スパース行列のガウス消去における最適ピボッテング・アルゴリズムとそのフォトランプログラム

スパース行列のガウス消去における最適ピボッテング・アルゴリズムとそのフォトランプログラム

喜屋武 盛 基* 白 川 功** 尾 崎 弘**

Optimal Pivoting Algorithm in Gaussian Elimination for Sparse Matrices and its FORTRAN Program.

Summary

In the process of solving the linear epuation by the Gaussian Elimination or other comparable technigues, a computational interest is the pivotal ordering of the coefficient matix for the given set of epuations.

This paper describes the algorithm fit for a large scale sparse matrices. The algorithm is essentially so called "minimum fill-in" but the method to obtain the minimum fill-in is unigue and some other criteria are added in oder to improve the optimality.

comparisions are made with Ghausi method by actual program and results are given.

1. まえがき

連立方程株の解を求める場合に重要なことに，係数行列を L/U 分解するときのピボットの順序決定がある。

本文では一般の非対称連立方程式をスパース技法を用いて解く場合の最適なピボットの順序決定のアルゴリズムをのべ，同アルゴリズムのFORTRANプログラムと，その結果について述べる。

2. 諸定義および定理

ここにあげる定義および定理は文献〔1〕に示したもののうち，特に本文のアルゴリズムに関連のあるものだけで証明などは略した。

定 義 1 n次の正方行列 $A=〔a_{ij}〕$ に対してブール行列 $X(A)=〔x_{ij}〕$をつぎのように定義する。

$$x_{ij}=\begin{cases}1: & a_{ij}\neq 0\\ 0: & a_{ij}=0\end{cases} \qquad (1)$$

定 義 2 n次の正方行列 $A=〔a_{ij}〕$の任意の $a_{ij}\neq 0$に注目し，行kおよび列lに関して何らかの操作を施し，かつその行および列を取り去って得られるn－1次の正方行列を $A^{*(k,l)}=〔a_{ij}^{*(k,l)}〕$としたとき，$X(A^{*(k,l)})=〔x_{ij}^{*(k,)}〕$と $X(A)=〔x_{ij}〕$の各要素に

$$x_{ij}^{*(k,l)}=x_{ij}\wedge(x_{il}\vee x_{kj}) \qquad (2)$$

の関係が成立すれば，この操作をa_{kl}に関する s-pivotingといい，$a_{k,l}$をこの s-pivoting における pivotという。たゞし∨および∧はブール和およびブール積の演算を表わす。

定 義 3 正方行列 $A=〔a_{ij}〕$の任意の $a_{k,l}\neq 0$をpivot とする s-pivoting において，順序対の集合F_{kl}を

$$F_{kl}=\{(i,\ j)\mid x_{ij}=0,\ x_{ij}^{*(k,l)}=1\} \qquad (3)$$

で定義し，この各元 $(i,\ j)\in F_{kl}$を，この s-pivoting による fill-inという。

定 義 4 正方行列 $A=〔a_{ij}〕$ に対して集合 I_k, J_l を

$$J_l=\{i\mid x_{il}=1\} \qquad (4)$$

$$I_k=\{j\mid x_{kj}=1\} \qquad (5)$$

が定義する。又以下では集合M，Nにたいしてその直積M×Nを

$$M\times N=\{(i,j)\mid j\in M,\ j\in N\} \qquad (6)$$

受付：1973年10月31日

＊琉球大学理工学部電気工学科

＊＊大阪大学工学部電子工学科

で表わし，M × Nの部分集合 (i, N)，(M, j) を

$$(i, N) = \{(i, j) \mid j \in N\} \quad (7)$$

$$(M, j) = \{(i, j) \mid i \in M\} \quad (8)$$

で表わす。

定理 1　n次の正方行列 $A = [a_{ij}]$ の任意の $a_{kl} \neq 0$ を pivot とする s-pivotingによって生ずる fill-inの集合F_{kl}は

$$F_{kl} = \left\{J_l - k\right\} \times \left\{I_k - l\right\} - \left[\left\{\bigcup_{i \in J_l - k} (i, I_l)\right\} \cup \left\{\bigcup_{j \in I_k - l} (J_i j)\right\}\right] \quad (9)$$

で与えられる。

定義 5　n次の正方行列 $A = [a_{ij}]$ に対応する。pivoting グラフG(A)とは節点集合 V (G) = S U T, $S = \{s_1, s_2, \cdots, s_n\}$, $T = \{t_1, t_2, \cdots, t_n\}$ に関する重みのある無向バイパータイトグラフ $G = [V(G); E(G)]$ であり，つぎのような構造をもつものとする。

i)　各$s_i \in S$は Aの行iに各$t_i \in T$は Aの列 Jにそれぞれ1対1に対応する。

ii)　$X(A) = [x_{ij}]$ に対して，$x_{ij} = 1$ であるとき，かつそのときに限り $(s_i, t_j) \in E(G)$であり，枝(s_i, t_j) は重み F_{ij} をもつものとする。

定義 6　n次の正方行列 $A = [a_{ij}]$ に対応するpivotingグラフG (A)に対して，任意の $a_{kl} \neq 0$ を pivot とする s-pivoting により得られる n − 1次の正方行列を $A^{*(k,l)}$ とし，これに対応する pivoting グラフを $G^{*(k,l)} = G(A^{*(k,l)})$ とする。このときGから$G^{*(k,l)}$ への変換操作を枝 (s_k, t_l)に関する縮約という。

定理 2　Aに対応するpivotingグラフG(A)において $F_{pq} = \phi$ (空集合) なる枝が存在するとき，任意の枝 (s_k, t_l) $(k \neq p, l \neq q)$に関する縮約により得られる pivoting グラフ $G^{*(k,l)}$ においても$F_{pq}^{*(k,l)} = \phi$である。

定理 3　pivotingグラフGの互いに隣接する枝(s_i, t_j), (s_k, t_l) $(i = k$ かつ $j \neq l$; または $i \neq k$ かつ $j = l)$ に対して$F_{ij} = F_{kl} = \phi$のとき，pivotingグラフ $G^{*(i,j)}$ および $G^{*(k,l)}$ は枝の重みをも含めて同形である。よって定理2および定理3からつぎの定理を得る。

