数値計算法（連立一次方程式の解法）

2004/05/19 平野拓一 （東京工業大学）

1. はじめに

行列方程式を解く方法について説明する。正則な正方行列を解くことが基本であり、ほとんどの連立一次方程式を解く場合はこの問題である。しかし、連立一次方程式を立てる際、条件の数が未知数よりも多くなったり少なくなったりしまうことも現実問題として起こる。そのような場合は正方行列ではなく、一般の長方形行列方程式を解く問題となる。このような一般の場合にも対処できるように興味ある人のために擬似逆行列の説明も加えた。

2. 直接解法

行列方程式を解くと言われると、線形代数で習うクラーメル(Cramer)の公式を使おうと思うかもしれない。クラーメルの公式は表現としては非常に美しく、あるときは見通しのよいビジョンを与えてくれるが、行列のサイズが大きな数値計算の問題で行列方程式を解こうとしたときには効率の悪い方法である。次に説明するガウスの消去法を用いるとずっと効率が良い。

2.1 ガウスの消去法(Gaussian Elimination)

掃き出し法とも言われる。

2.1.1 例による説明

$$\begin{aligned} 2u &+ v + w = 1 \\ 4u &+ v = -2 \\ -2u &+ 2v + w = 7 \end{aligned}$$

を解く問題を考える([1](p.8)より)。

(Step1)

前進消去

(forward elimination)

$\begin{aligned} 2u &+ v + w = 1 \\ 4u &+ v = -2 \\ -2u &+ 2v + w = 7 \end{aligned}$	行列表現	左から掛ける行列
② ④ ⑤	$\begin{bmatrix} 2 & 1 & 1 \\ 4 & 1 & 0 \\ -2 & 2 & 1 \end{bmatrix}\begin{bmatrix} u \\ v \\ w \end{bmatrix} = \begin{bmatrix} 1 \\ -2 \\ 7 \end{bmatrix}$	
	$A\mathbf{x} = \mathbf{b}$	$L_1 = \begin{bmatrix} 1 & 0 & 0 \\ -2 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}$
$\begin{aligned} 2u &+ v + w = 1 \\ &-1v - 2w = -4 \\ &3v + 2w = 8 \end{aligned}$	$\begin{bmatrix} 2 & 1 & 1 \\ 0 & -1 & -2 \\ 0 & 3 & 2 \end{bmatrix}\begin{bmatrix} u \\ v \\ w \end{bmatrix} = \begin{bmatrix} 1 \\ -4 \\ 8 \end{bmatrix}$	
⑥ ⑦ -3	$L_1 A\mathbf{x} = L_1\mathbf{b}$	$L_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 3 & 1 \end{bmatrix}$
$\begin{aligned} 2u &+ v + w = 1 \\ &-1v - 2w = -4 \\ &-4w = -4 \end{aligned}$	$\begin{bmatrix} 2 & 1 & 1 \\ 0 & -1 & -2 \\ 0 & 0 & -4 \end{bmatrix}\begin{bmatrix} u \\ v \\ w \end{bmatrix} = \begin{bmatrix} 1 \\ -4 \\ -4 \end{bmatrix}$	
前進消去過程終了	$L_2 L_1 A\mathbf{x} = L_2 L_1\mathbf{b}$	

なぜ右上三角行列も一気に 0 にせずに、最初に左下三角行列だけ 0 にするのかと言うと、その方が少しだけ効率が良いからである。左下三角行列と右上三角行列を同時に 0 にする方法を**ガウ**

ス・ジョルダン法(Gauss-Jordan method)と呼ぶ。それに対して上のように最初は左下三角行列だけ 0 にする方法を **LU 分解(LU factorization)**と呼ぶ。後で説明するが、ガウス・ジョルダン法は計算負荷が$O(N^3)$であるのに対し、LU 分解は$O(N^3/3)$である。N が非常に大きいときは 1/3 の係数など些細なものだが、ほぼ同じ方法にしては多少なりとも効率が良く、単純計算で 3 倍計算速度が速い。また、後述するが、LU 分解だと後から同じ係数行列だが異なる右辺ベクトルを持つ方程式を解きたくなった場合にはその負荷は$O(N^2/2)$で済むが、ガウス・ジョルダン法だと最初からやり直さなければならず、再び$O(N^3/3)$の計算が必要になるのである。

(Step2)

後退代入

(back-substitution)

$$\begin{aligned} 2u + v + w &= 1 \\ -1v - 2w &= -4 \\ w &= 1 \end{aligned}$$

$$\begin{bmatrix} 2 & 1 & 1 \\ 0 & -1 & -2 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} u \\ v \\ w \end{bmatrix} = \begin{bmatrix} 1 \\ -4 \\ 1 \end{bmatrix}$$

$$D_1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1/4 \end{bmatrix}$$

$$D_1 L_2 L_1 A\mathbf{x} = D_1 L_2 L_1 \mathbf{b}$$

$$\begin{aligned} 2u + v &= 0 \\ -1v &= -2 \\ w &= 1 \end{aligned}$$

$$\begin{bmatrix} 2 & 1 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} u \\ v \\ w \end{bmatrix} = \begin{bmatrix} 0 \\ -2 \\ 1 \end{bmatrix}$$

$$U_1 = \begin{bmatrix} 1 & 0 & -1 \\ 0 & 1 & 2 \\ 0 & 0 & 1 \end{bmatrix}$$

$$U_1 D_1 L_2 L_1 A\mathbf{x} = U_1 D_1 L_2 L_1 \mathbf{b}$$

$$\begin{aligned} 2u + v &= 0 \\ v &= 2 \\ w &= 1 \end{aligned}$$

$$\begin{bmatrix} 2 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} u \\ v \\ w \end{bmatrix} = \begin{bmatrix} 0 \\ 2 \\ 1 \end{bmatrix}$$

$$D_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$D_2 U_1 D_1 L_2 L_1 A\mathbf{x} = D_2 U_1 D_1 L_2 L_1 \mathbf{b}$$

$$\begin{aligned} 2u &= -2 \\ v &= 2 \\ w &= 1 \end{aligned}$$

$$\begin{bmatrix} 2 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} u \\ v \\ w \end{bmatrix} = \begin{bmatrix} -2 \\ 2 \\ 1 \end{bmatrix}$$

$$U_2 = \begin{bmatrix} 1 & -1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$U_2 D_2 U_1 D_1 L_2 L_1 A\mathbf{x} = U_2 D_2 U_1 D_1 L_2 L_1 \mathbf{b}$$

$$\begin{aligned} u &= -1 \\ v &= 2 \\ w &= 1 \end{aligned}$$

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} u \\ v \\ w \end{bmatrix} = \begin{bmatrix} -1 \\ 2 \\ 1 \end{bmatrix}$$

$$D_3 = \begin{bmatrix} 1/2 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

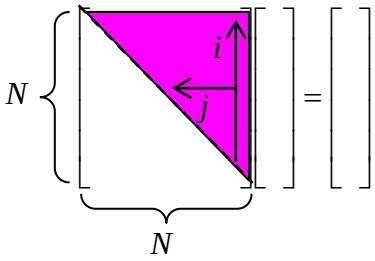
$$D_3 U_2 D_2 U_1 D_1 L_2 L_1 A\mathbf{x} = D_3 U_2 D_2 U_1 D_1 L_2 L_1 \mathbf{b}$$

後退代入は普通上のように考えず、下から未知数が求まっていくので、次のように計算を行う（計算量は同じなのでどちらでもよいのだが）。U を右上三角行列、$\mathbf{x}$ を未知列ベクトル、$\mathbf{c}$ を既知列ベクトルとすると、

$U\mathbf{x} = \mathbf{c}$

から次のように $\mathbf{x}$ を求める。

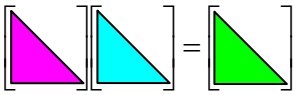
$$x_i = \frac{1}{u_{ii}}\left(c_i - \sum_{j=i+1}^{N} u_{ij} x_j \right) \qquad (i = N, \cdots, 1)$$



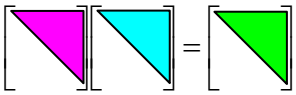
2.1.2 三角行列に関する諸定理

L: Lower-left Triangular Matrix, U: Upper-right Triangular Matrix

$L_i \in$ 左下三角行列 $\Rightarrow$ $L_m L_n \in$ 左下三角行列



$U_i \in$ 右上三角行列 $\Rightarrow$ $U_m U_n \in$ 右上三角行列



$LD \in$ 左下三角行列、$DL \in$ 左下三角行列

$UD \in$ 右上三角行列、$DU \in$ 右上三角行列

$DA = AD$： D は交換可能

$L \in$ 左下三角行列 $\Rightarrow$ $L^{-1} \in$ 左下三角行列

$U \in$ 右上三角行列 $\Rightarrow$ $U^{-1} \in$ 右上三角行列

(∵)

説明 1)

各前進消去のステップを逆に戻す操作を想像すればよい。

$$L = \begin{bmatrix} 1 & 0 & 0 \\ a & 1 & 0 \\ b & 0 & 1 \end{bmatrix}$$ ならば、元に戻す操作は $$L^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ -a & 1 & 0 \\ -b & 0 & 1 \end{bmatrix}$$ となる（$LL^{-1} = L^{-1}L = I$）。

説明 2)

$$A=\begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \qquad (b=c=f=0 \text{ とする})$$

の逆行列をクラーメル(Cramer)の公式を使って求める過程を考えるとわかる。

$$A^{-1}=\frac{1}{\det(A)}\begin{bmatrix} \begin{vmatrix} e & f \\ h & i \end{vmatrix} & -\begin{vmatrix} d & f \\ g & i \end{vmatrix} & \begin{vmatrix} d & e \\ g & h \end{vmatrix} \\ -\begin{vmatrix} b & c \\ h & i \end{vmatrix} & \begin{vmatrix} a & c \\ g & i \end{vmatrix} & -\begin{vmatrix} a & b \\ g & h \end{vmatrix} \\ \begin{vmatrix} b & c \\ e & f \end{vmatrix} & -\begin{vmatrix} a & c \\ d & f \end{vmatrix} & \begin{vmatrix} a & b \\ d & e \end{vmatrix} \end{bmatrix}^t = \frac{1}{\det(A)}\begin{bmatrix} \begin{vmatrix} e & 0 \\ h & i \end{vmatrix} & -\begin{vmatrix} d & 0 \\ g & i \end{vmatrix} & \begin{vmatrix} d & e \\ g & h \end{vmatrix} \\ -\begin{vmatrix} 0 & 0 \\ h & i \end{vmatrix} & \begin{vmatrix} a & 0 \\ g & i \end{vmatrix} & -\begin{vmatrix} a & 0 \\ g & h \end{vmatrix} \\ \begin{vmatrix} 0 & 0 \\ e & 0 \end{vmatrix} & -\begin{vmatrix} a & 0 \\ d & 0 \end{vmatrix} & \begin{vmatrix} a & 0 \\ d & e \end{vmatrix} \end{bmatrix}^t$$

$$=\frac{1}{\det(A)}\begin{bmatrix} \begin{vmatrix} e & 0 \\ h & i \end{vmatrix} & -\begin{vmatrix} d & 0 \\ g & i \end{vmatrix} & \begin{vmatrix} d & e \\ g & h \end{vmatrix} \\ 0 & \begin{vmatrix} a & 0 \\ g & i \end{vmatrix} & -\begin{vmatrix} a & 0 \\ g & h \end{vmatrix} \\ 0 & 0 & \begin{vmatrix} a & 0 \\ d & e \end{vmatrix} \end{bmatrix}^t$$

