超一様分布の数理

M2006MM014 黒田　智章
指導教員　伏見　正則

1　はじめに

航空工学や電気工学など科学技術計算が利用される分野において数値積分の占める割合は大きい。その計算をする際、次元が高くなると中点則や台形則などの方法では精度の高い解を求めるのに多くの時間が必要となる。そこでモンテカルロ法により定積分値を求めることが盛んに行われてきた。しかしこの方法でも解の精度を一桁上げるためには 100 倍の計算時間が必要となり、解の収束に時間がかかる。そこで定積分の解を高速に求めるために、超一様分布列と呼ばれる特殊な点列が考えられた。

超一様分布列の発生方法は複数あるが、そのどれを用いても次元がそれほど大きくない場合では、モンテカルロ法に比べ早い収束を実現できる。しかし、高次元の数値積分においては従来の発生方法では、モンテカルロ法よりも非効率であるという予測が立てられた。そこで代表的な超一様分布列である Halton 列、Faure 列に改良を加え、次元が高くなった場合でもモンテカルロ法を上回る収束速度を実現する点列を発生させる方法を提案する。

本論文の構成は以下のとおりである。第 2 節では Halton 列の説明し、実際に数値積分をすることによりモンテカルロ法の場合と比較する。その後 Halton 列の問題点を補うための方法を提案し、その実験結果を記載する。3 節では Faure 列の説明する。その後 Faure 列の改善方法の説明、および実験結果を記す。

2　Halton 列

Halton 列の発生方法を説明するため、はじめに van der Corput 列について述べる。 $b \geq 2$ を整数とし、整数 $n \geq 0$ に対して、その b 進展開を

$$n = d_0 + d_1 b + d_2 b^2 + \cdots + d_m b^m \quad 0 \leq d_i < b$$

とすると、基底 b の基底逆関数は

$$\phi_b(n) = \frac{d_0}{b} + \frac{d_1}{b^2} + \cdots + \frac{d_m}{b^{m+1}}$$

と表される。
このとき $\phi_b(0), \phi_b(1), \phi_b(2), \phi_b(3), \cdots$ を順に並べた点列を van der Corput 列と呼ぶ。
この点列を高次元に拡張した

$$X_n = (\phi_{b_1}(n), \cdots, \phi_{b_s}(n)) \qquad n = 0, 1, 2, \cdots$$

を Halton 列と呼ぶ。ただし $b_1, \cdots, b_s$ はすべて素数とする。

では次に実際にこれらの点を用いて $\int_0^1 e^x \, dx$ を計算し、誤差を調べる。計算方法は以下の通りである。
$x_0, x_1, x_2, \cdots$ を van der Corput 列とし、サンプルの数を N とすると誤差は

$$\left| \int_0^1 e^x \, dx - \frac{1}{N} \sum_{n=0}^{N-1} e^{x_n} \right|$$

とあらわされる。
実験では基底 2 の van der Corput 列と乱数を使用し、その相対誤差を調べた。計算に使用するサンプルはそれぞれ 1000 個とした。それらの結果を以下に示す。

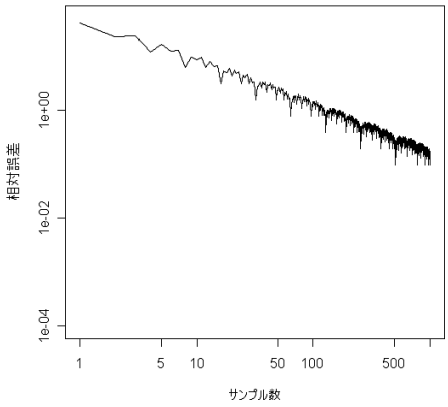

図 1 :van der Corput 列 ($b = 2$) の相対誤差

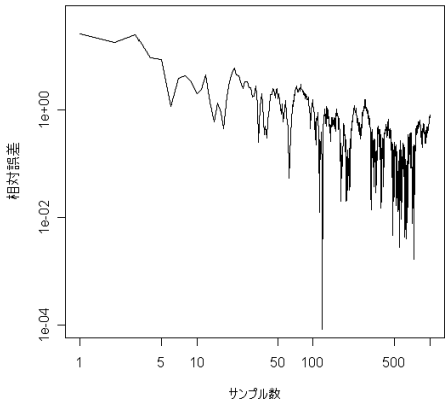

図 2 :乱数の相対誤差

1000 点のサンプルから求まった相対誤差は、van der Corput 列では 0.12、乱数では 0.81 となった。 これらの結果より、乱数でも精度の良い解が見つかる部分があるが、収束が遅いことがわかる。それに対し、超一様分布列の場合では極端に解が良くなることは少ないが、安定して真の値に近づいていることがわかる。今回の実験では積分可能な関数を用いたが、現実の問題では真の値がわからない場合に使用される。そのため求まる解の幅が広いということはそれだけ多くのリスクを負うことになる。両者の有効性の違いは明らかである。

