Comp. & Maths. with Appls. Vol. 12A, No. 6, pp. 815–824, 1986
Printed in Great Britain

0886-9553/86 $3.00 + .00
Pergamon Journals Ltd

A SIXTH-ORDER IMBEDDED RUNGE–KUTTA ALGORITHM WITH CONTINUOUSLY VARIABLE WEIGHTS

CURTIS OUTLAW, LEROY DERR and DIRAN SARAFYAN
University of New Orleans, New Orleans, LA 70148, U.S.A.

Abstract—Sarafyan's continuous method for approximate solution of initial value problems is extended to the sixth-order case. The approximations obtained are continuous throughout an interval. The main formula is a polynomial of fifth degree in c, $0 < c < 1$, such that for each c a valid fifth-order approximation of Runge–Kutta type is obtained for the solution at $x = x_0 + ch$. At $c = 1$ the approximation improves to sixth order. Lower order continuous formulas are imbedded, using polynomials of lesser degree, which may be used for error estimation or step-size selection. These formulas are differentiated to give continuous approximations to the first and second derivatives of the solution. The formulas are valid for systems as well as for a single differential equation.

INTRODUCTION

As is well known, Runge–Kutta methods are discrete processes—they provide a discrete set of approximate values for the solution of a differential equation. To obtain approximations between members of the discrete set, it was necessary to reduce the step-size and reapply the formula. The same can be said of the "differential quadrature" method recommended by Bellman[1] for the approximation of derivatives in initial value problems. In 1983, this situation was changed when Sarafyan[7] established a continuous Runge–Kutta process. With only six evaluations of f, the directional function, for the system of differential equations

$$dy/dx = f(x, y), \quad y(x_0) = y_0, \tag{1}$$

he obtains not only a fifth-order approximation $\tilde{y}_5(x_0 + h)$ at full step-size h, but also approximations $\tilde{y}_4(x_0 + ch)$, $\tilde{y}_3(x_0 + ch)$, $\tilde{y}_2(x_0 + ch)$ (the subscript indicates the order) for all c in $[0,1]$. The main formula, not obtained by an interpolation process, is a fourth-degree polynomial in $c = (x - x_0)/h$; for any c in $[0,1)$ it provides a fourth-order Runge–Kutta approximation to $y(x_0 + ch)$! At $c = 1$ the approximation is of fifth order. The derivatives of this polynomial give continuous approximations to the derivatives of $y(x_0 + ch)$ not only at a discrete set of *equally spaced* points as Bellman does, but for all c in $[0,1]$, without requiring additional evaluations of f. While attention is usually restricted to c in $[0,1]$, the formulas may even be used advantageously for c in the extended interval $[-0.5,1.5]$.

The formulas were derived using the algebraic equations for the fifth-order scalar case (for a single differential equation), but the coefficients obtained for the formulas satisfy conditions which make the algebraic equations for the scalar and vector (system) cases reduce to the same equivalent system (see [8]). Thus the formulas are valid for both the scalar and vector cases.

Such a package of imbedded formulas has clear advantages over traditional Runge–Kutta formulas, such as ease of run-time error estimation and adaptation of step-size, for starting predictor–corrector methods, for block integration and for approximation of y'.

In this paper we present a sixth-order nine-stage Runge–Kutta formula with fifth- and lower-order formulas continuously imbedded. The notation is

$$\tilde{y}_6(x_0 + h) = y_0 + \textstyle\sum_{i=0}^{8} w_i k_i, \quad k_0 = hf(x_0, y_0), \quad k_i = hf(x_0 + a_i h, y_0 + \sum_{j=0}^{i-1} b_{ij} k_j),$$
$$i = 1, 2, \ldots, 8, \quad a_i = \textstyle\sum_{j=0}^{i-1} b_{ij},$$

where h is the chosen step-size. The formulas are derived with conditions in force which make the systems of algebraic equations for scalar and vector cases equivalent, so that the formulas are again valid for both cases.

For the derivation of conventional Runge–Kutta algorithms, the Taylor series of $y(x_0 + h)$ in powers of h and in terms of the directional function f is obtained. In the present work it is necessary to obtain the expansion of $y(x_0 + ch)$ about x_0 in powers of $c = (x - x_0)/h$, where h is considered fixed, and for any c in $(0,1]$. Using an operational method due to Huta[4], with

$$_pf_q = \frac{\partial^{p+q}f}{\partial x^p\, \partial y^q}, \quad D^{(n)}f_i = \Sigma_{j=0}^{n} \binom{n}{j} f^j \cdot {}_{n-j}f_{i+j}$$

and each function evaluated at (x_0, y_0), we obtain for the case of a single equation