定理 4　ϕなる重みをもつ枝を含むようなpivotingグラフGに対して，

step 1)　$G_0 \leftarrow G$, $k \leftarrow 1$としてつぎへ移れ

step 2)　G_0において，任意の $F_{pq} = \phi$ なる枝に対して

$$\alpha_k \leftarrow p, \quad \beta_k \leftarrow q$$

としてつぎへ移れ

step 3)　$G_0 \leftarrow G^*(\alpha_k, \beta_k)$ としてつぎへ移れ

step 4)　G_0に重みがϕなる枝があれば $k \leftarrow k+1$ としてつぎへ移れ

step 4)　G_0に重みがϕなる枝があれば $k \leftarrow k+1$ として step 2)へ戻りなければ操作完了。

以上の操作を施して得られる pivotingグラフG_0は step 2) の p, q の選び方にかゝわらず，重みを含めて同形である。

3.　最適ピボッティングのアルゴリズム

step 1)　与えられたn次の正方行列 A_0に対するピボッティンググラフをG_0として，$G \leftarrow G_0$, $\triangle \leftarrow 0$, $N \leftarrow n$, としてつぎの操作より始める。

step 2)　$N = 1$であれば操作完了。$N > 1$のとき，Gに$F_{kl} = \phi$なる枝(s_p, t_q) が存在するかどうかしらべる。もし，存在しなければつぎへ移り，存在すれば，

$$G \leftarrow G^{*(k,l)}$$

$$\begin{pmatrix} V(G) \leftarrow V(G) - \{s_k, t_l\} \\ E(G) \leftarrow E(G) - \{(s_k, t_j) \mid t_j \in P(s_k)\} \\ \quad - \{(s_i, t_l) \mid s_i \in P(t_k)\} \end{pmatrix}$$

$$N \leftarrow N - 1$$

として step 2) をくり返す。

step 3)　Gにおいて

$$K_1 = \{(s\alpha, t\beta) \mid |F_{\alpha\beta}| = \min_{(s_i, t_j) \in E(G)} [|F_{ij}|]\}$$

を求め

$|K_1| = 1$ ならば，$K_1 = \{(s_k, t_l)\}$ として

$G \leftarrow G^{*(k,l)}$

$\triangle \leftarrow \triangle + |F_{kl}|$

$N \leftarrow N - 1$

として step 2) へ戻り，$|K_2| > 1$ならば，つぎへ移れ。

step 5)　各枝$(s\gamma, t\delta) \in K_2$に対して

$$\max_{(s\gamma, t\delta) \in K_2} [|\bigcup_{tj \Gamma (s_\alpha)} F_{\alpha j}|]$$ を与える枝 $(s\gamma, t\mu)$ の集合K_3を求め，任意の$(s_k, t_l) \in K_3$に対して

$G \leftarrow G^{*}(k, l)$

$\triangle \leftarrow \triangle + |F_{kl}|$

$N \leftarrow N - 1$

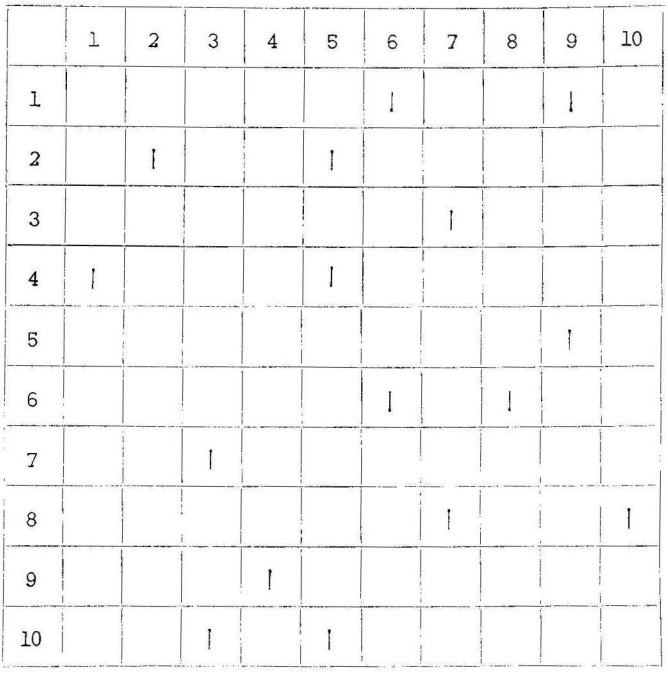
として Step 2）へ戻る

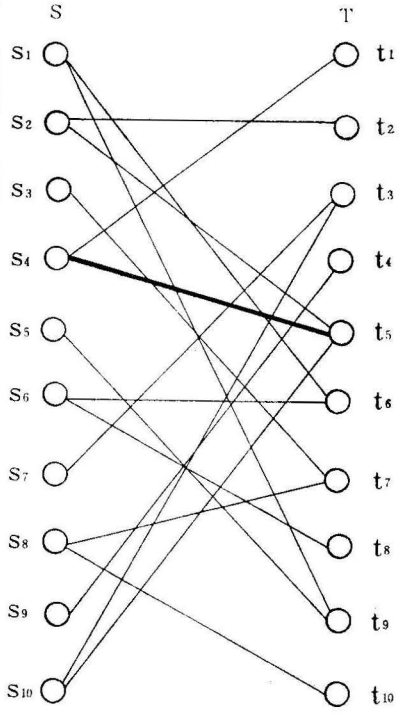
4. データ構造とフォートランプログラム

前述のバイパータイトグラフで表現されるスパース行列を電子計算機で処理する場合に重要なことは，データ構造と処理技法との関係である。

一般にグラフは接続行列で表わすこともできるが，記憶容量がn×n個，必要となり処理の範囲もそれだけ広がるので手間もかさみ得策ではない。[5] 我々が用いた方法はつぎの通りである。大きさNの1次元配列IND(N)と非零要素の総数NZの大きさの1次元配列INZ（NZ），およびNUBF（NZ）で枝の接続関係と枝の重み（Fill-inの数）を表わし更に処理を簡単にする目的でINDとINZに対称なJNDとJNZを利用したので記憶容量は2×(N+NZ)+NZ　程度必要とする。これはスパース行列に適したデータ構造である。

次に10×10の行列とバイパータイトグラフおよびデータ構造の例を示す。





N = 10 , NZ = 16

IND (10) = 2 , 4 , 5 , (7) , 8 , 10 , 11 , 13 , 14 , 16 ,

INZ (16) = 6 , 9 , 2 , 5 , 7 , (1 , (5)) 9 , 6 , 8 , 3 , 7 , 10 , 4 , 3 , 5 ,

JND (10) = 1 , 2 , 4 , 5 , 8 , 10 , 12 , 13 , 15 , 16 ,

JNZ (16) = 4 , 2 , 7 , 10 , 9 , 2 , 4 , 10 , 1 , 6 , 3 , 8 , 6 , 1 , 5 , 8

NUBF (16) = 1 , 1 , 0 , 2 , 0 , 0 , (2) , 0 , 1 , 0 , 0 , 1 , 0 , 0 , 1 , 2

枝(4，5）はIND(4)=7，IND(4−1)+1=6，IND(6～7)={1，5} INZ(7)=5となり枝(4，5）の重み，すなわちFill-inの数はNUBF(7)の中に収めてあり，値は2となっている。また逆にNUBF(3)=0，すなわち|F|=ϕであるNUBFの中の最初の枝はINZ(3)=2=J，JND(2)=2，IND(2−1)=2，INZ(2)=2=Iとなり，枝(I，J)=(2，2)がそれである。この例のように，NUBF(K)又はINZ(K)のKを与えて枝(I，J)を求める操作は割に頻繁に現われるので簡単なサブルーチンRETV(K,I,J)

