A Modification of the Runge-Kutta Fourth-Order Method

By E. K. Blum

1. Introduction. Consider the system of n first-order ordinary differential equations,

$$y_k' = f_k(t, y_1(t), \cdots, y_n(t)), \qquad k = 1, \cdots, n, \tag{1.1}$$

with the initial values,

$$y_k(t_0) = a_k . \tag{1.2}$$

Under suitable conditions on the f_i, a unique solution of (1.1) satisfying (1.2) exists for some interval, $t_0 \leqq t \leqq b$. For example, it is sufficient that the f_i be continuous and satisfy a Lipschitz condition in some neighborhood of the initial point, $(t_0, a_1, \cdots, a_n)$. We shall assume that such conditions obtain, so that the initial value problem (1.1), (1.2) has a unique solution.

To simplify the notation, we define $y_0 \equiv t$ and $f_0 \equiv 1$. We now let y be the vector, $(y_0, y_1, \cdots, y_n)$, and f the vector-valued function, $(f_0, f_1, \cdots, f_n)$. The initial value problem can then be written as

$$y' = f(y), \tag{1.3}$$

$$y(t_0) = a. \tag{1.4}$$

The Runge-Kutta fourth-order method for the numerical solution of (1.3), (1.4) yields approximate values, y_j, of y on a finite set of points, $t_j = t_0 + jh$, $j = 1, 2, \cdots, m$. It is usually summarized in formulas (1.5)–(1.9) below, which specify the calculations to be carried out for each integration step; i.e. for each value of j.

$$k_1 = hf(y_j), \tag{1.5}$$

$$k_2 = hf(y_j + k_1/2), \tag{1.6}$$

$$k_3 = hf(y_j + k_2/2), \tag{1.7}$$

$$k_4 = hf(y_j + k_3), \tag{1.8}$$

$$y_{j+1} = y_j + (k_1 + 2k_2 + 2k_3 + k_4)/6. \tag{1.9}$$

A variant of this method was derived by S. Gill [1]. The two advantages of Gill's variant are (1) in automatic computers, it requires $3n + B$ storage registers whereas the Runge-Kutta formulas as given above, require $4n + B$, where B is some constant; (2) the computation can be arranged so that rounding errors are reduced appreciably. In the present paper, we shall show how, by means of a fairly simple modification of (1.5)–(1.9), both of these advantages can be made to accrue to the classical Runge-Kutta method. All the constants in this modification are rational, whereas Gill's variant contains some irrational constants. The modifica-

Received September 18, 1961.

tion is achieved by extracting from Gill's method its main virtue, the rather ingenious device for reducing the rounding error, and applying it to a rearrangement of (1.5)–(1.9).

2. The Exact Modification. In an automatic digital computer, real numbers are replaced by what von Neumann and Goldstine [2] call "digital numbers," that is, by real numbers rounded to a prescribed number of digits. Further, exact arithmetic operations are replaced by "pseudo-operations" since results must be rounded. The main advantage of the modified Runge-Kutta formulas to be presented in Section 3 is that they reduce considerably the rounding error arising from the unavoidable use of digital numbers and pseudo-operations. The saving of n storage registers is a secondary consideration in large computers. The same is true of the Gill variant.

In this section we shall present a preliminary version of the proposed method. We shall refer to it as the "exact modification" since all operations will be assumed to be exact operations on real numbers. The form of the exact modification will demonstrate clearly how the saving of n storage registers is effected.

Using vector notation, as in (1.3)–(1.9), we can write the exact modification in a recursive form as follows:

$$\text{(2.1)} \qquad \begin{cases} z_0 = y_j, \\ q_0 = y_j, \\ P_0 = hf(z_0), \end{cases}$$

$$\text{(2.2)} \qquad \begin{cases} z_1 = z_0 + P_0/2, \\ q_1 = P_0, \\ P_1 = hf(z_1), \end{cases}$$

$$\text{(2.3)} \qquad \begin{cases} z_2 = z_1 + P_1/2 - q_1/2, \\ q_2 = q_1/6, \\ P_2 = hf(z_2) - P_1/2, \end{cases}$$

$$\text{(2.4)} \qquad \begin{cases} z_3 = z_2 + P_2, \\ q_3 = q_2 - P_2, \\ P_3 = hf(z_3) + 2P_2, \end{cases}$$

$$\text{(2.5)} \qquad y_{j+1} \equiv z_4 = z_3 + q_3 + P_3/6.$$

(Strictly speaking, each of the vectors, z_i, q_i, P_i, should have a second subscript, j, to indicate that the sequence (2.1)–(2.5) is repeated for each step of the solution. This subscript has been dropped for reasons of economy, just as the subscript which indicates the components of the vectors has been dropped.)

