Journal of the Operations Research Society of Japan
Vol. 45, No. 4, December 2002


# オプション価格評価から見た 低食い違い量列におけるランダム性の効果*

田村 勉
株式会社格付投資情報センター

(受理 2001 年 9 月 28 日 ; 再受理 2002 年 6 月 25 日)

**和文概要** 収束性が高いと言われる低食い違い量列は幾つか報告されているが，その具体的な生成アルゴリズムは公表されていない．例えば一般化 $Faure$ 列はその 1 つであるが，従来の点列を拡張したため未知の要素が増える結果となった．この部分は収束性に影響を及ぼすと考えられるにも拘わらずその選択基準はやはり明らかではないが，ある種のランダム性を与えるような仕組みを取り入れることにより確実に収束性が向上することは確認されている．

一方，不確定な要素を確率項ととらえることで準モンテカルロ法では従来確立されていなかった誤差評価に利用する手法が提案されている．しかし一般的には計算負荷が非常に掛かる手続きをとるために，低食い違い量列が担う計算の効率化という本来の目的に相反する可能性がある．

本論文では，収束性を高めるように元々ランダム性を与える仕組みをもった点列をその枠組みを継承しつつさらに誤差評価が可能となるように変更する．これは誤差評価に際し問題となる計算負荷の軽減に有効であると考えられる簡便な枠組みをもった点列である．この簡易的な点列の，より複雑な仕組みを持つ点列との数値計算での比較を通じて，パフォーマンスの検証を行った．収束性や計算負荷といった面から十分比較対照となり，ベンチマークとして利用可能であることが確認された．

## 1. はじめに

代表的な低食い違い量列（$low-discrepancy$ 列）として $Faure$ 列や $Sobol'$ 列などが知られているが，これらの点列を単純に利用するだけでは，多くの場合収束性に問題が残り，現実の使用に耐えるものとは言い難い．この問題を解決するような収束性の高い低食い違い量列は幾つか報告されているものの，その具体的な生成アルゴリズムは公表されていない．

低食い違い量列の改良は，既存の点列を基にして行うことが定石である．収束性が高いと言われる一般化 $Faure$ 列や一般化 $Sobol'$ 列といった点列もその名前が示すとおり $Faure$ 列や $Sobol'$ 列といったオリジナルの点列に基づき構築されている．オリジナルの点列に比べ飛躍的な収束精度の向上を実現するものとして Ninomiya and Tezuka[4] などに報告されているが，一般化とあるようにその定義はオリジナルの点列を拡張した形で与えられている．例えば，一般化 $Faure$ 列では未知の要素が増える結果となったが，この部分は収束性に深刻な影響を及ぼすと考えられるにも拘わらずその選択基準は同様に明らかではない．

一方，このような不確定な要素を積極的に利用しようとする取り組みもある．Morohoshi and Fushimi[2]，Owen[5] では，準モンテカルロ法では確立されていなかった誤差評価の方法を提案・検証している．不確定要素を確率項と考えることで誤差評価を行うものである．しかし一般的には計算負荷が非常に掛かる手続きをとるために，低食い違い量列が担う計算の効率化という本来の目的に相反する可能性がある．

田村・白川 [6] では，$Faure$ 列を基に簡単な改良を加えることで計算精度の向上を可能と

*本論文は執筆者の個人的意見であって，いかなる意味においても所属組織を代表するものではない

する実例を示した．点列の分布に対する一様性の尺度である *discrepancy* とその点列から生まれるサンプル・パスの挙動に着目し，生成過程である種のランダム性を加えることが収束性の向上に重要な要素となることを指摘した．このランダム性を与える手段を確率的な方法にすると，上記の誤差評価に必要となる確率的変動を与える手段として利用することができる．そこで本論文では，誤差評価を行うことを前提にランダム性を得る手段として乱数を用いるが，計算負荷の軽減に有効であると考えられるため生成方法の大枠は継承した点列を中心に議論する．この簡易な点列がより複雑な仕組みを持つ点列と比較し，そのパフォーマンスを通してベンチマークとして利用可能であるかを検証する．

本論文では以降，第2節で低食い違い量列の基本概念を，第3節で準モンテカルロ法における誤差評価の方法を紹介する．第4節では，誤差評価を視野に入れた上で第2節に基づいた点列の構成を考える．第5節において，数種のオプションの価格評価および誤差評価を通して第4節で構築した各点列の比較・検証を行う．最後にまとめとして，結果と今後の課題を第6節で述べる．

## 2. 低食い違い量列

ここでは，Niederreiter[3]，Tezuka[7] らにより築かれた低食い違い量列の基本概念を紹介する．

準モンテカルロ法では，低食い違い量列という確定的に生成される点列を用いる．これは特に点列の分布の一様性に重点を置いて，計算精度の効率化を図る手法である．その理論的な根拠となっているのは次の評価式である：

$$\left| \int_{[0,1]^k} f(\mathbf{x}) d\mathbf{x} - \frac{1}{N} \sum_{n=1}^{N} f(\mathbf{x}_n) \right| \leq V_f D_N^{(k)} \tag{2.1}$$

ここで $\mathbf{x}_n$ は $[0,1]^k$ の点列である．$V_f$ は $Hardy-Krause$ の意味での変動と言い，被積分関数 $f$ の積分領域上での変動を表し，$V_f$ が有限ならば被積分関数は有界変動となる．また，$D_N^{(k)}$ は次に定義されるように *discrepancy* を表し，有限個の点列の分布が理想的な一様分布からどの程度のずれを持っているかを表す尺度である．被積分関数が単純な関数であれば，積分計算の収束性が良いことは想像されるが上式からもそのことがわかる．

*discrepancy* は点列の一様性を測る尺度であり，小さいほど一様性が高いことを表す．

**定義 2.1** (*discrepancy*) $P = \{\mathbf{x}_n : n = 0, 1, \cdots, N-1\}$ を $[0,1]^k$ の点集合とし，$\mathbf{y} = (y_1, \cdots, y_k)$ を $[0,1]^k$ の点とする．$E(\mathbf{y})$ を $[0,1]^k$ の部分集合，つまり $[0, y_1) \times \cdots \times [0, y_k)$ として定義する．$\#(E(\mathbf{y}); N)$ は $E(\mathbf{y})$ の中の要素 $\mathbf{x}_n, n = 0, 1, \cdots, N-1$ の個数を表すとする．そのとき点集合 $P$ の *discrepancy* は，$L_\infty-$ ノルムに関して

$$D_N^{(k)} = \sup_{\mathbf{y} \in [0,1]^k} \left| \frac{\#(E(\mathbf{y}); N)}{N} - \prod_{i=1}^{k} y_i \right|$$

で定義される．

列 $\mathbf{x}_n, n = 0, 1, \cdots$ が $[0,1]^k$ で一様に分布しているときかつそのときにかぎり

$$\lim_{N \to \infty} D_N^{(k)} = 0$$

である．よって生成列を出来る限り一様分布に近づけつつ，生成個数を増やすと式 (2.1) の左辺を 0 に近づけることが可能となる．この *discrepancy* を小さくすることができれば，有

限個の生成点は一様分布に近づくことになる．*discrepancy* があるオーダーよりも低い点列を低食い違い量列と言い，*discrepancy* $D_N^{(k)}$ によって次のように定義される：

**定義 2.2（低食い違い量列）** $k$ 次元単位立方体 $[0,1]^k$ に属する（無限）点列 $\mathbf{x}_0, \mathbf{x}_1, \cdots$ は，任意の $N > 1$ に対して，初めの $N$ 点の *discrepancy* が

$$D_N^{(k)} \leq c(k)\frac{(\log N)^k}{N}$$

を満たすとき，低食い違い量列と呼ぶ．ここで $c(k)$ は次元 $k$ にのみ依存する定数である．

代表的な低食い違い量列として *Faure* 列や *Sobol'* 列が以前から知られている．本論文では，*Faure* 列をベースとして派生する点列を考えていく．

**定義 2.3（*Faure* 列）** 点列 $\mathbf{X}_n = (X_n^{(1)}, X_n^{(2)}, \cdots, X_n^{(k)})$ は次のように与えられる：基数 $b$ を次元 $k$ 以上の最小の素数とする．第 1 次元 $X_n^{(1)}$ はこの基数 $b$ による基底逆関数により得られ，基数 $b$ による整数 $n$ の *digit* 展開を

$$n = a_r b^r + a_{r-1}b^{r-1} + \cdots + a_1 b + a_0$$

とすると

$$X_n^{(1)} = \frac{a_0}{b} + \frac{a_1}{b^2} + \cdots + \frac{a_{r-1}}{b^r} + \frac{a_r}{b^{r+1}}$$

である．以降，第 $i(\geq 2)$ 次元目 $X_n^{(i)}$ は

$$X_n^{(i)} = \frac{a_0^{F(i)}}{b} + \frac{a_1^{F(i)}}{b^2} + \cdots + \frac{a_{r-1}^{F(i)}}{b^r} + \frac{a_r^{F(i)}}{b^{r+1}}$$

となる．ただし

$$a_j^{F(i)} = \sum_{l \geq j} {}_lC_j (i-1)^{l-j} a_l \pmod b$$

で与えられる．この展開は次のようにパスカル行列で簡潔に行列表現がなされる：

$$\begin{pmatrix} a_0^{F(1)} \\ a_1^{F(1)} \\ \vdots \\ a_r^{F(1)} \end{pmatrix} = \begin{pmatrix} 1 & & & \\ & 1 & & \\ & & 1 & \\ & & & \ddots \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_r \end{pmatrix} \tag{2.2}$$

$$\begin{pmatrix} a_0^{F(i)} \\ a_1^{F(i)} \\ \vdots \\ a_r^{F(i)} \end{pmatrix} = \begin{pmatrix} {}_0C_0 & {}_1C_0 & {}_2C_0 & \cdots \\ & {}_1C_1 & {}_2C_1 & \cdots \\ & & {}_2C_2 & \cdots \\ & & & \ddots \end{pmatrix}^{i-1} \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_r \end{pmatrix} \pmod b, i \geq 2 \tag{2.3}$$

*Faure* 列は次のように一般化される：

**定義 2.4（一般化 *Faure* 列）** 一般化 *Faure* 列の第 $i$ 次元の $b$ 進ベクトルは，$GF(b)$ 上の正則な下三角行列と式 (2.2)，式 (2.3) のオリジナルの *Faure* 列の第 $i$ 次元の $b$ 進ベクトルにより

$$\begin{pmatrix} a_0^{GF(i)} \\ a_1^{GF(i)} \\ \vdots \\ a_r^{GF(i)} \end{pmatrix} = \begin{pmatrix} l_{00}^{(i)} & & & \\ l_{10}^{(i)} & l_{11}^{(i)} & & \\ l_{20}^{(i)} & l_{21}^{(i)} & l_{22}^{(i)} & \\ \vdots & \vdots & \vdots & \ddots \end{pmatrix} \begin{pmatrix} a_0^{F(i)} \\ a_1^{F(i)} \\ \vdots \\ a_r^{F(i)} \end{pmatrix} \pmod b, i \geq 1 \tag{2.4}$$

