Universiti Malaysia Terengganu Journal of Undergraduate Research
Volume 1 Number 4, October 2019: 1-8

eISSN: 2637-1138

ON NUMERICAL METHODS FOR SECOND-ORDER NON-LINEAR ORDINARY DIFFERENTIAL EQUATIONS (ODEs): A REDUCTION TO A SYSTEM OF FIRST-ORDER ODEs

LOY KAK CHOON[A] AND PUTERI NURUL FATIHAH BINTI MOHAMAD AZLI[B,*]

[a,b]*Faculty of Ocean Engineering Technology and Informatics, Universiti Malaysia Terengganu, 21030 Kuala Nerus, Terengganu, Malaysia.*

* *Corresponding author: puteri_fatihah96@yahoo.com*

Abstract: 2^{nd}-order ODEs can be found in many applications, e.g., motion of pendulum, vibrating springs, etc. We first convert the 2^{nd}-order nonlinear ODEs to a system of 1^{st}-order ODEs which is easier to deal with. Then, Adams-Bashforth (AB) methods are used to solve the resulting system of 1^{st}-order ODE. AB methods are chosen since they are the explicit schemes and more efficient in terms of shorter computational time. However, the step size is more restrictive since these methods are conditionally stable. We find two test cases (one test problem and one manufactured solution) to be used to validate the AB methods. The exact solution for both test cases are available for the error and convergence analysis later on. The implementation of 1^{st}-, 2^{nd}- and 3^{rd}-order AB methods are done using Octave. The error was computed to retrieve the order of convergence numerically and the CPU time was recorded to analyze their efficiency.

Keywords: Ordinary differential equations, Adams-Bashforth, system of ODE, manufactured solution

Introduction

An equation involving a function with its derivatives is known as differential equation. Ordinary differential equations (ODEs) are differential equations which involve differentiation of only one independent variable while partial differential equations (PDEs) are differential equations which involve differentiation of more than one independent variable. In general, differential equations have their derivatives of various orders. In this paper, we focus only on ordinary differential equations.

Ordinary differential equations are used to model real applications such as motion of pendulum, population growth and decay etc. A density-dependent growth of prey population of Rosenzweig-MacArthur was modeled using the predator-prey model governed by a system of an ODE (Dimitrov & Kojouharov, 2005). The Rosenzweig-MacArthur model was presented in American Naturalist during 1963 where it was motivated by the living world behaviour (Hurvoka, 2013).

It is called n^{th} order differential equations if the ordinary differential equations contains $\frac{d^n y}{dt^n}$, where $n \in N$ is the highest value that can be found. An ODE is called first-order ODE because it involves function and first derivatives of function y, $\frac{dy}{dt}$. The general form of first-order ODEs can always be expressed as

$$y' = f(t,y), \qquad y(t_0) = y_0.$$

Here, the variable t is denoted as the 'independent variable' or 'time' and $y = y(t)$, is denoted as the 'dependent variable'. Equations which contain function variables of two or more derivatives of dependent variables with respect to one or more independent variables are called higher-order ordinary differential equations. The general form of higher-order ODEs can be expressed as follows:

$$f(y^{(k)}, y^{(k-1)}, \ldots, y; t) = 0$$

where the initial values are given as

$$y^{(k)}(t_0) = \alpha_0$$
$$y^{(k-1)}(t_1) = \alpha_1,$$
$$\vdots$$
$$y(t_k) = \alpha_k.$$

The main interest of this research is to approximate the solution of $y(t)$ only within a certain interval of $a< t < b$, and $a = t_0$ will be assumed as the initial time. One of the methods that can be used to solve the ordinary differential equations is qualitative analysis. It can be used to present a visual picture of the solutions behaviour to an ODE and verify numerical and analytical solutions utilized by the plot of directional fields (Tomas, 2013). However, qualitative analysis cannot give a very precise answer. To solve this problem, we often used numerical methods which can give approximate solutions to the differential equations. Numerical methods involve many iterations which give several disadvantages on the total time required to obtain acceptable approximation, but on the other side, they can solve complex problems both physically and geometrically. Due to the fact that ODEs use complicated algebraic manipulations and time consuming during the solving process, it is shown that ODEs sometimes are problematic and not practical to be solved analytically (Junior *et al.*, 2018).

