Quadrature Formulas with the Error Estimating Ability

MASATSUGU TANAKA*

1. *Preface*

In this paper, we study some of those formulas of a Runge-Kutta type which have the error estimating ability and which are to solve the ordinary differential equations;

$$y'=f(x), \qquad y(x_0)=y_0 \tag{1}$$

Their application to numerical integration is also investigated. Here we assume it possible for $f(x)$ to be differentiated to the required order.

The general expression of the formulas with the ability of error estimation is

$$k_i=hf(x_n+\alpha_i h) \qquad (n=0, 1, 2,\cdots,\ i=1, 2,\cdots,\ q) \tag{2}$$

$$y_{n+1}=y_n+\sum_{i=1}^{p}\nu_i k_i \tag{3}$$

$$y'_{n+1}=y_n'+\sum_{i=1}^{q}\mu_i k_i \qquad (p\leqq q,\ y_0'=y_0) \tag{4}$$

$$T=y_{n+1}-y'_{n+1} \tag{5}$$

where α_i, ν_i and μ_i are constants, y_{n+1} is a formula to obtain the numerical solution, and y'_{n+1} is a formula with one order or more higher rate of accuracy than that of y_{n+1}, T, the difference between these two formulas gives us an estimated value of the accumulated truncation error.

In 2, we investigate what rate of accuracy we can give to the quadrature formula, and then show the most appropriate formula in a sense concerning to each of the cases where $\alpha_i\neq 0$, $q=3$, 4 and 5.

In 3, as an application of the formulas in 2 to the numerical integration, we devise automatic integrators of the adaptive type. In 4, giving some numerical examples, we show the usefulness of automatic integrators by the author.

The formulas presented in this paper are more interesting when used as quadrature formulas than as numerical methods of ordinary differential equations.

Though we have automatic integrators of various kinds (See [1]), those by the author seem to have such advantages as follows:

With them,

(1) The algorithm is simple.

This paper first appeared in Japanese in Joho Shori (the Journal of the Information Processing Society of Japan), Vol. 12, No. 3 (1971), pp. 135-144.

* Faculty of Engineering, Yamanashi University.

(2) The changes of step sizes can be made completely at will, and so the application to complicated situations is possible.

(3) We can get the high rate of of accuracy in error estimations.

(4) We can get good efficiency when $f(x)$ is complicated. Especially this characteristic is vivid when the integration is done in the neighborhood of a singular point.

2. *The formulas and their existence theorem*

Theorem 1. Concerning to the Runge-Kutta formula (4) with q functional computations per step, if we name the highest order attainable

$$r,\ q \leqq r. \tag{6}$$

The proof is easily made. Actually the formula of $2q$th order is attained, too.

Theorem 2. Concerning to the general formula (2) to (5) with q functional computations per step, the following can be said.

(1) We can give the error-estimating ability to the $(q-1)$th order Runge-Kutta method (3).

(2) We can not give it the qth order one (3).

In the above cases to make the error estimation possible, we need to have the difference of at least one order between y_{n+1} and y'_{n+1}. The proof is easily made.

As is clear from theorem 2, we can not give to y_{n+1} the accuracy of the rate higher than $(q-1)$th order. Accordingly, E, the truncation error of y_{n+1} is:

$$E=h^q K_1 f^{(q-1)}(x_n)+h^{q+1}K_2 f^{(q)}(x_n)+\cdots \tag{7}$$

while y'_{n+1} being the method of Sth order, its truncation error E' is given as:

$$E'=h^{s+1}K_1' f^{(s)}(x_n)+h^{s+2}K_2' f^{(s+1)}(x_n)+\cdots \tag{8}$$

where

$$K_1=\{\sum_{i=1}^{p}\nu_i\alpha_i{}^{(q-1)}-1/q\}/(q-1)! \tag{9}$$

$$K_2=\{\sum_{i=1}^{p}\nu_i\alpha_i{}^{q}-1/(q+1)\}/q! \tag{10}$$

$$K_1'=\{\sum_{i=1}^{q}\mu_i\alpha_i{}^{s}-1/(s+1)\}/s! \tag{11}$$

$$K_2'=\{\sum_{i=1}^{q}\mu_i\alpha_i{}^{s+1}-1/(s+2)\}/(s+1)! \tag{12}$$

With these K_i, $K_i'(i=1, 2)$, we measure the accuracy of truncation error in y_{n+1} and y'_{n+1}.

