Practical error estimation in adaptive multidimensional quadrature routines

Jarle BERNTSEN *
Department of Informatics, University of Bergen, 5000 Bergen, Norway

Received 25 April 1988

Abstract: In this paper we discuss various ways of improving the reliability of adaptive multidimensional quadrature routines. We discuss the reasons that estimates of the error over a given region of integration may fail in representing an upper bound for the actual error. On basis of this discussion we suggest different actions to be taken in order to improve the reliability of the quadrature routine. In the numerical experiments we study how these new error estimates work in practice.

Keywords: Automatic multidimensional quadrature, error estimation.

1. Introduction

There are three main ingredients in adaptive quadrature routines:
(1) The algorithm for dividing the region of integration into subregions.
(2) A basic rule for estimating the integral over each subregion.
(3) A procedure for estimating the error over each subregion.

The algorithms used in multidimensional adaptive quadrature are very similar to the corresponding one-dimensional algorithms, see [6]. Basic rules for different regions of integration in different dimensions have also been studied by several authors, see [6,15].

In adaptive quadrature routines the decision on whether a given region of integration should be further divided is taken on basis of the size of the error estimate over the region. Both the efficiency and the reliability of a quadrature routine therefore depend heavily on the procedure for estimating errors. In the last edition of Davis and Rabinowitz’s book on numerical integration [6] there is a section on practical error estimation containing references to several papers on this topic. However, it is this author’s opinion that error estimating procedures deserve more investigation because of their importance in adaptive quadrature routines. In particular this is the case for multidimensional quadrature routines.

Usually the error over a given subregion is estimated by applying two different quadrature rules over the region. The absolute value of the difference in the results given by the two rules is then chosen to represent the error in the assumed best result. Our experience is that at least the

* Supported by The Norwegian Research Council for Sciences and Humanities and Statoil, Norway.
Present address: EDB-senteret, University of Bergen, 5007 Bergen, Norway.
multidimensional quadrature routines that apply this error estimating procedure may be very unreliable, see [8,4], and we will in this paper discuss various ways of improving the reliability.

In Section 2 we discuss how to scale different error estimates by using the weights of null rules, see Lyness [11]. In Section 3 we discuss why error estimates may be unreliable, and in Sections 4, 5 and 6 we consider different precautions that one may take in order to increase the reliability. In Section 7 our numerical experiments are described.

2. On the scaling of null rules

The quadrature rules used in both one- and multi-dimensional quadrature routines all have the form

\[
I[f] = \int_{\mathcal{R}} f(x) \, dx \approx \sum_{j=1}^{m} w_j f(x_j) \equiv Q[f]
\]

where \( \mathcal{R} \) is the region of integration, \( x_j \) are the evaluation points and \( w_j \) the corresponding weights, \( j = 1, \ldots, m \). We are interested in producing estimates of the error

\[
E[f] = I[f] - Q[f].
\]

Let \( Q_m \) be a quadrature rule of polynomial degree \( m \). Normally an estimate for \( E[f] \) is produced by applying two quadrature rules \( Q_{m_1} \) and \( Q_{m_2} \), \( m_1 > m_2 \), over the actual region, and an estimate of the error in the approximation given by \( Q_{m_1} \) is given by

\[
E_{m_1}[f] = \left| Q_{m_1}[f] - Q_{m_2}[f] \right|.
\]

When discussing how to estimate errors, we find it appropriate to introduce the concept null rules, see Lyness [11].

\[
N_{m_2} = Q_{m_1} - Q_{m_2}, \quad m_1 > m_2.
\]

\( N_{m_2} \) defined by (4) will be a null rule of degree \( m_2 \). If we apply \( N_{m_2} \) on any polynomial of degree less or equal to \( m_2 \), the result will be zero. The error estimate (3) may now be replaced by

\[
E_{m_1}[f] = \left| N_{m_2}[f] \right|.
\]

Very often the error estimate (3) in adaptive quadrature routines is scaled by some factor \( \lambda \) in order to balance reasonably between efficiency and reliability. This is the same as scaling the weights of the null rule \( N_{m_2} \) by \( \lambda \). Now define \( Q_{m_2}(\lambda) \) by, see [5],

\[
Q_{m_2}(\lambda) = \lambda Q_{m_2} + (1 - \lambda) Q_{m_1}
\]

which also is a quadrature rule of degree \( m_2 \) for \( \lambda \neq 0 \). Scaling (3) or (5) by \( \lambda \) will now be the same as using the error estimate

\[
E_{m_1}[f] = \left| Q_{m_1}[f] - Q_{m_2}(\lambda)[f] \right|.
\]

To choose a reasonable \( \lambda \) is therefore the same as choosing a reasonable comparison rule \( Q_{m_2}(\lambda) \).

The reason that we focus on this is that \( Q_{m_2} \) usually is computed as a solution of some nonlinear system of equation, the constraint on \( Q_{m_2} \) only being that it shall have degree \( m_2 \). Very often \( Q_{m_2} \) is computed by removing one point from the set of evaluation points used by
Q_{m1}, and we have several choices when deciding which point to remove. As shown in [5],
different choices may give us different members of a family of rules of the type (6) and therefore
null rules of different strengths. By taking the first computed Q_{m2}, we are therefore in a way
choosing the scaling of the null rule N_{m2} blindly.