よって、下三角行列である。

$D_3U_2D_2U_1D_1L_2L_1A=I$

D_i に交換規則を適用し、結合規則（重要！）を適用する。

$(U_2U_1)(D_3D_2D_1)(L_2L_1)A=I$

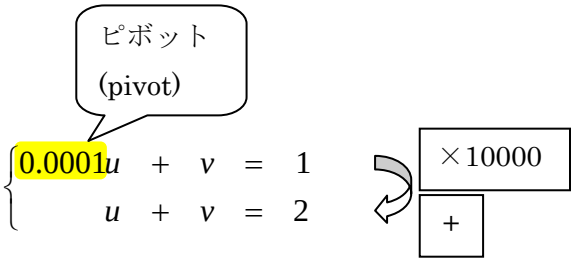
$UDLA=I$

$A=L^{-1}D^{-1}U^{-1}$

$A=L'D'U'$ (LU 分解, LDU 分解)

上の説明では逆行列を求めないとならないと思うかもしれないが、実は Step1 だけを見ていればすぐに LU 分解できたことになる。2.2 節で Step1 の例を通して具体的に L と U を求める方法を説明する。

2.1.3 まるめ誤差



$$\begin{cases} 0.0001u + v = 1 \\ -9999v = -9998 \end{cases}$$

$$\begin{cases} 0.0001u + v = 1 \\ v = 0.99989998\cdots \end{cases}$$

$$\begin{cases} 0.0001u + v = 1 \\ v \cong 0.9999 \end{cases}$$

$$\begin{cases} 0.0001u \cong 0.0001 \\ v \cong 0.9999 \end{cases}$$

$$\begin{cases} u \cong 1 \\ v \cong 1 \end{cases}$$

正しい（良い近似）

$$\begin{cases} 0.0001u + v = 1 \\ v \cong 1 \end{cases}$$

コンピュータのメモリには 2 進数で記憶するため、必然的に「まるめ誤差」が発生する。

$$\begin{cases} 0.0001u \cong 0 \\ v \cong 1 \end{cases}$$

$$\begin{cases} u \cong 0 \\ v \cong 1 \end{cases}$$

間違い

数値計算で犠牲になる（まるめ誤差となって無視されて消え失せる）のは小さな値である。元々小さな値（とくべき問題の中の最大値、平均値に比べて小さな値）はほとんど無視できる重要でない数値であり、誤差や雑音（ノイズ）みたいなものなのに、それをわざわざ大きく増幅して他の式と平等に扱い続けているのがよくない。具体的には最初の変形で、1 行目を 10000 倍して 2 行目に足したが、そのときに 2 行目の v の係数の 1 や右辺の 2 が軽視されて、誤差に埋もれてしまうのである。大切なのは小さな値でなく、大きな値である。測定データを処理するときのことを考えても、大きな値に意味があり、小さな値は雑音（ノイズ）と考えなければならない。

今の手法の問題点は小さな値（それは数値誤差やノイズと考えるべきである）を他の数値を消去するための**ピボット（pivot, 中心、要、軸、枢軸）**として用いていたことであり、ピボットは重要な役割を担っているのだから、誤差として消え失せるべき小さな値にそのような重要なポジションを与えてはいけない。そこで、なるべく大きな値をピボットに選ぶことによって、今の問題は回避できる。

部分選択(partial pivoting)を行った場合

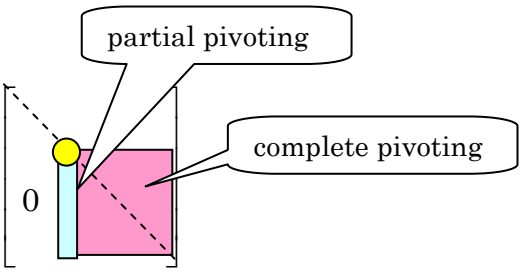
$$\begin{cases} u + v = 2 \\ 0.0001u + v = 1 \end{cases}$$

$$\begin{cases} u + v = 2 \\ 0.9999v = 0.9998 \end{cases}$$

$$\begin{cases} u + v = 2 \\ v \cong 1 \end{cases}$$

$$\begin{cases} u \cong 1 \\ v \cong 1 \end{cases}$$

部分選択では行の入れ替えだけ行ったが、列も入れ替えて最大の係数をピボットに用いる方法を**完全選択(complete pivoting)**と言う。完全選択は計算時間がより多くかかるし、入れ替えた順番を保持しておくために必要なメモリ容量も増える。部分選択は完全選択とほぼ同じ程度に良いことが数学的に正確に述べることができる[2](p.51)し、実用上部分選択で十分なので、部分選択が数値計算でよく用いられる。



他にも数値計算では

- まるめ誤差　　2 進数に変換するときのまるめ(round)による。メモリでは２進数で記憶するし、ALU で演算するときにも２進数で行うか

ら当然そのときにまるめ誤差が発生する。

- 打切り誤差　(例) $\sum_{n=1}^{\infty} a_n \cong \sum_{n=1}^{N} a_n$
- 桁落ち誤差　ほとんど同じ値同士の引き算で有効桁数が激減するときにおこる。(例) 8.3425071-8.3418425=0.0006646

などの誤差があることを絶えず意識してプログラミングしなければならない。

2.1.4 複数の右辺ベクトルの解を同時に得る方法、逆行列の計算法

2.1.1 節では

$A\mathbf{x}=\mathbf{b}$

を解くことが目的であり、単一の右辺ベクトル$\mathbf{b}$（物理、工学の問題では外力、励振などの影響に相当する）の場合を考察していた。しかし現実問題として、Aは変わらないが複数右辺ベクトル$\mathbf{b}_i\,(i=1,\cdots,M)$に対して

$$\begin{cases} A\mathbf{x}=\mathbf{b}_1 \\ \quad\vdots \\ A\mathbf{x}=\mathbf{b}_M \end{cases}$$

の全てを解きたいことがよくある。例えば、複数の励振箇所を持つアンテナや、複数のポートを持つ導波管を用いたマイクロ波回路を考えてみよう。散乱行列(scattering matrix)を求めたいことがよくあるが、その際、様々な場所から励振するが、リアクション行列Aは変化せず、$\mathbf{b}$が変化するだけである。この場合には 2.1.1 のガウスの消去法をM回繰り返す必要はない。

次のような列拡大行列という概念を用い、2.1 のガウスの消去法の操作を一度に行えばよい。

$A\left[\mathbf{x}_1 \quad \cdots \quad \mathbf{x}_M\right]=\left[\mathbf{b}_1 \quad \cdots \quad \mathbf{b}_M\right]$

上の$\mathbf{x}_i$は$A\mathbf{x}=\mathbf{b}_i$の解であり、$\left[\mathbf{x}_1 \quad \cdots \quad \mathbf{x}_M\right]$は$\mathbf{x}_i$を列ベクトルに持つ行列である。$\left[\mathbf{b}_1 \quad \cdots \quad \mathbf{b}_M\right]$は$\mathbf{b}_i$を列ベクトルに持つ行列である。

また、$A^{-1}\,(A^{-1}A=AA^{-1}=I$を満たす)を求めたいこともあるが、$A^{-1}$を求めるにも上のテクニックを使うことができる。$I$を$A$と同じ次元の単位行列として、

$A\left[\mathbf{x}_1 \quad \cdots \quad \mathbf{x}_M\right]=I$

をガウスの消去法を用いて$\left[\mathbf{x}_1 \quad \cdots \quad \mathbf{x}_M\right]$を求めれば、最終的にはこれが$A^{-1}$として求まるのである。

2.2 LU 分解

Lを対角成分を含む左下三角行列、Dを対角行列、Uを対角成分を含む右上三角行列とすると、2.1 節で

$A=LU$（$A=LDU$）

と書くことができることを説明した。

解くべき方程式は

$A\mathbf{x}=\mathbf{b}$

$LU\mathbf{x}=\mathbf{b}$

となる。

ここで、

$\mathbf{y}=U\mathbf{x}$

とおくと、

$L\mathbf{y}=\mathbf{b}$

ガウスの消去法の後退代入と同じ手法によって $\mathbf{y}$ を求める($\mathbf{y}=L^{-1}\mathbf{b}$ だが、L^{-1} を計算するのではない！)。この作業の計算時間は $N^2/2$ のオーダーである（この節の最後に説明)。
さらに同様に、$\mathbf{y}$ が求まったので、

$U\mathbf{x}=\mathbf{y}$

から、$\mathbf{x}$ を求める。この作業の計算時間は $N^2/2$ のオーダーである。
従って、全体の計算時間は $O(N^2)$ であり、ガウスの消去法を最初からやり直すと $O(N^3)$ なのでずっと速い。

よって、複数の右辺ベクトル $\mathbf{b}_i\,(i=1,\cdots,M)$ に対する $A\mathbf{x}=\mathbf{b}_i$ の解を求める際、ガウスの消去法、では複数の $\mathbf{b}_i$ に対して同時に前進消去と後退代入を実行したが、後から更に別の新たな右辺ベクトル $\mathbf{b}$ に対して同様に $A\mathbf{x}=\mathbf{b}$ の解を求めようとするとき、また最初からやり直さなければならない。そこで、$A=LU$ と分解しておく（つまり、ガウスの消去法の過程で出てくる行列を残しておく）と、上記の説明のように後退代入の作業を２回行って新たな右辺ベクトル $\mathbf{b}$ に対する解を得ることができる。別の言い方をすると、ガウスの消去法では右辺ベクトル $\mathbf{b}$ はあらかじめわかっていなければならなかったが、LU 分解ではその必要はない。LU 分解は右辺ベクトル $\mathbf{b}$ が与えられないときに $A=LU$ と分解しておくことと言うこともできる。

LU 分解の例

Step1 の最後より、前進消去が終わったときの右上三角行列を U と置くと、

$U=D_1L_2L_1A$

よって、

$$A=\underbrace{(D_1L_2L_1)^{-1}}_{=L}U=L_1^{-1}L_2^{-1}D_1^{-1}U=LU$$

と LU 分解できる。U は前進消去終了時の行列だが、L は前進消去した過程の逆の操作なので、第何式を何倍して第何式から引いたかを左下三角行列に記録していくだけであり、次のようになる。

