JOURNAL OF COMPUTATIONAL PHYSICS **123,** 267–273 (1996)
ARTICLE NO. 0023

Multidimensional Quadrature Algorithms at Higher Degree and/or Dimension

SIMON CAPSTICK

Supercomputer Computations Research Institute and Department of Physics, Florida State University, Tallahassee, Florida 32306

AND

B. D. KEISTER

*Physics Division, National Science Foundation, 4201 Wilson Boulevard, Arlington, Virginia 22230 and Department of Physics, Carnegie Mellon University, Pittsburgh, Pennsylvania 15213**

Received January 27, 1995; revised May 30, 1995

The accuracy of two multidimensional quadrature algorithms is examined with some simple test integrals. It is found that a scheme introduced by Genz is considerably more accurate and is simpler to implement for higher degree and/or dimension than one due to McNamee and Stenger. © 1996 Academic Press, Inc.

1. INTRODUCTION

Multidimensional quadrature methods have been a topic of interest for many years. For suitably well-behaved functions, the most naive procedure is to cast each of the variables on its own grid with a standard one-dimensional quadrature, i.e., a product rule. However, this method involves substantial, if not prohibitive, increases in the total number of quadrature points when the desired precision is increased. Monte Carlo methods offer an attractive alternative, providing reasonable precision with far fewer points, especially in higher dimensions. However, the approach to greater precision varies only as $1/\sqrt{N}$, where N is the total number of points.

Another alternative is a non-product quadrature algorithm. Of interest here are methods which exactly integrate a set of monomials up to some overall degree, but which use fewer total points than a corresponding product rule. An overview can be found in Ref. [1]. Several authors have addressed the specific case where the integrand or its weight function has specific symmetry properties [2–11]. Our focus in this work is a pair of algorithms developed by McNamee and Stenger [10] and by Genz [2] for weight functions which are symmetric and separable in the coordinates. Their method is in principle generalizable to arbitrarily high degree and/or dimension, but results of McNamee and Stenger were reported only for relatively low degree and dimension.

In this paper, we report results of an attempt to systematize as much as possible the McNamee–Stenger method [10] to arbitrary degree and/or dimension and to compare the results of test integrals using their method alongside that of Genz. For the numerical examples, we use integrands with Gaussian weight functions, because of their potential utility in a wide variety of physical applications, such as the calculation of quantum mechanical matrix elements in atomic, nuclear, and particle physics.

2. MULTIDIMENSIONAL ALGORITHMS

The basic task is to approximate N-dimensional integrals

$$\int d\mathbf{x}\, w(\mathbf{x}) f(\mathbf{x}), \tag{1}$$

where $f(\mathbf{x})$ is the integrand and $w(\mathbf{x})$ is the weight function.

Consider a class of integrals whose integrands are polynomials whose overall degree in the variables is limited to a value r. A product rule can integrate not only degree-r quadratures, but also monomials $x_1^r\, x_2^r \ldots$ with a product of degree-r one-dimensional quadratures, corresponding to an overall degree nr. For integrals where the *overall* degree is the important element, the product rule uses too many points to achieve a given level of accuracy. Thus, we seek a set of points and weights which is capable of exactly integrating all monomials up to the overall degree r. For any M such monomials, this requirement gives rise to M equations relating linear combinations of powers of the coordinates of the quadrature points, multiplied by the weights. These equations are linear in the weights, but nonlinear in the points. The solutions can thus be extremely

* Permanent address.

0021-9991/96 $18.00

difficult to obtain and the points and/or weights may be complex, or there may even be no solution.

The methods considered here apply to a more restricted set of integrals, where the weight function $w(\mathbf{x})$ is *symmetric* under the interchange and/or inversion of the individual coordinates and is a *separable* function of the coordinates:

$$w(\mathbf{x}) = w_1(x_1) \cdots w_N(x_N). \tag{2}$$

(The actual condition in the McNamee–Stenger method is slightly more relaxed than this, but the examples we use satisfy the more restricted condition.) In this case, the set of quadrature points must also be symmetric with respect to interchange or inversion. This means that the quadrature point information can be summarized in terms of *generators.* Let u_1, ..., u_p be the values of the evaluation coordinates (nodes). A generator is described by a list of non-zero node values which are to be permuted among the coordinates. For example, the generator $g = [u_1 u_2 u_2 u_4]$, or, more simply, $g = [1224]$, corresponds to the set of all points with one coordinate $x_a = \pm u_1$, two coordinates x_b, $x_c = \pm u_2$, and another $x_d = \pm u_4$, with the remaining coordinates equal to zero.

