

準モンテカルロ法における誤差推定

諸星 穂積 (Hozumi Morohosi) 伏見 正則 (Masanori Fushimi)

東京大学大学院工学系研究科

1 はじめに

s 次元の単位立方体 $[0, 1]^s$ 上で定義された関数 $f(\mathbf{x})$ の積分値

$$I = \int_{[0,1]^s} f(\mathbf{x}) d\mathbf{x} \quad (1)$$

を数値的に求めること、およびその計算値にどの程度の誤差が含まれるかを評価することは、実用上重要な問題である。 $s = 1$ の場合については、多くの研究によって、実用的な積分法およびその誤差評価法がほぼ確立している。一方、 $s > 1$ の場合には問題は格段に難しくなる。比較的低次元の場合は、1次元で有効な積分法をそのまま拡張した方法や、優良格子点法などが、ある程度実用的な手法として利用できるが、より高い次元での利用は難しい。また、数千、数万というような高次元の問題については、(疑似)乱数を用いたモンテカルロ法を適用することになり、誤差評価については、中心極限定理に基づいて行うことができる。ただし、積分計算値の収束は極めて遅い。

準モンテカルロ法は多次元数値積分の手法の中で、ある程度高次元まで利用でき、かつ収束が速いという特徴をもっており、(1)の積分を後述する“差異の小さい点列” $\{\mathbf{x}_i\} \in [0, 1]^s$ による均等重みの和

$$I_N = \frac{1}{N} \sum_{i=1}^N f(\mathbf{x}_i) \quad (2)$$

で近似する方法である。本論では、その誤差評価法を検討する。この方法については、理論的な誤差の上界の評価式が存在するだけで、実用的に利用できるような誤差評価法がまだ存在しない。近年、モンテカルロ法を参考にした確率的誤差評価法が提案されている。ここでは、これらの手法について、数値実験を行った結果を述べる。

2 差異の小さい点列

$f(\mathbf{x})$ を s 次元の単位立方体 $[0, 1]^s$ 上で定義された有界変動関数とする。点列 $P = \{\mathbf{x}_i\}_{i=1}^N \in [0, 1]^s$ の差異 D_N は次式で与えられる。

$$D_N = \sup_J \left| \frac{A(J; P)}{N} - V(J) \right|. \quad (3)$$

ここで $\sup$ は $J = [0, t_1) \times \cdots \times [0, t_s) \subset [0, 1]^s$ なる全ての半開区間にわたり、 $A(J; P)$ は J に含まれる P の点の個数を表す。差異の小さい点列が多次元の数値積分に有効であるとされる

根拠は、次の Koksma-Hlawka の不等式である。

$$\left| \frac{1}{N} \sum_{i=1}^N f(\mathbf{x}_i) - \int_{[0,1]^s} f(\mathbf{x}) d\mathbf{x} \right| \leq V(f) D_N. \quad (4)$$

ここで、 $V(f)$ は関数 f の (Hardy-Krause の意味での) 全変動とよばれる量である。差異の小さい点列として、以下に述べる (t, s) -sequence を用いると、 D_N が漸近的に $O((\log N)^s/N)$ となり (s は点列の次元)、高次元の数値積分を高速に行なうことが可能であると考えられている。

ここで、 (t, s) -sequence の定義を述べておく [3]。そのために、いくつかの予備的定義を行なう。

定義 1 (基本区間) 整数 $s \geq 1$ と $b \geq 2$ が与えられたとき、 b を基数とする基本区間を

$$E = \prod_{j=1}^s \left[\frac{a_j}{b^{d_j}}, \frac{a_j+1}{b^{d_j}} \right) \quad (5)$$

で定義する。ここで、 d_j, a_j は整数で、 $d_j \geq 0, 0 \leq a_j < b^{d_j}$ を満たすものとする。

定義 2 ((t, m, s)-net) s, m, t, b は整数で $s \geq 1, m \geq 0, 0 \leq t \leq m, b \geq 2$ とする。 s 次元の単位立方体 I^s 内の点集合 $\{\mathbf{z}_i : i = 1, \dots, b^m\}$ が、任意の体積 b^{t-m} の基本区間にちょうど b^t 個含まれるとき、この点集合は基数 b の (t, m, s) -net であるという。

