

ESTIMATION METHOD AND SYSTEM FOR COMPLEX SECURITIES
USING LOW-DISCREPANCY DETERMINISTIC SEQUENCES

The United States Government has certain rights to this invention pursuant to grants CCR-91-14042 and IRI-92-12597 awarded by the National Science Foundation, and to grant AFOSR-91-0347 awarded by the U.S. Air Force.

5

This application is a continuation-in-part of pending application No. 08/____,____ which was filed July 30, 1997 as a continuation application of now-abandoned application No. 08/285,902 filed August 4, 1994.

10

Background of the Invention

The invention relates to financial securities transactions such as buying and selling of stocks, bonds and financial derivative instruments, including futures, options, collateralized mortgage obligations, single-premium deferred annuities and insurance contracts, for example.

In financial securities transactions, which includes the initial sale, the value of a security may be estimated, e.g., based on expected future cash flow. Such cash flow may depend on variable interest rates, for example, and these and other relevant variables may be viewed as stochastic variables.

It is well known that the value of a financial security which depends on stochastic variables can be estimated in terms of a multi-dimensional integral. The dimension of such an integral may be very high.

990 4 133 - 7

19 AUG 1999



In collateralized mortgage obligations (CMO), for example, instruments or securities variously called tranches, shares, participations, classes or contracts have cash flows which are determined by dividing and distributing the cash flow of an underlying collection or pool of mortgages on a monthly basis according to pre-specified rules. The present value of a tranche can be estimated on the basis of the expected monthly cash flows over the remaining term of the tranche, and an estimate of the present value of a tranche can be represented as a multi-dimensional integral whose dimension is the number of payment periods of the tranche. For a typical instrument with a 30-year term and with monthly payments, this dimension is 360.

Usually, such a high-dimensional integral can be evaluated only approximately, by numerical integration. This involves the generation of points in the domain of integration, evaluating or "sampling" the integrand at the generated points, and combining the resulting integrand values, e.g., by averaging. Well known for numerical integration in securities valuation is the so-called Monte Carlo method in which points in the domain of integration are generated at random.

With integrands arising in financial securities valuation, the computational work in combining the sampled values is negligible as compared with producing the integrand values. Thus, numerical integration methods in securities valuation may be compared based on the number of

This Sheet Has Been Replaced



samples required for obtaining a sufficiently accurate approximation to the integral.

Summary Of The Invention

5 A preferred method for estimating the value of a financial security involves numerical integration unlike Monte Carlo integration in that an integrand is sampled at deterministic points having a low-discrepancy property. As compared with the Monte Carlo method, significant advantages
10 are realized with respect to speed, accuracy, and dependability.

 A computed estimate can be used in setting an offering price, e.g., somewhat higher than the estimate. An estimate can be used also in deciding whether to buy, hold or sell
15 the security. For example, if the estimate exceeds the price at which the security is being offered for sale, a "buy" recommendation is appropriate.

 A computed estimate can represent the value of other complex securities, e.g. a debt or insurance instrument.

20

Brief Description Of The Drawing

 Fig. 1 is a schematic of a programmed computer system in accordance with a preferred embodiment of the invention.

 Fig. 2 is a graphic representation of performance data
25 obtained in computer trial runs with an exemplary embodiment of the invention as compared with two Monte Carlo computations.

 Fig. 3 is a flow diagram of a preferred embodiment of a method in accordance with the invention.

This Sheet Has Been Replaced





Fig. 4 is a flow diagram of a preferred further embodiment of the invention, including the use of multiple processors.

Fig. 5 is a flow diagram of initially selling a complex security in accordance with a preferred embodiment of the invention.

Fig. 6 is a flow diagram of buying/holding/selling a complex security in accordance with a preferred further embodiment of the invention.

Further included is an Appendix with two computer algorithms in "C" source language, respectively for computing Sobol points and Halton points. For a description of C, see B.W. Kernighan et al., The Programming Language C, Prentice-Hall, 1978.

Detailed Description Of Preferred Embodiments

Fig. 1 shows a stored-program computer 11 connected to input means 12, e.g., a keyboard, for entering financial securities data, and connected to output means 13, e.g., a visual display device, for displaying an estimated value of the financial security. The computer 11 includes a working memory 111 (M), a low-discrepancy deterministic point generator 112 (P), an integrand evaluator 113 (E), and an integrand-value combiner 114 (C).

In estimating the value of a multi-dimensional integral in financial securities valuation, a multivariate integrand is sampled at points corresponding to a low-discrepancy deterministic sequence of points in the multivariate unit cube as defined below. If the multivariate unit cube is



also the domain of integration, the points of the low-discrepancy deterministic sequence can be used as sample points directly. In the case of a more general domain of integration, sample points correspond to points of a low-
 5 discrepancy deterministic sequence in the multivariate unit cube via a suitable transformation or mapping.

When a sufficiently large number of sampled values has been computed, an approximation of the integral is obtained by suitably combining the computed values, e.g., by
 10 averaging or weighted averaging.