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

$$=\underset{L}{\begin{pmatrix}1 & 0 & 0\\ 2 & 1 & 0\\ -1 & -3 & 1\end{pmatrix}}\underset{U}{\begin{pmatrix}2 & 1 & 1\\ 0 & -1 & -2\\ 0 & 0 & -4\end{pmatrix}}$$

$$=\underset{L}{\begin{pmatrix}1 & 0 & 0\\ 2 & 1 & 0\\ -1 & -3 & 1\end{pmatrix}}\underset{D}{\begin{pmatrix}2 & 0 & 0\\ 0 & -1 & 0\\ 0 & 0 & -4\end{pmatrix}}\underset{U}{\begin{pmatrix}1 & 1/2 & 1/2\\ 0 & 1 & 2\\ 0 & 0 & 1\end{pmatrix}}$$

計算時間のオーダー予測（LU 分解）

ガウスの消去法で左下三角部分を 0 にする時間を見積もる。

積和演算(積[乗除算]を取って和[加減算]を取る計算 1 回分$ab+c$)の回数から計算時間のオーダーを予測(order estimation)する。

for $k=1,2,\cdots,N-1$ {　　(列のループ)

　for $i=k+1,k+2,\cdots,N$ {

　　$w_{ik}=a_{ik}/a_{kk}$

　　for $j=k+1,k+2,\cdots,N+1$ {

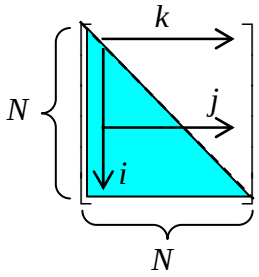
　　$a'_{ij}=a_{ij}-w_{ik}a_{kj}$

　　}

　}

}

右辺ベクトルも考慮



積和演算の回数は

$$\sum_{k=1}^{N-1}\sum_{i=k+1}^{N}\left(1+\sum_{j=k+1}^{N+1}1\right)=\sum_{k=1}^{N-1}\sum_{i=k+1}^{N}\left(1+(N+1-k)\right)=\sum_{k=1}^{N-1}\sum_{i=k+1}^{N}\left(N-k+2\right)$$

$$=\sum_{k=1}^{N-1}\left(N-k+2\right)\sum_{i=k+1}^{N}1=\sum_{k=1}^{N-1}\left(N-k+2\right)(N-k)$$

$$=\sum_{k=1}^{N-1}\left(N^2-Nk+2N-Nk+k^2-2k\right)$$

$$=\sum_{k=1}^{N-1}\left\{k^2-2(N+1)k+N^2+2N\right\}$$

$$=\sum_{k=1}^{N-1}k^2-2(N+1)\sum_{k=1}^{N-1}k+(N^2+2N)\sum_{k=1}^{N-1}1$$

$$=\frac{(N-1)N(2N-1)}{6}-2(N+1)\frac{(N-1)N}{2}+(N^2+2N)(N-1)$$

$$=\frac{N^3}{3}+\cdots=O(N^3)$$

公式:

```
In[1]:= Sum[i, {i, 1, n}]

Out[1]= 1/2 n (1 + n)

In[2]:= Sum[i^2, {i, 1, n}]

Out[2]= 1/6 n (1 + n) (1 + 2 n)
```

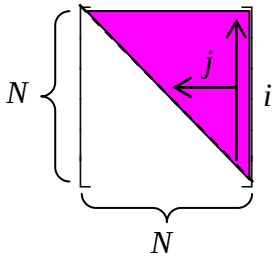
計算時間のオーダー予測（後退代入）

後退代入のオーダーを見積もる。U を右上三角行列、**x** を未知列ベクトル、**c** を既知列ベクトルとし、

$U\mathbf{x}=\mathbf{c}$

から **x** を求める。

$$x_i=\frac{1}{u_{ii}}\left(c_i-\sum_{j=i+1}^{N}u_{ij}x_j\right)\qquad(i=N,\cdots,1)$$



for $i=N,\cdots,1$ {

$x_i:=c_i$

for $j=i+1,\cdots,N$ {

$x_i:=x_i-u_{ij}x_j$

}

$x_i := \frac{x_i}{u_{ii}}$

}

積和演算の回数は

$$\sum_{i=1}^{N}\left(\sum_{j=i+1}^{N}1+1\right)=\sum_{i=1}^{N}\{(N-i)+1\}=(N+1)\sum_{i=1}^{N}1-\sum_{i=1}^{N}i$$

$$=(N+1)N-\frac{N(N+1)}{2}=\frac{N(N+1)}{2}$$

$$=\frac{N^2}{2}+\cdots=O(N^2)$$

2.3 Gauss-Jordan 法

Gauss の消去法では最初に前進消去で左下三角部分を 0 にし、次に後退代入で解を求めたが、前進消去過程で最初から対角行列を得ようと右上三角部分も 0 にしてしまう方法を Gauss-Jordan 法と言う。下のように Gauss-Jordan 法のオーダーは$N^3+\cdots$となる。Gauss の消去法、つまり LU 分解と後退代入を合わせたものでも後退代入は$O(N^2/2)$なので Gauss の消去のオーダーは$\frac{N^3}{3}+\cdots$となって 1/3 の係数があるので単純に考えても Gauss-Jordan 法はあまり優れていない。まあ、係数 1/3 の係数なんてたいしたことないのだが、LU 分解は後からでも$O(N^2/2)$で他の右辺ベクトルに対する解が得られるという点でも優れている。

計算時間のオーダー予測(Gauss-Jordan 法)

LU 分解の場合の計算とほとんど同じだが、LU 分解での$i=k+1,k+2,\cdots,N$が$i=1,\cdots,k-1,k+1,\cdots,N$(実際関係するのは i の値でなく、ループ回数なので、$i=1,\cdots,N-1$)となるだけである。

積和演算の回数は

$$\sum_{k=1}^{N-1}\sum_{i=1}^{N-1}\left(1+\sum_{j=k+1}^{N+1}1\right)=\sum_{k=1}^{N-1}\sum_{i=1}^{N-1}\left(1+(N+1-k)\right)=\sum_{k=1}^{N-1}\sum_{i=1}^{N-1}\left(N-k+2\right)$$

$$=\sum_{k=1}^{N-1}\left(N-k+2\right)\sum_{i=1}^{N-1}1=\sum_{k=1}^{N-1}\left(N-k+2\right)(N-1)$$

$$=(N-1)\sum_{k=1}^{N-1}\left(N-k+2\right)$$

$$=(N-1)\left\{\left(N+2\right)\sum_{k=1}^{N-1}1-\sum_{k=1}^{N-1}k\right\}$$

$$=(N-1)\left\{\left(N+2\right)(N-1)-\frac{(N-1)N}{2}\right\}$$

$$=\frac{N^3}{2}+\cdots=O(N^3)$$

文献[2]によると、$N^3+\cdots$ と N^3 の係数は 1 になると書かれているが、上の計算より、$N^3/2$ と思われる。

3. 反復法

直接解法の計算時間$O(N^3)$は膨大で、現実問題としてもっと速く求めたいという要求がある。また、直接解法では$O(N^3)$の計算時間で最後まで解き終わるまで何も得られない。ここではより速く、途中で計算を止めても少しは解ベクトルに近いベクトルを得ることが目的である。本節で説明する反復法は、初期ベクトルをだんだん解ベクトルに収束させていく方法で、途中で計算を止めてもまあまあ良い解が得られているという点がメリットである。

$A\mathbf{x}=\mathbf{b}$において、$A=S-T$と分解すると、

$(S-T)\mathbf{x}=\mathbf{b}$

$S\mathbf{x}=T\mathbf{x}+\mathbf{b}$

となる。

$\mathbf{x}^{(0)}$を初期ベクトルとして、

$$S\mathbf{x}^{(k+1)}=T\mathbf{x}^{(k)}+\mathbf{b} \qquad (1)$$

を繰り返し適用し、$\mathbf{x}^{(k+1)}$があるベクトルに収束したらそれは$A\mathbf{x}=\mathbf{b}$の解である。なぜならば、そのとき、$\mathbf{x}^{(k+1)}=\mathbf{x}^{(k)}=\tilde{\mathbf{x}}$となっており、$S\tilde{\mathbf{x}}=T\tilde{\mathbf{x}}+\mathbf{b}$，$(S-T)\tilde{\mathbf{x}}=\mathbf{b}$，$A\tilde{\mathbf{x}}=\mathbf{b}$となって与えられた方程式を満たすからである。

ただし、式(1)を見てもわかるように、各ステップで次のステップのために$\mathbf{x}^{(k+1)}$を計算しなければならないから、Sの逆行列が簡単に計算できる、あるいは直接解法を用いずに簡単に$\mathbf{x}^{(k+1)}$を求めることができなければ意味が無い（それが$A=S-T$と分解した目的である）。極端な場合には$S=A$，$T=0$と選べば1ステップで解が求まるが、それでは直接解法を用いるのと同じことであり意味がない。また、このアイデアは簡単であるが、問題は$\mathbf{x}^{(k+1)}$が収束するかどうかである。つまり、式(1)のアイデアは簡単であるが、Sの逆が簡単に計算でき、かつ解が収束するようにS,Tを選ばなければならない。

解の収束と誤差について

厳密解$\mathbf{x}$が満たす

$S\mathbf{x}=T\mathbf{x}+\mathbf{b}$

から式(1)を辺々引くと

$S(\mathbf{x}-\mathbf{x}^{(k+1)})=T(\mathbf{x}-\mathbf{x}^{(k)})$

ここで、ステップkの誤差ベクトル$\mathbf{e}^{(k)}=\mathbf{x}-\mathbf{x}^{(k)}$を導入して、

$S\mathbf{e}^{(k+1)}=T\mathbf{e}^{(k)}$

$\mathbf{e}^{(k+1)}=S^{-1}T\mathbf{e}^{(k)}$

つまり、誤差が0に収束する、別の言い方をすると$\mathbf{x}^{(k+1)}$が真の解に収束するには$S^{-1}T$の固有値$\lambda_i\ (i=1,\cdots,N)$の絶対値$|\lambda_i|$の最大値$\max_i|\lambda_i|$が1未満であれば良い。なぜならば、固有ベクトル

$\mathbf{u}_i$ ($(S^{-1}T)\mathbf{u}_i = \lambda_i \mathbf{u}_i$)は完備(complete)なので任意のベクトルはそれらの線形結合で表現することができるから、

$$\mathbf{e}^{(0)} = \sum_{i=1}^{N} c_i \mathbf{u}_i$$

と書くことができる。

$$\mathbf{e}^{(k)} = \sum_{i=1}^{N} c_i (S^{-1}T)^k \mathbf{u}_i = \sum_{i=1}^{N} c_i \lambda_i{}^k \mathbf{u}_i$$

ノルムの性質より、

$$\left\|\mathbf{e}^{(k)}\right\| \le \sum_{i=1}^{N} |c_i| \left|\lambda_i{}^k\right| \left\|\mathbf{u}_i\right\|$$

