

LDU 分解時における固有値についての行列式の微分

柏木 光博*

Derivative of Determinant with respect to Eigenvalue in LDU Decomposition

by
Mitsuhiro KASHIWAGI

(Received: September 30, 2012, Accepted: February 20, 2013)

Abstract

This paper presents a derivative of determinant with respect to eigenvalue in LDU decomposition and the numerical algorithm. The proposed method is effective in especially the band matrix that can save a storage capacity sharply compared with the densed matrix. This method computes the singular point in nonlinear analysis and eigenvalue problem using proposed method. Outputs of numerical examples show that the proposed idea works well and a validity through numerical simulation.

Key Words : Derivative of determinant, LDU decomposition, Band matrix, Non-symmetric matrix

1. はじめに

著者は、共役勾配法の理論の中の係数行列の逆行列算定式とトレース理論を使って固有値問題の行列式の固有値による微分式を提案した¹⁾。特に、固有値解析法に応用し効果的であることを示してきた¹⁻⁶⁾。しかし、ここでの提案式は共役勾配法などの反復法用であり LDU 分解などの直接法には使えない。ここでは、共役勾配法に関する式¹⁾を参考にして、直接法の代表的解法である LDU 分解時において非対称行列の固有値についての行列式の微分を示す。また、ニュートン・ラフソン法における解は行列式の微分によって変化するので、それを増分量として利用できる。増分量の設定が自動化できるので、非線形解析への影響が大きい基本的パラメーターを減らすことができる。提案法は密行列の LDU 分解や密行列に比べ記憶容量を大幅に節約できる帯行列の LDU 分解に対してもバンド幅内の要素を使って計算できるので、密行列の計算を、あたかも帯行列の計算をするかのようにできる。提案式を含まない帯行列の計算時間と比べてもそれほど増加にならない。以下に、その理論と詳細なアルゴリズムおよび移流拡散方程式を離散化した

非対称行列による数値実験例を示す。

2. LDU 分解時における固有値による行列式の微分式

固有値問題は次のように表される。 \tilde{A} を $n \times n$ の非対称行列とすると、標準固有値問題は

$$\tilde{A}x = \lambda x \quad (2.1)$$

として示される。ここに、 λ は固有値を、 x は固有ベクトルを表している。式(2.1)が自明な解をもつためには、 $\tilde{A} - \lambda I$ が特異でなければならないので

$$\det[\tilde{A} - \lambda I] = 0 \quad (2.2)$$

ここで、 $f(\lambda)$ を以下のようにおく。

$$f(\lambda) = \det[\tilde{A} - \lambda I] \quad (2.3)$$

トレース理論によって

$$\begin{aligned} \frac{f'(\lambda)}{f(\lambda)} &= \text{trace} \left[[\tilde{A} - \lambda I]^{-1} [\tilde{A} - \lambda I] \right] \\ &= -\text{trace} \left[[\tilde{A} - \lambda I]^{-1} \right] \end{aligned} \quad (2.4)$$

* 産業工学部教授

$$A = \tilde{A} - \lambda I$$

$$= [a_{ij}] = \begin{bmatrix} a_{11} & & & \\ a_{21} & a_{22} & \text{sym.} & \\ \vdots & \ddots & & \\ a_{n1} & \cdots & a_{nm} & \end{bmatrix} \quad (2.5)$$

とおく. LDU 分解時には $A = LDU$ になる. ここに,

$$L = [\ell_{ij}] = \begin{bmatrix} 1 & & & 0 \\ \ell_{21} & 1 & & \\ \vdots & & \ddots & \\ \ell_{n1} & \cdots & \ell_{n(n-1)} & 1 \end{bmatrix} \quad (2.7)$$

$$D = [d_{ij}] = \begin{bmatrix} d_{11} & & & 0 \\ & d_{22} & & \\ & & \ddots & \\ 0 & & & d_{nn} \end{bmatrix} \quad (2.8)$$

$$U = [u_{ij}] = \begin{bmatrix} 1 & u_{12} & \cdots & u_{1n} \\ & 1 & & \vdots \\ & & \ddots & u_{(n-1)n} \\ 0 & & & 1 \end{bmatrix} \quad (2.9)$$

L^{-1} を以下のようにおく.