Therefore, after having produced some Q_{m2} and the corresponding N_{m2}, we should try to
produce a reasonable λ such that the null rule

\[ N_{m2}(\lambda) = \lambda(Q_{m1} - Q_{m2}) \]

will provide reasonably strong error estimates

\[ E_{m1}[f] = |N_{m2}(\lambda)[f]|. \]

In particular, it is important that all error estimates have approximately the same strength when
we want to look at several error estimates of the form (9) in order to provide a final error
estimate. There is as far as we can see no obvious way of achieving this, see [5], but one practical
way is to scale all used null rules in such a way that the sum of the absolute values of their
weights is equal to some constant. In the experiments to be described we have thus scaled all null
rules such that

\[ \sum_{j=1}^{n} |v_j| = 2^{DIM} \]

where v_j, j = 1, \ldots, n, are the weights of the null rule after scaling, DIM the dimension of the
integral and [-1, 1]^DIM the region of integration.

3. Some reasons that error estimates may fail

If the error estimates over all subregions that are processed in an automatic quadrature
routine, satisfy the inequality

\[ E[f] \geq |I[f] - Q[f]|, \]

the quadrature routine will be 100 per cent reliable. We will therefore in this section discuss the
reasons that error estimates given by expressions like (3) may fail to satisfy (11). When these
reasons are established, we will start to look for remedies that may prevent (11) from failing.

The assumption behind the expression (3), see De Boor [7], is that Q_{m1}[f] is a much better
approximation to I[f] than Q_{m2}[f], m1 > m2, and therefore

\[ E_{m1}[f] = |Q_{m1}[f] - Q_{m2}[f]| - |I[f] - Q_{m2}[f]| \gg |I[f] - Q_{m1}[f]|. \]

If (12) is satisfied, then certainly (11) should be satisfied. The error expansion for the rule Q_m
will in Haber’s notation, see [10], look like

\[ I[f] - Q_m[f] = h^{m+2} \sum_{|i|=m+1} c_i f^{(i)}(\xi_i) \]

where i = (i_1, \ldots, i_{DIM}) and |i| = i_1 + \cdots + i_{DIM} and the region of integration is [-h, h]^DIM. f
is assumed to be analytical. The c_i’s are constants specific to Q_m. The \xi_i’s are vectors of
dimension DIM that depend on f and Q_m.
We will draw attention to three important reasons that error estimates of the form (3) may fail to satisfy (11), and the expression (13) is important in this discussion. The three reasons are not disjunct, but we find it useful to distinguish between them when considering remedies for improving the reliability.

Reason 1. Few or badly placed evaluation points

The problem of providing reliable error estimates is most pronounced when there are one or more difficult spots within the region of integration. This difficult spot may be a peak, a discontinuity or a singular point. If there are too few evaluation points or the evaluation points are badly placed in relationship to this difficult spot, the error may be estimated to almost zero in cases where the actual error is very large. In Section 4 we discuss what we can do in order to avoid this problem.

Reason 2. Nonasymptotic \( h \)

From (13) we see that for \( h \) small enough we may expect (11) to be satisfied. The main problem in automatic quadrature routines is that we assume (11) to be satisfied also for large values of \( h \). Very often we may notice in error estimates of the form (3) that there is some difficulty within the region, but the actual error will often be underestimated. In Section 5 we describe a procedure for testing whether we are in the asymptotic region of \( h \) or not.

Reason 3. “Phase factors”

For small enough \( h \), that is after enough subdivisions of the original region of integration, (11) will be satisfied most of the time. However, as we shall see in the following there are no guarantees that this is the case. Even if we are in the asymptotic range, it may happen that (11) fails. Lyness and Kaganove [12] describe how such events may occur. They apply two different quadrature rules to a function that varies with some parameter and plot the errors of the estimates delivered by the two rules as a function of this parameter. The problem arises when the distance between the two curves for some value of the parameter becomes less than the actual error in the best approximation because of “phase factors”, see [12, p. 76]. In [2] this author shows that this may be a problem also in multidimensional adaptive quadrature routines. If we look at the expression (13), we notice that if the \( f^{(u)} \)'s are oscillatory and the \( \xi_i \)'s are chosen carefully, the error in \( Q_{m_2}[f] \) may be smaller than the error in \( Q_{m_1}[f] \) even if \( m_2 < m_1 \) and \( h \) is small. Of course the problem with “phase factors” is present also for large values of \( h \), but they are the explanation for the phenomena that error estimates may be unreliable also for \( h \) in the asymptotic range. In Section 6 we discuss what to do in order to avoid the problem with “phase factors”.

4. Using more of the available information in the error estimation

In the previous section we argued that if there are no evaluation points close enough to some difficult spot, no expression of the form (3) will be able to detect the difficulty. This will always be a problem in adaptive quadrature routines. However, there are two precautions that one may take in order to reduce this problem.
The first one is connected to the distribution of evaluation points within the region of integration. When constructing quadrature rules, the quality measure for the rules most commonly used in adaptive quadrature routines, is the polynomial degree of exactness. Quadrature rules of high polynomial degree, however, very often tend to gather all evaluation points close to the edges of the region of integration and thereby leave large empty (or nearly empty) spaces around the center of the region. In earlier experiments, see [2], we have noticed the effects of this, but a systematic study of this phenomena remains to be done. However, we regard such a study to be a separate topic, and in this paper we concentrate on what we can do with the evaluation points at hand for any given quadrature rule.

