1996

An Efficient Runge-Kutta (4,5) pair

P. Bogacki
Old Dominion University

L. F. Shampine

Original Publication Citation

Bogacki, P., & Shampine, L. F. (1996). An efficient Runge-Kutta (4,5) pair. *Computers & Mathematics with Applications, 32*(6), 15-28. doi:10.1016/0898-1221(96)00141-1

Pergamon

Computers Math. Applic. Vol. 32, No. 6, pp. 15–28, 1996

0898-1221/96 $15.00 + 0.00

PII: S0898-1221(96)00141-1

An Efficient Runge-Kutta (4,5) Pair

P. BOGACKI
Department of Mathematics and Statistics, Old Dominion University
Norfolk, VA 23529, U.S.A.
bogacki@math.odu.edu

L. F. SHAMPINE
Mathematics Department, Southern Methodist University
Dallas, TX 75275, U.S.A.
shampine@na-net.ornl.gov

(Received and accepted May 1996)

Abstract—A pair of explicit Runge-Kutta formulas of orders 4 and 5 is derived. It is significantly more efficient than the Fehlberg and Dormand-Prince pairs, and by standard measures it is of at least as high quality. There are two independent estimates of the local error. The local error of the interpolant is, to leading order, a problem-independent function of the local error at the end of the step.

Keywords—Ordinary differential equations, Runge-Kutta, RKSUITE, Continuous extension, Local error.

1. INTRODUCTION

Fourth-order explicit Runge-Kutta formulas have always been popular for the solution of the initial value problem for a first-order system of ordinary differential equations (ODEs)

$$y' = f(x, y), \qquad a \leq x \leq b, \quad y(a) \text{ given.}$$

The error made in a step, the local error, is estimated by taking each step with a fifth-order formula and estimating the error in the fourth-order formula by comparison. A natural measure of the cost of a Runge-Kutta formula is the number of stages involved—the number of times $f(x, y)$ is evaluated. By embedding the evaluation of one formula in the other, it is possible to make evaluation of the pair very much cheaper than separate evaluation of the individual formulas. At least six stages are needed for a fifth-order formula, and it is possible to derive pairs that require only six stages.

The landmark paper of Hull *et al.* [1] considers how to assess the effectiveness of methods for the numerical solution of ODEs. There, the six stage F(4,5) pair due to Fehlberg [2] proved to be very effective. Provided that the stability of the fifth-order formula is acceptable, advancing the integration with the higher-order result, called local extrapolation [3], results in a more accurate integration at no additional cost. The comparison [4] shows the considerable advantages of implementing F(4,5) in this way. For quite some time the F(4,5) pair in local extrapolation mode

The authors are grateful for the advice of I. Gladwell that has considerably improved this paper.

was generally accepted as the best way to proceed at these orders. Dormand and Prince [5] achieved a considerable improvement by exploiting the idea of FSAL, First Same As Last. The idea is to form the result for advancing the integration, form the first stage of the *next* step, and use this stage as a last stage in the current step for the formation of the other formula of the pair. Provided that the step is a success, an extra stage is obtained for "free" in this way. Theoretical arguments [6] and experience say that the Dormand-Prince pair, DP5(4)7M, is considerably more efficient than F(4,5).

In this paper, we present the BS(4,5) pair that represents about as great an increase in efficiency over the Dormand-Prince pair as that pair represents over the Fehlberg pair. The stability of the new pair is about the same as DP5(4)7M on an equal cost basis. A very unusual aspect of the new pair is that we provide two fourth-order formulas so as to obtain two independent estimates of the local error and enhance the robustness of error control. These and other improvements are obtained by going to a pair that involves seven stages. Subsequently, other authors [7] recognized the advantages of using one more than the minimal number of stages.

A Runge-Kutta formula starts with an approximate solution of the differential equation at a point x_n, and computes an approximation at $x_{n+1} = x_n + h$. It is possible to derive a family of Runge-Kutta formulas depending on a parameter σ such that the member corresponding to σ provides an approximate solution at $x_n + \sigma h$, an "interpolant." By reusing the stages formed in the course of the basic step to x_{n+1}, at most a few additional stages are needed for the evaluation of all members of the family. Horn [8] derived such a scheme for F(4,5). One practical issue is how smoothly the interpolant for $[x_n, x_{n+1}]$ connects with the interpolant for an adjacent interval $[x_{n+1}, x_{n+2}]$. Horn's interpolant is not even continuous, but globally C^1 interpolants are now available [9]. Shampine and his coworkers [10,11], produced a number of interpolants for the Dormand-Prince pair, as did Dormand and Prince themselves [12]. At present, the best interpolant for that pair appears to be that of Calvo, Montijano and Randez [13]. The accuracy of these interpolants depends on the problem, but there are interpolants [11] with an error at $x_n + \sigma h$ that is a known multiple of the error at $x_n + h$ *independent* of the problem, at least asymptotically. Recent research [14] exploits such interpolants for control of the defect. We have derived an interpolant for the BS(4,5) pair that has this very desirable property.

In a section devoted to numerical tests we report some experiments of our own made with a version of the well-known code RKF45 [15] modified so that we could compare several pairs of formulas. We also describe briefly some substantial experiments of Kraut [16]. She compared the state-of-the-art suite of explicit Runge-Kutta codes RKSUITE [17–19], to codes in the NAG [20], and IMSL [21] libraries. The BS(4,5) pair is the standard choice in RKSUITE. Kraut's experiments and other experience have shown RKSUITE to be so effective that it was added to the IMSL library, and it replaced the explicit Runge-Kutta code of the NAG library. In this sense, the BS(4,5) pair has been a very successful pair.