In the d -dimensional unit cube $D = [0, 1]^d$, a low-discrepancy deterministic sequence $z_1, z_2, \dots$ of points in D can be characterized as follows:

For a point $t = [t_1, \dots, t_d]$ in D , define
 15 $[0, t) = [0, t_1) \times \dots \times [0, t_d),$

where $[0, t_i)$ denotes a left-closed, right-open interval, and denote with $\chi_{[0, t)}(\cdot)$ the characteristic or indicator function of $[0, t)$. For points $z_1, \dots, z_n$ in D , define

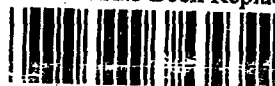
$R_n(t; z_1, \dots, z_n) = (\sum_{k=1}^n \chi_{[0, t)}(z_k)) / n - t_1 t_2 \dots t_d,$
 20 and define the discrepancy of $z_1, \dots, z_n$ as the L_∞ -norm of the function $R_n(\cdot; z_1, \dots, z_n)$, i.e.,

$$\|R_n(\cdot; z_1, \dots, z_n)\|_\infty = \sup_{t \in D} |R_n(t; z_1, \dots, z_n)|.$$

The sequence $z_1, z_2, \dots$ is said to be a low-discrepancy deterministic sequence provided

25 $\|R_n(\cdot; z_1, \dots, z_n)\|_\infty = O((\log n)^d / n).$

Low-discrepancy deterministic sequences are described in the published literature; see, e.g., H. Niederreiter, "Random Number Generation and Quasi-Monte Carlo Methods", CBMS-NSF, 63, SIAM, Philadelphia, 1992 and S. Paskov,



"Average Case Complexity of Multivariate Integration for Smooth Functions", Journal of Complexity, Vol. 9 (1993), pp. 291-312. Well-known examples of low-discrepancy deterministic sequences are the so-called Hammersley points, Halton points, Sobol points, and hyperbolic-cross points.

For illustration, in the case of Sobol points in a single dimension ($d=1$), a constructive definition may be given as follows: Choose a primitive polynomial

$$P(x) = x^n + a_1x^{n-1} + \dots + a_{n-1}x + 1$$

(whose coefficients a_i are either 0 or 1) and define so-called direction numbers v_i , $i > n$ by the following recurrence formula:

$$v_i = a_1v_{i-1} \oplus a_2v_{i-2} \oplus \dots \oplus a_{n-1}v_{i-n+1} \oplus v_{i-n} \oplus (v_{i-n}/2^n),$$

where $\oplus$ denotes a bit-by-bit "exclusive or" operation.

The initial numbers $v_1 = m_1/2$, ..., $v_n = m_n/2^n$ can be chosen freely provided m_i is odd and $0 < m_i < 2^i$ for $i = 1, 2, \dots, n$.

Using the direction numbers v_i so defined, now define the one-dimensional Sobol sequence $x_1, x_2, \dots$ by

$$x_k = b_1v_1 \oplus b_2v_2 \oplus \dots \oplus b_wv_w, \quad k \geq 0$$

where $k = \sum_{i=0}^{\lfloor \log k \rfloor} b_i 2^i$ is the binary representation of k .

For higher dimensions ($d > 1$), the first d primitive polynomials $P_1, P_2, \dots, P_d$ are used. If $\{x_k^i\}_{k=1}^\infty$ denotes the one-dimensional Sobol sequence generated by the

polynomial P_i , the d -dimensional Sobol points are defined as $x_k = (x_k^1, x_k^2, \dots, x_k^d)$.

While this definition can be implemented as a computer algorithm directly, faster techniques are known which produce these points in a "shuffled" or permuted sequence.



In particular, this applies to the computer algorithm given in the Appendix.

For specificity in the following, a preferred embodiment of the invention is described as applied to a collateralized mortgage obligation known as CMO FN, 89-23. This has thirty-year maturity and consists of the following tranches:

PAC tranches 23-A, 23-B, 23-C, 23-D, 23-E
supporting tranches 23-G, 23-H, 23-J
10 residual tranche 23-R
accrual tranche 23-Z

The monthly cash flow is divided and distributed according to pre-specified rules which are included in a formal prospectus. Some of the basic rules may be stated as
15 follows:

First from the monthly cash flow, the coupon is paid to the tranches. The remaining amount, called Principal Distribution Amount, is used for repayment of the principal. Prior to a fixed date in the future, the Principal Amount
20 will be allocated sequentially to the tranches 23-G, 23-H, 23-J and 23-Z. After that date, the Principal Distribution Amount will be allocated sequentially to the tranches 23-A, 23-B, 23-C, 23-D and 23-E according to a planned schedule. Any excess amount of the Principal Distribution Amount over
25 the planned schedule will be allocated sequentially to the tranches 23-G, 23-H, 23-J and 23-Z. A distribution of principal of the tranche 23-R will be made only after all other tranches have been retired.



In deriving an estimate for the present value of the security at the time of issue, the following notation is used below:

5 C - the monthly payment on the underlying mortgage pool;

i_k - the projected interest rate in month k , $k = 1, 2, \dots, 360$;

w_k - the percentage of mortgages prepaying in month k ;

$a_{360-k+1}$ - the remaining annuity after month k .