THEOREM 1. *The exact modification, (2.1)–(2.5), is equivalent to the classical Runge-Kutta method and requires only 3n + B storage registers.*

Proof. To show that (2.1)–(2.5) is equivalent to (1.5)–(1.9), we first observe that $P_0 = k_1$. Then $z_1 = y_j + k_1/2$, which implies $P_1 = k_2$. Since $q_1 = k_1$, it follows that $z_2 = (y_j + k_1/2) + k_2/2 - k_1/2 = y_j + k_2/2$. Thus,

$$P_2 = k_3 - k_2/2$$

and $q_2 = k_1/6$. From (2.4), it now follows that $z_3 = (y_j + k_2/2) + (k_3 - k_2/2) =$

$y_j + k_3$, and $q_3 = k_1/6 - (k_3 - k_2/2)$, whence $P_3 = k_4 + 2k_3 - k_2$. Combining these expressions in (2.5), we get

$$y_{j+1} = (y_j + k_3) + \left(\frac{k_1}{6} - k_3 + \frac{k_2}{2}\right) + \frac{1}{6}(k_4 + 2k_3 - k_2),$$

$$y_{j+1} = y_j + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4).$$

The order of computation of the components of the vectors z_{i+1}, q_{i+1}, and P_{i+1}, $i = 0, 1, 2, 3$, should be as follows. First, compute the components of z_{i+1} and q_{i+1} together. Each such component involves only the corresponding component of z_i, q_i, and P_i. Hence, as each component of z_{i+1} and q_{i+1} is computed, it can be placed in the storage occupied by the corresponding component of z_i and q_i, respectively. After all components of z_{i+1} and q_{i+1} have been computed, the components of P_{i+1} can be computed, replacing the corresponding components of P_i in storage. Thus, $3n + B$ storages suffice.

3. The Finite-Precision Modification. In this section we shall consider the rounding errors which arise in actual computation when digital numbers and pseudo-operations are used in (2.1)–(2.5). We shall adopt the notation of [2] for digital numbers, denoting a digital number by a letter with a bar over it, and similarly for vectors; e.g. $\bar{y}_i$ is a vector having digital numbers as components. However, we shall not introduce special symbols for pseudo-operations. Instead, we prefer to write all formulas with exact operations and introduce special terms to denote the rounding error caused by the pseudo-operations. Besides the usual arithmetic operations, we require two "shifting" operations. These are best described informally.

For the remainder of this section, let us assume that a digital number is represented by a sequence of s decimal digits, and that the decimal point is at the extreme left. (The first digit immediately to the right of the decimal point is said to be in position 1.) For m a non-negative integer, we define the operator, R_m, ("shift right m places") as follows. If $\bar{y}$ is a digital number, then $R_m\bar{y}$ is the digital number obtained by shifting the digits of $\bar{y}$ m positions to the right, "rounding off" the digits shifted into positions $s + 1, \cdots, s + m$, and inserting zeros into positions $1, \cdots, m$. The usual method of rounding off is to add (if $\bar{y} \geqq 0$) or subtract (if $\bar{y} < 0$) the digit "5" in position $s + 1$, and then drop the digits beyond position s. The operator, L_m, ("shift left m places") is defined similarly. Thus, $L_m\bar{y}$ is the digital number obtained by shifting the digits of $\bar{y}$ to the left m places, dropping those digits which are then to the left of the decimal point, and inserting zeros into positions $s - m + 1, \cdots, s$. When applied to vectors, R_m and L_m are considered to operate on each of the component digital numbers. If y is a real number, then we define $R_m y = 10^{-m}y$ and $L_m y = 10^m y$.

To briefly motivate the formulas to be given below, let us consider the exact modification, (2.1)–(2.5). If this procedure were carried out with digital numbers and pseudo-operations, an analysis of the rounding error would show that, under suitable conditions on the partial derivatives, $\partial f_k/\partial y_i$, the main source of error is in the computation of the z_i. The error there arises from the fact that q_i and P_i

are usually smaller than z_i in magnitude, since they are of order h. This means that either q_i and P_i must be computed with fewer significant digits than z_i, or else a shift right must be performed before q_i and P_i can be added to z_i. In either case an appreciable rounding error is incurred. The procedure explained below reduces this particular error. We shall refer to it as the "finite-precision modification" of the Runge-Kutta method to emphasize that it is designed for actual computation with digital numbers having a "finite precision" of s places. The finite-precision modification is derived from the exact modification by introducing the special quantities, r_i, as in Gill's formulas. To facilitate the error analysis, we shall write the finite-precision modification first with real numbers and exact operations, (3.1)–(3.5), and then with digital numbers and error terms, $\overline{(3.1)}$–$\overline{(3.5)}$.

