A Nested Partitioning Procedure for Numerical Multiple Integration

JEROME H. FRIEDMAN
Stanford Linear Accelerator Center
and
MARGARET H. WRIGHT
Stanford University

An algorithm is presented for adaptively partitioning a multidimensional coordinate space on the basis of optimization of a scalar function of the coordinates. The goal is to construct a set of hyperrectangular regions, such that the variation of function values within each region is small. These regions are then used as the basis for a stratified sampling estimate of the definite integral of the function.

Key Words and Phrases. numerical integration, quadrature, bounds-constrained optimization, recursive partitioning
CR Categories: 4.29, 5.16, 5.41

1. INTRODUCTION

Although the numerical evaluation of multiple integrals has received considerable attention, it remains a difficult problem. Most general-purpose techniques today apply to integrands that are relatively "well behaved" in that the integrand can be reasonably well approximated by a low-order polynomial within the region of integration. One technique for trying to deal with integrands that are not so well behaved is to partition the region of integration "adaptively" into subregions, chosen so that the integrand is well behaved within each subregion [6, 11, 12, 14, 15, 17]. Standard techniques can then be applied to evaluate the integral in each subregion, and the integral over the entire region is taken as the sum of the integrals over the subregions.

This paper presents an adaptive partitioning algorithm for multidimensional quadrature in which the subdivision strategy is based on the use of numerical optimization. Although the function evaluations required to develop the partition

The work of J.H Friedman was partially supported by the U.S Department of Energy under Contract DE-AC03-76-SF00515 The work of M.H. Wright was supported in part by the U.S. Department of Energy under contract DE-AS03-76-SF00326, in part by National Science Foundation Grant MCS76-20019-A01, and in part by the U.S. Army Research Office under Contract DAAG29-79-C-0110
Authors' addresses: J.H. Friedman, Stanford Linear Accelerator Center, Stanford, CA 94305, and European Organization for Nuclear Research, CERN, Geneva, Switzerland; M.H. Wright, Department of Operations Research, Stanford University, Stanford, CA 94305.

are not retained for the final integration, this approach appears to be competitive with existing methods on high-dimensional problems. This is because the initial investment in function values for the optimization and subdivision is repaid by the efficiencies resulting from a well-constructed partition.

Several automatic partitioning strategies based on estimated properties of the integrand have been proposed. Some of these strategies rely on factored approximations [14, 15, 17], while the others are general multidimensional adaptive procedures [6, 11, 12]. All of these adaptive techniques (as well as the one presented here) are iterative and are based on top-down successive refinement. At each iteration a particular region is considered; initially this region is the entire integration region. A sampling of the integrand within the region is used to estimate various properties of the integrand, which then guide a strategy for division into several subregions. This process continues until the partitioning has achieved some specified reduction in the estimated error of the approximate integral.

The effectiveness of a partitioning strategy depends, in large part, on the degree to which the properties of the integrand, deduced during the sampling, accurately reflect the characteristics of the function. If the function is badly behaved, its predicted and actual behavior may not even resemble one another; this may lead to ineffective or counterproductive partitions. This possibility is especially likely in high dimensions where even samplings of large cardinality are very sparse; for example, in ten dimensions a sampling of 60,000 points is equivalent to about 3 points per coordinate. On the other hand, a complex partitioning strategy that requires a very large number of integrand evaluations may be inefficient because the additional evaluations might be better expended simply to increase the number of points used to compute the final integral estimate. Thus a good partitioning strategy must be based on a computationally feasible way of assessing the integrand's behavior.

The main distinctions of the new partitioning strategy from previously proposed methods are

(1) the behavior of the integrand within a region is estimated by means of multiparameter optimization rather than by sampling;

(2) all subregions are defined by simple bounds on the coordinates.

2. OVERVIEW OF THE ADAPTIVE REFINEMENT PROCEDURE

Consider a hyperrectangular region R, defined by simple bounds on each coordinate:

$$R = \{x \mid x_i^{\mathrm{L}} \leq x_i \leq x_i^{\mathrm{U}}\}, \tag{1}$$

where x is the vector $(x_1, x_2, \ldots, x_n)^{\mathrm{T}}$. The essence of an adaptive strategy for partitioning R can be specified by three attributes:

(1) a measure $s(R)$ that indicates the "badness" of the integrand's behavior within R;
(2) a method for subdividing the region after $s(R)$ has been determined;
(3) a procedure for processing the new subregions and for terminating the partitioning (a global stopping criterion).

The quantity used in the new algorithm to characterize the integrand is the difference of extreme values within R, weighted by the volume of R. Let

$$v(R) = \max_{x \in R} f(x) - \min_{x \in R} f(x), \tag{2}$$

where $f(x)$ is a scalar-valued function (presently, the integrand function). The *spread* $s(R)$ is then defined by

$$s(R) = v(R) \cdot \mathrm{vol}(R). \tag{3}$$

The spread measure $s(R)$ bounds the uncertainty of a quadrature or Monte Carlo estimate of the integral over R and is taken to indicate the contribution of R to the uncertainty of a global estimate of the integral.

The choice of the measure (3) depends in two crucial ways on the simply bounded form (1) of R. First, the volume of such a region is easy to compute:

$$\mathrm{vol}(R) = \prod_{i=1}^{n} (x_i^{\mathrm{U}} - x_i^{\mathrm{L}}).$$

This would not be true if more complicated regions were allowed. The other term (2) may seem at first glance to be computationally intractable since two optimization problems must be solved to calculate it. However, methods for optimization with simple bounds on the variables are well developed, and thus the subproblems associated with (2) can be solved quite efficiently if f is a reasonable function. Section 3 gives some details of the optimization procedure.