10 A remaining annuity a_k can be expressed as

$$a_k = 1 + v_0 + \dots + v_0^{k-1}, \quad k = 1, 2, \dots, 360,$$

with $v_0 = 1/(1+i_0)$, where i_0 is the current monthly interest rate. Thus, after k months, the remaining amount of principal borrowed is $C \cdot a_k$.

15 It is assumed that the interest rate i_k can be expressed as

$$i_k = K_0 \exp(\xi_k) i_{t-1},$$

where $\exp(\cdot)$ denotes exponentiation and where

20 $\xi_1, \xi_2, \dots, \xi_{360}$ are independent, normally distributed random variables with mean 0 and variance σ^2 , and K_0 is a given constant. For the present example, $\sigma^2 = 0.0004$ is chosen.

It is assumed further that w_k as a function of i_k can be computed as

$$w_k = K_1 + K_2 \arctan(K_3 i_k + K_4),$$

25 where K_1, K_2, K_3, K_4 are given constants.

Under these assumptions, the cash flow in month k , $k = 1, 2, \dots, 360$ is

$$C(1-w_1(\xi_1)) \dots (1-w_{k-1}(\xi_1, \dots, \xi_{k-1})) (1-w_k(\xi_1, \dots, \xi_k) + w_k(\xi_1, \dots, \xi_k) a_{360-k+1}),$$

This Sheet Has Been Replaced



where

$$w_k(\xi_1, \dots, \xi_k) = K_1 + K_2 \arctan(K_3 K_0^k i_0 \exp(\xi_1 + \dots + \xi_k) + K_4).$$

This cash flow is distributed according to the rules of FN, 89-23. Then, the cash flow for each of the tranches is multiplied by the discount factor

$$v_1(\xi_1) \dots v_k(\xi_1, \dots, \xi_k), \text{ with}$$

$v_j(\xi_1, \dots, \xi_j) = 1/(1 + K_0^j i_0 \exp(\xi_1 + \dots + \xi_j))$, $j=1, 2, \dots, 360$, to find the present value for month k . Summing of the present values for every month gives the present value PV_T , for each tranche T .

The expected or estimated value,

$$E(PV_T) = E(PV_T(\xi_1, \dots, \xi_{360})),$$

upon a change of variables is represented by

$$E(PV_T) = \int_D PV_T(y_1(x_1), \dots, y_{360}(x_{360})) dx_1 \dots dx_{360},$$

where

$$x_1 = (2\pi\sigma)^{-1/2} \int_{-\infty}^{y_1} \exp(-t^2/(2\sigma)) dt.$$

Thus, for each tranche T , a 360-variate integrand has to be integrated over the 360-dimensional unit cube, .

After generating a point

$$(x_1, x_2, \dots, x_{360})$$

of a low-discrepancy deterministic sequence in the unit cube, the point

$$(y_1, y_2, \dots, y_{360})$$

is computed by finding the value of the inverse normal cumulative distribution function at each x_k . Then, for each tranche T , the function value

$$PV_T(y_1, y_2, \dots, y_{360})$$

is computed. These are the function values used in numerical integration.

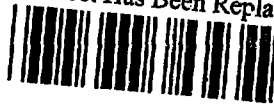
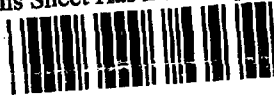


Fig. 2 shows results from trial runs for CMO FN, 89-23 with a preferred method using Sobol points generated by the corresponding computer algorithm given in the Appendix, as compared with Monte Carlo integration. Two Monte Carlo
5 computations were carried out, with different "seeds" or starting values of a congruential pseudo-random number generator known as RAN2; see W. Press et al., Numerical Recipes in C, Cambridge University Press, 1992. It is apparent that the preferred method reaches a steady value
10 more rapidly. In this and other trial runs, with typical requirements of precision and confidence, a speed-up by a factor of 3 to 5 was realized as compared with Monte Carlo integration. Much greater speed-up can be expected when higher precision or/and higher levels of confidence are
15 sought.

In a further trial run with CMO FN, 89-23, instead of Sobol points, Halton points were used as generated by the corresponding computer algorithm given in the Appendix. It is felt that Sobol points may be preferred over Halton
20 points for integrals of high dimension. However, this preference may not apply in the case of lower-dimensional integrals, e.g., with dimension up to 5 or so.

A computation as described may be terminated after a predetermined number of function evaluations.
25 Alternatively, e.g., after every function evaluation or after a predetermined incremental number of function evaluations, a current approximation may be compared with one or several preceding approximations, for termination once a suitable condition depending on the difference



between approximations is met. Such termination criteria may be called automatic. Automatic termination is particularly reliable where a sequence of approximations settles down smoothly; see, e.g., the curve in Fig. 2
5 corresponding to Sobol points.

