講　座

スパース行列処理技法(2)*

大附辰夫** 川北建次**

3. 行列の分割とピボット順序

同じスパース構造を持ち非零要素の値のみが異なる行列を多数処理する場合には，ピボット順序をあらかじめ決定しておいて，その順序に従った数値計算を毎回繰り返すのが能率良いとされている[37]．この場合，誤差の問題を無視して，数値計算の実行時間および非零要素の記憶場所を節約することだけを目標としてピボット順序を決定するのが普通である***．

スパース行列といっても，単にスパースであるというだけでなく，非零要素の配置にある規則的な構造が認められることが多い．応用によっては，あらかじめ規則的構造が明らかな場合もあるが，一般にはある種の形の分割（後述）を目標として零行列の範囲が広くなるように行や列を入れ換える手順が重要である．与えられた大きな行列が多数の部分小行列に分割された場合，部分小行列ごとに逆行列，行列和，行列積などの計算――**ブロック消去**という――を行うことによって，全体の計算の能率を高めることができる．

行列の分割の中で――分割が本質的に一意に定まるという意味で――最も基本的なものは**図-3(a)**のような**ブロック三角行列** (block-triangular matrix) への分解である．特に対称性構造を持つ場合には，同図(b)のような**ブロック対角行列** (block-diagonal matrix) と呼ばれる形になる．このような分割に着目すれば，各々の主座小行列ごとにそれを係数とする線形連立方程式を解く問題に分解できることから，適当に行や列を入れ換えて2個以上のブロックを持つ三角行列に変換できる行列は**分解可能** (reducible) であるといわれる．そうでない行列は**分解不可能** (irreducible) であるといわれる．

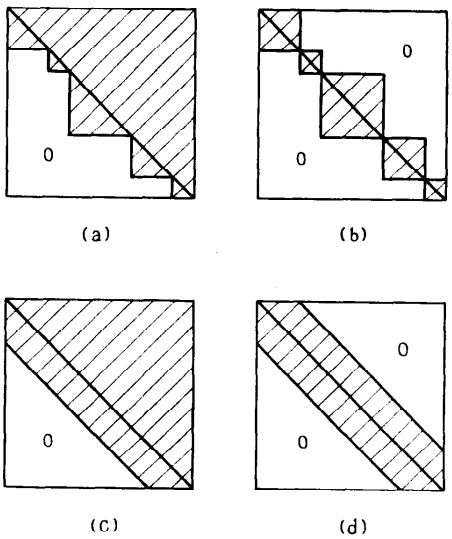


図-3 ブロック三角行列(a)，ブロック対角行列(b)，帯三角行列(c)および帯行列(d)

線形連立方程式の係数行列を先ずブロック三角行列の形に整理してからガウス消去法などの求解操作を行うことは，単に計算の能率を高めるためばかりでなく，誤差を低くおさえる面からも重要である．すなわち，ブロック三角行列の主座小行列以外の非零要素をピボットに選ぶと，大きな丸め誤差が発生する[38]．

偏微分方程式の求解や有限要素法などの分野では，図-3(c),(d)に示すような帯構造を持つ行列――それぞれ**帯三角行列** (band triangular matrix)，**帯行列** (band matrix) という――に適した数値計算法が開発されている[39)~42)]．

行列の帯構造に着目した数値解析を行う場合には，その前処理として，最も**バンド幅** (bandwidth) を小さくするようなピボット順序（行と列の置換）を定める問題が重要であり，これに関しても多くの研究成果が報告されている[43)~48)]．

行列のスパース構造によっては，ブロック三角型に分解できなかったり，帯行列化しようとしてもバンド

---

* Sparse Matrix Technology by Tatsuo OHTSUKI and Kenji KAWAKITA (Central Research Laboratories, Nippon Electric Co., Ltd.)

** 日本電気(株)中央研究所

*** 誤差が大きすぎる場合には，2.7 節，2.8 節に述べたような手法を並用して，精度を改善するという方策が用いられる．

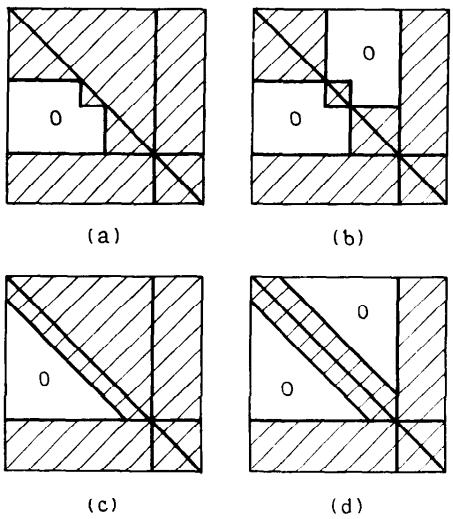


図-4 準ブロック三角 (bordered block-triangular) 行列 (a), 準ブロック対角 (bordered block-diagonal) 行列(b), 準帯三角 (bordered band triangular) 行列(c), 準帯 (bordered band) 行列 (d)

幅が大きくなりすぎる場合がある．このような行列に対して**図-4 (a)～(d)** のような分割に着目すると零行列の部分が広く取れることがある．この場合，右下のブロックに対応した行と列を取り除いた部分についてブロック三角行列または帯行列を対象としたアルゴリズムを適用すれば，全体の計算能率が高められる[49)~51)]．行列を図-4 のような形に整理する問題に対しても，零行列の範囲を最大にすることを目標としたヒューリスティックな算法がいくつか提案されている[52)~58)]が，厳密な最適解を与えるための能率よい算法は知られていない．

電気回路解析などに介入する大規模行列はランダムなスパース構造を持つ場合が多く，上記のような規則的な構造を目標としたスパース処理を行うのは得策ではない．というのは，そのような分割を行ったとしても，非零ブロック (図-3,4 の斜線の部分) もかなりスパースになり，零行列の部分が広く取れないからである．ランダムなスパース構造を前提としてピボット順序を決定する場合，次の二通りの目標が考えられる．

(I) 消去型 (積型) 逆行列の中に発生する非零要素の数を最小にする．これは同時に代入操作の演算回数を最小にしていることにもなる．

(II) 消去型 (積型) 逆行列を求めるのに必要な演算回数を最小にする

上記2つの目標は同一ではないが，片方に対する最適の順序付けは他方に対しても最適に近いものであるので，実用上はいずれか片方だけを目標とした順序付けのアルゴリズムを適用するのが普通である．いずれの目標に対しても，厳密な最適順序を与えるための能率的な算法は知られていないが，実用上許される計算の手間の範囲内で，最適に近い順序付けを与えることを狙ったアルゴリズムは数多く――消去型逆行列を前提としたものとして文献2),29),31),59)～71) など，積型逆行列を前提としたものとして文献61),72)～77) などがある――提案されている．このほかにも，電子回路網，電力系統などへの応用を前提として，問題の特殊性に適したピボット順序付けを求めようとする研究も多い[29),57),78)~87)]．