In the general expression (2) to (5) we think of the cases where $q=3$, 4 and 5. y'_{n+1} in (4) is the formula obtained by the application of Gauss-Legendre quadrature formula, which uses q functional values, to

$$\int_{x_n}^{x_{n+1}} f(x)dx \tag{13}$$

In this case, y'_{n+1} becomes the formula with the highest rate of accuracy of all those of Runge-Kutta type which need the functional computations of the same times per step as those of ours. Generally, in the case where $q=k$ the formula is $B-(k-2)$ shown in Table 1. $\alpha_i(i=1, 2, \cdots, k)$ in the formula (2) to (5) are

Table 1. The formulas with error estimating ability.

formula	α_1	q	r	s	Coefficient	K_1	K_2	K_1'	K_2'
B-1		3	2	6	$\alpha_1=0.8872983346$, $\alpha_2=0.1127016654$, $\alpha_3=0.5$, $\nu_1=\nu_2=0.5$, $\mu_1=\mu_2=5/18$, $\mu_3=4/9$	3.33×10^{-2}	1.67×10^{-2}	-4.96×10^{-7}	-2.48×10^{-7}
B-2	$\neq0$	4	3	8	$\alpha_1=0.06943184420$, $\alpha_2=0.3300094782$, $\alpha_3=0.9305681558$, $\alpha_4=0.6699905218$, $\nu_1=0.04519229241$, $\nu_2=0.6521451549$, $\nu_3=0.3026625527$, $\mu_1=0.1739274226$, $\mu_2=0.3260725774$, $\mu_3=0.1739274226$, $\mu_4=0.3260725774$	2.89×10^{-3}	1.45×10^{-5}	-7.09×10^{-9}	-3.54×10^{-8}
B-3		5	4	10	$\alpha_1=0.04691007703$, $\alpha_2=0.2307653449$, $\alpha_3=0.7692346551$, $\alpha_4=0.9530899230$, $\nu_1=\nu_4=0.04083499337$, $\nu_2=\nu_3=0.4591650066$, $\mu_1=\mu_4=0.1184634425$, $\mu_2=\mu_3=0.2393143352$, $\mu_5=0.2844444444$	-1.76×10^{-4}	-8.82×10^{-5}	-3.17×10^{-8}	-2.16×10^{-8}

q: number of functional evaluation per step r: order of y_n s: order of y_n'

the zero points of Legendre polynomial $P_k(x)$ of the k'th degree. (See [2]) As $(\alpha_1, \alpha_2, \cdots, \alpha_{k-1})$, we select, out of the possible ${}_kC_{k-1}$ pairs, the one that makes y_{n+1} a $(k-1)$th order method with the highest rate of accuracy attainable.

3. *The application to automatic integrators*

Leaving the definition of the automatic integrator to P. J. Davis and others, we go on with our adaptive and non-iterative methods [1].

If we set $y_0=y_0'=0$, the formula (2)—(5) becomes, as it is, a quadrature formula with the ability of error estimation. Then y_n and y_n' denote the approximate solutions of the definite integral

$$\int_{x_0}^{x_n} f(x)dx \tag{14}$$

In the formula of Runge-Kutta type which is a one step method, we can adjust pitches quite freely. Accordingly, if we make use of this advantage and integrate the interval from the left hand side to the right controlling pitches in accordance with the accuracy, it will be more effective when $f(x)$ changes suddenly than the methods that divide the interval into equal pieces. Then we study the algorithm to adjust the pitches in the case where the formulas in 2 are used.

When the definite integral

$$\int_a^b f(x)dx \tag{15}$$

is to be found, we set the allowable error for the whole interval of the numerical

integration as ε, and the order of the integral formula y_{n+1} as r.

When the integration up to the nth step has been found within the bounds of the allowable error, the question is how to decide the pitch at the $(n+1)$th step. If the first pitch at the $(n+1)$th step is set as h°_{n+1}, the local truncation error t°_{n+1} at this step is

$$t^\circ_{n+1}=T^\circ_{n+1}-T_n \tag{16}$$

where T°_{n+1} denotes the difference between y_{n+1} and y'_{n+1} when the pitch is h°_{n+1}, and T_n denotes the difference between them in the case one step earlier.
If

$$t^\circ_{n+1}>\frac{h^\circ_{n+1}\varepsilon}{b-a} \tag{17}$$

the pitch is excessively large, and so a new pitch h_{n+1} has to be found out. When t_{n+1} denotes the local truncation error for the new pitch h_{n+1},

$$t_{n+1}\fallingdotseq(h_{n+1})^{r+1}C \tag{18}$$

where C is constant. Similarly,

$$t^\circ_{n+1}\fallingdotseq(h^\circ_{n+1})^{r+1}C \tag{19}$$

Therefore

$$C\fallingdotseq\frac{t^\circ_{n+1}}{(h^\circ_{n+1})^{r+1}} \tag{20}$$

If (20) is substituted for (3.5), we obtain

$$t_{n+1}\fallingdotseq\left(\frac{h_{n+1}}{h^\circ_{n+1}}\right)^{r+1}t^\circ_{n+1} \tag{21}$$