Given that (3) is the spread measure, the second element of the partitioning algorithm involves dividing the single region (1) into disjoint simply bounded subregions. The strategy for subdivision is based on the assumption that the same quadrature rule will be applied in each subregion at the conclusion of the partitioning so that the aimed-for final result is a list of regions with "similar" spread measures. The method by which a given region is refined to achieve this goal is described in Section 4.

Finally, after R has been partitioned, the daughter subregions are merged into the list of all regions. If the global stopping criteria are satisfied, the partitioning terminates; otherwise the list of regions is scanned for the one with the largest spread measure, which is then considered for refinement at the next iteration. Details of this aspect of the algorithm are given in Section 4.

Figure 1 illustrates the partitioning achieved by applying this recursive partitioning procedure to the function

$$\begin{aligned} f(x_1, x_2) = {} & \exp\{-15[x_1^2 + (x_2 - 0.5)^2]\} \\ & + \exp\{-15[(x_1 + 0.433)^2 + (x_2 + 0.25)^2]\} \\ & + \exp\{-15[(x_1 - 0.433)^2 + (x_2 + 0.25)^2]\} \end{aligned}$$

with $-1 \le x_1 \le 1$ and $-1 \le x_2 \le 1$. Figure 1a shows an isometric representation of the surface defined by $y = f(x_1, x_2)$, Figure 1b displays some isopleths of the function on the plane, and Figure 1c shows the partitioning of the plane achieved by applying the above procedure recursively, in this case creating eleven subregions. The numbers indicate the order in which the corresponding cuts were made.

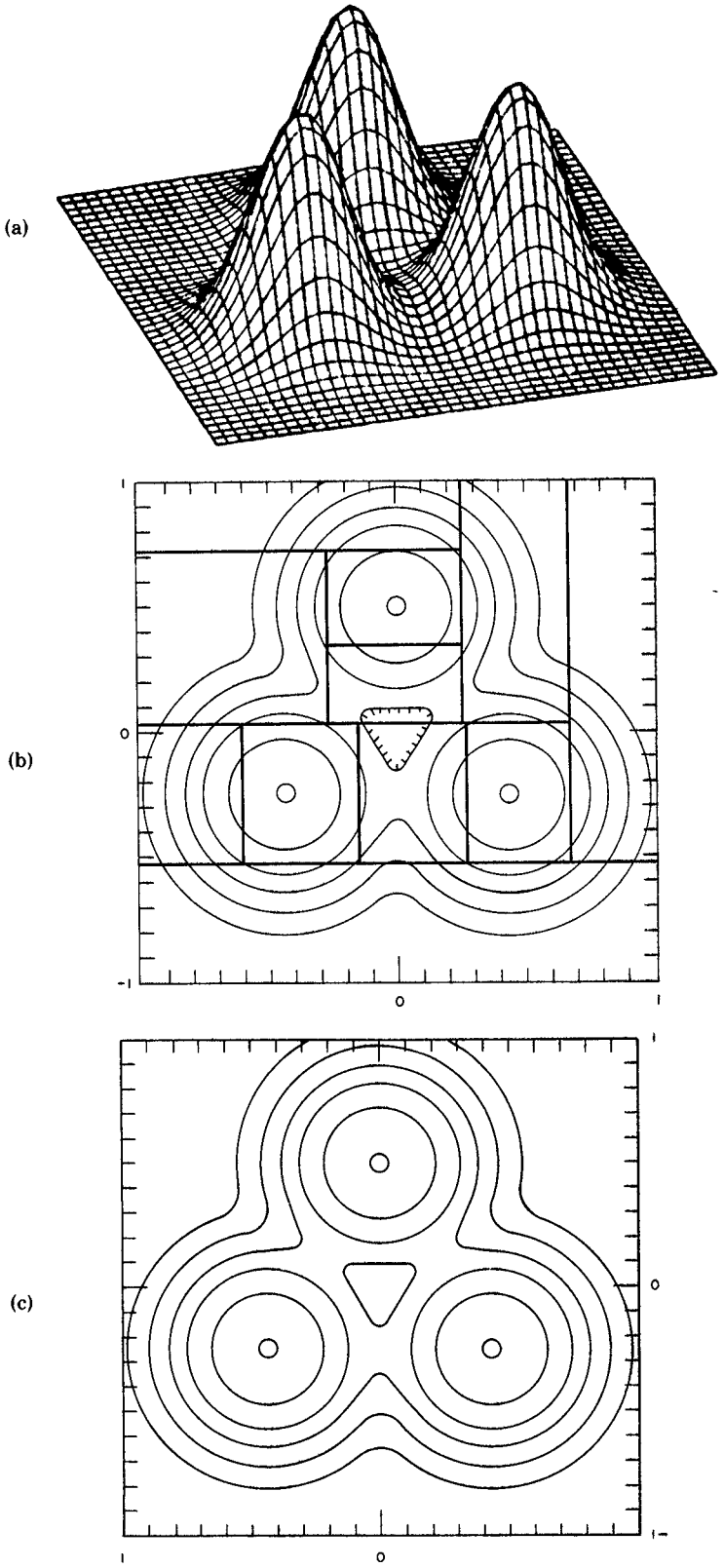

Figure 1.

3. OPTIMIZATION WITH SIMPLY BOUNDED VARIABLES

To compute the spread measure (3) at each step of the partitioning procedure, it is necessary to solve two bounds-constrained optimization problems of the form

$$\begin{aligned} \text{M1:} \quad & \min f(x) \\ & \text{subject to } x^{\mathrm{L}} \leq x \leq x^{\mathrm{U}} \\ \text{M2:} \quad & \max f(x) \\ & \text{subject to } x^{\mathrm{L}} \leq x \leq x^{\mathrm{U}}, \end{aligned}$$

where the scalar-valued function f drives the partitioning; and the vectors x^{L} and x^{U} contain, respectively, the lower and upper bounds that define the desired region.