2.1. *McNamee–Stenger Scheme*

The scheme of McNamee and Stenger [10] (hereafter denoted as MS) uses nodes which are the zeroes of orthogonal polynomials associated with a one-dimensional quadrature:

$$\int dx_1\, w_1(x_1) f(x_1) = \sum_j w_j f(x_{1j}). \tag{3}$$

The one-dimensional orthogonal polynomials are relevant because the integrals under consideration have a symmetric, separable weight function. With the weights known, the nonlinear part of the problem described above is then solved. What remains is a set of equations involving the set of monomials which is *linear* in the unknown weights. Apart from cases where the coefficient matrix is singular (see below), the resulting weights and points can be obtained and they will be real.

The monomials are defined as

$$I_{j_1 j_2 \cdots} = \int d\mathbf{x}\, w(\mathbf{x}) x_1^{j_1} x_2^{j_2} \ldots. \tag{4}$$

All monomials odd in one or more coordinates integrate to zero. The task is therefore to find the weight w_g associated with each generator g such that all *even* monomials are exactly integrated. The relation between generators and monomials is given by a set of moment equations,

$$\sum_g w_g \sum_{\mathbf{x}_g} x_1^{2j_1} x_2^{2j_2} \cdots = I_{2j_1 2j_2 \cdots}, \tag{5}$$

where the inner sum is over all points generated by g.

Each monomial can be characterized by an *axis count* α_m, which is the number of axes containing a non-vanishing power of coordinate. One can also define an axis count α_g for a generator, which is the number of non-zero coordinate values. The contribution of a particular generator (i.e., its associated points) vanishes if $\alpha_g < \alpha_m$. A key element of the MS paper is the fact that only a small subset of monomials with a given overall degree and α_m is needed to determine both the remaining monomials and all of the generators. Therefore, for each subset of monomials with an axis count α_m, one can associate a set of generators with $\alpha_g \geq \alpha_m$. If the monomials and generators are grouped from smallest to largest α, then the coefficient matrix which relates the unknown weights for the generators to the (known) monomials will be block triangular; that is, the matrix will have a block along the diagonal for each subset of monomials and generators with a given axis count plus, in general, non-zero elements above the diagonal.

For each axis count α, there are always at least as many generators as there are monomials. There is thus some freedom in choosing the generators. There can be substantial variation in the number of points associated with a given generator. For example, for degree 11 in six dimensions, the generator [11111] has 192 associated points, while the generator [11223] has 5760. Certainly it is preferred to have a set of generators with the smallest number of integrand evaluations. McNamee and Stenger speculated that the coefficient matrix would always yield a solution when one used the set of generators yielding that smallest number. For the examples they chose, this speculation turns out to be true. However, it is not true in general. For example, consider degree 15 in six dimensions. The MS scheme uses four monomials of degree 14 or less with $\alpha_m = 5$: I_{22222}, I_{22224}, I_{22226}, and I_{22244}. The natural choice of generators in this scheme is the set requiring a minimum number of integrand evaluations: [11111], [22222], [33333], and [44444]. However, the coefficients which connect the weights for any [*iiiii*] to the degree-14 monomials I_{22226} and I_{22244} are the same, namely, $2^5 \cdot u_i^{14}$. The submatrix for $\alpha = 5$ has two identical rows and is, therefore, analytically singular. This problem can be fixed, at the expense of an increased number of integrand evaluations, by swapping one of these generators with another generator with $\alpha = 5$ which has at least two different non-zero node values.

A second problem is that the coefficient matrix, while not analytically singular, can still be ill conditioned. This behavior tends to yield weights which are large and of alternating sign. The effect of this on actual integrals is discussed below.

