An extended doubly-adaptive quadrature method based on the combination of the Ninomiya and the FLR schemes

An extended doubly-adaptive quadrature method based on the combination of the Ninomiya and the FLR schemes

Takemitsu Hasegawa, Susumu Hibino, Yohsuke Hosoda and Ichizo Ninomiya
Department of Information Science, University of Fukui, Fukui, 910-8507, Japan

November 15, 2006 / March 16, 2007

Abstract. An improvement is made to an automatic quadrature due to Ninomiya (1980) of adaptive type based on the Newton-Cotes rule by incorporating a doubly-adaptive algorithm due to Favati, Lotti and Romani (1991). We compare the present method in performance with some others by using various test problems including Kahaner's ones (1971).

Keywords: automatic quadrature, adaptive algorithm, numerical comparison

Mathematics Subject Classification: 65D30, 65D32

1. Introduction

Given a function $f(x)$ and an error tolerance ε, an automatic quadrature algorithm computes an approximation to the definite integral

$$I(f) = \int_a^b f(x)\,dx, \tag{1}$$

of the form

$$Q(f) = \sum_{k=0}^{N} w_k f(x_k), \tag{2}$$

where w_k are weights and x_k are sample points, satisfying the condition

$$|I(f) - Q(f)| \le \varepsilon. \tag{3}$$

Generally, the algorithm generates the sequence of approximations with their error estimates until an estimated error is not greater than ε. The algorithm is classified into order adaptive and partition adaptive (Venter and Laurie, 2002).

A partition adaptive method splits the integration interval into subintervals to which a fixed integration rule is applied to approximate each integral until a local error estimate is not greater than the partitioned tolerance on the subinterval. Ninomiya's routine AQNN9 (1980), (Davis and Rabinowitz, 1984, p.341) based on the Newton-Cotes (abbreviated

to N-C) rules and the routine QAG (QAGS) in QUADPACK (Piessens, de Doncker-Kapenga, Überhuber and Kahaner) are partition adaptive. The partition adaptive method copes with non-regular integrands. See also Plaskota and Wasilkowski (2005).

An order adaptive method is suitable for smooth functions. This method consists of the sequence of integration rules with their order increasing by distributing and increasing sample points uniformly in the integration interval according to the corresponding integration rules.

A doubly-adaptive method (Favati, Lotti, Di Marco and Romani, 1994; Oliver, 1971; Oliver, 1972) is both order and partition adaptive (Venter and Laurie). This scheme chooses either to apply the order adaptive method to the current subinterval or to further split the subinterval, by detecting the local regularity of the integrand. An improved version QXG (QXGS) due to Favati, Lotti and Romani (1991) (we call FLR henceforth) of QAG (QAGS) is a doubly-adaptive algorithm based on recursive monotone stable (RMS) formulas (1991; 1994). Ninomiya's method is less effective for oscillatory integrands than the FLR method.

The purpose of this paper is to construct an improved automatic quadrature by the proper combination of Ninomiya's method and the FLR one. By using Kahaner's twenty one problems (1971) we compare in performance the present method with Ninomiya's method, the FLR method and an adaptive Lobatto procedure with Kronrod extension due to Gander and Gautschi (2000). We also give the comparison results with Venter-Laurie's method (2002) for 32 test problems.

This paper is organized as follows. In section 2 we outline Ninomiya's algorithm. In section 3 the sequence of RMS formulas is reviewed. In section 4 we show how to incorporate the FLR method into the Ninomiya scheme to construct an improved scheme. In section 5 the present algorithm is outlined. Section 6 demonstrates the performance of the present routine by using various test problems.

2. Ninomiya's method

We outline Ninomiya's partition adaptive (9-point) N-C method. The N-C rules suit partition adaptive schemes, where computed function values are stored and reused after the subdivision of the interval. In his algorithm Ninomiya makes effective use of the stack and three improvements: refining the error estimation, relaxing the convergence criterion and detecting extraordinary points.