として用いた。

以上説明したデータ構造を基に，前述のアルゴリズムをフローチャートにしたのが，次の各々のルーチンである。

つぎに MAIN ROUTINEの各部の説明を行う。

① ARYは原始データから IND, INZ, JND, JNZ をつくるサブルーチンで原始データは枝を(I, J)で表わして電子計算機に入力する。前例の 10×10 の行列の場合には（1，6)，(1，9)，(2，2)，(2，5)，(3，7)……となって合計32の数字である。

② はFill-in の計算で，こゝでは2個のサブルーチン IJFLNと NUBKLN を使う。IJFLN (I, J, NUB)は任意の枝 (I, J) を与えてそのFill-inの数NUBを計算し，同時に COMMON 領域にある1次元配列 IFLN に Fill-in の枝番号の対をたくわえるサブルーチンで flow-chart 2 に詳細が示してある。NUBFLN (I, J) は任意の枝 (I, J)，のFill-in の数をCOMMON 領域中の1次元配列 NUBF(NZ)の適当な場所に収めるサブルーチンで詳細はflow-chart 3 に示してある。

flow-chart 4 が MAIN ROUTINE の内部にある Fill-inの計算の詳細なフローチャートである。これですべての枝の Fill-in の計算をし，その値を NUBF (NZ)の中に収容する。

③ Subroutine IFZERO (NZ, N, I, J, IJYN)は配列NUBF(NZ)の中から零を見つけ対応する枝 (I, J) を与えるルーチンでflow-chart 5 に詳細を示す。零がみつかった時IJYN＝1，0がないときIJYN＝0となるのでつぎの判断を経て，KONE(前述のK1，Step3)へ行くか，又はG* (I, J) をつくって Step2)へもどることになる。

④ Subroutin KONE(NZ, K1ST, KCNT)はStep3)のK1を求めるルーチンである。1次元配列 NUBF (NZ)を探索して最少のFill-inを持つ枝 (I, J) の集合 K1 の1次元配列KISTの中に収容し，枝の数 | K1 | をKCNTとする。詳細はflow-chart 6 に示す。

⑤ KTWO (N, NZ, K1ST, KCNT, KCN2) は | K1 | ＞1であるとき，各々の枝 $(\alpha\ \beta) \in$ K1，(KISTの中に収容されている。)について消去を行い，その結果できるG＊(α，β)の各枝の重みの最少のものをK2としてK2STの中に入れ，K2の個数をKCN2とするルーチンである。この操作にはいる直前のGをすべこのK1についての操作が完了するまで保存しなければならないので，そのためにサブルーチンESCAPEを用いた。詳細は割愛する。

⑥ Subroutine KTHR(N, NZ, K2ST, KCN2, K3) は | K2 | ＞1のとき更に Step5) に示す条件を満す枝を探すルーチンである。詳細は紙面の都合上割愛する。

⑦ IJE1M(I, J, NZ, N, NEWNZ)は任意の枝(I, J)をピボットしてE(G* (I, J))をつくるサブルーチンで，行列ではI行J行の消去にあたる操作である。詳細をflow-chart 7 に示す。

⑧ ADIJ(I, J, NZ, N, NEWN8)は任意に枝 (I, J) をGに加え新しいグラフをつくるルーチンである。flow-chart 8 がその詳細である。

⑨ GHAUSI(NZ, N, K, K1ST, KCNT) は，文献〔3〕による HSIEHおよび GHAUSIによるアルゴリズムによる順序決定の手法で我々のアルゴリズムとの比較のために用いた。詳細は flow-hart 9 に示す。

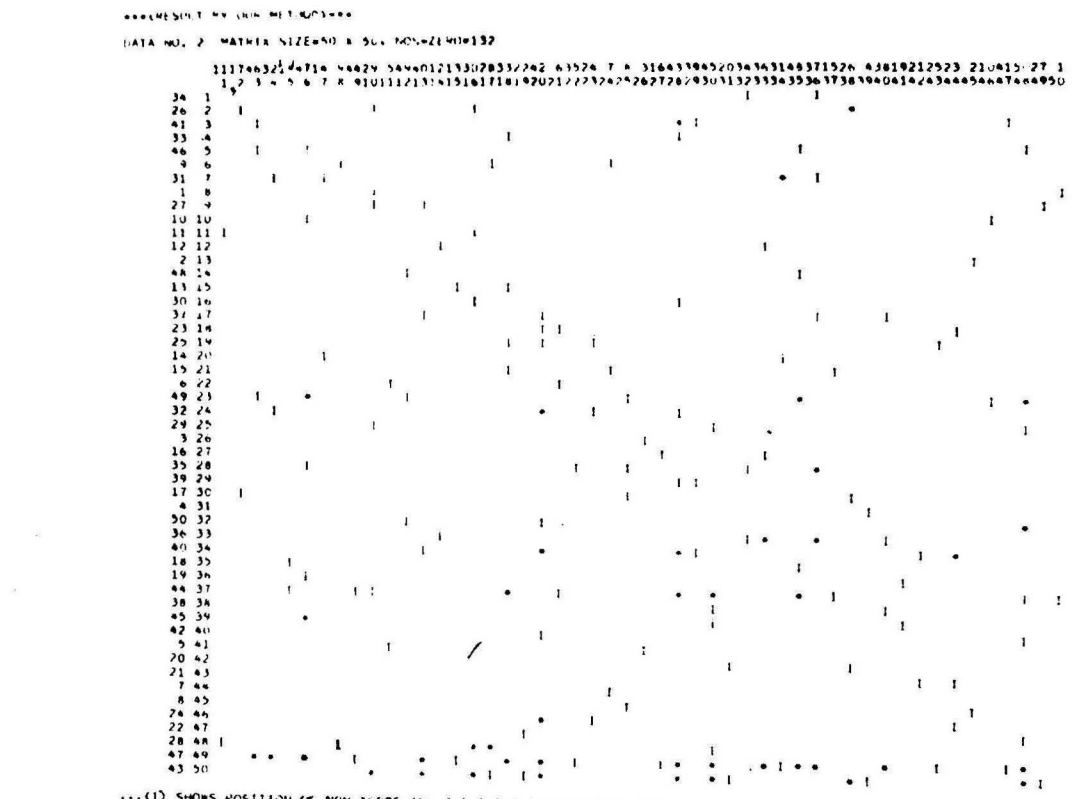
⑩ Subroutine PICTURは 図1および図2のように結果を図示するためのルーチンである。ピボットの順序は外側の番号でも示されるが IORDR, JORDRとして印字される。Fill-inは＊で示され，非零要素はIで示される。またFill-inの発生する順にIJFLとしてFill-inの枝番号が対で印字される。

