kutta Type Method for Solving Diffusive Lotka–Volterra System

Younis A. Sabawi 1, Mardan A. Pirdawood 2, Mohammed I. Sadeeq 3

1,2 Department of Mathematics, Faculty of Science and Health, Koya University, Koya KOY45, Kurdistan Region - F.R. Iraq
3 Department of Mathematics, University of Duhok, Kurdistan Region - Iraq.
younis.abid@koyauniversity.org, mardan.ameen@koyauniversity.org, mohammed.sadeeq@uod.ac

Abstract. This paper aims to developed a high-order and accurate method for the solution of one-dimensional Lotka-Volterra-diffusion with Neuman boundary conditions. A fourth-order compact finite difference scheme for spatial part combined with implicit-explicit Runge Kutta scheme in temporal are proposed. Furthermore, boundary points are discretized by using a compact finite difference scheme in terms of fourth order accuracy. A key idea for proposed scheme is to take full advantage of method of line (MOL), this is consequently enabling us to use implicit-explicit Runge Kutta method, that are of fourth order in time. We constructed fourth order accuracy in both space and time and is unconditionally stable. This is consequently leading to a reduction in the computational cost of the scheme. Numerical experiments show that the combination of the compact finite difference with IMEX- RK methods give an accurate and reliable for solving the Lotka-Volterra-diffusion.

Keywords: A compact difference methods, IMEX-RK methods, Lotka–Volterra System, A compact Fourth-Order Implicit-Explicit Runge-Kutta Type Method.

1. Introduction

The finite element (FEM) and compact finite difference (CFDM) methods consider are the most of flexibility common technique used for dealing with partial differential equations. In context of Dirichlet boundary conditions for linear and nonlinear parabolic problems are gaining increasing interest and there is a significant implementation of the method now are understandable and available in the literature [1-9]. However, there is less progress has been made comparatively in the deriving high order in terms of Neumann boundary conditions [10–12]. Cao al el [10] have developed a fourth-order compact finite difference scheme for solving the convection–diffusion equation with Neumann boundary conditions. They used compact finite difference scheme of fourth-order to discrete interior and boundary points. Yao al el [11] derived a fourth-order compact finite difference scheme for solving the model equation of simulated moving bed. They presented two different methods, direct method and pseudo grid point method to address the difficulty of the boundary condition. Fu al el [12] have used a high-order exponential scheme convection-diffusion equation with Neumann boundary conditions. In their analyses, they derived fourth-order compact exponential difference scheme in spatial discretization at all interior and boundary points. The aim of this work is to propose a numerical method for solving (1) and (2) that is fourth-order accurate in both space and time components. The idea is to discetise both spatial derivative and time integration of fourth order accurate by using a compact finite difference approximation and implicit-explicit Runge Kutta method, respectively. This is leading to a nonlinear system of ordinary differential equations. Some numerical methods have been developed to solve the Diffusive Lotka–Volterra System with Neumann boundary conditions, but only first-order or second-order at the
boundary points are considered, see [13, 14]. The main difficulties in our work in constructing fourth-order compact method for boundary point and dealing with the nonlinear reaction term. These challenges are addressed by employing techniques introduced by [12] for boundary point and IMEX-RK, introduced by Ruuth and Spiteri (21) for nonlinear term. It is worth noting the main reason of IMEX-RK technique is to lead us utilise implicit methods for linear part and explicit method for nonlinear part.

The rest of this paper is structured as follows. In Section 2, the model problem is introduced with derived fourth order compact scheme for both interior and boundary points. Section 3, implicit explicit Runge Kutta method are presented. Numerical Experiments are shown in Section 4, Finally, conclusions are given in Section 5.

Consider the one-dimensional Lotka-Volterra equation with Neumann boundary conditions:
\[
\frac{\partial u}{\partial t} = d_1 \frac{\partial^2 u}{\partial x^2} + au(1 - u) - \beta uv, \quad (x, t) \in [a, b] \times [0, T]
\]
\[
\frac{\partial v}{\partial t} = d_2 \frac{\partial^2 v}{\partial x^2} - \gamma v + \varepsilon uv, \quad (x, t) \in [a, b] \times [0, T]
\]
with the following initial conditions:
\[
\begin{align*}
  u(x, 0) &= u_0(x), & 0 \leq x \leq L \\
  v(x, 0) &= v_0(x), & 0 \leq x \leq L
\end{align*}
\]
and Neumann boundary conditions
\[
\begin{align*}
  \frac{\partial u(0, t)}{\partial x} &= 0, & \frac{\partial u(L, t)}{\partial x} &= 0 \\
  \frac{\partial v(0, t)}{\partial x} &= 0, & \frac{\partial v(L, t)}{\partial x} &= 0.
\end{align*}
\]
Here \( u \) represents the number of fish (prey), \( v \) represents the number of sharks (predator), and \( d_1 \) is diffusion coefficient of the prey. We also use the parameter \( d_2 \) is diffusion coefficient of the predator, to scale the predator, \( \alpha \) to represent of the prey, \( \gamma \) to represent the death rate of the predator, and \( \beta, \varepsilon \) they represent the rate of interaction between the prey and the predator which facilitates the killing of prey by predator.

