岡山理科大学紀要 第40号 A pp. 1-7(2004)

裾の重い分布に従う乱数の発生

林　聡美・竹中　茂夫*
岡山理科大学大学院理学研究科応用数学専攻
*岡山理科大学応用数学科

(2004年3月25日受付、2004年11月5日受理)

1　緒言

種々の分野で裾の重い分布に従う確率変数が使われ始めている。モデルと現実の比較のためには、計算機シミュレーションが重要な役目を果たしているが、このシミュレーションに使われている乱数は本当に裾が重いのだろうか。裾の重い、すなわち平均や分散が無限であるような確率分布に従う乱数を、有限性に縛られた計算機で発生する事は本質的に不可能である。従って問題は、現実にシミュレーションで使用されている擬似乱数が無限大とみなせる程大きな平均又は分散を持つのか、またそうでなければこの点を改善できるような乱数発生が可能であるのかということになる。

乱数に関わる話題では、使用する擬似乱数発生ルーティンが主要な問題とされがちだが、ここではそれを避けるために、物理乱数発生ボードで得られた乱数を、15ビット（又は31ビット）の理想乱数と考えてシミュレーションを行っている。物理乱数が理想乱数であるかどうかという議論については、今は考えないこととする。

2　組込乱数

通常の計算機言語では、次に示す線形合同法を用いた疑似乱数を与える関数が用意されている事が多い。

まず、初期値（乱数の種–seed）x_0 を用意する。これから次式で発生される数列 $\{x_n; n = 0, 1, 2, \dots\}$ が線形合同法による疑似乱数と呼ばれる。

$x_n \equiv a x_{n-1} + b \mod (M)$,

a, $n > 0$, n, b は整数、$M > 0$ は十分に大きい整数。

M は計算機で扱える整数の範囲内で最大の素数を使うか、もしくは、$M = 2^N$（N は整数）とする。

この数列は周期性を持ち、その周期は $M(= 2^N)$ を越えない。周期を最大にするための必要十分条件として、(1)〜(3) が知られている。

(1) b が M と互いに素である。
(2) $a-1$ が M を割り切るすべての素数の倍数である。
(3) M が4の倍数であれば、$a-1$ も4の倍数である。

例えば、マイクロソフト社の Visual C++ Version 6 では、組み込み乱数 rand( ) は、15ビットの整数型 (short int) の乱数を与えてくれる。ただし、周期が 2^{31} である事を考慮に入れれば、単純に単精度整数型の線形合同法を使用してはいないようである。現在のCPUのレジスター長が32ビットであることを考えると、31ビットの線形合同法を使って発生させた乱数の上位15ビットを採用して、周期を表面に出さない工夫がなされているようである。

こういった、組み込み乱数を使う上での問題点は、

(1) 解像度が15ビットしかない、すなわち、高々3万程度の異なる値が与えられるだけである。
(2) 周期が２０億（$=2^{31}$）以下
(3) 理論的に独立な乱数列とはいえない。

が挙げられる。(1),(2) はプログラム上の工夫で回避できる可能性があるが、線形合同法であるかどうかを超えてアルゴリズムを使って発生させる以上 (3) の独立性は回避できない。本論文では、これらの点を踏まえて、裾の思い分布に従う乱数列の発生について考慮していく。

3　裾の重い分布に従う乱数

3.1　裾の重い分布

ここで扱う裾の重い分布は、密度が

$$f(x) = \frac{(\beta - 1)}{x^{\beta}}, x \in [1, \infty), \ 1 < \beta < 3,$$

であるものとする。

これに限定する理由は、この分布が $\alpha = \beta - 1$ の安定分布の吸引域に属する、特に $\beta < 2$ では、正の片側安定分布の吸引域に属することであり、安定分布の