In global adaptive quadrature algorithms based on bisection of subregions, we build up binary trees of subregions as the computation proceeds. The final estimates of the integral and the error are based on the corresponding estimates from the subregions represented by the leaves of such a binary tree. Going from one level in the tree to the next, all information from the “old” function evaluations are very often forgotten. At least this is the case for the QUADPACK, see [13], routine QAG and multidimensional ADAPT-like routines, see [16,9]. The result of this may be that a problem spot discovered through a large error estimate on one level of the tree, may be “lost” again on the next level. On the next level there are twice as many points, but their position relative the difficult spot may be unfortunate.

In order to avoid this problem at least to some extent one may use two-level error estimates. Let $Q_i$ be a quadrature approximation over a given region and $Q^{(i)}_1$, $i = 1, 2$, two approximations over the two subregions we get by bisecting the first. We then get a two-level error estimate by the expression

$$E_1 = |Q_1 - (Q^{(1)}_1 + Q^{(2)}_2)|.$$  

(14)

This error estimate we will combine with the local error estimates $E^{(i)}_2$ and $E^{(2)}_2$ in order to produce the final estimated error over each of the two new regions. We may for instance add to each local error estimate a piece of $E_1$ that is proportional to the original local error estimate. 

$$E^{(i)}_2 = E^{(i)}_2 + \frac{E^{(i)}_2}{(E^{(1)}_2 + E^{(2)}_2)} E_1, \quad i = 1, 2.$$  

(15)

The weakness with this approach is that if for instance $E^{(1)}_2$ by mistake is estimated to zero, the expression (15) will not help. Another approach will therefore be to share the two level error estimate $E_1$ equally between the two new regions

$$E^{(i)}_2 = E^{(i)}_2 + \frac{1}{2} E_1, \quad i = 1, 2.$$  

(16)

By using (16), the information that we have a problem spot, will be transferred from one level of the tree of subregions to the next. The problem is that maybe the problem spot is only one of the two new subregions, and it may then be a mistake to inforce the original local error estimate through (16) in both. Such mistakes may cause a lot of unnecessary subdivisions. In the experiments to be described in Section 7, we have tried to combine the two expressions (15) and (16). The inforcement of the local error estimates will result in a more reliable routine, but also in a more expensive one. We have to use more function evaluations before the routine reports that our accuracy request is satisfied. In the error estimating procedure used in the numerical
experiments we have therefore introduced two heuristics \( c_1 \) and \( c_2 \). We may then try to select \( c_1 \) and \( c_2 \) in such a way that the cost of increasing the reliability will be reasonable.

\[
E_2^{(i)} = E_2^{(i)} + c_1 \frac{E_2^{(i)}}{E_2^{(1)} + E_2^{(2)}} E_1 + c_2 \frac{E_1}{2}, \quad i = 1, 2.
\]

(17)

Procedures similar to the procedures above have also been used by Sørevik [14]. In [14] these procedures were also used in combination with the procedure for estimating the local error

\[
E_{\text{scaled}}[f] = \frac{c E[f]}{\hat{Q}[f]}^\alpha, \quad c > 1, \quad \alpha > 1
\]

(18)

where \( E[f] \) is first computed by an expression similar to (3). \( \hat{Q}[f] = Q[|f - \tilde{f}|] \) and \( \tilde{f} = q[f]/\text{volume} \) for some quadrature rule \( Q \). This procedure is introduced in QUADPACK with \( c = 200 \) and \( \alpha = 1.5 \), and has proved to produce very reliable adaptive one-dimensional quadrature routines. However, the results from [14] indicate that it will be very expensive to use (18) as local error estimating procedure in adaptive multidimensional quadrature routines. In the next two sections we therefore introduce local error estimating procedures that we hope will be more cost efficient than (18) in multidimensional routines.

5. Checking the validity of the asymptotic assumption

The asymptotic assumption that error estimates like (3) are based upon is given by (12). If we have a sequence of quadrature rules \( Q_{2m-i}, i = -1, 1, 3, \ldots, 2m - 3 \), and the assumption that a higher degree approximation is better than a lower degree approximation is satisfied, then

\[
\frac{|I[f] - Q_{2m-i}[f]|}{|I[f] - Q_{2m-i-2}[f]|} \leq r_i < 1, \quad i = 1, 1, 3, \ldots, 2m - 3.
\]

(19)

If we knew \( I[f] \), we could by using (19) check the validity of the asymptotic assumption. The problem is of course that we do not know \( I[f] \), but we may replace \( I[f] \) in (19) with the presumptive best approximation \( Q_{2m+1}[f] \) and check whether the inequalities

\[
\frac{|Q_{2m+1}[f] - Q_{2m-1}[f]|}{|Q_{2m+1}[f] - Q_{2m-2}[f]|} \leq r_i < 1, \quad i = 1, 3, \ldots, 2m - 3
\]

(20)

are satisfied. In terms of the null rules \( N_{2m-i} = Q_{2m+1} - Q_{2m-i}, i = 1, 3, \ldots, 2m - 1 \), (20) may be written

\[
\frac{|N_{2m-i}[f]|}{|N_{2m-i-2}[f]|} \leq r_i < 1, \quad i = 1, 3, \ldots, 2m - 3.
\]

(21)

If the inequalities in (20) or (21) are not satisfied, we must choose some way of strengthening the local error estimates.

