応用力学論文集 Vol.7,pp.247-254 （2004年8月） 土木学会

区分的に値を持つ基底ベクトルを用いた対称行列用連立一次方程式の反復解法

An Iterative Method Using Piecewise Valued Basic Vectors for Solving Linear Systems with Symmetric Coefficient Matrix

片山拓朗*, 柏木光博**, 大脇信一***, 山尾敏孝****
Takuro KATAYAMA, Mitsuhiro KASHIWAGI, Shin-ichi OHWAKI and Toshitaka YAMAO

*正会員 博士(学術) 崇城大学助教授 工学部環境建設工学科(〒860-0082 熊本市池田4丁目22番1号)
**正会員 工学博士 九州東海大学教授 工学部建築学科(〒862-8652 熊本市渡鹿9丁目1番1号)
***理学博士 熊本大学教授 理学部数理科学科(〒860-8555 熊本市黒髪2丁目39番1号)
****正会員 工学博士 熊本大学教授 工学部環境システム工学科(〒860-8555 熊本市黒髪2丁目39番1号)

This paper presents an iterative method using piecewise valued basic vectors for solving linear systems with a symmetric coefficient matrix. The solution of linear systems is easily obtained by the proposed iterative method which repeats the following four operation. 1) Make the some independent systems in which total number of freedoms is reduced by Galerkin's method using the piecewise valued basic vectors. 2) Compute the approximate solution of these systems by other iterative method. 3) Obtain the best approximate solution of original linear systems by linking each approximate solution. 4) Update the proposed basic vectors by using the best approximation. Validity and efficiency of the proposed iterative method are studied and shown in several numerical examples.

Key Words: *the piecewise valued basic vector, Galerkin's method, linear systems, symmetric matrix*

1. はじめに

大規模構造物の動的問題や弾塑性問題の数値解を求める場合，大型対称行列の連立一次方程式を高速かつ低コストで解く必要に迫られる．PC クラスターなどの分散メモリー型並列計算機はこれを解決する有力な手段である．ただし，分散メモリー型並列計算機はノード間通信が必要であり，ノード間通信の性能はノード内の CPU と DRAM の通信性能に比べて劣る場合が多いので，計算機の性能を最大限発揮するためには通信量と通信回数が少ない連立一次方程式の解法が必要となる．

大規模な連立一次方程式の解法としては，AMG法[1]とCG法[2]を組み合わせた方法が有力である．AMG 法の反復計算の主要部は緩和法を含むので，分散メモリー型並列計算機への実装においては注意が必要である．藤井等は領域分割によるAMG法の並列化について研究を行い，AMG 法と CG 法を組み合わせた方法が ICCG 法[3]に比べて3倍以上高速であることを報告している[4]．

本論文では区分的に値を持つ基底ベクトルを用いた大型対称行列の連立一次方程式の並列解法を提案する[5),6]．提案法は以下の 4 つの操作を繰り返すことにより，連立一次方程式の解を求める方法である．

1) 区分的に値を持つ基底ベクトルを用いる Galerkin 近似により，未知数の数を減らした複数の連立一次方程式を作る．
2) それらの減次された方程式の近似解を他の反復法で計算する．
3) 得られた近似解をつなぎ合わせて元の方程式の最良近似解を構成する．
4) 得られた最良近似解を用いて提案の基底ベクトルを更新する．

提案法は減次された方程式をノード単独で解くことができるので，分散メモリー型並列計算機向きの解法と考えられる．提案法は連立一次方程式の未知数をいくつかの集合に分け，この集合を基に基底ベクトルを設定するため，領域の概念が無い問題にも適用ができる．なお，提案法は AMG 法と Domain Decomposition Methods[7] および Subspace Correction Methods[8] と異なる方法である．

提案法のアルゴリズムの妥当性を検証するために，ポアソン方程式の差分近似問題を例題として，16 ノードの PC クラスターを用いて数値実験を行った．数値実験により提案法の基本的な収束の性質を確認した．

2. 理論

2. 1 区分的に値を持つ基底ベクトルの定義

(1) 連立一次方程式および自由度の集合の定義

正値対称行列 $A \in \mathbb{R}^{n\times n}$ と既知の列ベクトル $\boldsymbol{b} \in \mathbb{R}^n$ および未知の列ベクトル $\boldsymbol{x} \in \mathbb{R}^n$ からなる次の連立一次方程式を考える.

$$A\,\boldsymbol{x} = \boldsymbol{b} \tag{1}$$

$$\boldsymbol{x} \equiv \left[x_1, x_2, \ldots, x_n\right]^{\mathrm{T}} \tag{2}$$

$$\boldsymbol{b} \equiv \left[b_1, b_2, \ldots, b_n\right]^{\mathrm{T}} \tag{3}$$

$$A \equiv \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \cdots & a_{nn} \end{bmatrix} \tag{4}$$

ここに, n は未知数の数を表す. また, $\mathbb{R}$ と $\mathbb{R}^n$ および $\mathbb{R}^{n\times m}$ はそれぞれ実数と n 次実列ベクトルおよび $n\times m$ 次実行列の集合を表す. 上付き添え字Tは行列の転置を表す.

未知数 x_i の番号 i を単に自由度と呼ぶものとする. 1から n までの自由度の集合を Ω として次式で定義する.

$$\Omega \equiv \left\{\, i \,|\, 1 \le i \le n \,\right\} \tag{5}$$

集合 Ω は m 個の互いに独立な部分集合 $\Omega_k, 1 \le k \le m$ の和集合として次式で表せるものとする.

$$\Omega = \bigcup_{k=1}^{m} \Omega_k, \quad \Omega_k \cap \Omega_j = \varnothing, \quad k \ne j \tag{6}$$

ここに, Ω_k の自由度数を ω_k とする. Ω の自由度数は $n = \omega_1 + \omega_2 + \cdots + \omega_m$ となる.

また, Ω_k に関する自由度について行列 A の要素を集めた部分行列を $\overline{A}^{(k)} \in \mathbb{R}^{\omega_k\times\omega_k}$ とすると,

$$\overline{A}^{(k)} \equiv \left[\, a_{ij} \,|\, i \in \Omega_k, \, j \in \Omega_k \,\right] \tag{7}$$

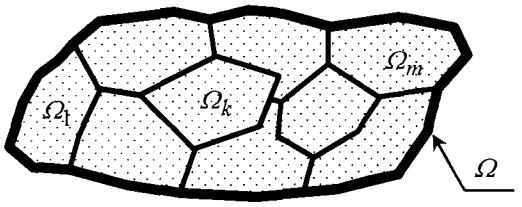
と表せる. $\overline{A}^{(k)}$ はいかなる列と行の置換を行っても対角ブロック行列に変換できないものとする.