以上述べたように，大規模問題の解析において行列のスパース性を利用しようとする場合，図-4 のような規則的な構造に分解することを狙うアプローチと，ランダムなスパース構造を前提として，最適なピボット順序を求める問題に帰着させるアプローチがある．両者をどう使い分けるかという議論はあまりなされていないが，主として前者は主記憶と二次記憶を用いて非零要素に関するデータの変換を行う場合に，後者は主記憶の中だけで処理を行う場合を対象としている．

4. グラフ理論の応用

前章で論じた行列の分割やピボット順序の問題は，行列のスパース構造，すなわちどの要素が0であるか否かという非数値的な情報だけを議論の対象としているので，これをグラフで表現することによって全体の見通しが良くなることが多い．以下で用いるグラフの用語は特にことわらない限り，本学会誌の講座[88)]に従うものとする．

正方行列 $A=\{a_{ij};\ i,j=1,2,\cdots,n\}$ のスパース構造は**2部グラフ** $G=(V,E)$; $V=V_1\cup V_2$ によって表現できる．ここで G は各節点対 $v_i\in V_1$, $v_j\in V_2$ に対して，$a_{ij}\neq 0$ のとき，しかもそのときに限り v_i と v_j を結ぶ枝が存在する $(v_i,v_j\in E)$ という性質を満たすグラフである．例えば**図-5 (a)** (次頁参照) の行列のスパース構造は同図(b)のグラフで表わされる．行列 A の行や列の置換によって得られる行列は，適当な2つの置換行列 P,Q を用いて $\bar{A}=PAQ^T$ と表現できる (2.6 節参照) が，全ての P,Q の組に対して，$\bar{A}$ を表わす2部グラフは**同形**である．

行列のスパース構造の分析は，対応する2部グラフ

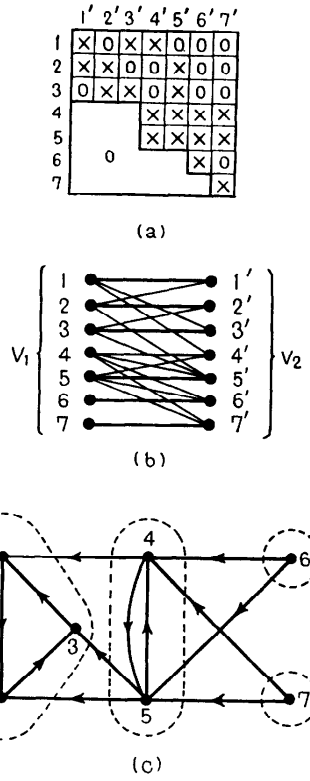


図-5 ブロック三角行列(a)とその2部グラフ表現(b)および有向グラフ表現(c)

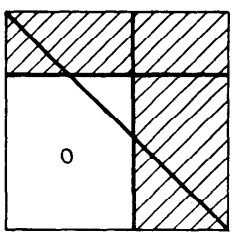


図-6 一意解を持たない連立方程式の係数行列

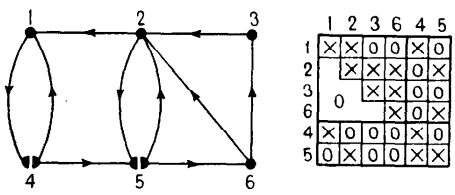
の**完全マッチング**を求めることから始まる．例えば，図-5(b)の2部グラフにおける完全マッチングの1つを太線で示してある．完全マッチングが存在しない場合は，その行列は**図-6**のような形をしていることになり，それを係数行列とする連立方程式は一意解を持たない．1つの完全マッチングが求まれば，これに対応する行または列の片方の置換（他方は固定する）によって，対角項に非零要素を配置することができる．

対角項だけをピボットにとることに限定して，ピボット順序を決定する場合は，対角項を節点に，非零要素を枝に対応させたグラフを用いると便利である．すなわち正方行列 $A=\{a_{ij};\ i,j=1,2,\cdots,n\}$ のスパース構造を表わす有向グラフ $G=(V,E)$ は，任意の異なる節点対 $v_i,v_j\in V$ に対して，$a_{ij}\neq 0$ のとき，しかもそのときに限り，v_j から v_i に向う枝が存在するように定めれば良い．対称構造を持つ行列に対しては，無向グラフを用いれば十分である．すなわち，相違なる節点対 v_i,v_j に対して $a_{ij}\neq 0,(a_{ji}\neq 0)$ のとき，しかもそのときに限り，v_i と v_j の間に枝が存在するように G の接続関係を定めれば良い．例えば図-5(a)の行列のスパース構造は同図(c)のグラフによって表わされる．行列 A の対角項を固定したもとで，行と列の対称置換によって得られる行列は，置換行列 P を用いて $\tilde{A}=PAP^T$ と表現できる（**2.6**参照）が，全ての P に対して，$\tilde{A}$ に対応するグラフは同形である．

行列 A のスパース構造を表わす有向グラフ G が，**強連結**でない場合は，それを強連結成分に分解し，それらの半順序関係を調べることによってもとの行列のブロック三角行列分解が得られる．例えば図-5(c)の各々の強連結成分（点線で囲んだ部分）の間の半順序関係は同図(a)の行列のスパース構造を与えている．一般に完全マッチングが存在する限り，行列のブロック三角型分解は，完全マッチングの選び方とは独立に一意に定まり[89]，また，それを具体的に求めるためのアルゴリズムも知られている[90]．行列をブロック三角型に分解することは，2部グラフの完全マッチングを求める問題と，有向グラフを強連結成分に分解する問題とから成っているが，最近の研究により，前者は $n^{2.5}$ に比例する手間で[91]，後者は枝（非零要素）の数に比例する手間で[92]解けることが明らかになっている．

行列が分解不可能 (irreducible)，すなわち対応する有向グラフが強連結の場合には，上記のブロック三角型分解を行うことはできない．分解不可能な行列を図-3(c),(d)あるいは図-4のような形に整理する問題もグラフを用いて表現すると便利である[93),94)]が，最適化問題としての定式化そのものが複雑*であり，あまり有効な結果は得られていない．図-4(a)または(c)の特殊な場合として，行列を**準三角型** (bordered triangular) に整理する問題は，対応する有向グラ

* 例えば，準ブロック三角化する場合，右下の対角ブロックを小さくすることと，左上の各対角ブロックを小さく且つ大きさの平均化をはかることは同等に重要であり，目的関数の設定の仕方があいまいである．



(a)　　(b)

最小帰還節点集合((a)の◐印)と対応した準三角行列(b)

図-7

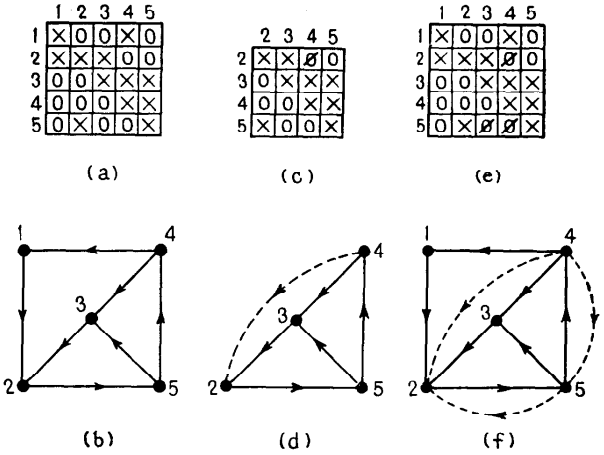
フから最小数の節点を取り除いて有向閉路の全てを消滅させるという有名な最小帰還節点集合を求める問題に帰着される(図-7参照). この問題の定式化は極めて単純であり, 準最適解を狙ったヒューリスティック算法がいくつか提案されている[95),96)]が, 厳密な最適解を能率良く求める算法は発見されていない.

