# ACCELERATION OF RUNGE-KUTTA INTEGRATION SCHEMES

PHAILAUNG PHOHOMSIRI AND FIRDAUS E. UDWADIA

*Received 24 November 2003*

A simple accelerated third-order Runge-Kutta-type, fixed time step, integration scheme that uses just two function evaluations per step is developed. Because of the lower number of function evaluations, the scheme proposed herein has a lower computational cost than the standard third-order Runge-Kutta scheme while maintaining the same order of local accuracy. Numerical examples illustrating the computational efficiency and accuracy are presented and the actual speedup when the accelerated algorithm is implemented is also provided.

## 1. Introduction

The behavior of many systems in nature and society is described by sets of differential equations that are nonlinear in character. Obtaining closed-form solutions for these equations is therefore, generally speaking, impossible, especially for systems where the state vector has a high dimension. Thus numerical methods for solving such equations become very important.

One of the most common methods for numerically integrating sets of nonlinear differential equations is the Runge-Kutta (RK) family of single-step methods. The RK integrator of order $n$ has a local error over a time step $h$ of $O(h^{n+1})$. For the third- and fourth-order (RK3 and RK4, respectively) methods, the number of function evaluations required over a single time step are three and four, respectively. In this paper, we propose a simple scheme that uses an RK-type third-order procedure [1, 2] but which utilizes just two function evaluations at each step. The scheme has the same order of local accuracy as the standard RK3 method and hence appears to provide improvement in computational efficiency without any significant effect on computational accuracy. The drawback of the scheme, however, is that it is, in reality, a two-step scheme and therefore needs to be initially "started" using an accurate one-step method, like a higher-order RK integrator.

Discrete Dynamics in Nature and Society 2004:2 (2004) 307–314
2000 Mathematics Subject Classification: 65D30, 65L05, 65L06
URL: http://dx.doi.org/10.1155/S1026022604311039

## 2. Numerical scheme

Consider the ordinary differential equation

$$y' := \frac{dy}{dt} = f(y), \tag{2.1}$$

where $y \in R^n$. We will assume that the function $f$ is Lipschitz and the solution of the equation starting with the initial conditions $y(0) = y_0$ is unique. In this sequel, we will consider the following RK-type scheme to integrate this equation. We denote

$$\begin{aligned} k_1 &= f(y(t_n)) = f(y_n) = f_n, \\ k_{-1} &= f(y(t_n - h)) = f(y_{n-1}) = f_{n-1}, \\ k_2 &= f(y_n + \beta h k_1), \\ k_{-2} &= f(y_{n-1} + \beta h k_{-1}). \end{aligned} \tag{2.2}$$

The approximate solution of (2.1) at time $t_{n+1}$ is then given by

$$y_{n+1} = y_n + h[a_1 k_1 + a_{-1} k_{-1} + b(k_2 - k_{-2})], \tag{2.3}$$

where $h$ is the fixed time step. We will determine the four parameters $a_1$, $a_{-1}$, $b$, and $\beta$ in relation (2.3) by comparing relation (2.3) with the Taylor series of $y(t_n + h)$. Using (2.1), we have

$$\begin{aligned} y''(t_n) &= y_n'' = f_{y_n} f_n, \\ y'''(t_n) &= y_n''' = [f_{yy} f^2]|_{t_n} + [f_y^2 f]|_{t_n}. \end{aligned} \tag{2.4}$$

The Taylor's series expansion of $y(t_n + h)$ is simply

$$y_{n+1} := y(t_n + h) = y(t_n) + h y'(t_n) + \frac{h^2}{2} y''(t_n) + \frac{h^3}{6} y'''(t_n) + O(h^4). \tag{2.5}$$

Using relations (2.1), (2.2), and (2.4), we get

$$\begin{aligned} a_1 k_1 &= a_1 y'(t_n), \\ a_{-1} k_{-1} &= a_{-1} y'(t_n - h) = a_{-1} y'(t_n) - a_{-1} h y''(t_n) + a_{-1} \frac{h^2}{2} y'''(t_n) + O(h^3), \\ b k_2 &= b f(y_n) + b\beta h f_{y_n} f_n + b \frac{\beta^2 h^2}{2} f_{y_n y_n} f_n^2 + O(h^3) \\ &= b y'(t_n) + b\beta h y''(t_n) + b \frac{\beta^2 h^2}{2} \left[ y'''(t_n) - f_y^2 f|_{t_n} \right] + O(h^3), \end{aligned}$$

$$\begin{aligned} bk_{-2} &= bf_{n-1} + b\beta h f_{y_{n-1}} f_{n-1} + b\frac{\beta^2 h^2}{2} f_{y_{n-1}y_{n-1}} f_{n-1}^2 + O(h^3), \\ &= by'(t_n - h) + b\beta h y''(t_n - h) + b\frac{\beta^2 h^2}{2}\left[y'''(t_n - h) - f_y^2 f\,|_{t_{n-1}}\right] + O(h^3) \\ &= by'(t_n) - bhy''(t_n) + b\frac{h^2}{2}y'''(t_n) + O(h^3) \\ &\quad + b\beta h y''(t_n) - b\beta h^2 y'''(t_n) + O(h^3) \\ &\quad + b\frac{\beta^2 h^2}{2}y'''(t_n) - b\frac{\beta^2 h^2}{2} f_y^2 f\Big|_{t_n} + O(h^3). \end{aligned} \tag{2.6}$$

