研究ノート

連立一次方程式の解法について

木　村　　等
高　橋　知　子

I　はじめに

連立一次方程式 $Ax=b$（A：正方行列，x，b：ベクトル）を計算機で解くとき，唯一の解を持つ場合は従来の Gauss 消去法と，それよりは良い結果を出す筈の反復改良法がある。これとはまったく別に，もともとは逆行列を求めるためのアルゴリズムから出発したらしいが，それの適用範囲が実は拡張できて，rank 落ちしたときの正方形行列や，長方形行列の一般逆行列を求め，一つのある解を求めることが考えられる。各々の方法の紹介とあるデータについて計算機を使用したときの結果を比較する。

記号

A，B などで m 行 n 列の行列を表わし，ij 成分を a_{ij}，b_{ij} などで表わす。x，y などでベクトルを，α，a，t_2，r などは適当なスカラー量とする。A* は A が実行列のときは転置行列を，複素行列のときは転置共役行列を表わす。I で単位行列を，A^{-1} で行列式がゼロでないときの逆行列を，A^{+} で行列式がゼロのとき等の一般逆行列を示す。det(A) で A の行列式をあらわす。

行列のノルム

説明の都合上，近さを測るために2種類のノルムを利用しているが，両者の関係を述べておく。

$A=(a_{ij})$　$i=1,\cdots m$; $j=1,\cdots n$

$\mathrm{rank}(A)=r$

$\lambda_i(A^*A)$　$i=1,\cdots, r$　は A^*A の固有値とする。

各 λ_i は正である。大きさの順に $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_r$ とするとき，

$\|A\|_E = (\sum_{i,j} |a_{ij}|^2)^{\frac{1}{2}}$ ：コークリッド・ノルム

$\|A\|_S = (\lambda_1(A^*A))^{\frac{1}{2}}$ ：スペクトラル・ノルム

と定義する。

$B=(b_{ij})$ の trace を $\mathrm{trace}(B)=\sum_{i=1}^{n} b_{ii}$ とあらわすと

$$\mathrm{trace}(A^*A)=\sum_{i,j}|a_{ij}|^2=\|A\|_E^2$$

また trace は固有値の和でもあるから，

$$\mathrm{trace}(A^*A)=\sum_{i=1}^{r}\lambda_i(A^*A)$$

$$\therefore \lambda_1(A^*A) \leq \mathrm{trace}(A^*A) \leq r\cdot\lambda_1(A^*A)$$

$$\therefore \|A\|_S^2 \leq \|A\|_E^2 \leq r\cdot\|A\|_S^2$$

$$\therefore \frac{1}{\sqrt{r}}\|A\|_E \leq \|A\|_S \leq \|A\|_E$$

従って，2番目の不等号より，ユークリッド・ノルムで測ってゼロに近いものは，スペクトラル・ノルムで測ってもゼロに近い。1番目の不等号より，ユークリッド・ノルムで測って大きいものは，スペクトラル・ノルムで測っても大きい。すなわち，どちらのノルムで収束性を論じてあっても結果は同である。

行列式の微分

$A=(a_{ij})$ $i=1,\cdots n$; $j=1,\cdots n$

$a_{ij}=a_{ij}(x)$

のとき，行列式の微分は次のようである。

$$\frac{d\det(A)}{dx}=\det\begin{pmatrix} a_{11}'(x) & a_{12}'(x) & \cdots & a_{1n}'(x) \\ a_{21}(x) & & \cdots & a_{2n}(x) \\ \vdots & & & \vdots \\ a_{n1}(x) & & \cdots & a_{nn}(x) \end{pmatrix}+\det\begin{pmatrix} a_{11}(x) & a_{12}(x) & \cdots & a_{1n}(x) \\ a_{21}'(x) & a_{22}'(x) & \cdots & a_{2n}'(x) \\ \vdots & \vdots & & \vdots \\ a_{n1}(x) & a_{n2}(x) & \cdots & a_{nn}(x) \end{pmatrix}$$

$$\cdots+\det\begin{pmatrix} a_{11}(x) & a_{12}(x) & \cdots & a_{1n}(x) \\ a_{21}(x) & a_{22}(x) & \cdots & a_{2n}(x) \\ \vdots & \vdots & & \vdots \\ a_{n1}'(x) & a_{n2}'(x) & \cdots & a_{nn}'(x) \end{pmatrix}$$

証明は，$\det(A)$ の計算をしてしまった後，x で微分した式が，$\frac{d\det(A)}{dx}$ の右辺の式になることを確かめることによって行われる。

II 反復改良法

係数行列の要素が対称であるとか，対角要素線にそって0以外の要素が帯状に並んでいるとかの特徴をそなえていず，まんべんなく0以外の要素が散らばっているとき，連立一次方程式 $A\boldsymbol{x}=\boldsymbol{b}$ を解くアルゴリズムで，ガウスの消去法にまさるものはこれまで知られていないと思われる。

ガウスの消去法の代数的な基礎は，係数行列Aの三角分解である。

〔定理1〕

n次正方行列 A のはじめの k行 k列からなる主小行列を A_k であらわす。$k=1,2,\cdots n-1$ に対して $\det(A_k)\neq 0$ と仮定すると，対角要素が1の下三角行列 $L=(l_{ij})$ と，上三角行列 $U=(u_{ij})$ で $LU=A$ となるものが一意に存在する。

（証明は大抵の教科書にあるので略す。）

従って，$A\boldsymbol{x}=\boldsymbol{b}$ は $LU\boldsymbol{x}=\boldsymbol{b}$ を解くこととなり，これは $L\boldsymbol{y}=\boldsymbol{b}$ から $\boldsymbol{y}$ を求め，$U\boldsymbol{x}=\boldsymbol{y}$ を解いて $\boldsymbol{x}$ を求めれば良いことを意味する。係数行列が上，下三角行列であることから，この2つの方程式を解くことはやさしい。なぜなら $L\boldsymbol{y}=\boldsymbol{b}$ を各要素に書き下すと，

$$\begin{cases} l_{11}y_1=b_1 & (1)\\ l_{21}y_1+l_{22}y_2=b_2 & (2)\\ l_{31}y_1+l_{32}y_2+l_{33}y_3=b_3 & (3)\\ \vdots\\ l_{n1}y_1+l_{n2}y_2+\cdots\cdots+l_{nn}y_n=b_n & (4) \end{cases}$$

(1)より $y_1=b_1/l_{11}$

(2)に(1)を代入して，$y_2=(b_2-l_{21}y_1)/l_{22}$

(3)に(1)(2)を代入して，……という形で順に y_1，y_2，y_3……y_n が求まる。

$U\boldsymbol{x}=\boldsymbol{y}$ についてもほぼ同様で，U が上三角であるため，x_n，x_{n-1}，x_{n-2}，……x_1 の順に求まる。

反復改良法は，1度 Gauss の消去法を行って得た答を，L，U をそのまま利用して高精度の解を求めようというものである。Gauss の消去法は三角分解をするのに殆どの時間を費すものであるから，1度求めた L，U を記憶装置に残しておいて利用するこの方法では時間を大して増やさずに，しかも最初の解が1桁でも正しければ，たいていの問題では計算機の精度一杯近くまで改良される。

概略を説明する。最初の解 $\boldsymbol{x}_1$ が求まれば，残差 $\boldsymbol{r}_1=\boldsymbol{b}-A\boldsymbol{x}_1$ を計算する。$\boldsymbol{x}_1$ は真の解

x に近い筈だから $b-Ax_1$ の計算はかなりきわどい。桁落ちの心配があるので，この残差計算を2倍精度で行なう。これがこの方法の key point である。r_1 が求まれば，$Az_1=r_1$ を解く。この時，最初の LU 分解を利用して z_1 を計算する。もし z_1 が正確に求まれば $x_2=x_1+z_1$ は $Ax=b$ の正確な解である。実際，

$$Ax_2=Ax_1+Az_1=b-r_1+r_1=b$$ である。

z_1 が正確でなく，2，3桁だけ求まるなら x_2 は普通 x_1 より精度が高くなる。（x_1 の有効数字の最初のq桁が正しいならば次の z_1 も q 桁正しいと考えられ，$x_1+z_1=x_2$ は約 $2q$ 桁正しく，$r_2=b-Ax_2$，$Az_2=r_2$，$x_3=x_2+z_2\cdots$ と，この方法を繰り返すごとに q 桁ずつ修正されることが証明されている。）計算機が有限桁計算であることから x_1, x_2, x_3, … は普通一定のベクトルにすぐ近づき，これは $Ax=b$ の解で，1倍精度では完全に精密である。