2. PRELIMINARIES

We consider the initial value problem for a system of ordinary differential equations

$$y'(x) = f(x, y(x)), \qquad a \le x \le b, \tag{1}$$

$$y(a) \text{ given.} \tag{2}$$

A Runge-Kutta process produces a sequence of approximations $\hat{y}_n$ to $y(x_n)$ for $a = x_0 < x_1 < \cdots < x_N = b$. Each step from x_n to $x_{n+1} = x_n + h$ involves two approximations to $y(x_{n+1})$, namely $\hat{y}_{n+1}$ and y_{n+1}. The pair of formulas used for this purpose has the form

$$\hat{y}_{n+1} = \hat{y}_n + h \sum_{i=1}^{s} \hat{b}_i k_i, \tag{3}$$

$$y_{n+1} = \hat{y}_n + h\sum_{i=1}^{s} b_i k_i, \tag{4}$$

where the stages

$$\begin{aligned} k_1 &= f(x_n, \hat{y}_n), \\ k_i &= f\left(x_n + c_i h, \hat{y}_n + h\sum_{j=1}^{i-1} a_{i,j} k_j\right), \qquad i = 2, 3, \ldots, s, \end{aligned} \tag{5}$$

and

$$c_i = \sum_{j=1}^{i-1} a_{i,j}, \qquad i = 2, 3, \ldots, s.$$

The local solution $u(x)$ is the solution of (1) that satisfies $u(x_n) = \hat{y}_n$. The local error of formula (3) at $(x_{n+1}, \hat{y}_{n+1})$ is $\hat{y}_{n+1} - u(x_n + h)$. The closely related concept of local truncation error refers to the solution $y(x)$ of (12). For a Runge-Kutta formula, the local error is the local truncation error associated with $u(x)$, so we, like other authors, may use either term when dealing with the solution of (1). For smooth functions f, a Taylor series expansion of this error has the form

$$\hat{y}_{n+1} - u(x_n + h) = \sum_{j=1}^{\infty} h^j \left\{\sum_{k=1}^{r_j} \hat{\tau}_k^{(j)} D_k^{(j)}\right\}. \tag{6}$$

The analogous expansion for the local error of formula (4) is

$$y_{n+1} - u(x_n + h) = \sum_{j=1}^{\infty} h^j \left\{\sum_{k=1}^{r_j} \tau_k^{(j)} D_k^{(j)}\right\}. \tag{7}$$

The $D_k^{(j)}$ here are elementary differentials, sums of products of partial derivatives of components of f evaluated at $(x_n, \hat{y}_n)$, and as such, they depend only on the problem. The $\hat{\tau}_k^{(j)}$ and $\tau_k^{(j)}$, are the truncation error coefficients of formulas (3) and (4), respectively; they depend only on the coefficients $(a_{i,j}, c_i, \hat{b}_i, b_i)$ defining the formulas. Explicit expressions for the truncation error coefficients are listed for $k = 1, \ldots, 6$ in [5]. Bettis and Horn [22] list such expressions for $k \leq 10$, and provide a FORTRAN program to evaluate the coefficients for a given Runge-Kutta formula.

A formula (3) of order five must satisfy the equations of condition

$$\hat{\tau}_k^{(j)} = 0, \qquad k = 1, 2, \ldots, r_j, \quad j = 1, 2, 3, 4, 5, \tag{8}$$

and a formula (4) of order four must satisfy

$$\tau_k^{(j)} = 0, \qquad k = 1, 2, \ldots, r_j, \quad j = 1, 2, 3, 4. \tag{9}$$

When y_{n+1} is of order four and $\hat{y}_{n+1}$ is of order five, it is easy to see that the difference $y_{n+1} - \hat{y}_{n+1}$ estimates the local error of the lower-order formula.

In contrast to the Dormand-Prince pair DP5(4)7M, the BS(4,5) pair that we present here does not directly assume that local extrapolation is done. In the derivation of the pair some choices were influenced by the intended mode of implementation, but the pair could be used efficiently in either mode. Derivation of an interpolant is different because, the mode of use affects the order of accuracy that is appropriate and the data that is available for the construction of the interpolant. As we feel that formula pairs should be used in local extrapolation mode, the discussion and analysis of interpolants that follows is based on this assumption.

A novelty of our approach is that we did not use the minimum number of stages ($s = 6$) necessary for a formula of order five. Instead, we took $s = 7$ in the hope that the additional

flexibility would allow a new pair to be constructed that would be more efficient than 6 stage pairs. Since our investigation, other authors [7] have recognized the advantages of an additional stage. A natural question at this point is, why just 7 stages, why not even more? Work on Runge-Kutta formulas has concentrated on measures of quality that are scaled by the cost of the formulas. Nevertheless, the absolute cost of a step must be considered. One reason for this is the cost of a failed step. A failed step with F(4,5) wastes 5 evaluations of f. The DP5(4)7M pair gains a "free" stage, only when the step is a success, so that the first evaluation of the next step is reused. On a failed step it wastes 6 evaluations of f, just like the BS(4,5) pair. In this situation the Fehlberg pair has a small advantage. (As a consequence, the "tuning" of codes based on the other pairs should place a little more emphasis on avoiding failed steps.)

Another matter is more difficult to quantify. To describe it in concrete terms, let us consider one of the situations investigated experimentally in [23]. There the efficiency of a basic pair was compared to the efficiency of a pair obtained from two steps with the basic pair. On an equal cost basis, the two pairs obviously have exactly the same behavior, but the absolute cost of a step with one pair is twice the cost of the other. Experiments show that the pair involving more stages is considerably less efficient for two reasons. One is that the step size is less frequently adapted to the behavior of the solution. The other is that the pair involving more stages must take step sizes twice as big, hence, must predict the step size twice as far into the future. The predictions are not as good for the pair involving more stages, so it has more failed steps. This is already less efficient, but it is aggravated by the large difference in the cost of a failed step. These experiments and related ones, support the plausible argument that even when two pairs have comparable properties on an equal cost basis, if one pair involves many more stages than the other pair, it will be less satisfactory in practice. It is these considerations of absolute cost that prevent us from adding many stages with a view to increasing efficiency. The difference in absolute cost between the Fehlberg and Dormand-Prince pairs has not in our experience led to practical differences of the kind described, and we anticipate that the same will be true of the small difference in cost of our pair.

The principal goals in the derivation of a Runge-Kutta pair are efficiency and stability. There is no question that stability is important, but in the typical computation the steps are chosen to yield accuracy rather than stability. For this reason we aimed to derive a pair that would be significantly more accurate than DP5(4)7M, but might be only comparable in terms of stability. With 7 stages it is possible to construct a formula of order six, so there is no question that we can construct formulas of order five that are as accurate as we wish. What is not clear is whether we can find accurate formulas with acceptable stability or that there will be an embedded acceptable fourth-order formula. We proceeded as follows. We started with a family of 7 stage formulas of order five that includes formulas of order six. There are many free parameters. To narrow the search for an accurate formula, we started from a formula of order six. There is a family of such formulas, and we first searched for an accurate, stable formula. On finding that there is a formula of order six in the family that has a stability comparable to that of the DP5(4)7M pair on an equal cost basis, we had reason to expect that we could find a formula of order five with the properties desired that had parameters not too different from those of the formula of order six. In point of fact, the fifth-order formula was constructed by modifying the sixth-order one. The result is described in the next section; here we describe the family and the search for our starting point—the sixth-order formula.