2. Development of fourth-order compact finite differencing schemes

2.1 The interior spatial points

This section aims to discretise on spatial domain, and this is consequently leading to a set of system of nonlinear ordinary differential equations. To do this, we discretise (1) and (2) by a four-order compact difference approximation for the space part setting \( \frac{\partial}{\partial x} = \frac{\partial u}{\partial x} \) and \( m(x) = \frac{\partial v}{\partial x} \) gives

Recalling (1) and (2), gives
\[
\begin{align*}
  d_1 \frac{\partial^2 u}{\partial x^2} + au - au^2 - \beta uv &= l(x), \\
  d_2 \frac{\partial^2 v}{\partial x^2} - \gamma v + \varepsilon uv &= m(x).
\end{align*}
\]

Applying \( \delta_x^2 u_l \) and \( \delta_x^2 v_l \) in (3) and (4) which is a second-order central difference along the x-direction, gives
\[
\begin{align*}
  d_1 \delta_x^2 u_l + au_l - au_l^2 - \beta u_l v_l - \tau^1_l &= l_l, \\
  d_2 \delta_x^2 v_l - \gamma v_{l+1} v_l + \varepsilon u_l v_l - \tau^2_l &= m_l.
\end{align*}
\]
where
\[
\begin{align*}
  \delta_x^2 u_l &= \frac{u_{l+1} - 2u_l + u_{l-1}}{h^2}, & \delta_x^2 v_l &= \frac{v_{l+1} - 2v_l + v_{l-1}}{h^2},
\end{align*}
\]
\( u_i = u(x_i), l_i = l(x_i) \) and \( m_i = m(x_i) \). The truncation error \( \tau_i^1 \) and \( \tau_i^2 \) so that

\[
\tau_i^1 = \frac{d_1 h^2}{12} \left( \frac{d^4 u}{dx^4} \right) + O(h^4), \quad \tau_i^2 = \frac{d_2 h^2}{12} \left( \frac{d^4 v}{dx^4} \right) + O(h^4).
\] (8)

To obtain higher order scheme yield \( O(h^4) \), this may accomplish by differentiating (3) and (4) respect to \( x \), gives

\[
\frac{d^3 u}{dx^3} = -\alpha \frac{du}{dx} + 2\alpha u \frac{du}{dx} + \beta \frac{d^2 v}{dx^2} + \frac{\beta}{d_1} \frac{dv}{dx} + \frac{\gamma}{d_1} \frac{d^2 l}{dx^2} + \frac{2\beta}{d_1} \frac{du}{dx} + \frac{1}{d_1} \frac{dl}{dx},
\] (9)

\[
\frac{d^4 u}{dx^4} = -\alpha + 2\alpha u + \beta \frac{d^2 u}{dx^2} + 2\alpha \left( \frac{du}{dx} \right)^2 + \frac{\beta}{d_1} \frac{d^2 v}{dx^2} + \frac{2\beta}{d_1} \frac{du}{dx} + \frac{1}{d_1} \frac{d^2 l}{dx^2} + \frac{1}{d_1} \frac{d^2 l}{dx^2},
\] (10)

\[
\frac{d^3 v}{dx^3} = \gamma \frac{dv}{dx} + \frac{\epsilon}{d_1} \frac{dv}{dx} + \frac{\epsilon}{d_1} \frac{du}{dx} + \frac{1}{d_1} \frac{dl(x)}{dx},
\] (11)

\[
\frac{d^4 v}{dx^4} = \frac{d_2}{d_1} \frac{dv}{dx} - \frac{d_2}{d_1} \frac{u}{dx} + \frac{d_2}{d_1} \frac{v}{dx} + \frac{1}{d_1} \frac{d^2 l}{dx^2} + \frac{1}{d_1} \frac{d^2 l}{dx^2}.
\] (12)

Replacing (9-12) into (8), this becomes

\[
\tau_i^1 = \frac{h^2}{12} \left( (-\alpha + 2\alpha u_i + \beta v_i) \delta_i^2 u_i + 2\alpha (\delta_x u_i)^2 + \beta u_i \delta_i^2 v_i \right) + O(h^4)
\] (13)

\[
+ \frac{h^2}{12} \left( 2\beta \delta_{x_i} u_i, \delta,v_i + \delta_i^2 l_i \right)
\]

\[
\tau_i^2 = \frac{h^2}{12} \left( \gamma \delta_i^2 v_i - \epsilon u_i \delta_i^2 v_i - 2\epsilon \delta_x u_i, \delta v_i - \epsilon v_i \delta_i^2 u_i \right).
\] (14)

Go back to (5) and (6), and substituting above equations in (5) and (6), reads

\[
d_1 \delta_i^2 u_i + \alpha u_i - \alpha u_i^2 - \beta u_i v_i - \frac{h^2}{12} \left( (-\alpha + 2\alpha u_i + \beta v_i) \delta_i^2 u_i + 2\alpha (\delta_x u_i)^2 + \beta u_i \delta_i^2 v_i + 2\beta \delta_{x_i} u_i \delta v_i \right)
\]

\[
= \frac{h^2}{12} \delta_i^2 l_i^1 + l_i + O(h^4),
\] (15)

\[
d_2 \delta_i^2 v_i - \gamma v_i + \epsilon u_i v_i - \frac{h^2}{12} \left( \gamma \delta_i^2 v_i - \epsilon u_i \delta_i^2 v_i - 2\epsilon \delta_x u_i \delta v_i - \epsilon v_i \delta_i^2 u_i \right)
\]

\[
= \frac{h^2}{12} \delta_i^2 l_i^2 + l_i + O(h^4).
\] (16)

If we put (7) along with \( \delta_x u_i = \frac{u_{i+1} - u_{i-1}}{2h} \), \( \delta_x v_i = \frac{v_{i+1} - v_{i-1}}{2h} \) in (15) and (16), this gives

\[
\frac{du_{i+1}(t)}{dt} + 10 \frac{du_{i}(t)}{dt} + \frac{du_{i-1}(t)}{dt} = f_1 u_i(t) + f_2 u_{i+1}(t) + f_3 u_{i-1}(t),
\] (17)

\[
\frac{dv_{i+1}(t)}{dt} + 10 \frac{dv_{i}(t)}{dt} + \frac{dv_{i-1}(t)}{dt} = g_1 v_i(t) + g_2 v_{i+1}(t) + g_3 v_{i-1}(t),
\] (18)

where

\[
f_1 = \left( 10\alpha - \frac{24d_1}{h^2} \right) u_i(t) + 8\alpha u_i(t) - \beta \left( u_{i+1}(t) + u_{i-1}(t) \right),
\]

\[
f_2 = \left( \alpha + \frac{12d_1}{h^2} \right) u_{i+1}(t) + \alpha u_{i+1}(t) - \beta \left( \frac{u_{i+1}(t) - u_{i-1}(t)}{2} \right) - \frac{\alpha}{2} u_{i+1}(t) - 2\alpha u_{i+1}(t) - \beta v_i(t),
\]

