# A Stochastic Algorithm for High Dimensional Integrals over Unbounded Regions with Gaussian Weight *

Alan Genz †
Department of Mathematics
Washington State University
Pullman, WA 99164-3113
alangenz@wsu.edu

John Monahan
Department of Statistics
North Carolina State University
Raleigh, NC 27695-8203
monahan@stat.ncsu.edu

## Abstract

Details are given for a Fortran implementation of an algorithm that uses stochastic spherical-radial rules for the numerical computation of multiple integrals over unbounded regions with Gaussian weight. The implemented rules are suitable for high dimensional problems. A high dimensional example from a computational finance application is used to illustrate the use of the rules.

## 1 Introduction

An important applications problem is to numerically compute integrals in the form

$$I(f) = (2\pi)^{-n/2} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} e^{-\mathbf{x}^t\mathbf{x}/2} f(\mathbf{x}) dx_1 dx_2 \cdots dx_n.$$

with $\mathbf{x} = (x_1, x_2, ..., x_n)^t$. There has been much recent interest (see for example, Paskov and Traub (1995), Papageorgiou and Traub (1996), and Joy, Boyle and Tan (1996)) in integrals in the $I(f)$ form that come from computational finance applications where the dimension is high ( e.g. $n > 100$).

This type of integration problem has traditionally been handled using Monte Carlo algorithms (see books by Hammersley and Handscomb, 1964, Stroud, 1971, and Davis and Rabinowitz, 1984). A simple Monte Carlo algorithm for estimating $I(f)$ might use

$$I(f) \approx I_N = \frac{1}{N} \sum_{j=1}^{N} f(\mathbf{x}_j),$$

where the points $\mathbf{x}_j$ are chosen with independent random coordinates $x_{ij} \sim Normal(0,1)$. This method is often effective, but it can have low accuracy and slow convergence. However, an important feature of simple Monte Carlo algorithms is the availability of practical and robust error estimates. Let $\hat{\sigma}$ denote the standard error for the sample. Then

$$\hat{\sigma} = \Big( \sum_{j=1}^{N} \frac{(f(\mathbf{x}_j) - I_N)^2}{N(N-1)} \Big)^{\frac{1}{2}},$$

*Accepted for publication in *J. Comp. Appl. Math.*
†Partially supported by NATO Collaborative Research Grant CRG 940139

and $\text{Prob}(|I(f) - I_N| < \alpha\hat{\sigma}) \approx \int_{-\alpha}^{\alpha} \frac{e^{-t^2/2}}{\sqrt{2\pi}} dt$.

The primary purpose of this paper is to describe an implementation of integration rules for $I(f)$ that have higher accuracy and better convergence properties than the simple Monte Carlo methods, but also have practical and robust error estimates. The basic integration rules are randomized deterministic integration rules that have higher degrees of accuracy than simple Monte Carlo rules. This randomization also allows the construction of simple Monte Carlo error estimates for the new stochastic rules. Randomized polynomial rules have been studied by Ermakov and Zolotukhin (1960), Hammersley and Handscomb (1964), Haber (1969) and Siegel and O'Brien (1985), but the work by these authors did not provide higher degree methods that had feasible implementations for multidimensional problems. The basic rules considered here are products of rules for the hyperspherical surface and rules for the Gaussian weighted interval $(-\infty, \infty)$. Randomization of these rules was first discussed by Genz and Monahan (1996) and Monahan and Genz (1997). This paper describes an implementation used in the Fortran subroutine RANRTH for these rules, and some generalizations, and considers the use of these rules for high dimensional problems.

## 2 Stochastic Spherical-Radial Rules

The description of the stochastic spherical-radial rules requires a change of variables to a spherical-radial coordinate system. Let $\mathbf{x} = r\mathbf{z}$, with $\mathbf{z}^t\mathbf{z} = 1$, so that $\mathbf{x}^t\mathbf{x} = r^2$, for $r \in [0, \infty)$. Then, for $n \geq 1$,

$$\begin{aligned} I(f) &= (2\pi)^{-n/2} \int_{\mathbf{z}^t\mathbf{z}=1} \int_0^\infty e^{-r^2/2} r^{n-1} f(r\mathbf{z}) dr d\mathbf{z} \\ &= \frac{(2\pi)^{-n/2}}{2} \int_{\mathbf{z}^t\mathbf{z}=1} \int_{-\infty}^\infty e^{-r^2/2} |r|^{n-1} f(r\mathbf{z}) dr d\mathbf{z}. \end{aligned}$$

Integration rules for $I(f)$ can be constructed as products of rules for the radial interval $(-\infty, \infty)$ with weight $e^{-r^2/2}|r|^{n-1}$, and rules for the surface of the unit n-sphere, $U_n$. Centrally symmetric rules will be used for the spherical surface, and in this case there is no inefficiency introduced by the use of the second expression above for $I(f)$, which allows for the use of symmetric radial rules. Averages of properly chosen samples of these rules can provide unbiased estimates for $I(f)$, and standard errors for the samples can be used to provide robust error estimates for the $I(f)$ estimates.