裾が漸近的にこの分布に従うことであり、さらに自己相似性があり理論及びプログラムとの相性が良いからである。

この分布に従う確率変数は、$1 < \beta < 3$ では、分散が無限大となり、$1 < \beta \leq 2$ では、平均も無限となる。

3.2 一様乱数からの変換

U を $[0,1]$ の一様分布にしたがう確率変数とする。すなわち、分布が

$$P(a < U \leq b) = b - a,\ 0 \leq a \leq b \leq 1$$

で与えられる確率変数とする。

一般の確率分布に従う確率変数 X は、その分布関数

$$F_X(c) \equiv P(X \leq c)$$

の逆関数を用いて、確率変数 U からの変換

$$X = F_X^{-1}(U)$$

で得られる。 すなわち、分布関数を計算すると

$$\begin{aligned} P(X \leq c) &= P(F_X^{-1}(U) \leq c) \\ &= P(U \leq F_X(c) \\ &= F_X(c)。 \end{aligned}$$

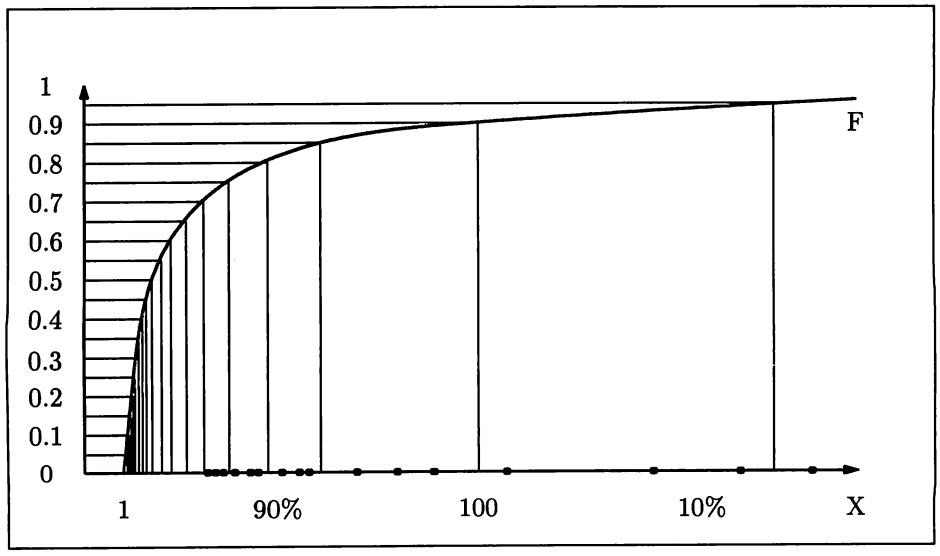
現実に計算機で得られるのは、整数型の疑似乱数であるが、それを $[0,1)$ に値を持つようノーマライズして得られる疑似乱数を、上の U の近似として考えてみよう。U が有限な離散値しか取らなければ、それを変換して得られる X もやはり有限個の離散値しか取りえない。従って、上に有界である。これでは、平均、分散といった X の特性量は有限でとなり、無限の値は取り得ないことは明白である。

ここでは、分布関数として

$$F(x) = 1 - \frac{1}{x^{\beta-1}}$$

を取りあげ、さらに U を 15 ビットの擬似乱数として話を進めていく。(実際、よく使われている Visual C の組込み乱数は、15 ビット乱数であるが)。

次の図で分かるように 0.9 から 1 の範囲には、U の約 32,000 個の離散値のうち 10% 約 3,200 個ががあり、それらが X の値としては無限大を含む区間に再配置されている。これでは、確率は低いが平均とか分散に対する寄与の大きい部分の近似度が、荒すぎる、これが、裾の重い乱数を発生させる上で 1 つの問題点である。



3.3 自己相似性を用いた接木

密度関数が

$$f(x) = (\beta - 1)\frac{1}{x^{\beta}}, \quad 1 \leq x$$

であるような、独立な確率変数列 $X, X_1, X_2, X_3, \cdots$ を考える。正数 C を 1 つ留めて考える。

$$Y_1(\omega) = \begin{cases} X(\omega) & \text{if } X(\omega) \leq C \\ C \times X_1(\omega) & \text{if } X(\omega) > C \end{cases}$$