以上の定義の下で、 (t, s) -sequence を次のように定義する。

定義 3 ((t, s)-sequence) $t \geq 0$ を整数とする。点列 $\{\mathbf{z}_n\}$ において、任意の整数 $k \geq 0, m > t$ に対して $\{\mathbf{z}_n : n = kb^m, \dots, (k+1)b^m - 1\}$ が (t, m, s) -net になるとき、基数 b の (t, s) -sequence であるという。

(t, s) -sequence としては、Sobol' 列や Faure 列が知られている。Sobol' 列は、上記の定義に従えば、 $GF(2)$ 上の原始多項式 $p_i(x)$ を次数が低いほうから s 個とってきたとき、それらの原始多項式の次数の総和 $\sum_{i=1}^s \deg p_i(x)$ を求め、 $t = \sum_{i=1}^s \deg p_i(x) - s$ とした基数 2 の (t, s) -sequence であり、Faure 列は次元 s 以上の最小の素数 p を基数とする $(0, s)$ -sequence である。実際の計算では有限個の点しか使わないわけで、 (t, s) -sequence を利用する利点は、任意の m に対する (t, m, s) -net を容易に作成できることにある。

実際にこの点列を使って数値積分を行ないその誤差評価をしようとする、関数の全変動を計算することは事実上不可能であるので、Koksma-Hlawka の不等式を利用することはできない。次節以降で本論で扱う誤差評価手法を述べる。

3 確率的変動

以下では (t, m, s) -net を対象とし、確率的な誤差評価法を与える。これは、もともとの (t, m, s) -net に確率的に独立な変動を与えて (複数の) 新たな点集合をつくり、それらで (2) を評価しそのばらつきをもとに誤差を見積もる方法である。確率的な変動を与える方法として以下の 2 つを考えた。

点列の scramble [4,5]

点列 $\{\mathbf{z}_i\}$ を (t, m, s) -net とする. $\mathbf{z}_i = (z_i^1, \dots, z_i^s)$ と座標成分で表示したとき, 各成分を $z_i^j = \sum_{k=1}^{\infty} z_{ijk} b^{-k}$, $0 \leq z_{ijk} < b$ と書く. この $\{\mathbf{z}_i\}$ から $\{\mathbf{x}_i\}$, $\mathbf{x}_i = (x_i^1, \dots, x_i^s)$, $x_i^j = \sum_{k=1}^{\infty} x_{ijk} b^{-k}$ を次のように決める.

$$\begin{aligned} x_{ij1} &= \pi_j(z_{ij1}), \\ x_{ij2} &= \pi_{jz_{ij1}}(z_{ij2}), \\ &\vdots \\ x_{ijk} &= \pi_{jz_{ij1}z_{ij2}\dots z_{ij,k-1}}(z_{ijk}). \end{aligned}$$

各 π は $0, 1, \dots, b-1$ の置換で, 全置換 $b!$ 個の上で一様に分布しているとする. π_j は全ての i について各 z_i^j の最初の桁を置換する. $\pi_{jz_{ij1}}$ は同様に第 2 桁を置換するが, 第 1 桁の値毎に互いに独立な置換とする. 以下同様に, 第 k 桁の置換は, $k-1$ 桁までの値に依存して決まる.

この方法は次のような操作を行なっているものと解釈できる. 積分領域を各座標軸ごとに b 分割して得られた b 個の小領域をランダムに並べ換え, 次に各小領域の中で更に b 分割を行なってランダムに並べ換えを行なう. 各小領域での並べ換えが互いに独立になるようにすることが, $\pi_{jz_{ij1}}$ という置換を用いたことに対応する. 以下分割で得られた小領域のなかで, この操作を繰り返す.

