# 孫子の定理による
# 乗算合同法乱数の縮減計算

**中澤直也**[1] / **中澤 宏**[1]

**(July 1, 2024)**

ここでは『孫子の定理』による魔術的計算減縮』とでも名付けるべき事柄を議論します。孫子は古代中国の南北朝時代、5 から 6 世紀頃に互いに素な複数の法を伴う計算を考え、定理を与えたそうです。そんなに難しい証明ではないと思うものの、考え込むと矢張り『魔術か』とも思われてなりません。元々の定理は算術的な装いですが、この 20-21 世紀にもなってそれまで考えられた事もなかったであろう巨大な 2 つ以上の部分法の役割に基づいて『乗算合同法乱数生成問題』に明快な解決を与えている事に気付きました。ここでは『乱数問題』の状況を説明した後、元々の孫子の定理の説明、そのより応用数値計算に適した形の導出を与えます、最適な推論方法だといいのですが。『どう議論したらよいかが分った』と感じて頂ければ私達の喜びです。

## 1. 乗算合同 (MC) 法乱数生成機構

この論文では乗算合同 (multiplicative congruential, MC) 法乱数生成機構を、特徴的な孫子の定理の構造と関連して議論します。MC 生成機構は整数の組 $(d, z, n)$、正の『法』と呼ばれる $d > 0$、正で $d$ とは素な (共通素因数を持たない)『乗数』の $z$、そして正で再び $d$ とは素な『種 seed、或いは初期値』$n > 0$、の三つ組みで構成されます。これらは整数の列 $\{X_0, X_1, X_2, \cdots\}$ を区間 $(0, d)$ に次の再帰的な方法で生成します:

$$X_0 = \mathrm{mod}(n,\, d), \quad X_{k+1} = \mathrm{mod}(zX_k,\, d), \quad k = 0, 1, 2, \cdots.$$

この手続きは次の様に表す事も出来ます:

$$\{X_k = \mathrm{mod}(nz^k,\, d) |\ \ k = 0, 1, 2, \cdots\}.$$

[1]©枚方乱数工房 (HRF): 〒 573-0081 枚方市釈尊寺町 28-18-103.

上の関数 $\mathrm{mod}(A,\, d) = R$ は様々な計算プログラムでは多用されるものです。その定義は幾分『数学関数』としてはスマートではない部分があって、理論ではより明快な『法 $d$ の合同関係』に置き換えられる、というか代用される事も多くあります。しかし数値計算に用いるには非負の整数変数 $A$ に唯一つの整数値を対応させる関数でなければ困る、その様なものでなければならないので、次の定義を用います:

**定義 1.** 非負の任意整数 $A$ に対して、整数 $d > 0$ で割った整数の商を $Q \geq 0$、余りを $R \geq 0$ の等式

$$R = A - Qd$$

できまる $0 \leq R < d$ を $A$ に対する関数として、それを

$$R = \mathrm{mod}(A,\, d) = A - Qd, \quad 0 \leq \mathrm{mod}(A,\, d) < d$$

と定義する。 **(定義 1 終り)**

## 2. MC 一様乱数

(有理数の)MC 一様乱数 $\{V_k|\ k = 0, 1, 2, \cdots\}$ は開集合 $(0, 1)$ 内の数列

$$\{V_k = X_k/d|\ \ k = 0, 1, 2, \cdots, \quad 0 < V_k < 1\}$$

です。これらは一様に分布した (望むらくは統計的に独立な) 乱数の『見本列』です。列はシミュレーションやゲームに使用して不足しないように十分長くなければなりません。以下の議論を素早く滑らかに進めるために、まず術語を準備します。2 つの整数 $A$ と $B$ は次の式

$$A - B = \alpha d, \quad \text{or} \quad B = A + \beta d$$

が成り立つ時『$d$ を法として互いに同値』と言います。上や以下ではギリシャ文字 $\alpha, \beta, \cdots$ はそれぞれある整数を表すと約束します。関数 $\mathrm{mod}(A,\, d)$ は既に**定義 1** で不明確なく区間 $[0, d)$ に定義されました。幾分この定義を不細工と感じられるかも知れませんが、関数 $\mathrm{mod}(A,\, d)$ を我々は数値計算で省く事はできないので御了承下さい。

乗算合同 MC 法一様独立乱数の生成の実相の観察に移ります。MC 法一様乱数生成機構 $(d, z, n)$ を設計する際には得られる乱数が統計的な一様性と独立性の姿を再現する様にあらゆる努力を払わねばなりません。

