On Numerical Solutions to One-Dimensional Integration Problems with Applications to Linear Light Sources

MARC J. OUELLETTE and EUGENE FIUME
University of Toronto

Many key problems in computer graphics require the computation of integrals. Due to the nature of the integrand and of the domain of integration, these integrals seldom can be computed analytically. As a result, numerical techniques are used to find approximate solutions to these problems. While the numerical analysis literature offers many integration techniques, the choice of which method to use for specific computer graphic problems is a difficult one. This choice must be driven by the numerical efficiency of the method, and ultimately, by its visual impact on the computed image.

In this paper, we begin to address these issues by methodically analyzing deterministic and stochastic numerical techniques and their application to the type of one-dimensional problems that occur in computer graphics, especially in the context of linear light source integration. In addition to traditional methods such as Gauss-Legendre quadratures, we also examine Voronoi diagram-based sampling, jittered quadratures, random offset quadratures, weighted Monte Carlo, and a newly introduced method of compounding known as a *difficulty driven compound quadrature*.

We compare the effectiveness of these methods using a three-pronged approach. First, we compare the frequency domain characteristics of all the methods using periodograms. Next, applying ideas found in the numerical analysis literature, we examine the numerical and visual performance profiles of these methods for seven different one-parameter problem families. We then present results from the application of the methods for the example of linear light sources. Finally, we summarize the relative effectiveness of the methods surveyed, showing the potential power of difficulty-driven compound quadratures.

Categories and Subject Descriptors: F.2.1 [**Analysis of Algorithms and Problem Complexity**]: Numerical Algorithms and Problems—*Computation of transforms*; G.1.0 [**Numerical Analysis**]: General—*Error analysis*; G.1.4 [**Numerical Analysis**]: Quadrature and Numerical Differentiation—*Adaptive and iterative quadrature*; *Error analysis*; *Gaussian quadrature*; *Iterative methods*; G.3 [**Probability and Statistics**]: *Probabilistic algorithms* (*including Monte Carlo*); I.3.0 [**Computer Graphics**]: General; I.3.7 [**Computer Graphics**]: Three-Dimensional Graphics and Realism

General Terms: Algorithms, Experimentation, Performance, Reliability

The financial support of the Natural Sciences and Engineering Research Council of Canada (NSERC), and of Communications and Information Technology, Ontario (CITO), is gratefully acknowledged. Our computer graphics laboratory also benefited from the generosity of Alias|Wavefront.

Authors' addresses: M. Ouellette, now at Alias|Wavefront, 210 King Street East, Toronto, Ont., Canada, M5A 1J7; e-mail: vv1@dgp.toronto.edu; E. Fiume, Department of Computer Science, University of Toronto, 10 King's College Road, Toronto, Ont., Canada, M5S 3G4.

Additional Key Words and Phrases: Difficulty-driven compound quadrature, integration, linear light sources, numerical quadrature, performance profile, periodogram, rendering, stochastic sampling, Voronoi sampling

1. INTRODUCTION

In computer graphics, many key problems require the solution of integrals for which analytical solutions are not always available or tractable. For these reasons, numerical techniques are needed to find approximate solutions. The choice of numerical integration technique, whether it be deterministic, or stochastic, must often be done on an ad hoc basis. In this paper, we present a novel approach to solving one-dimensional integration problems in computer graphics, namely, *difficulty-driven compound quadratures*. We then present the results of an in-depth study of this technique and of other current methods used to solve the types of one-dimensional integration problems that occur in computer graphics. This study demonstrates the power of the new technique, in the context of certain classes of integration problems, as well as the strengths and weaknesses of existing methods.

We will compare the effectiveness of these methods using a three-pronged approach. First, we will use a frequency domain analysis tool, the periodogram, to compare the frequency domain characteristics of the one-dimensional integration methods. We will then use a numerical analysis tool, performance profiles, to compare the performance of these methods over families of integration problems, with each family consisting of a set of one-dimensional integration problems with similar integration difficulties, and each family representing a type of difficulty encountered in computer graphics problems. Finally, we will compare the performance of the methods over four different types of linear light source integration problems, each representing a family of problems with a specific integration difficulty.

1.1 Numerical Integration in Computer Graphics

Before we can compare the relative performance of various integration methods, we need to take into account some considerations that are specific to computer graphics problems. While we do want to compare the numerical efficiency of these techniques, we must do so in the context of needing to achieve a solution that is visually acceptable. Toward achieving this goal, we will take into account three important factors contributing to this efficiency: the need for a moderately accurate solution, the need for a consistent solution, and the need for a low-cost solution.

From a numerical point of view, the most important measure of the quality of a solution to an integration problem is its numerical accuracy. In computer graphics, the most important feature of any solution to a rendering problem is the quality of the final rendered image. Quantifying this quality is an extremely difficult problem [Rushmeier et al. 1995]: Whereas one can measure the numerical difference between images, there is no universally accepted metric for measuring the perceptual difference between images. While quantifying perceptual difference is beyond the scope of the paper, we will, however, take into consideration the visual appearance of rendered images, such as the presence of noise or other unwanted artifacts, in our evaluation of numerical techniques.

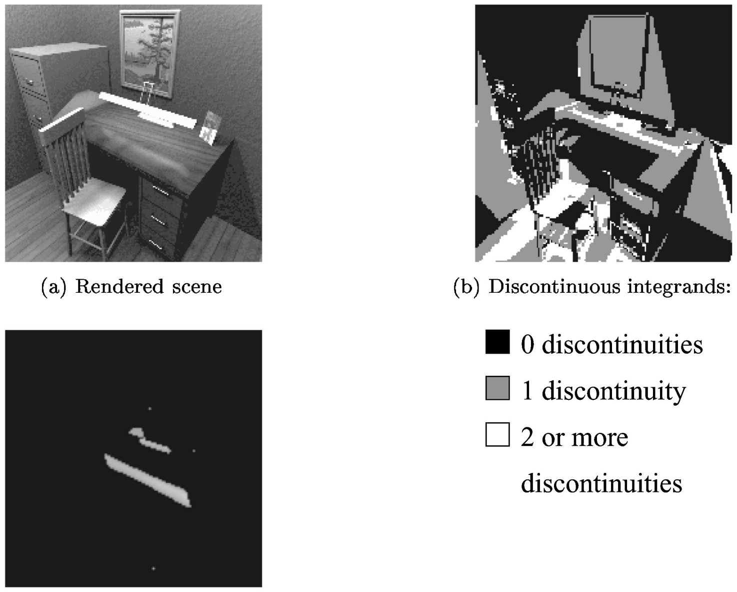
Because of this dichotomy of the definition of quality, the desired accuracy in computing an integrand can be quite moderate. For example, it is common to allow eight bits of accuracy for each of the three color components of a rendered image. This limited numerical accuracy suggests that, depending on the subsequent usage of the result, the accuracy of each integration problem may have to be only on the order of $\frac{1}{256}$. In general, numerical solutions in computer graphics only need to be moderately accurate, say, within a relative error from 10^{-1} to 10^{-3}.

In addition, numerical quality is not always positively correlated to visual quality: It is possible for an image with superior numerical accuracy to appear noticeably worse, visually, than an image with inferior numerical accuracy. Perhaps even more important than the numerical accuracy of a solution is the consistency of this accuracy. For example, if the numerical accuracy of an approximation technique varies in a periodic fashion, unwanted artifacts can become visible in a rendered image. In some cases, it is better, visually, to have less variance in numerical results, even at the expense of poorer average approximations. At the end of this section, we provide an example which demonstrates that results which are, in some sense, numerically equivalent, may be visually quite different.

One final consideration controlling the choice of integration method is the large number of integration problems that must be solved in the rendering of a single image. For direct illumination alone, we may easily have to solve millions of lighting integrals in order to achieve an acceptable image quality for even a moderately sized image. Given such a potentially large number of problems to solve, it would be unacceptable to use thousands of function evaluations to solve a single definite integration problem. We must therefore limit the number of function evaluations used to solve each problem; in this study, we have chosen to consider at most 300 function evaluations per integral.

1.2 Context of Work

As two-thirds of the analysis presented in this paper is presented in a general context, we would like to stress that this general analysis is necessary in several ways. First, the analysis leads to the the obvious application to the problem of computing the integral resulting from a linear light, as will be seen in Section 5. Next, there are other possible applications to one-dimensional integration problems, such as motion blurring [Meredith-Jones 2000], and light transport path integration in participating media [Pauly et al. 2000]. Finally, the analysis lays the ground work for a similar analysis of two-dimensional integration problems, such as the computation of area light sources, the filtering of textures, or antialiasing over a pixel.



(a) Rendered scene

(b) Discontinuous integrands:

■ 0 discontinuities
▨ 1 discontinuity
□ 2 or more discontinuities

(c) Integrands with a peak: the intensity of each pixel is proportional to the sharpness of the resulting peak in the integrand.

Fig. 1. Characteristic classification of a scene.

1.3 Examples

Finally, before we move on to the analysis sections, it is useful to present two examples; the first illustrates the type of one-dimensional integration problem that we will be investigating, whereas the second illustrates the difference between numerical and visual error.

1.3.1 *Integration Difficulties.* Consider the scene in Figure 1(a) (also shown at a higher resolution in Figure 28). It consists of an office environment illuminated by two linear light sources on the ceiling, and by a third linear light source in the desk lamp. In order to render this scene, we need to compute illumination integrals resulting from all three of these light sources, for a number of points on the objects in the scene. The resulting integrands will have different types of difficulties, depending on the geometry of the illumination problem, the nature of the surface, and the nature of the light source. If all three light sources are diffuse, and if we are only interested in computing the primary illumination for this scene, we can characterize the types of difficulty present in the resulting integrands as shown in Figure 1(b) and 1(c).

In Figure 1(b), the black regions represent areas that have a continuous integrand, because each light source is either fully visible or fully occluded. The grey regions represent areas where one or more of the light sources is partially hidden by another object in the scene, causing the resulting integrands to have precisely one discontinuity. Finally, the white regions represent areas of the scene where one or more of the resulting integrands has more than one

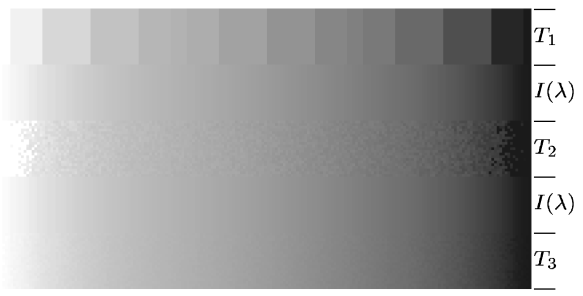


Fig. 2. Integration plots for step function $s(x, \lambda)$. For each of the five strips, the abscissa represents the parameter λ as it varies from 0 to 1, and the intensity of each pixel corresponds to the evaluation/approximation of $\int_0^1 s(x, \lambda)\, dx$ for the given value of λ.

discontinuity. The areas near the back of the chair result in integrands that have many discontinuities.

In Figure 1(c), the grey regions represent the areas of the scene where the integrand has a significant peak over the interval of integration. The brightness of each pixel is directly related to the sharpness of the peak. These regions correspond to those in Figure 1(a) that have significant highlights, such as the area at the front of the desk. The black pixels all result in integrands that are relatively smooth.

1.3.2 *Numerical vs. Visual Error.* Finally, we provide an example[1] that illustrates the difference between numerical and visual error. Consider the step function $s(x, \lambda)$,

$$s(x, \lambda) = \begin{cases} 0 & \text{if } x \le \lambda \\ 255 & \text{if } x > \lambda \end{cases} \quad \lambda \in [0, 1],$$

and its definite integral $I(\lambda)$ over the interval $[0, 1]$. The definite integral has an analytic solution, namely, $I(\lambda) = 255(1 - \lambda)$.

We computed this integral for 199 different values of λ, using three different numerical integration techniques (T_1, T_2, and T_3), giving three approximations, $I_1(\lambda)$, $I_2(\lambda)$, and $I_3(\lambda)$. We repeated these calculations 21 different times, using the same set of values for λ. Since the value of $I(\lambda)$ is bounded above by 255, it is straightforward to plot the results as a grey-scale intensity strip.

In Figure 2, we have shown the results in the following way. The image shown consists of five strips of 21 scanlines, with each scanline being 199 pixels wide. The ith pixel on a scanline corresponds to the parameter $\lambda_i = \frac{i-1}{198}$ and has an intensity corresponding to either the quantity $I(\lambda_i)$, or the approximation calculated by the appropriate technique.

The top 21 scanlines, as delimited by the tick marks on the right hand side and labeled T_1, correspond to the 21 different calculations of the approximation, each computed using technique T_1. The next 21 scanlines are identical, and correspond to the analytic solution $I(\lambda)$. The following 21 scanlines correspond

[1]This example will be revisited in Section 4.

to the 21 different approximations of $I(\lambda)$ using technique T_2. The next 21 scanlines are identical, and once again correspond to the analytic solution $I(\lambda)$. The last 21 scanlines correspond to the 21 different approximations of $I(\lambda)$ using technique T_3.

For the strips generated by T_1, T_2, and T_3, we calculated the average absolute error with respect to the analytical strip $I(\lambda)$. The average absolute error for each of the three approximate strips is 0.60, that is, each strip has the same numerical error. The T_1 strip was generated using a deterministic technique, and hence the 21 scanlines are identical. Visually, the error shows up as bands of uniform intensity, with a noticeable discontinuity between bands. The T_2 strip was generated using a space-filling technique. Notice that the scanlines are all different: this is due to the stochastic nature of the integration technique. The visual error appears as uncorrelated noise dispersed throughout the strip. Finally the T_3 strip was generated using a difficulty-driven compound quadrature.[2] Once again, the visual error appears as uncorrelated noise, but is less noticeable than for the T_3 strip. Thus, even though all three strips have the same numerical error, they all have different visual errors. The strip generated by T_3 is the less objectionable of the three approximate strips.

In the remainder of this paper, we combine ideas from the numerical analysis and computer graphics literature to analyze the effectiveness of a number of one-dimensional integration techniques, and their application to families of problems that occur in computer graphics image synthesis. In Section 2, we present difficulty-driven compound quadratures, a novel method of approximating integrals in computer graphics, and provide an overview of existing one-dimensional integration techniques that will be used in our analysis of one-dimensional integration problems. We then compare the effectiveness of these methods using three different approaches. First, in Section 3, we compare the frequency domain characteristics of all the methods using periodograms. Next, in Section 4, we apply ideas found in the numerical analysis literature [Lyness and Kaganove 1976], and examine the numerical and visual performance profiles of these methods for seven different one-parameter problem families, with each family representing a type of difficulty that can occur in computer graphics problems. In Section 5, we present results from the application of the methods to common one-dimensional rendering problems. Finally, in Section 6, we summarize the effectiveness of difficulty-driven compound quadratures, present the strengths and weaknesses of all the analyzed techniques, and suggest areas of future research.

2. THE NUMERICAL INTEGRATION METHODS

We begin by presenting the different one-dimensional numerical integration methods that we consider in the analysis found in this paper. While we can not possibly consider every possible numerical integration technique, we have chosen techniques that are representative of scientific computation. We have also

[2]The space-filling technique, which is Voronoi sampling, and the difficulty-driven compound quadrature will be presented in Section 2.

rejected methods which are not as well suited for solving the one-dimensional integration problems that we are examining in this paper.

In this paper, given an integrand f and a domain of integration $[a, b]$, we want to find an approximation to the definite integral of f over this domain; that, is we want to compute

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

We can approximate this integral by computing the weighted sum $Q(f)$, where

$$Q(f) = \sum_{i=1}^{n} w_i f(x_i). \tag{2}$$

The particular choice of the n point locations x_i, together with the choice of the n weights w_i, defines a numerical integration method known as a *quadrature rule* [Davis and Rabinowitz 1984; Kahaner et al. 1989].

A well-known method that we have decided to not examine in this paper is the (pure) *Monte Carlo* method. In such a method, the weights are all $1/n$, and the point locations are random variables distributed uniformly over the interval $[a, b]$. In one dimension, the convergence of such a method is only $O(n^{-1/2})$, whereas a convergence rate of $O(n^{-1})$ can be achieved with a deterministic quadrature rule, even if the function f has a discontinuity. Note that Bakhvalov's theorem [Shreider 1966] states that these orders cannot be improved, even if f has a bounded, continuous first derivative. Since the functions that we consider are not *very poor*, that is, they are either continuous or piecewise smooth, a pure Monte Carlo method is not indicated [Shreider 1966].

It should be noted, of course, that a pure Monte Carlo method may very well be acceptable with integration problems of greater dimensionality. We will, however, examine several stochastic techniques for approximating integrals, including variations of the Monte Carlo method such as jittered or stratified sampling. In the remainder of this section, we define the one-dimensional integration techniques that we will be examining, namely, regular compound quadratures, difficulty-driven compound quadratures, jittered quadratures, randomly offset quadratures, weighted Monte Carlo, and Voronoi sampling.