The problem M2 can be treated as a minimization problem involving $(-f(x))$; therefore all subsequent discussion will concern minimization only.

In a typical quadrature problem, $f(x)$ will be twice continuously differentiable, or at least nonsmooth only at isolated points. The algorithm selected to solve problem M1 should consequently be able to perform well on a smooth function. However, in most instances the derivatives of f will not be available so that the method of choice should require function values only. On the basis of these considerations, the optimization method used in the partitioning algorithm is a bounds-constrained quasi-Newton method with finite-difference approximations to first derivatives.

Quasi-Newton methods for unconstrained optimization have a remarkable history, beginning with Davidon [2] and Fletcher and Powell [4]; a recent summary of their motivation and properties is given by Dennis and Moré [3]. Quasi-Newton methods have been extremely successful on a wide variety of problems; if properly implemented, they are quite robust and usually display superlinear convergence. The idea of a quasi-Newton method is to build up second-order information about the function to be minimized by incorporating the observed changes in the gradient into a matrix that approximates the underlying matrix of second partial derivatives (Hessian matrix), so that the method should eventually behave like Newton's method.

A typical iteration of an unconstrained quasi-Newton method begins with the current iterate, x; the gradient vector of f, g; and an approximation to the Hessian, the matrix B.

(i) If the norm of the gradient is sufficiently small, the procedure terminates. Otherwise proceed to step (ii).

(ii) Solve the linear system

$$Bp = -g$$

for the search direction p. In practice, numerical stability is ensured by using a Cholesky factorization of the matrix B, so that p is always a direction of descent for f. This essential feature is due to Gill and Murray [7].

(iii) Find a step length $\alpha > 0$ that yields a sufficient decrease in f, so that

$$f(x + \alpha p) < f(x).$$

The step-length algorithm used in the current procedure is the safeguarded quadratic interpolation procedure implemented by Gill and Murray [8].

(iv) Evaluate the gradient at $x + \alpha p$, and produce an updated Hessian approximation by

modifying the Cholesky factorization of B with the BFGS quasi-Newton update (see [9]). Return to step (i) with $x + \alpha p$ as the next iterate.

In the present algorithm, it is asumed that the analytic gradient of f is not available, so that the calculation of the vector g is carried out using finite differences.

When the variables are constrained to be between simple bounds, the above algorithm can be modified in a straightforward manner. At each iteration, it is determined whether a given variable is "free" to vary or is to be held "fixed" at one of its bounds. After this decision, the unconstrained algorithm is applied with the following changes:

(1) The gradient, direction of search, and approximate Hessian represent the free variables only.
(2) The step length in step (iii) may need to be restricted to prevent a free variable from violating a bound during an iteration. In this case the variable subsequently becomes fixed on that bound.
(3) The updates to B involve only the free variables.
(4) The test for convergence in step (i) is based on the norm of the gradient with respect to the free variables. When this quantity is sufficiently small, it is necessary to check whether freeing any variable currently held fixed on its bound will lead to a reduction in f. This determination is made by checking the sign of the gradient with respect to all fixed variables; for example, if the ith variable is fixed on a lower bound and the ith component of the gradient is negative, the ith variable can be released from its bound.

The many additional details of the algorithm are given in full in the software documentation [5], including user-controlled tolerances that define, for example, "sufficiently small" in step (i).

Before beginning to solve problems M1 and M2, the function f is evaluated at a random sample of points in R, and the point with the smallest (largest) function value is used as the initial point for the minimization (maximization). (The size of this sample is user controllable; values of 50 to 100 seem to be adequate, depending on the problem.) Although for some regions the extrema computed at previous states could be used as part of the initial sample, this information is not retained in order to improve robustness. Especially in high dimensions, convergence to a spurious saddle point might preclude any further search for the true extremum since the convergence criteria would be satisfied at the initial point.

An additional feature of the algorithm that is designed to improve robustness is a "local search," to be used if the gradient is small at the initial point. The idea is again to avoid a spurious indication of convergence at a saddle point. The details of the local search are rather complicated, and only the general idea is sketched here. First, a point perturbed from the initial point is generated by moving a small, feasible amount along each coordinate until the function value changes sufficiently. Next, a feasible descent direction is constructed at whichever point is lower and an exact line search is carried out along that direction. Then, a second feasible descent direction, orthogonal to the first, is generated at the lowest point found, and a second exact line search is performed. If this transfer fails to yield a "sufficiently lower" function value, the initial point is accepted; otherwise the quasi-Newton procedure begins at the new point.

This local search cannot be guaranteed to move away from a saddle point since the function is evaluated only at a finite number of points along two directions. In practice, the local search has been quite successful on all the examples tested.

4. REFINEMENT INTO SUBREGIONS

Given that the spread measure (3) has been computed by the solution of two optimization problems, we next consider the numerical procedure by which the region is subdivided. In this section we are concerned with a particular region R defined by (1). Let $x^{\max}$ and $x^{\min}$ denote the points in R where f achieves its maximum and minimum, respectively, with $f^{\max}$ and $f^{\min}$ the corresponding function values. For simplicity, we assume that there are no other local extrema in R; the implemented procedure contains provisions to handle situations where this assumption is not satisfied.