5. 結果の例

Nの値を最大50までのスパースな係数行列について約40例を実行し，GHAUSIの方法と比較したところ40例中11例がFill-inの総数で改善されたが我々の方法がGHAUSIより悪い例はまだ1例もみつかっていない。実行時間については，計算機の都合で大ざっぱにしか計れなかったが，Fill-inの総和が0かそれに近い場合には我々の方法が早く，Fill-in の総和が多い場合，例えば図1および図2の場合にはGHAUSIの方法が早い。

6. むすび

試みた例がわずか40例であり，これだけでは優劣を確定的には言えないかも知れないが，11例に改善がみられたことと，我々の方法が悪い例がまだ1例もみつかっていないということからFill-inの数においては我々の方法が有利であると云えよう。ただ実行時間においては，多少劣る点もあるが，これはFill-inの数における優位とのかねあいできまることである。例えば数秒余計にかけてFill-inの数を1個へらしたとしても，その後で何千何万回と実行する数値解では，計算時間がはるかに短縮されることが期待できよう。

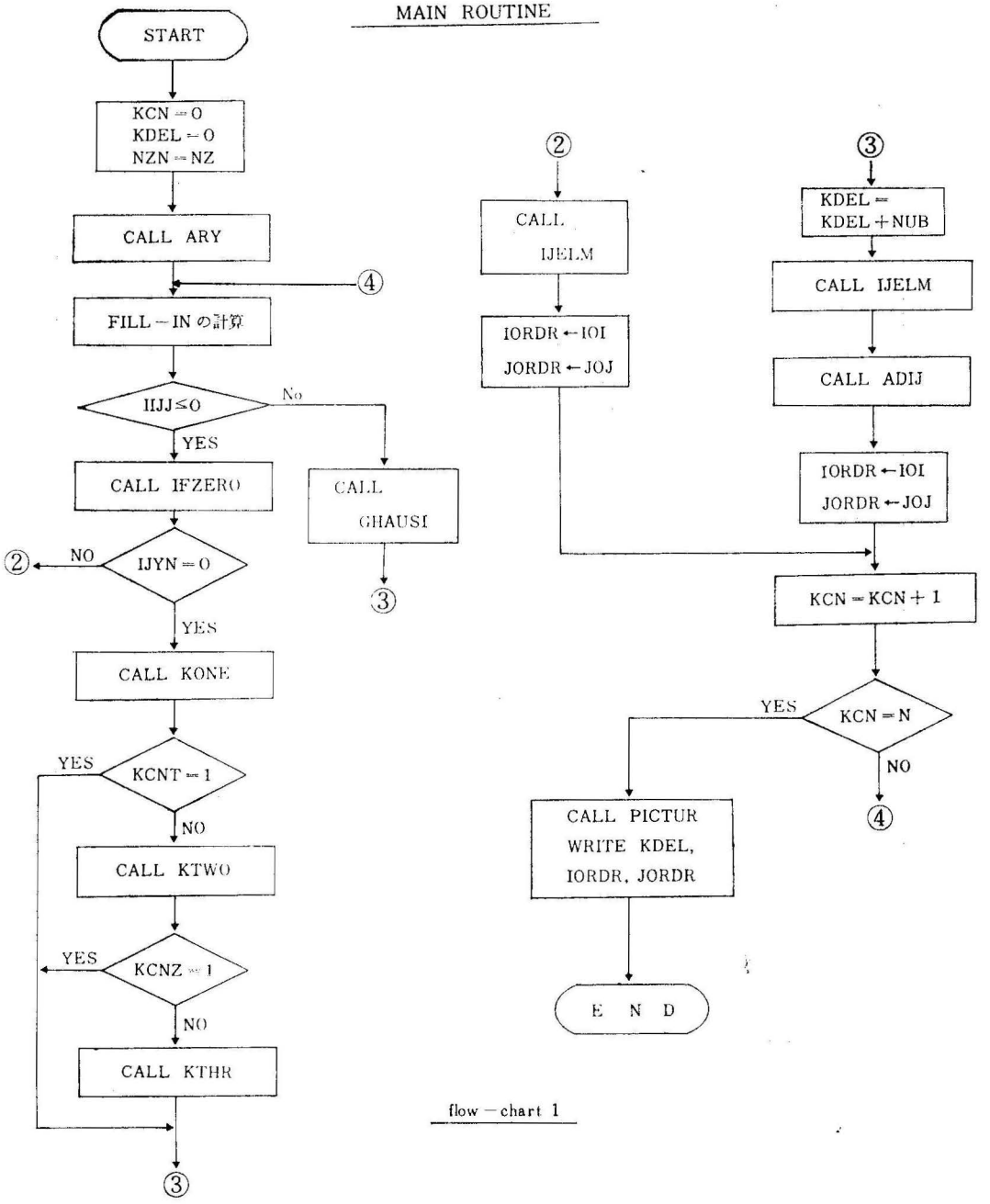


```
***(RESULT BY (RCM METHOD)***

DATA NO. 2  MATRIX SIZE=50 X 50, NON-ZERO=132

...(I) SHOWS POSITION OF NON-ZEROS AND (*) SHOWS THAT OF FILL-INS
...OUTER-MOST NUMBERS SHOW OPTIMAL ORDERING
...NUMBER OF FILL-INS  = 43
...POSITIONS OF FILL-INS (JFL, READ PAIR-WISE
(JFL
      48  17  23   6  48  16  33  33  49  18   7  34  37  15  49  33   2  38  37  35  39   6  50  38  34  44  50  48  46  20
      34  20  24  20  49  27  50  10  50  16  50  13  50  30  37  50  50  28  49   4  49  36  49  28  37  28  28  36  35  36
      49   6  49  40  49  13  49  30   3  27  34  28  50  20  49   3  32  48  49  48  49  35  23  35  23  48
...OPTIMAL ORDERING  (ODR (NOS) AND JODR (CODES)
(ODR
       8  13  26  31  41  22  44  45   4  10  11  12  15  20  21  27  30  35  36  42  43  47  18  46  19   2   9  48  25  16
       7  24   4   1  28  33  17  38  29  34   3  40  50  37  39   5  49  14  23
JODR
      50  45  26  39  11  21  24  25   8  46   1  14  19   7  37  27   2   5  41  31  42  19  44  23  43  38  49  17  10  16
      34   4  18  32  22  35  36  40  29  13  47  20  28   9  30   3   6  35  12
```

