Mathematical Modelling and Analysis
Volume 11 Number 3, 2006, pages 315–330
 ISSN 1392-6292 print, ISSN 1648-3510 online

ERROR ESTIMATION FOR QUADRATURE FORMULAS BASED ON EQUALLY SPACED NODES

K. PLUKAS and D. PLUKIENĖ

Kaunas University of Technology
Studentų 50-407, 3031 Kaunas, Lithuania
E-mail: Kostas.Plukas@ktu.lt

Received October 15, 2005; revised April 04, 2006; published online September 15, 2006

Abstract. The error estimation for quadrature formulas based on equally spaced nodes is discussed in this paper. The error estimates use embedded formulas and they are obtained for Newton-Cotes and Hermitian quadrature formulas. The coefficients of these formulas and error estimates are presented. The locally adaptive integration procedures implementing the truncation error estimation method proposed in this paper were developed in MATLAB and results of appropriate comparative tests are presented.
Key words: numerical integration, quadrature formula, error estimate, locally adaptive procedure

1. Introduction

Numerical approximation of integrals $If = \int_a^b f(x)\,dx$ is one of the most frequently used numerical procedure. Numerous techniques that are available for analytical evaluation of integrals are inadequate for many problems that arise in the real physical world. Consequently the value of the definite integral must be approximated by some numerical methods.

Most numerical integration methods involve constructing the interpolating polynomial of $f(x)$ and then integrating this polynomial in order to obtain an approximation to the integral of $f(x)$. All interpolatory quadrature formulas are uniquely characterized by the choice of their nodes $x_1, x_2, \ldots, x_n$. The most important sets of interpolatory quadrature formulas are the following.

Closed Newton-Cotes formulas. Closed Newton-Cotes formulas Q_n^N, $n \geqslant 2$ are interpolatory formulas based on equidistant nodes

$$x_i = a + (i-1)h, \;\; i = 1, 2, \ldots, n, \; h = \frac{b-a}{n-1}$$

and can be presented as

$$Q_n^N f = h \sum_{i=1}^{n} w_i y_i, \tag{1.1}$$

where w_i are the coefficients called the weights, $y_i = f(x_i)$ are the values of the integrand evaluated at the nodes x_i, and h is an integration step.

Closed Hermitian quadrature formulas. Closed Hermitian quadrature formulas Q_n^H are also based on interpolation functions constructed on equidistant nodes and can be presented as

$$Q_n^H f = h \sum_{i=1}^{n} w_i^H y_i + h^2 \sum_{i=1}^{n} w_i^{H^*} y_i', \tag{1.2}$$

where $w_i^H, w_i^{H^*}$ are coefficients called the weights, $y_i = f(x_i)$ and $y_i' = f'(x_i)$ are the values of the integrand and its derivative evaluated at the nodes x_i.

Gaussian quadrature formulas. Gauss-Legendre quadrature formulas Q_n^G are also based on interpolation functions constructed on the nodes chosen as zeros of orthogonal Legendre polynomials and can be presented as

$$Q_n^G f = h \sum_{i=1}^{n} w_i^G y_i, \tag{1.3}$$

where w_i^G are coefficients called the weights.

DEFINITION 1. [2]. A quadrature formula Q_n has degree d if it integrates exactly all polynomials of degree $\leq d$ and fails to integrate exactly $f(x) = x^{d+1}$.

DEFINITION 2. The truncation error for the quadrature formula is the difference between the numerical and analytical (exact) values of the integral.

The truncation error for the quadrature formulas (1.1)-(1.3) is given in the form

$$Ch^{\alpha} f^{(\beta)}(\xi), \tag{1.4}$$

where C is a constant, α is an integer dependent on the degree of quadrature formula, β is an integer indicating the order of the derivative, and ξ is some point in the interval (a, b).

Suppose that we want to approximate integral $\int_a^b f(x)\,dx$ to within a specified tolerance $\varepsilon > 0$. For solving this problem we need: a quadrature formula, an integration strategy, and a formula for estimation of the truncation error.

An important problem associated with automatic numerical integration problem is that of error estimation. It is difficult to compute error estimates that are both reliable and accurate. An obvious way to obtain an error estimate is to compare a pair of integration formulas Q_{n1} and Q_{n2}. The assumption that Q_{n2} returns significantly better results than Q_{n1} implies that error estimate formula

$$\Delta = \left| Q_{n1}f - Q_{n2}f \right| \tag{1.5}$$

gives a sufficiently accurate estimate of the truncation error for a large class of integrands. A necessary condition in this technique is the assumption that the degree of accuracy Q_{n2} is larger than Q_{n1}.

A simple way to obtain two different estimates Q_{n1} and Q_{n2} for If is to use compound rules. Integration formula Q_n is applied first to $[a, b]$ and then to the two subintervals $[a, (a+b)/2]$ and $[(a+b)/2, b]$. Adding up the results of both subintervals leads to the formula Q_{2n} which is more accurate than Q_n. If f is a sufficiently smooth function and Q_n has the degree p then formula

$$E = \left| \frac{Q_{2n}f - Q_nf}{2^p - 1} \right| \tag{1.6}$$

may be used as an error estimation formula.

The main drawback of formula (1.6) is necessity to approximate definite integral twice. Therefore for minimization of computation costs the estimates Q_{n1} and Q_{n2} are calculated using embedded formulas. Suppose the $n + 1$ nodes are fixed and a unique interpolatory quadrature formula $Q_{n2} = Q_{n+1}$ of degree d is constructed. The formula Q_{n1} may be constructed removing one or some points from the initial set of nodes checking that the interpolatory rule based on the remaining nodes actually has lower degree of accuracy than the formula Q_{n2}.

