系列現象の統計的解析—Ⅳ

モンテカルロ法へのリレー計算機の利用について

赤池弘次・三枝八重子

(1957年4月受付)

Statistical Analysis of Serial Phenomena—Ⅳ

Monte Carlo Methods Performed with Relay Computer

By Hirotugu AKAIKE and Yaeko SAIGUSA

Having used the FACOM-128 relay computer for some types of Monte Carlo works, we have seen that there are some fundamental rules for efficient programmings for these problems. We have analysed this fact from both sides of computer and problem. Some results of our analysis are given in this paper, and they are illustrated by some examples.

These examples contain programmings for generating quasi-random numbers, generating random numbers with negative exponential distribution, generatting random numbers with arbitrary given discrete and finite distribution, getting histogram from observed data, porker test and simulation of single server queue.

Institute of Statistical Mathematics

§1. 序及び謝辞

系列的な現象の統計的な解析は，先づその現象を適切に表現するような確率過程を求め，ついでその確率過程の各種の特性値間の関係をもとにして必要な数値に関する情報を求めることによつて行われる．しかし現実の問題は，多数の条件の組合せられたものが多いので，各々の条件の確率論的な表現は比較的簡単なものでも，これを全部組合せて得られる確率過程は多くの場合数式的な取扱いが極めて困難となる．そのような場合に特性値間の関係を求めたり，或は取上げた確率過程が実際によく現実の現象を表現しているかをたしかめたりするために確率過程の実現値を求める場合が多い．これは乱数をもちいれば，各種の条件の各々は比較的簡単に実現できるから，それの単純な組合せとして求められる．このように乱数による確率変数の実現値をもちいて，各種の推定の問題を処理する方法は一般にモンテカルロ法と呼ばれているが，特に現実の系列的な現象をそのまま表現する確率過程の実現値をとりあつかう場合には simulation method（相似法）と呼ばれることもある．

さて，一般にモンテカルロ法を利用する際には，単純な操作を極めて多数回繰返すことを要求される，これはひとつの実現値を得るために単純な操作を組合せることが必要なのと，推定の精度を上げるためには多数の実現値が要求されるためである．ところで系列的な現象をまねる場合には途中の誤りは次々の結果に影響を及ぼしてしまうので特に注意してさけなくてはならない．このよう

なことからモンテカルロ法を用いる際には，速度及び正確さの点から高速自動計算機によらなければならない場合が多い．たまたま統計数理研究所に設置された FACOM-128 継電器式万能自動計算機は通常の四則演算能力の他に各種の判別能力を備えているため特にモンテカルロ法に利用するに便利であると考えれる．われわれは各種の系列的な現象のモンテカルロ法による解析を FACOM-128 によつて実施したが．その中に現われたプログラミングのいくつかは一般的に用いられるものであり，更にこれらのプログラミングは計算機の機能に則して統一的に把握することができるものと考えられるのでそれを報告する．この統一的な見地は，一見極く自明のこととも考えられるが，無意識に行われている個々のプログラミングを通して，計算機の機能がはつきりその存在を現わしている点が興味深い．プログラミングの有効な定式化への一歩を試みるものとして報告したい．なお本告は参考文献[1]の知識を前提している．本報告作製に際し種々ご指導下さつた富士通信機関係の方々と第三研究部第二研究室の方々に厚くお礼申し上げたい．

§2. 判断について

モンテカルロ法を実施する際に通常の計算の場合と異つて特に計算機に要求される機能は判断の能力である．通常の数値計算に於ては判断が演算の終結，開始等の決定のために補助的に利用されるのに反してモンテカルロ法を実施する場合には判断が最も主要な動作となる．従つて以下この判断についてやや一般的に考察し，次いでその具体例を見ることにする．