よって、$|\lambda_i| < 1$ならば誤差は 0 に収束する。

絶対値$|\lambda_i|$の最大値$\max_i |\lambda_i|$は**スペクトル半径(spectral radius)**と呼ばれ、次のように書く。

$$\rho(S^{-1}T) = \max_i |\lambda_i|$$

ステップ数が多いとき、収束の程度は上のパラメータが支配的となり、最悪値を与える。
誤差を10^{-p}に減少させるのに必要な反復回数Kはρをスペクトル半径として

$$\rho^K = 10^{-p}$$
$$\ln \rho^K = \ln 10^{-p}$$
$$K \ln \rho = -p \ln 10$$
$$K = \frac{p \ln 10}{-\ln \rho}$$

と見積もられる。

初期ベクトルについて

初期ベクトル$\mathbf{x}^{(0)}$は普通何も情報が分かっていないときは

$$x_i^{(0)} = \frac{b_i}{a_{ii}} \qquad (i = 1, \cdots, N)$$

と取る。なぜならば、よく出てくる物理、工学の問題では対角成分が一番大きく、左下と右上部分は小さくなる。すると、上の仮定はあまり間違っていないと考えられるからである。
他にも、物理、工学の問題で解のおおまかな形があらかじめ予想できる場合はそれを初期値とすると収束が速くなる。この作業を**前処理(pre-process, pre-conditioning)**と呼ぶ。例えば、アンテナの問題を考える。自由空間中のダイポールアレーのモーメント法解析などでは各ダイポール上の電流分布は相互結合を無視した場合、つまり 1 つのダイポールが自由空間に置かれている場合の電流分布を全てのアンテナの初期値としておくと、後は相互結合の補正が行われるだけなので、

収束は速くなる。

3.1 Jacobi 法

$A = L + D + U$

と表現できたとする。L, D, U はそれぞれ「対角成分が 0 の左下三角行列(Aの左下三角部分)」、「対角成分のみ値を持つ行列(Aの対角部分)」、「対角成分が 0 の右上三角行列(Aの右上三角部分)」を表す。

ヤコビ法(Jacobi method)と呼ばれる方法では、S の逆が簡単に計算できる最も簡単な例で、$S = D$, $T = -L - U$ と取る。

$D\mathbf{x}^{(k+1)} = (-L - U)\mathbf{x}^{(k)} + \mathbf{b}$

要素ごとの演算を書くと、

$$a_{ii} x_i^{(k+1)} = -\sum_{j=1}^{i-1} a_{ij} x_j^{(k)} - \sum_{j=i+1}^{N} a_{ij} x_j^{(k)} + b_i \qquad (i = 1, \cdots, N) \qquad (2)$$

各ステップで新たな$\mathbf{x}^{(k+1)}$を得るための計算量は$O(N^2)$の行列とベクトルの積、続いて$O(N^2)$の後退代入を行うので$O(N^2)$である。

Jacobi 法の収束について

Aが対角優位(diagonal dominant)であれば収束する。

一般の問題については言えないが、文献[2](p.626)より、Dirichlet 条件を課した楕円型微分方程式の離散解法では行列サイズNの問題に対して$\rho \cong 1 - \frac{\pi^2}{2N}$となるので、$K \cong \frac{1}{2} pN$ となるそうである。つまり、各ステップで$O(N^2)$であり、それを$O(N)$繰り返すので結局$O(N^3)$の計算時間がかかり、理論的興味はあるが実用的にはあまり魅力的でない。

3.2 Gauss-Seidel 法

Jacobi 法では式(2)の左辺の$\mathbf{x}^{(k+1)}$の更新のときに全て右辺のように 1 つ前のステップの$\mathbf{x}^{(k)}$を用いていたが、$i = 1, \cdots, N$ と順番に計算するならば、要素$x_i^{(k+1)}$の計算のときに今のステップで求まった$x_l^{(k+1)} (l < i)$をすぐに使ってしまう方法も考えられる。驚くべきことに、これは悪い方法ではない。これは**ガウス・ザイデル法(Gauss-Seidel method)**と呼ばれ、要素ごとの演算を書くと次のようになる。

$$a_{ii} x_i^{(k+1)} = -\sum_{j=1}^{i-1} a_{ij} x_j^{(k+1)} - \sum_{j=i+1}^{N} a_{ij} x_j^{(k)} + b_i \qquad (i = 1, \cdots, N)$$

$$\sum_{j=1}^{i} a_{ij} x_i^{(k+1)} = -\sum_{j=i+1}^{N} a_{ij} x_j^{(k)} + b_i$$

これは行列表現すると、$(L+D)\mathbf{x}^{(k+1)} = -U\mathbf{x}^{(k)} + \mathbf{b}$ となる。つまり、$S = L + D$, $T = -U$ と取ったことに等しい。

Gauss-Seidel 法の収束について

A が対称正定値行列であれば収束する。

一般の問題については言えないが、文献[2](p.626)より、Dirichlet 条件を課した楕円型微分方程式の離散解法では行列サイズ N の問題に対して $\rho \cong 1 - \frac{\pi^2}{N}$ となるので、$K \cong \frac{1}{4} pN$ となるそうである。つまり、Jacobi 法に比べて係数が 1/2 となっており、同じ精度を得るためのステップ数は半分になるが、オーダー的には改善が見られない。

3.3 SOR 法(Successive Overrelaxation method)

ガウス・ザイデル法は

$(L+D)\mathbf{x}^{(k+1)} = -U\mathbf{x}^{(k)} + \mathbf{b}$

と書けた。次のように変形する。

$D\mathbf{x}^{(k+1)} = -L\mathbf{x}^{(k+1)} - U\mathbf{x}^{(k)} + \mathbf{b}$

$\mathbf{x}^{(k+1)} = -D^{-1}L\mathbf{x}^{(k+1)} - D^{-1}U\mathbf{x}^{(k)} + D^{-1}\mathbf{b}$

$\mathbf{x}^{(k+1)}$ の項でなぜ L を基準にしなかったかと言うと、$O(N)$ の D^{-1} は後退代入を利用する $O(N^2)$ の L^{-1} よりも簡単だからである。

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + (\underbrace{-D^{-1}L\mathbf{x}^{(k+1)} - D^{-1}U\mathbf{x}^{(k)} + D^{-1}\mathbf{b}}_{\tilde{\mathbf{x}}^{(k+1)}} - \mathbf{x}^{(k)})$$

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \delta$$

ここで、

$\delta = \tilde{\mathbf{x}}^{(k+1)} - \mathbf{x}^{(k)}$

は 1 回の反復での変化量を表す。

SOR 法(Successive Overrelaxation method, 逐次加速緩和法, 逐次過緩和法)とは、加速緩和係数、あるいは過緩和パラメータ(overrelaxation parameter)と呼ばれる ω を導入して、上の式の修正 δ を大きくしたり小さくしたりする方法である。当然 $\omega = 1$ のときはガウス・ザイデル法に一致する。

$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \omega\delta$

$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \omega(-D^{-1}L\mathbf{x}^{(k+1)} - D^{-1}U\mathbf{x}^{(k)} + D^{-1}\mathbf{b} - \mathbf{x}^{(k)})$

$D\mathbf{x}^{(k+1)} = D\mathbf{x}^{(k)} + \omega(-L\mathbf{x}^{(k+1)} - U\mathbf{x}^{(k)} + \mathbf{b} - D\mathbf{x}^{(k)})$

$(D+\omega L)\mathbf{x}^{(k+1)} = \left[(1-\omega)D - \omega U\right]\mathbf{x}^{(k)} + \omega\mathbf{b}$

上の表現は文献[1]の p.328 の式(27)と一致する。新たな $\mathbf{x}^{(k+1)}$ を得るための計算量は $O(N^2)$ の三角行列(下記参照)とベクトルの積、続いて $O(N^2)$ の後退代入を行うので $O(N^2)$ である。

SOR 法の収束について

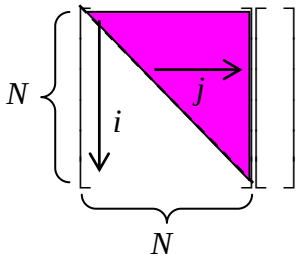
Aが対称正定値行列であり、$0<\omega<2$であれば収束する。

一般の問題については言えないが、文献[2](p.627)より、Dirichlet 条件を課した楕円型微分方程式の離散解法では行列サイズ N の問題に対して $\rho \cong 1-\frac{2\pi}{\sqrt{N}}$ となるので、$K \cong \frac{1}{3} p\sqrt{N}$ となるそうである。つまり、Jacobi 法や Gauss-Seidel 法に比べてオーダーが小さくなっている。

計算時間のオーダー予測（三角行列とベクトルの積）

$U\mathbf{x}$

$$x_i' = \sum_{j=i}^{N} u_{ij} x_j \qquad (i = 1, \cdots, N)$$



for $i = 1, \cdots, N$ {

$x_i' := 0$

for $j = i, \cdots, N$ {

$x_i' := x_i' + u_{ij} x_j$

}

}

積和演算の回数は

$$\sum_{i=1}^{N}\sum_{j=i}^{N} 1 = \sum_{i=1}^{N}\{N-(i-1)\} = \sum_{i=1}^{N}(N+1-i)$$

$$= (N+1)\sum_{i=1}^{N} 1 - \sum_{i=1}^{N} i$$

$$= (N+1)N - \frac{N(N+1)}{2} = \frac{N(N+1)}{2}$$

$$= \frac{N^2}{2} + \cdots = O(N^2)$$

3.4 共役傾斜法(CG 法, Conjugate Gradient method)

3.4.1 まえがき

有限要素法(FEM, Finite Element Method)では微分方程式(その解を y とする)を直接解く代わりに、**汎関数(functional)**と呼ばれる関数 y を引数とする関数 F[y](普通の関数と明示的に区別するために、引数の指定には()でなく、[]を用いる)を考える。つまり、F は関数の関数である。F は１つの極小を持ち、F が極小となるときの y は元の微分方程式の解となっているような F を導出し(汎関数の作成)、微分方程式を解く問題を汎関数 F を最小化する問題に置き換えて解く。汎関数 F の関数 y による変化率を**変分(variation)**(関数の変数による変化率は**微分(derivation)**と言うのに対応)と言い、ある法則(例えば微分方程式)から y を直接導く代わりに、その汎関数 F[y]の極値、停留点を求めて間接的に y を求める方法を**変分法(Variational method)** [5] [6]と言う。このような問題はよく扱うので、変分学と言われる数学の一分野がある。変分学で扱う問題を変分問題と言う。