A few examples of the numerical methods used to solve ODEs are Euler's method, Runge-Kutta method, Adams-Bashforth method etc. In one of the latest development, the first-order fractional ordinary differential equation which governs the epidemic model was solved using implicit Adams methods (Ameen & Novati, 2016). Senthilnathan (2018) proved the accuracy by solving the initial value problems for ODE using two numerical methods of Euler and 4^{th}-order of Runge Kutta method and compared its numerical solutions with the exact solutions. In a separate literature, the fractional ODE is solved using an explicit Adams-Bashforth methods (Garrappa, 2007) and the backward Euler method is used to solve the discrete epidemic model (Enatsu *et al*., 2010).

Here, we start by introducing the nonlinear differential equation. If an ordinary differential equation consists of one or more nonlinear terms involving or without involving the derivatives, the differential equation is then called as a nonlinear ordinary differential equation. In this research, we will focus on the approximation of higher-order nonlinear ordinary differential equations by using so-called a reduction to a system of first-order ODEs. The system of first-order ODEs will then be solved using the chosen numerical method which is Adams-Bashforth (AB) method. Similar work of solving higher-order ODEs using AB-method after reducing them into system of first-order ODEs can be found in many literatures. The approximation of high-order ODEs can be done by reducing them into the system of first-order ODEs before solving them using both third-order Adams-Bashforth and Adams-Moulton methods (Beeman, 1975).

Numerical Methods for High-Order ODEs

In this section, we will solve two test cases which consist of high-order ODEs using Adam-Bashforth method. The first test case is a higher-order ODE with a simple analytical formula. For the second test case, we use a manufactured solution which gives a great convenience to get an ansatz solution for any desired ODEs.

Lipschitz Continuity

We can prove the existence and uniqueness of the test problem by showing the Lipschitz bound. For systems of $s > 1$ ordinary differential equations, $u(t) \in \mathbb{R}$ and $f(u,t)$ is a function mapping $\mathbb{R}^s \times \mathbb{R} \rightarrow \mathbb{R}^s$. We can say that the function f is Lipschitz continuous in some norm $\|.\|$ if there is a constant L such that

$$\| f(u,t) - f(u^*,t) \| \leq L \| u-u^* \|,$$

for all u in a neighbourhood of u^* (LeVeque, 2004). By the equivalence of finite-dimensional norms as well, if f is Lipschitz continuous in one norm then it is Lipchitz continuous in any other norm, though the Lipschitz constant may depend on the norm chosen. The same theorem on existence and uniqueness can also be applicable to systems of ODEs.

Adams-Bashforth Method

Adams-Bashforth methods are chosen to solve our test cases in this project. Adams-Bashforth methods can be categorised as Linear Multistep Methods (LMMs) since they require one or more values from previous steps to approximate the next value. The first three terms of the method can be expressed as follows:

1-step: $U^{n+1} = U^n + kf(U^n)$,

2-step: $U^{n+2} = U^{n+1} + \frac{k}{2}(-f(U^n)+3f(U^{n+1}))$,

3-step: $U^{n+3} = U^{n+2} + \frac{k}{12}(5f(U^n)-16f(U^{n+1})+23f(U^{n+2}))$.

In this research, we will only focus on the 1-, 2- and 3-step Adams-Bashforth methods. These methods will produce 1st-, 2nd- and 3rd-order of accuracy for the global error, respectively. Next, we will investigate the zero-stability of 1-, 2- and 3-step Adams-Bashforth methods by finding the characteristic equation for each method. According to LeVeque (2004), "an r -step method is said to be zero-stable if the roots of the characteristic polynomial $p(\xi)$ satisfies the conditions" as follows:

i) $|\xi_j| \leq 1$ for $j = 1, 2, ..., n$,

ii) if ξ_j is a repeated root, then $|\xi_j| \leq 1$.