判断は与えられた情報を一定の基準に従つて分類し，その分類結果に対応する命令を発することによつて完成する．計算機の役目は情報集団の要素と命令集団の要素との対応を実現することにあるわけで，この対応は通常上述の通り情報の分類という演算とその演算結果にもとづく命令の駆動という演算によつて実現する，このように考えるとプログラミングの立場から見るとき，命令集団の構造と情報集団の構造とを同時に考えながらプログラミングをすることが必要となることは明かであるが，命令集団の構造はこれを計算機の機能の側から見ると判別 (11) の結果によつて制御されるものと非正規化された数によつて制御されるものの二種類しかない．従つて具体的な問題が与えられた時，その命令集団はこの二種類の何れの型によつて表現されるかを考えることが先づ必要である．何れの型によつても表現される場合には，通常非正規化型の方がより有利なプログラミングを与えることが多いが，これは分類演算の型式に関係する．命令集団の構造がこのように決定されるとあとは分類演算だけが問題となる．分類演算は (0～10) の演算を用いる場合と，更に下位の判断として (11) によつて制御される比較的単純な演算の組合せを用いる場合とに分けられる．これは主として {(11), (12)}, {(11), (20)} 型である．分類を情報集団とその分類基準の側から見ると次のようになる，先づ命令集団は前述の二種類に限られるわけであるから，何れにせよこれを制御するためには情報は一次元の数値に変換されなくてはならない．この変換は (1→1) 変換だけではなく (多)→1) 変換が行われるわけでここに分類の必要が生ずるのである．ところでわれわれの考える情報は常に一次元若くは多次元の数値の組として表現されるわけであるから分類はこの空間を問題の規定する基準に従つてそれぞれの領域に分割し各領域に属する情報に対して一定の数値を与えることに帰着する．以下 (11,12). (11,20) 型演算を D型演算と呼ぶことにする．

ところで計算機の機能の側から見ると前述の通り FACOM-128 の分類能力は通常の演算 (0～10) に基くものと判別 (11) に基くものの二通りに分けられる．ここで一応 D 型演算 ((11, 12) (11, 20)) 以外の判断の複合によつて構成される分類を上位の判断として除外すると分類は演算 (0～10) と D 型演算によるものに限定される．前者による分類は与えられた情報の計量的な特性のみを利用する直接的な変数変換である．後者を用いる分類は，分類される数値と分類結果として対応させられる数値との間には何等計量的な関係を保つ必要はない．

更に D 型演算を用いて領域の分割を行う際には，与えられた情報がどの領域に属するかを決定

するために通常領域に一定の直線的な順序がつけられ，この順序に従つて順次与えられた情報がある領域に属するか否かを調べて行くという演算を行う場合が殆んどである．このような場合にはどの情報についても必ず行われる分類演算と情報毎に演算回数の変化する繰返し型の部分とを分離し繰返し型のためには独立したひとつのルーティンを設ける必要がある．中でも D 型演算の繰返し回数の記録をとる演算が頻繁に用いられる．以下この演算を counting と呼ぶことにするが筆者はこのために特に適していると考えられる演算を考案した．これについては次の §3 で述べる．

要約するに，分類は情報の計量的な性質を利用して (0～10) 演算で行う場合と D 型演算を用いる場合とに分けられ，D 型演算の繰返しを行う場合にはしばしば counting が有効であるといえる．勿論実際の問題はこのような単純な判断の複雑な組合せによつて解かれるわけである．

更に判断全般を通じて通常の演算に比して非正規化された数を取扱う必要が極めて大であることは命令集団の構造からも当然のことと考えられるが，特にモンテカルロ法へのリレー計算機の利用を考えるとき非正規化された数の取扱いについて十分な理解を持つことが必要と考えられる．このため各記憶装置の特性，(8, 9, 10) の特殊演算，及び (0～7) の通常の演算と非正規化された数との関係について十分な研究が必要である．筆者等もこの点については未だ極く一部の経験しか持たないがその有効さは次の § のいくつかの例でも十分に察することができるであろう．

§3. プログラミングの実例

1. 命令集団の型に注意すること:

［問題］ $x_i \leq 0$ のとき　x_i を 1 に加え
　　x_i^2 を 2 に加える
　　$x_i < 0$ のとき　x_i を 3 に加え
　　x_i^2 を 4 に加える．