2.1 Regular Compound Quadrature Rules

As was stated in Eq. (2), a quadrature rule is an integration method defined by a set of n point locations and weights. If the point locations and weights are fixed, the method is a *deterministic* quadrature rule; otherwise it is a *stochastic* quadrature rule. A deterministic quadrature rule Q is said to be of *degree* d over the domain D if it integrates exactly all polynomials of degree d or less, but fails to integrate exactly some polynomial of degree $d + 1$.

In one dimension, the *Gauss* (also known as Gauss-Legendre) quadrature rule on n points is defined by the choice of n locations and n weights that gives the quadrature rule of highest possible degree. An n-point Gauss quadrature rule (Gn) has degree $2n - 1$ [Kahaner et al. 1989]. In this paper, we will study the Gauss quadrature rules on 1, 3, 5, 7, 9, 15, and 30 points.

Note that the quality of a quadrature rule often relies on the smoothness of the integrand over the domain of integration. Should a singularity or other nonsmooth behavior arise, the quality of the quadrature is cast into doubt.

A *compound* quadrature rule is obtained if we subdivide D into *panels* and apply a quadrature rule to each panel. The first type of compounding that we shall study is *regular compounding*, in which for a given problem, the interval of integration is subdivided into panels of equal size. Notice that a regular compound quadrature rule with G1 applied to each panel is often referred to as *supersampling* in the computer graphics literature.

2.2 Difficulty-Driven Compound Quadratures

Traditionally, the choice of the number of panels in a compound quadrature provides something of a divide-and-conquer integration approach. It also allows one to choose panel sizes and locations that can isolate a difficulty in the integrand or can take advantage of the dispersion of quadrature sample points to get better visual results for the same number of points. We shall exploit both possibilities in the new technique we shall be presenting.

Suppose that we have a function to integrate over the interval [0, 1], and that this function has an integration difficulty at a location $\lambda \in [0, 1]$. For example, this difficulty could be a peak, a discontinuity, or a singularity. If we know the approximate location of λ, we should be able to exploit this knowledge to obtain a more efficient numerical integration technique and a more visually reliable solution.

Let $\tilde{\lambda}$ be an estimate of the location of λ, and let $P \geq 2$ be the total number of panels desired for a *difficulty-driven compound quadrature* rule. We subdivide the integration interval at $\tilde{\lambda}$ into two subintervals of integration, and apply regular compounding to each of the resulting subintervals. We define P_0, the number of panels in $[0, \tilde{\lambda}]$, and P_1, the number of panels in $[\tilde{\lambda}, 1]$, such that the number of panels in each subinterval is proportional to its length, and each subinterval has at least one panel. Specifically,

(1) $P_0 = \max(P - 1, \min(1, \lfloor \frac{1}{2} + (\tilde{\lambda} P) \rfloor))$,

(2) $P_1 = P - P_0$.

For a variety of problems, we will examine the effects of difficulty-driven compound quadratures that are based on the approximate location of the difficulty $\tilde{\lambda}$. In particular, given an uncertainty ξ, we will approximate the location of the difficulty by randomly choosing the value of $\tilde{\lambda}$ from a uniform distribution over the interval $[\lambda - \xi, \lambda + \xi]$. When $\xi = 0$, we would be using exact knowledge of the location of the difficulty.

In the problem families of Section 4, the integrands are defined such that the location of the difficulty λ is known. In the integrands encountered in Section 5, the location of λ must be determined using heuristics. The specifics of these heuristics is beyond the scope of this paper, but the interested reader can find extensive details in Ouellette and Fiume [1999b] and Ouellette [2002].

2.3 Jittered Quadratures

Although the idea of jittering in itself is not new to computer graphics [Dippé and Wold 1985; Cook 1986; Shirley 1991], the theory and application of jittering to numerical quadratures is novel. Given an arbitrary (deterministic) quadrature rule, we generate a (random) *jittered* quadrature rule by perturbing the location of each sample. While these jittered quadrature rules are necessarily suboptimal numerically, they can be useful in an undersampled context (e.g., when the integrand is dominated by high-frequency content). The interested reader can refer to Shreider [1966] for a brief introduction to the general concept of random quadratures and for pointers to additional references.

Let Q be an n-point quadrature rule, as defined in Eq. (2). Let V_i be the *Voronoi cell* associated with x_i, that is:

$$V_i = \{x \in [a,b] \mid |x - x_j| \leq |x - x_i|, \text{for all } j \in [1,n]\}. \tag{3}$$

We define the location jittered quadrature J to be

$$J(f) = \sum_{i=1}^{n} w_i f(u_i), \tag{4}$$

where u_i is a uniformly distributed random variable in V_i. We also define the partial location jittered quadrature J_p to be

$$J_p(f) = \sum_{i=1}^{n} w_i f((1-p)x_i + pu_i), \tag{5}$$

where $p \in [0,1]$. Notice that $J_0 = Q$ and that $J_1 = J$. We will use the abbreviation Jn to refer to the fully jittered quadrature rule based on Gn, the n-point Gauss quadrature rule. Notice also that if Q is the Gauss quadrature with 1 point and regular compounding, then J1 is the well-known jittered (or stratified) sampling.

2.4 Random Offset Quadratures

An alternative to jittering every point of a quadrature is to add a common offset to all point locations, giving in effect, a *random offset* quadrature [Pauley et al. 2000]. Once again, let Q be an n-point quadrature rule, as defined in Eq. (2). Let ξ be chosen randomly from a uniform distribution over the interval $[a,b]$. We then define the random offset quadrature rule R to be

$$Q(f) = \sum_{i=1}^{n} w_i f(\tilde{x}_i), \tag{6}$$

where

$$\tilde{x}_i = \begin{cases} \xi + x_i & \text{if } \xi + x_i \leq b, \\ \xi + x_i - b + a & \text{otherwise.} \end{cases} \tag{7}$$

In Pauly et al. [2000], the quadrature Q is the 1-point Gauss quadrature rule, and regular compounding is used. In the remainder of this paper, we shall only consider this particular version of random offset quadratures, and the term *random offset quadrature* will refer to this version specifically. The abbreviation Rn will be used to denote a random offset quadrature rule (based on G1) regularly compounded with n panels.

2.5 Weighted Monte Carlo

The *weighted Monte Carlo* method [Yakowits et al. 1978] is a variant of the pure Monte Carlo method wherein the points are chosen in a similar fashion, but the weights of a point vary according to the distance to its nearest neighbor(s). In order to construct an $n > 1$-point weighted Monte Carlo method, $n - 2$ point locations are chosen randomly from a uniform distribution over the interval $[a, b]$. In addition, the end points a and b are used as sample points. If we let $x_1, x_2, \ldots, x_n$ be the resulting n sample points sorted in increasing order, then the weight w_i assigned to each point is

$$\begin{aligned} w_1 &= \frac{(x_2 - a)}{2(b-a)}, \\ w_i &= \frac{(x_{i+1} - x_{i-1})}{2(b-a)} \quad \text{if } 1 < i < n, \\ w_n &= \frac{(b - x_{n-1})}{2(b-a)}. \end{aligned} \tag{8}$$

In Yakowits et al. [1978], this method is shown to have a convergence of $O(n^{-4})$ when the function $f(x)$ has a continuous second derivative. Note, however, that many of the one-dimensional integration problems that are encountered in this paper, and in computer graphics in general, will not meet this requirement (e.g., a discontinuous integrand). The abbreviation Wn will be used to denote a weighted Monte Carlo method on n points.

2.6 Voronoi Sampling

The final method that we will examine is *Voronoi sampling* [Ouellette and Fiume 1999b; Deussen et al. 2000]. It is a method similar to the quasi-random methods described in Spanier and Maize [1994], since the sequence of points generated is *uniformly distributed*, that is, it ultimately assigns to each subinterval a number of points proportional to its length. Given an interval $[a, b]$, we construct a Voronoi sampling sequence by first randomly choosing a sample point x_1 from a uniform distribution over (a, b). We then choose sample points iteratively by constructing the Voronoi cells corresponding to the location of the sample points in the existing sequence, and augmenting the sequence by the midpoint of the largest cell.

Formally, let $x_{-1} = a$, let $x_0 = b$, and let $x_1, x_2, \ldots, x_{n-1}$ be the first $n - 1$ points of the Voronoi sampling sequence. Let V_i be the Voronoi cell of x_i; that is,

$$V_i = \{x \in [a, b] \mid |x - x_j| \leq |x - x_i|, \text{for all } j \in [-1, n]\}, \tag{9}$$

for $i \in [-1, n]$. It is clear that each cell V_i is a line segment. Let V_M be the longest such line segment, with the proviso that ties are broken randomly.

Then the next sample point in the Voronoi sequence is m_M, the midpoint of the line segment corresponding to V_M. In our implementation, the points are all given the same weight. The abbreviation Vn will be used to denote a Voronoi sampling sequence of n points.

We now define some general properties of Voronoi sampling that will be needed in Section 4. Consider a specific instance of a Voronoi sampling sequence over the interval [0, 1]. Once the initial seed σ has been chosen, we can determine precisely how many points will be chosen to the left and to the right of this seed. If $\sigma \in [1/3, 2/3]$, then it is easy to see that the Voronoi sequence must be built by alternately subdividing the left range $[0, \sigma]$ and the right range $[\sigma, 1]$.

The initial subdivision of a range generates one point, namely, the midpoint between the seed and the appropriate endpoint of the range. The nth subdivision adds 2^n points to the range, which are simply the midpoints between those points produced by the preceding subdivision. If not all 2^n points need to be added to attain the desired number of points, the points to be added are chosen randomly. We say that a subdivision is *full* if all points have been added, and that it is *partial* otherwise. For example, if a Voronoi sequence is generated from a seed $\sigma \in [1/3, 2/3]$, then such a sequence will be full with both 191 and 255 points, but will be partial at all values in between. Similarly, if $\sigma \in [1/5, 1/3] \cup [2/3, 4/5]$, then the resulting subdivisions will be full with both 159 and 191 points, but will be partial at all values in between.

3. FREQUENCY DOMAIN ANALYSIS

It is common to study sampling strategies using frequency domain analysis. It is less common to do the same for quadrature schemes, despite their evident connection to sampling. In this section we compare the integration methods by examining their characteristics in the frequency domain [Ziemer and Tranter 1990]. To do this, we treat both deterministic and stochastic methods as sampling patterns over the interval [0, 1], and assume that each pattern is made periodic by tiling the unit interval over $\mathbb{R}$. We will use a frequency analysis tool called the *periodogram* to compare all methods. A full derivation of the definition of the periodogram is given in the Appendix.

Given a set of N points $\{x_j\}$ together with their weights w_j, for $j = 1 \ldots N$, we define the sampling pattern $s(x) : [0, 1] \mapsto \mathbb{R}$ to be

$$s(x) = \sum_{j=1}^{N} w_j \delta(x - x_j). \tag{10}$$

Here, δ is the Dirac delta function, and $s(x)$ is made periodic by tiling the unit interval over the entire real line. Since $s(x)$ is real and periodic, it is equivalent to its Fourier Series representation. That is,

$$s(x) = \sum_{k=-\infty}^{\infty} c_k e^{i2\pi kx}, \tag{11}$$

where the Fourier Series coefficient c_k is

$$c_k = \sum_{j=1}^{N} w_j e^{-i2\pi k x_j}. \tag{12}$$

Since the sampling function $s(x)$ is periodic, its *spatial autocorrelation function* $\mathcal{R}(\tau)$ is defined to be

$$\mathcal{R}(\tau) = \int_0^1 s(x)s(x+\tau)\,dx. \tag{13}$$

The Fourier Series coefficients of the autocorrelation function are equivalent to the magnitude squared of the Fourier Series coefficients of the original signal $s(x)$. The plot of the Fourier Series coefficients of $\mathcal{R}(\tau)$ with respect to the frequency k is called a *periodogram*. We obtain the periodogram of $s(x)$ by plotting $\|c_k\|^2$ as a function of the frequency k. Because $s(x)$ is a real function, $\|c_k\|^2 = \|c_{-k}\|^2$, and therefore the periodogram of $s(x)$ is symmetric about the origin.

We use the periodogram to analyze the energy in the autocorrelation function of the sampling pattern corresponding to each integration method. For instance, power closer to the origin represents low-frequency energy in the autocorrelation function and long-distance spatial correlation. For sampling patterns, low-frequency energy is responsible for disturbing visual artifacts known as *aliasing*, which is manifested as coherent replicated noise patterns. When energy is dispersed equally in all frequencies, the resulting incoherent distribution of error is called *white noise*. The pure Monte Carlo method is an example of a sampling pattern with a white noise characteristic. A *blue noise* sampling pattern, as defined by Ulichney [Ulichney 1987], has the characteristic of a low noise floor in the passband, with a higher, well-distributed noise floor in the high frequencies. The resulting errors in an image tend to be less visually objectionable, that is, there is less coherent aliasing for high-frequency signals. The Voronoi sampling technique is an example of a pattern with a blue noise frequency characteristic.

We now present periodograms for the various one-dimensional numerical integration techniques. On each periodogram, the abscissa is the frequency k, and the ordinate is the corresponding value of the coefficient $\|c_k\|^2$. Note that given a sampling rate of $2N$, *Shannon's Sampling Theorem* states that a function can be reconstructed exactly provided that the function is 0 outside the frequency interval $[-N, N]$ [Pratt 1991]. On each periodogram, we will denote the Nyquist frequency as N, where $2N$ is the number of samples used by the methods being illustrated. In our analysis, we will calculate the *average periodogram* from 1000 instances of periodograms for a particular random process.

In Figure 3, we illustrate the periodograms for the deterministic quadratures G3 through G30. The ranges for all periodograms were truncated to include up to three times the Nyquist frequency. Outside of the plotted range, the periodograms for all of the Gauss quadratures are jagged. Notice that all the quadrature rules have a band of low frequencies with low energy that is delimited by the Nyquist frequency.

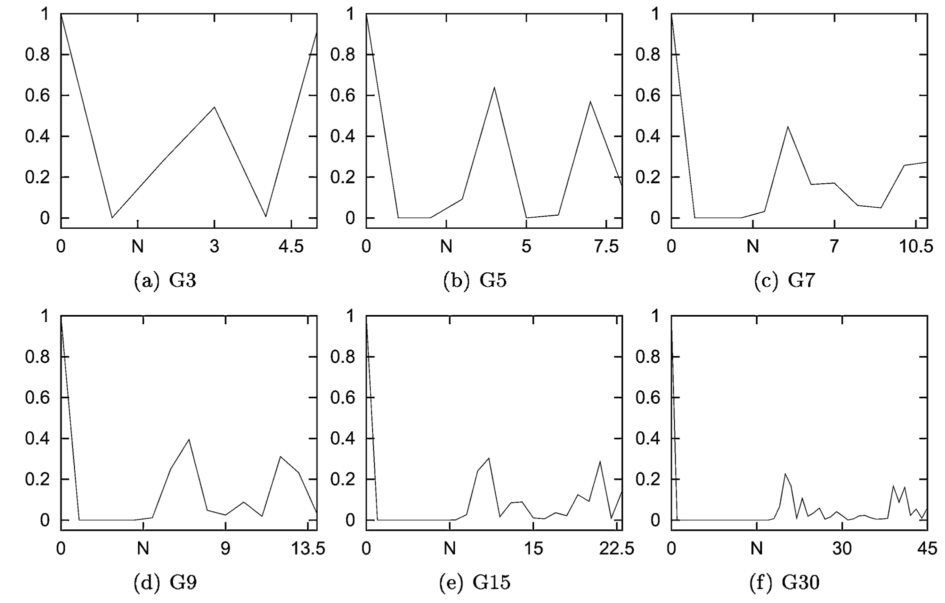
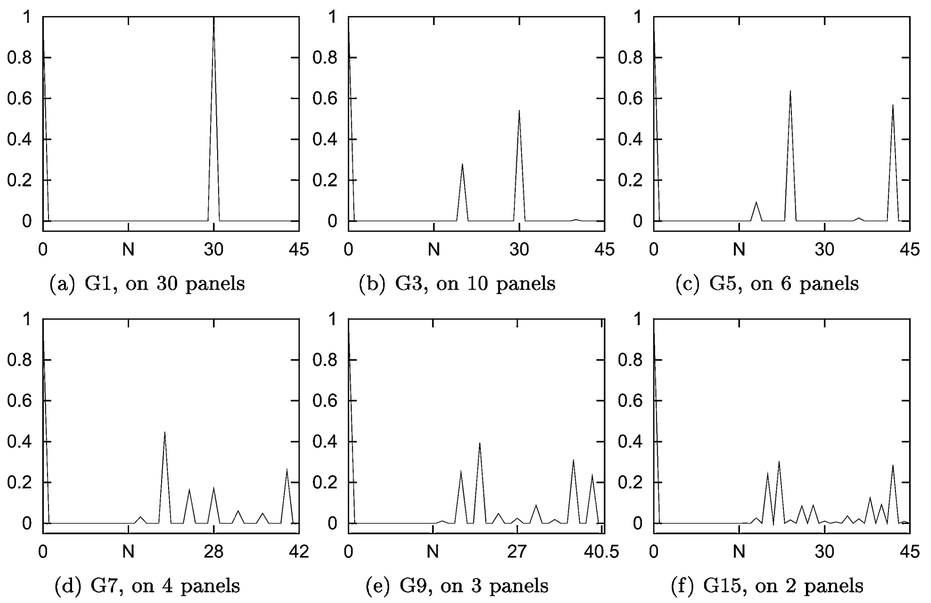


Fig. 3. Periodograms for deterministic quadratures. The abscissa is the frequency k, and the ordinate is the magnitude squared of the Fourier Series coefficient, that is, $\|c_k\|^2$. The Nyquist frequency is denoted by the symbol N on the abscissa. Note that these are also the periodograms for the random offset quadrature based on the given quadrature rule.