点列の random shift

$\mathbf{u}$ を $[0, 1]^s$ 上一様分布するベクトルとして, $\mathbf{x}_i = \mathbf{z}_i + \mathbf{u} \pmod{1}$ とする. つまり, もとの点列 $\{\mathbf{z}_i\}$ を一斉にある方向に平行移動させて, 領域 $[0, 1]^s$ からはみ出した点については, 周期性条件によって $[0, 1]^s$ 内に引きもどす.

以上の確率変動を M 回独立に繰返し, それによって得られた積分の計算値を $\hat{I}^{(1)}, \dots, \hat{I}^{(M)}$ とし, これらから積分 I の推定値

$$\bar{I} = \frac{1}{M} \sum_{j=1}^M \hat{I}^{(j)}, \quad (6)$$

を求める. また, 各計算値から誤差の推定値として分散

$$\hat{\sigma}^2 = \frac{1}{M(M-1)} \sum_{j=1}^M (\hat{I}^{(j)} - \bar{I})^2, \quad (7)$$

を計算する.

4 数値実験

[1] で提案されている多次元数値積分の試験関数を用い, 前節で述べた手法について, 推定量の精度と, 誤差評価の有効性を検証した. 使用した関数は以下の通りであり, 積分区間は $[0, 1]^s$ である. 各関数は s 次元で定義されており, $\mathbf{x} = (x_1, \dots, x_s)$ である. 各関数の後ろにはその関数の特徴が記されている.

$$f_1(\mathbf{x}) = \cos(2\pi u_1 + \sum_{j=1}^s a_j x_j) \quad (\text{Oscillatory})$$

$$f_2(\mathbf{x}) = \prod_{j=1}^s (a_j^{-2} + (x_j - u_j)^2) \quad (\text{Product Peak})$$

$$f_3(\mathbf{x}) = (1 + \sum_{j=1}^s a_j x_j)^{-s-1} \quad (\text{Corner Peak})$$

$$f_4(\mathbf{x}) = \exp(-\sum_{j=1}^s a_j^2 (x_j - u_j)^2) \quad (\text{Gaussian})$$

$$f_5(\mathbf{x}) = \exp(-\sum_{j=1}^s a_j |x_j - u_j|) \quad (C_0)$$

$$f_6(\mathbf{x}) = \exp(-\sum_{j=1}^s a_j x_j) 1_{x_1 > u_1} 1_{x_2 > u_2} \quad (\text{Discontinuous})$$

f_6 で $1_{x_1 > u_1}$ は $x_1 > u_1$ なる x_1 の領域で 1, 他で 0 となる関数である ($1_{x_2 > u_2}$ も同様). u_i は $[0, 1)$ 上の一様乱数であり, a_j は $[0, 1)$ 上の一様乱数をそれぞれの f_k に対して, $\sum_{j=1}^s a_j = h_k s^{-e_k}$ となるように規格化した. 規格因子である e_k, h_k については, $(e_k) = (1.5, 2, 2, 1, 2, 2)$, $(h_k) = (110, 600, 600, 100, 150, 100)$ とした ([1] で推奨している値をそのまま使った). a_j の値が大きいほど数値積分は難しくなる. u_j は関数の対称性による影響を除くためのものである. 10 次元の f_1 に対して, a_j, u_j を 10 組与え, それぞれの組で決まる f_1 について 30 回の scramble や random shift を行なって式 (6) によって推定値を求めた. 差異の小さい点列としては, Faure 列, Sobol' 列, および Larcher et al.[2] による (t, m, s) -net を用いた.

scramble と random shift の実装は以下のように行なった. scramble では, 各座標毎, 各桁毎に置換を割り当ててののだが, 第 2 桁目以降は, その上位桁のとり値毎に互いに独立な置換を割り当てなければならない. 利用した (t, m, s) -net は b 進で小数点以下 m 桁の数なので, m 桁目まで置換を行なうとして, 1 回の scramble に必要な置換の総数は, $s \times (1 + b + b^2 + \dots + b^{m-1}) = s(b^m - 1)/(b - 1)$ である. 各回毎に, これらの置換を最初に発生させて配列上に記憶して, 各サンプル点毎に小数点以下の各桁に対応する置換を配列上で探索して scramble した点を求めた. random shift は各回の実行毎に $[0, 1]^s$ のランダムベクトルを 1 個発生させて計算を行なった.