We followed the procedure of Butcher [24] to obtain a family of sixth-order, seven stage formulas with four free parameters, which we chose to be $\bar{c}_2$, $\bar{c}_3$, $\bar{c}_5$ and $\bar{c}_6$. (We use bars to distinguish quantities associated with the sixth-order formula.) We want to choose these parameters so that the formula will be very accurate. Because the local error (67) depends on the problem, no formula can minimize it for all problems. To get a formula that is "usually" accurate, it is conventional to minimize some norm of the vector of the truncation error, coefficients of the leading term of the local error. This is all somewhat vague, but the approach can be interpreted

rigorously as minimizing (to leading order) a bound on the local error [11]. Other measures of quality involve higher-order terms, and their size is measured in the same way. Like Dormand and Prince, we use a Euclidean norm, hence, concern ourselves with

$$\bar{T}_j = \sqrt{\sum_{k=1}^{r_j} \left(\bar{\tau}_k^{(j)}\right)^2}.$$

Other authors have used different norms in this context, but it seems to matter little, provided that they are used consistently.

When an explicit Runge-Kutta method is applied to the test equation $y' = \lambda y$ with step size h, $\hat{y}_{n+1} = P(z)\hat{y}_n$ where $z = \lambda h$. When $\mathrm{Re}(\lambda) \leq 0$, the equation is stable. The Runge-Kutta method is also stable in that portion of the complex plane where $\mathrm{Re}(\lambda) \leq 0$ and $|P(z)| \leq 1$, the absolute stability region of the method. For a method of order p, the stability condition can be written in terms of the coefficients defining the formula as

$$\left| \sum_{j=0}^{p} \frac{z^j}{j!} + \sum_{j=p+1}^{s} w_j z^j \right| \leq 1, \tag{10}$$

where

$$w_j = \tau_{r_j}^{(j)} + \frac{1}{j!} = \sum_{m_1=1}^{s} \cdots \sum_{m_{j-1}=1}^{s} b_{m_1} a_{m_1,m_2} a_{m_2,m_3} \cdots a_{m_{j-2},m_{j-1}} c_{m_{j-1}}.$$

It turns out that in the case of the sixth-order formulas of seven stages considered, the only undetermined quantity in (10) is $\bar{w}_7$, and this quantity has the simple form

$$\bar{w}_7 = \frac{\bar{c}_3\,(1 - 3\bar{c}_3)}{720\,(15\bar{c}_3^2 - 10\bar{c}_3 + 1)}.$$

Note that $\bar{w}_7$ depends only on one of the four free parameters available in the sixth-order formula. Searching among the values of $\bar{c}_3$ we found that if $\bar{w}_7 \approx 1/5040$ ($\bar{c}_3 \approx 2/9$ or $\bar{c}_3 \approx 1/4$), then on an equal cost basis (rescaling the region by a factor of 6/7), the stability region is almost exactly the same as that of the fifth-order formula of DP5(4)7M, except that near the imaginary axis it is somewhat better. This was an important result since it gave us a good reason to think that we could find a more accurate formula than that of Dormand and Prince, which is of at least comparable stability. With simplicity of coefficients in mind, we further investigated the two possibilities $\bar{c}_3 = 2/9$ and $\bar{c}_3 = 1/4$. A search for parameters that minimize $\bar{T}_7$ led to choosing the former value along with $\bar{c}_2 = 1/6$, $\bar{c}_5 = 2/3$, and $\bar{c}_6 = 3/4$. These values correspond to $\bar{T}_7 \approx 2.133 \times 10^{-4}$.

3. THE BS(4,5) PAIR

With a stable and accurate sixth-order formula, we can consider "spoiling" it to obtain a fifth-order formula, and furthermore, developing a companion fourth-order formula for the control of error. The pair of our choice is presented in the subroutine CONST of [17]. In this section, we discuss the criteria we considered in developing the new pair and some of its properties.

The major objectives we had in mind when looking for a fifth-order formula were that:

(i) its accuracy had to be significantly better than that of DP5(4)7M, and

(ii) its absolute stability region had to be comparable.

An important associated task is to provide an interpolant. It is a little awkward to discuss this issue. Although we have chosen to describe interpolation in a separate section, it is not independent of the construction of the basic pair—we considered other pairs with nearly the

same properties that we discarded because they were less well suited to interpolation. One of the minor goals considered might be mentioned at this point; we wanted to avoid large coefficients. The quantity $D = \max(|a_{i,j}|, |\hat{b}_i|, |b_i|, |c_i|)$ displayed in Table 1 for the three pairs shows that we were quite successful in this respect.

Although the goals (i) and (ii) seem straightforward, the first involves fundamental issues of quality. We cannot use $\hat{T}_6$ alone to measure how well goal (i) is satisfied, because it can be made arbitrarily small. The difficulty caused by an extremely small value of $\hat{T}_6$ is that the leading term of the truncation error then dominates only for very small step sizes h; for all practical purposes the formula is of sixth-order. To reveal this situation, we examine more than the leading term in the truncation error expansion. We concentrated on the ratio $\hat{T}_7/\hat{T}_6$, which we insist not to be too large, so that the formula is genuinely of fifth-order. It is necessary to specify an acceptable value for this ratio, and we chose to do this by adopting the corresponding value for DP5(4)7M, namely 9.9. Our fifth-order formula has $\hat{T}_7/\hat{T}_6 \approx 9.6$. Still higher-order terms were also investigated. Table 1 displays $\hat{T}_8$ and $\hat{T}_9$, for all three pairs. For our pair, these quantities are similar in size to $\hat{T}_7$, which we take to mean that the size of the leading term $\hat{T}_6$, is a reasonable measure of accuracy of the new formula.