In Figure 4 we compare the effects of regular compounding with deterministic Gauss quadratures. Each Gauss rule was compounded a number of times to bring the total number of point samples as close as possible to 30. This can be done exactly for G1, G3, G5, G15, and G30. Four panels were chosen for G7, for a total of 28 points; three panels were chosen for G9, for a total of 27 points. Note that since G30 already uses 30 points, the resulting periodogram is that of Figure 3(f). Observe that the 30-point compound quadrature based on G1 has a low-frequency band that is twice as large as for any of the other compound quadratures, extending all the way to twice the Nyquist frequency. On the other hand, the height of the spike in G1 is much higher than in any of the other quadrature rules. In general, the height of the spikes decreases with increasing degree of the quadrature. For all of the compound quadrature rules, the energy fluctuates greatly in the frequencies above the low-frequency band.

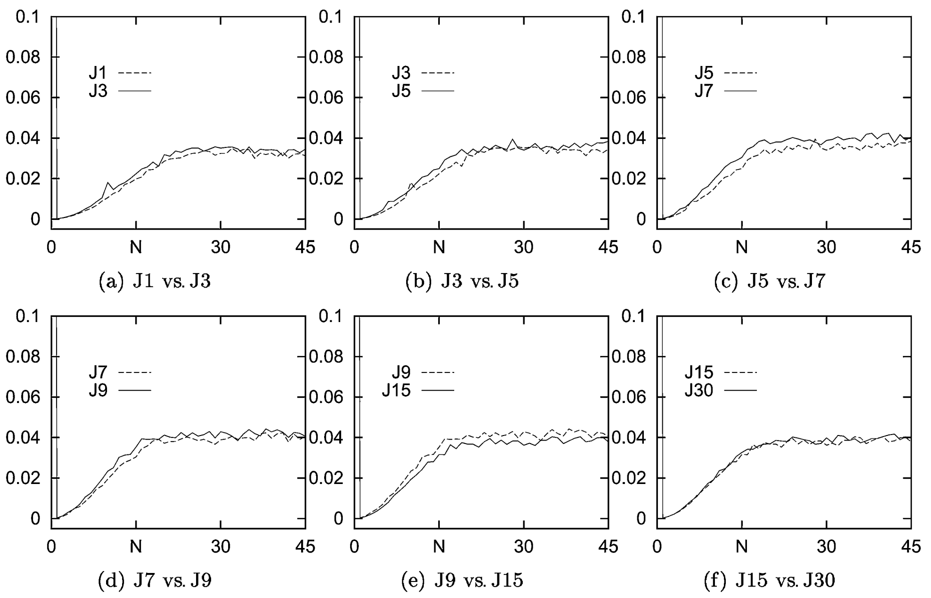
In Figure 5, we compare all of the compound quadrature techniques with full location jittering. On each periodogram, we compare consecutive pairs of jittered compound quadratures J1 through J30. For clarity, the vertical range has been truncated at 0.1 instead of 1. Notice that all of the quadrature rules exhibit the blue noise response to some degree, with attenuation of the low frequencies and evenly distributed energy in the higher frequencies. All of the quadrature techniques have a similar noise attenuation zone in the lower frequencies. With the exception of J7 and J9 (the two methods that use fewer than 30 points), the height of the noise floor is directly related to the number of points in each quadrature rule, and is at its lowest for J1.



(a) G1, on 30 panels (b) G3, on 10 panels (c) G5, on 6 panels

(d) G7, on 4 panels (e) G9, on 3 panels (f) G15, on 2 panels

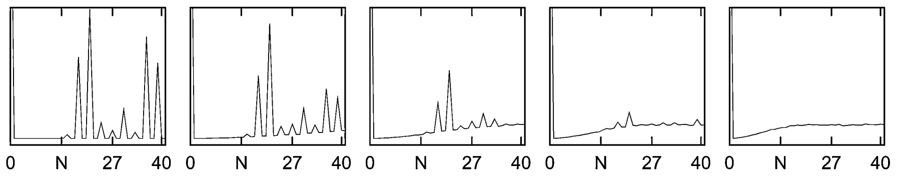
Fig. 4. Periodograms for regular compound quadratures, with the same deterministic technique being applied to each panel. Note that these are also the periodograms for the random offset quadrature based on the given quadrature rule, with the given compounding factor.



(a) J1 vs. J3 (b) J3 vs. J5 (c) J5 vs. J7

(d) J7 vs. J9 (e) J9 vs. J15 (f) J15 vs. J30

Fig. 5. Periodograms for compound jittered quadratures, with approximately 30 points.



(a) $J9_{0.00}$ (G9) (b) $J9_{0.25}$ (c) $J9_{0.50}$ (d) $J9_{0.75}$ (e) $J9_{1.00}$ (J9)

Fig. 6. Average periodograms for jittered 9-point Gauss quadrature. Ordinate is truncated at 0.4, and jitter varies from 0 to 1, by increments of 0.25.

If we compare the periodograms of the deterministic compound quadratures in Figure 4 to their jittered counterparts in Figure 5, we can notice the following:

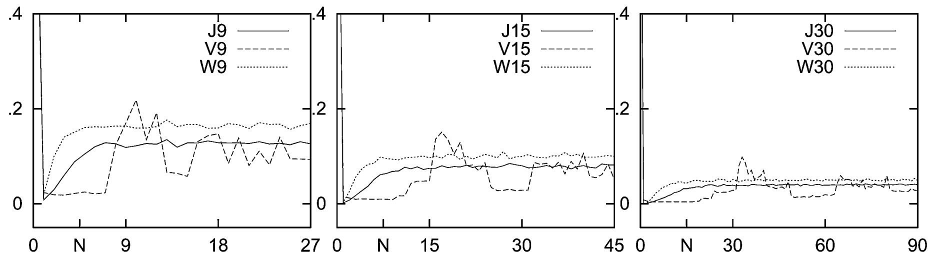
—Each deterministic quadrature has a lower and flatter noise floor than its jittered counterpart.
—Beyond the low-energy band, the energy in each jittered quadrature is nearly constant, but oscillates greatly in the deterministic counterpart.

It is informative to take a closer look at how gradual jittering affects the periodogram of a quadrature rule. Since the effect is similar for all quadratures, we will only present the results for the compound quadrature based on G9. In Figure 6 we have shown the periodogram for the jittered quadrature with jitter values 0, 0.25, 0.5, 0.75, and 1, with an abscissa ranging from 0 to 3 times the Nyquist frequency, and the ordinate ranging from 0 to 0.4, for clarity. The periodograms in Figure 6 illustrate what happens when we progressively jitter the location of the points for G9. There are a few interesting observations to be made from this figure:

—The nonjittered quadrature (0.00) has no power in the lowest frequencies, but has energy that oscillates in all other frequencies.
—As the jitter is increased from 0.25 to 1.0, the power in the noise floor increases slightly and the power in the higher frequencies becomes more uniform.
—All of the jittered quadratures exhibit the blue noise characteristic to some degree.

As we will see in later sections, the practical effect of jittering will be to reduce the aliasing that would result from the application of G9 to a difficult integral, but to introduce noise where an application of G9 to a simpler integral would have resulted in a correct numerical solution.

By definition, the sampling pattern generated by a random offset quadrature is simply a shifted version of the sampling pattern generated by the original quadrature rule. For example, a random offset quadrature R30 (i.e., based on G1 compounded with 30 equal panels) is a shifted version of G1 regularly compounded with 30 panels. Since Fourier transformations are shift invariant, both R30 and G1 compounded with 30 panels have exactly the same periodogram, namely, Figure 4(a). Therefore, the periodograms for the random



(a) 9 points (b) 15 points (c) 30 points

Fig. 7. Periodograms for weighted Monte Carlo and Voronoi sampling, compared to jittered quadratures, with ordinate truncated at 0.4.

offset quadratures based on G3 through G30 are found in Figure 3, and the periodograms for the random offset quadratures based on G1 through G15, with regular compounding, can be found in Figure 4.

In Figure 7 we illustrate the periodograms for a weighted Monte Carlo method and a Voronoi sampling, for 9, 15, and 30 samples, respectively. For comparison purposes, we have included the jittered periodogram of J9, J15, and J30, respectively. Each periodogram includes up to six times the Nyquist frequency and for clarity, the ordinate has been truncated at 0.4. Notice that both the Voronoi and the weighted Monte Carlo method exhibit a blue noise response first observed in the jittered quadrature periodograms of Figure 5. The Voronoi samplings all exhibit a blue noise response, with attenuation of the low frequencies up to and slightly beyond the Nyquist frequency and evenly distributed energy in the higher frequencies. This method also has a midfrequency peak (at about twice the Nyquist frequency), a characteristic of Poisson Disk [Cook 1986; McCool and Fiume 1992] sampling distributions. Note that the attenuation is also present in both the jittered quadrature and weighted Monte Carlo periodographs, but to a lesser degree in each, respectively.

4. PERFORMANCE PROFILES

One of the many difficulties in comparing the effectiveness of quadrature rules in solving integration problems is that for each technique, there are problems for which it will work perfectly, and problems for which it will not work at all. As is pointed out in Lyness [1981], it is possible for a technique to perform admirably on one problem, only to fail completely for a slightly different problem. In computer graphics, where inconsistent results can cause visual artifacts in rendered images, the behavior of a particular technique on a family of integrands is as important as its behavior on a particular member of such a family.

The idea of a *performance profile*, as taken from the numerical integration literature [Lyness and Kaganove 1976; Lyness 1981], is to test a one-parameter set of numerical problems called a *problem family*, examining some aspect of the performance of one or more quadrature rules. The performance profile is then a plot of a chosen aspect of the performance, the abscissa being the varying parameter, conventionally denoted by λ.

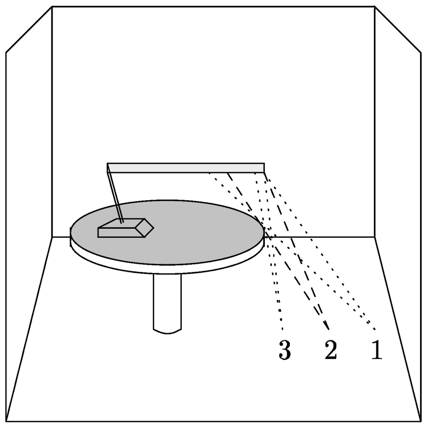


Fig. 8. Simple scene illustrating a family of integration problems with a discontinuity as a common difficulty.

In this section, we define seven different families of problems that each possess a difficulty that can be encountered in one-dimensional integration problems in computer graphics. We then define two different types of performance profiles, and use these tools to analyze the seven families of problems.

4.1 Problem Families

Before we define the seven families of one-dimensional integration problems that we will be analyzing, it is useful to present an example corresponding to a well-known integration problem in computer graphics. Consider a scene consisting of a round table, a fluorescent lamp on this table, and a floor, as shown in Figure 8. As we move along the floor from point 1, to point 2, to point 3, we are able to see a progressively smaller portion of the lamp. The problem of determining the portion of the light that is visible from neighboring points on the floor corresponds to the problem of finding the location of a jump discontinuity in an integrand.

The location of the jump discontinuity is defined by the location of the edge of the table as it blocks the view of the light source, and is different for each instance of the visibility problem.[3] Thus these visibility problems form a family of similar yet distinct one-dimensional integration problems. The problem of finding these discontinuities has been previously examined, notably in Ouellette and Fiume [1999b] and Heidrich et al. [2000] for linear light sources, and in Hart et al. [1999] and Ouellette and Fiume [1999a] for area light sources.

We define a problem family $\mathcal{F}$ as a set of functions to be integrated, each with similar difficulties such that

$$\mathcal{F} = \{f(\lambda, x) \mid \lambda \in [L, H]\}. \tag{14}$$

[3]The computation of a *discontinuity mesh* isolates continuous domains of integration, but their computation is prohibitive for complex scenes or scenes with nonpolygonal objects [Drettakis and Fiume 1994; Stewart and Ghali 1994].

Table I. Problem Families and Difficulties

Family	$[L, H]$	Difficulty	Rendering Problem Example
1	$[0, 1]$	C^2 discontinuity at λ	C^2 discontinuity in a shadow, used as a secondary light source
2	$[0, 1]$	C^1 discontinuity at λ	C^1 discontinuity in a shadow, used as a secondary light source
3	$[0, 1]$	C^0 discontinuity at λ	Linear light source partially hidden
4	$[0, 1]$	Oscillating function	High-frequency texture used as a light source
5	$[0, 1]$	Peak at λ	Specular reflection from a glossy surface
6	$[1, 2]$	Sharp peak at λ	Specular reflection from a very glossy surface
7	$[0, 1]$	Weak singularity at λ	Form factors between adjoining surfaces

The parameter λ, chosen from the domain $[L, H]$, controls the location of the difficulty. For each problem family, the domain of the parameter λ coincides with the limits of integration, that is, we will want to compute definite integrals of the form

$$I(\lambda) = \int_L^H f(\lambda, x)\, dx. \tag{15}$$

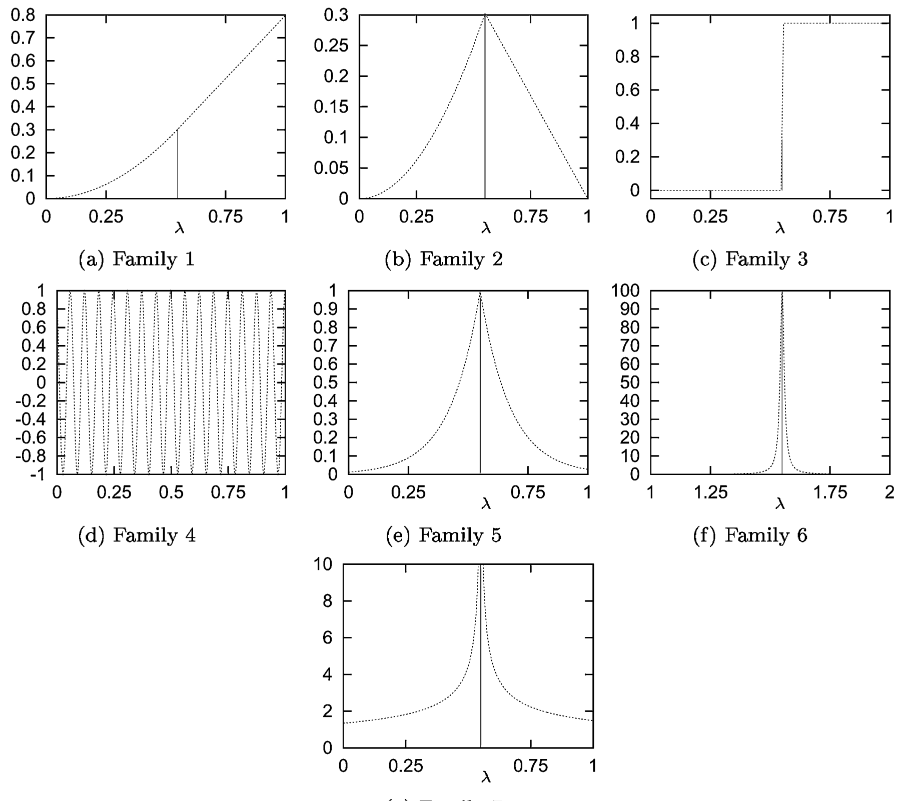
The problem families we study all have analytic solutions and all have distinct difficulties that are related to problems encountered in computer graphics. A list of problem families, difficulties, and an examples of corresponding rendering problems can be found in Table I, and a sample member from each family is shown in Figure 9. For each family, the actual definition of $f(\lambda, x)$ is given further on in this section.

For each problem family, we create similar yet distinct integration problems by varying the parameter λ. The purpose is to determine the behavior of each integration method on a set of these distinct problems, instead of on just one particular member of a family. This allows us to evaluate the consistency of the performance of each method on the entire family. From a visual point of view, such tests can provide valuable insight, as slowly varying the location of the parameter λ corresponds to varying the location of a difficulty in the computation of an image.

4.2 Performance Profiles

In this paper, for each of the chosen problem families, we will study two different aspects of the performance of the integration methods. Since each problem family has an analytical solution, we can easily compare a numerical approximation to the exact solution. The two aspects that we will examine are the numerical and the visual convergence of the approximate solutions. To study both aspects of the performance, it is useful to prescale the problem families such that the results can be readily interpreted. In our tests, each problem family is prescaled such that the integration of any of its members results in a value in the range $[0, 255]$. This results in values that are suitable for plotting as a grey-scale intensity.