また、変分という概念は物理学でよく使われており、ミクロに現象を観察した後でマクロな原理を記述する方法がよく行われる。つまり、光学、電磁波の例で言うと、屈折の法則を記述するスネルの法則(Snell's law)「屈折率 n1 の媒質から n2 の媒質に光が入射するとき $n_1 \sin\theta_i = n_2 \sin\theta_t$ の関係を満たす」は現象をミクロに観察して導出した法則であるが、その現象をマクロにとらえた説明として、フェルマーの原理(Fermat's principle)「光線は到達時間が最小になるような経路を選ぶ」がある。後者の表現は曖昧に思えるかもしれないが、そんなことはなく、後者から前者の法則を導くことができ等価である。どちらかと言うと後者の表現は問題を厳密に設定していないという意味で、前者よりもより多くの情報を網羅する。フェルマーの原理はあまりにマクロに現象をとらえているので初学者にとっては非常に納得しにくく、ミクロな現象が見えてこないので分かりにくいが、このようによりマクロに成り立ち、それからミクロな法則を簡単に導くことができる法則を見出すのが物理の発展した形と見なされている[4]。フェルマーの原理は経路 C という関数が引数の汎関数 F[C]と見なすことができ、これが最小となるCが光の経路ということになる。他にも物理学で知られる運動方程式、マクスウェルの方程式、シュレーディンガー方程式などのさまざまな法則に対する汎関数が知られており、物理学、特に物理学の中の１分野である解析力学でよく用いられる。

3.4.2 CG 法における汎関数

前おきが長くなったが、CG 法も概念は変分法と同じである。方程式

$$A\mathbf{x} = \mathbf{b}$$

を直接解く代わりに

$$f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T A\mathbf{x} - \mathbf{x}^T\mathbf{b} = \frac{1}{2}\mathbf{x}^T A\mathbf{x} - \mathbf{b}^T\mathbf{x}$$

が極小値を取る $\mathbf{x}$ を探索することによって、$A\mathbf{x} = \mathbf{b}$ を満たす $\mathbf{x}$ を求めるのである。$f(\mathbf{x})$ は多変数の２次関数(２次形式)である。$f(\mathbf{x})$ が極値を取るとき($\nabla f(\mathbf{x}) = 0$)、$\mathbf{x}$ は $A\mathbf{x} = \mathbf{b}$ を満たす。つまり、$\mathbf{x}$ は $A\mathbf{x} = \mathbf{b}$ の解になっているのである。ただし、そのためには $f(\mathbf{x})$ が唯一の極値を持た

なければならない(数値探索などしたときに2つ極値が見つかったらどちらが本当の解かわからないから。また、その場合にはまだ $A\mathbf{x}=\mathbf{b}$ に $\mathbf{x}$ を代入して解なのかどうか確認すれば済むが、峠の尾根、または馬の背中にのせる鞍のように極値が無限にある場合にはどうしようもないから)。$f(\mathbf{x})$ が唯一の極値を持つための条件は A が対称正定値行列であることが必要十分である。$f(\mathbf{x})$ は極値を与える $\mathbf{x}$ が $A\mathbf{x}=\mathbf{b}$ の解となる汎関数と言うこともできる。

CG 法による行列方程式の解法は、行列方程式の解法と言うよりは関数の最小化問題と言った方がわかりやすいかもしれない。

3.4.3 CG 法アルゴリズム

初期値設定

$\varepsilon := 10^{-4}$

$\mathbf{x}_0$ (初期解ベクトル)

$\mathbf{r}_0 := \mathbf{b} - A\mathbf{x}_0$ (初期残差[勾配]ベクトル)

$\mathbf{p}_0 := \mathbf{r}_0$ (初期探索方向ベクトル)

$k := 0$

while ($|\mathbf{r}_k| > \varepsilon|\mathbf{b}|$) {

$\alpha_k := \dfrac{\mathbf{p}_k{}^T\mathbf{r}_k}{\mathbf{p}_k{}^T A\mathbf{p}_k}$ ($\alpha_k := \dfrac{\mathbf{r}_k{}^T\mathbf{r}_k}{\mathbf{p}_k{}^T A\mathbf{p}_k}$)

$\mathbf{x}_{k+1} := \mathbf{x}_k + \alpha_k\mathbf{p}_k$

$\mathbf{r}_{k+1} := \mathbf{b} - A\mathbf{x}_{k+1}$ ($\mathbf{r}_{k+1} := \mathbf{r}_k - \alpha_k A\mathbf{p}_k$)

$\beta_k := -\dfrac{\mathbf{r}_{k+1}{}^T A\mathbf{p}_k}{\mathbf{p}_k{}^T A\mathbf{p}_k}$ ($\beta_k := \dfrac{\mathbf{r}_{k+1}{}^T\mathbf{r}_{k+1}}{\mathbf{r}_k{}^T\mathbf{r}_k}$)

$\mathbf{p}_{k+1} := \mathbf{r}_{k+1} + \beta_k\mathbf{p}_k$

$k := k + 1$

}

解説

(1)

$\mathbf{x}_k$ の位置から $\mathbf{p}_k$ 方向に探索し最小となる位置 $\mathbf{x}_{k+1}$ まで移動する。つまり、

$f(\mathbf{x}_k + \alpha\mathbf{p}_k)$

の値が最小となる $\alpha = \alpha_k$ を求め、

$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k\mathbf{p}_k$

とする。

その α_k は次のように求めることができる。

$$\frac{\partial}{\partial \alpha} f(\mathbf{x}_k + \alpha \mathbf{p}_k) = 0$$

ここで、

$$\begin{aligned} f(\mathbf{x}_k + \alpha \mathbf{p}_k) &= \frac{1}{2}(\mathbf{x}_k + \alpha \mathbf{p}_k)^T A(\mathbf{x}_k + \alpha \mathbf{p}_k) - (\mathbf{x}_k + \alpha \mathbf{p}_k)^T \mathbf{b} \\ &= \frac{1}{2}(\mathbf{x}_k{}^T + \alpha \mathbf{p}_k{}^T)(A\mathbf{x}_k + \alpha A\mathbf{p}_k) - (\mathbf{x}_k{}^T + \alpha \mathbf{p}_k{}^T)\mathbf{b} \\ &= \frac{1}{2}(\mathbf{x}_k{}^T A\mathbf{x}_k + \alpha \mathbf{x}_k{}^T A\mathbf{p}_k + \alpha \mathbf{p}_k{}^T A\mathbf{x}_k + \alpha^2 \mathbf{p}_k{}^T A\mathbf{p}_k) - (\mathbf{x}_k{}^T \mathbf{b} + \alpha \mathbf{p}_k{}^T \mathbf{b}) \\ &= \frac{1}{2}\mathbf{x}_k{}^T A\mathbf{x}_k - \mathbf{x}_k{}^T \mathbf{b} + \frac{1}{2}(\alpha \mathbf{x}_k{}^T A\mathbf{p}_k + \alpha \mathbf{p}_k{}^T A\mathbf{x}_k + \alpha^2 \mathbf{p}_k{}^T A\mathbf{g}_k) - \alpha \mathbf{p}_k{}^T \mathbf{b} \\ &= f(\mathbf{x}_k) + \frac{1}{2}(\alpha \mathbf{x}_k{}^T A\mathbf{p}_k + \alpha \mathbf{p}_k{}^T A\mathbf{x}_k + \alpha^2 \mathbf{p}_k{}^T A\mathbf{p}_k) - \alpha \mathbf{p}_k{}^T \mathbf{b} \end{aligned}$$

$$\frac{1}{2}(\mathbf{x}_k{}^T A\mathbf{p}_k + \mathbf{p}_k{}^T A\mathbf{x}_k + 2\alpha \mathbf{p}_k{}^T A\mathbf{p}_k) - \mathbf{p}_k{}^T \mathbf{b} = 0$$

$$\underline{\mathbf{x}_k{}^T A\mathbf{p}_k} + \underline{\mathbf{p}_k{}^T A\mathbf{x}_k} + 2\alpha \mathbf{p}_k{}^T A\mathbf{p}_k - 2\mathbf{p}_k{}^T \mathbf{b} = 0$$

スカラーの転置はまたスカラーであり、転置しても値は変わらない。A は対称行列なので、

例えば、$\left(\mathbf{x}_k{}^T A\mathbf{p}_k\right)^T = \mathbf{p}_k{}^T A^T (\mathbf{x}_k{}^T)^T = \mathbf{p}_k{}^T A\mathbf{x}_k$

$$2\mathbf{p}_k{}^T A\mathbf{x}_k + 2\alpha \mathbf{p}_k{}^T A\mathbf{p}_k - 2\mathbf{p}_k{}^T \mathbf{b} = 0$$

$$\alpha \mathbf{p}_k{}^T A\mathbf{p}_k + \mathbf{p}_k{}^T (A\mathbf{x}_k - \mathbf{b}) = 0 \qquad \cdots (A)$$

$$\alpha = -\frac{\mathbf{p}_k{}^T (A\mathbf{x}_k - \mathbf{b})}{\mathbf{p}_k{}^T A\mathbf{p}_k}$$

つまり、

$$\alpha_k = \frac{\mathbf{p}_k{}^T \mathbf{r}_k}{\mathbf{p}_k{}^T A\mathbf{p}_k}$$

とし、

$\mathbf{x}_{k+1} := \mathbf{x}_k + \alpha_k \mathbf{p}_k$

と更新し、$\mathbf{p}_k$ 方向で f が最小となる位置に移動する。

(2)

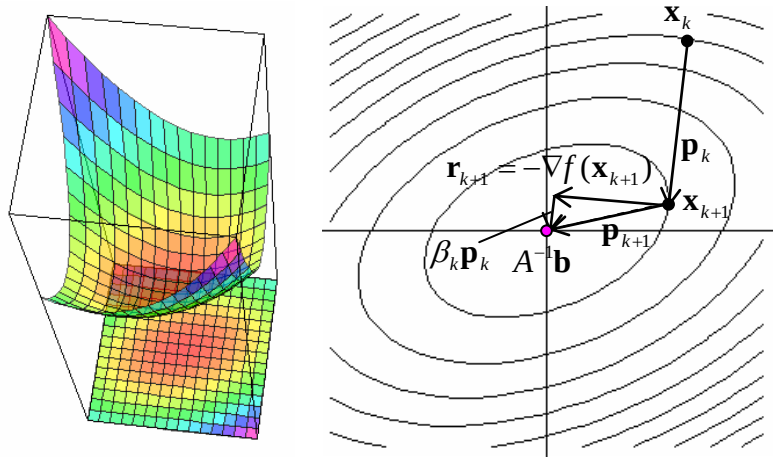
残差ベクトル

$\mathbf{r}_k = \mathbf{b} - A\mathbf{x}_k$

を定義する。

実は、この方向は次のように $\mathbf{x}_k$ の位置での $f(\mathbf{x}_k)$ 勾配の下り方向 $-\nabla f(\mathbf{x}_k)$ である。

$-\nabla f(\mathbf{x}_k) = -(A\mathbf{x}_k - \mathbf{b}) = \mathbf{b} - A\mathbf{x}_k$



式(A)より、

$\alpha \mathbf{p}_k{}^T A\mathbf{p}_k + \mathbf{p}_k{}^T (A\mathbf{x}_k - \mathbf{b}) = 0$

$\mathbf{p}_k{}^T A(\underbrace{\mathbf{x}_k + \alpha\mathbf{p}_k}_{\mathbf{x}_{k+1}} - A^{-1}\mathbf{b}) = 0$