Table 1. A comparison of three Runge-Kutta pairs of orders 5 and 4.

	s	$\hat{T}_6$	$\hat{T}_7$	$\hat{T}_8$	$\hat{T}_9$	B_2	C_2	D
BS(4,5)	7	.000022	.00021	.00035	.00042	1.27	1.19	1.16
DP5(4)7M	6	.00040	.0040	.0043	.0042	1.54	1.67	11.6
F(4,5)	6	.0034	.0068	.0081	.0080	3.16	1.36	8.00

The formula presented in [17] has $\hat{T}_6 \approx 2.22 \times 10^{-5}$, as compared to $\hat{T}_6^{DP} \approx 3.99 \times 10^{-4}$ and $\hat{T}_6^{F} \approx 3.36 \times 10^{-3}$. Here we use "DP" and "F," to identify quantities associated with the DP5(4)7M and F(4,5) pairs, respectively. One way of comparing the efficiency of formulas is to compare the step sizes that would yield a given accuracy ϵ, taking the cost of each step into account (cf. [6], where this is referred to as the "second measure of efficiency"). As we have observed, the Taylor series expansion (6) shows that the errors behave in a different way not just because the truncation error coefficients are different for the formulas considered, but also because these coefficients are weighted by the problem-dependent elementary differentials. To make it possible to compare the formulas' efficiency without having to refer to a specific problem, it is conventionally assumed that "on average" the errors are proportional to the norms of the leading local truncation error coefficients: $\hat{T}_6$, $\hat{T}_6^{DP}$, and $\hat{T}_6^{F}$ for the methods considered here. This assumption implies that the step size yielding the accuracy ϵ is proportional to $(\epsilon/\hat{T}_6)^{1/6}$, $(\epsilon/\hat{T}_6^{DP})^{1/6}$, and $(\epsilon/\hat{T}_6^{F})^{1/6}$, respectively. It was stated in [6] that the ratio of step size per unit cost for DP5(4)7M and F(4,5) is

$$\left(\frac{\hat{T}_6^{F}}{\hat{T}_6^{DP}}\right)^{1/6} \approx 1.43,$$

which shows a large gain in efficiency for the formula of Dormand and Prince, compared to the one of Fehlberg. This gain has been confirmed in numerical tests, and the DP5(4)7M pair has become widely accepted as the most effective pair at these orders. Comparing our pair to the Dormand-Prince pair in the same way after scaling for equal cost, we find

$$\left(\frac{6}{7}\right)\left(\frac{\hat{T}_6^{DP}}{\hat{T}_6}\right)^{1/6} \approx 1.39. \tag{11}$$

When accuracy determines the step size, as it usually does, this rough comparison suggests an improvement over DP5(4)7M comparable to the improvement that DP5(4)7M offers over F(4,5).

In constructing the fifth-order formula of [17], we were able to retain the stability properties of the sixth-order formula used as a starting point, but we were not able to improve them. The absolute stability region is defined by (10) with

$$\hat{w}_6 = \frac{17291}{12418560} \quad \text{and} \quad \hat{w}_7 = \frac{269}{1379840}.$$

When scaled for equal cost, the region closely matches that of the fifth-order formula of Dormand and Prince, for most angles $\arg(\lambda)$ in the second and third quadrants—the radius is never smaller by more than 2%, and may be greater by as much as 4%. This is the situation for most angles, but near the imaginary axis, the new formula is much better (up to about 40%). The regions are so similar that a plot contrasting them is not helpful.

We now proceed to the construction of the fourth-order companion used for error estimation. Having already chosen the values c_i and $a_{i,j}$, only one degree of freedom remains for the determination of b_1, b_3, b_4, b_5, and b_6. (Constraints on the family require that $b_2 = 0$ and $b_7 = \hat{b}_7$.) Prince and Dormand [25] list some measures of quality for the companion and the error estimator. Just as with the fifth-order formula, we want to avoid a large value of $B_2 = T_6/T_5$. As we observed earlier, a large value means that the leading term in the truncation error dominates only for very small step sizes. It is this leading term that is estimated for the control of error, so when B_2 is large, we anticipate that the error estimate might be unreliable when the step size is not very small. The quantity T_5 measures only the general size of the coefficients $\tau_1^{(5)}, \ldots, \tau_9^{(5)}$. If one of the coefficients should vanish, there would be a class of problems for which the leading term of the truncation error vanishes and the formula is of order five. Obviously, we must avoid zero coefficients, but we went farther and attempted to make the coefficients all of about the same size so that the formula would have a uniform behavior. To assess this we computed the ratio of the largest $|\tau_k^{(5)}|$ to the smallest. The value of 18 for the BS(4,5) pair is rather better than the value of 64 for F(4,5), and 74 for DP5(4)7M; this gives us reason to hope that its behavior will be a little more uniform. The error estimate itself has a Taylor series expansion that we obtain from the difference of the expansions of the local truncation error for the two formulas. The quantity

$$C_2 = \frac{\sqrt{\sum_{k=1}^{20} \left[\tau_k^{(6)} - \hat{\tau}_k^{(6)}\right]^2}}{T_5}$$

measures the dominance of the leading term of the error in the expansion of the estimate. A large value of C_2 implies that the error estimate might be unreliable when the step size is not very small.

The criteria listed so far still leave us with a lot of freedom, especially when deciding how accurate we want to make the fourth-order formula. Although these criteria prevent us from minimizing the leading error term, there is no limitation with respect to increasing it. If we make it "large," we get a small value of B_2 and a leading term in the truncation error expansion that strongly dominates subsequent terms. We have already described this as desirable, and now we need to explain why it should not be carried to an extreme. There are two common ways of measuring efficiency that are discussed in [6]. We have mentioned comparing the step sizes that would yield a given accuracy ϵ, the "second measure of efficiency." The "first measure" compares the largest step sizes that would satisfy a tolerance of ϵ on the local error. These measures are different when local extrapolation is done. The accuracy achieved is determined by the formula used to advance the step, the fifth-order formula when local extrapolation is done. The local error controlled is that of the lower-order formula, the fourth-order formula. When B_2 is exceptionally small, the other criteria imply that the fifth-order formula is very much more accurate than the fourth-order formula. Used in local extrapolation mode, this implies that the error of $\hat{y}_{n+1}$ is very much smaller than the error tolerance. A pair of Zonneveld [26] is shown by Shampine and

Watts [6] to behave in just this way. It is preferable to have a closer connection between the input tolerance and the accuracy achieved. Depending on how efficiency is measured, a very small value of B_2 can mean an inefficient pair. Indeed, it has been argued [6] that the DP5(4)7M pair is unduly conservative in this regard. Despite this argument, in our search for a fourth-order formula, we insisted that our pair have the same relation to DP5(4)7M in both measures of efficiency. Thus, we insisted that

$$\left(\frac{6}{7}\right)\left(\frac{T_5^{DP}}{T_5}\right)^{1/5} \approx \left(\frac{6}{7}\right)\left(\frac{\hat{T}_6^{DP}}{\hat{T}_6}\right)^{1/6},$$

which requires that we construct our fourth-order formula so that

$$T_5 \approx T_5^{DP}\left(\frac{\hat{T}_6}{\hat{T}_6^{DP}}\right)^{5/6} \approx 1.06 \times 10^{-4}. \tag{12}$$

The final selection of the fourth-order formula coefficients was made taking the stability properties into account. In local extrapolation mode it is not crucial to make the lower-order formula very stable, but it is desirable to match the regions reasonably well. The stability regions of the fourth- and fifth-order formulas of our pair are almost identical.