Since each family is prescaled to achieve a maximum definite integral value of 255 within the domain $[L, H]$, an absolute tolerance of ϵ corresponds to a



(a) Family 1 (b) Family 2 (c) Family 3

(d) Family 4 (e) Family 5 (f) Family 6

(g) Family 7

Fig. 9. Plots of a sample member from each problem family, with $\lambda = L + 0.55$. Family 1: C^2 discontinuity; family 2: C^1 discontinuity; family 3: C^0 discontinuity; family 4: oscillating function; family 5: function with a peak; family 6: function with a sharp peak; family 7: function with a weak singularity.

relative error of $\epsilon/255$. The absolute tolerance that will most often be used is $\epsilon = 0.5$, since this implies exact solutions for images that only require 8 bits of accuracy per color. Notice that an absolute error of 0.5 corresponds to a relative error of 0.2%.

4.2.1 *Numerical Convergence.* The first aspect that we examine is the numerical convergence of each technique as a function of the number of function evaluations used by the technique. For quadrature rules, this is done using an increasing number of panels resulting from either a regular or difficulty-driven compound rule. The same quadrature rule is used over each panel. For the Voronoi and the weighted Monte Carlo methods, an equivalent number of function evaluations is used.

To define the numerical convergence over a problem family, we subdivide the domain $[L, H]$ into n segments, and for each $\lambda = L + \frac{j}{n}(H - L)$ such that $j \in \{0, 1, \ldots, n-1\}$, we compute the integral $I(\lambda)$. We thus evaluate each technique for n different members of a problem family. For a given absolute tolerance ϵ, we will plot the percentage of members that were integrated within ϵ as a function of the total number of function evaluations used to approximate the integrals. A method has achieved *100% numerical convergence* with respect to ϵ if all of the members are integrated within ϵ. On all the convergence plots, the abscissa is the number of function evaluations, and the ordinate is the percentage of convergence.

Note that our definition of numerical convergence is closely related to the information-based ϵ-complexity discussed in Traub et al. [1988]. In essence, we have defined a class of problems, and we want each problem from that class to be solved within a tolerance of ϵ. Achieving 100% numerical convergence is akin to finding the cost, in terms of function evaluations, of the worst case ϵ-complexity of the family. In other words, the worst case ϵ-complexity of a problem family is the cost associated with solving the member of the family that requires the greatest number of function evaluations in order to solve it. Similarly, achieving a lower convergence rate, say, $100\% - \delta$, is akin to finding the probabilistic ϵ-complexity of the problem, that is, finding the number of function evaluations required so that all but at most a fraction δ of the problems have been evaluated within a tolerance of ϵ.

4.2.2 *Visual Convergence.* We will examine the visual convergence of each numerical integration technique by plotting the result of the integration of the n different members of a problem family. Since the families were prescaled such that these values range between 0 and 255, we can display them as an 8-bit grey-scale intensity line. An obvious but important observation is that given a choice of members to integrate, the resulting intensity line will be uniquely defined if the technique is deterministic. If the technique is stochastic, each application of the technique results in a potentially different intensity line.

In computer graphics, integrands are often highly correlated, and depending on the type of solution employed to approximate each integral, different visual characteristics will emerge for different types of solution. Typically, approximate solutions derived from deterministic techniques may be prone to banding or aliasing, where as those derived from stochastic techniques may instead be prone to noise. For these reasons, it is important to study the visual convergence of several intensity lines for a given problem family.

For a given number of function evaluations, we examine the visual convergence of each technique by defining an *intensity strip* consisting of a catenation of intensity lines. Given a set of n different members (to be integrated) from a problem family, and a desired number of intensity lines h, we define a strip[4] of width n and height h. This strip is an array of values $S[i, j]$ such that the entry corresponding to $[i, j]$ is the jth computation of $I(L + \frac{i}{n}(H - L))$, using a given numerical integration technique. If the technique is deterministic, $S[i, j] = S[i, k]$ for any value of j and k, and the intensity lines are all identical. If the technique is stochastic, $S[i, j]$ and $S[i, k]$ are not necessarily the same, and we can potentially have h different intensity lines.

We define the *true strip* to be the intensity strip generated by the analytical solution of the same set of integration problems. Since the analytical solution

[4]In our study, we generate strips of width $n = 199$ and of height $h = 21$.

is deterministic, the true strip will consist of a catenation of identical intensity lines. To examine the visual convergence of different numerical integration techniques, we generate an intensity strip for each numerical technique. We then create an image by catenating these intensity strips, with a true strip inserted between every numerically generated intensity strip. This enables us to compare the visual convergence of the various techniques by comparing their strip to the true strip. We will say, informally, that a numerical method has *converged visually* if its corresponding intensity strip is perceptually nearly indistinguishable from the true strip. In all visual profiles, the true strip will appear, unlabeled, between each pair of intensity strips generated by a numerical integration technique.

4.3 Performance Profiles of Test Families

In this section we examine the performance profiles resulting from the application of the previously introduced numerical methods to seven different problem families. For each family, we will present the most relevant numerical and visual results.

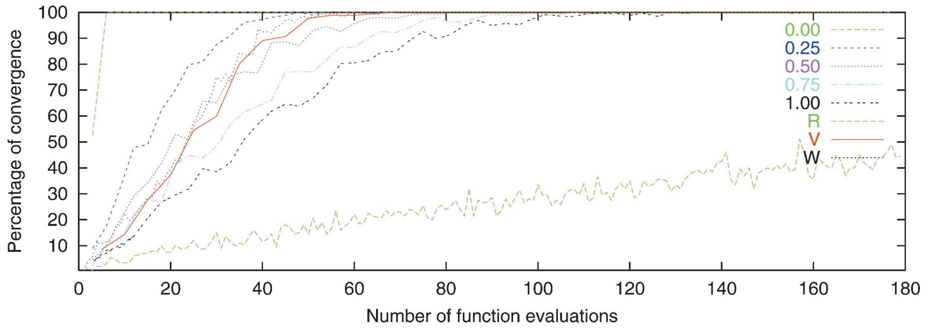
Note that in both the numerical and visual profiles, the following abbreviations will be used to indicate the method used: Gn for the n point Gauss quadrature, Jn for the fully jittered n point Gauss quadrature rule, Rn for the random offset quadrature, V for the Voronoi method, and W for the weighted Monte Carlo method. In addition, a partially jittered quadrature will be indicated as either Jn_p, where n indicates an n-point Gauss quadrature rule, and p indicates the amount of jittering, or simply as p if the Gauss quadrature rule is clear.

4.3.1 *Problem Family* 1: C^2 *Discontinuity*. The first problem family is the following:

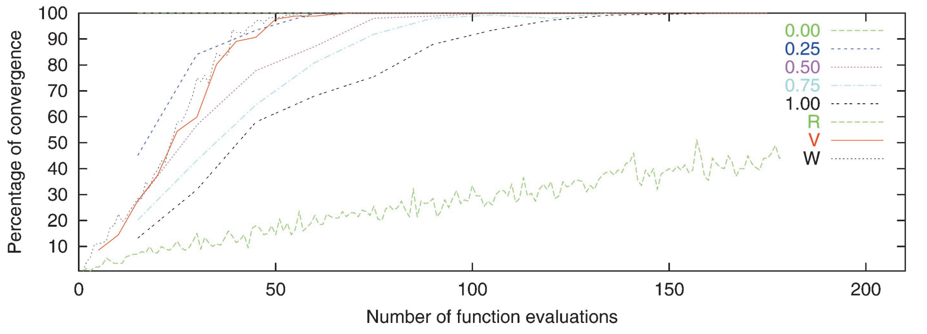
$$\begin{aligned} f_1 &= \begin{cases} x^2 & \text{if } x \le \lambda \\ 2\lambda x - \lambda^2 & \text{if } x > \lambda \end{cases} \quad \lambda \in [0,1], \\ I_1 &= \int_0^1 f_1\, dx = \frac{1}{3}\lambda^3 - \lambda^2 + \lambda. \end{aligned}$$

This is a piecewise continuous function, with a second derivative discontinuity at λ.

This family is very well behaved for all choices of $\lambda \in [0, 1]$. As a result, all of the deterministic quadratures with 5 or more points achieve 100% convergence without compounding. The 3-point Gauss and the 1-point Gauss method converge almost as fast; they achieve 100% convergence after using a total of 6 and 15 function evaluations, respectively. Since all methods with 3 or more points converge fully using at most two panels, there is no advantage to be gained by using a difficulty-driven compound quadratures scheme based on the knowledge of λ, the location of the difficulty. After 30 function evaluations, the resulting strips are identical to the true strip if we use any of the deterministic methods. The jittered quadratures, on the other hand, require 180 or more function evaluations to converge numerically.



(a) J3



(b) J15

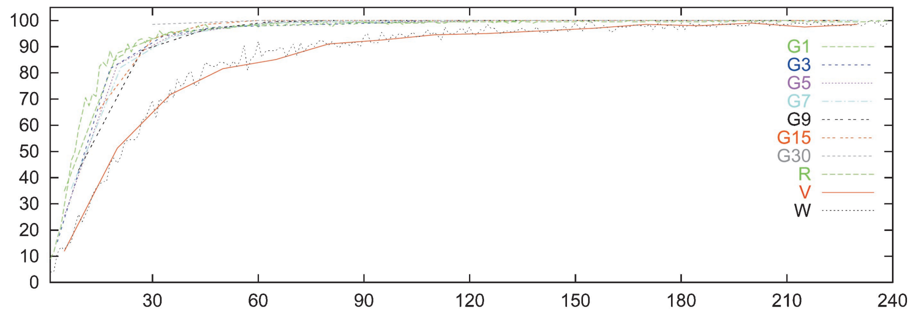
Fig. 10. Numerical convergence plots for problem family 1, with $\epsilon = 0.5$, for partially jittered quadratures J3 and J15. The random offset, weighted Monte Carlo, and Voronoi methods are included for comparison.

$\overline{V}$
$\overline{R}$

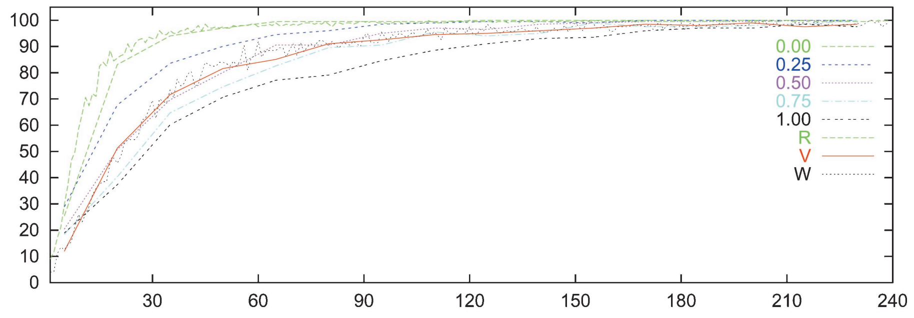
Fig. 11. Visual convergence plots for problem family 1, after 30 function evaluations.

Figure 10 illustrates the relative convergence rates of partial location jittering for an absolute tolerance of 0.5, for both J3 and J15, with the jittering varying from 0.00 (deterministic) to 1.00 (fully jittered). For both quadrature rules, the rate of convergence decreases as the amount of jittering increases. This is true for all of the quadrature rules. The Voronoi method converges at about the same rate as the slightly jittered quadratures. Numerically, the weighted Monte Carlo method converges at about the same rate as the Voronoi method, and the random offset method converges at the slowest rate, both numerically and visually.

All of the quadrature based strips (both deterministic and jittered) have converged visually after 30 function evaluations, as well as both the Voronoi and weighted Monte Carlo methods. Figure 11 shows the resulting strips after 30 function evaluations have been used with the Voronoi and random offset



(a) Relative convergence of Gauss, random offset, Voronoi and weighted Monte Carlo.



(b) Effects on convergence of location jittering with J5.

Fig. 12. Numerical convergence plots for problem family 2, with $\epsilon = 0.5$.

methods, interspersed by the true strip. The jittered quadratures and the weighted Monte Carlo methods gave strips similar to the Voronoi strip, and only the random offset method did not converge visually after 30 function evaluations.

4.3.2 *Problem Family* 2: C^1 *Discontinuity*. The second problem family is the following:

$$f_2 = \begin{cases} x^2 & \text{if } x \le \lambda \\ \frac{\lambda^2}{1-\lambda}(1-x) & \text{if } x > \lambda \end{cases} \quad \lambda \in [0, 1],$$

$$I_2 = \int_0^1 f_2\, dx = \frac{1}{2}\lambda^2 - \frac{1}{6}\lambda^3.$$

This is a piecewise C^2 function, with a first derivative discontinuity at λ.

The members of this family are also relatively easy to integrate for all values of $\lambda \in [0, 1]$. As shown in Figure 12(a), all of the deterministic quadrature rules, as well as the random offset method, achieve over 90% convergence after 30 function evaluations, for a tolerance of 0.5. The convergence rates of all these methods are nearly indistinguishable. The Voronoi and the weighted Monte Carlo methods have a slightly slower convergence rate. Increasing the jittering, as shown for the 5-point Gauss quadrature J5 in Figure 12(b), decreases the convergence rate for all quadrature rules.

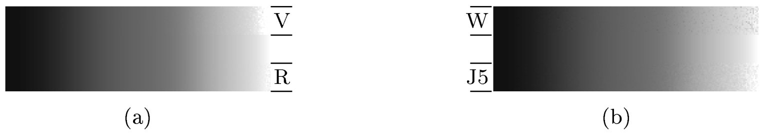


Fig. 13. Visual convergence plots for problem family 2, after 30 function evaluations.

The intensity strips for the deterministic quadratures have all converged visually after 30 function evaluations. The Voronoi strip and the random offset strips are almost converged, whereas the weighted Monte Carlo and jittered strips are slightly noisier. The strips in Figure 13(a) show the results after 30 function evaluations for the Voronoi and random offset methods, and those in Figure 13(b) show the results after 30 function evaluations for weighted Monte Carlo and for the fully jittered quadrature J5. Notice that the weighted Monte Carlo strip (W) has more noise in the form of bright black and white dots.

For this problem family, using a difficulty-driven compound quadrature is only slightly advantageous. From a numerical perspective, if we subdivide the interval of integration into two panels, with the boundary coinciding with λ, then we will integrate any function exactly provided that we are using a method of degree at least 2. This is evident since any member of the problem family will be continuous over each panel, and will be of degree at most 2. From a visual perspective, we can only fairly compare the methods after at least 30 function evaluations have been used, since the basic quadratures tested have up to 30 points. Since the regular compound quadratures have all converged visually by this point, there is little advantage to be gained by knowing the location of λ: for the quadratures with less than 15 points, we would, however, achieve earlier visual convergence.

4.3.3 *Problem Family* 3: *C^0 Discontinuity.* The third problem family is the following:

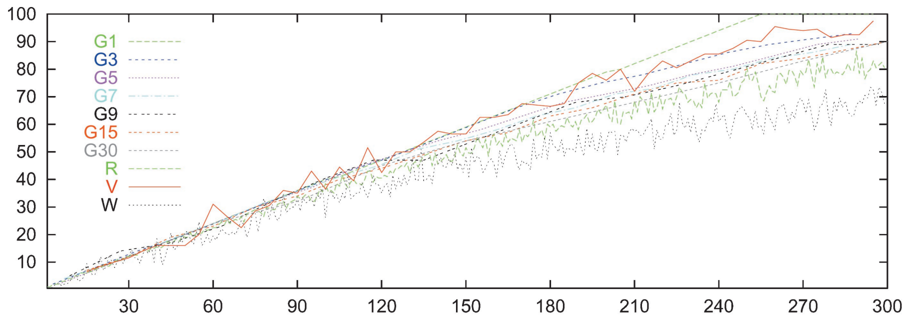
$$f_3 = \begin{cases} 0 & \text{if } x \leq \lambda \\ 1 & \text{if } x > \lambda \end{cases} \quad \lambda \in [0, 1],$$

$$I_3 = \int_0^1 f_3\, dx = 1 - \lambda.$$

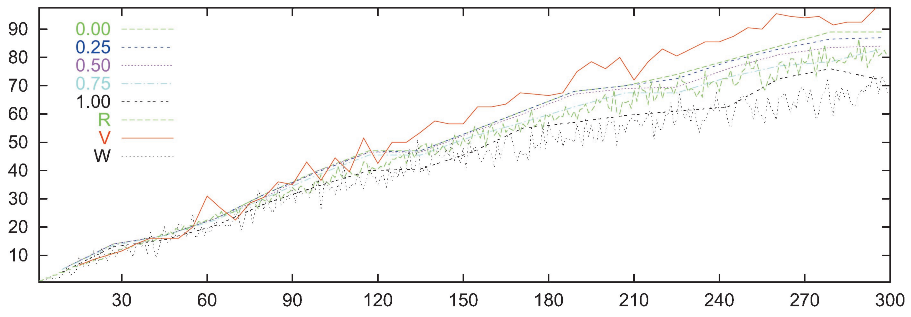
This is a step function with a discontinuity at λ.