この問題は MC 乱数生成機構の設計の根本であり、その実現のためには MC 法の最大の特徴、相続く $l$ 個の乱数が必ず $l$ 次元格子を作る事、その格子が理想とすべき形は格子の単位胞 unit cell が幾何学的な正単体である正格子である事、が我々の困難な努力の結果確立しています。その様な理想に近い幾何学格子を実現するものとして既に HRF は

2018 年に『#001』の名で実用周期 $2^{52}$ を持つもの

2020 年に『#003』の名で実用周期 $2^{51}$ をもつもの

を発見し、優れた 2 つの MC 乱数生成機構を与えています。

この『検定』の問題にここでは立ち入りません。全周期に亘る検定で良好な統計性が保障された MC 乱数をシミュレーションやゲームに用いるのは当然です。そこには科学技術者、ゲームのクリエータ達に立ち入って頂く必要はありません。

残る問題の第一は『(実用) 周期の長さ』の選択です。実用周期 T とは、素数 $p = d$ の法と原始根乗数 z の場合が分り易いのですが、フェルマーの小定理 $\mathrm{mod}(z^{p-1},\, p) = 1$ が成り立ちますから、列の前半最後は

$$\mathrm{mod}(z^{(p-1)/2},\, p) = p - 1, \quad \text{2 乗して 1 になる}$$

数です。つまり $-1$ と法 $p$ で合同であるため、それからあと後半の乗数 $z$ の冪は前半列にマイナスを掛けただけのもので『独立ではない』のです。独立乱数として使えるのは MC 乱数列の前半だけ、これが『実用 (になる) 周期』です。

MC 乱数列の構造に少し容易ならぬものを感得して下さい。さらに D. E. Knuth 先生の浩瀚な教科書 *The Art of Computer Programming* には大切な注意があります。シミュレーションに用いる (MC) 乱数は『再現可能 reproducible』であり、かつ『移転可能 transportable』でなければなりません。実数のような終りの方の (例えば 10 桁目、20 桁目の) 数はまあまあではなく、何度生成しても全桁がピシャリと一致しなければなりません、これが『reproducibility 再現可能性』です。コンピュータや計算言語を別のものに移しても全く同じ乱数が再現されなければない、これが『transportability 移植可能性』です。なにしろ乱数の生成回数は膨大ですから、変な事がある乱数のたとえ桁の途中、終りの方でも、に入る様ではそれ以後の発展は全く別物になります。乱数というものの世界の恐ろしさ、劇薬に過ぎるかも知れませんが、整数演算と本質的に結ばれている世界の恐ろしさを御理解下さい。

## 3. MC 一様独立乱数の周期と計算時間

この小文の特別な関心は MC 乱数の計算に要する計算時間にあります、 20 世紀乱数のコンピュータの展開はメルセンヌ素数 $d = 2^{31} - 1$ を法、その原始根 $z$ を乗数とする MC 乱数から始まりました。余り認識されていない事柄なので、代表的な MC 乱数についてその周期などを纏めておきます。周期は用い得る周期であり、『減縮』とは孫子の定理による減縮で 100,000 個を計算する時間を示します。

| 生成機構 | 法 | 周期 | CPU 時間 | 縮減 | 使える時間 |
|---|---|---|---|---|---|
| 原始根 | $p = 2^{31} - 1$ | $T = 2^{30} - 1$ | $0.09sec$ | | 9 秒 |
| #M001 | $p \approx 2^{34}$ | $T \approx 2^{33.00}$ | $2.5sec$ | | 数分 |
| #003 | $d \approx 2^{54}$ | $T \approx 2^{51}$ | $2.5sec$ | $0.25sec$ | 1.8 年 |
| #001 | $d \approx 2^{54}$ | $T \approx 2^{52}$ | $2.5sec$ | $0.25sec$ | 3.6 年 |

なお、メルセンヌ素数 $2^{31} - 1$ とその原始根による成機機構は、全周期を生成するのに現在の中級程度の卓上型コンピュータで 9sec 程度しか要しない短さで、シミュレーションやゲームには適しません。

上で『縮減』と記した項目は、下で説明する孫子の定理の部分法に基づく部分 MC 乱数生成機構を利用した計算時間で、法 $d$ を用いて 4 倍精度実数に変換して行った計算時間を $1/10$ に短縮します。明快な孫子に定理に基づき、MC 生成方式の原理に依拠する乱数全周期に亘る検定基準で、与えられるごく単純明快な計算式方式はどの様なシミュレーションの目的、ゲームであろうと、強く薦められる計算方式です。