The fourth-order formula discussed here provides an error estimate of high quality, without requiring any function evaluations beyond those already needed for the fifth-order formula. However, the constraint $b_7 = \hat{b}_7$ implies that the error estimate $y_{n+1} - \hat{y}_{n+1}$ depends on function evaluations at c_1, c_3, c_4, c_5 and c_6, which range from 0 to 0.75, but not at $c_7 = 1$. If the solution should have a sharp change between $x_n + 0.75h$ and the end of the step, this might not be "noticed" by the error estimate. The matter is much more serious when solving stiff problems [27] because quasidiscontinuities are not unusual in that context. Still, we would like our method to be robust in the presence of discontinuities in f, and as was observed by Gollwitzer, the present error estimate can be deceived then. To enhance the robustness of the method, we decided to supplement the pair with an additional fourth-order formula,

$$\tilde{y}_{n+1} = \hat{y}_n + h\left(\sum_{i=1}^{7} \tilde{b}_i k_i + \tilde{b}_8 f\left(x_{n+1}, \hat{y}_{n+1}\right)\right),$$

and a corresponding error estimate that takes account of the solution at the end of the step. Because the BS(4,5) pair is not itself FSAL, we have at our disposal the first evaluation of the next step for this purpose. Notice that here we must specify that local extrapolation is to be done. When selecting the values of $\tilde{b}_i$ (see [17]), we were guided by the same criteria as those considered during the selection of the values b_i. In particular, the truncation error coefficients of the new formula also satisfy (12), with both ratios B_2 and C_2 approximately equal 1.04. Also, the stability region of the formula closely matches those of the other formulas developed here.

As implemented in RKSUITE, two error estimates are formed at every step. The first error estimate is $y_{n+1} - \hat{y}_{n+1}$. If the estimated error is too large, the step is rejected and this estimate is used for the selection of a step size for another try. If the first error test is passed, the step is completed and the second error estimate, $\tilde{y}_{n+1} - \hat{y}_{n+1}$, formed. The step is accepted or rejected and the next step size selected using this estimate. As always with FSAL, a failed step costs a function evaluation, but this is even less likely than usual for the BS(4,5) method because for a smooth problem, a failed step is almost certain to be recognized by the first estimate.

4. INTERPOLATION

We now wish to construct a family of formulas depending on a parameter σ such that the result $\hat{y}_{n+\sigma}$ approximates $y(x_{n+\sigma})$ where $x_{n+\sigma} = x_n + \sigma h$. It is permissible for $x_{n+\sigma}$ to lie outside

the interval $[x_n, x_{n+1}]$, but the formula is usually much less accurate then, so we restrict our attention to $0 \leq \sigma \leq 1$. Providing for "interpolation" complicates considerably the development of Runge-Kutta methods, and there are a number of questions about the goals and compromises that must be made. When we derived the BS(2,3) pair [28], we found that the "natural" interpolant has a remarkable property. Namely, to leading order the relative accuracy of the intermediate results is *independent* of the problem. More precisely, the accuracy at $x_{n+\sigma}$ is related to the local error controlled by the code in a known way that is independent of the problem, and the error is never worse than the local error controlled at each step. Along with its interpolant, the BS(2,3) method provides a $C^1[a,b]$ solution that can be substituted into the differential equation to define its residual, or defect. Higham [14] shows that if a Runge-Kutta method has a C^1 interpolant that has an error independent of the problem, it is possible to derive a robust, inexpensive control of the defect. In this section, we derive an interpolant for the BS(4,5) pair with properties similar to that of the BS(2,3) pair. Indeed, we go much further here because the interpolant maintains the accuracy of the higher-order result of the pair. RKSUITE implements three pairs of formulas of which only two have interpolants, namely BS(2,3) and BS(4,5). This is the only major explicit Runge-Kutta code with interpolants to which Higham's approach to error control is applicable.

One issue to be addressed in the construction of interpolants is how smoothly the interpolant for one interval connects with those of adjacent intervals. The family of formulas is constructed from the stages formed in taking the step from x_n to $x_n + h$ with the pair (34), and possibly additional stages that result in a total of s^* stages. The member corresponding to σ is

$$\hat{y}_{n+\sigma} = \hat{y}_n + \sigma h \sum_{i=1}^{s^*} \hat{b}_i(\sigma) k_i. \tag{13}$$

Here the k_i are defined as in (5), except for the indices i running from $s+1$ to s^*. The important point is that none of the stages depends on σ, so that the cost in evaluations of f of constructing this family is independent of the number of points $x_{n+\sigma}$ at which answers are desired. By restricting the $\hat{b}_i(\sigma)$ of (13) to be polynomials of degree at most 5, the right hand side of (13) is a polynomial in σ, and we shall describe it as the "interpolant" for the pair. Just as with (6), for a specific σ the local error of the formula (13) can be expanded as

$$\hat{y}_{n+\sigma} - u(x_n + \sigma h) = \sum_{j=1}^{\infty} (\sigma h)^j \left\{ \sum_{k=1}^{r_j} \hat{\tau}_k^{(j)}(\sigma) D_k^{(j)} \right\}. \tag{14}$$

For the interpolant (13) to be of order p^*, the truncation error coefficients $\hat{\tau}_k^{(j)}(\sigma)$ have to satisfy the appropriate equations of condition

$$\hat{\tau}_k^{(j)}(\sigma) = 0, \qquad k = 1, 2, \ldots, r_j, \quad j = 1, 2, \ldots, p^*, \tag{15}$$

which are related in a simple manner to (8), cf. [12].

It has become generally accepted that the interpolant should be globally C^1. To accomplish this, it is obviously necessary that the slope at the end of the step be included among the extra stages

$$k_8 = f(x_{n+1}, \hat{y}_{n+1}),$$

(hence, $c_8 = 1$, and $a_{8,i} = \hat{b}_i$ for $i = 1, 2, \ldots, 7$). Just as with the FSAL technique, this stage is "free" when the integration is continued because it is the first stage of the next step. One difference, though, is that interpolation is only done after a successful step, so there is no waste on failed steps as with FSAL. For the interpolant to be globally C^1, the polynomials $\hat{b}_i(\sigma)$ must satisfy [29]

$$\begin{aligned} \hat{b}_i(0) &= \delta_{i1}, \\ \hat{b}_i(1) &= \hat{b}_i, \\ \hat{b}_i + \hat{b}_i'(1) &= \delta_{i8}, \end{aligned} \tag{16}$$

where $\delta_{i,j}$ is the usual Kronecker delta, and $\hat{b}_i = 0$ for $i = 8, \ldots, s^*$.