Figure 14(a) shows the convergence plots for an absolute tolerance of 0.5 for all of the deterministic quadratures, as well as the plots for the random offset, Voronoi, and weighted Monte Carlo methods. All of these techniques have similar convergence rates, with random offset and weighted Monte Carlo having a slightly slower rate.

Figure 14(a) shows that the compound quadrature rules that converge most quickly use fewer points in the base quadrature. For a given number of function evaluations, the number of panels over which we apply the base method is inversely proportional to the number of points used by this method. Since



(a) Relative convergence of Gauss, random offset, Voronoi and weighted Monte Carlo.



(b) Effects on convergence of location jittering with J9.

Fig. 14. Numerical convergence plots for problem family 3, with $\epsilon = 0.5$.

the function is constant everywhere but at the discontinuity, all of the base quadrature rules will integrate the function exactly over any panel that does not contain the discontinuity. Conversely, the only error incurred will be bounded by the length of the panel containing the discontinuity. Since the length of this panel is inversely related to the number of points in the the base quadrature rule, the error will also be inversely related to the number of points. Since the integrand has a discontinuity, Bakhvalov's theorem tells us that the $O(1/n)$ convergence rate is the best that we can achieve. Therefore, for this particular problem, the regular compound quadrature based on G1 (i.e., super sampling) achieves the optimal convergence rate.

The effect of jittering the location of the quadrature points is similar for all quadrature rules. Figure 14(b) illustrates the convergence plots for one of the quadrature rules, namely, J9, with the amount of location jittering varying from 0 (deterministic) to 1 (fully jittered). All of these methods have similar convergence rates, with the convergence rate being inversely related to the amount of jittering. The convergence rates are similar to the rate for the Voronoi method, but slightly poorer.

Figure 15 shows some of the resulting strips after 30 function evaluations have been used with each technique. The strips on the left-hand side were

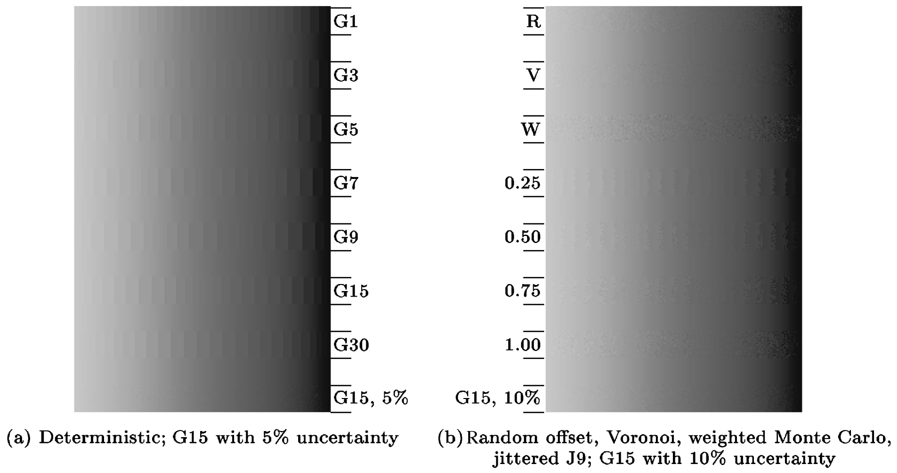


Fig. 15. Visual convergence plots for problem family 3, after 30 function evaluations.

generated using all the deterministic quadratures with regular compounding, with the exception of the last strip, which was generated using a difficulty-driven compound quadrature based on G15, with an uncertainty of 5%. Notice that slight banding occurs in all of the deterministic strips. Banding is not visible for the difficulty-driven strip.

The strips on the right-hand side were generated using random offset, Voronoi, weighted Monte Carlo, partial location jittering of J9, and a difficulty-driven compound quadrature based on the G15, with an uncertainty of 10%. The Voronoi strip is visually pleasing, with no banding, and only a small amount of noise. The random offset strip is slightly noisier than the Voronoi strip, but less noisy than the weighted Monte Carlo strip. In the J9 strips, banding is progressively reduced as we increase the amount of jittering; the aliasing patterns are transformed into less visually objectionable random noise. Since the percentage of numerical convergence actually decreased as we increased the jittering, this illustrates the fact that numerical accuracy and visual accuracy are not always positively correlated.

For this problem family, the difficulty-driven compound quadrature schemes are helpful as long as the uncertainty for the location of the difficulty is less than 10%, that is, as long as we guarantee that a panel border will be located within 10% of the jump discontinuity. For a given uncertainty, the choice of the underlying quadrature rule does not affect the visual quality of the results. For this reason, we have chosen to illustrate these results using only one of the quadratures rules, namely, G15, and using only two uncertainties, namely $\xi = 5\%$ and $\xi = 10\%$. The 5% difficulty-driven strips are almost visually perfect, with only a minute amount of noise, whereas the 10% strips are of similar quality to the V30 strip. Finally, since the integrand is constant everywhere but at λ, a difficulty driven quadrature with zero uncertainty will converge numerically with a total of at most two panels, even for G1.

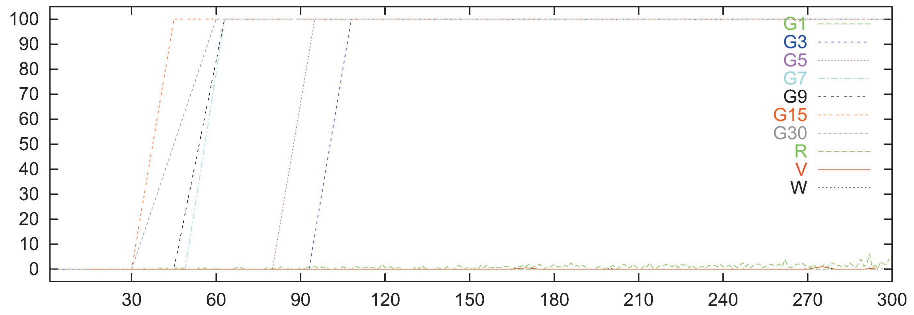


Fig. 16. Numerical convergence plots for problem family 4, with an absolute tolerance of $\epsilon = 0.5$. Only the deterministic quadratures G3 through G30 converge at all.

4.3.4 *Problem Family* 4: *Oscillating Function.* The fourth problem family is the following:

$$f_4 = \cos(\lambda + 100x), \quad \lambda \in [0, 1],$$
$$I_4 = \int_0^1 f_4\,dx = \frac{1}{100}\sin(100 + \lambda) - \sin\lambda.$$

This is a rapidly oscillating function in the interval [0, 1], with slightly less than 16 full cycles in that interval.

Figure 16 shows the convergence plots for an absolute tolerance of 0.5 for all of the deterministic quadratures, and for the random offset, Voronoi, and weighted Monte Carlo methods. The convergence is either fully achieved or not achieved at all. As we can see in Figure 16, only the deterministic Gauss quadratures with 3 or more points converge. Due to the particular difficulty of this integral, the higher-degree quadratures converge faster, as can also be seen in Figure 16.

None of the jittered quadratures, random offset, Voronoi, or weighted Monte Carlo methods achieve full convergence. The highest percentage of convergence achieved by any of the stochastic methods is only 6%, namely, by the random offset quadrature.

In Figure 17, we show the resulting intensity strips generated with deterministic compound Gauss quadratures, and the random offset, Voronoi, and weighted Monte Carlo methods. The strips in Figure 17(a) were generated using 30 function evaluations, and the strips in Figure 17(b) were generated using 60 function evaluations.

The strips after 30 function evaluations are completely different from the true strip. The deterministic strips are either black or white, while the random offset, Voronoi, and weighted Monte Carlo strips are very noisy. This is aliasing due to the undersampling of the high-frequency sinusoid.

After 60 function evaluations, G9, G15, and G30 have converged completely; G7 is too bright; G5 is almost converged; G3 is too dark. The G1 strip is almost completely black, and the ones generated by the random offset, Voronoi, and weighted Monte Carlo methods are still very noisy. We found that even after 300 function evaluations, the strips generated by these methods, as well as by the jittered methods, were very noisy.

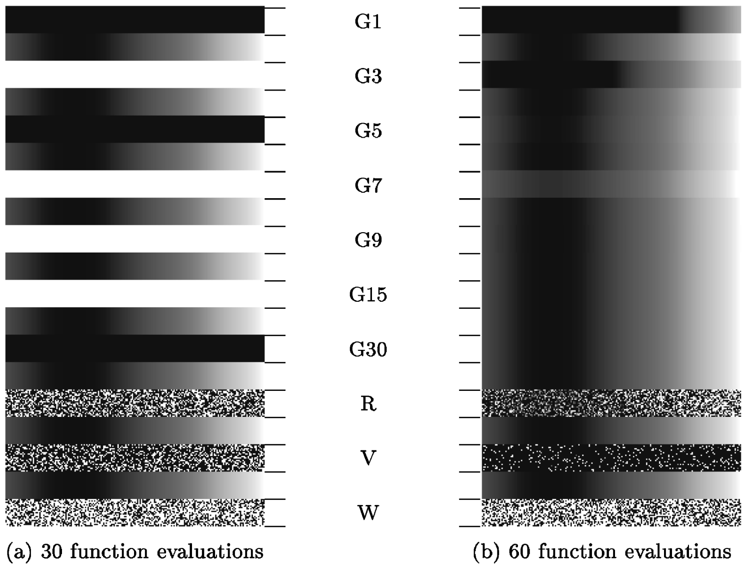


Fig. 17. Visual convergence plots for problem family 4 for deterministic quadratures, random offset, Voronoi, and weighted Monte Carlo, after 30 and 60 function evaluations.

This problem family appears very sensitive to jittering. Only the deterministic quadratures are able to provide converged solutions. Through experimentation, we found that partial jittering as low as 0.1 results in very noisy images. This problem family is the only one in which the parameter λ does not correspond to the location of a single difficulty. Examining the integrand reveals that the parameter is a phase factor which shifts the cosine wave. There is no advantage to be gained by subdividing the interval of integration based on the parameter λ, since this location does not correspond to a single difficulty of the integrand.

4.3.5 *Problem Family* 5: *Function with a Peak.* The fifth problem family is the following:

$$\begin{aligned} f_5 &= e^{-8|x-\lambda|}, \quad \lambda \in [0,1], \\ I_5 &= \int_0^1 f_5 \, dx = \frac{1}{4} - \frac{1}{8}(e^{-8\lambda} + e^{8\lambda-8}). \end{aligned}$$

This is a function with a moderate peak at λ.

Figure 18 shows the convergence plots for an absolute tolerance of 0.5 for all of the deterministic quadratures, and for the random offset, Voronoi, and weighted Monte Carlo methods. Once again, we find that all of the deterministic quadrature rules have faster numerical convergence than the stochastic methods. The deterministic quadratures have similar convergence rates, with the lower-degree methods converging slightly faster than the higher-degree methods. The Voronoi method clearly has a faster convergence rate than the jittered quadratures (J1 shown), random offset, and weighted Monte Carlo methods.

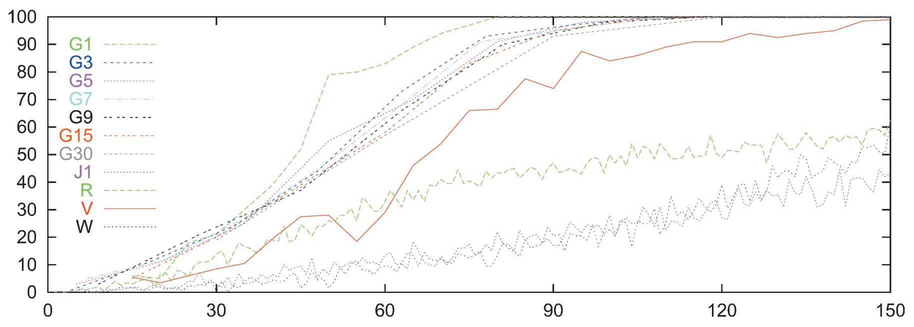
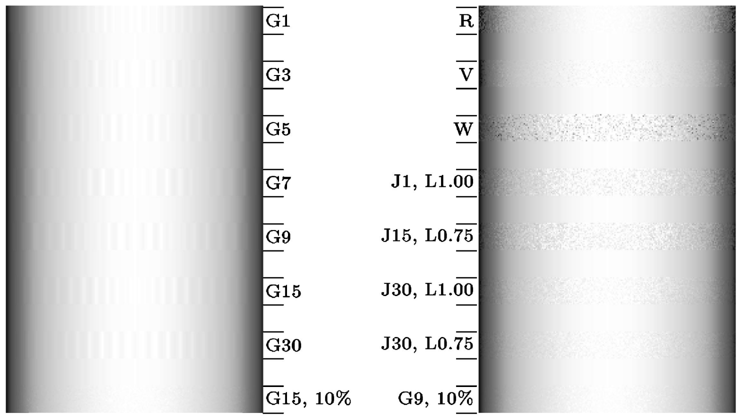


Fig. 18. Numerical convergence plots for problem family 5, with an absolute tolerance of $\epsilon = 0.5$.



(a) Deterministic; G15 with 10% uncertainty (b) Random offset, Voronoi; weighted Monte Carlo, jittered quadratures; G9 with 10% uncertainty

Fig. 19. Visual convergence plots for problem family 5, after 30 function evaluations.

The latter three methods have similar convergence rates, although initially the random offset method converges faster.

Figure 19(a) illustrates the results after 30 function evaluations for a variety of methods. All deterministic methods show aliasing in the form of banding; these bands are least visible with the 1-point Gauss quadrature. Notice also that the Voronoi strip shows no aliasing and is only slightly noisy, whereas the weighted Monte Carlo strip has a lot of noise in the form of bright black and white dots.

The random offset method and the jittered quadratures are moderately successful at removing banding artifacts while not introducing too much noise, but are not as visually acceptable as the Voronoi strip. Strips generated with the random offset method and some of the jittered techniques are shown in Figure 19(b). The location jittered quadratures are all successful at removing the banding artifacts. The 0.75 location jittered quadratures give the best results; this is illustrated for J15 and J30 in Figure 19. The exception is the J1 quadrature, which achieves better results with full jittering.

Table II. Numerical Convergence Plots for Difficulty Driven Compound Quadratures After 30 Function Evaluations, with an Absolute Tolerance of $\epsilon = 0.5$

λ uncertainty	G1	G3	G5	G7	G9	G15
10%	19.5%	21%	23%	25%	29%	50%
5%	19%	21%	32%	44.5%	49.5%	60%
2.5%	18%	41.5%	50.5%	51.5%	65.5%	82%
1%	30%	58.5%	66%	81.5%	91%	98.5%
0%	0%	100%	100%	100%	100%	100%

Even though numerical convergence is not achieved before at least 90 function evaluations, all deterministic methods have converged visually after 60 function evaluations; however, the stochastic strips are still slightly noisy.

Finally, for this problem family, knowing λ, the location of the peak, is advantageous as long as our uncertainty for the estimation of the location of the peak is at most 10%. When this is the case, all the deterministic techniques used in combination with the difficulty-driven compounding have converged visually after at most 30 function evaluations. This is illustrated by the last two strips of Figure 19: the last strip of Figure 19(a) was produced using a 10% error and the 15-point Gauss quadrature with two panels, and the last strip of Figure 19(b) was produced using a 10% error and the 9-point Gauss quadrature with three panels. If more than 10% error is allowed for the approximate location of the peak, the strips become slightly noisy.

In Table II the degree of numerical convergence achieved by each quadrature rule is shown for a variety of uncertainties on the value of λ, the location of the peak. Recall that the numerical convergence is the percentage of problems of the problem family that were integrated with an absolute error of less than 0.5. The results in this table show that G1 is not well suited to this particular type of difficulty-driven compounding. This is to be expected, since all the other deterministic methods are of higher degree and will be more successful at integrating the peak than G1. In particular, the degree of numerical convergence increases with the degree of the basic method used, even though the total number of points used (30) remains constant. Given that no traditional method achieved even 25% convergence after 30 function evaluations, the results achieved by the difficulty-driven compound quadratures are significant improvements over those of the former methods.

4.3.6 *Problem Family* 6: *Function with a Sharp Peak.* The sixth problem family is the following:

$$f_6 = \frac{0.01}{(x-\lambda)^2 + 0.0001}, \quad \lambda \in [1, 2],$$

$$I_6 = \int_1^2 f_6\, dx = \arctan(100\lambda - 100) - \arctan(100\lambda - 200).$$

This is a function with a sharp peak at λ. It is a shifted and scaled variant of Runge's function [Kahaner et al. 1989]

$$R(x) = \frac{1}{1 + 25x^2}. \tag{16}$$

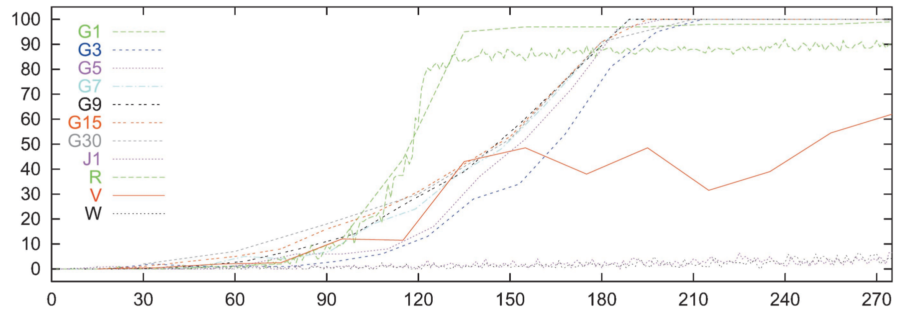


Fig. 20. Numerical convergence plots for problem family 6, with an absolute tolerance of $\epsilon = 0.5$.