筆者の興味は一般逆行列の方にあるので，この証明はここには紹介しない。あとの計算例で，実際に解が改良される例をあげておいた。詳しいことは G.E. Forsyth, C.B.Moler の Computer Solution of Linear Algebraic Systems. (Prentice Hall. 1967年) に載っている。

III Frame の方法

n 次行列 A の余因子行列 adj(A)，行列式 det(A)，逆行列 A^{-1}，固有多項式 $f(\lambda)$ をいちどきに計算する次のような方法がある。

固有多項式を，c_i をスカラーとして，

$$f(\lambda)=(-1)^n\det(\lambda I-A)=(-1)^n[\lambda^n-c_1\lambda^{n-1}-c_2\lambda^{n-2}\cdots-c_i\lambda^{n-i}\cdots-c_n]$$

とおく。

行列 adj$(\lambda I-A)$ の ij 要素は，$\lambda I-A$ の $(n-1)$ 次の小行列式であるから，高々 $(n-1)$ 次の多項式である。故に，

$$a_{ij}^{(0)}\lambda^{n-1}+a_{ij}^{(1)}\lambda^{n-2}+a_{ij}^{(2)}\lambda^{n-3}+\cdots+a_{ij}^{(n-2)}\lambda+a_{ij}^{(n-1)}$$

として，同じ次数の係数を集めて

$$A_0=(a_{ij}^{(0)}),\ A_1=(a_{ij}^{(1)}),\ \cdots,\ A_{n-1}=(a_{ij}^{(n-1)})$$

なる n 次行列の形に書ける。故に，

$$\text{adj}(\lambda I-A)=\lambda^{n-1}A_0+\lambda^{n-2}A_1+\cdots+\lambda^{n-i-1}A_i+\cdots+\lambda A_{n-2}+A_{n-1} \qquad (5)$$

この時，次のことが成立する。

〔定理2〕

i $A_0=I$

ii $\mathrm{trace}(AA_{k-1})=kc_k \quad (k=1, 2, \cdots, n)$

iii $A_k=AA_{k-1}-c_kI \quad (k=1, 2, \cdots, n-1)$ (6)

iv $AA_{n-1}=c_nI$

＜証明＞

余因子行列 adj(A) の定義と各成分ごとの比較から，

$$(\lambda I-A)\cdot \mathrm{adj}(\lambda I-A)=\det(\lambda I-A)\cdot I \tag{7}$$

であることがわかる。従って，

$$(\lambda I-A)(\lambda^{n-1}A_0+\lambda^{n-2}A_1+\cdots+A_{n-2}+A_{n-1})=(\lambda^n-c_1\lambda^{n-1}\cdots-c_n)\cdot I$$

$$\therefore \lambda^nA_0+\lambda^{n-1}(A_1-AA_0)+\cdots+\lambda^2(A_{n-2}-AA_{n-3})+\lambda(A_{n-1}-AA_{n-2})-AA_{n-1}$$
$$=\lambda^nI-\lambda^{n-1}c_1I-\cdots-c_nI$$

両辺を比べて

$$A_0=I$$

$$A_k-AA_{k-1}=-c_kI \qquad \therefore A_k=AA_{k-1}-c_kI \quad (k=1, 2, \cdots, n-1)$$

$$AA_{n-1}=c_nI$$

よって，i，iii，iv が成立する。ii を証明するために，(7) の両辺の trace をとる。

$$\mathrm{trace}[\lambda\cdot \mathrm{adj}(\lambda I-A)]-\mathrm{trace}[A\cdot \mathrm{adj}(\lambda I-A)]=n\cdot f(\lambda)$$

$$\therefore \mathrm{trace}[A\cdot \mathrm{adj}(\lambda I-A)]=\lambda\cdot \mathrm{trace}[\mathrm{adj}(\lambda I-A)]-n\cdot f(\lambda) \tag{8}$$

ところで，

$$f'(\lambda)=\frac{df(\lambda)}{d\lambda}=\frac{d}{d\lambda}\det\begin{pmatrix} a_{11}-\lambda & a_{12} & \cdots & a_{1n} \\ \vdots & & & \vdots \\ a_{m1} & & \cdots & a_{nn}-\lambda \end{pmatrix}=\det\begin{pmatrix} -1 & 0 & \cdots & 0 \\ a_{21} & a_{22}-\lambda & \cdots & a_{2n} \\ \vdots & & & \vdots \\ a_{n1} & & \cdots & a_{nn}-\lambda \end{pmatrix}+$$

$$\cdots+\det\begin{pmatrix} a_{11}-\lambda & a_{12} & \cdots & \\ 0 & -1 & 0\cdots & 0 \\ \vdots & & & \vdots \\ a_{n1} & & \cdots & a_{nn}-\lambda \end{pmatrix}+\cdots+\det\begin{pmatrix} a_{11}-\lambda & \cdots & a_{1n} \\ \vdots & & \vdots \\ a_{n-11} & \cdots & a_{n-1n} \\ 0 & \cdots\cdots & 0 \quad -1 \end{pmatrix}$$

$$=-\mathrm{trace}[\mathrm{adj}(A-\lambda I)]=\mathrm{trace}[\mathrm{adj}(\lambda I-A)]$$

であることを用い，(5) 式に注意すれば，(8) 式は，

$$\sum_{k=1}^{n}\lambda^{n-k}\mathrm{trace}(AA_{k-1})=\lambda\cdot f'(\lambda)-n\cdot f(\lambda)$$
$$=\lambda[n\lambda^{n-1}-(n-1)c_1\lambda^{n-2}\cdots-c_{n-1}]-n(\lambda^n-\cdots-nc_n)$$

$$=\sum_{k=1}^{n}\lambda^{n-k}c_k(-n+k+n)$$

係数比較により，

$$\mathrm{trace}(AA_{k-1})=kc_k \quad (k=1,\ 2,\ \cdots,\ n)$$

ゆえに ii が成立する。　　（証明終）

〔定理3〕 余因子行列，行列式，逆行列，固有多項式は次の式によって求めることができる。

（i） $\mathrm{adj}(A)=(-1)^{n-1}A_{n-1}$

（ii） $\det(A)=(-1)^{n-1}c_n$

(iii) $A^{-1}=c_n{}^{-1}A_{n-1}$ （但し，$\det(A)\neq 0$ のときのみ）

(iv) $f(\lambda)=(-1)^n(\lambda^n-c_1\lambda^{n-1}-\cdots\cdots-c_n)$

＜証明＞

（i） $\mathrm{adj}(\lambda I-A)=\lambda^{n-1}A_0+\lambda^{n-2}A_1+\cdots\cdots+\lambda A_{n-2}+A_{n-1}$

恒等式ゆえ $\lambda=0$ とおけば，

$$\mathrm{adj}(-A)=A_{n-1}$$

また，$\mathrm{adj}(-A)=(-1)^{n-1}\mathrm{adj}(A)$

$$\therefore \mathrm{adj}(A)=(-1)^{n-1}A_{n-1}$$

（ii） $\det(A-\lambda I)=(-1)^n(\lambda^n-c_1\lambda^{n-1}-\cdots\cdots-c_{n-1}\lambda-c_n)$

恒等式ゆえ $\lambda=0$ とおけば，

$$\det(A)=(-1)^n(-c_n)=(-1)^{n-1}c_n$$

(iii) 定理1の iv より $c_n\cdot I=AA_{n-1}$

$\det(A)\neq 0$ のとき (ii) より $c_n\neq 0$ であり A^{-1} が存在するから，左から A^{-1}，右から $(c_n\cdot I)^{-1}$ を両辺に乗ずると，

$$A^{-1}=c_n{}^{-1}\cdot A_{n-1}$$

(iv) 自明　　（証明終）

定理2の結果を利用すると，次のアルゴリズムで c_1，A_1，c_2，$A_2\cdots$ と計算することにより，定理3の各結果を求められることがわかる。

$$c_1=\mathrm{trace}(A), \qquad A_1=A-c_1\cdot I$$

$$c_2=\frac{1}{2}\mathrm{trace}(AA_1), \qquad A_2=AA_1-c_2\cdot I$$

$$c_3=\frac{1}{3}\mathrm{trace}(AA_2), \qquad A_3=AA_2-c_3\cdot I \qquad (9)$$

$$\vdots$$

$$c_{n-1}=\frac{1}{n-1}\mathrm{trace}(AA_{n-2}),\qquad A_{n-1}=AA_{n-2}-c_{n-1}\cdot I$$

$$c_n=\frac{1}{n}\mathrm{trace}(AA_{n-1})$$

Frame のこの方法は，正方行列で，$\det(A)\neq 0$ のときについて述べているものであるが，実はこれの類似のアルゴリズムで一般逆行列が求まることを後で述べる。