We then find the characteristic equations for each step of the Adams-Bashforth methods. We start with the 1-step method of Adams-Bashforth method, i.e.

$$U^{n+1} = U^n + kf(U^n).$$

First, we let $U^{n+1} = \xi^{n+1}$ and $U^n = \xi^n$ to yield the characteristic polynomial

$$p(\xi) = \xi^{n+1} - \xi^n = 0.$$

We then multiply ξ^{-n} to the characteristic polynomial to obtain ξ-1= 0. Therefore, we obtain ξ=1. With the root, the general solution has the form $y_n = C_1$. As a conclusion, 1-step method of Adams-Bashforth is zero-stable.

Formulation: Test Case I

A second-order ODE problem can simply be written as:

$$y'' + y = t \text{ given } \quad y(0)=1 \text{ and } y'(0)=0.$$

The ODE is then converted to a 1st-order ODE system which finally gives

$$y'_1(t) = y_2(t),$$

$$y'_2(t) = t - y_1(t),$$

incorporating with the initial values $y_1(0) = 1$ and $y_2(0) = 0$.

Using Laplace Transform and Inverse Laplace Transform, the ODE is then solved analytically and the solution can be expressed as:

$$y(t) = t - \sin t + \cos t.$$

We then prove the existence and uniqueness of the test problem using Lipschitz continuity and we have L=1:

$$\| f(u) - f(u^*) \|_\infty \leq \| u - u^* \|_\infty.$$

The solution of the test problem hence is unique for all $t \in \mathbb{R}$ or in another words, the IVP is globally Lipschitz.

Numerical Schemes: Test Case I

In this section, we write numerical schemes for the 1st, 2nd- and 3rd-order of the Adams-Bashforth methods for both test cases before we implement them using Octave. First of all, we define the system of 1st-order ODE for the test problem using f_1 and f_2 such that

$$f_1 = y'_1(t) = y_2(t),$$

$$f_2 = y'_2(t) = t - y_1(t).$$

Using the functions f_1 and f_2 above in the AB methods we have the following numerical schemes.

1-step AB-method:

$$y_1^{n+1} = y_1^{n} + h(y_2^{n}),$$

$$y_2^{n+1} = y_2^{n} + h(t_n - y_1^{n}).$$

2-step AB-method:

$$y_1^{n+2} = y_1^{n+1} + \frac{h}{2}\left(-y_2^{n} + 3y_2^{n+1}\right),$$

$$y_2^{n+2} = y_2^{n+1} + \frac{h}{2}\left(-(t_n - y_1^{n}) + 3(t_{n+1} - y_1^{n+1})\right).$$

3-step AB-method:

$$y_1^{n+3} = y_1^{n+2} + \frac{h}{12}\left(5(y_2^{n}) - 16(y_2^{n+1}) + 23(y_2^{n+2})\right),$$

$$y_2^{n+3} = y_2^{n+2} + \frac{h}{12}\left(5(t_n - y_1^{n}) - 16(t_{n+1} - y_1^{n+1}) + 23(t_{n+2} - y_1^{n+2})\right).$$

The computation of the test problem is done at $t \in [0,1]$.