図—1に Ω と Ω_k のイメージを示す. 太い実線で囲まれた複数の閉領域は集合 $\Omega_1, \Omega_2, \cdots, \Omega_m$ を, それらを囲む極太の実線の内側が集合 Ω をイメージしたものである。図中の細かな点は(1)式の自由度をイメージしたものである.

(2) 隣接する集合の定義

ここでは集合 Ω_k と Ω_l の隣接について定義を行う. Ω_k と Ω_l が次の条件を満たす時, Ω_l は Ω_k に隣接すると定義する.

$$\mathrm{Norm}(A, \Omega_k, \Omega_l) \ne 0, \quad l \ne k \tag{8}$$



図—1 集合 Ω, 部分集合 $\Omega_k, 1 \le k \le m$ および自由度のイメージ

ここに, $\mathrm{Norm}(A, \Omega_k, \Omega_l)$ は適当な自由度番号の集合である Γ と Φ, および行列 A の非対角要素を用いて定義される次のノルムとする.

$$\mathrm{Norm}(A, \Gamma, \Phi) \equiv \sum_{i\in\Gamma} \left(\sum_{j\in\Phi,\, j\ne i} \left| a_{ij} \right| \right) \tag{9}$$

Ω_k に隣接する集合 Ω_l の個数を π_k とし, Ω_l の番号 l からなる集合を Π_k とし, 次式で定義する.

$$\Pi_k \equiv \left\{\, l \,\middle|\, \mathrm{Norm}(A, \Omega_k, \Omega_l) \ne 0, \; l \ne k \right\} \tag{10}$$

(3) 区分的に値を持つ基底ベクトル

ここで, α_k 個の直交ベクトル $\hat{\boldsymbol{v}}_i^{(k)} \in \mathbb{R}^n$, $1 \le i \le \alpha_k$ を次式で定義する.

$$\hat{\boldsymbol{v}}_i^{(k)} \equiv \left[\, v_{i,j} \,\middle|\, j \in \Omega; \quad v_{i,l} = 0, \; l \notin \Omega_k \,\right], \quad 1 \le i \le \alpha_k, \quad 1 \le k \le m \tag{11}$$

本論文ではベクトル $\hat{\boldsymbol{v}}_i^{(k)}$ を基底ベクトルとする解ベクトル $\boldsymbol{x}$ の部分空間近似を考える. 基底ベクトル $\hat{\boldsymbol{v}}_i^{(k)}$ は集合 Ω_k に属する自由度のみ意味のある値を持ち, その他の自由度は意味のある値を持たないので, $\hat{\boldsymbol{v}}_i^{(k)}$ を区分的に値を持つ基底ベクトルと呼ぶ[9), 10), 11)].

次に, $\hat{\boldsymbol{v}}_i^{(k)}, 1 \le i \le \alpha_k$ を基底ベクトルとする部分空間を $\hat{V}^{(k)} \in \mathbb{R}^{n\times\alpha_k}$ として, 次式で定義する.

$$\hat{V}^{(k)} \equiv \left[\hat{\boldsymbol{v}}_1^{(k)}, \hat{\boldsymbol{v}}_2^{(k)}, \cdots, \hat{\boldsymbol{v}}_{\alpha_k}^{(k)} \right], \quad 1 \le k \le m \tag{12}$$

基底ベクトル $\hat{\boldsymbol{v}}_i^{(k)}$ の直交条件は次式となる. ここに, I_n は n 次の単位行列とする.

$$\hat{V}^{(k)\mathrm{T}}\, \hat{V}^{(k)} = \mathrm{I}_{\alpha_k} \tag{13}$$

さらに, $\hat{V}^{(1)}, \hat{V}^{(2)}, \ldots, \hat{V}^{(m)}$ はその基底ベクトルの構造より互いに独立かつ直交しているので, これらが張る部分空間を $V \in \mathbb{R}^{n\times\alpha}$ として, 次式で定義する.

$$V \equiv \left[\hat{V}^{(1)}, \hat{V}^{(2)}, \ldots, \hat{V}^{(m)} \right] \tag{14}$$

ここに, $\alpha = \alpha_1 + \alpha_2 + \cdots + \alpha_m$ とする.

2. 2 Galerkin 近似の基礎式

(1) Ω_k を内包する集合 Φ_k の定義

部分集合 Ω_k を内包するように自由度の集合 Φ_k を次のように定める.

$$\Omega_k \subset \Phi_k \subset \Omega, 1 \le k \le m \tag{15}$$

$$\Omega_i \not\subset \Phi_k, i \ne k, 1 \le k \le m \tag{16}$$

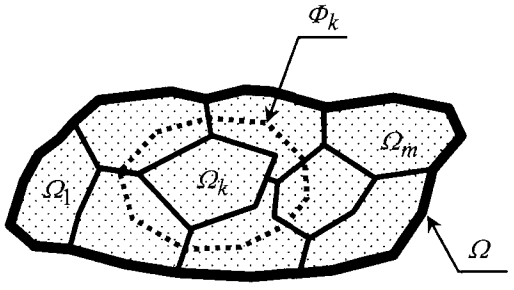
ここに，Φ_k の自由度数を ϕ_k とする．(16)式は Φ_k が Ω_k 以外の集合 Ω_i を内包しない条件を示す．

また，Φ_k に関する自由度について行列 A の要素を集めた部分行列を $\bar{A}_{\Phi}^{(k)} \in \mathbb{R}^{\phi_k \times \phi_k}$ とすると，

$$\bar{A}_{\Phi}^{(k)} \equiv \left[a_{ij} \,\middle|\, i \in \Phi_k, j \in \Phi_k \right] \tag{17}$$

と表される．$\bar{A}_{\Phi}^{(k)}$ はいかなる行と列の置換を行っても対角ブロック行列に変換できないものとする．

図—2に Ω_k と Φ_k のイメージを示す．Ω_k を囲む太い点線の内部が Φ_k を表している．なお，Φ_k と重なる部分集合 $\Omega_l, l \ne k$ の集合番号 l からなる集合を Γ_k とし，Γ_k の要素数を γ_k とする．よって，$k \notin \Gamma_k$ となる．