上の最後の 3 つの MC 生成機構のうち#M001 は単精度整数の規格を越える素数 $p = 17179869989 > 2^{32}$ を法とし、そのある原始根を乗数とするものです。MC 法の漸化式計算では、整数 $\{X_k | \, k = 0, 1, 2, \cdots\}$ は倍精度整数として扱い、

$$X_{k+1} = \mathrm{mod}(zX_k, \, p)$$

の扱いでは、我々は『再現性』を尊重しなければならないから、4 倍精度実数への変換と演算そして倍精度整数への逆変換、を必要とします。この、現在では余り大きいとは言えない素数 $p$ の法でその労苦を経ても実現される (実用) 周期は $2^{33}$ に過ぎず、計算プログラムは単純明快です

が数分で全周期を使い切ります。孫子の定理による減縮が可能な#003 や #001 とは比べものになりません。この経験で我々に言える事は、2 つ以上の部分法を持つ孫子縮減が可能な MC 乱数こそ本命であって、シミュレーションやゲームに使用すべきだとの展望を強く述べます。とにかく現在では孫子の定理の与える原理的簡単明瞭は越え難く、その優位は疑うべくもありません。

孫子の定理による計算時間の減縮の説明はこの小文の大目的です。結局の所、乱数全周期に亘る優れた幾何学的検定、『すべての乱数生成方法を遍く比較する検定』は MC 法乱数にしか可能ではありません。そこで働く孫子の定理の重要性に思いを致して下さい。

## 4. 孫子の定理 I.

考え込まなければ、孫子の定理の基本形は単純明快です。

**孫子の定理 I.** 整数の法 $d > 0$ はすべて異なる正の素数 $p_1, p_2, \cdots, p_m$ の積であると仮定する:

$$d = p_1 p_2 \cdots p_m.$$

任意の負ではない整数 $A < d$ に対して $m$ 個の整数の組 $\{A_1, A_2, \cdots, A_m\}$,

$$A_1 = \mathrm{mod}(A,\, p_1),\ A_2 = \mathrm{mod}(A,\, p_2),\ \cdots,\ A_m = \mathrm{mod}(A,\, p_m) \quad (*)$$

は一意に定まり、逆に $m$ 個の整数の任意の組

$$\{A_k |\ 0 \le A_k < p_k, \quad 1 \le k \le m\} \quad (**)$$

は $d = p_1 p_2 \cdots p_m$ 未満の非負整数 $A$ を唯一つ定める。
**(証明)** 与えられた整数 $0 \le A < d$ から $(*)$ の整数の組 $\{A_1, A_2, \cdots, A_m\}$ は一意に定まります。仮に整数 $0 \le B < d$ があって $(*)$ と同じ値の組、

$$A_1 = \mathrm{mod}(B,\, p_1),\ A_2 = \mathrm{mod}(B,\, p_2),\ \cdots,\ A_m = \mathrm{mod}(B,\, p_m) \quad (\#)$$

を与えるなら、$(*)$ と $(\#)$ の差 $A - B$ は $p_1, p_2, \cdots, p_m$ それぞれを法として 0 であり、互いに異なる素な整数 $p_1, p_2, \cdots . p_m$ の倍数で、$d$ の倍数です。$A, B$ 共に区間 $[0, d)$ 内で定義されているから $A = B$ でなければなりません。 ■

## 5. 孫子の定理 II.

上の孫子の定理 I では、部分法での値 $A_k = \mathrm{mod}(A,\, p_k)$ が $1 \leq k \leq m$ で定められれば、もとの整数 $A$ は一意に決まると示されました。部分法での値の組から $A$ がどのように決まるのか、それが孫子の定理には重要です。以下は通常の数学議論としてはスマートではありませんが、円滑な議論には欠かせません。

殆ど自明ですが、補題を準備します。

**補題.** $A, B$ は非負の整数、$e$ は正の整数の法とする。$A' = \mathrm{mod}(A,\, e)$, $B' = \mathrm{mod}(B,\, e)$ と記すと次が成り立つ:

**(1)** $\mathrm{mod}(A,\, e) = \mathrm{mod}(A',\, e)$.

**(2)** $\mathrm{mod}(A + B,\, e) = \mathrm{mod}(A' + B,\, e) = \mathrm{mod}(A' + B',\, e)$.

**(3)** $\mathrm{mod}(AB,\, e) = \mathrm{mod}(A'B,\, e) = \mathrm{mod}(A'B',\, e)$.

**(証明) (1)** $A' = \mathrm{mod}(A,\, e)$ だから次は mod 関数の定義から明らか:

$$A = \mathrm{mod}(A,\, e) + \alpha e = A' + \alpha e.$$

ここで $\alpha$ は『何らかの整数である』とだけ言えば十分です。
**(2)** 整数 $A = \mathrm{mod}(A,\, e) + \alpha e = A' + \alpha e$ がある整数 $\alpha$ で成り立ち、同様に整数 $B = \mathrm{mod}(B,\, e) + \beta e$ もある整数で成り立ちます。故に