Advantageously in computing function values, a cluster or network of multiple parallel processors or workstations can be used. This may involve a master or host processor providing points of a low-discrepancy sequence to slave
10 processors and combining function values returned by the slave processors into an approximate value for the integral. Thus, the computation can be speeded up in proportion to the number of processors used.

Advantageous further, in combination with numerical
15 integration as described above, is the use of error reduction techniques analogous to variance reduction in Monte Carlo integration as described, e.g., by M. Kalos et al., Monte Carlo Methods, John Wiley & Sons, 1986. This may involve a change of variables or/and variation reduction,
20 for example.

The specific embodiment described above, involving valuation of a collateralized mortgage obligation, can be interpreted as follows: The definite integral represents the expected value of the sum of a series of payments on the
25 obligation, namely the regular payments of mortgage principal and interest, plus any mortgage prepayments. The likelihood of prepayment depends in part on the prevailing level of interest for mortgage refunding.



For each payment, the value of the payment is weighted by the probability of all interest rate scenarios up to the time of the payment. Thus, for example, the amount of next month's payment will be valued based on the probability of all possible interest rates likely to prevail next month, and the impact of each of those possibilities on the likelihood of prepayment of principal. For the month after, valuation takes into account the interest rate scenarios of both months, their probabilities of occurrence, and the effect of interest rates on prepayment. In addition to next month's valuation, the month-after estimate will include an additional term corresponding to the month-after payment, represented by an integral over the possibilities of the two-month process. Further monthly payments, i.e. third, fourth et seq., are represented analogously. With each additional integral term representing a scheduled or unscheduled payment amount, the over-all value of the CMO is represented by the sum of the terms.

The value of the CMO can be decomposed further, e.g., by valuing the interest payment and the principal payment separately in each period to represent an "interest-only tranche" and a "principal-only tranche". Thus, the value of the CMO can be represented by the sum of a yet-greater number of integrals. Conveniently, the value of each of the individual integrals in the sum can be estimated as described above, using a low-discrepancy deterministic sequence of sample points.

A sum of payment integrals can also be used to represent the value of a portfolio of two or more CMOs,



giving the same result as a single integral over the sum of the payments. The range of possibilities over which the integration is effected is the same in both cases.

5 A CMO portfolio can be augmented in other ways, e.g., by the inclusion of one or several bonds. For valuation purposes, a high-grade bond differs from a CMO in that prepayment will take place for the entire issue or not at all. Again, valuation depends on the fluctuation of interest rates, taking into account all possible future
10 interest rate scenarios. This applies whether there is one payment (from a single CMO), several payments (from several CMOs), or payments from a combination of CMOs and bonds.

Other than CMOs and bonds, a complex security portfolio may include securities such as single-premium deferred
15 annuities (SPDA). An SPDA is a contract sold by an insurance company to an annuitant investor. Under the contract, the insurance company takes money from the investor, invests this money in a portfolio of various investments, and credits a portion of the investment
20 proceeds to the investor's account. The SPDA is an asset of the insurance company. There is an initial investment by the company in the cost of producing the contract. There is a predictable return for the company on the investment portfolio, of interest and principal payments by the various
25 borrowers less the contractual crediting to the annuitant's account. The unpredictable factor in the valuation of the portfolio is again random movements of interest rates, as these influence prepayment of investment assets as well as unexpected withdrawals and cash demands by the annuitant.



The same methodology of valuation can be used after inclusion of the SPDA asset in the complex security.

Valuation, by the insurance company, of a portfolio including an SPDA contract involves estimating the value of the contract itself. In pricing a new contract, its value is estimated in advance of issue by making assumptions about the kinds of assets that will be in the portfolio along with the SPDA. The pricing of the SPDA before issue and its valuation after issue are equivalent to the calculation presented above for a CMO.

A life insurance policy, or any other kind of insurance contract differs from the above in that, in addition to the interest rate variable, there is an additional stochastic variable representing health risk, mortality risk or fire risk, for example. Under inclusion of this variable, the same techniques as described above can be used for valuation.

Many other types of securities are amenable to corresponding valuation and estimation, where each period's prospective payment is valued depending upon all previous values of a stochastic variable. For example, in the case of an interest rate option, the stochastic variable is interest rate and the payment is the value of a bond. In the case of a stock option, the variable is the price of the stock. In the case of a currency option, the variable is the relative value of two currencies.



A P P E N D I X

```

/*****

```

```

void sobol(int n)

```

This is the function sobol for generating Sobol points. It returns the n-th d-dimensional Sobol point. (The point (0,0, ..., 0) is skipped.) The point is implicitly returned through the array x. The function sobseq from Numerical Recipes, 1992 was used as a basis, but changed significantly to accommodate the parallel distributed approach. The function sobol can generate Sobol points skipping an initial part of the sequence. The constant MAXDIM, see Numerical Recipes, is extended to 360 which required adding more initializing data to the arrays ip (the primitive polynomials), mdeg (their degrees), and iv (the initial direction numbers). The polynomial x is used to generate the first coordinate of Sobol points.

```

*****/

