博士論文（要約）

論文題目

並列有限要素法における線形ソルバの統一的フレームワークに関する研究

氏名　　森田 直樹

目次

図目次

表目次

第１章　序論

第１章は，単行本もしくは雑誌掲載等の形で刊行される予定があるため，インターネット公表できません．

第 2 章　並列有限要素解析のための線形ソルバ

2.1　緒言

領域分割に基づく並列有限要素法の基礎となる領域分割法は，non-overlapping 型領域分割と overlapping 型領域分割に大別される．本章では，non-overlapping 型領域分割と overlapping 型領域分割の両者に対し，分割領域から得られる行列と解くべき連立一次方程式 $\boldsymbol{Ax} = \boldsymbol{b}$ の対応について述べる．さらに，non-overlapping 型領域分割と overlapping 型領域分割の両者に対し，Schur complement system による静的縮約を導出する過程を述べる．

次に，有限要素離散化から得られる連立一次方程式を解くための線形ソルバに関し，直接法と反復法の両者について述べる．反復法については，一般的な非定常反復法の他に，高い並列計算性能を実現するための手法として注目されている通信隠蔽型非定常反復法についても取り扱う．さらに，反復法を安定に解くための重要な技術である，反復法前処理について述べる．

2.2　Non-overlapping 型領域分割法

non-overlapping 型領域分割は，図 2.1 に示す領域 Ω を，式 (2.1)，式 (2.2) のように分割する．ここで，分割領域 $\Omega^{(i)}$，ND は領域分割数である．

$$\Omega = \bigcup_{i}^{\mathrm{ND}} \Omega^{(i)} \tag{2.1}$$

$$\bigcap_{i}^{\mathrm{ND}} \Omega^{(i)} = \emptyset \tag{2.2}$$

このとき，他の領域と隣り合う境界 Γ は，領域 i の境界 $\partial\Omega^{(i)}$ を用いて，式 (2.3) で定義される．

$$\Gamma = \bigcap_{i}^{\mathrm{ND}} \partial\Omega^{(i)} \tag{2.3}$$

また，領域 Ω の境界 $\partial\Omega$ は，分割領域の境界のうち境界 $\partial\Omega$ に属する境界 $\partial\Omega^{(i)}_{\mathrm{BOUND}}$ を用いて式 (2.4) で表される．

$$\partial\Omega = \bigcup_{i}^{\mathrm{ND}} \partial\Omega^{(i)}_{\mathrm{BOUND}} = \bigcup_{i}^{\mathrm{ND}} \partial\Omega^{(i)} \backslash \Gamma = \bigcup_{i}^{\mathrm{ND}} \{\partial\Omega^{(i)} \cap (\text{not } \Gamma)\} \tag{2.4}$$

以上の定義に従って，領域 Ω の有限要素離散化を行い，式 (2.5) のような連立一次方程式を得る．ここで，$\boldsymbol{A}$ は係数行列，$\boldsymbol{f}$ は外力ベクトル，$\boldsymbol{x}$ は解ベクトルである．

$$\boldsymbol{Ax} = \boldsymbol{f} \tag{2.5}$$

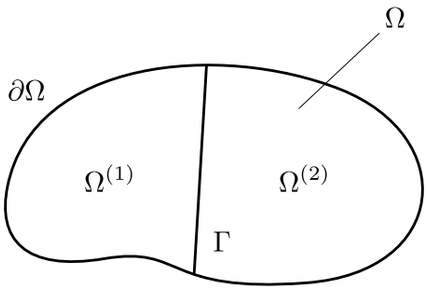


Fig. 2.1 Schematic representation of non-overlapping domain decompotision method (number of subdomains ND = 2)

式 (2.5) の行列，ベクトルを行列形式で書き換えると，式 (2.6)，式 (2.7)，式 (2.8) で表される．

$$\boldsymbol{A} = \begin{pmatrix} \boldsymbol{A}_{II}^{(1)} & 0 & \boldsymbol{A}_{\Gamma I}^{(1)} \\ 0 & \boldsymbol{A}_{II}^{(2)} & \boldsymbol{A}_{\Gamma I}^{(2)} \\ \boldsymbol{A}_{I\Gamma}^{(1)} & \boldsymbol{A}_{I\Gamma}^{(2)} & \boldsymbol{A}_{\Gamma\Gamma} \end{pmatrix} \tag{2.6}$$

$$\boldsymbol{x} = \begin{pmatrix} \boldsymbol{x}_I^{(1)} \\ \boldsymbol{x}_I^{(2)} \\ \boldsymbol{x}_\Gamma \end{pmatrix} \tag{2.7}$$

$$\boldsymbol{f}_i = \begin{pmatrix} \boldsymbol{f}_I^{(1)} \\ \boldsymbol{f}_I^{(2)} \\ \boldsymbol{f}_\Gamma \end{pmatrix} \tag{2.8}$$

このとき，$\boldsymbol{A}_{\Gamma\Gamma}$，$\boldsymbol{f}_{\Gamma\Gamma}$ はそれぞれ式 (2.9)，式 (2.10) で定義される．

$$\boldsymbol{A}_{\Gamma\Gamma} = \boldsymbol{A}_{\Gamma\Gamma}^{(1)} + \boldsymbol{A}_{\Gamma\Gamma}^{(2)} \tag{2.9}$$

$$\boldsymbol{f}_{\Gamma\Gamma} = \boldsymbol{f}_{\Gamma\Gamma}^{(1)} + \boldsymbol{f}_{\Gamma\Gamma}^{(2)} \tag{2.10}$$

ここで，$\boldsymbol{A}_{II}^{(i)}$ は領域 i の内部領域 I に関する係数行列の要素，$\boldsymbol{A}_{I\Gamma}^{(i)}$ は領域 i の内部領域 I で境界 Γ に関する係数行列の要素，$\boldsymbol{A}_{\Gamma I}^{(i)}$ は領域 i の境界 Γ で内部領域 I に関する係数行列の要素，$\boldsymbol{A}_{\Gamma\Gamma}$ は境界 Γ に関する係数行列の要素，$\boldsymbol{f}_I^{(i)}$ は領域 i の内部領域に関する外力ベクトルの要素，$\boldsymbol{f}_\Gamma$ は境界 Γ に関する外力ベクトルの要素，$\boldsymbol{x}_I^{(i)}$ は領域 i の内部領域に関する解ベクトルの係数，$\boldsymbol{x}_\Gamma$ は境界 Γ に関する解ベクトルの要素であり，$\boldsymbol{A}_{\Gamma\Gamma}^{(i)}$ は $\boldsymbol{A}_{\Gamma\Gamma}$ のうち領域 i が寄与する要素，$\boldsymbol{f}_\Gamma^{(i)}$ は $\boldsymbol{f}_\Gamma$ のうち領域 i が寄与する要素である．

以上の定義から，式 (2.6)，式 (2.7) を分割領域の行列形式でそれぞれ書き直すと，式 (2.11)，式 (2.12) を得る．

$$\boldsymbol{A} = \bigoplus_i^{\mathrm{ND}} \boldsymbol{A}^{(i)} = \bigoplus_i^{\mathrm{ND}} \begin{pmatrix} \boldsymbol{A}_{II}^{(i)} & \boldsymbol{A}_{\Gamma I}^{(i)} \\ \boldsymbol{A}_{I\Gamma}^{(i)} & \boldsymbol{A}_{\Gamma\Gamma}^{(i)} \end{pmatrix} \tag{2.11}$$

$$\boldsymbol{f} = \bigoplus_{i}^{\mathrm{ND}} \begin{pmatrix} \boldsymbol{f}_I^{(i)} \\ \boldsymbol{f}_\Gamma^{(i)} \end{pmatrix} \tag{2.12}$$

ここで，$\boldsymbol{A}^{(i)}$ は領域 i に関する係数行列の要素，$\bigoplus$ は有限要素法におけるアセンブリ作用素であり，各領域のローカル節点番号をグローバル節点番号に変換し，全体係数行列に足し込む演算を示す．アセンブリ作用素は，各領域のローカル節点番号をグローバル節点番号に変換する 0-1 行列 $\boldsymbol{N}$ を用いて式 (2.13) のように書き換えることができる．

$$\boldsymbol{A} = \bigoplus_{i}^{\mathrm{ND}} \boldsymbol{A}^{(i)} = \sum_{i=1}^{\mathrm{ND}} \boldsymbol{N}^{(i)} \boldsymbol{A}^{(i)} \boldsymbol{N}^{(i)t} \tag{2.13}$$

2.3 Overlapping 型領域分割法

overlapping 型領域分割は，図 2.2 に示す領域 Ω を，式 (2.14)，式 (2.15) のように分割する．ここで，分割領域 $\Omega^{(i)}$，ND は領域分割数である．

$$\Omega = \bigcup_{i}^{\mathrm{ND}} \Omega^{(i)} + \Gamma \tag{2.14}$$

$$\bigcap_{i}^{\mathrm{ND}} \Omega^{(i)} \cap \Gamma = \emptyset \tag{2.15}$$

このとき，領域 $\Omega^{(i)}$ と領域 Γ の接する境界 $\partial\Gamma^{(i)}$ は，領域 i の境界 $\partial\Omega^{(i)}$ と境界 $\partial\Gamma$ を用いて，式 (2.16) で定義される．

$$\partial\Gamma^{(i)} = \partial\Omega^{(i)} \cap \partial\Gamma \tag{2.16}$$

また，領域 Ω の境界 $\partial\Omega$ は，分割領域の境界のうち境界 $\partial\Omega$ に属する境界 $\partial\Omega^{(i)}_{\mathrm{BOUND}}$ と領域 Γ の境界のうち境界 $\partial\Omega$ に属する境界 $\partial\Gamma^{(i)}_{\mathrm{BOUND}}$ を用いて式 (2.17) で表される．

$$\begin{aligned} \partial\Omega = \bigcup_{i}^{\mathrm{ND}} \partial\Omega^{(i)}_{\mathrm{BOUND}} \cup \partial\Gamma_{\mathrm{BOUND}} &= \bigcup_{i}^{\mathrm{ND}} (\partial\Omega^{(i)} \backslash \partial\Gamma) \cup \left(\partial\Gamma \backslash \bigcup_{i}^{\mathrm{ND}} \partial\Omega^{(i)} \right) \\ &= \bigcup_{i}^{\mathrm{ND}} \{\partial\Omega^{(i)} \cap (\mathrm{not}\ \Gamma)\} \cup \left\{ \partial\Gamma \cap \left(\mathrm{not} \bigcup_{i}^{\mathrm{ND}} \partial\Omega^{(i)} \right) \right\} \end{aligned} \tag{2.17}$$

以上の定義に従って，領域 Ω の有限要素離散化を行うと，non-overlapping 型領域分割と同様な連立一次方程式 (2.5) を得る．overlapping 型領域分割法は，オーバーラップ深さを任意に指定することができる．本研究では，オーバーラップ深さは 1，すなわち有限要素 1 つ分のオーバーラップ深さのみ考慮する．式 (2.5) の行列を，ベクトルを行列形式で書き換えて，式

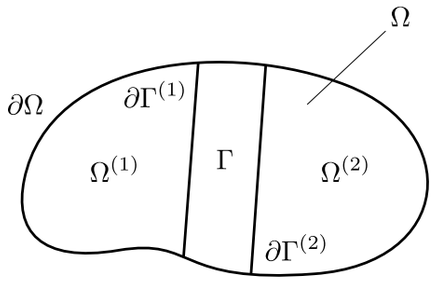


Fig. 2.2 Schematic representation of overlapping domain decompotision method (number of subdomains ND = 2)

(2.18)，式 (2.19)，式 (2.20) を得る．

$$\boldsymbol{A} = \begin{pmatrix} \boldsymbol{A}_{II}^{(1)} & \boldsymbol{A}_{\Gamma I}^{(1)} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{A}_{I\Gamma}^{(1)} & \boldsymbol{A}_{\Gamma\Gamma}^{(1)} & \boldsymbol{A}_{\Gamma\Gamma}^{(1,2)} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{A}_{\Gamma\Gamma}^{(2,1)} & \boldsymbol{A}_{\Gamma\Gamma}^{(2)} & \boldsymbol{A}_{\Gamma I}^{(2)} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{A}_{I\Gamma}^{(2)} & \boldsymbol{A}_{II}^{(2)} \end{pmatrix} \tag{2.18}$$

$$\boldsymbol{x} = \begin{pmatrix} \boldsymbol{x}_{I}^{(1)} \\ \boldsymbol{x}_{\Gamma}^{(1)} \\ \boldsymbol{x}_{\Gamma}^{(2)} \\ \boldsymbol{x}_{I}^{(2)} \end{pmatrix} \tag{2.19}$$

$$\boldsymbol{f}_i = \begin{pmatrix} \boldsymbol{f}_{I}^{(1)} \\ \boldsymbol{f}_{\Gamma}^{(1)} \\ \boldsymbol{f}_{\Gamma}^{(2)} \\ \boldsymbol{f}_{I}^{(2)} \end{pmatrix} \tag{2.20}$$

ここで，$\boldsymbol{A}_{II}^{(i)}$ は領域 i の内部領域 I に関する係数行列の要素，$\boldsymbol{A}_{I\Gamma}^{(i)}$ は領域 i の内部領域 I で境界 Γ に関する係数行列の要素，$\boldsymbol{A}_{\Gamma I}^{(i)}$ は領域 i の境界 Γ で内部領域 I に関する係数行列の要素，$\boldsymbol{A}_{\Gamma\Gamma}^{(i)}$ は領域 Γ に関する係数行列の要素，$\boldsymbol{A}_{\Gamma\Gamma}^{(i,j)}$ は領域 $\Gamma^{(i)}$ と領域 $\Gamma^{(j)}$ に関する係数行列の要素，$\boldsymbol{f}_{I}^{(i)}$ は領域 i の内部領域に関する外力ベクトルの要素，$\boldsymbol{f}_{\Gamma}^{(i)}$ は領域 Γ に関する外力ベクトルの要素，$\boldsymbol{x}_{I}^{(i)}$ は領域 i の内部領域に関する解ベクトルの係数，$\boldsymbol{x}_{\Gamma}^{(i)}$ は領域 Γ に関する解ベクトルの要素である．

以上の定義から，式 (2.18) を分割領域の行列形式でそれぞれ書き直すと，式 (2.21) を得る．

$$\boldsymbol{A} = \bigoplus_{i}^{\mathrm{ND}} \boldsymbol{A}^{(i)} = \bigoplus_{i}^{\mathrm{ND}} \begin{pmatrix} \boldsymbol{A}_{II}^{(i)} & \boldsymbol{A}_{\Gamma I}^{(i)} & \boldsymbol{0} \\ \boldsymbol{A}_{I\Gamma}^{(i)} & \boldsymbol{A}_{\Gamma\Gamma}^{(i)} & \boldsymbol{A}_{\mathrm{HALO}}^{(i)} \end{pmatrix} \tag{2.21}$$

ここで，$\boldsymbol{A}^{(i)}$ は領域 i に関する係数行列の要素，$\boldsymbol{A}_{\mathrm{HALO}}^{(i)}$ は領域 i とその隣接領域に関する要素であり，領域 i の隣接領域の添字集合 $\Lambda_j^{(i)}$ $(1 \leq j \leq \mathrm{NNEIB}^{(i)})$ を用いて，式 (2.22) で定

義される．NNEIB$^{(i)}$ は，領域 i に隣接する領域数である．

$$\boldsymbol{A}_{\mathrm{HALO}}^{(i)} = \left[\boldsymbol{A}_{\Gamma\Gamma}^{(i,\Lambda_1^{(i)})}, \boldsymbol{A}_{\Gamma\Gamma}^{(i,\Lambda_2^{(i)})}, \ldots, \boldsymbol{A}_{\Gamma\Gamma}^{(i,\Lambda_{\mathrm{NNEIB}^{(i)}}^{(i)})}\right] \tag{2.22}$$

non-overlapping 型領域分割法と同様に，アセンブリ作用素は，各領域のローカル節点番号をグローバル節点番号に変換する 0-1 行列 $\boldsymbol{N}$ を用いて式 (2.23) のように書き換えることができる．

$$\boldsymbol{A} = \bigoplus_{i}^{\mathrm{ND}} \boldsymbol{A}^{(i)} = \sum_{i=1}^{\mathrm{ND}} \boldsymbol{N}^{(i)} \boldsymbol{A}^{(i)} \boldsymbol{N}^{(i)t} \tag{2.23}$$

2.4 Schur Complement System Method

連立一次方程式 (2.5) の求解を考える最初の手順として，式 (2.6) における内部自由度 $\boldsymbol{A}_{II}^{(i)}$ の消去（静的縮約）を考える．静的縮約を行うために，式 (2.24)，式 (2.25) に示す方程式の変形を考える．

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{y} \tag{2.24}$$

$$\begin{bmatrix} \boldsymbol{A}_{11} & \boldsymbol{A}_{21} \\ \boldsymbol{A}_{12} & \boldsymbol{A}_{22} \end{bmatrix} \begin{Bmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \end{Bmatrix} = \begin{Bmatrix} \boldsymbol{y}_1 \\ \boldsymbol{y}_2 \end{Bmatrix} \tag{2.25}$$

式 (2.25) の関係を書き直して，式 (2.25) を得る．

$$\begin{cases} \boldsymbol{A}_{11}\boldsymbol{x}_1 + \boldsymbol{A}_{21}\boldsymbol{x}_2 = \boldsymbol{y}_1 \\ \boldsymbol{A}_{12}\boldsymbol{x}_1 + \boldsymbol{A}_{22}\boldsymbol{x}_2 = \boldsymbol{y}_2 \end{cases} \tag{2.26}$$

$\boldsymbol{x}_1$ を静的縮約するために，第 1 式から式 (2.27) を得る．

$$\boldsymbol{x}_1 = -\boldsymbol{A}_{11}^{-1}\boldsymbol{A}_{21}\boldsymbol{x}_2 + \boldsymbol{A}_{11}^{-1}\boldsymbol{y}_1 \tag{2.27}$$

式 (2.27) を式 (2.26) の第 2 式に代入して，式 (2.28) を得る．

$$\begin{aligned} -\boldsymbol{A}_{12}\boldsymbol{A}_{11}^{-1}\boldsymbol{A}_{21}\boldsymbol{x}_2 + \boldsymbol{A}_{12}\boldsymbol{A}_{11}^{-1}\boldsymbol{y}_1 + \boldsymbol{A}_{22}\boldsymbol{x}_2 &= \boldsymbol{y}_2 \\ (\boldsymbol{A}_{22} - \boldsymbol{A}_{12}\boldsymbol{A}_{11}^{-1}\boldsymbol{A}_{21})\boldsymbol{x}_2 &= \boldsymbol{y}_2 - \boldsymbol{A}_{21}\boldsymbol{A}_{11}^{-1}\boldsymbol{y}_1 \end{aligned} \tag{2.28}$$