で定義される．この下三角行列を恒等行列とすれば，オリジナルの *Faure* 列となる．

さらに，一般化 *Niederreiter* 列の構築理論 [3]，[7] を適用することで式 (2.4) の第 $i$ 次元の $b$ 進ベクトルは

$$\begin{pmatrix} a_0^{GN(i)} \\ a_1^{GN(i)} \\ \vdots \\ a_r^{GN(i)} \end{pmatrix} = \begin{pmatrix} \pi_0^{(i)}(a_0^{GF(i)}) \\ \pi_1^{(i)}(a_1^{GF(i)}) \\ \vdots \\ \pi_r^{(i)}(a_r^{GF(i)}) \end{pmatrix} \pmod{b}, i \geq 1 \tag{2.5}$$

とすることが可能である．ここで $\pi_j^{(i)}, i \geq 1, j = 0, 1, ..., r$ は $\{0, 1, ..., b-1\}$ 上の一対一の置換であり，これを恒等変換とすると一般化 *Faure* 列である（さらに，生成行列による変換前の $b$ 進ベクトルに対しても置換が適用可能であるが，本論文では取り扱わない）．このような置換を本論文では後方置換と呼ぶことにする．なお一般化 *Niederreiter* 列は低食い違い量列である．

## 3. 誤差評価の方法

### 3.1. モンテカルロ法の誤差評価

モンテカルロ法における誤差は次のように評価される．求めたい定積分を

$$I = E[f(X)] = \int_{[0,1]^k} f(\mathbf{x})d\mathbf{x}$$

とする．$\mathbf{x}_n$ を $[0,1]^k$ の点列とし

$$\hat{\mu}_{MC} = \hat{I}_N = \frac{1}{N}\sum_{n=1}^{N} f(\mathbf{x}_n) \tag{3.1}$$

とする．この式は $\hat{I}$ がモンテカルロ法による $I$ の推定量であることを表している．これは不偏推定量 $E[\hat{I}] = I$ であり，分散による誤差は

$$Var[\hat{I}_N] = \frac{1}{N}\int_{[0,1]^k} (f(\mathbf{x}) - I)^2 d\mathbf{x}$$

で評価される．この推定量としては

$$\hat{\sigma}_{MC}^2 = \hat{Var}[\hat{I}_N] = \frac{1}{N(N-1)}\sum_{n=1}^{N}(f(\mathbf{x}_n) - \hat{\mu}_{MC})^2 \tag{3.2}$$

を用いる．

モンテカルロ法では，生成される 1 点それぞれが独立という前提があるためこのような評価が可能となる．分散減少法と呼ばれる手法は，この分散を小さくし計算時間の短縮を図る方法である．ところが準モンテカルロ法に用いる低食い違い量列は，人為的に点列を配置していく手続きをとる．したがって新たに生成される 1 点はそれ以前に生成した点列を確実に反映していることから，準モンテカルロ法にはこの誤差評価を適用することはできない．

### 3.2. 準モンテカルロ法の誤差評価

Morohoshi and Fushimi[2], Owen[5] では，準モンテカルロ法の誤差評価方法の提案・検証を行っている．これはモンテカルロ法の場合とは意味合いの異なる方法であり，点列の集合 $\hat{I}$ に確率的な変動を与え誤差評価を行うものである．確率的な変動を独立に $r$ 回与えることで $\hat{I}_N$ の $r$ 個の標本 $\hat{I}_N^{(j)}, j = 1, ..., r$ が得られたとする．このとき

$$\hat{\mu}_{QMC} = \bar{I}_N^r = \frac{1}{r}\sum_{j=1}^{r} \hat{I}_N^{(j)} \tag{3.3}$$

とする．ここで $Var[\hat{I}_N]$ の推定量 $\hat{Var}[\hat{I}_N]$ を

$$\hat{Var}[\hat{I}_N] = \frac{1}{r-1}\sum_{j=1}^{r}(\hat{I}_N^{(j)} - \bar{I}_N^r)^2$$

とすると，$Var[\bar{I}_N^r] = \frac{Var[\hat{I}_N]}{r}$ であるから，誤差 $Var[\bar{I}_N^r]$ の推定量は

$$\hat{\sigma}_{QMC}^2 = \hat{Var}[\bar{I}_N^r] = \frac{1}{r(r-1)}\sum_{j=1}^{r}(\hat{I}_N^{(j)} - \hat{\mu}_{QMC})^2 \tag{3.4}$$

となる．この平方根は標準偏差の不偏推定量ではないが，今後 $\hat{\sigma}_{QMC}$ を利用する．

モンテカルロ法では，$n$ 番目の点を生成した時点でそれ以前に生成された点列を利用して誤差評価ができるのに対して，準モンテカルロ法の誤差評価方法では，$n$ 番目の点における誤差評価をするためには確率的に独立な $n$ 番目の点が複数個必要ということになる．$n$ 個の点を生成するまでを 1 セットとして確率的に独立な複数個のセットを生成しなければならないために，準モンテカルロ法の利点である収束性を高めて計算負荷を軽減することに影響しかねない．

確率的な変動を与える方法は幾つか提案されているが，Owen[5] の *scrambled net* はその代表的な手法である．これは点列の分布状況を判断する理論である $(t, m, k)-$ ネット，$(t, k)-$ 列（一般化 *Niederreiter* 列もこれに基づいている．詳細は Tezuka[7] を参照のこと）の性質を維持しつつ確率変動を与えることができる一般形，言い換えれば置換を適用し得る究極の形式を表したものである（一般化 *Faure* 列，一般化 *Niederreiter* 列を拡張した考え方とも言える．下三角行列による変換および置換により *scrambled net* の性質を満たす置換となっている）．そのためフルスケールを適用することになれば，1 回毎の計算において大きな計算負荷が掛かることが想像される．誤差評価を行う場合には前述のとおり複数回の計算を要することを考え合わせると，パフォーマンスとのバランスを見ながらより簡易的な手法を選択することも意味がある．

このような課題を視野に入れ，次節では，田村・白川 [6] を継承するより簡単な方法を考える．

## 4. 点列のアルゴリズム

*Faure* 列のメリットの 1 つに次元の増加に対してプログラミング上で柔軟に対応できる点が挙げられる．しかし Ninomiya and Tezuka[4] などで指摘されているように，特に高い次元にオリジナルを用いる場合は，その収束性に深刻な問題が残り現実の使用に耐えるものとは言い難い．

田村・白川 [6] は，*Faure* 列に簡単な改良を加えることで収束性を向上させる方法を具体的に示した．そのためにはオリジナルの *Faure* 列が持つ規則性を崩し，ある種のランダムな性

質を与えることが大きな要素となる．ここで言う規則性とは，例えば生成の初期段階の点列が同一線上に並んでしまうといったことである．これは一様性の測度である *discrepancy* を悪化させるとともにサンプル・パスの単調性や言わば不自然性をも引き起こし，このことが収束性の悪化の大きな原因になると考えられる．ランダム性を取り込むことで *discrepancy* の向上とサンプル・パスの改善を図った結果，オリジナルの点列に対して収束性を格段に高めることに成功した．まずそのアプローチを簡単に述べる．

改良のキーとなるランダム性を与えるには次の2つのアプローチが考えられる：

**改良 4.1（生成行列による改良）** 定義2.4の一般化 *Faure* 列における改良であり，その下三角行列に着目する．

**改良 4.2（後方置換による改良）** 第2節の一般化 *Niederreiter* 列における改良であり，$b$ 進ベクトルに対する置換に着目する．

これらの組合せの中でより簡単なものを採用している．その枠組みを行列形式で表現すると次のようになる：

$$\begin{pmatrix} a_0^{TS(i)} \\ a_1^{TS(i)} \\ \vdots \\ a_r^{TS(i)} \end{pmatrix} = \begin{pmatrix} a_0^{GF(i)} \\ a_1^{GF(i)} \\ \vdots \\ a_r^{GF(i)} \end{pmatrix} + \begin{pmatrix} s_0^{(i)} \\ s_1^{(i)} \\ \vdots \\ s_r^{(i)} \end{pmatrix} \pmod b, i \geq 1 \tag{4.1}$$

ここで $s_j^{(i)}, i \geq 1, j = 0, 1, ..., r$ は $\{0, 1, ..., b-1\}$ 上の整数とする．式 (2.5) において $\pi_j^{(i)}(a) = a + s_j^{(i)} \pmod b, i \geq 1, j = 0, 1, ..., r$ としたものである．

具体的には以下に述べるように3つの点列を用意した．まず改良4.1に関しては

- 各次元の生成行列にその該当する次元数を掛ける．つまり式 (2.4) で $l_{jj}^{(i)} = l^{(i)} = i, j = 0, 1, ..., r$ とする（その他の成分は0）．例えば1次元には1，2次元には2を掛ける，といった次第である．この点列を *GFaureDN* 列と呼ぶことにする．
- 各次元の生成行列に原始根の累乗 $\pmod b$ を掛ける．つまり式 (2.4) で $l_{jj}^{(i)} = l^{(i)} = p^i \pmod b, j = 0, 1, ..., r$ とする（その他の成分は0）．ここで $p$ は基数 $b$ を法とした原始根とする．

という手段を用いた．生成行列にある整数を掛ける操作は，一般化 *Faure* 列の下三角行列として対角成分が同一の対角行列を採用することである．さらに改良4.2を組み合わせたものとして，

- 上述の原始根を使う方法に加え，後方置換としてその該当する次元数を足す．つまり式 (4.1) で $s_j^{(i)} = s^{(i)} = i, j = 0, 1, ..., r$ とする．例えば1次元には1，2次元には2を足す，といった次第である．この点列を *GNiedePR+* 列と呼ぶことにする．

という方法をとった．原始根はランダム性と確定的な方法を両立するための方便として採用した．以上が，原始根を用いた従来の方法である．

ランダム性を取り込むためにはどのような方法でも構わないし，何れにせよこれが得られればオリジナルの点列の悪い特性が取り除かれる可能性が高い．本論文では，誤差評価を行うことを前提として，上述の原始根に代わり乱数の利用を考える．しかし計算負荷の軽減の観点から基本的な枠組みを維持した次のような点列を考える：

- 改良4.1として，式 (2.4) の $l_{jj}^{(i)} = l^{(i)}, j = 0, 1, ..., r$ を乱数により割り当てる（その他の成分は0）．この点列を *GFaureRN* 列と呼ぶことにする．