$$\text{(3.1)} \qquad \begin{cases} z_0 = y_j, \\ L_m q_0 = L_m q_{4j}, \\ L_m P_0 = (L_m h) f(z_0), \end{cases}$$

$$\text{(3.2)} \qquad \begin{cases} \begin{cases} r_1 = R_m(\frac{1}{2} L_m P_0 - L_m q_0), \\ z_1 = z_0 + r_1, \end{cases} \\ \begin{cases} L_m q_1 = 3 L_m r_1 - (\frac{1}{2} L_m P_0 - L_m q_0), \\ L_m P_1 = (L_m h) f(z_1), \end{cases} \end{cases}$$

$$\text{(3.3)} \qquad \begin{cases} \begin{cases} r_2 = R_m(\frac{1}{2}(L_m P_1 - L_m q_1)), \\ z_2 = z_1 + r_2, \end{cases} \\ \begin{cases} L_m q_2 = -L_m r_2 - \frac{1}{3} L_m q_1 + \frac{1}{2} L_m P_1, \\ L_m P_2 = (L_m h) f(z_2) - \frac{1}{2} L_m P_1, \end{cases} \end{cases}$$

$$\text{(3.4)} \qquad \begin{cases} \begin{cases} r_3 = R_m(L_m P_2), \\ z_3 = z_2 + r_3, \end{cases} \\ \begin{cases} L_m q_3 = -L_m r_3 + L_m q_2, \\ L_m P_3 = (L_m h) f(z_3) + 2 L_m P_2, \end{cases} \end{cases}$$

$$\text{(3.5)} \qquad \begin{cases} r_4 = R_m(\frac{1}{6} L_m P_3 + L_m q_3), \\ y_{j+1} = z_4 = z_3 + r_4, \\ L_m q_{4,j+1} = 3\,[L_m r_4 - (\frac{1}{6} L_m P_3 + L_m q_3)]. \end{cases}$$

Remark 1. Regarding all operations in (3.1)–(3.5) as exact, we can replace R_m by 10^{-m} and L_m by 10^m. A straightforward calculation then shows that (3.1)–(3.5) is equivalent to (2.1)–(2.5).

Remark 2. The quantities, r_i, are redundant if all operations in (3.1)–(3.5) are considered to be exact. They play a significant role only when digital numbers and pseudo-operations are introduced.

Remark 3. The r_i require only one additional storage register rather than n. The order of computation should be as follows. First, compute together the components of r_i, z_i, and $L_m q_i$, as indicated by the inner brackets. Then the components of $L_m P_i$ can be computed. Since the computation of a component of z_i and $L_m q_i$ requires only the corresponding component of r_i, and since r_i is not used after the ith stage, one storage suffices for all components of all r_i.

Remark 4. It is obvious that the quantity, $L_m q_{4j}$, is always zero if exact operations are used. For pseudo-operations this is not the case. However, $L_m q_{40}$ can always be taken to be zero to start the computation.

In practice, the finite-precision procedure, (3.1)–(3.5), would be carried out with digital numbers and pseudo-operations. The operation "+" would be executed as pseudo-addition, and both R_m and L_m would be executed as shift operations rather than as multiplications. To analyze the rounding error, it is convenient to rewrite (3.1)–(3.5) in a mixed form, $(\overline{3.1})$–$(\overline{3.5})$, involving digital numbers, exact operations, and error terms. It is to be understood that the digital numbers are thereby treated as real decimal numbers having zeros in all positions beyond position s. The effect of pseudo-operations is shown by the presence of a single error term denoted by an expression of the form, $e(u)$.

$$\overline{(3.1)} \qquad \begin{cases} \bar{z}_0 = \bar{y}_j \\ \overline{L_m q_0} = \overline{L_m q_{4j}}\,, \\ \overline{L_m P_0} = (L_m \bar{h}) f(\bar{z}_0) + e(L_m P_0), \end{cases}$$

$$\overline{(3.2)} \qquad \begin{cases} \bar{r}_1 = R_m(\tfrac{1}{2}\overline{L_m q_0} - \overline{L_m P_0} + e(r_1), \\ \bar{z}_1 = \bar{z}_0 + \bar{r}_1\,, \\ \overline{L_m q_1} = 3L_m \bar{r}_1 - (\tfrac{1}{2}\overline{L_m P_0} - \overline{L_m q_0}) + e(L_m q_1), \\ \overline{L_m P_1} = (L_m \bar{h}) f(\bar{z}_1) + e(L_m P_1), \end{cases}$$