で定義される確率変数 Y_1 が、X と同分布であることは簡単な計算で分かる。これは、簡単な入れ子構造となっている。すなわち、Y_1 が X_1 と同分布であることを用いて上式 X_1 の代わりに、Y_1 の独立なコピーを代入しても、得られる確率変数は又、基の X と同分布となる。これを続けると、

$$Y = \begin{cases} X(\omega) & \text{if } X(\omega) \leq C \\ C \cdot X_1(\omega), & \text{if } C < X(\omega), \\ & C \cdot X_1(\omega) \leq C^2 \\ C^2 \cdot X_2(\omega), & \text{if } C < X(\omega), X_1(\omega), \\ & C^2 \cdot X_2(\omega) \leq C^3 \\ C^3 \cdot X_3(\omega), & \text{if } C < X, X_1, X_2, \\ & C^3 \cdot X_3 \leq C^4 \\ \cdots & \cdots \end{cases}$$

で得られる確率変数 Y もまた、X と同じ裾の重い分布に従うことがわかる。

3.4 再帰的プログラム

前節で得られた Y は、理論的には同語反復的なものでしかないが、現実の計算機の世界では意味を持ってくる。

まず、$X, X_1, X_2, \cdots$ が、離散的な擬似乱数から変換された近似的乱数だとする。Y_1 では、0.9〜1.0 の区間に 3,200 の点を分布させる代わりに X_1 による 32,000 個の点を分布させている。すなわち、無限という特性につながる部分をよりよく近似した点の分布となっている。従ってこれをくりかえした Y では、無限に関連する部分をどんどん点の数を増やして（その分その点が出現する確率を減らして）近似していくことになる。当然、数学的な（無限回の入れ子を許せば）平均は無限となる。

実際に $C = 100$、と置いたプログラム例を示す。プログラム中の関数¥get_rand( ) は、再帰なく定義された X の離散的近似とする。また N は再帰の深さを示す大域変数とする。すなわち、得られる乱数が計算機で扱える最大数（10^{100} 程度）を超えないように再帰の深さを制限するために使われている。たとえ確率 0 であろうが無限の繰り返しを含むものはアルゴリズムとは呼べない（すなわち、正しいプログラムとはいえない）ことを配慮して再帰の深さを 49 以下と制限している。これによる影響は次の節で検討する。

```
double makerandom(unsigned long N)
{
   double  IZ,RN;
   IZ= get_rand();       //get X
   if( ( IZ <= 100) || ( N > 49 ))
   {
       RN =IZ * (pow(100.0, N ));
       N=0;
       return( RN );
   }
   else {
       N++;
       return (makerandom( N ));
            // recursive call
   }
}
```

3.5 再帰によるオーバーヘッド

例えば、$\beta = 1.5$ と取った場合を考えよう。$P(X > 100) = 0.1$ であることを考慮すると、X_1 が使われる確率は 0.1。同様にして、X_n が使われる確率は 10^{-n} である。すなわち、乱数を求める部分が計算時間の主要部分と仮定すると、前節のプログラムの実行時間は、単純に X を求める時間を 1 とすれば、例え再帰の深さに制限を設けないでも

$$1 + 0.1 + 0.001 + 0.001 + \cdots = 1.111\ldots$$

と、約 11%増加するだけである。

2 つの 15 ビット乱数を、おのおの上位及び下位 15 ビットとして 30 ビット乱数として用いるという工夫（正規分布、安定分布に従う擬似乱数として良く用いられている方法は本質的にはこれである）はこのオーバーヘッドが 100% である事と比較すれば、この方法の利点は明らかであろう。