What we are doing in order to check the validity of the asymptotic assumption when going from (19) to (20), is to assume that the assumption in beforehand is satisfied. Therefore, the tests (20) are no rigid tests on whether the assumption (12) is satisfied, and it is necessary to take a closer look at what may happen when using tests like (20). If the asymptotic assumption is
satisfied, the tests (20) will be passed as long as the approximations $Q_{2m-i}$, $i = 3, 5, \ldots, 2m - 3$, are not computed to machine accuracy, and the local error estimates will not be strengthened. If the asymptotic assumption is not satisfied, we do not want the tests (20) to be passed. However, because the assumption $Q_{2m+1} \sim I[f]$ may be very wrong, the inequalities in (20) may still be satisfied. How often this may happen depends of course on the size of the $r_i$'s. The worst that can happen when using the inequalities (20) or (21) to check the validity of the asymptotic assumption, is therefore that the normally used error estimate (3) is used also in some cases where the assumption is not satisfied.

In the experiments to be described in Section 7, we use a degree 7 quadrature rule over each subregion and null rules of degree 5, 3 and 1 in tests like (21). $r_1$ and $r_3$ are chosen heuristically. If one of the tests fails, we strengthen the local error estimate by using the expression

$$E[f] = c_3 \cdot \max(\, |N_5[f]|, \, |N_3[f]|, \, |N_1[f]|) \cdot$$

(22)

Otherwise we choose $E[f] = |N_5[f]|$. The error estimate (22) including $c_3$ is of course another heuristic choice.

6. A precaution against the “phase factors”

In Section 3 we described how the “phase factors” may cause severe problems when estimating the error. In [2] we illustrated the phenomena by estimating the integral of $f_2(x)$, see Section 7, in three dimensions over the unit cube. $\alpha_1$, $\alpha_2$ and $\alpha_3$ were chosen equal to 90 and $\beta_2$ and $\beta_3$ equal to 1.0. $\beta_1$ is varied from 0.0 to 1.0. One degree 7 rule and two degree 5 rules are used and the errors in the approximations are plotted in Fig. 1 as functions of $\beta_1$.

We see that if we use one of the degree 5 rules as a rule of reference, there are two small regions in $\beta_1$ where an error estimate of the form (3) will not satisfy (11). If we use the other degree 5 rule, there is one greater region where (11) fails. However, the intersection of these regions is empty. This observation gave us the idea of introducing the error estimate

$$E[f] = \max(\, |N_{m,1}[f]|, \, |N_{m,2}[f]|) \cdot$$

(23)

where $N_{m,1}$ and $N_{m,2}$ are two linearly independent null rules of degree $m$. The constraint that the

![Fig. 1.](image-url)
rules must be linearly independent or not members of the same λ-family, see Section 2, is necessary to force curves like those in Fig. 1 out of phase.

This error estimate will satisfy the inequality (11) for all values of βi in the example above. We are aware that in other cases (11) may still fail even if we use the error estimate (23), but at least the probability that this happens will be considerably reduced.

We should also note that using a two level error estimate like (16) probably will remove some of the “phase factor” problems. On the other hand the error estimate (15) will be of little or no help.

When we in our experiments use (23) in combination with the tests (21), we first select

\[ N_5[f] = \max(|N_{5,1}[f]|, |N_{5,2}[f]|) \]

7. Numerical experiments

In [3] we have run experiments in 2, 4, 6, 8, 10 and 15 dimensions in order to study how the precautions introduced in Sections 4, 5 and 6 work in practice. The experiments were most extensive in 2 dimensions and the results from this dimension also gave best insight into how the precautions work. We will therefore present here some results from our 2-dimensional experiments. Each of the introduced remedies may be used alone or in any combination. The goal is to produce a quadrature routine with acceptable reliability. The number of integrand evaluations the routine needs before it reports that our accuracy request is met, must on the other hand not increase too much. There are several heuristics that we may play around with in order to balance reasonably between these two desires.

We use a quadrature rule of degree 7 to approximate the integral over each subregion. The quadrature rule used in two dimensions has the form

\[
Q_j[f] = w_1 \sum f(0,0) + w_2 \sum f(\alpha_1, 0) + w_3 \sum f(\alpha_2, 0) + w_4 \sum f(\alpha_3, 0) + w_5 \sum f(\beta, \beta) + w_6 \sum f(\lambda, \lambda)
\]

where all sums are fully symmetric sums over all permutations of coordinates, sign changes included. \( \alpha_1 \sim 0.4, \alpha_2 \sim 0.6, \alpha_3 \sim 1.0, \beta \sim 0.7 \) and \( \lambda \sim 1.0 \). The weights \( w_1, \ldots, w_6 \) are chosen such that the quadrature rule will be of degree 7. The above quadrature rule uses 21 integrand evaluations. This is more than strictly necessary in order to achieve a degree 7 quadrature rule, see for instance [16,9]. However, by introducing more fully symmetric sums, we may produce two linearly independent null rules of degree 5. We thus have two linearly independent null rules of degree 5, \( N_{5,1} \) and \( N_{5,2} \), one null rule of degree 3, \( N_3 \), and one null rule of degree 1, \( N_1 \), for estimating the errors. They are all scaled according to (10). In our experiments reported in [3] we have used 10 test families, see [9,3]. Here we report on the results for the following 3 families:

**Attributes:**

1. \( f_1(x) = \exp\left(- \sum_{i=1}^{n} \alpha_i |x_i - \beta_i|\right) \) – \( C_0 \) function
2. \( f_2(x) = \prod_{i=1}^{n} \left( \alpha_i x_i^2 + (x_i - \beta_i)^2 \right)^{-1} \) – product peak
3. \( f_3(x) = \cos\left(2\pi \beta_1 + \sum_{i=1}^{n} \alpha_i x_i \right) \) – oscillatory
The parameters \( \beta_i, i = 1, \ldots, n \), are picked randomly from \([0, 1]\). The \( \alpha \) vectors are also first picked randomly from \([0, 1]\) and then scaled according to

\[
n^n \sum_{i=1}^{n} \alpha_i = d_j.
\]

The parameters \( d_j \) controls, for constant \( n \), the difficulty of the integrands in family \( j \), see [8]. The exponents \( e_j \) are chosen to offset the natural increase with \( n \) of the difficulty in integrand \( j \).

In the experiments reported here we have chosen:

\[
d = (300., 600., 110.), \quad e = (2., 2., 1.5).
\]

The regions of integration are for all test families the unit square \([0, 1]^2\).

We report the results we get when using the following error estimating procedures.

**Procedure 1.** \( E_7[f] = |N_{5,1}[f]| \).

**Procedure 2 (see Section 6).** \( E_7[f] = \max(|N_{5,1}[f]|, |N_{5,2}[f]|) \).

**Procedure 3 (see Section 5).** If

\[
\frac{|N_{5,1}[f]|}{|N_3[f]|} \leq 0.5 \quad \text{and} \quad \frac{|N_5[f]|}{|N_1[f]|} \leq 0.5,
\]

then

\[
E_7[f] = |N_{5,1}[f]|.
\]

else

\[
E_7[f] = 10 \times \max(|N_{5,1}[f]|, |N_{5,2}[f]|, |N_5[f]|, |N_1[f]|).
\]

**Procedure 4 (see Section 4).** In each subdivision step let \( Q_{7,1}[f] \) be the quadrature approximation of the parent region and \( Q_{7,2}^{(i)}, i = 1, 2 \), the approximations over the children subregions.

Let \( E_{7,1}[f] \) be the two level error estimate computed by

\[
E_{7,1}[f] = |Q_{7,1}[f] - (Q_{7,2}^{(1)}[f] + Q_{7,2}^{(2)}[f])|.
\]

Let the error estimates over the children subregions \( E_{7,2}^{(i)}, i = 1, 2 \), first be computed by

\[
E_{7,2}^{(i)}[f] = |N_{5,1}[f]|, \quad i = 1, 2.
\]

The final error estimate over each new subregion will now be

\[
E_{7,2}^{(i)}[f] = E_{7,2}^{(i)}[f] + 0.5 \times \frac{E_{7,2}^{(i)}[f]}{(E_{7,2}^{(1)}[f] + E_{7,2}^{(2)}[f])} E_{7,1}[f] + 0.25 \times E_{7,1}[f], \quad i = 1, 2.
\]

**Procedure 5.** The same as Procedure 4 except that the errors over the children subregions first are computed by

\[
E_{7,2}^{(i)}[f] = \max(|N_{5,1}[f]|, |N_{5,2}[f]|), \quad i = 1, 2.
\]

For each \( i \) we then impose the test:

if

\[
\frac{|E_{7,2}^{(i)}[f]|}{|N_3[f]|} \leq 0.5 \quad \text{and} \quad \frac{|N_5[f]|}{|N_1[f]|} \leq 0.5,
\]
then $E^{(i)}_{7,2}[f]$ is kept unmodified, else

$$E^{(i)}_{7,2}[f] = 10 \times \max(E^{(i)}_{7,2}[f], |N_2[f]|, |N_1[f]|)$$

before the final estimates

$$E^{(i)}_{7,2}[f] = E^{(i)}_{7,2}[f] + 0.5 \times \frac{E^{(i)}_{7,2}[f]}{(E^{(i)}_{7,2}[f] + E^{(i)}_{7,2}[f])} E_{7,1}[f] + 0.25 \times E_{7,1}[f], \quad i = 1, 2,$$

are computed.

The error estimating Procedure 5 is a combination of the Procedures 2, 3 and 4.

We have used 200 samples for each test family in dimension 2. We have requested relative accuracies $10^{-1}, 10^{-2}, \ldots, 10^{-5}$, and allowed at maximum 300 000 integrand evaluations.

In the appendix we have listed the average numbers of function evaluations, $\nu_m$, where $m$ indicates the number of samples. We list the number of cases where the routine reports that our accuracy request is met, $N_1$, and in the parentheses the number of cases where the actual error is greater than the error request. We list the number of cases where the computation is stopped because the maximum number of integrand evaluations is reached, $N_2$, and in the parentheses the number of cases where the actual error is greater than the estimated error. Finally, we list the average number of digits in the computed approximations to the integrals.

In the main loop of a global adaptive quadrature algorithm the subregion with the greatest estimated error is usually divided in two equal parts. Each of the two new subregions is then processed. That is: approximations to the integral and the error are computed. Our numerical experiments are run on an Alliant FX/8 with 6 processors in double precision. In each step of the algorithm the 3 subregions with greatest estimated errors are divided in two equal parts along the direction with greatest fourth divided difference. The 6 resulting new subregions are then processed in parallel. Even if we by doing this have reduced the CPU time by approximately a factor 5.5, see [1], the experiments reported in [3] have taken 52 hours of CPU.