2.1. Refinement of error estimation

Let $S_{[a,b]}$ be an $n+1$-point N-C rule to an integral on an interval $[a,b]$. Let $c=(a+b)/2$. A way to get an estimate $|\tilde{e}_{n+1}|$ of the error e_{n+1} for $S_{[a,b]}$ might be $|S_{[a,b]}-(S_{[a,c]}+S_{[c,b]})|$ with new n function evaluations. Ninomiya observes that the error estimation requires only two additional sample points, which can be reused after the interval is bisected. In fact, let $n\geq 2$ be an even integer, then $e_{n+1}\propto h^{n+3}f^{(n+2)}$, where $h=(b-a)/2$. Noting that $f^{(n+2)}$ can be estimated by using $n+3$ function evaluations (the divided difference (Berntsen and Espelid, 1991)), we see that two additional function evaluations suffice since $n+1$ function values have been evaluated in constructing the $n+1$-point N-C rule $S_{[a,b]}$.

Ninomiya chooses two mid-points of outermost subintervals (squares in Fig. 1 for 5-point and 9-point rules (circles)). He actually gives $\tilde{e}_{n+1}$ and new rules $Q_{n+3}=S_{[a,b]}-\tilde{e}_{n+1}$ for $n=4,6$ and 8. Although the 9-point N-C rule is known to be numerically unstable, the 11-point rule $Q_{11}=\{\sum_{k=0}^{4}v_k(f_k+f_{10-k})+v_5f_5\}h/u$ is stable in that all weights hv_k/u are positive, $v_0=18447429$, $v_1=77594624$, $v_2=63216384$, $v_3=150355296$, $v_4=81233152$, $v_5=154791780$, $u=468242775=\sum_{k=0}^{4}v_k+v_5/2$. In the present method we use Q_{11}.

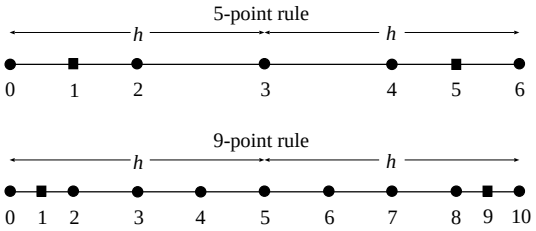

Figure 1. Arrangement of sample points (circles) for Newton-Cotes rules. Squares denote additional sample points used for error estimates.

2.2. Relaxation of convergence criterion

In a partition adaptive method a convergence test must be performed locally and separately for every subinterval. Recall that ε is the required accuracy (overall criterion) in (3) for the whole interval of the width $2h_0=(b-a)$. Let ε_i be the local error criterion for the i-th subinterval of the width $2h_i$. Then the proportional allocation $\varepsilon_i=\varepsilon h_i/h_0$ is safe but extremely conservative. In his paper (1980) Ninomiya proposed a rather cautious scheme, $\varepsilon_i=\varepsilon h_i/h_0\log(h_0/h_i)$ to positively relax the error criterion for small subintervals.

After the publication of his paper Ninomiya made further numerical experiments to get a rather ambitious but efficient criterion

$$\varepsilon_i = \varepsilon h_i/h_0\sqrt{h_0/h_i}, \tag{4}$$

with enhanced relaxation since $\sqrt{h_0/h_i} > \log(h_0/h_i)$.

2.3. Treatment of extraordinary points

Suppose that a partition adaptive method is applied to an integral having singular points in the integration interval. Then the process of bisection goes on extremely until the method might fail. Let m be a positive integer. If a singular point is at one of 2^m equally divided points in a given interval, then the singular point is at the end of some subinterval after the bisection process proceeds. From the asymptotic behavior of the error estimates of the approximations on the intervals having endpoint singularity, Ninomiya detects and treats analytically three types of singularities, discontinuity, algebraic singularity, and logarithmic singularity without any additional function evaluations.

Remark: In the FLR scheme an extrapolation method (ε-algorithm) is used to eliminate the effects of integrand singularities. Although Ninomiya's method is an improved N-C automatic quadrature by virtue of three devices above, his method might be less effective, say for oscillatory functions or when the tolerance ε is very small.

3. RMS sequence of quadrature rules

Recursive monotone stable (RMS) formulas due to FLR (1991) constitutes a nice family of quadrature rules with increasing precision, $I_1, \ldots, I_n$, which are symmetric interpolatory rules. This sequence of rules is successively constructed without wasting function values previously computed since the set of the sample points of I_{k-1} is a subset of that of I_k (embedded rules). In addition the family of rules has the following properties,

- the degree of precision of I_k is sufficiently larger than the one of I_{k-1},
- the distance between sample points is shorter near the ends of the interval,
- all the formulas in the family are numerically stable, namely all the weights are positive.