$$\begin{aligned}
y(x_0 + ch) &= y_0 + cfh + \frac{c^2}{2!}[Df]h^2 + \frac{c^3}{3!}[D^{(2)}f + f_1 Df]h^3 \\
&+ \frac{c^4}{4!}[D^{(3)}f + 3Df_1Df + f_1D^{(2)}f + f_1^2Df]h^4 \\
&+ \frac{c^5}{5!}[D^{(4)}f + f_1D^{(3)}f + f_1^2D^{(2)}f + f_1^3Df + 4D^{(2)}fDf_1 \\
&+ 6D^{(2)}f_1Df + 7f_1Df_1Df + 3f_2(Df)^2]h^5 \\
&+ \frac{c^6}{6!}[D^{(5)}f + f_1D^{(4)}f + f_1^2D^{(3)}f + f_1^3D^{(2)}f + f_1^4Df \\
&+ 5D^{(3)}fDf_1 + 9f_1D^{(2)}fDf_1 + 12f_1^2DfDf_1 + 10D^{(3)}f_1Df \\
&+ 16f_1D^{(2)}f_1Df + 10D^{(2)}f_1D^{(2)}f + 10f_2D^{(2)}fDf \\
&+ 13f_1f_2(Df)^2 + 15Df_2(Df)^2 + 15(Df_1)^2Df]h^6 + O(h^7).
\end{aligned}$$

We must solve the algebraic equations obtained by matching terms of the above expansion with those of the expansion of

$$\bar{y}_5(x_0 + ch) = y_0 + \Sigma_{i=0}^{8} w_i(c)k_i,$$

where h is fixed and c is variable in $[0,1]$, to obtain the continuous fifth-order approximation, and the matching must be such that when $c = 1$, the approximation $[\bar{y}_6(x_0 + h),\ w_i = w_i(1)]$ is sixth order. This is to be accomplished with only the $w_i(c)$'s depending on c, so that the k_i's are calculated only once. Huta's method is applied to expand each

$$k_i = hf(x_0 + a_ih,\ y_0 + \Sigma_{j=0}^{i-1} b_{ij}k_j).$$

For the sixth-order scalar case, the system resulting from the matching is, with

$$A_{ik} = \Sigma_{j=1}^{i-1} a_j^k b_{ij}, \quad i = 2, \ldots, 8, \quad k = 1, \ldots, 4,$$

1. $\Sigma_{i=0}^{8} w_i(c) = c,$
2–5. $\Sigma_{i=1}^{8} a_i^p w_i(c) = \dfrac{c^{p+1}}{p+1}, \quad p = 1, \ldots, 4,$
6. $\Sigma_{i=1}^{8} a_i^5 w_i = \frac{1}{6},$
7–9. $\Sigma_{i=2}^{8} w_i(c)A_{ip} = \dfrac{c^{p+2}}{(p+1)(p+2)}, \quad p = 1, 2, 3,$
10. $\Sigma_{i=2}^{8} w_iA_{i4} = \frac{1}{30},$
11, 12. $\Sigma_{i=2}^{8} w_i(c)a_iA_{ip} = \dfrac{c^{p+3}}{(p+1)(p+3)}, \quad p = 1, 2,$
13. $\Sigma_{i=2}^{8} w_ia_iA_{i3} = \frac{1}{24},$
14. $\Sigma_{i=2}^{8} w_i(c)a_i^2A_{i1} = c^5/10,$
15. $\Sigma_{i=2}^{8} w_ia_i^3A_{i1} = \frac{1}{12},$
16. $\Sigma_{i=2}^{8} w_ia_i^2A_{i2} = \frac{1}{18},$
17. $\Sigma_{i=2}^{8} w_i(c)A_{i1}^2 = c^5/20,$