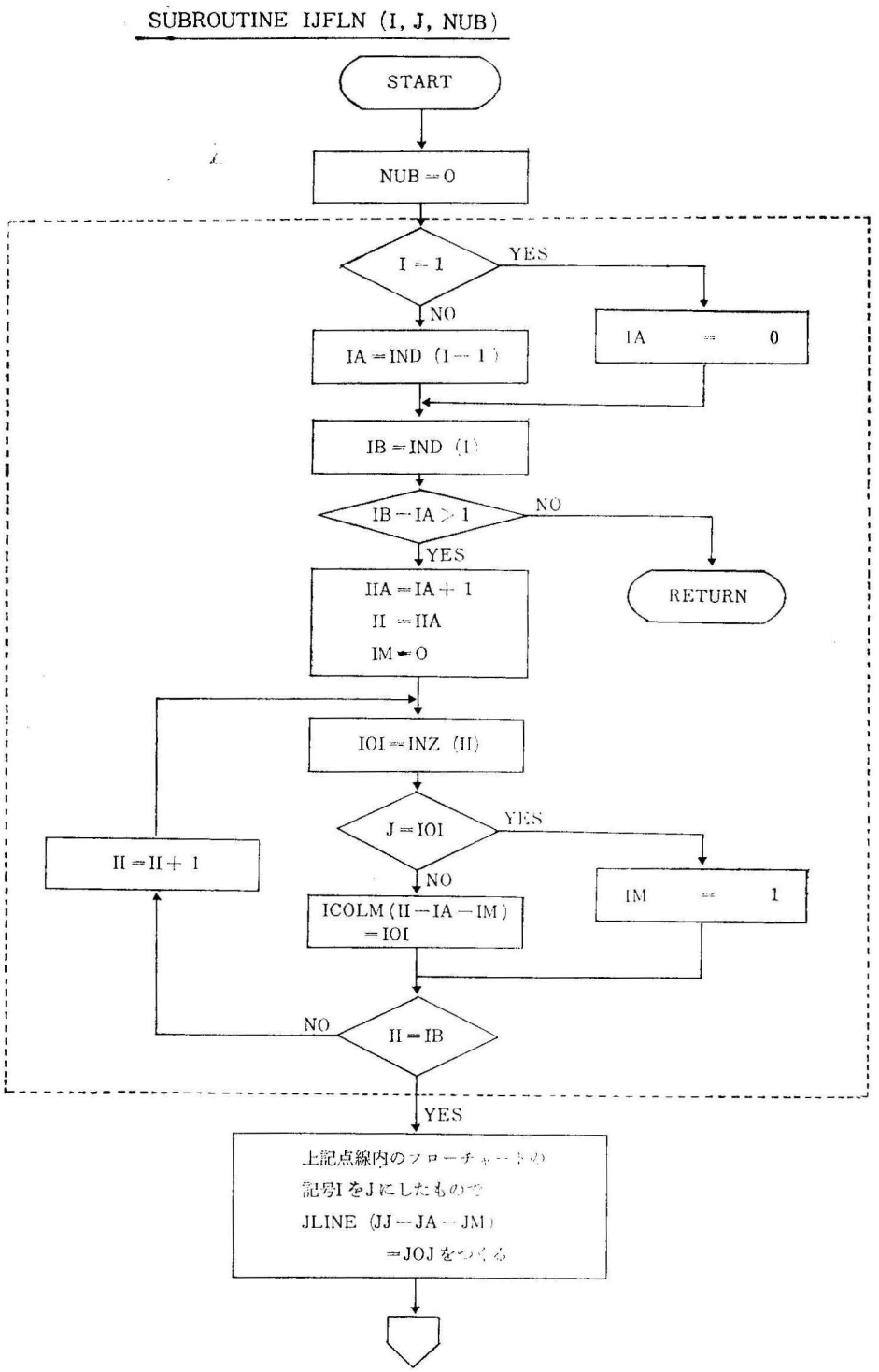
図　1

```
***(RESULT BY GAUSS METHOD)***

DATA NO. 2  MATRIX SIZE=50 * 50, NON-ZERO=132

          1111111111222222222233333333334444444444555555555566666666667777777777888888888899999999990
 1234567890123456789012345678901234567890123456789012345678901234567890123456789012345678901234567890
 1 2 3 4 5 6 7 8 9012345678901234567890123456789012345678901234567890123456789012345678901234567890
 23  1                                                                  I                I
 33  2  I                    I                   I                     •                   •
 48  3    I     •                          •                           • I          •                    I
 29  4                                  I                              I
 42  5    I     I                          •                                        I                    I
  9  6                I                   I                   I
 49  7      I     I                                                              •   I
  2  8                      I                                                                         I
 34  9                      I     I                                                                     I
 10 10                I                                                                        I
 11 11  I                                  I
 12 12                           I                                        I
  3 13                                                                                    I
 24 14                         I                                        I
 13 15                               I     I
 25 16                               I                              I
 47 17                         I                                    •   •          I       I
 26 18                                                  I I                                  I
 28 19                                        I     I     I                                 I
 14 20                   I                                                  I
 15 21                                        I                 I                    I
  8 22                         I                     I
 30 23    I     •               I                           I                    •                    I
 48 24       I                                  •     I           I                 •
 34 25                      I                                    I                                    I
  5 26                                                 I
 16 27                                              I     I
 41 28                I                                     I     I           I     •
 31 29                                                    I I
 17 30  I                                                 I                       I
  6 31                                                                              I
 39 32                         I                     I                       •
 45 33                               I               •                 •   I •     •          I
 37 34                            I                  •                 • I                I   •
 18 35                   I                                                  I
 19 36                   I                                                                 I
 46 37                   I           I I                   •   • I           •   •         •   I     I
 32 38                                                                                 I
 43 39                   •                             •                     I                I
 38 40                                                 I                                          I
  7 41                         I                                    I
 20 42                                                       I                    I
 21 43                                                                                I   I
  1 44                                                  I
  4 45                                                     I                                   I
 27 46                                              •     I                                I
 22 47                                           I                                                I
 35 48  I                    I                 • •                 •   I
 50 49                  •     I     I           I     •     •   I         I •     • I   •          I       I
 44 50                      •     •     I     I •                 •   • I                 • I                • I

...(I) SHOWS POSITION OF NON-ZEROS AND (•) SHOWS THAT OF FILL-INS
...OUTER-MOST NUMBERS SHOW OPTIMAL ORDERING
...NUMBER OF FILL-INS  = 47
...POSITIONS OF FILL-INS (IFL, READ PAIR-WISE
IFL
      48  27  23   6  44  16  33  35  49  19   7  34  37  18  49  33   2  38  37  35  39   6  50  38  34  44  50  48  28  36
      33  36  29  35  32  35   2  28  48  28  46  20  34  20  24  20  49  20  49  28  37  28   3   6   3  35   3  28  34  28
      17  30  33  30  50  10  50  28  50  13  50  30  37  30  50  20  17  28   5  20  37  20   3  20  49   6  49  36  39  20
      33  20  24  36
...OPTIMAL ORDERING  (ODR (ROW) AND JODR (COL.)
IODR
      44   8  13  45  26  31  41  27   6  10  11  12  15  20  21  27  30  35  36  42  43  47   1  14  16  18  46  19   4  23
      29  38   2   9  48  25  34  40  32   3  26   5  39  50  33  37  17  24   7
JODR
      24  50  49  25  26  39  11  21   8  46   1  14  15   7  37  27   2   5  41  31  42  19  32  12  16  44  23  43  10   3
      29  40  38  45  17  10  13  48  35  47  22   6  50  28  33   9  20   4  36
```