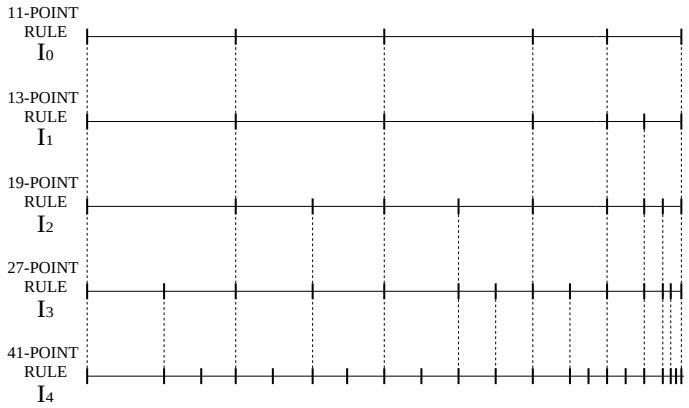

Figure 2. Arrangement of sample points in the right-half interval

The case KEY= 2 in the routine QXG in FLR (1991) uses a family of four formulas in the RMS formulas, namely, 13-point (I_1), 19-point (I_2), 27-point (I_3), 41-point (I_4) rules. Including Ninomiya's stable 11-point rule (I_0) yields a sequence of rules, $I_0, I_1, \ldots, I_4$, with increasing accuracy, see Fig. 2 for the arrangement of sample points in the right-half interval with the sample points symmetrically arranged in the left-half interval. From Fig 2 we find that the sample points for I_k are all reused in I_{k+1}, $0 \le k \le 3$. Further, they can be also reused after the bisection of the interval. See also Sugiura and Sakurai (1989).

4. Combination of Ninomiya's method and the FLR method

4.1. Selection criterion

Combining properly the Ninomiya and the FLR schemes requires a selection criterion to decide at each stage during the integration process whether the current interval is bisected (Ninomiya's scheme) or a higher order rule is applied (the FLR scheme). To this end, usually the ratio of the error estimates of two quadrature rules is used (Venter and Laurie).

Suppose that there exists a sequence of rules $I_0, I_1, \ldots, I_n$ with increasing accuracy, namely I_n is the most accurate rule. If an integrand has no singularity in the interval, we apply I_k, $k = 0, 1, \ldots$, consecutively until a satisfactory approximation is obtained. Let $E_k > 0$ be an error estimate for I_k. Let `hint` be defined by

$$\texttt{hint} = E_k/E_{k-1}, \tag{5}$$

then `hint`< 1. The value of `hint` indicates the smoothness of integrands. The smaller is `hint`, the less singular is the integrand in the interval. We choose to apply the higher order rule if `hint` is smaller than a fixed value P, namely

$$\texttt{hint} < P, \tag{6}$$

otherwise we choose to bisect the current interval. In this connection the determination of the parameter P plays an important role in the efficiency of an automatic quadrature method.

We note that in computing $Q_{11}(= I_0)$ and its error estimate $\tilde{e}_9(= E_0)$ in Ninomiya's scheme the error estimate $\tilde{e}_5(= E_{-1})$ for the 5-point N-C rule $Q_5(= I_{-1})$ is also available without any further computational costs. By using these $\tilde{e}_5$ and $\tilde{e}_9$ we determine `hint` by

$$\texttt{hint} = |\tilde{e}_9/\tilde{e}_5|. \tag{7}$$

A comprehensive numerical experiment on integrals of various types (particularly, twenty one problems of Kahaner, 1971 with varied values of tolerance ε) with changing P in (6) between 0.01 and 0.5, reveals that $P = 0.2$ might be a good choice so as to minimize the numbers of function evaluations required to approximate the integrals. In the FLR method (1991) below $P = 0.16$ is chosen.

In summary, if `hint` given by (7) is smaller than 0.2, we choose to apply the higher order rule (the 13-point rule I_1 of the RMS family), see § 4.2 below. Otherwise we choose to bisect the current interval.

4.2. Incorporation of the FLR method

Once the higher order rule is chosen in § 4.1 above we proceed to the FLR scheme, which is also a doubly-adaptive scheme. Again we begin by computing the approximations and the error estimates followed by performing the convergence test before the selection criterion test.

