# 卓上型電子計算機によるいくつかの計算例

## その1　配列をもつデータの処理

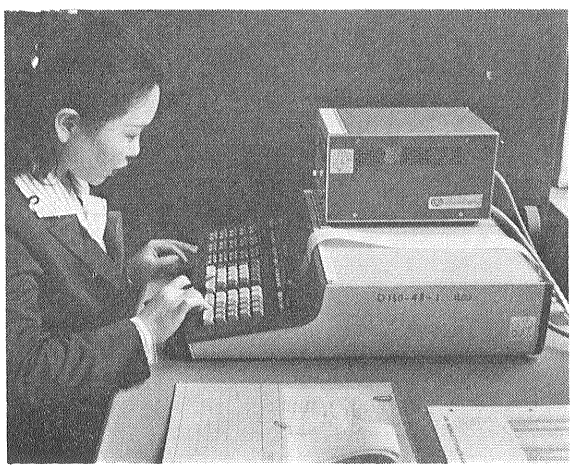
吉井守正（鉱　床　部）

### はじめに

プログラムが組めて　データに対してもある程度の記憶容量をもつ卓上型電子計算機が　最近盛んに使われるようになって来た．　これらの計算機は　いわゆるミニコンピュータとかマイクロコンピュータとか呼ばれる物とは異なり　いわゆる“電卓”や“電子そろばん”と言われる電子式の計算機が次第にグレードアップして出来上った物と考える事ができよう．　最近では記憶容量の点では　ミニコンなどに決して劣らないものもあるが　もともと“電卓出身”のこの種の計算機では　取り扱いなどの点でいろいろ簡便にできており　プログラミングなども容易なため　初心者も十分使いこなす事ができる．

筆者が所属する部にも　数年前に横河ヒューレット・パッカード20型（YHP—20）という計算機が導入され　それ以来いろいろの技術計算が行なわれ　その結果はすでにいくつかの論文や講演を通じて公表されて成果をおさめている．

そこで　この計算機などにたずさわった人が手分けして　自分らが組んだ計算プログラムや　それらに関係するノウハウについて順次御紹介して行きたい．　筆者らは決してプログラミングの専門家ではなく　筆者に至ってはまったくの“電卓育ち”であるから　幼稚な事を麗麗しく述べたてる事もあろうが　これらについては是非有益な御助言をたまわりたい．　計算プログラムを組む思想そのものは大型機も卓上機も大差ないと思われるが　卓上型計算機の場合　価格が安いという利点がある反面　おそらくそのためにハードウエア上に種々の制約があり　細部にわたっては　大部様子がちがって来る．　しかし一方　卓上型計算機は初心者に対する間口の広さをもちまた小回りのきく便利さは　大型機の比ではない．　とにかく　筆者らの体験談などもまじえながら　卓上型計算機のためのソフトウエアについて　気楽に話を進めよう．



第1図　卓上型電子計算機の1例
筆者らが使っているYHP—20計算機は6kw のメモリー容量をもつ．プログラムやデータはキーボードから入力され　それぞれ磁気カードまたは磁気テープにレコードされる．　計算機の上にのっている装置は　カセットテープレコーダ．

### 1. 二次元配列をもつデータの処理

卓上型計算機にはさまざまな種類があるが　それらのハイクラスのものを除くと配列宣言のできない機種が多い．　このために　配列をもつデータ　すなわち数表という視覚によく訴える並びをもつ一団の数値を　計算機のメモリーに直接割り付ける事ができないのである．筆者らが使っているYHP—20もこの部類に属する（最新型の YHP—25は配列が可能である）．　ところが筆者らの経験からすると　配列をもつ多量のデータを処理する計算に対する需要は大変多いのである．　そこで筆者は配列宣言のできない計算機でも　たとえば FORTRAN などで DIMENSION　宣言をしたのと同じように　メモリーの割り付けができるソフトウエアを考えついた．

原理は簡単である．　いま二次元配列が1個ある場合について考えてみよう．　その配列が$m$行$n$列の規模をもつとき　その$i$行$j$列目の要素に対応するメモリーの番地$y$は

$$y=(i-1)n+j+c \quad \cdots\cdots(1)$$

という関係式で求める事ができる．　これらの関係を第2図に示す．　ここに$c$は定数であり　具体的にはメモリーのもっとも若い番地から$c$番地までを　配列要素以外に使うためである．　筆者は$c=30$としている．　こ