IV 一般逆行列

連立一次方程式 $A\boldsymbol{x}=\boldsymbol{b}$（$A$ は n 次正方行列，$\boldsymbol{x}$，$\boldsymbol{b}$ はベクトル）は，$\det(A)\neq 0$ のとき唯一の解を必ず持ち，それは $\boldsymbol{x}=A^{-1}\boldsymbol{b}$ であらわされる。

$\det(A)=0$ のとき，または A が長方形行列のとき，もはや解は存在しないか，存在しても1つとは限らなくなり，複数個の解のうちから適切なものを選びとりたいときがある。A の逆行列は A の要素の連続関数である。$\det(A)\neq 0$ なら A^{-1} に一致するような形で，A^{-1} によく似た性質を持つ A の一般逆行列を定め，$\boldsymbol{x}=A^{-1}\boldsymbol{b}$ に準じた扱いもできると好都合である。

A が $m\times n$ 行列のとき，次の4つの条件をみたす $n\times m$ 型行列 X を A の一般逆行列と呼び A^+ であらわす。

1. $(AX)^*=AX$
2. $(XA)^*=XA$
3. $AXA=A$
4. $XAX=X$

〔定理4〕

A が与えられたとき，1～4を満たす X は必ず存在し，しかも unique に決まる。

＜証明＞

$A=0$ ならば，$A^+=0$ とすれば明らかに1～4を満たす。

$A\neq 0$，$m=n$，$\det(A)\neq 0$ ならば A^{-1} が1～4を満たすから $A^+=A^{-1}$ である。

$A\neq 0$，$\mathrm{rank}\,A=r>0$，$r\leq\min(m,n)$ または，$m=n$，$r<m$ ならば，$m\times r$ 行列 B と $r\times n$ 行列 C が存在して $A=BC$ と表わされ，$A^+=C^*(CC^*)^{-1}(B^*B)^{-1}B^*$ とすれば，これが1～4を満たすことがわかる。

A に対して，X と Y が同時に1～4を満たすと仮定すると

$$X=XAX=X(AX)^*=XX^*\cdot A^*=XX^*\cdot A^*Y^*A^*$$

$$=X(AX)\cdot(AY)=XAY=X\cdot AYA\cdot Y=(XA)^*(YA)^*Y$$
$$=A^*X^*\cdot A^*Y^*Y=A^*Y^*Y=YAY=Y$$
$$\therefore X=Y$$

よって A^+ は unique にきまる。

（証明終）

〔定理5〕

A は $m\times n$ 行列，$\boldsymbol{x}$，$\boldsymbol{b}$ は n 次ベクトルのとき，$A\boldsymbol{x}=\boldsymbol{b}$ が解を持つための必要十分条件は，$AA^+\boldsymbol{b}=\boldsymbol{b}$ が成立することである。

＜証明＞

$A\boldsymbol{x}=\boldsymbol{b}$ ならば，$AA^+A\boldsymbol{x}=AA^+\boldsymbol{b}$

$\therefore A\boldsymbol{x}=AA^+\boldsymbol{b}$

$\therefore \boldsymbol{b}=AA^+\boldsymbol{b}$

逆に，$AA^+\boldsymbol{b}=\boldsymbol{b}$ ならば，$\boldsymbol{x}=A^+\boldsymbol{b}$ とすればこれは解の1つであるから，$A\boldsymbol{x}=\boldsymbol{b}$ は解をもつ。

（証明終）

〔定理6〕

$A\boldsymbol{x}=\boldsymbol{b}$ が解を持つとき，任意の n 次ベクトル $\boldsymbol{h}$ に対し，

$$\boldsymbol{x}=A^+\boldsymbol{b}+(I-A^+A)\boldsymbol{h}$$

と表わせる $\boldsymbol{x}$ は，この連立方程式の解である。

＜証明＞

$$A\boldsymbol{x}=AA^+\boldsymbol{b}+A(I-A^+A)\boldsymbol{h}$$
$$=\boldsymbol{b}\quad+(A-AA^+A)\boldsymbol{h}$$
$$=\boldsymbol{b}\quad+(A-A)\boldsymbol{h}$$
$$=\boldsymbol{b}$$

よって，上記 $\boldsymbol{x}$ は解である。

（証明終）

（$\boldsymbol{h}=\boldsymbol{o}$ のとき，$\boldsymbol{x}=A^+\boldsymbol{b}$ であることに注意。）

$A\boldsymbol{x}=\boldsymbol{b}$ が解を無数に持つとき，1つの解 $\boldsymbol{x}=A^+\boldsymbol{b}$ はどんな特徴を持っているだろうか？　また，別の定理によれば，$A\boldsymbol{x}=\boldsymbol{b}$ が解を持つための必要十分条件は，$\mathrm{rank}(A)=\mathrm{rank}(A,\boldsymbol{b})$ である〔ここに $(A,\ \boldsymbol{b})$ は，行列 A に，ベクトル $\boldsymbol{b}$ を右端に列ベクトル

としてつけ加えた行列とする〕。従って，$\mathrm{rank}(A)\neq\mathrm{rank}(A,\ \boldsymbol{b})$ ならば解は存在しないのだが，定理4によれば A^+ は必ず1つ存在し，$A^+\boldsymbol{b}$ も存在する。解のないときの $A^+\boldsymbol{b}$ とは，一体何を計算したことになるのだろうか？　この疑問にこたえるのが定理7である。

〔定義〕

方程式 $t(x)=g$ の，あるノルムに関する最適近似解とは，任意の x について

$$\|t(x)-g\|>\|t(x_0)-g\|$$

であるか，

$$\|t(x)-g\|=\|t(x_0)-g\|$$

である場合には，

$$\|x\|\geq\|x_0\|$$

である x_0 のことである。

今，ノルムとして $\|A\|_E=\{\sum_{i,j}|a_{ij}|^2\}^{\frac{1}{2}}$ なるユークリッド・ノルムを考える。あきらかに

$$\begin{cases}\|A\|_E=\{\mathrm{trace}(A^*A)\}^{\frac{1}{2}}\\ \|A\|_E\geq 0 \text{ で等号は } A=0 \text{ のときに限る}\end{cases}$$

である。$\|A\|_E$ は，A の各成分の連続関数であり，A と B の全ての成分が近いときは $\|A-B\|_E$ の値はゼロに近くなる。よって，最適近似解とは，最小自乗法による近似解のうち，ある種の長さ（ここでは $\|x\|_E$ のこと）が，最小のものを指している。定義からわかるように，最適近似解は唯一つであるとは限らない。しかし，今考えている方程式については次のことが成立する。

〔定理7〕

$A^+\boldsymbol{b}$ は方程式 $A\boldsymbol{x}=\boldsymbol{b}$ の唯一の最適近似解である。

＜証明＞　（以下，添字Eは省略する）

任意の行列 P，Q に対して次のことが成立する。

$$\|AP+(I-AA^+)Q\|^2=\|AP\|^2+\|(I-AA^+)Q\|^2 \tag{10}$$

なぜなら

$$\begin{aligned}&\{AP+(I-AA^+)Q\}^*\{AP+(I-AA^+)Q\}\\&\quad=(AP)^*AP+Q^*(I-AA^+)^*AP+(AP)^*(I-AA^+)Q\\&\qquad+\{(I-AA^+)Q\}^*\{(I-AA^+)Q\}\end{aligned} \tag{11}$$

ところで

$$(I-AA^+)^*AP=AP-(AA^+)^*AP=AP-AA^+AP$$
$$=AP-AP=0$$

ゆえに $(AP)^*(I-AA^+)=0$ でもある。

$\therefore$ (11)式$=(AP)^*(AP)+\{(I-AA^+)Q\}^*\{(I-AA^+)Q\}$

$\therefore \|AP+(I-AA^+)Q\|^2=\|AP\|^2+\|(I-AA^+)Q\|^2$

従って，最適近似解 $A^+\boldsymbol{b}$ と，他の近似解 $\boldsymbol{x}$ のひき起こす残差がどんなものかに着目して計算すると

$$\|A\boldsymbol{x}-\boldsymbol{b}\|^2=\|A\boldsymbol{x}-A(A^+\boldsymbol{b})+AA^+\boldsymbol{b}-\boldsymbol{b}\|^2$$
$$=\|A(\boldsymbol{x}-A^+\boldsymbol{b})-(I-AA^+)\boldsymbol{b}\|^2$$
$$=\|A(\boldsymbol{x}-A^+\boldsymbol{b})\|^2+\|(I-AA^+)(-\boldsymbol{b})\|^2$$
$$\geq\|A(A^+\boldsymbol{b})-\boldsymbol{b}\|^2 \qquad (12)$$