Numerical Results: Test Case I

i) Error Analysis

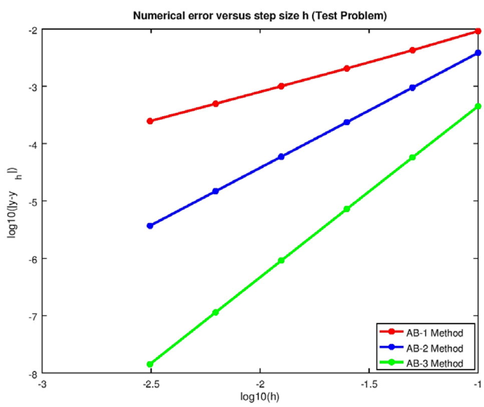

Figure 1 : Numerical Error versus Step Size *h* Used to Compute the Test Problem

Figure 1 shows a graph of numerical error of 1-, 2- and 3-step of AB methods versus the step size *h* chosen to compute the test problem. As we can see in the graph, the numerical errors of 1-, 2- and 3-step of AB methods get smaller as the step size *h* decreases. The AB-3 method produces the smallest error of all three methods with a fixed the step size *h*. At h = 0.00625, AB-1 method has the numerical error of 4.95 x 10^{-4} while the numerical error for the AB-2 method and AB-3 method is 1.47 x 10^{-5} and 1.14 x 10^{-7}, respectively. At h=0.003125, the error for the AB-2 method is 3.69 x 10^{-6}. The numerical error for AB-1, AB-2 and AB-3 method are 4.21 x 10^{-3}, 9.46 x 10^{-4} and 5.72 x 10^{-5}, respectively as the value of step size *h* is fixed at 0.05. So, we conclude that the numerical methods are consistent or, in other words, the smallest error is obtained as step size *h* gets larger.

ii) Efficiency Analysis

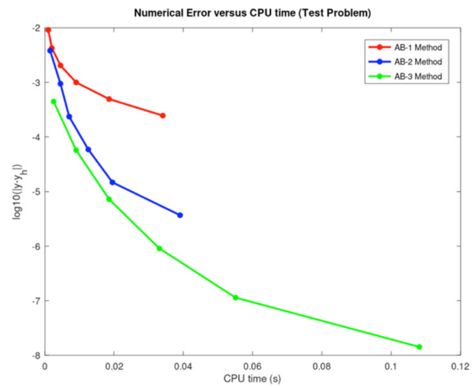

Figure 2 : Numerical Error versus CPU Time Required to Compute The Test Problem

Figure 2 above shows the results on the numerical error against the CPU time for each step size *h*. By referring to the graph, the AB-1 method produces the CPU time of 2x10^{-3}s, the AB-2 method produces 4.48x10^{-3}s of CPU time and the AB-3 method produces 9x10^{-3}s of the CPU time at the step size of h=0.05. As we can see in the figure above, the CPU time increases in nonlinear fashion when the numerical error of each method decreases. At h=0.025, the CPU time of AB-1 method is 4.5x10^{-3}s and for AB-2 7x10^{-3}s is recorded. For AB-3 method, 1.85x10^{-2}s is required to complete the computation. The CPU times for AB-1 method are 9.76x10^{-4}s and 3.4x10^{-2}s at h=0.1 and 0.003125 respectively. As the step size *h* decreases, the CPU time is increasing in a nonlinear fashion and this applies to all the three steps of AB methods. We suspected that to reach arbitrary small error even for AB-3 method requires a very long time in an asymptotical manner. To increase the efficiency, higher order AB method (>3 order) should be used.

Formulation : Test Case II

For the second test case, we have a manufactured solution for an ODE problem which takes the form of

$$y'' + 4y' + 5e^{y} = \frac{1}{t^2} + \frac{1}{t}$$

where the suggested initial condition for the manufactured solution are $y(1)=0$ and $y'(1)=-1$. Then, we rewrite the manufactured solution above as a system of 1st-order ODE which takes the form of

$$y'_1(t) = y_2(t)$$

$$y'_2(t) = -4y_2(t)-5e^{y_1(t)} + \frac{1}{t^2} + \frac{1}{t}$$

with the initial conditions $y_1(1) = -1$ and $y_2(1) = 0$. The exact solution exists everywhere except when $t = 0$ and the function is only belongs to a continuous function $C^0(\mathbb{R}\backslash\{0\})$. We then prove whether or not the manufactured solution has a unique solution by using Lipschitz continuity. So, we have Lipschitz continuity with $= 5e^{y_1}+4$:

$$\| f(u) - f(u^*) \|_\infty \leq (5e^{y_1}+4) \| u-u^* \|_\infty$$

$$\| f(u) - f(u^*) \|_\infty \leq (5e^{v}+4) \| u-u^* \|_\infty$$

where $v = \max\{y_1, y_2\} = \| u \|_\infty$. The solution of the manufactured solution is not globally unique since $y_1 \to \infty$, when $L \to \infty$. Also, the solution does not exist when $t \to 0$. Since the interval is chosen for the computation is taken at $t \in [1, 2]$, the local solution exist and is unique.