Using (2.6), we obtain

$$\begin{aligned} y_{n+1} - y_n := y(t_n + h) - y(t_n) \\ &= (a_1 + a_{-1})hy'(t_n) + (-a_{-1} + b)h^2 y''(t_n) \\ &\quad + (a_{-1} - b + 2b\beta)\frac{h^2}{2}y'''(t_n) + O(h^4). \end{aligned} \tag{2.7}$$

Comparing (2.7) with (2.5), we require

$$a_1 + a_{-1} = 1, \qquad -a_{-1} + b = \frac{1}{2}, \qquad \beta b = \frac{5}{12} \tag{2.8}$$

so that our numerical scheme is $O(h^4)$.

Using $b$ as a free parameter, we now obtain some sample formulae using the scheme proposed in (2.2) and (2.3):

$$a_1 = \frac{1}{4}, \qquad a_{-1} = \frac{3}{4}, \qquad b = \frac{5}{4}, \qquad \beta = \frac{1}{3}, \tag{2.9a}$$

$$a_1 = \frac{1}{2}, \qquad a_{-1} = \frac{1}{2}, \qquad b = 1, \qquad \beta = \frac{5}{12}, \tag{2.9b}$$

$$a_1 = \frac{3}{4}, \qquad a_{-1} = \frac{1}{4}, \qquad b = \frac{3}{4}, \qquad \beta = \frac{5}{9}, \tag{2.9c}$$

$$a_1 = 1, \qquad a_{-1} = 0, \qquad b = \frac{1}{2}, \qquad \beta = \frac{5}{6}. \tag{2.9d}$$

Each set of values given in (2.9) represents an $O(h^4)$ RK-type scheme. We note that our integration scheme (2.2) and (2.3) requires just two function evaluations at each time step. To compute $y_{n+1}$, we need to calculate at time $t = t_n$ only the two quantities $k_1$

and $k_2$; the quantities $k_{-1}$ and $k_{-2}$ are already known at time $t_n$ since they were computed at the previous time $t_{n-1}$. A two-step computational algorithm is thus obtained. To start the integration process, one would need a one-step integrator like RK4. However, after the start-up, only two function evaluations are required cutting the computation cost when compared to the standard RK3 algorithm, which is also accurate to $O(h^4)$, by about a third.

It should be noted that each of the specific accelerated RK3 schemes obtained by using the numerical values illustrated in (2.9) will have, in general, different truncation errors (which will also depend on the specific differential equations being integrated) and different numerical stability characteristics. The examples described here seem to indicate that the parameters given in (2.9a) and (2.9b) may be an adequate set to use. Further investigation of these aspects will be taken up in a later communication.

## 3. Numerical examples

We consider two numerical examples to illustrate the accuracy of the method proposed herein. Comparisons of the error with the standard RK3 (see [1]) method are presented. The two equations considered are

$$y' = -y, \quad y(1) = 1, \tag{3.1}$$

$$y' = -\frac{ty}{1+t^2}, \quad y(1) = 1, \tag{3.2}$$

whose solutions are $y = e^{1-t}$ and $y = \sqrt{2}/\sqrt{(1+t^2)}$, respectively.

The differential equation (3.2) is nonautonomous. We represent it, in the usual way, as a coupled set of ordinary nonlinear differential equations as

$$\dot{y} = -\frac{yz}{1+z^2}, \qquad \dot{z} = 1, \tag{3.3}$$

where the dots refer to differentiation with respect to the parameter $\tau$, and the initial conditions are taken to be $y(\tau = 1) = z(\tau = 1) = 1$. In fact, in the results presented below, (3.1) and (3.2) are both simultaneously solved as a system of three first-order differential equations by augmenting (3.1) with (3.3). The results described below refer to the implementation of the standard RK3 algorithm [1] and the accelerated RK3 algorithm proposed in (2.2) and (2.3). All computations are done in the Matlab environment.

In each case, the accelerated RK3 scheme was started by using an RK 45 variable step size scheme (ODE45 in Matlab) with a relative error tolerance of $10^{-13}$ to compute the solution at the end of the first time step of duration $h$.