2.2. *The Genz–Patterson Scheme*

The quadrature scheme of Genz utilizes a set of multidimensional Lagrange interpolation polynomials [12] $P_{j_1 j_2 \cdots}(x_1, x_2, \ldots)$, where each j_i refers to the order of the polynomial in x_i. The quadrature points are the nodes of these interpolation polynomials, which are orthogonal with respect to the quadrature itself. Thus, the moment equations decouple,

$$\sum_g w_g \sum_{\mathbf{x}_g} P_{2j_1 2j_2 \cdots}(\mathbf{x}_g) = J_{2j_1 2j_2 \cdots}, \tag{6}$$

where

$$J_{2j_1 2j_2 \cdots} = \int d\mathbf{x}\, w(\mathbf{x}) P_{2j_1 2j_2 \cdots}(x_1, x_2, \ldots). \tag{7}$$

In this scheme, the node ordering is linked to the ordering of the interpolating polynomials: the node labeled λ_1 is associated with a second-order polynomial, λ_2 with a fourth-order polynomial, etc. This has two implications. First, there is much less freedom in the construction of generators than for the MS method. The *values* of some of the nodes can be permuted, but the possibility of generator swapping does not arise. Second, there is a one-to-one correspondence between generators and the moment integrals J (because the moment equations decouple), with the resulting weights given by an algebraic expression rather than a solution to a matrix equation.

The method can be made efficient by choosing node values, which are in principle arbitrary, that cause certain sets of weights w_g to vanish. One set of nodes considered by Genz is an embedded sequence developed by Patterson [13]. This has the advantage that previous integrand evaluations can be re-used if increased precision is desired.

Because the node values in the Genz scheme are in principle arbitrary, the total number of integrand evaluations in one dimension is roughly twice as high as that of a quadrature based upon orthogonal polynomials. Thus, in low dimensions, it is less efficient than a product rule. However, in higher dimensions, with a judicious choice of node values, the vanishing of many weights w_g results in a quadrature scheme that is more efficient than a product rule and often more efficient than a MS rule.

3. NUMERICAL IMPLEMENTATION

The algorithm was implemented in C++. Generators and monomials were each assigned classes whose member functions handled all bookkeeping information. By overloading `operator=( )` and using reference counting, the code to sort and swap generators could be written in a transparent fashion without additional memory overhead. A generator can be characterized by an array whose elements indicate the number of times a given node value appears. A monomial can be characterized by an array whose elements indicate the power of the coordinate on a given axis. In both cases, the vector contains a variable number of dimensions and a variable number of null elements. They can be constructed by means of a re-entrant function which produces the allowed vectors by calling itself repeatedly with a vector dimension which decrements itself by one unit with each call. The combinatorics needed to allocate arrays involve ratios of large factorials. To avoid integer size overflow, large numbers were manipulated using the multiple-precision Integer package in the GNU C++ library [14].

TABLE I

Results for the Integral $\int d\mathbf{x}\, e^{-\mathbf{x}^2} \cos|\mathbf{x}|$ for Various Generator Sets in Nine Dimensions Using the McNamee–Stenger Method

Degree	No. swaps	No. points	$\lvert W\rvert$	$\lvert[I - I_{\text{ex}}]/I_{\text{ex}}\rvert$
7	0	997	3,807	0.0305
9	0	3,973	112	1.3×10^{-3}
11	0	13,159	481,013	5.3×10^{-4}
11	1	13,159	130,251	6.4×10^{-3}
11	2	13,159	10,768	1.2×10^{-3}
11	3	14,503	505	2.4×10^{-5}
11	4	14,503	149	2.2×10^{-5}
13	0	36,967	452	4.6×10^{-6}
15	0	80,617	∞	∞
15	3	96,745	10^8	1.1×10^{-4}
17	0	197,353	∞	∞
17	5	237,673	19,536	1.2×10^{-6}
19	0	420,075	∞	∞
19	23	580,779	9,345	1.0×10^{-8}

4. SAMPLE RESULTS

To illustrate the method, consider the integral

$$I = \int d\mathbf{x}\, e^{-\mathbf{x}^2} \cos|\mathbf{x}| \tag{8}$$

in nine dimensions. The exact value is

$$I_{\text{ex}} = -\frac{895}{256} \frac{\pi^5 e^{-1/4}}{\Gamma(\frac{9}{2})} = -71.6\,332\,342\,909. \tag{9}$$