- 改良 4.1 として，式 (2.4) の $l_{jj}^{(i)} = l^{(i)}, j = 0, 1, ..., r$ を乱数により割り当てる（その他の成分は 0）．改良 4.2 として，式 (4.1) の $s_j^{(i)} = s^{(i)}, j = 0, 1, ..., r$ の値を乱数により割り当てる．この点列を $GNiedeRN+$ 列と呼ぶことにする．

さらに比較のため仕組みが比較的複雑な次の点列も用意する：

- 改良 4.1 として，式 (2.4) の $l_{jk}^{(i)}, i \geq 1, j = 0, 1, ..., r, k \geq j$ を乱数により割り当てる（その他の成分は 0）．改良 4.2 として，式 (2.5) の $\pi_j^{(i)}, i \geq 1, j = 0, 1, ..., r$ を採用する．つまり $\{0, 1, ..., b-1\}$ 上の一対一の置換を各次元およびその列毎に異なるパターンで設ける．それぞれの置換はランダム置換（例えば [1] を参照）により生成する．この点列を $GNiedeRN*$ 列と呼ぶことにする．

次節では，これらの点列を中心として簡単なオプション評価を通した議論を展開する．

## 5. オプション評価への適用

ここでは 2 種類の異なるタイプのオプションの価格評価を題材として，前述の議論に基づいた点列を用いた準モンテカルロ法を中心とした数値計算を行うことで各点列の比較・検証をする．いずれの計算もなるべく共通な条件設定をとることにする．つまり数値積分の次元数，サンプル・パスの各パラメータを共通な値とする．使用する点列は次のとおり：

- *Faure* 列（オリジナルをそのまま使用）
- *GFaureDN* 列（田村・白川 [6] により提案された一般化 *Faure* 列）
- *GNiedePR+* 列（田村・白川 [6] により提案された一般化 *Niederreiter* 列）
- *GFaureRN* 列
- *GNiedeRN+* 列
- *GNiedeRN** 列
- *CombTaus* 列（擬似乱数列 *Combined Tausworthe* 列 [8]）

各点列の構築方法は，第 2 節，第 4 節において説明しているとおり．確率的な変動を与える点列 *GFaureRN* 列，*GNiedeRN+* 列，*GNiedeRN** 列では，乱数により成分を入れ替えた生成行列による 30 パターンを使用する．*GNiedePR+* 列では使う原始根を変えた 30 パターンを使用する．*CombTaus* 列では初期シードを変えた 30 パターンを使用する．ただしいずれの点列もそれぞれのオプションに対して共通の 30 パターンを使用し，100 万個まで生成を行う．

### 5.1. 幾何平均コール・オプション

時点 $[0, T]$ を $0 = t_0 < t_1 < \cdots < t_n = T$ と $n$ 個に分割し，各時点の価格を $S(t_i)$ としたとき，時点 $T$ までの幾何平均

$$G(T) = (\prod_{i=0}^{n} S(t_i))^{1/(n+1)} \tag{5.1}$$

に関するコールオプションを評価する．

$x_i$ を標準正規乱数，$\Delta t = t_{i+1} - t_i, i = 0, 1, 2, .., n-1$ とすれば

$$S(t_{i+1}) = S(t_i) \exp\{(r - \frac{\sigma^2}{2})\Delta t + \sigma x_i \sqrt{\Delta t}\}, i = 0, 1, 2, .., n-1$$

によりサンプル・パスは生成される．今回の数値計算では，各パラメータは次のとおり：サンプル・パスの初期値 $S(t_0) = 110.0$，行使価格 100.0，$T = 1.0$，次元数 $n = 360$（$\Delta t = \frac{1}{360}$）

したがって基数 $b = 367$, $r = 0.1$, $\sigma = 0.2$. このオプションについては解析解 14.3924 を得ることが出来るが, この解に対する収束精度を確認するために取り上げたものである.

計算結果は, 図1から図6に示す:

- 図1: *Faure* 列, *GFaureDN* 列
- 図2: *GFaureDN* 列, *GNiedePR+* 列
- 図3: *GFaureRN* 列
- 図4: *GNiedeRN+* 列
- 図5: *GNiedeRN** 列
- 図6: *CombTaus* 列

各図における縦軸はオプションの価格, 横軸は点列の生成個数を表して, プロットは1万個毎に行っている. 図1では, オリジナルの *Faure* 列 (実線と黒い丸で表す), *GFaureDN* 列 (実線と白い丸で表す) を比較している. 図2では, *GFaureDN* 列 (実線と白い丸で表す) と *GNiedePR+* 列の異なる原始根による結果30セットを破線で表す. 図3, 図4, 図5においては, 破線は異なる生成行列による結果30セットの軌跡を表す. 同時に, 太い実線のうち中心の線が式 (3.3) の $\hat{\mu}_{QMC}$, それを挟む上下の線は式 (3.3), (3.4) により, 上が $\hat{\mu}_{QMC} + 3\hat{\sigma}_{QMC}$, 下が $\hat{\mu}_{QMC} - 3\hat{\sigma}_{QMC}$ を表す. 図6では, 破線は異なるシードによる結果30セットの軌跡を表す. また太破線によって式 (3.1) の $\hat{\mu}_{MC}$, および式 (3.1), (3.2) による $\hat{\mu}_{MC} + \hat{\sigma}_{MC}$, $\hat{\mu}_{MC} - \hat{\sigma}_{MC}$ を表す.

また定量的な結果を表1から表3に示す:

- 表1: 解析解との相対誤差の絶対値
- 表2: 式 (3.3) の平均 (擬似乱数は, 式 (3.1) の平均)
- 表3: 式 (3.4) の標準偏差 (擬似乱数は, 式 (3.2) の標準偏差)

なお表1では, *GNiedePR+* 列, *GFaureRN* 列, *GNiedeRN+* 列, *GNiedeRN** 列および *CombTaus* 列では30パターンのうち各時点で最も誤差の大きいものを選択している. また表2, 表3における生成個数は *CombTaus* 列とそれ以外の点列とでは意味が異なる. 最終的に100万個時点の評価をするために, 擬似乱数 *CombTaus* 列は100万個, その他の点列は100万個 $\times$30パターンを生成している. これは第3節で述べたとおりモンテカルロ法と準モンテカルロ法では誤差評価の方法が全く違うことによるもので, そのため直接比較できないことに注意が必要である.

図1から図6および表1から表3より次のようなことが見て取れる:

- オリジナルの *Faure* 列の場合, 収束性は他の点列と比べ悪いものとなる. 相対誤差を見ると *Faure* 列で100万回を経過した時点でも後方置換を適用する点列の5万回時点の値にも達していない.
- *GFaureDN* 列は元々の *Faure* 列と比べると飛躍的に収束性が改善される. しかし後方置換を組み込んだ点列のスケールで比べると収束ははるかに遅い.
- *GNiedePR+* 列では, 原始根の選択により結果に差が生じるものの *Faure* 列, *GFaureDN* 列と比べると収束性がさらに改善されている.
- *GFaureRN* 列は, *Faure* 列, *GFaureDN* 列に比べると収束性が改善される. 30パターンによる軌跡もいずれも似たようなものとなるが, 標準偏差は他の点列より大きい. また後方置換を組み込んだ点列と比較すると収束性には歴然とした差がある.

- *GNiedeRN+* 列は，元々の *Faure* 列，*GFaureDN* 列，*GFaureRN* 列らに比べると早い段階から解析解に近いところで推移し飛躍的に収束性が改善される．また乱数により生成行列を変更しても大きな影響なく，*GNiedePR+* 列における原始根の選択と比べその差は小さい．

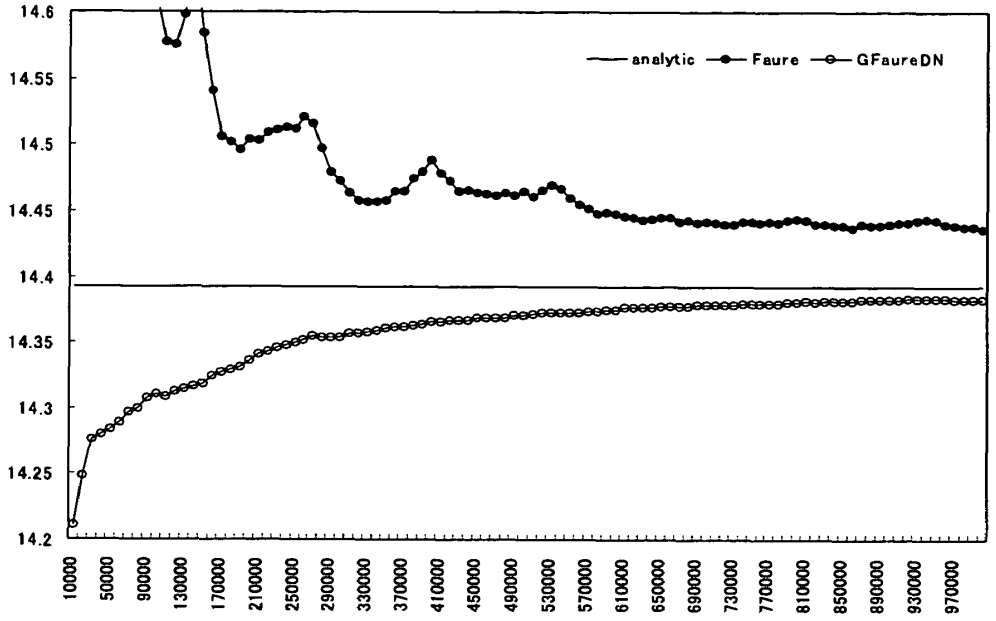


図 1　幾何平均コール・オプション
Faure 列，GFaureDN 列（田村・白川 [1] により提案された一般化列 Faure）の比較

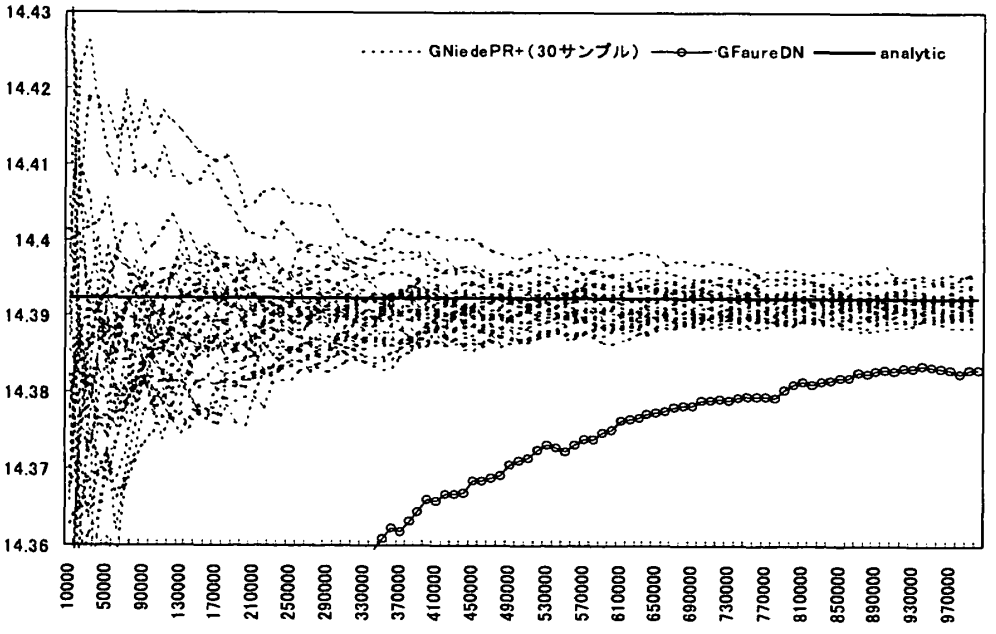


図 2　幾何平均コール・オプション
GFaureDN 列，GNiedePR+列（それぞれ田村・白川 [1] により提案された一般化 Faure 列および一般化 Niederreiter 列）の比較