このとき，Schur 補元 $\boldsymbol{S}$ は，式 (2.29) で与えられる．

$$\boldsymbol{S} = \boldsymbol{A}_{22} - \boldsymbol{A}_{12}\boldsymbol{A}_{11}^{-1}\boldsymbol{A}_{21} \tag{2.29}$$

式 (2.29) で式 (2.28) を書き換えて，式 (2.30) を得る．

$$\boldsymbol{S}\boldsymbol{x}_2 = \boldsymbol{y}_2 - \boldsymbol{A}_{21}\boldsymbol{A}_{11}^{-1}\boldsymbol{y}_1 \tag{2.30}$$

式 (2.30) から解ベクトル $\boldsymbol{x}_2$ が得られれば，式 (2.27) から解ベクトル $\boldsymbol{x}_1$ が得られ，解くべき方程式 (2.24) の全ての解ベクトルが得られることがわかる．non-overlapping 型領域分割では，式 (2.28) をサーフェース自由度問題と呼ぶ場合がある．

2.5 Non-overlapping 型領域分割の分割領域の静的縮約

non-overlapping 型領域分割の静的縮約を，式 (2.31) から考える（式 (2.6) の再掲）.

$$\boldsymbol{A} = \begin{pmatrix} \boldsymbol{A}_{II}^{(1)} & 0 & \boldsymbol{A}_{\Gamma I}^{(1)} \\ 0 & \boldsymbol{A}_{II}^{(2)} & \boldsymbol{A}_{\Gamma I}^{(2)} \\ \boldsymbol{A}_{I\Gamma}^{(1)} & \boldsymbol{A}_{I\Gamma}^{(2)} & \boldsymbol{A}_{\Gamma\Gamma} \end{pmatrix} \tag{2.31}$$

式 (2.31) と式 (2.25) を以下のように対応づけると，Schur 補元 $\boldsymbol{S}$ は式 (2.36) で表される.

$$\boldsymbol{A}_{11} = \begin{bmatrix} \boldsymbol{A}_{II}^{(1)} & 0 \\ 0 & \boldsymbol{A}_{II}^{(2)} \end{bmatrix} \tag{2.32}$$

$$\boldsymbol{A}_{12} = \begin{bmatrix} \boldsymbol{A}_{I\Gamma}^{(1)} & \boldsymbol{A}_{I\Gamma}^{(2)} \end{bmatrix} \tag{2.33}$$

$$\boldsymbol{A}_{21} = \begin{bmatrix} \boldsymbol{A}_{\Gamma I}^{(1)} \\ \boldsymbol{A}_{\Gamma I}^{(2)} \end{bmatrix} \tag{2.34}$$

$$\boldsymbol{A}_{22} = \begin{bmatrix} \boldsymbol{A}_{\Gamma\Gamma} \end{bmatrix} \tag{2.35}$$

$$\begin{aligned} \boldsymbol{S} &= \boldsymbol{A}_{22} - \boldsymbol{A}_{12}\boldsymbol{A}_{11}^{-1}\boldsymbol{A}_{21} \\ &= \begin{bmatrix} \boldsymbol{A}_{\Gamma\Gamma} \end{bmatrix} - \begin{bmatrix} \boldsymbol{A}_{I\Gamma}^{(1)} & \boldsymbol{A}_{I\Gamma}^{(2)} \end{bmatrix} \begin{bmatrix} \boldsymbol{A}_{II}^{(1)-1} & 0 \\ 0 & \boldsymbol{A}_{II}^{(2)-1} \end{bmatrix} \begin{bmatrix} \boldsymbol{A}_{\Gamma I}^{(1)} \\ \boldsymbol{A}_{\Gamma I}^{(2)} \end{bmatrix} \\ &= \boldsymbol{A}_{\Gamma\Gamma}^{(1)} + \boldsymbol{A}_{\Gamma\Gamma}^{(1)} - \boldsymbol{A}_{I\Gamma}^{(1)}\boldsymbol{A}_{II}^{(1)-1}\boldsymbol{A}_{\Gamma I}^{(1)} - \boldsymbol{A}_{I\Gamma}^{(2)}\boldsymbol{A}_{II}^{(2)-1}\boldsymbol{A}_{\Gamma I}^{(2)} \end{aligned} \tag{2.36}$$

これを，non-overlapping 型領域分割による分割領域 i の係数行列 $\boldsymbol{A}^{(i)}$ で書き直す.

$$\boldsymbol{A}^{(i)} = \begin{pmatrix} \boldsymbol{A}_{II}^{(i)} & \boldsymbol{A}_{\Gamma I}^{(i)} \\ \boldsymbol{A}_{I\Gamma}^{(i)} & \boldsymbol{A}_{\Gamma\Gamma}^{(i)} \end{pmatrix} \tag{2.37}$$

式 (2.37) と式 (2.25) を以下のように対応づけると，local Schur 補元 $\boldsymbol{S}^{(i)}$ は，式 (41) で表される.

$$\boldsymbol{A}_{11} = \boldsymbol{A}_{II}^{(i)} \tag{2.38}$$

$$\boldsymbol{A}_{12} = \boldsymbol{A}_{I\Gamma}^{(i)} \tag{2.39}$$

$$\boldsymbol{A}_{21} = \boldsymbol{A}_{\Gamma I}^{(i)} \tag{2.40}$$

$$\boldsymbol{A}_{22} = \boldsymbol{A}_{\Gamma\Gamma}^{(i)} \tag{2.41}$$

$$\begin{aligned}\boldsymbol{S}^{(i)} &= \boldsymbol{A}_{22} - \boldsymbol{A}_{12}\boldsymbol{A}_{11}^{-1}\boldsymbol{A}_{21} \\ &= \boldsymbol{A}_{\Gamma\Gamma}^{(i)} - \boldsymbol{A}_{I\Gamma}^{(i)}\boldsymbol{A}_{II}^{(i)-1}\boldsymbol{A}_{\Gamma I}^{(i)}\end{aligned} \tag{2.42}$$

従って，式 (2.36) と式 (2.42) により，式 (2.43) の関係を得る．

$$\boldsymbol{S} = \bigoplus_{i}^{\mathrm{ND}} \boldsymbol{S}^{(i)} \tag{2.43}$$

式 (2.43) は，分散メモリ環境において non-overlapping 型領域分割のサーフェース自由度問題を解く場合には，領域間通信を行って local Schur 補元 $\boldsymbol{S}^{(i)}$ を全て足し合わせる必要があることを意味している．通信が必要なデータ量は領域分割数とともに増加するため，並列計算性能が大きく劣化する可能性がある．

この解決のために，各領域で local Schur 補元 $\boldsymbol{S}^{(i)}$ の逆行列を計算して，反復法前処理として適用する方法がある．この手法は，Neumann-Neumann 前処理と呼ばれる．一方，local Schur 補元 $\boldsymbol{S}^{(i)}$ は一般に線形従属で逆行列を持たないので，擬似逆行列を計算する必要があり，前処理性能はそれほど高くない．

この問題を解決するために，local Schur 補元 $\boldsymbol{S}^{(i)}$ の null space として剛体モードを選定し疑似逆行列を得る手法が，BDD (balancing domain decomposition) 前処理である．

2.6 Overlapping 型領域分割の分割領域の静的縮約

overlapping 型領域分割の静的縮約を，式 (2.44) から考える（式 (2.18) の変形）．

$$\boldsymbol{A} = \begin{pmatrix} \boldsymbol{A}_{II}^{(1)} & 0 & \boldsymbol{A}_{\Gamma I}^{(1)} & 0 \\ 0 & \boldsymbol{A}_{II}^{(2)} & 0 & \boldsymbol{A}_{\Gamma I}^{(2)} \\ \boldsymbol{A}_{I\Gamma}^{(1)} & 0 & \boldsymbol{A}_{\Gamma\Gamma}^{(1)} & \boldsymbol{A}_{\Gamma\Gamma}^{(1,2)} \\ 0 & \boldsymbol{A}_{I\Gamma}^{(2)} & \boldsymbol{A}_{\Gamma\Gamma}^{(2,1)} & \boldsymbol{A}_{\Gamma\Gamma}^{(2)} \end{pmatrix} \tag{2.44}$$

式 (2.44) と式 (2.25) を以下のように対応づけると，Schur 補元 $\boldsymbol{S}$ は式 (2.49) で表される．

$$\boldsymbol{A}_{11} = \begin{bmatrix} \boldsymbol{A}_{II}^{(1)} & 0 \\ 0 & \boldsymbol{A}_{II}^{(2)} \end{bmatrix} \tag{2.45}$$

$$\boldsymbol{A}_{12} = \begin{bmatrix} \boldsymbol{A}_{I\Gamma}^{(1)} & 0 \\ 0 & \boldsymbol{A}_{I\Gamma}^{(2)} \end{bmatrix} \tag{2.46}$$

$$\boldsymbol{A}_{21} = \begin{bmatrix} \boldsymbol{A}_{\Gamma I}^{(1)} & 0 \\ 0 & \boldsymbol{A}_{\Gamma I}^{(2)} \end{bmatrix} \tag{2.47}$$

$$\boldsymbol{A}_{22} = \begin{bmatrix} \boldsymbol{A}_{\Gamma\Gamma}^{(1)} & \boldsymbol{A}_{\Gamma\Gamma}^{(1,2)} \\ \boldsymbol{A}_{\Gamma\Gamma}^{(2,1)} & \boldsymbol{A}_{\Gamma\Gamma}^{(2)} \end{bmatrix} \tag{2.48}$$

$$
\begin{aligned}
\boldsymbol{S} &= \boldsymbol{A}_{22} - \boldsymbol{A}_{12}\boldsymbol{A}_{11}^{-1}\boldsymbol{A}_{21} \\
&= \begin{bmatrix} \boldsymbol{A}_{\Gamma\Gamma}^{(1)} & \boldsymbol{A}_{\Gamma\Gamma}^{(1,2)} \\ \boldsymbol{A}_{\Gamma\Gamma}^{(2,1)} & \boldsymbol{A}_{\Gamma\Gamma}^{(2)} \end{bmatrix} - \begin{bmatrix} \boldsymbol{A}_{I\Gamma}^{(1)} & 0 \\ 0 & \boldsymbol{A}_{I\Gamma}^{(2)} \end{bmatrix} \begin{bmatrix} \boldsymbol{A}_{II}^{(1)-1} & 0 \\ 0 & \boldsymbol{A}_{II}^{(2)-1} \end{bmatrix} \begin{bmatrix} \boldsymbol{A}_{\Gamma I}^{(1)} & 0 \\ 0 & \boldsymbol{A}_{\Gamma I}^{(2)} \end{bmatrix} \\
&= \begin{bmatrix} \boldsymbol{A}_{\Gamma\Gamma}^{(1)} - \boldsymbol{A}_{I\Gamma}^{(1)}\boldsymbol{A}_{II}^{(1)-1}\boldsymbol{A}_{\Gamma I}^{(1)} & \boldsymbol{A}_{\Gamma\Gamma}^{(1,2)} \\ \boldsymbol{A}_{\Gamma\Gamma}^{(2,1)} & \boldsymbol{A}_{\Gamma\Gamma}^{(2)} - \boldsymbol{A}_{I\Gamma}^{(2)}\boldsymbol{A}_{II}^{(2)-1}\boldsymbol{A}_{\Gamma I}^{(2)} \end{bmatrix}
\end{aligned} \tag{2.49}
$$

これを，overlapping 型領域分割による分割領域 i の係数行列 $\boldsymbol{A}^{(i)}$ で書き直す．

$$
\boldsymbol{A}^{(i)} = \begin{pmatrix} \boldsymbol{A}_{II}^{(i)} & \boldsymbol{A}_{\Gamma I}^{(i)} & 0 \\ \boldsymbol{A}_{I\Gamma}^{(i)} & \boldsymbol{A}_{\Gamma\Gamma}^{(i)} & \boldsymbol{A}_{\mathrm{HALO}}^{(i)} \end{pmatrix} \tag{2.50}
$$

式 (2.50) と式 (2.25) を以下のように対応づけると，Local Schur 補元 $\boldsymbol{S}^{(i)}$ は，式 (2.55) で表される．

$$
\boldsymbol{A}_{11} = \boldsymbol{A}_{II}^{(i)} \tag{2.51}
$$

$$
\boldsymbol{A}_{12} = \boldsymbol{A}_{I\Gamma}^{(i)} \tag{2.52}
$$

$$
\boldsymbol{A}_{21} = \boldsymbol{A}_{\Gamma I}^{(i)} \tag{2.53}
$$

$$
\boldsymbol{A}_{22} = \boldsymbol{A}_{\Gamma\Gamma}^{(i)} \tag{2.54}
$$

$$
\begin{aligned}
\boldsymbol{S}^{(i)} &= \boldsymbol{A}_{22} - \boldsymbol{A}_{12}\boldsymbol{A}_{11}^{-1}\boldsymbol{A}_{21} \\
&= \boldsymbol{A}_{\Gamma\Gamma}^{(i)} - \boldsymbol{A}_{I\Gamma}^{(i)}\boldsymbol{A}_{II}^{(i)-1}\boldsymbol{A}_{\Gamma I}^{(i)}
\end{aligned} \tag{2.55}
$$

従って，式 (2.49) と式 (2.55) により，式 (2.56) の関係を得る．

$$
\boldsymbol{S} = \bigoplus_{i}^{\mathrm{ND}} \begin{bmatrix} \boldsymbol{S}^{(i)} & \boldsymbol{A}_{\mathrm{HALO}}^{(i)} \end{bmatrix} \tag{2.56}
$$

例えば，領域分割数が 2 の場合，$\boldsymbol{S}$ は，

$$
\boldsymbol{S} = \begin{bmatrix} \boldsymbol{S}^{(1)} & \boldsymbol{A}_{\mathrm{HALO}}^{(1)} \\ \boldsymbol{A}_{\mathrm{HALO}}^{(2)} & \boldsymbol{S}^{(2)} \end{bmatrix} \tag{2.57}
$$

と表せられる．

式 (2.56) は，分散メモリ環境において overlapping 型領域分割のサーフェース自由度問題を解く場合に，領域間通信を行うことなく Schur 補元 $\boldsymbol{S}$ が構築できることを意味している．一方，overlapping 型領域分割の Schur 補元 $\boldsymbol{S}$ の自由度は，non-overlapping 型領域分割の 2 倍になることに注意する．また，$\boldsymbol{A}_{\mathrm{HALO}}^{(i)}$ は，疎行列の非零パターンを持っているため，この部分の特徴を適切に捉える必要がある．

これらの考察から，領域間通信を行うことなく Schur 補元 $\boldsymbol{S}$ が構築できる overlapping 型領域分割が，分散メモリ環境における並列計算に適していると考えられる．従って，本研究では，直接法・反復法の統一的計算する手法の基盤として，overlapping 型領域分割を採用する．

2.7 直接法

直接法は，Gauss の消去法 (LU 分解と前進後退代入) に基づいており，有限回の演算で解を得る堅固な手法である．直接法の特徴は，LU 分解の過程において元々の行列で要素の値が零である部分に零以外の値が出現する，フィルインが発生することである．

2.8 グラフ

グラフ $\boldsymbol{G}$ とは，節点集合 $\boldsymbol{V} = \{1, 2, \cdots, n\}$ と，辺集合 $\boldsymbol{E} = \{(i, j) \mid i, j \in \boldsymbol{V}\}$ から決まる構造 $\boldsymbol{G} = (\boldsymbol{V}, \boldsymbol{E})$ である．ここで n は節点数である．有向グラフは辺 (i, j) と辺 (j, i) を異なる辺として取扱い，無向グラフは辺 (i, j) と辺 (j, i) を同一として取り扱う．節点 i の隣接点 j は，隣接集合 $\mathrm{Adj}(i) = \{j \mid (i, j) \in \boldsymbol{E}\}$ で表される．

2.9 係数行列とグラフ

係数行列 $\boldsymbol{A}$ から得られる隣接リスト $\mathcal{A}_{i*}$，$\mathcal{A}_{*j}$ は式 (2)，式 (3) で表される．

$$\mathcal{A}_{i*} = \{j \mid a_{ij} \neq 0,\ i \neq j\} = \mathrm{Adj}(i) = \{j \mid (i, j) \in \boldsymbol{E}\} \tag{2.58}$$
$$\mathcal{A}_{*j} = \{i \mid a_{ij} \neq 0,\ i \neq j\} = \mathrm{Adj}(j) = \{i \mid (i, j) \in \boldsymbol{E}\} \tag{2.59}$$

隣接リスト $\mathcal{A}_{i*}$，$\mathcal{A}_{*j}$ に対し，係数行列 $\boldsymbol{A}$ の対角成分 a_{ii} に対応する self-edge (i, i) を考えると，係数行列 $\boldsymbol{A}$ 成分の非零要素と一致する．このとき，係数行列 $\boldsymbol{A}$ から得られるグラフ構造を $\boldsymbol{G_A}$ で表す．

2.10 フィルインとシンボリック分解

正定値対称行列 $\boldsymbol{A}$ は，$\boldsymbol{A} = \boldsymbol{L}\boldsymbol{L}^t$ の形に Cholesky 分解することができる．ここで，$\boldsymbol{L}$ は下三角行列である．このとき，Cholesky 分解をすることで現れる，$\boldsymbol{G_A}$ に含まれない辺集合をフィルイン（fill-in）と呼び，$\boldsymbol{G_F}$ と表す．フィルインを考慮したグラフ構造 $\boldsymbol{G_A^+}$ は，式 (4) で表される．

$$\boldsymbol{G_A^+} = \boldsymbol{G_{L+L^t}} = \boldsymbol{G_{A+F}} \tag{2.60}$$

ここで，$\boldsymbol{G_L}$ は下三角行列 $\boldsymbol{L}$ の非零要素を示すグラフである．

Cholesky 分解の効率的な計算のために，グラフ構造 $\boldsymbol{G_A^+}$ を予め決定しておく必要がある．グラフ構造 $\boldsymbol{G_A}$ からフィルイン $\boldsymbol{G_F}$ を求め，$\boldsymbol{G_A^+}$ を決定することを，シンボリック分解（symbolic factorization）と呼ぶ．

```
1: for i = 1, 2, 3, ··· , n do
2:    for j ∈ A_i do
3:       for k ∈ A_i ∧ j < k do
4:          A_j ⇐ A_j ∪ k
5:       end for
6:    end for
7: end for
```