\[
f_3 = \left( \alpha + \frac{12d_1}{h^2} \right) u_{i-1}(t) + \beta \left( \frac{v_{i+1}(t) - v_{i-1}(t)}{2} \right) - \frac{\alpha}{2} u_{i-1}(t) - 2\alpha u_{i+1}(t) - \beta v_i(t),
\]

\[
g_1 = 10\gamma - \frac{24d_2}{h^2} + \epsilon \left( u_{i+1}(t) + 8u_i(t) + u_{i-1}(t) \right),
\]

\[
g_2 = \frac{12d_2}{h^2} \gamma - \frac{\epsilon}{2} (u_{i+1}(t) + 2u_i(t) - u_{i-1}(t)),
\]

\[
g_3 = \frac{12d_2}{h^2} \gamma - \frac{\epsilon}{2} (u_{i+1}(t) - 2u_i(t) - u_{i-1}(t)).
\]
2.2 The boundary spatial points

The main goal here is to construct a fourth order accurate on the spatial points for the left and right boundaries. This is may have accomplished by using techniques deduced in [12]. For simplicity to present our work, we will derive for \( u \) and for \( v \) will follow the same rules, to do this, begin with lift Neumann boundary condition, yields,

\[
\frac{\partial u(x_0,t)}{\partial x} = -3u(x_0,t) + 4u(x_1,t) - u(x_2,t) + \frac{\partial^3 u(x_0,t)}{\partial x^3} + \frac{h^3}{4} \frac{\partial^4 u(x_0,t)}{\partial x^4} + O(h^4)
\]

\[
\frac{\partial u(x_n,t)}{dx} = -3u(x_{n-2},t) + 4u(x_{n-1},t) - u(x_n,t) + \frac{\partial^3 u(x_n,t)}{\partial x^3} + \frac{h^3}{4} \frac{\partial^4 u(x_n,t)}{\partial x^4} + O(h^4)
\]

Substituting (9) and (10) into (19), this becomes

\[
3 \frac{du_0(t)}{dt} - 2 \frac{du_1(t)}{dt} - \frac{du_2(t)}{dt} = L_1 u_0(t) + L_2 u_1(t) + L_3 u_2(t)
\]

where

\[
L_1 = (-3 \frac{6d_1}{h^2} + \alpha) + 3 \left( \beta (v_0(t) - 2v_1(t) + v_2(t)) + 2\alpha u_0(t) + \beta v_0(t) \right) u_0(t),
\]

\[
L_2 = 6 \frac{d_1}{h^2} + \alpha - 12\alpha u_0(t) + 6\beta v_0(t),
\]

\[
L_3 = 6 \frac{d_1}{h^2} + \alpha - 12\alpha u_0(t) + 6\beta v_0(t).
\]

Go back to (19), to construct right boundary point, this can be obtained by substituting (9) and (10) in (20), imply

\[
3 \frac{du_{n-1}(t)}{dt} - 2 \frac{du_n(t)}{dt} - \frac{du_{n+1}(t)}{dt} = L_{11} v_n(t) + L_{22} v_{n-1}(t) + L_{33} v_{n-2}(t),
\]

where

\[
L_{11} = 3 \left( \frac{6d_1}{h^2} + \alpha \right) - 3 \left( 2\alpha u_n(t) + \beta (v_{n-2}(t) - 2v_{n-1}(t) + 2v_n(t)) \right),
\]

\[
L_{22} = -6 \left( \frac{4d_1}{h^2} + \alpha \right) - 3 \left( 2\alpha u_n(t) + \beta v_n(t) \right) u_{n-2}(t) + 6 \left( 2\alpha u_n(t) + \beta v_n(t) \right),
\]

\[
L_{33} = 3 \left( \frac{2d_1}{h^2} + \alpha \right) - 3 \left( 2\alpha u_n(t) + \beta v_n(t) \right).
\]

Similarly, for gives

\[
3 \frac{dv_1(t)}{dt} - 2 \frac{dv(t)}{dt} - \frac{dv(t)}{dt} = w_1 v_0(t) + w_2 v_1(t) + w_3 v_2(t),
\]

\[
\frac{dv_{n-2}(t)}{dt} + 3 \frac{dv_{n-1}(t)}{dt} - 3 \frac{dv_n(t)}{dt} = w_{11} v_n(t) + w_{22} v_{n-1}(t) + w_{33} v_{n-2}(t),
\]

where

\[
w_1 = 3 \left( \gamma - \frac{6d_2}{h^2} \right) - 3\beta (u_0(t) - 2u_1(t) + u_2(t)),
\]

\[
w_2 = 6 \left( \frac{4d_2}{h^2} - \gamma \right) + 6\epsilon u_0(t), \quad w_3 = 3 \left( \gamma - \frac{2d_2}{h^2} \right) - 3\epsilon u_0(t),
\]

\[
w_{11} = 3 \left( \frac{6d_1}{h^2} + \alpha \right) - 3 \left( 2\alpha u_n(t) + \beta (v_{n-2}(t) - 2v_{n-1}(t) + 2v_n(t)) \right),
\]

\[
w_{22} = 6 \left( \gamma - \frac{4d_2}{h^2} \right) - 6\epsilon u_n(t), \quad w_{33} = 6 \left( \gamma - \frac{4d_2}{h^2} \right) + 3\epsilon u_n(t).
\]