等号は，$\|A(\boldsymbol{x}-A^+\boldsymbol{b})\|^2=0$ すなわち $A\boldsymbol{x}=AA^+\boldsymbol{b}$ が成立するときのみである。ところで，連立方程式が解を持つ場合は，定理6より等号を成立させる $\boldsymbol{x}$ は無数に存在しうることがわかるから，最適近似解の定義の第2式を検討してみる。(10)式で A を A^+ に，P を B に，Q を $\boldsymbol{x}$ におきかえ $(A^+)^+=A$ であることを用いると，

$$\|A^+\boldsymbol{b}+(I-A^+A)\boldsymbol{x}\|^2=\|A^+\boldsymbol{b}\|^2+\|(I-A^+A)\boldsymbol{x}\|^2$$

$A\boldsymbol{x}=AA^+\boldsymbol{b}$ を用いて

$$(I-A^+A)\boldsymbol{x}=\boldsymbol{x}-A^+A\boldsymbol{x}=\boldsymbol{x}-A^+AA^+\boldsymbol{b}=\boldsymbol{x}-A^+\boldsymbol{b}$$

$\therefore \|A^+\boldsymbol{b}+\boldsymbol{x}-A^+\boldsymbol{b}\|^2=\|A^+\boldsymbol{b}\|^2+\|\boldsymbol{x}-A^+\boldsymbol{b}\|^2$

$\therefore \|\boldsymbol{x}\|^2=\|A^+\boldsymbol{b}\|^2+\|\boldsymbol{x}-A^+\boldsymbol{b}\|^2\geq\|A^+\boldsymbol{b}\|^2 \qquad (13)$

等号が成立するのは $\|\boldsymbol{x}-A^+\boldsymbol{b}\|=0$ のとき，すなわち $\boldsymbol{x}=A^+\boldsymbol{b}$ のときのみである。

(12)及び(13)式から，$A^+\boldsymbol{b}$ が最適近似解であり，等号の場合を考えることによって唯一であることが証明された。

(証明終)

残差を $\boldsymbol{e}=A\boldsymbol{x}-\boldsymbol{b}$ とすると，$\boldsymbol{e}$ は $\boldsymbol{x}$ の関数である。定理7より，連立一次方程式 $A\boldsymbol{x}=\boldsymbol{b}$ が解を持たないときは，$A^+\boldsymbol{b}$ は $\boldsymbol{e}^*\boldsymbol{e}$ を最小にするもの，即ち最小自乗法の意味で $A\boldsymbol{x}-\boldsymbol{b}$ を最も小さくするベクトルで，最小のものであることがはっきりした。

V　一般逆行列を求めるアルゴリズム（1）

〔定理8〕 A を $m\times n$ 行列，C を $n\times n$ 行列とし，$A^*A=C(A^*A)^2$ が成立するならば，

$A^+=CA^*$ である。

＜証明＞

$A^+A^{+*}A^+$ を証明すべき式の右からかける。

$$A^*AA^+A^{+*}A^+=C(A^*A)^2A^+A^{+*}A^+ \tag{14}$$

$A=AA^+A$ ゆえ，$A^*=A^*(AA^+)^*=A^*AA^+$ を用いると，

(14) の左辺 $=A^*A^{+*}A^+=(A^+A)^*A^+=A^+AA^+=A^+$

(14) の右辺 $=CA^*A\cdot A^*A\cdot A^+A^{+*}A^+=CA^*AA^*A^{+*}A^+=CA^*A(A^+A)^*A^+$

$=CA^*A\cdot A^+AA^+=CA^*AA^+=CA^*$

$\therefore\ A^+=CA^*$ （証明終）

IV 節の定理 4 より A^+ はただ 1 つ存在することがわっているから条件を満たす C が見つかればよい。

上記定理と Frame の方法（III 節）に本質的に似た形で，A^+ を計算するアルゴリズムが次の形で提示される。

〔定理 9〕 A を $m\times n$ 行列，rank は r とする。

i $B=A^*A$

ii $C_{(1)}=I$

iii $C_{(i+1)}=I\cdot\dfrac{\mathrm{trace}(C_{(i)}B)}{i}-C_{(i)}B \quad i=1,\ 2,\ \cdots,\ r-1$

iv $C=\dfrac{r\cdot C_{(r)}}{\mathrm{trace}(C_{(r)}B)}A^*$

i ～iii の順で計算すると $C_{(r+1)}B=0$，$\mathrm{trace}(C_{(r)}B)\neq 0$ で，iv の C が求める A^+ である。

＜証明＞ 証明が長くなるので step に分ける。

1-step 成分表示

各段階の成分は次のようにあらわせる。

$$\begin{cases} C_{(1)\alpha\beta}=\delta_{\alpha\beta} \\ C_{(j+1)\alpha\beta}=\dfrac{1}{j!}\displaystyle\sum_{r_1\cdots r_j}\begin{vmatrix} b_{r_1r_1} & \cdots & b_{r_1r_j} & {}_j\delta_{r_1\beta} \\ \vdots & & \vdots & \vdots \\ b_{r_jr_1} & \cdots & b_{r_jr_j} & \delta_{r_j\beta} \\ b_{\alpha r_1} & \cdots & b_{\alpha r_j} & \delta_{\alpha\beta} \end{vmatrix} \quad (j=1,2,\cdots) \end{cases} \tag{15}$$

ここに $\delta_{\alpha\beta}$ は $\alpha=\beta$ のとき 1，$\alpha\neq\beta$ のとき 0 を示す記号，$|\cdots|$ の 2 本線は行列式を，$\sum_{r_1\cdots r_j}$ は $\{1,2,\cdots,n\}$ の中から j 個とったすべての順序集合の和とする。従って一度選んだ

$r_1, r_2, \cdots, r_j$ について $j!$ 個の和をとっていることになるので $j!$ で割ると丁度1回づつの和を意味している。上式を最後の列について展開すると，次式と同等であることがわかる。

$$C_{(j+1)\alpha\beta}=\delta_{\alpha\beta}\frac{1}{j!}\sum_{r_1\cdots r_j}\begin{vmatrix} b_{r_1r_1} & \cdots & b_{r_1r_j} \\ \vdots & & \vdots \\ b_{r_jr_1} & \cdots & b_{r_jr_j} \end{vmatrix}-\frac{j}{j!}\sum_{r_1\cdots r_{j-1}}\begin{vmatrix} b_{r_1r_1} & \cdots & b_{r_1r_{j-1}} & b_{r_1\beta} \\ \vdots & & \vdots & \vdots \\ b_{r_{j-1}r_1} & \cdots & b_{r_{j-1}r_{j-1}} & b_{r_{j-1}\beta} \\ b_{\alpha r_1} & \cdots & b_{\alpha r_{j-1}} & b_{\alpha\beta} \end{vmatrix} \qquad (16)$$

第1項は (15) 式の右下隅 $\delta_{\alpha\beta}$ についての展開形。第2項は $\delta_{r_j\beta}$ についての展開形で頭に負号がつき，$r_j=\beta$ のときのみ値があることを利用。1度選んだ $r_1, \cdots, r_j$ について，例えば $r_1=\beta$ ならば1行目を (15) 式の r_j 行に，1列目を r_j 列に持ってきて展開する。$\sum$ は順列和ゆえ，β に一致するのは j 回あるので $\sum$ の前に j がくくり出される。