註．以下 1, 2, 3, ‥‥等の数字は記憶装置の番号を示すものとする．

この例では分類は当然 x_i と 220 (常数 0) との大小判別による．従つてその判別結果によつて命令を制御すると考えると判別型の命令集団と考えやすい．しかしこれをよく見ると実は分類結果に対応する各命令は全く同一型で加算される記憶装置の番号が変化しているだけである．従つてこれはアドレス変更を利用すれば非正規化型の命令集団として扱える．この各々の見方に対応する実際のプログラミングは次の通りになる．

判別型 (x_i は 191 (数値テープ) にあるものとする)

1.	191	220	000	11	1	0	0	0	0	x_i と 0 (220) との第1大小判別
2.	210	210	000	03	1	0	0	0	0	210 (x_i-0) の2乗
3.	000	000	000	20	0	0	1	1	0	$x_i \geq 0$ なら読め，$x_i < 0$ ならジャンプ
4.	210	000	001	01	0	0	0	0	0	x_i-0 を 1 に加算
5.	211	000	002	01	0	0	0	0	0	x_i^2 を 2 に加算
6.	000	000	000	20	0	0	0	0	1	$x_i < 0$ なら読め
7.	210	000	003	01	0	0	0	0	0	x_i-0 を 3 に加算
8.	211	000	004	01	0	0	0	0	0	x_i^2 を 4 に加算
9.	000	000	000	20	0	0	1	1	1	常に読め

非正規化型 (数 -0.0000000×10^7 を 6, 数 -0.0002002×10^7 を 7 に入れておく)

1.	191	220	000	11	1	0	0	0	0	x_i と 0 との第 1 大小判別
2.	210	210	000	03	1	0	0	0	0	$(x_i-0)^2 \rightarrow 211$
3.	006	007	198	12	0	0	0	0	0	$x_i \geqq 0$ なら SAB = SAC = 0 $x_i < 0$ なら SAB = SAC = 2
4.	210	000	000	01	0	0	0	1	1	x_i を [SAB + 1] に加える
5.	211	000	000	01	0	0	0	1	1	x_i^2 を [SAB + 1] に加える

(20)（ルーティンジヤンプ）の速度は小であるから上のふたつのプログラミングでは当然非正規化型の方が遙かにすぐれている．この例はプログラミングに際しては先づ命令集団の構造を考察すべきであることを示している．

2. 多次元 ⟶ 一次元化の例:

［問題］ ポーカー検定．連続する四数 (4-digits) について (a, a, a, a) (a, a, a, b) (a, a, b, b) (a, a, b, c) (a, b, c, d) の型の分布を求める．

この問題を解くには連続する四数中に於ける数 0～9 の頻度を求めその二乗和をつくると，

$$\begin{aligned}
(a,\ a,\ a,\ a) &\longrightarrow & 4^2 &= 16\\
(a,\ a,\ a,\ b) &\longrightarrow & 3^2+1 &= 10\\
(a,\ a,\ b,\ b) &\longrightarrow & 2^2+2^2 &= 8\\
(a,\ a,\ b,\ c) &\longrightarrow & 2^2+1+1 &= 6\\
(a,\ b,\ c,\ d) &\longrightarrow & 1+1+1+1 &= 4
\end{aligned}$$

となつて一次元化される．この一次元化された数字の度数分布を求めればよい（ヒストグラムの項参照）．これは多次元 → 一次元 (0～10) 型分類のひとつの例としてあげた，

3. 擬似乱数の作製（特殊演算の利用）:

〔問題〕 congruent method によつて擬似乱数を作れ (congruent method とは適当な数 x_0 を出発値とし，定数 k を用いて $x_{n+1} \equiv k x_n \bmod \alpha$ として $0 \leq x_{n+1} < \alpha$ を逐次作り出す方法．ここでは $\alpha = 10^8$ とし，k は八桁以内の数とする)．