図 1 では各試験関数 $f_1, \dots, f_6$ について, 積分のサンプル点数を横軸にとり, scramble や random shift して得られた積分推定値の相対誤差を, パラメータ a_j, u_j の 10 組について平均値を取りプロットした. Faure 列と Sobol' 列の間には顕著な相違は見られない. Larcher et al. による (t, m, s) -net は, この実験に関しては, サンプル点数の増加に伴う精度の向上が見られない場合があった.

また, 誤差推定の有効性の検証については, 各 f_k についてパラメータの値を 1 つ固定して, Faure 列と Sobol' 列についてのみ行なった. 図 2, 3 では横軸にサンプル点数を取り, scramble と random shift 双方について式 (7) で得られた $\hat{\sigma}$ を用い, $3\hat{\sigma}$ を誤差の推定値として $\bar{I} \pm 3\hat{\sigma}$ を表示した. 点線は積分の真値である. 真値が $\bar{I} \pm 3\hat{\sigma}$ の範囲に入っていることが観察される. この実験に関しては, Faure 列のほうが Sobol' 列よりも精密な誤差推定を与えた. また, scramble と random shift を比較すると scramble のほうが誤差推定が精密であった.

5 考察・課題

準モンテカルロ法に対する 2 つの確率的誤差評価の数値実験結果を報告した. 2 つの手法を比較すると, どちらの方法でも実験的には誤差の推定には成功しており, 今回の実験結果のみからは, 誤差推定精度 (標本分散の小ささ) は scramble のほうが良好であることが観察された. しかし, scramble は実行に際してサンプル点数に比例して増大する多大な記憶領域と, その領域上での探索のための計算時間が必要であるのに対し, random shift は余計な記憶領域を必要

としないし、必要な計算時間も少い。この点を考慮すると、推定法として random shift は有効であると考えられる。

手法の理論的根拠については、random shift については、個々の推定値が互いに独立になると考えられるので、標本分散による推定法は妥当であろうと考えられるが、scramble については、推定値の独立性や、中心極限定理の成立について、少なくとも [5] では明らかにされていない。[4] では、scramble による積分の 2 乗誤差の期待値を、Haar 関数を拡張した b -adic Haar 関数を用いて解析している。簡単のため、1 次元 b -adic Haar 関数を示すと、それは次のようなものである。まず、 $[0, 1]$ 上の b 個の関数 $\psi_c(x)$, $c = 0, 1, \dots, b-1$ を次のように定義する。

$$\psi_c(x) = \begin{cases} \sqrt{b} - \frac{1}{\sqrt{b}}, & \frac{c}{b} \leq x < \frac{c+1}{b}, \\ -\frac{1}{\sqrt{b}}, & \text{otherwise.} \end{cases} \quad (8)$$

$\psi_c(x)$ をもとに、 b -adic Haar 関数を次のように定義する： $k \geq 0$, $0 \leq t < b^k$ を整数とする。 $\psi_c(x)$ を用いて、

$$\psi_{ktc}(x) = b^{\frac{k}{2}} \psi_c(b^k x - t) \quad (9)$$

とする。被積分関数をこの関数系 $\{\psi_{ktc}\}$ で展開し、基数 b の (t, m, s) -net をサンプル点として数値積分を行なうと、展開項中の k の値が $m - t$ 以下である ψ_{ktc} は 0 になる。この事実を利用して、数値積分の 2 乗誤差の期待値を計算しているのが [4] である。さらに、この b -adic Haar 関数の展開係数がある速度で減衰するような関数のクラスに限定して、誤差解析を行なった研究がいくつか報告されている。

今回の数値実験で観察されたいくつかの事実の説明や、あるいは、2つの推定法 (random shift と scramble) の間になんらかの関係があるのか、などの問題について、このような b -adic Haar 関数を利用する誤差解析の手法を用いることは今後の課題である。