Collecting together (17), (18), (21), (22), (23) and (24), this will lead to a system of nonlinear ordinary differential equations as follows

\[
A \frac{du(x,t)}{dt} = B_1 u(x,t)
\]

\[
A \frac{dv(x,t)}{dt} = B_2 v(x,t),
\]

where
such that

\[ B_1^{11} = -3 \left( \frac{6d_1}{h^2} + \alpha \right) + c_1^{11}, \quad B_1^{12} = 6 \left( \frac{4d_1}{h^2} + \alpha \right) + c_1^{12}, \quad B_1^{13} = -3 \left( \frac{2d_1}{h^2} + \alpha \right) + c_1^{13}, \]

\[ 2B_1^{21} = 4 \frac{d_1}{h^2} + R_1^{11}, \quad B_1^{22} = 10\alpha - \frac{24d_1}{h^2} + R_1^{12}, \quad B_1^{23} = \alpha + \frac{12d_1}{h^2} + R_3^{13}, \]

\[ B_1^{n-1m-2} = \alpha + \frac{12d_1}{h^2} + R_1^{n-1,m-2}, \quad B_1^{n-1m-1} = 10\alpha - \frac{24d_1}{h^2} + R_2^{n-1,m-1}, \]

\[ B_1^{nm-2} = 3 \left( \frac{2d_1}{h^2} + \alpha \right) + c_1^{m-2}, \quad B_1^{nm-1} = -6 \left( \frac{4d_1}{h^2} + \alpha \right) + c_1^{nm-1}, \]

\[ B_1^{nm} = 3 \left( \frac{\alpha}{h^2} + \alpha \right) + c_1^{nm}. \]

and

\[ c_1^{11} = 3 \left( \beta (v_0(t) - 2v_1(t) + v_2(t)) + (2\alpha u_0(t) + \beta v_0(t)) \right), \]

\[ c_1^{12} = -6 \left( 2\alpha u_0(t) + \beta v_0(t) \right), \]

\[ c_1^{13} = 3 \left( 2\alpha u_0(t) + \beta v_0(t) \right), \]

\[ R_1^{ij} = \frac{\beta}{2} (v_{i+1}(t) - v_{i-1}(t)) - \frac{\alpha}{2} u_{i-1}(t) - 2\alpha u_i(t) - \beta v_{i1}(t), \quad i = 1, \ldots, n-1, j = 1, \ldots, m-2. \]

\[ R_2^{ij} = -8\alpha u_i(t) - \beta (v_{i+1}(t) - 12v_i(t) + v_{i-1}(t)), \quad i = 1, \ldots, n-1, j = 2, \ldots, m-1. \]

\[ R_3^{ij} = \alpha u_{i-1}(t) - \frac{\beta}{2} (v_{i+1}(t) - v_{i-1}(t)) - \frac{\alpha}{2} u_{i1}(t) - 2\alpha u_i(t) - \beta v_{i1}(t), \]

\[ i = 1, \ldots, n-1, j = 3, \ldots, m. \]

\[ c_1^{n-2} = -3(2\alpha u_n(t) + \beta v_n(t)), \]

\[ c_1^{n-1} = 6(2\alpha u_n(t) + \beta v_n(t)), \]

\[ c_1^{nm} = -3 \left( (2\alpha u_n(t) + \beta v_n(t)) + \beta (v_{n-2}(t) - 2v_{n-1}(t) + v_n(t)) \right). \]

\[ B_2 = \begin{bmatrix} B_2^{11} & B_2^{12} & B_2^{13} \\ B_2^{21} & B_2^{22} & B_2^{23} \\ \vdots & \vdots & \vdots \\ B_2^{nm-2} & B_2^{nm-1} & B_2^{nm} \end{bmatrix}, \]

where

\[ B_2^{11} = 3 \left( \gamma - \frac{2d_1}{h^2} \right), \quad B_2^{12} = 6 \left( \frac{4d_1}{h^2} - \gamma \right), \quad B_2^{13} = 3 \left( \gamma - \frac{2d_1}{h^2} \right), \]

\[ B_2^{21} = -\frac{2d_1}{h^2} + \gamma + Q_1^{11}, \quad B_2^{22} = 10\gamma - \frac{24d_1}{h^2} + Q_2^{12}, \quad B_2^{23} = \frac{12d_1}{h^2}, \quad B_2^{33} = \gamma + Q_3^{13}, \]

\[ B_2^{n-1m-2} = \frac{12d_1}{h^2} - \gamma + Q_1^{n-1,m-2}, \quad B_2^{n-1m-1} = 10\gamma - \frac{24d_1}{h^2} + Q_2^{n-1,m-1}, \]

\[ B_2^{nm-2} = 3 \left( \frac{2d_1}{h^2} - \gamma \right) + 3\varepsilon v_n(t), \quad B_2^{nm-1} = 6 \left( \gamma - \frac{4d_1}{h^2} \right) + 6\varepsilon v_n(t), \quad B_2^{nm} = 3 \left( \frac{6d_1}{h^2} - \gamma \right). \]

and

\[ Q_1^{ij} = -\frac{\varepsilon}{2} (u_{i+1}(t) - 2u_i(t) - u_{i-1}(t)), \quad i = 1, \ldots, n-1, j = 1, \ldots, m-2. \]
3. Implicit-Explicit Runge Kutta methods (IMEXRK)

The aim of this section is to combine the high-order compact difference scheme presented in section 2 with IMEXRK schemes to obtain a new and high-order method for solving (1) and (2). The curtail idea of IMEXRK is to split in two methods, the first is implicit Runge Kutta and the second is explicit Runge Kutta method and consequently leading to the linear part is treated by implicit scheme while nonlinear term is treated by explicit method we refer the readers to [15-25] in details. To do this, go back for the system of nonlinear ordinary differential equation in (28) and (29), since A is invertible, and for brevity, we will use the following notions

\[ Q_{ij}^e = \varepsilon (u_{i+1}(t) + 8u_i(t) + u_{i-1}(t)), \quad i = 1, \ldots, n - 1, j = 2, \ldots, m - 1. \]

\[ Q_{ij}^s = \frac{\varepsilon}{2} (u_{i+1}(t) + 2u_i(t) - u_{i-1}(t)), \quad i = 1, \ldots, n - 1, j = 3, \ldots, m. \]

The key idea of IMEX-RK methods is to decompose the right-hand side of (28) and (29) into stiff (\( F_1 \) and \( G_1 \)) and nonstiff (\( F_2 \) and \( G_2 \)) terms. Note that the no stiff term is treated by explicit Runge Kutta method while stiff term is treated by single diagonal implicit method. The resulting system of ODE’s can be very efficiently integrated using the IMEX approach.