Fig. 2.3 Fill-in detection based on Theorem 1 and Theorem 2

2.11 フィルインの決定

フィルイン $\boldsymbol{G_F}$ の決定のために，グラフ構造 $\boldsymbol{G_A}$ を用いる．ここで，$\mathcal{A}_i$ を $\boldsymbol{A}$ の第 i 列の隣接リスト，$\mathcal{L}_i$ を $\boldsymbol{L}$ の第 i 列の隣接リストとする．このとき，$\mathcal{A}_1$ と $\mathcal{L}_1$ は一致する．

最も簡単なフィルインの決定方法は，定理 1，定理 2 を用いて，$\mathcal{L}_1$ から順に，全ての非零要素を確認していくものである．図 2.3 にフィルイン決定のアルゴリズムを示す．定理 1，定理 2 に基づく方法は，演算量 $\mathcal{O}(m^2)$ と見積もられる．ここで，m は隣接リスト $\mathcal{A}$ の要素数である．

定理 1 Cholesky 分解 $\boldsymbol{A} = \boldsymbol{L}\boldsymbol{L}^t$ において，数値の桁落ちを無視すれば，$a_{ij} \neq 0 \Rightarrow l_{ij} \neq 0$ が成り立つ．

定理 2 Cholesky 分解 $\boldsymbol{A} = \boldsymbol{L}\boldsymbol{L}^t$ において，数値の桁落ちを無視すれば，$i < j < k \wedge l_{ji} \neq 0 \wedge l_{ki} \neq 0 \Rightarrow l_{kj} \neq 0$ が成り立つ．

図 2.3 に示したフィルイン決定のアルゴリズムは，行列サイズと隣接リストの要素数が大きくなるに従って，多くの計算時間を要する．この解決のため，図 2.4 に，定理 3 に基づいたフィルイン決定のアルゴリズムを示す．定理 3 に基づいた方法は，演算量 $\mathcal{O}(m)$ と見積もられる．

定理 3 Cholesky 分解 $\boldsymbol{A} = \boldsymbol{L}\boldsymbol{L}^t$ において，節点 j が節点 i の親であれば，$\mathcal{L}_i \setminus j \in \mathcal{L}_j$ が成り立つ．ここで，節点 i の親 j は $\min\{j \mid j > i,\ l_{ij} \neq 0\}$ である．

2.11.1 リオーダリング

直接法によって疎行列を求解するときは，フィルインの数が元の行列の非零要素数に比べて非常に大きくなり，計算量とメモリ使用量が増大する．フィルインの増大を抑制させるために，適当な置換行列を用いて行と列を置換するリオーダリングを行う必要がある．

```
1: for i = 1, 2, 3, ··· , n do
2:     j = min {j | j > i, j ∈ L_i}
3:     A_j ⇐ A_j ∪ A_i \ j
4: end for
```

Fig. 2.4 Fill-in detection based on Theorem 3

```
1: for k = 1, 2, 3, ..., n do
2:     for i = k + 1, k + 2, ..., n do
3:         a_{i,k} = a_{i,k}/a_{kk}
4:     end for
5:     for j = k + 1, k + 2, ..., n do
6:         for i = k + 1, k + 2, ..., n do
7:             a_{i,j} = a_{i,j} − a_{i,k}a_{k,j}
8:         end for
9:     end for
10: end for
```

Fig. 2.5 Outer product (right looking) factorization

リオーダリングの例としては，minimum degree 法，Cuthill-Mackee 法と nested dissection 法が挙げられる [1, 2, 3, 4]．minimum degree 法と Cuthill-Mackee 法は，LU 分解の各ステップで生じるフィルインを評価値として，フィルインが少なくなるように置換行列を決定する幅優先探索アルゴリズムの一種である．nested dissection 法は，ひとつの領域を分割するセパレータ領域最小化するように，二つの領域に分割することを再帰的に繰り返し，フィルインを削減する手法である．

2.11.2 LU 分解

LU 分解は，正則な正方行列 $\boldsymbol{A}$ を，下三角行列 $\boldsymbol{L}$ と上三角行列 $\boldsymbol{U}$ を用いて，$\boldsymbol{A} = \boldsymbol{LU}$ と行列分解する手法である．LU 分解には，数学的に等価である様々な分解手順があり，ここでは外積形式分解，内積形式分解，Crout 法，frontal 法について示す．

図 2.5 に，外積形式分解（outer product factorization）を示す．外積形式分解は，ピボット i を用いた i 番目の LU 分解において，残った行列成分を i 行と i 列の外積を用いて更新する手法である．ピボット i に対して，i よりも大きい行列成分を参照するため，right looking アルゴリズムと呼ばれる．

図 2.6 に，内積形式分解（inner product factorization）を示す．内積形式分解は，ピボット i を用いた i 番目の LU 分解において，i 列自身を分解済みの成分の内積を用いて更新する手法

```
1: for k = 1, 2, 3, ..., n do
2:   for j = 1, 2, 3, ..., k − 1 do
3:     for i = j + 1, j + 2, ..., n do
4:       a_{i,k} = a_{i,k} − a_{i,j} a_{j,k}
5:     end for
6:   end for
7:   a_{k,k} = a_{k,k}^{−1}
8:   for i = k + 1, k + 2, ..., n do
9:     a_{i,k} = a_{i,k} a_{k,k}
10:  end for
11: end for
```

Fig. 2.6 Inner product (left looking) factorization

である．ピボット i に対して，i よりも小さい列成分を参照するため，left looking アルゴリズムと呼ばれる．

図 2.7 に，Crout 法を示す．Crout 法は，ピボット i を用いた i 番目の LU 分解において，i 行と i 列自身を分解済みの成分の内積を用いて更新する手法である．行列成分の更新順序の特徴から，共有メモリ型並列計算機に適していることが特徴である．

図 2.8 に，frontal 法を示す．frontal 法は，フロンタル行列 $\boldsymbol{F}$ とアップデート行列 $\boldsymbol{U}$ を用いて LU 分解する，外積形式の直接法である．計算手順を理解するために，正定値対称行列 $\boldsymbol{A}$ を Frontal 法によって Cholesky 分解 $\boldsymbol{A} = \boldsymbol{L}\boldsymbol{L}^{\boldsymbol{t}}$ を計算することを考える．ここで，$\boldsymbol{A}$ は正定値対称行列，$a_{i,j}$ は $\boldsymbol{A}$ の第 (i, j) 番目の行列要素，$\boldsymbol{F}$ はフロンタル行列，$\boldsymbol{U}$ はアップデート行列，$\boldsymbol{L}$ は下三角行列，$l_{i,j}$ は $\boldsymbol{L}$ の第 (i, j) 番目の行列要素である．

特に図 2.8 は，行列 $\boldsymbol{A}$ が密行列の場合を示している．ここで，$\boldsymbol{F}^k$ は，ある行番号 $k(1 \leq k \leq n)$ に対するフロンタル行列で，$(n - k + 1) \times (n - k + 1)$ の大きさをもつ．

```
1: a_{11} = 1/a_{11}
2: for i = 2, 3, ..., n do
3:     a_{1,i} = a_{1,i} a_{1,1}
4: end for
5: for k = 1, 2, 3, ..., n do
6:     for j = 1, 2, 3, ..., k do
7:         for i = k, k + 1, ..., n do
8:             a_{i,k} = a_{i,k} − a_{i,j} a_{j,k}
9:         end for
10:    end for
11:    a_{k,k} = a_{k,k}^{−1}
12:    for j = 1, 2, ..., k do
13:        for i = k + 1, k + 2, ..., n do
14:            a_{k,i} = a_{k,j} − a_{k,j} a_{j,i}
15:        end for
16:    end for
17:    for j = k + 1, k + 2, ..., n do
18:        a_{k,j} = a_{k,j} a_{k,k}
19:    end for
20: end for
```

Fig. 2.7 Crout factorization

1: $\boldsymbol{U}^0 = \boldsymbol{0}$
2: **for** $k = 1, 2, 3, \ldots, n$ **do**
3: $\boldsymbol{F}^k = \begin{pmatrix} a_{k,k} & a_{k,k+1} & a_{k,k+2} & \cdots & a_{k,n} \\ a_{k+1,k} & 0 & 0 & \cdots & 0 \\ a_{k+2,k} & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & & \vdots \\ a_{n,k} & 0 & 0 & \cdots & 0 \end{pmatrix}$
4: $\boldsymbol{F}^k = \boldsymbol{F}^k + \boldsymbol{U}^{k-1}$
5: $l_{k,k} = 1/\sqrt{f^k_{1,1}}$
6: **for** $i = 1, 2, 3, \ldots, n-k$ **do**
7: $l_{k+i,k} = f^k_{i,k} l_{k,k}$
8: **end for**
9: **for** $j = 1, 2, 3, \ldots, n-k$ **do**
10: **for** $i = j, j+1, j+2, \ldots, n-k$ **do**
11: $u^k_{i,j} = f^k_{i,j} - l_{k+i,k} l_{k+j,k}$
12: **end for**
13: **end for**
14: **end for**

Fig. 2.8 Frontal method

2.12 直接法における並列計算

これまでに述べた LU 分解のアルゴリズムは，逐次的な計算に基づくため，並列計算を行うには別途，並列計算のための計算手順を考慮する必要がある．ここでは特に，分散メモリ型並列計算機における LU 分解の手法として，nested dissection method，multi-frontal method，SPIKE msethod について述べる．

2.12.1 Nested Dissection Method

nested dissection 法は，ある領域を 2 分割する最小なセパレータ領域を再帰的に決定することで提案された，並列直接法である．式 (2.61) のような連立一次方程式を考える．

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{f} \tag{2.61}$$

ここで，セパレータ領域 Γ で 2 分割した領域を考え，その係数行列を行列形式で書き換えると，式 (2.62) で表される．

$$\boldsymbol{A} = \begin{pmatrix} \boldsymbol{A}_{II}^{(1)} & 0 & \boldsymbol{A}_{I\Gamma}^{(1)t} \\ 0 & \boldsymbol{A}_{II}^{(2)} & \boldsymbol{A}_{I\Gamma}^{(2)t} \\ \boldsymbol{A}_{I\Gamma}^{(1)} & \boldsymbol{A}_{I\Gamma}^{(2)} & \boldsymbol{A}_{\Gamma\Gamma} \end{pmatrix} \tag{2.62}$$

式 (2.62) より，内部領域 $\boldsymbol{A}_{II}^{(1)}$ と $\boldsymbol{A}_{II}^{(2)}$ は，それぞれ独立に LU 分解可能であり，その LU 分解による行列成分の更新は，セパレータ領域 $\boldsymbol{A}_{\Gamma\Gamma}$ に集約されることがわかる．このような領域分割を再帰的に内部領域 $\boldsymbol{A}_{II}^{(i)}$ に行うことで，LU 分解の並列計算が可能となる．

2.12.2 Multi-frontal Method

multi-frontal 法は，frontal 行列の演算順序の依存性を考慮することで提案された，並列直接法である．図 2.9 に，multi-frontal 法のアルゴリズムを示す．ここで，$\boldsymbol{A}$ は正定値対称行列，$a_{i,j}$ は $\boldsymbol{A}$ の第 (i,j) 番目の行列要素，$\boldsymbol{F}$ はフロンタル行列，$\boldsymbol{U}$ はアップデート行列，$\boldsymbol{L}$ は下三角行列，$l_{i,j}$ は $\boldsymbol{L}$ の第 (i,j) 番目の行列要素，$\mathcal{A}_i$ は係数行列 $\boldsymbol{A}$ の第 i 行の非零要素，q_n は $\mathcal{A}_i$ の非零要素数，$\mathcal{L}_i$ は下三角行列 $\boldsymbol{L}$ の第 i 行の非零要素，$\mathrm{index}(i, \mathcal{L}_k)$ は $\mathcal{L}_k$ の第 i 番目の非零要素の行番号であり，$\bigoplus$ は拡張和を示す．

1: $\boldsymbol{U}^0 = \boldsymbol{0}$
2: **for** $k = 1, 2, 3, \ldots, n$ **do**
3: $\boldsymbol{F}^k = \begin{pmatrix} a_{k,k} & a_{k,q_1} & a_{k,q_2} & \cdots & a_{k,q_n} \\ a_{q_1,k} & 0 & 0 & \cdots & 0 \\ a_{q_2,k} & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & & \vdots \\ a_{q_n,k} & 0 & 0 & \cdots & 0 \end{pmatrix}$
$(q_1, q_2, \ldots, q_n \in \mathcal{A}_k)$
4: $\boldsymbol{F}^k = \boldsymbol{F}^k \bigoplus \boldsymbol{U}^{k-1}$
5: $l_{k,k} = 1/\sqrt{f^k_{1,1}}$
6: **for** $i \in \mathcal{L}_k$ **do**
7: $p = \mathrm{index}(i, \mathcal{L}_k)$
8: $l_{i,k} = f^k_{p,k} l_{k,k}$
9: **end for**
10: **for** $j \in \mathcal{L}_k$ **do**
11: **for** $i \in \mathcal{L}_k$ **do**
12: $p = \mathrm{index}(i, \mathcal{L}_k)$
13: $q = \mathrm{index}(j, \mathcal{L}_k)$
14: $u^k_{p,q} = f^k_{p,q} - l_{i,k} l_{j,k}$
15: **end for**
16: **end for**
17: **end for**

Fig. 2.9 Multi-frontal method

2.12.3 SPIKE Method

SPIKE 法は，帯行列の係数行列をもつ方程式に対し提案された並列直接法であり，係数行列として帯行列をもつ連立一次方程式 (2.63) を解く手法である．ここで，$\boldsymbol{A}$ は $b \ll n$ であるバンド幅 b をもつ帯行列，$\boldsymbol{x}$ は解ベクトル，$\boldsymbol{f}$ は右辺ベクトルである．

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{f} \tag{2.63}$$

式 (2.63) に対し，対角ブロック行列 $\boldsymbol{D} = \mathrm{diag}\{\boldsymbol{A}_1 \boldsymbol{A}_2 \dots \boldsymbol{A}_p\}$ を定義して，対角ブロック行列の逆行列 $\boldsymbol{D}^{-1}$ を左から乗ずることで，スパイク方程式 (2.64) を得る．

$$\boldsymbol{S}\boldsymbol{x} = \boldsymbol{g} \tag{2.64}$$

その後，式 (2.64) に対し，方程式の依存関係から縮退方程式 (2.65) を定義する．この縮退方程式 (2.65) を解くことで，解くべき連立一次方程式 (2.63) の解を得る手法である．

$$\hat{\boldsymbol{S}}\hat{\boldsymbol{x}} = \hat{\boldsymbol{g}} \tag{2.65}$$

SPIKE 法は，本研究の基盤となる手法であるため，第 3 章で詳細について記述する．

2.13 定常反復法

定常反復法は，連立一次方程式 $\boldsymbol{Ax} = \boldsymbol{b}$ に対し，適当な初期値 $\boldsymbol{x}^{(0)}$ を与えたとき，ある定常な操作 $\boldsymbol{M}$ によって，反復的に解を求めるものである．具体的には，係数行列を $\boldsymbol{A} = \boldsymbol{M} + \boldsymbol{N}$ と分解したとき，式 (2.66) に従って解を修正する．

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{M}^{-1}(\boldsymbol{b} - \boldsymbol{N}\boldsymbol{x}^{(k)}) \tag{2.66}$$

ここで $\boldsymbol{M}$ が $\boldsymbol{A}$ であれば，反復は 1 回で収束する．従って，$\boldsymbol{M} \simeq \boldsymbol{A}$ であるような $\boldsymbol{M}$ を選定することが重要となる．

また，あらかじめ係数行列 $\boldsymbol{A}$ は，式 (2) のように分解できるとする．ここで，$\boldsymbol{D}$ は対角行列，$\boldsymbol{L}$ は狭義下三角行列，$\boldsymbol{U}$ は狭義上三角行列である．

$$\boldsymbol{A} = \boldsymbol{L} + \boldsymbol{D} + \boldsymbol{U} \tag{2.67}$$

2.13.1 Jacobi Method

Jacobi 法では，$\boldsymbol{M} = \boldsymbol{D}$ として，解の修正を行うものである [5]．従って式 (2.66) は，式 (2.68) のように書き直される．

$$\begin{aligned} \boldsymbol{x}_{k+1} &= \boldsymbol{D}^{-1}\{\boldsymbol{b} - (\boldsymbol{L} + \boldsymbol{U})\boldsymbol{x}_k\} \\ \boldsymbol{x}_{k+1} &= -\boldsymbol{D}^{-1}(\boldsymbol{L} + \boldsymbol{U})\boldsymbol{x}_k + \boldsymbol{D}^{-1}\boldsymbol{b} \end{aligned} \tag{2.68}$$

これを成分表示すれば，式 (2.69) のように表される．

$$x_i^{(k+1)} = \frac{1}{a_{ii}} \left(b_i - \sum_{\substack{j=1 \\ j \neq i}}^{n} a_{ij} x_j^{(k)} \right) \tag{2.69}$$

2.13.2 Gauss-Sidel (GS) Method

ガウスザイデル法 (Gauss-Sidel method, GS method) では，$\boldsymbol{M} = \boldsymbol{D} + \boldsymbol{L}$ として，解の修正を行うものである．従って式 (2.66) は，式 (2.70) のように書き直される．

$$\begin{aligned} \boldsymbol{x}_{k+1} &= (\boldsymbol{D} + \boldsymbol{L})^{-1}(\boldsymbol{b} - \boldsymbol{U}\boldsymbol{x}_k) \\ \boldsymbol{x}_{k+1} &= -(\boldsymbol{D} + \boldsymbol{L})^{-1}\boldsymbol{U}\boldsymbol{x}_k + (\boldsymbol{D} + \boldsymbol{L})^{-1}\boldsymbol{b} \end{aligned} \tag{2.70}$$

これを成分表示すれば，式 (2.71) のように表される．

$$x_i^{(k+1)} = \frac{1}{a_{ii}} \left(b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(k+1)} - \sum_{j=i+1}^{n} a_{ij} x_j^{(k)} \right) \tag{2.71}$$

2.13.3 Sucsessive Over Relaxation (SOR) Method

逐次過緩和法 (sucsessive over relaxation method, SOR method) では，ある加速係数 ω を用いて，連立一次方程式 $\boldsymbol{Ax} = \boldsymbol{b}$ を連立一次方程式 $\omega\boldsymbol{Ax} = \omega\boldsymbol{b}$ に変換する [6]．ここで，$\omega\boldsymbol{A}$ を

$$\omega\boldsymbol{A} = (\boldsymbol{D} + \omega\boldsymbol{L}) + \{\omega\boldsymbol{U} - (1-\omega)\boldsymbol{D}\} \tag{2.72}$$

