Journal of the Nigerian Mathematical Society
Vol. 35, pp. 424-438, 2016

SOLVING IVPs IN ODEs BY USING SOME $\mathcal{L}$-STABLE METHODS IN VARIABLE STEP-SIZE FORMULATION

GURJINDER SINGH, V. KANWAR[1] AND SAURABH BHATIA

ABSTRACT. In this paper, we propose a one-parameter family of the $\mathcal{L}$-stable modified trapezoidal method for solving numerically initial value ordinary differential equations (ODEs). The proposed family has second algebraic order of convergence and is $\mathcal{L}$-stable. Further, variable step-size formulation of the proposed methods is considered as embedded-type methods. A comparison of numerical results made by the proposed methods and by the existing classical ODE solver is given.

Keywords and phrases: Ordinary Differential Equations; Initial Value Problems; Stability
2010 Mathematical Subject Classification: 65Lxx, 65L05, 65L20

1. Introduction

The present article is concerned with the following initial value problem (IVP)

$$y'(x) = f(x, y(x)); \quad y(a) = y_0, \quad x \in [a, b], \tag{1}$$

where we assume for now $y, f \in \mathbb{R}$. It is assumed that the initial value problem (1) is well-posed, that is, it has a unique continuously differential solution, say $y(x)$. To solve numerically the given IVP (1), we proceed as usual by discretizing the interval $[a, b]$ as follows: $a = x_0 < x_1 < x_2 < \cdots < x_N = b, \quad x_n = x_0 + nh, \quad n = 0, 1, 2, 3, \ldots, N, \ h = \frac{(b-a)}{N}$. Here h is called the step-size, grid-size or mesh-size, which may be constant or variable along the integration interval. In this article, we first derive the numerical methods with constant step-size and then consider their variable step-size formulation.

Different codes are available in the literature to cope efficiently with initial-value problems having different characters, for example, stiff, singular and singularly-perturbed etc. But, it is also true that a single code cannot cope efficiently with each type of initial-value problems. This is a major reason behind the continuous development of different codes having different features according to the problem to deal with. In

Received by the editors December 02, 2015; Revised: August 04, 2016; Accepted: September 07, 2016
[1]Corresponding author

virtually, all modern codes for ODEs, the step-size is selected automatically to achieve reliability and efficiency [1]. Practically speaking, any discretization method with constant step-size performs poorly if the solution varies rapidly in some parts of the integration interval and slowly in other large parts of the integration interval. As some authors have remarked, to be efficient, an integrator based on a particular formula must be suitable for a variable step-size formulation [2-3].

There are different situations where low accuracy methods are needed, for example, real time simulation and time integration of partial differential equations, etc. As Fehlberg has remarked in the NASA report [4]

... low order Runge-Kutta formulas are of interest in some heat transfer problems. It is well-known that the parabolic partial differential equations of such problems can be reduced to ordinary differential equations. For instance, by a discretization of the space variable(s) of the problem, we obtain a system of ordinary differential equations with the time as the independent variable. Such a system can be integrated by Runge-Kutta methods.
However, it is also well-known that the application of Runge-Kutta methods to such problems is often very time consuming. Higher order Runge-Kutta formulas do not offer advantages in this respect, since stability considerations, resulting from the exponential character of the solution, exclude an increase of the intgration step-size that would make such high order formulas meaningful. Therefore, low order Runge-Kutta formulas (second or third order) can be expected to solve such system more efficiently than any high order formula. On the other hand, they are potentially more efficient than the standard difference formulas obtained by discretization of the space variable(s) as well as the time variable.
For more details, see for references [4-6].

The aim of this article is to derive variants of the low accuracy modified trapezoidal method [7] and to show their importance in variable step-size mode. Some useful references can be found in [1-37].

The paper is organized as follows: In section 2, derivation of the methods is considered. In section 3. error analysis of proposed methods is carried out and linear stability analysis is considered in section 4. Section 5. is concerned with the formulation of the proposed methods in variable step-size mode. Applicability of the proposed methods to a system of first order ODEs is discussed in section 6. In section 7, some implementation details of the methods are considered and numerical results are presented in section 8. Finally, some conclusions are given in section 9.

2. Derivation

In [8], a one-parameter class of the implicit Euler's method is proposed for solving numerically initial value problems (1). We will use this class of methods to modify the method given in [7]. Here, we briefly recall derivation of the class of implicit Euler's method [8].

