A Tenth-Order Runge-Kutta Method with Error Estimate

T. Feagin

Abstract—A tenth-order explicit Runge-Kutta method with embedded results of order eight is exhibited. The difference between the results of orders eight and ten can be used to estimate the local truncation error and thus to vary the stepsize. Numerical experiments demonstrate that the method compares favorably with other high-order embedded methods.

Index Terms—Embedded Runge-Kutta methods, high-order, initial value problems

I. INTRODUCTION

Obtaining an analytical solution to the first-order system of ordinary differential equations:

\[ \frac{dx}{dt} = f(t, x) \]

subject to the initial condition:

\[ x(t_0) = x_0, \]

is often difficult, especially when \( f(t, x) \) is a nonlinear function of \( x \). Numerical methods are frequently used in order to obtain approximate solutions valid over some finite range of interest, \([t_0, t_f]\). It is well known that numerical solutions are subject to errors due to truncation, rounding, and (potentially) numerical instability. Consequently, it is important to have estimates or bounds for these errors. Investigators frequently have reasonable estimates for those regions where numerical instability or rounding errors might dominate the calculations and would naturally employ moderate step sizes, neither too large or too small, so as to avoid such regions. As a result, the principal problem is obtaining reasonably accurate estimates of the truncation errors that would allow for the selection of an appropriate step size for a given error tolerance.

Runge-Kutta methods are particularly advantageous for those problems (1) in which the character of the solution changes so rapidly that a major change in the step size must be employed in order to track the solution accurately. Multistep methods, on the other hand, often have difficulty adjusting the step size adequately in response to such rapid changes, especially if the strategy is limited to halving and doubling the step size.

A number of authors [1, 2, 3, 4, 5, 6] have developed explicit Runge-Kutta methods that provide an estimate for the local truncation error. The estimates are usually provided using the difference between the final \( m \)-th order result and the result of an embedded formula of order \( m-1 \). Most notably, in the 1960’s, Fehlberg [7] developed a number of very efficient, embedded explicit Runge-Kutta methods using this approach. His high-order methods include those of order 5(6), 6(7), 7(8), and 8(9). The first number, in each case, represents the order of the embedded, lower-order result. In the present work, the embedded result is of order \( m-2 \) instead of \( m-1 \), which allows for a more efficient estimate of the errors [8]. In fact, the method described herein requires only seventeen stages, the minimum known value for the tenth order.

An \( n \)-stage, explicit Runge-Kutta method obtains an approximation to \( x(t_0 + h) \), the solution of (1) at the point \( t_0 + h \), using the formula:

\[ x_n = x_0 + h \sum_{k=0}^{n-1} c_k f(t_k, x_k) \] (2a)

where \( t_k = t_0 + \alpha_k h \) and the intermediate approximations \( x_k \) are computed from

\[ x_k = x_0 + h \sum_{j=0}^{k-1} \beta_j f(t_j, x_j) \] (2b)

for \( k = 1, \ldots, n-1 \). The parameters or coefficients \( \{c_k\}_{k=0}^{n-1} \), \( \{\alpha_k\}_{k=0}^{n-1} \), and \( \{\beta_j\}_{j=0}^{k-1} \) are simply real constants. Note that \( \alpha_0 = 0 \) (as is normally assumed); that is, the first evaluation is \( f(t_0, x_0) \). The method is said to be of order \( m \) if the local truncation error, i.e. the difference between the approximation \( x_n \) and the true solution \( x(t_0 + h) \), is \( O(h^{m+1}) \).

The above notation follows that of Fehlberg, Bettis, and Horn [1, 7, 9]. The form (2) is also known as the Runge-Kutta ansatz.

In order to monitor and control the local truncation error, a lower-order result:

\[ \hat{x}_n = x_0 + h \sum_{k=0}^{n-1} \hat{c}_k f(t_k, x_k) \] (3)

is also computed. The difference,

\[ \hat{x}_n - x_n = h \sum_{k=0}^{n-1} (\hat{c}_k - c_k) f(t_k, x_k) \] (4)

is then used as an estimate of the local truncation error encountered in the current step. Note that this estimate is of the error in \( \hat{x}_n \), not of the error in \( x_n \). Nevertheless, the estimate has been shown to be useful for monitoring the accuracy of the numerical integration and for varying the
stepsizes in such a way that the truncation errors are more uniform [10, pp. 184-196].