```

```

#include "nrutil.h"
#define MAXBIT 30
#define MAXDIM 360

```

```

extern int d; /* actual dimension of the points */
extern double *x; /* This returns implicitly the n-th Sobol point in x */

```

```

void sobol(int n)

```

```

{
    int j,k,l;
    unsigned long i,im,ipp;
    static double fac;
    static unsigned long in,ix[MAXDIM+1],*iu[MAXBIT+1];
    static unsigned long mdeg[MAXDIM+1]={0,MAXBIT,1,2,3,3,4,4,
/*16*/
        5,5,5,5,5,5,
        6,6,6,6,6,6,
        7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,
        8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,
        9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,
        9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,
        9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,
/*20*/
        10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,
        10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,
        10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,
        11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,
        11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,
        11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,
        11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,
        11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,
        11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,
        11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,
        11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,
        11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,
        11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,
        11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,
        12,12,12,12,12,12,12,12,12,12,12,12,12,12,12,12,12,12,
        12,12,12
    };
}

```

[REDACTED]

A29823-FWC-A

[illegible]

[illegible][illegible][illegible][illegible]

-17-

This Sheet Has Been Replaced



A29823-FWC-A

1,85,67,49,125,25,109,43,89,69,113,47,55,97,3,37,83,103,27,13,33,115,41,79,17,29,119,75,73,105,
7,59,65,21,3,113,61,89,45,107,
1,85,67,49,125,25,109,43,89,69,113,47,55,97,3,37,83,103,27,13,33,115,41,79,17,29,119,75,73,105,
7,59,65,21,3,113,61,89,45,107,
1,85,67,49,125,25,109,43,89,69,113,47,55,97,3,37,83,103,27,13,33,115,41,79,17,29,119,75,73,105,
7,59,65,21,3,113,61,89,45,107, /*end 7 */

1,255,79,147,141,89,173,43,9,25,115,97,19,97,197,101,255,29,203,65,195,177,105,17,47,169,109,1
49,15,213,135,253,191,155,175,63,47,7,23,39,
1,255,79,147,141,89,173,43,9,25,115,97,19,97,197,101,255,29,203,65,195,177,105,17,47,169,109,1
49,15,213,135,253,191,155,175,63,47,7,23,39,
1,255,79,147,141,89,173,43,9,25,115,97,19,97,197,101,255,29,203,65,195,177,105,17,47,169,109,1
49,15,213,135,253,191,155,175,63,47,7,23,39,
1,255,79,147,141,89,173,43,9,25,115,97,19,97,197,101,255,29,203,65,195,177,105,17,47,169,109,1
49,15,213,135,253,191,155,175,63,47,7,23,39,
1,255,79,147,141,89,173,43,9,25,115,97,19,97,197,101,255,29,203,65,195,177,105,17,47,169,109,1
49,15,213,135,253,191,155,175,63,47,7,23,39,
1,255,79,147,141,89,173,43,9,25,115,97,19,97,197,101,255,29,203,65,195,177,105,17,47,169,109,1
49,15,213,135,253,191,155,175,63,47,7,23,39,
1,255,79,147,141,89,173,43,9,25,115,97,19,97,197,101,255,29,203,65,195,177,105,17,47,169,109,1
49,15,213,135,253,191,155,175,63,47,7,23,39,
1,255,79,147,141,89,173,43,9,25,115,97,19,97,197,101,255,29,203,65,195,177,105,17,47,169,109,1
49,15,213,135,253,191,155,175,63,47,7,23,39, /*end 8 */

1,257,465,439,177,321,181,225,235,103,411,233,59,353,329,463,385,111,475,451,263,19,249,275,36
9,393,167,333,473,469,101,21,451,229,247,297,403,497,61,361,
1,257,465,439,177,321,181,225,235,103,411,233,59,353,329,463,385,111,475,451,263,19,249,275,36
9,393,167,333,473,469,101,21,451,229,247,297,403,497,61,361,
1,257,465,439,177,321,181,225,235,103,411,233,59,353,329,463,385,111,475,451,263,19,249,275,36
9,393,167,333,473,469,101,21,451,229,247,297,403,497,61,361,
1,257,465,439,177,321,181,225,235,103,411,233,59,353,329,463,385,111,475,451,263,19,249,275,36
9,393,167,333,473,469,101,21,451,229,247,297,403,497,61,361,
1,257,465,439,177,321,181,225,235,103,411,233,59,353,329,463,385,111,475,451,263,19,249,275,36
9,393,167,333,473,469,101,21,451,229,247,297,403,497,61,361,
1,257,465,439,177,321,181,225,235,103,411,233,59,353,329,463,385,111,475,451,263,19,249,275,36
9,393,167,333,473,469,101,21,451,229,247,297,403,497,61,361,
1,257,465,439,177,321,181,225,235,103,411,233,59,353,329,463,385,111,475,451,263,19,249,275,36
9,393,167,333,473,469,101,21,451,229,247,297,403,497,61,361,
1,257,465,439,177,321,181,225,235,103,411,233,59,353,329,463,385,111,475,451,263,19,249,275,36
9,393,167,333,473,469,101,21,451,229,247,297,403,497,61,361,
/*end 9 */