(15)(16) 式ともに iii の成分表示になっていることを帰納法で証明する。

$j=1$ のとき

$$C_{(2)\alpha\beta}=\delta_{\alpha\beta}\sum_{r_1}b_{r_1r_1}-b_{\alpha\beta}=\left(I\cdot\frac{1}{1}\mathrm{trace}(B)-B\right)_{\alpha\beta}$$

$$=\left(I\cdot\frac{1}{1}\mathrm{trace}(C_{(1)}B)-C_{(1)}B\right)_{\alpha\beta}$$

$$(C_{(1)}=I \text{ ゆえ})$$

$j-1$ のとき成立するとして j を調べる。

$$\left(I\cdot\frac{1}{j}\mathrm{trace}(C_{(j)}B)-C_{(j)}B\right)_{\alpha\beta}$$

$$=\delta_{\alpha\beta}\frac{1}{j}\sum_{i=1}^{n}\left(\sum_{k=1}^{n}C_{(j)ik}\,b_{ki}\right)-\sum_{k=1}^{n}C_{(j)\alpha k}\,b_{k\beta}$$

$$\text{第1項}=\delta_{\alpha\beta}\frac{1}{j}\sum_{i=1}^{n}\left[\sum_{k=1}^{n}\left\{\frac{1}{(j-1)!}\sum_{r_1\cdots r_{j-1}}\begin{vmatrix} b_{r_1r_1} & \cdots & b_{r_1r_{j-1}} & \delta_{r_1k} \\ \vdots & & \vdots & \vdots \\ b_{ir_1} & \cdots & b_{ir_{j-1}} & \delta_{ik} \end{vmatrix}\right\}b_{ki}\right]$$

（上式の δ_{r_ik} の余因子を Δ_{r_ik} で示すと）

$$=\delta_{\alpha\beta}\frac{1}{j!}\sum_{i=1}^{n}\sum_{r_1\cdots r_{j-1}}\left(\sum_{k=1}^{n}\Delta_{r_1k}\,\delta_{r_1i}\,b_{ki}+\cdots+\Delta_{r_{j-1}k}\,\delta_{r_{j-1}i}\,b_{ki}+\Delta_{ik}\,\delta_{ii}\,b_{ki}\right)$$

$$=\delta_{\alpha\beta}\frac{1}{j!}\sum_{i=1}^{n}\sum_{r_1\cdots r_{j-1}}(\Delta_{r_1k}\,b_{r_1i}+\Delta_{r_2k}\,b_{r_2i}+\cdots+\Delta_{r_{j-1}k}\,b_{r_{j-1}i}+\Delta_{ik}\,b_{ii})$$

$$=\delta_{\alpha\beta}\frac{1}{j!}\sum_{r_1\cdots r_{j-1}}\left[\sum_{i=1}^{n}(\quad\text{〃}\quad)\right]$$

$$=\delta_{\alpha\beta}\frac{1}{j!}\sum_{r_1\cdots r_j}\begin{vmatrix} b_{r_1r_1} & \cdots & b_{r_1r_j} \\ \vdots & & \vdots \\ b_{r_jr_j} & \cdots & b_{r_1r_j} \end{vmatrix},\qquad (\because i \text{ の動きを } r_j \text{ とあらわした。})$$

$$\text{第2項}=-\frac{1}{(j-1)!}\sum_{k=1}^{n}\left[\sum_{r_1\cdots r_{j-1}}\left\{\begin{vmatrix} b_{r_1r_1} & \cdots & b_{r_1r_{j-1}} & \delta_{r_1k} \\ \vdots & & \vdots & \vdots \\ b_{r_{j-1}r_1} & & b_{r_{j-1}r_{j-1}} & \delta_{r_{j-1}k} \\ b_{\alpha r_1} & & b_{\alpha r_{j-1}} & \delta_{\alpha k} \end{vmatrix}\right\} b_{k\beta}\right]$$

$$=-\frac{1}{(j-1)!}\sum_{r_1\cdots r_{j-1}}\sum_{k=1}^{n}(\Delta_{r_1k}\,\delta_{r_1k}\,b_{k\beta}+\cdots+\Delta_{r_{j-1}k}\,\delta_{r_{j-1}k}\,b_{k\beta}+\Delta_{\alpha k}\,\delta_{\alpha k}b_{k\beta})$$

$$=-\frac{1}{(j-1)!}\sum_{r_1\cdots r_{j-1}}(\Delta_{r_1k}\,b_{r_1\beta}+\cdots+\Delta_{r_{j-1}k}b_{k\beta}+\Delta_{\alpha k}\,b_{\alpha\beta})$$

$$=-\frac{j}{j!}\sum_{r_1\cdots r_{j-1}}\begin{vmatrix} b_{r_1r_1} & \cdots & b_{r_1r_{j-1}} & b_{r_1\beta} \\ \vdots & & \vdots & \vdots \\ b_{r_{j-1}r_1} & \cdots & b_{r_{j-1}r_{j-1}} & b_{r_{j-1}\beta} \\ b_{\alpha r_1} & \cdots & b_{\alpha r_{j-1}} & b_{\alpha\beta} \end{vmatrix}$$

故に j のとき (16) 式即ち (15) 式が成立する。

2-step $C_{(r+1)}B=0$, $\mathrm{trace}(C_{(r)}B)\neq 0$ であること

1-step の証明中より

$$(C_{(j)}B)_{\alpha\beta}=\frac{1}{(j-1)!}\sum_{r_1\cdots r_{j-1}}\begin{vmatrix} b_{r_1r_1} & \cdots & b_{r_1\beta} \\ \vdots & & \vdots \\ b_{r_{j-1}r_1} & \cdots & b_{r_{j-1}\beta} \\ b_{\alpha r_1} & \cdots & b_{\alpha\beta} \end{vmatrix}$$

$$\mathrm{trace}(C_{(j)}B)=\frac{1}{j!}\sum_{r_1\cdots r_j}\begin{vmatrix} b_{r_1r_1} & \cdots & b_{r_1r_j} \\ \vdots & & \vdots \\ b_{r_jr_1} & \cdots & b_{r_jr_j} \end{vmatrix}$$

$$\therefore\ (C_{(r+1)}B)_{\alpha\beta}=\frac{1}{r!}\sum_{r_1\cdots r_r}\begin{vmatrix} b_{r_1r_1} & \cdots & b_{r_1\beta} \\ \vdots & & \vdots \\ b_{r_rr_1} & \cdots & b_{r_r\beta} \\ b_{\alpha r_1} & \cdots & b_{\alpha\beta} \end{vmatrix}=0$$

($\because$ B の $(r+1)$ 次小行列式は 0 であり，その和だから)

また

$$\mathrm{trace}(C_{(r)}B)=\frac{1}{r!}\sum_{r_1\cdots r_r}\begin{vmatrix} b_{r_1r_1} & \cdots & b_{r_1r_r} \\ \vdots & & \vdots \\ b_{r_rr_1} & \cdots & b_{r_rr_r} \end{vmatrix}\neq 0$$

($\because$ B の r 次小行列式はゼロでなく，B は Positive semi definite ゆえ r 次小行列式はゼロ以上。)

3-step C が A^+ であること

$\mathrm{rank}(B)=r$ ならば $C_{(r+1)}B=0$, $\mathrm{trace}(C_{(r)}B)\neq 0$ であった。

$$\therefore\ 0=C_{(r+1)}B=I\cdot\frac{1}{r}\mathrm{trace}(C_{(r)}B)B-C_{(r)}B^2$$

$$\therefore\ \frac{\mathrm{trace}(C_{(r)}B)}{r}\cdot B=C_{(r)}B^2$$

$$\therefore\ B=\frac{r\cdot C_{(r)}}{\mathrm{trace}(C_{(r)}B)}\cdot B^2$$

故に，

$$D=\frac{r\cdot C_{(r)}}{\mathrm{trace}(C_{(r)}B)}\ \text{とおくと}\ B=DB^2$$

〔定理8〕より $C=DA^*$ とおけばこれは A^+ である。

（証明終）

証明方法は全く異なっているが，定理9のアルゴリズムは A の rank が次数より落ちる時に対して，Frame の方法を拡張したことに他ならない。

勿論，$\det(A)\neq 0$ のときの A^{-1} を求めるときにも利用できる。さらに A の rank が前もってわからない時は，$C_{(r+1)}B=0$，$\mathrm{trace}(C_{(r)}B)\neq 0$ を判定条件として iii の反復を止めることができ，かつ A の rank を計算できる。

VI　一般逆行列を求めるアルゴリズム　（2）

与えられた方程式 $F(z)=0$ を解くための方法として数値計算では，よく Newton 法が用いられる。概略だけを述べる。解の近くで $F(z)$ は微分可能とし，微分可能領域内の点 a を中心として Tayler 展開すると，

$$F(z)=F(a)+(z-a)F'(a)+(z-a)^2F''(\xi)\qquad(|z-\xi|<|z-a|),$$

z と a が近いとみなして，2次の項を捨てると近似的に

$$F(z)\simeq F(a)+(z-a)F'(a)$$

ゆえ，$F(z)=0$ を $F(a)+(z-a)F'(a)=0$

なる1次式で近似，これを解いて，

$$z=a-\frac{F(a)}{F'(a)}\ \text{を得る。}$$

a を解の第 k 近似 z_k，z を解の第 $(k+1)$ 近似 z_{k+1} と考えると，

$$z_{k+1}=z_k-\frac{F(z_k)}{F'(z_k)}$$

数列 $\{z_k\}$ が解に収束する為の条件の議論はさておき，これに端を発して $z_{k+1}=z_k+M_kr_k$（M_k，r_k は各段階のある量を表わす。）