ここでは前述の文献〔1〕に述べられていない特殊乗算 (10) についてその性質を述べよう．

A 欄の内容が 1.2345678×10^a

B 欄の内容が 0.0010000×10^b

であつたとしよう．ここで

A	*B*	000	10	1	2	0	0	0

なる命令を出すと

211 の内容が $0.0001234 \times 10^{b-a+1}$

212 の内容が $5.6780000 \times 10^{b-a-7}$

となる．一般の A, B については指数部の法則はこの例の通りで，A と B との指数を無視した掛算を行い，その結果を 211 と 212 に並べて末尾が 212 の末尾に来るようにし，211 の頭の方の空間に零をつめ上述の指数をそれぞれ 211, 212 に与えるものと考えればよい．

ここで $x_n \times 10^0$ を 181, $k \times 10^7$ を 39 に入れておく．

1.	181	039	000	10	1	2	0	0	0	x_n と k とを特殊乗算
2.	212	000	191	00	00	0	0	0	0	下の 8 桁を打出す

これだけの命令の繰返しで上述の問題はとける．

4. 変数変換:

〔問題〕 一様分布をする乱数から，与えられたヒストグラムをその確率分布とするような乱数を作れ.

この問題は結局次の形の問題をとけばよい．分点 $(d_1, d_2 \cdots\cdots d_k)$ が与えられたとき $(d_i < d_{i+1}$ $i = 1 \cdots\cdots k-1)$ 一様分布に従う乱数 x_i に対し

$$x_i < d_1 \text{ なら } y_1$$

$$d_{h-1} \leq x_i < d_h \text{ なら } y_h \quad (h = 2 \cdots\cdots k)$$

を対応させ，この y_h を取出す．このためのプログラミングは d_i を i に y_i を $k+i$ に入れておき，x_i はテープ 191 にあるものとして次の通り.

$R-2$ (ルーティン 2)

1.	191	000	000	00	0	0	0	0	0	x_i を 0 に送る
2.	001	000	000	15	0	0	0	0	0	ルーティン 1 を呼ぶ

$R-1$ (ルーティン 1)

1.	000	000	000	11	1	0	1	0	0	d_i と x_i との大小判別
2.	002	001	001	17	0	0	0	0	0	$d_i > x_i$ なら $R-2$ $d_i \leq x_i$ なら $R-1$ 呼出し

$R-2$ (ルーティン つづき)

3.	k	000	191	00	0	0	2	0	0	$k+h$ を 191 に打出す
4.	000	000	000	13	1	0	0	0	0	SA を帰零する

実際にこのプログラミングを利用する際には更に速度を増すために細工が必要である．特に k が大となるとこのままでは使えない．この例は D 型の分類を用い，判別回数を SA に記録する counting の一種を示している．ここで更に筆者の考案した counting の例を示すために次の問題を考えよう.

5. 単位時間内の事象発生回数を求めること：

〔問題〕 相隣る事象間の時間長の記録 $\{t_i\}$ が与えられたとき，$((k-1)T, kT]$ 中に起きた事象の数 n_k の記録 $\{n_k;\ k = 1, 2, \cdots\cdots\}$ を作る.

$\{t_i\}$ がテープ 191 に，定数 $T(>0)$ が 183, 184 に，入つているものとし，220 を 182 に送つておく.

$R-1$

1.	191	182	182	01	0	0	0	0	0	$t_i + \sum t_i \rightarrow 182$
2.	183	182	000	11	1	0	0	0	1	kT と $\sum t_i$ との大小判別
3.	001	001	002	17	0	0	0	0	0	$kT \geq \sum t_i$ なら $R-1$ $kT < \sum t_i$ なら $R-2$ 呼出し