消去型逆行列のスパース性を保存するという意味で最適のピボット順を求める問題は, グラフを用いることによって明解な定式化ができる[56),97)~101)]. 与えられた行列を A (ピボットは常に対角項から取ると仮定), これのスパース構造を表わすグラフを $G=(V, E)$ とする. 今, A の対角項の1つをピボットに選んでガウス消去法のステップ(2.1における $A^{(1)}=A$ から $A^{(2)}$ を得る操作)を行ったと仮定すると, $A^{(2)}$ の主要部($A^{(2)}$ からピボット要素に対応する行と列を取り除いた行列)のスパース構造を表わすグラフ $G^{(2)}$ は $G^{(1)}=G$ のピボットに対応する節点——例えば $a\in V$ とする——に対して次の操作をほどこすことによって得られる.

(1) $b\in V$ から a を通って $c\in V$ に至る長さ2の道が存在するような各々の節点対 (b,c) に対して, b から c に向う枝を付加(もともと存在する場合はそのままにしておく)し, 次に

(2) 節点 a, およびこれに接続している全ての枝を除去する.

グラフ G におけるこの操作を**節点縮約** (vertex elimination). また, いくつかの節点について節点縮約を行って得られるグラフを G の**縮約グラフ** (elimination graph) という. 例えば, 図-8(a)の行列の(1,1)要素をピボットした結果は同図(c)のようになるが, これは同図(b)のグラフ G から同図(d)の縮約グラフを得ることに対応する. 対称行列(無向グラフ)の場合には上記(1)の操作を次のような操作に置き代えることができる.



(a)　(c)　(e)

(b)　(d)　(f)

図-8 節点縮約(非対称行列の場合)——
行列 $A^{(1)}=A$(a) と対応するグラフ $G^{(1)}=G$(b), ピボットを一回行った結果 $A^{(2)}$(c) と縮約グラフ $G^{(2)}$(d), および消去型逆行列のスパース構造 A^*(e) と対応するグラフ $G[\alpha]$(f)
ϕ: 新たに発生した非零要素
⇢⋯: 新たに発生した枝

(1)' $b,c\in V$ 共に a に隣接しているような各々の節点対 (b,c) に対して b と c を結ぶ枝を付加(もともと枝がある場合はそのままにしておく)する.

図-9(a)~(d)(次頁参照)は無向グラフにおける節点縮約の例を示している.

一般に行列 A の対角項をある順序でピボットすることは, グラフ G の節点に1から n までの番号付けをし, この番号順に節点縮約をほどこすことに対応する. 今, グラフ $G=(V,E)$; $|V|=n$ の節点のある順序付け α に対して, この順番に節点縮約を行ったとき新たに発生する枝の集合を F とすれば, F は A の消去型逆行列の中に新たに発生する非零要素の集合を表わすことになる. 更に G の**スーパーグラフ**

$$G[\alpha]=(V,\hat{E});\ \hat{E}=E\cup F \qquad (4,1)$$

は消去型逆行列のスパース構造, すなわち2.1の表現を用いれば

$$A^*=\sum_{i=1}^{n}U_i+\sum_{i=1}^{n}L_i \qquad (4.2)$$

のスパース構造を表わしていることになる. 図-8(e), (f)および図-9(e), (f)はそれぞれ非対称行列, 対称行列について番号順にピボットした場合の例を示す.

以上の議論から明らかなように, 行列のスパース性

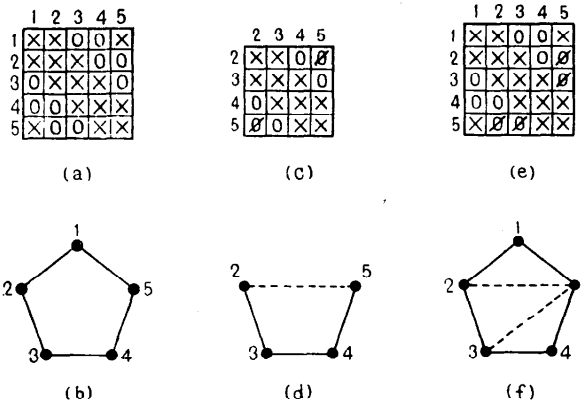


図-9 節点縮約（対称行列の場合）——
行列 $A^{(1)}=A$(a) と対応するグラフ $G^{(1)}=G$(b)，ピボットを一回行った結果 $A^{(2)}$(c) と縮約グラフ $G^{(2)}$(d)，および消去型逆行列のスパース構造 A^*(e) と対応するグラフ $G[\alpha]$(f).
ϕ: 新たに発生した非零要素
……: 新たに発生した枝

を最も保存するピボット順を求める問題は，対応するグラフ G の節点の順序付け α の中でスーパーグラフ $G[\alpha]$ に付加される枝の数 $|F|$ を最小にするものを求める問題と解釈できる．厳密な最適解を求めるためには，“虱潰し”に毛がはえた程度の方法しか知られていないが，前節に紹介したように準最適解を狙った算法は種々提案されている[2),29),31),59)~71)]．これらの方法は，局所的に良いと思われる判断に基づいて次々と縮約すべき節点を選んで行くもので，基本的には各段階の縮約グラフにおいて，次のような選択を行っている．

（A） 接続している枝の数が最小の節点を選ぶ．

（B） 入り込んでいる枝の数と出て行く枝の数の積が最小となる節点を選ぶ．

（C） 節点縮約によって新たに発生する枝の数が最小となる節点を選ぶ．

広く使われている**マルコビッツ法**[2)] は(B)を適用したものである．無向グラフの場合は，(A)と(B)は一致し，**最小次数法** (minimum degree method) と呼ばれているものに相当する．一般に(C)は(A)や(B)よりも良い順序付けを与えるが，それだけ縮約すべき節点を発見するための手間を要するという欠点を持つ．最近はもっと質の良い順序付けを求めることを狙って，上記の判断規準を二段階の節点縮約まで考慮して適用する方法[67)]や，どれか1つの規準では区別できない複数個の節点に対して他の規準を適用する方法[69),70)]などが提案されている．一般に手の込んだ判定を行えばそれだけ得られる順序付けの質が良くなるのは当然であるので，今後は，順序付けを求めるための手間と得られた結果の良さとのトレードオフをどこに設定するか，という議論が重要であろう．

最適な順序付けを与える実用的な算法は知られていないと述べたが，特殊な構造の無向グラフについては，ある意味での最適な順序付けを簡単な手続きによって求める方法が提案されている．すなわち正方形グリッド ($m\times m$) の構造を持つ無向グラフ（節点の数は $n=m^2$）においては，上記の最小次数法あるいは George の方法[102),104)]によって得られる順序付けは $m^2\log m$ に比例する個数の非零要素を発生するが，この結果はメッシュの大きさに対する非零要素の数の増加の割合という意味で最適である[103)]．但し，非零要素の絶対的な個数を最小にする問題は未解決である．