のように分解し，$\boldsymbol{M} = \boldsymbol{D} + \omega\boldsymbol{L}$ として，解の修正を行うものである．従って式 (2.66) は，式 (2.73) のように書き直される．

$$\boldsymbol{x}_{k+1} = (\boldsymbol{D} + \omega\boldsymbol{L})^{-1}\{\omega\boldsymbol{b} - \omega\boldsymbol{U}\boldsymbol{x}_k + (1-\omega)\boldsymbol{D}\boldsymbol{x}_k\} \tag{2.73}$$

これを成分表示すれば，式 (2.74) のように表される．

$$x_i^{(k+1)} = \frac{1}{a_{ii}}\left\{\omega b_i - \omega\sum_{j=1}^{i-1} a_{ij}x_j^{(k+1)} - \omega\sum_{j=i+1}^{n} a_{ij}x_j^{(k)} + (1-\omega)a_{ii}x_i^{(k)}\right\} \tag{2.74}$$

SOR 法は，$\omega = 1$ のとき，Gauss-Sidel 法と一致することが知られている．

2.14 非定常反復法

定常反復法は，連立一次方程式 $\boldsymbol{Ax} = \boldsymbol{b}$ に対し，適当な初期値 $\boldsymbol{x}^{(0)}$ を与えたとき，解ベクトルのみを反復的に更新して解を求める手法であった．これに対し，非定常反復法は，解の更新に拘束条件や最適化条件を加えることで，より高速に収束解を得る手法である．

非定常反復法において，解くべき問題の係数行列が正定値性対称行列である連立一次方程式 $\boldsymbol{Ax} = \boldsymbol{b}$ の場合，式 (2.75) に示す関数の最小化問題を解くと考えることもできる．

$$f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^t\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}^t\boldsymbol{x} \tag{2.75}$$

2.14.1 Steepest Descent (SD) Method

最急勾配法 (steepest descent method, SD method) は，解の修正方向を関数 $f(\boldsymbol{x})$ の勾配 $\dfrac{\partial f(\boldsymbol{x})}{\partial \boldsymbol{x}}$ として，反復させる手法である．最急勾配法を，図 2.10 に示す．ここで，$\boldsymbol{A}$ は係数行列，$\boldsymbol{x}$ は解ベクトル，$\boldsymbol{r}$ は残差ベクトル，α は残差ベクトルの係数，ε は収束判定値である．

関数 $f(\boldsymbol{x})$ の勾配 $\dfrac{\partial f(\boldsymbol{x})}{\partial \boldsymbol{x}}$ は残差ベクトル $\boldsymbol{r}_k$ と一致する．このとき，式 (2.75) から係数 α_k を考慮した残差ベクトル $\boldsymbol{r}_k$ を考えて，式 (2.76) を得る．

$$\begin{aligned} f(\boldsymbol{x}_k) &= \frac{1}{2}(\boldsymbol{x}_k + \alpha_k\boldsymbol{r}_k)^t\boldsymbol{A}(\boldsymbol{x}_k + \alpha_k\boldsymbol{r}_k) - \boldsymbol{b}^t(\boldsymbol{x}_k + \alpha_k\boldsymbol{r}_k) \\ &= f(\boldsymbol{x}_k) + \frac{1}{2}\alpha_k^2\boldsymbol{r}_k^t\boldsymbol{A}\boldsymbol{r}_k - \alpha_k\boldsymbol{r}_k^t\boldsymbol{r}_k \end{aligned} \tag{2.76}$$

1: $\boldsymbol{r}_0 = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0$
2: **for** $i = 1, 2, 3, \ldots,$ **do**
3: $\quad \alpha_i = (\boldsymbol{r}_{i-1}, \boldsymbol{r}_{i-1})/(\boldsymbol{r}_{i-1}, \boldsymbol{A}\boldsymbol{r}_{i-1})$
4: $\quad \boldsymbol{x}_i = \boldsymbol{x}_{i-1} + \alpha_i \boldsymbol{r}_{i-1}$
5: $\quad \boldsymbol{r}_i = \boldsymbol{r}_{i-1} - \alpha_i \boldsymbol{A}\boldsymbol{r}_{i-1}$
6: $\quad$ **if** $(\|\boldsymbol{r}_i\|_2/\|\boldsymbol{r}_0\|_2 \leq \varepsilon)$ **then stop**
7: **end for**

Fig. 2.10 Steepest descent method

この式を最小にする $\alpha^{(k)}$ は，

$$\frac{\partial f(\boldsymbol{x})}{\partial \alpha^{(k)}} = \alpha^{(k)} {\boldsymbol{r}^{(k)}}^t \boldsymbol{A}\boldsymbol{r}^{(k)} - {\boldsymbol{r}^{(k)}}^t \boldsymbol{r}^{(k)} = 0 \tag{2.77}$$

となればよいことがわかる．

従って，以下の関係

$$\begin{aligned} \alpha_k &= \frac{\boldsymbol{r}_k^t \boldsymbol{r}_k}{\boldsymbol{r}_k^t \boldsymbol{A}\boldsymbol{r}_k} \\ &= \frac{(\boldsymbol{r}_k, \boldsymbol{r}_k)}{(\boldsymbol{r}_k, \boldsymbol{A}\boldsymbol{r}_k)} \end{aligned} \tag{2.78}$$

から，係数 α_k を決定し，解ベクトル $\boldsymbol{x}_k$ と残差ベクトル $\boldsymbol{r}_k$ を更新すればよい．

2.14.2 Conjugate Gradient (CG) Method

共役勾配法 (conjugate gradient method, CG method) は，Krylov 部分空間による反復法として広く知られており，解の修正ベクトル $\boldsymbol{p}_k$ を残差ベクトル $\boldsymbol{r}_i$ $(1 \leq i \leq k-1)$ の一次結合として定めるものである [7]．図 2.11 に，共役勾配法を示す．図 2.12 に，前処理つき共役勾配法を示す．ここで，$\boldsymbol{A}$ は係数行列，$\boldsymbol{M}$ は前処理行列，$\boldsymbol{b}$ は右辺ベクトル，$\boldsymbol{x}$ は解ベクトル，$\boldsymbol{r}$ は残差ベクトル，$\boldsymbol{p}$ は探索方向ベクトル，α，β は探索方向ベクトルの係数，ε は収束判定値である．

共役勾配法の特徴は，1 反復あたりの演算量とメモリ使用量が少ないこと，解くべき行列の次元数 n に対して理論上 n 回反復すれば収束することである．共役勾配法の収束のしやすさは，最大固有値と最小固有値の比で表される条件数 (comdition number) を指標にすることができる．

2.14.3 Conjugate Residual (CR) Method

共役残差法 (conjugate residual method, CR method) は，ある任意の個数の残差ベクトル $\boldsymbol{r}_k$ が係数行列 $\boldsymbol{A}$ に対し各々直交する条件で，解を修正するものである．[8] 図 2.13 に，共役

$\boldsymbol{r}_0 = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0$
$\boldsymbol{p}_0 = \boldsymbol{r}_0$
for $i = 1, 2, 3, \ldots,$ **do**
  $\alpha_i = (\boldsymbol{r}_{i-1}, \boldsymbol{r}_{i-1})/(\boldsymbol{p}_{i-1}, \boldsymbol{A}\boldsymbol{p}_{i-1})$
  $\boldsymbol{x}_i = \boldsymbol{x}_{i-1} + \alpha_i \boldsymbol{p}_{i-1}$
  $\boldsymbol{r}_i = \boldsymbol{r}_{i-1} - \alpha_i \boldsymbol{A}\boldsymbol{p}_{i-1}$
  if $(\|\boldsymbol{r}_i\|_2/\|\boldsymbol{r}_0\|_2 \leq \varepsilon)$ **then stop**
  $\beta_i = (\boldsymbol{r}_i, \boldsymbol{r}_i)/(\boldsymbol{r}_{i-1}, \boldsymbol{r}_{i-1})$
  $\boldsymbol{p}_i = \boldsymbol{r}_i + \beta_i \boldsymbol{p}_{i-1}$
end for

Fig. 2.11 CG method

$\boldsymbol{r}_0 = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0$, $\boldsymbol{u}_0 = \boldsymbol{M}^{-1}\boldsymbol{r}_0$, $\boldsymbol{p}_0 = \boldsymbol{u}_0$
for $i = 1, 2, 3, \ldots,$ **do**
  $\boldsymbol{s} = \boldsymbol{A}\boldsymbol{p}_{i-1}$
  $\alpha_i = (\boldsymbol{r}_{i-1}, \boldsymbol{u}_{i-1})/(\boldsymbol{s}, \boldsymbol{p}_{i-1})$
  $\boldsymbol{x}_i = \boldsymbol{x}_{i-1} + \alpha_i \boldsymbol{p}_{i-1}$
  $\boldsymbol{r}_i = \boldsymbol{r}_{i-1} - \alpha_i \boldsymbol{s}_{i-1}$
  if $(\|\boldsymbol{r}_i\|_2/\|\boldsymbol{r}_0\|_2 \leq \varepsilon)$ **then stop**
  $\boldsymbol{u}_i = \boldsymbol{M}^{-1}\boldsymbol{r}_i$
  $\beta_i = (\boldsymbol{r}_i, \boldsymbol{u}_i)/(\boldsymbol{r}_{i-1}, \boldsymbol{u}_{i-1})$
  $\boldsymbol{p}_i = \boldsymbol{u}_i + \beta_i \boldsymbol{p}_{i-1}$
end for

Fig. 2.12 Preconditioned CG method

残差法を示す．図 2.14 に，前処理つき共役残差法を示す．ここで，$\boldsymbol{A}$ は係数行列，$\boldsymbol{M}$ は前処理行列，$\boldsymbol{b}$ は右辺ベクトル，$\boldsymbol{x}$ は解ベクトル，$\boldsymbol{r}$ は残差ベクトル，$\boldsymbol{p}$ は探索方向ベクトル，α，β は探索方向ベクトルの係数，ε は収束判定値である．

共役残差法は，共役勾配法のように，高々 n 回の反復で解を得ることができる保証はない．しかし，共役残差法反復中において，

$$\boldsymbol{p}_i = \boldsymbol{r}_i + \sum_{j=1}^{i} \beta_{i-j} \boldsymbol{p}_{i-j} \tag{2.79}$$

が計算できれば，n 回の反復で収束することが知られている．

$\boldsymbol{r}_0 = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0$
$\boldsymbol{p}_0 = \boldsymbol{r}_0$
for $i = 1, 2, 3, \ldots,$ **do**
  $\alpha_i = (\boldsymbol{r}_{i-1}, \boldsymbol{A}\boldsymbol{r}_{i-1})/(\boldsymbol{p}_{i-1}, \boldsymbol{p}_{i-1})$
  $\boldsymbol{x}_i = \boldsymbol{x}_{i-1} + \alpha_i \boldsymbol{p}_{i-1}$
  $\boldsymbol{r}_i = \boldsymbol{r}_{i-1} - \alpha_i \boldsymbol{A}\boldsymbol{p}_{i-1}$
  if $(\|\boldsymbol{r}_i\|_2/\|\boldsymbol{r}_0\|_2 \leq \varepsilon)$ **then stop**
  $\beta_i = (\boldsymbol{r}_i, \boldsymbol{A}\boldsymbol{r}_{i-1})/(\boldsymbol{r}_{i-1}, \boldsymbol{A}\boldsymbol{r}_{i-1})$
  $\boldsymbol{p}_i = \boldsymbol{r}_i + \beta_i \boldsymbol{p}_i$
end for

Fig. 2.13 CR method

$\boldsymbol{r}_0 = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0$
$\boldsymbol{p}_0 = \boldsymbol{M}^{-1}\boldsymbol{r}_0$
for $i = 1, 2, 3, \ldots,$ **do**
  $\alpha_i = (\boldsymbol{M}^{-1}\boldsymbol{r}_{i-1}, \boldsymbol{A}\boldsymbol{M}^{-1}\boldsymbol{r}_{i-1})/(\boldsymbol{M}^{-1}\boldsymbol{A}\boldsymbol{p}_{i-1}, \boldsymbol{A}\boldsymbol{p}_{i-1})$
  $\boldsymbol{x}_i = \boldsymbol{x}_{i-1} + \alpha_i \boldsymbol{p}_{i-1}$
  $\boldsymbol{r}_i = \boldsymbol{r}_{i-1} - \alpha_i \boldsymbol{A}\boldsymbol{p}_{i-1}$
  if $(\|\boldsymbol{r}_i\|_2/\|\boldsymbol{r}_0\|_2 \leq \varepsilon)$ **then stop**
  $\beta_i = (\boldsymbol{M}^{-1}\boldsymbol{r}_i, \boldsymbol{A}\boldsymbol{M}^{-1}\boldsymbol{r}_{i-1})/(\boldsymbol{M}^{-1}\boldsymbol{r}_{i-1}, \boldsymbol{A}\boldsymbol{M}^{-1}\boldsymbol{r}_{i-1})$
  $\boldsymbol{p}_i = \boldsymbol{M}^{-1}\boldsymbol{r}_i + \beta_i \boldsymbol{p}_i$
end for

Fig. 2.14 Preconditioned CR method

2.14.4 Biconjugate Gradient (BiCG) Method

双共役勾配法 (biconjugate gradient method, BiCG method) は，非対称行列に対応した Krylov 部分空間法である [9]．双共役勾配法は，Lanczos 双直交化過程に基づいており，以下の 2 つの Krylov 部分空間

$$\mathcal{K}_m(\boldsymbol{A}, \boldsymbol{v}) = \mathrm{span}\{\boldsymbol{v}, \boldsymbol{A}\boldsymbol{v}, \boldsymbol{A}^2\boldsymbol{v}, \ldots, \boldsymbol{A}^{m-1}\boldsymbol{v}\} \tag{2.80}$$
$$\mathcal{K}_m(\boldsymbol{A}^t, \boldsymbol{w}) = \mathrm{span}\{\boldsymbol{w}, \boldsymbol{A}\boldsymbol{w}, \boldsymbol{A}^2\boldsymbol{w}, \ldots, \boldsymbol{A}^{m-1}\boldsymbol{w}\} \tag{2.81}$$

に対し，$(\boldsymbol{v}_i, \boldsymbol{v}_j) = 0$ $(i \neq j)$ を満たす双直交基底を求めるものである．

図 2.15 に，双共役勾配法を示す．図 2.16 に，前処理つき双共役勾配法を示す．ここで，$\boldsymbol{A}$ は係数行列，$\boldsymbol{M}$ は前処理行列，$\boldsymbol{b}$ は右辺ベクトル，$\boldsymbol{x}$ は解ベクトル，$\boldsymbol{r}$ は残差ベクトル，$\boldsymbol{r}^*$ は

1: $\boldsymbol{r}_0 = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0$
2: $(\boldsymbol{r}_0^*, \boldsymbol{r}_0) \neq 0$
3: $\boldsymbol{p}_0 = \boldsymbol{r}_0$
4: $\boldsymbol{p}_0^* = \boldsymbol{r}_0^*$
5: **for** $i = 1, 2, 3, \ldots,$ **do**
6: $\quad \alpha_i = (\boldsymbol{r}_{i-1}, \boldsymbol{r}_{i-1}^*)/(\boldsymbol{A}\boldsymbol{p}_{i-1}, \boldsymbol{p}_{i-1}^*)$
7: $\quad \boldsymbol{x}_i = \boldsymbol{x}_{i-1} + \alpha_i \boldsymbol{p}_{i-1}$
8: $\quad \boldsymbol{r}_i = \boldsymbol{r}_{i-1} - \alpha_i \boldsymbol{A}\boldsymbol{p}_{i-1}$
9: $\quad \boldsymbol{r}_i^* = \boldsymbol{r}_{i-1}^* - \alpha_i \boldsymbol{A}^t \boldsymbol{p}_{i-1}^*$
10: $\quad$ **if** $(\|\boldsymbol{r}_i\|_2 / \|\boldsymbol{r}_0\|_2 \leq \varepsilon)$ **then stop**
11: $\quad \beta_i = (\boldsymbol{r}_i, \boldsymbol{r}_i^*)/(\boldsymbol{r}_{i-1}, \boldsymbol{r}_{i-1}^*)$
12: $\quad \boldsymbol{p}_i = \boldsymbol{r}_i + \beta_i \boldsymbol{p}_{i-1}$
13: $\quad \boldsymbol{p}_i^* = \boldsymbol{r}_i^* + \beta_i \boldsymbol{p}_{i-1}^*$
14: **end for**

Fig. 2.15 BiCG method

シャドウ残差ベクトル，$\boldsymbol{p}$ は探索方向ベクトル，$\boldsymbol{p}^*$ はシャドウ探索方向ベクトル，α，β は探索方向ベクトルの係数，ε は収束判定値である．

2.14.5 Conjugate Gradient Squared (CGS) Method

共役勾配自乗法 (conjugate gradient squared method, CGS method) は，非対称行列に対応した Krylov 部分空間法であり，双共役勾配法の改良手法として提案された [10]．残差ベクトル $\boldsymbol{r}$ を多項式で表すと，残差ベクトルとシャドウ残差ベクトルの内積が最小になるような条件が多項式の 2 乗で示されることに基づいている手法である．図 2.17 に，共役勾配自乗法を示す．図 2.18 に，前処理つき共役勾配自乗法を示す．ここで，$\boldsymbol{A}$ は係数行列，$\boldsymbol{M}$ は前処理行列，$\boldsymbol{b}$ は右辺ベクトル，$\boldsymbol{x}$ は解ベクトル，$\boldsymbol{r}$ は残差ベクトル，$\boldsymbol{r}^*$ はシャドウ残差ベクトル，$\boldsymbol{p}$，$\boldsymbol{u}$ は探索方向ベクトル，$\boldsymbol{p}^*$ はシャドウ探索方向ベクトル，α，β は探索方向ベクトルの係数，ε は収束判定値である．

2.14.6 Biconjugate Gradient Stabilized (BiCGStab) Method

双共役勾配安定化法 (biconjugate gradient stabilized method, BiCGStab method) は，非対称行列に対応した Krylov 部分空間法であり，双共役勾配法の改良手法として提案された [11]．共役勾配自乗法の収束性安定化のため，1 次の最小残差多項式を利用したものである．図 2.19 に，双共役勾配安定化法を示す．図 2.20 に，前処理つき双共役勾配安定化法を示す．ここで，$\boldsymbol{A}$ は係数行列，$\boldsymbol{M}$ は前処理行列，$\boldsymbol{b}$ は右辺ベクトル，$\boldsymbol{x}$ は解ベクトル，$\boldsymbol{r}$ は残差ベク