Numerical Schemes: Test Case II

The system of the 1st-order ODE of the manufactured solution can also be defined by f_1 and f_2 such as follows:

$$f_1 = y_1' = y_2(t)$$

$$f_2 = y_2' = -4y_2(t)-5e^{y_1(t)} + \frac{1}{t^2} + \frac{1}{t}$$

The numerical scheme for the AB-Methods for the manufactured solution can be written as below.

1-step AB-method:

$$y_1^{n+1} = y_1^{n} + hf_1(y_1^{n}, y_2^{n}, t_n),$$

$$y_2^{n+1} = y_2^{n} + hf_2(y_1^{n}, y_2^{n}, t_n).$$

2-step AB-method:

$$y_1^{n+2} = y_1^{n+1} + \frac{h}{2}(-f_1(y_1^{n}, y_2^{n}, t_n) + 3f_1(y_1^{n+1}, y_2^{n+1}, t_{n+1})),$$

$$y_2^{n+2} = y_2^{n+1} + \frac{h}{2}(-f_2(y_1^{n}, y_2^{n}, t_n) + 3f_2(y_1^{n+1}, y_2^{n+1}, t_{n+1})).$$

3-step AB-method:

$$y_1^{n+3} = y_1^{n+2} + \frac{h}{12}(5f_1(y_1^{n}, y_2^{n}, t_n) - 16f_1(y_1^{n+1}, y_2^{n+1}, t_{n+1}) + 23f_1(y_1^{n+2}, y_2^{n+2}, t_{n+2}),$$

$$y_2^{n+3} = y_2^{n+2} + \frac{h}{12}(5f_2(y_1^{n}, y_2^{n}, t_n) - 16f_2(y_1^{n+1}, y_2^{n+1}, t_{n+1}) + 23f_2(y_1^{n+2}, y_2^{n+2}, t_{n+2}).$$

The computation of the manufactured solution is done at $t \in [1, 2]$. When we compute the 2-step of AB-method, we require a special initialization for time-stepping algorithm at n=1. For this, the AB-1 method is used. For 3-step AB-method, the initializations of the time-stepping at n=1 and n=2 are done using Runge-Kutta-2 method. We make sure that these initializations produce at least 2nd-order of accuracy or higher hence RK-2 is chosen. RK method is likely chosen to solve ODEs because it is an accurate, stable and easy to program (Senthilnathan, 2018). The general form of Runge-Kutta-2 schemes for the test problem can be expressed in the form of

$$K_1^{n+1} = hf_1(y_1^{n}, y_2^{n}, t_n)$$

$$L_1^{n+1} = hf_2(y_1^{n}, y_2^{n}, t_n)$$

$$K_2^{n+1} = hf_1(y_1^{n}+0.5K_1^{n+1}, y_2^{n}+0.5L_1^{n+1}, t_n+0.5h)$$

$$L_2^{n+1} = hf_2(y_1^{n}+0.5K_1^{n+1}, y_2^{n}+0.5L_1^{n+1}, t_n+0.5h)$$

$$y_1^{n+1} = y_1^{n} + K_2^{n+1}$$

$$y_2^{n+1} = y_2^{n} + L_2^{n+1}$$

Similar to the previous test problem, we also need special initializations for the 2-step of AB method and the 3-step of AB method of our manufactured solution. We will use exactly the same way as has been implemented for the test problem. We will compute the AB-2 method with the initialization of AB-1 method for time-stepping algorithm at n=1 and the RK-2 method is used as the initialization at n=1 and n=2 when computing the 3-step AB-method.

Numerical Results: Test Case II

i) Error Analysis

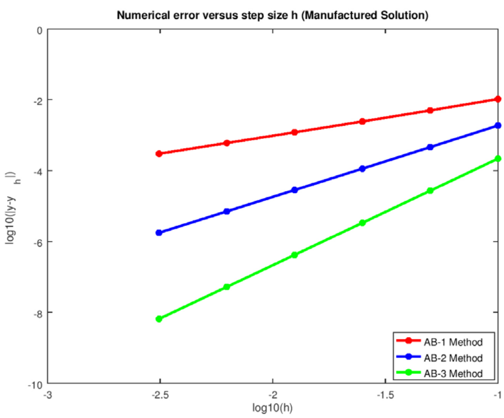

Figure 3: Numerical Error versus Step Size h Used to Compute the Manufactured Solution

Figure 3 shows the numerical error versus step size h for the manufactured solution. According to Figure 3, the numerical errors of 1-, 2- and 3-step of AB methods of the manufactured solution decline as expected as the step size h decreases. At step size h=0.1, the errors with magnitude 9.13x10^{-3} and 4.47x10^{-4} for the AB methods of order 1 and 3, respectively.