A partitioning strategy that allowed completely general regions might divide R into two disjoint parts with equal spread measures, as follows. Suppose that for a given value $\bar{f}$, $f^{\min} \le \bar{f} \le f^{\max}$, one could determine an isoplethic surface $\{x \mid f(x) = \bar{f}\}$ that separates R into two parts, $R^{\max}$ (which contains $x^{\max}$) and $R^{\min}$ (which contains $x^{\min}$). Under the uniqueness assumption, the differences of extreme function values within $R^{\max}$ and $R^{\min}$, respectively, would be $(f^{\max} - \bar{f})$ and $(\bar{f} - f^{\min})$. The desired choice for $\bar{f}$ would make the spread measures associated with $R^{\max}$ and $R^{\min}$ equal; that is,

$$(f^{\max} - \bar{f})\mathrm{vol}(R^{\max}) = (\bar{f} - f^{\min})\mathrm{vol}(R^{\min}). \tag{4}$$

The strategy just described is, of course, impractical because the determination of the isopleths would, in general, be an extremely complex numerical procedure. Since the resulting subregions $R^{\max}$ and $R^{\min}$ would no longer be defined in general by simple bounds, the next optimization problem would also be much more complicated to solve. Furthermore, it would be much more difficult to compute the volume of such a general region.

The strategy adopted in the present algorithm divides the region R into a collection of simply bounded subregions by constructing an axis-oriented hyperrectangular approximation to either $R^{\max}$ or $R^{\min}$. Either $f^{\max}$ or $f^{\min}$ is selected as the "major" extremum (f^{M}), that is, that for which the corresponding function value is farthest from the mean function value $\bar{f}$ in R ($\bar{f}$ is the average of function values at the initial random sample of points).

If $(f^{\max} + f^{\min})/2 \ge \bar{f}$, then $f^{\mathrm{M}} = f^{\max}$, $f^{\mathrm{m}} = f^{\min}$; otherwise $f^{\mathrm{M}} = f^{\min}$, $f^{\mathrm{m}} = f^{\max}$ (with the corresponding choices for x^{M}, x^{m}).

We then seek to define a region R^{M} containing x^{M} by two sets of "cuts" (δ^+, δ^-) along the positive and negative coordinate directions from x^{M}:

$$R^{\mathrm{M}} = \{x \mid x_i^{\mathrm{M}} - \delta_i^- \le x_i \le x_i^{\mathrm{M}} + \delta_i^+\}, \qquad \delta_i^-, \delta_i^+ \ge 0, \quad i = 1, \ldots, n$$

with δ^+, δ^- chosen such that

$$\begin{aligned} f(x^{\mathrm{M}} + \delta_i^+ e_i) &= \bar{f} \\ f(x^{\mathrm{M}} - \delta_i^- e_i) &= \bar{f}, \qquad i = 1, \ldots, n \end{aligned} \tag{5}$$

where e_i is the ith column of the identity matrix.

The equation to be satisfied by $\bar{f}$ is a rearrangement of (4),

$$\gamma f^{\mathrm{M}} + (1 - \gamma) f^{\mathrm{m}} = \bar{f}, \tag{6}$$

where $\gamma = \mathrm{vol}(R^{\mathrm{M}})/\mathrm{vol}(R)$.

Because R^M is defined by simple bounds,

$$\text{vol}(R^M) = \prod_{i=1}^{n} (\delta_i^+ + \delta_i^-).$$

Thus, the vectors δ^+, δ^- are the solution of the $2n$ nonlinear equations:

$$\begin{aligned} f(x^M + \delta_i^+ e_i) &= \frac{\prod_{i=1}^{n} (\delta_i^+ + \delta_i^-)}{\text{vol}(R)} f^M + \left[1 - \frac{\prod_{i=1}^{n} (\delta_i^+ + \delta_i^-)}{\text{vol}(R)}\right] f^m, \\ f(x^M - \delta_i^- e_i) &= \frac{\prod_{i=1}^{n} (\delta_i^+ + \delta_i^-)}{\text{vol}(R)} f^M + \left[1 - \frac{\prod_{i=1}^{n} (\delta_i^+ + \delta_i^-)}{\text{vol}(R)}\right] f^m, \\ & \qquad i = 1, \ldots, n. \end{aligned} \tag{7}$$

Several considerations affect the choice of solution method for the nonlinear system (7). It is undesirable to expend too much effort in solving (7) since the solution need not be computed with very high accuracy. This means that a Newton-type method (for solving nonlinear equations) based on standard finite differences is unacceptable because of the $2n$ function evaluations required at every iteration to compute the Jacobian. A reasonable alternative is to use a secant-type method, where the elements of the Jacobian are approximated by differencing the function values from the previous iteration. The switch to a secant method is worthwhile because such methods display local superlinear convergence and are typically as effective as a Newton-type method in moving from a poor initial estimate of the solution to a reasonably good one (which is all that is required in this case) [16].

Even a secant method for solving (7) could be considered objectionable because it requires the solution of a $2n \times 2n$ linear system at each iteration. However, the special form of (7) allows it to be transformed to an equivalent, but simpler, nonlinear system.

The right-hand side of (7) is a vector with equal components, and hence the vectors δ^+, δ^- also satisfy the $2n$ nonlinear equations

$$\begin{aligned} f(x^M + \delta_1^+ e_1) - f(x^M + \delta_2^+ e_2) &= 0 \\ &\vdots \\ f(x^M + \delta_{n-1}^+ e_{n-1}) - f(x^M + \delta_n^+ e_n) &= 0 \\ f(x^M + \delta_n^+ e_n) - f(x^M - \delta_1^- e_1) &= 0 \\ f(x^M - \delta_1^- e_1) - f(x^M - \delta_2^- e_2) &= 0 \\ &\vdots \\ f(x^M - \delta_{n-1}^-) - f(x^M - \delta_n^- e_n) &= 0 \\ f(x^M + \delta_1^+ e_1) - \tilde{f} &= 0. \end{aligned} \tag{8}$$

The attractive feature of the system (8) is that since each equation (except for the last) involves only two adjacent unknowns, the Jacobian displays the following

special structure:

$$\begin{array}{cccccccc} x & x & 0 & \cdot & \cdot & \cdot & \cdot & 0 \\ 0 & x & x & 0 & \cdot & \cdot & \cdot & 0 \\ 0 & 0 & x & x & 0 & \cdot & \cdot & 0 \\ \cdot & \cdot & & & & & & \cdot \\ \cdot & \cdot & & & & & & \cdot \\ \cdot & \cdot & & & & & & \cdot \\ 0 & 0 & 0 & \cdot & \cdot & 0 & x & x \\ x & x & x & \cdot & \cdot & \cdot & x & x \end{array} \tag{9}$$

If no interchanges are necessary, the matrix (9) will be reduced to upper triangular form very easily, by simply subtracting multiples of each successive row from the last. This means that solving the linear system at each iteration is extremely fast.