2.1 Halton 列の問題点

Halton 列を高次元の問題に適応させるのは大きなリスクを負うことになる。前述したように 1 次元 Halton 列（van der Corput 列）は基底 b によって一意的に決まる。基底に 2 をとった場合では点列が非常に一様なものになるが、次元が大きくなるとその生成法ゆえ一様性は失われていく。つまり基底はできるだけ小さい素数が望ましいのである。しかし次元が増えれば増えるほど多くの基底が必要となり、小さい順に素数を選んでも大きな基底を使わざるを得ない。実際に次元があまりに大きくなると、モンテカルロ法よりも精度が落ちるという予測も立てられている。精度の良い解を高速で求めるために作られた点列であるのに、これでは有効であるとは言いがたい。

2.2 改良 Halton 列 1

ここでは前節までの Halton 列に工夫をすることにより次元が高くなった場合でのリスクを減らす方法を提案する。一様性が失われる一番の要因は、基底が大きくなるとサンプルの多くが、ひとつ前、ひとつ後の点と近いものになる。その点を改善するため基底逆関数の工夫をする。前節までの $\phi_b(n)$ のかわりに

$$\phi_b(n;\pi) = \frac{\pi_b(a_0)}{b} + \frac{\pi_b(a_1)}{b^2} + \cdots + \frac{\pi_b(a_m)}{b^{m+1}}$$

を使用する。
ここで出てくる、$a_j, j = 0, 1, 2, \cdots, m$ は n の b 進展開 $n = a_0 + a_1 b + \cdots + a_m b^m$ における係数とする。π_b は $\{0, 1, 2, \cdots, b-1\}$ 上の置換であり、そのときの点列は

$$X_n = (\phi_{b_1}(n;\pi_{b_1}), \phi_{b_2}(n;\pi_{b_2}), \cdots, \phi_{b_k}(n;\pi_{b_s}))$$

となる。これをスクランブル Halton 列と呼ぶ。これまでに紹介した Halton 列はすべての a, b に対して $\pi_b(a) = a$(恒等置換) を当てはめたものである。

では次に置換 π_b をどう選ぶかという問題が生じる。ここでは基底が変わるたびにランダムに置換する方法を提案する。そもそも超一様分布とはモンテカルロ法の「乱数」を使うことのリスクを考え、代わりに決定論的な数列を用いて解の変動幅を狭めようとしたものであった。その考えからするならば、乱数を使うということは逆戻りなのかもしれない。しかしこのアルゴリズムを利用し実験した結果、基底が大きくなった場合のリスクをある程度抑えることができるようになった。

2.3 改良 Halton 列 2

ランダムに入れ替えるという方法では、基底が大きくなれば入れ替えの組み合わせは増え、ランダムな要素が持つ影響力は増える。つまり次元が非常に高くなってしまった場合、モンテカルロ法とほぼ等しい物になってしまい、元の Halton 列の利点がすべて消えてしまう。そこで次に規則的に π_b を決める方法を提案する。まずはじめに π_b のテーブルを 0 以外すべて反転させる (reverse permutation という)。
$\pi = (0, 1, 2, 3, \cdots, b-1) \ \rightarrow \ (0, b-1, b-2, b-3, \cdots, 1)$
となる。

一次元目の点列 (基底 2 の場合) はこれを π とする。次にそれぞれの基底が担当する次元に応じて π_b を右方向にシフトしていく。ただしその際 π_0 は 0 で固定する。また π_{b-1} の次のシフトは π_0 とする。シフトの仕方は s 次元の数値積分をする際の t 次元目の関数の基底が b とするとき、$t/s \cdot b$(小数点以下切捨て) とする。たとえば 16 次元の計算をする際の基底 11 を考えてみると、はじめに 0 以外のすべてのテーブルを反転し $\pi = (0, 10, 9, 8, \cdots, 1)$ とする。次に基底 11 は 5 番目の次元を担当するので、$5/16 \cdot 11 = 3.4\cdots$ となるので、0 を固定して右方向に 3 つずらす、すると $\pi = (0, 3, 2, 1, 10, \cdots, 4)$ となる。この操作をすべての次元で行って点列を作る。