の形の反復法をニュートン法と呼ぶ。

今 A^{-1} に対する第 k 近似を Y_k と表わし

$$R_k=I-AY_k$$

$$Y_{k+1}=Y_k+M_kR_k$$

とおく。R_k は AA^+ と AY_k の差であり，k が大きくなるほど R_k の各成分はゼロに近づいて欲しい。

$$M_k=Y_k$$

とおくと

$$Y_{k+1}=Y_k+Y_kR_k=Y_k(I+R_k)=Y_k(2I-AY_k)$$

この反復法は A^{-1} を求めるのに役に立つであろうか？ また $\det(A)=0$ の場合は何を計算することになるのだろうか？ V 節に述べたのは，A^+ を求めるために高々 A の次元までの有限回の操作を行う直接法であった。

上記反復法は 1933 年 G. Schulz によって $\det(A)\neq 0$ の場合が述べられ，1965 年 Adi Ben-Israel によって A^+ にも用いられることが証明されているものである。

〔定理10〕 $0<\alpha<\dfrac{2}{\lambda_1(A^*A)}$ なる α に対して，

$$\begin{cases} Y_0=\alpha A^* \\ Y_{k+1}=Y_k(2I-AY_k) \quad (k=0,1,2\cdots) \end{cases} \tag{17}$$

で定義される行列 $\{Y_k\}$ に対して，

$\lim_{k\to\infty} Y_k=A^+$ である。

証明のために，次の補助定理を 3 つ用意する。

〔補助定理 a〕

(17) 式で定義される Y_k は A^*B_k および C_kA^* の形をしており，

$Y_k=A^+AY_k=Y_kAA^+$ が成り立つ。

＜証明＞

帰納法によって前半を証明する。

$k=0$ のとき

$Y_0=\alpha A^*$ ゆえ $B_0=\alpha I$，$C_0=\alpha I$ と考えれば，

$Y_0=A^*B_0$，$Y_0=C_0A_0^*$ ゆえに成立する。

k のとき成立すると仮定して，$k+1$ のときを調べる。

$Y_{k+1}=Y_k(2I-AY_k)=A^*B_k(2I-AA^*B_k)$ ゆえに

$B_{k+1}=B_k(2I-AA^*B_k)$ と見れば，$Y_{k+1}=A^*B_{k+1}$

$$Y_{k+1}=(2I-Y_kA)Y_k=(2I-C_kA^*A)C_kA^*$$

ゆえに

$C_{k+1}=(2I-C_kA^*A)C_k$ と見れば，$Y_{k+1}=C_{k+1}A^*$

後半の証明は

$A=AA^+A$，従って $A^*=A^+AA^*=A^*AA^+$ を用いれば

$Y_k=A^*B_k=A^+AA^*B_k=A^+AY_k$

$Y_k=C_kA^*=C_kA^*AA^+=Y_kAA^+$

（証明終）

〔補助定理b〕

$\|AA^+-\alpha AA^*\|_s<1$ が成立するための必要十分条件は

$$0<\alpha<\frac{2}{\lambda_1(A^*A)}$$

＜証明＞

$(AA^+)AA^+=AA^*A^{+*}A^*=AA^*$

$(AA^+)AA^*=AA^*$

ゆえに AA^* と AA^+ は同じユニタリ行列で対角化できるから AA^+-AA^* の固有値は，それぞれの固有値の差である。

$(AA^+)^2=AA^+$ ゆえ AA^+ の固有値は1か0。

$\mathrm{rank}(AA^*)=\mathrm{rank}(AA^+)=\mathrm{rank}(A)$ ゆえ AA^* と AA^+ のゼロでない固有値の個数は同じである。

ゆえに $AA^+-\alpha AA^*$ の固有値は $1-\alpha\lambda_i(A^*A), i=1,2,\cdots r$

∴ $|1-\alpha\lambda_i(A^*A)|<1$ が成立するための必要十分条件は

$-1<1-\alpha\lambda_i(A^*A)<1$

∴ $0<\alpha\lambda_i(A^*A)<2$

∴ $0<\alpha<\frac{2}{\lambda_i(A^*A)}$

よって $0<\alpha<\frac{2}{\lambda_1(A^*A)}$ とすればよい。 （証明終）

（注） 実際計算では固有値がわかっていない事もあるので，α の評価には Gershgorin の定理を用いる。

$A^*A=(b_{ij})$ $i=1,\cdots n\ ;\ j=1\cdots n$ のとき

$\lambda_1(A^*A)\leq\max\limits_{1\leq i\leq n}\sum\limits_{j=1}^{n}|b_{ij}|$ ゆえ

$0<\alpha<\dfrac{2}{\max\limits_{1\leq i\leq n}\sum\limits_{j=1}^{n}|b_{ij}|}$ とする。

〔補助定理 c〕

$AY=AA^+$, $Y=A^+AY=YAA^+$ なる Y は A^+ である。

＜証明＞

IV 節の A^+ の定義の 1～4 の条件について調べる。

1. $AYA=AA^+\cdot A=A$ $\quad\therefore\ AYA=A$
2. $YAY=Y\cdot AA^+=Y$ $\quad\therefore\ YAY=Y$
3. $(AY)^*=(AA^+)^*=AA^+=AY$ $\quad\therefore\ (AY)^*=AY$
4. $(YA)^*=(A\cdot A^+Y\cdot A)^*=(A^+\cdot AA^+\cdot A)^*=(A^+A)^*=A^+A=YA$

$\therefore\ (YA)^*=YA$

（証明終）

＜定理 1 の証明＞

$AA^+-AY_{k+1}=AA^+-AY_k(2I-AY_k)$

ここで

$AA^+=AA^+AA^+$

$AY_k=AA^+AY_k$

$AY_k=AY_kAA^+$（補助定理 a より）

を左辺に代入すると，

$$\begin{aligned}AA^+-AY_{k+1}&=AA^+AA^+-AA^+AY_k-AY_kAA^++(AY_k)^2\\&=AA^+(AA^+-AY_k)-AY_k(AA^+-AY_k)\\&=(AA^+-AY_k)(AA^+-AY_k)\end{aligned}$$

$\therefore\ \|AA^+-AY_{k+1}\|_s\leq\|AA^+-AY\|_s^2\leq\|AA^+-\alpha AA^*\|_s^{2^k}$ $\quad(k=0,1,2\cdots)$

補助定理 b により

$0<\alpha<\dfrac{2}{\lambda_1(AA^*)}$ なる α に対しては

$\|AA^+-\alpha AA^*\|_s<1$

$\therefore\ \lim\limits_{k\to\infty}AY_{k+1}=AA^+$

ゆえに $\lim_{k\to\infty} Y_k=Y$ とすると $AY=AA^+$ が成立する。

補助定理 a により，$Y=A^+AY=YAA^+$。ゆえに補助定理 c より $Y=A^+$ 即ち

$\lim_{k\to\infty} Y_k=A^+$ である。

（証明終）

（収束の程度は証明中からわかるように2次である。）

VII　計算機による実験

II節で述べた方法を反復改良法，V節で述べた方法を Frame 法，VI節で述べた方法を，Newton 法と呼ぶことにする。次の4つの例についてそれぞれプログラムを組んでみた。

EX.1

係数行列として $(a-b)^{m-1}$ の係数を m 行目に下三角形に並べた。たとえば5次の行列だと

$$A=\begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 1 & -1 & 0 & 0 & 0 \\ 1 & -2 & 1 & 0 & 0 \\ 1 & -3 & 3 & -1 & 0 \\ 1 & -4 & 6 & -4 & 1 \end{pmatrix}$$

この行列は次数に関係なく $A=A^{-1}$ であり，$\operatorname{rank} A=(A$ の次数$)$ となるものである。$A\boldsymbol{x}=\boldsymbol{b}$ で $\boldsymbol{x}^*=(1,1,\cdots,1)$ となるように右辺 $\boldsymbol{b}$ を設定して計算した。結果は表1のとうりであり，$\boldsymbol{x}$ が何桁正しいかを示している。次数が大きくなるほど計算困難。