の(1)式を実行するサブルーチンを作っておき　必要のつどこれを呼び出せば　その配列要素がもつ行番号$i$と列番号$j$から　それに対応するメモリーの番地を　いつでも自動的に求める事ができる．　このサブルーチンを配列用サブルーチンと呼ぶ事にするが　これを使う事によって使用者は　その配列の要素がメモリーの何番地に対応するかなどをまったく考えなくとも　プログラミングや　そのプログラムによる計算ができるのである．

配列の行数$m$　列数$n$などはシステムを通じて　一定の番地に入れる事にすれば　どのプログラムにも共通できる．　しかも$m$や$n$の値は配列をもつデータと一緒にレコードされるので　それ以後使用者はそのデータの配列規模についても　とくに記憶しておかなくともよい．

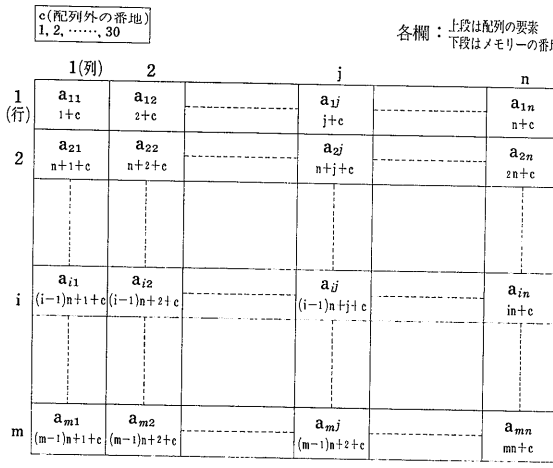
配列用サブルーチンを使ったプログラムの例として配列をもつデータを入力する場合を第３図で示す．

さて　(1)式に示すメモリーの番地は　配列の同じ行の１列目から２列目　３列目……の順に一連番号で増していく．　このような並びを　$Y$並びと呼ぶ事にしよう．これに対して　これと行・列の関係を逆した並び方　すなわち　配列要素に対応する番地$x$が

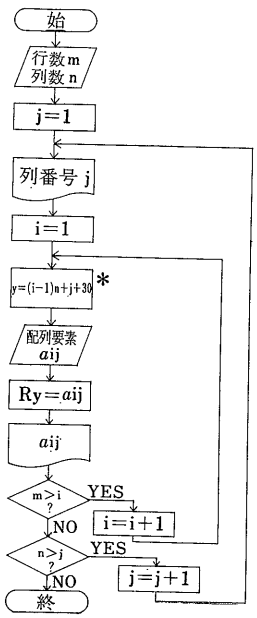
$$x=(j-1)m+i+c \quad \cdots\cdots(2)$$

という関係式で求められるような並び方を　$X$並びと呼ぶ事にする．　これらどちらの並び方を採用するかは自由であり　計算機のメカニズムなども考えて　つごうのよい並び方でシステムを統一すればよい．　もちろんこれらの並び方を互いに変換するプログラムも作る事ができる．　筆者はもっぱら$Y$並びを使っている．　なおFORTRAN で DIMENSION 宣言をしたときのメモリーの割り付けは　$X$並びで行なっている．

いま(1)式を眺めると　式の中に行数$m$の項が含まれていない事に気付く．　これは　$Y$並びの場合は配列の列数はあらかじめ定めておかねばならないが　行数についてはとくに制限がなく　いわば成り行きになっている事




第２図　二次元配列とメモリーの番地との対応
これは$Y$並び（本文）の場合である．　$Y$並びでは同じ行の１列目から２列目　３列目…の順にメモリーの番地が一連番号でふえて行く．
一般に $a_{ij}$ に対応する番地が　$(i-1)n+j+c$　となり　この関係を配列用サブルーチンに使う．