Definition 3. [2]. A rule

$$Nf = \sum_{i=1}^{n} u_i f(x_i) \tag{1.7}$$

is a *null rule* if it has at least one nonzero weight, and in addition

$$\sum_{i=1}^{n} u_i = 0.$$

A *null rule* is said to have degree d if it integrates to zero all polynomials of degree $\leq d$ and fails to do so with $f(x) = x^{d+1}$. A *null rule* N and interpolatory rule Q, based on the same set of points, are *equally strong* if the null rule is scaled such that $\|N\|_2 = \|Q\|_2$, where

$$\|N\|_2 = \Big(\sum_{i=1}^{n} u_i^2 \Big)^{1/2}, \quad \|Q\|_2 = \Big(\sum_{i=1}^{n} w_i^2 \Big)^{1/2}$$

and w_i are the coefficients of rule Q. The term *null rule* was first used in 1965 by J. N. Lyness [15]. The general theory of *null rules* was presented in 1991 by J. Berntsen and T.O. Espelid in [2].

Gauss formulas have one severe drawback: two arbitrary Gauss formulas Q_{n1}^G and Q_{n2}^G with $n_2 > n_1$ do not have any identical nodes (except,

perhaps, for the interval center). The usual procedure for obtaining a practical error estimate by formula (1.5) requires too many integrand values and therefore too much computational effort. This disadvantage was overcome by using a seemingly obvious method proposed by A. S. Kronrod in 1965 [17]. Suppose that n nodes $x_1, x_2, \ldots, x_n$ are the abscissas of Q_{n1}^G, then formula Q_{n2}^G with $(2n+1)$ nodes is constructed in such a way that its degree of accuracy is the highest possible, i.e. new nodes are contained in the intervals $(a, x_1), (x_1, x_2), \ldots, (x_n, b)$.

Since the first automatic algorithm for the numerical calculation of definite integrals was given by McKeeman [16] in 1963, many new algorithms, including non-adaptive, adaptive (locally and globally), and double adaptive (locally and globally), have been developed [3, 4, 5, 6, 7, 8, 9, 13, 14].

Recently Gander and Gautschi [9] published a paper describing two new adaptive quadrature Matlab codes *adaptsim* and *adaptlob* based on Simpson's and the four-point Gauss-Lobatto rules respectively. Espelid in [3] discussed null rules based on divided differences and constructed error estimator using sequences of null rules with the intention to increase the reliability for both *adaptsim* and *adaptlob* codes. In addition two new Matlab codes *coteda* and *coteglob* are developed, they use a locally and a globally double adaptive strategy respectively. Both algorithms use sequences of null rules in their local error estimations. These new codes make use of both the five-point closed Newton-Cotes rule and the nine-point closed Newton-Cotes rule.

The aims of this paper are:

1. To compute the null rules with the smallest truncation error for Newton-Cotes and Hermitian quadrature formulas.
2. To develop and to test adaptive Matlab procedures based on the constructed null rules and the nine-point closed Newton-Cotes and the four-point closed Hermitian quadrature formulas.
3. To compare numerical results with Gander–Gautschi and Espelid results from the well known test kit given in [3, 9].

2. Truncation Error Evaluation

Suppose that quadrature Q is based on nodes $x_0 = a, x_1, x_2, \ldots, x_n = b$ and it has the degree d and quadrature Q_1 is based on the same nodes except of nodes $x_{k_1}, \ldots, x_{k_q}, k_i \in \{1, 2, \ldots, n-1\}, i = \overline{1,q}, q \geq 1$ and it has degree $d_1 \leq (d-q)$. Usually, the nodes $x_{k_i}, i = \overline{1,q}$ are chosen in such a way that the quadrature formula Q_1 would have the smallest theoretical truncation error, i.e. that the absolute value of the constant C in the truncation error formula (1.4) would be the smallest.

Suppose that Δ and Δ_1 are the values of truncation error for formulas Q and Q_1 respectively. Thus,

$$If = Qf + \Delta = Q_1 f + \Delta_1. \tag{2.1}$$

Equivalently, (2.1) can be rewritten as

$$\delta = \left|Qf - Q_1 f\right| = \left|\Delta_1 - \Delta\right|. \tag{2.2}$$

It is obvious that formula (2.2) is the null rule of degree $d_0 = \min(d, d_1)$. It implies that formula (2.2) can be used for local error estimation.

We may expect that error estimate formula (2.2) on the narrow integration interval will be exact enough because

$$|f(x) - P_n(x)| \approx |P_{n+1}(x) - P_n(x)|,$$

when interpolating step is sufficiently small, where $P_n(x)$ and $P_{n+1}(x)$ are interpolating polynomials for $f(x)$ [1]. This statement can be proved by using the Taylor series in the interval $[x_j, x_{j+1}]$.

Formula (2.2) for the closed Newton-Cotes rules can be written as

$$\delta = \left|Qf - Q_1 f\right| = \left|h(w_1^\delta y_1 + w_2^\delta y_1 + \ldots + w_n^\delta y_n)\right|, \tag{2.3}$$

where $w_i^\delta = w_i^Q - w_i^{Q_1}, i = 0, \ldots, n$, and the coefficients w_i^Q and $w_i^{Q_i}$ are the weights in formula (1.1) for the quadrature formulas Q and Q_1 respectively.

In a similar way formula (2.2) for closed Hermitian rule can be written as

$$\delta = \left|h(w_1^\delta y_1 + w_2^\delta y_1 + \ldots + w_n^\delta y_n) + h^2(w_1^{\delta^*} y_1' + w_2^{\delta^*} y_2' + \ldots + w_n^{\delta^*} y_n')\right|. \tag{2.4}$$

3. The Error Estimation for Newton-Cotes Quadrature Formulas

Suppose that the closed Newton-Cotes quadrature formula Q is based on equally spaced nodes $x_i, i = \overline{0,n}$ and it has the degree n. In this section we will discuss a procedure for construction of a closed interpolatory quadrature formula Q_1, which is based on the same nodes x_i except of one node $x_k, k \in \{1, 2, \ldots, n-1\}$ and the truncation error of which is the smallest possible.

The construction of quadrature Q_1 consists of the following steps.