Results are shown in Table I for the MS method with various generator sets. Those with zero "swaps" correspond to sets which yield the smallest number of quadrature points. For degrees 15 and 17, this "minimal" set in fact yields algebraic singularities, and these are denoted by ∞ in the tables. By swapping in and out different generators with identical values of α, it is possible to eliminate these

TABLE II

Results for the Integral $\int d\mathbf{x}\, e^{-\mathbf{x}^2} \cos|\mathbf{x}|$ in Nine Dimensions Using the Genz–Patterson Method

Degree	No. points	$\|W\|$	$\|[I - I_{\text{ex}}]/I_{\text{ex}}\|$
7	871	20.8	0.0148
9	3,481	44.3	8.61×10^{-4}
11	11,851	74.3	3.62×10^{-5}
13	36,253	97.7	1.14×10^{-6}
15	100,207	160	2.95×10^{-8}
17	254,737	214	8.34×10^{-10}
19	608,547	219	1.39×10^{-10}

algebraic singularities, at the expense of adding more quadrature points.

In addition, one can see from the table the effect of weights which are large and of alternating sign. In principle, one would prefer an integration scheme in which the points all lie inside the region of integration and the weights are all positive [9]. In the approaches considered in this paper, the first criterion is satisfied, but the second is not. A "figure of merit" is the weight sum [2]

$$|W| := \left[\sum_g |w_g| \sum_{\mathbf{x}_g} 1\right] \Big/ \left[\sum_g w_g \sum_{\mathbf{x}_g} 1\right]. \tag{10}$$

If all of the weights are positive, then $|W| = 1$. For larger dimensions and degrees, such as in the case considered here, individual weights can be positive or negative and can differ from each other by several orders of magnitude. One can see from the table that $|W|$ can vary by a considerable amount. These large weights are a consequence of a coefficient matrix which is nearly singular numerically, although not algebraically singular. It is possible in many cases to "tame" this behavior by further swapping of generators with identical values of α. The value of $|W|$ is an important ingredient in improving convergence of the integration. For example, the degree-15 result in Table I uses three times the number of points as that for degree 13, but with lower accuracy. Since the value of $|W|$ is much larger in the degree-15 case, we might have expected this outcome.

Table II illustrates the results for the same integral in nine dimensions using the Genz–Patterson method (hereafter denoted as GP). Note that the weight sum increases with increasing degree, but the increase is monotonic and is accompanied by a monotonic decrease in numerical error.

Note that both methods are much more efficient than a product rule. For the integral of Eq. (8), a product rule with two points per axis (degree 3) requires 512 integrand evaluations in nine dimensions and yields an error (as defined in the tables) of +0.26; a rule with three points per axis (degree 5) requires 19,583 evaluations in nine dimensions and has an error of −0.014. The latter error is comparable to that of either the MS or GP method with degree 7, requiring less than 1000 evaluations.

As another example, consider the integral

$$I = \int d\mathbf{x}\, e^{-\mathbf{x}^2} \sqrt{1 + \mathbf{x}^2} \tag{11}$$

in a degree-7 quadrature in four dimensions. This integral cannot be performed analytically. Analytic angular integration, coupled with numerical radial quadrature, yields the result $I_{\text{ex}} = 16.6\,743\,812$. In the MS method, the relevant monomials are I_0, I_2, I_4, I_{22}, I_{24}, and I_{222}. The number of generators with axis count 0, 1, 2, and 3, respectively, is 1, 2, 3, 4. For $\alpha = 2, 3$ there are extra generators available for swapping, giving a total of 12 possible generator combinations. The results are summarized in Fig. 1, which shows the associated value of $|W|$ for each rule compared to the deviation from the "exact" result obtained from a high-precision product rule. The number of integrand evaluations ranges from 97 to 185. It is clear from the figure that, for this example, smaller values of $|W|$ are generally associated with better accuracy. Since it is straightforward to examine all 12 possible rules for degree 7 in four dimensions, the value $|W| = 6.3$ serves as a lower bound for this set of rules.

The results for the integral (11) in four dimensions using the GP method through degree 29 are shown in Table III. The node values were generated using an extension to the Patterson sequence for Gaussian weight functions [15]. Convergence is not as rapid as with the cosine integral.