1: $\boldsymbol{r}_0 = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0$
2: $(\boldsymbol{r}_0^*, \boldsymbol{r}_0) \neq 0$
3: $\boldsymbol{p}_0 = \boldsymbol{M}^{-1}\boldsymbol{r}_0$
4: $\boldsymbol{p}_0^* = \boldsymbol{M}^{-1*}\boldsymbol{r}_0^*$
5: **for** $i = 1, 2, 3, \ldots,$ **do**
6: $\quad \alpha_i = (\boldsymbol{M}^{-1}\boldsymbol{r}_{i-1}, \boldsymbol{r}_{i-1}^*)/(\boldsymbol{A}\boldsymbol{p}_{i-1}, \boldsymbol{p}_{i-1}^*)$
7: $\quad \boldsymbol{x}_i = \boldsymbol{x}_{i-1} + \alpha_i \boldsymbol{p}_{i-1}$
8: $\quad \boldsymbol{r}_i = \boldsymbol{r}_{i-1} - \alpha_i \boldsymbol{A}\boldsymbol{p}_{i-1}$
9: $\quad \boldsymbol{r}_i^* = \boldsymbol{r}_{i-1}^* - \alpha_i \boldsymbol{A}^t \boldsymbol{p}_{i-1}^*$
10: $\quad$ **if** $(\|\boldsymbol{r}_i\|_2/\|\boldsymbol{r}_0\|_2 \leq \varepsilon)$ **then stop**
11: $\quad \beta_i = (\boldsymbol{M}^{-1}\boldsymbol{r}_i, \boldsymbol{r}_i^*)/(\boldsymbol{M}^{-1}\boldsymbol{r}_{i-1}, \boldsymbol{r}_{i-1}^*)$
12: $\quad \boldsymbol{p}_i = \boldsymbol{M}^{-1}\boldsymbol{r}_i + \beta_i \boldsymbol{p}_{i-1}$
13: $\quad \boldsymbol{p}_i^* = \boldsymbol{M}^{-1*}\boldsymbol{r}_i^* + \beta_i \boldsymbol{p}_{i-1}^*$
14: **end for**

Fig. 2.16 Preconditioned BiCG method

1: $\boldsymbol{r}_0 = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0$
2: $(\boldsymbol{r}_0^*, \boldsymbol{r}_0) \neq 0$
3: $\boldsymbol{p}_0 = \boldsymbol{r}_0$
4: $\boldsymbol{u}_0 = \boldsymbol{r}_0$
5: **for** $i = 1, 2, 3, \ldots,$ **do**
6: $\quad \alpha_i = (\boldsymbol{r}_{i-1}, \boldsymbol{r}_0^*)/(\boldsymbol{A}\boldsymbol{p}_{i-1}, \boldsymbol{r}_0^*)$
7: $\quad \boldsymbol{q}_i = \boldsymbol{u}_{i-1} - \alpha_i \boldsymbol{A}\boldsymbol{p}_{i-1}$
8: $\quad \boldsymbol{x}_i = \boldsymbol{x}_{i-1} + \alpha_i(\boldsymbol{u}_{i-1} + \boldsymbol{q}_i)$
9: $\quad \boldsymbol{r}_i = \boldsymbol{r}_{i-1} + \alpha_i \boldsymbol{A}(\boldsymbol{u}_{i-1} + \boldsymbol{q}_i)$
10: $\quad$ **if** $(\|\boldsymbol{r}_i\|_2/\|\boldsymbol{r}_0\|_2 \leq \varepsilon)$ **then stop**
11: $\quad \beta_i = (\boldsymbol{r}_i, \boldsymbol{r}_0^*)/(\boldsymbol{r}_{i-1}, \boldsymbol{r}_0^*)$
12: $\quad \boldsymbol{u}_i = \boldsymbol{r}_i + \beta_i \boldsymbol{q}_i$
13: $\quad \boldsymbol{p}_i = \boldsymbol{u}_i + \beta_i(\boldsymbol{q}_i + \beta_i \boldsymbol{p}_{i-1})$
14: **end for**

Fig. 2.17 CGS method

1: $\boldsymbol{r}_0 = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0$
2: $(\boldsymbol{r}_0^*, \boldsymbol{r}_0) \neq 0$
3: $\boldsymbol{p}_0 = \boldsymbol{r}_0$
4: $\boldsymbol{u}_0 = \boldsymbol{r}_0$
5: **for** $i = 1, 2, 3, \ldots,$ **do**
6: $\quad \alpha_i = (\boldsymbol{r}_{i-1}, \boldsymbol{r}_0^*)/(\boldsymbol{M}^{-1}\boldsymbol{A}\boldsymbol{p}_{i-1}, \boldsymbol{r}_0^*)$
7: $\quad \boldsymbol{q}_i = \boldsymbol{u}_{i-1} - \alpha_i \boldsymbol{M}^{-1}\boldsymbol{A}\boldsymbol{p}_{i-1}$
8: $\quad \boldsymbol{x}_i = \boldsymbol{x}_{i-1} + \alpha_i \boldsymbol{M}^{-1}(\boldsymbol{u}_{i-1} + \boldsymbol{q}_i)$
9: $\quad \boldsymbol{r}_i = \boldsymbol{r}_{i-1} + \alpha_i \boldsymbol{A}\boldsymbol{M}^{-1}(\boldsymbol{u}_{i-1} + \boldsymbol{q}_i)$
10: $\quad$ **if** $(\|\boldsymbol{r}_i\|_2/\|\boldsymbol{r}_0\|_2 \leq \varepsilon)$ **then stop**
11: $\quad \beta_i = (\boldsymbol{r}_i, \boldsymbol{r}_0^*)/(\boldsymbol{r}_{i-1}, \boldsymbol{r}_0^*)$
12: $\quad \boldsymbol{u}_i = \boldsymbol{r}_i + \beta_i \boldsymbol{q}_i$
13: $\quad \boldsymbol{p}_i = \boldsymbol{u}_i + \beta_i(\boldsymbol{q}_i + \beta_i \boldsymbol{p}_{i-1})$
14: **end for**

Fig. 2.18 Preconditioned CGS method

トル，$\boldsymbol{r}^*$ はシャドウ残差ベクトル，$\boldsymbol{p}$，$\boldsymbol{s}$ は探索方向ベクトル，α，β，ω は探索方向ベクトルの係数，ε は収束判定値である．

2.14.7 Generalized Product-type Biconjugate Gradient (GPBiCG) Method

一般化双共役勾配法 (generalized product-type biconjugate gradient method, GPBiCG method) は，非対称行列に対応した Krylov 部分空間法である [12]．図 2.21 に，双共役勾配安定化法を示す．ここで，$\boldsymbol{A}$ は係数行列，$\boldsymbol{M}$ は前処理行列，$\boldsymbol{b}$ は右辺ベクトル，$\boldsymbol{x}$ は解ベクトル，$\boldsymbol{r}$ は残差ベクトル，$\boldsymbol{r}^*$ はシャドウ残差ベクトル，$\boldsymbol{p}$，$\boldsymbol{t}$ は探索方向ベクトル，$\boldsymbol{v}$，$\boldsymbol{w}$，$\boldsymbol{y}$，$\boldsymbol{z}$ は一時ベクトル，ζ，η，α，β は探索方向ベクトルの係数，ε は収束判定値である．

2.14.8 Generalized Minimal Residual (GMRES) Method

一般化最小残差法 (generalized minimal residual method, GMRES method) は，非対称行列に対応した反復解法であり，Arnoldi 原理からヘッセンベルグ行列を経て，解ベクトルを得る手法である [13]．図 2.22 に，一般化最小残差法を示す．図 2.23 に，前処理つき一般化最小残差法を示す．ここで，$\boldsymbol{A}$ は係数行列，$\boldsymbol{M}$ は前処理行列，$\boldsymbol{b}$ は右辺ベクトル，$\boldsymbol{x}$ は解ベクトル，$\boldsymbol{r}$ は残差ベクトル，$\boldsymbol{v}_i$ は変換行列 $\boldsymbol{V}$ の第 i 番目の列ベクトル，h_{ij} はヘッセンベルグ行列 $\boldsymbol{H}$ の第 i 番目の列ベクトル，$\boldsymbol{w}$，$\boldsymbol{r}$ は一時ベクトル，β は一時ベクトルの係数，ε は収束判定値である．

1: $\boldsymbol{r}_0 = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0$
2: $(\boldsymbol{r}_0^*, \boldsymbol{r}_0) \neq 0$
3: $\boldsymbol{p}_0 = \boldsymbol{r}_0$
4: **for** $i = 1, 2, 3, \ldots,$ **do**
5: $\quad \alpha_i = (\boldsymbol{r}_{i-1}, \boldsymbol{r}_0^*)/(\boldsymbol{A}\boldsymbol{p}_{i-1}, \boldsymbol{r}_0^*)$
6: $\quad \boldsymbol{s}_i = \boldsymbol{r}_{i-1} - \alpha_i \boldsymbol{A}\boldsymbol{p}_{i-1}$
7: $\quad \omega_i = (\boldsymbol{A}\boldsymbol{s}_i, \boldsymbol{s}_i)/(\boldsymbol{A}\boldsymbol{s}_i, \boldsymbol{A}\boldsymbol{s}_i)$
8: $\quad \boldsymbol{x}_i = \boldsymbol{x}_{i-1} + \alpha_i \boldsymbol{p}_{i-1} + \omega_i \boldsymbol{s}_i$
9: $\quad \boldsymbol{r}_i = \boldsymbol{s}_{i-1} - \omega_i \boldsymbol{A}\boldsymbol{s}_{i-1}$
10: $\quad$ **if** $(\|\boldsymbol{r}_i\|_2/\|\boldsymbol{r}_0\|_2 \leq \varepsilon)$ **then stop**
11: $\quad \beta_i = \alpha_i/\omega_i \times (\boldsymbol{r}_i, \boldsymbol{r}_0^*)/(\boldsymbol{r}_{i-1}, \boldsymbol{r}_0^*)$
12: $\quad \boldsymbol{p}_i = \boldsymbol{r}_i + \beta_i(\boldsymbol{p}_{i-1} + \omega_i \boldsymbol{A}\boldsymbol{p}_{i-1})$
13: **end for**

Fig. 2.19 BiCGStab method

1: $\boldsymbol{r}_0 = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0$
2: $(\boldsymbol{r}_0^*, \boldsymbol{r}_0) \neq 0$
3: $\boldsymbol{p}_0 = \boldsymbol{r}_0$
4: **for** $i = 1, 2, 3, \ldots,$ **do**
5: $\quad \alpha_i = (\boldsymbol{r}_{i-1}, \boldsymbol{r}_0^*)/(\boldsymbol{A}\boldsymbol{M}^{-1}\boldsymbol{p}_{i-1}, \boldsymbol{r}_0^*)$
6: $\quad \boldsymbol{s}_i = \boldsymbol{r}_{i-1} - \alpha_i \boldsymbol{A}\boldsymbol{M}^{-1}\boldsymbol{p}_{i-1}$
7: $\quad \omega_i = (\boldsymbol{A}\boldsymbol{M}^{-1}\boldsymbol{s}_i, \boldsymbol{s}_i)/(\boldsymbol{A}\boldsymbol{M}^{-1}\boldsymbol{s}_i, \boldsymbol{A}\boldsymbol{M}^{-1}\boldsymbol{s}_i)$
8: $\quad \boldsymbol{x}_i = \boldsymbol{x}_{i-1} + \alpha_i \boldsymbol{M}^{-1}\boldsymbol{p}_{i-1} + \omega_i \boldsymbol{M}^{-1}\boldsymbol{s}_i$
9: $\quad \boldsymbol{r}_i = \boldsymbol{s}_{i-1} - \omega_i \boldsymbol{A}\boldsymbol{M}^{-1}\boldsymbol{s}_{i-1}$
10: $\quad$ **if** $(\|\boldsymbol{r}_i\|_2/\|\boldsymbol{r}_0\|_2 \leq \varepsilon)$ **then stop**
11: $\quad \beta_i = \alpha_i/\omega_i \times (\boldsymbol{r}_i, \boldsymbol{r}_0^*)/(\boldsymbol{r}_{i-1}, \boldsymbol{r}_0^*)$
12: $\quad \boldsymbol{p}_i = \boldsymbol{r}_i + \beta_i(\boldsymbol{p}_{i-1} + \omega_i \boldsymbol{A}\boldsymbol{p}_{i-1})$
13: **end for**

Fig. 2.20 Preconditioned BiCGStab method

$\boldsymbol{r}_0 = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0$, $\boldsymbol{r}_0^* = \boldsymbol{r}_0$
$\beta_{-1} = 0$
for $i = 0, 1, 2, \ldots,$ **do**
  $\boldsymbol{p}_i = \boldsymbol{r}_i + \beta_{i-1}(\boldsymbol{p}_{i-1} - \boldsymbol{u}_{i-1})$
  $\alpha_i = \dfrac{(\boldsymbol{r}_0^*, \boldsymbol{r}_i)}{(\boldsymbol{r}_0^*, \boldsymbol{A}\boldsymbol{p}_i)}$
  $\boldsymbol{y}_i = \boldsymbol{t}_{i-1} - \boldsymbol{r}_i - \alpha_i \boldsymbol{w}_{i-1} + \alpha_i \boldsymbol{A}\boldsymbol{p}_i$
  $\boldsymbol{t}_i = \boldsymbol{r}_i - \alpha_i \boldsymbol{A}\boldsymbol{p}_i$
  $\zeta_i = \dfrac{(\boldsymbol{y}_i, \boldsymbol{y}_i)(\boldsymbol{A}\boldsymbol{t}_i, \boldsymbol{t}_i) - (\boldsymbol{y}_i, \boldsymbol{t}_i)(\boldsymbol{A}\boldsymbol{t}_i, \boldsymbol{y}_i)}{(\boldsymbol{A}\boldsymbol{t}_i, \boldsymbol{A}\boldsymbol{t}_i)(\boldsymbol{y}_i, \boldsymbol{y}_i) - (\boldsymbol{y}_i, \boldsymbol{A}\boldsymbol{t}_i)(\boldsymbol{A}\boldsymbol{t}_i, \boldsymbol{y}_i)}$
  $\eta_i = \dfrac{(\boldsymbol{A}\boldsymbol{t}_i, \boldsymbol{A}\boldsymbol{t}_i)(\boldsymbol{y}_i, \boldsymbol{t}_i) - (\boldsymbol{y}_i, \boldsymbol{A}\boldsymbol{t}_i)(\boldsymbol{A}\boldsymbol{t}_i, \boldsymbol{t}_i)}{(\boldsymbol{A}\boldsymbol{t}_i, \boldsymbol{A}\boldsymbol{t}_i)(\boldsymbol{y}_i, \boldsymbol{y}_i) - (\boldsymbol{y}_i, \boldsymbol{A}\boldsymbol{t}_i)(\boldsymbol{A}\boldsymbol{t}_i, \boldsymbol{y}_i)}$
  $\boldsymbol{v}_i = \zeta_i \boldsymbol{A}\boldsymbol{p}_i + \eta_i(\boldsymbol{t}_{i-1} - \boldsymbol{r}_i + \beta_{i-1}\boldsymbol{u}_{i-1})$
  $\boldsymbol{z}_i = \zeta_i \boldsymbol{r}_i + \eta_i \boldsymbol{z}_{i-1} - \alpha_i \boldsymbol{u}_i)$
  $\boldsymbol{x}_{i+1} = \boldsymbol{x}_i + \alpha_i \boldsymbol{p}_i + \boldsymbol{z}_i$
  $\boldsymbol{r}_{i+1} = \boldsymbol{t}_i - \eta \boldsymbol{y}_i - \zeta \boldsymbol{A}\boldsymbol{t}_i$
  if $(\|\boldsymbol{r}_{i+1}\|_2 / \|\boldsymbol{r}_0\|_2 \leq \varepsilon)$ **then stop**
  $\beta_i = \dfrac{\alpha_i}{\eta_i} \dfrac{(\boldsymbol{r}_0^*, \boldsymbol{r}_{i+1})}{(\boldsymbol{r}_0^*, \boldsymbol{r}_i)}$
  $\boldsymbol{w}_i = \boldsymbol{A}\boldsymbol{t}_i + \beta_i \boldsymbol{A}\boldsymbol{p}_i$
end for

Fig. 2.21 GPBiCG method

$\boldsymbol{r}_0 = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0$
$\beta = \|\boldsymbol{r}_0\|_2$
$\boldsymbol{v} = \boldsymbol{r}_0 / \beta$
for $i = 1, 2, 3, \ldots, m$ **do**
  $\boldsymbol{w}_i = \boldsymbol{A}\boldsymbol{v}_j$
  for $j = 1, 2, 3, \ldots, i$ **do**
    $h_{ji} = (\boldsymbol{w}_i, \boldsymbol{v}_j)$
    $\boldsymbol{w}_i = \boldsymbol{w}_i - h_{ji}\boldsymbol{v}_j$
  end for
  $h_{i+1,i} = \|\boldsymbol{w}_i\|_2$
  $\boldsymbol{v}_i = \boldsymbol{w}_i / h_{i+1,i}$
end for
Compute $\boldsymbol{y}_m$ the minimizer of $\|\beta \boldsymbol{e}_1 - \boldsymbol{H}_m \boldsymbol{y}_m\|_2$
$\boldsymbol{x}_m = \boldsymbol{x}_0 + \boldsymbol{V}_m \boldsymbol{y}_m$

Fig. 2.22 GMRES method

1: $\boldsymbol{r}_0 = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0$
2: $\beta = \|\boldsymbol{r}_0\|_2$
3: $\boldsymbol{v} = \boldsymbol{r}_0/\beta$
4: **for** $i = 1, 2, 3, \ldots, m$ **do**
5: $\quad \boldsymbol{w}_i = \boldsymbol{A}\boldsymbol{v}_j$
6: $\quad$ **for** $j = 1, 2, 3, \ldots, i$ **do**
7: $\quad\quad h_{ji} = (\boldsymbol{w}_i, \boldsymbol{v}_j)$
8: $\quad\quad \boldsymbol{w}_i = \boldsymbol{w}_i - h_{ji}\boldsymbol{v}_j$
9: $\quad$ **end for**
10: $\quad h_{i+1,i} = \|\boldsymbol{w}_i\|_2$
11: $\quad \boldsymbol{v}_i = \boldsymbol{w}_i/h_{i+1,i}$
12: **end for**
13: Compute $\boldsymbol{y}_m$ the minimizer of $\|\beta\boldsymbol{e}_1 - \boldsymbol{H}_m\boldsymbol{y}_m\|_2$
14: $\boldsymbol{x}_m = \boldsymbol{x}_0 + \boldsymbol{V}_m\boldsymbol{y}_m$