1,771,721,1013,759,835,949,113,929,615,157,39,761,169,983,657,647,581,505,833,139,147,203,81,3
37,829,989,375,365,131,215,733,451,447,177,57,471,979,197,251,
1,771,721,1013,759,835,949,113,929,615,157,39,761,169,983,657,647,581,505,833,139,147,203,81,3
37,829,989,375,365,131,215,733,451,447,177,57,471,979,197,251,
1,771,721,1013,759,835,949,113,929,615,157,39,761,169,983,657,647,581,505,833,139,147,203,81,3
37,829,989,375,365,131,215,733,451,447,177,57,471,979,197,251,
1,771,721,1013,759,835,949,113,929,615,157,39,761,169,983,657,647,581,505,833,139,147,203,81,3
37,829,989,375,365,131,215,733,451,447,177,57,471,979,197,251,
1,771,721,1013,759,835,949,113,929,615,157,39,761,169,983,657,647,581,505,833,139,147,203,81,3
37,829,989,375,365,131,215,733,451,447,177,57,471,979,197,251,
1,771,721,1013,759,835,949,113,929,615,157,39,761,169,983,657,647,581,505,833,139,147,203,81,3
37,829,989,375,365,131,215,733,451,447,177,57,471,979,197,251,
1,771,721,1013,759,835,949,113,929,615,157,39,761,169,983,657,647,581,505,833,139,147,203,81,3
37,829,989,375,365,131,215,733,451,447,177,57,471,979,197,251,

This Sheet Has Been Replaced



A29823-FWC-A

```
1,771,721,1013,759,835,949,113,929,615,157,39,761,169,983,657,647,581,505,833,139,147,203,81,3
37,829,989,375,365,131,215,733,451,447,177,57,471,979,197,251,
1,771,721,1013,759,835,949,113,929,615,157,39,761,169,983,657,647,581,505,833,139,147,203,81,3
37,829,989,375,365,131,215,733,451,447,177,57,471,979,197,251,
/*end 10 */
```

```
1,1285,823,727,267,833,471,1601,1341,913,1725,2021,1905,375,893,1599,415,605,819,975,915,1715,
1223,1367,663,629,525,469,981,1667,1587,1251,451,481,721,483,1209,1457,415,1435,
1,1285,823,727,267,833,471,1601,1341,913,1725,2021,1905,375,893,1599,415,605,819,975,915,1715,
1223,1367,663,629,525,469,981,1667,1587,1251,451,481,721,483,1209,1457,415,1435,
1,1285,823,727,267,833,471,1601,1341,913,1725,2021,1905,375,893,1599,415,605,819,975,915,1715,
1223,1367,663,629,525,469,981,1667,1587,1251,451,481,721,483,1209,1457,415,1435,
1,1285,823,727,267,833,471,1601,1341,913,1725,2021,1905,375,893,1599,415,605,819,975,915,1715,
1223,1367,663,629,525,469,981,1667,1587,1251,451,481,721,483,1209,1457,415,1435,
1,1285,823,727,267,833,471,1601,1341,913,1725,2021,1905,375,893,1599,415,605,819,975,915,1715,
1223,1367,663,629,525,469,981,1667,1587,1251,451,481,721,483,1209,1457,415,1435,
1,1285,823,727,267,833,471,1601,1341,913,1725,2021,1905,375,893,1599,415,605,819,975,915,1715,
1223,1367,663,629,525,469,981,1667,1587,1251,451,481,721,483,1209,1457,415,1435,
1,1285,823,727,267,833,471,1601,1341,913,1725,2021,1905,375,893,1599,415,605,819,975,915,1715,
1223,1367,663,629,525,469,981,1667,1587,1251,451,481,721,483,1209,1457,415,1435,
1,1285,823,727,267,833,471,1601,1341,913,1725,2021,1905,375,893,1599,415,605,819,975,915,1715,
1223,1367,663,629,525,469,981,1667,1587,1251,451,481,721,483,1209,1457,415,1435,
/*end 11 */
```

```
1,3855,4091,987,1839,4033,2515,579,3863,977,3463,2909,3379,1349,3739,347,387,2881,2821,1873,19
59,1929,2389,3251,1149,243,3609,1131,1701,143,1339,3497,2499,1571,983,4021,1625,3217,1163,2977
```

```
1,3855,4091,987,1839,4033,2515,579,3863,977,3463,2909,3379,1349,3739,347,387,2881,2821,1873,19
59,1929,2389,3251,1149,243,3609,1131,1701,143,1339,3497,2499,1571,983,4021,1625,3217,1163,2977
```

```
1,3855,4091,987,1839,4033,2515,579,3863,977,3463,2909,3379,1349,3739,347,387,2881,2821,1873,19
59,1929,2389,3251,1149,243,3609,1131,1701,143,1339,3497,2499,1571,983,4021,1625,3217,1163,2977
```

```
1,3855,4091,987,1839,4033,2515,579,3863,977,3463,2909,3379,1349,3739,347,387,2881,2821,1873,19
59,1929,2389,3251,1149,243,3609,1131,1701,143,1339,3497,2499,1571,983,4021,1625,3217,1163,2977
```

```
1,3855,4091,987,1839,4033,2515,579,3863,977,3463,2909,3379,1349,3739,347,387,2881,2821,1873,19
59,1929,2389,3251,1149,243,3609,1131,1701,143,1339,3497,2499,1571,983,4021,1625,3217,1163,2977
```

```
1,3855,4091,987,1839,4033,2515,579,3863,977,3463,2909,3379,1349,3739,347,387,2881,2821,1873,19
59,1929,2389,3251,1149,243,3609,1131,1701,143,1339,3497,2499,1571,983,4021,1625,3217,1163,2977
```

```
1,3855,4091,987,1839,4033,2515,579,3863,977,3463,2909,3379,1349,3739,347,387,2881,2821,1873,19
59,1929,2389,3251,1149,243,3609,1131,1701,143,1339,3497,2499,1571,983,4021,1625,3217,1163,2977
```

```
1,3855,4091,987,1839,4033,2515,579,3863,977,3463,2909,3379,1349,3739,347,387,2881,2821,1873,19
59,1929,2389,3251,1149,243,3609,1131,1701,143,1339,3497,2499,1571,983,4021,1625,3217,1163,2977
```

```
1,3855,4091,987,1839,4033,2515,579,3863,977,3463,2909,3379,1349,3739,347,387,2881,2821,1873,19
59,1929,2389,3251,1149,243,3609,1131,1701,143,1339,3497,2499,1571,983,4021,1625,3217,1163,2977
/*end 12 */
```

```
};
if (n < 0) {
    for(j=12;j<=MAXBIT;j++) iv[1+j*MAXDIM]=1; /* Initialize all direction
        numbers for the first
        coordinate to 1 */
    for (j=1,k=0;j<=MAXBIT;j++,k+=MAXDIM) iu[j] = &iv[k];
    for (k=1;k<=MAXDIM;k++) {
        for (j=1;j<=mdeg[k];j++) iu[j][k] <= (MAXBIT-j);
    }
}
```