$$L^{-1} = [g_{ij}] = \begin{bmatrix} 1 & & & 0 \\ g_{21} & 1 & & \\ \vdots & & \ddots & \\ g_{n1} & \cdots & g_{n(n-1)} & 1 \end{bmatrix} \quad (2.10)$$

$LL^{-1} = I$ (I は単位行列) を展開して整理すると,

$$i = j+1 \quad g_{ij} = -\ell_{ij},$$

$$i \geq j+2 \quad g_{ij} = -\ell_{ij} - \sum_{k=j+1}^{i-1} \ell_{ik} g_{kj} \quad (2.11)$$

次に, U^{-1} を以下のようにおく.

$$U^{-1} = [h_{ij}] = \begin{bmatrix} 1 & h_{12} & \cdots & h_{1n} \\ & 1 & & \vdots \\ & & \ddots & h_{(n-1)n} \\ 0 & & & 1 \end{bmatrix} \quad (2.12)$$

$U^{-1}U = I$ を展開して整理すると,

$$j = i+1 \quad h_{ij} = -u_{ij},$$

$$j \geq i+2 \quad h_{ij} = -u_{ij} - \sum_{k=i+1}^{j-1} u_{kj} h_{ik} \quad (2.13)$$

式(2.11)と(2.13)から, g_{ij} と h_{ij} は全要素について計算する必要があるが, 半帯幅外の要素については $\ell_{ij} = 0$ および $u_{ij} = 0$ となるので, 半帯幅内の ℓ_{ij} と u_{ij} を使って計算できる. また, 半帯幅が狭いほど計算時間は短くなる.

行列式の微分は, 式(2.4)より, 式(2.6)の逆行列の対角要素だけを求めて加算すればよい. 式(2.7),(2.8),(2.9),(2.10),(2.12)から, $U^{-1}D^{-1}L^{-1}$ を展開し対角要素の和を求めると, 式(2.4)は以下ようになる.

$$\frac{f'(\lambda)}{f(\lambda)} = -\sum_{i=1}^n \left(\frac{1}{d_{ii}} + \sum_{k=i+1}^n \frac{g_{ki} h_{ik}}{d_{kk}} \right) \quad (2.14)$$

以上から, 行列式の微分には LDU 分解時の D 行列と L 行列および U 行列の逆行列の要素から成っていることが分かる. 上述のように, バンド幅内の要素を使って計算できるので, 密行列の計算を, あたかも帯行列の計算をするかのようにできる. よって, 提案式を含まない帯行列の計算時間と比べてもそれほどの増加にならないと予想される.

LDU 分解プログラムに補足したプログラム (ベクトルを 2 本追加) により, 簡単に f'/f が得られる. この値はニュートン・ラフソン法やデュラン・カーナー法などによる数値解析 (固有値に近づく $f(\lambda)$ は 0 に近づくが, 実際の計算では f/f' を使うので問題ない) に便利に利用できる. また, 提案法は連立一次方程式を解く過程で簡単に求めることができる方法となっている.

3. 提案法のアルゴリズム

3-1. 密行列用アルゴリズム

最初に, 密行列に対するアルゴリズムを示す. 配列や変数は以下による.

1) LDU 分解と行列式の固有値による微分値の算定

①入力データ

- A : given coefficient matrix,
- 2-dimension array as A(n,n)
- b : work vector, 1-dimension array as b(n)
- c : work vector, 1-dimension array as c(n)
- n : given order of matrix A and vector b
- eps : parameter to check singularity of the matrix output

②出力データ

- A : L matrix, D matrix and U matrix,

2-dimension array as $A(n,n)$
 fd : derivative of determinant
 ierr : error code
 =0, for normal execution
 =1, for singularity

③LDU分解

```
do i=1,n
<d(i,i)>
do k=1,i-1
A(i,i)=A(i,i)-A(i,j)*A(j,j)*A(j,i)
end do
if (abs(A(i,i))<eps) then
ierr=1
return
end if
<l(i,j) and u(i,j)>
do j=i+1,n
do k=1,i-1
A(j,i)=A(j,i)-A(j,k)*A(k,k)*A(k,i)
A(i,j)=A(i,j)-A(i,k)*A(k,k)*A(k,j)
end do
A(j,i)=A(j,i)/A(i,i)
A(i,j)=A(i,j)/A(i,i)
end do
end do
ierr=0
```