The radial rules developed by Genz and Monahan (1996) have the form

$$R(h) = \sum_{i=0}^{k} w_i (h(\rho_i) + h(-\rho_i))/2,$$

with $\rho_0 = 0$. These rules are designed to approximate the weighted infinite range integrals $T(h) = \int_{-\infty}^{\infty} e^{-r^2/2}|r|^{n-1}h(r)dr$. The weights are standard symmetric interpolatory weights defined by $w_i = T(\prod_{j=0,\neq i}^{l} \frac{r^2-\rho_j^2}{\rho_i^2-\rho_j^2})$, so that $R(h)$ has polynomial degree $2k+1$. Results in the Genz and Monahan papers generalize a method developed by Siegel and O'Brien (1985) to show how averages of samples of the rules $R(h)$ can be used to provide unbiased estimates for $T(h)$. For $k = 1$, the point $\rho_1$ needs to be chosen randomly with probability density proportional to $\rho_1^{n+1}e^{-\rho_1^2/2} \sim Chi(n+2)$ in order to provide unbiased degree three rules for $T(h)$. For $k = 2$, the points $\rho_1$ and $\rho_2$ need to be chosen randomly with joint probability density proportional to $(\rho_1\rho_2)^{n+1}e^{-(\rho_1^2+\rho_2^2)/2}(\rho_1 - \rho_2)^2(\rho_1 + \rho_2)$, in order to provide unbiased degree five rules for $T(h)$. Genz and Monahan show that if $r \sim Chi(2n+7)$

and $q \sim Beta(n+2, \frac{3}{2})$, then $\rho_1 = r\sin(\frac{\sin^{-1}(q)}{2})$ and $\rho_2 = r\cos(\frac{\sin^{-1}(q)}{2})$ have the correct joint distribution. It is a standard statistical procedure to generate random numbers with a *Chi* and *Beta* densities (see Fishman, 1996). Higher degree rules can also be constructed, but the joint probability density functions for the $\rho$ parameters for unbiased rules become more complicated and more difficult to sample from.

The stochastic spherical-radial rules derived by Genz and Monahan (1996) use rules for the spherical surface. Let

$$S(s) = \sum_{j=1}^{p} \tilde{w}_j s(\mathbf{z}_j),$$

with $\mathbf{z}_j^t \mathbf{z}_j = 1$ for all $j$, be an integration rule that approximates an integral of a function $s(\mathbf{z})$ over the surface $U_n$ of the unit $n$-sphere defined by $\mathbf{z}^t\mathbf{z} = 1$. If $Q$ is an $n \times n$ orthogonal matrix then

$$S_Q(s) = \sum_{j=1}^{p} \tilde{w}_j s(Q\mathbf{z}_j)$$

is also an integration rule for $s$ over $U_n$, because $||Q\mathbf{z}|| = ||\mathbf{z}||$. Furthermore, if $S$ has polynomial degree $d$, then so does $S_Q$, because $s(Q\mathbf{z})$ has the same polynomial degree as $s(z)$. Efficient spherical surface rules are given in the books by Stroud (1972) and Mysovskikh (1981), and the review paper by Mysovskikh (1980). If $Q$ is a random orthogonal matrix chosen with Haar distribution from the set of all matrices in the orthogonal group, then averages of rules $S_{Q_i}(s)$ provide unbiased degree $d$ estimates for $U_n$ integrals.

There are many choices that could be used for $S$. The book by Stroud (1971) lists several rules that use combinations of the standard Euclidean basis vectors for integrand evaluation points. Rules that use combinations of unit vertex regular simplex vertices were described by Mysovskikh (1980, 1981) and these rules are more efficient for the higher degrees than the rules in Stroud's book. These simplex vertex rules are used in RANRTH. The rules that are used with radial rules have degree one, three, five and seven.

A simple degree 1 rule is

$$S^1(s) = \frac{|U_n|}{2}(s(-\mathbf{z}) + s(+\mathbf{z})),$$

where $\mathbf{z}$ is any point on $U_n$.

The higher degree rules used by RANRTH use a set vertices $\{\mathbf{v}_j\}$ for a unit vertex regular $n$-simplex. One such set is given in Stroud (1971, page 345, with a minor misprint corrected), where

$$v_{i,j} = \begin{cases} 0 & \text{if} \quad i > j \\ \sqrt{\frac{(n+1)(n-i+1)}{n(n-i+2)}} & \text{if} \quad i = j \\ -\sqrt{\frac{n+1}{(n-i+1)n(n-i+2)}} & \text{if} \quad i < j \end{cases} \quad \text{for } i = 1, 2, \ldots, n;\ j = 1, 2, \ldots, n+1.$$

A degree three rule that uses simplex vertices is (Mysovskikh, 1980)

$$S^3(s) = \frac{|U_n|}{2(n+1)} \sum_{j=1}^{n+1} (s(-\mathbf{v}_j) + s(+\mathbf{v}_j)),$$

An efficient general degree five rule (Mysovskikh, 1980) has the form

$$S^5(s) = \frac{|U_n|}{2n(n+1)^2(n+2)} \quad ( \quad (7-n)n^2 \sum_{j=1}^{n+1} (s(-\mathbf{v}_j) + s(+\mathbf{v}_j)) + 4(n-1)^2 \sum_{k=1}^{n(n+1)/2} (s(-\mathbf{y}_k) + s(+\mathbf{y}_k)) \quad ).$$

The points $\mathbf{y}_k$ are projections of midpoints of unit vertex regular $n$-simplex edges onto $U_n$. The set $\{\mathbf{y}_k\}$ is $\{(\mathbf{v}_i + \mathbf{v}_j)/\sqrt{2(n-1)/n} \ : \ i < j \ \}$. The rule $S^5(s)$ uses $(n+1)(n+2)$ $s(\mathbf{z})$ values.