図—2　Ω_k を内包する Φ_k のイメージ

(2) Galerkin 近似

次に，i 行要素が 1 となる単位ベクトルを $\boldsymbol{e}_i \in \mathbb{R}^n$ とし，$\boldsymbol{e}_i$ を基底ベクトルとする部分空間 $G_k \in \mathbb{R}^{n \times \phi_k}$ を次式で定義する．

$$G_k \equiv \left[\boldsymbol{e}_i \,\middle|\, i \in \Phi_k \right], \quad 1 \le k \le m \tag{18}$$

ここで，(14)式の部分空間 V と(18)式の部分空間 G_k を用いた未知ベクトル $\boldsymbol{x}$ の部分空間近似を $\boldsymbol{u}_k \in \mathbb{R}^n$ として，次式で表す．

$$\boldsymbol{u}_k = G_k \, \boldsymbol{y}_k + V \, \boldsymbol{z}_k, \quad 1 \le k \le m \tag{19}$$

ここに，$\boldsymbol{y}_k \in \mathbb{R}^{\phi_k}$ と $\boldsymbol{z}_k \in \mathbb{R}^{\alpha}$ は係数ベクトルである．

ところで，部分集合 Φ_k に属する自由度について，(19)式では右辺第1項 G_k と第2項 V の張る空間が重複している．G_k は単位ベクトルを基底ベクトルとしているので，Φ_k に属する自由度については，(19)式の第1項は(1)式の情報を完全に引き継いでいる．よって，Φ_k に属する自由度については G_k による表現で十分であり，これに相当する V の成分は省略することが可能である．

そこで，$\hat{\boldsymbol{v}}_i^{(l)}$ の Φ_k に属する自由度の成分を強制的にゼロとしたベクトルを $\hat{\boldsymbol{v}}_i^{(l,k]} \in \mathbb{R}^n$ として，次式で定義する．

$$\hat{\boldsymbol{v}}_i^{(l,k]} \equiv \left[v_{i,j} \,\middle|\, j \in \Omega; v_{i,j} = 0, j \notin \Omega_l \text{ or } j \in \Phi_k \right], \; l \ne k \tag{20}$$

さらに，$\hat{\boldsymbol{v}}_i^{(l,k]}$ からなる部分空間を $\hat{V}^{(l,k]}$ として次式で定義する．

$$\hat{V}^{(l,k]} \equiv \left[\hat{\boldsymbol{v}}_1^{(l,k]}, \hat{\boldsymbol{v}}_2^{(l,k]} \ldots, \hat{\boldsymbol{v}}_{\alpha_l}^{(l,k]} \right], \quad l \ne k \tag{21}$$

なお，$l \notin \Gamma_k$ となる l については，Ω_l と Ω_k は重ならないので，$\hat{V}^{(l,k]} = \hat{V}^{(l)}$ となる．ここで，$l = k$ の時 $\hat{\boldsymbol{v}}_i^{(l,k]} = 0$ となることに注意して，$\hat{V}^{(l,k]}$ からなる部分空間 $V^{[k]} \in \mathbb{R}^{n \times \alpha^{[k]}}$ を次式で定義する．

$$V^{[k]} \equiv \left[\hat{V}^{(1,k]}, \ldots, \hat{V}^{(k-1,k]}, \hat{V}^{(k+1,k]}, \ldots, \hat{V}^{(m,k]} \right] \tag{22}$$

ここに，$\alpha^{[k]}$ は $V^{[k]}$ を張る基底ベクトルの本数であり，$\alpha^{[k]} = \alpha - \alpha_k$ となる．

(18)式の部分空間 G_k と(22)式の部分空間 $V^{[k]}$ を用いた未知ベクトル $\boldsymbol{x}$ の部分空間近似 $\boldsymbol{u}_k$ は次式で表される．

$$\boldsymbol{u}_k = G_k \, \boldsymbol{y}_k + V^{[k]} \, \boldsymbol{z}_k^{[k]}, \quad 1 \le k \le m \tag{23}$$

(23)式の係数ベクトル $\boldsymbol{y}_k$ と $\boldsymbol{z}_k^{[k]} \in \mathbb{R}^{\alpha^{[k]}}$ は次式を解いて得られる．

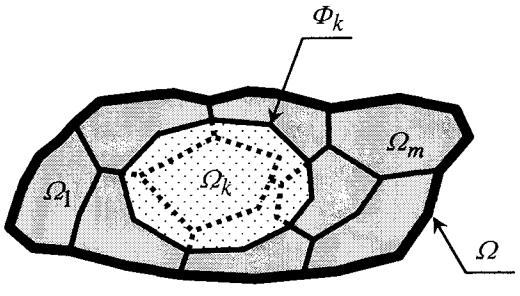
$$C_k^{[k]} \, \boldsymbol{w}_k^{[k]} = \boldsymbol{f}_k^{[k]}, \quad 1 \le k \le m \tag{24}$$

ここに，$C_k^{[k]} \in \mathbb{R}^{(\alpha^{[k]}+\phi_k) \times (\alpha^{[k]}+\phi_k)}$ と $\boldsymbol{w}_k^{[k]} \in \mathbb{R}^{\alpha^{[k]}+\phi_k}$ および $\boldsymbol{f}_k^{[k]} \in \mathbb{R}^{\alpha^{[k]}+\phi_k}$ は次式で定義される．

$$C_k^{[k]} \equiv \begin{bmatrix} G_k^{\mathrm{T}} A G_k & G_k^{\mathrm{T}} A V^{[k]} \\ V^{[k]\mathrm{T}} A G_k & V^{[k]\mathrm{T}} A V^{[k]} \end{bmatrix} \tag{25}$$

$$\boldsymbol{w}_k^{[k]} \equiv \begin{bmatrix} \boldsymbol{y}_k \\ \boldsymbol{z}_k^{[k]} \end{bmatrix} \tag{26}$$

$$\boldsymbol{f}_k^{[k]} \equiv \begin{bmatrix} G_k^{\mathrm{T}} \boldsymbol{b} \\ V^{[k]\mathrm{T}} \boldsymbol{b} \end{bmatrix} \tag{27}$$



図—3　$C_k^{[k]} \, \boldsymbol{w}_k^{[k]} = \boldsymbol{f}_k^{[k]}$ のイメージ

図―3は(24)式のイメージを表したものである．図中の灰色の部分は部分空間 $V^{[k]}$ が表す空間のイメージで，太い実線で囲まれた細かな点の部分は部分空間 G_k が表す空間のイメージである．太い点線は改良前の部分空間 V の形を示したものである．