4 計算機実験

4.1 組み込み乱数の問題点

前節の乱数生成法では、確率変数列 $X, X_1, X_2, \cdots$ の独立性が仮定されていた。しかし、2.2 で述べたように、組み込み乱数は線形合同法 $x_{x+1} \equiv a \times x_n + b$, $\bmod (2^{31})$ を用いて発生されている。上位 15 ビット

を採用しているといっても、これでは生成された乱数は独立な系であるとはいえない。

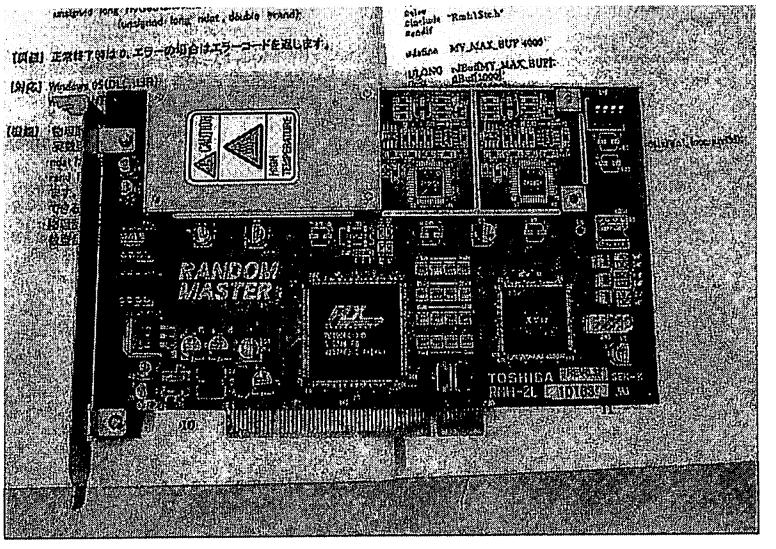
実際、2個の独立な確率変数を用いて安定型確率変数を作るような場合、隣り合わせた乱数ではなく、いくつかの乱数を読み捨てていくつか先の値を用いている。しかも、いくつ読み捨てるかが Know How となっていることは、計算機シミュレーションでははは常識化している。このことは、計算機で乱数を生成させようとする限りどんな擬似乱数についても大なり小なりいえることである。

4.2 物理乱数ボード

前節の理論を実験してみるに当たって、乱数列の独立性、周期性の影響をさけるため、レファレンスとして内蔵型の物理乱数ボードを使用した。これは、半導体の熱雑音を利用して乱数を得るものである。乱数としては 15, 16 ビット、31, 32 ビットの整数型、さらに浮動小数型もあるが、ここでは 15 ビット及び 31 ビットの整数型の乱数発生器として用いている。内部構造が完全には明らかにされていないので、空間及び時間に関する一様性には（実のところ）不安が残るが、検定の結果このボードにより得られた物理乱数には **"一様性がなくはない"** であろうと考えている。独立性については、マクロな意味で独立性を破るメカニズムが考えにくいという消極的な理由ではあるが、（独立になり得ないという積極的な理由のあるプログラムによる擬似乱数と比べて）より独立性が大きいと考えている。

再現性という観点で物理乱数を嫌う議論もあるが、デバックの段階では再現性のある擬似乱数、デバックのすんだ段階で、乱数発生部分を物理乱数という手段で充分にカバーできるだろう。

最近、通信のセキュリティの為の信頼できる使い捨て乱数発生器として、1チップの熱雑音利用の乱数発生器が開発されたことを述べておく。今後、CPU の一部として、熱雑音発生器が組み込まれる可能性もあると考えている。



4.3 実験結果

密度関数 $f(x) = (\beta - 1)\frac{1}{x^\beta}$ において、$\beta = 1.5$ としておく。また、**3.3** の正数を $C = 100$ とする。1回の実験について 10^7 個の乱数を発生させ、10^7 個の乱数の平均を求める実験を行った。物理乱数の 31 ビット、物理乱数の 15 ビット、組み込み乱数（15 ビット）について 10^7 個の乱数の平均を求める実験をそれぞれ 65 回ずつ行った。また、実験で求めた平均値 65 回分の平均と分散についても求めた。theory とは、理論平均を示し、アイテレーションなしのときは平均を上から抑え、アイテレーションあり（**3.4** のプログラムでの N の値を $N = 49$ としたとき）のときは、理論的に平均を下から抑えた値である。