Let y_n denotes the numerical approximation to the theoretical solution $y(x)$ of (1) at the grid point $x = x_n$ and consider the following function

$$I(x) = e^{\alpha x}(\beta x + \gamma), \tag{2}$$

where $\alpha, \beta,\ \gamma \in \mathbb{R}$.
Since there are three unknowns in (2), therefore for obtaining a numerical method with one free parameter, we impose the following interpolatory conditions

$$y(x_{n+1}) = I(x_{n+1}),\ y(x_n) = I(x_n),\ y'(x_{n+1}) = I'(x_{n+1}). \tag{3}$$

Using these interpolatory conditions, the following class of numerical methods [8] is obtained

$$y_{n+1} = y_n + h(1 - \alpha h)y'_{n+1}, \tag{4}$$

where $y_{n+1} \simeq y(x_{n+1)},\ y_n \simeq y(x_n)$ and $y'_{n+1} \simeq f(x_{n+1}, y_{n+1})$.
This is a one-parameter family of the implicit Euler's method [8]. By putting $\alpha = 0$ in (4), the classical implicit Euler's method can be obtained.

Note. It must be corrected here that the above family of methods is $\mathcal{A}$-stable (also $\mathcal{L}$-stable) for $\alpha \leq 0$. In this case, there is no restriction on the step-size. This is the reason that such type of methods are suitable for solving stiff systems. But, in the article [8], it is bad written that this class of methods is $\mathcal{L}$-stable for $|\alpha h| \ll 1$.

In [7], the following modification of the classical trapezoidal method is proposed

$$y_{n+1} = y_n + \frac{h}{2}[f(x_n, \hat{y_n}) + f(x_{n+1}, y_{n+1})], \tag{5}$$

where $\hat{y_n} = y_{n+1} - hf(x_{n+1}, y_{n+1})$ is obtained from the classical Euler's formula when applied backward at $x = x_{n+1}$ in the negative x-direction. The classical trapezoidal method has second algebraic order of convergence and is $\mathcal{A}$-stable. The modified trapezoidal method (5) has second algebraic order of convergence and is $\mathcal{L}$-stable, that is, in this way the classical method switched from $\mathcal{A}$-stable to $\mathcal{L}$-stable.

We further modify the method given in (5) by using the modified value of $\hat{y_n} = y_{n+1} - h(1 - \alpha h)y'_{n+1}$ obtained from (4) in (5). In this way, we obtain the following expression

$$y_{n+1} = y_n + \frac{h}{2}[f(x_n, \hat{y_n}) + f(x_{n+1}, y_{n+1})], \tag{6}$$

where $\hat{y_n} = y_{n+1} - h(1-\alpha h)y'_{n+1}$.
This is a new one-parameter family of the modified trapezoidal method (5) for solving (1) numerically. By putting $\alpha = 0$ in (6), the classical modified trapezoidal method (5) can be obtained.

3. Error Analysis

It is very practical and necessary to know how the local errors behave in the implementation of any numerical method. In this section, we will present error analysis of the proposed methods (6).

Consider the family given in (6) in the following difference operator form

$$\mathcal{L}[z(x);h] = z(x+h) - z(x) - \frac{h}{2}\left[\hat{z}'(x) + z'(x+h)\right], \tag{7}$$

where $z(x)$ is an arbitrary analytic function defined on $[a,b]$. Expanding the above expression by Taylor series about x and collecting terms in h, after substituting $z(x)$ by the solution $y(x)$ of (1) and x by x_n, one can obtain the following local truncation error for the methods (6) as follows

$$LTE = \left(\frac{-y_n^{(3)} + (3y''_n - 6\alpha y'_n)g_n}{12}\right)h^3 + \mathcal{O}(h^4). \tag{8}$$

where y''_n and $y_n^{(3)}$ denote the numerical approximations to the second and third order derivatives of $y(x)$ at the grid point x_n respectively and $g_n = \frac{\partial f}{\partial y}$ at x_n. Hence, the family given in (6) has second algebraic order of convergence.
The LTE given in (8) is not very practical due to the presence of second and third order derivatives. But for the autonomous case $y'(x) = f(y(x))$, it may be simplified yielding

$$LTE = \frac{1}{12}\left((3 - 6\alpha + f_y)ff_y - f_{yy}f^2\right)h^3 + \mathcal{O}(h^4), \tag{9}$$

where f_y and f_{yy} are the usual notations for partial derivatives of f w.r.t. y at $x = x_n$. The principal term in (9) could be used as an estimate of the local truncation error.