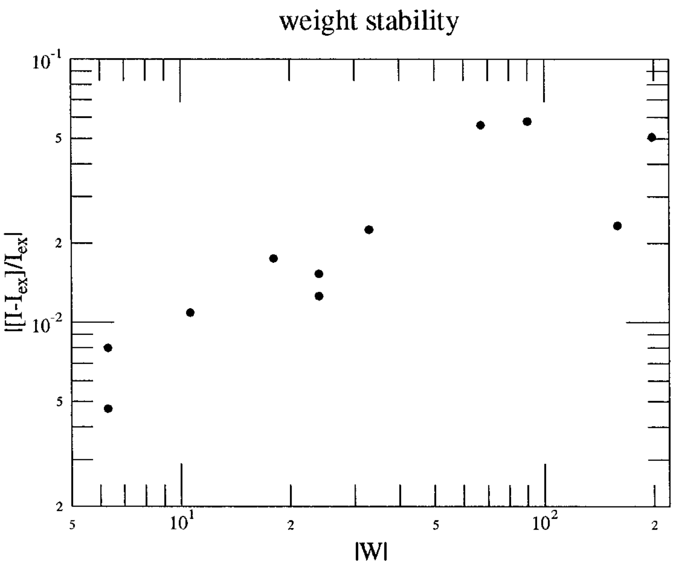

FIG. 1. Effect of large, cancelling weights on numerical evaluation of integral, as discussed in the text.

TABLE III

Results for the Integral $\int d\mathbf{x}\, e^{-\mathbf{x}^2} \sqrt{1 + \mathbf{x}^2}$ in Four Dimensions Using the Genz–Patterson Method

Degree	No. points	$\lvert W\rvert$	$\lvert[I - I_{\rm ex}]/I_{\rm ex}\rvert$
7	81	4.51	0.012
9	201	4.85	3.9×10^{-3}
11	449	3.05	1.4×10^{-3}
13	825	6.77	1.3×10^{-3}
15	1,521	4.36	3.7×10^{-4}
17	2,625	3.06	1.4×10^{-4}
19	4,105	4.17	9.9×10^{-5}
21	6,441	28.6	2.6×10^{-4}
23	9,441	59.5	2.8×10^{-4}
25	13,521	44.9	1.1×10^{-4}
27	18,777	34.1	1.8×10^{-4}
29	25,433	109	1.5×10^{-4}

This is not surprising, since $\cos\lvert\mathbf{x}\rvert$ has a uniformly convergent power series, whereas $\sqrt{1 + \mathbf{x}^2}$ has a unit radius of convergence, with an integration region which includes values of the coordinates outside this radius. Nevertheless, the performance of the method is practically better than that of McNamee and Stenger.

The results for product-rule integration are comparable to those of GP for similar numbers of evaluations at lower degree. The error with four points per axis (degree 7, with 256 evaluations) is 2×10^{-3} and is comparable to the error of the degree-9 GP result. However, with 30 points per axis (degree 29, with 50,625 evaluations), the error is 6×10^{-4}, comparable to the degree-15 GP result (3.7×10^{-4}) using less than one-tenth the number of evaluations.

For larger degree and/or dimension, the number of possible rules using the MS method becomes very large, given the combinatorics of possible generator swaps. For example, in the degree-19, dimension-9 example shown in Table I, there is a total of 2001 generators, from which are chosen 83 to correspond to the allowed monomials. It has already been noted that a rule which generates the smallest number of quadrature points for each axis count can sometimes yield a singular or ill-conditioned matrix equation for the weights. For these cases, some swapping of generators is required, and this will result in a higher number of integrand evaluations. There are some general observations which we can make regarding the construction of "optimal" rules (those which yield relatively small values of $\lvert W\rvert$ for a modest expense in extra integrand evaluations), at least for Gaussian weights. In the previous example, the smallest values of $\lvert W\rvert$ were obtained under the following generator swaps: [11] $\leftrightarrow$ [12], together with [111] $\leftrightarrow$ [222] or [111] $\leftrightarrow$ [122]. Since u_1 is the smallest positive node value, generators consisting *entirely* of u_1 elements represent sets of points clustered closest to the origin, where the Gaussian weight factor is largest. In general, $\lvert W\rvert$ is significantly reduced when these generators are swapped with others containing larger node values. However, choosing generators solely on the basis of the highest node values does not improve the situation and, in fact, can make it worse. The points should also be distributed through the integration volume as much as possible.