18. $\Sigma_{i=2}^{8} w_i a_i A_{i1}^2 = \frac{1}{24}$,
19. $\Sigma_{i=2}^{8} w_i A_{i1} A_{i2} = \frac{1}{36}$,
20, 21. $\Sigma_{i=3}^{8} w_i(c) (\Sigma_{j=2}^{i-1} b_{ij} A_{jp}) = \dfrac{c^{p+3}}{12(p^2+1)}$, $p = 1, 2$,
22. $\Sigma_{i=3}^{8} w_i(\Sigma_{j=2}^{i-1} b_{ij} A_{j3}) = \frac{1}{120}$,
23. $\Sigma_{i=3}^{8} w_i a_i (\Sigma_{j=2}^{i-1} b_{ij} a_j A_{j1}) = \frac{1}{48}$,
24. $\Sigma_{i=4}^{8} w_i(c)(\Sigma_{j=3}^{i-1} b_{ij}[\Sigma_{m=2}^{j-1} b_{jm} A_{m1}]) = c^5/120$,
25. $\Sigma_{i=4}^{8} w_i(\Sigma_{j=3}^{i-1} b_{ij}[\Sigma_{m=2}^{j-1} b_{jm} A_{m2}]) = \frac{1}{360}$,
26. $\Sigma_{i=5}^{8} w_i(\Sigma_{j=4}^{i-1} b_{ij}[\Sigma_{m=3}^{j-1} b_{jm}\{\Sigma_{n=2}^{m-1} b_{mn} A_{n1}\}]) = \frac{1}{720}$,
27. $\Sigma_{i=3}^{8} w_i(c)(\Sigma_{j=2}^{i-1} \{a_i + a_j\} b_{ij} A_{j1}) = 7c^5/120$,
28. $\Sigma_{i=3}^{8} w_i(c)(\Sigma_{j=2}^{i-1} \{a_i + a_j\} b_{ij} A_{j2}) = \frac{1}{40}$,
29. $\Sigma_{i=3}^{8} w_i(\Sigma_{j=2}^{i-1} \{a_i^2 + a_j^2\} b_{ij} A_{j1}) = \frac{2}{45}$,
30. $\Sigma_{i=3}^{8} w_i(\Sigma_{j=2}^{i-1} [2A_{i1} A_{j1} + A_{j1}^2] b_{ij}) = \frac{13}{360}$,
31. $\Sigma_{i=4}^{8} w_i(\Sigma_{j=3}^{i-1} b_{ij}[\Sigma_{m=2}^{j-1} b_{jm}\{a_i + a_j + a_m\} A_{m1}]) = \frac{1}{60}$.

Each equation which has a power of c on its right-hand side must be satisfied for arbitrary c so that the matching for $\bar{y}_5(x_0 + ch)$ yields a fifth-order formula; the other equations need only be satisfied for $c = 1$ so that $\bar{y}_6(x_0 + h)$ will be a sixth-order approximation.

In [8] the authors showed that imposing certain equivalence criteria guarantees that any solution of the algebraic equations associated with the scalar case for a fifth- or sixth-order Runge–Kutta process will also satisfy the equations for the vector case—indeed, the criteria cause the equations for both cases to reduce to the same simpler equivalent system. Although the situation here is more general because the weights depend on c, it can be shown as in [8] that if we impose the criteria

$$\text{(A)}\quad \begin{cases} a_8 = 1, \quad \Sigma_{i=j+1}^{8} w_i b_{ij} = w_j(1 - a_j), \quad j = 2, \ldots, 7, \\ A_{i1} = a_i^2/2, \quad i = 2, \ldots, 8, \end{cases}$$

where

$$A_{ik} = \Sigma_{j=1}^{i-1} a_j^k b_{ij}, \quad i = 2, \ldots, 8, \quad k = 1, \ldots, 4,$$

then the combined scalar and vector systems for fifth- and sixth-order formulas reduce to [with $w_i = w_i(1)$]

$$\begin{aligned}
&w_1(c) = 0, \quad \Sigma_{i=2}^{8} w_i(c) b_{i1} = 0,\\
&\Sigma_{i=2}^{8} w_i(c) a_i b_{i1} = 0, \quad \Sigma_{i=2}^{8} w_i A_{i1} b_{i1} = 0,\\
&\Sigma_{i=3}^{8} w_i(c)(\Sigma_{j=2}^{i-1} b_{ij} b_{j1}) = 0,\\
&\Sigma_{i=3}^{8} w_i(\Sigma_{j=2}^{i-1} \{a_i + a_j\} b_{ij} b_{j1}) = 0,\\
&\Sigma_{i=4}^{8} w_i(\Sigma_{j=3}^{i-1} b_{ij}[\Sigma_{m=2}^{j-1} b_{jm} b_{m1}]) = 0,\\
&\Sigma_{i=0}^{8} w_i(c) = c, \quad \Sigma_{i=2}^{8} a_i w_i(c) = c^2/2, \quad \Sigma_{i=2}^{8} a_i^2 w_i(c) = c^3/3,\\
&\Sigma_{i=2}^{8} a_i^3 w_i(c) = c^4/4, \quad \Sigma_{i=2}^{8} a_i^4 w_i(c) = c^5/5,\\
&\Sigma_{i=2}^{8} a_i^5 w_i = \tfrac{1}{6}, \quad \Sigma_{i=2}^{8} w_i(c) A_{i2} = \tfrac{1}{12},\\
&\Sigma_{i=2}^{8} w_i(c) A_{i3} = \tfrac{1}{20}, \quad \Sigma_{i=2}^{8} w_i(c) a_i A_{i2} = \tfrac{1}{15},\\
&\Sigma_{i=2}^{8} w_i a_i A_{i3} = \tfrac{1}{24}, \quad \Sigma_{i=3}^{8} w_i(c)(\Sigma_{j=2}^{i-1} b_{ij} A_{j2}) = \tfrac{1}{60},\\
&\Sigma_{i=2}^{8} w_i a_i^2 A_{i2} = \tfrac{1}{18}, \quad \Sigma_{i=3}^{8} w_i(\Sigma_{j=2}^{i-1} b_{ij} A_{j3}) = \tfrac{1}{120},\\
&\Sigma_{i=4}^{8} w_i(\Sigma_{j=3}^{i-1} b_{ij}[\Sigma_{m=2}^{j-1} b_{jm} A_{m2}]) = \tfrac{1}{360},\\
&\Sigma_{i=3}^{8} w_i(\Sigma_{j=2}^{i-1} \{a_i + a_j\} b_{ij} A_{j2}) = \tfrac{1}{40},\\
&\Sigma_{i=3}^{8} w_i(\Sigma_{j=2}^{i-1} \{a_i^2 + a_j^2\} b_{ij} A_{j1}) = \tfrac{2}{45}.
\end{aligned}$$