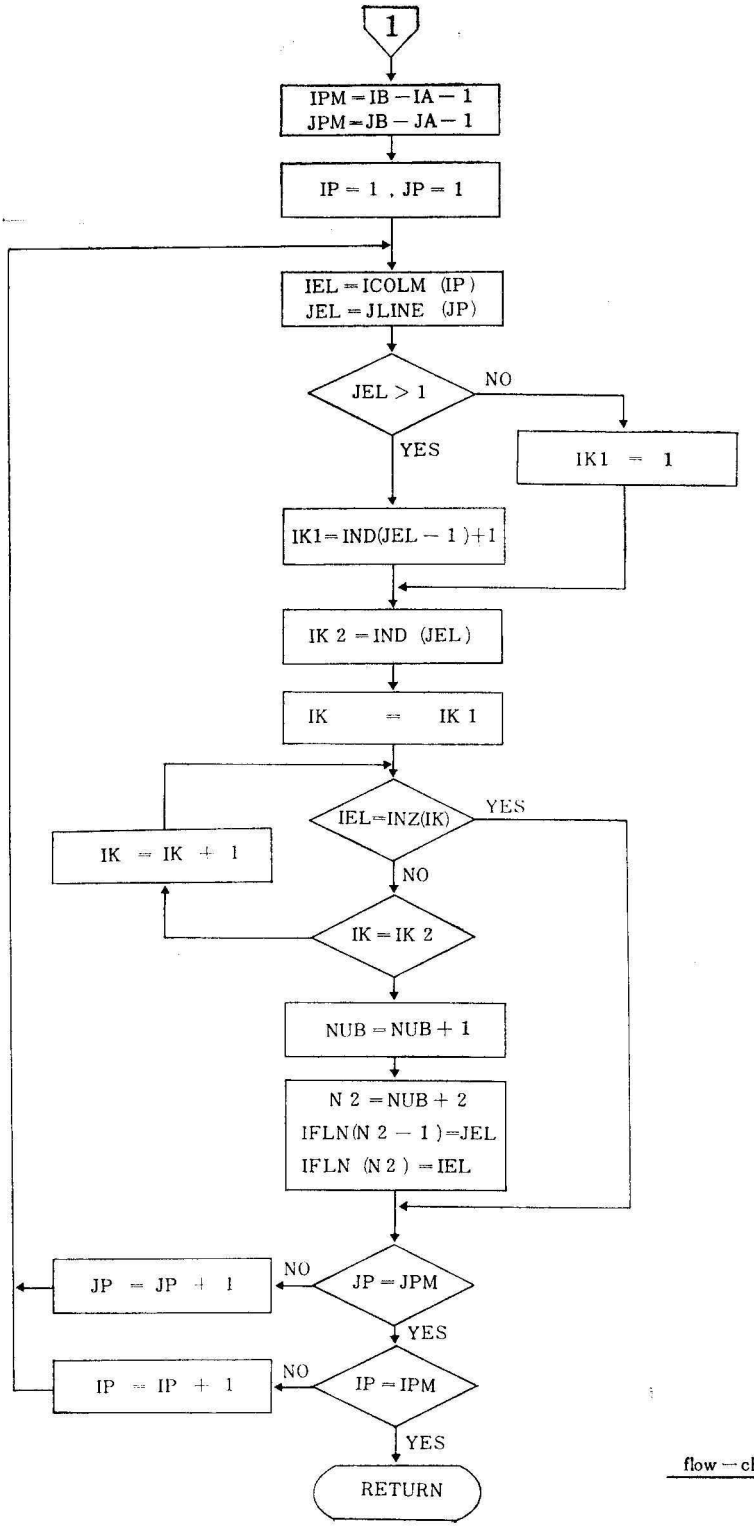
図 2


MAIN ROUTINE
START
KCN = 0
KDEL = 0
NZN = NZ
CALL ARY
④
FILL－IN の計算
IIJJ≦0
No
YES
CALL IFZERO
CALL GHAUSI
②
NO
IJYN = 0
YES
③
CALL KONE
YES
KCNT = 1
NO
CALL KTWO
YES
KCNZ = 1
NO
CALL KTHR
③
②
CALL IJELM
IORDR ← IOI
JORDR ← JOJ
③
KDEL = KDEL + NUB
CALL IJELM
CALL ADIJ
IORDR ← IOI
JORDR ← JOJ
KCN = KCN + 1
YES
KCN = N
NO
④
CALL PICTUR
WRITE KDEL,
IORDR, JORDR
E N D