On the other hand, we must satisfy the condition

$$t_{n+1}\leq\frac{h_{n+1}\varepsilon}{b-a} \tag{22}$$

Therefore

$$h_{n+1}\leq\sqrt[r]{\frac{(h^\circ_{n+1})^{r+1}\varepsilon}{(b-a)t^\circ_{n+1}}} \tag{23}$$

That is, we have only to give the new pitch h_{n+1} the value that satisfies

$$h_{n+1}=\alpha\sqrt[r]{\frac{(h^\circ_{n+1})^{r+1}\varepsilon}{(b-a)t^\circ_{n+1}}} \tag{24}$$

where α being a constant, satisfies

$$0<\alpha\leq1 \tag{25}$$

Next, if (22) can be satisfied, we advance to the $(n+2)$th step. In this case, the procedure to decide the pitch is just the same as that in the preceding case. If we assume

$$t^\circ_{n+2}\fallingdotseq(h^\circ_{n+2})^{r+1}C \tag{26}$$

the new pitch h°_{n+2} is

$$h^\circ_{n+2}=\alpha\sqrt[r]{\frac{(h_{n+1})^{r+1}\varepsilon}{(b-a)t_{n+1}}} \tag{27}$$

where α is the constant above stated. Henceforth we will call $1/\alpha$ the factor of

safety. (The f. s. as an abbreviation) The f. s. ought to be decided in accordance with the circumstance taking each kind of cause into consideration. We explain here the relation between several causes and the f. s.

(1) When pitch h is large, it is rather questionable that we express the terms of truncation error only by the principal term. Then the f. s. has to be large enough. The converse is also true.

(2) When the accuracy of the solution is in question, the f. s. has to be large enough. When high efficiency is desired rather than accuracy, the f. s. has only to be the value around 1.

(3) Around any sigular point of $f(x)$, it is better to set the f. s. large. According to our experiences, α is enough when 0.9 or 1.0 so long as $f(x)$ has not particularly bad sort of character.

Through the above study, we have y_n as the solution of

$$\int_{x_0}^{x_n} f(x)dx \tag{28}$$

and in order to find its error estimate we have y_n' the formula with a higher rate of accuracy. In this way, we can estimate the error, but if we use, as the solution, y_n' which has too higher a rate of accuracy, it won't be efficient. In the case of automatic integrators the high rate of efficiency is more desirable than that of accuracy if only its necessary accuracy is achieved. From such a view point, we set the allowable error in the small interval with the pitch h_{n+1} as

$$\frac{\beta h_{n+1}\varepsilon}{b-a} \tag{29}$$

where β is a constant and $\beta \geq 1$. In the numerical examples in the succeeding section, we show the cases where $\beta=1$, 100 and 200.

4. *Numerical examples and conclusion*

To compare the automatic integrators by the author with ordinary ones, we take into consideration the following three integrations which have the allowable error 10^{-4}.

$$(1)\quad \int_0^1 e^x dx \tag{30}$$

$$(2)\quad \int_0^1 \frac{dx}{1+x} \tag{31}$$

$$(3)\quad \int_0^{0.99} \frac{dx}{1-x} \tag{32}$$

The ordinary automatic integrators we have used are:

(a) The ALGOL routine by M. Shibuya which uses Simpson's rule and, by halving the pitch gets to the necessary rate of accuracy [4].

(b) The translation of Romberg integration in FORTRAN made by R. H.

Pennington, which appears on p. 250 in [5], into ALGOL routine, where the relative error estimation is replaced by the absolute one.

(c) The translation of FORTRAN routine made by D. L. Russel which, using the mid-point rule, is adaptive and non-iterative, into ALGOL routine [1].

(d) The ALGOL routine by W. M. Mckeeman and L. Tesler which uses three point rule and seven point rule by Simpson and is also adaptive and iterative [6], [3].