$R-1$ の第二ステップが counting である．このプログラミングを用いると $R-1$ を通つた回数が 197 (SAC) に記録される．このプログラミングを用いると，195, 196 が他の条件によつて使えない場合等にも演算に干渉することなく演算度数の記録が得られる．この点が筆者の着想であるがこれは1ステップでふたつの演算を行わせることに相当するわけで極めて応用範囲の広い有利なプログラミングである．このプログラミングは FACOM-128 で完全に用いられることが確認されている．ルーティンの回転数を特にそのためのステップを設けずに演算中のアドレス変更量の変化を利用して記録する方法を一般に counting と呼ぶことにすると前例もこの例もそれぞれひとつの

counting を用いていることになる．counting を用いる際には必ずそのためにひとつのルーティンを独立させなくてはならない．

$R-2$

1.	197	221	191	02	0	0	0	0	0	SAC—1 を正規化して 191 へ
2.	000	000	000	13	1	0	0	0	1	SAC を帰零
3.	184	183	183	01	0	0	0	0	0	$kT \rightarrow (k+1)T$
4.	183	182	000	11	1	0	0	0	1	$(k+1)T$ と $\sum t_i$ との大小判別
5.	001	001	002	17	0	0	0	0	0	$(k+1)T \geq \sum t_i$ なら $R-1$ 呼出し $(k+1)T < \sum t_i$ なら $R-2$ 呼出し

6. ヒストグラム:

〔問題〕 与えられたデータ $\{x_i\}$ のヒストグラムをつくれ．

ここではデータの範囲を適当な数の等間隔の区間に分割し，その区間内の度数を見ることにする．区間の巾を d とし，$1/d$ を 183 に入れておく．各 x_i は ≥ 0 としておく．$\{x_i\}$ はテープ 191 に入つているものとする．

1.	191	183	000	03	1	0	0	0	0	$x_i \times 1/d \rightarrow 211$
2.	219	211	196	08	0	1	0	0	0	211 を非正規化 → SAB
3.	196	000	197	00	0	0	0	0	0	SAB → SAC
4.	221	000	000	01	0	0	0	2	2	数 1 を [SAB] に加算

この繰返しでヒストグラムが求められる．この例では (0～10) 型の分類が利用されている．これを D 型で行うと全く実用にならない程速度が落ちてしまう．

7. Queue の実験:

〔問題〕 第 i 番目と第 $i+1$ 番目の input の間隔が t_i，第 i 番目の人のサービス時間が s_i のとき，第1番目のの人の到着からサービスが開始されるものとして窓口通過後の第 $i-1$ 番目と第 i 番目の人の間隔 d_i，第 i 番目の人の待ち時間 W_i，或は第 $i-1$ 番目の人がサービス終了後第 i 番目の人が到着するまでの窓口の手空き時間 r_i を求めよ．

これを解くために次の式を用いる．

$$r_i = \left(\sum_{j=1}^{i-1} t_j - \sum_{j=1}^{i-1} d_j\right) \vee O, \quad r_1 \equiv 0, \quad d_1 \equiv s_1$$

$$r_i - w_i = \sum_{j=1}^{i-1} t_j - \sum_{j=1}^{i-1} d_j$$

$$d_i = r_i + s_i$$

$\{t_i\}$ が 191 $\{s_i\}$ が 192 のテープに入つているものとする．$d_i (i \geq 2)$ を 191，$r_i - w_i (i \geq 2)$ を 192 に打出すためには次のプログラミングでよい．(t_i の桁数4桁程度として実用になろう．)

1.	210	191	000	01	1	0	0	0	0	$\sum t_i \rightarrow 210$
2.	210	001	192	11	0	0	0	0	0	$\sum t_i$ と $\sum d_i$ の大小判別をし $\sum t_i - \sum d_i \rightarrow 192$
3.	182	220	000	12	0	0	0	0	0	$(\sum t_i - \sum d_i) \vee O \rightarrow 0$
4.	000	192	191	01	0	0	0	0	0	$(\sum t_i - \sum d_i) \vee O + s_k = d_k \rightarrow 191$
5.	181	001	001	01	0	0	0	0	0	$\sum d_i + d_k \rightarrow 1$