Fig. 2.23 Preconditioned GMRES method

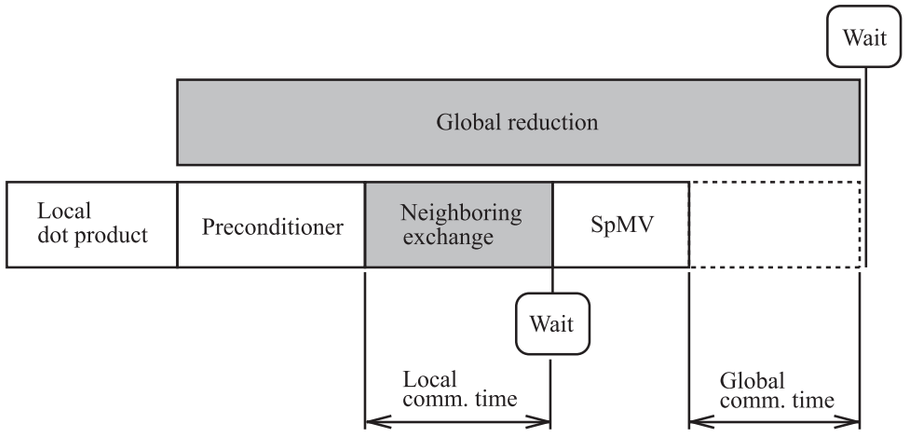


Fig. 2.24 Schematic representation of communication hiding

2.15 通信隠蔽型反復法

本節では，ベクトル内積に起因する大域的な通信が占める通信時間を隠蔽する，通信隠蔽型反復法について述べる．

反復法のアルゴリズムは，疎行列ベクトル積 (sperse matrix vector multiplication, SpMV)，ベクトル内積 (dot product)，ベクトル和 (alpha x Plus y, AXPY) による線形演算で構成されている．反復法を分散メモリ環境の並列計算機を用いて計算する場合，疎行列ベクトル積とベクトル内積の演算は，並列プロセス間の通信を行う必要がある．

領域分割法に基づく疎行列ベクトル積の演算は，各領域と隣接する領域との間で局所的な通信を行うことで，並列計算を実現できる．ベクトル内積の演算は，各領域での内積を計算した後，大域的な通信を行うことで，並列計算を実現できる．一方，ベクトル和の演算は，並列計算を行う場合でも領域間の通信を必要としない．

並列計算機のコア数の増加に伴い，並列プロセス数が増加すると，大域的な通信が必要なベクトル内積は，局所的な通信が必要な疎行列ベクトル積に比べ，並列プロセス数が n のとき $O(\log n)$ 倍の通信コストがかかる．例えば，従来の共役勾配法では，一反復あたり 2 回のベクトル内積が必要であり，大規模並列計算を行う場合には，ベクトル内積に起因する大域的な通信が占める通信時間が無視できなくなっている．

この解決のため，反復法を数学的に等価に変換することで，大域的な通信が必要な内積計算の通信時間による遅延を隠蔽する手法が提案された．図 2.24 に，通信隠蔽部分の概略図を示す．図 2.24 より，ベクトル内積の大域的通信に並行して，前処理や疎行列ベクトル積の計算を行うことで，通信時間の隠蔽を可能にしていることがわかる．この実現のため，MPI 3.0 準拠の非同期全体通信関数 MPI_Iallreduce を用いて計算を行う [14]．

1: $\boldsymbol{r}_0 = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0$, $\boldsymbol{u}_0 = \boldsymbol{M}^{-1}\boldsymbol{r}_0$
2: $\boldsymbol{p}_0 = \boldsymbol{u}_0$, $\boldsymbol{s}_0 = \boldsymbol{A}\boldsymbol{p}_0$, $\gamma_0 = (\boldsymbol{r}_0, \boldsymbol{u}_0)$
3: **for** $i = 0, 1, 2, \ldots,$ **do**
4: $\quad \delta = (\boldsymbol{p}_i, \boldsymbol{s}_i)$
5: $\quad \boldsymbol{q}_i = \boldsymbol{M}^{-1}\boldsymbol{s}_i$
6: $\quad \alpha_i = \gamma_i / \delta$
7: $\quad \boldsymbol{x}_{i+1} = \boldsymbol{x}_i + \alpha_i \boldsymbol{p}_i$
8: $\quad \boldsymbol{r}_{i+1} = \boldsymbol{r}_i - \alpha_i \boldsymbol{s}_i$
9: $\quad$ **if** $(\|\boldsymbol{r}_{i+1}\|_2 / \|\boldsymbol{r}_0\|_2 \leq \varepsilon)$ **then stop**
10: $\quad \boldsymbol{u}_{i+1} = \boldsymbol{u}_i - \alpha_i \boldsymbol{q}_i$
11: $\quad \gamma_{i+1} = (\boldsymbol{r}_{i+1}, \boldsymbol{u}_{i+1})$
12: $\quad \boldsymbol{w}_{i+1} = \boldsymbol{A}\boldsymbol{u}_{i+1}$
13: $\quad \beta_{i+1} = \gamma_{i+1} / \gamma_i$
14: $\quad \boldsymbol{p}_{i+1} = \boldsymbol{u}_{i+1} + \beta_{i+1}\boldsymbol{p}_i$
15: $\quad \boldsymbol{s}_{i+1} = \boldsymbol{w}_{i+1} + \beta_{i+1}\boldsymbol{s}_i$
16: **end for**

Fig. 2.25 Grop asynchronous CG method

2.15.1 Grop asynchronous CG Method

Grop asynchronous CG 法は，共役勾配法を数学的に等価に変換することで，前処理と疎行列ベクトル積のそれぞれの計算にあわせて，ベクトル内積の通信時間を隠蔽することのできる手法である [15]．また，前処理の適用と疎行列ベクトル積の適用が分離していることが特徴として挙げられる．図 2.25 に，Grop asynchronous CG method を示す．ここで，$\boldsymbol{A}$ は係数行列，$\boldsymbol{M}$ は前処理行列，$\boldsymbol{b}$ は右辺ベクトル，$\boldsymbol{x}$ は解ベクトル，$\boldsymbol{r}$ は残差ベクトル，$\boldsymbol{p}$ は探索方向ベクトル，$\boldsymbol{s}$，$\boldsymbol{q}$，$\boldsymbol{w}$，$\boldsymbol{u}$ は一時ベクトル，δ，α，β，γ は探索方向ベクトルの係数，ε は収束判定値である．

2.15.2 Pipelined Chronopoulos Gear CG Method

pipelined chronopoulos Gear CG 法は，共役勾配法を数学的に等価に変換することで，前処理と疎行列ベクトル積の計算にあわせて，ベクトル内積の通信時間を隠蔽することのできる手法である [16, 17, 15]．Grop asynchronous CG 法では，前処理と疎行列ベクトル積の計算が独立に行われていたのに対し，pipelined chronopoulos Gear CG 法では，前処理と疎行列ベクトル積を一連に計算することで，1 反復あたりのベクトル内積演算を 1 回に抑えている．図 2.26 に，chronopoulos Gear CG method を示す．図 2.27 に，pipelined chronopoulos Gear

1: $\boldsymbol{r}_0 = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0$
2: $\boldsymbol{u}_0 = \boldsymbol{M}^{-1}\boldsymbol{r}_0$
3: $\boldsymbol{w}_0 = \boldsymbol{A}\boldsymbol{u}_0$
4: $\alpha_0 = (\boldsymbol{r}_0, \boldsymbol{u}_0)/(\boldsymbol{w}_0, \boldsymbol{u}_0)$
5: $\beta_0 = 0$
6: $\gamma_0 = (\boldsymbol{r}_0, \boldsymbol{u}_0)$
7: **for** $i = 0, 1, 2, \ldots,$ **do**
8: $\quad \boldsymbol{p}_i = \boldsymbol{u}_i + \beta_i \boldsymbol{p}_{i-1}$
9: $\quad \boldsymbol{s}_i = \boldsymbol{w}_i + \beta_i \boldsymbol{s}_{i-1}$
10: $\quad \boldsymbol{x}_{i+1} = \boldsymbol{x}_i + \alpha_i \boldsymbol{p}_i$
11: $\quad \boldsymbol{r}_{i+1} = \boldsymbol{r}_i - \alpha_i \boldsymbol{s}_i$
12: $\quad$ **if** $(\|\boldsymbol{r}_{i+1}\|_2/\|\boldsymbol{r}_0\|_2 \leq \varepsilon)$ **then stop**
13: $\quad \boldsymbol{u}_{i+1} = \boldsymbol{M}^{-1}\boldsymbol{r}_{i+1}$
14: $\quad \boldsymbol{w}_{i+1} = \boldsymbol{A}\boldsymbol{u}_{i+1}$
15: $\quad \gamma_{i+1} = (\boldsymbol{r}_{i+1}, \boldsymbol{u}_{i+1})$
16: $\quad \delta = (\boldsymbol{w}_{i+1}, \boldsymbol{u}_{i+1})$
17: $\quad \beta_{i+1} = \gamma_{i+1}/\gamma_i$
18: $\quad \alpha_{i+1} = \gamma_{i+1}/(\delta - \beta_{i+1}\gamma_{i+1}/\alpha_i)$
19: **end for**

Fig. 2.26 Chronopoulos Gear CG method

CG method を示す．ここで，$\boldsymbol{A}$ は係数行列，$\boldsymbol{M}$ は前処理行列，$\boldsymbol{b}$ は右辺ベクトル，$\boldsymbol{x}$ は解ベクトル，$\boldsymbol{r}$ は残差ベクトル，$\boldsymbol{p}$ は探索方向ベクトル，$\boldsymbol{s}$，$\boldsymbol{q}$，$\boldsymbol{w}$，$\boldsymbol{u}$ は一時ベクトル，δ，α，β，γ は探索方向ベクトルの係数，ε は収束判定値である．

1: $\boldsymbol{r}_0 = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0$
2: $\boldsymbol{w}_0 = \boldsymbol{A}\boldsymbol{r}_0$
3: **for** $i = 0, 1, 2, \ldots,$ **do**
4: $\quad \gamma_i = (\boldsymbol{r}_i, \boldsymbol{r}_i)$
5: $\quad \delta = (\boldsymbol{w}_i, \boldsymbol{r}_i)$
6: $\quad \boldsymbol{q}_i = \boldsymbol{A}\boldsymbol{w}_i$
7: $\quad$ **if** $i > 0$ **then**
8: $\quad\quad \beta_i = \gamma_i/\gamma_{i-1}$; $\alpha_i = \gamma_i/(\delta - \beta_i\gamma_i/\alpha_{i-1})$
9: $\quad$ **else**
10: $\quad\quad \beta_0 = 0$; $\alpha_0 = \gamma_0/\delta$
11: $\quad$ **end if**
12: $\quad \boldsymbol{z}_i = \boldsymbol{q}_i + \beta_i\boldsymbol{z}_{i-1}$
13: $\quad \boldsymbol{s}_i = \boldsymbol{w}_i + \beta_i\boldsymbol{s}_{i-1}$
14: $\quad \boldsymbol{p}_i = \boldsymbol{r}_i + \beta_i\boldsymbol{p}_{i-1}$
15: $\quad \boldsymbol{x}_{i+1} = \boldsymbol{x}_i + \alpha_i\boldsymbol{p}_i$
16: $\quad \boldsymbol{r}_{i+1} = \boldsymbol{r}_i - \alpha_i\boldsymbol{s}_i$
17: $\quad$ **if** $(\|\boldsymbol{r}_{i+1}\|_2/\|\boldsymbol{r}_0\|_2 \leq \varepsilon)$ **then stop**
18: $\quad \boldsymbol{w}_{i+1} = \boldsymbol{w}_i - \alpha_i\boldsymbol{z}_i$
19: **end for**

Fig. 2.27 Pipelined Chronopoulos Gear CG method

1: $\boldsymbol{r}_0 = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0$, $\boldsymbol{u}_0 = \boldsymbol{M}^{-1}\boldsymbol{r}_0$, $\boldsymbol{w}_0 = \boldsymbol{A}\boldsymbol{u}_0$
2: **for** $i = 0, 1, 2, \ldots,$ **do**
3: $\quad \boldsymbol{m}_i = \boldsymbol{M}^{-1}\boldsymbol{w}_i$
4: $\quad \gamma_i = (\boldsymbol{w}_i, \boldsymbol{u}_i)$
5: $\quad \delta = (\boldsymbol{m}_i, \boldsymbol{w}_i)$
6: $\quad \boldsymbol{n}_i = \boldsymbol{A}\boldsymbol{m}_i$
7: $\quad$ **if** $i > 0$ **then**
8: $\quad\quad \beta_i = \gamma_i/\gamma_{i-1}$; $\alpha_i = \gamma_i/(\delta - \beta_i\gamma_i/\alpha_{i-1})$
9: $\quad$ **else**
10: $\quad\quad \beta_0 = 0$; $\alpha_0 = \gamma_0/\delta$
11: $\quad$ **end if**
12: $\quad \boldsymbol{z}_i = \boldsymbol{n}_i + \beta_i\boldsymbol{z}_{i-1}$
13: $\quad \boldsymbol{q}_i = \boldsymbol{m}_i + \beta_i\boldsymbol{q}_{i-1}$
14: $\quad \boldsymbol{p}_i = \boldsymbol{u}_i + \beta_i\boldsymbol{p}_{i-1}$
15: $\quad \boldsymbol{x}_{i+1} = \boldsymbol{x}_i + \alpha_i\boldsymbol{p}_i$
16: $\quad \boldsymbol{r}_{i+1} = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_{i+1}$
17: $\quad$ **if** $(\|\boldsymbol{r}_{i+1}\|_2/\|\boldsymbol{r}_0\|_2 \leq \varepsilon)$ **then stop**
18: $\quad \boldsymbol{u}_{i+1} = \boldsymbol{u}_i - \alpha_i\boldsymbol{q}_i$
19: $\quad \boldsymbol{w}_{i+1} = \boldsymbol{w}_i - \alpha_i\boldsymbol{z}_i$
20: **end for**

Fig. 2.28 Preconditioned pipelined CR method

2.15.3 Pipelined CR Method

pipelined CR method 法は，共役残差法を数学的に等価に変換することで，疎行列ベクトル積の計算にあわせて，ベクトル内積の通信時間を隠蔽することのできる手法である [15]．図 2.28 に，preconditioned pipelined CR method を示す．ここで，$\boldsymbol{A}$ は係数行列，$\boldsymbol{M}$ は前処理行列，$\boldsymbol{b}$ は右辺ベクトル，$\boldsymbol{x}$ は解ベクトル，$\boldsymbol{r}$ は残差ベクトル，$\boldsymbol{p}$ は探索方向ベクトル，$\boldsymbol{m}$，$\boldsymbol{q}$，$\boldsymbol{w}$，$\boldsymbol{u}$ は一時ベクトル，δ，α，β，γ は探索方向ベクトルの係数，ε は収束判定値である．

2.15.4 Pipelined CG Method

pipelined CG method は，全体通信に必要な同期による遅延を隠蔽するために，Ghysels と Vanroose によって提案された手法である [15]．図 2.28 に，preconditioned pipelined CG method を示す．ここで，$\boldsymbol{A}$ は係数行列，$\boldsymbol{M}$ は前処理行列，$\boldsymbol{b}$ は右辺ベクトル，$\boldsymbol{x}$ は解ベクトル，$\boldsymbol{r}$ は残差ベクトル，$\boldsymbol{p}$ は探索方向ベクトル，$\boldsymbol{m}$，$\boldsymbol{n}$，$\boldsymbol{q}$，$\boldsymbol{u}$，$\boldsymbol{w}$，$\boldsymbol{z}$ は一時ベクトル，δ，α，

1: $\boldsymbol{r}_0 = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0$, $\boldsymbol{u}_0 = \boldsymbol{M}^{-1}\boldsymbol{r}_0$, $\boldsymbol{w}_0 = \boldsymbol{A}\boldsymbol{u}_0$
2: **for** $i = 0, 1, 2, \ldots,$ **do**
3: $\quad \gamma_i = (\boldsymbol{r}_i, \boldsymbol{u}_i)$
4: $\quad \delta = (\boldsymbol{w}_i, \boldsymbol{u}_i)$
5: $\quad \boldsymbol{m}_i = \boldsymbol{M}^{-1}\boldsymbol{w}_i$
6: $\quad \boldsymbol{n}_i = \boldsymbol{A}\boldsymbol{m}_i$
7: $\quad$ **if** $i > 0$ **then**
8: $\quad\quad \beta_i = \gamma_i/\gamma_{i-1}$; $\alpha_i = \gamma_i/(\delta - \beta_i\gamma_i/\alpha_{i-1})$
9: $\quad$ **else**
10: $\quad\quad \beta_0 = 0$; $\alpha_0 = \gamma_0/\delta$
11: $\quad$ **end if**
12: $\quad \boldsymbol{z}_i = \boldsymbol{n}_i + \beta_i\boldsymbol{z}_{i-1}$
13: $\quad \boldsymbol{q}_i = \boldsymbol{m}_i + \beta_i\boldsymbol{q}_{i-1}$
14: $\quad \boldsymbol{s}_i = \boldsymbol{w}_i + \beta_i\boldsymbol{s}_{i-1}$
15: $\quad \boldsymbol{p}_i = \boldsymbol{u}_i + \beta_i\boldsymbol{p}_{i-1}$
16: $\quad \boldsymbol{x}_{i+1} = \boldsymbol{x}_i + \alpha_i\boldsymbol{p}_i$
17: $\quad \boldsymbol{r}_{i+1} = \boldsymbol{r}_i - \alpha_i\boldsymbol{s}_i$
18: $\quad \boldsymbol{u}_{i+1} = \boldsymbol{u}_i - \alpha_i\boldsymbol{q}_i$
19: $\quad \boldsymbol{w}_{i+1} = \boldsymbol{w}_i - \alpha_i\boldsymbol{z}_i$
20: $\quad$ **if** $(\|\boldsymbol{r}_{i+1}\|_2/\|\boldsymbol{r}_0\|_2 \leq \varepsilon)$ **then stop**
21: **end for**