TABLE IV

Results for the Integral $\int d\mathbf{x}\, e^{-\mathbf{x}^2} \cos\lvert\mathbf{x}\rvert$ in 25 Dimensions Using the McNamee–Stenger Method

Degree	No. points	$\lvert W\rvert$	$\lvert[I - I_{\rm ex}]/I_{\rm ex}\rvert$
5	1,251	118	2.04
7	20,901	108,361	0.75
9	244,101	15,564	0.07

With a little experience, it is relatively straightforward to find a set of generator swaps which yields a workable rule for a particular degree and dimension. However, with increasing degree, the task becomes more and more tedious, because there are more and more sources of algebraic singularities and matrix ill-conditioning. Furthermore, we have not found a way to make the procedure systematic. The present method is an iterative procedure in which we start with a set of generators, solve the matrix equation for the weights, examine the solution in each subspace as identified by axis count, make additional generator swaps in subspaces with singular or near-singular behavior, and then repeat the process.

We show in Table IV some results for the cosine integral defined in Eq. (8) for 25 dimensions using the MS method. The exact integral is $I = -1.35\,691 \times 10^6$. No generator swaps were attempted. It is interesting to note that $\lvert W\rvert$ for degree 9 is surprisingly small. The total of 244,101 integrand evaluations would correspond to 1.6 points per axis in a standard product rule.

The results for 25 dimensions using the GP method are shown in Table V. For this particular example, the weight sums for degree 7 using the two methods are quite different

TABLE V

Results for the Integral $\int d\mathbf{x}\, e^{-\mathbf{x}^2} \cos\lvert\mathbf{x}\rvert$ in 25 Dimensions Using the Genz–Patterson Method

Degree	No. points	$\lvert W\rvert$	$\lvert[I - I_{\rm ex}]/I_{\rm ex}\rvert$
5	1,251	118	2.04
7	19,751	583	0.446
9	227,001	2,229	0.06

TABLE VI

Results for the Integral $\pi^{-2} \int d\mathbf{x}\, e^{-\mathbf{x}^2} \left[C + \sum_{i<j} (x_i - x_j)^2 \right]^{1/2}$ in Four Dimensions Using the Genz–Patterson Algorithm

Degree	$C = 0$	$C = 1$	$C = 4$
7	3.10786626	2.64609065	3.10723549
9	1.98840410	2.45734391	3.07389967
11	2.44883891	2.52937917	3.08486334
13	1.90244769	2.47063587	3.07895652
15	2.35023326	2.51352820	3.08246780
17	2.19225845	2.49893286	3.08136666
19	2.32214897	2.50743604	3.08179546
21	0.94947077	2.48519194	3.08134499
23	3.87096406	2.52460747	3.08189775
25	1.64319448	2.49493263	3.08149716
27	3.92043757	2.52547659	3.08188046
29	0.69059716	2.48276668	3.08138056

in magnitude, yet the accuracy is similar. As a general observation, however, we find a persistent correlation between small weight sums and lower numerical error for any given quadrature degree.

Note that a product rule in 25 dimensions is essentially out of the question. With only two points per axis (degree 3), $2^{25} = 33{,}554{,}432$ evaluations would be required.

An example of a more general integrand which does not depend solely upon a radial variable is a Jastrow-type correlation,

$$I = \pi^{-N/2} \int d\mathbf{x}\, e^{-\mathbf{x}^2} \left[C + \sum_{i<j} (x_i - x_j)^2 \right]^{1/2}. \qquad (12)$$

The value of C determines the extent to which the square root can be expanded in a power series inside the dominant region $|\mathbf{x}| \leq 1$ prescribed by the Gaussian. The limiting case $C = 0$ does not correspond to a power series at all, and increasing values of C correspond to increasing radii of convergence of the power series. The results using the GP method in four dimensions are shown in Table VI. Not surprisingly, the quadrature works best for $C = 4$, and very poorly for $C = 0$. Indeed, the GP method applied to Eq. (12) with $C = 0$ in nine dimensions can yield negative numbers! Another example of such integrals is

$$I = \pi^{-N/2} \int d\mathbf{x}\, \mathbf{e}^{-\mathbf{x}^2} \prod_{i<j} [C + |x_i - x_j|]. \qquad (13)$$

Such a form containing a modulus does not have a rapidly converging power-series expansion for small values of C, and the numerical results have the same features as those for Eq. (12). These examples clearly indicate that the quadrature algorithms presented here can be used very efficiently with integrands which can be approximated as polynomials, but that caution must be used to verify that an integrand indeed has such behavior.