参考文献

- [1] Genz, A.: Testing Multidimensional Integration Routines. In *Tools, Methods and Languages for Scientific and Engineering Computation*, B. Ford, J. C. Rault, F. Thomasset(editors), pp. 81-94, North-Holland, 1984.
- [2] Larcher G., A. Lauss, H. Niederreiter, and W. Ch. Schmid: Optimal Polynomial for (t, m, s) -net and Numerical Integration of Multivariate Walsh Series. *SIAM Journal on Numerical Analysis*, Vol. 33, No. 6, pp. 2239-2253, 1996.
- [3] Niederreiter, H.: *Random Number Generation and Quasi-Monte Carlo Methods*. CBMS-NSF Regional Conference Series in Applied Mathematics No.63, SIAM, 1992.
- [4] Owen, A. B.: Monte Carlo Variance of Scrambled Equidistribution Quadrature. Technical Report No. 466, Department of Statistics, Stanford University, November 1994.
- [5] Owen, A. B.: Randomly Permuted (t, m, s) -Nets and (t, s) -Sequences. Technical Report No. 464, Department of Statistics, Stanford University, September 1994.

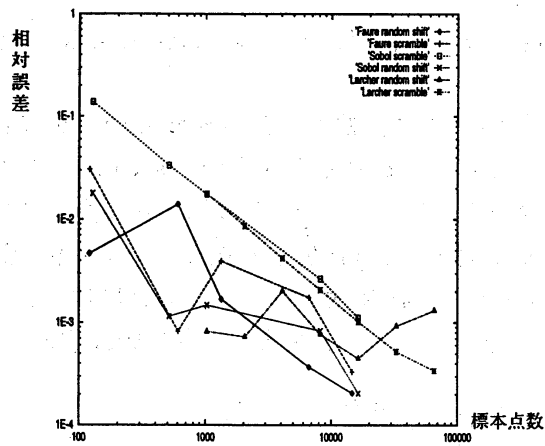
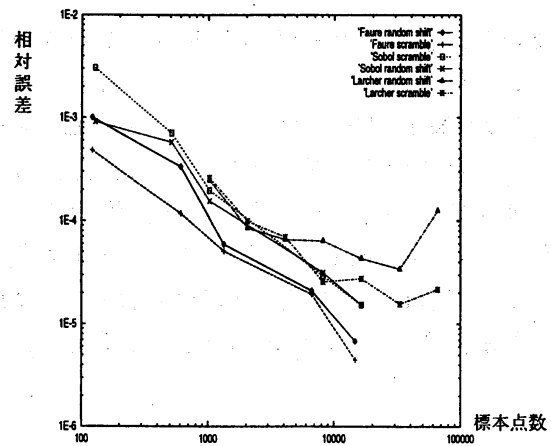
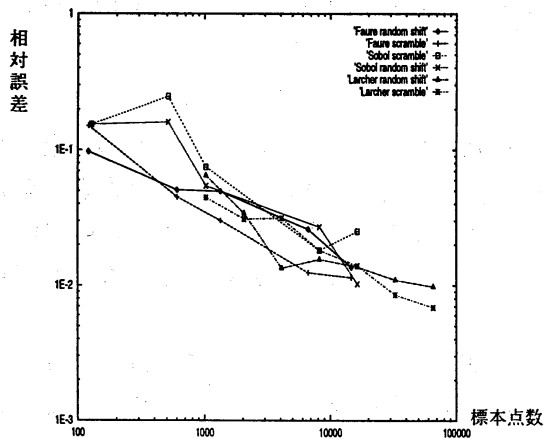
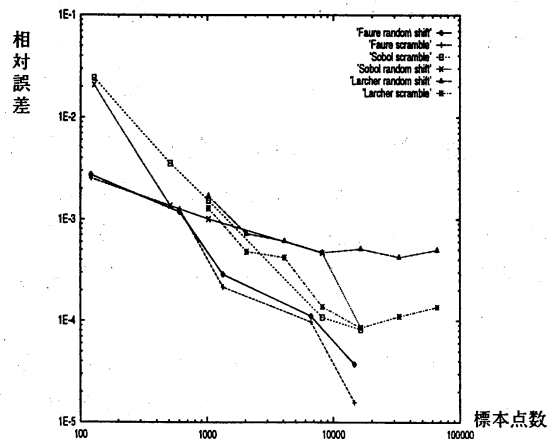
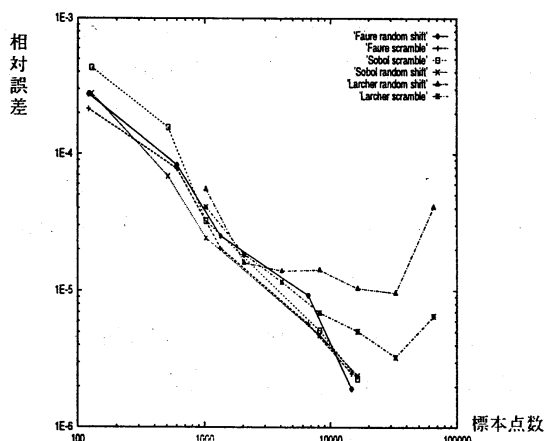
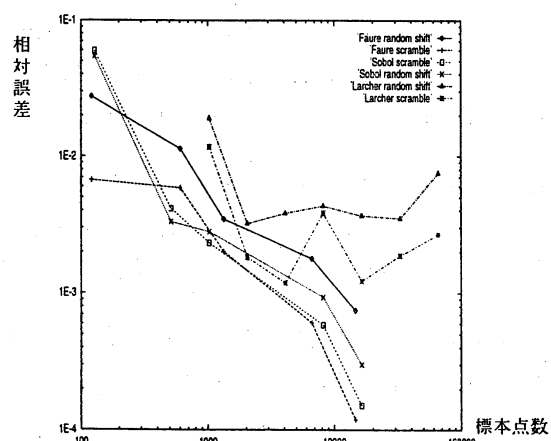
(a) f_1 (b) f_2 (c) f_3 (d) f_4 (e) f_5 (f) f_6