表1

次数＼方法	反復改良法	Frame 法	Newton 法	Newton 法の公式変形
3			6 桁 正 し い	反復11回 7桁正しい
5		7 桁 正 し い	1 桁 正 し い	反復18回 7桁正しい
7		7 桁 正 し い	求 ま ら ず	反復26回 7桁正しい
9		求 ま ら ず		反復30回で 求まらず
14	Gauss 法で 7桁正しい	overflow		
25	7回反復改良 6桁正しい			
30	求 ま ら ず			

（斜線は計算しなかったところ）

反復改良法による解の回復は著しいと思われるので，$n=25$ の結果を載せておく。

Gauss 法による解（反復改良するときの初期値）　　解（7回反復改良後）

表2

IMPRUV NO SYOKICHI

1	0.10000000E+01
2	0.10000000E+01
3	0.99999960E+00
4	0.99999840E+00
5	0.99999770E+00
6	0.99999050E+00
7	0.99999960E+00
8	0.99984200E+00
9	0.99996670E+00
10	0.99847600E+00
11	0.99942050E+00
12	0.99659030E+00
13	0.99398490E+00
14	0.10131800E+01
15	0.10966950E+01
16	0.13636060E+01
17	0.21159630E+01
18	0.40882680E+01
19	0.88016820E+01
20	0.18070360E+02
21	0.27810870E+02
22	-0.21294570E+01
23	-0.26342070E+03
24	-0.14967640E+04
25	-0.63005740E+04

表3

1	0.10000000E+01
2	0.10000000E+01
3	0.10000000E+01
4	0.10000000E+01
5	0.10000000E+01
6	0.10000000E+01
7	0.10000000E+01
8	0.10000000E+01
9	0.10000000E+01
10	0.10000000E+01
11	0.10000000E+01
12	0.10000000E+01
13	0.10000000E+01
14	0.10000000E+01
15	0.10000000E+01
16	0.10000000E+01
17	0.10000000E+01
18	0.10000000E+01
19	0.10000000E+01
20	0.10000000E+01
21	0.10000000E+01
22	0.99999990E+00
23	0.99999960E+00
24	0.99999870E+00
25	0.99999800E+00

表2の数値からわかるように，Gauss 法によって求まった答は，斜線より下の部分が解とは大幅に違っている。それが，7回の反復改良後，表3に示されているように，ほぼ6桁正しくなった。

表1に「Newton 法の公式変形」の項があるが，これについて説明しておく。アルゴリズムは

$$X_{k+1}=X_k(2I-AX_k) \tag{18}$$

であった。このとうりプログラムを組んだのが「Newton 法」とした項である。ところが，アルゴリズムの導かれた過程をよく見ると

$$X_{k+1}=X_k+X_k(I-AX_k) \tag{19}$$

であり，AX_k は X_k が A^{-1} の良い近似ならば I に近い筈である。有限桁の計算機では $(I-AX_k)$ の計算は微妙で，桁落ち現象（＝有効数字が失われる）を起す可能性が大きい。そこでこの部分の計算だけを倍精度にして (19) 式でプログラムを組んだ結果を「Newton 法の公式変形」としてまとめた。表1から明らかなように，(18) 式を (19) 式として一部分を倍精度計算（そのために記憶場所6 word，プログラムステップ10だけ増加。）に直しただけで結果はかなり良好になることがわかる。計算機使用の時，桁落ちに対する注意を常にしなければならないことを示す典型的な例である。

また Newton 法は VI 節で述べたごとく，$\lim_{k\to\infty} X_k = A^+$ で2次収束だから，かなり早く A^+ に近づく筈だが $n=6$ では15回反復してもそれらしき様子は見られない。これは，VI 節のは，計算機の丸め誤差を全く考慮にいれてない，純粋に数学的な収束の証明であって，計算機による数値計算は誤差論をぬきにしては考えられないことを示している。

Frame 法については $n=14$ のとき，アルゴリズム途中で overflow（計算機による桁あふれ）してしまう。これは，乗算を何度も用いるアルゴリズムは不利である事が考えられる。

一般に $\det(A)\neq 0$ のときは，Gauss 消去法の反復法が他の2つよりもかなり精度よく求まる。Frame 法も Newton 法も解が求まる場合でも，補助記憶のためにかなり場所をとるし，乗算が多くて演算時間が長くなるという欠点があり，やはり $\det(A)=0$ のときのためのアルゴリズムと考えた方が良さそうである。

EX.2

$$A=\begin{pmatrix} 1 & 0 & -2 \\ 0 & 1 & -1 \\ -1 & 1 & 1 \\ 2 & -1 & 2 \end{pmatrix},\quad \boldsymbol{b}=\begin{pmatrix} -1 \\ 0 \\ 1 \\ 3 \end{pmatrix},\quad \operatorname{rank} A=3$$

$$A^+=\frac{3}{225}\begin{pmatrix} 20 & 25 & 5 & 30 \\ 10 & 50 & 40 & 15 \\ -15 & 0 & 15 & 15 \end{pmatrix}$$

EX.3

$$A=\begin{pmatrix} -1 & -2 \\ 0 & 0 \\ 2 & 4 \\ 1 & 2 \\ 3 & 6 \end{pmatrix},\quad \boldsymbol{b}=\begin{pmatrix} -3 \\ 0 \\ 6 \\ 3 \\ 9 \end{pmatrix},\quad \operatorname{rank} A=1$$

$$A^+=\frac{1}{75}\begin{pmatrix} -1 & 0 & 2 & 1 & 3 \\ -2 & 0 & 4 & 2 & 6 \end{pmatrix}$$

表4

	Frame 法	Newton 法
EX.2	rank 3 解は6桁正しい。	反復8回後， 解は6桁正しい。
EX.3	rank 1 解は6桁正しい。	反復2回後， 解は3桁正しい。

EX.4

$$A=\begin{pmatrix}3 & 2 & 5\\ 2 & 1 & 3\\ 6 & -3 & 3\end{pmatrix},\quad \boldsymbol{b}=\begin{pmatrix}10\\ 6\\ 6\end{pmatrix},\quad \operatorname{rank}A=2$$

これは第1列と第2列の和が第3列になっているものである。

表5

反復改良法	Frame 法	Newton 法
解として $\begin{pmatrix}-1\\ -1\\ 3\end{pmatrix}$ を7桁正しく求める。	rank 2, 解として $\frac{1}{3}\begin{pmatrix}2\\ 2\\ 4\end{pmatrix}$ を6桁正しく求める。	6回反復後, 解として $\frac{1}{3}\begin{pmatrix}2\\ 2\\ 4\end{pmatrix}$ を6桁正しく求める。

$A_3=\begin{pmatrix}3 & 2\\ 2 & 1\end{pmatrix}$ で $\det(A_2)=-1\neq 0$ ゆえ 定理1より三角分解可能。反復改良法は一応の答を出しているが，これは $\|\boldsymbol{x}\|$ 最小のものではない。Frame 法，Newton 法は共に $\|\boldsymbol{x}\|$ 最小のものを6桁精度で出している。以上の実験では，Frame 法は相当正確に rank を計算していることがわかる。

VIII 反復改良法のプログラム

連立一次方程式 $A\boldsymbol{x}=\boldsymbol{b}$ に対する解法として Gauss 消去法の反復改良法は，計算センターなどに置いておくと役に立ちそうである。詳しいアルゴリズムの説明や収束の証明は省いて，プログラムとその使用法だけ説明する。

サブルーチン名 HANPUK (N, A, UL, B, X, ISW, II)