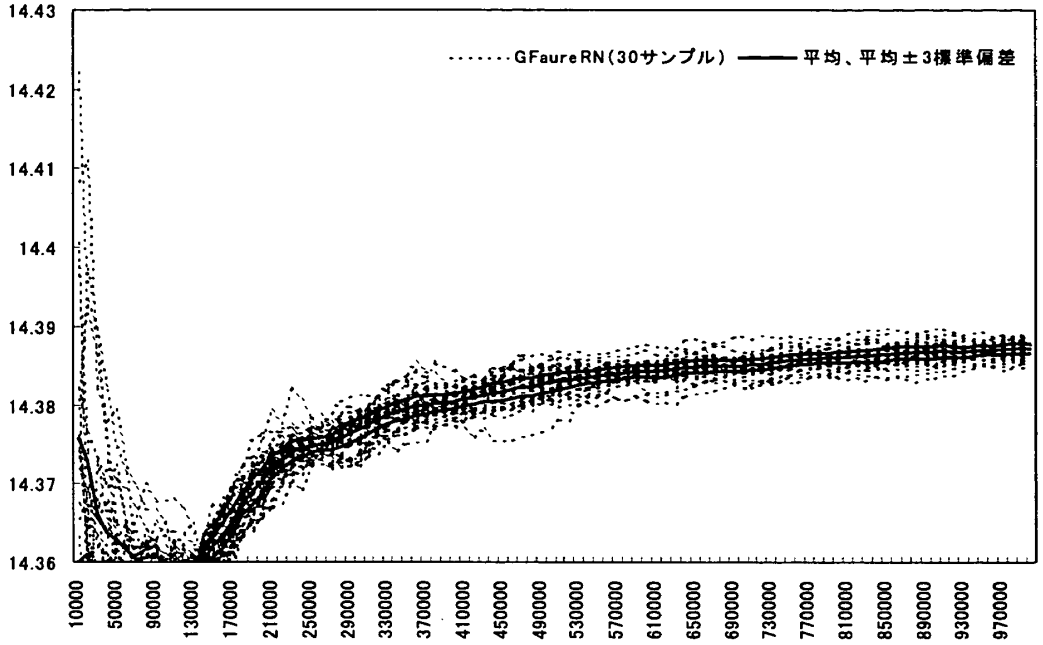


図 3　幾何平均コール・オプション
GFaureRN 列（第 4 節で構築した一般化 Faure 列）の比較

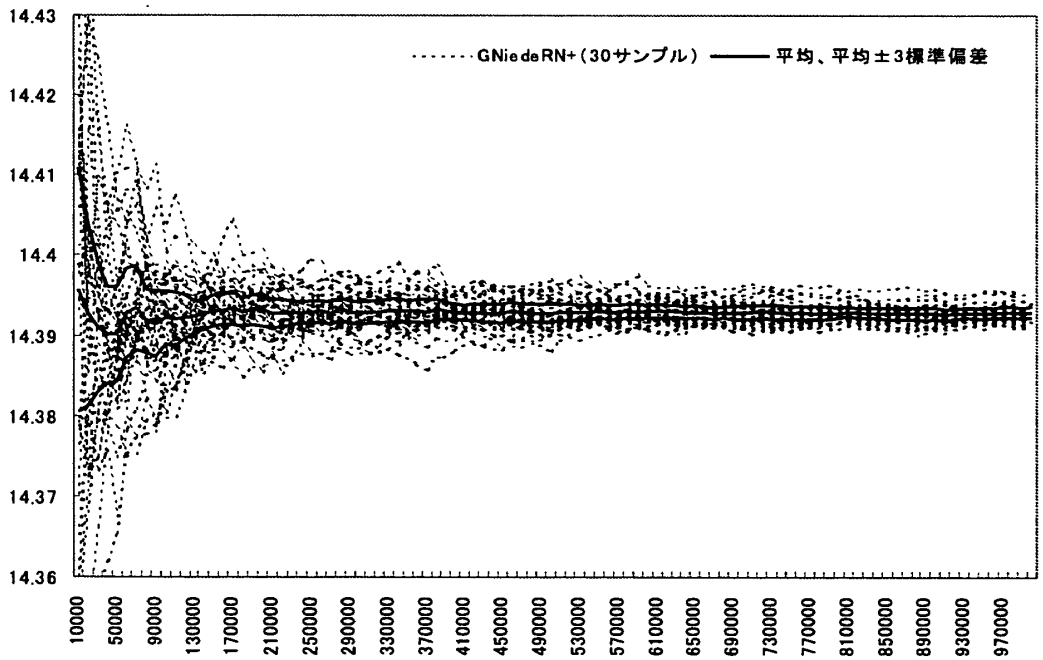


図 4　幾何平均コール・オプション
GNiedeRN+列（第 4 節で構築した一般化 Niederreiter 列）の比較

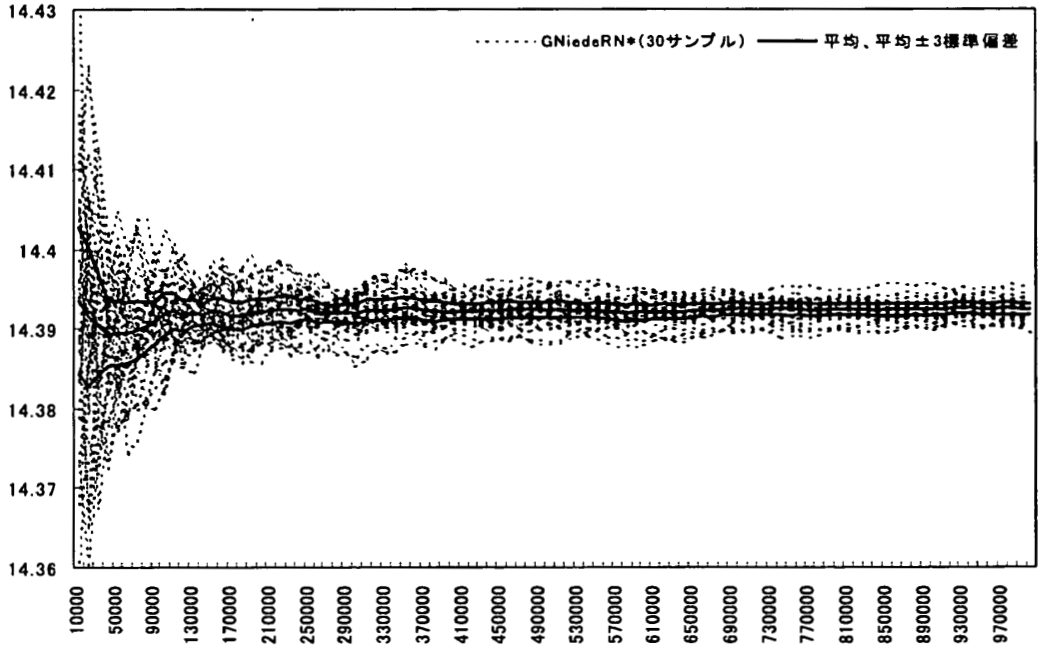


図 5　幾何平均コール・オプション
GNiedeRN* 列（第 4 節で構築した一般化 Niederreiter 列）の比較

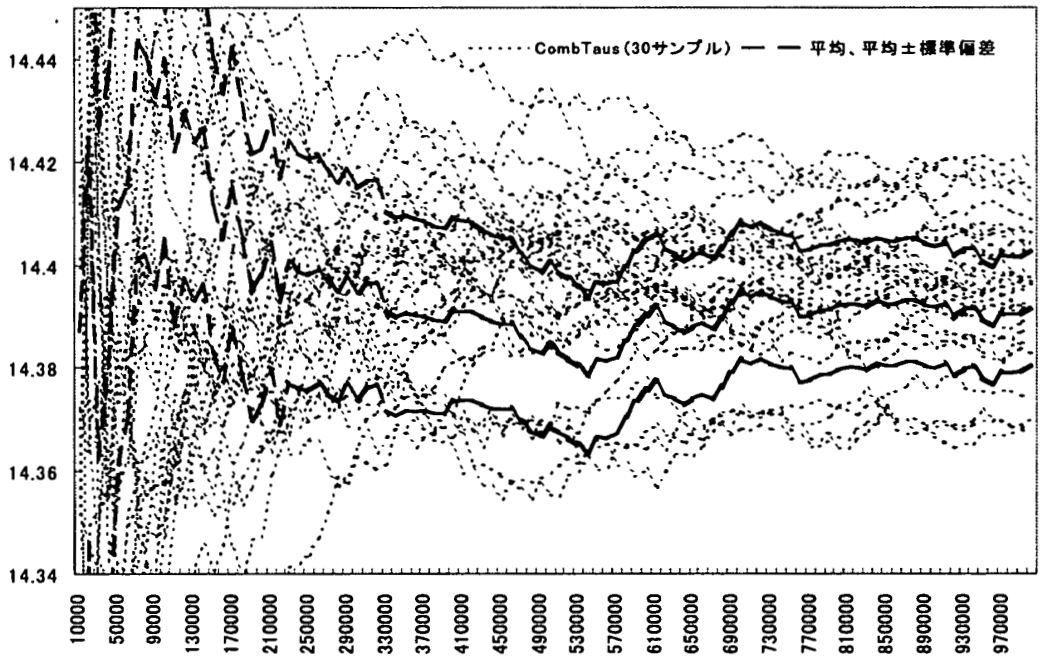


図 6　幾何平均コール・オプション
CombTaus 列（擬似乱数 Combined Tausworthe 列 [8]）の比較

表 1 幾何平均コール・相対誤差

Faure 列，GFaureDN 列（田村・白川 [6] により提案された一般化 Faure 列），GNiedePR+列（田村·白川 [6] により提案された一般化 Niederreiter 列），GFaureRN 列（第 4 節で構築した一般化 Faure 列），GNiedeRN+列（第 4 節で構築した一般化 Niederreiter 列），GNiedeRN* 列（第 4 節で構築した一般化 Niederreiter 列），CombTaus 列（擬似乱数 Combined Tausworthe 列 [8]）の比較