1. Substitute the integrand function $f(x)$ with the Lagrange interpolating polynomial $P_{n-1}(x)$ that agrees with $f(x)$ at nodes $x_i, i = \overline{0,n}, i \neq k, k_i \in \{1, 2, \ldots, n-1\}$.
2. Write the resulting approximation of the integral

$$\int_a^b f(x)\,dx \approx Q_1 f = \int_a^b P_{n-1}(x)\,dx = h \sum_{i=0, i\neq k}^{n} w_i^{Q_1} y_i. \tag{3.1}$$

3. For each $k = 1, 2, \ldots, n-1$ evaluate the truncation errors of formula (3.1).
4. Choose the smallest theoretical truncation error. Let x_k be a node for which this error is received.
5. Calculate the coefficients $w_i^{Q_i}$ in formula (3.1).

The theoretical truncation error Δ_1 for quadrature Q_1 is calculated using the error formula of the interpolating polynomial $P_{n-1}(x)$ and the Intermediate Value Theorem. It can be written as

$$\Delta_1 = \frac{f^{(n)}(\xi)}{n!} \int_{x_0=a}^{x_n=b} (x-x_0)(x-x_1)\dots(x-x_{k-1})(x-x_{k+1})\dots(x-x_n)\,dx, \quad (3.2)$$

where ξ is some point in the interval (a,b). Using substitution $t = \frac{x-a}{h} - l$, where $h = \frac{b-a}{n}, l = \frac{n}{2}$, formula (3.2) can be written as

$$\Delta_1 = \frac{f^{(n)}(\xi)}{n!} h^{n+1} \int_{-l}^{l} (t+l)(t+l-1)\dots(t+l-k+1)(t+l-k-1)\dots(t-l)\,dt. \quad (3.3)$$

Formula (3.3) is more convenient for calculation, because it does not depend on the actual values of x_i.

The coefficients in formula (3.1) can be computed using the method of undetermined coefficients. Formula (3.1) must be exact for polynomials of degree $\leq (n-1)$. In this case, we need to solve the following system of linear equations

$$\begin{cases} w_0^{Q_1}\varphi_j(l_0) + w_1^{Q_1}\varphi_j(l_1) + \dots + w_{k-1}^{Q_1}\varphi_j(l_{k-1}) \\ \qquad + w_{k+1}^{Q_1}\varphi_j(l_{k+1}) + \dots + w_n^{Q_1}\varphi_j(l_n) = m_j, \quad j = \overline{0,n-1}, \end{cases} \quad (3.4)$$

where $\varphi_j(l_i) = (-l+i)^j$, $m_j = \int_{-l}^{l} x^j dx$, for all $i = 0,1,2,\dots,k-1,k+1,\dots,n$, and $j = \overline{0,n-1}$.

Table 1 shows coefficients of the closed Newton-Cotes quadrature formulas, closed interpolatory quadrature formulas Q_1 and error estimation formula

$$\Delta = c_\Delta(\tilde{p}) \frac{f^{(\tilde{p}-1)}(\xi)}{(\tilde{p}-1)!} h^{tildep}.$$

The table is divided into several parts, each of which consists of the three rows. The coefficients of the closed Newton-Cotes quadrature formula (quadrature Q) of the degree p are put in the first row of each part. The calculated coefficients of the quadrature Q_l with the smallest truncation error and of the degree $(p-2)$ are given in the second row, and the coefficients of the truncation error formula (2.3) are listed in the third row (denoted "δ_n"). These coefficients give the difference between coefficients in the first and in the second rows.

The column "n" represents the number of equally sized subintervals into which integration interval is divided. The column "p" gives the degree of quadrature formulas, and the column "c_Δ" displays theoretical formulas for evaluation of the truncation error. For example, if $n = 4$, then the truncation

Table 1. The coefficients of the symmetric embedded Newton-Cotes formulas and truncation error estimation formulas.

n	p	w_0	w_l	w_2	w_3	w_4	w_5	c_Δ	$\tilde{p}$
	2	$\frac{1}{3}$	$\frac{4}{3}$	$\frac{1}{3}$				$\frac{-4}{15}$	5
2	1	1	0	1				$\frac{-4}{3}$	3
	δ_2	$\frac{1}{3}$	$\frac{4}{3}$	$\frac{1}{3}$					
	4	$\frac{14}{45}$	$\frac{64}{45}$	$\frac{24}{45}$	$\frac{64}{45}$	$\frac{14}{45}$		$\frac{-128}{21}$	7
4	3	$\frac{2}{9}$	$\frac{16}{9}$	0	$\frac{16}{9}$	$\frac{2}{9}$		$\frac{32}{15}$	5
	δ_4	$\frac{4}{45}$	$\frac{-16}{45}$	$\frac{24}{45}$	$\frac{-16}{45}$	$\frac{4}{45}$			
	6	$\frac{41}{140}$	$\frac{216}{140}$	$\frac{27}{140}$	$\frac{272}{140}$	$\frac{27}{140}$	$\frac{216}{140}$	$\frac{-1296}{5}$	9
6	5	$\frac{14}{50}$	$\frac{81}{50}$	0	$\frac{110}{50}$	0	$\frac{81}{50}$	$\frac{324}{35}$	7
	δ_6	$\frac{9}{700}$	$\frac{-54}{700}$	$\frac{135}{700}$	$\frac{-180}{700}$	$\frac{135}{700}$	$\frac{54}{700}$		
	8	$\frac{3956}{14175}$	$\frac{23552}{14175}$	$\frac{-3712}{14175}$	$\frac{41984}{14175}$	$\frac{-18160}{14175}$	$\frac{41984}{14175}$	$\frac{-606208}{33}$	11
8	7	$\frac{1908}{6615}$	$\frac{10496}{6615}$	0	$\frac{16128}{6615}$	$\frac{-4144}{6615}$	$\frac{16128}{6615}$	$\frac{-118784}{315}$	9
	δ_8	$\frac{-928}{99225}$	$\frac{7424}{99225}$	$\frac{-25984}{99225}$	$\frac{51968}{99225}$	$\frac{-64960}{99225}$	$\frac{51968}{99225}$		
	10	$\frac{80335}{299376}$	$\frac{531500}{299376}$	$\frac{-242625}{299376}$	$\frac{1362000}{299376}$	$\frac{-1302750}{299376}$	$\frac{2136840}{299376}$	$\frac{-53854\cdot 10^4}{273}$	13
10	9	$\frac{11690}{40824}$	$\frac{65125}{40824}$	0	$\frac{97500}{40824}$	$\frac{-23250}{40824}$	$\frac{106110}{40824}$	$\frac{-647\cdot 10^4}{99}$	11
	δ_{10}	$\frac{-16175}{898128}$	$\frac{16175}{898128}$	$\frac{-727875}{898128}$	$\frac{1941000}{898128}$	$\frac{-3396750}{898128}$	$\frac{4076100}{898128}$		