はじめの改良の問題点は、次元にかかわらずすべて同様にランダムに入れ替える、という方法をとったことにある。2 節のはじめに述べたように、問題なのは基底が大きくなった場合の van der Corput である。つまり基底が小さい場合でまで π を大きく動かす必要はなく、むしろそうすることで Halton 列の良い面を潰してしまう。この方法では 16 次元の場合基底 2,3,5 は π に変化が無く、最後の方の次元は reverse permutation とほぼ一致してしまうが、基底が増えるに従い π のずれが大きくなっていく。また一つひとつの次元をずらしすぎることにより失われる Halton 列の利点をある程度保つことができる。この点列を用いて実験を行った結果、Halton 列、改良 Halton 列 1、乱数の場合に比べ早い収束が実現できた。

2.4 実験結果

以下は 16 次元の関数の積分 $\theta_f = \int_{[0,1]^s} \exp(\mathbf{u} \cdot \mathbf{z}) d\mathbf{z}$ を Halton 列、改良 Halton 列 1、Halton 列 2 を用いて計算した結果である。ただし u は [0,1) の一様乱数であり、次元ごとに別々の値を取る。1000 点のサンプルより求まった相対誤差は、Halton 列が 3.29696、改良 Halton 列が 1.37445、改良 Halton 列 2 が 0.477928 となった。以下の図より Halton 列は確かに解の幅は小さいが収束していく様子が見えてこない。改良 1 では Halton 列よりも誤差は小さいが、解の幅が広いのと改悪に向かう部分が見られる、これはランダムな要素を取り入れることのリスクであり次元が増えればよりこのリスクは高まる。改良 2 では誤差が安定して右下がりになっており、最終的な解も他の二つの方法に比べ優れていることが確