2．3 最良近似解の構成と収束判定

(1) 最良近似解の構成

(23)式で計算される部分空間近似 $\boldsymbol{u}_k$ を用いた解の構成について述べる．$V^{[k]}$ と G_k の基底ベクトルの構造より，$\boldsymbol{u}_k$ の自由度のうち Ω_k に属する自由度は未知ベクトル $\boldsymbol{x}$ のそれの最良近似となる．そこで，$\boldsymbol{u}_k$ の自由度のうち Ω_k に属さない自由度の値を強制的にゼロに置き換えたベクトルを $\hat{\boldsymbol{u}}_k^{(k)} \in \mathbb{R}^n$ とする．$\hat{\boldsymbol{u}}_k^{(k)}$ は形式的に次式で表せる．

$$\begin{aligned}\hat{\boldsymbol{u}}_k^{(k)} &\equiv \left[\, u_{k,i} \,\middle|\, i \in \Omega;\quad u_{k,i} = 0, i \notin \Omega_k \,\right] \\ &= H_k\, H_k^{\mathrm{T}}\, \boldsymbol{u}_k\end{aligned} \tag{28}$$

ここに，$H_k \in \mathbb{R}^{n\times\omega_k}$ は単位ベクトルを基底ベクトルとする次式で定義される部分空間である．

$$H_k \equiv \left[\, \boldsymbol{e}_i \,\middle|\, i \in \Omega_k \,\right] \tag{29}$$

ここで，(23)式を(28)式に代入し，(20)式，(21)式，(22)式で示す $V^{[k]}$ の構造を考慮すると次式を得る．

$$\hat{\boldsymbol{u}}_k^{(k)} = H_k\, H_k^{\mathrm{T}}\, G_k\, \boldsymbol{y}_k \tag{30}$$

(30)式は形式的なものであり，$\hat{\boldsymbol{u}}_k^{(k)}$ は $\boldsymbol{y}_k$ より必要な自由度の値を抽出する簡単な操作で計算できる．

最終的には，(1)式の未知ベクトル $\boldsymbol{x}$ の最良近似を $\tilde{\boldsymbol{x}} \in \mathbb{R}^n$ として次式で定義する．

$$\tilde{\boldsymbol{x}} \equiv \hat{\boldsymbol{u}}_1^{(1)} + \hat{\boldsymbol{u}}_2^{(2)} + \cdots + \hat{\boldsymbol{u}}_m^{(m)} \tag{31}$$

(2) 残差ベクトルによる収束判定

近似ベクトル $\tilde{\boldsymbol{x}}$ における(1)式の残差ベクトル $\boldsymbol{r} \in \mathbb{R}^n$ を次式で定義する．

$$\begin{aligned}\boldsymbol{r} &\equiv \left[r_1, r_2, \ldots, r_m\right]^{\mathrm{T}} \\ &= \boldsymbol{b} - A\,\tilde{\boldsymbol{x}}\end{aligned} \tag{32}$$

ベクトル $\boldsymbol{r}$ の内，集合 Ω_k に属さない自由度の値をゼロとしたベクトルを $\hat{\boldsymbol{r}}^{(k)} \in \mathbb{R}^n$ として次式で定義する．

$$\begin{aligned}\hat{\boldsymbol{r}}^{(k)} &\equiv \left[\, r_i \,\middle|\, i \in \Omega;\quad r_i = 0, i \notin \Omega_k \,\right] \\ &= H_k\, H_k^{\mathrm{T}}\, \boldsymbol{r}\end{aligned} \tag{33}$$

また，ベクトル $\boldsymbol{b}$ の内，集合 Ω_k に属さない自由度をゼロとしたベクトルを $\hat{\boldsymbol{b}}^{(k)} \in \mathbb{R}^n$ とし，次式で定義する．

$$\begin{aligned}\hat{\boldsymbol{b}}^{(k)} &\equiv \left[\, b_i \,\middle|\, i \in \Omega;\quad b_i = 0, i \notin \Omega_k \,\right] \\ &= H_k\, H_k^{\mathrm{T}}\, \boldsymbol{b}\end{aligned} \tag{34}$$

(33)式と(34)式を用いた収束判定を次式で定義する．

$$\varepsilon \equiv \frac{\|\boldsymbol{r}\|}{\|\boldsymbol{b}\|} = \frac{\sum_{k=1}^{m} \left\|\hat{\boldsymbol{r}}^{(k)}\right\|}{\sum_{k=1}^{m} \left\|\hat{\boldsymbol{b}}^{(k)}\right\|} \le \varepsilon_a \tag{35}$$

ここに，ε は相対残差ノルムで，ε_a は収束判定値とする．$\|\ \|$ はベクトルのユークリッドノルムを示す．

2．4 部分空間 $\hat{V}^{(k)}$ の初期値と更新

(1) 部分空間の初期値

提案法で反復を開始するには基底ベクトルの初期値 $\hat{\boldsymbol{v}}_1^{(k)}$ を決めなければならない．予め近似ベクトル $\tilde{\boldsymbol{x}}$ が得られている場合は，次式により $\hat{\boldsymbol{v}}_1^{(k)}$ を設定できる．

$$\hat{\boldsymbol{v}}_1^{(k)} = \frac{H_k H_k^{\mathrm{T}}\, \tilde{\boldsymbol{x}}}{\left\| H_k H_k^{\mathrm{T}}\, \tilde{\boldsymbol{x}} \right\|} \tag{36}$$

提案法と異なる解法を用いて(1)式を解くことにより $\tilde{\boldsymbol{x}}$ を作ることもできるが，より簡便な方法で初期ベクトル $\hat{\boldsymbol{v}}_1^{(k)}$ を設定するのが望ましい．

$\hat{\boldsymbol{b}}^{(k)} \neq 0$ であれば，次式を $\boldsymbol{y}_k$ について反復法や直接法で解き，

$$G_k^{\mathrm{T}} A G_k\, \boldsymbol{y}_k = \hat{\boldsymbol{b}}^{(k)},\quad 1 \le k \le m \tag{37}$$