An efficient general degree seven rule (Mysovskikh, 1981) has the form

$$\begin{aligned} S^7(s) = \frac{|U_n|}{36n(n+1)^3(n+2)(n+4)} \quad ( \quad & n^3(9n^2 - 793n + 1800) \sum_{j=1}^{n+1} (s(-\mathbf{v}_j) + s(+\mathbf{v}_j)) \\ & + 144(n-1)^3(4-n) \sum_{k=1}^{n(n+1)/2} (s(-\mathbf{y}_k) + s(+\mathbf{y}_k)) \\ & + 486(n-2)^3 \sum_{k=1}^{(n-1)n(n+1)/6} (s(-\mathbf{u}_k) + s(+\mathbf{u}_k)) \\ & + (10n-6)^3 \sum_{k=1}^{n(n+1)} (s(-\mathbf{w}_k) + s(+\mathbf{w}_k)) \quad ). \end{aligned}$$

The points $\mathbf{u}_k$ are projections of the centroids of unit vertex regular $n$-simplex faces onto $U_n$. The set $\{\mathbf{u}_k\}$ is $\{(\mathbf{v}_i + \mathbf{v}_j + \mathbf{v}_l)/\sqrt{3(n-2)/n} \ : \ i < j < l \ \}$. The points $\mathbf{w}_k$ are projections of the selected edge points of the unit vertex regular $n$-simplex onto the surface of $U_n$. The set $\{\mathbf{w}_k\}$ is $\{(\mathbf{v}_i + 3\mathbf{v}_j)/\sqrt{(10n-6)/n} \ : \ i \neq j \ \}$. The rule $S^7(s)$ uses $(n+1)(n^2+8n+6)/3$ $s(\mathbf{z})$ values.

The stochastic rules for the unbounded radial interval and the spherical surface can be combined to give stochastic rules for the original integration region. The general form for a stochastic spherical-radial $(SR)$ rule with a degree $l$ rule for the spherical surface integral and a degree $m$ rule for the radial integral is

$$SR^{(l,m)}(f) = \sum_{i=0}^{\lfloor m/2 \rfloor} w_i (S_Q^l(f(-\rho_i \mathbf{z}) + S_Q^l(f(\rho_i \mathbf{z}))/2.$$

The spherical surface rules used in RANRTH are all centrally symmetric, and $\rho_0 = 0$, so

$$SR^{(l,m)}(f) = w_0 f(\mathbf{0}) + \sum_{i=1}^{\lfloor m/2 \rfloor} w_i S_Q^l(f(\rho_i \mathbf{z}))$$

There are various combinations for $l$ and $m$ that could be used in an implementation. The combinations used in RANRTH are discussed briefly here.

A degree one rule constructed from $S^1$ and $R^1$ is

$$SR^{(1,1)} = \frac{f(-\rho Q \mathbf{z}) + f(+\rho Q \mathbf{z})}{2}.$$

But here it is unnecessary to separately generate a random $Q$, a random $\mathbf{z}$ and a random $\rho$ because a combined vector $\mathbf{x}$ with components $x_i \sim Normal(0,1)$ already has the correct distribution. This

can be seen by considering $\mathbf{x} = \rho\mathbf{z}$ with $\mathbf{z} = \mathbf{x}/||\mathbf{x}||$. If $x_i \sim Normal(0,1)$, then $\rho \sim Chi(n)$ and $\mathbf{z}$ is uniform on $U_n$. Multiplying $\mathbf{z}$ by a random $Q$ does not change its distribution proerties. Therefore, all that is needed for $SR^{(1,1)}$ is to generate the components $x_i \sim Normal(0,1)$ and use the form

$$SR^{(1,1)} = \frac{f(-\mathbf{x}) + f(\mathbf{x})}{2}.$$

This is a well-known Monte Carlo integration rule for $I(f)$, sometimes referred to as crude Monte Carlo with antithetic variates.

A degree three rule constructed from the rules $S^3$ and $R^3$ is

$$SR^{(3,3)}(f) = (1 - \frac{n}{\rho^2})f(\mathbf{0}) + \frac{n}{\rho^2}S^3(f(\rho Q\mathbf{z})).$$

For this rule $\rho \sim Chi(n+2)$.

A degree five rule constructed from the rules $S^l$ and $R^5$ is

$$\begin{aligned} SR^{(l,5)}(f) = & \quad (1 - \frac{n(\rho^2 + \delta^2 - (n+2))}{\rho^2\delta^2})f(\mathbf{0}) \\ & + \frac{n(n+2-\delta^2)}{\rho^2(\rho^2-\delta^2)}S^m(f(\rho Q\mathbf{z})) \\ & + \frac{n(n+2-\rho^2)}{\delta^2(\delta^2-\rho^2)}S^m(f(\delta Q\mathbf{z})). \end{aligned}$$

For this rule, if $r \sim Chi(2n+7)$ and $q \sim Beta(n+2, \frac{3}{2})$, then $\rho = r\sin(\frac{\sin^{-1}(q)}{2})$ and $\delta = r\cos(\frac{\sin^{-1}(q)}{2})$ have the correct joint probability density for an unbiased rule. RANRTH implements $SR^{(l,5)}$ rule for $l = 5$ and $l = 7$.

The rules $SR^{(1,1)}$, $SR^{(3,3)}$, $SR^{(5,5)}$ and $SR^{(7,5)}$ require 2, $1 + 2(n+1)$ and $1 + 2(n+1)(n+2)$ and $1 + 2(n+1)(n^2+8n+6)/3$ $f$ values, respectively. Random samples of one of these rules can be generated, and the sample average used to estimate $I(f)$. The standard error for the sample can be used to provide an error estimate.