Recall that in the RMS family above I_k $(k = 1, \ldots, 4)$ denote the approximations to $I(f)$ (1), where the integration interval is assumed to be a current subinterval after bisections. Let M_k $(k = 1, \ldots, 4)$ denote the approximations to $I(|f - I_k/(b-a)|) \approx I(|f - I(f)/(b-a)|)$. By noting that the ratios of the numbers N_k of sample points for I_k, $k = 2, 3, 4$ are $N_k/N_{k-1} \approx 3/2$, the errors $|I(f) - I_k|$ is estimated by

$$E_k := M_k * \min(1, 200|I_k - I_{k-1}|/M_k)^{3/2}, \quad k = 2, 3, 4, \tag{8}$$

if $M_k \neq 0$, otherwise $E_k := |I_k - I_{k-1}|$. We use Ninomiya's 11-point rule I_0 to estimate the error $|I(f) - I_1|$ by $E_1 = |I_1 - I_0|$.

For each value of $k = 1, \ldots, 4$ we perform the following steps. We accept the approximation I_k of the current interval if the convergence criterion (of the FLR, not relaxed)

$$E_k \leq \varepsilon(h_i/h_0), \tag{9}$$

is satisfied. Otherwise, if $1 \leq k \leq 3$ and the value of `hint` for the selection criterion defined by

$$\texttt{hint} = E_k/E_{k-1}, \tag{10}$$

is smaller than 0.16, which is the value determined in the FLR method, the higher order rule I_{k+1} is applied to the current interval. Otherwise the current interval is bisected to apply Ninomiya's scheme. Note that in the original FLR method the selection criterion by `hint` (10) is examined only once, namely $\texttt{hint} = E_3/E_2$.

5. Present routine AQN9D

We outline the present automatic quadrature routine AQN9D based on the combination of Ninomiya's method and the FLR scheme.

Purpose

Given the integrand $f(x)$, the interval $[a, b]$, the tolerance ε and the machine epsilon EPMACH, the routine AQN9D computes an approximation S which hopefully satisfies the condition,

$$\Big| \int_a^b f(x)dx - S \Big| \leq \max(\varepsilon, \text{EPMACH}).$$

Outline of the algorithm

Output is the approximation S and the estimated error E.

1. Initialization

- $S = 0\,;\ E = 0\,;\ N = 0\,;$ (where N denotes the stack pointer)
- Prepare to treat the current interval $[a, b]$.

2. Treatment of the interval

- $k = 0\,;$
- Compute $\widetilde{e}_5$, $E_0 = \widetilde{e}_9$, $I_0 = Q_{11}$, `hint` and the relaxation factor.
- If the convergence criterion is satisfied, do Step 3.

(The FLR method with some modifications)

- While $k < 4$ and `hint`$< P$

  (where $P = 0.2$ if $k = 0$, otherwise $P = 0.16$)

  - $k = k + 1\,;$
  - Compute I_k, E_k and `hint`.
  - If the convergence criterion is satisfied, do Step 3.

end

- If $k = 0$, examine the extraordinary points. If an extraordinary point is detected, do Step 3 after the treatment of the singularity (replace the value of I_0 with the new result and modify E_0).
- $N = N + 1$;
- Store the data on the current interval into the stack.
- Bisect the current interval.
- Prepare to treat the new left-hand subinterval and repeat from Step 2.

3. Exit or preparation for the right subinterval

- $S = S + I_k\,;\ E = E + E_k\,;$
- If $N = 0$, exit with S and E.
- Take the data from the stack to treat the remaining right-hand subinterval.
- $N = N - 1$; and repeat from Step 2.

Table I. Kahaner's Test Problems.