$$\overline{(3.3)} \qquad \begin{cases} \bar{r}_2 = R_m(\tfrac{1}{2}(\overline{L_m P_1} - \overline{L_m q_1})) + e(r_2), \\ \bar{z}_2 = \bar{z}_1 + \bar{r}_2\,, \\ \overline{L_m q_2} = -L_m \bar{r}_2 - \tfrac{1}{3}\overline{L_m q_1} + \tfrac{1}{2}\overline{L_m P_1} + e(L_m q_2), \\ \overline{L_m P_2} = (L_m \bar{h}) f(\bar{z}_2) - \tfrac{1}{2}\overline{L_m P_1} + e(L_m P_2), \end{cases}$$

$$\overline{(3.4)} \qquad \begin{cases} \bar{r}_3 = R_m(\overline{L_m P_2}) + e(r_3), \\ \bar{z}_3 = \bar{z}_2 + \bar{r}_3\,, \\ \overline{L_m q_3} = -L_m \bar{r}_3 + \overline{L_m q_2}\,, \\ \overline{L_m P_3} = (L_m \bar{h}) f(\bar{z}_3) + 2\overline{L_m P_2} + e(L_m P_3), \end{cases}$$

$$\overline{(3.5)} \qquad \begin{cases} \bar{r}_4 = R_m(\tfrac{1}{6}\overline{L_m P_3} + \overline{L_m q_3}) + e(r_4), \\ \bar{y}_{j+1} = \bar{z}_4 = \bar{z}_3 + \bar{r}_4\,, \\ \overline{L_m q_{4,j+1}} = 3[L_m \bar{r}_4 - (\tfrac{1}{6}\overline{L_m P_3} + \overline{L_m q_3})] + e(L_m q_4). \end{cases}$$

Remark 5. It is seen that $e(z_i) = 0$ for $i = 0, 1, \cdots, 4$ because the pseudo-operation of addition gives the same result as the exact operation. This is true because of our assumption that in all digital numbers the decimal point is in a fixed position. If "floating-point" numbers are used, the pseudo-operation of addition can introduce a rounding error. We shall discuss this in the next section.

We are now in a position to estimate the rounding error in (3.1)–(3.5). After some preliminaries, we shall formulate the results as Theorem 2 and its corollary.

For any quantity, u, we define $\epsilon(u) = \bar{u} - u$; i.e. $\epsilon(u)$ is the total rounding error in u. We are interested in $\epsilon(y_j)$. However, it will turn out that the quantity,

$$\bar{y}_j^* = \bar{y}_j - \frac{R_m}{3}\overline{L_m q_{4j}}\,,$$

is a better approximation to y_j than is $\bar{y}_j$. Thus, we shall consider $\epsilon(y_j^*)$ instead, where

$$y_j^* = y_j - \frac{R_m}{3} L_m q_{4j} = y_j - \frac{1}{3} q_{4j}\,.$$

By remark 4, $q_{4j} = 0$, so that $y_j^* = y_j$. Hence,

$$y_j = \bar{y}_j^* - \epsilon(y_j^*).$$

We note that

$$\epsilon(y_j^*) = \epsilon(y_j) - \frac{R_m}{3}\epsilon(L_m q_{4j}).$$

It is convenient to deal with the norm of a vector, u, which we define as

$$\| u \| = \max_k | u_k |,$$

where u_k are the components of u. For a matrix, A, with elements, a_{ik}, we define

$$\| A \| = \max_i \{ \sum_k | a_{ik} | \}.$$

In particular, we shall be concerned with matrices for which $a_{ik} = \partial f_i/\partial y_k$, where the partial derivatives are evaluated at different points for each i and k. A matrix of this type will be denoted by the symbol, "J."

THEOREM 2. *For any of the quantities, u, computed in* (3.1)–(3.5), *let the error term, $e(u)$, be subject to the condition,*

$$\| e(u) \| \leqq \frac{M}{2} 10^{-s}. \tag{i}$$

Let the bounds on the partial derivatives, $\partial f_i/\partial y_k$, be such that for any matrix, J,

$$\| J \| \leqq L. \tag{ii}$$

Let $h = 10^{-m}$, $0 < m < s$. Then the total rounding error in y_j^* incurred in one integration step is not greater than $2M10^{-s-m}$ in absolute value.