Naturally, we do not want to add any more stages than necessary to do interpolation, but we should not be too concerned about the cost of extra stages: extra stages are formed only on steps where interpolation is done, and it is not likely that there will be a large number of such steps. Further, the extra stages are formed only once per step, regardless of the number of interpolations to be done at points in the span of the step. It is natural to ask, what is the smallest number of extra stages that will provide an interpolant of acceptable quality? It turns out that with no extra stages, the highest-order possible is four. Some authors favor interpolants of this order of accuracy for general use with a (4,5) pair. Although we favor order five because we want to relate the accuracy of the interpolant to the accuracy of the result used to advance the integration, we have derived a "free" fourth-order interpolant for use when the accuracy of intermediate solution values is not critical (for details see [30]).

Adding a ninth stage to the pair in [17] permits the derivation of fifth-order interpolants. Unfortunately, the accuracy of these interpolants is not very good. When the formulas have the same order p, the accuracy of the formula producing $\hat{y}_{n+\sigma}$ can be compared to that of the basic formula producing $\hat{y}_{n+1}$ in the conventional way. Thus, we introduce

$$\hat{T}_j(\sigma) = \sigma^j \sqrt{\sum_{k=1}^{r_j} \left(\hat{\tau}_k^{(j)}(\sigma)\right)^2},$$

and look at $\hat{T}_6(\sigma)/\hat{T}_6$ to assess the accuracy of the interpolant. The error of $\hat{y}_{n+1}$ is estimated and the step rejected if this error is bigger than a specified tolerance. The step size is then adjusted so that the predicted error of the next try will be smaller than, but comparable to, the tolerance, and the step is repeated until it succeeds. By relating the accuracy of the interpolated value $\hat{y}_{n+\sigma}$ to that of an accepted solution $\hat{y}_{n+1}$, we can assess the accuracy at points interior to the span of the step. For this reason we do not want to have a ratio that is much bigger than 1 for any σ in $[0,1]$. There is no point being too fastidious about this though, because the way we compare accuracy is very rough, and the relative accuracy certainly depends on the problem. With $s^* = 9$ we were able to construct interpolants for which

$$\max_{0\le\sigma\le 1} \frac{\left[\hat{T}_6(\sigma)\right]}{\hat{T}_6}$$

is about 12, and this is the best that can be done. With $s^* = 10$ the most accurate interpolants we found have a ratio of about 3. We did not consider these interpolants to be sufficiently accurate, hence, resorted to $s^* = 11$ to get a satisfactory interpolant. With this many stages we can make an important qualitative improvement in the interpolant that we explain in a moment. Considering the number of stages required to get an acceptable interpolant with the other pairs and the extra stage we have at our disposal, it was surprising to us that so many extra stages appeared to be necessary. We attribute this to the greater demands placed on the accuracy of the interpolant by the extremely small truncation error coefficients of our fifth-order formula.

We would describe an "ideal" interpolant as one for which

$$\|\hat{y}_{n+\sigma} - u(x_n + \sigma h)\| \le \|\hat{y}_{n+1} - u(x_n + h)\|, \qquad \text{for all } 0 \le \sigma \le 1. \tag{17}$$

Besides the obvious wish for accurate approximations, this property means that the usual control of the error at the end of the step controls the error of the interpolant as well. It seems unlikely that (17) could hold for all problems and all h, but there are examples for which it is true asymptotically, one of which is BS(2,3) [11]. Considering the crudity of the conventional comparison of relative accuracy, we consider this to be a very desirable property. None of the interpolants for

F(4,5) or DP5(4)7M have it. For (17) to hold to leading order it is necessary that

$$\sigma^6 \begin{pmatrix} \hat{\tau}_1^{(6)}(\sigma) \\ \hat{\tau}_2^{(6)}(\sigma) \\ \vdots \\ \hat{\tau}_{20}^{(6)}(\sigma) \end{pmatrix} = p(\sigma) \begin{pmatrix} \hat{\tau}_1^{(6)} \\ \hat{\tau}_2^{(6)} \\ \vdots \\ \hat{\tau}_{20}^{(6)} \end{pmatrix}, \tag{18}$$

where $p(\sigma)$ is a scalar function of σ such that $|p(\sigma)| \leq 1$ for $0 < \sigma < 1$. We found that with $s^* = 11$, it is possible to construct an interpolant for the BS(4,5) pair that does satisfy (17) to leading order. Even more is true; the error of $\hat{y}_{n+\sigma}$ is, to leading order, precisely $|p(\sigma)|$ times the error of $\hat{y}_{n+1}$, and we know what $p(\sigma)$ is. The interpolant with $s^* = 11$ is presented in the subroutine CONST of [17]. Recall that the quantity D presented in Table 1 is a bound on the magnitude of the coefficients defining the basic pair. The interpolant was constructed so that the coefficients $\hat{b}_i(\sigma)$ of (13) defining the interpolated value $\hat{y}_{n+\sigma}$ satisfy the same bound D. The quantity $p(\sigma)$ is plotted in Figure 1, where it can be seen that the leading term of the error has a very smooth behavior throughout the step. Note that $p(\sigma)$ does not vanish on $(0, 1)$; if it did, the fifth-order interpolant would degenerate there to a formula of order six.