For an explicit Runge-Kutta method, the number of stages, \( n \), is the number of times the derivative function \( f(t,x) \) must be evaluated during each step of the integration. Obviously, in order to obtain an approximation to the solution \( x(t) \) valid over the entire range \( [t_0, t_f] \), a number of steps would normally have to be taken. At each step, the formula (2) could be used to provide an approximation to \( x(t_0 + h) \) and the formula (4) could be used to estimate the local truncation error. If the estimate of the error were within a user-specified tolerance, then the step would be accepted and the values of \( t_0 \) and \( x_0 \) would be updated accordingly. If the estimated error were not within the tolerance, the step would be rejected. In either event, the stepsize \( h \) would be adjusted in such a way that the expected value of the next estimated error would be within the tolerance. Of course, for the very last step, \( h \) would be reduced, if necessary, in order to stop the integration precisely at \( t_f \).

II. BACKGROUND

The first Runge-Kutta methods were developed many years ago. Euler [11] developed his first-order method in about 1768. In 1895, Runge [12] generalized Euler’s method as shown in (2). Kutta [13] further developed the idea, about 1768. In 1895, Runge [12] generalized Euler’s method as shown in (2). Kutta [13] further developed the idea, during the last several decades, numerous high-order explicit Runge-Kutta methods [4, 7, 8, 15, 16, 17, 18, 19] have been developed. A number of enhancements of the basic methods have also been studied including the abilities to vary the order [9, 20, 21], to estimate the local truncation error [1, 2, 3, 4, 5, 6, 7], and to provide for dense output (i.e., to make accurate, intermediate results available at any point within a given step) [9, 22, 23]. The main focus here is on the estimation of truncation errors.

III. THE COEFFICIENTS OF AN EXPLICIT RUNGE-KUTTA METHOD OF ORDER TEN

A tenth-order Runge-Kutta method requires the solution of 1,205 nonlinear algebraic equations. With the help of a Mathematica program, a Runge-Kutta method of order ten with an embedded eighth-order result has been determined with seventeen stages and will be referred to as RK8(10). In comparison, Fehlberg’s highest order embedded method, RK8(9), and Hairer’s tenth-order method [18] (without an embedded result) also require seventeen stages.

The method is based on the six-point Gauss-Lobatto quadrature formula. The coefficients are given to sixty significant digits in Table 1. Fifty-four of the 170 coefficients (more than 30%) are zero. The maximum absolute value of any coefficient is less than 5.785.
The coefficients $\beta_{ij}$ represent new weights for an embedded result without additional evaluations of $f(t_k, x_k)$. Therefore, $\beta_{ij}$ can be used to estimate the local truncation errors.

Originally, the tenth-order method of RK8(10), was developed without consideration for estimating the local truncation errors. However, the method was later extended to admit an eighth-order embedded result without additional evaluations of $f(t_k, x_k)$. Let $\tilde{x}_k$ represent an approximation to $x(t_k + h)$ of order $m < n$ such that

$$\tilde{x}_k = x_k + h \sum_{\ell=0}^{n-1} \frac{\ell + 1}{\ell!} f(t_k, x_k) .$$

The coefficients $\tilde{c}_j^{i-1}$ represent new weights for an embedded lower-order method. For RK8(10), the largest value of $m$ that is possible is eight. In other words, any
method of order nine that uses the same values for the coefficients \( \{\alpha_k\}_{k=0}^{n-1} \) and \( \{\beta_k\}_{k=0}^{n-1} \) must also have \( \hat{c}_k = c_k \) for \( k = 0, \ldots, n-1 \) and consequently would not be distinct from the tenth-order result.

The eighth-order result for \( x_n \) is obtained by taking \( \hat{c}_k = c_k \) for all values of \( k \) except for \( k = 1 \) and \( k = 15 \). Choosing, for example, the values \( \hat{c}_1 = 1/36 \) and \( \hat{c}_{15} = -1/36 \) (in contrast to \( c_1 = 1/40 \) and \( c_{15} = -1/40 \) for the tenth-order result) causes \( \hat{x}_n \) to be an eighth-order result. The difference

\[
\hat{x}_n - x_n = \frac{1}{360} h \left( f(t_i, x_i) - f(t_{i+1}, x_{i+1}) \right)
\]

is then used as an estimate of the local truncation error of the eighth-order result, which in turn is used to adjust the stepsize of the numerical integration and to provide for a more efficient algorithm overall. However, note that (like Fehlberg’s high-order methods) the difference \( \hat{x}_n - x_n \) vanishes for problems in which \( f = f(t) \) only (i.e., for quadrature problems). Therefore, the error estimate must be used with caution.