error estimate formula (2.3) for the five point closed Newton-Cotes quadrature is presented as

$$\delta_4 = \frac{4}{45}\Big|h(y_0 - 4y_1 + 6y_2 - 4y_3 + y_4)\Big|. \tag{3.5}$$

Remark 1. If $n = 2$ or 4, then we can use the composite rule for estimation of the truncation error: the composite trapezoid rule for $n = 2$ and the composite Simpson's rule for $n = 4$. In these cases we will get four times less the truncation error values than the corresponding values reported in Table 1.

Remark 2. Table 2 shows the ratio between the absolute values of the theoretical truncation errors of error estimation formulas $\delta_n(x_k)$. Here $\delta_n(x_k)$ denotes the theoretical error estimate for the n degree quadrature formula based on equally spaced nodes $x_0, x_1, \ldots, x_{k-1}, x_{k+1}, \ldots, x_n$. Values $|\delta_n(x_k)|$ and $|\delta_n(x_{n-k})|$ are equal.

It follows from Table 2 that optimal error estimator is obtained for $\delta_n(x_2)$. This fact can be explained in the way that for the *even integer* n the value of integral $\int_{x_0=a}^{x_n=b} (x - x_0)(x - x_1)\ldots(x - x_{k-1})(x - x_{k+1})\ldots(x - x_n)\,dx$ has the smallest value if $x_k = x_2$. It based on the properties of polynomial $(x - x_0)(x - x_1)\ldots(x - x_{k-1})(x - x_{k+1})\ldots(x - x_n)$ [12].

Table 2. The values of the ratio $|\frac{\delta_n(x_k)}{\delta_n(x_2)}|$.

n	x_1	x_2	x_3	x_4	x_5
4	4	1	4		
6	20	1	$\frac{68}{9}$	1	20
8	$\frac{644}{29}$	1	$\frac{164}{29}$	$\frac{227}{116}$	$\frac{164}{29}$
10	$\frac{6378}{647}$	1	$\frac{1362}{647}$	$\frac{5211}{4529}$	$\frac{35614}{23645}$

4. The Truncation Error Estimation for Hermitian Quadrature formulas

The construction of the $(2n+1)$-th order Hermitian quadrature formula Q, which is based on equidistant nodes, consists of the following steps.

1. Divide the given integration interval $[a,b]$ into equally sized subintervals by nodes $x_i = x_0 + ih, i = \overline{0,n}$, where $x_0 = a$, $h = (b-a)/n$, $x_n = b$.
2. Substitute integrand function $f(x)$ with the Hermite interpolating polynomial $H(x)$ obeying the following equalities:

$$H(x_i) = y_i, \quad H'(x_i) = y_i', \quad y_i = f(x_i), \quad y_i' = f'(x_i), \quad i = \overline{0,n}.$$

3. Write the resulting approximation in the form

$$\int_a^b f(x)\,dx \approx \int_a^b H(x)\,dx = h(w_0y_0 + w_1y_1 + \cdots + w_ny_n) + h^2(w_0^*y_0' + w_1^*y_1' + \cdots + w_n^*y_n'). \quad (4.1)$$

4. Calculate the coefficients w_i and w_i^*.

The coefficients in formula (4.1) are computed using the undetermined coefficients method. Hence, like for the Newton-Cotes formulas, we need to solve the following system of linear equations

$$\begin{cases} w_0\varphi_j(l_0) + w_1\varphi_j(l_1) + \cdots + w_n\varphi_j(l_n) \\ \quad + w_0^*\varphi_j'(l_0) + w_1^*\varphi_j'(l_1) + \cdots + w_n^*\varphi_j'(l_n) = m_j, \quad j = \overline{0,2n+1}, \end{cases} \quad (4.2)$$

where

$$\varphi_j(l_i) = (-[n/2]+i)^j, \quad \varphi_j'(l_i) = j(-[n/2]+i)^{j-1}, \quad i = \overline{0,n},$$

$$m_j = \int_{-[n/2]}^{n-[n/2]} x^j\,dx, \quad j = \overline{0,2n+1}.$$

The theoretical truncation error Δ for Hermitian quadrature Q can be computed using the truncation error formula of the Hermitian interpolating polynomial and the Intermediate Value Theorem. This formula like (3.3) can be written as follows

$$\Delta = \frac{f^{(2n+2)}(\xi)}{(2n+2)!} h^{2n+3} \int_0^n \prod_{l=0}^{n} (t-l)^2 \, dt, \tag{4.3}$$

where $t = \frac{x-a}{h}$, $h = \frac{b-a}{n}$.

Suppose that Hermitian quadrature formula Q is based on equally spaced nodes $x_i, = \overline{0,n}$ and it has degree $p = 2n+1$. Then Hermitian quadrature formula Q_1 of degree $(p-q)$ for integer $q > 0$ is obtained by the following steps.