Figure 20 shows the convergence plots for an absolute tolerance of 0.5 for all of the deterministic quadratures, for J1, and for the random offset, Voronoi, and weighted Monte Carlo methods. We find that all of the deterministic quadratures converge faster than their stochastic counterparts, except for the random offset method. Notice also that the jittered quadratures (J1 only shown) and the weighted Monte Carlo methods converge at very similar rates.

Most of the deterministic quadrature rules have similar convergence rates, although the higher-degree methods converge slightly faster, with super sampling being the exception. As shown in Figure 20, G1 converges much faster than the other deterministic quadratures, until it reaches 95% convergence after 125 function evaluations. It then slowly converges to 98% after 300 function evaluations. Conversely, the other deterministic quadratures have converged to less than 40% after 130 function evaluations, but have converged fully after at most 213 function evaluations. While G1 is better at quickly isolating the peak in a single interval, it cannot evaluate the area under this peak very reliably. The random offset method has a very similar behavior, which is to be expected since it consists of a shifted version of G1.

This particular problem family results in a peculiar behavior for the Voronoi method. If we examine the convergence plot of the Voronoi method in Figure 20, we notice that the convergence is nonmonotonic. This is especially evident in the range 155 to 190, and again from 190 to 255. In both ranges, the convergence of the Voronoi method actually decreases as we increase the number of samples, before beginning to increase again.

This decrease can be explained if we recall the way in which the points are generated for the Voronoi method. For this problem family, Voronoi sequences that result in partial subdivisions are on average less accurate than Voronoi sequences where the points generated by these partial subdivisions are simply removed (see Section 2.6 for definitions of the terminology).

We now present the results of an experiment using Voronoi processes over the interval [0, 1] whose initial seed σ was constrained by $\sigma \in [1/3, 2/3]$. We examined 100,000 integration problems from family 6, each generated by choosing a random value of $\lambda \in [1, 2]$. For each problem, we generated a random seed $\sigma \in [1/3, 2/3]$, and constructed the Voronoi sequences ranging from 191 to 255 points. We then approximated each integral using each Voronoi sequence and determined if a tolerance of 0.5 had been achieved.

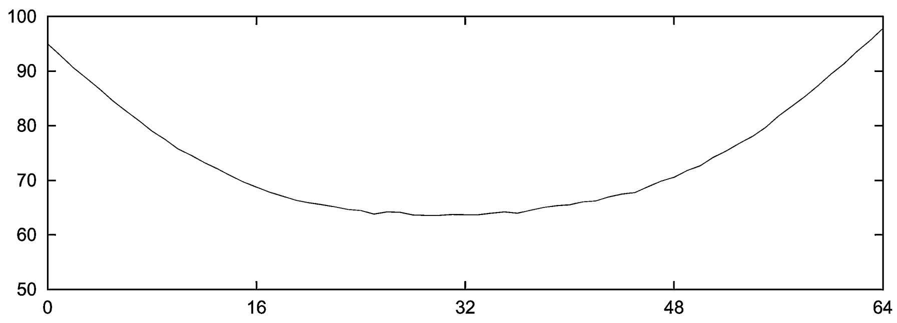


Fig. 21. Voronoi experiment: the abscissa is number of points in each method (minus 191); the ordinate is the percentage of numerical convergence, with a tolerance of $\epsilon = 0.5$.

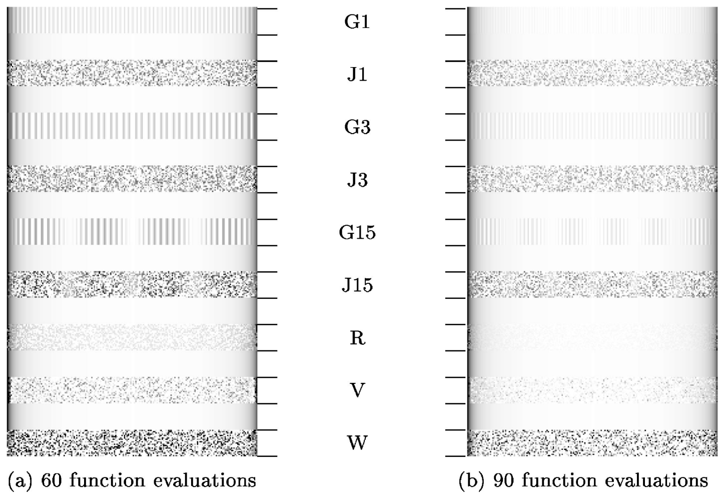


Fig. 22. Visual convergence plots for problem family 6, for G1, J1, G3, J3, G15, J15, random offset, Voronoi, and weighted Monte Carlo methods.

As was shown in Section 2.6, a Voronoi process whose seed is constrained as stated in our experiment will have full subdivision if 191 points are generated. As we increase the number of points from 191 to 255, the number of partial subdivisions increases from 0 to 64, with 64 being once again a full subdivision. In Figure 21, we illustrate the percentage of convergence for the Voronoi processes, with the number of points in the sequence ranging from 191 to 255. It is instructive to show the number of partial subdivisions as the abscissa.[5] Due to the nature of the problems from family 6, increasing the number of partial subdivisions actually decreases the accuracy of the calculated integral! This accuracy is at its worst when roughly half the intervals are being subdivided, and only increases once the subdivision has become almost full again.

As for visual convergence, some of the resulting strips after 60 function evaluations are shown in Figure 22(a), and after 90 function evaluations in

[5]The number of points in the Voronoi process is the sum of the abscissa and of 191.

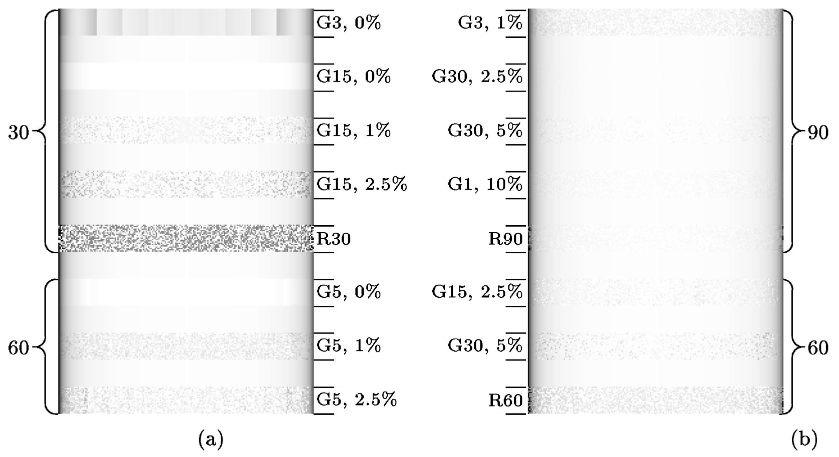


Fig. 23. Difficulty-driven visual convergence plots for problem family 6. Top left: 30 function evaluations. Bottom left and bottom right: 60 function evaluations. Top right: 90 function evaluations.

Figure 22(b). With fewer than 120 function evaluations, none of the methods tested achieved visual convergence. Whereas numerical convergence is achieved after 120 function evaluations only for G1, visual convergence is reached at 120 function evaluations for all of the deterministic methods.

After 60 (and 90) function evaluations, all of the deterministic quadratures produce aliasing in the form of bands of brightness. The deterministic and jittered quadratures G1, G3, G15, the random offset, Voronoi, and weighted Monte Carlo methods were used to generate the strips in Figure 22. In general, we find that the low-degree Gauss quadratures respond better to jittering than the higher-degree quadratures. This is evident in Figure 22 if we compare the location jittered strips for J1 and J3 to the location jittered strip for J15. The banding is removed in the first two strips, but not in the latter. Observe also that after both 60 and 90 function evaluations, the weighted Monte Carlo and the jittered quadrature strips are the noisiest, the Voronoi strip is less noisy, and the random offset strip one is the closest, visually, to the true strip, and has almost converged visually after 90 function evaluations.

This particular problem family benefits from difficulty-driven compounding. For this family, the parameter λ indicates the location of a sharp peak. After 30 function evaluations, we obtain results that are far superior to both regular quadrature compounding and the stochastic methods. In Figure 23(a), the top five test strips all use 30 function evaluations, with the last strip, R30, included as a reference. We include the random offset strip as a reference since it was the method which came the closest to achieving visual convergence. If λ is known exactly and difficulty-driven compounding is used, G1 and G3 (G3 shown) both show banding, whereas G5 through G15 (G15 shown) are all smooth, but have not converged visually. When we increase the uncertainty on λ to 1%, G5 through G15 (G15 shown) become slightly noisy, but less noisy than R30. If we further increase the uncertainty to 2.5%, all strips become slightly noisier, while only G15 (G15 shown) remains band-free. For uncertainties of 5% or more, the difficulty-driven compounding gives results equivalent to or worse than R30.

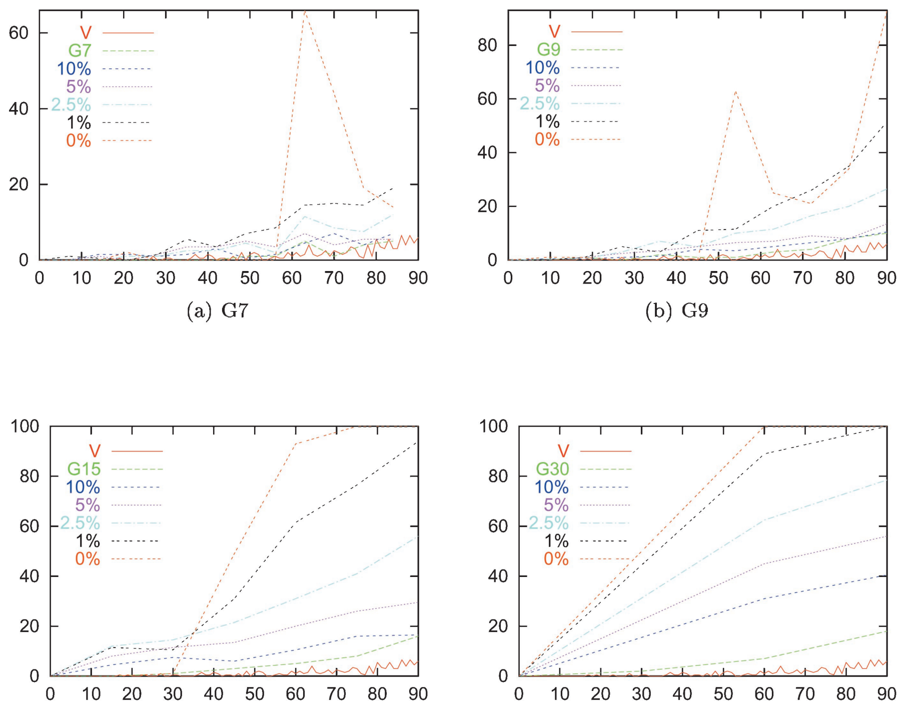
After 60 function evaluations, we again obtain results that are superior to both regular quadrature compounding and the stochastic methods (i.e., jittered quadratures, random offset, Voronoi, and weighted Monte Carlo). In Figure 23(a) and Figure 23(b), the bottom three test strips all use 60 function evaluations, with the last strip, R60, incorporated as a reference. If λ is known exactly and difficulty-driven compounding is used, G1 through G5 (G5 shown) are very smooth, but have not converged visually. The strips for G7 through G30 have all converged visually. When we increase the uncertainty on λ to 1%, G1 through G9 (G5 shown) become slightly noisy, but less noisy than R60, whereas G15 and G30 have converged. If we further increase the uncertainty to 2.5%, G1 through G9 (G5 shown) are both as noisy as R60, whereas G15 and G30 are very slightly noisy (G15 shown). For an uncertainty of 5%, only G30 (shown) remains better than R60, with the other techniques giving results similar to R60. Finally for uncertainties of 10% or greater, no advantage is obtained via difficulty-driven compounding.

After 90 function evaluations, we obtain results that are slightly superior to both regular quadrature compounding and the stochastic methods. In Figure 23(b), the top five test strips all use 90 function evaluations, with the last strip, R90, incorporated as a reference. If λ is known exactly and difficulty-driven compounding is used, all methods have converged visually. If λ is known within an uncertainty of 1%, all strips except G3 have converged visually; G3 (shown) is too dark, and of quality slightly worse than R90. With an uncertainty of 2.5%, only G1, G15, and G30 (G30 shown) are still better than R90, with all three of them with very slight noise. When the uncertainty is increased to 5%, G1 and G30 (G30 shown) are unchanged, whereas the other techniques are no better than R90. Finally, with an uncertainty of 10%, G1 (shown) is unchanged and very slightly noisy, and slightly better than R90.

Numerically, the difficulty-driven compound quadrature rules of higher-degree converge faster than the regularly compounded quadratures, and than the stochastic methods. The plots in Figure 24 show that G15 and G30 both converge noticeably faster with difficulty-driven compounding. The speed of convergence increases as we decrease the uncertainty of the location of the peak. The intermediate methods G7 and G9 also converge faster than both the stochastic methods and the regularly compounded quadrature, provided that the uncertainty is at most 1% and 2.5%, respectively. The intermediate method G5 only converges significantly faster than the Voronoi or the regularly compounded method if the location of the peak is known exactly. Numerically, the convergence of either G1 or G3 is not improved by using difficulty-driven compounding. Note that we have included the Voronoi plots, and not the random offset plots, since in the examined range, the former method actually converges faster numerically.

4.3.7 *Problem Family* 7: *Function with a Weak Singularity.* The seventh problem family is the following:

$$f_7 = |x - \lambda|^{-\frac{1}{2}}, \quad \lambda \in [0, 1],$$



(a) G7 (b) G9

(c) G15 (d) G30

Fig. 24. Numerical convergence plots of difficulty-driven compounding, for a tolerance of $\epsilon = 0.5$, for Voronoi (V), regular compounding (Gn), and difficulty-driven compounding with various uncertainties (10%, 5%, 2.5%, 1%, and 0%).

$$\begin{aligned} I_7 &= \int_0^1 f_7\,dx \\ &= 2(\sqrt{\lambda} + \sqrt{1-\lambda}). \end{aligned}$$

This is family 1 as taken from the test suite in [Espelid and Sorevik 1989]. It has a weak singularity at λ.

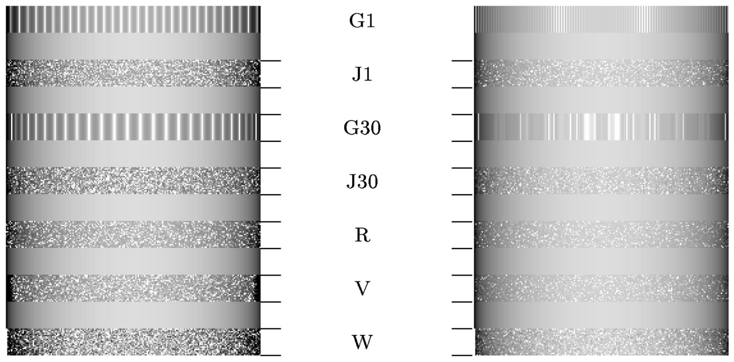
This is the most difficult[6] problem family. Even after 300 function evaluations, none of the methods have achieved more than 2.5% convergence within a tolerance of 0.5. In Table III, the maximum convergence for the deterministic quadrature rules, the random offset, Voronoi, and weighted Monte Carlo methods have been listed. The degree of convergence increases as the tolerance is relaxed: a tolerance of 50.0 represents a maximum intensity error of 50, on a scale of 0 to 255.

This problem family was also found to be the most challenging in terms of achieving good visual results. Figure 25(a) shows some of the strips generated after 30 function evaluations, while Figure 25(b) shows some of the strips after 300 function evaluations. Even after 300 function evaluations, none of the

[6]In practice, techniques such as singular value decomposition and special quadrature rules can be used to evaluate such integrals [Davis and Robinowitz 1984; Schröder 1993].

Table III. Maximum Numerical Convergence for at Most 300 Function Evaluations, as a Function of Tolerance ϵ

ϵ	G1	G3	G5	G7	G9	G15	G30	R	V	W
0.5	2%	1%	1%	1%	0%	0%	2%	3%	3%	3%
1.0	2%	2%	2%	1%	1%	1%	2%	4%	3%	5%
5.0	9%	10%	10%	10%	11%	11%	12%	15%	13%	16%
10.0	20%	21%	22%	20%	24%	21%	24%	25%	24%	25%
50.0	94%	94%	94%	92%	94%	93%	90%	96%	98%	93%



(a) 30 function evaluations (b) 300 function evaluations

Fig. 25. Visual convergence plots for problem family 7, for G1, J1, G30, J30, random offset, Voronoi, and weighted Monte Carlo.

techniques produced a strip visually indistinguishable from the true strip. This is not surprising considering the numerical convergence results for this problem family. After 300 function evaluations, all deterministic methods still show aliasing in the form of bands, as shown for G1 and G30.