As it turns out, the eighth-order result described above satisfies all but twenty of the 286 additional ninth-order equations. Also note that it is the tenth-order result, \( x_n \), that is used to propagate the solution to the next step.

V. SOME NUMERICAL EXPERIMENTS

The tenth-order method RK8(10), given in the previous section, has been applied to a number of initial value problems of the form (1). A couple of such applications are described in this section. The performance of the method with respect to its ability to obtain an accurate numerical solution over the range \([t_0, t_f]\) for these problems has been compared with that of several other methods using efficiency diagrams. For a given method displayed in these diagrams, the negative logarithm (base 10) of the absolute value of the largest component of the error (denoted by \( E \)) is plotted as a function of the logarithm (base 10) of the number of evaluations of \( f(t, x) \) required (denoted by \( NF \)). For problems in which \( f(t, x) \) is computationally expensive, the time of computation is proportional to \( NF \). On such a diagram, the curve connecting the individual experimental results is often a nearly straight line with slope proportional to the order of the method for regions where the largest source of error is truncation error.

Figure 1 exhibits an efficiency diagram for the initial value problem (1) where the differential equations are the equations of motion for the two-body problem:

\[
\begin{align*}
\frac{dx_1}{dt} &= x_2, & \frac{dx_2}{dt} &= -x_1(x_1^2 + x_2^2)^{\frac{3}{2}}, \\
\frac{dx_3}{dt} &= x_4, & \frac{dx_4}{dt} &= -x_3(x_1^2 + x_2^2)^{\frac{3}{2}}
\end{align*}
\]

subject to the initial conditions:

\[
\begin{align*}
x_1(0) &= 1, & x_2(0) &= e, \\
x_3(0) &= 0, & x_4(0) &= \sqrt{1-e^2}
\end{align*}
\]

where the eccentricity \( e = 0.4 \). For such an eccentricity, it is essential to be able to adjust the stepsize of the integration in order to preserve a fairly uniform truncation error.

For each experimental result, the error is measured at \( t = 4\pi \). In the diagram, the results for RK8(10) are connected by the blue line, the results for RK7(8) due to Fehlberg [7] are shown by the purple line, and the results for Fehlberg’s RK8(9) are shown by the green line. Each line shown in the diagram connects the points of more than twenty individual experimental results (which are fairly evenly distributed).
It can be seen from the diagram that as \( NF \) increases (i.e., as the stepsize decreases), the RK8(10) method is the most efficient of the methods shown when an accuracy above nine significant digits is desired. For lower accuracy, Fehlberg’s RK7(8) provides the most efficient results.

Figure 2 exhibits an efficiency diagram for the initial value problem (1) where the differential equations are the well-known predator-prey equations:

\[
\begin{align*}
\frac{dx_1}{dt} &= x_1 (2 - x_2), \quad \frac{dx_2}{dt} = x_2 (x_1 - 1), \\
\text{subject to the initial conditions:} \\
x_1(0) &= 2, \quad x_2(0) = 2.
\end{align*}
\]

For each experimental result in the diagram, the error is measured at the final point, \( t = 4 \).

Once again, each line shown in the diagram connects the points of more than twenty individual experimental results. It can be seen from the diagram that when highly accurate solutions are required, the RK8(10) method is the most efficient of the methods shown for this problem as well.

VI. CONCLUSIONS

A new explicit Runge-Kutta method of order ten (with seventeen stages) has been exhibited. The method contains an embedded method of order eight. The difference between the eighth- and tenth-order results at the end of a step can be used as an estimate of the local truncation error. This estimate can be used to monitor the accuracy of the numerical integration and to control the stepsize in a meaningful way. Numerical experiments illustrates the effectiveness of the technique and indicates that the new method offers promise for those problems requiring high accuracy.

VII. REFERENCES