Based on the blue line on Figure 3 which represents the AB-2 method, the numerical error is 1.14x10^{-4} at h=0.025 and 7.08x10^{-6} at h=0.00625. For h=0.0125, the AB-1 method and AB-2 method produces the error of about 1.21x10^{-3} and 2.84x10^{-5}, respectively while the AB-3 method produces 4.2x10^{-7}, which is the smallest error. The AB-1, -2 and -3 methods produce the numerical error of 1.04x10^{-2}, 1.88x10^{-3} and 2.18x10^{-4} respectively, when the step size h is set to be 0.1. Hence, we can conclude that AB-3 method produces the smallest magnitude of error.

ii) Efficiency Analysis

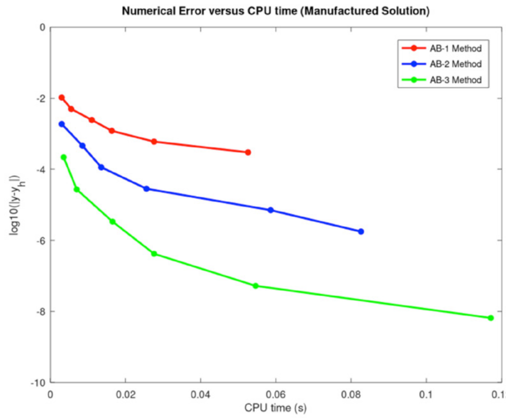

Figure 4 : Numerical Error versus CPU Time Required to Compute The Manufactured Solution

Figure 4 shows the plot of numerical error for AB-methods versus the CPU time required to complete the calculations. The higher the numerical error of 1-, 2- and 3-step of AB methods, the smaller CPU times are required. The AB-1 and -2 method produce a CPU time of 5.25x10^{-2}s and 8.26x10^{-2}s, respectively, while the AB-3 method requireds 1.17x10^{-1}s of CPU time at step size h=0.003125. At h=0.0125, the AB-1, -2 and -3 methods has the CPU times of 1.63x10^{-2}s, 2.55x10^{-2}s and 2.75x10^{-2}s, respectively. By referring to the Figure 4, the green line represents the CPU time of the AB-3 method. The CPU time of about 7x10^{-3}s and 5.46x10^{-2}s at step size h=0.5 and h=0.00625. As a conclusion, the AB-3 method is the most efficient method among of all the three methods.

We consider y is the exact solution. Then, we denote y_h as the numerical approximation where it depends on small parameter h which is the step size. Here we denote κ is the convergence rate of a numerical method. So, the relationship between the numerical errors can be expressed in the form as follows:

$$\kappa = \frac{\log_{10}\left(\frac{|y - y_h|}{\left|y - y_{\frac{h}{2}}\right|}\right)}{\log_{10}(2)}.$$

Based on the graph in Figure 1, the convergence rate for the test problem can be obtained by calculating the gradient of the graph for each method. In the end, we obtain the rate of convergence of the AB-1 method is 1 while the AB-2 method is 2 and for the AB-3 method, we obtain a convergence rate of 3.

The similar pattern occurs in the manufactured problem as in the test problem. We can refer to the Figure 4 to verify the rate of convergence of 1-, 2- and 3-step of AB-methods for the manufactured solution. As a result, we obtain an order of convergence of 1 for the AB-1 method, 2 for AB-2 method and order of 3 for the AB-3 method. We have shown that our numerical method reproduces the theoretical order of convergence in both test cases.