Proof. From (3.2) and (3.2) we obtain

$$\epsilon(z_1) = \epsilon(z_0) + R_m(\tfrac{1}{2}\epsilon(L_m P_0) - \epsilon(L_m q_0)) + e(r_1).$$

From (3.1), (3.1), if we assume that $h = \bar{h}$, we have

$$\epsilon(L_m P_0) = L_m(h)(f(\bar{y}_j) - f(y_j)) + e(L_m P_0).$$

Now, for each component, f_i, $i = 0, 1, \cdots, n$, we have

$$f_i(\bar{y}_j) - f_i(y_j) = \sum_{k=0}^{n} \frac{\partial f_i}{\partial y_{jk}} \epsilon(y_{jk}),$$

or, in matrix notation,

$$f(\bar{y}_j) - f(y_j) = J\epsilon(y_j).$$

This gives

$$\begin{aligned} \epsilon(L_m P_0) &= L_m(h) J \epsilon(y_j) + e(L_m P_0), \\ \epsilon(z_1) &= \epsilon(y_j) + \frac{h}{2} J\epsilon(y_j) - R_m \epsilon(L_m q_{4j}) + \frac{R_m}{2} e(L_m P_0) + e(r_1). \end{aligned} \tag{3.6}$$

Proceeding in this way, we obtain

$$\epsilon(z_2) = \left(I + \frac{h}{2}J + \frac{h^2}{4}J^2\right)\epsilon(y_j) + e(r_2) - \frac{1}{2}e(r_1) + z \tag{3.7}$$

where

$$z = \frac{hJ}{2}e(r_1) + \frac{R_m}{2}e(L_m P_1) - \frac{R_m}{2}e(L_m q_1) - \frac{hR_m}{2}J[\epsilon(L_m q_{4j}) - e(L_m P_0)],$$

$$\epsilon(z_3) = \left(1 + hJ + \frac{h^2}{2}J^2 + \frac{h^3}{4}J^3\right)\epsilon(y_j) - \frac{1}{2}e(r_1) + e(r_2) + e(r_3) \tag{3.8}$$
$$+ R_m e(L_m P_2) - \frac{R_m}{2}e(L_m q_1) + hJe(r_2) - \frac{h}{2}Je(r_1) + hJz,$$

$$\epsilon(y_{j+1}) = \epsilon(y_j) - \frac{R_m}{3}\epsilon(L_m q_{4j}) + hJv + \frac{R_m}{6}W + e(r_4), \tag{3.9}$$

where

$$v = \tfrac{1}{6}(\epsilon(y_j) + 2\epsilon(z_1) + 2\epsilon(z_2) + \epsilon(z_3)),$$

and

$$W = e(L_m P_0) + 2e(L_m P_1) + 2e(L_m P_2) + e(L_m P_3) + 6e(L_m q_2) - 2R_m e(L_m q_1).$$

Using (3.6)–(3.8), we obtain

$$v = \left(1 + \frac{h}{2}J + \frac{h^2}{6}J^2 + \frac{h^3}{24}J^3\right)\epsilon(y_j) - \frac{R_m}{3}\epsilon(L_m q_{4j}) \tag{3.10}$$
$$+ \frac{1}{12}e(r_1) + \frac{1}{2}e(r_2) + \frac{1}{6}e(r_3) + \mu_1 + \mu_2,$$

where

$$\mu_1 = \frac{R_m}{6}(e(L_m P_0) + e(L_m P_1) + e(L_m P_2)) - \frac{R_m}{4}e(L_m q_1),$$

$$\mu_2 = \frac{z}{3} + \frac{h}{6}J(e(r_2) - \frac{1}{2}e(r_1) + z).$$

From (3.5) and the fact that $q_{4j} = 0$, we obtain

$$\epsilon(L_m q_{4,j+1}) = 3L_m e(r_4) + e(L_m q_4). \tag{3.11}$$

If we multiply (3.11) by $R_m/3$ and subtract from (3.9), we obtain

$$\epsilon(y^*_{j+1}) = \epsilon(y_j^*) + hJv + \frac{R_m}{6}W - \frac{R_m}{3}e(L_m q_4). \tag{3.12}$$

Applying the properties of the norm and using conditions (i) and (ii), we get

$$\| z \| \leqq h \| J \| \cdot \| e(r_1) \| + \frac{R_m}{2} \| e(L_m P_1) \| + \frac{R_m}{2} \| e(L_m q_1) \|$$

$$+ \frac{hR_m}{2} \| J \| \cdot \| e(L_m P_0 \| + \frac{hR_m}{2} \| J \| \cdot \| \epsilon(L_m q_{4j}) \|,$$

$$\leqq \frac{1}{2}(hL + R_m)\, 10^{-s} M + \frac{LhR_m}{4}\, 10^{-s} M + \frac{hR_m}{2} \| \epsilon(L_m q_{4j}) \| L.$$