④行列式の固有値による微分

```
fd=0
<(i,i)>.
do i=1,n
fd=fd-1/A(i,i)
end do
<(i,j)>
do i=1,n-1
do j=i+1,n
b(j)=-A(j,i)
c(j)=-A(i,j)
do k=1,j-i-1
b(j)=b(j)-A(j,i+k)*b(i+k)
c(j)=c(j)-A(i+k,j)*c(i+k)
end do
fd=fd-b(j)*c(j)/A(j,j)
end do
end do
```

2) 解の算定

①入力データ

A : L matrix , D matrix and U matrix,
 2-dimension array as $A(n,n)$
 b : given right hand side vector,

1-dimension array as $b(n)$

n : given order of matrix A and vector b

②出力データ

b : work and solution vector, 1-dimension array

③前進代入

```
do i=1,n
do j=i+1,n
b(j)=b(j)-A(j,i)*b(i)
end do
end do
```

④後退代入

```
do i=1,n
b(i)=b(i)/A(i,i)
end do
do i=1,n
ii=n-i+1
do j=1,ii-1
b(j)=b(j)-A(j,ii)*b(ii)
end do
end do
```

3-2. バンド行列用アルゴリズム

次に、バンド行列に対するアルゴリズムを示す。以下に帯行列の概要を図-1 に示す。ここに、 nl は左バンド幅であり、 nu は右バンド幅である。それぞれに対角要素は含まないので、帯幅は $nl+1+nu$ である。

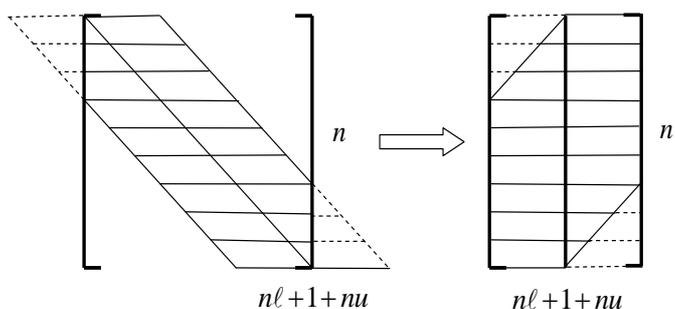


図-1 帯行列と帯幅

1) LDU分解と行列式の固有値による微分値の算定

①入力データ

A : given coefficient band matrix,
 2-dimension array as $A(n,nl+1+nu)$
 b : work vector, 1-dimension array as $b(n)$
 c : work vector, 1-dimension array as $c(n)$
 n : given order of matrix A
 nl : given left half band width of matrix A
 nu : given right half band width of matrix A
 eps : parameter to check singularity of the matrix

②出力データ

A : L matrix , D matrix and U matrix,
2-dimension array as A(n,nl+1+nu)

fd : derivative of determinant

ierr : error code

=0, for normal execution

=1, for singularity

③LDU分解

do i=1,n

<d(i,i)>

do j=max(1,i-min(nl,nu)),i-1

A(i,nl+1)=A(i,nl+1)-A(i,nl+1-(i-j))*

A(j,nl+1)*A(j,nl+1+(i-j))

end do

if (abs(A(i,nl+1))<eps) then

ierr=1

return

end if

<l(i,j)>

do j=i+1,min(i+nl,n)

sl=A(j,nl+1-(j-i))

do k=max(1,j-nl),i-1

sl=sl- A(j,nl+1-(j-k))*A(k,nl+1)

*A(k,nl+1+(i-k))

end do

A(j,nl+1-(j-i))=sl/A(i,nl+1)

end do

<u(i,j)>

do j=i+1,min(i+nu,n)

su=A(i,nl+1+(j-i))

do k=max(1,j-nu),i-1

su=su- A(i,nl+1-(i-k))*A(k,nl+1)