\[
\begin{align*}
\frac{du}{dt} &= F_1(u, v) + F_2(u, v), \\
\frac{dv}{dt} &= G_1(u, v) + G_2(u, v).
\end{align*}
\]

(27)

Notes that the matrices \( F_1, G_1 = (a_{i,j}), a_{i,j} = 0 \) for \( j \geq i \) and \( F_2, G_2 = (\hat{a}_{i,j}) \), where \( F_1, G_1 \) is implicit scheme and \( F_2, G_2 \) is explicit scheme. An IMEX Runge-Kutta scheme is characterized by these two matrices and the coefficient vectors \( c^f = (c_1, \ldots, c_s)^T \), \( c^e = (c_{s+1}, \ldots, c_s)^T \). The above methods can be represented by a double tableau in the usual Butcher notation. Using the Kronecker product, this can be written in the Butcher tableau is

\[
\begin{array}{cccccccc}
  c_1 & a_{11} & 0 & 0 & \cdots & 0 \\
  c_2 & a_{21} & a_{22} & 0 & \cdots & 0 \\
  c_3 & a_{31} & a_{32} & a_{33} & \cdots & 0 \\
  \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\
  c_s & a_{s1} & a_{s2} & a_{s3} & \cdots & a_{ss} \\
\end{array}
\]

\[
\begin{array}{cccccccc}
  b_1 & b_2 & b_3 & \cdots & b_s \\
\end{array}
\]

\[
\begin{array}{cccccccc}
  \hat{c}_1 & 0 & 0 & 0 & \cdots & 0 \\
  \hat{c}_2 & \hat{a}_{21} & 0 & 0 & \cdots & 0 \\
  \hat{c}_3 & \hat{a}_{31} & \hat{a}_{32} & 0 & \cdots & 0 \\
  \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\
  \hat{c}_s & \hat{a}_{s1} & \hat{a}_{s2} & \hat{a}_{s3} & \cdots & 0 \\
\end{array}
\]

By using implicit-explicit Runge-Kutta scheme for above, gives

\[
\mathbf{u}_i = \mathbf{u}^0 + \Delta t \sum_{j=1}^{s} a_{ij}F_1(t_0 + c_j\Delta t, \mathbf{u}(t), \mathbf{v}(t)) + \Delta t \sum_{j=1}^{s-1} \hat{a}_{ij}F_2(t_0 + c_j\Delta t, \mathbf{u}(t), \mathbf{v}(t))
\]
\[ v_i = v_0 + \Delta t \sum_{j=1}^{s} a_{ij} g_1 \left( t_0 + c_i \Delta t, u_i(t), v_i(t) \right) + \Delta t \sum_{j=1}^{i-1} \hat{a}_{ij} g_2 \left( t_0 + c_i \Delta t, u_i(t), v_i(t) \right) \]

\[ u_i = u_0 + h \sum_{j=1}^{s} b_j f_1 \left( t_0 + c_i \Delta t, u_i(t), v_i(t) \right) + h \sum_{j=1}^{s} \hat{b}_j f_2 \left( t_0 + c_i \Delta t, u_i(t), v_i(t) \right) \]

\[ v_i = v_0 + h \sum_{j=1}^{s} b_j g_1 \left( t_0 + c_i \Delta t, u_i(t), v_i(t) \right) + h \sum_{j=1}^{s} \hat{b}_j g_2 \left( t_0 + c_i \Delta t, u_i(t), v_i(t) \right), \]

where

\[
A = \begin{bmatrix}
3 & -2 & -1 \\
1 & 10 & 1 \\
1 & 2 & -3
\end{bmatrix},
\]

\[
F_1 = \begin{bmatrix}
-3 \left( \frac{6d_1}{h^2} + \alpha \right) & 6 \left( \frac{4d_1}{h^2} + \alpha \right) & -3 \left( \frac{2d_1}{h^2} + \alpha \right) \\
2d_1 \alpha + \frac{12d_1}{h^2} & 10 \alpha - \frac{24d_1}{h^2} & 2d_1 \alpha + \frac{12d_1}{h^2} \\
3 \left( \frac{2d_1}{h^2} + \alpha \right) & -6 \left( \frac{4d_1}{h^2} + \alpha \right) & 3 \left( \alpha + \frac{6d_1}{h^2} \right)
\end{bmatrix},
\]

\[
F_2 = \begin{bmatrix}
c_{i,1}^{1,1} & c_{i,1}^{1,2} & c_{i,1}^{1,3} \\
R_{i,1}^{1,1} & R_{i,1}^{1,2} & R_{i,1}^{1,3} \\
\vdots & \vdots & \vdots \\
R_{i,n-1,m-2}^{1,1} & R_{i,n-1,m-1}^{1,1} & R_{i,n-1,m}^{1,1} \\
c_{i,n,m-2}^{1,1} & c_{i,n-1,m}^{1,1} & c_{i,n,m}\end{bmatrix},
\]

where \( c_{i,1}^{1,1} = 3 \left( \beta \left( v_0(t) - 2v_1(t) + v_2(t) \right) + \left( 2\alpha u_0(t) + \beta v_0(t) \right) \right), \)

\( c_{i,1}^{1,2} = -6 \left( 2\alpha u_0(t) + \beta v_0(t) \right), \)

\( c_{i,1}^{1,3} = 3 \left( 2\alpha u_0(t) + \beta v_0(t) \right), \)

\( R_{i,j}^{1,1} = \frac{\beta}{2} \left( v_{i+1}(t) - v_{i-1}(t) \right) - \frac{\alpha}{2} u_{i-1}(t) - 2\alpha u_i(t) - 1\beta v_i(t), \ i = 1, ..., n - 1, j = 1, ..., m - 2. \)

\( R_{i,j}^{1,2} = -8\alpha u_i(t) - \beta \left( v_{i+1}(t) - 12v_i(t) + v_{i-1}(t) \right), \quad i = 1, ..., n - 1, j = 2, ..., m - 1. \)