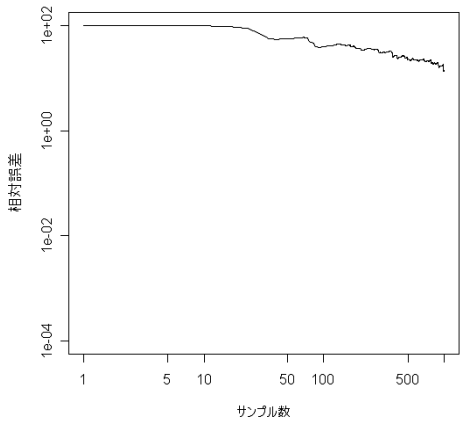

図 3 :Halton 列の相対誤差の推移

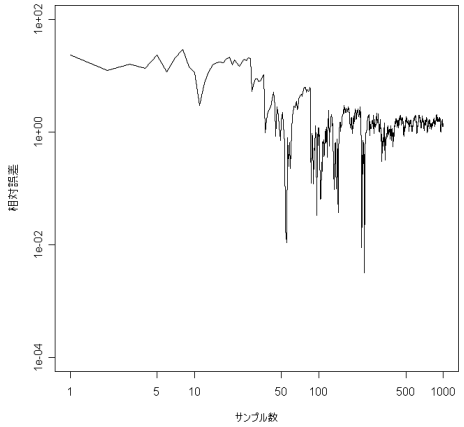

図 4 :改良 Halton 列 1 の相対誤差の推移

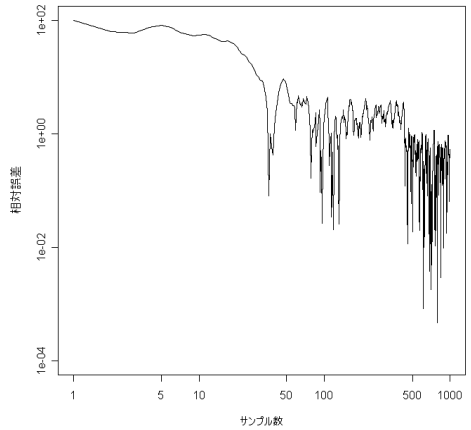

図 5 :改良 Halton 列 2 の相対誤差の推移

認できる。一様乱数を変えて複数回の実験をおこなった場合でもほぼすべての場合で、改良 2 が最も優れていた。以上より改良 2 は有効な方法であることわかる。

3 Faure 列

ここでは前節までに使用した Halton 列とは別の生成法をとり、次元が高くなる際の問題の改善に取り組む。Faure 列も Halton 列同様、基本となるアイデアは van der Corput 列を元にするという点では同じである。Halton 列では次元ごとに基底を変えることで高次元での点列を作成した。それに対し Faure 列では基底は同一のまま、Pascal 行列を利用することで van der Corput 列を高次元に拡張する。まずは 2 章で説明した van der Corput 列同様、各 n に対して b 進展開の係数 $a'_j = 0, 1, 2, \cdots, m$ を求める。それから次のものを計算する。

$$\begin{pmatrix} a_0^{(1)} \\ a_1^{(1)} \\ \vdots \\ a_m^{(1)} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & \cdots \\ 0 & 1 & 0 & \cdots \\ 0 & 0 & 1 & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_m \end{pmatrix}$$

$$\begin{pmatrix} a_0^{(i)} \\ a_1^{(i)} \\ \vdots \\ a_m^{(i)} \end{pmatrix} = \begin{pmatrix} {}_0\mathrm{C}_0 & {}_1\mathrm{C}_0 & {}_2\mathrm{C}_0 & \cdots \\ 0 & {}_1\mathrm{C}_1 & {}_2\mathrm{C}_1 & \cdots \\ 0 & 0 & {}_2\mathrm{C}_2 & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{pmatrix}^{i-1} \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_m \end{pmatrix}$$

$$(\mathrm{mod}\ b), i \geq 2$$

こうして計算される $(a_0^{(i)}, a_1^{(i)}, \cdots)$ より次元 s の点列は

$$\mathbf{X}_n = (\phi_b^{(1)}(n), \cdots, \phi_b^{(s)}(n))$$

と表される。ただし $\phi_b^{(i)}(n)$ は

$$\phi_b^{(i)}(n) = \frac{a_0^{(i)}}{b} + \frac{a_1^{(i)}}{b^2} + \cdots + \frac{a_m^{(i)}}{b^{m+1}}$$

とする。ただし基底 b は次元 s 以上の最小の素数とする。Faure 列も van der Corput 列を利用する以上, 前章で説明した基底が大きくなると一様性が失われるという欠点を持っている。しかし Halton で 16 次元を実現しようとした際には、16 番目の素数である 47 を基底に持つ van der Corput 列を利用しなくてはならない。それに対し Faure 列ではすべての次元で、次元以上の最小の素数を基底に利用するため、16 よりも大きい最小の素数 17 で 16 次元目の基底を済ますことができる。つまり次元が高くなることのリスクをなくすことはできないが軽減することができるのである。

3.1 Faure 列の問題点

次元が小さい場合では Faure 列は Halton 列よりも劣るが、次元が高い場合では Faure 列のほうが優れている (16 次元での複数回の実験では、そのほぼ全てにおいて収束速度、解の変動幅ともに Faure 列のほうが優れていた)。しかしここでも Halton 列同様、次元が高くなると解の変動幅を減らすため、収束速度を犠牲にしている。

また van der Corput 列をそのまま使用しているため、全ての次元において最初の b(基底) 個は全て同じ点列が並んでしまうという問題もある。

3.2 改良 Faure 列

Faure 列では Halton 列のときのように π の置換を行って改善することは難しい。なぜなら Faure 列は置換に対して繊細であり、多くのずらしを取り入れることでオリジナルのものから離れ、改悪になってしまう (事実 Halton 列のとき有効だった置換を行っても改悪になった)。また生成行列の一部を変更することでも同様の結果となった。最終的な解の精度を維持するためには変更点を減らさなくてはならず、そうすることでオリジナルのものとほとんど同様の結果になってしまう。

そこで、それぞれの次元ごとに $a_0^{(i)}, \cdots, a_m^{(i)}$ を P_i 倍する方法を提案する。具体的な P_i については 1 次元目は 1 倍 (オリジナルのもの)、それ以降では基底を b とすると、奇数次元では $b/2$ を超える最小の素数 (16 次元では 11)。偶数次元ではその次の素数 (16 次元では 13) とする。