NO	A	B	EXACT	INTEGRAND
(1)	0.0	1.0	1.7182818285e+00	exp(x)
(2)	0.0	1.0	7.0000000000e−01	(int) min(x/0.3, 1)
(3)	0.0	1.0	6.6666666666e−01	sqrt(x)
(4)	−1.0	1.0	4.7942822669e−01	0.92*cosh(x)−cos(x)
(5)	−1.0	1.0	1.5822329637e+00	1/(pow(x, 4)+pow(x, 2)+0.9)
(6)	0.0	1.0	4.0000000000e−01	x∗sqrt(x)
(7)	0.0	1.0	2.0000000000e+00	1/sqrt(x)
(8)	0.0	1.0	8.6697298734e−01	1/(pow(x, 4)+1)
(9)	0.0	1.0	1.1547006690e+00	2/(2+sin(31.4159∗x))
(10)	0.0	1.0	6.9314718056e−01	1/(1+x)
(11)	0.0	1.0	3.7988549304e−01	1/(exp(x)+1)
(12)	0.0	1.0	7.7750463411e−01	x/(exp(x)−1)
(13)	0.1	1.0	9.0986452566e−03	sin(314.159∗x)/(3.14159∗x)
(14)	0.0	10.0	5.0000021117e−01	sqrt(50)∗exp(−50∗3.14159∗pow(x, 2))
(15)	0.0	10.0	1.0000000000e+00	25∗exp(−25∗x)
(16)	0.0	10.0	4.9936380287e−01	50/3.14159/(2500∗pow(x, 2)+1)
(17)	0.01	1.0	1.1213956963e−01	50∗pow(sin(50.0∗3.14159∗x) /(50.0∗3.14159∗x), 2)
(18)	0.0	π	8.3867632338e−01	cos(cos(x)+3∗sin(x)+2∗cos(2∗x) +3∗sin(2∗x)+3∗cos(3∗x))
(19)	0.0	1.0	-1.000000000e+00	log(x)
(20)	−1.0	1.0	1.5643964441e+00	1/(pow(x, 2)+1.005)
(21)	0.0	1.0	2.1080273550e−01	pow(1/cosh(10∗(x−0.2)), 2) +pow(1/cosh(100∗(x−0.4)), 4) +pow(1/cosh(1000∗(x−0.6), 6)

6. Numerical examples

Table II. Comparison of Performance of Adaptive Quadrature Routines.

ERROR REQUIREMENT 1.0e−06

	quadl		DQXG2		AQNN9		**AQN9D**	
NO	N	error	N	error	N	error	N	error
(1)	18	1.4e−13	19	2.2e−16	**11**	2.2e−16	**11**	2.2e−16
(2)	**198**	2.4e−09	673	1.7e−08	211	2.9e−08	211	2.9e−08
(3)	*78*	1.3e−06*	387	4.0e−09	**91**	2.5e−12	**91**	2.4e−12
(4)	18	3.3e−10	41	2.2e−16	**11**	3.8e−14	**11**	3.8e−14
(5)	48	6.8e−11	41	1.3e−15	41	1.0e−11	**25**	5.0e−10
(6)	48	4.6e−08	67	7.8e−09	**41**	5.1e−08	**41**	5.1e−08
(7)	*408*	2.3e−06*	1547	2.0e−07	**91**	1.7e−10	**91**	1.2e−10
(8)	18	1.8e−07	41	2.2e−16	21	3.3e−11	**13**	1.2e−08
(9)	468	3.9e−09	321	4.0e−14	221	9.3e−09	**157**	2.6e−08
(10)	18	5.6e−09	41	2.2e−16	**11**	3.9e−10	**11**	3.9e−10
(11)	18	1.6e−12	27	2.2e−16	**11**	2.4e−14	**11**	2.5e−14
(12)	18	4.0e−15	27	2.2e−16	**11**	4.4e−16	**11**	4.4e−16
(13)	1218	8.1e−11	641	2.2e−16	641	1.1e−10	**469**	3.4e−08
(14)	138	6.3e−09	173	3.5e−14	91	3.4e−10	**75**	7.5e−10
(15)	168	3.9e−11	147	2.2e−16	81	5.5e−10	**69**	5.3e−10
(16)	168	1.3e−10	281	4.6e−12	101	1.5e−09	**97**	3.6e−10
(17)	*768*	8.7e−06*	641	3.8e−13	481	5.6e−09	**365**	1.3e−09
(18)	228	2.3e−13	81	1.6e−12	111	1.6e−09	**75**	4.9e−09
(19)	228	2.4e−07	757	2.9e−08	**91**	2.8e−10	**91**	3.4e−11
(20)	48	5.5e−11	41	0.0e+00	**21**	1.1e−08	**21**	1.1e−08
(21)	*168*	1.1e−03*	*241*	1.1e−03*	*111*	1.1e−03*	*101*	1.1e−03*
	214	81%	297	95%	119	95%	97	95%

The examples in this section are computed in double precision; the machine precision is $2.22\ldots \times 10^{-16}$.

we begin by using Kahaner's test problems (Table I) to compare the present routine AQN9D with `quadl`, a Matlab function which is *based* on `adaptlob` due to Gander and Gautschi, DQXG with KEY= 2 (called DQXG2) due to Favati, Lotti and Romani and AQNN9 due to Ninomiya. Both DQXG2 and AQNN9 are written in FORTRAN while AQN9D is in C and quadl is in MATLAB. Tables II and III show the numbers (N) of function evaluations with actual absolute errors required to satisfy the tolerances $\varepsilon = 10^{-6}$ and 10^{-9}, respectively. The numbers with asterisk mean the failure to satisfy the tolerances. At the bottom of each table, success rates in percentage as well as average numbers of function evaluations are shown for the compared routines.