無向グラフにおける節点縮約の順序付け問題に関しては，最近，理論的に興味があり，実用的にもかなり有効な成果が得られている．先ず，スーパーグラフ $G[\alpha]=G(V, E\cup F)$ において $|F|=0$，すなわち新しい非零要素を発生させない順序付け α——これを**完全順序** (perfect ordering) という——が存在するようなグラフ $G=(V, E)$ は如何なる性質を持つか，というのが理論的に興味ある問題である．これに対して Rose は

定 理: グラフ G に完全順序が存在するための必要十分条件は G が**三角化グラフ**である．

という結果を導いた[98),100)]．また，グラフが三角化グラフであるか否かを枝(行列の非零要素)の数に比例する手間で判定するアルゴリズムも最近発見された[107)]．

さて上記の定理によれば，最適のピボット順序を決定する問題は，与えられたグラフに最小数の枝を付加してこれを三角化グラフにする——**最小三角化** (minimum triangulation) という——問題と解釈できる．最小三角化について実用的な結果は何も得られていないが，これに準ずるものとして，

定義: グラフ G の節点のある順序付けを α——$G[\alpha]=G(V, E\cup F)$——とする．今，他の如何なる順序付け α'——$G[\alpha']=G(V, E\cup F')$——に対しても，$F'$ が F の真部分集合とならない時，α は**極小三角化順序** (minimal triangula-

tion ordering) と呼ばれる*.

という概念を導入することができる．これに関しては，与えられた順序付けが極小三角化順序であるための必要十分条件[99]，更にはこれを（節点の数)×(枝の数）に比例する手間――前述の(A),(B),(C)に基づくものよりも低い限界を与えている――で求める算法[105)~107)]など興味ある結果が得られている．

極小三角化順序と(A),(B),(C)の規準に基づく順序付けの最適解への近さを比較すると，これはグラフの性質に大きく依存し，どちらが良いかは単純には判断できない．最近，この両者の長所を生かしてより良い順序付けを求めることを狙った研究もある[71)]．

三角化グラフの概念を有向グラフに拡張すること[101)]も興味ある問題であるが，無向グラフの場合に比べて，問題が急に複雑になり，明確な理論体系は今のところ得られていない．

5. データ構造とアクセス手法

スパース行列処理プログラムの性能は，使用するメモリーと計算に要するCPU時間で評価される．この2つの量をできるだけ少くすることが必要であるが，一般にメモリー量とCPU時間は両立しない性格をもっており，いかにバランスをとるかが問題である．そのためには，アルゴリズムの選択，スパース行列を表現するデータ構造とデータ構造へのアクセス手法の設計がきめてとなる．

本章ではスパース行列処理における基本的なデータ構造について説明するが，この問題は計算機のアーキテクチャ，プログラミング技術と密接な関係にあり，実際にプログラムを作成する時にはそれらに依存する面があることを注意しておく．

問題の規模が大きくなるとメインメモリーには入りきらなくなるため，外部記憶装置を使用せざるを得ない場合がある．このような場合には，ファイルの編成方法，データのアクセス順序とアクセス手法の設計が非常に重要な問題となり，高度のプログラミング技術が必要となる．現在，構造解析プログラムや線形計画法プログラムで大規模な問題（行列の次数が数千～数万）を対象とするものは外部記憶装置を使用しているが，対象とする問題の性質と使用する計算機システムにかなり依存した設計が行われているのが現状であり**，一般的な議論の対象とはなりにくいので，本章ではそれらの説明は行わない．本章のデータ構造の説明では，物理的なメモリー階層を意識しないことにする．すなわち，データ構造のマッピングとそれに対するアクセスは十分大きいアドレス空間で行われているものとする．

スパース行列処理で使われるデータ構造は，基本的には次の3種類に分類される．

- ビットマップ方式
- 行/列インデクス方式
- リスト方式

これらは基本的な考え方の分類をしたものであって，必ずしもこれらの方式がそのまま実際に使用されているというわけではない．実際には，問題の性質，問題の定式化，使用する計算機システム，設計目標などに従って総合的な判断の下に方式の設計が行われ，これらを変形した方式や混合した方式がとられることがある．以下，3つの方式についてその説明をする．

（1） ビットマップ方式

図-10 のように行列 A が与えられている時，行列の非零要素，零要素にそれぞれビット1，ビット0を対応してできるブール行列がこの方式の基本となっている．**図-11**（次頁参照）の表現方法が**ビットマップ方式**と呼ばれているものである．BITMAP は，ビットストリングの配列で，各要素が各行のスパース構造をあらわしている．行列の非零要素の値は，配列 VALUE に A の行方向順につめた形で入っており，配列 ROWINDEX によって各行の先頭の非零要素の場所が与えられる．ビットマップ方式は構造が簡単であるが，行列の次数が大きくなると BITMAP のためのエリアが大きくなる上，非零要素をサーチするための時間が多く必要となるため，有利な方法とはいえない．また，VALUE はつめた形で入っており，一般に処理中新しい非零要素が発生するので，新しい場所

$$A=\begin{pmatrix} 5 & 0 & 2 & 0 \\ 1 & 3 & 0 & 0 \\ 0 & 8 & 7 & 0 \\ 4 & 0 & 0 & 9 \end{pmatrix}$$

図 10 行列 A

* 最小三角化順序は極小三角化順序であるが，逆は必らずしも成立しない．

** OS のデータ管理機能が通常提供しているアクセス手法では，必ずしも能率の良い処理ができない場合があるので，問題に適したチャネルプログラムを作るなど特殊なアクセス法を設計することがある．

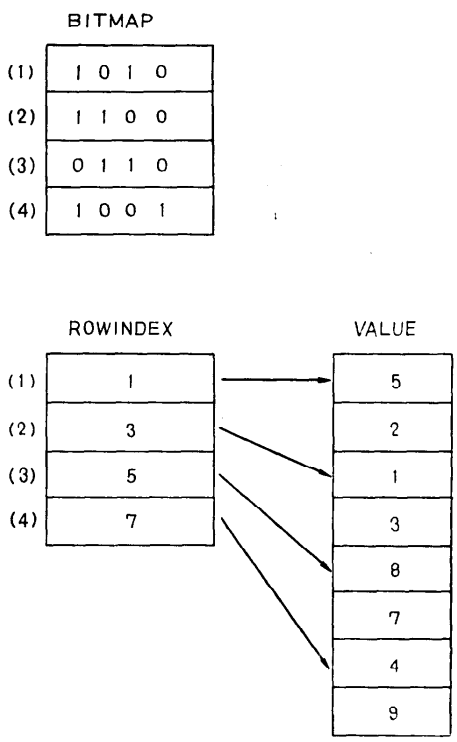


図-11 ビットマップ方式

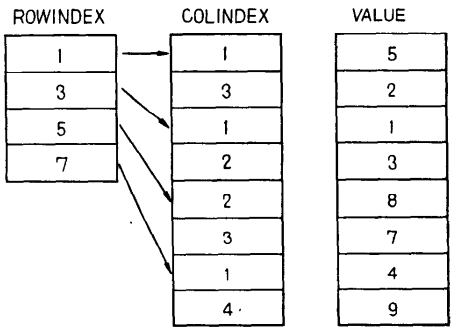


図-12 行/列インデクス方式

に正しい順序でつめ直す操作が必要である．

（2） 行/列インデクス方式[109)~111)]