flow－chart 1

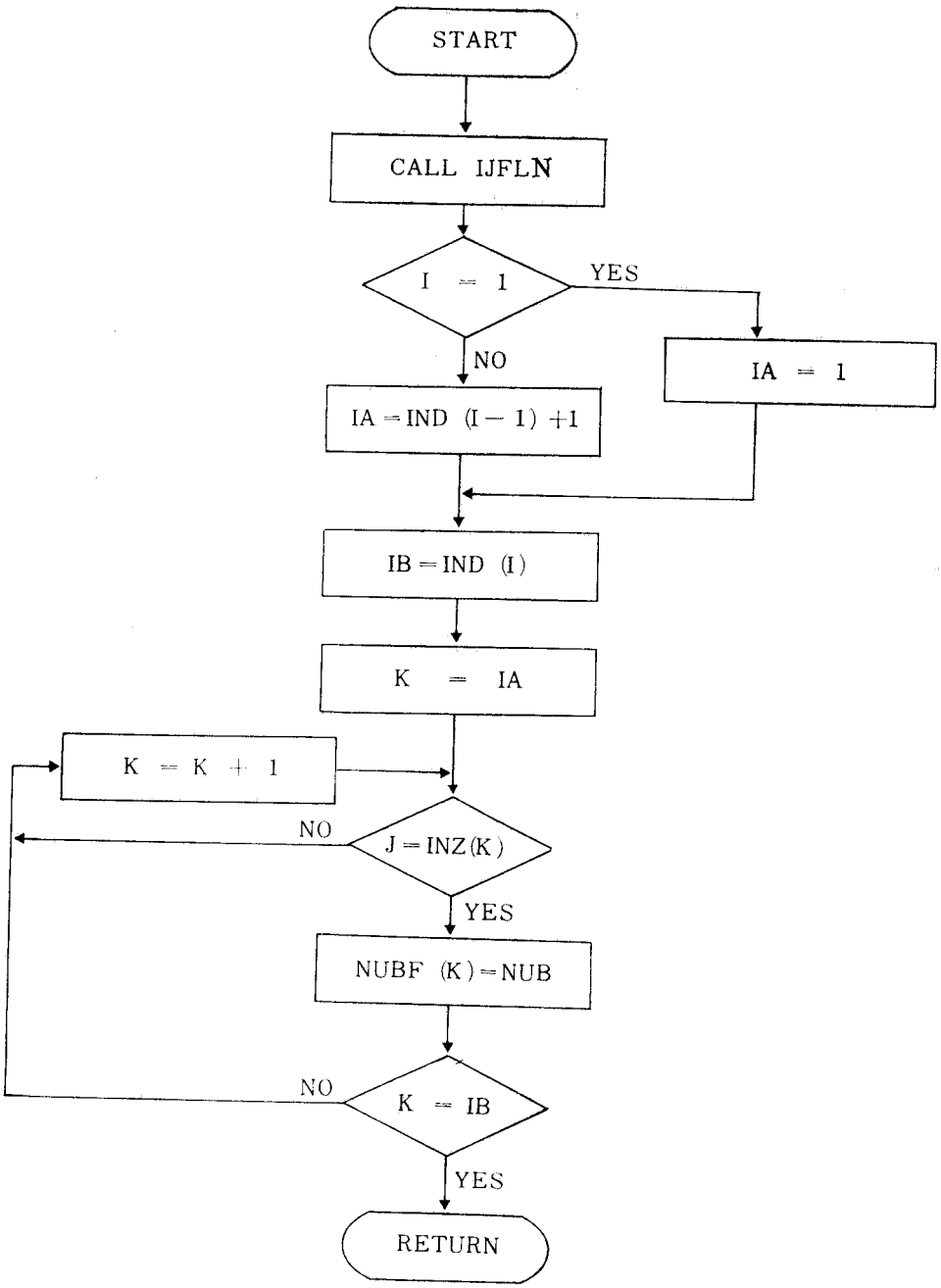


flow－chart 2 －a

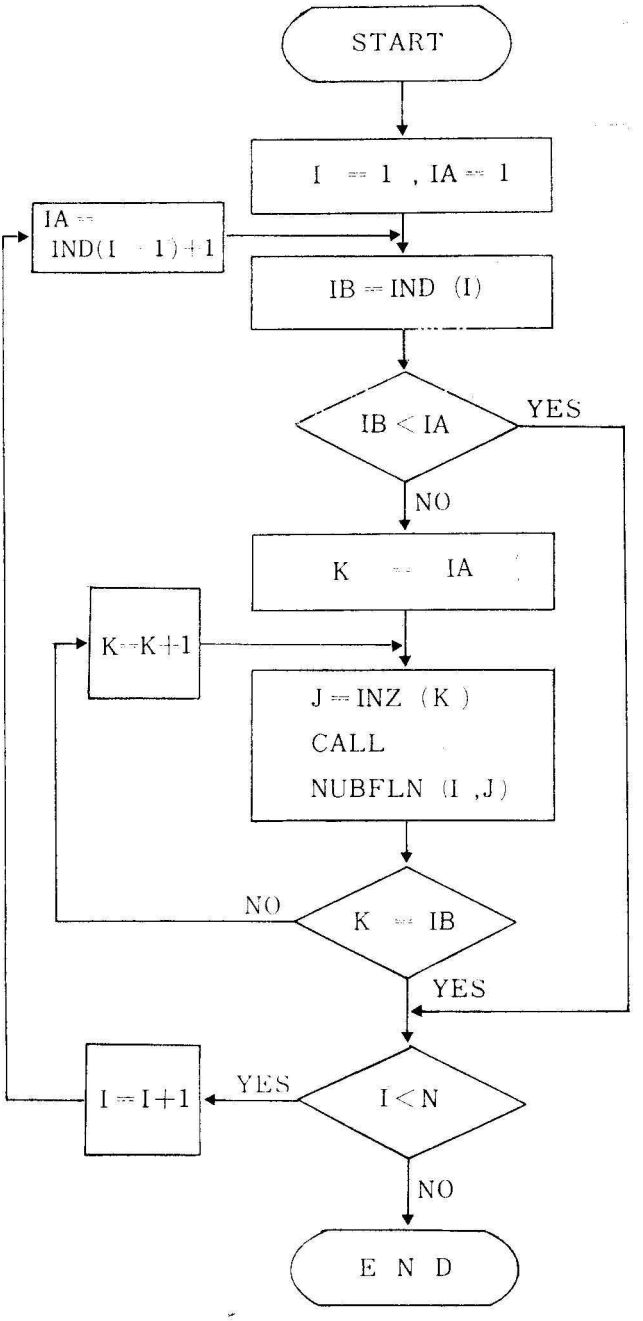


flow－chart 2 －b

SUBROUTINE NUBFLN (I, J)

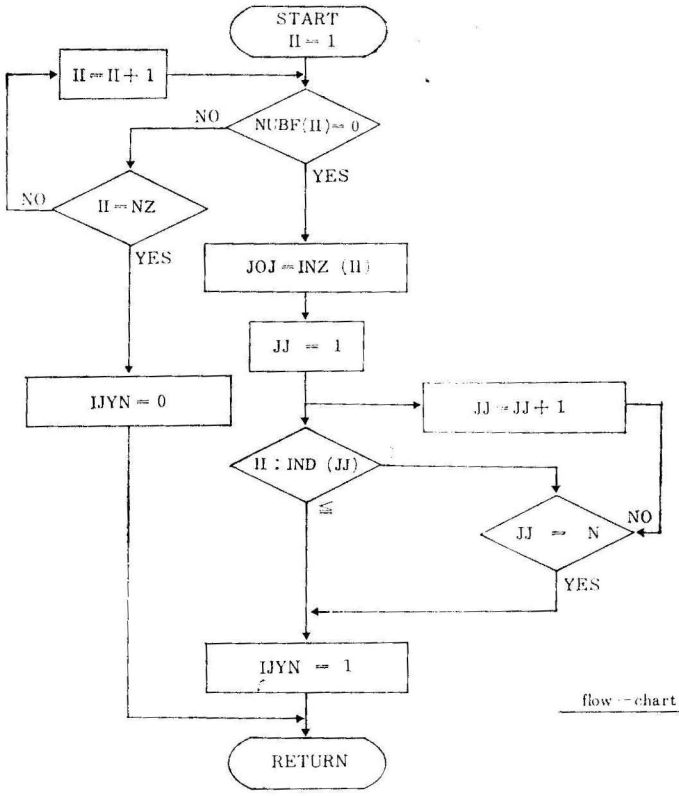


flow − chart 3


START
I = 1 , IA = 1
IA = IND(I - 1)+1
IB = IND (I)
IB < IA
YES
NO
K = IA
K=K+1
J = INZ (K)
CALL
NUBFLN (I ,J)
NO
K = IB
YES
I = I+1
YES
I < N
NO
E N D