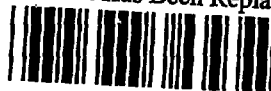
```

    for (j=mdeg[k]+1;j<=MAXBIT;j++) {
ipp=ip[k];
i=iu[j-mdeg[k]][k];
i ^= (i >> mdeg[k]);
for (l=mdeg[k]-1;l>=1;l--) {
    if (ipp & 1) i ^= iu[j-l][k];
    ipp >>= 1;
}
iu[j][k]=i;
}
}
fac=1.0/(1L << MAXBIT);
in=0;
}
else
{
    /* Check if the (n-1)-th number was generated in the previous call
    to sobol. If not, update in and ix */

    if(in!=n-1) {
unsigned long gray;

/* Set ix to 0 */
for (k=1;k<=IMIN(d,MAXDIM);k++) ix[k]=0;
in=n-1;
gray=in^(in>>1); /* Find gray code of in */
for (j=1;j<=MAXBIT;j++) {
    if(gray&1) { /* Only digits which are 1 are used */
        im=(j-1)*MAXDIM;
        for (k=1;k<=IMIN(d,MAXDIM);k++) ix[k] ^= iv[im+k];
    }
    gray>>=1;
}
    im=in; /* Calculate the next vector in the sequence */
    for (j=1;j<=MAXBIT;j++) { /* Find the rightmost zero bit */
if (!(im & 1)) break;
im >>= 1;
    }
    if (j > MAXBIT) nrerror("MAXBIT too small in sobseq");
    im=(j-1)*MAXDIM;
    for (k=1;k<=IMIN(d,MAXDIM);k++) {
ix[k] ^= iv[im+k];
x[k-1]=ix[k]*fac;
    }
    in++;
}
}
}
#undef MAXBIT
#undef MAXDIM

```



```

/*****

```

```

void halton(int n)

```

```

    This is the function halton for generating Halton points.
    It returns the n-th d-dimensional Halton point. The point is implicitly
    returned through the array x. The last two digits of n-1 in base p[j]
    are kept in q1[j] and q2[j]. When both digits become p[j]-1, the
    radical inverse function is computed again. That way the accumulation
    of round-off error is avoided. In practice, there are not any upper bounds
    on the values of d and n.