4. Linear stability analysis

The linear stability analysis of the above schemes is examined as usually by applying them to the Dahlquist's test problem

$$y'(x) = \lambda y(x),\ \lambda < 0. \tag{10}$$

The true solution of this problem is $y(x) = e^{\lambda x}$ and will be damped out as $x \to \infty$. It is expected that application of a given numerical method to this problem has the similar behavior as the true solution of the problem. Since, the family of methods (6) contains a one free parameter, therefore it will be convenient to use the Routh-Hurwitz criterion to carry out the linear stability analysis of the methods.

Theorem 4.1. *The family of modified trapezoidal method given in* (6) *is $\mathcal{L}$-stable for $\alpha \leq 0$.*

Proof. By applying the family of methods given in (6) to the Dahlquist's test equation (10), the following difference equation is obtained

$$y_{n+1} = y_n + \frac{\bar{h}}{2}\left[2 - (1-\alpha h)\bar{h}\right] y_{n+1}, \tag{11}$$

where $\bar{h} = \lambda h$ and $\bar{h} < 0$.

Therefore, the stability function of the proposed family of methods (6) is given by

$$R(\bar{h}) = \frac{2}{2 - 2\bar{h} + k\bar{h}^2}, \tag{12}$$

where $k = 1 - \alpha h$.

Now, consider the corresponding characteristic equation

$$\pi(\xi) = (2 - 2\bar{h} + k\bar{h}^2)\xi - 2 = 0. \tag{13}$$

By using the substitution $\xi = \dfrac{1+z}{1-z}$, we have the following transformed equation

$$\hat{\pi}(z) = (2 - 2\bar{h} + k\bar{h}^2)(1+z) - 2(1-z) = 0. \tag{14}$$

After collecting the coefficients of z in (14), we have

$$\hat{\pi}(z) = (4 - 2\bar{h} + k\bar{h}^2)z + (k\bar{h}^2 - 2\bar{h}) = 0. \tag{15}$$

Then, the Routh-Hurwitz criterion will be satisfied for $\alpha \leq 0$. Hence, the absolute stability interval of the proposed family of methods (6) is $(-\infty, 0)$ for $\alpha \leq 0$.

Further, if we consider λ is a complex number with $\mathrm{Re}(\lambda) < 0$, then it can be verified that $|R(\bar{h})| \to 0$ as $\mathrm{Re}(\bar{h}) \to -\infty$, for $\bar{h} = \lambda h$ and $\mathrm{Re}(\bar{h}) < 0$. Therefore, the family of methods (6) is $\mathcal{L}$-stable. □

Note. The class of methods given in (6) has one free parameter $\alpha \in \mathbb{R}$. Following the stability considerations, one must have to choose the value of the free parameter α such that $\alpha \leq 0$. In the numerical experiments, the numerical methods for different values of the free parameter are tested on several IVPs having different behavior of solutions.

5. Formulation in variable step-size mode

In the previous sections, the methods given in (6) have been considered using a fixed step-size h. As some authors have remarked, to be efficient, an integrator must be suitable in variable-step size formulation [2-3]. The formulation in variable step-size mode of the methods given in (6) is considered as follows. The methods given in (6) may be formulated as embedded pairs by combining with the forward Euler's method. In embedded pairs, the values needed by the method of less order must be used by the other method also, and so there is no additional cost in the computation of these values. The lower order method is just used to

estimate the local error at each step and higher order method is used to advance the integration process. We will follow Shampine *et. al.* [9] to formulate the given methods (6) in variable step-size mode. Here, we will discuss the process of variable step-size formulation in general, that is, for methods of order p and order $p+1$ as discussed in [9]. Assume that the local error in using the method y_{n+1} of order p is given by

$$le_n = y(x_n + h) - y_{n+1}, \tag{16}$$

where $y(x)$ is the theoretical solution.
Now, if we apply a method of order $p+1$ to compute a result y^*_{n+1} on this step. Then we have

$$\begin{aligned} est &= y^*_{n+1} - y_{n+1} \\ &= [y(x_n + h) - y_{n+1}] - [y(x_n + h) - y^*_{n+1}] \\ &= le_n + \mathcal{O}(h^{p+2}). \end{aligned} \tag{17}$$