$\boldsymbol{y}_k$ から次式により $\hat{\boldsymbol{v}}_1^{(k)}$ を設定できる．

$$\hat{\boldsymbol{v}}_1^{(k)} = \frac{H_k H_k^{\mathrm{T}} G_k\, \boldsymbol{y}_k}{\left\| H_k H_k^{\mathrm{T}} G_k\, \boldsymbol{y}_k \right\|} \tag{38}$$

この方法は，$n \gg \phi_k$ であれば(37)式を解く計算量は(1)式を解くことに比べてはるかに少なく，確実に初期ベクトルを設定できる．ただし，$\hat{\boldsymbol{b}}^{(k)} = 0$ となる場合は使えない．

最も簡単な方法は $\hat{\boldsymbol{v}}_1^{(k)}$ の要素を次式で決める方法である．

$$v_{1,i} = \begin{cases} \dfrac{1}{\sqrt{\omega_k}}, & i \in \Omega_k \\ 0, & i \notin \Omega_k \end{cases} \tag{39}$$

我々の数値実験では(37)式を解く方法に比べて僅かに反復数が増加するものの，この方法で不都合は生じていない．

(2) 部分空間の更新

近似解 $\tilde{\boldsymbol{x}}$ の誤差を反復と共に小さくするためには部分空間 $V^{[k]}$ を改良する必要がある．そのためには部分空間 $\hat{V}^{(k)}$ を改良すれば良い．Ω_k の自由度に対してベクトル $\hat{\boldsymbol{u}}_k^{(k)}$ は解ベクトルの最良近似となるので，$\hat{\boldsymbol{u}}_k^{(k)}$ は基底ベクトル $\hat{\boldsymbol{v}}_i^{(k)}, 1 \le i \le \alpha_k$ と直交する成分を含むものと考えられる．これより，$\hat{\boldsymbol{u}}_k^{(k)}$ を用いる $\hat{V}^{(k)}$ の更新法を考えることができる．

アルゴリズム—1は，$\hat{\boldsymbol{v}}_i^{(k)}, 1 \le i \le \alpha_k$ を基準として $\hat{\boldsymbol{u}}_k^{(k)}$ をこれらに直交化させ，これを新しい基底ベクトルとする部分空間の更新方法を示している．この方法では反復の進行と共に基底ベクトルの数が 1 個ずつ増加するので，計算機の主記憶容量に制限があり所定の精度の近似ベクトルを得るのに多くの反復を必要とする問題においては，提案法のアルゴリズムに決定的な破綻を生じる可能性がある．これを回避するには，提案の基底ベクトルの本数を制限できる部分空間 $\hat{V}^{(k)}$ の更新法が必要である．

アルゴリズム—1 $\hat{\boldsymbol{u}}_k^{(k)}$ による $\hat{V}^{(k)}$ の更新法
（k は部分集合 Ω_k の番号を示す．）

for $j = 1, 2, \ldots, \alpha_k$

$$\hat{\boldsymbol{u}}_k^{(k)} \leftarrow \hat{\boldsymbol{u}}_k^{(k)} - \left(\hat{\boldsymbol{u}}_k^{(k)\mathrm{T}}\, \hat{\boldsymbol{v}}_j^{(k)}\right) \hat{\boldsymbol{v}}_j^{(k)}$$

end (j loop)

$$\hat{\boldsymbol{v}}_{\alpha_k+1}^{(k)} = \frac{\hat{\boldsymbol{u}}_k^{(k)}}{\left\|\hat{\boldsymbol{u}}_k^{(k)}\right\|}$$

$$\alpha_k \leftarrow \alpha_k + 1$$

アルゴリズム—2は $\hat{\boldsymbol{u}}_k^{(k)}$ を基準にして基底ベクトル $\hat{\boldsymbol{v}}_i^{(k)}, 1 \le i \le \alpha_k$ を $\hat{\boldsymbol{u}}_k^{(k)}$ に再直交化する方法を示したものである．この方法では近似ベクトル $\hat{\boldsymbol{u}}_k^{(k)}$ に関する最新の情報を常に保有するので，既存の一部の基底ベクトルを省略することにより基底ベクトルの本数を制限できる．これにより提案法のアルゴリズムの破綻を回避できるが，既存の基底ベクトルが全て書き換えられることにより $C_k^{[k]}, \boldsymbol{b}_k^{[k]}$ の要素の一部が書き換えられるので，それに伴う通信時間や計算時間のロスが生じる．

アルゴリズム—2 $\hat{\boldsymbol{u}}_k^{(k)}$ による $\hat{V}^{(k)}$ の制限付更新法
（k は部分集合 Ω_k の番号を示す．）

$$\hat{\boldsymbol{v}}_{1,new}^{(k)} = \frac{\hat{\boldsymbol{u}}_k^{(k)}}{\left\|\hat{\boldsymbol{u}}_k^{(k)}\right\|}$$

for $j = 1$ to $\min(\alpha_k, \alpha_{\max} - 1)$

for $i = 1$ to j

$$\hat{\boldsymbol{v}}_j^{(k)} \leftarrow \hat{\boldsymbol{v}}_j^{(k)} - \left(\hat{\boldsymbol{v}}_j^{(k)\mathrm{T}}\, \hat{\boldsymbol{v}}_{i,new}^{(k)}\right) \hat{\boldsymbol{v}}_{i,new}^{(k)}$$

end (i loop)

$$\hat{\boldsymbol{v}}_{j+1,new}^{(k)} = \frac{\hat{\boldsymbol{v}}_j^{(k)}}{\left\|\hat{\boldsymbol{v}}_j^{(k)}\right\|}$$

end (j loop)

$$\alpha_k \leftarrow \min(\alpha_k + 1, \alpha_{\max})$$

2．5 $C_k^{[k]}\, \boldsymbol{w}_k^{[k]} = \boldsymbol{f}_k^{[k]}$ の解法

(24)式の減次された連立一次方程式は反復法により解くものとしている．また，(24)式はノード単独で解かれるので，逐次計算において高速な反復解法を採用するのが望ましい．本論文では，係数行列 $C_k^{[k]}$ が対称行列であるため，前処理付き CG 法を採用する．前処理法としてはしきい値付き不完全LU分解[12]の対称版であるしきい値付き不完全LDL$^{\mathrm{T}}$分解（ILDL$^{\mathrm{T}}$）を採用する．

2．6 計算アルゴリズム

前節まで説明した提案法の計算方法を計算アルゴリズムとしてアルゴリズム—3にまとめる．提案法の計算アルゴリズムは準備(Preparation)と初期設定(Initialization)および主反復（Main iteration）の 3 つに大きく分けられる．