	物理乱数 31 ビット		物理乱数 15 ビット		組み込み乱数 15 ビット	
iteration	なし	あり	なし	あり	なし	あり
1	1.288E+08	8.418E+08	5.406E+04	1.071E+07	5.204E+04	2.489E+06
2	4.031E+07	4.566E+06	5.420E+04	2.517E+09	5.546E+04	2.921E+07
3	9.229E+07	5.431E+07	5.347E+04	1.792E+09	5.311E+04	1.785E+07
4	2.606E+07	7.444E+07	5.369E+04	2.501E+07	5.388E+04	5.246E+07
5	5.162E+06	3.226E+08	4.895E+04	4.787E+06	5.470E+04	3.059E+08
6	5.878E+06	8.287E+06	5.559E+04	4.405E+07	5.309E+04	2.768E+07
7	5.212E+07	3.339E+10	5.703E+04	7.034E+08	5.340E+04	1.045E+08
8	1.475E+07	1.227E+07	5.270E+04	7.148E+07	5.184E+04	1.404E+11
9	2.367E+09	4.752E+07	5.532E+04	1.526E+07	5.513E+04	1.699E+08
10	7.820E+06	4.828E+08	5.581E+04	1.611E+07	5.857E+04	9.578E+07
11	3.621E+07	6.969E+06	4.970E+04	1.964E+08	5.121E+04	2.196E+07
12	4.793E+07	4.435E+07	5.333E+04	4.418E+10	5.127E+04	3.108E+07
13	1.530E+09	6.653E+06	5.192E+04	6.902E+07	5.472E+04	1.403E+07
14	1.063E+08	8.267E+07	5.405E+04	3.847E+07	5.147E+04	3.745E+08
15	3.359E+07	9.738E+10	5.304E+04	8.688E+06	5.572E+04	2.416E+09
16	1.354E+08	3.017E+06	5.286E+04	4.646E+07	5.320E+04	4.137E+07
17	3.306E+07	2.821E+07	5.651E+04	5.332E+07	5.609E+04	1.354E+07
18	1.349E+08	5.342E+07	5.493E+04	4.598E+07	5.018E+04	8.569E+06
19	9.531E+06	5.160E+07	5.293E+04	1.557E+08	5.170E+04	5.885E+06
20	8.747E+07	1.392E+08	5.059E+04	1.707E+09	5.216E+04	2.965E+07
21	8.235E+07	1.986E+07	5.304E+04	7.261E+06	5.554E+04	9.667E+06
22	8.490E+06	9.414E+06	5.385E+04	1.208E+08	4.852E+04	7.707E+07
23	1.381E+07	1.230E+07	5.408E+04	6.318E+06	5.236E+04	3.446E+07
24	2.068E+07	3.966E+07	5.349E+04	4.675E+08	5.139E+04	2.955E+08
25	9.808E+06	4.466E+08	5.529E+04	4.460E+06	5.246E+04	7.320E+07
26	1.310E+10	7.305E+08	5.538E+04	1.699E+07	5.783E+04	7.707E+07
27	1.924E+08	9.488E+07	5.439E+04	3.521E+08	5.205E+04	1.522E+08
28	1.373E+07	2.452E+07	5.521E+04	1.700E+08	5.482E+04	1.223E+08
29	2.104E+07	5.639E+07	5.183E+04	6.402E+06	5.463E+04	7.751E+08
30	5.496E+07	3.401E+08	5.395E+04	2.081E+07	5.199E+04	2.081E+06
31	7.839E+08	3.956E+07	5.504E+04	2.842E+07	5.298E+04	4.322E+09
32	5.663E+06	8.927E+06	5.787E+04	1.345E+07	5.469E+04	9.621E+06
33	1.613E+08	1.650E+08	5.208E+04	4.198E+07	5.659E+04	4.328E+09
34	1.093E+07	1.969E+07	5.490E+04	5.900E+06	5.475E+04	4.932E+07
35	1.316E+09	1.492E+07	5.023E+04	2.491E+07	5.468E+04	1.367E+07
36	9.026E+06	3.504E+08	5.448E+04	1.796E+07	5.430E+04	5.512E+07
37	2.539E+08	1.318E+07	5.346E+04	9.296E+07	5.856E+04	8.091E+07
38	2.592E+07	4.375E+07	5.492E+04	2.479E+07	5.488E+04	1.651E+07
39	3.911E+06	7.622E+07	5.416E+04	5.007E+07	5.101E+04	6.311E+06
40	2.256E+06	3.145E+08	5.394E+04	1.707E+08	5.411E+04	7.025E+06