Each iteration of the secant method for solving the nonlinear system (8) requires $2n$ function evaluations for the latest values of $\{\delta_i^+, \delta_i^-\}$. Typically, three or four iterations are required in order for (8) to be satisfied with reasonable accuracy.

Numerous safeguards are included in the secant procedure—in particular, each variable δ_i^+, δ_i^- is constrained to remain within the range where the solution must lie, and the norm of the vector that is the left-hand side of (8) is required to decrease at every iteration.

For simplicity of exposition, certain complications were not included in the preceding discussion of the partitioning strategy—in particular, the fact that not all possible directions are considered as candidates for cuts. Since the bookkeeping overhead for any cut is the same, it is prudent to disregard cuts that appear to be ineffective. Certain directions are eliminated for two reasons. First, x^M may be very close to an upper or lower bound, so that the cut would be insignificant. Therefore, no cut is made along the ith positive direction if

$$x_i^U - x_i^M \leq \beta(x_i^U - x_i^L),$$

nor along the ith negative direction if

$$x_i^M - x_i^L \leq \beta(x_i^U - x_i^L),$$

where $0 < \beta < 1$ (currently, $\beta = 0.05$).

Furthermore, the solution values for $\delta_i^{\pm}$ are constrained to satisfy

$$\begin{aligned} 0 < \delta_i^+ &\leq \alpha(x_i^U - x_i^M), \\ 0 < \delta_i^- &\leq \alpha(x_i^M - x_i^L), \end{aligned} \tag{10}$$

where $0 < \alpha < 1$ (currently, $\alpha = \frac{1}{2}$). Before beginning the iteration procedure to solve for the cuts, f is evaluated at the points where $\delta_i^{\pm}$, $i = 1, \ldots, n$, are at the upper bounds given by (10); the values of γ and $\tilde{f}$ from (6) are then computed at this initial configuration. Assume that $f^M = f^{max}$ (a similar analysis holds when $f^M = f^{min}$), and consider a possible positive cut along the ith coordinate. By assumption, f is monotonic along the ith coordinate (moving away from the extremum) so that the value of f corresponding to the maximum δ_i^+ will be smaller than $\tilde{f}$ at any other δ_i^+ in the acceptable range. In addition, the initial $\tilde{f}$ will be the maximum possible value. Thus, if f at the initial δ_i^+ exceeds $\tilde{f}$, there

can be no solution to (7) in the desired range, and so no attempt will be made to find δ_i^+ (the ith upper bound remains unaltered).

The above procedure is carried out for all possible directions. It should be noted that if any direction is eliminated, the values of γ and $\tilde{f}$ in (6) must be recomputed.

The inner region R^M is hyperrectangular and can be inserted directly into the global list of current regions. The outer part is hyperrectangular only if a single solution for a $\delta_i^\pm$ exists (which is quite often the case). If solutions for several $\delta_i^\pm$ exist, then the outer region must be decomposed into hyperrectangular regions by using the values of the $\delta_i^\pm$ for successive nested bisections [11]. These regions are then inserted into the global list. This procedure can divide the original hyperrectangle into as many as $2n + 1$ subregions (if x^M is not located on a boundary). However, the restrictions imposed by (10) usually greatly reduce the number of subregions, which can be (and quite often are) as small as two.

5. MULTIPLE INTEGRATION

The result of applying the nested refinement procedure described in the previous three sections is a set of hyperrectangular subregions of the domain of integration that are mutually exclusive and collectively cover the integration region. The variation of integrand values within the regions has been designed to be substantially less than between the regions. Since the regions are mutually exclusive, the integrals within each can be summed to form the global integral estimate.

A variety of methods exist for evaluating the definite integral within each subregion. (For an excellent and rather complete survey, see Stroud [18].) These methods can be crudely characterized by the regularity they require of the integrand (and/or its derivatives to various orders) and their accuracy per integrand evaluation. At one extreme are the Monte Carlo methods [10], which usually require very little of the integrand (and thus are quite robust), but which converge rather slowly. Monte Carlo methods also yield a simple uncertainty estimate. At the other extreme are the high-degree quadrature formulas [18], which can be applied only to very regular integrands, but which can yield high accuracy for very few integrand evaluations.

Limited experience has indicated that the more robust methods perform best in conjunction with this partitioning method. Of these, the greatest success has, so far, been obtained with the quasi-uniform Monte Carlo methods of Korobov [13]. Reasonable success has also been achieved with simple pseudorandom Monte Carlo methods [10].

The integration method used in each subregion for the examples of the next section is the quasi-uniform Monte Carlo method of Korobov [13], as described in Stroud [18]. (One exception is I_a of Table III, for which pseudorandom Monte Carlo was used.) The rate of convergence of this quasi-uniform method with increasing N depends on the smoothness of the integrand, but it is never slower than $1/N$. This method (like most quadrature methods) does not provide a simple estimate of the uncertainty associated with the integral evaluation in each subregion. It has been found empirically that the quantity

$$\sigma_i = \frac{1}{2}\frac{S_i}{N}$$

provides a reliable (and usually quite conservative) estimate of the uncertainty associated with this method. Here S_i is the spread of the ith region, N is the number of sample points, and the factor $\frac{1}{2}$ is introduced because of the convention of reporting uncertainty as a symmetric (plus or minus) half-value about the estimate. Since the integral estimates in each subregion are independent, the total uncertainty σ is taken as the square root of the sum of squares of the individual region uncertainties

$$\sigma = \frac{1}{2N}\left[\sum_{i=1}^{M} S_i^2\right]^{1/2}. \tag{11}$$

Here M is the number of subregions.