## 3 Some Implementation Details

All of the $SR$ rules with degree $> 1$ require random orthogonal matrices. If $V$ is used to denote the $n \times (n+1)$ matrix with simplex vertices $\mathbf{v}_i$ as columns, then the vertices for an $SR$ rule sample are the columns of the matrix $QV$, for some random orthogonal $n \times n$ matrix $Q$. The generation of the $Q$ matrices can contribute a significant overhead cost for high dimensional problems when standard algorithms for generating random orthogonal matrices are used. Efficient algorithms for generating random orthogonal matrices are given by Stewart (1980) and Anderson, Olkin and Underhill (1987). A variation on the Stewart algorithm that combines the generation of $Q$ with the multiplication $QV$ is used in the $SR$ rule implementation for RANRTH in the following form:

**Algorithm for Generating Randomly Transformed Simplex Vertices**

1. **Input** $n$.
2. $\mathbf{x} \leftarrow \mathbf{0}$.
3. Intialize the matrix $V$ of vertices for the standard unit vertex $n-$simplex.
4. For $k = n-1, n-2, \ldots, 1$
   (a) For $i = k, k+1, \ldots, n$ generate random $x_i \sim Normal(0,1)$.
   (b) $s \leftarrow ||\mathbf{x}||$, $x_k \leftarrow x_k - s$, $\beta \leftarrow 1/(x_k s)$
   (c) $V \leftarrow V + (\beta \mathbf{x})(\mathbf{x}^t V)$
5. **Output** $V$

This algorithm uses a sequence of random reflectors to modify the initial upper triangular simplex vertex matrix $V$. The first reflector vector $\mathbf{x}$ has only two nonzero elements and changes only the lower right $2 \times 3$ block of $V$. Subsequent reflector vectors have successively more nonzeros until the final reflector vector has all $n$ elements typically nonzero and changes every element of $V$. For a particular $k$ value, the cost of the modification of $V$ is $O((n-k)^2)$ floating point operations (flops), so the final cost for generating $QV$ is $O(n^3)$ flops, plus the cost of generating $O(n^2)$ $Normal(0,1)$ random numbers. With $SR^{(l,3)}$ rules, the columns of $QV$ are used for the evaluation points for $2(n+1)$ integrand values, so the overhead cost per integrand value is $O(n^2)$ flops plus the cost of generating $O(n)$ $Normal(0,1)$ random numbers. Once an integrand evaluation point is available, the cost for the evaluation of the integrand could be as low as $O(n)$, but not lower, because there are $n$ components for the input variable for the integrand, so the overhead cost for transforming the simplex vertices could be significant compared to the integrand evaluation cost when $n$ is large, for problems where the integrand evaluation cost is only $O(n)$. $SR^{(l,5)}$ rules have a $QV$ overhead cost per evaluation point that decreases to $O(n)$ flops, which should not be significant compared to the integrand evaluation cost, and this will also be true for $SR^{(l,7)}$ rules.

Algorithms in RANRTH for the $SR$ rules with $l, m > 1$ have a similar general structure.

**$SR^{(l,m)}$ Rule Integration Algorithm**

1. **Input** $\epsilon$, $n$, $l$, $m$, $f$ and $N_{max}$.
2. $N \leftarrow 0$, $I \leftarrow 0$, $E \leftarrow 0$, $F_0 \leftarrow w_0 f(\mathbf{0})$
3. **Repeat**
   (a) $N \leftarrow N+1$, $SR \leftarrow F_0$.
   (b) Generate a randomly transformed simplex vertex matrix $V$.
   (c) Generate the random $\rho$'s for an $SR^{(l,m)}$ rule.
   (d) For $i = 1, 2, \ldots, \lfloor m/2 \rfloor$: $SR \leftarrow SR + w_i S_Q^l(f(\rho_i \mathbf{z}))$
   (e) $D \leftarrow (SR - I)/N$, $I \leftarrow I + D$, $E \leftarrow (N-2)E/N + D^2$.

   **Until** $E < \epsilon^2$ or $N = N_{max}$.
4. **Output** $I \approx I(f)$, $\hat{\sigma}_E = \sqrt{E} \approx < |I - I(f)|$, and $N$.

The input $\epsilon$ is an error tolerance, the input $N_{max}$ provides a limit on the time for the algorithm, and the output $\hat{\sigma}_E$ is the standard error for the integral estimate $I$. The algorithm computes $I$ and $E$ using a modified version of a stable one-pass algorithm (Chan and Lewis, 1979) for sample means and variances. The unscaled sample standard error $\hat{\sigma}_E$ will usually provide an error bound with approximately 68% certainty. An appropriate scaling $\hat{\sigma}_E$ can be used to produce a higher degree of confidence in $I$. The error estimates obtained by scaling $\hat{\sigma}_E$ with this algorithm should be used with some caution for low $N$ values. These error estimates use the assumption that the sample averages $SR$ are approximately $Normal$. For large $N$, a scaled $\hat{\sigma}_E$ should provide a robust, statistically sound error estimate.

The supporting software for RANRTH is self-contained and includes functions for the generation of $Uniform$, $Normal$, $Chi$ and $Beta$ pseudo-random numbers. The $Normal$, $Chi$ and $Beta$ generators all use the Uniform generator developed by L'Ecuyer (1996). The $Normal$ generator uses a method developed by Marsaglia and Tsang (1984) and the $Chi$ and $Beta$ generators use methods described by Fishman (1996).

Many applications problems require the numerical approximation of several, closely related, integrals. The algorithm for the $SR$ rules is easily adapted to these problems by replacing the scalar $f$ in the original problem definition with a vector $\mathbf{f}$ and adding extra loops in the appropriate places. The Fortran subroutine RANRTH includes this extension, and it allows for an arbitrary number of components for $\mathbf{f}$.