| 生成個数 | Faure | GFaureDN | GNiedePR+ | GFaureRN | GNiedeRN+ | GNiedeRN* | Comb Taus |
|---|---|---|---|---|---|---|---|
| 50000 | 8.8807% | 0.7519% | 0.2191% | 0.3530% | 0.1873% | 0.1052% | 1.0440% |
| 100000 | 1.5167% | 0.5697% | 0.1506% | 0.3224% | 0.0879% | 0.0730% | 0.5143% |
| 150000 | 1.3307% | 0.5109% | 0.1342% | 0.2448% | 0.0530% | 0.0417% | 0.5244% |
| 200000 | 0.7750% | 0.3875% | 0.1161% | 0.1858% | 0.0572% | 0.0467% | 0.4722% |
| 250000 | 0.8347% | 0.2965% | 0.0874% | 0.1408% | 0.0496% | 0.0368% | 0.3383% |
| 300000 | 0.5562% | 0.2632% | 0.0694% | 0.1411% | 0.0339% | 0.0487% | 0.3340% |
| 350000 | 0.4550% | 0.2194% | 0.0648% | 0.1201% | 0.0360% | 0.0395% | 0.3517% |
| 400000 | 0.6695% | 0.1837% | 0.0603% | 0.1062% | 0.0290% | 0.0246% | 0.3357% |
| 450000 | 0.4963% | 0.1667% | 0.0542% | 0.1179% | 0.0290% | 0.0257% | 0.2909% |
| 500000 | 0.5011% | 0.1480% | 0.0404% | 0.1062% | 0.0318% | 0.0301% | 0.2878% |
| 550000 | 0.4660% | 0.1391% | 0.0383% | 0.0862% | 0.0291% | 0.0270% | 0.2629% |
| 600000 | 0.3857% | 0.1202% | 0.0416% | 0.0818% | 0.0270% | 0.0310% | 0.2659% |
| 650000 | 0.3655% | 0.1042% | 0.0378% | 0.0759% | 0.0250% | 0.0251% | 0.2069% |
| 700000 | 0.3428% | 0.0934% | 0.0302% | 0.0705% | 0.0261% | 0.0189% | 0.2300% |
| 750000 | 0.3470% | 0.0892% | 0.0272% | 0.0658% | 0.0234% | 0.0229% | 0.1842% |
| 800000 | 0.3557% | 0.0789% | 0.0267% | 0.0612% | 0.0245% | 0.0192% | 0.2004% |
| 850000 | 0.3221% | 0.0725% | 0.0265% | 0.0640% | 0.0245% | 0.0239% | 0.1844% |
| 900000 | 0.3300% | 0.0652% | 0.0279% | 0.0622% | 0.0237% | 00231.% | 0.1900% |
| 950000 | 0.3483% | 0.0630% | 0.0209% | 0.0512% | 0.0225% | 0.0196% | 0.2003% |
| 1000000 | 0.3059% | 0.0655% | 0.0267% | 0.0526% | 0.0208% | 0.0213% | 0.1793% |

表 2 幾何平均コール・平均

GFaureRN 列（第 4 節で構築した一般化 Faure 列），GNiedeRN+列（第 4 節で構築した一般化 Niederreiter 列），GNiedeRN* 列（第 4 節で構築した一般化 Niederreiter 列），CombTaus 列（擬似乱数 Combined Tausworthe 列 [8]）の比較

| 生成個数 | GFaureRN | GNiedeRN+ | GNiedeRN* | Comb Taus |
|---|---|---|---|---|
| 50000 | 14.3573 | 14.3931 | 14.3896 | 14.4110 |
| 100000 | 14.3565 | 14.3944 | 14.3919 | 14.4401 |
| 150000 | 14.3619 | 14.3925 | 14.3923 | 14.4116 |
| 200000 | 14.3702 | 14.3924 | 14.3922 | 14.3975 |
| 250000 | 14.3748 | 14.3923 | 14.3926 | 14.3980 |
| 300000 | 14.3770 | 14.3922 | 14.3919 | 14.3946 |
| 350000 | 14.3795 | 14.3922 | 14.3927 | 14.3908 |
| 400000 | 14.3805 | 14.3925 | 14.3922 | 14.3911 |
| 450000 | 14.3817 | 14.3924 | 14.3923 | 14.3886 |
| 500000 | 14.3828 | 14.3926 | 14.3924 | 14.3845 |
| 550000 | 14.3837 | 14.3927 | 14.3924 | 14.3818 |
| 600000 | 14.3844 | 14.3925 | 14.3921 | 14.3902 |
| 650000 | 14.3849 | 14.3927 | 14.3923 | 14.3879 |
| 700000 | 14.3850 | 14.3926 | 14.3926 | 14.3948 |
| 750000 | 14.3857 | 14.3926 | 14.3925 | 14.3927 |
| 800000 | 14.3860 | 14.3926 | 14.3925 | 14.3920 |
| 850000 | 14.3864 | 14.3926 | 14.3924 | 14.3925 |
| 900000 | 14.3868 | 14.3927 | 14.3925 | 14.3917 |
| 950000 | 14.3870 | 14.3926 | 14.3925 | 14.3896 |
| 1000000 | 14.3872 | 14.3924 | 14.3925 | 14.3917 |

表 3 幾何平均コール・標準偏差
GFaureRN 列（第 4 節で構築した一般化 Faure 列），GNiedeRN+列（第 4 節で構築した一般化 Niederreiter 列），GNiedeRN* 列（第 4 節で構築した一般化 Niederreiter 列），CombTaus 列（擬似乱数 Combined Tausworthe 列 [8]）の比較

| 生成個数 | GFaureRN | GNiedeRN+ | GNiedeRN* | Comb Taus |
|---|---|---|---|---|
| 50000 | 0.1855% | 0.1980% | 0.1346% | 4.9819% |
| 100000 | 0.0974% | 0.1098% | 0.0890% | 3.5296% |
| 150000 | 0.0474% | 0.0650% | 0.0509% | 2.8887% |
| 200000 | 0.0527% | 0.0660% | 0.0523% | 2.5045% |
| 250000 | 0.0309% | 0.0453% | 0.0437% | 2.2373% |
| 300000 | 0.0421% | 0.0474% | 0.0425% | 2.0416% |
| 350000 | 0.0372% | 0.0457% | 0.0444% | 1.8905% |
| 400000 | 0.0319% | 0.0324% | 0.0297% | 1.7649% |
| 450000 | 0.0376% | 0.0388% | 0.0354% | 1.6624% |
| 500000 | 0.0363% | 0.0376% | 0.0341% | 1.5782% |
| 550000 | 0.0270% | 0.0296% | 0.0295% | 1.5062% |
| 600000 | 0.0274% | 0.0279% | 0.0312% | 1.4423% |
| 650000 | 0.0256% | 0.0257% | 0.0285% | 1.3843% |
| 700000 | 0.0244% | 0.0276% | 0.0244% | 1.3320% |
| 750000 | 0.0242% | 0.0254% | 0.0259% | 1.2869% |
| 800000 | 0.0234% | 0.0229% | 0.0234% | 1.2464% |
| 850000 | 0.0263% | 0.0222% | 0.0251% | 1.2094% |
| 900000 | 0.0257% | 0.0235% | 0.0251% | 1.1764% |
| 950000 | 0.0195% | 0.0228% | 0.0242% | 1.1454% |
| 1000000 | 0.0205% | 0.0230% | 0.0237% | 1.1161% |

- *GNiedeRN** 列は，*GNiedeRN+* 列とほぼ同様の結果であり大きな差はない．標準偏差は相対的に小さいようであるがそれも明確なものではない．
- 擬似乱数列 *CombTaus* 列は，シードの選択により結果に大きなブレが生じ，*GNiedeRN+* 列，*GNiedeRN** 列に比べて収束性は良くない．

### 5.2. ルックバック・オプション

時点 $[0,T]$ を $0=t_0<t_1<\cdots<t_n=T$ と分割し，各時点の価格を $S(t_i)$ としたとき，

$$S(t_n)-\min\{S(t_0),S(t_1),\cdots,S(t_n)\} \tag{5.2}$$

を満期のペイオフとする．

$x_i$ を標準正規乱数，$\Delta t=t_{i+1}-t_i, i=0,1,2,..,n-1$ とすれば

$$S(t_{i+1})=S(t_i)\exp\{(r-\frac{\sigma^2}{2})\Delta t+\sigma x_i\sqrt{\Delta t}\}, i=0,1,2,..,n-1$$

としてサンプル・パスは生成される．今回の数値計算では，各パラメータは次のとおり：サンプル・パスの初期値 $S(t_0)=110.0$，行使価格 100.0，$T=1.0$，次元数 $n=360$（$\Delta t=\frac{1}{360}$）したがって基数 $b=367$，$r=0.1$，$\sigma=0.2$．このオプションについては，連続型の解析解を求めることは可能であるがこのような離散型とは乖離が大きいため比較はしない．

計算結果は，図 7 から図 12 に示す：

- 図7：$Faure$列、$GFaureDN$列
- 図8：$GFaureDN$列、$GNiedePR+$列
- 図9：$GFaureRN$列
- 図10：$GNiedeRN+$列
- 図11：$GNiedeRN*$列
- 図12：$CombTaus$列

各図における縦軸はオプションの価格，横軸は点列の生成個数を表して，プロットは1万個毎に行っている．図7では，オリジナルの$Faure$列（実線と黒い丸で表す），$GFaureDN$列（実線と白い丸で表す）を比較している．図8では，$GFaureDN$列（実線と白い丸で表す）と$GNiedePR+$列の異なる原始根による結果30セットを破線で表す．図9，図10，図11においては，破線は異なる生成行列による結果30セットの軌跡を表す．同時に，太い実線のうち中心の線が式(3.3)の$\hat{\mu}_{QMC}$，それを挟む上下の線は式(3.3)，(3.4)により，上が$\hat{\mu}_{QMC}+3\hat{\sigma}_{QMC}$，下が$\hat{\mu}_{QMC}-3\hat{\sigma}_{QMC}$を表す．図12では，破線は異なるシードによる結果30セットの軌跡を表す．また太破線によって式(3.1)の$\hat{\mu}_{MC}$，および式(3.1)，(3.2)による$\hat{\mu}_{MC}+\hat{\sigma}_{MC}$，$\hat{\mu}_{MC}-\hat{\sigma}_{MC}$を表す．