*A(k,nl+1+(j-k))

end do

A(i,nl+1+(j-i))=su/A(i,nl+1)

end do

end do

ierr=0

④行列式の固有値による微分

fd=0

<(i,i)>

do i=1,n

fd=fd-1/A(i,nq)

end do

<(i,j)>

do i=1,n-1

do j=i+1,min(i+nl,n)

b(j)=-a(j,nl+1-(j-i))

do k=1,j-i-1

b(j)=b(j)-a(j,nl+1-(j-i)+k)*b(i+k)

end do

end do

do j=i+nl+1,n

b(j)=0.d0

do k=1,nl

b(j)=b(j)-a(j,k)*b(j-nl-1+k)

end do

end do

do j=i+1,min(i+nu,n)

c(j)=-a(i,nl+1+(j-i))

do k=1,j-i-1

c(j)=c(j)-a(i+k,nl+1+(j-i)-k)*c(i+k)

end do

end do

do j=i+nu+1,n

c(j)=0.d0

do k=1,nu

c(j)=c(j)-a(i,nl+1+k)*c(j-nl-1+k)

end do

end do

do j=i+1,n

fd=fd-b(j)*c(j)/a(j,nl+1)

end do

2) 解の算定

①入力データ

A : given decomposed coefficient band matrix,
2-dimension array as A(n,nl+1+nu)

b : given right hand side vector, 1-dimension
array as b(n)

n : given order of matrix A and vector b

nl : given left half band width of matrix A

nu : given right half band width of matrix A

②出力データ

b : solution vector, 1-dimension array

③前進代入

do i=1,n

do j=max(1,i-nl),i-1

b(i)=b(i)-A(i,n+1-(i-j))*b(j)

end do

end do

④後退代入

do i=1,n

ii=n-i+1

b(ii)=b(ii)/a(ii,nl+1)

do j=ii+1,min(n,ii+nu)

b(ii)=b(ii)-a(ii,nl+1+(j-ii))*b(j)

end do

end do

4. 数値実験

提案式の有効性を示すために、非対称行列を有する標準固有値問題に関する例題による数値実験を行った。ここでの数値実験は、すべて倍精度演算とし、求める固有解の個数は10個とした。アルゴリズムと数値実験結果を以下に示す。計算には、今回はスーパーコンピュータを対象としていないこと、および最近のパーソナルコンピュータの性能向上などを考慮し、'Intel(R) Core™ i7 3.2GHz, RAM12.0 GB, Windows 7, PGI Fortran Workstation (pgiwsx64-106)'を使用した。以下に詳細を記す。

4-1. アルゴリズム

ここでは、逆ベキ乗法とシフト逆ベキ乗法の提案式用アルゴリズムの詳細を示す。これらのアルゴリズムにより数値実験を行った。

A は $n \times n$ 行列、 λ は固有値、 x は対応する右固有ベクトル、 y は対応する左固有ベクトル、 λ_0 を原点移動量とすると、標準固有値問題は、 $[A - \lambda_0 I]x = (\lambda - \lambda_0)x$ および $[A - \lambda_0 I]^T y = (\lambda - \lambda_0)y$ のように表せる。

4-1-1. 逆ベキ乗法

(0) $\lambda_0 = 0$ として $A - \lambda_0 I$ の LDU 分解を行う。本アルゴリズムでは常に $\lambda_0 = 0$ である。

(1) $\ell = 0$

初期ベクトル $x^{(0)}$ と $y^{(0)}$ を一様乱数によって生成する。最大反復回数を設定する。グラム・シュミットの双直交化法により、既に得られた i 個の右固有ベクトル u と左固有ベクトル v の抜き取りと正規化および初期近似固有値を次式で求める。

$$\begin{aligned} \tilde{x}^{(0)} &= x^{(0)} - \sum_{j=1}^i (x^{(0)T} v_j / v_j^T u_j) u_j, \\ x^{(0)} &= \tilde{x}^{(0)} / \left(\tilde{x}^{(0)T} \tilde{x}^{(0)} \right)^{\frac{1}{2}}, \\ \tilde{y}^{(0)} &= y^{(0)} - \sum_{j=1}^i (y^{(0)T} u_j / u_j^T v_j) v_j, \\ y^{(0)} &= \tilde{y}^{(0)} / \left(\tilde{y}^{(0)T} \tilde{y}^{(0)} \right)^{\frac{1}{2}}, \quad \lambda^{(0)} = y^{(0)T} A x^{(0)} \end{aligned}$$