The Fortran subroutine RANRTH also includes a restart feature, which allows a computation that has been stopped because of the work limit ($N_{max}$), to be continued. The results from successive calls to RANRTH with the same integrand are combined using a variance weighted averaging algorithm. To be specific, let $I_1, I_2, \ldots, I_k$ and $E_1, E_2, \ldots, E_k$ be results from $k$ successive applications of the general $SR$ rule integration algorithm. The final integral estimate $\tilde{I}_k$ and standard error $\hat{\sigma}_E = \sqrt{\tilde{E}_k}$ are obtained using

$$\tilde{I}_k = \frac{\frac{I_1}{E_1} + \frac{I_2}{E_2} + \cdots + \frac{I_k}{E_k}}{\frac{1}{E_1} + \frac{1}{E_2} + \cdots + \frac{1}{E_k}} \quad \text{and} \quad \tilde{E}_k = \frac{1}{\frac{1}{E_1} + \frac{1}{E_2} + \cdots + \frac{1}{E_k}}.$$

Algebraically equivalent formulas used by RANRTH that require results from only one previous step are $\tilde{I}_k = \tilde{I}_{k-1} + W_k(I_k - \tilde{I}_{k-1})$ and $\tilde{E}_k = W_k E_k$, with $W_k = \tilde{E}_{k-1}/(E_k + \tilde{E}_{k-1})$, for $k > 1$.

The general algorithm, and RANRTH, can easily be parallelized for use with large scale problems. Depending on the target parallel computer architecture, parallelization could be introduced at any one of several levels. The easiest parallelization is a very large grained parallelization introduced at a main program level, where parallel calls to RANRTH can be used, and results combined by the main program. A large grained parallelization within RANRTH can be introduced by parallelizing the loop at step 3 in the general $SR$ rule algorithm. Finer grained parallelism can be introduced inside the procedure for the computation of the $U_n$ rules, or at the integrand evaluation level.

# 4 An Example

The example in this section has been used by Caflisch and Morokoff (1996), and Ninomiya and Tezuka (1996) to illustrate the use of a quasi-Monte Carlo method for a computational finance problem. The integrand, which has a Gaussian weight, is

$$P(\mathbf{x}) = C \sum_{k=1}^{n} \frac{((1 - w_k(\mathbf{x})) + w_k(\mathbf{x})c_k) \prod_{j=1}^{k-1}(1 - w_j(\mathbf{x}))}{\prod_{j=0}^{k-1}(1 + i_k(\mathbf{x}))},$$

where $i_k(\mathbf{x}) = i_0 K_0^k e^{\hat{\sigma}(x_1+x_2+\cdots+x_k)}$, $K_0 = e^{-\hat{\sigma}^2/2}$, $w_k(\mathbf{x}) = K_1 + K_2 \tan^{-1}(K_3 i_k(\mathbf{x}) + K_4)$ and $c_k = \sum_{j=0}^{n-k}(1+i_0)^{-1}$, with an initial interest rate $i_0$ and constants $(K_1, K_2, K_3, K_4, \hat{\sigma})$ given. $I(P)$ is the present value of a security backed by mortgages of length $n$ months with a fixed monthly interest rate $i_0$. A related integral $I(A)$, which determines the average life for the mortgages, was $A(\mathbf{x})$ defined by

$$A(\mathbf{x}) = \sum_{k=1}^{n} k w_k(\mathbf{x}) \prod_{j=1}^{k-1} (1 - w_j(\mathbf{x})).$$

Caflisch and Morokoff (1996) studied two types of problem: a "nearly linear" problem and a "nonlinear" problem. With the "nearly linear" case $(K_1, K_2, K_3, K_4) = (0.01, -0.005, 10, 0.5)$, and in the "nonlinear" case $(K_1, K_2, K_3, K_4) = (0.04, 0.0222, -1500, 7)$. In both cases $C = 1$, $i_0 = 0.007$ and $\hat{\sigma} = 0.02$. Ninomiya and Tezuka (1996) used $C = 2000$, $i_0 = 0.075/12$, $\hat{\sigma} = 0.2$ and $(K_1, K_2, K_3, K_4) = (0.24, 0.134, -26.11, 12.72)$. The problem was considered for total time lengths as long as thirty years ($n = 360$ months). Results in Figures 4.1-4.6 are given for the Caflisch and Morokoff cases only, for $n = 90, 180, 360$. If $M$ is the total number of $f$ values needed for the integral estimate $I_M$, $\log_{10}(\hat{\sigma}_E/I_M)$ is plotted as a function of $log_2$(M), for each ($n$, case) pair.