Table III. Comparison of Performance of Adaptive Quadrature Routines.

ERROR REQUIREMENT 1.0e−09

	quadl		DQXG2		AQNN9		**AQN9D**	
NO	N	error	N	error	N	error	N	error
(1)	18	1.4e−13	19	2.2e−16	**11**	2.2e−16	**11**	2.2e−16
(2)	318	3.1e−10	1017	1.7e−11	**311**	2.8e−11	**311**	2.8e−11
(3)	228	1.0e−09	667	2.7e−12	**131**	2.5e−12	**131**	2.4e−12
(4)	18	3.3e−10	41	2.2e−16	21	5.6e−17	**13**	8.3e−16
(5)	108	2.7e−11	**41**	1.3e−15	61	1.8e−11	45	1.8e−11
(6)	108	1.2e−10	227	7.6e−12	91	8.9e−12	**73**	1.4e−11
(7)	*1128*	1.8e−09*	2347	2.0e−10	251	1.7e−13	**155**	1.0e−12
(8)	48	1.5e−13	41	2.2e−16	41	5.0e−15	**27**	3.3e−16
(9)	1038	4.0e−14	321	4.0e−14	421	5.8e−12	**261**	2.4e−11
(10)	48	1.4e−15	41	2.2e−16	21	3.9e−13	**13**	3.8e−12
(11)	18	1.6e−12	27	2.2e−16	**11**	2.4e−14	**11**	2.5e−14
(12)	18	4.0e−15	27	2.2e−16	**11**	4.4e−16	**11**	4.4e−16
(13)	4068	2.5e−16	**641**	2.2e−16	1271	6.3e−14	777	1.6e−12
(14)	228	1.8e−14	213	0.0e+00	131	2.8e−12	**95**	4.2e−10
(15)	288	3.4e−11	147	2.2e−16	121	5.5e−13	**95**	1.9e−13
(16)	438	8.2e−14	321	2.2e−16	201	4.8e−12	**123**	1.6e−12
(17)	2208	3.9e−13	**641**	3.8e−13	981	1.1e−12	651	2.5e−12
(18)	738	1.6e−14	121	1.9e−14	201	1.1e−12	**103**	8.4e−13
(19)	*498*	1.4e−09*	1101	1.3e−11	171	1.7e−11	**139**	3.0e−13
(20)	48	5.5e−11	41	0.0e+00	61	2.3e−13	**37**	1.9e−14
(21)	**738**	5.1e−13	*241*	1.1e−03*	*221*	1.1e−03*	*143*	1.1e−03*
	588	90%	394	95%	226	95%	154	95%

The examination of Tables II and III reveals the following facts.

1. Integrals of nonsingular functions such as Problems 4, 8, 10, 11 and 12 are easy for adaptive routines to approximate.

2. Integrals of functions with peaks (Problems 14, 15 and 16 as well as 21) are rather easy except for Problem 21, which is very difficult to approximate because of the integrand with three sharp peaks in very small regions. Table III shows that 'quadl' succeeds in approximating the Problem 21 with $\varepsilon = 10^{-9}$.

3. Integrals of singular or discontinuous functions (Problems 2, 3, 7 and 19) appear to be rather difficult problems for DQXG2. The present AQN9D can effectively approximate these problems by the combination of the Ninomiya and the FLR methods and by virtue of the detection of extraordinary points in the Ninomiya scheme.