We also performed the integrations in some of the cases above using adaptive Monte Carlo methods. For an example of such a method, see Ref. [16]. For the square-root integrand of Eq. (11), a run with 500,000 integrand evaluations yielded a statistical uncertainty estimate of 2%. This is reasonable performance from a Monte Carlo method for a positive-definite integrand peaked at the origin, although comparable accuracy can be achieved via the Genz–Patterson scheme with less than 1000 points. On the other hand, Monte Carlo results for the cosine integral of Eq. (8) were much worse, not surprisingly, given the sign changes of the integrand and the fact that the Monte Carlo algorithm used importance sampling.

5. SUMMARY

We have considered here two multidimensional quadrature algorithms which exactly integrate the set of all polynomials of a given limiting degree, for integrands with symmetric, separable weight functions. The methods give similar numerical accuracy and are simple to implement, in cases of low degree. However, as the desired degree of polynomial increases, the method of McNamee and Stenger becomes increasingly more difficult to implement in a way which produces increasing accuracy. The difficulty stems from the fact that the moment equations which determine the quadrature weights have ill-conditioned or even singular coefficient matrices for higher degree and/or dimension. The method of Genz using the embedded coordinate series of Patterson is easier both to implement and to extend to higher degree and provides systematically better accuracy with increasing degree. It should also be stressed that both rules are more efficient than product rules for integrals with higher degree and/or dimension. The results of tests on various sample integrands suggests that the degree of success of the method in applications is linked to the extent that the integrand (modulo the weight function) can be expanded in a power series.

ACKNOWLEDGMENTS

One of us (B.D.K.) wishes to thank Dr. A. Genz for several helpful conversations. We wish to thank Ms. Xiaoqing (Cathy) Cao for her help with the implementation of the McNamee–Stenger algorithm, who was supported by the National Science Foundation's Research Experience for Undergraduates program at The Florida State University under Grant PHY-9200095. This work was also partially supported by the National Science Foundation under Grant PHY-9319641 (B.D.K), by the U.S. Department of Energy through Contract DE-FG05-86ER40273, and by the Florida State University Supercomputer Computations Research Institute which is partially funded by the Department of Energy through Contract DE-FC05-85ER250000 (S.C.).

REFERENCES

1. A. H. Stroud, *Approximate Calculation of Multiple Integrals* (Prentice–Hall, Englewood Cliffs, NJ, 1971).
2. A. Genz, *SIAM J. Numer. Anal.* **23,** 1273 (1986).
3. A. C. Genz and A. A. Malik, *SIAM J. Numer. Anal.* **20,** 580 (1983).
4. P. Keast and J. N. Lyness, *SIAM J. Numer. Anal.* **16,** 11 (1979).
5. V. I. Lebedev, *Zh. vȳchisl. Mat. Mat. Fiz.* **15,** 48 (1975).
6. V. I. Lebedev, *Zh. vȳchisl. Mat. Mat. Fiz.* **16,** 293 (1976).
7. J. N. Lyness, *Math. Comput.* **19**, 260 (1965).
8. J. N. Lyness and P. Keast, *Numer. Math.* **30,** 51 (1978).
9. F. Mantel and P. Rabinowitz, *SIAM J. Numer. Anal.* **14,** 391 (1977).
10. J. McNamee and F. Stenger, *Numer. Math.* **10,** 327 (1967).
11. P. Rabinowitz and N. Richter, *Math. Comput.* **23,** 765 (1969).
12. A. C. Genz, *SIAM J. Sci. Stat. Comput.* **3,** 160 (1982).
13. T. N. L. Patterson, *Math. Comput.* **22,** 847 (1968).
14. D. Lea, *The* GNU C++ *Library User's Manual*, 1988.
15. A. C. Genz and B. D. Keister, *J. Comput. Appl. Math.,* submitted.
16. G. P. Lepage, *J. Comput. Phys.* **27,** 192 (1978).