\( R_{i,j}^{1,3} = \alpha u_{i-1}(t) - \frac{\beta}{2} \left( v_{i+1}(t) - v_{i-1}(t) \right) - \frac{\alpha}{2} u_{i+1}(t) - 2\alpha u_i(t) - \beta v_i(t), \)

\( i = 1, ..., n - 1, j = 3, ..., m. \)
\[ c_i^{n,m-2} = -3(2\alpha u_n(t) + \beta v_n(t)), \]
\[ c_i^{n,m-1} = 6(2\alpha u_n(t) + \beta v_n(t)), \]
\[ c_i^{n,m} = -3 \left( (2\alpha u_n(t) + \beta v_n(t)) + \beta(v_{n-2}(t) - 2v_{n-1}(t) + v_n(t)) \right), \]

and
\[
G_1 = \begin{bmatrix}
3 \left( \gamma - \frac{6d_2}{h^2} \right) & 6 \left( \frac{4d_2}{h^2} - \gamma \right) & 3 \left( \gamma - \frac{2d_2}{h^2} \right) \\
\frac{12d_2}{h^2} - \gamma & 10\gamma - \frac{24d_2}{h^2} & \frac{12d_2}{h^2} - \gamma \\
\frac{12d_2}{h^2} - \gamma & 10\gamma - \frac{24d_2}{h^2} & \frac{12d_2}{h^2} - \gamma \\
3 \left( \frac{2d_2}{h^2} - \gamma \right) & 6 \left( \gamma - \frac{4d_2}{h^2} \right) & 3 \left( \frac{6d_2}{h^2} - \gamma \right)
\end{bmatrix}
\]

\[
G_2 = \begin{bmatrix}
N_1 & N_2 & N_3 \\
Q_1^{i,1} & Q_2^{i,2} & Q_3^{i,3} \\
\vdots & \vdots & \vdots \\
Q_1^{n-1,m-2} & Q_2^{n-1,m-1} & Q_3^{n-1,m} \\
M_1 & M_2 & M_3
\end{bmatrix}
\]

such that
\[ N_1 = -3\varepsilon (2u_0(t) - 2u_1(t) + u_2(t)), N_2 = 6\varepsilon u_0(t), N_3 = -3\varepsilon u_0(t), \]
\[ Q_1^{i,j} = -\frac{\varepsilon}{2} (u_{i+1}(t) - 2u_i(t) - u_{i-1}(t)), \quad i = 1, \ldots, n-1, j = 1, \ldots m-2 \]
\[ Q_2^{i,j} = \varepsilon (u_{i+1}(t) + 8u_i(t) + u_{i-1}(t)), \quad i = 1, \ldots, n-1, j = 2, \ldots m-1 \]
\[ Q_3^{i,j} = \frac{\varepsilon}{2} (u_{i+1}(t) + 2u_i(t) - u_{i-1}(t)), \quad i = 1, \ldots, n-1, j = 3, \ldots m \]
\[ M_1 = 3\varepsilon u_0(t), M_2 = -6\varepsilon u_0(t), \quad M_3 = 3\varepsilon (u_{n-2}(t) - 2u_{n-1}(t) + u_n(t)). \]

4. Numerical experiments
We shall now illustrate the performance of a presented method, through an implementation based on the Matlab programming. This section aims to present the relation between sharks and fish populations. Two cases will focus on it in this paper. Firstly, for which conditions sharks avoid killing off the fish population. Secondly, what conditions can sharks and fish coexist so that their populations oscillate, but neither group ever is extinguished. In this article, we will consider different methods of IMEX-RK \( (p, s_1, s_2) \) the number of \( p \) is the order of the scheme, \( s_2 \) is the number of stage implicit schemes and \( s_2 \) is the number of stage explicit schemes [19].

Table 4.1. implicit-explicit Runge–Kutta IMEXRK \((1, 1, 1)\) methods based on Butcher tableaux.

| 1 | 1/2 | 1/2 |
|---|-----|-----|
| 1 | 1 | 0 | 1/2 | 1/2 |
Table 4.2. implicit-explicit Runge–Kutta IMEXRK(2,2,2) methods based on Butcher tableaux.

| 0   | 0   | 0   | 1/2 | 1/2 | 0   | 0   | 0   | 1/2 | 1/2 |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| 1/2 | 1/2 | 0   | 1/2 | 1/2 | 1/4 | 1/4 | 0   | 1/4 | 1/4 |
| 1   | 1/2 | 1/2 | 0   | 1   | 1/3 | 1/3 | 1/3 | 1/3 | 1/3 |

Table 4.3. implicit-explicit Runge–Kutta IMEXRK(2,3,3) methods based on Butcher

\begin{align*}
\alpha & \quad 0 \\
0 & 0 & 0 & 0 & 0 & -\alpha & \alpha & 0 & 0 \\
1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 - \alpha & \alpha & 0 \\
1/2 & 0 & 1/4 & 1/4 & 0 & 1 & \beta & 1/4 & 1/2 - \beta - \eta - \alpha & \alpha \\
\end{align*}

\begin{align*}
\alpha & = 0.24169426078821, \\
\beta & = \alpha / 4, \\
\eta & = 0.12915286960590.
\end{align*}