この方式は，非零要素の列番号と要素値をつめた形で格納するものである．図-10の行列が与えられた時，**図-12**のようなデータ構造で表現する方式を**行/列インデクス方式**という．VALUEにはAの行方向順にその非零要素値が入っており，COLINDEXにはVALUEの各要素に対応してその列番号が入っている．ROWINDEXが各行の先頭の非零要素の場所を

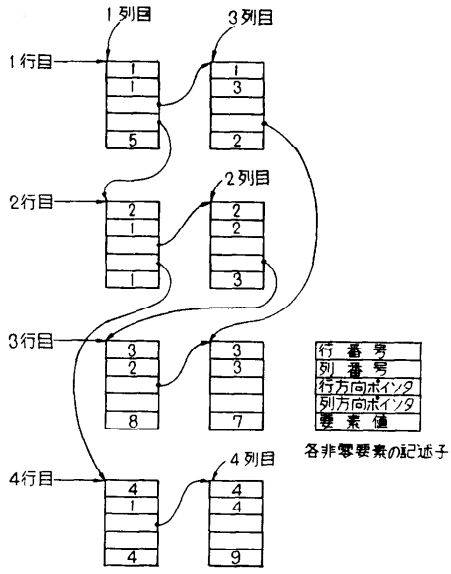


図-13 リスト方式

与える．この方式は比較的多く採用されており，そのプログラム設計も比較的簡単であるが，一般に新しい非零要素が発生するので，新しい場所にVALUE，COLINDEXを正しい順序でつめ直す操作が必要である．

（3） リスト方式[13),111),118)]

リスト方式は，**図-13**のように各非零要素に対し記述子を設け，行列のスパース構造を記述子のリスト構造で表現する方法である．記述子は，行番号，列番号，要素値，行方向ポインタ，列方向ポインタによって構成される．行方向，列方向ポインタはそれぞれ，同じ行，列における次の記述子の場所を与える．次の記述子がない時は，それを意味する値またはマークが入っている．各行，各列における記述子チェインの先頭場所は，ポインタの配列によって与えられる．ポインタは絶対アドレスまたは，記述子がまとめて格納されているエリアにおけるオフセット値である．この方式は，新しい非零要素が発生した場合でも，その記述子の場所の確保とポインタチェインのつけかえだけで処理できることにあり，スパース行列処理のためには便利なデータ構造であるが，ポインタのための場所が必要であるため，行/列インデクス方式に比べれば余計メモリーが必要である．**図-13**の場合，要素値は記述子中に入れられているが，要素値は別の場所にあって，記述子にその場所をさすポインタを入れておく

方式もある．また，処理のつごう上，行，列方向に逆方向のポインタチェインを追加する方式もある．実際のプログラムでは，問題の性質と処理のつごうにより，図-13の構造を基本としていろいろな変種が使われている．

以上述べた方式は，行列のスパース構造がランダムな場合の基本的なデータ構造とアクセス手法を与えるものであるが，行列が規則性をもっている場合（例えば，帯行列，ブロック三角行列）には，その規則性を有効に使うことにより更に能率の良いデータ構造を設計することができる．帯行列の特殊性を利用した方式を特に**帯インデクス法** (band indexing schema)[111] ということがある．帯インデクス法では，帯の内側は密であると仮定して処理を行うものが多いが，内側がスパースになっている場合は，内側に対してもスパース処理をすることにより更に能率を高めることができる．図-3, 4のような規則性をもっている行列の場合は行列の適当なブロック毎に消去を行えばよい．したがって，大規模な行列の場合は，1つのブロックをメインメモリ内に，他のブロックを外部記憶装置におき，演算をしながらブロック毎に I/O をすることにより，少ないメインメモリ量で能率の良い処理を行うことができる．偏微分方程式の数値解法や，構造解析などであらわれる大規模な帯行列，準帯行列はこのような技法で解かれる．

スパース行列処理では，新しい非零要素の発生あるいは演算の数を少なくするようにピボット順序の決定を行うことが多い．このためには行/列の入れかえが必要となるが，実際にメモリ内でデータの移送をしてつめなおす操作をすることはばかげたことである．ふつう，行/列の入れかえは**置換ベクトル** (permutation vector) と呼ばれる一次元配列を使って，ピボット番号と行/列番号の対応関係を記憶しておくことにより行われる．置換ベクトルの第 i 番要素は，i 番目のピボットに対応する行/列番号を与える．

6. コード発生方式

スパース行列を処理する方式はいろいろあるが，方式は対象とする問題の性格によってきめられるものである．例えば，1度だけ解けば良い場合と，同じスパース構造を持ち非零要素の値が異なる行列を多数回解く必要のある場合とでは，その処理方式はかなり異なる．スパース行列処理が特に威力を発揮するのは後者の場合である．

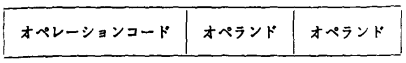


図-14 擬似コード

処理方式は大きく分けて次の2つがある．

- 解釈実行方式 (interpretive method)[113]
- コード発生方式 (code generation method)[37],[114]

解釈実行方式は，基本的にはデータ構造をみながら計算を実行していく方式で，通常作られるプログラムはこれに属する．行列のスパース構造をあらわすデータ構造をみながら実行する方式が普通であるが，まずスパース構造をあらわすデータ構造をもとに消去または分解の過程をシミュレーションして，計算の順序をあらわす別のデータ構造を発生し，発生されたデータ構造をみながら実行する方式がある．発生されるデータ構造は次の2種類のものがある．

- 擬似コード (pseudo code)[113]
- アドレス表 (address table)[109]

擬似コードは**図-14**のように，オペレーション部，オペランド部（オペランドの数は2または3が普通）から成り，オペレーションは普通＋，－，＊，÷であるが，それ以外のオペレーションを定義することもある．オペランド部は非零要素の場所を与える．

ガウス消去法などの単純な計算の場合は，1つの要素に対する演算手順はきまっているので，擬似コードにおけるオペレーション部を省略することができる．これを**アドレス表**という．解釈実行方式は，比較的単純な問題の場合には有効な方式であるが，より複雑な問題――例えば非線形回路網の解析――の場合にはあまり計算スピードの向上は望めない．非線形回路解析の場合，**図-15**のように非線形方程式解法ループ，数値積分計算ループ，モンテカルロ法によるパラメータ変更のループが3重にネストした計算構造になっており，全体で数千～数万回同じ構造をもった連立方程式