準備とは集合 Ω_k と Φ_k の決定，係数ベクトル $G_k^{\mathrm{T}}\boldsymbol{b}$ と係数行列 $G_k^{\mathrm{T}}\, A\, G_k$ の作製であり，いわゆる問題行列の作製である．これは離散化の手法や計算モデルのデータ構造とも密接に関係する．初期設定とは部分空間 $\hat{V}^{(k)}$ の初期値の設定と前処理行列の計算で構成される．主反復は係数行列 $C_k^{[k]}$ の計算，連立方程式 $C_k^{[k]}\, \boldsymbol{w}_k^{[k]} = \boldsymbol{f}_k^{[k]}$ の求解，近似解 $\hat{\boldsymbol{u}}_k^{(k)}$ の構成，収束判定および部分空間 $\hat{V}^{(k)}$ の更新で構成される．準備と初期設定は主反復を実行する前に一度だけ実行すればよい．主反復は近似ベクトル $\tilde{\boldsymbol{x}}$ が収束するまで繰り返し実行される．

アルゴリズム—3 提案法の計算アルゴリズム
（k は部分集合 Ω_k の番号を示す）

(Preparation)

P1: Define Ω_k and Φ_k.

P2: Set $G_k^{\mathrm{T}}\boldsymbol{b}$ and $G_k^{\mathrm{T}}\, A\, G_k$.

(Initialization)

I1: Compute an initial subspace $\hat{V}^{(k)}$.

I2: Compute ILDL$^{\mathrm{T}}$ of $G_k^{\mathrm{T}} A G_k$.

(Main iteration)

M1: Compute $C_k^{[k]}$ and $\boldsymbol{f}_k^{[k]}$.

M2: Solve $\boldsymbol{w}_k^{[k]}$ from $C_k^{[k]}\, \boldsymbol{w}_k^{[k]} = \boldsymbol{f}_k^{[k]}$.

M3: Compute $\hat{\boldsymbol{u}}_k^{(k)}$.

M4: Compute $\hat{\boldsymbol{r}}^{(k)}$.

M5: Check $\varepsilon = \|\boldsymbol{r}\| / \|\boldsymbol{b}\| \le \varepsilon_a$.

M6: Update $\hat{V}^{(k)}$.

Go to M1.

3. 数値計算例

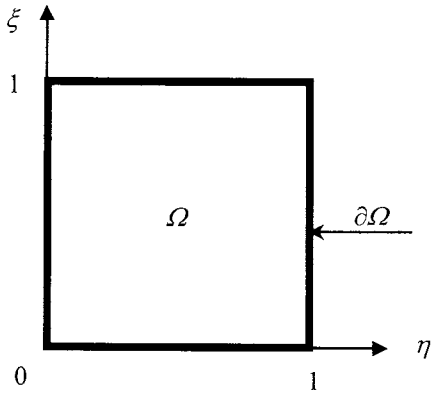
3. 1 計算モデルと計算機

図—4に示す正方領域 Ω と境界 $\partial\Omega$ で定義される次のポアソン方程式の未知関数 $x(\eta,\xi)$ の数値解を求める問題で，提案法のアルゴリズムを検証する．

$$\frac{\partial^2 x(\eta,\xi)}{\partial\eta^2}+\frac{\partial^2 x(\eta,\xi)}{\partial\xi^2}=f(\eta,\xi) \quad \text{in } \Omega \tag{40}$$

$$x(\eta,\xi)=0, \quad \text{on } \partial\Omega \tag{41}$$

$$f(\eta,\xi)=\begin{cases}1 & ,0\le\eta\le1,\ 0\le\xi\le0.5\\ 0 & ,0\le\eta\le1,\ 0.5<\xi\le1\end{cases} \tag{42}$$



図—4 計算領域の形状

ここに，$f(\eta,\xi)$ は既知関数である．5 点差分を用いて(40)式を離散化すると(1)式の連立一次方程式を得る．差分格子は正方格子とする．表—1は計算モデルにおける差分格子の分割数と総自由度数を整理したものである．

計算には自作の PC クラスターを用いた．ノード数は 16 であり，ノードは 1 個の CPU を実装している．各ノードは 24 ポート×100Base-TX のスイッチングハブで結合した．ノードの仕様を表—2に示す．OS にはLinuxを，通信ライブラリーにはmpich-1.2.1[13]を用いた．計算プログラムはC言語を用いて記述した．

表—1 計算モデルの自由度

Model	Number of divisions		Number of freedoms, n
	axis ξ	axis η	
A	501	501	250,000
B	601	601	360,000
C	701	701	490,000

表—2 自作PCクラスターのノード仕様

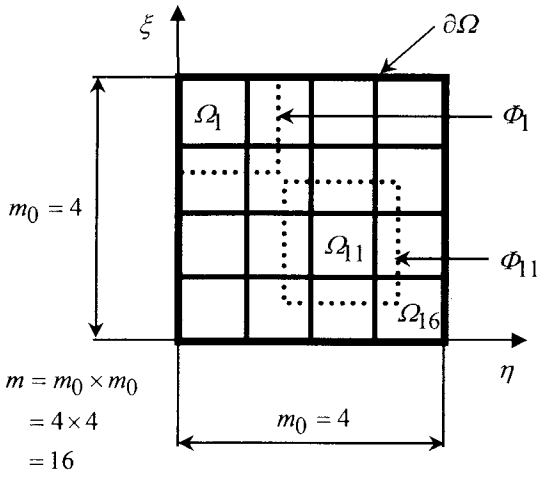
Node	N_1-N_8	N_9-N_{16}
CPU	Pentium Ⅲ(933MHz)	Celeron(1.7GHz)
DRAM	512MB	512MB
LAN	100Base-TX	100Base-TX

3. 2 部分集合 Ω_k と Φ_k の定義

図—5は，部分集合の数が $m=16$ となる例題を用いて，提案法における自由度の部分集合 Ω_k と Φ_k の定義を説明するものである．境界 $\partial\Omega$ で囲まれた領域 Ω を，η 方向に $m_0=4$ 分割，同じく ξ 方向に $m_0=4$ 分割し，実線で示す $m=m_0\times m_0=16$ 個の矩形領域で覆い，k 番目の矩形の中にある自由度で部分集合 Ω_k を構成するものとする．各部分集合の自由度数が大体同じ程度になるように分割する．