Table IV. Comparison with the Venter and Laurie method by 32 problems

ERROR REQUIREMENT 1.0e−07

No	Integral	Exact	Venter-Laurie			**AQN9D**	
			P	O	C	N	error
A1	$\int_1^{100} dx/x$	$2\log 10$	153	**63**	**63**	77	9.0e−10
A2	Kahaner.5		93	31	31	**25**	5.0e−10
A3	Kahaner.12		33	15	15	**11**	4.4e−16
A4	$\int_{-4}^{4} dx/(1+x^2)$	$2\tan^{-1} 4$	153	127	77	**61**	1.1e−08
A5	Kahaner.1		33	31	31	**19**	0.0e+00
A6	$\int_0^{\pi/2} dx/(1+\sin^2 x)$	$\pi\sqrt{2}/4$	33	31	31	**21**	5.8e−10
A7	$\frac{1}{120}\int_0^5 \prod_{k=1}^5 (x-k)dx$	$-\frac{47.5}{144}$	33	15	15	**11**	0.0e+00
A8	$\int_{-1}^{1}(\frac{23}{25}\cosh x - \cos x)dx$	(∗1)	33	15	15	**11**	3.8e−14
B9	Kahaner.3		183	**63**	**63**	111	2.5e−12
B10	$\int_{-1}^{1} dx/\sqrt{1-x^2}$	π	2193	8895	2205	**341**	1.1e−09
B11	$\int_0^1 \sqrt{\lvert x^2-0.25\rvert}dx$	(∗2)	363	381	**141**	281	1.1e−09
B12	Kahaner.19		483	1065	495	**111**	3.5e−11
B13	$\int_0^1 \log(\sin(\pi x))\,dx$	$-\log 2$	963	2385	975	**201**	3.7e−10
B14	$\int_0^1 \sqrt{x}\log x\,dx$	$-4/9$	243	**127**	**127**	201	6.6e−11
B15	$\int_0^1 \lvert x-0.4\rvert\,dx$	0.26	213	1207	273	**121**	1.9e−10
B16	$\int_0^1 e^{-2\lvert x-0.4\rvert}dx$	(∗3)	213	1207	273	**121**	3.8e−10
B17	$\int_1^2 \frac{10^{-4}}{(x-1.4)^2+10^{-8}}dx$	(∗4)	603	2733	457	**255**	2.2e−09
B18	$\int_0^1 \log\lvert x-0.3\rvert\,dx$	(∗5)	603	3843	615	**289**	6.4e−09
C19	$\int_0^1 \frac{1}{(x+0.01)^5}dx$	(∗6)	183	127	**123**	189	3.7e−09
C20	$\int_0^1 \frac{1}{\sqrt{x+0.0001}}dx$	(∗7)	333	255	213	**129**	1.8e−10
C21	$\int_0^1 \frac{1}{x+0.0001}dx$	log10001	363	525	307	**157**	1.1e−09
C22	$\int_0^1 \frac{1}{(230x-30)^2+1}dx$	(∗8)	273	953	229	**123**	2.7e−10
C23	$\int_0^1 \frac{1}{x+0.01}dx$	log101	153	**63**	**63**	77	2.7e−10
C24	$\int_0^{10} \frac{50}{\pi(1+2500x^2)}dx$	$\frac{\tan^{-1} 500}{\pi}$	273	255	199	**97**	3.6e−10
D25	Kahaner.9		603	255	309	**181**	2.5e−10
D26	Kahaner.13		2793	**255**	309	567	3.0e−09
D27	$\int_0^{2\pi} x\sin(30x)\cos x dx$	$-\frac{60}{899}\pi$	2289	**255**	721	459	9.1e−10
D28	$\int_0^1 f_1(x)\,dx$	$-20\pi/99$	903	**127**	233	185	2.2e−11
D29	$\int_0^{2\pi} e^{-x}\sin(10x)\,dx$	$\frac{1-e^{-2\pi}}{10.1}$	513	**63**	185	143	1.0e−10
E30	$\int_0^1 f_2(x)\,dx$	1.95	1113	8197	1173	**471**	2.8e−09
E31	$\int_0^1 f_3(x)\,dx$	(∗9)	543	4035	555	**251**	5.1e−10
E32	Kahaner.2		603	4447	615	**251**	1.8e−09
average number of function evaluations →			549	1314	348	173	

P, O and C denote PART, ORDER and CHOICE, respectively.

(∗1) $\frac{46}{25}\sinh 1 - 2\sin 1$ (∗2) $\frac{\pi}{16}+\frac{\sqrt{3}}{4}-\frac{\log(2+\sqrt{3})}{8}$ (∗3) $1-\frac{e^{-0.8}+e^{-1.2}}{2}$

(∗4) $\tan^{-1} 6000 + \tan^{-1} 4000$ (∗5) $0.3\log(0.3)+0.7\log(0.7)-1$

(∗6) $\{10^8-(1.01)^{-4}\}/4$ (∗7) $2\sqrt{1.0001}-\frac{1}{50}$

(∗8) $\frac{\tan^{-1} 200+\tan^{-1} 30}{230}$ (∗9) $0.4+e-e^{0.4}$

4. Integrals of oscillatory functions (Problems 9, 13, 17 and 18) are difficult for adaptive routines. The present AQN9D copes with these problems better than other routines except for the Problems 13 and 17 with $\varepsilon = 10^{-9}$.