Figures 3.1 and 3.2 show the relative errors in the solution $y(t_n)$ over a 10- second interval of time using a time step of 0.01. Our accelerated RK3 scheme is compared with the standard RK3 scheme provided in [1]. Both schemes give the same order of magnitude of the relative local error. Figure 3.1 corresponds to the solution of the first example

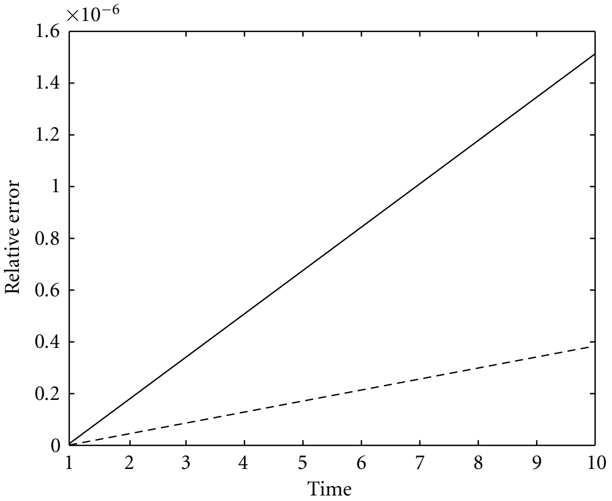

Figure 3.1. The solid line shows the accelerated RK3 scheme given in (2.9b) for (3.1). The dashed line is for the standard RK3 algorithm that uses three function evaluations. The time step for the integration is $h = 0.01$.

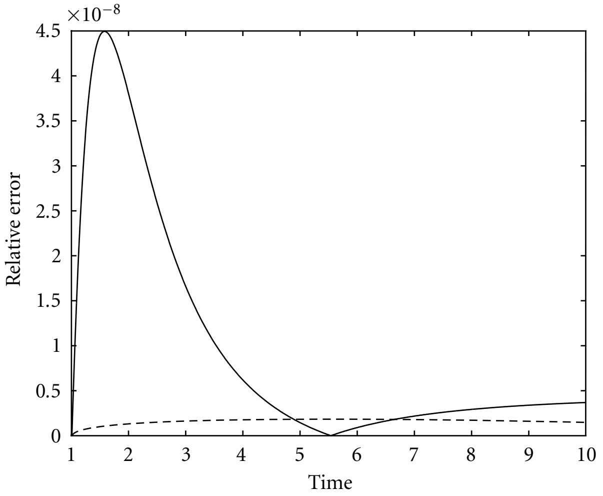

Figure 3.2. The solid line shows the accelerated RK3 scheme given in (2.9b) for (3.2). The dashed line is for the standard RK3 algorithm that uses three function evaluations per step.

(equation (3.1)) and Figure 3.2 to that of the second (equation (3.2)). For both examples, the accelerated RK3 scheme uses the parameters given in (2.9b).

Figure 3.3 shows the absolute value of the errors in the computed results for the first example problem at the end of 10 seconds, for 20 different values of the time step $h$. These errors are obtained by comparing the computed solution with the exact solution for

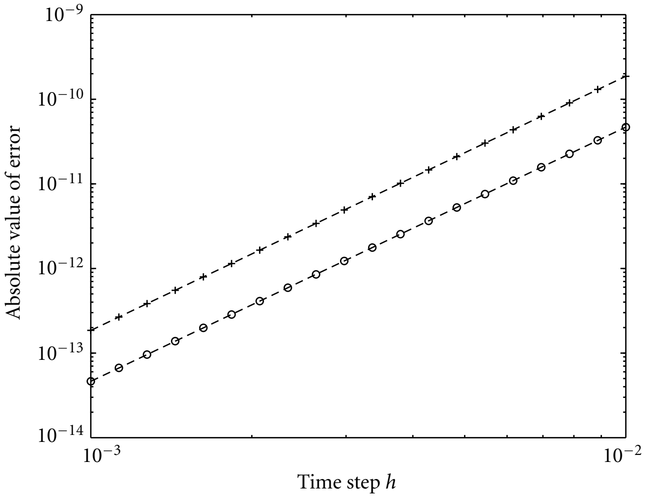

Figure 3.3. Absolute value of error at $t = 10$ in solution of (3.1) versus step size. The pluses show results for the accelerated RK3 scheme using the parameters given in (2.9b) and the circles show the standard RK3.