また，点線で示す矩形で k 番目の矩形を囲み，点線の矩形の中にある自由度で部分集合 Φ_k を構成する．図中では Ω_1 と対応する Φ_1 と，Ω_{11} と対応する Φ_{11} を例示している．

なお，分割数 m_0 は 3, 4, 5, 6, 7 の 5 種類，つまり部分集合の数 m は 9, 16, 25, 36, 49 の 5 種類とする．



図—5 Ω_k と Φ_k の形状と分割モデルの一例

3. 3 収束の様子

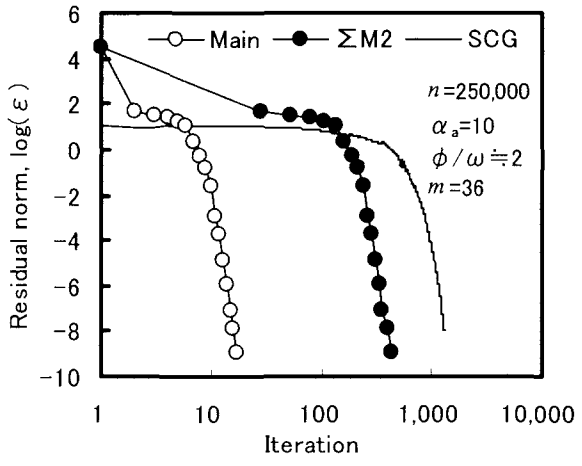
図—6は Model-A における提案法の相対残差ノルム ε の収束の様子を，提案法における主反復の回数(Main と表示)とアルゴリズム—3の M2 における PCG 法の反復回数の累積回数(ΣM2 と表示)の二つの指標で示したものである．反復回数の累積回数は回数が最も多い領域中央部 Ω_{15} のものとした．比較のために，並列機への実装が容易で並列効果の高い対角スケーリング付きCG法(SCG

法)の収束も示す．提案法の計算条件は図中に示す．SCG 法は ε が 10^{-8} より小さくなるのに 1296 回の反復を必要としたが，提案法は主反復 16 回で収束し，PCG 法の反復回数の最大累積回数は 428 回であった．

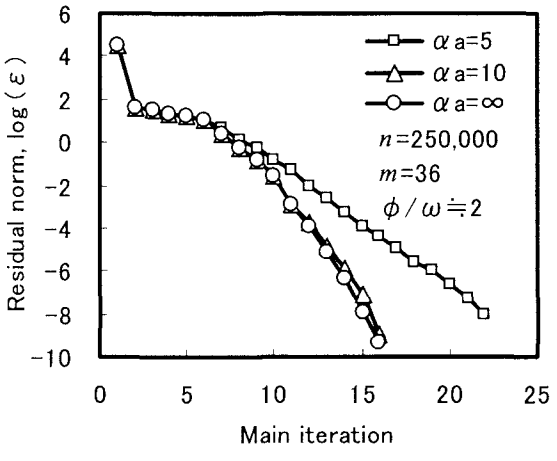
図―7は，基底ベクトルの最大本数 α_a を5本，10本，無制限とした場合の収束を比較したものである．他の条件は図―6と同じである．最大本数が多くなると収束が強くなる傾向を確認できる．今回の例では無制限の場合，16 回の反復で残差ノルムが 10^{-8} より小さくなっている．ただし，これは最大本数を10本とする場合の収束と大きな違いは認められない．

図―8は部分集合 Ω_k と Φ_k の自由度数の比 ϕ/ω を概ね 2, 3, 4 と変化させた場合の収束を比較したものである．他の条件は図―6と同じである．ϕ/ω を大きくすると収束が強くなることが確認できる．

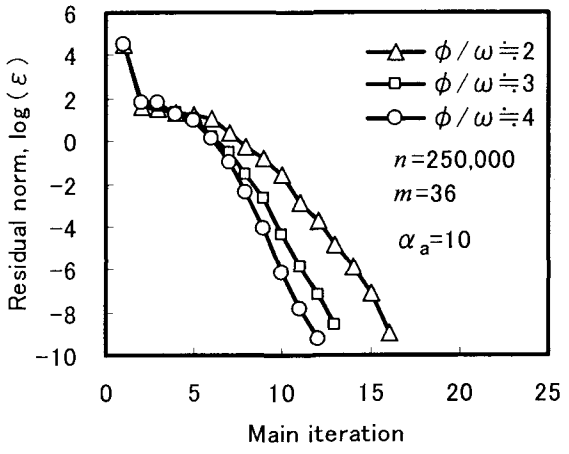
図―9は，部分集合の数 m と $\varepsilon \le 10^{-8}$ となるのに要した反復回数の関係を整理したものである．他の条件は図―6と同じである．m の増加に伴い反復回数が増加する傾向が確認できる．ただし，反復回数の増加の割合は m の増加の割合に比べて少なく，m が大きくなると増加の割合が少なくなる傾向を確認できる．