図 1: Faure 列, Sobol' 列, Larcher net の scramble と shift による相対誤差

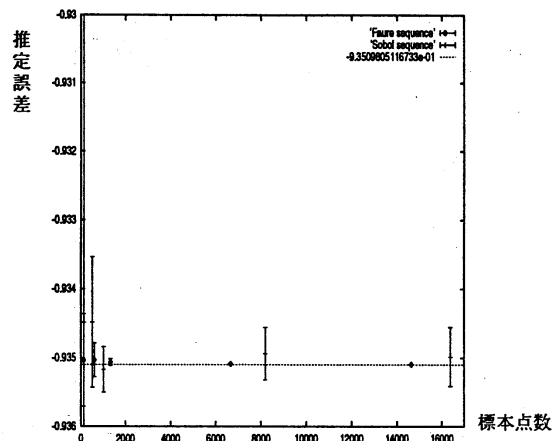
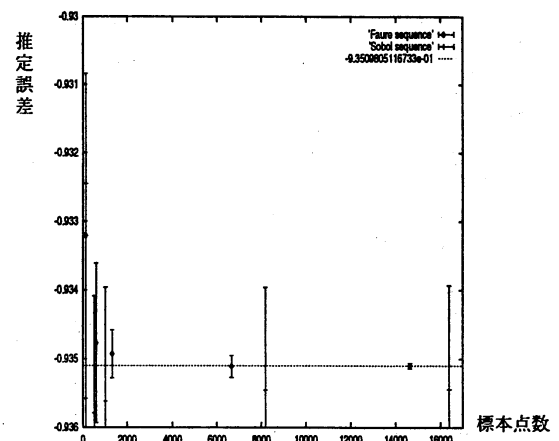
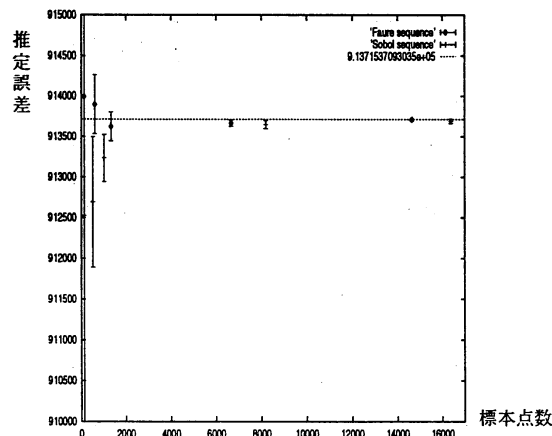
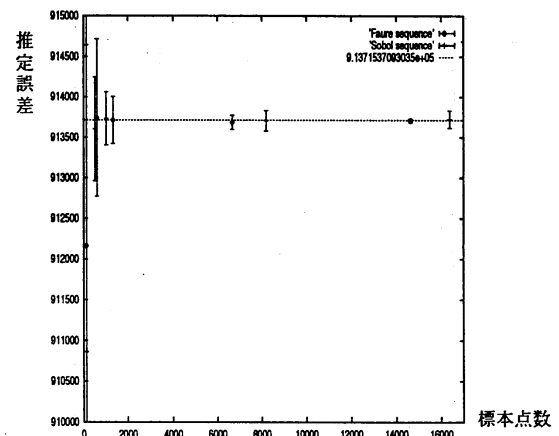
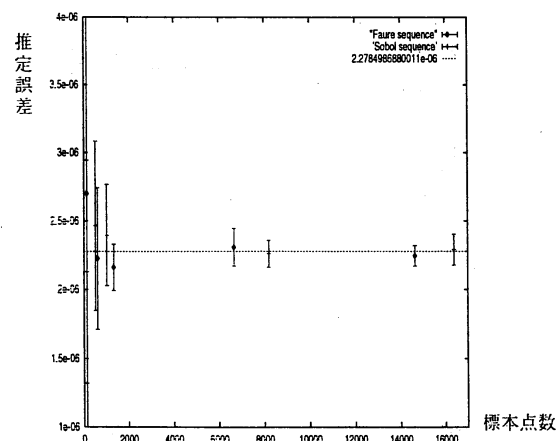
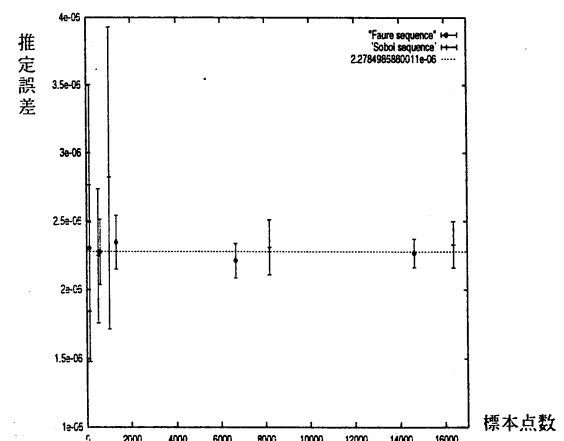
(a) f_1 scramble(b) f_1 random shift(c) f_2 scramble(d) f_2 random shift(e) f_3 scramble(f) f_3 random shift

図 2: Faure 列, Sobol' 列の scramble と shift による誤差推定 (1)