$\mathbf{p}_k{}^T A(A^{-1}\mathbf{b} - \mathbf{x}_{k+1}) = 0$

$\mathbf{p}_{k+1}$ はこの方向(共役な方向)を向かせたい

解がある方向は $\mathbf{p}_k$ と A 直交する。その方向を共役な方向と言い、共役な方向に動けば今 $\mathbf{p}_k$ 方向に動いて最適化したことが無駄にならない（最急傾斜法, Steepest Descent method では放物体の断面が円のときは良いが、楕円のときはそれまでの方向での最適化の労力がいくらか無駄になる)。

$\mathbf{p}_k{}^T (\mathbf{b} - A\mathbf{x}_{k+1}) = 0$

$\mathbf{p}_k{}^T \mathbf{r}_{k+1} = 0$ ・・・ (B)

$\mathbf{p}_{k+1} = \mathbf{r}_{k+1} + \beta_k \mathbf{p}_k$

$\mathbf{p}_{k+1}$ は $\mathbf{p}_k$ と A 直交するようにする。

$\mathbf{p}_{k+1}{}^T A\mathbf{p}_k = (\mathbf{r}_{k+1} + \beta_k \mathbf{p}_k)^T A\mathbf{p}_k = (\mathbf{r}_{k+1}{}^T + \beta_k \mathbf{p}_k{}^T)A\mathbf{p}_k$

$= \mathbf{r}_{k+1}{}^T A\mathbf{p}_k + \beta_k \mathbf{p}_k{}^T A\mathbf{p}_k = 0$

$$\beta_k = -\frac{\mathbf{r}_{k+1}{}^T A\mathbf{p}_k}{\mathbf{p}_k{}^T A\mathbf{p}_k}$$

$$\mathbf{r}_{k+1} = \mathbf{b} - A\mathbf{x}_{k+1} = \mathbf{b} - A(\mathbf{x}_k + \alpha_k \mathbf{p}_k) = (\mathbf{b} - A\mathbf{x}_k) + \alpha_k A\mathbf{p}_k$$
$$= \mathbf{r}_k - \alpha_k A\mathbf{p}_k$$

$$\alpha_k = \frac{\mathbf{p}_k{}^T \mathbf{r}_k}{\mathbf{p}_k{}^T A\mathbf{p}_k},\ \ \beta_k = -\frac{\mathbf{r}_{k+1}{}^T A\mathbf{p}_k}{\mathbf{p}_k{}^T A\mathbf{p}_k}$$

と求まったが、計算負荷が減るようにより簡単な表現を求める。

■

$$\mathbf{p}_k{}^T \mathbf{r}_k = (\mathbf{r}_k + \beta_k \mathbf{p}_{k-1})^T \mathbf{r}_k = (\mathbf{r}_k{}^T + \beta_k \mathbf{p}_{k-1}{}^T)\mathbf{r}_k = \mathbf{r}_k{}^T \mathbf{r}_k + \beta_k \underbrace{\mathbf{p}_{k-1}{}^T \mathbf{r}_k}_{=0} = \mathbf{r}_k{}^T \mathbf{r}_k$$

式(B)より

■

$$\mathbf{r}_{k+1}{}^T \mathbf{r}_{k+1} = \mathbf{r}_{k+1}{}^T (\mathbf{r}_k - \alpha_k A\mathbf{p}_k) = \underbrace{\mathbf{r}_{k+1}{}^T \mathbf{r}_k}_{=0} - \alpha_k \mathbf{r}_{k+1}{}^T A\mathbf{p}_k$$
$$= -\alpha_k \mathbf{r}_{k+1}{}^T A\mathbf{p}_k$$
$$= -\frac{\mathbf{p}_k{}^T \mathbf{r}_k}{\mathbf{p}_k{}^T A\mathbf{p}_k}(-\beta_k \mathbf{p}_k{}^T A\mathbf{p}_k) = \beta_k \mathbf{p}_k{}^T \mathbf{r}_k = \beta_k \mathbf{r}_k{}^T \mathbf{r}_k$$

$$\beta_k = \frac{\mathbf{r}_{k+1}{}^T \mathbf{r}_{k+1}}{\mathbf{r}_k{}^T \mathbf{r}_k}$$

よって、

$$\alpha_k = \frac{\mathbf{r}_k{}^T \mathbf{r}_k}{\mathbf{p}_k{}^T A\mathbf{p}_k},\ \ \beta_k = \frac{\mathbf{r}_{k+1}{}^T \mathbf{r}_{k+1}}{\mathbf{r}_k{}^T \mathbf{r}_k}$$

$\mathbf{r}_i$の直交性と$\mathbf{p}_i$の共役性

$$\mathbf{r}_k{}^T \mathbf{r}_{k+1} = \mathbf{r}_k{}^T (\mathbf{r}_k - \alpha_k A\mathbf{p}_k) = \mathbf{r}_k{}^T \mathbf{r}_k - \alpha_k \mathbf{r}_k{}^T A\mathbf{p}_k$$
$$= \mathbf{r}_k{}^T \mathbf{r}_k - \frac{\mathbf{p}_k{}^T \mathbf{r}_k}{\mathbf{p}_k{}^T A\mathbf{p}_k}\mathbf{r}_k{}^T A\mathbf{p}_k = \mathbf{r}_k{}^T \mathbf{r}_k - \frac{\mathbf{p}_k{}^T \mathbf{r}_k \mathbf{r}_k{}^T A\mathbf{p}_k}{\mathbf{p}_k{}^T A\mathbf{p}_k}$$
$$= \mathbf{r}_k{}^T \mathbf{r}_k - \frac{\mathbf{p}_k{}^T (\mathbf{r}_k \mathbf{r}_k{}^T) A\mathbf{p}_k}{\mathbf{p}_k{}^T A\mathbf{p}_k} = \mathbf{r}_k{}^T \mathbf{r}_k - (\mathbf{r}_k \mathbf{r}_k{}^T)\frac{\mathbf{p}_k{}^T A\mathbf{p}_k}{\mathbf{p}_k{}^T A\mathbf{p}_k} = 0$$

スカラーだから

$\mathbf{r}_i$は直前のものに直交に、$\mathbf{p}_i$は直前のものに共役にしているが、実際には次のように互いに直交および互いに共役になる。

$\mathbf{r}_i^T \mathbf{r}_j = 0 \quad (i \neq j)$　　互いに直交

$\mathbf{p}_i^T A \mathbf{p}_j = 0 \quad (i \neq j)$　　互いに共役

$\mathbf{r}_i$の直交性の証明

帰納法で証明する。

(i) 仮定($\leq k$)

$i \leq k, j \leq k$で

$\mathbf{r}_i^T \mathbf{r}_j = 0\,(i \neq j)$, $\mathbf{p}_i^T A \mathbf{p}_j = 0\,(i \neq j)$ ($\mathbf{p}_i$の共役性の証明で示す)が成り立つとする。

(ii) 範囲の拡大($\leq k+1$)

$i \leq k$で$\mathbf{r}_{k+1}^T \mathbf{r}_i = 0$が成立することを示す。

$$\mathbf{r}_{k+1}^T \mathbf{r}_i = (\mathbf{r}_k - \alpha_k A\mathbf{p}_k)^T \mathbf{r}_i = (\mathbf{r}_k^T - \alpha_k (A\mathbf{p}_k)^T)\mathbf{r}_i$$
$$= (\mathbf{r}_k^T - \alpha_k \mathbf{p}_k^T A^T)\mathbf{r}_i = (\mathbf{r}_k^T - \alpha_k \mathbf{p}_k^T A)\mathbf{r}_i$$
$$= \underbrace{\mathbf{r}_k^T \mathbf{r}_i}_{=0} - \alpha_k \mathbf{p}_k^T A\mathbf{r}_i = -\alpha_k \mathbf{p}_k^T A\mathbf{r}_i$$

ここで、

$\mathbf{p}_i = \mathbf{r}_i + \beta_{i-1}\mathbf{p}_{i-1}$

$\mathbf{r}_i = \mathbf{p}_i - \beta_{i-1}\mathbf{p}_{i-1}$

より、

$$= -\alpha_k \mathbf{p}_k^T A(\mathbf{p}_i - \beta_{i-1}\mathbf{p}_{i-1})$$
$$= -\alpha_k (\underbrace{\mathbf{p}_k^T A\mathbf{p}_i}_{=0} - \beta_{i-1} \underbrace{\mathbf{p}_k^T A\mathbf{p}_{i-1}}_{=0})$$
$$= 0$$

<Q.E.D>

$\mathbf{p}_i$の共役性の証明

帰納法で証明する。

(i) 仮定($\leq k$)

$i \leq k, j \leq k$ で

$\mathbf{p}_i{}^T A\mathbf{p}_j = 0\ (i \neq j),\ \mathbf{r}_i{}^T\mathbf{p}_j = 0$ が成り立つとする。

(ii) 範囲の拡大($\leq k+1$)

$$\mathbf{p}_{k+1}{}^T A\mathbf{p}_i = (\mathbf{r}_{k+1} + \beta_k\mathbf{p}_k)^T A\mathbf{p}_i = (\mathbf{r}_{k+1}{}^T + \beta_k\mathbf{p}_k{}^T)A\mathbf{p}_i$$
$$= \mathbf{r}_{k+1}{}^T A\mathbf{p}_i + \beta_k\mathbf{p}_k{}^T A\mathbf{p}_i$$
$$= \mathbf{r}_{k+1}{}^T A\mathbf{p}_i - \frac{\mathbf{r}_{k+1}{}^T A\mathbf{p}_k}{\mathbf{p}_k{}^T A\mathbf{p}_k}\mathbf{p}_k{}^T A\mathbf{p}_i \qquad \cdots(A)$$

$i = k$ のとき、

$$(A) = \mathbf{r}_{k+1}{}^T A\mathbf{p}_k - \frac{\mathbf{r}_{k+1}{}^T A\mathbf{p}_k}{\mathbf{p}_k{}^T A\mathbf{p}_k}\mathbf{p}_k{}^T A\mathbf{p}_k = 0$$

次に、

$i \neq k\ (i < k)$のとき、(i)の仮定から、(A) $\mathbf{p}_k{}^T A\mathbf{p}_i$ は 0 になり、

$$(A) = \mathbf{r}_{k+1}{}^T A\mathbf{p}_i$$
$$= (\mathbf{r}_k - \alpha_k A\mathbf{p}_k)^T A\mathbf{p}_i = (\mathbf{r}_k{}^T - \alpha_k\mathbf{p}_k{}^T A)A\mathbf{p}_i$$
$$= \mathbf{r}_k{}^T A\mathbf{p}_i - \alpha_k\mathbf{p}_k{}^T A(A\mathbf{p}_i)$$

Krylov 部分空間

$$\mathcal{K}_n(A,\mathbf{x}) \overset{def.}{=} \mathrm{span}(\mathbf{x}, A\mathbf{x}, A^2\mathbf{x}, \cdots, A^{n-1}\mathbf{x})$$