Concerning to (d), we have the study by J. N. Lyness, and we follow his

Table 2. The numerical solution of $\int_0^1 e^x dx$ (true solution 1.718281828)

method		α	β	y_n	y_n'	error estimate of y_n	true error of y_n	true error of y_n'	number of functional evalution
author's automatic integrator	B-1	0.9	1.0	1.718365417	1.718281809	8.3608×10^{-5}	8.3589×10^{-5}	-1.9×10^{-8}	78
		0.9	100.0	1.724783492	1.718281820	6.501672×10^{-3}	6.501664×10^{-9}	-8.0×10^{-9}	18
	B-2	0.9	1.0	1.718358954	1.718281820	7.7134×10^{-5}	7.7126×10^{-5}	-8.0×10^{-9}	20
		0.9	100.0	1.722301652	1.718281822	4.01983×10^{-3}	4.019824×10^{-5}	-6.0×10^{-9}	8
		0.9	200.0	1.722301652	1.718281822	4.01983×10^{-3}	4.019824×10^{-5}	-6.0×10^{-9}	8
	B-3	0.9	1.0	1.718258034	1.718281818	-2.3784×10^{-5}	-2.3794×10^{-5}	-1.0×10^{-8}	15
		0.9	100.0	1.718049192	1.718281821	-2.32629×10^{-4}	-2.32636×10^{-4}	-7.0×10^{-9}	10
		0.9	200.0	1.718049192	1.718281821	-2.32629×10^{-4}	-2.32636×10^{-4}	-7.0×10^{-9}	10
ordinary automatic integrator	Simpson				1.718281792			-3.6×10^{-8}	17
	Romberg				1.718281839			1.1×10^{-8}	9
	Midpoint				1.718212257 (10 division)			-6.9571×10^{-6}	34
					1.718237070 (20 division)			-4.4758×10^{-6}	60
					1.718221915 (30 division)			-5.9913×10^{-5}	44
	ASQ				1.718281916			8.8×10^{-8}	19

Table 3. The numerical solution of $\int_0^1 \frac{dx}{1+x}$ (true solution 0.6931471806)

method		α	β	y_n	y_n'	error estimate of y_n	true error of y_n	true error of y_n'	number of functional evaluation
author's automatic integrator	B-1	1.0	1.0	0.6932348564	0.6931471801	8.76763×10^{-5}	8.76758×10^{-5}	-5.0×10^{-10}	45
		1.0	100.0	0.6974634834	0.6931469402	4.316543×10^{-5}	4.3163028×10^{-3}	-2.404×10^{-7}	12
	B-2	1.0	1.0	0.6931002361	0.6931471797	-4.69436×10^{-5}	-4.69445×10^{-5}	-9.0×10^{-10}	20
		1.0	100.0	0.6910161808	0.6931469724	-2.1307916×10^{-3}	-2.1309998×10^{-3}	-2.082×10^{-7}	12
		1.0	200.0	0.6902996006	0.6931467799	-2.8471793×10^{-3}	-2.8471793×10^{-3}	-4.007×10^{-7}	8
	B-3	1.0	1.0	0.6931207423	0.6931471800	-2.64377×10^{-5}	-2.64383×10^{-5}	-6.0×10^{-10}	20
		1.0	100.0	0.6927032092	0.6931471698	-4.439606×10^{-4}	-4.439714×10^{-4}	-2.68×10^{-8}	10
		1.0	200.0	0.6927032092	0.6931471698	-4.439606×10^{-4}	-4.439714×10^{-4}	-2.68×10^{-8}	10
ordinary automatic integrator	Simpson				0.6931476552			4.716×10^{-7}	17
	Romberg				0.6931471812			6.0×10^{-10}	17
	Midpoint				0.6930927017 (10 division)			-5.44789×10^{-5}	30
					0.6931004666 (20 division)			-4.6714×10^{-6}	32
					0.6931124655 (30 division)			-3.47251×10^{-5}	30
	ASQ				0.6931474753			2.947×10^{-7}	19

Table 4. The numerical solution of $\int_0^{0.99} \frac{dx}{1-x}$ (true solution 4.6051701860)