This is computable estimate of the local error of the lower order method because le_n is $\mathcal{O}(h^{p+1})$ and so dominates in (17) for small enough values of h. It must be mentioned here that we can estimate the error in y_{n+1} by comparing it to the more accurate solution y^*_{n+1}. But the trick in embedded pairs to make local error estimation practical is to find a pair of methods that share as many function evaluations as possible. In this way of proceeding, we do not know precisely how small the local error is at each step of integration, but we believe that it is rather smaller than the estimated local error [9].
A local error tolerance *tol* is specified and, if the estimated error is too large relative to this tolerance, the step is rejected and another attempt is made with a smaller step-size.
Now, we discuss the strategy that how to change the step-size. From (16), we have

$$y(x + h) - y_{n+1} = h^{p+1}\phi(x_n) + \mathcal{O}(h^{p+2}). \tag{18}$$

If we were try to take a step from x_n with a new step-size σh, then the error would be

$$\begin{aligned} (\sigma h)^{p+1}\phi(x_n) + \mathcal{O}((\sigma h)^{p+2}) &= \sigma^{p+1}h^{p+1}\phi(x_n) + \mathcal{O}(h^{p+2}) \\ &= \sigma^{p+1}est + \mathcal{O}(h^{p+2}). \end{aligned} \tag{19}$$

The largest step-size that we predict will pass the error test corresponds to choosing σ so that

$$|\sigma^{p+1}est| \approx tol.$$

Then the new step-size is given by

$$h\left(\frac{tol}{|est|}\right)^{1/(p+1)}. \tag{20}$$

Many authors, based on extensive numerical experimentations recommend the inclusion of a safety factor ν in (20) as follows

$$h_{new} = \nu h \left(\frac{tol}{|est|} \right)^{1/(p+1)}, \tag{21}$$

where ν is a suitable adjustment factor $\nu \approx 0.9$ [10]. Here, p is the order of the lower order method, and $0 < \nu < 1$ is a safety factor whose purpose is to avoid failed steps.
Normally some restrictions must be considered in order to avoid large fluctuations in step-size: The step-size is allowed to remain in the following limits

$$h_{mini} \leq h_{new} \leq h_{maxi},$$

where h_{mini} and h_{maxi} are the allowed minimum and maximum step-sizes, respectively. This strategy is applied successively to predict the step-size for the next step after a successful step, i.e. when $\|est\| < tol$.

There are different strategies for selecting the size of the initial step, that we call h_{ini}, (see [13-14]), we can simply take a very small starting step value as in [22], and then the algorithm will correct this value if necessary, according with the step-size strategy.

6. Applicability to differential systems

The above integrators may be applied to a system of first-order ordinary differential equations. Consider a system of m equations, which may be written in vector form as

$$\mathbf{y}' = \mathbf{f}(x, \mathbf{y}), \quad \mathbf{y}(a) = \mathbf{y}_0, \qquad a \leq x \leq b$$

where
$\mathbf{y} = (y_1, \ldots, y_m)^T, \mathbf{f}(x, \mathbf{y}) = (f_1(x, y_1, \ldots, y_m), \ldots, f_m(x, y_1, \ldots, y_m))^T$
and $\mathbf{y}_0 = (y_{1,0}, \ldots, y_{m,0})^T$.
The numerical integrators given in (6) for scalar equations being one-step methods, may be written as

$$y_{n+1} = y_n + h\, \Phi_f(x_n, y_n, y_{n+1}, h),$$

where $\Phi_f(x_n, y_n, y_{n+1}, h)$ is the incremental function, and the subscript f on the right-hand side indicates that the dependence of Φ_f on its variables is through the function f. Applying this method to each of the scalar equations in the differential system results in

$$\mathbf{y}_{n+1} = \mathbf{y}_n + h\, \Phi(x_n, \mathbf{y}_n, \mathbf{y}_{n+1}, h)$$

where

$$\Phi(x_n, \mathbf{y}_n, \mathbf{y}_{n+1}, h) = (\Phi_{f_1}, \ldots, \Phi_{f_m})^T.$$

Here

$$\Phi_{f_i} = \Phi_{f_i}(x_n, y_{1,n} \ldots, y_{m,n}, y_{1,n+1} \ldots, y_{m,n+1}, h), \ i = 1, 2, \ldots, m.$$