を定義する。アルゴリズムの式($\mathbf{p}_0 := \mathbf{r}_0, \mathbf{r}_{k+1} := \mathbf{r}_k - \alpha_k A\mathbf{p}_k, \mathbf{p}_{k+1} := \mathbf{r}_{k+1} + \beta_k\mathbf{p}_k$)を見ると、

$\mathbf{p}_n \in \mathcal{K}_{n+1}(A,\mathbf{p}_0) = \mathrm{span}(\mathbf{p}_0, \mathbf{p}_1, \cdots, \mathbf{p}_n)$

$A\mathbf{p}_i \in \mathcal{K}_{i+2}(A,\mathbf{p}_0) \subseteq \mathcal{K}_{k+1}(A,\mathbf{p}_0) = \mathrm{span}(\mathbf{p}_0, \mathbf{p}_1, \cdots, \mathbf{p}_k)$

$\mathbf{r}_k \in \mathcal{K}_{k+1}(A,\mathbf{p}_0)$

より、(i)の仮定から、

$= 0$

<Q.E.D>

共役, A 直交の意味について

$$\mathbf{p}^T A\mathbf{q} = 0 \qquad (1)$$

が成り立つとき、$\mathbf{p}$ と $\mathbf{q}$ は A 直交、あるいは(A に関して)共役であると言う（$A = I$ (単位行列)

のとき、単に直交と言う)。ここで、共役の意味について考えてみよう。Aが対称正値行列(エルミート行列の要素)のとき、ユニタリー行列Qと対角行列$\Lambda = \begin{bmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_N \end{bmatrix} = \mathrm{diag}(\lambda_1, \cdots, \lambda_N)$を用いて

$$Q^T A Q = \Lambda$$

と対角化できるので、

$$A = Q \Lambda Q^T \qquad (2)$$

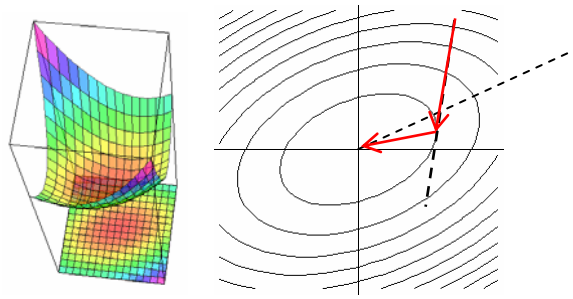
と表現できる。

今、式(1)の考察をするために次の２次形式を考えよう。

$$f(\mathbf{x}) = \mathbf{x}^T A \mathbf{x} \qquad (3)$$

Aは対称正値行列なので、$f(\mathbf{x})$の曲面は任意の２つの軸を選んで(他の軸を固定して)描いた曲面は１つの極値を持つ放物体となる。式(2)を式(3)に代入して、

$$f(\mathbf{x}) = \mathbf{x}^T Q \Lambda Q^T \mathbf{x} = (Q^T \mathbf{x})^T \Lambda Q^T \mathbf{x}$$



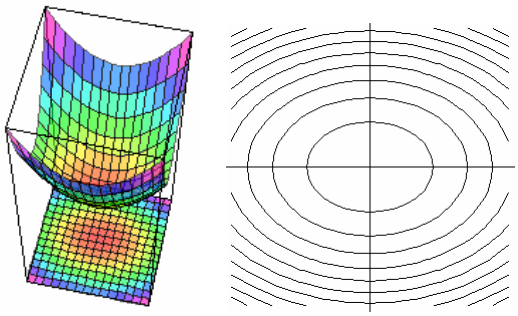
ここで、

$$\mathbf{y} = Q^T \mathbf{x}$$

とおくと、

$$f(\mathbf{x}) = g(\mathbf{y}) = \mathbf{y}^T \Lambda \mathbf{y}$$

この操作は主軸変換であり、$\mathbf{y}$の任意の２つの軸を選んで(他の軸を固定して)描いた$g(\mathbf{y})$は、その２つの軸を含む面での放物体のカット面は楕円であり、その楕円の２つの軸は選んだ２つの軸に一致する。



さらにここで、

$$M = \begin{bmatrix} \sqrt{\lambda_1} & & \\ & \ddots & \\ & & \sqrt{\lambda_N} \end{bmatrix} = \mathrm{diag}(\sqrt{\lambda_1}, \cdots, \sqrt{\lambda_N})$$

とおくと、

$$f(\mathbf{x}) = g(\mathbf{y}) = \mathbf{y}^T MM\mathbf{y} = \mathbf{y}^T M^T M\mathbf{y} = (M\mathbf{y})^T M\mathbf{y}$$

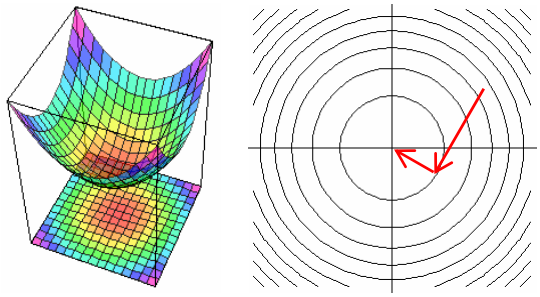
ここで、

$$\mathbf{z} = M\mathbf{y} = MQ^T\mathbf{x}$$

とおくと、

$$f(\mathbf{x}) = g(\mathbf{y}) = h(\mathbf{z}) = \mathbf{z}^T\mathbf{z}$$

この操作は主軸変換して、さらに放物体の断面が楕円でなく、完全な円になるように各座標軸を伸縮して変換したものである。



つまり、式(1)の共役、あるいはA直交$\mathbf{p}^T A\mathbf{q} = 0$が成り立つということは、主軸変換して、さらに放物体の断面が円になるように軸を伸縮させたときに$\mathbf{p}$と$\mathbf{q}$は直交していることを意味する。

放物体の断面が完全な円ならば、放物体のある２軸で$h(\mathbf{z})$を最小化するとき、どの２つの方向でも直交する方向に２回動けば最小化は完了する。放物体のN軸全てで最小化するときも同様で、N 回直交する方向に動けば最小化は完了し、そのとき$h(\mathbf{z})$の真の最小（極小）に辿り着く。CG法ではこのように明示的に主軸変換は行わないが、間接的にこのような動作を実現しているのである。

計算速度とメモリ容量の比較

	Methods	Costs	
		CPU	MEM
Direct method	Gauss-Jordan	$O(N^3)$	$O(N^2)$
	Gaussian Elimination	$O(N^3)$	$O(N^2)$
	LU factorization	$O(N^3)$	$O(N^2)$
Iterative method	Jacobi method	$O(KN^2)$	$O(N^2)$
	Gauss-Seidel method	$O(KN^2)$	$O(N^2)$
	SOR method	$O(KN^2)$	$O(N^2)$
Direct? Iterative?	CG method	$O(N^3)$	$O(N^2)$
	Bi-CG method	$O(N^3)$	$O(N^2)$
	PCG method	$O(N^3) \cong O(N^2)$	$O(N^2)$

Convergence is fast!

反復法の反復回数 Kは最悪の場合無限大である。しかし、前処理によって最初の解を真の解に近い値にすることで収束を速めることができる。CG 法は最悪 N回反復すれば必ず真の解を得ることができる（その意味で反復法と直接法の中間にある）。前処理を行うと収束を速めることができ、様々な前処理法が提案されている。

行列が疎行列（帯行列）の場合には、CPU, MEM コストのオーダーの指数部を全て 1 減らすことができる。

4. 一般の（正方行列を含む）長方形行列の逆行列について

2 章では正則な正方行列の解法について説明した。それに対して、A を $m \times n$ の行列とし、

$$A\mathbf{x} = \mathbf{b}$$

において、条件が未知数よりも多く、A が縦長の長方形行列あるいは、未知数よりも条件の数が少ない横長の長方形行列だった場合はどうなるだろうか[1] [3]。実は、このような一般の長方形行列の方程式に対しても近似的に解を求める理論が確立されている。本章ではその方法の概要を説明する。そのような一般の長方形行列に対する逆行列 $A^{\dagger}$ を**擬似逆行列 (Pseudo Inverse)**または**一般化逆行列 (Generalized Inverse)**あるいは考案者にちなんで**ムーア・ペンローズ逆行列 (Moore-Penrose Inverse)**と呼ぶ。

$$\bar{\mathbf{x}} = A^{\dagger}\mathbf{b}$$

$\bar{\mathbf{x}}$ は $A\mathbf{x} = \mathbf{b}$ に対する誤差ノルム $\|A\mathbf{x} - \mathbf{b}\|$ が最小になる最適解を表す。

(i), (ii), (iii)の 3 つの場合に分けて説明する。

(i) 正則な正方行列の場合（未知数の数 n ＝条件の数 m ）

2 章で説明したように、ガウスの消去法やその他の効率的な数値計算法、または理論として綺麗なのは Cramer の公式（実用的ではないが）を用いて解くことができる。この場合は喜んで

$$A^{\dagger} = A^{-1}$$

とすることができる。

(ii) 縦長の長方形行列の場合（未知数の数 n ＜条件の数 m ）

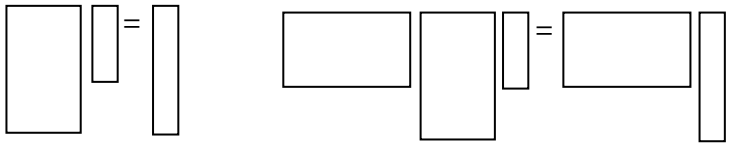
条件の数が未知数よりも多い場合には、(i)の場合のように方程式を解いて解を求めるという考え方ではなく、誤差ノルムを最小にする、つまり $\|A\mathbf{x} - \mathbf{b}\|$ を最小にするように $\mathbf{x}$ の最適解 $\bar{\mathbf{x}}$ を求める。これは数値計算でよく使われる**最小２乗法**と全く同じことである。

解法としては、

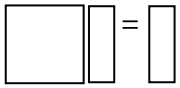
$$A\mathbf{x} = \mathbf{b}$$

の左から A^T を掛け、

$$\left(A^T A\right)\mathbf{x} = A^T\mathbf{b}$$



とする。すると A の列ベクトルが全て線形独立ならば $A^T A$ は正則な正方行列となり、(i)の問題にすることができる。



$$\mathbf{x} = \left(A^T A\right)^{-1} A^T\mathbf{b}$$

よって、

$$A^{\dagger} = \left(A^T A\right)^{-1} A^T$$

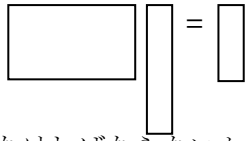
と求めることができる。

（補足）さて、左から A の転置行列をかけて出来た正方行列が正則ならばこのやり方で問題無いが、本当に A^TA は正則なのだろうか？ A^TA は $n\times n$ 行列であり、これは A^T の列空間内、つまり A の行空間内にある。つまり、 A の m 個の n 次元行ベクトルのうち、線形独立なものが n 個含まれていれば、それらのベクトルで張られ、かつ個数が m よりも少ない n 個の n 次元ベクトルは線形独立になり得るから $\mathrm{rank}(A^TA)=n$ 、つまり正則となり得る。
さて、 $\mathrm{rank}(A)=\mathrm{rank}(A^T)$ であるから $\mathrm{rank}(A)<n$ ならば未知数を消去して $\mathrm{rank}(A)=n$ の行列に帰着することができ、再びこの方法を使って $A^\dagger$ を求めることができる。