```

```

    *****/

```

```

extern int d; /* actual dimension of the points */
extern int *q1,*q2; /* q1[j] is the last digit of n-1 in base p[j], q2[j] is
                    the digit before the last one */
extern double *x; /* This will contain the Halton point */
extern int *p; /* the first d prime numbers */
extern int *p_1; /* first d prime numbers minus 1 */
extern double *incr1,*incr2; /* incr1[j] is 1/p[j] and 1/(p[j]*p[j]) */

```

```

double find_fi(int p, int n); /* See below */

```

```

void halton(int n)

```

```

{
    double a;
    int j,nn;
    static int ins_n; /* The default value of ins_n is 0 */

    /* Check if the (n-1)-th number was generated in the previous call
       to halton. If not, update q1, q2, and x */

    if(ins_n!=n-1)
    {
        ins_n=n-1;
        for (j=0; j<d; j++)
        {
            q1[j]=ins_n%p[j];
            q2[j]=(ins_n/p[j])*p[j];
            x[j]=find_fi(p[j],ins_n);
        }
        ins_n++;
    }
    for(j=0;j<d;j++)
    {
        if(q1[j]<p_1[j])
        {
            /* It is easy to update when the last digit is less than p[j]-1 */

            q1[j]++;
            x[j]=x[j]+incr1[j];
        }
        else if(q2[j]<p_1[j])
        {
            /* This is the case when the last digit is p[j]-1 and the digit
               before the last one is less than p[j]-1 */

            q1[j]=0;
            q2[j]++;
        }
    }
}

```



```

    x[j]=x[j]+incr1[j]+incr2[j]-1.0;
}
    else
{
    /* This is the case when the last digit is p[j]-1 and the digit
       before the last one is also p[j]-1 */

    q1[j]=0;
    q2[j]=0;
    nn=n/(p[j]*p[j]);
    a=nn*p[j];
    nn=nn/p[j];
    if(nn) x[j]=(a+find_fi(p[j],nn))*incr2[j]*incr1[j];
    else x[j]=a*incr2[j]*incr1[j];
}
}

/*****
find_fi(int p, int n)

This returns the radical inverse function fi(p,n) at n for the prime p.

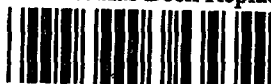
*****/
double find_fi(int p, int n)
{
    int p2,nn;
    double s,fi,incr;

    incr=1.0/p;
    p2=p*p;
    nn=n/p2;

    fi=0.0;
    s=incr;
    while (nn > 0)
    {
        fi+= (nn*p)*s;
        nn=nn/p;
        s*=incr;
    }

    /* The two largest components of fi are added later to avoid possible
       loss of precision */
    fi=((n/p*p)+ fi)/ p2;
    fi+=(n*p)/ (double) p;
    return fi;
}

```



WE CLAIM:

- 1 1. A method for one of buying, holding and
2 selling a complex security, comprising:
3 (i) deriving at least one multivariate integrand
4 which, when integrated over a domain of integration,
5 represents an estimated value of at least a portion of the
6 complex security;
7 (ii) calculating, by computer, integrand values at
8 points in the domain of integration which are obtained from
9 a low-discrepancy deterministic sequence; and
10 (iii) effecting, based on a total value which
11 includes the estimated value, one of buying, holding and
12 selling the complex security.

- 1 2. The method according to claim 1, wherein the
2 complex security comprises a derivative instrument.

- 1 3. The method according to claim 1, wherein the
2 complex security comprises a financial security.

- 1 4. The method according to claim 1, wherein the
2 complex security comprises a collateral mortgage obligation.

- 1 5. The method according to claim 1, wherein the
2 complex security comprises a bond.

- 1 6. The method according to claim 1, wherein the
2 complex security comprises a single-premium deferred
3 annuity.





1 7. The method according to claim 1, wherein the
2 complex security comprises an insurance contract.

1 8. The method according to claim 1, wherein the
2 complex security comprises an interest rate option.

1 9. The method according to claim 1, wherein the
2 complex security comprises a stock option.

1 10. The method according to claim 1, wherein the
2 complex security comprises a currency option.

1 11. The method according to claim 1, wherein the
2 complex security comprises a plurality of securities
3 selected from the group consisting of a derivative
4 instrument, a financial security, a collateral mortgage
5 obligation, a bond, a single-premium deferred annuity, an
6 insurance contract, an interest rate option, a stock option
7 and a currency option.

1 12. The method according to claim 1, wherein
2 combining the integrand values comprises averaging.

1 13. The method according to claim 1, wherein the
2 number of integrand values is predetermined.

1 14. The method according to claim 1, wherein the
2 number of integrand values is determined automatically.





1 15. The method according to claim 1, further
2 comprising allocating integrand evaluations among a
3 plurality of processors.

1 16. The method according to claim 1, wherein the low-
2 discrepancy deterministic sequence comprises Sobol points.

1 17. The method according to claim 1, wherein the low-
2 discrepancy deterministic sequence comprises Halton points.

1 18. The method according to claim 1, wherein the low-
2 discrepancy deterministic sequence comprises Hammersley
3 points.

1 19. The method according to claim 1, wherein the low-
2 discrepancy sequence comprises hyperbolic-cross points.

1 20. A method for selling a complex security,
2 comprising:

3 (i) deriving at least one multivariate integrand
4 which, when integrated over a domain of integration,
5 represents an estimated value of at least a portion of the
6 complex security;

7 (ii) calculating, by computer, integrand values at
8 points in the domain of integration which are obtained from
9 a low-discrepancy deterministic sequence; and

10 (iii) selling the complex security at an offering
11 price which is not less than the estimated value.