To further simplify this reduced system, we also require that

$$
\text{(B)} \quad \begin{cases} A_{i2} = a_i^3/3, \quad i = 3, \ldots, 8, \\ A_{i3} = a_i^4/4, \quad i = 5, \ldots, 8, \\ w_2(c) = w_3(c) = w_4(c) = 0, \\ b_{31} = b_{41} = \cdots = b_{81} = 0, \\ b_{52} = b_{62} = b_{72} = b_{82} = 0. \end{cases}
$$

With all these conditions in force, the combined system for fifth and sixth order reduces to (A), (B) and

$$
\text{(C)} \quad \begin{cases} w_0(c) + \Sigma_{i=5}^{8} w_i(c) = c, \\ \Sigma_{i=5}^{8} w_i(c) a_i^p = \dfrac{c^{p+1}}{p+1}, \quad p = 1, 2, 3, 4, \\ \Sigma_{i=5}^{8} w_i a_i^5 = \frac{1}{6}\,. \end{cases}
$$

A solution can be obtained with a_1 and b_{84} as free parameters. We found it convenient to use $a_1 = \frac{1}{32}$ and $b_{84} = \frac{125}{154}$. If another value of a_1 is used, then one must modify b_{21} and b_{20} by

$$
b_{21} = \frac{1}{2a_1}\left(\frac{1}{24}\right)^2 \text{ and } b_{20} + b_{21} = \frac{1}{24}\,.
$$

If b_{84} is changed, then the following must be changed accordingly:

$$
\begin{aligned}
b_{73} &= \frac{244}{231} + \frac{16}{125} b_{84}, \quad b_{74} = \frac{1}{352}\left(\frac{2375}{6} - 77 b_{84}\right), \\
b_{75} &= \frac{-37}{24} + \frac{77}{500} b_{84}, \quad b_{76} = \frac{1}{112}\left(\frac{505}{6} - \frac{77}{125} b_{84}\right), \\
b_{83} &= \frac{-256}{7}\left(\frac{2}{125} b_{84} + \frac{5}{21}\right), \quad b_{85} = \frac{124}{21} - \frac{88}{125} b_{84}, \\
b_{86} &= \frac{-379}{147} + \frac{22}{875} b_{84}, \\
b_{70} &= \frac{3}{4} - \Sigma_{j=3}^{6} b_{7j}, \quad b_{80} = 1 - \Sigma_{j=3}^{7} b_{8j}.
\end{aligned}
$$

The solution is

i	a_i	b_{ij},	$j = 0, 1, \ldots, 7$,				
1	$\frac{1}{32}$	$\frac{1}{32}$					
2	$\frac{1}{24}$	$\frac{1}{72}$	$\frac{1}{36}$				
3	$\frac{1}{16}$	$\frac{1}{64}$	0	$\frac{3}{64}$			
4	$\frac{1}{5}$	$\frac{53}{125}$	0	$-\frac{204}{125}$	$\frac{176}{125}$		
5	$\frac{1}{4}$	$\frac{1}{96}$	0	0	$\frac{4}{33}$	$\frac{125}{1056}$	