	物理乱数 31 ビット		物理乱数 15 ビット		組み込み乱数 15 ビット	
iteration	なし	あり	なし	あり	なし	あり
41	1.690E+07	3.909E+08	5.589E+04	6.547E+07	5.091E+04	4.473E+08
42	3.569E+07	1.204E+07	5.392E+04	5.785E+08	5.276E+04	1.516E+08
43	1.629E+07	2.703E+07	5.332E+04	5.130E+07	5.773E+04	2.617E+07
44	1.045E+08	1.047E+08	5.444E+04	3.310E+07	5.259E+04	5.054E+06
45	1.278E+07	3.092E+08	5.547E+04	4.305E+08	5.436E+04	2.759E+07
46	5.180E+07	6.152E+07	5.250E+04	2.658E+07	5.205E+04	1.006E+08
47	1.603E+07	6.250E+08	5.432E+04	1.056E+08	5.122E+04	3.149E+06
48	2.576E+07	3.507E+07	5.620E+04	4.888E+06	5.360E+04	1.521E+07
49	8.835E+06	3.157E+08	5.415E+04	4.965E+07	5.721E+04	7.045E+08
50	8.172E+07	1.108E+10	5.287E+04	1.095E+07	5.653E+04	1.415E+07
51	1.012E+07	3.487E+07	5.375E+04	1.419E+07	5.421E+04	1.290E+07
52	1.041E+08	1.403E+08	5.435E+04	3.070E+06	5.227E+04	6.949E+06
53	7.574E+07	8.545E+07	5.385E+04	9.768E+06	5.485E+04	7.425E+07
54	3.376E+07	6.698E+07	5.575E+04	3.732E+07	5.630E+04	2.185E+07
55	3.540E+06	2.087E+07	5.349E+04	7.546E+06	5.069E+04	1.382E+08
56	2.428E+07	1.380E+08	5.670E+04	9.980E+06	5.080E+04	8.834E+07
57	2.730E+06	4.358E+07	5.417E+04	5.192E+06	5.664E+04	7.778E+06
58	4.413E+07	9.366E+07	5.535E+04	7.622E+06	5.375E+04	1.829E+07
59	1.226E+07	9.018E+09	5.336E+04	1.485E+08	5.062E+04	8.555E+06
60	5.574E+06	5.067E+06	5.435E+04	2.159E+08	5.239E+04	1.325E+07
61	2.211E+07	1.242E+07	5.620E+04	4.996E+07	5.551E+04	1.720E+08
62	5.266E+07	1.242E+08	5.227E+04	7.956E+06	5.096E+04	3.142E+08
63	3.009E+06	2.321E+07	5.501E+04	2.007E+08	5.423E+04	1.480E+08
64	1.085E+09	5.999E+09	5.431E+04	8.119E+06	5.234E+04	1.382E+08
65	1.828E+08	1.288E+08	5.524E+04	3.347E+07	5.417E+04	5.635E+06
mean	3.541E+08	2.535E+09	5.405E+04	8.533E+08	5.363E+04	2.421E+09
variance	5.556E+08	4.437E+09	1.206E+03	1.440E+09	1.818E+03	4.363E+09
theory	4.295E+09	2.147E+58	6.554E+04	3.277E+53	6.554E+04	3.277E+53