$$\begin{aligned} \mathrm{mod}(A + B,\, e) &= \mathrm{mod}(A' + \alpha e + B' + \beta e,\, e) \\ &= \mathrm{mod}(A' + B', e) \end{aligned}$$

です。最後の mod 関数は非負の $A' + B'$ が $e$ より大きくなるかも知れず欠かせません。
**(2)** 次は明らかです:

$$\begin{aligned} \mathrm{mod}(AB, e) &= \mathrm{mod}(\{A' + \alpha e\}B,\, e) = \mathrm{mod}(\{A' + \alpha e\}\{B' + \beta e\},\, e) \\ &= \mathrm{mod}(\{A'B + \gamma e\},\, e) = \mathrm{mod}(A'B' + \delta e,\, e) \\ &= \mathrm{mod}(A'B,\, e) = \mathrm{mod}(A'B',\, e). \end{aligned}$$

ここで $\gamma, \delta$ はある整数です。最後の mod 関数は $A'B, A'B'$ が $e$ より大きくなるかも知れないので欠かせません。 ■

**孫子の定理 II.** $d = p_1 p_2 \cdots p_m$ が正の整数の法 $d$ の正で互いに異なる素数

の部分法 $\{p_1, p_2, \cdots, p_m\}$ による分解とする。次の整数の組を定義する:[2]

$$\{D_k = d/p_k |\ k = 1, 2, \cdots, m,$$
$$D_k{}^{-1} \text{ は法 } p_k \text{ での } D_k \text{ の逆数、} 1 \leq k \leq m\}.$$

また整数 $A$ からその法 $p_k$ での値

$$\{A_k = \mathrm{mod}(A,\, p_k) |\ 1 \leq k \leq m\}$$

を定義する。このとき次が成り立つ：

$$A = \mathrm{mod}(A_1 D_1 D_1{}^{-1} + A_2 D_2 D_2{}^{-1} + \cdots + A_m D_m D_m{}^{-1},\, d). \quad (\#)$$

**(証明)**、式 $(\#)$ の右辺を整数 $B$ と記します。素数 $p_k$ を取って $B$ の関数値 $\mathrm{mod}(B,\, p_k)$ 考えます。まず右辺全体は法 $d$ での値だから、

$$B = A_1 D_1 D_1{}^{-1} + A_2 D_2 D_2{}^{-1} + \cdots + A_m D_m D_m{}^{-1} + \alpha d$$

と、ある整数 $\alpha$ 倍の $d$ を加えるか引くかして等式で表す事ができます。法 $p_k$ での値では、この $\alpha d$ は消え、また $A_k D_k D_k{}^{-1}$ 以外の項はすべて $p_k$ を含むので消えます。残るのは

$$\mathrm{mod}(B,\, p_k) = \mathrm{mod}(A_k D_k D_k{}^{-1},\, p_k)$$

だけです。法 $p_k$ の mod 関数内で見る積は**補題 (2), (3)** によって

$$\begin{aligned}
\mathrm{mod}(B,\, p_k) &= \mathrm{mod}(\{A_k{}' + \alpha p_k\}(D_k D_k{}^{-1})' + \beta p_k\},\, p_k) \\
&= \mathrm{mod}(\{A_k + \gamma p_k\}\{D_k D_k{}^{-1} + \delta p_k\},\, p_k) \\
&= \mathrm{mod}(A_k,\, p_k) = \mathrm{mod}(\{A + \epsilon p_k\},\, p_k) \\
&= \mathrm{mod}(A,\, p_k) = A_k
\end{aligned}$$

と演算されます; $(D_k D_k{}^{-1})' = \mathrm{mod}(D_k D_k{}^{-1},\, p_k) = 1$ に注意。最終的には簡明な

$$\mathrm{mod}(B,\, p_k) = A_k$$

だけが残るのです。まるで魔法です。こうしてすべての $k = 1, 2, \cdots, m$ について $(*)$ 式の $\mathrm{mod}(B,\, p_k)$ は $A_k$ と一致する事がわかります。$A, B$ 共に区間 $[0, d)$ 内だから、**孫子の定理 I** によって $A = B$ と示されました。■