1. Substitute integrand function $f(x)$ with the Hermite interpolating polynomial $H_1(x)$ obeying the following equations

$$H_1(x) = y_i, \quad i = \overline{0,n}, \quad H_1'(x) = y_i', \quad i = \overline{0,n}, \; i \neq k_1, k_2, \ldots, k_q,$$

where $x_i, i = \overline{0,n}$ are the nodes of the quadrature Q. The nodes

$$x_{k_i} \in \{x_0, x_1, \ldots, x_n\}, \quad i = \overline{1,q}$$

are chosen in such a way that the theoretical truncation error for quadrature Q_1 would be the smallest.

2. Write the resulting approximation in the form

$$\int_a^b f(x)\,dx \approx \int_a^b H_1(x)\,dx = h \sum_{i=0}^{n} w_i y_i + h^2 \sum_{i=0}^{n} w_i^* y_i', \tag{4.4}$$

where $w_{k_i}^* = 0$, $i = \overline{1,q}$.

3. Calculate the coefficients w_i and w_i^*.

The coefficients in (4.4) are computed by using the undetermined coefficients method. Formula (4.4) has degree $p_1 \leq (2n+1-q)$.

The theoretical truncation error Δ_1 for Hermitian quadrature Q_1 is calculated using the error formula of the Hermite interpolating polynomial and the Intermediate Value Theorem. This formula like (4.3) can be written as follows

$$\Delta_1 = \frac{f^{(2n+2-q)}(\xi)}{(2n+2-q)!} h^{2n+3-q} \int_0^n \prod_{l=0}^{n} (t-l)^s \, dt, \tag{4.5}$$

where $t = \frac{x-a}{h}$, $h = \frac{b-a}{n}$ and if $l = k_i$, $i = \overline{1,q}$ then $s = 1$ else $s = 2$.

Tables 3 and 4 show calculated coefficients of the Hermitian quadrature formulas Q and Q_1. The structure of both tables is the same as that of Table 1.

Tables 5 and 6 show coefficients of the truncation error formulas of Hermitian quadrature.

Table 3. The coefficients of the embedded Hermitian quadrature formulas.

n	δ	w_0	w_1	w_2	w_3	w_4	w_5	w_6
1	3	$\frac{1}{2}$	$\frac{1}{2}$					
	2	$\frac{1}{3}$	$\frac{2}{3}$					
2	5	$\frac{7}{15}$	$\frac{16}{15}$	$\frac{7}{15}$				
	4	$\frac{4}{15}$	$\frac{16}{15}$	$\frac{10}{15}$				
3	7	$\frac{93}{224}$	$\frac{243}{224}$	$\frac{243}{224}$	$\frac{93}{224}$			
	5	$\frac{39}{80}$	$\frac{81}{90}$	$\frac{81}{90}$	$\frac{39}{80}$			
4	9	$\frac{3202}{8505}$	$\frac{8192}{8505}$	$\frac{11232}{8505}$	$\frac{8192}{8505}$	$\frac{3202}{8505}$		
	7	$\frac{434}{945}$	$\frac{1024}{945}$	$\frac{864}{945}$	$\frac{1024}{945}$	$\frac{434}{945}$		
5	11	$\frac{319085}{912384}$	$\frac{691875}{912384}$	$\frac{1270000}{912384}$	$\frac{1270000}{912384}$	$\frac{691875}{912384}$	$\frac{319085}{912384}$	
	9	$\frac{175515}{435456}$	$\frac{455625}{435456}$	$\frac{457500}{435456}$	$\frac{457500}{435456}$	$\frac{455625}{435456}$	$\frac{175515}{435456}$	
6	13	$\frac{3310219}{10010000}$	$\frac{5014656}{10010000}$	$\frac{11161125}{10010000}$	$\frac{21088000}{10010000}$	$\frac{1161125}{10010000}$	$\frac{5014656}{10010000}$	$\frac{3310219}{10010000}$
	11	$\frac{144341}{385000}$	$\frac{377784}{385000}$	$\frac{475875}{385000}$	$\frac{314000}{385000}$	$\frac{475875}{385000}$	$\frac{377784}{385000}$	$\frac{144341}{385000}$

Table 4. The coefficients of the embedded Hermitian quadrature formulas.

n	δ	w_0^*	w_1^*	w_2^*	w_3^*	w_4^*	w_5^*	w_6^*	c_Δ	$\tilde{p}$
1	3	$\frac{1}{12}$	$\frac{-1}{12}$						$\frac{1}{30}$	5
	2	0	$\frac{-1}{6}$						$\frac{1}{12}$	4
2	5	$\frac{1}{15}$	0	$\frac{-1}{15}$					$\frac{16}{105}$	7
	4	0	$\frac{-4}{15}$	$\frac{-2}{15}$					$\frac{4}{15}$	6
3	7	$\frac{57}{1120}$	$\frac{-81}{1120}$	$\frac{81}{1120}$	$\frac{-57}{1120}$				$\frac{81}{70}$	9
	5	$\frac{3}{40}$	0	0	$\frac{1}{40}$				$\frac{81}{140}$	7
4	9	$\frac{116}{2835}$	$\frac{-512}{2835}$	0	$\frac{512}{2835}$	$\frac{-116}{2835}$			$\frac{10240}{693}$	11
	7	$\frac{20}{315}$	0	0	0	$\frac{-20}{3155}$			$\frac{2048}{315}$	9
5	11	$\frac{36975}{1064448}$	$\frac{-314375}{1064448}$	$\frac{-272500}{1064448}$	$\frac{272500}{1064448}$	$\frac{314375}{1064448}$	$\frac{-369975}{1064448}$		$\frac{3610625}{12012}$	13
	9	$\frac{1725}{36288}$	$\frac{-3750}{36288}$	0	0	$\frac{3750}{36288}$	$\frac{-1725}{36288}$		$\frac{68125}{924}$	11
6	13	$\frac{30711}{10010000}$	$\frac{-409536}{10010000}$	$\frac{-726975}{10010000}$	0	$\frac{726975}{1001\cdot 10^4}$	$\frac{409536}{1001\cdot 10^4}$	$\frac{-30711}{1001\cdot 10^4}$	$\frac{1306368}{143}$	15
	11	$\frac{111}{2750}$	$\frac{-486}{2750}$	0	0	0	$\frac{486}{2750}$	$\frac{-111}{2750}$	$\frac{8374752}{5005}$	13