(2) $\ell = 1, 2, \dots$

逆ベキ乗法による反復計算を行う。

① ℓ 反復時での $x^{(\ell)}$ と $y^{(\ell)}$ は、 $x^{(\ell)} = [A - \lambda_0 I]^{-1} x^{(\ell-1)}$ と $y^{(\ell)} = [A - \lambda_0 I]^T y^{(\ell-1)}$ によって求める。ここに、 $-T$ は転置行列の逆行列を表している。グラム・シュミットの双直交化法により、既に得られた固有ベクトルの抜き取りと正規化および近似固有値を次式で求める。

$$\begin{aligned} \tilde{x}^{(\ell)} &= x^{(\ell)} - \sum_{j=1}^i (x^{(\ell)T} v_j / v_j^T u_j) u_j, \\ x^{(\ell)} &= \tilde{x}^{(\ell)} / \left(\tilde{x}^{(\ell)T} \tilde{x}^{(\ell)} \right)^{\frac{1}{2}}, \\ \tilde{y}^{(\ell)} &= y^{(\ell)} - \sum_{j=1}^i (y^{(\ell)T} u_j / u_j^T v_j) v_j, \\ y^{(\ell)} &= \tilde{y}^{(\ell)} / \left(\tilde{y}^{(\ell)T} \tilde{y}^{(\ell)} \right)^{\frac{1}{2}}, \quad \lambda^{(\ell)} = y^{(\ell)T} A x^{(\ell)} \end{aligned}$$

②収束判定を行う。最大反復回数（本数値実験では

$\ell_{\max} = 100000$) を超えたら計算をストップ。 ε_1 と ε_2 を収束判定値とすると

$$\left| \frac{\lambda^{(\ell)} - \lambda^{(\ell-1)}}{\lambda^{(\ell)}} \right| < \varepsilon_1 \quad (\text{本数値実験では } \varepsilon_1 = 10^{-14})$$

あるいは、残差ベクトル： $r^{(\ell)} = A x^{(\ell)} - \lambda^{(\ell)} x^{(\ell)}$ を計算し

$$\frac{\|r^{(\ell)}\|}{\|A x^{(\ell)}\|} < \varepsilon_2 \quad (\text{本数値実験では } \varepsilon_2 = 10^{-8})$$

であれば収束。次の固有値を求める場合、(1)に戻るが、求めない場合、(3)へ。収束していない場合、(2)の①に戻る。

(3)最後に、求められた全ての固有値と固有ベクトルに対して、固有値と固定原点移動量 λ_0 との差の絶対値が小さい方から順に並び替えを行う。

4-1-2. シフト逆ベキ乗法

(0) $\lambda_0 = 0$ として $A - \lambda_0 I$ の LDU 分解を行う。

(1) $\ell = 0$

初期ベクトル $x^{(0)}$ と $y^{(0)}$ を一様乱数によって生成する。最大反復回数を設定する。グラム・シュミットの双直交化法により、既に得られた i 個の右固有ベクトル u と左固有ベクトル v の抜き取りと正規化および初期近似固有値を次式で求める。

$$\begin{aligned} \tilde{x}^{(0)} &= x^{(0)} - \sum_{j=1}^i (x^{(0)T} v_j / v_j^T u_j) u_j, \\ x^{(0)} &= \tilde{x}^{(0)} / \left(\tilde{x}^{(0)T} \tilde{x}^{(0)} \right)^{\frac{1}{2}}, \\ \tilde{y}^{(0)} &= y^{(0)} - \sum_{j=1}^i (y^{(0)T} u_j / u_j^T v_j) v_j, \\ y^{(0)} &= \tilde{y}^{(0)} / \left(\tilde{y}^{(0)T} \tilde{y}^{(0)} \right)^{\frac{1}{2}}, \quad \lambda^{(0)} = y^{(0)T} A x^{(0)} \end{aligned}$$

(2) $\ell = 1, 2, \dots$

シフト逆ベキ乗法による反復計算を行う。

① ℓ 反復時での $x^{(\ell)}$ と $y^{(\ell)}$ は、 $x^{(\ell)} = [A - \lambda_0 I]^{-1} x^{(\ell-1)}$ と $y^{(\ell)} = [A - \lambda_0 I]^T y^{(\ell-1)}$ によって求める。ここに、 $-T$ は転置行列の逆行列を表している。グラム・シュミットの双直交化法により、既に得られた固有ベクトルの抜き取りと正規化および近似固有値を次式で求める。

$$\begin{aligned} \tilde{x}^{(l)} &= x^{(l)} - \sum_{j=1}^i (x^{(l)T} v_j / v_j^T u_j) u_j, \\ x^{(l)} &= \tilde{x}^{(l)} / \left(\tilde{x}^{(l)T} \tilde{x}^{(l)} \right)^{\frac{1}{2}}, \\ \tilde{y}^{(l)} &= y^{(l)} - \sum_{j=1}^i (y^{(l)T} u_j / u_j^T v_j) v_j, \\ y^{(l)} &= \tilde{y}^{(l)} / \left(\tilde{y}^{(l)T} \tilde{y}^{(l)} \right)^{\frac{1}{2}}, \quad \lambda^{(l)} = y^{(l)T} A x^{(l)} \end{aligned}$$