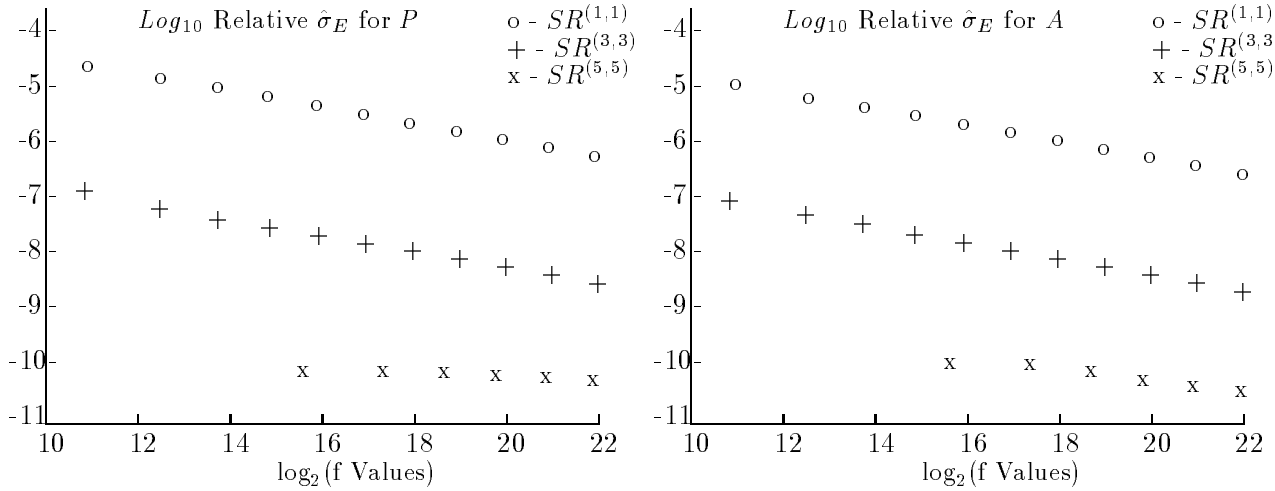


Fig. 4.1: "Nearly Linear" Problem, with $n = 90$.

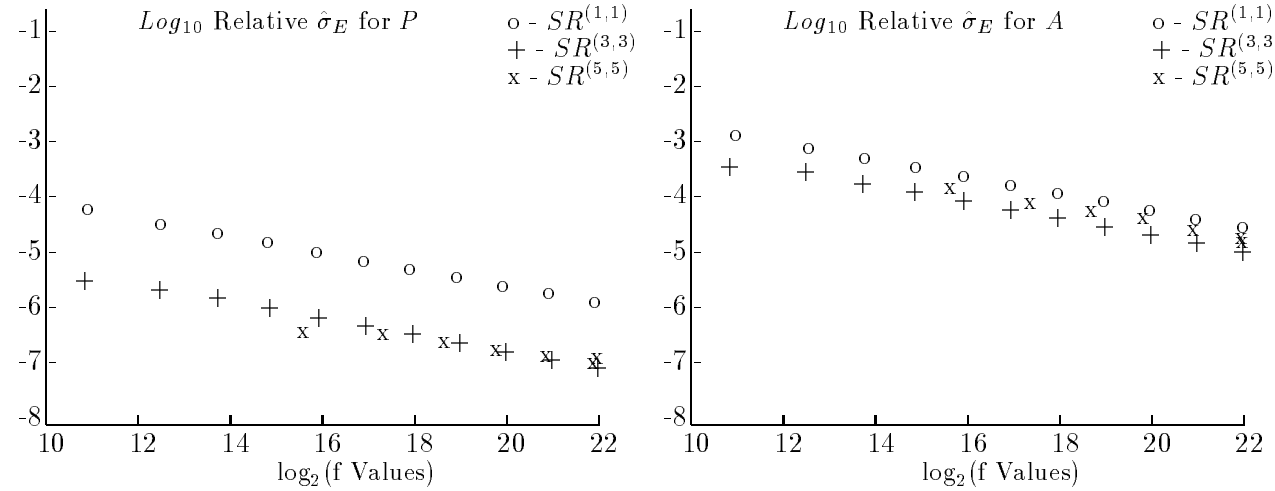


Fig. 4.2: "Nonlinear" Problem, with $n = 90$.

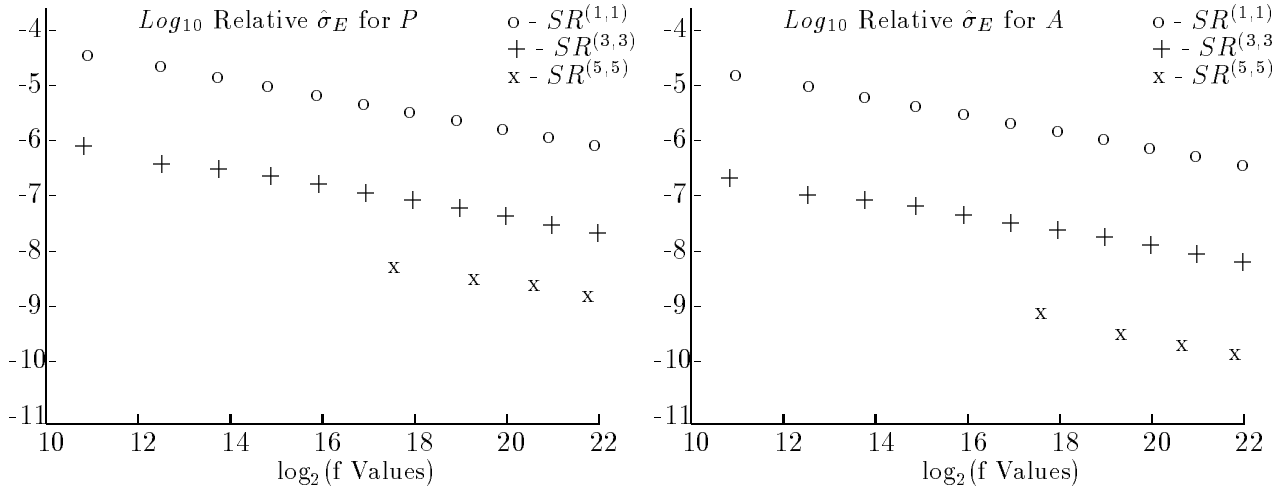


Fig. 4.3: "Nearly Linear" Problem, with $n = 180$.