For the stability analysis of the methods given in (6) for the system of first-order differential equations it is sufficient to consider the following system (see [23])

$$\mathbf{y}' = \mathbf{\Lambda}\,\mathbf{y}\,, \qquad \mathbf{\Lambda} = \mathrm{diag}(\lambda_1, \lambda_2, ..., \lambda_m)\,.$$

If we apply the integrator given in (6) to this test equation we get

$$\mathbf{y}_{n+1} = \mathbf{R}\,\mathbf{y}_n\,, \qquad \mathbf{R} = \mathrm{diag}(R_1, R_2, ..., R_m).$$

This decouples into m independent numerical methods, one for each component, where each R_i is a rational function of the type in (12). For this reason the stability considerations can be restricted to a scalar test equation.

7. Implementation details

The methods given in (6) are of implicit nature. They can be used as an iteration method by using some form of Newton-Raphson's iteration method or by some of its variants for solving the resulting equation on each step. But, generally it is more efficient to use the methods like in (6) in predictor-corrector mode by using a suitable predictor. In the present article, the forward Euler's method is used as a predictor and the methods given in (6) are used as correctors. Recall that, the forward Euler's method is also used to estimate the local error at step of integration as we have formulated in section 5.

8. Numerical results

In this section, some numerical experiments have been presented by implementing the proposed methods (6) and the exiting classical method (5). The numerical experiments have been performed on several IVPs having different behavior of solutions. The computational work has been done using Matlab version 7.9.0.529 (R2009b) on a personal computer (32 bit operating-system). In the following tables, the notations Meth-1 and Meth-2 are used to denote the new method (6) and the classical method (5) respectively. Further, the notation $MaxErr(y(x))$ stands for the maximum absolute error along the integration interval and whereas $E(x_N)$ denotes the absolute error at the final point of integration interval in approximating the true solution $y(x)$.

Implementation with constant step-size

Firstly, we present some numerical experiments by implementing both the methods, that is, Meth-1 and Meth-2 with constant step-size.

8.1. **A nonlinear problem.** First consider a nonlinear IVP given by

$$y'(x) = (\cos(y(x)))^2, \quad x \in [0, 1]$$

with $y(0) = \dfrac{\pi}{4}$. The true solution of the problem is

$$y(x) = \arctan(1 + x).$$

For this problem, the classical method (5) and a new proposed method (6) for $\alpha = \dfrac{-19}{20}$ are considered for comparison. In Table 1, absolute errors at the final point $x_N = 1$ and maximum absolute errors along the integration interval are considered. Different step-sizes $h = \dfrac{1}{2^j}, j = 8, 9, 10, 11$ have been used for obtaining numerical results. It can be observed from Table 1 that the proposed method (6) has a good performance as compared with the existing method (5).

TABLE 1. Data for problem 8.1

h	$Method$	$E(y(x_N))$	$MaxErr(y(x))$
$\dfrac{1}{2^8}$	Meth-1	3.4253×10^{-7}	3.8191×10^{-7}
	Meth-2	1.6713×10^{-6}	1.6744×10^{-6}
$\dfrac{1}{2^9}$	Meth-1	8.4941×10^{-8}	9.4790×10^{-8}
	Meth-2	4.1707×10^{-7}	4.1783×10^{-7}
$\dfrac{1}{2^{10}}$	Meth-1	2.1148×10^{-8}	2.3611×10^{-8}
	Meth-2	1.0417×10^{-7}	1.0436×10^{-7}
$\dfrac{1}{2^{11}}$	Meth-1	5.2764×10^{-9}	5.8921×10^{-9}
	Meth-2	2.6031×10^{-8}	2.6078×10^{-8}

8.2. **A nonlinear problem.** Now, again consider another nonlinear IVP given by

$$y'(x) = \frac{1}{y(x)}, \quad x \in [0, 2]$$

with $y(0) = 1$. The true solution of the problem is

$$y(x) = \sqrt{2x + 1}.$$

For this problem, the classical method (5) and a new proposed method (6) for $\alpha = \dfrac{-1}{2}$ are considered for comparison. In Table 2, absolute errors at the final point $x_N = 2$ and maximum absolute errors along the integration interval are considered. Different step-sizes $h = \dfrac{1}{2^j}, j = 8, 9, 10$ have been used for obtaining numerical results. It can be observed from Table 2 that the proposed method (6) performs better as compared with the existing method (5).