Fig. 2.29 Preconditioned pipelined CG method

β，γ は探索方向ベクトルの係数，ε は収束判定値である．

本手法は，新たに追加した一時ベクトルによって，図 2.28 の 3，4 行目と 5，6 行目に関わる計算をオーバーラッピングすることが可能である．

反復中に必要なベクトルの本数は一般的な共役勾配法で x，r，s，u の 4 本，PipeCG で x，r，s，u，z，q，n，m，w の 9 本である．そのため，メモリ使用量と一反復あたりの演算量は増加するが，数値計算では効率的なメモリアクセスが可能であるため，計算時間の増加は無視できる．一般的な共役勾配法と数学的に等価なことが証明されているが，一時ベクトルを多く扱うことによる丸め誤差の影響が大きい．

2.15.5 Pipelined BiCGSTAB Method

pipelined BiCGSTAB method は，BiCGSTAB 法を数学的に等価に変換することで，前処理と疎行列ベクトル積の計算にあわせて，ベクトル内積の通信時間を隠蔽することのできる手

$\boldsymbol{r}_0 = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0$
$\boldsymbol{w}_0 = \boldsymbol{A}\boldsymbol{r}_0$
$\boldsymbol{t}_0 = \boldsymbol{A}\boldsymbol{w}_0$
$\alpha = (\boldsymbol{r}_0, \boldsymbol{r}_0)/(\boldsymbol{r}_0, \boldsymbol{w}_0)$
$\beta_{-1} = 0$
$c_0 = (\boldsymbol{r}_0, \boldsymbol{r}_0)$
for $i = 0, 1, 2, \ldots,$ **do**
  $\boldsymbol{p}_i = \boldsymbol{r}_i + \beta_{i-1}(\boldsymbol{p}_{i-1} - \omega_{i-1}\boldsymbol{s}_{i-1})$
  $\boldsymbol{s}_i = \boldsymbol{w}_i + \beta_{i-1}(\boldsymbol{s}_{i-1} - \omega_{i-1}\boldsymbol{z}_{i-1})$
  $\boldsymbol{z}_i = \boldsymbol{t}_i + \beta_{i-1}(\boldsymbol{z}_{i-1} - \omega_{i-1}\boldsymbol{v}_{i-1})$
  $\boldsymbol{q}_i = \boldsymbol{r}_i - \alpha_i\boldsymbol{s}_i$
  $\boldsymbol{y}_i = \boldsymbol{w}_i - \alpha_i\boldsymbol{z}_i$
  $\gamma = (\boldsymbol{q}_i, \boldsymbol{y}_i)$
  $\delta = (\boldsymbol{y}_i, \boldsymbol{y}_i)$
  $\boldsymbol{v}_i = \boldsymbol{A}\boldsymbol{z}_i$
  $\omega_i = \gamma/\delta$
  $\boldsymbol{x}_{i+1} = \boldsymbol{x}_i + \alpha_i\boldsymbol{p}_i + \omega_i\boldsymbol{q}_i$
  $\boldsymbol{r}_{i+1} = \boldsymbol{q}_i - \omega_i\boldsymbol{y}_i$
  if $(\|\boldsymbol{r}_{i+1}\|_2/\|\boldsymbol{r}_0\|_2 \leq \varepsilon)$ **then stop**
  $\boldsymbol{w}_{i+1} = \boldsymbol{y}_i - \omega_i(\boldsymbol{t}_i - \alpha_i\boldsymbol{v}_i)$
  $c_1 = (\boldsymbol{r}_0, \boldsymbol{r}_{i+1})$
  $c_2 = (\boldsymbol{r}_0, \boldsymbol{w}_{i+1})$
  $c_3 = (\boldsymbol{r}_0, \boldsymbol{s}_i)$
  $c_4 = (\boldsymbol{r}_0, \boldsymbol{z}_i)$
  $\boldsymbol{t}_{i+1} = \boldsymbol{A}\boldsymbol{w}_{i+1}$
  $\beta_i = (\alpha_i/\omega_i)c_1/c_0$
  $\alpha_{i+1} = c_1/(c_2 + \beta_i c_3 - \beta_i\omega_i c_4)$
  $c_0 = c_1$
end for

Fig. 2.30 Pipelined BiCGSTAB method

法である [18]．図 2.30 に，pipelined BiCGSTAB method を示す．図 2.31 に，preconditioned pipelined BiCGSTAB method を示す．ここで，$\boldsymbol{A}$ は係数行列，$\boldsymbol{M}$ は前処理行列，$\boldsymbol{b}$ は右辺ベクトル，$\boldsymbol{x}$ は解ベクトル，$\boldsymbol{r}$ は残差ベクトル，$\boldsymbol{r}^*$ はシャドウ残差ベクトル，$\boldsymbol{s}$，$\boldsymbol{t}$，$\boldsymbol{q}$，$\boldsymbol{y}$，$\boldsymbol{v}$，$\boldsymbol{w}$，$\boldsymbol{z}$，$\boldsymbol{p}^*$，$\boldsymbol{s}^*$，$\boldsymbol{q}^*$，$\boldsymbol{w}^*$，$\boldsymbol{z}^*$ は一時ベクトル，ω，α，β，γ，δ，c_0，c_1，c_2，c_3，c_4 は探索方向ベクトルの係数，ε は収束判定値である．

1: $\boldsymbol{r}_0 = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0$, $\boldsymbol{r}_0^* = \boldsymbol{M}^{-1}\boldsymbol{r}_0$
2: $\boldsymbol{w}_0 = \boldsymbol{A}\boldsymbol{r}_0^*$, $\boldsymbol{w}_0^* = \boldsymbol{M}^{-1}\boldsymbol{w}_0$
3: $\boldsymbol{t}_0^* = \boldsymbol{A}\boldsymbol{w}_0^*$
4: $\alpha = (\boldsymbol{r}_0, \boldsymbol{r}_0)/(\boldsymbol{r}_0, \boldsymbol{w}_0)$
5: $\beta_{-1} = 0$
6: $c_0 = (\boldsymbol{r}_0, \boldsymbol{r}_0)$
7: **for** $i = 0, 1, 2, \ldots,$ **do**
8: $\quad \boldsymbol{p}_i^* = \boldsymbol{r}_i^* + \beta_{i-1}(\boldsymbol{p}_{i-1}^* - \omega_{i-1}\boldsymbol{s}_{i-1}^*)$
9: $\quad \boldsymbol{s}_i = \boldsymbol{w}_i + \beta_{i-1}(\boldsymbol{s}_{i-1} - \omega_{i-1}\boldsymbol{z}_{i-1})$
10: $\quad \boldsymbol{s}_i^* = \boldsymbol{w}_i^* + \beta_{i-1}(\boldsymbol{s}_{i-1}^* - \omega_{i-1}\boldsymbol{z}_{i-1}^*)$
11: $\quad \boldsymbol{z}_i = \boldsymbol{t}_i + \beta_{i-1}(\boldsymbol{z}_{i-1} - \omega_{i-1}\boldsymbol{v}_{i-1})$
12: $\quad \boldsymbol{q}_i = \boldsymbol{r}_i - \alpha_i\boldsymbol{s}_i$
13: $\quad \boldsymbol{q}_i^* = \boldsymbol{r}_i^* - \alpha_i\boldsymbol{s}_i^*$
14: $\quad \boldsymbol{y}_i = \boldsymbol{w}_i - \alpha_i\boldsymbol{z}_i$
15: $\quad \gamma = (\boldsymbol{q}_i, \boldsymbol{y}_i)$
16: $\quad \delta = (\boldsymbol{y}_i, \boldsymbol{y}_i)$
17: $\quad \boldsymbol{z}_i^* = \boldsymbol{M}^{-1}\boldsymbol{z}_i$
18: $\quad \boldsymbol{v}_i = \boldsymbol{A}\boldsymbol{z}_i^*$
19: $\quad \omega_i = \gamma/\delta$
20: $\quad \boldsymbol{x}_{i+1} = \boldsymbol{x}_i + \alpha_i\boldsymbol{p}_i^* + \omega_i\boldsymbol{q}_i^*$
21: $\quad \boldsymbol{r}_{i+1} = \boldsymbol{q}_i - \omega_i\boldsymbol{y}_i$
22: $\quad$ **if** $(\|\boldsymbol{r}_{i+1}\|_2/\|\boldsymbol{r}_0\|_2 \leq \varepsilon)$ **then stop**
23: $\quad \boldsymbol{r}_{i+1}^* = \boldsymbol{q}_i^* - \omega_i(\boldsymbol{w}_i^* - \alpha\boldsymbol{z}_i^*)$
24: $\quad \boldsymbol{w}_{i+1} = \boldsymbol{y}_i - \omega_i(\boldsymbol{t}_i - \alpha_i\boldsymbol{v}_i)$
25: $\quad c_1 = (\boldsymbol{r}_0, \boldsymbol{r}_{i+1})$, $c_1 = (\boldsymbol{r}_0, \boldsymbol{r}_{i+1})$, $c_3 = (\boldsymbol{r}_0, \boldsymbol{s}_i)$, $c_4 = (\boldsymbol{r}_0, \boldsymbol{z}_i)$
26: $\quad \boldsymbol{w}_{i+1}^* = \boldsymbol{M}^{-1}\boldsymbol{w}_{i+1}$
27: $\quad \boldsymbol{t}_{i+1} = \boldsymbol{A}\boldsymbol{w}_{i+1}^*$
28: $\quad \beta_i = (\alpha_i/\omega_i)c_1/c_0$
29: $\quad \alpha_{i+1} = c_1/(c_2 + \beta_i c_3 - \beta_i\omega_i c_4)$
30: $\quad c_0 = c_1$
31: **end for**

Fig. 2.31 Preconditioned Pipelined BiCGSTAB method

1: $\boldsymbol{M} = \boldsymbol{0}$
2: **for** $i = 1, 2, 3, \ldots, n$ **do**
3: $\quad m_{ii} = a_{ii}$
4: **end for**

Fig. 2.32 Diagonal scaling preconditioner

1: $\boldsymbol{M} = \boldsymbol{0}$
2: **for** $i = 1, 2, 3, \ldots, n$ **do**
3: $\quad \boldsymbol{M}_{ii} = \boldsymbol{A}_{ii}$
4: **end for**

Fig. 2.33 Block diagonal scaling preconditioner

2.16 反復法前処理

反復法前処理は，解くべき係数行列の固有値分布を改善し元の係数行列より小さな条件数を得ることで，反復法を安定に解くための重要な技術である．本節では，その中でも代表的な前処理である，diagonal scaling 前処理，successive over-relaxation 前処理，incomplete LU(p) 分解前処理について述べる．その他の前処理として，AINV 前処理，SAINV 前処理，ISAINV 前処理，RIF 前処理，IRIF 前処理が挙げられるが，これらについては第 4 章で取り扱う．

2.16.1 Diagonal Scaling

対角スケーリング (diagonal scaling) 前処理は，前処理行列 $\boldsymbol{M}$ の対角成分を係数行列 $\boldsymbol{A}$ の対角成分として用いるものである．図 2.32 に，対角スケーリング前処理を示す．ここで，m_{ii} は前処理行列 $\boldsymbol{M}$ の i 番目の対角要素，a_{ii} は係数行列 $\boldsymbol{A}$ の i 番目の対角要素である．対角スケーリング前処理は，計算負荷が小さく，完全に並列計算が可能であるのが特徴である．係数行列 $\boldsymbol{A}$ の強い近似ではないが，対角優位な行列に対して有効な前処理能力をもつ場合がある．

ブロック対角スケーリング前処理では，対角ブロックの逆行列を計算して前処理行列を生成する．図 2.33 に，ブロック対角スケーリング前処理を示す．ここで，$\boldsymbol{M}_{ii}$ は前処理行列 $\boldsymbol{M}$ の i 番目の対角ブロック，$\boldsymbol{A}_{ii}$ は係数行列 $\boldsymbol{A}$ の i 番目の対角ブロックである．ブロック対角スケーリング前処理では，対角成分の逆行列は陽に保持せず，LU 分解後の成分を保持することで，計算効率を高める方法が採用される場合がある．

```
1: M = A
2: for k = 1, 2, 3, ..., n do
3:   for j = k + 1, k + 2, ..., n do
4:     if m_jk ≠ 0 then
5:       m_jk = m_jk / m_kk
6:     end if
7:     for i = k + 1, k + 2, ..., n do
8:       if m_ji ≠ 0 then
9:         m_ji = m_ji − m_jk m_ki
10:      end if
11:    end for
12:  end for
13: end for
```

Fig. 2.34 Incomplete LU(0) preconditioner

2.16.2 Successive Over-Relaxation (SOR)

逐次過緩和前処理 (successive over-relaxation, SOR) は，式 (2.82) に示すように係数行列 $\boldsymbol{A}$ を行列の和で表すことで得られる分離型前処理である．このとき，前処理行列 $\boldsymbol{M}$ を式 (2.83) のように生成する．

$$\boldsymbol{A} = \boldsymbol{L} + \boldsymbol{D} + \boldsymbol{U} \tag{2.82}$$

$$\boldsymbol{M} = \left(\frac{\boldsymbol{D}}{\omega} + \boldsymbol{L}\right) \frac{\omega}{2-\omega} \boldsymbol{D}^{-1} \left(\frac{\boldsymbol{D}}{\omega} + \boldsymbol{U}\right) \tag{2.83}$$

ここで，$\boldsymbol{L}$ は狭義下三角行列，$\boldsymbol{D}$ は対角行列，$\boldsymbol{U}$ は狭義上三角行列，ω は緩和パラメータであり $0 < \omega < 2$ をとる．

2.16.3 Imcomplete LU(p) (ILU) factorization

不完全 LU 分解前処理 (incomplete LU factorization, ILU) は，Gauss の消去法に基づいて近似的な LU 分解を施すことで前処理行列 $\boldsymbol{M} = \boldsymbol{LU}$ を生成する手法である．特に，分解時の前処理行列の非零プロファイルを係数行列と同じものに固定して分解を行うフィルインを考慮しない不完全 LU 分解前処理は，ILU(0) と呼ばれる．図 2.34 に，フィルインを考慮しない不完全 LU 分解前処理 (ILU(0)) を示す．ここで，m_{ii} は前処理行列 $\boldsymbol{M}$ の i 番目の対角要素である．

このアルゴリズムでは，下三角行列 $\boldsymbol{L}$ の対角成分が 1 であることを自明として，LU 分解を行う ILU 前処理は，解く問題によって分解が不安定になり，前処理行列の対角項に負の値が発生する場合がある．そのような問題には，ILU 分解前の前処理行列 $\boldsymbol{M}$ の対角項を，あらかじ

```
1: fill-in level = p, F = 0, M = A
2: for all i, j do
3:   if a_ij = 0 then
4:     f_ij = 1
5:   end if
6: end for
7: define level p fill-in of F
8: for k = 1, 2, 3, ..., n do
9:   for j = k + 1, k + 2, ..., n do
10:    if f_jk ≠ 0 then
11:      m_jk = m_jk / m_kk
12:    end if
13:    for i = k + 1, k + 2, ..., n do
14:      if f_ji ≠ 0 then
15:        m_ji = m_ji − m_jk m_ki
16:      end if
17:    end for
18:  end for
19: end for
```

Fig. 2.35 Incomplete LU(p) preconditioner

め修正係数を用いて定数倍することで，LU 分解の破綻を防ぐ方法がとられる場合がある．

前述した不完全 LU 分解前処理は，前処理行列の非零プロファイルを係数行列と同じものに固定して分解を行った．一般に数段のフィルインを考慮して非零プロファイルの数を増やせば，フィルインを考慮しない場合に比べて，前処理行列が係数行列の良い近似に近づくため，強い前処理能力をもつ．このような，p 段のフィルインを考慮した不完全 LU 分解前処理は，ILU(p) と呼ばれる．図 2.35 に，p 段のフィルインを考慮した不完全 LU 分解前処理を示す．ここで，m_{ii} は前処理行列 $\boldsymbol{M}$ の i 番目の対角要素，a_{ii} は係数行列 $\boldsymbol{A}$ の i 番目の対角要素，p はフィルインの段数，$\boldsymbol{F}$ はフィルインを決定するために用いる行列である．

1: **for** $i = 1, 2, 3, \ldots, n$ **do**
2: $\quad \boldsymbol{r}_i = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_i$
3: $\quad \boldsymbol{A}\boldsymbol{p} = \boldsymbol{r}_i$
4: $\quad \boldsymbol{x}_{i+1} = \boldsymbol{x}_i + \boldsymbol{p}$
5: **end for**

Fig. 2.36 Iterative refinement method

2.17 Iterative Refinement Method

iterative refinement method は，連立一次方程式 $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ の解の精度を改善する手法である [8]．図 2.36 に，反復改良法を示す．ここで，$\boldsymbol{r}$ は残差ベクトル，$\boldsymbol{p}$ は一時ベクトルである．iterative refinement method は，主に直接法の解の改良手法として採用される．また，線形問題に対し Newton-Raphson 法を適用した場合にも，iterative refinement method と同等の計算になる．

2.18 反復法における並列計算

並列反復法の計算は，アルゴリズムの性質から，行列ベクトル積，ベクトル内積，ベクトル和，前処理の並列計算を実現すればよい．これらの演算は，前述した領域分割法の枠組みに基づいて計算される．一方，並列反復法の前処理は，一般に分割領域ごとに局所化した前処理が適用される．これは，領域分割間にまたがる係数行列の要素を計算に考慮しようとすると，直接法的な計算アプローチが必要であり並列性が阻害されることから，前処理適用時の計算効率が大きく低下するためである．また，領域分割数が大きくなるに従って，分割領域間にまたがる係数行列の要素の数が大きくなると，前処理局所化の際に無視される要素の数が増大し，一般に対角スケーリング前処理相当の前処理性能に漸近する．

2.19 反復法による条件数の推定

連立一次方程式の係数行列の性質を評価する指標のひとつとして，条件数が挙げられる．条件数は，反復法の収束のしやすさを評価する重要な指標である．条件数は，式 (2.84) で定義される．

$$\kappa(\boldsymbol{A}) = \|\boldsymbol{A}\|\|\boldsymbol{A}^{-1}\| \tag{2.84}$$

ここで，ノルム $\|\boldsymbol{A}\|$ の定義は任意に定めてよい．このとき特に，ノルムの定義が 2 ノルムで，行列 $\boldsymbol{A}$ が正定値対象であれば，条件数は式 (2.85) のように，最大固有値と最小固有値の比と

なる．

$$\kappa(\boldsymbol{A}) = \frac{\lambda_{max}(\boldsymbol{A})}{\lambda_{min}(\boldsymbol{A})} \tag{2.85}$$