② ある反復回数毎 (ここでは、100 反復回数) に原点移動量を $\lambda_0 = \lambda^{(l)}$ と変更し、 $A - \lambda_0 I$ の LDU 分解を行う。①の計算を実行する。次に、今回求めた $FD (= f'/f)$ と前回求めた FD が同符号なら原点移動量を $\lambda_0 = \lambda^{(l)} - 1/FD$ と変更し、 $A - \lambda_0 I$ の LDU 分解を行う。①の計算を実行する。

③ 収束判定を行う。最大反復回数 (本数値実験では $\ell_{\max} = 100000$) を超えたら計算をストップ。 ε_1 と ε_2 を収束判定値とする

$$\left| \frac{\lambda^{(l)} - \lambda^{(l-1)}}{\lambda^{(l)}} \right| < \varepsilon_1 \quad (\text{本数値実験では } \varepsilon_1 = 10^{-14})$$

あるいは、残差ベクトル: $r^{(l)} = Ax^{(l)} - \lambda^{(l)} x^{(l)}$ を計算し

$$\frac{\|r^{(l)}\|}{\|Ax^{(l)}\|} < \varepsilon_2 \quad (\text{本数値実験では } \varepsilon_2 = 10^{-8})$$

であれば収束。次の固有値を求める場合、(1)に戻るが、求めない場合、(3)へ。収束していない場合、(2)の①に戻る。

(3)最後に、求められた全ての固有値と固有ベクトルに対して、固有値と固定原点移動量 λ_0 との差の絶対値が小さい方から順に並び替えを行う。

4-2. 標準固有値問題 (移流拡散方程式)

ここでは、定常問題における移流拡散方程式を離散化して生じる係数マトリックスを扱った。多くの極近接根をもつ求めにくい対象である。自由度と原点移動量を変化させながら数値実験した。以下のような2次元モデルを考える (図-1)。境界条件は境界値を0とする。移流拡散方程式に中心差分を用いて離散化を行うと、 $Ax = \lambda x$ の固有値問題が得られる。ここで、 $A = (a_{k,l})$ とおけば、係数行列 A の各要素は次式となる。

$$l = (j-1)n_x + i$$

$$a_{l+ny,l} = -\frac{v}{2\Delta y} - \frac{\chi}{\Delta y^2}, \quad (1 \leq i \leq n_x, 1 \leq j < n_y)$$

$$a_{l+1,l} = -\frac{u}{2\Delta x} - \frac{\chi}{\Delta x^2}, \quad (1 \leq i < n_x, 1 \leq j \leq n_y)$$

$$a_{l,l} = \frac{2\chi}{\Delta x^2} + \frac{2\chi}{\Delta y^2}, \quad (1 \leq i \leq n_x, 1 \leq j \leq n_y)$$

$$a_{l,l+1} = \frac{u}{2\Delta x} - \frac{\chi}{\Delta x^2}, \quad (1 \leq i < n_x, 1 \leq j \leq n_y)$$

$$a_{l,l+ny} = \frac{v}{2\Delta y} - \frac{\chi}{\Delta y^2}, \quad (1 \leq i \leq n_x, 1 \leq j < n_y)$$

$$a_{l,m} = 0 \quad \text{上記以外の } l, m$$

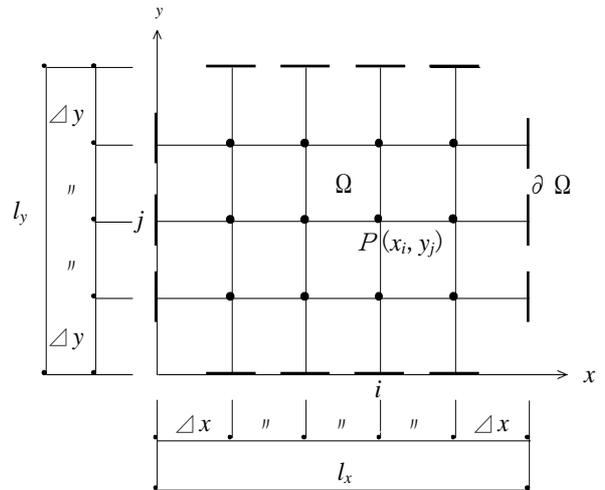


図-2 2次元モデルの一例

表-1 2次元モデルのパラメーター

type	lx	ly	nx	ny	n
A	50	50	50	50	2500
B	100	100	100	100	10000
C	150	150	150	150	22500
D	200	200	200	200	40000