6	$\frac{1}{2}$	$-\frac{19}{24}$	0	0	$\frac{64}{33}$	$-\frac{875}{264}$	$\frac{8}{3}$		
7	$\frac{3}{4}$	$-\frac{11}{16}$	0	0	$\frac{268}{231}$	$\frac{125}{132}$	$-\frac{17}{12}$	$\frac{251}{336}$	
8	1	$\frac{1211639}{222222}$	0	0	$-\frac{14848}{1617}$	$\frac{125}{154}$	$\frac{16}{3}$	$-\frac{376}{147}$	$\frac{8}{7}$

$$w_1(c) = w_2(c) = w_3(c) = w_4(c) = 0,$$

$$w_5(c) = (128c^2/3)\left[\frac{3}{16} - \frac{13}{24}c + \frac{9}{16}c^2 - \frac{1}{5}c^3\right],$$

$$w_6(c) = 64c^2\left[-\frac{3}{32} + \frac{19}{48}c - \frac{1}{2}c^2 + \frac{1}{5}c^3\right],$$

$$w_7(c) = (128c^2/3)\left[\frac{1}{16} - \frac{7}{24}c + \frac{7}{16}c^2 - \frac{1}{5}c^3\right],$$

$$w_8(c) = (32c^2/3)\left[-\frac{3}{64} + \frac{11}{48}c - \frac{3}{8}c^2 + \frac{1}{5}c^3\right],$$

$$w_0(c) = c - (32c^2/3)\left[\frac{25}{64} - \frac{35}{48}c + \frac{5}{8}c^2 - \frac{1}{5}c^3\right].$$

$$\tilde{y}_6(x_0 + h) = y_0 + \frac{1}{90}[7(k_0 + k_8) + 32(k_5 + k_7) + 12k_6], \tag{2}$$

$$\tilde{y}_5(x_0 + ch) = y_0 + ck_0 + (c^2/6)A + \frac{2}{9}c^3B + \frac{4}{3}c^4C + \frac{32}{15}c^5D, \tag{3}$$

where

$$A = -25k_0 + 48k_5 - 36k_6 + 16k_7 - 3k_8,$$

$$B = 35k_0 - 104k_5 + 114k_6 - 56k_7 + 11k_8,$$

$$C = -5k_0 + 18k_5 - 24k_6 + 14k_7 - 3k_8,$$

$$D = k_0 - 4k_5 + 6k_6 - 4k_7 + k_8.$$

Similarly, we obtain the lower order approximations

$$\tilde{y}_4(x_0 + ch) = y_0 + ck_0 + (c^2/3)E + \frac{8}{3}c^3F + \frac{8}{3}c^4G, \tag{4}$$

where

$$E = -11k_0 + 18k_5 - 9k_6 + 2k_7,$$

$$F = 2k_0 - 5k_5 + 4k_6 - k_7,$$

$$G = -k_0 + 3k_5 - 3k_6 + k_7;$$

$$\tilde{y}_3(x_0 + ch) = y_0 + ck_0 + c^2H + \frac{8}{3}c^3I, \tag{5}$$

where

$$H = -3k_0 + 4k_5 - k_6,$$

$$I = k_0 - 2k_5 + k_6.$$

The approximations for $y'(x_0 + ch)$ are (the subscripts indicate the orders of approximation)

$$[\bar{y}_5(x_0 + ch)]'_4 = (1/h)[k_0 + (c/3)A + \tfrac{2}{3}c^2B + \tfrac{16}{3}c^3C + \tfrac{32}{3}c^4D], \tag{6}$$

$$[\bar{y}_4(x_0 + ch)]'_3 = (1/h)(k_0 + \tfrac{2}{3}cE + 8c^2F + \tfrac{32}{3}c^3G), \tag{7}$$

$$[\bar{y}_3(x_0 + ch)]'_2 = (1/h)(k_0 + 2cH + 8c^2I). \tag{8}$$

One can even use the following approximations for $y''(x_0 + ch)$:

$$[\bar{y}_5(x_0 + ch)]''_3 = (1/h^2)(\tfrac{1}{3}A + \tfrac{4}{3}cB + 16c^2C + \tfrac{128}{3}c^3D)$$

$$[\bar{y}_4(x_0 + ch)]''_2 = (1/h^2)(\tfrac{2}{3}E + 16cF + 32c^2G).$$

As we mentioned earlier, all of the above formulas are valid for systems of differential equations as well as for a single equation.

Examples

We shall discuss two initial-value problems:

$$\text{(I)}\begin{cases} dy/dx = -30y, \\ y(0) = \tfrac{1}{3}, \quad y = \tfrac{1}{3}e^{-30x} \end{cases} \qquad \text{(II)}\begin{cases} dy/dx = 10y^2, \\ y(0) = 1, \quad y = 1/(1 - 10x). \end{cases}$$

The first involves a stiff differential equation; it was studied in some detail by Burden *et al.*[2]. With step size $h = 0.1$, they found that Euler's method, the classical fourth-order Runge–Kutta formula and Adams' predictor corrector method give, respectively at $x = 1.5$: $-10{,}922$, 40 and 803,840, rounded to the nearest integer. The exact value is $(0.954\ 172\ldots)10^{-20} \approx 10^{-20}$. Definitely the emerging pattern is confusing, and all three approximations are utterly meaningless. This is due to the fact that the step-size $h = 0.1$ was large for this problem and for all three processes used. We will see that our imbedded formulas can be used to detect and discard step-sizes which are too large.