Table 4.4. implicit-explicit Runge–Kutta IMEXRK(3,4,4) methods based on Butcher

\begin{align*}
\alpha_{21} & = 0.39098372452428 \\
\alpha_{31} & = 1.09436646160460 \\
\alpha_{32} & = 0.33181504274704 \\
\alpha_{41} & = 0.14631668003312 \\
\alpha_{42} & = 0.69488738277516 \\
\alpha_{43} & = 0.46893381306619 \\
\alpha_{51} & = -1.33389883143642 \\
\alpha_{52} & = 2.90509214801204 \\
\alpha_{53} & = -1.06511748457024 \\
\alpha_{54} & = 0.27210900509137 \\
\beta_{1} & = b_{1} = a_{51} \\
\beta_{2} & = b_{2} = a_{52} \\
\beta_{3} & = b_{3} = a_{53} \\
\beta_{4} & = b_{4} = a_{54} \\
\beta_{5} & = b_{5} = a_{55} \\
\end{align*}

\begin{align*}
a_{11} & = 1/4 \\
a_{21} & = 0.34114705729739 \\
a_{22} & = 1/4 \\
a_{31} & = 0.80458720789763 \\
a_{32} & = -0.07095262154540 \\
a_{33} & = 1/4 \\
a_{41} & = -0.52932607329103 \\
a_{42} & = 1.15137638494253 \\
a_{43} & = -0.80248263237803 \\
a_{44} & = 1/4 \\
a_{51} & = 0.11933093090075 \\
a_{52} & = 0.5512553134927 \\
a_{53} & = -0.1216872844994 \\
a_{54} & = 0.20110104014943 \\
a_{55} & = 1/4 \\
\end{align*}

Example 4.1

A basic model for studying the interaction of species in population biology is the predator-prey model. The predator-prey model is a planar system representing the behaviour of a population of prey \( u \), say Fish, and a population of predators \( v \), say Sharks.

With initial conditions

\[ u(x,0) = f(x) = \frac{1}{10} \sin^2 \left( \frac{24}{10} \pi x \right) + \frac{28}{100} \sin^2 \]
Figure 4.1 show that fish population increases, while sharks population remains almost constant in first few time steps with sitting the rate of contact (β = 0.5) between the two species is higher compared to the rate of death (ε = 0.05). However, the Sharks population quickly starts to grow in the areas in the absence of fish, and the result of this can be seen in Figure 4.2. This is consequently leading to Sharks population will dominate the ecosystem and fish population will extinguished with the increased number of sharks as shown in Figure 4.3, 4.4. In the period, fish population start to grown, while the predator population becomes quite small as indicated in Figure 4.5, 4.6. Eventually, both prey and predator repeat themselves, producing a new travelling front which moves again from the top to the bottom. Its formation can be seen in Figure 4.6, 4.7, 4.8. Under normal conditions both prey and predator, always tend to co-exist, no species is extinguished from the ecosystem. This only happens, with the absence of external forces such as human interference in the ecosystem, where fishing is one of the activity in this particular case, which destroy
Figure 4.3: Example 1. Numerical solution by using IMEXRK (2,2,2) scheme for $\alpha = 0.01$, $\beta = 0.01, \gamma = 3.42, \epsilon = 1, d_1 = 0.001, d_2$

Figure 4.4: Example 1. Numerical solution by using IMEXRK(2,2,2) scheme for $\alpha = 0.5$, $\beta = 1, \gamma = 3.8, \epsilon = 1, d_1 = 0.001, d_2 = 0.001$

Figure 4.5: Example 1. Numerical solution by IMEXRK (3,3,2) $\alpha = 0.5, \beta = 2, \gamma = 3.35$

Figure 4.6: Example 1. Numerical solution by using IMEXRK2 scheme for $\alpha = 2$, $\beta = 1, \gamma = 6, \epsilon = \frac{1}{15}, d_1 = 0.001, d_2 = 0.001$ for $t = 45$
Example 4.2

\[
\frac{\partial u}{\partial t} = d_1 \frac{\partial^2 u}{\partial x^2} + \alpha u - \beta uv
\]

\[
\frac{\partial v}{\partial t} = d_2 \frac{\partial^2 v}{\partial x^2} - \gamma v + \varepsilon uv,
\]

boundary conditions
\frac{\partial u}{\partial x}_{x=0} = 0, \quad \frac{\partial u}{\partial x}_{x=1} = 0, \quad \frac{\partial v}{\partial x}_{x=0} = 0, \quad \frac{\partial v}{\partial x}_{x=1} = 0,

and initial condition at t = 0:

\begin{align*}
u(x, 0) &= u_0(x) = 70x^2(x - 1)^2e^{-3.5x}, \\
v(x, 0) &= v_0(x) = 70x^2(x - 1)^2e^{3.5(x-1)}.
\end{align*}

Figure 4.10: Example 2. numerical solution by using IMEXRK (4,5,5) scheme

5. Conclusion

This paper is devoted to propose a high-order compact scheme for solving the one-dimensional Lotka-Volterra-diffusion equation. The curtail idea is to combine a fourth-order compact finite difference scheme to discretise the spatial derivative and implicit-explicit Runge Kutta method to the time integration, and this leads to nonlinear system of ordinary differential equations. The proposed scheme has fourth order in space and time. The main use for these combinations is in splitting a nonlinear system by a pair of methods (F,G) where the method F is implicit and the method G is explicit. An important factor in our proposed method is to reduce the number of iterations and consequently leading to a reduction in the computational cost of the scheme. It is clearly confirmed that the proposed method is a good in term of the computational cost and stability. Numerical experiments through Matlab programming also confirm that the proposed method are reliable and efficient for solving Lotka-Volterra-diffusion equation. This approach may be extended to tackle stochastic equations with growth model [26, 27]. Another interesting of this work is to use discontinuous Galerkin methods for estimating this type of problem in terms of $L^\infty(L^2)$ and $L^\infty(H^1)$ [28 – 31].

References

[1] Younis A. Sabawi, “A Posteriori L^\infty(H^1)- Error Bound in Finite Element Approximation of
Semilinear Parabolic Problems, “2019 First International Conference of Computer and Applied Science (CAS), Baghdad, Iraq, 2019, pp. 102-106, doi: 10.1109/CAS47993.2019.9075699.

[2] Sabawi YA. A Posteriori Error Analysis in Finite Element Approximation for Fully Discrete Semilinear Parabolic Problems. InFinite Element Methods and Their Applications 2020 Dec 10. IntechOpen

[3] Fu Y. Compact fourth-order finite difference schemes for Helmholtz equation with high wave numbers. Journal of Computational Mathematics. 2008 Jan 1:98-11

[4] Manaa SA, Mohemmed MA, Hussein YA. A Numerical Solution for Sine-Gordon Type System. Tikrit Journal of PureScience. 2010;15(3):106-13.