また定量的な結果を次のとおり示す：

- 表4：式(3.3)の平均（擬似乱数は、式(3.1)の平均）
- 表5：式(3.4)の標準偏差（擬似乱数は、式(3.2)の標準偏差）

なお表4，表5における生成個数は$CombTaus$列とそれ以外の点列とでは意味が異なる．最終的に100万個時点の評価をするために，擬似乱数$CombTaus$列は100万個，その他の点列は100万個$\times$30パターンを生成している．これは第3節で述べたとおりモンテカルロ法と準モンテカルロ法では誤差評価の方法が全く違うことによるもので，そのため直接比較できないことに注意が必要である．

図7から図8および表4，表5より次のようなことが見て取れる：

- オリジナルの$Faure$列の場合，収束性は他の点列と比べはるかに悪いものとなる．このルックバック・オプションでは特に顕著である．
- $GFaureDN$列は元々の$Faure$列と比べると飛躍的に収束性が改善される．しかし後方置換を組み込んだ点列のスケールで比べると収束は見劣りする．
- $GNiedePR+$列では，原始根の選択により結果に差が生じるものの$Faure$列，$GFaureDN$列と比べると収束性がさらに改善されている．
- $GFaureRN$列は，$Faure$列に比べると収束性が改善されるが，$GFaureDN$列と比べると見劣りする．30パターンによる軌跡はいずれも似たようなものとなる．このオプションでは標準偏差は他の点列と差は見られないが，後方置換を組み込んだ点列と比較すると収束性に歴然とした差がある．
- $GNiedeRN+$列は，元々の$Faure$列，$GFaureDN$列，$GFaureRN$列らに比べると飛躍的に改善される．$GFaureRN$列と比べ後方置換をするかしないかで収束性が大きく違う．また30パターンの軌跡からは，乱数により生成行列を変更しても大きな影響がないことがわかり，$GNiedePR+$列における原始根の選択のそれと比べ影響は小さい．両者の差は，第5.1節の幾何平均コールの場合よりも顕著である．
- $GNiedeRN*$列は，$GNiedeRN+$列とほぼ同様の結果であり大きな差はない．このオプションでは標準偏差は相対的に大きいようである．

- 擬似乱数列*CombTaus*列は，シードの選択により結果に大きなブレが生じ，*GNiedeRN+*列，*GNiedeRN∗*列に比べて収束性は良くない．

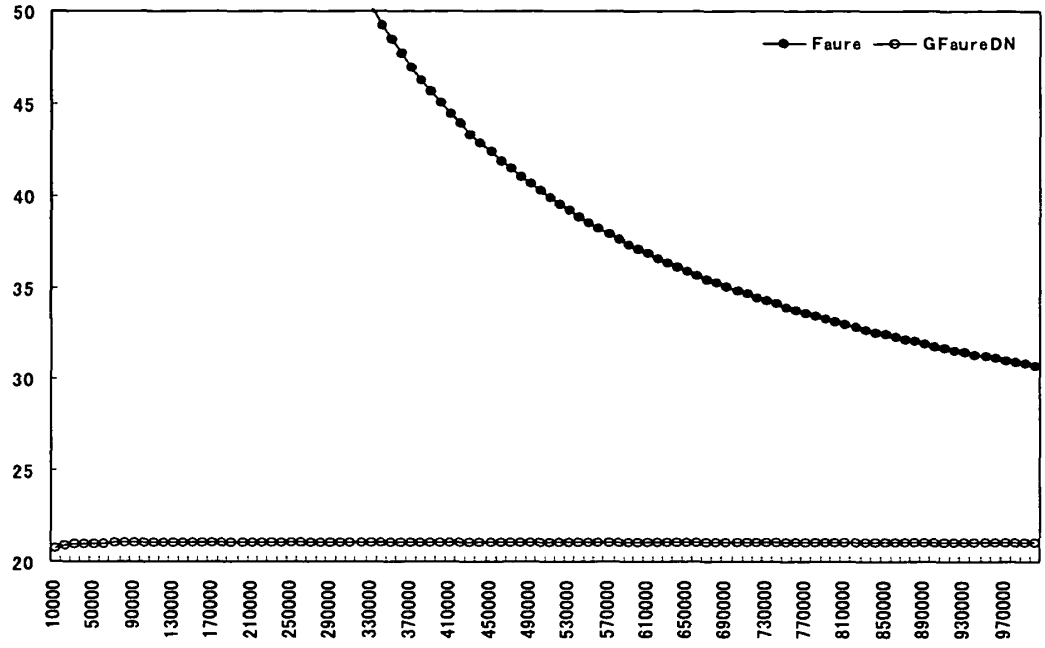


図 7　ルックバック・オプション
Faure 列，GFaureDN 列（田村・白川 [6] により提案された一般化 Faure 列）の比較

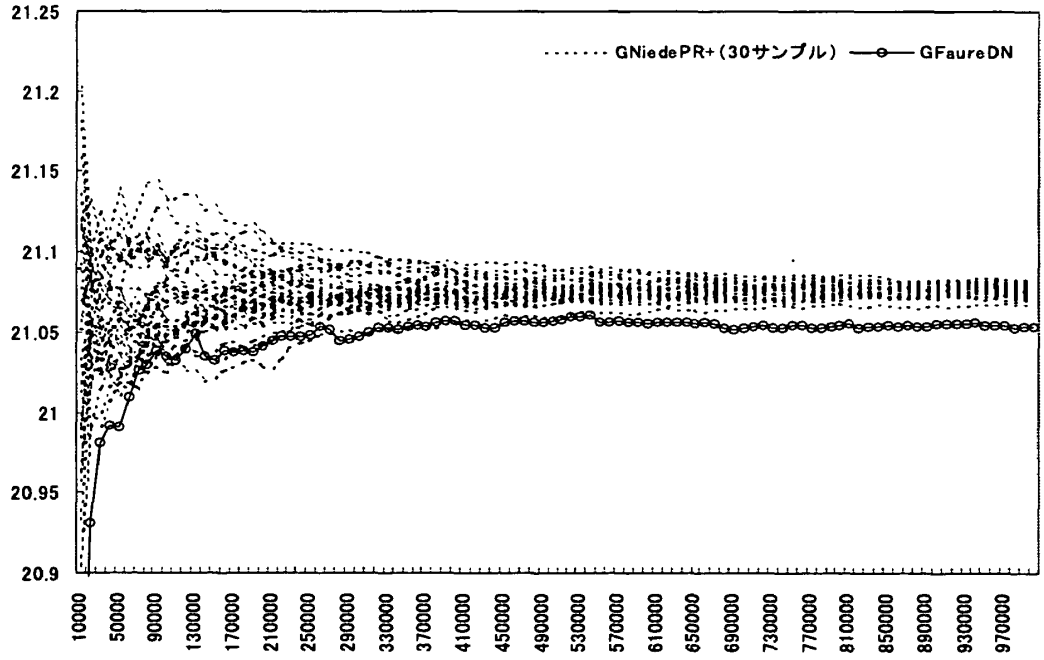


図 8　ルックバック・オプション
GFaureDN 列，GNiedePR+列（それぞれ田村・白川 [6] により提案された一般化 Faure 列および一般化 Niederreiter 列）の比較