The results from our experiments in the higher dimensions are to a great extent supportive to the 2-dimensional results, and below we list the most important conclusions.

1. The usual error estimating procedure (3) produces on many problems very unreliable error estimates. This observation is in accordance with the results of previous experiments, see [8, 4, 5, 2, 14], and justifies our efforts on improving the reliability of multidimensional quadrature routines.

2. By adding different remedies to the error estimating Procedure 1, we are able to reduce the number of failures. That is the number of cases where the actual accuracy is greater than the required accuracy, or in the cases where the maximum allowed number of function evaluations have been used, the actual error is greater than the reported error. No remedy alone, at least with the heuristics chosen here, is able to give the routine an acceptable accuracy. However, very often different precautions remove different failures, and by combining them as we do in error estimating Procedure 5, we get a routine with in our opinion acceptable reliability.

3. There is of course a cost connected to this improved reliability. However, a very large failure percentage indicates that too little work has been spent on solving the problem and therefore we must expect this to happen. The increased cost seems also to depend on how bad the reliability was before our precautions were introduced. For the oscillatory problems the usual
error estimating procedure is reliable enough in most cases, and for these problems the error estimating Procedure 5 gives only a small additional cost. The error estimating Procedure 5 therefore seems to work as intended. The extra cost of introducing this error estimating procedure is greatest in those cases where it is most highly required.

(4) The final choice of heuristics in an adaptive routine will be a political decision. If too much emphasis is put on reliability, we may have the computers running 5 or 10 times longer than strictly necessary. On the other hand if we are too concerned about economy, we may frequently be misled by the delivered error estimates.

8. Conclusions

In earlier experiments with adaptive quadrature routines we have seen that to base the error estimation on the absolute difference between two quadrature approximations alone, may produce routines that are very unreliable for many quadrature problems. Especially this is the case for multidimensional automatic quadrature routines. In recent years we have therefore tried to build up an understanding of the reasons that error estimates may fail. These reasons have previously been described by other authors, but they have not had the impact on practical error estimating procedures that they in this author’s opinion should have had. Based on the acquired understanding we have introduced several precautions that may prevent error estimates from failing. Through the numerical experiments we have given numerical evidence that shows that these precautions also work in practise, and the extra cost of introducing the new error estimating procedure is greatest in those cases where it is most highly required to spend more work. We believe that the insight gained through the work on practical error estimation may be used to produce multidimensional adaptive quadrature routines with acceptable reliability.

Appendix. Two-dimensional experiments

Test family 1 — Procedure 1

<table>
<thead>
<tr>
<th>Error req.</th>
<th>$v_{200}$</th>
<th>N1</th>
<th>N2</th>
<th>Mean. dig.</th>
</tr>
</thead>
<tbody>
<tr>
<td>$10^{-1}$</td>
<td>578</td>
<td>200 (116)</td>
<td>0 (0)</td>
<td>0.99</td>
</tr>
<tr>
<td>$10^{-2}$</td>
<td>953</td>
<td>200 (129)</td>
<td>0 (0)</td>
<td>1.78</td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>1838</td>
<td>200 (154)</td>
<td>0 (0)</td>
<td>2.62</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>4197</td>
<td>200 (143)</td>
<td>0 (0)</td>
<td>3.71</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>8976</td>
<td>200 (138)</td>
<td>0 (0)</td>
<td>4.79</td>
</tr>
</tbody>
</table>
Test family 1—Procedure 2

<table>
<thead>
<tr>
<th>Error req.</th>
<th>$p_{200}$</th>
<th>N1</th>
<th>N2</th>
<th>Mean. dig.</th>
</tr>
</thead>
<tbody>
<tr>
<td>$10^{-1}$</td>
<td>655</td>
<td>200 (80)</td>
<td>0 (0)</td>
<td>1.21</td>
</tr>
<tr>
<td>$10^{-2}$</td>
<td>1111</td>
<td>200 (67)</td>
<td>0 (0)</td>
<td>2.23</td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>2366</td>
<td>200 (51)</td>
<td>0 (0)</td>
<td>3.32</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>5099</td>
<td>200 (25)</td>
<td>0 (0)</td>
<td>4.50</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>10485</td>
<td>200 (8)</td>
<td>0 (0)</td>
<td>5.60</td>
</tr>
</tbody>
</table>

Test family 1—Procedure 3

<table>
<thead>
<tr>
<th>Error req.</th>
<th>$p_{200}$</th>
<th>N1</th>
<th>N2</th>
<th>Mean. dig.</th>
</tr>
</thead>
<tbody>
<tr>
<td>$10^{-1}$</td>
<td>1149</td>
<td>200 (15)</td>
<td>0 (0)</td>
<td>1.81</td>
</tr>
<tr>
<td>$10^{-2}$</td>
<td>1726</td>
<td>200 (69)</td>
<td>0 (0)</td>
<td>2.27</td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>2884</td>
<td>200 (119)</td>
<td>0 (0)</td>
<td>2.95</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>5354</td>
<td>200 (111)</td>
<td>0 (0)</td>
<td>3.95</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>10105</td>
<td>200 (110)</td>
<td>0 (0)</td>
<td>5.00</td>
</tr>
</tbody>
</table>