The solution of the second initial-value problem, $y = 1/(1 - 10x)$, is not defined at $x = 0.1$, and for all $x > 0.1$, $y < 0$. Yet with $h = 0.1$, Euler's method and the classical fourth-order Runge–Kutta formula yield 2 and 9.729 816 66... , respectively, as approximations to the solution at $x = 0.1$. With $h = 0.2$, the same formulas yield 3 and 878 as approximations to $y(0.2) = -1$. A single application of our main formula (2) and the imbedded formulas (3) and (4) would reveal from the start that either of these step-sizes is too large.

Before applying our imbedded formulas to these problems, we give a rule which makes the selection of an appropriate step-size faster and easier.

Rule of Thumb. For the initial-value problem (1), the formula

$$h_t = \min\{|y_0/f_0|, 1\}, \quad f_0 = f(x_0, y_0) \neq 0$$

yields a useful trial step-size h_t to begin the application of any Runge–Kutta algorithm.

It should be pointed out that this simple but useful rule does not require any additional evaluations of the directional function f, since f_0 constitutes the first stage of any Runge–Kutta algorithm, regardless of its order and the *step-size* used.

Example 1. In spite of its simple appearance, the initial-value problem $dy/dx = -30y$, $y(0) = \tfrac{1}{3}$, is tricky and requires particularly careful selection of the step-size. If one chooses blindly $h = 0.2$, a single application of our formulas (2) and (4) gives

$$y(0.2) = 0.000\,826\ldots,$$
$$\tilde{y}_6(0.2) = -0.782\,502\ldots,$$
$$\tilde{y}_4(0.2) = 1.517\,190\ldots.$$

(The exact value is listed for comparison.) The sixth- and fourth-order approximations do not even agree in sign. Definitely the chosen step-size is too large. The Rule of Thumb gives $h_t = 1/30 = 0.0333\ldots < 0.2$. Thus the use of the Rule of Thumb would prevent waste by indicating from the start that 0.2 as well as 0.1 are larger than 1/30 and should be discarded.

The choice of $h = 0.02 < 1/30$ is in agreement with the above rule. With this step-size and a single application of the imbedded formulas, we find at $x = x_0 + ch = 0.02c$, $c =$ 0.2, 0.4, 0.6, 0.8, 1 (E_i and e_i refer to the corresponding absolute and relative errors) (see Table 1).

Notice that at $x = 0.02$ ($c = 1$) we find $|\tilde{y}_6(0.02) - \tilde{y}_4(0.02)| = 0.000\,004\ldots$; thus it may be assumed that $\tilde{y}_4(0.02)$, and consequently $\tilde{y}_6(0.02)$ which is more accurate, are approximations to $y(0.02)$ correct to five significant figures. Actually, the table shows that $\tilde{y}_6(0.02)$ is correct to six significant figures. Observe that this accuracy is nearly maintained throughout the interval.

It should be pointed out that the partition of the interval [0, 0.02] into five equal subintervals was arbitrary. One may divide this interval into any number of equal or unequal parts. The partitioning used here as an illustration lends itself admirably to block integration by six-point formulas as specified, for instance, by Milne[5] and Rosser[6]. (However, there is a discrepancy between the two blocks of formulas listed by these authors, specifically, formulas 19-10 of Milne[5, p. 48] and A20 of Rosser[6, p. 447], which should have been identical. It appears that Rosser's coefficients are the correct ones.) These matters need to be investigated in detail, but that is outside of the scope of this paper. The present authors plan to treat block integration, and other topics to which our continuously imbedded Runge–Kutta formulas may be advantageously applied, in a subsequent paper.

Comparison of the continuous fourth- and fifth-order approximations allows easy run-time adjustment of the step-size. We used such a (double precision) adaptive process for this example and obtained

$$\tilde{y}_6(1.5) = 1.134\,85 \times 10^{-20}$$

as an approximation to

$$y(1.5) = 0.954\,172 \times 10^{-20}.$$

Table 1.

c, x	$y, \tilde{y}_5, \tilde{y}_4$	E_5, E_4	e_5, e_4
0.2	0.295 640 145 572...		
0.004	0.295 639 929 827...	0.000 000 215 745 2...	0.000 000 729 756 2...
	0.295 639 612 898...	0.000 000 532 674 3...	0.000 001 801 766 0...
0.4	0.262 209 287 022...		
0.008	0.262 209 132 681...	0.000 000 154 340 9...	0.000 000 588 617 5...
	0.262 208 921 273...	0.000 000 365 748 9...	0.000 001 394 874 2...
0.6	0.232 558 775 357...		
0.012	0.232 558 554 371...	0.000 000 220 985 6...	0.000 000 950 235 8...
	0.232 558 322 298...	0.000 000 453 058 9...	0.000 001 948 148 3...
0.8	0.206 261 130 602...		
0.016	0.206 260 426 438...	0.000 000 704 163 3...	0.000 003 413 940 9...
	0.206 260 025 568...	0.000 001 105 033 8...	0.000 005 357 450 7...
1.0	0.182 937 212 031...		
0.02	0.182 937 385 960...	0.000 000 173 928 7...	0.000 000 950 756 6...
	0.182 941 386 436...	0.000 004 174 404 6...	0.000 022 818 783 6...