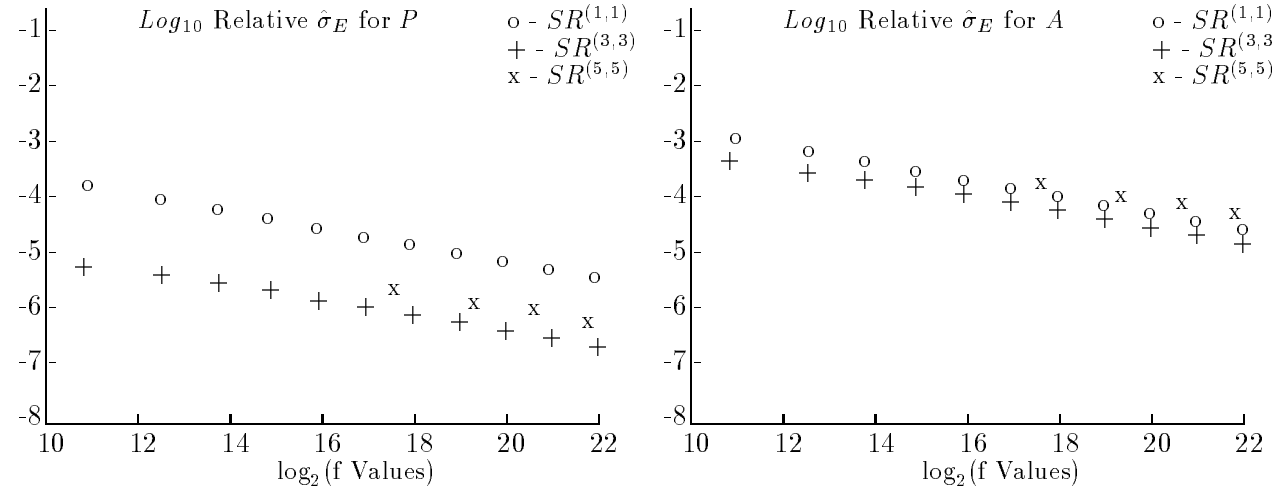


Fig. 4.4: "Nonlinear" Problem, with $n = 180$.

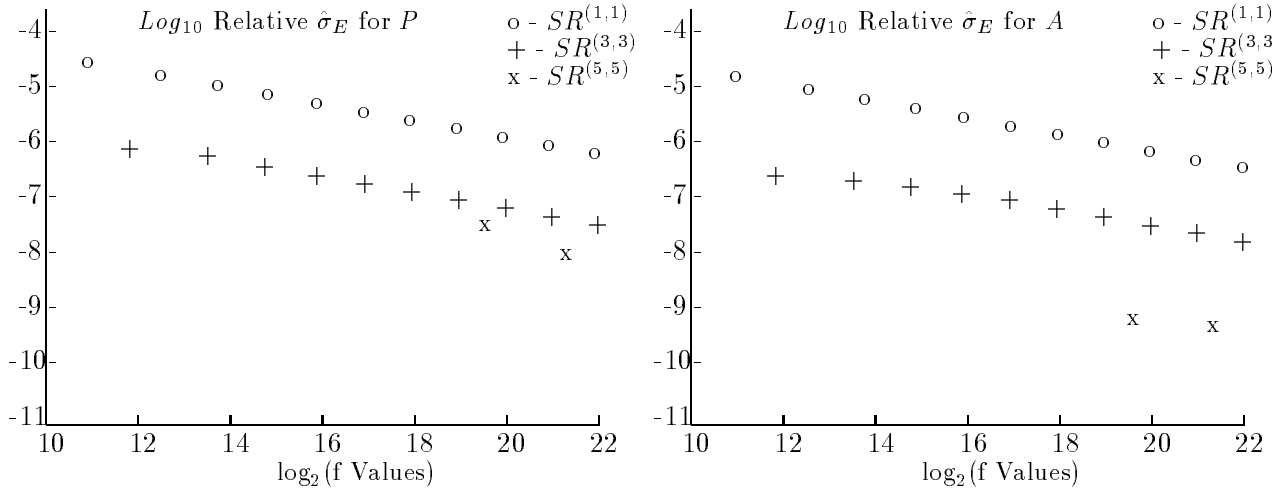


Fig. 4.5: "Nearly Linear" Problem, with $n = 360$.

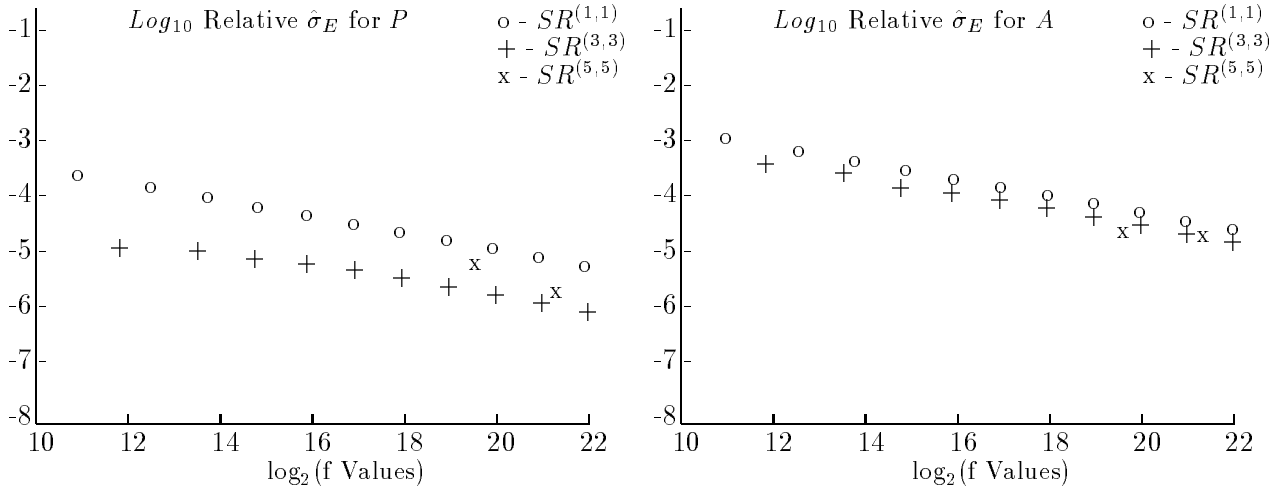


Fig. 4.6: "Nonlinear" Problem, with $n = 360$.