TABLE 2. Data for problem 8.2

h	$Method$	$E(y(x_N))$	$MaxErr(y(x))$
$\frac{1}{2^8}$	Meth-1	5.1092×10^{-9}	4.3323×10^{-7}
	Meth-2	1.3703×10^{-6}	1.4749×10^{-6}
$\frac{1}{2^9}$	Meth-1	1.6495×10^{-9}	1.0777×10^{-7}
	Meth-2	3.4189×10^{-7}	3.6790×10^{-7}
$\frac{1}{2^{10}}$	Meth-1	4.5729×10^{-10}	2.6878×10^{-8}
	Meth-2	8.5386×10^{-8}	9.1871×10^{-8}

Implementation in variable step-size mode

Now, we present some numerical experiments by implementing both the methods, that is, Meth-1 and Meth-2 with variable step-sizes. For both the methods, Meth-1 and Meth-2, the forward Euler's method has been used to estimate the local error at each step. At each step of integration, the local error $E(i)$ in the i^{th} component of the solution is estimated and is required to be less than or equal to the acceptable tolerance, which is a function of two user-defined tolerances $RelTol$ (Relative error tolerance) and $AbsTol$ (Absolute error tolerance). The following criterion for controlling the local error at each integration step has been used for both the methods

$$E(i) \leq \max(RelTol * \text{abs}(y(i)), AbsTol),$$

where $RelTol$ and $AbsTol$ are positive real numbers.

8.3. **A stiff problem.** Consider a stiff IVP

$$y'(x) = 49e^{-50x} - y(x), \quad x \in [0, 1]$$

with $y(0) = 1$. The true solution of the problem is

$$y(x) = 2e^{-x} - e^{-50x}.$$

For this problem, the classical method (5) and new proposed method (6) for $\alpha = \frac{-11}{5}$ are considered for comparison. In Table 3, absolute errors at the final point $x_N = 1$ are considered. Different values for $h_{ini} = AbsTol = RelTol = k$ have been used for obtaining numerical results. It can be observed from Table 3 that the proposed method (6) has a good performance as compared with the existing method (5) in terms of accuracy and less number of function evaluations.

TABLE 3. Data for problem 8.3

k	N	$Method$	$E(y(x_N))$
$\frac{1}{100}$	19	Meth-1	1.8106×10^{-3}
	20	Meth-2	1.1474×10^{-2}
$\frac{1}{1000}$	58	Meth-1	2.8961×10^{-4}
	59	Meth-2	1.4933×10^{-3}
$\frac{1}{10000}$	185	Meth-1	4.5201×10^{-5}
	187	Meth-2	1.7327×10^{-4}

8.4. **A stiff system.** Here, we consider the following system of first order ODEs

$$\begin{aligned} y_1'(x) &= \frac{y_1(x)}{y_2(x)} - 2y_1(x) - e^{-x}; \qquad && y_1(0) = 1 \\ y_2'(x) &= -y_2(x); && y_2(0) = 1. \end{aligned}$$

on the interval $[0, 1]$. The true solution of the system is

$$\begin{cases} y_1(x) = e^{-2x} \\ y_2(x) = e^{-x}. \end{cases} \tag{22}$$

For this problem, the Meth-1, that is, the method for $\alpha = \frac{-100}{101}$ in (6) is considered. Absolute errors at the final point $x_N = 1$ are reported in Table 4. For this problem, different values for $h_{ini} = AbsTol = RelTol = k$ have been used. Table 4 is a numerical evidence which demonstrates the better performance of the proposed method compared with the existing method.

TABLE 4. Data for problem 8.4

k	N	$Method$	$E(y_1(x_N))$	$E(y_2(x_N))$
$\frac{1}{100}$	12	Meth-1	9.7181×10^{-4}	1.2416×10^{-3}
	13	Meth-2	4.0857×10^{-3}	2.8583×10^{-3}
$\frac{1}{1000}$	35	Meth-1	7.8214×10^{-5}	9.4051×10^{-5}
	36	Meth-2	4.1005×10^{-4}	3.0738×10^{-4}
$\frac{1}{10000}$	105	Meth-1	7.5261×10^{-6}	8.4913×10^{-6}
	105	Meth-2	4.0887×10^{-5}	3.1712×10^{-5}