この結果より，$\sum r_i$，$\sum r_i^2$，$\sum w_i$，$\sum w_i^2$ を求めるのには例1のプログラミングをつかえばよ

い．この例のプログラミングは簡単であるが速度大で実用性も大である．

6. 指数分布をする乱数（やや複雑な分類の例）：

〔問題〕 0～1 の間に一様分布をする乱数から e^{-x} を密度函数としてもつ乱数をつくれ．

このために J. von Neumann の考案になる次の結果を使う〔2〕．

$\{X_i\}$ を $[0,1]$ に一様分布をする独立な確率変数とする．このとき Y を次のように定義すると，Y は密度函数 e^{-x} を持つ確率変数である．

$$X_1 > X_2 > \cdots\cdots > X_{n_1-1} \qquad X_{n_1-1} \leq X_{n_1}$$

となつた時 n_1 が偶数ならば $Y = X_1$ とする．n_1 が奇数ならば，更に上と同様にして

$$X_{n_1+1} > X_{n_1+2} > \cdots\cdots > X_{n_1+n_2-1}, \qquad X_{n_1+n_2-1} \leq X_{n_1+n_2}$$

となつた時，n_2 が偶数ならば $Y = X_{n_1+1}$ とする．n_2 が奇数の時は更につづけて

$$X_{n_1+n_2+1} > X_{n_1+n_2+2} > \cdots\cdots > X_{n_1+n_2+n_3-1}, \qquad X_{n_1+n_2+n_3-1} \leq X_{n_1+n_2+n_3}$$

となつた時，n_3 が偶数ならば $Y = X_{n_1+n_2+1} + 2$ とする．n_3 が奇数のときは更に同様なことを繰返す．このようにして n_i を定義していくとき一般に

$$X_{\sum_{i=1}^{k} n_i+1} > X_{\sum_{i=1}^{k} n_i+2} > \cdots\cdots > X_{\sum_{i=1}^{k} n_i+n_{k+1}-1}, \qquad X_{\sum_{i=1}^{k} n_i+n_{k+1}-1,} \leq X_{\sum_{i=1}^{k+1} n_i}$$

となつて $n_1, \cdots\cdots n_k$ が奇数で n_{k+1} がはじめて偶数のとき $Y = X_{\sum_{i=1}^{k} n_i+1} + k$ とする*．

これを使つて次のようにプログラミングする．

一様分布をする乱数 $\{x_i\}$ が与えられたとしよう．上の分類の過程は次のような構造をしている．先づ $x_1 \leq x_2$ か $x_1 > x_2$ であるかによつての大分類が行われる．$x_1 > x_2$ の場合に限つて更に分類が進められるのであるが，これ以降の分類は同一形式の判別の繰返しとして表現される．即ち $x_2 > x_3$ なら更に x_3 と x_4 の大小判別を行い．$x_2 \leq x_3$ ならば加算量に増分1を加え，x_3 の代りに新たな x_4 をとり x_4 と x_5 の大小判別を行う．前の場合 $x_3 \leq x_4$，後の場合 $x_4 \leq x_5$ となつたら分類は終了してそれぞれ x_1, x_4+1 が打出される．また $x_3 > x_4$，或は $x_4 > x_5$ の場合には x_2 と x_3 の大小判別以後の過程を x_2 の代りにそれぞれ x_4，x_5 を，x_3 の代りにそれぞれ x_5，x_6 を用い等して繰返せばよい．従つてこのプログラミングは次の通りになる．$\{x_i\}$ は 191 のテープに入つているとする．加算量を0に，打出される候補者を1におくことにする．

		$R-1$								
1.	191	000	001	00	0	0	0	0	0	$x_i \to 1$
2.	191	000	002	00	0	0	0	0	0	$x_{i+1} \to 2$
3.	001	002	000	11	1	0	0	0	0	x_i と x_{i+1} の大小判別
4.	002	001	001	17	0	0	0	0	0	$x_i > x_{i+1} \to R-2$, $x_i \leq x_{i+1} \to R-1$
5.	000	001	191	01	0	0	0	0	0	$0+1 \to 191$
6.	220	000	000	00	0	0	0	0	0	0 を帰零

* $P_{\mathrm{rob}}(X_1 > X_2 > \cdots\cdots > X_{n-1}) = P_{\mathrm{rob}}(E_{n-1}) = \dfrac{1}{(n-1)!}$,