The results show that the $SR$ rules can be feasibly implemented for high dimensional problems, and that they can provide much more accurate results than simple Monte Carlo methods for comparable work. The results for the $SR^{(5,5)}$ rules are given only for larger sample sizes because these rules require $2(n+1)(n+2)+1$ $f$ values for the first sample and $2(n+1)(n+2)$ $f$ values for subsequent samples, with a minimum of two samples needed to produce an error estimate. For example, with $n = 180$, 131769 $f$ values are needed to start. These rules did not perform better than the $SR^{(3,3)}$ rules for the "nonlinear" case for $n = 180$ and $n = 360$. The accuracies that were obtained are comparable to accuracies obtained by Caflisch and Morokoff using quasi-Monte Carlo methods, with similar numbers of integrand evaluations. Caflisch and Morokoff used a "Brownian bridge" change of variables designed to precondition the problem so that most of the variation in the integrand was caused by changes in the variables with lowest indices. These low index variables had better uniformity properties than the high indexed variables for the quasi-Monte Carlo method that Caflisch and Morokoff used. It is unlikely that $SR$ rule results would benefit from such a change of variable, unless it reduced the overall variance in the integrand. However, it should be relatively straightforward to develop a quasi-random version of the $SR$ rule algorithm described here. A possible problem is the generation of quasi-random orthogonal matrices, but this has already been studied by Fang and Wang (1994). It should be possible to use techniques described in their book for the development of "quasi-stochastic" $SR$ rule algorithms. This is currently under investigation.

# References


Anderson, T.W., Olkin, I. and Underhill L.G. (1987), Generation of Random Orthogonal Matrices, *SIAM Journal on Scientific Computation* **8**, pp. 625–629.

Chan, T. F. and Lewis, J. G. (1979), Computing Standard Deviations: Accuracy, *Communications of the ACM* **22**, pp. 526–531.

Caflisch, R. and Morokoff, W. (1996), Quasi-Monte Carlo Computation of a Finance Problem, in *Proceedings of the Workshop on Quasi-Monte Carlo Methods and Their Applications*, Statistics Research and Consultancy Unit, Hong Kong Baptist University, pp. 15–30.

Davis, P.J., and Rabinowitz, P. (1984), *Methods of Numerical Integration*, Academic Press, New York.

Ermakov, S.M. and Zolotukhin, V.G. (1960), in *Theory of Probability and Applications, Amer. Math. Soc. Transl. 5*, pp. 428-431.

Fang, K.-T. and Wang, Y. (1994), *Number Theoretic Methods in Statistics*, Chapman and Hall, London.

Fishman, G.S. (1996), *Monte Carlo*, Springer-Verlag, New York.

Genz, A., and Monahan, J. (1998), Stochastic Integration Rules for Infinite Regions, *SIAM Journal on Scientific Computation*, **19**, pp. 426–439.

Haber, S. (1969), Stochastic Quadrature Formulas, *Math. Comp.* **23**, pp. 751–764.

Hammersley, J.M. and Handscomb, D.C. (1964), *Monte Carlo Methods*, Chapman and Hall, London.

Joy, C., Boyle, P. and Tan, T. (1996), Quasi-Monte Carlo Methods in Numerical Finance, *Management Science* **42**, pp. 926–938.

L'Ecuyer, Pierre (1996) Combined Multiple Recursive Random Number Generators, *Operations Research* **44**, pp. 816–822.

Marsaglia, G. and Tsang, W.W. (1984), A Fast, Easily Implemented Method for Sampling from Decreasing or Symmetric Unimodal Density Functions, *SIAM J. Sci. Stat. Comput.* **5**, pp. 349–359.

Monahan, J., and Genz, A. (1997), Spherical-Radial Integration Rules for Bayesian Computation, *Journal of the American Statistical Association* **92**, pp. 664–674.

Mysovskikh, I. P. (1980), The Approximation of Multiple Integrals by using Interpolatory Cubature Formulae, in *Quantitative Approximation*, R.A. DeVore and K. Scherer (Eds.), Academic Press, New York, pp. 217–243.

Mysovskikh, I. P. (1981), *Interpolatory Cubature Formulas* (Russian), Izd 'Nauka', Moscow-Leningrad.

Siegel, A.F. and O'Brien, F. (1983), Unbiased Monte Carlo Integration Methods with Exactness for Low Order Polynomials, *SIAM. J. Sci. Stat. Comput.* **6**, pp. 169–181.

Ninomiya, S. and Tezuka, S (1996), Toward Real-Time Pricing of Complex Financial Derivatives, *Applied Mathematical Finance* **3**, pp. 1–20.

Papageorgiou, A. and Traub, J. (1996), Beating Monte Carlo, *Risk*, **9**:6, pp. 63–65.

Paskov, S. and Traub, J. (1995), Faster Valuation of Financial Derivatives, *Journal of Portfolio Management*, Fall 1995, pp. 113–120.

Stewart, G.W. (1980), The Efficient Generation of Random Orthogonal Matrices with An Application to Condition Estimation, *SIAM J. Numer. Anal.* **17**, pp. 403–409.

Stroud, A. H. (1971), *The Approximate Calculation of Multiple Integrals*, Prentice Hall, Englewood Cliffs, New Jersey.