[5] 2N order compact finite difference scheme with collocation method for solving the generalized Burger’s–Huxley and Burger’s–Fisher equations

[6] Yang X, Ge Y, Zhang L. A class of high-order compact difference schemes for solving the Burgers’ equations. Applied Mathematics and Computation. 2019 Oct 1;358:394-417.

[7] Boscaino S, Filbet F, Russo G. High order semi-implicit schemes for time dependent partial differential equations. Journal of Scientific Computing. 2016 Sep 1;68(3):975-1001.

[8] Bhatt H, Chowdhury A. A compact fourth-order implicit-explicit Runge-Kutta type scheme for numerical solution of the Kuramoto-Sivashinsky equation. arXiv preprint arXiv:1911.12183. 2019 Nov 27.

[9] Hammad DA, El-Azab MS. 2N order compact finite difference scheme with collocation method for solving the generalized Burger’s–Huxley and Burger’s–Fisher equations. Applied Mathematics and Computation. 2015 May 1;258:296-311.

[10] Cao HH, Liu LB, Zhang Y, Fu SM. A fourth-order method of the convection–diffusion equations with Neumann boundary conditions. Applied Mathematics and Computation. 2011 Jul 15;217(22):9133-41.

[11] Fu Y, Tian Z, Liu Y. A Compact Exponential Scheme for Solving 1D Unsteady Convection-Diffusion Equation with Neumann Boundary Conditions. arXiv preprint arXiv:1805.05728. 2018 May 15.

[12] Yao C, Zhang Y, Chen J, Ling X, Jing K, Lu Y, Fan E. Development of a fourth-order compact finite difference scheme for simulation of simulated-moving-bed process. Scientific reports. 2020 May 8;10(1):1-3.

[13] Nyaanga, P . P . (2004) “The interaction of species in population biology”, www.ii.uib.no/~pius/thesis3.pdf.

[14] Qasem AF, Manaa SA. Numerical solution of non-linear prey-predator system using finite elements method. AL-Rafidain Journal of Computer Sciences and Mathematics. 2007;4(2):113-33.

[15] Hussein YA. Combination Between Single Diagonal Implicit and Explicit Runge Kutta (SDIMEX-RK) Methods for solving stiff Differential equations. Tikrit Journal of Pure Science. 2011;16(1):94-101.

[16] Pareschi L, Russo G. Implicit–explicit Runge–Kutta schemes and applications to hyperbolic systems with relaxation. Journal of Scientific computing. 2005 Oct 1;25(1):129-55.

[17] Ascher UM, Ruuth SJ, Spiteri RJ. Implicit–explicit Runge–Kutta methods for time-dependent partial differential equations. Applied Numerical Mathematics. 1997 Nov 1;25(2-3):151-67

[18] Younis A, Sabawi, Mardan A. Pirdawood, Hemm M. Rasool. Model Reduction and Implicit-Explicit Runge-Kutta Methods for Nonlinear Stiff Initial-Value Problems, “Seventh International Scientific Conference Iraqi Al Khwarizmi Society”, Mousl, Iraq, 2021. Accepted for publication.

[19] Pareschi L, Russo G. Implicit–explicit Runge–Kutta schemes and applications to hyperbolic systems with relaxation. Journal of Scientific computing. 2005 Oct 1;25(1):129-55.

[20] Pao CV, Wang YM. Numerical solutions of a three-competition Lotka–Volterra system. Applied Mathematics and computation. 2008 Oct 1;204(1):423-40.

[21] Owolabi KM, Patidar KC. Higher-order time-stepping methods for time-dependent reaction–
diffusion equations arising in biology. Applied Mathematics and Computation. 2014 Aug 1;240:30-50.

[22] Ascher UM, Ruuth SJ, Spiteri RJ. Implicit-explicit Runge-Kutta methods for time-dependent partial differential equations. Applied Numerical Mathematics. 1997 Nov 1;25(2-3):151-67.

[23] Hundsdorfer W, Ruuth SJ. IMEX extensions of linear multistep methods with general monotonicity and boundedness properties. Journal of Computational Physics. 2007 Aug 10;225(2):2016-42.

[24] Li D, Zhang C, Wang W, Zhang Y. Implicit–explicit predictor–corrector schemes for nonlinear parabolic differential equations. Applied mathematical modelling. 2011 Jun 1;35(6):2711-22.

[25] Zhang K, Wong JC, Zhang R. Second-order implicit–explicit scheme for the Gray–Scott model. Journal of Computational and Applied Mathematics. 2008 Apr 1;213(2):559-81.

[26] Tesfay A, Tesfay D, Khalaf A, Brannan J. Mean exit time and escape probability for the stochastic logistic growth model with multiplicative $\alpha$-stable Lévy noise. Stochastics and Dynamics. 2020 Aug 28:2150016

[27] Khalaf AD, Abouagwa M, Mustafa, Wang X. Stochastic Volterra integral equations with jumps and the strong superconvergence of the Euler–Maruyama approximation. Journal of Computational and Applied Mathematics. 2021 Jan;382:113071.

[28] Younis A. Sabawi, Adaptive discontinuous Galerkin methods for interface problems, PhD Thesis, University of Leicester, Leicester, UK (2017).

[29] Cangiani, Andrea, Emmanuil H. Georgoulis, and Younis A. Sabawi Adaptive discontinuous Galerkin methods for elliptic interface problems, Math. Comp. 87 (2018), no. 314, 2675–2707.

[30] Andrea Cangiani, Emmanuil H. Georgouils, and Younis A. Sabawi, Convergence of an adaptive discontinuous Galerkin method for elliptic interface problems, Journal of Computational and Applied Mathematics 367 (2020):112397

[31] Sabawi YA. A Posteriori $L_\infty(L_2) + L_2(H^1)$-Error Bounds in Discontinuous Galerkin Methods For Semidiscrete Semilinear Parabolic Interface Problems. Baghdad Science Journal 2021;18(3):0522-.