5. Numerical Results

The MATLAB locally adaptive (the subintervals are processed from left to right until the integral over each subinterval satisfies the relative error requirement) integration procedures: *ncg*9 and *hermit*5 were developed. Procedure *ncg*9 uses the closed nine-point Newton-Cotes quadrature formula. Procedure

Table 5. The coefficients of the truncation error formula of Hermitian quadrature.

n	δ	w_0	w_1	w_2	w_3	w_4	w_5	w_6
1	δ_1^1	$\frac{1}{6}$	$\frac{-1}{6}$					
2	δ_2^1	$\frac{1}{5}$	0	$\frac{-1}{5}$				
3	δ_3^2	$\frac{-81}{1120}$	$\frac{81}{1120}$	$\frac{81}{1120}$	$\frac{-81}{1120}$			
4	δ_4^2	$\frac{-704}{8505}$	$\frac{-1024}{8505}$	$\frac{3456}{8505}$	$\frac{-1024}{8505}$	$\frac{-704}{8505}$		
5	δ_5^2	$\frac{-1021875}{19180064}$	$\frac{-5518125}{19180064}$	$\frac{6540000}{19180064}$	$\frac{6540000}{19180064}$	$\frac{-5518125}{19180064}$	$\frac{-1021875}{19180064}$	
6	δ_6^2	$\frac{-442647}{10010000}$	$\frac{-4807728}{10010000}$	$\frac{-1211625}{10010000}$	$\frac{12924000}{10010000}$	$\frac{-1211625}{10010000}$	$\frac{-4807728}{10010000}$	$\frac{-442647}{10010000}$

Table 6. The coefficients of the truncation error formula of Hermitian quadrature.

n	δ	w_0^*	w_1^*	w_2^*	w_3^*	w_4^*	w_5^*	w_6^*
1	δ_1^1	$\frac{1}{12}$	$\frac{1}{12}$					
2	δ_2^1	$\frac{1}{15}$	$\frac{4}{15}$	$\frac{1}{15}$				
3	δ_3^2	$\frac{-27}{1120}$	$\frac{-81}{1120}$	$\frac{81}{1120}$	$\frac{27}{1120}$			
4	δ_4^2	$\frac{-64}{2835}$	$\frac{-512}{2835}$	0	$\frac{512}{2835}$	$\frac{64}{2835}$		
5	δ_5^2	$\frac{-40875}{3193344}$	$\frac{-613125}{3193344}$	$\frac{-817500}{3193344}$	$\frac{613125}{3193344}$	$\frac{40875}{3193344}$		
6	δ_6^2	$\frac{-9693}{10010000}$	$\frac{-232632}{10010000}$	$\frac{-726975}{10010000}$	0	$\frac{726975}{10010000}$	$\frac{232632}{10010000}$	$\frac{-9693}{10010000}$

*hermit*5 uses the closed five-point Hermite quadrature formula. Both formulas have degree of precision equal to nine. Formulas (2.3) and (2.4) are used for local error estimation in procedures *ncg*9 and *hermit*5 respectively.

Rounding error may influence both the estimate of the local integral and the evaluation of truncation error. We will, as in [3], define a certain noise level for the problem. All local errors below this noise level in both procedures *ncg*9 and *hermit*5 are defined to be zero, thus avoiding subdividing such intervals further. The noise level in both procedures is: $is * 10^{-17}$, where $is \neq 0$, is a rough estimate of a modulus of the integral.

In both procedures we are using the conventional termination criteria over each subinterval $[u, v]$ of the initial interval $[a, b]$:

$$(is + \delta = is) \text{ or } (m = u) \text{ or } (m = \nu), \tag{5.1}$$

where $m = (u + \nu)/2$ and δ is the truncation error value over $[u, v]$.

In comparing adaptive quadrature routines, the most important characteristics are [9]:

- *efficiency*, as measured by the number of function evaluations required to meet a given error tolerance;
- *reliability*, the extent to which the requested error tolerance is achieved;

- *tolerance responsiveness*, as measured by the efficiency of sensitiveness to changes in the error tolerance.

We have tested both procedures *ncg*9 and *hermit*5 discussed in this paper on 23 test problems used in [3, 9]. They are picked from two different sources: the 21 first problems come from Kahaner [11] and the last two are picked from [10].

1. $\int_0^1 \exp(x)\,dx$. 2. $\int_0^1 f(x)\,dx$, where $f(x) = \begin{cases} 1, & \text{if } x > 0.3, \\ 0, & \text{else.} \end{cases}$

3. $\int_0^1 \sqrt{x}\,dx$. 4. $\int_{-1}^1 \left(\frac{23}{25}\cosh(x) - \cos(x)\right) dx$. 5. $\int_0^1 \frac{dx}{x^4 + x^2 + 0.9}$.

6. $\int_0^1 \sqrt{x^3}\,dx$. 7. $\int_0^1 \frac{1}{\sqrt{x}}\,dx$. 8. $\int_0^1 \frac{1}{x^4}\,dx$. 9. $\int_0^1 \frac{2}{2 + \sin(10\pi x)}\,dx$.

10. $\int_0^1 \frac{1}{1+x}\,dx$. 11. $\int_0^1 \frac{1}{1+e^x}\,dx$. 12. $\int_0^1 \frac{x}{e^x - 1}\,dx$.

13. $\int_{0,1}^1 \sin\left(\frac{100\pi x}{\pi x}\right) dx$. 14. $\int_0^{10} \frac{\sqrt{50}}{\exp(50\pi x^2)}\,dx$. 15. $\int_0^{10} \frac{25}{\exp(25x)}\,dx$.