Table 2.

c, x	$y, \bar{y}_5, \bar{y}_4$	E_5, E_4
0.9	10.000 000...	2.
0.09	7.510 019...	2.489 980...
	6.440 026...	3.559 973...
1	undefined	
0.10	15.927 508...	...
	9.168 115...	...
1.1	−10.000 000...	
0.11	32.793 498...	42.793 498...
	12.986 622...	22.986 622...

When we programmed the Fehlberg sixth-order Runge–Kutta formula with fifth order imbedded [3] in a (double precision) adaptive routine as in [2], the best approximation we could get for this example was

$$\bar{y}(1.5) = -0.623\,998 \times 10^{-20}.$$

However, Fehlberg's formula needed a minimum step-size of 10^{-8}, whereas our formula needed a minimum step-size of only 10^{-5}.

Example 2. The application of the Rule of Thumb gives in this case $h_t = \min\{\frac{1}{10}, 1\} = 0.1$. [Recall that since the solution of this problem is $y = 1/1(1 - 10x)$, the solution fails to

Table 3.

c, x	$y, \bar{y}_5, \bar{y}_4$	E_5, E_4
−0.5	0.888 888 888...	
−0.0125	0.888 449 747...	0.000 439 141...
	0.889 378 872...	0.000 489 983...
−0.3	0.930 232 558...	
−0.0075	0.930 149 267...	0.000 083 290...
	0.930 334 022...	0.000 101 464...
−0.1	0.975 609 756...	
−0.0025	0.975 605 565...	0.000 004 190...
	0.975 615 346...	0.000 005 590...
0.1	1.025 641 025...	
0.0025	1.025 639 537...	0.000 001 488...
	1.025 643 185...	0.000 002 160...
0.3	1.081 081 081...	
0.0075	1.081 077 966...	0.000 003 114...
	1.081 085 695...	0.000 004 614...
0.5	1.142 857 142...	
0.0125	1.142 855 385...	0.000 001 756...
	1.142 858 839...	0.000 001 697...
0.7	1.212 121 212...	
0.0175	1.212 115 939...	0.000 005 272...
	1.212 126 391...	0.000 005 179...
0.9	1.290 322 580...	
0.0225	1.290 315 245...	0.000 007 335...
	1.290 299 935...	0.000 022 644...
1	1.333 333 333...	
0.025	1.333 332 047...	0.000 001 286...
	1.333 235 335...	0.000 097 997...
1.1	1.379 310 344...	
0.0275	1.379 322 258...	0.000 011 913...
	1.379 038 865...	0.000 271 478...
1.3	1.481 481 481...	
0.0325	1.481 521 132...	0.000 039 650...
	1.480 250 386...	0.001 231 095...
1.5	1.600 000 000...	
0.0375	1.599 913 082...	0.000 086 917...
	1.596 089 511...	0.003 910 488...

exist at $x = 0.1$.] With $h = 0.1$ the imbedded formulas (2)–(4) give at $x = x_0 + ch = (0.1)c$, $c = 0.9, 1, 1.1$ (see Table 2).

We find $|\bar{y}_5(0.09) - \bar{y}_4(0.09)| = 1.069\ldots$ (with $c = 1$), $|\bar{y}_6(0.1) - \bar{y}_4(0.1)| = 6.759\ldots$, $|\bar{y}_5(0.11) - \bar{y}_4(0.11)| = 19.816\ldots$. These results indicate that the chosen step-size, $h = 0.1$, is substantially large.

With $h = 0.05$, a single application of the imbedded formulas yields

$$y(0.05) = 2.000\,000\ldots,$$
$$\bar{y}_6(0.05) = 1.999\,360\ldots,$$
$$\bar{y}_4(0.05) = 1.989\,871\ldots.$$

Since $|\bar{y}_6(0.05) - \bar{y}_4(0.05)| = 0.009\,489\ldots$, we assume that $\bar{y}_6(0.05)$ is correct to one decimal place. Actually the approximation in question is correct to two decimal places.