続いて、1 回の実験について 2×10^9 個の乱数を発生させ、2×10^9 個の乱数の平均を求める実験を行った。物理乱数の 31 ビット、物理乱数の 15 ビット、組み込み乱数（15 ビット）について 2×10^9 個の乱数の平均を求める実験をそれぞれ 3 回ずつ行った。

	物理乱数 31 ビット		物理乱数 15 ビット		組み込み乱数 15 ビット	
iteration	なし	あり	なし	あり	なし	あり
1	6.074E+08	1.152E+10	5.395E+04	3.652E+09	5.393E+04	3.672E+10
2	3.187E+09	5.698E+09	5.399E+04	7.948E+08	5.392E+04	3.675E+10
3	5.266E+09	9.120E+09	5.396E+04	7.544E+09	5.392E+04	3.806E+10

5 結論

5.1 最後に

実験結果でもわかるように、使う乱数の数が周期 $2^{31} = 2 \times 10^9$ よりも十分小さいときは、15 ビットの組み込み擬似乱数 + 再帰が、結構良い結果を出してる。また、31 ビット乱数であれば、再帰なしでも良い結果を出している。すなわち、ここで見る限りでは、2 個の独立な 15 ビット乱数を用いる（よく使われている）擬似乱数は、必要な乱数の数が小さい限り（すなわち周期が無視できる 10^8 程度以下）限り、平均に関しては十分大きな値を出しているといえる。

5.2 今後

ここでは、単に平均の大きさのみを問題としているが、実際には多くの問題が残っている。

(1) 平均が無限となる母集団から取ったサンプルの平均とは何を意味するのか。
(2) そのサンプル数に関する挙動は。
(3) 裾の重さ β は、検定可能か。（理論と実際）
(4) 生成した現実の乱数の（理論）平均値はあの程度

でいいのか。必要があれば構造型のデータを使えばもっと大きな数（$100^{2^{32}}$）でも扱えるようにできるがそこまで必要か。（出現頻度が極端に小さくなる）今のままでも、$N = 49$ となる確率は 10^{49} であり、実際の計算機のクロック（10^9）と比べればわかるがとても起きそうもない。

5.3 最後に

この論文は、第一著者の林聡美が、修士論文用の研究として、第二著者竹中の指導下に行った研究の要約である。 内容は、2003 年 9 月にチェコ共和国プラハ市の科学アカデミィでの国際研究集会および、2003 年 10 月の統計数理研究所の共同研究集会で著者 2 人の共同研究として発表されている。

参考文献

[1] Feller,W.: An Introduction to Probability Theory and its Applications, Vol.2,2nd ed. ,Wiley (1971),New York.

[2] Samorodnitsky,G and Taqqu,M.S.: Stable Non-Gaussian Random Processes, Chapman & Hall (1994), New York

[3] 高嶋 恵三:擬似乱数と Random walk 検定, 神戸大学理学部数学教室、2002 年 2 月

Generation of Random Numbers with Heavy Taniled Distribution

Satomi Hayashi and Shigeo Takenaka *

Graduate school of Science

** Department of Applied Mathematics, Facalty of Science*

Okayama University of Science

Ridai-cho 1-1, Okayama 700-0005, Japan

(Received March 25, 2004; accepted November 5, 2004)

Recent 10 years, many authors start to use stable random variables for their stochastic models because stable distribution has a heavy tail comparing the Gaussian distribution.There exeist many phenomena having heavy tailed relation in biology, physics etc.In some of their works computer simulations play important rolls. There happens a natural question. "Can computer generate heavy tailed random numbers." We are interseted in Generation of Random Numbers with heavy tailed distribution.