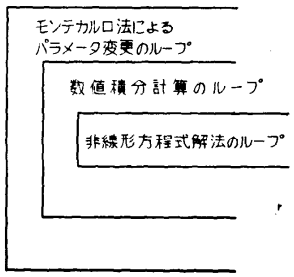


図-15 非線形回路網解析の計算構造

を解くことが必要になるため，1回当りの計算時間をできるだけ縮めなければならない．また，この問題の場合，行列の非零要素として ±1，定数，微分オペレータ，非線形オペレータがあらわれるので特別の処理が必要となる．このような問題を能率よく解くために開発された方式が**コード発生方式**[37),114)]と呼ばれるもので，まずスパース構造をあらわすデータ構造をもとに，消去または分解の過程をシミュレーションして，計算手順をプログラムとして発生する．発生されるプログラムには，ループやブランチがまったくないのが特徴であり，実行スピードは非常に速い．発生される言語は，**高級言語** (FORTRAN など)[37)] または**機械語**[114)] である．高級言語で発生を行う場合には，実行するためにコンパイラによる処理が必要となるが，通常のコンパイラでは処理可能なステートメント数に制限があるだけでなく非常に時間がかかる．このため，専門のコンパイラを開発することが必要となる[37)]．一方，機械語を使う場合には，発生されたプログラムがそのまま実行可能であるため，コンパイルなどの手続が不用となるだけでなく，レジスタ割付けの最適化を直接行うことができ非常に能率が良い．この方式を特に**機械語発生方式** (machine code generation method) という．

図-15 のような計算構造をもつ問題をコード発生方式によって処理する方法を電気回路網の場合を例にとって説明する．

電気回路網の場合，回路の状態を記述する方程式はキルヒホフの電流則，電圧則，素子特性（線形，非線形），微分方程式の組から成る．これをまとめて，$Ax=b$ の形に書くことができる．A は図-16 のように，その非零要素が ±1，定数，パラメータ，微分オペレータ，非線形オペレータから成っているオペレータの行列である．行列 A を特に**スパースタブロー**と呼ぶ[114)~116)]．この方程式を数値的に解くためには，数値

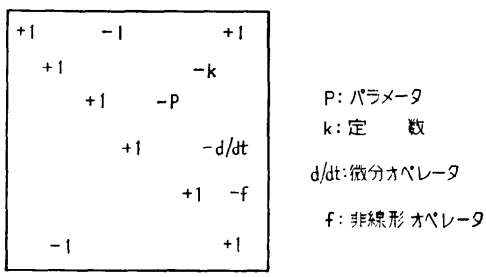


図-16 スパースタブロー

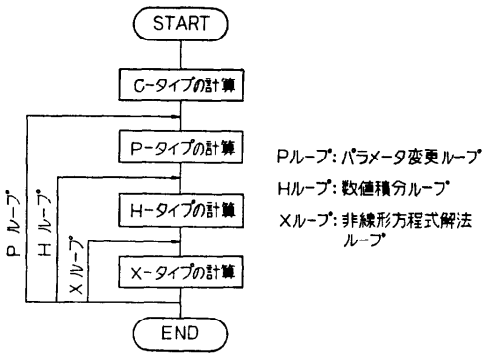


図-17 コードの実行

積分公式，非線形方程式の解法を適用することが必要である．全体の計算構造は図-15 のようになっているので，計算の能率を上げるためには各ループの内側では本質的に必要な計算だけを行うようにすることがきめてとなる．計算構造に対応して計算の内容を次のように分類することができる[114)~117)]．

C—タイプ（定数同志だけからなる計算）
P—タイプ（パラメータに関係する計算）
H—タイプ（数値積分に関係する計算）
X—タイプ（非線形方程式の解法に関係する計算）

このように分類されたコードが発生された場合，その計算構造は図-17 のようになる．

分類されたコードは，基本的には次のように発生される．まず，各非零要素をそのタイプによって分類することが必要である．非零要素を例えば

+1, −1：接続関係をあらわす要素
C ：定数
P ：パラメータ
d/dt ：微分オペレータ
f ：非線形オペレータ
H ：数値積分に関係する要素
X ：非線形方程式解法に関係する要素

と分類し，各非零要素のタイプをあらわすデータ構造を設ける．スパース構造をリスト方式であらわせば，例えば図-18（次頁参照）のような要素記述子となる．まず，スパース構造をあらわすデータ構造をみながら消去または分解のシミュレーションを行う．シミュレーション段階では，新しく発生する非零要素の決定とタイプコードの変更を行う．この場合，結果の値が簡単にわかるもの（例えば −1*1 など）は，シミュレーション段階で行い，コード発生で無駄な命令列が発

タイプコード
行番号
列番号
値の場所
行方向ポインタ
列方向ポインタ

図-18 要素記述子

生されないようにする．この後，コード発生段階に入る．コード発生段階では，要素値が入る場所の決定，結果が入るコードの種類（C-タイプ，X-タイプ等）の決定，命令列の発生，バッファへの書き込み，外部記憶装置への出力を行う．

発生されるコードの数学的アルゴリズムとしては，ガウス消去法あるいはクラウト分解法がふつう採用されるが，コードの実行能率としてはクラウト分解法が良い．クラウト分解法の場合はレジスタ中で内積累和を行うことができ，ガウス消去法の場合よりもLOAD, STORE 命令が少なくなるからである．1つの非零要素に対する演算を発生する場合，その演算はさらにいくつかのコードに分割される．たとえば，X-タイプの結果をもつ要素に対する演算は，一般には次のようになっている．

X-タイプの結果$=D*\{(C$-タイプの計算$)+(P$-タイプの計算$)+(H$-タイプの計算$)+(X$-タイプの計算$)\}$

D は，± 1 または，上のいずれかのタイプをもった値である．したがって，タイプに従ってコードを分割して発生することにより，内側の計算ループでは本質的に必要な計算だけが行われることになる．

参考文献

(既出分への追加)

37) F. G. Gustavson, W. Liniger and R. A. Willoughby: Symbolic Generation of an Optimal Crout Algorithm for Sparse Systems of Linear Equations, J. ACM, Vol. 17, No. 1, pp. 87～109 (1970).

38) R. A. Willoughby: Some Pivot Considerations in Direct Methods for Sparse Matrices, Proc. Mexico International IEEE Conf. on Systems, Networks and Computers, Oaxtepec, Mexico (1971).

39) D. Thurnau: Algorithm 195. Bandsolve, Comm. ACM, Vol. 6, p. 441 (1963).

40) R. S. Martin and J. H. Wilkinson: Handbook Series Linear Algebra. Symmetric Decomposition of Positive Definite Band Matrices, Numer. Math., Vol. 7, pp. 355～361 (1965).

41) R. P. Tewarson: Solution of Linear Equations with Coefficient Matrix in Band Form, BIT, Vol. 8, pp. 53～58 (1968).

42) G. Cantin: An Equation Solver of Very Large Capacity, Internt. J. Numer. Methods Eng., Vol. 3, pp. 379～388 (1971).

43) R. Rosen: Matrix Bandwidth Minimization, Proc. 23rd Nat. Conf. ACM Publ., pp. 585～595, Brandon Systems Press, Princeton (1968).

44) F. A. Akyuz and S. Utku: An Automatic Relabeling Scheme for Bandwidth Minimization of Stiffness Matrices, AIAA J., Vol. 6, pp. 728～730 (1968).