8.5. **A linear system.** As a last example, consider the following system of first order ODEs

$$\begin{aligned} y_1'(x) &= -100y_1(x) + 9.901y_2(x); \qquad & y_1(0) &= 1 \\ y_2'(x) &= 0.1y_1(x) - y_2(x); & y_2(0) &= 1. \end{aligned}$$

on the interval $[0, 1]$. The theoretical solution of the system is

$$\begin{cases} y_1(x) = e^{-0.99x} \\ y_2(x) = 10e^{-0.99x}. \end{cases} \tag{23}$$

For this problem, Meth-1, that is, the method for $\alpha = \dfrac{-13}{14}$ in (6) is considered. Absolute errors at final point $x_N = 1$ are reported in Table 5. For this problem, different values for $h_{ini} = AbsTol = RelTol = k$ have been used. The numerical results given in Table 5 show the good performance of the proposed method.

TABLE 5. Data for problem 8.5

k	N	$Method$	$E(y_1(x_N))$	$E(y_2(x_N))$
$\frac{1}{100}$	53	Meth-1	3.1162×10^{-2}	6.5789×10^{-3}
	56	Meth-2	2.5854×10^{-2}	1.4525×10^{-2}
$\frac{1}{1000}$	79	Meth-1	1.5593×10^{-3}	4.5270×10^{-4}
	79	Meth-2	2.0463×10^{-3}	1.3188×10^{-3}

9. Concluding Remarks

This article proposes some novel $\mathcal{L}$-stable variable step-size methods for solving numerically initial value problems (1). The aim of this article is to provide alternatives of the existing $\mathcal{L}$-stable method showing their good performance in variable step-size formulation. A comparison of numerical results obtained by the proposed methods and by the existing ODE solver is presented. The numerical results show the good performance of the proposed methods over the existing method considered for comparison. These methods are good alternatives to the existing method in the literature and may be used to solve numerically a scalar ODE and a system of first order ODEs.

ACKNOWLEDGEMENTS

The authors would like to thank the anonymous referees for their comments which improved the original version of this manuscript. We are also pleased and thankful to the authority of TEQIP II of University Institute of Engineering and Technology, Panjab University, Chandigarh

for the financial support under the project entitled "New Numerical Schemes to Solve Initial Value Systems of Ordinary Differential Equation" vide no. 1059 TEQIP-II/UIET, dated 26/5/2014.

References

[1] Uri. M. Ascher and Linda. R. Petzold, *Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations*, Siam, 1998.

[2] J. D. Lambert, *Numerical Methods for Ordinary Differential Systems: The Initial Value Problem*, first ed., John Wiley, New York, 1991.

[3] E. Hairer, S.P. Nörsett and G. Wanner, *Solving Ordinary Differential Equations-I*, Springer, Berlin, 1993.

[4] E. Fehlberg, *Low order classical Runge-Kutta formulas with step-size control and their application to some heat transfer problems*, Nasa Technical Report No. 315, George C. Marshall Space Flight Center, Hurtsville, 1969

[5] C. W. Gear, *Simulation: Conflicts Between Real Time and Software* in J. R. Rice (ED.), Mathematical Software III, Academic Press, New York, 1977.

[6] C. W. Gear, *Numerical solution of ordinary differential equations: is there anything left to do?*, SIAM, Rev. **23** 10-24, 1981.

[7] M.M. Chawla and M.A. AL-Zanaidi, *New L-stable modified trapezoidal formulas for the numerical integration of* $y' = f(x, y)$, Int. J. Comp. Maths. **63** 279-288, 1997.

[8] G. Singh, V. Kanwar and S. Bhatia, *A class of the backward Euler's method for initial value problems,* Res. Jour. Engg. Tech. **6** (1) 207-211, 2015.

[9] L. F. Shampine, I. Gladwell and S. Thompson, *Solving ODEs with Matlab*, Cambridge University Press, 2003.

[10] J. Stoer and R. Bulirsch, *Introduction to Numerical Analysis,* Springer, 2002.

[11] G. Dahlquist, *A special stability problem for linear multi-step methods,* BIT, **3** 27-43, 1963.

[12] C. C. Jibunoh, *An exponential method for accurate and automatic integration of nonlinear (stiff and non-stiff) ODE systems,* J. Nigerian. Math. Soc. **34** (2) 143-159, 2015.

[13] C. C. Jibunoh, *Accurate and automatic integration of stiff and non-stiff (ODE) systems via spectral decomposition,* J. Nigerian. Math. Soc. **29** 21-39, 2010.