Continuing in this way, we have

$$\| \mu_2 \| \leqq \frac{1}{3} \| z \| + \frac{hL}{8} 10^{-s} M + h \| z \| L,$$

$$\| \mu_1 \| \leqq \tfrac{3}{8} R_m 10^{-s} M,$$

$$\| W \| \leqq 6(10^{-s}) M + R_m 10^{-s} M,$$

$$\| v \| \leqq \left(1 + \frac{hL}{2} + \frac{h^2L^2}{6} + \frac{h^3L^3}{24}\right) \| \epsilon(y_j) \| + \frac{R_m}{3} \| \epsilon(L_m q_{4j}) \| + \tfrac{3}{8} 10^{-s} M + \| \mu_1 \| + \| \mu_2 \|,$$

$$\| \epsilon(y^*_{j+1}) \| \leqq \| \epsilon(y_j^*) \| + Lh \| v \| + \frac{R_m}{6} \| W \| + \frac{R_m}{3} \| e(L_m q_4) \|,$$

$$\| \epsilon(y^*_{j+1}) \| \leqq \| \epsilon(y_j^*) \| + \left(hL + \frac{h^2L^2}{2} + \frac{h^3L^3}{6} + \frac{h^4L^4}{24}\right) \| \epsilon(y_j) \| + hR_m \left(\frac{L}{3} + \frac{hL^2}{6} + \frac{h^2L^3}{2}\right) \| \epsilon(L_m q_{4j}) \|,$$

(3.13)

$$+ \left(\frac{4}{3} R_m + \frac{3}{8} h\right) 10^{-s} M + \left(\frac{R_m^2}{6} + \frac{13}{24} hR_m + \frac{7h^2}{24}\right) 10^{-s} M + \left(\frac{h^3}{2} + \frac{7h^2 R_m}{12}\right) 10^{-s} M + \left(\frac{h^3 R_m}{4}\right) 10^{-s} M.$$

Now, to estimate the rounding error incurred in *one* integration step, say from j to $j + 1$, we assume that all quantities obtained at the jth step are exact. Thus, in (3.13) we set $\epsilon(y_j^*) = 0$, $\epsilon(L_m q_{4j}) = 0$, and $\epsilon(y_j) = 0$. Denoting the one-step rounding error by $\epsilon_1(y^*_{j+1})$, we have

$$\| \epsilon_1(y^*_{j+1}) \| \leqq \left(\frac{4}{3} R_m + \frac{3}{8} h\right) 10^{-s} M + \left(\frac{R_m^2}{6} + \frac{13}{24} hR_m + \frac{7h^2}{24}\right) 10^{-s} M + \left(\frac{h^3}{2} + \frac{7}{12} h^2 R_m\right) 10^{-s} M + \frac{h^3 R_m}{4} 10^{-s} M.$$

(3.14)

Since $h = 10^{-m}$, we can take $R_m = h$ and get

$$\| \epsilon_1(y^*_{j+1}) \| \leqq \left[\frac{41}{24} 10^{-s-m} + 10^{-s-2m} + \frac{13}{12} 10^{-s-3m} + \frac{1}{4} 10^{-3-4m}\right] M, \tag{3.15}$$

which proves the theorem.

COROLLARY. *A bound for the accumulated rounding error, under the hypotheses of Theorem 2, is given by*

$$\| \epsilon(y_j^*) \| \leqq \| \epsilon(y_0^*) \| e^{hjL} + (1 - e^{hjL}) \left(\frac{f(h)}{1 - e^{hL}}\right) 10^{-s} M, \tag{3.16}$$

where $f(h) = \dfrac{h}{48} \left(130 + 160h + 100h^2 + 27h^3 + \dfrac{h^4}{3}\right)$.

Proof. From the definition of y_j^* we obtain

$$\| \epsilon(y_j) \| \leqq \| \epsilon(y_j^*) \| + \frac{R_m}{3} \| \epsilon(L_m q_{4j}) \| .$$

From (3.11), we have

$$R_m \| \epsilon(L_m q_{4j}) \| \leqq \frac{3}{2} (10^{-s})M + \frac{R_m}{2} 10^{-s} M,$$

Using (3.13), we get

$$\| \epsilon(y_{j+1}^*) \| \leqq e^{hL} \| \epsilon(y_j^*) \| + (f_1(h) + f_2(R_m , h))10^{-s} M, \tag{3.17}$$

where

$$f_1(h) = \frac{h}{48} (66 + 110h + 64h^2 + h^3),$$

$$f_2(h) = \frac{R_m}{48} \left(64 + 8R_m + 42h + 36h^2 + 26h^3 + \frac{h^4}{3}\right).$$

Setting $R_m = h$, and solving the difference equation corresponding to (3.17), we obtain (3.16).