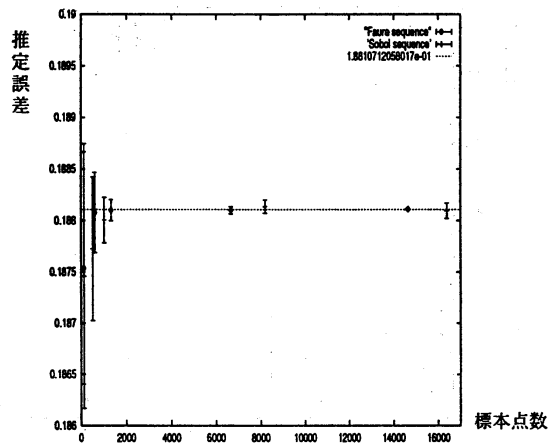
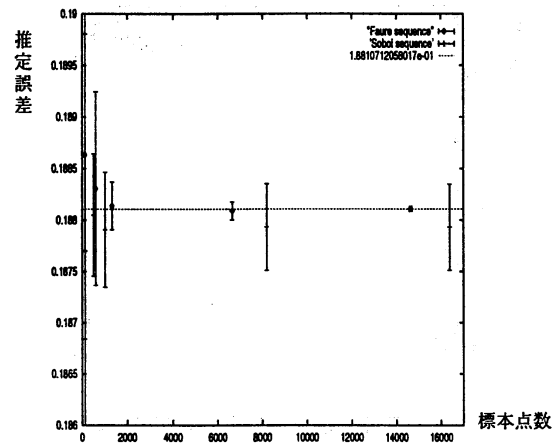
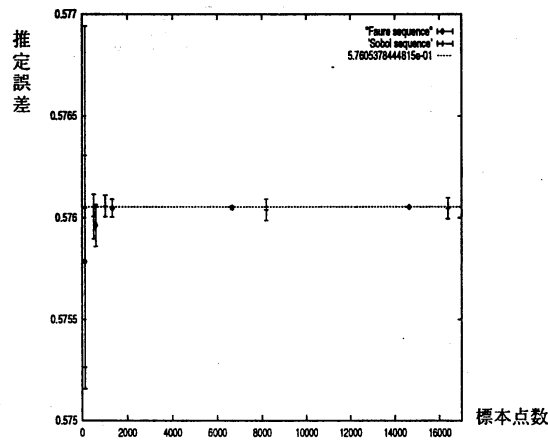
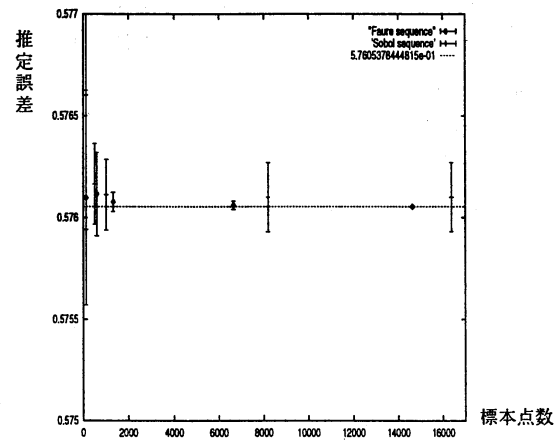
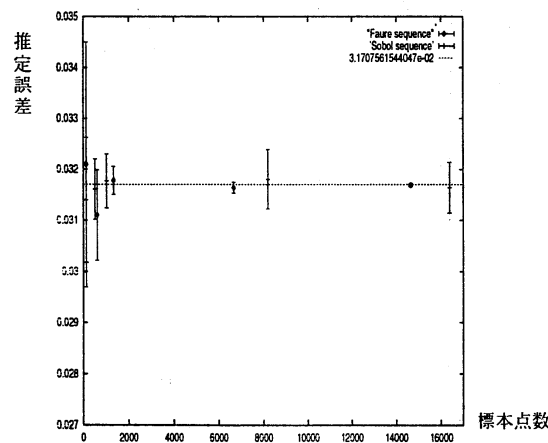
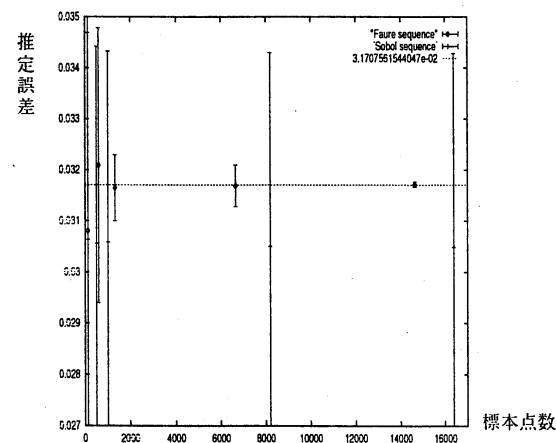
(a) f_4 scramble(b) f_4 random shift(c) f_5 scramble(d) f_5 random shift(e) f_6 scramble(f) f_6 random shift

図 3: Faure 列, Sobol' 列の scramble と shift による誤差推定 (2)