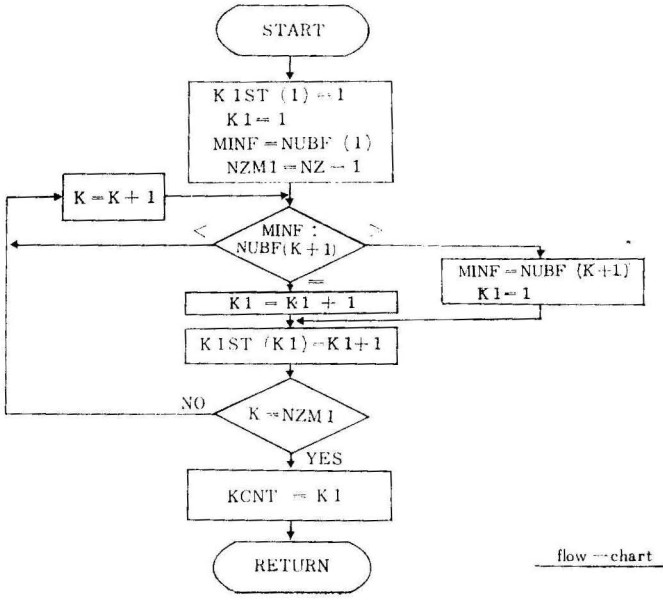
flow－chart 4

SUBROUTINE IFZERO (NZ, N, IOI, JOJ, IJYN)



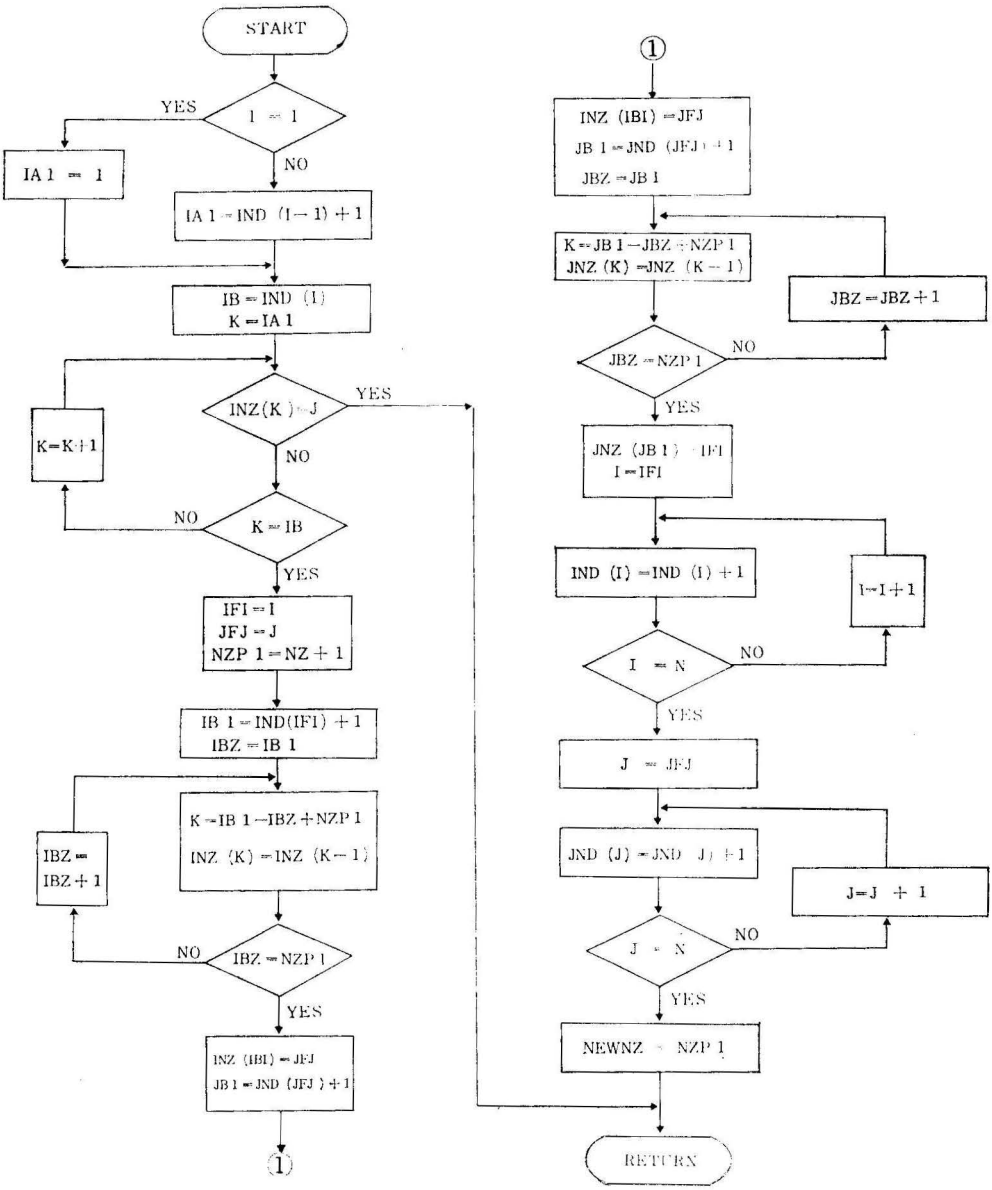
flow-chart 5

SUBROUTINE KONE (NZ, KIST, KCNT



flow-chart 6

SUBROUTINE ADIJ (I, J, NZ, N, NEWNZ)



flow − chart 7

以上述べた結果から我々の手法の優位性がほゞ確められたが，さらに確定的にすすめるために，Ghausi等の文献〔2〕のアルゴリズムによるプログラムを開発中であり，つぎの機会に，それと我々との比較報告を行ないたい。また我々のピボット順序づけの有効性が確められた後、この方法を利用した巨大行列の数値解法の開発も残された研究課題の一つである。

謝　　辞

おわりにこの研究の遂行にあたり，幾多の便宜と御支援を下さった大阪大学基礎工学部嵩忠雄教授，ならびに琉球大学電子計算機室長山下崇教授に深謝します。

参考文献

〔1〕 喜屋武，白川，尾崎，〃線形系解析における最適Pivoting順序とこれに付随するバイパータイト・グラフ〃 電子通信学会回路とシステム 理論研究会資料 CT72—68(1973—01)

〔2〕 H.Y. Hsieb aud M.S Ghausi "A probabilistic approach to optimal pivoting and prediction of fill-in for random sparse matrices" IEEE Trans. on Circuit Theory, Vol, CT-A, No4, pp329-336 July 1972.

〔3〕 H. Y. Hsieh, M. S. Ghausi, " On optimal-pivoting algorithm in sparse matirces " IEEE Trans, on Circuit Theory (Corresp) Vol. CT-19, pp93-96, Jan.1972

〔4〕 R. D. Berry," An optimal ordering of electronic circuit equations for a sparse matrix solutions" IEEE Trans. on Circuit Theory, Vol.CT-18, p40-50, Jan.1971

〔5〕 伊理正夫〃グラフ的構造を有する情報の処理技法について〃，昭和47年度電気四学会連合大会講演論文集