第３図
配列をもつデータの入力

(a)　流れ図
配列をもつデータを処理するもっとも簡単なプログラムの例である．　まず　行番号$i$と列番号$j$を定め　つぎに＊印で示される配列用サブルーチンを実行すればよい．
データの処理を行単位で行なうか列単位で行なうかは$i$と$j$のどちらのループを大ループにするかで決まり　$Y$並びや$X$並び（本文）という並び方とは無関係である．

```
0:
"E";ENT "L?";R0;
"C?";R1⊢
1:
1→B⊢
2:
SPC ;FXD 0;PRT B
;1→A⊢
3:
GSB "Y1"⊢
4:
ENT "DATA";RY;
FXD 6;PRT RY⊢
5:
IF R0>A;A+1→A;
JMP -2⊢
6:
IF R1>B;B+1→B;
JMP -4⊢
7:
SPC 8;STP`⊢
8:
GTO "E"⊢
9:
"Y1";(A-1)R1+B+3
0→Y;RET ⊢
```

(b)　プログラムリスト
流れ図にもとづく　プログラムリストである．　YHP—20の言語は特殊だが　大体のところはお読みいただけると思う．
配列用サブルーチンは第９ラインにあり“Y１”というラベルが付いている．　これを第３ラインで呼び出している．　入力命令（ENT）は第４ラインにあり　入力する番地のデータレジスタは$RY$という間接指定になっている．　システムを通じて配列の行数$m$に$R0$　列数$n$に$R1$　行番号$i$にA　列番号$j$にBのデータレジスタを　それぞれ割り当てる事にしている．

を意味する．　第3図で見てもわかるように　$m$は　行番号$i$の計数器のための単なる指標にすぎない．　この性質を利用すると　データを行単位でいつでも追加できる事になる．　一連の研究の途上で新しいデータが加わるというのは　ごく普通にある事だから　この性質は大いに利用すべきである．　これを行数可変の配列と呼ぶ事にすると　これを使ったプログラムをいくつか組んで使っているが　実際に便利である．　なお　$X$並びの場合は同様の事が列に対して言えるのは当然である．

## 2. 配列をもつデータの編集

卓上型計算機でのデータの入力は　普通はキーボードから直接行ない　パンチカードなどを使うのはまれである．　このため入力の操作が簡単であり　これがこの種の計算機の特長でもある．　しかしその反面一度入力したデータをあとから加除訂正するのは　とくにデータが多量の時には面倒である．　そこでデータの部分的な修正のために全部のデータをはじめから入力し直すという作業をしないで済むように　データの編集をするシステムを考えた．　これによると　任意の配列要素の訂正のほかに　データを行または列単位で加除・入れ替えできる．　それらのプログラムをつぎに列挙する．

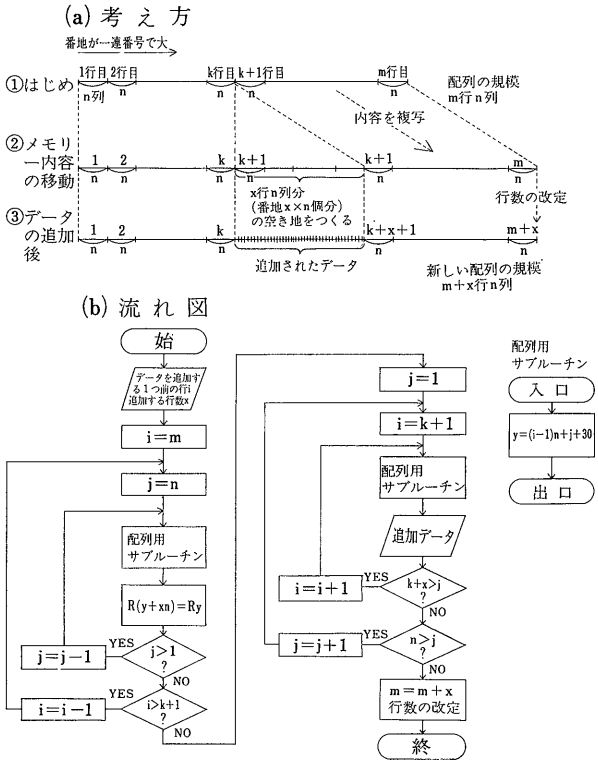
a．配列要素の訂正.
b．行または列の追加.
c．行または列の削除.
d．行または列の入れ替え.
e．配列を任意の行を境に2分し別々にレコード.
f．2つの配列の合併.

これらの中から　行の追加と列の削除の例について簡単に説明する．　行の追加は内容的には任意の行のあと（および第1行の前）に任意の行数を追加するものである．　データの追加後に配列の行数が自動的に改定される．　その考え方と行程を第4図に示す．　列の削除は任意の列以下の数列を削除するもので　削除後に列数が自動的に改定される．　これらを第5図に示す.