method		α	β	y_n	y_n'	error estimate of y_n	true error of y_n	true error of y_n'	number of functional evalution
author's automatic integrator	B-1	1.0	1.0	4.605270338	4.605170443	9.9895×10^{-5}	1.00152×10^{-4}	2.57×10^{-7}	2,583
		0.8	100.0	4.610930730	4.605170125	5.760605×10^{-3}	5.760544×10^{-3}	-6.1×10^{-8}	288
	B-2	0.8	1.0	4.605249593	4.605170115	7.9478×10^{-5}	7.9407×10^{-5}	-7.1×10^{-8}	283
		0.7	100.0	4.606963975	4.605170091	1.793884×10^{-3}	1.793789×10^{-3}	-9.5×10^{-8}	124
		0.7	200.0	4.607343052	4.605170100	2.172952×10^{-3}	2.172866×10^{-3}	-8.6×10^{-8}	124
	B-3	0.7	1.0	4.605112857	4.605170119	-5.7262×10^{-5}	-5.7329×10^{-5}	-6.7×10^{-8}	145
		0.5	100.0	4.600307008	4.605168882	-4.861874×10^{-3}	-4.863178×10^{-3}	-1.305×10^{-6}	75
		0.5	200.0	4.594715129	4.605161797	-1.0446668×10^{-2}	-1.0455057×10^{-2}	-8.389×10^{-6}	95
ordinary automatic integrator	Simpson				4.605170159			-2.7×10^{-8}	2,049
	Romberg				4.605168163			-2.023×10^{-6}	1,025
	Midpoint				4.603898194 (10 division)			-1.271992×10^{-3}	205
					4.604906008 (20 division)			-2.64178×10^{-4}	284
					4.605037487 (30 division)			-1.32699×10^{-4}	350
	ASQ				4.605171919			1.733×10^{-6}	186

advise to decide the allowable error in the sub-intervals [3].

In Table 2, 3 and 4, the results of the computations of examples are given, where Simpson, Romberg, Midpoint and ASQ denote automatic integrators (a), (b), (c) and (d) respectively. As the methods by the author, we take up three automatic integrators, each of which uses B-1, B-2 and B-3 respectively.

In the case of automatic integrator (c), the efficiency varies widely according to the decision of primary pitches. We seek for the solution setting the primary pitch as 1/10, 1/20 and 1/30 of the whole interval.

From Table 1—Table 4, we can get the following conclusion.

(1) When B-1—B-3 are used as quadrature formulas, the formulas of higher orders seem to have higher efficienty.

(2) In the case of the integration around a singular point, such as (32), the automatic integrator (a) and (b) are inefficient. As the methods proposed by the author are thorough especially about adaptability, the one that uses a formula with such a high order as B-3 is greatly efficient for these problems.

(3) In such cases as (30) and (31) where $f(x)$ doesn't make any sudden changes, and the interval of the integration is not near the sigular point, such interative method as (a), (b) which divides the interval into equal minute pieces has, among the ordinary ones, greater efficiency, while the methods by the author are very effective as well. Generally, for the integration under such circumstances the f. s. need not be so large. In the above cases, for example, the inverse of the f. s. will be appropriate when in the range of 0.8~1.0.

On the other hand, for the integration near a singular point such as (32), it will be more efficient that we take the bigger f. s. in the auther's non-iterative

automatic integrators, because we can avoid unnecessary repetitions. For example, the inverse of the f. s. in this case will be appropriate when 0.5—0.8. In the author's routine, the trial to enlarge the f. s. will have to be made to increase the efficiency when meaningless repetition goes on.

(4) In Formulas B-1—B-3, y'_{n+1} has a much higher rate of accuracy than y_{n+1}. Therefore, it has much smaller error than the allowable one even when $\beta=200$.

In conclusion, we recommend the formula B-3 as the quadrature formula. When very hard conditions are given, we had better let $\beta=1.0$ and find the numerical solution y_n together with the error y_n-y_n'. We think the automatic integrator that uses the formula B-3 has the higher efficiency than any other ordinary one when $f(x)$ is complicated.

Acknowledgement

The auther expresses here his sincere gratitude to Professor Shigeichi Moriguchi, University of Tokyo, for his helpful suggestions.

References

[1] Davis, P. J. and P. Rabinowitz: Numerical Integration, Blaisdell Publishing Company, Tronto. London (1967), 159-174.
[2] Singer, J.: Element of Numerical Analysis, *Academic Press*, New York. London (1964).
[3] Lyness, J. N.: Notes on the Adaptive Simpson Quadrature Routine, *J. ACM, 16*, No. 3 (1969).
[4] Yamauchi, Z., S. Moriguchi and S. Hitotumatu: *Computational Methods for Digital Computers, 1*, Baifukan, Tokyo (1965), 76-77 (in Japanese).
[5] Pennington, R. H.: Introductery Computer Methods and Numerical Analysis, 2nd edition, The Macmillan Company, Collier-Macmillan Limited, London (1970).
[6] Mckeeman, W. M. and L. Tesler: Algorithm 182, nonrecursive adaptive integration, *comm. ACM, 6*, No. 6, p. 135.