The uncertainty estimate (11) can be used as a basis for terminating the partitioning. At a given stage of partitioning, let there be M subregions and $N_p(M)$ integrand evaluations. If one wishes to estimate the integral with (prespecified) uncertainty σ_0, then from (11)

$$N_I(M) = \frac{1}{2\sigma_0}\left[\sum_{i=1}^{M} S_i^2\right]^{1/2} \tag{12}$$

integrand evaluations will be required to estimate the integral in *each* subregion. Therefore, the total number of integrand evaluations needed to achieve accuracy σ_0 if the partitioning is stopped after M regions is

$$N_T(M) = N_p(M) + N_I(M) \cdot M. \tag{13}$$

As the partitioning proceeds (increasing M), $N_p(M)$ increases (approximately linearly) while $N_I(M)$ decreases due to the reduction in spread. This reduction tends to be very rapid initially, tapering off to slow reduction for large M. Therefore, $N_T(M)$ tends to decrease for small (increasing) M, reaches a minimum, and then starts to increase slowly. An optimal strategy is to terminate partitioning at the point at which $N_T(M)$ achieves its minimum value. Since it is not possible to know (in advance) this optimum value, we terminate the partitioning at the first point for which $N_T(M)$ (13) fails to decrease for several (five) successive iterations. Equation (12) is then used to determine the number of evaluation points $N_I(M)$ needed to perform the final integration in each subregion.

6. EXAMPLES

In this section, we attempt to illustrate some of the properties of this nested refinement procedure for multiple integration by applying it to several examples presented by others to illustrate their integration procedures.

Tables I–III present the results. For each example, we show the estimated integral value with associated uncertainty (11), total number of evaluations of the integrand (partitioning plus integral evaluation) N_T, the number used for the partitioning stage alone N_p, and the resulting number of subregions. The value of N_p includes the function evaluations used in (a) the initial random sampling, (b) solving the two optimization problems, and (c) solving the nonlinear system (8), in each subregion. For each case the value of σ_0 in (12) was chosen to be the uncertainty achieved by the method with which the example was published. The results of an n product of an m-point Gauss–Legendre rule (see [1, Table 25.4])

Table Ia

$$I_p = \left(\frac{10}{\sqrt{\pi}}\right)^p \int_0^1 d^p x \exp\left[-100 \sum_{i=1}^{p} \left(x_i - \frac{1}{2}\right)^2\right] = 1.0$$

Integral	This method	LePage [15]	Gauss-Legendre
I_4	0.999 ± 0.007 $N_T = 7403$ $N_p = 3851$ 24 regions	0.994 ± 0.007 $N_T = 10000$	0.892 $N_T = 10000$
I_9	1.01 ± 0.008 $N_T = 277238$ $N_p = 83626$ 388 regions	1.001 ± 0.005 $N_T = 100000$	71.364 $N_T = 2.0 \times 10^6$ 0.774 $N_T = 10^9$

Table Ib

$$I_p = \frac{1}{2}\left(\frac{10}{\sqrt{\pi}}\right)^p \int_0^1 d^p x \left\{\exp\left[-100 \sum_{i=1}^{p} \left(x_i - \frac{1}{3}\right)^2\right] + \exp\left[-100 \sum_{i=1}^{p} \left(x_i - \frac{2}{3}\right)^2\right]\right\} = 1.0$$

Integral	This method	LePage [15]	Gauss-Legendre
I_2	1.000 ± 0.003 $N_T = 2279$ $N_p = 1339$ 20 regions	0.999 ± 0.002 $N_T = 300000$	0.999 $N_T = 2304$
I_4	0.998 ± 0.007 $N_T = 10230$ $N_p = 4662$ 29 regions	1.003 ± 0.006 $N_T = 300000$	0.927 $N_T = 10000$
I_7	0.994 ± 0.005 $N_T = 190894$ $N_p = 43449$ 185 regions	0.991 ± 0.007 $N_T = 2.4 \times 10^6$	2.27 $N_T = 279936$
I_9	1.03 ± 0.025 $N_T = 303228$ $N_p = 151033$ 305 regions	0.96 ± 0.04 $N_T = 1.5 \times 10^6$	240.08 $N_T = 262144$ 0.0065 $N_T = 1.95 \times 10^6$

are also shown with each example. (The number of points taken on each coordinate m is the nth root of the total number of points given.)

Table I shows results for a series of integrals presented by LePage [15]. LePage employs a factored approximation of the form

$$\hat{f}(x) = \prod_{i=1}^{n} f_i(x_i), \tag{14}$$

which is used as a basis for pseudorandom Monte Carlo importance sampling within the region of integration. This procedure should be especially suitable for