## 3. 三次元配列をもつデータの処理

これまでに述べたのは　二次元配列をもつデータの処理法であったが　つぎに同じ配列規模をもつ多数の配列がある場合について考えてみよう．　いま　$m$行$n$列の配列が$k$個あったとすると　その$k$番目の配列要素に対応するメモリーの番地$y$は　$Y$並びの場合

$$y=(i-1)n+j+(k-1)mn+c \cdots\cdots(3)$$



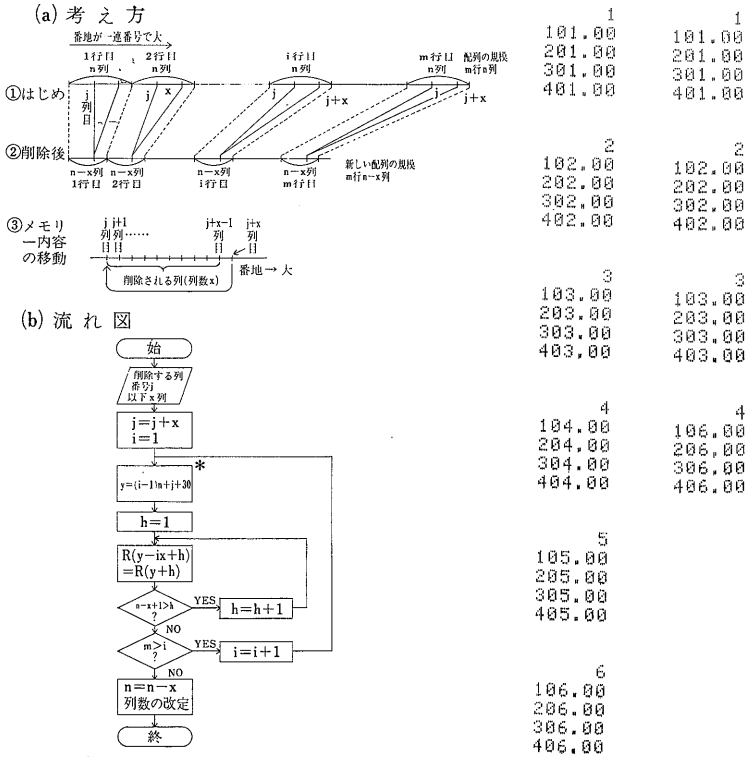

第4図
行を追加するプログラム

(a) 考 え 方
① 配列をメモリーに割り付けたところを示す．　メモリーの番地は図の左から右へ一連番号でふえる.
② 配列の$k$行目のつぎから$x$行分のデータを追加するため　$k+1$行目以下の番地の内容を高い番地へ移動（実際には複写）して　番地に$x\times n$個分の空き地を作る.
③ $k+1$行目以下にデータを入力しこのあと行数は$m+x$に改定される.

(b) 流 れ 図
データの追加をするためのメモリー内容の移動は行番号$i$　列番号$j$の計数器をともに減算しながら行なう事に注意.

(c) 計 算 例
配列の各列についてその第5行目のあとに3行分のデータを追加する例．　左側は追加前　右側は追加後.

| 1 | 1 |
|---|---|
| 101.00 | 101.00 |
| 201.00 | 201.00 |
| 301.00 | 301.00 |
| 401.00 | 401.00 |
| 501.00 | 501.00 |
| 601.00 | 5991.00 |
| 701.00 | 6991.00 |
| 801.00 | 7991.00 |
| 901.00 | 601.00 |
| 1001.00 | 701.00 |
| | 801.00 |
| 2 | 901.00 |
| 102.00 | 1001.00 |
| 202.00 | |
| 302.00 | |
| 402.00 | 2 |
| 502.00 | 102.00 |
| 602.00 | 202.00 |
| 702.00 | 302.00 |
| 802.00 | 402.00 |
| 902.00 | 502.00 |
| 1002.00 | 5992.00 |
| | 6992.00 |
| | 7992.00 |
| 3 | 602.00 |
| 103.00 | 702.00 |
| 203.00 | |
| 303.00 | 802.00 |
| 403.00 | 902.00 |
| 503.00 | 1002.00 |
| 603.00 | |
| 703.00 | |
| 803.00 | 3 |
| 903.00 | 103.00 |
| 1003.00 | 203.00 |
| | 303.00 |
| | 403.00 |
| | 503.00 |
| | 5993.00 |
| | 6993.00 |
| | 7993.00 |
| | 603.00 |
| | 703.00 |
| | 803.00 |
| | 903.00 |
| | 1003.00 |