[2]具体的に下の $D_k{}^{-1}$ を求めるには整数 $X = 0, 1, \cdots, p_k - 1$ を取って $D_k$ に掛け合わせ、$\mathrm{mod}(XD_k,\, p_k) = 1$ となる $X$ を $D_k{}^{-1}$ とすればよい。

## 6. 孫子の定理による MC 乱数計算の減縮

5-6 世紀に数論的孫子の定理が述べられてから、20 世紀に MC 乱数の計算が考えられるまで、定理が計算技術に持つ重要な意味は考えられる機会がなかったと考えます。2021 年に中澤直也は互いに素な 2 つの正の部分法 $p_1, p_2$ の積である法 $d = p_1 p_2$ が、孫子の定理で保障される重要な MC 乱数の高速算出を与えている事に気付きました。それは前の英語報告 ea4e.pdf に述べられている通り、条件 $d = p_1 p_2$ は $2d$ が倍精度整数内である場合には、すべてに倍精度整数演算だけを用いて、再現可能性を保ち、4 倍精度実数の利用に比べて 10 倍も高速な MC 乱数計算を可能にします。上に得られた事は条件を緩めて、$m$ 個 $(m = 2.3, \cdots)$ の単精度素数の積の法 $d = p_1 p_2 \cdots p_m$ を用いても『再現可能性』を保ちながら MC 乱数を算出する事ができる、という事、巨大な法と伴われる巨大な周期とを必要とする 21 世紀の MC 乱数問題に直面して初めて、気付かれた新しい技術です。

$(d, z, n)$MC 乱数の生成を考えます。計算したいのは再現可能な $(d, z, n)$MC 乱数整数列 $\{X_0, X_1, X_2, \cdots\}$ で法が $d = p_1 p_2 \cdots p_m$ と異なる $m$ 個の正素数の積である場合です。具体的にはまず整数列

$$\{X_j = \mathrm{mod}(nz^j,\, d) |\; j = 0, 1, 2, \cdots\}$$

と、その関数 $\mathrm{mod}(X_j,\, p_k)$ を算出しなければなりません; **孫子の定理 II** の $(*)$ 式参照。それは

$$\begin{aligned} \mathrm{mod}(X_j,\, p_k) &= \mathrm{mod}(\{\mathrm{mod}(nz^j,\, d)\},\, p_k) \\ &= \mathrm{mod}(\{nz^j + \alpha d\},\, p_k) \\ &= \mathrm{mod}(\{\mathrm{mod}(n,\, p_k) + \beta p_k\}\{z + \gamma p_k\}^j,\, p_k) \\ &= \mathrm{mod}(n_k {z_k}^j,\, p_k) \end{aligned}$$

という事になります。ここで

$$n_k = \mathrm{mod}(n,\, p_k), \quad z_k = \mathrm{mod}(z,\, p_k)$$

です。ややこしい。しかし結果 $(\#)$ は綺麗です。法 $d$ が互いに素な (或いは異なる) 正奇素数 $m$ 個の積 $d = p_1 p_2 \cdots p_m$ であれば,$(d, z, n)$MC 乱数は今や外側の $\mathrm{mod}(\cdots,\, d)$ から容易に外す事ができます; 整数 $A \geq 0$ に対しある整数 $\alpha$ による表現 $\mathrm{mod}(A,\, d) = A + \alpha d$ を $A = \mathrm{mod}(nz^j,\, d)$ として用いれば、次の結果が得られるのです。

**定理 III. 孫子の定理による MC 乱数計算の減縮**

整数の MC$(d, z, n)$ 乱数で法 $d$ が $m$ 個の部分法、異なる正の奇素数、の積 $d = p_1 p_2 \cdots p_m$ であるとする。このとき任意の番号 $j = 0, 1, 2, \cdots$ について

$$X_j \ = \mathrm{mod}(\{n_1 {z_1}^j D_1 {D_1}^{-1} + n_2 {z_2}^j D_2 {D_2}^{-1} + \cdots + n_m {z_m}^j D_m {D_m}^{-1}\},\ d)$$

が成り立つ。言葉で言えば第 $j$ 整数乱数 $X_j$ は、部分法での部分 MC 整数乱数生成機構

$$\{(p_k, z_k, n_k) |\ k = 1, 2, \cdots, m, \quad z_k = \mathrm{mod}(z,\ p_k), \quad n_k = \mathrm{mod}(n,\ p_k)\}$$