Let us suppose that more accurate approximations are desired. With $h = 0.025$, a single application of the imbedded formulas yields at $x = x_0 + ch = (0.025)c$, $c = -0.5, -0.3, -0.1, 0.1, 0.3, 0.5, 0.7, 0.9, 1, 1.1, 1.3, 1.5$ (see Table 3).

We see at the very first application of the imbedded sixth-order formula ($c = 1$) we have $\bar{E}_6(0.025) = |\bar{y}_6(0.025) - \bar{y}_4(0.025)| = 0.000\,096\,712\ldots$ as an estimate to $E_6 = 0.000\,001\,286\ldots$, the absolute error in $\bar{y}_6(0.025)$. Thus $\bar{E}_6$ constitutes a conservative but safe estimate for E_6. Notice also that c is allowed to take values smaller than 0 and larger than 1; nevertheless satisfactory approximations are obtained even outside of the interval $[x_0, x_0 + h]$.

Table 4.

c, x	$y', (\bar{y}_5)'_4, (\bar{y}_4)'_3$	$\lvert y' - (\bar{y}_5)'_4\rvert, \lvert y' - (\bar{y}_4)'_3\rvert$
−0.5	7.901 234...	
−0.0125	8.022 732...	0.121 498...
	7.774 045...	0.127 189...
−0.3	8.653 326...	
−0.0075	8.687 298...	0.033 971...
	8.613 766...	0.039 559...
−0.1	9.518 143...	
−0.0025	9.522 244...	0.004 100...
	9.512 774...	0.005 369...
0.1	10.519 395...	
0.0025	10.518 563...	0.000 831...
	10.520 633...	0.001 237...
0.3	11.687 363...	
0.0075	11.687 619...	0.000 256...
	11.686 903...	0.000 459...
0.5	13.061 224...	
0.0125	13.061 146...	0.000 077...
	13.061 146...	0.000 077...
0.7	14.692 378...	
0.0175	14.691 255...	0.001 122...
	14.692 926...	0.000 548...
0.9	16.649 323...	
0.0225	16.650 425...	0.001 102...
	16.631 804...	0.017 519...
1.0	17.777 777...	
0.025	17.781 630...	0.003 852...
	17.731 892...	0.045 885...
1.1	19.024 970...	
0.0275	19.031 511...	0.006 541...
	19.927 341...	0.097 628...
1.3	21.947 873...	
0.0325	21.947 738...	0.000 134...
	21.629 100...	0.318 773...
1.5	25.600 000...	
0.0375	25.532 706...	0.067 293...
	24.786 643...	0.813 356...

We now list below the approximations to $y'(x)$, $x = x_0 + ch = (0.025)c$ for the indicated values of c and obtained through the use of formulas (6) and (7) (see Table 4).

At $x = 0.025$ ($c = 1$), that is, with a single application of formulas (6) and (7), we find $|(\bar{y}_5)'_4 - (\bar{y}_4)'_3| = 0.049\,738\ldots$ as an estimate to $|y' - (\bar{y}_5)'_4| = 0.003\,852$. These results are not as accurate as the previous ones, which is not surprising because the derivative algorithms are of one unit lower order than the corresponding formulas for approximating $y(x_0 + h)$, and we are close to a point where the solution does not exist.

Finally, it should be pointed out that the continuous approximations obtained in this paper are legitimate Runge–Kutta approximations of the specified order; they are not the result of an interpolative process.

REFERENCES

1. R. Bellman and J. Casti, Differential Quadrature and Long-Term Integration. *J. Math. Anal. Appl.* **34**, 235–238 (1971).
2. R. L. Burden, J. D. Faires and A. C. Reynolds, *Numerical Analysis*. Prindle, Weber, and Schmidt, Boston (1981).
3. Erwin Fehlberg, Classical Fifth-, Sixth-, Seventh-, and Eighth-order Runge–Kutta Formulas with Stepsize Control. NASA Technical Report R-287 (October 1968).
4. A. Huta, Une Amélioration de la Méthod de Runge–Kutta–Nyström pour la Résolution Numérique des Équations Différentielles du Premier Ordre. *Acta Fac. Rerum. Nat. Univ. Comenian. Math.* **1**, 201–224 (1956).
5. W. E. Milne, *Numerical Solution of Differential Equations*. Wiley, New York (1953).
6. J. B. Rosser, A Runge–Kutta for all Seasons. *SIAM Rev.* **9**, 417–452 (1967).
7. D. Sarafyan, Continuous Approximate Solution of Ordinary Differential Equations and Their Systems. *Comp. Math. Appl.* **10**, 139–159 (1984).
8. D. Sarafyan, C. L. Outlaw and L. J. Derr, An Investigation of Runge–Kutta Processes, and Equivalence of Scalar and Vector Cases. *J. Math. Anal. Appl.* **104**, 568–588 (1984).