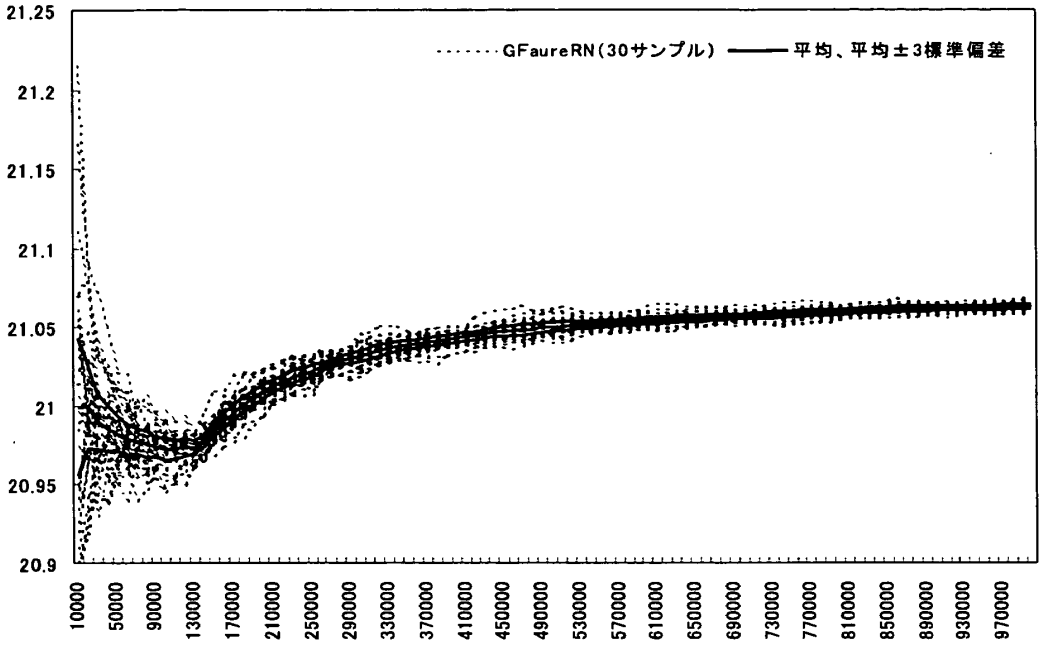


図 9 ルックバック・オプション
GFaureRN 列（第 4 節で構築した一般化 Faure 列）の比較

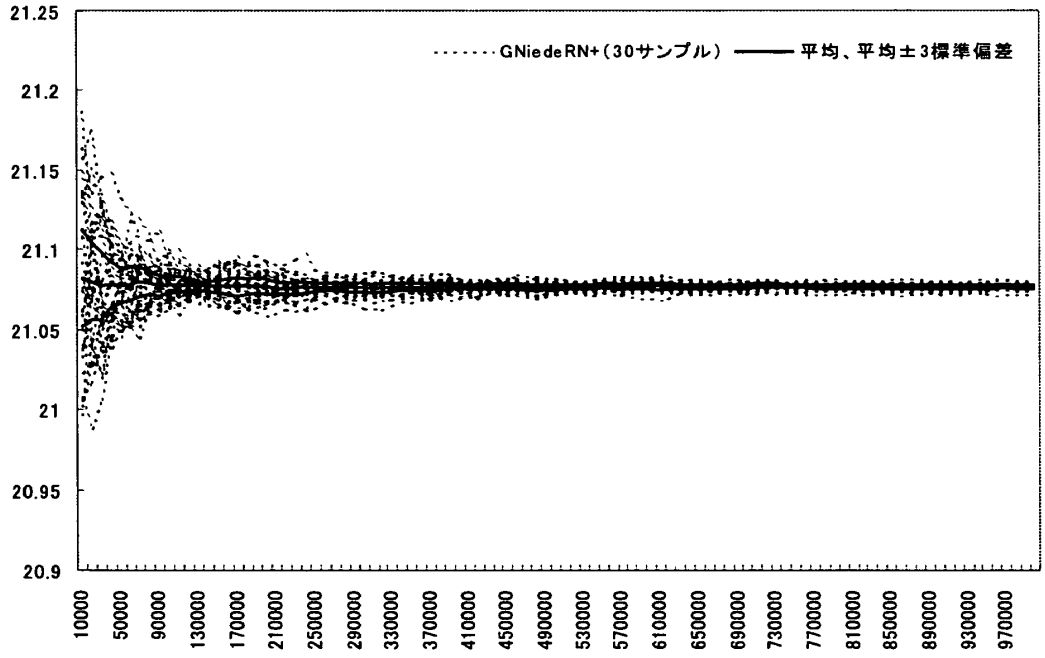


図 10 ルックバック・オプション
GNiedeRN+列（第 4 節で構築した一般化 Niederreiter 列）の比較

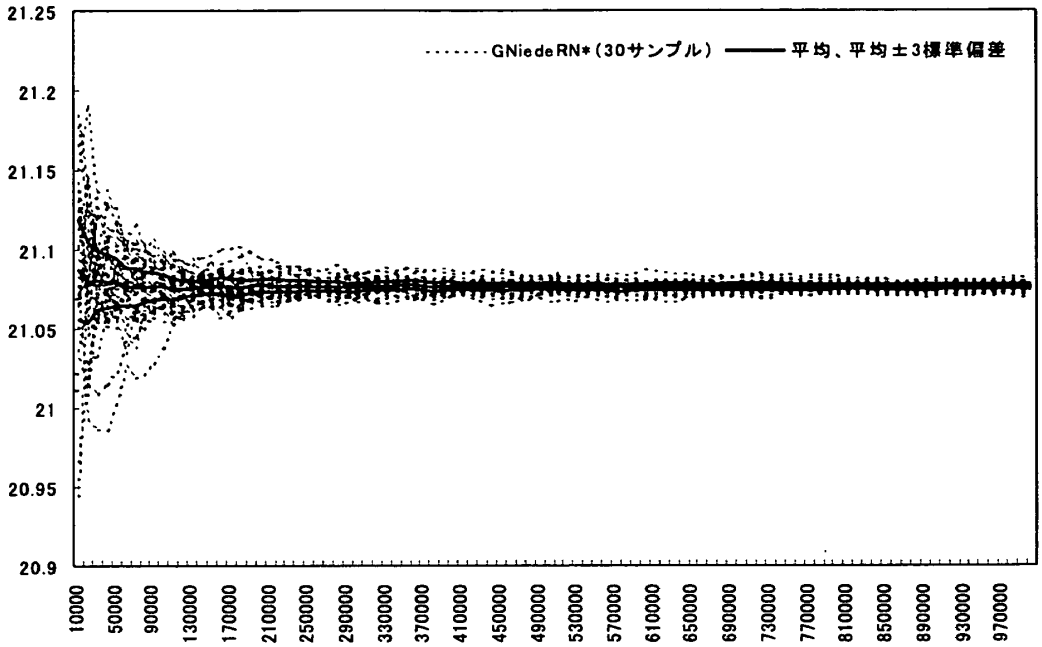


図 11 ルックバック・オプション
GNiedeRN* 列（第 4 節で構築した一般化 Niederreiter 列）の比較

表 4 ルックバック・オプション・平均
GFaureRN 列（第 4 節で構築した一般化 Faure 列），GNiedeN+列（第 4 節で構築した一般化 Niederreiter 列），GNiedeRN* 列（第 4 節で構築した一般化 Niederreiter 列），CombTaus 列（擬似乱数 Combined Tausworthe 列 [8]）の比較

| 生成個数 | GFaureRN | GNiedeRN+ | GNiedeRN* | Comb Taus |
|---|---|---|---|---|
| 50000 | 20.9825 | 21.0772 | 21.0797 | 21.1629 |
| 100000 | 20.9724 | 21.0782 | 21.0770 | 21.1827 |
| 150000 | 20.9850 | 21.0768 | 21.0770 | 21.1202 |
| 200000 | 21.0090 | 21.0769 | 21.0777 | 21.1056 |
| 250000 | 21.0223 | 21.0764 | 21.0774 | 21.1063 |
| 300000 | 21.0320 | 21.0760 | 21.0762 | 21.0967 |
| 350000 | 21.0386 | 21.0766 | 21.0779 | 21.0823 |
| 400000 | 21.0430 | 21.0768 | 21.0771 | 21.0877 |
| 450000 | 21.0473 | 21.0767 | 21.0765 | 21.0915 |
| 500000 | 21.0503 | 21.0762 | 21.0767 | 21.0800 |
| 550000 | 21.0523 | 21.0774 | 21.0766 | 21.0708 |
| 600000 | 21.0543 | 21.0771 | 21.0763 | 21.0755 |
| 650000 | 21.0558 | 21.0767 | 21.0767 | 21.0705 |
| 700000 | 21.0571 | 21.0771 | 21.0767 | 21.0809 |
| 750000 | 21.0585 | 21.0772 | 21.0766 | 21.0770 |
| 800000 | 21.0596 | 21.0766 | 21.0767 | 21.0812 |
| 850000 | 21.0612 | 21.0767 | 21.0766 | 21.0803 |
| 900000 | 21.0620 | 21.0765 | 21.0764 | 21.0767 |
| 950000 | 21.0623 | 21.0769 | 21.0767 | 21.0734 |
| 1000000 | 21.0630 | 21.0767 | 21.0765 | 21.0754 |

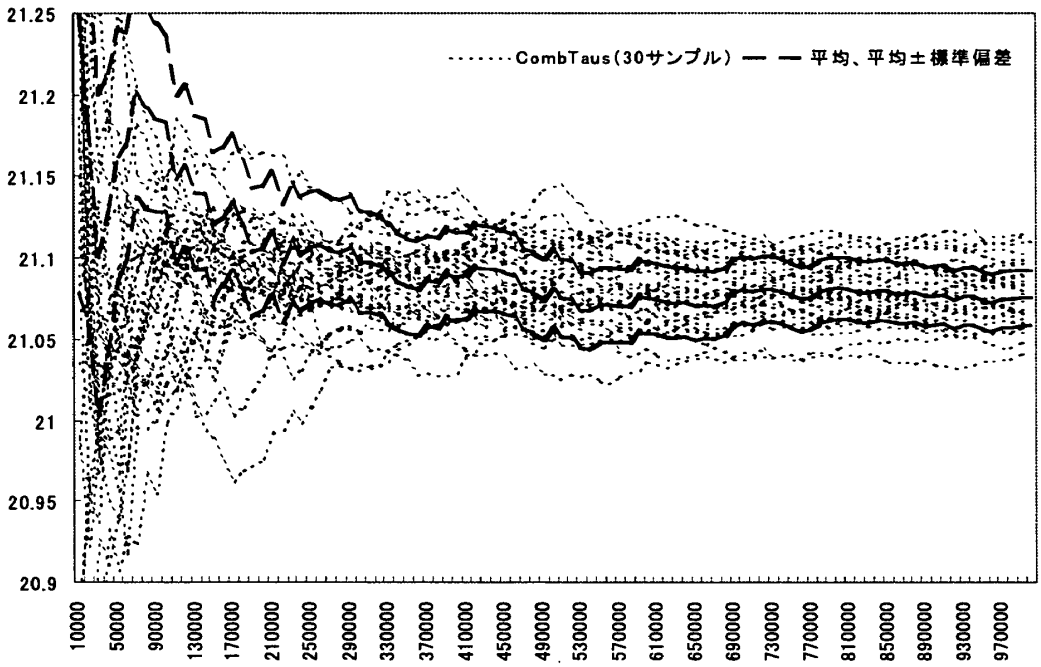


図 12 ルックバック・オプション
CombTaus 列（擬似乱数 Combined Tausworthe 列 [8]）の比較

表 5 ルックバック・オプション・標準偏差
GFaureRN 列（第 4 節で構築した一般化 Faure 列），GNiedeN+列（第 4 節で構築した一般化 Niederreiter 列），GNiedeRN* 列（第 4 節で構築した一般化 Niederreiter 列），CombTaus 列（擬似乱数 Combined Tausworthe 列 [8]）の比較

| 生成個数 | GFaureRN | GNiedeRN+ | GNiedeRN* | Comb Taus |
|---|---|---|---|---|
| 50000 | 0.3657% | 0.3596% | 0.4997% | 7.7049% |
| 100000 | 0.2354% | 0.1849% | 0.2534% | 5.4444% |
| 150000 | 0.1356% | 0.1371% | 0.1510% | 4.4337% |
| 200000 | 0.1443% | 0.1561% | 0.1250% | 3.8323% |
| 250000 | 0.1231% | 0.0872% | 0.0959% | 3.4282% |
| 300000 | 0.1173% | 0.0881% | 0.0929% | 3.1315% |
| 350000 | 0.0958% | 0.0838% | 0.0907% | 2.8976% |
| 400000 | 0.0819% | 0.0531% | 0.0769% | 2.7112% |
| 450000 | 0.0980% | 0.0548% | 0.0826% | 2.5537% |
| 500000 | 0.0962% | 0.0563% | 0.0653% | 2.4202% |
| 550000 | 0.0683% | 0.0504% | 0.0650% | 2.3065% |
| 600000 | 0.0693% | 0.0539% | 0.0665% | 2.2079% |
| 650000 | 0.0607% | 0.0488% | 0.0642% | 2.1209% |
| 700000 | 0.0518% | 0.0421% | 0.0681% | 2.0435% |
| 750000 | 0.0627% | 0.0382% | 0.0607% | 1.9740% |
| 800000 | 0.0525% | 0.0433% | 0.0522% | 1.9119% |
| 850000 | 0.0567% | 0.0414% | 0.0532% | 1.8553% |
| 900000 | 0.0501% | 0.0415% | 0.0496% | 1.8026% |
| 950000 | 0.0495% | 0.0344% | 0.0473% | 1.7552% |
| 1000000 | 0.0511% | 0.0368% | 0.0515% | 1.7110% |

### 5.3. 考察

上述のとおり2つのオプションの価格評価における数値実験では，同じ点列を用いて比較を行ったがどちらもほぼ同様の結果を得た．$Faure$ 列，生成行列を改良した $GFaureDN$ 列および $GFaureRN$ 列，生成行列の改良および後方置換を施した点列である $GNiedePR+$ 列，そして同様の仕組みで乱数を用いる $GNiedeRN+$ 列および $GNiedeRN*$ 列という順で明らかに収束性が改善されて行く．