のかき混ぜ shuffling の $\mathrm{mod}(\cdots, d)$ であり、その法 $d$ 内での線形結合係数は

$$\{D_k {D_k}^{-1} |\ k = 1, 2, \cdots, m\}$$

である。 **(定理 III 終り)**

この shuffling 構造が中澤直也の発見です。美しいと思います。

実際に MC 生成機構#001 と#003 で孫子の定理による CPU 時間の縮減が 1/10 程度になる事を述べました。これらは $m = 2$ 個の単精度整数の大きさ限界近い部分法 $p_1, p_2$ を持ちます。$m = 3$ 個の部分法となると、計算を要する部分 MC 乱数生成機構が増しますから、計算時間も比例して増します。微妙な釣り合いの上に立つ状況ですが、$m = 3$ 個の MC 乱数生成機構が頂点なのかも知れません。いずれにせよ現在 $m = 2$ の生成機構は完成していますし、実現された実用周期 $2^{52}$ から $2^{53}$ は十分な長さです。読者の御活用は発明者の幸甚です。

## 付録. MC 乱数生成機構#001 の FORTRAN 計算プログラム

古い書式で恐縮ですが次のページに FORTRAN での MC 乱数生成機構#001 の MC プログラムを載せ、孫子減縮技術の要点を説明します。

**図 1. #001MC 乱数生成プログラム**

```
program main
implicit integer*8(i-n), real*8(a-h, o-z)
common ip1,ip2,id,ad,iz1,iz2,mz1,mz2,ip2mp1,ip1mp2
ip1=134265023
ip2=134475827
id=ip1*ip2                    ! id ≈ 2^54
ad=id
iz1=19061252
iz2=77600525
n1=10                         !  seed1 is 10
n2=13                         !  seed2 is 13
ip2mp1=52577007
ip1mp2=81816271
mz1=mod(n1,ip1)
mz2=mod(n2,ip2)
do i=1,10000000
call random(rand)
end do
end

subroutine random(rand)
implicit integer*8(i-n), real*8(a-h,o-z)
common ip1,ip2,id,ad,iz1,iz2,mz1,mz2,ip2mp1,ip1mp2
mz1=mod(mz1*iz1,ip1)
mz2=mod(mz2*iz2,ip2)
mz1a=mod(mz1*ip2mp1,ip1)
mz2a=mod(mz2*ip1mp2,ip2)
az=ip2*mz1a+ip1*mz2a
rand=az/ad
rand=mod(rand,1d0)
return
end
```

**(図 1 終り)**

上の $10^7$ 個の乱数出力に対する CPU 時間は 0.25 秒です。これは孫子の定理による部分生成機構への分解の威力で、これでも再現可能性は保障されています。分解を行わず 4 倍精度実数への代入を経る方法でも再現可能性は保障されますが、同数の出力にほぼ 10 倍の CPU 時間がかかります。

この#001 のプログラムでは上の部分法での生成の後も倍精度整数計算で進む事ができます。ここで新たに議論した可能性は、部分法整数生成機構の出力を次々に倍精度実数に代入して実数計算で乱数出力を得る方法で、部分法達は再現可能性を保ちながら、部分乱数の個数 $m$ には直ちには制限が生じません。但し計算速度は $m/2$ 倍になるので、実際に使える $m$ の値には上限がありますが、実際使用の自由度を広げている事は確かです。活用して頂ければ HRF としては喜びです。

コンピュータへの 4 倍精度実数規格の実装は少なくないと思われますので、同じ#001MC 乱数で同個数をそれで出力させるプログラムも載せます。

**図 2. 4 倍精度実数を用いる#001 計算 FORTRAN プログラム**

```
program main
implicit integer*8(i-p), real*8(a-h,r-z), real*16(q)
p1=134265023
p2=134475827
id=p1*p2                        !id=d は大体 2^54 です
qd=id
iz=7759097958782935d0
n=14899790517668688d0     !図 1 と同じ初期値です
qmz=n
qz=iz
do i=1, 10000000
rand=mod(qmz,qd)/qd
qmz=mod(qmz*qz,qd)
end do
end
```

**(図 2 終り)**

簡潔明瞭なプログラムです。上に述べた通り、CPU 時間は 10 倍程度にもなりますが、もし必要なら同じ乱数出力が得られるかを 2 つのプログラムで計算して比較する事もできます。御利用下さい。