The location jittered versions of all of the deterministic methods show error in the form of incoherent noise, as illustrated for G1 and G30, but are successful at removing the banding artifacts. The random offset, Voronoi, and weighted Monte Carlo methods are about as equally effective at removing the banding while introducing the least amount of uncorrelated noise in the strip.

This family is the most difficult test because of the weak singularity in the integral. If we know the location of the singularity, we can get better numerical and visual convergence, especially if the uncertainty for λ is 0%. In such a case, we don't have convergence for low values of epsilon; however, we do obtain convergence for an epsilon of 10, after 210 function evaluations for G15, and after 120 function evaluations for G30. Even though numerical convergence is still very slow, it is an improvement over both regular compounding and the other stochastic methods.

In Figure 26, we have shown the visual convergence for one of the quadrature rules (G15) assuming an uncertainty of 0% on λ. The strips are shown in progression from 30 function evaluations to 300 function evaluations. Even with an uncertainty of 0%, we do not obtain visual convergence using the difficulty-driven compounding. We do, however, have much less objectionable artifacts. The results for all quadratures tested (G1 through G30) are very

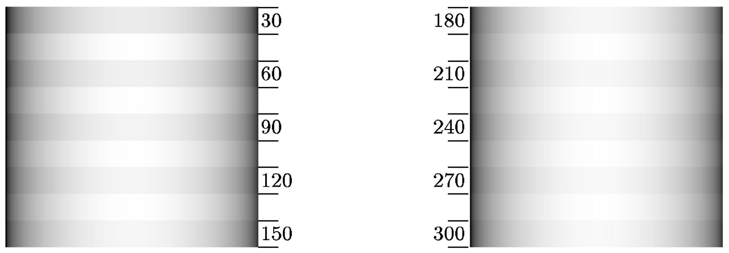


Fig. 26. Visual convergence plots for difficulty-driven compounding, using a 0% uncertainty with G15, and the number of points indicated.

similar. Notice that in each strip, the grey-scale gradation is too dark, but none the less smoother than any of the strips previously shown in Figure 25.

5. APPLICATIONS

In this section, we will compare the performance of the various one-dimensional integration techniques by using them to solve linear light source integration problems that occur in the course of rendering the office scene shown in Figure 28. We compare deterministic quadratures (based on both regular compounding and difficulty-driven compounding), jittered quadratures, and the random offset, Voronoi, and weighted Monte Carlo methods. The integrands examined will have one or more of the following four characteristics: smooth, single discontinuity, high-frequency oscillation/variation, or a sharp peak. Since we are considering only the integrals resulting from the computation of primary illumination from linear light sources, none of the integrals will contain a singularity.

Let us first review the problem of computing the direction illumination from a linear light source. This is a problem that has been the subject of several research papers in computer graphics [Nishita et al. 1985; Poulin and Amanatides 1990; Ouellette and Fiume 1999b; Heidrich et al. 2000]. As illustrated in Figure 27, we define a *linear light source* to be the line between two points L_1 and L_2, which emits light along its entire length, and illuminates an element of area dA. Consider an element of the light source dL, consisting of an infinitesimally small segment of the light source. As illustrated in Figure 27, dL subtends an angle of $d\alpha$ with respect to dA.

Let α_{12} be the angle defined by L_1, dA, and L_2, as illustrated in Figure 27. Let $d\alpha$ be an element of angle measured in the plane defined by dA, L_1, and L_2, and at an angular distance of α from the edge defined by dA and L_1. Let $L(\alpha)$ be the radiance received through $d\alpha$ by dA, and $f_r(\alpha; \theta_r, \phi_r)$ be the radiance reflected by dA through an element of solid angle $d\omega_r$ in the direction defined by the pair of polar angles (θ_r, ϕ_r). Then L_r, the total radiance reflected through $d\omega_r$, is defined as

$$L_r = \int_{\alpha=0}^{\alpha=\alpha_{12}} L_i(\alpha) f_r(\alpha; \theta_r, \phi_r) \cos\theta_i(\alpha)\, d\alpha, \tag{17}$$

where $\cos\theta_i(\alpha)$ is the angle of declination as measured from Z, the surface normal with respect to dA, and is a function of the parameter α.

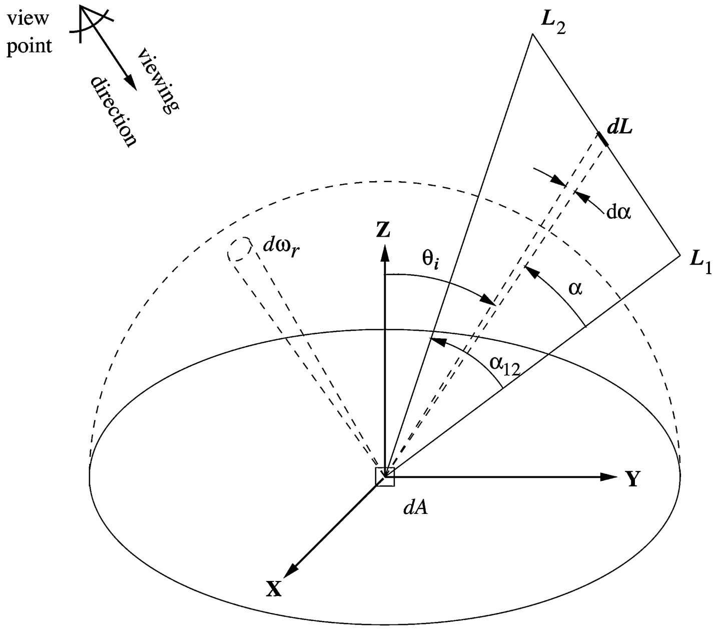


Fig. 27. Linear light source illumination problem.

We will now compare the effectiveness of numerical methods in handling the integrand derived from the illumination integral associated with one or more linear light sources. We have designed an office scene such that each of the four previously mentioned types of integrand characteristics is represented. This scene consists of an office illuminated by three diffuse linear light sources, as shown in Figure 28. Two of the sources are located on the ceiling (and out of view), and the third is located in the desk lamp. The scene is rendered using a ray caster that calculates the primary illumination by computing the light integral associated with each light source.

The resulting light source integrals present one or more of the integration difficulties that were present in the problem families of the previous section. Each box in Figure 28 corresponds to one of the outlined classes. The surfaces that are not in shadow and that do not have a significant highlight (e.g., Figure 28(a)) give rise to a smooth integrand. The regions that are in penumbra and that do not have a significant highlight (e.g., Figure 28(b)) produce a smooth integrand with a single discontinuity. The regions that have a high-frequency occlusion pattern obscuring the light source (e.g., Figure 28(c)) have an integrand with a high-frequency variation. Finally, the regions that show a significant highlight (e.g., Figure 28(d)) have an integrand with a sharp peak.

5.1 Smooth Integrands

For smooth integrands, numerical quadratures of low precision will integrate the primary illumination integral with high accuracy. In the following example, the region of the wall to the right of the painting, as delimited by box (a) in

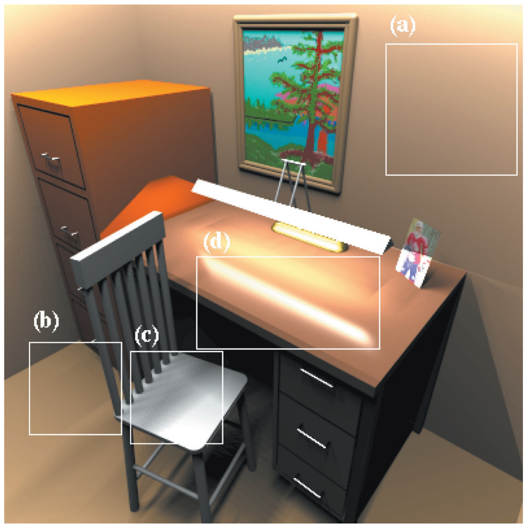


Fig. 28. Test regions of an office scene, with no antialiasing performed.

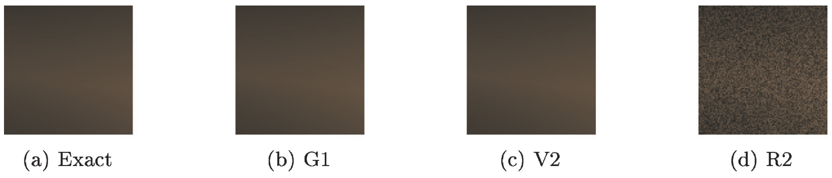


Fig. 29. Smooth region, approximated with at most 2 function evaluations.

Figure 28, has been chosen to compare the effects of the various quadrature rules on integrals of smooth integrands.

Figure 29(a) shows the exact solution. Figure 29(b) shows the solution if only a 1-point quadrature is used to integrate each linear light source. This solution is almost identical to the true solution, both visually and numerically (no pixel differs by more than 3). If we increase the quadrature rule to a 2-point quadrature, no pixel differs by more than 1. If instead of a quadrature rule, we use a stochastic method, then Figure 29(c) and (d) show the results for the Voronoi and the random offset method, respectively. Notice that the Voronoi method is visually accurate, but the random offset method is fairly noisy. Since we cannot generate a weighted Monte Carlo method with fewer than 2 points, this method was not considered for this test. For this type of smooth integrand, jittering the quadrature's point location provides no benefit, since the convergence of the deterministic quadratures is achieved immediately.

5.2 Single Discontinuity

In this example, the penumbral region of the floor in front of the filing cabinet, as delimited by box (b) in Figure 28, has been chosen to compare the effects of the various quadrature rules on integrals whose integrand contains a single

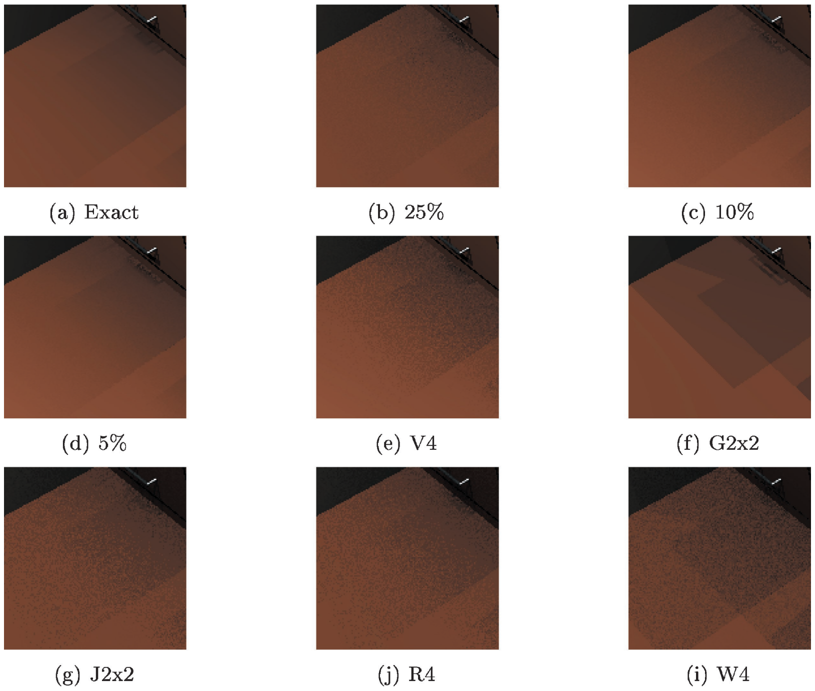
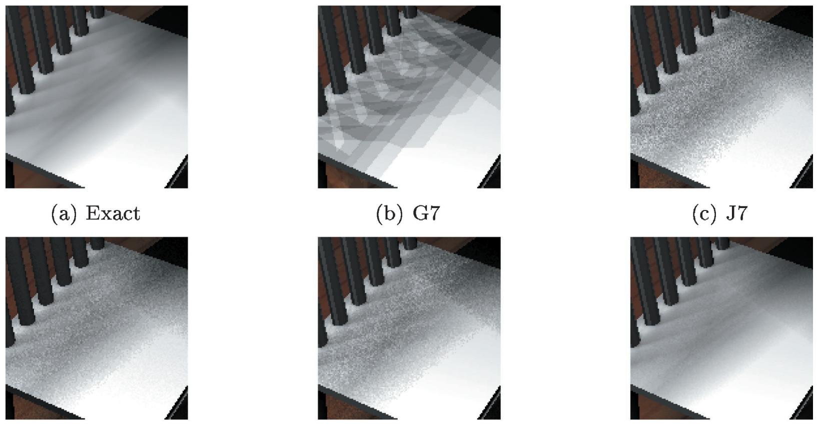


Fig. 30. Discontinuous region, approximated with 4 function evaluations.

discontinuity. The points on the floor have a view of the two ceiling light sources that is partially obstructed by the filing cabinet and the desk, and a completely occluded view of the desk lamp. Since the chair does not affect the primary illumination of this area, it has been removed to allow a better view of the penumbral region.

Figure 30(a) shows the exact solution for the area in front of the bottom of the filing cabinet. The other images have all been computed using at most four function evaluations per light source. In Figure 30(b), 30(c) and 30(d), we used a 2-point Gauss quadrature rule, compounded using a difficulty-driven algorithm. In each case, the discontinuity was calculated within the given uncertainty,[7] the light source was subdivided into two panels at this discontinuity, and a 2-point Gauss quadrature rule was used to integrate both panels. The results are surprisingly good even for a 25% uncertainty. Notice that in all three cases, the handle on the filing cabinet drawer causes more than one discontinuity in the light source integrand, which explains the presence of artifacts in the shadow cast by the handle. Figure 30(e) shows the results of applying a 4-point Voronoi method; the image is noticeably noisier than the previous three images. Figure 30(f) shows the result of applying a regular two panel compounding with the same 2-point Gauss quadrature. The image is not noisy, but has artifacts in the form of banding. Jittering the location of the

[7]See Ouellette and Fiume [1999] for a discussion of the problem of approximating the location of a discontinuity.



(a) Exact (b) G7 (c) J7

(d) R7 (e) V7 (f) V30

Fig. 31. High-frequency region, approximated with 7 or 30 function evaluations.

quadrature points (J2 $\times$ 2) eliminates the banding, replacing it with noise as shown in Figure 30(g). The 4-point random offset method (R4) shown in Figure 30(h) has a similar amount of noise as J2 $\times$ 2, but both are slightly noisier than V4. Finally, notice that the weighted Monte Carlo method produces the noisiest image, as shown in Figure 30(i).

5.3 High Frequency

In this example, the shadow region on the seat of the chair, as delimited by box (c) in Figure 28, has been chosen to compare the effects of the various quadrature rules on integrals with multiple discontinuities. The points on the chair have a view of one of the ceiling light sources that is partially obstructed by one or more of the bars on the back of the chair. The resulting integrands are similar to, but different from, the oscillating functions of the fourth problem family: the integrands are discontinuous and have a lower frequency. Figure 31(a) shows the exact solution for the shadow on the chair.

This signal is difficult to integrate due to the numerous discontinuities (there are up to 13 discontinuities in the integrand). As a result, convergence is slow. In Figure 31(b), (c), (d), and (e), we compare the convergence of four different methods using seven function evaluations per integral. In Figure 31(b), applying the 7-point Gauss quadrature (G7) causes aliasing: The shadow appears to be projected from several point light sources. In Figure 31(c), the aliasing is eliminated by using the same technique but with jittering of the sample points; however, the image is fairly noisy. Applying a 7-point random offset method results in an image without coherent aliasing, and that is a little cleaner than Figure 31(c). Using a 7-point Voronoi method provides the best visual result of any of the 7-point methods. Figure 31(e) is crisper than Figure 31(c), and marginally better than Figure 31(d), in that it is better at preserving features apparent in the exact image of Figure 31(a), notably the directional shadow near the back of the seat, and the solid shadow in the top-right corner of the image. The application of a 7-point weighted Monte Carlo method resulted in an

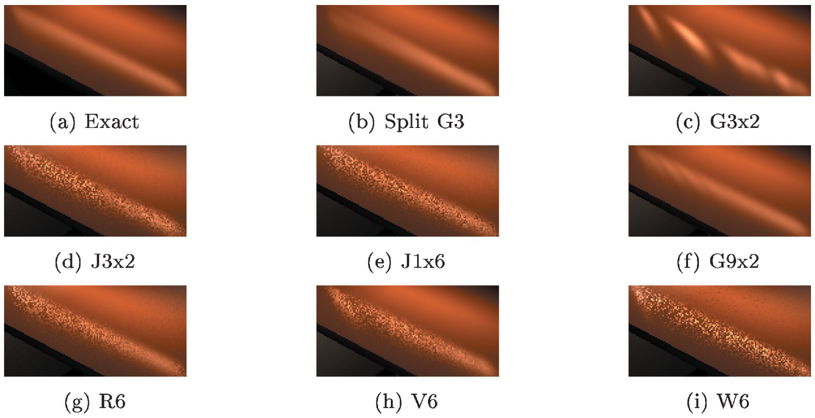


Fig. 32. Highlight region, approximated with six function evaluations.

image (not shown) slightly noisier than Figure 31(c), but without banding. Finally, none of the methods converge (visually or numerically) with fewer than 30 function evaluations. Figure 31(f) shows the result of using a 30-point Voronoi method, the best method among all 30-point methods.