16. $\int_0^{10} \frac{50}{\pi(2500x^2 + 1)}\,dx$. 17. $\int_{0,01}^1 \frac{50\sin(50\pi x)}{(50\pi x)^2}\,dx$.

18. $\int_0^{10} \cos\big(\cos(x) + 3\sin(x) + 2\cos(2x) + 3\sin(2x) + 3\cos(3x)\big)\,dx$.

19. $\int_0^1 f(x)\,dx$, $f(x) = \begin{cases} \log x, & \text{if } x > 10^{-15}, \\ 0, & \text{else.} \end{cases}$ 20. $\int_{-1}^1 \frac{1}{1.005 + x^2}\,dx$.

21. $\int_0^1 \sum_{i=1}^3 \frac{1}{\cosh(20^i(x - 21))}\,dx$. 22. $\int_0^1 4\pi^2 x \sin(20\pi x)\cos((2\pi x))\,dx$.

23. $\int_0^1 \frac{1}{1 + (230x - 30)^2}\,dx$.

We compared our results with results of MATLAB 7 function *quadl* [9] and with results of procedure *coteda* [3]. The procedures *quadl* and *coteda* as *ncg*9 and *hermit*5 have degree of precision nine. The function *quadl* uses

the same stopping criterion (1.5) and is based on a four-point Gauss-Lobatto rule with a three-point Kronrod extension. The second Kronrod extension of applied to the initial interval $[a, b]$ provides the estimate of is in formula (1.5) (procedures *ncg*9 and *hermit*5 also use this formula for rough estimate of the modulus of integral).

The procedure *coteda* uses both the five-point closed Newton-Cotes rule and the closed nine-point Newton-Cotes rule in a locally doubly adaptive fashion based on bisection. The code is allowed to stop either because the five-point estimate is considered good enough or because the nine-point estimate is considered good enough. The nine-point estimate requires four new points to the five-point estimate, but these are the same points needed in two applications of five-point rule after bisection of the the interval [3]. The procedure *coteda* uses a sequence of null rules of decreasing degrees [2].

We have tested both new procedures *ncg*9 and *hermit*5, as in [3] for twelve different error tolerances $\varepsilon = 10^{-1}, 10^{-2} \ldots, 10^{-12}$. Some results of calculations are presented in the following tables. Here we report the number of function evaluations used by each of the four codes. The results of column *coteda* are taken from [3] and all results of the other columns are calculated. A minus sign in front of number indicates that the requested accuracy was not achieved by the given code in this particular case.

Table 7. Test problem 3.

ε	*ncg*9	*herm*5	*quadl*	*coteda*
10^{-1}	9	10	18	9
10^{-2}	9	34	18	21
10^{-4}	41	66	48	41
10^{-6}	73	106	78	73
10^{-8}	121	162	168	113
10^{-10}	185	242	348	193
10^{-11}	265	330	438	241
10^{-12}	329	410	588	297

Table 8. Test problem 4.

ε	*ncg*9	*herm*5	*quadl*	*coteda*
10^{-1}	9	10	18	9
10^{-2}	9	10	18	9
10^{-4}	9	10	18	9
10^{-6}	9	10	18	9
10^{-8}	17	18	18	17
10^{-10}	33	34	48	33
10^{-11}	33	50	48	33
10^{-12}	49	50	48	65

Table 9. Test problem 13.

ε	*ncg*9	*herm*5	*quadl*	*coteda*
10^{-1}	-17	250	-18	513
10^{-2}	257	306	-78	513
10^{-4}	505	530	738	961
10^{-6}	969	1018	1218	1089
10^{-8}	1233	1938	3138	2017
10^{-10}	2041	2962	5718	3273
10^{-11}	3129	3970	6978	3977
10^{-12}	4025	4298	-10068	4081

Table 10. Test problem 17.

ε	*ncg*9	*herm*5	*quadl*	*coteda*
10^{-1}	33	66	18	41
10^{-2}	65	146	-48	265
10^{-4}	-193	362	-138	-297
10^{-6}	-321	674	618	905
10^{-8}	1025	954	1068	1185
10^{-10}	1529	2090	1998	2017
10^{-11}	2162	2506	2958	2273
10^{-12}	2537	3458	3888	2849

Table 11. Test problem 21.

ε	*ncg*9	*herm*5	*quadl*	*coteda*
10^{-1}	33	18	18	49
10^{-2}	48	-18	48	77
10^{-3}	73	74	78	-93
10^{-4}	-81	106	168	-129
10^{-5}	-105	-106	-228	-161
10^{-6}	-129	-186	-228	-193
10^{-7}	-185	-226	-348	381
10^{-8}	-233	-306	-438	461
10^{-9}	465	-402	888	581
10^{-10}	577	-434	1248	737
10^{-11}	729	946	1608	913
10^{-12}	953	1146	2268	1117

Table 12. Test problem 22.

ε	*ncg*9	*herm*5	*quadl*	*coteda*
10^{-1}	65	90	168	129
10^{-2}	105	122	168	129
10^{-3}	121	136	228	177
10^{-4}	145	242	408	257
10^{-5}	257	258	648	257
10^{-6}	257	354	798	289
10^{-7}	369	498	1038	465
10^{-8}	505	506	1248	505
10^{-9}	505	842	2208	529
10^{-10}	873	1002	3018	833
10^{-11}	1009	1122	3978	985
10^{-12}	1169	1866	5418	1033

The following observations can be made.

1. The results *ncg*9 and *coteda* are very similar in both reliability and efficiency. It can be explained that the code *ncg*8 uses the closed nine-point Newton-Cotes formula for integral approximation and the same strong 7-th degree of exactness null rule for locally error evaluation. The code *coteda*, as it was already mentioned, uses both closed five-point and nine-point Newton-Cotes formulas for integral approximation and a sequence of null rules of decreasing degrees for locally error evaluation. The code *coteda* in contrast to code *ncg*9 uses the locally doubly adaptive strategy [3]. It is known that usage of the the null rules of lower degrees usually requires computation of larger number of integrand functions. But a doubly adaptive strategy usually requires computation of less values of integrand function than a single adaptive strategy. Therefore only in 99 cases from all $23 \times 12 = 276$ cases code *ncg*9 required a smaller number of computations of integrand functions than code *coteda*.