この方法では π の置換や生成行列の一部を変更することは無いので、Faure 列の性質を著しく失うことを防ぐことができる。また全ての次元で初めの b 個が同じものになる問題も防ぐことができる。P_i の決め方については、全ての次元で同じ P_i とすると隣り合う次元の点列が非常に近いものになってしまう (特に低次元の場合は顕著)。$P_i = i$(恒等置換) としたときでも 2 次元目と 4 次元目、4 次元目と 8 次元目など、ある次元とその 2 倍、3 倍となる次元の点列は近いものになる場合が多く見られた。$1, \cdots, b-1$ をランダムに入れ替え、P_i とする方法では場合によっては悪い点列にってしまう。たとえば P_2=8、P_3=16 とするとそれらの次元の点列は近いものになってしまう。P_i の取り方を上記のようにしたのは、P_i をさまざまに設定し実験を行った結果、良い解が得られた場合が多かったためである。理論的な証明はないが、このように P_i をとると、それぞれの次元の点列が近いものに場合が少なかった。P_i の取り方は特殊な場合でない限り、オリジナルのものよりも改善する。

3.3 実験結果

右図は 2 節で用いた 16 次元関数を Faure 列、改良 Faure 列で計算した結果である。以下の結果より、Faure 列よりも改良 Faure の方が解の収束速度が速いことが確認できる。特に Faure 列では、18 番目のサンプル以降 (基底が 17 であることが要因) で誤差が跳ね上がるが、改良 Faure 列ではそれを軽減し、その後も解の幅を維持しつつ早く収束していることが確認できる。一様乱数を変え複数回の実験を行った場合でも同様の結果が得られた (別の関数を用いた場合でも、多くの場合で改良 Faure 列のほうが良い結果が得られた)。

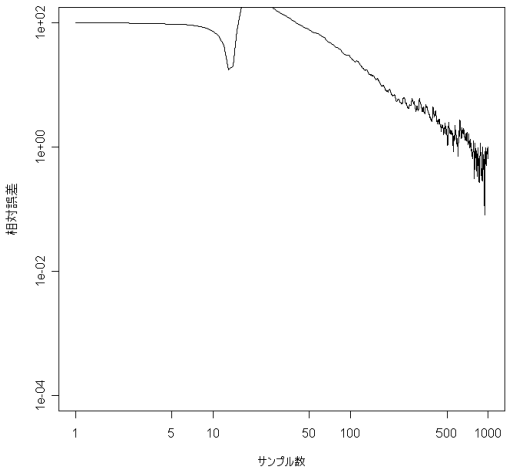

図 6 :Faure 列の相対誤差の推移

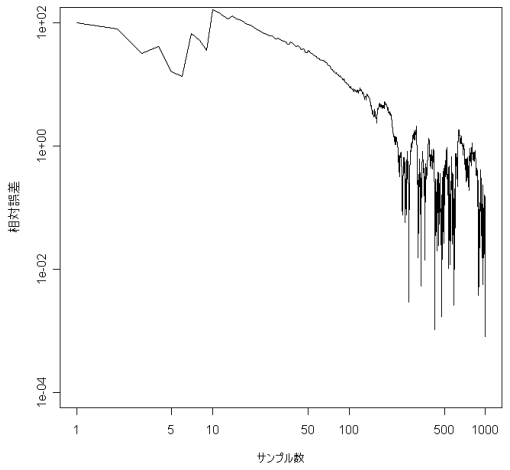

図 7 :改良 Faure 列の相対誤差の推移

参考文献

[1] Alan Genz：Website
http://www.math.wsu.edu/faculty/genz/homepage

[2] Hozumi Morohosi and Masanori Fushimi：準モンテカルロ法の誤差の解析手法の比較, 2000 年度日本オペレーションズリサーチ・学会春季研究発表会, 2000.

[3] Lucille development blog：Website
`http://lucille.asto-net.jp/blog`

[4] Harald Niederreiter：Random Number Generation and Quasi-Monte Carlo Methods, Society for Industrial Applied Mathematics, pp.23-45, 1992.

[5] L'Ecuyer Pierre：Website
`http://www.iro.umontereal.ca/~lecuyer`

[6] 田村勉. 白川浩：一般化 Faure 列による準乱数とそのオプション評価への応用, 平成１０年度研究論文, 東京工業大学, 1998.

[7] 手塚集：計算統計　, 第　部, pp.67-120, 岩波書店, 2000.