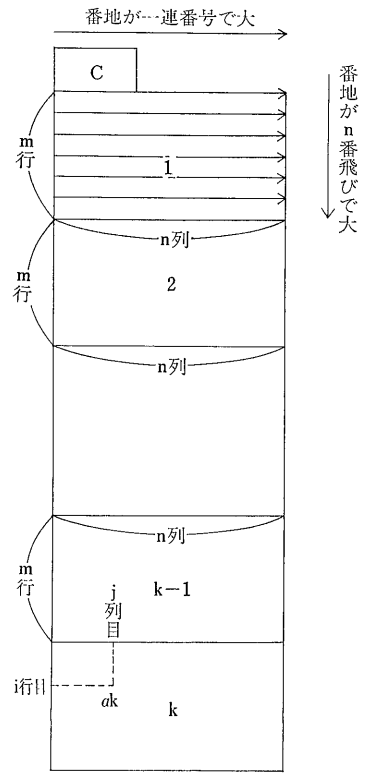

第5図　　列 を 削 除 す る プ ロ グ ラ ム

```
     1          1
101.00     101.00
201.00     201.00
301.00     301.00
401.00     401.00

     2          2
102.00     102.00
202.00     202.00
302.00     302.00
402.00     402.00

     3          3
103.00     103.00
203.00     203.00
303.00     303.00
403.00     403.00

     4          4
104.00     106.00
204.00     206.00
304.00     306.00
404.00     406.00

     5
105.00
205.00
305.00
405.00

     6
106.00
206.00
306.00
406.00
```

(a) 考 え 方
列の変更はY並びの場合　行の変更よりやや複雑な様相となる．③でみるようにj列目以下のx列を削るという事は　j＋x列目の内容をj列目に移すのと同じである．　データ編集用のプログラムでは　随所に"植木算"や"はしご算"が登場するので　注意を要する．

(b) 流 れ 図
＊印が配列用サブルーチン

(c) 計 算 例
配列の第4列以下の2列を削除する例．　左側は削除前　右側は削除後




第6図　三次元配列の要素とメモリーの番地との対応

メモリーの番地は　図の左から右へ一連番号で　また上から下へ向って$n$番飛びにふえるものとする．　この配列の組は　全体をひとつの三次元配列とみなす事もできるし　配列規模が同じ二次元配列の集りとも考えられる．いま第$k$番の配列を考えると　その1行1列目の要素に対応するメモリーの番地の前には$(k-1)mn+c$個の番地が並んでいる．したがってその配列の$i$行$j$列目の要素　$a_{ik}$に対応する番地は　本文の(3)式で表わされる．

という関係式で求める事ができる．　(3)式は(1)式とくらべてkという変数が加わったもので　これはとりもなおさず　三次元配列をもつデータとメモリーの番地との関係式でもある．　この式を実行するサブルーチンを使えば　二次元配列と同様な簡単さで　三次元配列をもつデータの処理ができる．　三次元配列での配列要素とメモリーの番地との関係を第6図に示す．

この三次元配列は　上にも述べたとおり　同じ配列規模をもつ複数の二次元配列とみなす事もできるので　この考えを使うと　行列の加減算や逆行列の計算ができる．そこで　もっと一般的な行列計算を可能にするため　異なった配列規模をもつ複数の配列を自由に処理できるように　これらの方式を拡張する事にした．　これについては次回で述べたい．

以上のような　配列用サブルーチンによる配列の処理法は　配列宣言のできない計算機の不便さを十分おぎなうものである．　また行数可変の配列という考え方は　配列を　FORTRAN　などよりずっと柔軟に取り扱う事を可能にするという点で　卓上型計算機にはふさわしい．

これらの方法はいずれも　メモリー容量が小さい計算機のメモリーの番地を無駄なく使う　という趣旨にもよく合致するものと思う．（つづく）