$GFaureDN$ 列のような簡単な改良でも元々の $Faure$ 列と比べると飛躍的に収束性が改善される．これはオリジナルの $Faure$ 列のもつ規則性を崩すことが有効であることを示している．ここで言う規則性とは，例えば生成の初期段階では全ての次元で同じ成分をもつことでその配置が同一線上に並んでしまうといったことに象徴される．これは一様性の測度である *discrepancy* を悪化させるとともに実際の価格計算において大きな影響を与えるサンプル・パスの単調性や言わば不自然性をも引き起こす．$Faure$ 列は言わば計算には無駄なサンプル・パスが初期時点から繰り返し生成され，これが最後まで収束性に影響し続ける．低食い違い量列であることと実際のサンプル・パスが効率的であることは必ずしもイコールではない．今回のような経路依存タイプのオプションのサンプル・パスでは，点列の1次元目が初期時点から1時点目の推移に，2次元目が1時点目から2時点目の推移に，といった対応になる．点列の生成において隣り合う次元間の影響が削減できればサンプル・パスは多様化され，つまりは効率化される．ルックバック・オプションでは，各サンプル・パスが辿った点を平均化する幾何平均に比べ，各サンプル・パス中の1点を選択するという形態であるためより経路依存性が高い．$Faure$ 列がルックバック・オプションで他の点列と乖離が大きいのは，サンプル・パスの不効率性が経路依存性により大きな影響を与えるためと考えられる．これに対して $GFaureDN$ 列の次元数を生成行列に掛ける操作でもその影響を幾分か緩和されたと考えられる．

さらなるサンプル・パスの効率化には，生成過程においてある種のランダム性を与えることが効果的である．そのために生成行列への改良と後方置換が利用できる．生成行列を改良する $GFaureRN$ 列では，やはり $Faure$ 列と比べると飛躍的に収束性が改善されるが，必ずしも $GFaureDN$ 列より収束性が良いとは言えない．$Faure$ 列が生み出すような極めて特異なサンプル・パス（例えば，初期値からずっと上昇し続けて，式 (5.1), (5.2) の値が非常に大きくなるもの）があれば，その影響で解析解あるいは収束解に対して（極めて）大きい値の方向から徐々に減少し収束して行く．しかし生成行列のみの改良点列では，そのような特異なものを排除している分，$Faure$ 列の場合ほど影響を与えるサンプル・パスは生成されず，初期値付近で推移する傾向がある（第4節で述べたとおり，田村・白川 [6] では生成行列の改良に乱数ではなく原始根を用いているが，この点を指摘している）．したがって $Faure$ 列と比べるとより解析解あるいは収束解に近い値から収束への過程が始まると考えられる．これは $GFaureDN$ 列の場合にも当てはまることであろう．

生成行列だけの改良となる $GFaureRN$ 列とこれにさらに後方置換を与える $GNiedeRN+$ 列と比較すると明らかに後者の方が良い結果を示した．これは後方置換とすることで行列では不可能な変換が可能となり，そのことが大きな役割を果たすためと考えられる．例えば $b$ 進ベクトルの最上位桁における0から0以外への変換がそうである．この部分は新しい点列を生成する都度値が入れ替わるとともに一番大きい値となる桁であるため，サンプル・パスの挙動に一番大きな作用をもたらす．よってこの部分をうまく利用した結果，つまり後方置換を用いたことにより，上で述べた $GFaureRN$ 列までの問題点を解消する（$Faure$ 列の場

合ほど単調かつ極端ではなく，しかし$GFaureRN$列のように初期値付近ばかりに集中しない）ようなサンプル・パスの早期の多様性・効率化を実現させることが可能となったと考えられる．

$GNiedeRN+$列は，後方置換として和を使う簡略・単純な方法であるが，生成行列の改良と合わせることが前提であり，それによってある程度のランダム性を確保できると考えられ，また一般的な後方置換とは違い細かな置換の対応表を持つ必要がないといった点も考慮される点であろう．結果として一般的な置換を後方置換としてもつ$GNiedeRN*$列との間に差は見られなかった．

$GNiedePR+$列と$GNiedeRN+$列はともに和を使う後方置換をもつ点列である．ともにパラメータを変え異なる30パターンを生成し比較したが，後者の方がパラメータ選択による影響が少なくまた収束性も良い．ここでも上述の次元間の影響に関連したと考えられる．$GNiedePR+$列では生成行列に原始根の次元数の累乗を掛ける，後方置換として次元数を和に使用する．これに対して$GNiedeRN+$列ではともに乱数を使用するため，比較して次元間の影響を排除できるからと思われる．

次に計算における負荷を比較するためそれぞれの点列の生成時間を計測した．360次元で100万個の点列を生成するのに要する（価格計算はしない）時間は，$Faure$列，$GFaureRN$列が513秒，$GNiedePR+$列，$GNiedeRN+$列が612秒，$GNiedeRN*$列が884秒，擬似乱数列である$CombTaus$列が358秒であった（PentiumIII 450MHz）．$Faure$列と$GFaureRN$列はともに生成行列による変換のみであるから両者に差はない．$Faure$列，$GFaureRN$列と$GNiedePR+$列，$GNiedeRN+$列の時間差は，後方置換を適用するかどうかによるものである．さらに$GNiedeRN*$列は，下三角行列による変換およびより細かい後方置換が加わる分一層時間を要し，$GNiedeRN+$列と比べて同様の収束性であるにもかかわらず計算時間は1.5倍近く掛かることになる．$CombTaus$列は他の点列と比べて生成時間が少ないが，上記の数値実験が示すとおり収束性が悪いためこのことは優位とは言えない．

## 6. 結論と今後の課題

以上の議論から次のような結論を得た：

- ランダム性を与えることによるサンプル・パスの早期効率的な生成が効果を生むと考えられる．したがって乱数によりランダム性を取り込むことは，オプション価格評価において収束性を高める手段となり，同時に準モンテカルロ法における誤差評価を可能とすることから有効な改良であると言える．逆に準モンテカルロ法における誤差評価の利用が低食い違い量列にランダム性を得るための手段として乱数を用いることに対する動機付けとなる．
- 良好な結果を示した2つの後方置換を適用した点列を比較した場合，置換として単純に和を使う簡易的な手法とより一般的な置換との間に大きな差異は見られない．この和による置換代用は計算負荷を大幅に減少させる手法となり得るものであり，少なくともこの程度のオプションの価格評価においてはベンチマークとしては十分有効であろう．

今後の課題としては次のようなことがあげられる：

- 生成行列のみの改良点列における収束の様子を見ると，どちらのオプション評価とも，生成初期段階で解（解析解あるいは収束解）の上方にあったとしてもその後解の下方に一旦大きく落ち込み，生成個数10万個あたりで反転し上昇を始めて解へ収束に向か

う，という傾向があった．加えて，生成個数10万個以降の収束の様子は，生成初期段階の位置に関係なく似通っていていずれも解の下方に偏ったものとなった．このような反転や偏りといった現象が起こる原因，何故生成個数10万個近辺で起こるのか，30パターンの生成行列で比較を行ったがそれ以外でもこのような結果が生じるのか，他のオプション評価においてもそうなるのか，また生成個数10万個以降で収束を始めるのであれば最初の10万個ほどを使用しないことで収束性向上の効果が得られるのか，といった点を解明する必要がある．

- 今回の2つのオプション評価による数値計算では，後方置換として和を使う場合とより一般的な置換の場合でパフォーマンスに大きな差異は見られなかったが，より多様なサンプル・パスを要求するような数値計算では結果が異なるかも知れない．したがってより複雑な仕組みをもつ金融商品の価格評価を取り上げ比較する必要がある．

## 参考文献

[1] J.E.Gentle: *Random Number Generation and Monte Carlo Methods* (Springer,1998).

[2] H.Morohoshi and M.Fushimi: A practical approach to the error estimation of quasi-monte carlo integrations. *Monte Carlo and Quasi-Monte Carlo Methods 1998* (Springer, 2000) 377-390.

[3] H.Niederreiter: Random number generation and quasi-monte carlo methods. *Number 63 in CBMS-NSF Regional Conference Series in Applied Mathematics* (SIAM, 1992).

[4] S.Ninomiya and S.Tezuka: Toward real-time pricing of complex financial derivatives. *Appl. Math. Finance*, 3 (1996) 1-20.

[5] A.Owen: Monte carlo variance of scrambled net quadrature. *SIAM J. Numer. Anal.*, **34**-5 (1997) 1884-1910.

[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．ジャフィー・ジャーナル［1999］金融技術とリスク管理の展開 (東洋経済新報社, 1999) 95-115.

[7] S.Tezuka: *Uniform Random Numbers: Theory and Practice* (Kluwer Academic Publishers, Boston, 1995).

[8] S.Tezuka and P.L.Ecuyer: Efficient and portable combined tausworthe random number generator. *ACM Transactions on Modeling and Computer Simulation* **1** (1991) 99-112.

田村 勉
株式会社格付投資情報センター
格付本部
ストラクチャードファイナンス部
〒103–0013　東京都中央区日本橋人形町3–8–1
E-mail: ttamura@r-i.co.jp

## ABSTRACT

# THE EFFECT OF RANDOMIZED LOW DISCREPANCY SEQUENCES IN OPTION PRICING

Tsutomu Tamura
*Rating and Investment Information, Inc.*

New derivative products usually have complex payoff structures depending on multiple risk factors. In such situation numerical computation methods, such as Monte Carlo and quasi-Monte Carlo methods, become very powerful tools because of difficulty in evaluating their pricing model analytically.

Low discrepancy sequences installed in quasi-Monte Carlo methods make it possible to produce the uniformity of distribution over the domain of integration, i.e. one or more dimensional unit cube, even for a small number of sample points, which makes numerical integrations to be efficient. Classical low discrepancy sequences, e.g. Faure sequences, are not always satisfactory for multi-dimensional integrations. However, some of generalized Faure sequences can attain quite high performance to compute high-dimensional integrations practically required in financial derivatives pricing, which have been reported in some papers. Unfortunately, none of detailed techniques for the practical construction of such high performanec generalized Faure sequences is shown in them. Incidentally, we can confirm that applying a kind of randomization to the classical sequences leads to realize better convergence performance than the original sequence, as we have reported.

And recently, some error estimation methods for quasi-Monte Carlo simulation were proposed and experiments taking up such error estimations in numerical evaluations were reported. These methods require certain probabilistic structures of their internal sequences and the most generalized class of them demands huge computational quantity, which is a major problem to be solved in analyzing the quasi-Monte Carlo errors.

In this paper, we try to modify low discrepancy sequences with randomized structures originally based on generalized Niederreiter sequences, to keep their performance of convergence and apply the quasi-Monte Carlo error estimation method to it. The structure of this sequence is simple, which can reduce computational complexity. These simplified sequences are applied to the numerical evaluation of path-dependent options in order to compare other low discrepancy and pseudo-random number sequences. We demonstrate that the sequences we proposed attain comparable performance of convergence, error estimation and evaluation time to sequences with more generalized and complicated probabilistic structures.