5.4 Peak

In this example, the highlighted region on top of the desk, as delimited by box (d) in Figure 28, has been chosen to compare the effects of the various quadrature rules on integrals with a peak in their domain of integration. The points on the desk have an unobstructed view of the desk lamp. Note that most of the irradiance falling on these points comes from the desk lamp.

It should be noted that for this type of problem, it is sometimes possible to apply importance sampling [Sobol 1994; Veach and Guibas 1995] to find an approximate solution. Essentially, if we know a function that is somewhat proportional to the integrand, and yet simple enough to model, we can use it to sample the integrand in a more efficient fashion. However, such functions are not always readily available, and increase in usefulness with an increased dimensionality of the integration problem.

Figure 32(a) shows the exact solution for this area. Figure 32(b) was generated using a 3-point Gauss quadrature rule, compounded using a difficulty-driven algorithm. For each integral, the domain of integration was split at the location where the integrand peaked, and the 3-point quadrature rule was applied to each resulting panel. In Figure 32(c), the same technique was applied, but the domain of integration was split exactly in half. Notice the resulting fragmentation of the highlight. Even if we use a deterministic 9-point Gauss quadrature on each half, the highlight is still not smooth, as can be seen in Figure 32(f). In Figure 32(d), a 3-point jittered quadrature is compounded on two intervals, resulting in a noisy highlight, without fragmentation. Applying a 1-point jittered quadrature compounded on six intervals results in a very similar image, as seen in Figure 32(e). If we use a random offset or Voronoi method with 6 points, we obtain a slightly cleaner highlight, as shown in Figure 32(g) and (h). Finally, if we use a 6-point weighted Monte Carlo method, the image, as shown in Figure 32(i), is noticeably noisier. The difficulty-driven compounding is the clear winner for all the methods using at most 6 points per integral.

6. CONCLUSIONS

In this paper, we have presented a thorough study of the comparative effectiveness of deterministic and stochastic numerical techniques for solving one-dimensional integration problems that commonly occur in computer graphics. We favorably compared a new type of numerical integration technique, difficulty-driven compound quadratures, to traditional deterministic quadrature compounding and to several stochastic techniques, namely, jittered quadratures, random offset quadratures, Voronoi sampling, and weighted Monte Carlo sampling.

We first used periodograms to compare the frequency domain characteristics of both deterministic and stochastic techniques. We observed that the *blue noise* characteristics of the jittered quadratures and of the Voronoi method and, to some degree, of the weighted Monte Carlo method, foreshadowed their effectiveness at eliminating aliasing. Similarly, the deterministic nature of Gauss quadrature was reflected in their periodograms, and these were an indicator that these techniques would have problems integrating signals with high-frequency content.

Applying ideas from the numerical analysis community, we then compared the same techniques using the concept of performance profiles. This allowed us to examine the behavior of each technique over entire classes of problems, similar to problems encountered in computer graphics. For seven different types of integration problems, we examined both the numerical and the visual performance profiles.

In general, we found that for smooth integrands (families 1 and 2), traditional deterministic quadratures were the best choice. For integrands where the location of a single difficulty corresponded to the value of a parameter λ, difficulty-driven compound quadratures outperformed all other methods, even if the location of λ was only known to a tolerance of 5–10%. For these same problems, the random offset and/or the Voronoi method were the next best alternative, and jittered quadrature was slightly less useful. Finally, for the problem family with a highly oscillating integrand, the best choice was again traditional deterministic quadratures, provided that a sufficient number of function evaluations were used. In general, we also observed that the weighted Monte Carlo method often had the slowest numerical convergence, and the noisiest images. The results are summarized in Table IV.

Finally, we compared the performance of the same numerical techniques for four different types of illumination integrals. These integration problems were obtained by considering the primary illumination from linear light sources, for four different regions of a scene. The four regions were chosen such that the resulting integrands would form four distinct classes of integration problems. In general, our conclusions were similar to those of the previous section. For problems with a single difficulty (e.g., integrand with a peak or a discontinuity),

Table IV. Summary of Problem Family Results

Family	Difficulty	Best Choice(s)	Next Best Alternative
1	C^2 discontinuity	Low-degree quadrature	Voronoi, weighted Monte Carlo
2	C^1 discontinuity	Low-degree quadrature	Voronoi, random offset
3	C^0 discontinuity	Low-degree quadrature with difficulty-driven compounding	Voronoi, random offset, or jittered quadrature
4	Oscillating	High degree quadrature	
5	Moderate peak	Moderate-degree quadrature with difficulty-driven compounding	Voronoi, random offset, or jittered quadrature
6	Sharp peak	High-degree quadrature with difficulty-driven compounding	Random offset
7	Weak singularity	High-degree quadrature with difficulty-driven compounding	

Table V. Summary of Primary Illumination Results

Integrand Characteristic	Best Choice(s)	Next Best Alternative
Smooth and continuous	Low-degree quadrature	Voronoi
Single discontinuity	Low-degree quadrature with difficulty-driven compounding	Voronoi, random offset, or jittered quadrature
Multiple discontinuities	Voronoi or jittered quadratures	
Moderate peak	Low-degree quadrature with difficulty-driven compounding	Voronoi or random offset

difficulty-driven compounding outperformed all other methods. For smooth integration problems, a deterministic quadrature was sufficient, whereas for high-frequency problems, a stochastic method such as the Voronoi or the random offset method gave more convincing results. Once again, the weighted Monte Carlo method provided the noisiest results for a given computational cost. The relative effectiveness of these methods is summarized in Table V.

The results of both the problem families and the rendering problems have demonstrated the potential effectiveness of difficulty-driven compounded quadratures. In computer graphics problem, we seldom expect to know the exact location of a difficulty, with discontinuity meshes in polygonal environments being an exception. However, our results demonstrate that even using approximate knowledge of the location of a difficulty can greatly increase the effectiveness of numerical techniques. We have also shown that stochastic techniques such as Voronoi sampling, random offset quadratures, and jittered quadratures can provide decent solutions where deterministic quadratures suffer from aliasing.

In the immediate future, we will continue our investigation of efficient algorithms for the approximate location of difficulties that arise in the computation of integrals, in common rendering problems. A priority is to investigate both the cost of determining the type and location of the difficulties, and also the degree of quadrature rule that is best suited for each particular situation.

As future research, we also would also like to extend this study to two-dimensional integrals. Of particular interest are the integration problems associated with rendering area light sources, and the potential benefits that can be derived from difficulty-driven compounding.

APPENDIX: DERIVATION OF PERIODOGRAM FORMULATION

In this appendix, we provide a formal derivation of the definition of a periodogram. Given a sampling function $s(x)$ that is periodic over the interval [0, 1], its *spatial autocorrelation function* $\mathcal{R}(\tau)$ is defined to be

$$\mathcal{R}(\tau) = \int_0^1 s(x)s(x+\tau)\,dx. \tag{18}$$

Since $s(x)$ is also real, it is equivalent to its Fourier Series representation, that is,

$$s(x) = \sum_{k=-\infty}^{\infty} c_k e^{i2\pi kx}, \tag{19}$$

where

$$c_k = \int_0^1 s(x)e^{-i2\pi kx}. \tag{20}$$

If we substitute this representation into the definition of $\mathcal{R}(\tau)$, we can derive the identity: $\mathcal{R}(\tau) = \sum_{k=-\infty}^{\infty} \|c_k\|^2 e^{i2\pi k\tau}$.

$$\begin{aligned}
\mathcal{R}(\tau) &= \int_0^1 s(x)s(x+\tau)\,dx \\
&= \int_0^1 \sum_{k=-\infty}^{\infty} c_k e^{i2\pi kx} \sum_{m=-\infty}^{\infty} c_m e^{i2\pi m(x+\tau)}\,dx \\
&= \sum_{k=-\infty}^{\infty} \sum_{m=-\infty}^{\infty} c_k c_m \int_0^1 e^{i2\pi kx} e^{i2\pi m(x+\tau)}\,dx \\
&= \sum_{k=-\infty}^{\infty} \sum_{m=-\infty}^{\infty} c_k c_m e^{i2\pi m\tau} \int_0^1 e^{i2\pi(k+m)x}\,dx.
\end{aligned} \tag{21}$$

Since $\int_0^1 e^{i2\pi(k+m)x}\,dx$ is 1 if $m = -k$, and is 0 otherwise, the identity reduces to

$$\begin{aligned}
\mathcal{R}(\tau) &= \sum_{k=-\infty}^{\infty} c_k c_{-k} e^{i2\pi(-k)\tau} \\
&= \sum_{k=-\infty}^{\infty} c_{-k} c_k e^{i2\pi k\tau} \\
&= \sum_{k=-\infty}^{\infty} \|c_k\|^2 e^{i2\pi k\tau}.
\end{aligned} \tag{22}$$

The last equality holds because $s(x)$ being a real periodic signal implies that $c_{-k} = \overline{c_k}$. The above identity shows that the Fourier Series coefficients of the autocorrelation function can be obtained by computing the magnitude squared of the Fourier Series coefficients of the original signal $s(x)$. The plot of the Fourier Series coefficients of $\mathcal{R}(\tau)$ with respect to the frequency k is called a *periodogram*.

Given a set of N points $\{x_j\}$ together with their weight w_j, for $j = 1 \ldots N$, we define the sampling pattern $s(x) : [0, 1] \mapsto \mathbb{R}$ to be

$$s(x) = \sum_{j=i}^{n} w_j \delta(x - x_j). \tag{23}$$

Here, δ is the Dirac delta function, and $s(x)$ is made periodic by tiling the unit interval over the entire real line.

The Fourier Series for $s(x)$ is

$$\sum_{k=-\infty}^{\infty} c_k e^{i2\pi kx}, \tag{24}$$

where

$$\begin{aligned} c_k &= \int_0^1 s(x)\, e^{-i2\pi kx} \\ &= \int_0^1 \sum_{j=1}^{n} w_j \delta(x - x_j)\, e^{-i2\pi kx} \\ &= \sum_{j=1}^{n} w_j \int_0^1 \delta(x - x_j)\, e^{-i2\pi kx} \\ &= \sum_{j=1}^{n} w_j e^{-i2\pi kx_j}, \end{aligned} \tag{25}$$

by the sifting property of the Dirac delta function. A *periodogram* of $s(x)$ can now be defined as the plot of $\|c_k\|^2$ as a function of the frequency k.

ACKNOWLEDGMENTS

The authors would like to thank Andria Hunter for proofreading various versions of this paper, and Dave Mould for many insightful discussions. We are also grateful to Doug MacLellan for his permission to use the photograph appearing in the frame sitting on top of the desk in Figures 1(a) and 28. We are grateful as well to the Alias|Wavefront rendering team for providing encouragement in the final phases of preparing this paper. The authors acknowledge the gracious advice of the anonymous referees who were of great help in improving the manuscript.

REFERENCES

COOK, R. L. 1986. Stochastic sampling in computer graphics. *ACM Trans. Graphics 5,* 1, 51–72.

DAVIS, P. AND RABINOWITZ, P. 1984. *Methods of Numerical Integration*. Academic Press, New York.

DEUSSEN, O., HILLER S., VAN OVERVELD, C., AND STROHOTTE, T. 2000. Floating points: A method for computing stipple drawings. *Comput. Graph. Forum 19*, 3, C41–C50.

DIPPÉ, M. AND WOLD, E. 1985. Antialiasing through stochastic sampling. *Comput. Graph. 19*, 3 (Jul.), 69–78.

DRETTAKIS, G. AND FIUM, E. 1994. A fast shadow algorithm for area light sources using backprojection. In *ACM SIGGRAPH '94 Conference Proceedings* (*August*), Annual Conference Series, ACM, New York, pp. 223–230.

Espelid, T. and Sorevik, T. 1989. A discussion of a new error estimate for adaptive quadrature. *Bit 29*, 283–294.

Hart, D., Dutré, P., and Greenberg, D. 1999. Direct illumination with lazy visibility evaluation. In *ACM SIGGRAPH*, 147–154.

Heidrich, W., Brabec, S., and Seidel, H.-P. 2000. Soft shadow maps for linear lights. In *Proceedings of the 11th Eurographics Workshop on Rendering* (Brno, Czech Republic), Springer-Verlag/Wien, New York, *'99 Conference Proceedings* (*August*), Annual Conference Series, ACM, New York, pp. 269–280.

Kahaner, D., Moler, C., and Nash, S. 1989. *Numerical Methods and Software*. Prentice Hall, Englewood Cliffs, New Jersey.

Lyness, J. 1981. *Remarks about Performance Profiles.*. Tech. Mem. 369, Argonne National Lab, Argonne, Ill.

Lyness, J. and Kaganove, J. 1976. Comments on the nature of automatic quadrature routines. *ACM Trans. Math. Softw. 2*, 1 (Mar.), 65–81.

McCool, M. and Fiume, E. 1992. Hierarchical poisson disk sampling distributions. In *Graphics Interface '92,* Canadian Information Processing Society, Toronto, Ont., Canada, pp. 94–105.

Meredith-Jones, R. 2000. Point sampling algorithms for simulating motion blur. M.S. dissertation, University of Toronto, Toronto, Ont., Canada.

Nishita, T., Okamura, I., and Nakamae, E. 1985. Shading models for point and linear light sources. *ACM Trans. Graphics 4*, 2 (Apr.), 124–146.

Ouellette, M. J. 2002. Numerical methods for illumination integrals. Ph.D. dissertation, University of Toronto, Toronto, Ont., Canada. In preparation.

Ouellette, M. J. and Fiume, E. 1999a. Approximating the location of integrand discontinuities for penumbral illumination with area light sources. In *Proceedings of the 10th Eurographics Workshop on Rendering* (Grenada, Spain), Springer-Verlag/Wien, New York, pp. 213–224.

Ouellette, M. J. and Fiume, E. 1999b. Approximating the location of integrand discontinuities for penumbral illumination with linear light sources. In *Graphics Interface '99,* Canadian Information Processing Society, Toronto, Ont., Canada, pp. 66–75.

Pauly, M., Kollig, T., and Keller, A. 2000. Metropolis light transport for participating media. In *Proceedings of the 11th Eurographics Workshop on Rendering* (Brno, Czech Republic), Springer-Verlag/Wien, New York, pp. 11–22.

Poulin, P. and Amanatides, J. 1990. Shading and shadowing with linear light sources. In *Eurographics '90*, *Comput. Graph., Forum 9.3*, pp. 377–386.

Pratt, W. 1991. *Digital Image Processing*. John Wiley & Sons, New York.

Rushmeier, H., Ward, G., Piatko, C., Sanders, P., and Rust, B. 1995. Comparing real and synthetic images: Some ideas about metrics. In *Proceedings of the 6th Eurographics Workshop on Rendering* (Dublin, Ireland), Springer-Verlag/Wien, New York, pp. 213–222.

Schröder, P. 1993. Numerical integration for radiosity in the presence of singularities. In *Proceedings of the 4th Eurographics Workshop on Rendering* (Paris, France), Springer-Verlag/Wien, New York, pp. 177–184.

Shirley, P. 1991. Physically based lighting calculations for computer graphics. Ph.D. dissertation, University of Illinois.

Shreider, Y. 1966. *The Monte Carlo Method*. Pergamon Press, New York.

Sobol, I. 1994. *A Primer for the Monte Carlo Method*. CRC Press, London.

Spanier, J. and Maize, E. 1994. Quasi-random methods for estimating integrals using relatively small samples. *SIAM Review 36,* 1 (Mar.), 18–44.

Stewart, J. A. and Ghali, S. 1994. Fast computation of shadow boundaries using spatial coherence and backprojections. In *ACM SIGGRAPH '94 Conference Proceedings* (*August*), Annual Conference Series, ACM, New York, pp. 231–238.

Traub, J., Wasilkowski, G., and Woźniakowski, H. 1998. *Information-Based Complexity*. Academic Press, San Diego, Calif.

Ulichney, R. 1987. *Digital Halftoning*. MIT Press, Cambridge, MA.

Veach, E. and Guibas, L. 1995. Optimally combining sampling techniques for Monte Carlo rendering. In *ACM SIGGRAPH '95 Conference Proceedings* (*August*), Annual Conference Series, ACM, New York, pp. 419–428.

YAKOWITS, S., KRIMMEL, J., AND SZIDAROVSZKY, F. 1978. Weighted Monte Carlo integration. *SIAM J. Numer. Anal.15*, 6 (Dec.), 1289–1300.

ZIEMER, R. AND TRANTER, W. 1990. *Principles of Communications: Systems, Modulation and Noise*, 3rd ed. Houghton Mifflin, Boston.

Received March 2001; revised October 2001; accepted October 2001