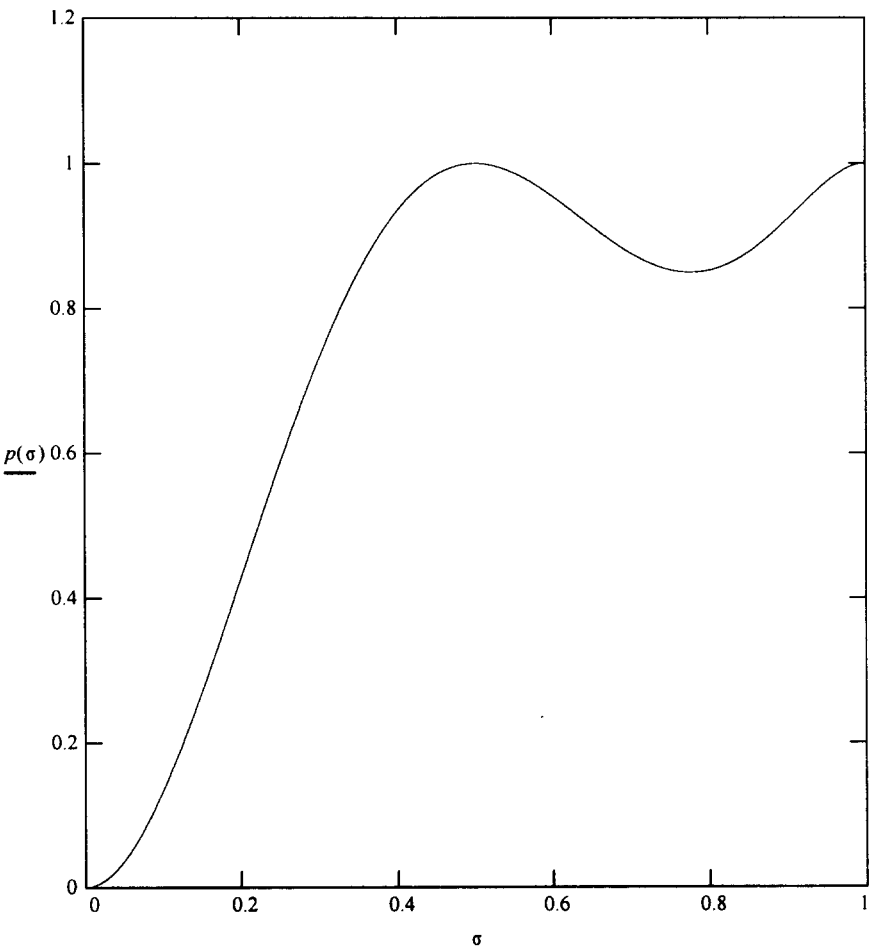

Figure 1.

Throughout this investigation we have been cautious about the assumption that the leading term dominates in the truncation error expansion. To assess the effect of higher-order terms we resort to the crude comparison of accuracy in terms of the relative sizes of truncation error co-

efficients. Specifically, we considered the seventh-order terms by examining the ratio $\hat{T}_7(\sigma)/\hat{T}_6(\sigma)$. The value of this ratio at $\sigma = 1$ is determined by the basic pair; it is about 9.6. We would like the ratio to be, not much larger for $0 < \sigma < 1$. A numerical investigation revealed no value of σ where it is bigger than 9.6. This increases our confidence that the interpolated value should be at least as accurate in the span of the step, as the result at the end of the step.

5. NUMERICAL TESTS

We believe that understanding of the quality of Runge-Kutta formulas has progressed to the point that the theoretical measures considered in the construction of our pair, do indicate how formulas will perform in practice. Of course, we claim only that a rough assessment of relative efficiency is provided by the theory. Because efficiency depends on the problem and the quality of the implementation of the formulas, we consider the role of numerical tests to be a confirmation of the theoretical predictions. We report here two substantial sets of experiments that are entirely consistent with our predictions.

We carried out one set of experiments ourselves [31]. It is generally accepted that F(4,5) is significantly less efficient than DP5(4)7M, so we compared these two pairs at the same time that we compared the Dormand-Prince pair to our own. The fact that the comparison of F(4,5) to DP5(4)7M is consistent with a considerable body of experience increases our confidence in the numerical comparison of DP5(4)7M to BS(4,5). We used two sets of problems. Six problems from [32], II.10 make up one set. The second problem in this set is the two body problem with eccentricity 0.5. The two body problem with eccentricities 0.1, 0.3, 0.5, 0.7, and 0.9 form an important family in the nonstiff DETEST test set of [1], so we added the other eccentricities as a second set of test problems. We modified the RKF45 code of Watts and Shampine [15] so that any one of the Fehlberg, Dormand-Prince, or Bogacki-Shampine pairs might be used. With each pair all the problems were integrated with a pure absolute error control for tolerances $10^{-3}, \ldots, 10^{-12}$. The maximum norm of the error at the end of the interval of integration was computed using "exact" values obtained with a high-order Runge-Kutta pair, and a tolerance of 10^{-14}. Plots of the efficiency of solution are found in [31]. To relate the computations to the theory, we want the relative cost of achieving a given error. Because the error achieved is not the same as the tolerance input, we had to interpolate the data gathered to determine these costs. This presents some difficulties at both extremes of the range of tolerances, because all three pairs had to achieve the accuracy if we were to compute the relative costs. Table 2 presents the mean of the relative costs for all the tolerances where the computations were meaningful, and the standard deviation of these costs for the first set of problems. We would anticipate that the relative efficiency of two pairs depends on the problem solved, but only weakly on the accuracy achieved for a given problem. As expected, there is an erratic dependence on the accuracy, especially at tolerances for which the absolute cost is small so that a difference of even one step affects the results. Theoretical arguments and computational experience say that the DP5(4)7M is significantly more efficient than F(4,5). The same theory says that BS(4,5) is about equally more efficient than DP5(4)7M. The numerical results of Table 2 are consistent with these predictions.

The very substantial tests of Kraut [16] had a different goal. By the time of her tests the BS(4,5) pair had been implemented, along with a (2,3) pair and a (7,8) pair, in a suite of production-grade Runge-Kutta codes called RKSUITE [17–19]. Her goal was to compare RKSUITE to the explicit Runge-Kutta codes of the widely-used IMSL [21], NAG [20], and SLATEC [33] libraries. As it happens, the code in the SLATEC library is a variant of the RKF45 code used in our tests. Kraut used the problems of our two test sets plus a discontinuous problem from [1], and two stiff problems from the stiff DETEST test set [34]. The last problems were added to those we used so as to investigate how well the codes responded to difficulties. She used a wider range of tolerances, partly to investigate how well the codes cope with crude tolerances and very stringent tolerances. Kraut also investigated the effectiveness of interpolation in the codes that have the

Table 2. Relative cost to achieve a given accuracy with F(4,5), DP5(4)7M, and BS(4,5).

Problem	nfe^F/nfe^{DP}		nfe^{DP}/nfe^{BS}	
	mean	std	mean	std
Jacobian elliptic functions	1.37	0.09	1.57	0.07
Two body problem	1.48	0.06	1.45	0.42
van der Pol equation	0.97	0.16	1.55	0.08
Brusselator	1.55	0.11	1.95	0.21
Hanging string	1.28	0.02	1.80	0.26
Pleiades	1.27	0.18	1.11	0.06

capability. The various codes implement formulas of different orders, so Kraut drew conclusions for three ranges of tolerances. Full details of her tests can be found in [16]. Here it will suffice to quote her conclusion that "For moderate accuracy requests the BS(4,5) pair is more efficient than the corresponding codes in the NAG, SLATEC, and IMSL libraries. ... No code without an interpolation capability, and no formula pair like the PD(7,8) pair in RKSUITE can compete with pairs that do have this capability, like the BS(2,3) and BS(4,5) pairs, when dense output is required." Subsequently RKSUITE was added to the IMSL library and it replaced the explicit Runge-Kutta code of the NAG library.