Remark 6. Theorem 2 gives an upper bound on the one-step rounding error. A somewhat better result can be obtained from a statistical estimate of this error, if one is willing to make certain assumptions. If we assume (1) that the components of the errors, $e(r_i)$ and $e(LP_i)$ are independent and have a uniform distribution between $-10^{-s}/2$ and $10^{-s}/2$, and (2) that the bias which would be introduced in $e(Lq_1)$ and $e(Lq_2)$ by the coefficient, $\frac{1}{2}$, is eliminated by 'rounding up' Lq_1 and 'rounding down' Lq_2, then a direct computation with (3.12) yields as the approximate standard deviation of a component of $\epsilon(y_j^*)$,

$$\sigma_i \doteq \frac{1}{6} \left[\frac{10}{3} \left(R_m^2 + \frac{h^2}{4} \sum_k (\partial f_i/\partial y_k)^2\right)\right]^{\frac{1}{2}} 10^{-s}. \tag{3.18}$$

This is approximately the standard deviation of an error which is uniformly distributed between $\pm R_m 10^{-s}/2$, so that the accuracy is the same as would be obtained with $s + m$ digits.

Remark 7. As an example, we follow Gill [1] and integrate $y' = y$ from $t = 0$ to $t = 1$, with $h = 0.1$ and $s = 6$. The results are given in Table 1. The values in parentheses are those obtained by Gill's method [1]. For $t = 1$, after ten steps, we should have the value of $e/10$. If we use $y - q/3$ for this value, we obtain 0.27182810, which is in error by -8×10^{-8}. (Note that in computing q_1 the result of multiplying by $\frac{1}{2}$ was rounded up, while in the computation of q_2, it was rounded down.)

Remark 8. It is of interest to compare the accumulated error of the above example with the bounds given by (3.16) of the corollary and by statistical estimates. Since $\epsilon(y_0^*) = 0$ in the example, and $t = hj = 1$, (3.16) becomes

$$| \epsilon(y_j^*) | \leqq (e - 1) \frac{f(h)}{(e^h - 1)} \times 10^6 \leqq 1.72 \frac{f(h) \times 10^{-6}}{(e^h - 1)} .$$

TABLE 1

Comparison of Gill's Method with the Modified Runge-Kutta Method

t	*Stage*	r	z	lq	LP
0.0	0	5 000	100 000	0	100 000
	1	250	105 000	100 000	105 000
	2	5 275	105 250	16 667	52 750
	3		110 525	− 36 083	216 025
0.1	4	− 8	(110 517) 110 517	(− 3) − 3	110 517
	1	5 526	116 043	110 518	116 043
	2	276	116 319	18 422	58 297
	3	5 830	122 149	− 39 878	238 744
0.2	4	− 9	(122 140) 122 140	(− 8) − 9	122 140
	1	6 108	128 248	122 161	128 248
	2	304	128 552	20 364	64 428
	3	6 443	134 995	− 44 066	263 851
0.3	4	− 9	(134 986) 134 986	(+ 4) + 3	134 986
	1	6 749	141 735	134 980	141 735
	2	338	142 073	22 494	71 206
	3	7 121	149 194	− 48 716	291 606
0.4	4	− 12	(149 182) 149 182	(− 14) − 15	149 182
	1	7 461	156 643	149 224	156 643
	2	371	157 014	24 870	78 693
	3	7 869	164 883	− 53 820	322 269
0.5	4	− 11	(164 872) 164 872	(− 4) − 3	164 872
	1	8 244	173 116	164 881	173 116
	2	412	173 528	27 478	86 970
	3	8 697	182 225	− 59 492	356 165
0.6	4	− 13	(182 212) 182 212	(+ 3) + 3	182 212
	1	9 110	191 322	182 197	191 322
	2	456	191 778	30 369	96 117
	3	9 612	201 390	− 65 751	393 624
0.7	4	− 15	(201 375) 201 375	(− 9) − 9	201 375
	1	10 070	211 445	201 403	211 445
	2	502	211 947	33 568	106 225
	3	10 623	222 570	− 72 662	435 020
0.8	4	− 16	222 554 222 554	(− 3) − 3	222 554
	1	11 128	233 682	222 560	233 682
	2	556	234 238	37 094	117 397
	3	11 740	245 978	− 80 306	480 772
0.9	4	− 18	(245 960) 245 960	(− 9) − 9	480 772
	1	12 299	258 259	245 981	258 259
	2	614	258 873	040 995	129 744
	3	12 974	271 847	− 88 745	531 335
1.0	4	− 19	(271 828) 271 828	(− 4) − 3	

Now,

$$g(h) = \frac{f(h)}{(e^h - 1)} \doteq \frac{130 + 160h + 100h^2 + 27h^3}{48(1 + h/2 + h^2/6 + h^3/24)},$$

and $g(.1) \doteq 2.92$. Hence,

$$| \epsilon(y_j^*) | \leqq 4.98 \times 10^{-6}.$$

To obtain a statistical estimate, we might assume that accumulated error is the sum of the one-step errors and that these errors are independent. Using (3.18), the standard deviation for one step is about 3.4×10^{-8}. The standard deviation after ten steps is $\sqrt{10}$ times this, or about 1.1×10^{-7}.