入力パラメーター

A……連立方程式の係数行列

N……Aの次元

B……連立方程式の右辺ベクトル

II……主プログラムで切る配列の大きさ

出力パラメーター

X……連立方程式の解ベクトル

ISW……0　正常に解を求める。

1　Aに0ばかりの行がある。

2　Pivot が0になった。

3　10回の改良でも解が収束しない。

4　行列Aの性質が悪く（最大固有値と最小固有値の比が大きすぎること），解を改良できない。

中間パラメーター

・　UL……Aを三角行列に分解したものを保存する。

```
0001            SUBROUTINE HANPUK(N,A,UL,B,X,ISW,II)
                C     GAUSS-SYOKYO HOO WO HANPUKU-KAIRYO SITE  A*X=B WO TOKU.
                C          INPUT
                C               N-----GYORETU NO JIGEN
                C               A-----KEISUU GYORETSU(HOZON SARERU)
                C               B-----HOOTEI-SIKI NO UHEN VECTOR
                C               II----MAIN PROGRAM NO HAIRETSU A NO JIGEN
                C          OUTPUT
                C               X-----KOTAE
                C               UL----A NO SANKAKU BUNKAI
                C               ISW---A NO SINGULARITY WO HANTEI SURU
                C                     ISW=0 --- SEIJYO
                C                     ISW=1 --- A NI ZERO GYO GA ARU
                C                     ISW=2 --- PIVOT GA ZERO NI NATTA
                C                     ISW=3 --- 10-KAI KAIRYO SITA GA---
                C                     ISW=4 --- A NO SEISITSU WARUKU KAIRYO NO IMI NASI
                C
                C
0002            DIMENSION A(II,II),UL(II,II),B(II),X(II),Y(30),Z(30)
0003            ISW=0
0004            CALL DECOMP(N,A,UL,ISW,II,Y)
0005            IF(ISW.NE.0) GOTO 100
0006            CALL SOLVE (N,UL,B,X,II)
0007            CALL IMPRUV (N,A,UL,B,X,ISW,II,Y,Z)
0008          100 RETURN
0009            END
```

```
0001            SUBROUTINE DECOMP(N,A,UL,ISW,II,SCALES)
0002            DIMENSION A(II,II),UL(II,II),SCALES(II)
0003            COMMON /DDD/ IPS(30)
                C
                C  INITIALIZE IPS,UL,SCALES
                C
0004            DO 5 I=1,N
0005            IPS(I)=I
0006            ROWNRM=0.0
0007            DO 2 J=1,N
0008            UL(I,J)=A(I,J)
0009            IF(ROWNRM-ABS(UL(I,J))) 1,2,2
0010          1 ROWNRM=ABS(UL(I,J))
0011          2 CONTINUE
0012            IF(ROWNRM) 3,4,3
0013          3 SCALES(I)=1.0/ROWNRM
0014            GOTO 5
```

```
0015          4     ISW=1
0016                RETURN
0017          5     CONTINUE
              C
              C
              C  GAUSSIAN ELIMINATION WITH PARTIAL PIVOTING
              C
0018                NM1=N-1
0019                DO 17 K=1,NM1
0020                BIG=0.0
0021                DO 11 I=K,N
0022                IP=IPS(I)
0023                SIZE=ABS(UL(IP,K))*SCALES(IP)
0024                IF(SIZE-BIG) 11,11,10
0025             10 BIG=SIZE
0026                IDXPIV=I
0027          11    CONTINUE
0028                IF(BIG) 13,12,13
0029          12    ISW=2
0030                RETURN
0031          13    IF(IDXPIV-K) 14,15,14
0032          14    J=IPS(K)
0033                IPS(K)=IPS(IDXPIV)
0034                IPS(IDXPIV)=J
0035          15    KP=IPS(K)
0036                PIVOT=UL(KP,K)
0037                KP1=K+1
0038                DO 16 I=KP1,N
0039                IP=IPS(I)
0040                EM=-UL(IP,K)/PIVOT
0041                UL(IP,K)=-EM
0042                DO 16 J=KP1,N
0043                UL(IP,J)=UL(IP,J)+EM*UL(KP,J)
              C
0044          16    CONTINUE
0045          17    CONTINUE
0046                KP=IPS(N)
0047                IF(UL(KP,N)) 19,18,19
0048          18    ISW=2
0049          19    RETURN
0050                END


0001                SUBROUTINE SOLVE(N,UL,B,X,II)
0002                DIMENSION UL(II,II),B(II),X(II)
0003                COMMON /DDD/ IPS(30)
0004                NP1=N+1
0005                IP=IPS(1)
0006                X(1)=B(IP)
0007                DO 2 I=2,N
0008                IP=IPS(I)
0009                IM1=I-1
0010                SUM=0.0
0011                DO 1 J=1,IM1
0012          1     SUM=SUM+UL(IP,J)*X(J)
0013          2     X(I)=B(IP)-SUM
              C
0014                IP=IPS(N)
0015                X(N)=X(N)/UL(IP,N)
0016                DO 4 IBACK=2,N
0017                I=NP1-IBACK
0018                IP=IPS(I)
0019                IP1=I+1
0020                SUM=0.0
0021                DO 3 J=IP1,N
0022          3     SUM=SUM+UL(IP,J)*X(J)
0023          4     X(I)=(X(I)-SUM)/UL(IP,I)
0024                RETURN
0025                END
```

```
0001            SUBROUTINE IMPRUV(N,A,UL,B,X,ISW,II,R,DX)
0002            DIMENSION A(II,II),UL(II,II),B(II),X(II),R(II),DX(II)
0003            DOUBLE PRECISION SUM,DA,DXX,DB
                C
0004            WRITE(6,8000) (X(I),I=1,N)
0005          8000 FORMAT(//,'IMPRUV NO SYOKICHI',//(10X,E16.8))
0006            EPS=1.0E-7
0007            ITMAX=10
                C
0008            XNORM=0.0
0009            DO 1 I=1,N
0010          1 XNORM=AMAX1(XNORM,ABS(X(I)))
0011            IF(XNORM) 3,2,3
0012          2 DIGITS=-ALOG10(EPS)
0013            RETURN
                C
0014          3 DO 9 ITER=1,ITMAX
0015            DO 5 I=1,N
0016            SUM=0.0
0017            DO 4 J=1,N
0018            DA=A(I,J)
0019            DXX=X(J)
0020          4 SUM=SUM+DA*DXX
0021            DB=B(I)
0022            SUM=DB-SUM
0023          5 R(I)=SUM
0024            CALL SOLVE(N,UL,R,DX,II)
0025            DXNORM=0.0
0026            DO 6 I=1,N
0027            T=X(I)
0028            X(I)=X(I)+DX(I)
0029            DXNORM=AMAX1(DXNORM,ABS(X(I)-T))
0030          6 CONTINUE
0031            IF(ITER-1) 8,7,8
0032          7 DIGITS=-ALOG10(AMAX1(DXNORM/XNORM,EPS))
0033            IF(DIGITS) 80,80,100
0034         80 WRITE(6,90)
0035         90 FORMAT(//,'MATRIX NO SEISITSU WARUKU KAIRYO DEKINAI')
0036            ISW=4
0037        100 WRITE(6,110) DIGITS
0038        110 FORMAT(//,' SUKUNAKU TO MO ',E12.3,'-KETA TADASII'/)
0039            IF(ISW-4) 8,20,8
0040          8 IF(DXNORM-EPS*XNORM)10,10,9
0041         10 WRITE(6,5000) ITER
0042       5000 FORMAT(/' KAIRYO KAISUU =',I3/)
0043            GO TO 20
0044          9 CONTINUE
                C
                C     HANPUKU SITEMO KAI GA SYUSOKU SINAI
                C
0045            ISW=3
0046         20 RETURN
0047            END
```

(サブルーチン HANPUK は，サブルーチン DECOMP, SOLVE, IMPRUV が一体となって機能するものである。)

HANPUK ルーチンの使用例

```
DIMENSION A(30, 30), UL(30, 30), B(30), X(30)
L=30
READ(5, 10) N
```

```
 10  FORMT(I2)
     READ(5, 20) ((A(I, J), J=1, N), I=1, N), (B(I), I=1, N)
 20  FORMAT(10F8. 2)
     CALL HANPUK (N, A, UL, B, X, ISW, L)
     IF(ISW) 30, 50, 30
 30  WRITE(6, 40) ISW
 40  FORMAT(▼ISW▼=, I3)
     STOP
 50  WRITE(6, 60) (X(I), I=1, N)
 60  FORMAT(▼KOTAE▼/7 (3X, E15. 7))
     STOP
     END
```

印刷結果

```
(イ) IMPRUV NO SYOKICHI
                    0.10000000E+01
                    0.10131800E+01
                    0.13636060E+01
(ロ) SUKUNAKU TO MO 0.222E+01−KETA TADASII
(ハ) KAIRYO KAISUU=3
(ニ) KOTAE
     0.10000000E+01
     0.10000000E+01
     0.10000000E+01
```

(イ)～(ハ)は，サブルーチン HANPUK で印刷したもの。(ニ)は主プログラムで印刷した解である。(イ)の「IMPRUV NO SYOKICI」とは，Gauss 消去法の答，(ロ)はそれが何桁まで信用できるかの目安である。(ハ)は何回改良したかのメッセージ。なお，行列Aの数値はサブルーチン HANPUK によっては変化しない。