2. Only for four tests: test 3 (the singular problem) for tolerance 10^{-1}, test 13 (the nonlinear oscillatory problem) for tolerance 10^{-1}, test 17 (the nonlinear oscillatory problem) for tolerance 10^{-6}, and test 21 (three peaks problem; peak in the vicinity $x = 0.6$ is very narrow) for tolerances 10^{-7} and 10^{-8} code *coteda* gives better results than *ncg*9. It can be explained that for this difficult tests the 7-th degree of exactness null rule is too "optimistic". The error estimation procedure based on sequence of null rules of decreasing degrees presents more exact results for these cases.

3. The codes *hermit*5 and *quadl* based on the nine degree Hermite and Gauss-Lobatto rule with a three-point Kronrod extension respectively in all cases demonstrate worse results in efficiency than codes *ncg*9 and *coteda*. It can be explained that in the procedures *hermit*5 and *quadl* integration step is larger than in codes *ncg*9 and *coteda*.

4. In terms of reliability the procedures *ncg*9, *hermit*5 and *coteda* demonstrate very close results. The reliability of code *quadl* is slightly worse than reliability of other discussed procedures.

6. Conclusions

In automatic quadrature algorithms the estimate of the truncation error is a very important step: it governs the decision on whether to return the current approximation and terminate or to continue calculations. Both the efficiency and the reliability therefore depend heavily on the error estimating procedure. Consequently the coefficients of the highest degree null rules with the smallest errors based on the same set of the points for 3, 5, 7, 9, and 11 point Newton-Cotes quadrature formulas are calculated in this paper. The coefficients of the 2, 3, 4, 5, 6, and 7 point Hermite quadrature formulas and the coefficients of their error estimation formulas are calculated. The calculated Hermitian error estimation formulas have the highest degree and the smallest error also.

In view of these (admittedly limited) test results we can conclude that both codes *cteda* and *ncg*9 show the best and very closed results in both reliability and efficiency. We can conclude:

1) The closed nine-point Newton-Cotes formula puts up better performance than the same degree Gauss-Lobatto formula;

2) Reliability of both error estimation formulas (based on one null rule and on a sequence null rules) are the same for the tolerances $\leq 10^{-4}$;

3) For the tolerances $\geq 10^{-3}$ the error estimation formula of procedure *ncg*9 gives less reliable results than the error estimation formula of code *cteda*. Therefore the error estimation formulas, presented in this paper, can be used for approximation of definite integrals, in particular, if the required tolerance $\leq 10^{-4}$. Efficiency of code *cteda* can be explained that it uses double adaptive integration procedure.

The procedure *hermit*5 shows results of reliability better than results of the same degree procedure *ncg*9. But efficiency of code *hermit*5 is worse than of procedure *ncg*9. Therefore, Hermitian quadrature can be used for approximated of integrals if calculation of the values of derivates of integrand function is not very complicated.

References

[1] N.S. Bakhvalov. *Numerical Methods*, volume 1. Moscow. (In Russian)

[2] J. Berntsen and T.O. Espelid. Error estimation in automatic quadrature routines. *ACM Transactions on Mathematical Software (TOMS)*, **17**(2), 233–252, 1991.

[3] T.O. Espelid. Doubly adaptive quadrature routines based on Newton-Cotes rules. *BIT Numerical Mathematics*, **43**, 319–337, 2003.

[4] T.O. Espelid. A test of QUADPACK and four double adaptive quadrature routines. *Reports in Informatics*, **281**, 1–22, 2004.

[5] T.O. Espelid and T. Sorevik. A discussion of a new error estimate for adaptive quadrature. *BIT Numerical Mathematics*, **29**(2), 283–294, 1989.

[6] P. Favati, G. Fiorentino, G. Lotti and F. Romani. Local error estimates and regularity tests for the implementation of double adaptive quadrature. *ACM Transactions on Mathematical Software (TOMS)*, **23**(1), 16–31, 1997.

[7] P. Favati, G. Lotti and F. Romani. Algorithm 691; improving QUADPACK automatic integration routines. *ACM Transactions on Mathematical Software (TOMS)*, **17**(2), 218–232, 1991.

[8] P. Favati, G. Lotti and F. Romani. Interpolatory integration formulas for optimal composition. *ACM Transactions on Mathematical Software (TOMS)*, **17**(2), 207–217, 1991.

[9] W. Gander and W. Gautschi. Adaptive quadrature-revisited. *BIT*, **40**, 84–101, 2000.

[10] S. Gariba, L. Quartapelle and G. Reina. Algorithm 36-SNIFF: Efficient self-tuning algorithm for numerical integration. *Computing*, **20**, 363–375, 1978.

[11] D.K. Kahaner. Comparison of numerical quadrature formulas. In: J. Rice(Ed.), *Mathematical Software*, New-York, Academic Press, 229–259, 1971.

[12] B. Kvedaras and M. Sapagovas. *Numerical methods*. Mintis, Vilnius, 1974. (In Lithuanian)

[13] D.P. Laurie. Sharper error estimates in adaptive quadrature. *BIT*, **23**, 258–261, 1983.

[14] D.P. Laurie. Anti-Gaussian quadrature formulas. *Mathematics of Computation*, **65**(214), 739–747, 1996.

[15] J.N. Lyness. Symmetric integration rules for hypercubes III. construction of integration rules using null rules. *Math. Comput.*, **19**, 625–637, 1965.

[16] W. M. McKeeman. Certification of algorithm 145. adaptive numerical integration by Simpson's rule. *Commun. ACM*, **6**, 167–168, 1963.

[17] Ch. W. Ueberhuber. *Numerical Computation 2: Methods, Software, and Analysis*. Springer-Verlag Berlin Heidelberg New York, 1997.