It is of interest to compare the above results with those obtained from the classical Runge-Kutta method, (1.5)–(1.9). These are tabulated below.

t	y
0.0	.100 000
0.1	110 517
0.2	122 140
0.3	134 986
0.4	149 183
0.5	164 873
0.6	182 213
0.7	201 377
0.8	222 556
0.9	245 963
1.0	271 831

4. Floating-point Arithmetic. Since many modern automatic computers provide "floating-point" operations, and since the finite-precision modification must be applied in a slightly different way when floating-point numbers are used, it seems worthwhile to devote a short section to this subject.

Let us begin by establishing certain conventions. A "digital number in normal floating-point form" consists of two parts, a modulus and an exponent. The modulus is an aggregate of s decimal digits, the decimal point being placed at the extreme left and the digit in position one being non-zero. The exponent consists of two digits and represents the power of ten which multiplies the modulus. An algebraic sign is associated with each modulus and exponent. Thus, the fixed-point number, .00113, would be written as +.113–02 in normal floating form, and −11.3 would be written as −.113 + 02. In floating-point arithmetic some of the shift operations of formulas (3.1)–(3.5) will be carried out automatically by the positioning which must take place in the process of addition or subtraction. The rounding error will then be governed by the magnitude of h and the relative magnitudes of y and y'. If $hy_j' < y_j$, then Theorem 2 will apply to the procedure (4.1)–(4.5) below, it being understood that all errors must be considered as relative errors. If $hy_j' > y_j$ for some j, the theorem no longer holds. Nevertheless, over an interval, there should be a preponderance of points for which $hy_j' < y_j$, so that (4.1)–(4.5) should reduce the overall rounding error.

To explain the meaning of the symbols L and R in (4.1)–(4.5), we must first

point out that in automatic computers, the exponent of a floating-point number is placed to the left of the modulus. Thus, a shift right m places will not affect the exponent if $m < s$. Now, in the computation of r_i, the exponent, μ, of the quantity in square brackets is compared with the exponent, ρ, of z_{i-1}. If $\mu < \rho$, then $R^{(i)} = R_{\rho-\mu}$ and $L^{(i)} = L_{\rho-\mu}$. If $\mu \geqq \rho$, then $R^{(i)} = R_0$ and $L^{(i)} = L_0$. With this interpretation of the shift operations, the finite-precision modification for floating-point arithmetic is as follows:

$$\begin{cases} z_0 = y_j, \\ q_0 = q_{4j}, \\ P_0 = hf(z_0), \end{cases} \tag{4.1}$$

$$\begin{cases} \begin{cases} r_1 = L^{(1)}R^{(1)}[\frac{1}{2}P_0 - q_0], \\ z_1 = z_0 + r_1, \\ q_1 = 3r_1 - (\frac{1}{2}P_0 - q_0), \end{cases} \\ P_1 = hf(z_1), \end{cases} \tag{4.2}$$

$$\begin{cases} \begin{cases} r_2 = L^{(2)}R^{(2)}[\frac{1}{2}(P_1 - q_1)], \\ z_2 = z_1 + r_2, \\ q_2 = -r_2 - \frac{1}{3}q_1 + \frac{1}{2}P_1, \end{cases} \\ P_2 = hf(z_2) - \frac{1}{2}P_1, \end{cases} \tag{4.3}$$

$$\begin{cases} \begin{cases} r_3 = L^{(3)}R^{(3)}[P_2], \\ z_3 = z_2 + r_3, \\ q_3 = -r_3 + q_2, \end{cases} \\ P_3 = hf(z_3) + 2P_2, \end{cases} \tag{4.4}$$

$$\begin{cases} r_4 = L^{(4)}R^{(4)}[\frac{1}{6}P_3 + q_3], \\ y_{j+1} = z_4 = z_3 + r_4, \\ q_{4,j+1} = 3[r_4 - (\frac{1}{6}P_3 + q_3)]. \end{cases} \tag{4.5}$$

Computation and Data Reduction Center
Space Technology Laboratories, Inc.
Los Angeles 45, California

1. S. GILL, "A process for the step-by-step integration of differential equations in an automatic digital computing machine," *Proc. Cambridge Philos. Soc.*, v. 47, pt. 1, p. 96–108.
2. J. VON NEUMANN, & H. H. GOLDSTINE, "Numerical inverting of high order matrices," *Bull. Amer. Math. Soc.*, v. 53, n. 11, November 1947.