[14] R. I. Okuonghae and M. N. O. Ikhile, *On the construction of high-order* $\mathcal{A}(\alpha)$*-stable hybrid linear multi-step methods for stiff IVPs in ODEs,* Num. Ana. Appl. **5** (3) 231-241, 2012.

[15] F. R. Villatoro, *Given a one-step numerical scheme, on which ordinary differential equations is it exact ?,* J. Comp. Appl. Math. **223** 1058-1065, 2009.

[16] A. Abdi, *Construction of high-order quadratically stable second derivative general linear methods for the numerical integration of stiff ODEs,* J. Comp. Appl. Math. **303** 218-228, 2016.

[17] H. Ramos, *A non-standard explicit integration scheme for initial-value problems,* Appl. Math. Comp. **189** 710-718, 2007.

[18] J. I. Ramos, *Linearization techniques for singularly-perturbed initial-value problems of ordinary differential equations,* Appl. Math. Comp. **163** 1143-1163, 2005.

[19] H. Ramos, G. Singh, V. Kanwar and S. Bhatia, *Solving first-order initial value problems by using an explicit non-standard A-stable one-step method in variable step-size formulation,* Appl. Math. Comp. **268** 796-805, 2015.

[20] H. A. Watts, *Starting step-size for an ODE solver,* J. Comp. Appl. Math. **9** 177-191, 1983.

[21] L. F. Shampine and M. K. Gordon, *Computer Solutions of Ordinary Differential Equations: The Initial Value Problem,* Freeman, San Francisco, CA, 1975.

[22] A. E. Sedgwick, *An effective variable order variable-step Adams method,* Dept. of Computer Science, Rept. 53, University of Toronto, Toronto, Canada, 1973.

[23] E. Hairer, *Unconditionally stable explicit methods for parabolic equations,* Numer. Math. **35** 57-68, 1980.

[24] A.R. Yaakub and D.J. Evans, *New L-stable modified trapezoidal methods for the initial value problems,* Int. J. Comp. Maths. **80** (1) 95-104, 2003.

[25] L. F. Shampine and A. Witt, *Control of local error stabilizes integrations,* J. Comp. Appl. Math. **62** 333-351, 1995 .

[26] X. Y. Wu, *Exact exponentially fitted method for solving stiff ordinary differential equations,* J. Nanjing University (Natural Science) **33** (1) 1-6, 1997.

[27] Xin-Yuan Wu and Jian-Lin Xia, *Two low accuracy methods for stiff systems,* Appl. Math. Comp. **123** 141-153, 2001.

[28] P. Bogacki and L. F. Shampine, *A* 3(2) *pair of Runge-Kutta formulas,* Appl. Math. Letter. **2** 1-9, 1998.

[29] J. R. Dormand and P.J. Prince, *A family of embedded Runge-Kutta formulae*, J. Comput. Appl. Math., **6** 19-26, 1980.

[30] H. H. Rosenbrock, *Some general implicit processes for the numerical solution of differential equations,* Comput. J. **5** 329-330, 1963 .

[31] L. F. Shampine and M. W. Reichelt, *The MATLAB ODE Suite,* SIAM J. Sci. Comput. **18** 1-22, 1997.

[32] R. Ashino, M. Nagase and R. Vaillancourt, *Behind and beyond the Matlab ODE suite,* Comput. Math. Appl. **40** 491-512, 2000.

[33] L. V. Fausett, *Applied Numerical Analysis Using Matlab,* Pearson Prentice Hall, 2008.

[34] H. Ramos, *Contributions to the development of differential systems exactly solved by multi step finite-difference schemes,* Appl. Math. Comp. **217** 639-649, 2010.

[35] M. Calvo, J. I. Montijano and L. Randez, On the change of step-size in multi-step codes, Numer. Alg. 4 (1993) 283-304.

[36] E. Hairer and G. Wanner, *Solving Ordinary Differential Equations-II,* Springer, Berlin, 1996.

[37] M. K. Jain, *Numerical Solution of Differential Equations,* Wiley, New Delhi, 1984.

UNIVERSITY INSTITUTE OF ENGINEERING AND TECHNOLOGY, PANJAB UNIVERSITY, CHANDIGARH-160014, INDIA
E-mail address: gurjinder11@gmail.com (G. Singh), vmithil@yahoo.co.in (V. Kanwar), s_bhatia@pu.ac.in (Saurabh Bhatia)