Table II

$$I_a = 100 \int_0^1 d^6x \exp\left[-\sum_{i=1}^{6} (0.6 + 0.4i)x_i\right] = 0.719022$$

$$I_b = 0.01 \int_0^1 \frac{d^6x}{\prod_{i=1}^{5} [0.1 + 0.01 + (x_i + x_{i+1})^2]}$$

$$I_c = 0.01 \int_0^1 \frac{d^6x}{[0\,1 + \sum_{i=1}^{6} 0.1i(x_i - 0.1i)^2]^3}$$

$$I_d = 100 \int_0^1 \frac{d^6x}{1.0 + [\sum_{i=1}^{6} (1.0 + 0.1i)x_i]^4}$$

Integral	This method	Sasaki [17] Method 1	Sasaki [17] Method 2	Gauss-Legendre
I_a	0.7193 ± 0.0005 N_T = 40526 N_p = 7010 28 regions	0.7182 ± 0.0005 N_T = 70000	0 7194 ± 0.0003 N_T = 84529	0.71902 N_T = 15625 0.71902 N_T = 46656
I_b	0.1740 ± 0.0004 N_T = 46656 N_p = 6183 27 regions	0 1718 ± 0.0004 N_T = 70000	0 1721 ± 0.0004 N_T = 84529	0.1722 N_T = 15626 0 1723 N_T = 46645
I_c	0 5022 ± 0 0004 N_T = 10676 N_p = 2889 13 regions	0.5005 ± 0 0004 N_T = 70000	0 5017 ± 0.0004 N_T = 84529	0.498 N_T = 4096 0.502 N_T = 15625
I_d	0.8193 ± 0.0008 N_T = 46663 N_p = 10798 45 regions	0 816 ± 0.001 N_T = 70000	0.8214 ± 0.0008 N_T = 84529	0.8208 N_T = 15626 0.8208 N_T = 46645

the integrals presented in Table Ia since the factored approximation of (14) is exact. As seen in Table Ia, it considerably outperforms the one presented here in high dimensionality. However, the best procedure in this case would be to integrate each one-dimensional function separately and then form the combined integral as the product of the one-dimensional integrals. The integrals presented in Table Ib do not conform exactly to the factored approximation of (14), and the comparison for these cases is more favorable to the partitioning method presented here.

Table II shows results of applying this procedure to four integrals presented by Sasaki [17]. He employs a factored approximation of the form

$$\hat{f}(x) = \prod_{i=1}^{n-1} f_i(x_i, x_{i+1}). \tag{15}$$

Each function $f_i(x_i, x_{i+1})$ is represented by an adaptive piecewise constant approximation on the plane. For the first two integrands of Table II, the factored approximation (15) is exact, while for the last two, it is not. The method presented here is seen to perform well in comparison to that of Sasaki for these integrands. However, as Table II indicates, these integrands are well approximated by low-

Table III

$$I_a = \int_0^1 d^{10}x \prod_{i=1}^{10} i x_i^{i-1} = 1.0$$

$$I_b = \int_0^1 d^5x\, f(x) = \frac{1}{54} = 0.01851851$$

$$f(x) = 1.0, \quad \text{if} \quad \begin{array}{l} 0 \le x_1 \le 1 \\ 0 \le x_2 \le 1/2 \\ 0 \le x_3 \le 1/3 \\ 0 \le x_4 \le 2/3 \\ 1/3 \le x_5 \le 1/2 \end{array}$$

$$f(x) = 0, \quad \text{otherwise}$$

$$I_k = \int_{-5}^{5} d^k x \prod_{i=1}^{k} \left[(2\pi)^{-1/2} \exp\left(-\frac{1}{2} x_i^2\right)\right] = 1.0$$

Integral	This method	Halton and Zeidman [11]	Gauss-Legendre
I_a	1.03 ± 0.02 $N_T = 91096$ $N_p = 50138$ 273 regions	$0\,944 \pm 0\,029$ $N_T = 205677$	0.921 $N_T = 59049$
I_b	$0.018519 \pm 0.21 \times 10^{-5}$ $N_T = 26509$ $N_p = 16809$ 100 regions	$0.018516 \pm 0.11 \times 10^{-4}$ $N_T = 120145$	0.01969 $N_T = 32768$
I_5	0.999 ± 0.003 $N_T = 15378$ $N_p = 7816$ 38 regions	$0.96 \pm 0\,02$ $N_T = 105821$	1.155 $N_T = 16807$
I_{10}	0.98 ± 0.01 $N_T = 68233$ $N_p = 28092$ 137 regions	0.90 ± 0.11 $N_T = 453872$	0.0076 $N_T = 1.05 \times 10^6$

order polynomials, and a simple Gauss-Legendre product rule outperforms both methods in this case.

Table III shows results on several of the integrals presented by Halton and Zeidman [11]. They describe a nested refinement procedure based on successive bisection. The procedure described in this report was motivated, to a substantial degree, by this Monte Carlo sequential stratification (MCSS) technique. Inspection of Table III indicates that partitioning based on optimization described in this report compares favorably with the MCSS procedure.

7. DISCUSSION

The examples of the previous section illustrate that the use of function optimization to construct a partition can be quite effective, even though at first sight the strategy might seem to be too expensive.

The purpose of applying the partitioning is to divide the integration region into subregions, such that the behavior of the integrand within each is relatively good

when compared to its behavior over the entire integration region. In this context, "bad behavior" is ideally defined as error associated with a particular integration method. Our choice of spread (3) as such a measure is motivated by the relative ease and reliability with which it can be estimated (using function optimization), as compared to other properties of the integrand (such as the variance) that require adequate sampling to be reliably estimated. The partitioning procedure will be most effective in those cases for which the spread measure closely corresponds to the uncertainty in the integral estimate.

The main objective of the isoplethic division strategy is to make finer divisions (locally) in those directions in which the integrand is most rapidly varying. The strategy tends to accomplish this even if the resulting hyperrectangle does not closely correspond to a function isopleth. However, the resulting reduction in spread will be greater the closer the approximation is to a function isopleth. Thus the partitioning strategy will be most effective when the function isopleths tend to be convex over the bulk of the integration region and somewhat less effective to the extent that this is not the case. (It should be noted that this generally *is* the case for the examples of the previous section.) As with most numerical integration methods, this method has difficulty with highly oscillatory integrands.