45) R. P. Tewarson: Sorting and Ordering Sparse Linear Systems, in 11), pp. 151～168 (1971).

46) R. P. Tewarson: Row Column Permutation of Sparse Matrices, Comput. J., Vol. 10, pp. 300～305 (1967).

47) P. Rabinowitz: Applications of Linear Programming to Numerical Analysis, SIAM Rev., Vol. 10, pp. 121～159 (1968).

48) Y. T. Chen: Permutation of Irreducible Sparse Matrices to Upper Triangular Form, IMA J. Vol. 10, pp. 15～18 (1972).

49) D. V. Steward: Partitioning and Tearing Systems of Equations, SIAM J., Numer. Anal., Vol. 2, pp. 345～365 (1965).

50) J. A. George: Block Elimination of Finite Element Systems of Equations, in 12), pp. 101～114 (1972).

51) D. J. Rose and J. R. Bunch: The Role of Partitioning in the Numerical Solution of Sparse Systems, ***ibid***, pp. 177～190 (1972).

52) V. K. Wenke: Praktishe Anwendung Linearer Wirtschaffs-Modelle, Unternehmensforshung, Vol. 8, pp. 33～46 (1964).

53) B. A. Carré: The Partitioning of Network Equations for Block Interation, Comput. J., Vol. 9, pp. 84～97 (1966).

54) R. B. Marimont: System Connectivity and Matrix Properties, Bull. Math. Biophs. Vol. 31, pp. 255～274 (1969).

55) J. Eufinger: Eine Untersuchung zur Auflösung Magerer Gleichungssysteme, J. Reine Angewandte Math., Vol. 245, pp. 208～220 (1970).

56) E. C. Ogbuobiri, W. F. Tinney and J. K. Walker: Sparsity Directed Decomposition for Gaussian Elimination on Matrices, IEEE Trans. PAS, Vol. PAS-89, pp. 141～155 (1970).

57) 有吉，白川，尾崎: Sparse アドミタンス行列の

計算手順に関する一方法，信学論（A），Vol. 53-A, No. 11, pp. 612～619 (1970).
58) 翁長：Sparse 行列の消去系列最適化のグラフ理論的考察，信学会回路システム理論研資，Vol. CT 73-1 (1973).
59) J. Carpentier: Ordered Elimination, Proc. Power Systems Comput. Conf., London (1963).
60) H. Edelman: Ordered Triangular Factorization of Matrices, ***ibid.*** (1963).
61) R. P. Tewarson: Solution of a System of Simultaneous Linear Equations with a Sparse Coefficient Matrix by Elimination Methods, BIT, Vol. 7, pp. 226～239 (1967).
62) W. R. Spillers and N. Hickerson: Optimal Elimination for Sparse Symmetric Systems as a Graph Problem, Quart. Appl. Math., Vol. 26, pp. 425～432 (1968).
63) E. C. Ogbuobiri: Dynamic Storage and Retrieval in Sparsity Programming, IEEE Trans. PAS, Vol. PAS-89, pp. 150～155 (1970).
64) U. Bertelé and F. Brioshi: On the Theory of the Elimination Process, J. Math. Anal. Appl., Vol. 35, pp. 48～57 (1971).
65) H. Y. Hsieh and M. S. Ghausi: On Optimal Pivoting Algorithms in Sparse Matrices, IEEE Trans. CT, Vol. CT-19, pp. 93～96 (1972).
66) H. Y. Hsieh and M. S. Ghausi: A Probabilistic Approch to Optimal Pivoting and Prediction of Fill-in for Random Sparse Matrices, ***ibid.***, pp. 329～336 (1972).
67) 坂本，白川，尾崎：スパース連立方程式におけるピボット操作の順序づけ，情報処理，Vol. 13, pp. 154～160 (1972).
68) 喜屋武，白川，尾崎：線形連立方程式のスパーシティとこれに付随するバイパータイト・グラフ，信学大全，p. 6 (1973).
69) M. Nakhla, K. Singhal and J. Vlach: An Optimal Pivoting Order for the Solution of Sparse Systems of Equations, IEEE Trans. CAS, Vol. CAS-21, pp. 222～225 (1974).
70) H. Y. Hsieh: Pivoting-Order Computation Method for Large Random Sparse Systems, ***ibid.***, pp. 225～230 (1974).
71) J. Jess: Sparse Matrix Pivoting—The Global Effect of Local Criteria, Proc. Internt. Symp., on Circuits and Systems, pp. 127～130 (1975).
72) L. J. Larson: A Modified Inversion Procedure for Product Form of Inverse in Linear Programming Codes, Comm. ACM, Vol. 5, pp. 382～383 (1962).
73) D. M. Smith and W. Orchard-Hays: Computational Efficiency in Product Form LP Codes, in "Recent Advances in Mathematical Programming (R. L. Graves and P. Wolfe, eds.), McGraw-Hill, New York", pp. 211～218 (1963).
74) J. C. Dickson: Finding Permutation Operations to Produce a Large Triangular Submatrix, 28th Nat. Meeting OR Soc. America, Houston (1965).
75) R. P. Tewarson: On the Product Form of Inverse of Sparse Matrices, SIAM Rev., Vol. 8, pp. 336～342 (1966).
76) R. P. Tewarson: On the Product Form of Inverse of Sparse Matrices and Graph Theory. ***ibid.***, Vol. 9, pp. 91～99 (1967).
77) W. Orchard-Hays: Advanced Linear Programming Computing Techniques, McGraw-Hill, New York (1968).
78) N. Sato and W. F. Tinney: Techniques for Exploiting the Sparsity of the Network Admittance Matrix, IEEE Trans. PAS, Vol. PAS-82, pp. 944～950 (1963).
79) W. F. Tinney and J. W. Walker: Direct Solutions of Sparse Network Equations by Optimally Ordered Triangular Factorization, Proc. IEEE, Vol. 55, pp. 1801～1809 (1967).
80) H. Edelman: Massnahmen zur Reduktion des Rechenaufwands bei der Berechnung Grosser Elektrisher Netze, Elektron. Rechenanlangen, Vol. 10, pp. 118～123 (1968).
81) A. Chang: Application of Sparse Matrix Methods in Electric Power System Analysis, in 10), pp. 113～121 (1969).
82) W. F. Tinner: Comments on Sparsity Techniques for Power System Problems, ***ibid.***, pp. 25～34 (1969).
83) R. D. Berry: An Optimal Ordering of Electronic Circuit Equations for a Sparse Matrix Solution, IEEE Trans. CT, Vol. CT-18, pp. 40～50 (1971).
84) R. S. Norin and C. Pottle: Effective Ordering of Sparse Matrices Arising from Nonlinear Electrical Networks, IEEE Trans. CT, Vol. CT-18, pp. 139～145 (1971).
85) A. M. Erisman: Sparse Matrix Approach to the Frequency Domain Analysis of Linear Passive Electrial Networks, in 12), pp. 31～40 (1972).
86) 篠田，梶谷：節点アドミタンス行列のスパーシティとその応用，信学論（A），Vol. 56-A, No. 9, pp. 544～550 (1973).
87) 相原，坪井：直交化法による回路解析の一手法とそのスパース処理，信学論（A），Vol. 57-A, No. 9, pp. 655～662 (1974).
88) 可児，大附：設計自動化におけるグラフ理論と