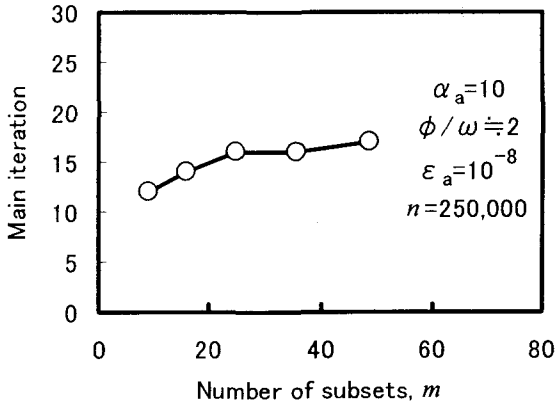
図―6 提案法の収束履歴の一例



図―7 基底ベクトルの最大本数と収束



図―8 自由度数比 ϕ/ω と収束



図―9 部分集合の数 m と反復回数

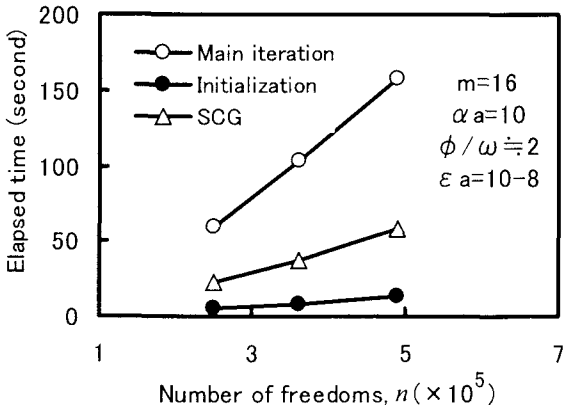
3．4 計算時間

図―10は提案法の計算時間(計算開始から計算終了までの経過時間)と自由度数の関係を調べたものである．計算の条件は図中に示す．提案法の計算時間は初期設定と主反復に分けて示した．参考のために SCG 法の計算時間を併記した．初期設定は $\mathrm{ILDL}^{\mathrm{T}}$ に要する時間である．初期設定の時間は主反復の10%程度である．主反復の計算時間は自由度数の増加に比例して増加している．提案法の主反復は SCG 法の3倍程度の計算時間を費やしている．なお，実験では計算性能に 2 倍程度の開きがある 2 種類のノードを使用しているが，提案法と SCG 法ともにノードの計算量が同程度になるように分担量を決めているため，性能の低いノードの計算時間が全体の計算時間に現れる．

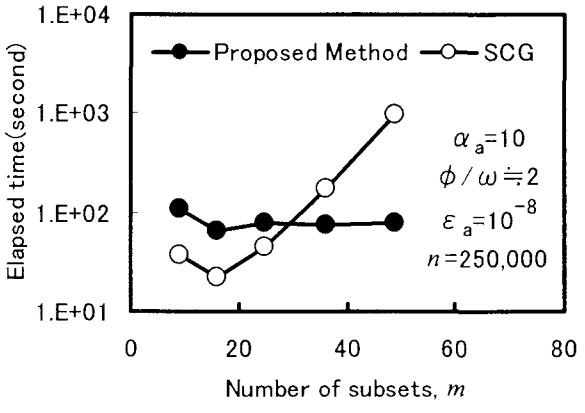
図―11は部分集合の数 m すなわちノード数を増加させた時の提案法の計算時間の変化を調べたものである．参考のために，ノード数を変化させたときの SCG 法の計算時間を併記する．提案法の計算条件は図中に示す．ただし，本計算は 16 ノードの PC クラスター上で mpich のマルチプロセス機能を利用した擬似的な並列計算の結果であり，16 を超えるノード数では1CPU 内でのプロセスの待ち時間が計算時間に加算されるので注意が必要である．

図―11より，提案法は m が 16 を超える擬似的な並列

計算環境において，m の増加に伴う計算時間の増加は見られない．これは数値実験のノード数の範囲においてノード数による並列効果が定性的に見込まれることを示している．SCG 法の計算時間が 16 を超えるノード数において急激に増加したのは，提案法に比べて通信回数が多く，特殊な計算環境下のプロセス実行待ち時間が通信の遅延等に結びついた結果と見られる．



図—10 自由度数 n と計算時間



図—11 部分集合の数 m と計算時間
（16 ノードの PC クラスターを使用）

4. まとめ

本論文では区分的に値を持つ基底ベクトルを用いた連立一次方程式の反復解法を提案し，16 ノードの PC クラスターを用いた数値実験により下記のことを明らかにした．

1) 提案法は SCG 法に比べて少ない主反復回数で方程式を解くことができる．
2) 提案法では基底ベクトルの本数を多く設定することにより反復回数を少なくできる．
3) 提案法では自由度数の比 ϕ/ω を大きくすることにより反復回数を少なくできる．
4) 提案法の反復回数は部分集合の数 m の増加に伴い僅かに増加する傾向がある．

今回の数値実験は，提案法の基本的な収束を確認するため，得られた解の検証が簡単な数学モデルで行った．今後は，工学や理学の応用問題において提案法の基本的な収束を検討する予定である．

また，実験に用いた計算環境が小規模であり，提案法のノード数に対する適用範囲や性能は明らかにできなかった．今後，ノード数が多くなる中・大規模な分散メモリー型並列計算機において提案法の性能を確認する必要がある．なお，紙面の都合上，ノード間通信のための計算手順と通信時間については機会をあらためて報告する．

参考文献

1) W. L. Briggs, V. E. Henson and S. F. McCormick: *A Multigrid Tutorial Second Edition*, SIAM, Philadelphia, pp. 137-161, 2001.
2) M. R. Hestenes and E. Steifel: Method of conjugate gradients for solving linear systems, *J. Res. Nat. Bur. Standards*, Vol.49, pp.409-436, 1952.
3) J. A. Meijerink and H. A. van der Vorst: An iterative solution method for linear systems of which the coefficient matrix is a symmetric M-matrix, *Math. Comp.*, Vol. 31, pp.148-162, 1977.
4) 藤井昭宏，西田晃，小柳義夫：領域分割による並列 AMG アルゴリズム，情報処理学会論文誌，Vol.44, No.SIG6(ACS1), pp.9-17, 2003.
5) 片山拓朗，平井一男：区分的に値を持つ基底ベクトルを用いた連立方程式の一並列反復解法，日本応用数理学会年会講演会予稿集，Vol.2002, pp.81-81, http://pjsiam.jstage.jst.go.jp/ja, 2002.
6) 片山拓朗：区分的に値を持つ基底ベクトルを用いた連立方程式・並列反復解法の収束，日本応用数理学会 2003 年度年会講演会予稿集，pp.120-121, 2003.
7) Y. Saad: *Iterative Methods For Sparse Linear Systems*, PWS Publishing Company, Boston, pp.383-424, 1996.
8) U. Trottenberg, C. W. Oosterlee, A. Schüller: *Multigrid*, Academic Press, London, pp.533-572, 2001.
9) 片山拓朗，平井一男，柏木光博：区分的に値を持つ基底ベクトルを用いた固有ベクトルの近似法，日本応用数理学会 2000 年度年会講演会予稿集，pp.528-529, 2000.
10) 片山拓朗，平井一男，柏木光博：区分的に値を持つ基底ベクトルを用いた逆べき乗法の高速化，土木学会第 56 回学術講演会概要集，I-B246, 2001.
11) 片山拓朗，平井一男，柏木光博：区分的に値を持つ基底ベクトルを用いた逆べき乗法の高速化法，日本応用数理学会 2001 年度年会講演会予稿集，pp.272-273, 2001.
12) Y. Saad, ILUT: A dual threshold incomplete ILU factorization, *Numerical Linear Algebra with Application*, 1, pp. 387-402, 1994.
13) http://www-unix.mcs.anl.gov/mpi/mpich

（2004年 4 月16日受付）