An important consequence of adopting the spread as an indication of difficulty is that one is less likely to be deceived into thinking that an integral estimate is accurate when it is not. Undersampling can cause both the integral and its associated error estimate to be seriously undervalued. This tendency is more pronounced the more difficult the problem. Owing to the fact that the estimation of the spread does not rely on sampling, it is less vulnerable to this problem.

The memory requirement associated with the method is not severe. For each of the M hyperrectangular regions, one must store the spread measure S_i (3) and the region boundaries x_i^U, x_i^L. The boundaries can be arranged in a binary tree requiring $3M - 2$ integers and $M - 1$ real quantities. Thus, in all, storage for $3M - 2$ integers and $2M - 1$ real numbers is required. For the examples of the previous sections, the largest number of regions was $M = 388$. Storage for several thousand regions could easily be accommodated on the most medium- to large-scale computers.

There are several avenues of investigation that are not addressed in this report. It has been assumed that the object function used to drive the partitioning was identical to the integrand. This is not a fundamental requirement and different choices may prove to be useful. For example, if several integrals are to be evaluated with similar integrands over the same region of integration, it might be that a partitioning based on one of them will be effective for integrating all of them. Many integration formulas have the property that they are exact for linear functions. Thus, within any region, the linear component of the integrand is exactly integrated and one would like a partitioning of the domain of integration, such that the range or spread of derivatives within each subregion is small. An object function of the form

$$f'(x) = \sum_{i=1}^{n} \left| \frac{\partial f(x)}{\partial x_i} \right| \qquad \text{or} \qquad f'(x) = \left\{ \sum_{i=1}^{n} \left[\frac{\partial f(x)}{\partial x_i} \right]^2 \right\}^{1/2}$$

might be useful for driving the partitioning in these cases. Another situation in which the integrand and object function might differ occurs when there are

factors or components in the integrand that are clearly oscillatory or unduly expensive to calculate. These factors could be replaced by average values or simple approximations for the purpose of determining the partitioning.

The possibility of using this partitioning method in conjuction with other adaptive methods might also be considered. Owing to its robustness, this procedure might be applied as the first stage of a combined procedure. In those subregions for which the spread measure is relatively large, one could apply an adaptive procedure based on sampling. The methods of Genz [6], LePage [15], and Kahaner and Wells [12] appear as good candidates for this combined application.

A FORTRAN program [5] implementing the nested refinement partitioning procedure described in this report, along with several numerical integration methods, is available from either author.

ACKNOWLEDGMENTS

Helpful discussions with David Carey, Fred James, and G. Peter LePage are gratefully acknowledged. We thank Eric Grosse for providing the graphical example. We also thank the referees for many helpful suggestions.

REFERENCES

1 ABRAMOWITZ, M., AND STEGUN, I.A. *Handbook of Mathematical Functions.* National Bureau of Standards Applied Mathematics Ser 55, NBS, Washington, D.C., 1964.
2. DAVIDON, W.C. Variable metric method for minimization. Rep. ANL-5990, Argonne Nat. Lab., Argonne, Ill., 1959.
3. DENNIS, J E , AND MORÉ, J J. Quasi-Newton methods, motivation and theory. *SIAM Rev 19* (1977), 46–89.
4 FLETCHER, R., AND POWELL, M.J.D. A rapidly convergent descent method for minimization. *Comput. J. 6* (1963), 163–168.
5. FRIEDMAN, J H , AND WRIGHT, M H User's guide for DIVONNE. Computation Res. Group Tech. Memo 193, Stanford Linear Accelerator Center, Stanford, Calif., 1979.
6. GENZ, A An adaptive multidimensional quadrature procedure. *Comput. Phys Commun. 4* (1972), 11–15.
7. GILL, P E , AND MURRAY, W. Quasi-Newton methods for unconstrained optimization. *J. Inst. Math. Appl. 9* (1972), 91–108.
8 GILL, P.E , AND MURRAY, W. Safeguarded steplength algorithms for optimization using descent methods. Rep. NAC 37, Nat. Phys. Lab., England, 1974.
9. GILL, P.E., AND MURRAY, W Quasi-Newton methods for linearly constrained optimization. In *Numerical Methods for Constrained Optimization*, P.E. Gill and W. Murray (Eds.), Academic Press, New York, 1974.
10 HALTON, J.H. A retrospective and prospective survey of the Monte Carlo method. *SIAM Rev. 12* (1970), 1–30
11. HALTON, J H , AND ZEIDMAN, E A The evaluation of multidimensional integrals by the Monte Carlo sequential stratification technique Tech. Rep. 137, Comput. Sci. Dep., Univ. of Wisconsin, 1971.
12 KAHANER, D K., AND WELLS, M.B. An experimental algorithm for N-dimensional adaptive quadrature. *ACM Trans. Math Softw. 5*, 1 (March 1979), 86–96.
13. KOROBOV, N M *Number-Theoretic Methods in Approximate Analysis* (in Russian), Goz. Izdat. Fiz.-Mat Lit , Moscow, MR 28#716.
14. LAUTRUP, B.E. An adaptive multidimensional integration technique In *Proc. Second Colloquium in Advanced Computing Methods in Theoretical Physics*, Centre National de la Recherche Scientifique, Marseille, France, 1971.
15. LEPAGE, G P A new algorithm for adaptive multidimensional integration. *J. Comput Phys 27* (1978), 192–203

16. MORÉ, J.J. The Levenberg-Marquardt algorithm: Implementation and theory. In *Numerical Analysis*, G.A. Watson (Ed.), Springer-Verlag, New York, 1977.
17. SASAKI, T. Multidimensional Monte Carlo integration based on factorized approximation functions. *SIAM J. Numer. Anal. 15* (1978), 938–952.
18. STROUD, A.H *Approximate Calculation of Multiple Integrals* Prentice-Hall, Englewood Cliffs, N J., 1971.

Received September 1977; revised June 1980; accepted September 1980.