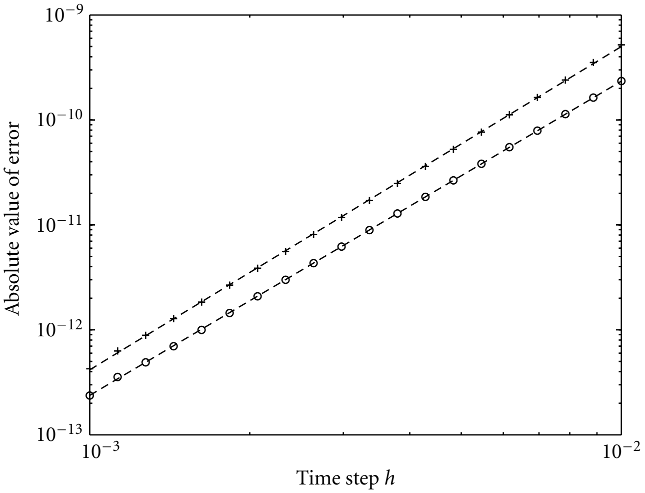

Figure 3.4. Absolute value of error at $t = 10$ in solution of (3.2) versus step size. The pluses show results for the accelerated RK3 scheme using the parameters given in (2.9b) and the circles show the standard RK3.

(3.1). The pluses indicate results for the accelerated RK3 scheme and the circles indicate those for the standard RK3. The linear least-squares fits to the computed data points are shown by the dashed lines. The slopes of these dashed lines are 3.004 (for the pluses) and 3.005 (for the circles) in close conformity with the expected accuracy of $O(h^4)$ per

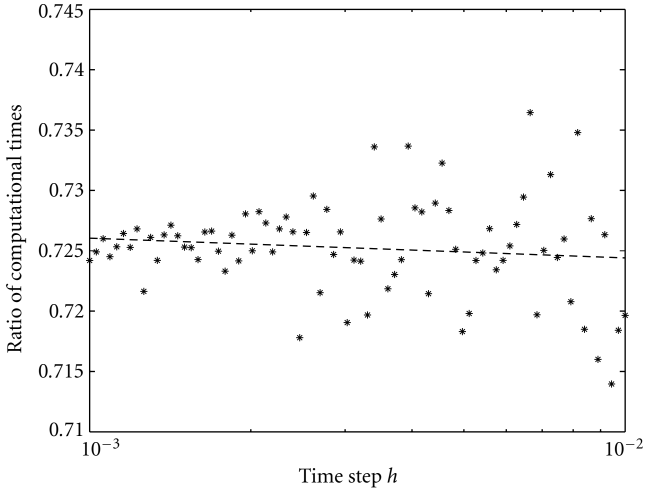

Figure 3.5. Ratio of computational time required for the accelerated RK3 scheme to that required for standard RK3 (see, [1]), showing an acceleration of at least 25% when the accelerated scheme is implemented.

integration step for each of the schemes. This leads to an accumulated error than of $O(h^3)$, as portrayed in the figure. Figure 3.4 shows similar data for the second example problem (equation (3.2)). The least-squares lines that fit the two sets of data for the accelerated RK3 (pluses) and the standard RK3 (circles) are shown in Figure 3.4. Their slopes are 3.08 (pluses) for the accelerated RK3 scheme and 2.99 (circles) for the standard RK3.

Figure 3.5 shows the (approximate) ratio of the computation time required for the accelerated RK3 scheme compared with that required for the standard RK3 scheme (at $\tau = 20$) when using 80 different time steps $h$. The Matlab function "etime" that was used gives the elapsed time approximately. While this ratio will differ from machine to machine and is dependent to some extent on the actual computer code used, it is indicative of the extent of acceleration one might expect when the accelerated RK3 scheme is actually implemented. In each case, the three first-order differential equations (3.1) and (3.3) are simultaneously solved as a system. The least-squares fit line to the data has a slope of about 0.004 and shows that the accelerated RK3 is on average about 27% faster than the standard RK3 algorithm while maintaining the same order of local accuracy.

## 4. Conclusions

A simple third-order, fixed time step Runge-Kutta-like scheme has been developed for numerical integration that requires just two function evaluations per time step. It requires that the procedure be initially started by a more accurate, higher-order one-step method. While being of the same order of local accuracy as the standard RK3 scheme, the method proposed herein is computationally more efficient. Truncation error analysis and stability analysis of the scheme will be presented in a later communication as will extensions to similar accelerated fourth- and higher-order RK schemes.

## References

[1] B. Carnahan, H. Luther, and J. Wilkes, *Applied Numerical Methods*, John Wiley & Sons, New York, 1969.

[2] L. Lapidus and J. H. Seinfeld, *Numerical Solution of Ordinary Differential Equations*, Mathematics in Science and Engineering, vol. 74, Academic Press, New York, 1971.

Phailaung Phohomsiri: Department of Aerospace and Mechanical Engineering, University of Southern California, Los Angeles, CA 90089-1453, USA

Firdaus E. Udwadia: Departments of Civil Engineering, Aerospace and Mechanical Engineering, Mathematics, and Operations and Information Management, University of Southern California, Los Angeles, CA 90089, USA

*E-mail address*: fudwadia@usc.edu