(iii) 横長の長方形行列の場合（未知数の数 n ＞条件の数 m ）

条件の数が未知数より少ない場合にも、もちろん(i)のような議論は通用しない。一見、(ii)と同じ方法が使えそうに思えるが、(ii)の補足に書かれているように正方行列の問題に直しても、その正方行列は正則になり得ないから同じやり方は通用しない。しかし、この場合にも(ii)のように誤差ノルム $\|A\mathbf{x}-\mathbf{b}\|$ を最小にするという意味で、最適解 $\bar{\mathbf{x}}$ を求めることができる。

$A\mathbf{x}=\mathbf{b}$



において、
$A\mathbf{x}$ は A の列空間内にある。従って、 $\mathbf{b}$ も A の列空間内になければならないから、 $\mathbf{b}$ を A の列空間に射影し、それを $\mathbf{p}=P\mathbf{b}$ とする（もし A が横長の長方形行列ならば当然列ベクトルの数は次元よりも多いのでその次元の空間を張っている可能性が高いから $\mathbf{b}$ を A の列空間に射影しても、しないのと同じことだが、ここでは(i)や(ii)の場合も含む一般の場合を考えたいのである）。

$A\bar{\mathbf{x}}=\mathbf{p}$

そして次に、 $\bar{\mathbf{x}}$ は A の行空間内にある。なぜならば、 A の行空間と A の零空間 $\mathcal{N}(A)=\mathrm{Ker}(A)$ は直交し、それだけでなく $\mathbf{R}^n$ の直交補空間である[1](p.94-95)。このことは、もし $\bar{\mathbf{x}}$ が A の行空間内に無ければ $\bar{\mathbf{x}}$ は A の零空間内にあり、 $A\bar{\mathbf{x}}=0$ となってしまうことを意味する。もちろん、 $\mathbf{p}=0$ ならばそれでよく、「 $\bar{\mathbf{x}}$ は A の零空間である」が解になるが、今はそのようなことを特に議論しているのではなく、 $\mathbf{p}\neq 0$ の話をしているのである。よって、

$A\bar{\mathbf{x}}=\mathbf{p}$

を満たし、 A の行空間内にあって、 $\|A\bar{\mathbf{x}}-\mathbf{p}\|$ が最小になるような $\bar{\mathbf{x}}$ が最適解である。

・・・とは言っても具体的にどのように $A^\dagger$ を構成すればいいのかわからないと思う。 $A^\dagger$ の計算法の説明をする。

[構成法 1]

□特異値分解

まず、Aの特異値分解について説明する。

$A^T A$ ← $n \times n$

を考える。$A^T A$は正方行列であり、かつ対称行列となる。これを対角化してみよう（固有値問題）。

$$A^T A \mathbf{x}_i = \lambda_i \mathbf{x}_i$$

ここで、$\mathbf{x}_i{}^T \mathbf{x}_i = 1$(単位ベクトル), $\mathbf{x}_i{}^T \mathbf{x}_j = 0 \ (i \neq j)$である。また、$A^T A$は対称行列だから、$\lambda_i \geq 0$

である[1](p.242)。

$\lambda_1, \cdots, \lambda_r$が正であり、$\lambda_{r+1}, \cdots, \lambda_n$が 0 であるとする。

$$A^T A[\mathbf{x}_1 \quad \cdots \quad \mathbf{x}_n] = [\lambda_1 \mathbf{x}_1 \quad \cdots \quad \lambda_n \mathbf{x}_n]$$

$n \times n$

$$= [\mathbf{x}_1 \quad \cdots \quad \mathbf{x}_n] \begin{bmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_n \end{bmatrix}$$

$$= ([\mathbf{x}_1 \quad \cdots \quad \mathbf{x}_n] \begin{bmatrix} \sqrt{\lambda_1} & & \\ & \ddots & \\ & & \sqrt{\lambda_n} \end{bmatrix}) \begin{bmatrix} \sqrt{\lambda_1} & & \\ & \ddots & \\ & & \sqrt{\lambda_n} \end{bmatrix}$$

$$A^T A[\mathbf{x}_1 \quad \cdots \quad \mathbf{x}_n] = ([\mathbf{x}_1 \quad \cdots \quad \mathbf{x}_n]\Lambda)\Lambda$$

$$\Lambda^\dagger [\mathbf{x}_1 \quad \cdots \quad \mathbf{x}_n]^T A^T A[\mathbf{x}_1 \quad \cdots \quad \mathbf{x}_n] = \Lambda$$

ここで、

$$\Lambda^\dagger = \begin{bmatrix} 1/\sqrt{\lambda_1} & & & & & \\ & \ddots & & & & \\ & & 1/\sqrt{\lambda_r} & & & \\ & & & 0 & & \\ & & & & \ddots & \\ & & & & & 0 \end{bmatrix}$$

$$(A[\mathbf{x}_1 \quad \cdots \quad \mathbf{x}_n]\Lambda^\dagger)^T A[\mathbf{x}_1 \quad \cdots \quad \mathbf{x}_n] = \Lambda$$

$$(A[\mathbf{x}_1 \quad \cdots \quad \mathbf{x}_n]\Lambda^\dagger)^T (A[\mathbf{x}_1 \quad \cdots \quad \mathbf{x}_n]\Lambda^\dagger) = I'$$

ここで、

$$I' = \begin{bmatrix} 1 & & & & \\ & \ddots & & & \\ & & 1 & & \\ & & & 0 & \\ & & & & \ddots \\ & & & & & 0 \end{bmatrix}$$

ここで、

$\mathbf{y}_i = \dfrac{A\mathbf{x}_i}{\mu_i}$, $\mu_i = \sqrt{\lambda_i}$ （μ_i を A の**特異値(singular value)**と言う）

とすると、

$$\begin{bmatrix} \mathbf{y}_1 & \cdots & \mathbf{y}_r \end{bmatrix}^T \begin{bmatrix} \mathbf{y}_1 & \cdots & \mathbf{y}_r \end{bmatrix} = \begin{bmatrix} 1 & & \\ & \ddots & \\ & & 1 \end{bmatrix}$$ （$m \times r$）（$\mathbf{y}_i$ は正規直交である）

ここで、

$\mathbf{y}_1, \cdots, \mathbf{y}_r$ は正規直交だが、$\mathbf{y}_i$ は m 次元ベクトルなので、$\mathbf{y}_1, \cdots, \mathbf{y}_r, \cdots, \mathbf{y}_m$ と $m-r$ 個のベクトルを追加して Gram-Schmidt の直交化法により m 個の正規直交基底を作ることが出来る（これだけ用意しておかないと後で Q_2 の逆が作りにくいから）。

$Q_1 = \begin{bmatrix} \mathbf{y}_1 & \cdots & \mathbf{y}_m \end{bmatrix}$, $Q_2 = \begin{bmatrix} \mathbf{x}_1 & \cdots & \mathbf{x}_n \end{bmatrix}$

とすると（Q_1, Q_2 はユニタリー行列である。ユニタリー行列 U は $U^T U = I$, $U^{-1} = U^T$ という定義・性質を持つ）、

$$Q_1{}^T A Q_2 = Q_1{}^T \begin{bmatrix} A\mathbf{x}_1 & \cdots & A\mathbf{x}_n \end{bmatrix} = \begin{bmatrix} \mu_1 & & & \\ & \ddots & & \\ & & \mu_r & \\ & & & \end{bmatrix} = \Sigma$$

$$A = Q_1 \Sigma Q_2{}^T$$ （$m \times m$, $n \times n$, $m \times n$）

この分解を A の**特異値分解(singular value decomposition: SVD)**と言う。正方行列に対する**対角化(diagonalization)**の一般の長方形行列バージョンである。

□一般逆の作成

$$A = Q_1 \Sigma Q_2{}^T$$

の左逆 $A^\dagger$ を求める（一般に右逆にはなっていない $AA^\dagger \neq I$。なっていることもある）。いつもの

ように左から個々の行列の逆行列をかけていくと、

$$Q_2\Sigma^{\dagger}Q_1{}^{T}A = I$$

ここで、Σ の左逆 $\Sigma^{\dagger} = \begin{bmatrix} 1/\mu_1 & & \\ & \ddots & \\ & & 1/\mu_r \end{bmatrix}$ は目の子で求まるところがミソである。

よって、

$$A^{\dagger} = Q_2\Sigma^{\dagger}Q_1{}^{T}$$

□本当かどうか確認
さて、上で

$$A^{\dagger} = Q_2\Sigma^{\dagger}Q_1{}^{T}$$

と一般逆が求まったが、これが本当に最小２乗の最適解になっているかどうか確認しよう。
ユニタリー行列の積はベクトルの長さを変えないから

$$\left\|A\mathbf{x}-\mathbf{b}\right\| = \left\|Q_1\Sigma Q_2{}^{T}\mathbf{x}-\mathbf{b}\right\| = \left\|\Sigma Q_2{}^{T}\mathbf{x}-Q_1{}^{T}\mathbf{b}\right\|$$

ここで、

$$\mathbf{y} = Q_2{}^{T}\mathbf{x} = Q_2{}^{-1}\mathbf{x}$$

とする。$\mathbf{y}$ のノルムは $\mathbf{x}$ のノルムと同じである。$\mathbf{y}$ は $\left\|\Sigma\mathbf{y}-Q_1{}^{T}\mathbf{b}\right\|$ を最小化すればよく、最適解は

$\bar{\mathbf{y}} = \Sigma^{\dagger}Q_1{}^{T}\mathbf{b}$ である。よって、$\bar{\mathbf{x}} = Q_2\bar{\mathbf{y}} = Q_2\Sigma^{\dagger}Q_1{}^{T}\mathbf{b}$ つまり、$A^{\dagger} = Q_2\Sigma^{\dagger}Q_1{}^{T}$ となる。

[構成法 2]
$A = \bar{L}\bar{U}$ を用いて
$A^{\dagger} = \bar{U}^{T}(\bar{U}\bar{U}^{T})^{-1}(\bar{L}^{T}\bar{L})^{-1}\bar{L}^{T}$
も A の擬似逆行列になっている。

[証明]
[1](p.159)

References

[1] G. ストラング著、線形代数とその応用、産業図書、平成 4 年

[2] W.H. Press et al., Numerical Recipes in C [日本語版], 技術評論社, 2003.

[3] G.H. Golub, C.F.V. Loan, Matrix Computations, Third Ed., The John Hopkins University Press., London, 1996.

[4] R.P. Feynman、ファインマン物理学 II 光 熱 波動、岩波書店、2002 年

[5] 寺沢寛一、数学概論[増訂版]、岩波書店、1995 年

[6] 寺沢寛一、数学概論 応用編、岩波書店、1995 年