組み合せ算法(1), 情報処理, Vol. 16, No. 5, pp. 440～449 (1975).
89) A. L. Dulmage and N. S. Mendelsohn: A Structure Theory of Bipartite Graphs of Finite Exterier Dimension, Trans. Roy. Soc. Can., Sec. III, Vol. 53, pp. 1～13 (1959).
90) A. L. Dulmage and N. S. Mendelsohn: Two Algorithms for Bipartite Graphs, SIAM J., Vol. 11, No. 1, pp. 183～194 (1963).
91) J. E. Hopcroft and R. M. Karp: A $n^{5/2}$ Algorithm for Maximum Matching in Bipartite Graphs, IEEE Conf. Record of the 12th Annual Symp. on Switching and Automata Theory, pp. 122～125 (1971).
92) R. Tarjan: Depth-First Search and Linear Graph Algorithms, SIAM J. Comput., Vol. 1, No. 2, pp. 146～160 (1972).
93) F. Harary: Sparse Matrices and Graph Theory, in 11), pp. 139～150 (1971).
94) F. Harary: Sparse Digraphs—Classification and Algorithms, IFIP Conf., Ljubljana, Yugoslavia (1971).
95) A. Lempel and I. Cederbaum: Minimum Feedback Arc and Vertex Sets of a Direct Graph, IEEE Trans. CT, Vol. CT-13, No. 4, pp. 399～403 (1966).
96) L. K. Cheung and E. S. Kuh: The Bordered Triangular Matrix and Minimum Essential Sets of a Digraph, IEEE Trans. CAS, Vol. CAS-21, No. 5, pp. 633～639 (1974).
97) S. Parter: The Use of Linear Graphs in Gauss Elimination, SIAM Rev. Vol. 3, pp. 119～130 (1961).
98) D. J. Rose: Triangulated Graphs and the Elimination Process, J. Math. Anal. Appl., Vol. 32, pp. 597～609 (1970).
99) T. Ohtsuki, L. K. Cheung and T. Fujisawa: On Minimal Triangulation of a Graph, Memo., No. ERL-M 351, Univ. of Calif., Berkeley (1972).
100) D. J. Rose: A Graph-Theoretic Study of the Numerical Solution of Sparse Positive Definite Systems of Linear Equations, in "Graph Theory and Computing (R. Read, ed.), Academic Press, New York", pp. 183～217 (1973).
101) L. Haskins and D. J. Rose: Toward Characterization ot Perfect Elimination Digraphs, SIAM J. Comput., Vol. 10, pp. 345～363 (1973).
102) J. A. George: Nested Dissection of a Regular Finite Element Mesh, SIAM J. Numer. Anal., Vol. 10, pp. 345～363 (1973).
103) A. J. Hoffman, M. S. Martin and D. J. Rose: Complexity Bounds for Regular Finite Difference and Finite Element Grids, ***ibid.*** pp. 364～369 (1973).
104) D. J. Rose and G. F. Whitten: Automatic Nested Dissection, Proc. ACM Annual Symp., pp. 82～88 (1974).
105) T. Fujisawa and H. Orino: An Efficient Algorithm of Finding a Minimal Triangulation of a Graph, Proc. IEEE Internat. Symp. on Circuits and Systems, pp. 172～175 (1974).
106) T. Ohtsuki: A Fast Algorithm for Finding on Optimal Ordering for Vertex Elimination on a Graph, SIAM J. Comput., to appear (1975).
107) D. J. Rose and R. Tarjan: Algorithm Aspects of Vertex Elimination on Graphs, Memo. ERL-M 483, Univ. of Calif., Berkeley (1974).
108) U. W. Pooch and A. Nieder: A Survey of Indexing Techniques for Sparse Matrices, Computing Surveys, Vol. 5, No. 2, pp. 109～133 (1973).
109) 熊本, 大附: スパースな係数行列をもつ連立一次方程式の解析プログラム, 情報処理学会第11回大会 (1970).
110) 高橋: 正値対称行列のスパース処理について, 情報処理学会第14回大会, pp. 169～170 (1973).
111) F. G. Gustavson: Some Basic Techniques for Solving Sparse Systems, in 12), pp. 41～52 (1972).
112) D. E. Knuth: The Art of Programming, Vol. 1, Addison Wesley, Reading, Mass. (1968).
113) H. B. Lee: An Implementation of Linear Equations, in 10), pp. 75～83 (1969).
114) G. D. Hachtel, R. K. Brayton and F. G. Gustavson: The Sparse Tableau Approach to Network Analysis and Design, IEEE Trans. CT, Vol. CT-18, pp. 101～113 (1971).
115) K. Kawakita and T. Ohtsuki: NECTAR 2-A Circuit Analysis Program Based on Piecewise-Linear Approach, IEEE Internat. Symp. on Circuits and Systems, pp. 92～95, (1975).
116) W. T. Weeks, et al: Algorithm for ASTAP-A Network Analysis Program, IEEE Trans. CT, Vol. CT-20, pp. 628～634 (1973).
117) G. E. Hachtel: Vector and Matrix Variability Type in Sparse Matrix Algorithms, in 12), pp. 53～64 (1972).
118) P. C. Kettler and R. L. Weil: An Algorithm to Provide Structure for Decomposition, in 10), pp. 11～17 (1969).
119) M. Larcombe: A List Processing Approach to the Solution of Large Sparse Sets of Matrix Equations and the Factorization of

the Overall Matrix, in 11), pp. 25～40 (1971).
120) E. Nuding and I. Kahlert-Warmbold: A Computer Oriented Representation of Matrices, Computing Vol. 6, pp. 1～8 (1970).
121) A. Jennings and A. Truff: A Direct Method for the Solution of Large Sparse Symmetric Simultaneous Equations, in 11), pp. 97～104 (1971).
122) A. Jennings: A Sparse Matrix Scheme for the Computer Analysis of Structures, Internat. J. Comput. Mach. Vol. 2, pp. 1～21 (1968).
123) C. W. MaCormick: Application of Partially Bordered Matrix Methods to Structural Analysis, in 10), pp. 155～158 (1969).
124) R. J. Allwood: Matrix Methods of Structural Analysis, in 11), pp. 17～24 (1971).
125) E. L. Palacol: The Finite Element Method of Structural Analysis, in 10), pp. 101～106 (1969).
126) S. J. Fenves, R. D. Logcher and S. P. Mauch: STRESS: A Reference Manual, A Problem Oriented Computer Language for Structural Engineering, MIT Press, Cambridge, Mass. (1965).
127) W. Weaver: Computer Programs for Structural Analysis, Van Nostrand Co., Inc., Princeton, N. J. (1967).
128) E. L. Wilson: SOLID SAP-A Static Solid Structures, Univ. of California Structural Engineering Laboratory Report No. UC-SESM 71-19 (1970).
129) The ASTRA System, The Boeing Company, D 2-125179-8 (1972).
130) B. Schumaker: An Introduction to ICES, Dept. of Civil Eng. Res. Rep. R 67-47, MIT (1967).

(昭和 50 年 10 月 31 日受付)