Venter and Laurie (2002) computes 32 problems shown in their paper with tolerance $\varepsilon = 10^{-7}$. We compare the present method with their results in Table IV, where the numbers of function evaluations required to satisfy the given tolerance are shown in the fourth to seventh columns. In Table IV, $f_1(x)$, $f_2(x)$ and $f_3(x)$ are given, respectively, by

$$f_1(x) = 4\pi^2 x \sin(20\pi x)\cos(2\pi x),$$

$$f_2(x) = \begin{cases} x, & x \le 0.1 \\ x+1, & 0.1 < x \le 0.45 \\ x+2, & x > 0.45, \end{cases} \qquad f_3(x) = \begin{cases} 1, & x < 0.4 \\ e^x, & x \ge 0.4. \end{cases}$$

Table IV shows that the present method is effective for four types of integrands, Group A (smooth), B (singular), C (peak) and E (discontinuous) except for A1, B9, B11, B14, C19, C23, and D26 $\sim$ D29 (oscillatory).

Acknowledgements

The authors thank Hiroshi Sugiura for his helpful comments. We also thank referees for their valuable suggestions to improve the presentation of the paper. Particularly, one of the referees suggested us to compare our routine in performance with that due to Venter and Laurie (2002) for 32 problems shown in their paper.

References

J. Berntsen and T. O. Espelid, Error estimation in automatic quadrature routines. *ACM Transaction on Mathematical Software*, 17:233–252, 1991.

P. J. Davis and P. Rabinowitz. *Methods of Numerical Integration 2nd edit.* Academic Press, Orlando, 1984.

P. Favati, G. Fiorentino, G. Lotti, and F. Romani, Local error estimates and regularity tests for the implementation of double adaptive quadrature. *ACM Transaction on Mathematical Software*, 23:16–31, 1997.

P. Favati, G. Lotti, and F. Romani. Interpolatory integration formulas for optimal composition. *ACM Transaction on Mathematical Software*, 17:207–217, 1991.

P. Favati, G. Lotti, and F. Romani. ALGORITHM 691 Improving QUADPACK automatic integration routines. *ACM Transaction on Mathematical Software*, 17:218–232, 1991.

P. Favati, G. Lotti, and F. Romani, Theoretical and practical efficiency measures for symmetric interpolatory quadrature formulas. *BIT*, 34:546–557, 1994.

P. Favati, G. Lotti, G. Di Marco and F. Romani, Asymptotic behavior of automatic quadrature. *J. Complexity*, 10:296–340,1994.

W. Gander and W. Gautschi, Adaptive quadrature—revisited. *BIT*, 40:84–101, 2000.

D. K. Kahaner, *Computation of numerical quadrature formulas.* In J. R. Rice editor, *Mathematical Software*, 229–259, 1971, Academic Press.

I. Ninomiya, Improvement of adaptive Newton-Cotes quadrature methods. *J. Information Processing*, 3:162–170, 1980.

J. Oliver, A practical strategy for the Clenshaw-Curtis quadrature method. *J. Inst. Maths Applics*, 8:53–56, 1971.

J. Oliver, A doubly-adaptive Clenshaw-Curtis quadrature method. *Computer J.*, 15:141–147, 1972.

R. Piessens, E. de Doncker-Kapenga, C. W. Überhuber and D. K. Kahaner, *QUADPACK, A subroutine Package for Automatic Integration.* Springer-Verlag, Berlin, 1983.

L. Plaskota and G. W. Wasilkowski, Adaption allows efficient integration of functions with unknown singularities. *Numerische Mathematik*, 102:123–144, 2005.

H. Sugiura and T. Sakurai, On the construction of high-order integration formulae for the adaptive quadrature method. *J. Computational and Applied Mathematics* 28:367–381, 1989.

A. Venter and D. P. Laurie, A doubly adaptive integration algorithm using stratified rules. *BIT*, 42:183–193, 2002.

Address for Offprints: Takemitsu Hasegawa, Department of Information Science, University of Fukui, Fukui, 910-8507, Japan