$P_{\mathrm{rob}}(x < X_1 < x+dx \mid E_{n-1}) = (n-1)x^{n-2}dx$ となることは最大値の分布より明か．

$\therefore\ P_{\mathrm{rob}}\{(x < X_1 < x+dx) \cap (E_{n-1} - E_n)\} = \left[\dfrac{x^{n-2}}{(n-2)!} - \dfrac{x^{n-1}}{(n-1)!}\right]dx$

$\therefore\ \sum_{m=1}^{\infty} P_{\mathrm{rob}}\{(x < X_1 < x+dx) \cap (E_{2m-1} - E_{2m})\} = e^{-x}dx \qquad (0 < x < 1)$

のこりの e^{-1} については X_{ni} の分布が $e^{-x}dx \qquad (0 < x < 1)$

従つて Y の分布としては $e^{-1}e^{-x}dx = e^{-y}dy \qquad (1 < y < 2)$

等となつて Y の密度函数は e^{-y} $(0 < y < +\infty)$ なることがわかる．

$R-2$

1.	191	000	003	00	0	0	0	0	0	$x_{i+2} \rightarrow 3$
2.	003	002	000	11	1	0	0	0	0	x_{i+2} と x_{i+1} の大小判別
3.	221	220	183	12	0	0	0	0	0	$x_{i+1} \leqq x_{i+2}$ なら 数 $1 \rightarrow 183$ $x_{i+1} > x_{i+2}$ なら 数 $0 \rightarrow 183$
4.	000	183	000	01	0	0	0	0	0	$0+183 \rightarrow 0$
5.	191	003	184	12	0	0	0	0	0	$x_{i+1} \leqq x_{i+2}$ なら $x_{i+3}' \rightarrow 184$ $x_{i+1} > x_{i+2}$ なら $x_{i+2} \rightarrow 184$
6.	184	000	003	00	0	0	0	0	0	$x_{i+3}' \rightarrow 3$ $x_{i+2} \rightarrow 3$
7.	003	001	182	12	0	0	0	0	0	$x_{i+1} \leqq x_{i+2}$ なら $x_{i+3}' \rightarrow 182$ $x_{i+1} > x_{i+2}$ なら $x_i \rightarrow 182$
8.	182	000	001	00	0	0	0	0	0	$x_{i+3}' \rightarrow 1$ $x_i \rightarrow 1$
9.	191	000	002	00	0	0	0	0	0	$x_{i+4}' \rightarrow 2$ $x_{i+3} \rightarrow 2$
10.	003	002	000	11	1	0	0	0	0	x_{i+3}' と x_{i+4}' x_{i+2} と x_{i+3} の大小判別
11.	002	001	001	17	0	0	0	0	0	$x_{i+3}' > x_{i+4}'$ $x_{i+2} > x_{i+3}$ なら $R-2$, 他は $R-1$

§4 結　　語

以上実例によつて §3 の考察を検討したのであるが，これらの実例は全く基本的なものであつて，実際に使用されるプログラミングは計算機の容量その他の制限のために更に複雑な考慮が要求される．しかしそれらもこのような基本的な判断の複合として簡単に解ける場合が多い．そしてそれはむしろ多ルーティン間の結合の問題になる．

ここにあらためて判断に関するプログラミングの基本的な規則と考えられるものをまとめて本報告を終りたい．

1. 命令集団の型に注意せよ〈判別型／非正規化型〉
2. 情報の分類法に注意せよ〈D型／(0～10)型〉
3. D型の不定回繰返しがある場合，繰返しのない部分と完全に分離してひとつの独立したルーティンを用いよ．

統計数理研究所

文　献

[1] 石井康雄：FACOM-128 について II. プログラミング. FUJI vol 6, No. 4, 1956.
[2] J. von Neumann: Various techniques used in connection with random digits. *Monte Carlo Method. National Bureau of Standards Applied Mathematics Series,* No. 12 (1951) P. 38.