REFERENCES

1. T.E. Hull, W.H. Enright, B.M. Fellen and A.E. Sedgwick, Comparing numerical methods for ordinary differential equations, *SIAM J. Numer. Anal.* **4**, 603–637, (1972).
2. E. Fehlberg, Klassische Runge-Kutta-Formeln vierter und niedriger Ordnung mit Schrittweiten-Kontrolle und ihre Anwendung auf Wärmeleitungsprobleme, *Computing* **7**, 61–71, (1970).
3. L.F. Shampine, Local extrapolation in the solution of ordinary differential equations, *Math. Comp.* **27**, 91–97, (1973).
4. L.F. Shampine, H.A. Watts and S.M. Davenport, Solving nonstiff ordinary differential equations—The state of the art, *SIAM Review* **18**, 376–411, (1976).
5. J.R. Dormand and P.J. Prince, A family of embedded Runge-Kutta formulas, *J. Comp. Appl. Math.* **6**, 19–26, (1980).
6. L.F. Shampine, Some practical Runge-Kutta formulas, *Math. Comp.* **46**, 135–150, (1986).
7. P.W. Sharp and E. Smart, Explicit Runge-Kutta pairs with one more derivative evaluation than the minimum, *SIAM J. Sci. Comput.* **14**, 338–348, (1993).
8. M.K. Horn, Fourth- and fifth-order, scaled Runge-Kutta algorithms for treating dense output, *SIAM J. Numer. Anal.* **20**, 558–568, (1983).
9. W.H. Enright, K.R. Jackson, S.P. Nørsett and P.G. Thomsen, Interpolants for Runge-Kutta formulas, *ACM Trans. Math. Soft.* **12**, 193–218, (1986).
10. L.F. Shampine, Interpolation for Runge-Kutta methods, *SIAM J. Numer. Anal.* **22**, 1014–1027, (1985).
11. I. Gladwell, L.F. Shampine, L. Baca and R.W. Brankin, Practical aspects of interpolation in Runge-Kutta codes, *SIAM J. Sci. Stat. Comput.* **8**, 322–341, (1987).
12. J.R. Dormand and P.J. Prince, Runge-Kutta triples, *Computers Math. Applic.* **12A** (9), 1007–1017, (1986).
13. M. Calvo, J.I. Montijano and L. Randez, New continuous extensions for the Dormand and Prince RK method, In *Computational Ordinary Differential Equations,* (Edited by J.R. Cash and I. Gladwell), pp. 135–144, Clarendon Press, Oxford, U.K., (1992).
14. D.J. Higham, Robust defect control with Runge-Kutta schemes, *SIAM J. Numer. Anal.* **26**, 1175–1183, (1989).
15. L.F. Shampine and H.A. Watts, Practical solution of ordinary differential equations by Runge-Kutta methods, Rept. SAND 76-0585, Sandia National Laboratories, Albuquerque, NM, (1976).
16. G.L. Kraut, A comparison of RKSUITE with Runge-Kutta codes from the NAG, SLATEC, and IMSL libraries, Math. Dept., Southern Methodist University, Dallas, TX, (1991).
17. RKSUITE: **http://www.netlib.org/ode/rksuite/**.
18. R.W. Brankin, I. Gladwell and L.F. Shampine, RKSUITE: A suite of Runge-Kutta codes for the initial value problem for ODEs, Softreport 91-1, Math. Dept., Southern Methodist University, Dallas, TX, (1991).
19. R.W. Brankin, I. Gladwell and L.F. Shampine, RKSUITE: A suite of explicit Runge-Kutta codes, World Scientific, Singapore, (1993).
20. NAG Fortran Library Manual, Mark 13, NAG Ltd., Wilkinson House, Jordanhill Road, Oxford, U.K., (1988).
21. IMSL Math/Library User's Manual, Version 1.1, IMSL, 2500 Park West Tower One, 2500 City West Boulevard, Houston, TX, (1989).

22. D.G. Bettis and M.K. Horn, Computation of truncation error terms for Runge-Kutta methods, Math. Inst. report, Tech. Univ. Munich, (private communication).
23. L.F. Shampine, The quality of Runge-Kutta formulas, Rept. SAND 76-0370, Sandia National Laboratories, Albuquerque, NM, (1976).
24. J.C. Butcher, *The Numerical Analysis of Ordinary Differential Equations*, Wiley, Chichester, U.K., (1987).
25. P.J. Prince and J.R. Dormand, High order embedded Runge-Kutta formulas, *J. Comp. Appl. Math.* **7**, 67–75, (1981).
26. J.A. Zonneveld, *Automatic Numerical Integration*, Math. Centre Tracts 8, Amsterdam, (1964).
27. L.F. Shampine, Implementation of Rosenbrock methods, *ACM Trans. Math. Software* **8**, 93–113, (1982).
28. P. Bogacki and L.F. Shampine, A 3(2) pair of Runge-Kutta formulas, *Appl. Math. Lett.* **2** (4), 1–9, (1989).
29. J.R. Dormand, M.A. Lockyear, N.E. McGorrigan and P.J. Prince, Global error estimation with Runge-Kutta triples, *Computers Math. Applic.* **18** (9), 835–846, (1989).
30. P. Bogacki, Efficient Runge-Kutta pairs and their interpolants, Ph.D. Thesis, Math. Dept., Southern Methodist University, Dallas, TX, (1990).
31. P. Bogacki and L.F. Shampine, An efficient Runge-Kutta (4,5) pair, Rept. 89-20, Math. Dept., Southern Methodist University, Dallas, TX, (1989).
32. E. Hairer, S.P. Nørsett and G. Wanner, *Solving Ordinary Differential Equations I Nonstiff Problems*, Springer, Berlin, (1987).
33. B.L. Buzbee, The SLATEC common mathematical library, Chapter 11, In *Sources and Development of Mathematical Software*, (Edited by W.R. Cowell), Prentice Hall, Englewood Cliffs, NJ, (1984).
34. W.H. Enright, T.E. Hull and B. Lindberg, Comparing numerical methods for stiff systems for O.D.E.'s, *BIT* **15**, 10–48, (1975).