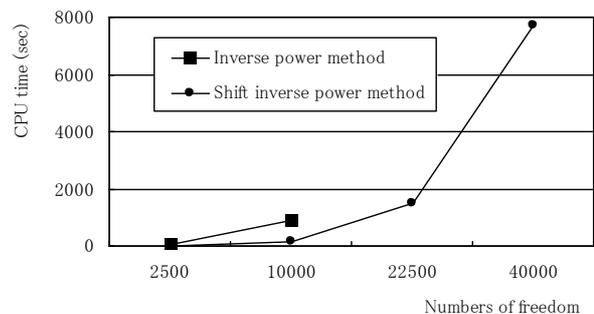


図-3 2次元モデルの計算時間

ここに、 $\Delta x, \Delta y$ はそれぞれ x, y 方向の分割幅である。

ここでは、 $u = 0, \Delta x = \Delta y = v = \chi = 1$ とした。セルベクレ数は 1 で行列 A は弱対角優位行列となる。数値実験に用いたパラメータ $l_x, l_y, n_x, n_y, n = n_x \times n_y$ を表-1 に示す。

図-3 は、表-1 のモデル A の 4 タイプについて、逆ベキ乗法とシフト逆ベキ乗法の計算時間を描いている。逆ベキ乗法では、タイプ A と B については解析できたが、タイプ C と D については解析できず、次元数が増加すると解析不能となった。しかし、提案のアルゴリズムによるシフト逆ベキ乗法では 4 タイプ共解析が可能であった。提案法は計算時間や反復回数も少なく安定した解法となっている。

5. 結び

直接法の代表的解法である LDU 分解時に同時に非対称行列の固有値による行列式の微分を求められることを示した。また、ニュートン・ラフソン法における解は行列式の微分によって変化するので、それを増分量として利用することが可能である。増分量の設定が自動化できるので、非線形解析への影響が大きい基本的パラメータを減らすことができる。提案法は LDU 分解を対象としており、帯行列は密行列の LDU 分解に比べ記憶容量を大幅に節約できる。 LDU 分解プログラムに補足したプログラム（ベクトルを 2 本追加）により、簡単に固有値による行列式の微分が得られる。この値はニュートン・ラフソン法やデュラン・カーナー法などによる数値解析に便利に利用できる。また、提案法は連立一次方程式を解く過程で簡単に求めることができる方法であり、非線形問題や固有値問題などに対して特に有効な方法と考えられる。

参考文献

- [1] 柏木 光博, 共役勾配法による最大あるいは最小固有解の一算定法, 日本計算工学会論文集, Vol.1,1-5,1999.5
- [2] 柏木光博, 共役勾配法による大次元スパース対称行列の固有解, 日本応用数学会論文集, Vol.15, 29-43, 2005.3
- [3] Kashiwagi,M., A Method for Determining Eigensolutions of Large, Sparse, Symmetric Matrices by the Preconditioned Conjugate Gradient Method in the Generalized Eigenvalue Problem, 日本建築学会構造系論文集, Vol.629, 1209-1217, 2008.7
- [4] 柏木 光博, ダブルシフト逆ベキ乗法によるスパース対称行列の中間固有解の一算定法, 日本応用数学会論文集, Vol.19, No.3, 23-38, 2009.9
- [5] 柏木 光博, 一般固有値問題における前処理付共役勾配法による大次元スパース対称行列の固有解の一算定法, 日本建築学会構造工学論文集, Vol.56B, 445-451, 2010.3
- [6] 柏木 光博, 一般固有値問題におけるランチョス逆ベキ乗法によるスパース対称行列の中間固有対の一算定法, 日本建築学会構造系論文集, Vol.664, 1181-1188, 2011.6