Conclusion

In this research, we focused on the numerical solution of high-order nonlinear ODE which was converted to a system of 1st-order ODE. For the numerical illustration, we used two test cases which were a test problem and a manufactured solution. The numerical methods that had been chosen was 1-, 2- and 3-step of the Adams-Bashforth schemes. To solve the test cases, Octave programming was used to implement the numerical schemes that had been constructed earlier. The results that were produced are the global errors, the rate of convergence and the CPU time and these numerical results depend heavily on the step size h. The results of numerical computations were showcased and we can see that there are four graphs that explain the numerical error against the step size h and the numerical error against the CPU time for each test cases.

According to numerical results obtained, we can see that the numerical error and the CPU time are closely related to the step size h. For both test cases, as the step size decreases, the numerical error of each method decreases as well while the CPU time increases. We also noted that the higher the order of the time stepping methods (linear multistep method), the more efficient they are and less error they produce. In this research project, we conclude that the AB-3 method is the most efficient method to solve high-order nonlinear ODE. No doubt, the AB-3 method has a high order of convergence which is 3. While conducting this research, we learnt that the Adams-Bashforth methods are fully explicit methods and therefore they are more efficient than implicit methods in term of shorter computational time. These methods are also very easy to derive as they do not require Newton's method to solve the arised fixed point problem occurring in the implicit methods.

Based on the discussions above, a few suggestions can be considered. First, we can try to solve other nonlinear ODEs by using other numerical methods such as Backward Euler and Adams-Moulton method. Though they are more complicated and require Newton's method, these methods can handle stiff ODE better. We can implement other types of initialization for the AB-3 method such as the RK-4 method or using Richardson's extrapolation but these have not been attempted in this project. Last but not least, we hope that the AB-3 method can be used to showcase its fullest potential to solve more practical problem such that system of ODE governing physical problem or many models found in real application. As we can see, AB-3 method is the most efficient method, and this has been proven in this project. Higher-order method such as AB-4 and AB-5 are likely to be more efficient. This can be a potential future work of this final year project.

Acknowledgements

Lab CERMAT which is under the administration of School of Informatics and Applied Mathematics, Universiti Malaysia Terengganu is kindly acknowledged for providing high performance computing facility to conduct this research.

References

Ameen and Novati, P. (2016). The solution of fractional order epidemic model by implicit Adams methods. *Applied Mathematical Modelling*. 10(43): 78-84.

Beeman, D. (1975). Some Multistep Methods for Use in Molecular Dynamics Calculations. *Journal of Computational Physics*. 6(20): 130-139.

Dimitrov, D.T. and Kojouharov, H.V. (2005). Positive and elementary stable nonstandard numerical methods with applications to predator-prey models. *Journal of Computational and Applied Mathematics* 2(189): 98-108.

Enatsu, Y., Nakata, Y. and Muroya, Y. (2010). Global Stability for A Class of Discrete SIR Epidemic Models. *Mathematical Biosciences and Engineering* 7(2): 347-361.

Garrappa, R. (2007). On some explicit Adams multistep methods for fractional differential equations. *Journal of Computational and Applied Mathematics*. 11(229): 392-399.

Hurvoka, J. (2013). *Predator-Prey Model with Ratio-dependent Food Processing Response*. United States: Department of Mathematics and Statistics, University of Minnesota Duluth.

Junior, C.F.L, Silva, G.S.D, Mendes, P.S.N., Machado, V.A.D.S., Ribeiro, B.N.M. (2018). Implementation of Numerical Methods of Euler and Runge-Kutta through MATLAB Software for the Solution of Ordinary Differential Equations Dedicated to Teaching. *Journal of Engineering Research and Application*, 2248-9622.

LeVeque, R.J. (2004). *Finite Difference Methods for Differential Methods*. USA: University of Washington.

Senthilnathan, C. (2018). A Numerical Solutions Of Initial Value Problems (Ivp) For Ordinary Differential Equations (Ode) with Euler and Higher Order of Runge Kutta Methods Using Matlab. *International Journal of Engineering Science Invention (IJESI)*, 2319-6734.

Tomas, B.C. (2013). *Methods of Applied Mathematics for Engineers and Scientist*. Cambridge University Press. USA: Michigan Technology University.