Test family 1—Procedure 4

<table>
<thead>
<tr>
<th>Error req.</th>
<th>$p_{200}$</th>
<th>N1</th>
<th>N2</th>
<th>Mean. dig.</th>
</tr>
</thead>
<tbody>
<tr>
<td>$10^{-1}$</td>
<td>899</td>
<td>200 (18)</td>
<td>0 (0)</td>
<td>1.73</td>
</tr>
<tr>
<td>$10^{-2}$</td>
<td>1847</td>
<td>200 (37)</td>
<td>0 (0)</td>
<td>2.54</td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>4065</td>
<td>200 (41)</td>
<td>0 (0)</td>
<td>3.52</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>8205</td>
<td>200 (31)</td>
<td>0 (0)</td>
<td>4.54</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>15543</td>
<td>200 (14)</td>
<td>0 (0)</td>
<td>5.57</td>
</tr>
</tbody>
</table>

Test family 1—Procedure 5

<table>
<thead>
<tr>
<th>Error req.</th>
<th>$p_{200}$</th>
<th>N1</th>
<th>N2</th>
<th>Mean. dig.</th>
</tr>
</thead>
<tbody>
<tr>
<td>$10^{-1}$</td>
<td>1484</td>
<td>200 (1)</td>
<td>0 (0)</td>
<td>2.30</td>
</tr>
<tr>
<td>$10^{-2}$</td>
<td>2842</td>
<td>200 (0)</td>
<td>0 (0)</td>
<td>3.18</td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>5468</td>
<td>200 (0)</td>
<td>0 (0)</td>
<td>4.20</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>9925</td>
<td>200 (0)</td>
<td>0 (0)</td>
<td>5.19</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>17680</td>
<td>200 (0)</td>
<td>0 (0)</td>
<td>6.15</td>
</tr>
</tbody>
</table>

Test family 2—Procedure 1

<table>
<thead>
<tr>
<th>Error req.</th>
<th>$p_{200}$</th>
<th>N1</th>
<th>N2</th>
<th>Mean. dig.</th>
</tr>
</thead>
<tbody>
<tr>
<td>$10^{-1}$</td>
<td>653</td>
<td>200 (120)</td>
<td>0 (0)</td>
<td>0.91</td>
</tr>
<tr>
<td>$10^{-2}$</td>
<td>1142</td>
<td>200 (140)</td>
<td>0 (0)</td>
<td>1.76</td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>2371</td>
<td>200 (131)</td>
<td>0 (0)</td>
<td>2.79</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>5244</td>
<td>200 (44)</td>
<td>0 (0)</td>
<td>4.47</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>11191</td>
<td>200 (11)</td>
<td>0 (0)</td>
<td>6.04</td>
</tr>
</tbody>
</table>

Test family 2—Procedure 2

<table>
<thead>
<tr>
<th>Error req.</th>
<th>$p_{200}$</th>
<th>N1</th>
<th>N2</th>
<th>Mean. dig.</th>
</tr>
</thead>
<tbody>
<tr>
<td>$10^{-1}$</td>
<td>776</td>
<td>200 (83)</td>
<td>0 (0)</td>
<td>1.17</td>
</tr>
<tr>
<td>$10^{-2}$</td>
<td>1381</td>
<td>200 (60)</td>
<td>0 (0)</td>
<td>2.32</td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>2826</td>
<td>200 (11)</td>
<td>0 (0)</td>
<td>3.94</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>5796</td>
<td>200 (1)</td>
<td>0 (0)</td>
<td>5.49</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>12159</td>
<td>200 (0)</td>
<td>0 (0)</td>
<td>6.78</td>
</tr>
</tbody>
</table>
### Test family 2 — Procedure 3

<table>
<thead>
<tr>
<th>Error req.</th>
<th>$p_{200}$</th>
<th>N1</th>
<th>N2</th>
<th>Mean. dig.</th>
</tr>
</thead>
<tbody>
<tr>
<td>$10^{-1}$</td>
<td>1830</td>
<td>200 (18)</td>
<td>0 (0)</td>
<td>2.10</td>
</tr>
<tr>
<td>$10^{-2}$</td>
<td>2623</td>
<td>200 (39)</td>
<td>0 (0)</td>
<td>2.66</td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>3650</td>
<td>200 (47)</td>
<td>0 (0)</td>
<td>3.62</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>5966</td>
<td>200 (16)</td>
<td>0 (0)</td>
<td>4.82</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>11721</td>
<td>200 (6)</td>
<td>0 (0)</td>
<td>6.25</td>
</tr>
</tbody>
</table>

### Test family 2 — Procedure 4

<table>
<thead>
<tr>
<th>Error req.</th>
<th>$p_{200}$</th>
<th>N1</th>
<th>N2</th>
<th>Mean. dig.</th>
</tr>
</thead>
<tbody>
<tr>
<td>$10^{-1}$</td>
<td>1146</td>
<td>200 (27)</td>
<td>0 (0)</td>
<td>1.78</td>
</tr>
<tr>
<td>$10^{-2}$</td>
<td>2422</td>
<td>200 (31)</td>
<td>0 (0)</td>
<td>2.89</td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>4272</td>
<td>200 (12)</td>
<td>0 (0)</td>
<td>4.28</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>7548</td>
<td>200 (0)</td>
<td>0 (0)</td>
<td>5.43</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>14118</td>
<td>200 (0)</td>
<td>0 (0)</td>
<td>6.63</td>
</tr>
</tbody>
</table>

### Test family 2 — Procedure 5

<table>
<thead>
<tr>
<th>Error req.</th>
<th>$p_{200}$</th>
<th>N1</th>
<th>N2</th>
<th>Mean. dig.</th>
</tr>
</thead>
<tbody>
<tr>
<td>$10^{-1}$</td>
<td>2274</td>
<td>200 (1)</td>
<td>0 (0)</td>
<td>2.86</td>
</tr>
<tr>
<td>$10^{-2}$</td>
<td>3545</td>
<td>200 (0)</td>
<td>0 (0)</td>
<td>3.58</td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>5218</td>
<td>200 (0)</td>
<td>0 (0)</td>
<td>4.70</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>8367</td>
<td>200 (0)</td>
<td>0 (0)</td>
<td>5.89</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>15227</td>
<td>200 (0)</td>
<td>0 (0)</td>
<td>7.11</td>
</tr>
</tbody>
</table>

### Test family 3 — Procedure 1

<table>
<thead>
<tr>
<th>Error req.</th>
<th>$p_{200}$</th>
<th>N1</th>
<th>N2</th>
<th>Mean. dig.</th>
</tr>
</thead>
<tbody>
<tr>
<td>$10^{-1}$</td>
<td>5937</td>
<td>200 (0)</td>
<td>0 (0)</td>
<td>4.71</td>
</tr>
<tr>
<td>$10^{-2}$</td>
<td>13142</td>
<td>200 (0)</td>
<td>0 (0)</td>
<td>6.27</td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>28545</td>
<td>200 (0)</td>
<td>0 (0)</td>
<td>7.83</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>61940</td>
<td>198 (0)</td>
<td>2 (0)</td>
<td>9.42</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>124730</td>
<td>187 (0)</td>
<td>13 (0)</td>
<td>10.90</td>
</tr>
</tbody>
</table>

### Test family 3 — Procedure 2

<table>
<thead>
<tr>
<th>Error req.</th>
<th>$p_{200}$</th>
<th>N1</th>
<th>N2</th>
<th>Mean. dig.</th>
</tr>
</thead>
<tbody>
<tr>
<td>$10^{-1}$</td>
<td>5952</td>
<td>200 (0)</td>
<td>0 (0)</td>
<td>4.78</td>
</tr>
<tr>
<td>$10^{-2}$</td>
<td>13149</td>
<td>200 (0)</td>
<td>0 (0)</td>
<td>6.31</td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>28559</td>
<td>200 (0)</td>
<td>0 (0)</td>
<td>7.82</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>61955</td>
<td>198 (0)</td>
<td>2 (0)</td>
<td>9.47</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>124745</td>
<td>187 (0)</td>
<td>13 (0)</td>
<td>10.91</td>
</tr>
</tbody>
</table>

### Test family 3 — Procedure 3

<table>
<thead>
<tr>
<th>Error req.</th>
<th>$p_{200}$</th>
<th>N1</th>
<th>N2</th>
<th>Mean. dig.</th>
</tr>
</thead>
<tbody>
<tr>
<td>$10^{-1}$</td>
<td>6002</td>
<td>200 (0)</td>
<td>0 (0)</td>
<td>4.82</td>
</tr>
<tr>
<td>$10^{-2}$</td>
<td>13170</td>
<td>200 (0)</td>
<td>0 (0)</td>
<td>6.30</td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>28558</td>
<td>200 (0)</td>
<td>0 (0)</td>
<td>7.84</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>61940</td>
<td>198 (0)</td>
<td>2 (0)</td>
<td>9.42</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>124730</td>
<td>187 (0)</td>
<td>13 (0)</td>
<td>10.90</td>
</tr>
</tbody>
</table>
Test family 3—Procedure 4

<table>
<thead>
<tr>
<th>Error req.</th>
<th>$p_{200}$</th>
<th>N1</th>
<th>N2</th>
<th>Mean. dig.</th>
</tr>
</thead>
<tbody>
<tr>
<td>$10^{-1}$</td>
<td>6164</td>
<td>200 (0)</td>
<td>0 (0)</td>
<td>5.00</td>
</tr>
<tr>
<td>$10^{-2}$</td>
<td>13365</td>
<td>200 (0)</td>
<td>0 (0)</td>
<td>6.41</td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>28801</td>
<td>200 (0)</td>
<td>0 (0)</td>
<td>7.95</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>62156</td>
<td>198 (0)</td>
<td>2 (0)</td>
<td>9.54</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>124929</td>
<td>187 (0)</td>
<td>13 (0)</td>
<td>10.93</td>
</tr>
</tbody>
</table>

Test family 3—Procedure 5

<table>
<thead>
<tr>
<th>Error req.</th>
<th>$p_{200}$</th>
<th>N1</th>
<th>N2</th>
<th>Mean. dig.</th>
</tr>
</thead>
<tbody>
<tr>
<td>$10^{-1}$</td>
<td>6224</td>
<td>200 (0)</td>
<td>0 (0)</td>
<td>5.06</td>
</tr>
<tr>
<td>$10^{-2}$</td>
<td>13386</td>
<td>200 (0)</td>
<td>0 (0)</td>
<td>6.43</td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>28814</td>
<td>200 (0)</td>
<td>0 (0)</td>
<td>7.97</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>62165</td>
<td>198 (0)</td>
<td>2 (0)</td>
<td>9.57</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>124935</td>
<td>187 (0)</td>
<td>13 (0)</td>
<td>10.93</td>
</tr>
</tbody>
</table>

References