これまで，条件数の推定方法はいくつか提案されているが，本研究では大規模問題でも推定が可能な Krylov 部分空間を利用した方法を用いる．共役勾配法は，式 (2.86) に示すように反復中から得られる係数 α，β を用いて，Lanczos 法の第 m 回の反復までに得られる三重対角行列に対応する行列 $\boldsymbol{T}_m$ を生成することができる．この三重対角行列 $\boldsymbol{T}_m$ の最小固有値と最大固有値を求めると，条件数の推定を行うことができる．GMRES 法では，計算途中で得られるヘッセンベルグ行列の最小固有値と最大固有値を求めると，条件数の推定を行うことができる．

三重対角行列の固有値計算は，QR 法などを用いて解かれる．本研究では，数値計算ライブラリ LAPACK[19] のルーチン DSTEQR を用いて QR 法による固有値計算を行った．

$$\boldsymbol{T}_m = \begin{pmatrix} \frac{1}{\alpha_0} & \frac{\sqrt{\beta_0}}{\alpha_0} & & & & & \boldsymbol{0} \\ \frac{\sqrt{\beta_0}}{\alpha_0} & \frac{\beta_0}{\alpha_0} + \frac{1}{\alpha_1} & \frac{\sqrt{\beta_1}}{\alpha_1} & & & & \\ & \frac{\sqrt{\beta_1}}{\alpha_1} & \frac{\beta_1}{\alpha_1} + \frac{1}{\alpha_2} & \frac{\sqrt{\beta_2}}{\alpha_2} & & & \\ & & & \ddots & & & \\ & & & \frac{\sqrt{\beta_{m-3}}}{\alpha_{m-3}} & \frac{\beta_{m-3}}{\alpha_{m-3}} + \frac{1}{\alpha_{m-2}} & \frac{\sqrt{\beta_{m-2}}}{\alpha_{m-2}} & \\ \boldsymbol{0} & & & & \frac{\sqrt{\beta_{m-2}}}{\alpha_{m-2}} & \frac{\beta_{m-2}}{\alpha_{m-2}} + \frac{1}{\alpha_{m-1}} & \end{pmatrix} \tag{2.86}$$

この方法は共役勾配法の係数を用いるため，収束自体が不十分な場合には，最小固有値を大きく推定することになる．最小固有値を大きく推定することは，条件数を小さく推定することになるため，推定には十分な反復を行うことに注意する．

前処理付共役勾配法や前処理付 BiCGSTAB 法に対して，式 (2.86) の行列 $\boldsymbol{T}_m$ から条件数推定を行うと，前処理行列を適用した後の行列 $\boldsymbol{M}^{-1}\boldsymbol{A}$ の条件数を推定することができる．

2.20 反復法ソルバの理論演算性能

2.20.1 3×3 block Compressed Sparse Row (CSR) storage format

compressed sparse row (CSR) storage format は，有限要素解析プログラムにおいてよく用いられる疎行列格納形式のひとつである．block compressed sparse row (BCSR) storage format は，CSR 形式をブロック行列に適用するために拡張した手法である．3 次元構造解析においてソリッド要素を用いた場合，1 節点あたり 3 自由度の値をもつため， 3×3 のサイズのブロック行列を用いて疎行列を良く表すことができることがわかる．このようなブロック行列に適した疎行列格納形式として，3×3 BCSR storage format が挙げられる．

図 2.37 に，3×3 BCSR 形式の概略図を示す．ここで，$value$ は非零要素を保持する倍精度浮動小数点数型配列，$index$ は非零要素の行情報を保持する整数型配列，$item$ は非零要素の列

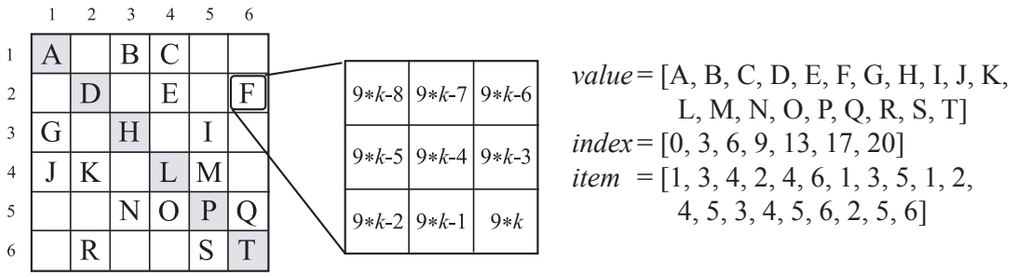


Fig. 2.37 Schematic representation of 3×3 BCSR storage format

情報を保持する整数型配列，k は一時整数型変数である．図 2.37 のように，それぞれの非零要素は 3×3 の小行列に格納されている．BCSR 形式は CSR 形式に比べ，ブロック小行列に注目することで，非零パターンの位置を示す整数型配列が少なくすむため，メモリアクセスの観点から効率よく計算できる．

2.20.2 SpMV kernel

疎行列ベクトル積は，共役勾配法などの反復法の計算時間の多くを占める重要な行列演算である．

図 2.38 に，3×3 BCSR 形式よる疎行列ベクトル積の fortran 90 プログラムカーネルを示す．ここで，$value$ は行列の非零要素，x は右辺ベクトル，y は解ベクトル，$index$ は非零要素の行情報を保持する配列，$item$ は非零要素の列情報を保持する配列，n は行列サイズ，i，j，jS，jE は一時整数型変数，$x1$，$x2$，$x3$，$y1$，$y2$，$y3$ は一時倍精度浮動小数点数型変数である．

ここで，3×3 BCSR 形式の疎行列ベクトル積カーネルの B/F 値 (byte per FLOPS ratio) は，以下のように計算できる．このとき，x，$x1$，$x2$，$x3$，y，$y1$，$y2$，$y3$ がキャッシュメモリ上で保持されることを仮定すると，最内側 j ループにおいて，9 つの倍精度浮動小数点数型 $value$ と 1 つの整数型配列 $item$ をロードする必要がある．ロードされる総データ量は 76 Bytes (倍精度浮動小数点数型 8 Bytes × 9 + 整数型配列 4 Bytes × 1) であり，総演算数は 18 FLOPS (和算 9 回 + 積算 9 回) である．以上から，3×3 BCSR 形式の疎行列ベクトル積カーネルの B/F 値は 4.22 (= 76 Bytes / 18 FLOPS) である．

同様に，3 次元構造解析において 1 節点あたり 6 自由度の値をもつ構造要素を用いた場合の，6×6 BCSR 形式の疎行列ベクトル積カーネルの B/F 値は 4.05 (= 292 Bytes / 72 FLOPS) である．

疎行列ベクトル積カーネルの B/F 値に基づけば，6×6 BCSR 形式の疎行列ベクトル積カーネルの方が，3×3 BCSR 形式の疎行列ベクトル積カーネルに比べ，高い演算性能を得ることができると考えられる．しかし，実際の演算性能はメモリのキャッシュミスやレジスタスピルな

```
!compute y=Ax
do i = 1,n
  jS = index(i-1) +1
  jE = index(i  )
  do j = jS, jE
    in = item(j)
    x1 = x(3*in-2)
    x2 = x(3*in-1)
    x3 = x(3*in  )
    y1 = y1 +value(9*j-8)*x1 +value(9*j-7)*x2 +value(9*j-6)*x3
    y2 = y2 +value(9*j-5)*x1 +value(9*j-4)*x2 +value(9*j-3)*x3
    y3 = y3 +value(9*j-2)*x1 +value(9*j-1)*x2 +value(9*j  )*x3
  enddo
  y(3*in-2) = y1
  y(3*in-1) = y2
  y(3*in  ) = y3
enddo
```

Fig. 2.38 Fortran90 program of SpMV kernel for $y = Ax$ in 3×3 BCSR storage format

どの影響を受ける場合がある．

2.21 結言

本章では，non-overlapping 型領域分割と overlapping 型領域分割の両者に対し，分割領域から得られる行列と解くべき連立一次方程式 $\boldsymbol{Ax} = \boldsymbol{b}$ の対応について述べた．さらに，non-overlapping 型領域分割と overlapping 型領域分割の両者に対し，Schur complement system による静的縮約を導出する過程を述べた．

また，有限要素離散化から得られる連立一次方程式を解くための線形ソルバに関し，直接法と反復法の両者について述べた．このとき，高い並列計算性能を実現するための手法として注目されている通信隠蔽型非定常反復法について述べた．次に，反復法を安定に解くための重要な技術である，反復法前処理について述べた．最後に，反復法ソルバによる条件数の推定と，反復法ソルバの理論演算性能について述べた．

第 3 章　線形ソルバの統一的フレームワークの提案

第 3 章は，単行本もしくは雑誌掲載等の形で刊行される予定があるため，インターネット公表できません.

第 4 章　悪条件問題に対する反復法前処理の提案

第 4 章は，単行本もしくは雑誌掲載等の形で刊行される予定があるため，インターネット公表できません.

第 5 章　強スケーリングに対する反復法前処理の提案

第 5 章は，単行本もしくは雑誌掲載等の形で刊行される予定があるため，インターネット公表できません.

第 6 章　3×3 ブロッキング構造要素の提案

6.1　緒言

シェル要素やビーム要素などの構造要素は，ソリッド要素などの連続体要素に比べて少ない自由度で複雑な構造の挙動を良く再現できることから，構造解析に広く用いられている．構造要素は解くべき問題を少ない自由度で表現できるので，計算時間やメモリ使用量などの計算コストを削減できることに優位性がある．構造要素が連続体要素と大きく異なる点として，連続体要素が各節点に並進 3 自由度をもつのに対し，Drilling 自由度を考慮した MITC4 シェル要素や Bernoulli-Euler 梁要素は，各節点に並進 3 自由度と回転 3 自由度の合計 6 自由度をもつことが挙げられる [20, 21, 22, 23, 24, 25, 26]．

これまでに，有限要素法から得られる行列は疎行列となり，各節点がもつ自由度に適したブロック小行列の疎行列格納形式を用いて格納されることを述べた．各節点に 3 自由度をもつ連続体要素と各節点に 6 自由度をもつ構造要素を，ひとつの解析モデルの中で併用する場合，疎行列格納形式が複雑になり，計算性能が劣化することが懸念されていた．さらに，並列計算の基盤となるライブラリプログラムや領域分割プログラムを，各節点の自由度が異なる場合の計算に対応させねばならない．

本章では，提案した線形ソルバの統一的フレームワークを効率的に利用するために，各節点に 6 自由度をもつ構造要素を 3×3 ブロック小行列に分割して格納する 3×3 ブロッキング構造要素を提案する．数値例では，提案手法が計算性能に与える影響について述べる．

6.2　3×3 ブロッキング構造要素の提案

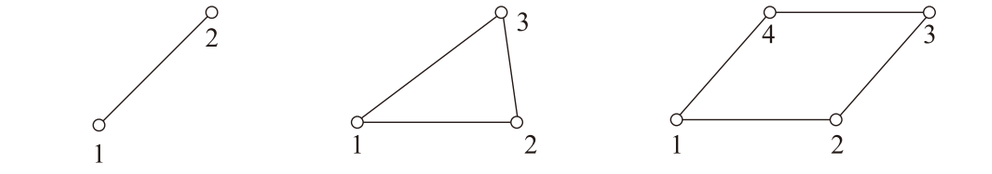
本章では，各節点に 6 自由度をもつ構造要素を 3×3 ブロック小行列に分割して格納する 3×3 ブロッキング構造要素を提案する [27]．提案手法は，有限要素解析プログラムに既に実装されているソリッド要素のためのデータ構造を利用する．提案手法による 2 節点梁要素，3 節点三角形シェル要素，4 節点四辺形シェル要素はそれぞれ，4 節点四面体要素，6 節点プリズム要素，8 節点六面体要素用のデータ構造を利用する．

図 6.1 に，従来の構造要素である，2 節点梁要素，3 節点三角形シェル要素，4 節点四辺形シェル要素を示す．梁要素の自由度の合計は 12，三角形シェル要素は 18，四角形シェル要素は 24 である．

図 6.2 に，提案手法の 3×3 ブロッキング構造要素である，2 つのダミー節点をもつ 4 節点梁要素，3 つのダミー節点をもつ 6 節点三角形シェル要素，4 つのダミー節点をもつ 8 節点四辺形シェル要素を示す．これら提案手法の構造要素は，従来の構造要素と同じ自由度を有している．

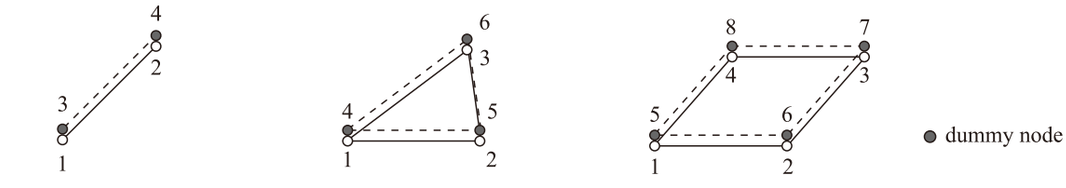
図 6.3 に，従来手法である 3 節点三角形シェル要素の概略図を示す．図 6.4 に，提案手法である 3 節点のダミー節点をもつ 6 節点三角形シェル要素の概略図を示す．ここで，u^i は並進自由度，θ^i は回転自由度で，上付き添字 i は節点番号を示す．従来手法である 3 節点三角形シェル要素は，各節点に並進 3 自由度，回転 3 自由度の合計 6 自由度を有している．一方，提案手法では，並進 3 自由度，回転 3 自由度をそれぞれ別の節点に割り当てる．特に提案手法では，並進 3 自由度を前半の 3 節点，回転 3 自由度を後半の 3 節点に割り当てている．また，ダミー節点を含む節点コネクティビティ情報は，要素の幾何情報を計算するために重要である．

提案手法は，領域分割に基づく並列有限要素法の領域分割においても有用性が挙げられる．従来の構造要素では，グラフ構造に基づいて領域分割を行う際に，節点あたりの自由度が異なるため，並列計算を効率的に行うためには，分割領域の演算量を等しくする必要があった．この手法の場合，ノードに自由度相当分の重みを考慮して計算しなければならないため，重み付きグラフの計算が必要となり，並列計算の基盤となるライブラリプログラムの大幅な修正が必要となる．また，この修正を経ても，行列ベクトル積のカーネル部分が複雑化するのを避けることはできない．



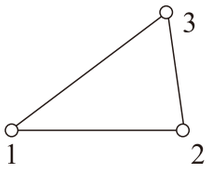
(a) 2-node beam element　(b) 3-node triangular shell element　(c) 4-node quadrilateral shell element

Fig. 6.1 Conventional structural elements



(d) 4-node beam element with 2 dummy nodes　(e) 6-node triangular shell element with 3 dummy nodes　(f) 8-node quadrilateral shell element with 4 dummy nodes

Fig. 6.2 3×3 DOF blocking structural elements

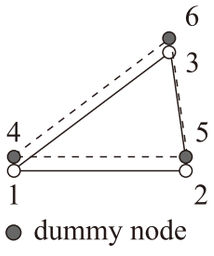


3 nodes × 6 DOFs/node (stored in 6×6 BCSR storage format)

$$\{ \quad \text{node 1} \quad , \quad \text{node 2} \quad , \quad \text{node 3} \quad \}$$

$$= \{ \quad \boldsymbol{u}^1 \quad \boldsymbol{\theta}^1 \quad , \quad \boldsymbol{u}^2 \quad \boldsymbol{\theta}^2 \quad , \quad \boldsymbol{u}^3 \quad \boldsymbol{\theta}^3 \quad \}$$

Fig. 6.3 3-node triangular shell element



6 nodes × 3 DOFs/node (stored in 3×3 BCSR storage format)

$$\{ \text{ node 1, node 2, node 3, node 4, node 5, node 6 } \}$$

$$= \{ \quad \boldsymbol{u}^1 \quad , \quad \boldsymbol{u}^2 \quad , \quad \boldsymbol{u}^3 \quad , \quad \boldsymbol{\theta}^4 \quad , \quad \boldsymbol{\theta}^5 \quad , \quad \boldsymbol{\theta}^6 \quad \}$$

Fig. 6.4 6-node triangular shell element

6.3 数値例

6.3.1 計算条件と計算環境

提案手法の有効性を示すために，数値例として，共役勾配法の疎行列ベクトル積部分のプロファイルを測定した．計算機として，奥田研究室 IC，奥田研究室 IC2，京コンピュータを用いた．奥田研究室 IC の計算機仕様を表 6.1 に示す．奥田研究室 IC2 の計算機仕様を表 6.2 に示す．京コンピュータの計算機仕様を表 6.3 に示す．

特に，京コンピュータは，82,944 ノード (663,552 コア) を有し，ピーク演算性能 10.62 PFLOPS をもつ大規模並列計算機であり，各ノードは，8 コア (2.0GHz/core) で，1 ノードあたりのピーク演算性能は 128 GFLOPS である．各ノードのピークメモリバンド幅は 64 GB/s であるため，B/F 値は 0.5 となる．また，STREAM ベンチマークで算定された実効メモリバンド幅は，各ノード 36 GB/s であり，B/F 値は 0.28 となる [28].

従って，STREAM ベンチマークで算定された実効メモリバンド幅と，前節で述べた 3×3 BCSR 形式の行列ベクトル積の計算カーネルの B/F 値より，この計算カーネルの理論性能はピーク演算性能の 6.63% (=0.28/4.22) となり，6×6 BCSR 形式を用いる場合，ピーク演算性能の 6.91% (=0.28/4.05) となる．また，京コンピュータにおける演算性能のプロファイルは，Fujitsu profiler を利用した．

Table 6.1 Specification of IC

Component	Configuration of a node
CPU	Xeon E5-2650v3 (10cores/CPU, 2.3GHz, 25 MB L3-Cache) × 2
Memory	768 GB (DDR4-2333 32GB × 24)
Peak performance	736 Gflops
Peak memory bandwidth	136.5 GB/sec
B/F rate (theoretically)	0.19
Compiler	GCC 4.8.5

Table 6.2 Specification of IC2

Component	Configuration of a node
CPU	Xeon E5-2695v3 (14cores/CPU, 2.3GHz, 35 MB L3-Cache) × 2
Memory	768 GB (DDR4-2333 32GB × 24)
Peak performance	1,030.4 GFlops
Peak memory bandwidth	136.5 GB/sec
B/F rate (theoretically)	0.13
Compiler	Intel Compiler 17.0.4

Table 6.3 Specification of the K-computer

Component	Configuration of a node
CPU	SPARC64 VIIIfx (8cores/node, 2.0GHz)
Memory	16 GB
Peak performance	128 Gflops
Peak memory bandwidth	64 GB/sec
B/F rate (theoretically)	0.50
B/F rate (STREAM)	0.28
Compiler	Fujitsu Compiler ver 1.2.0

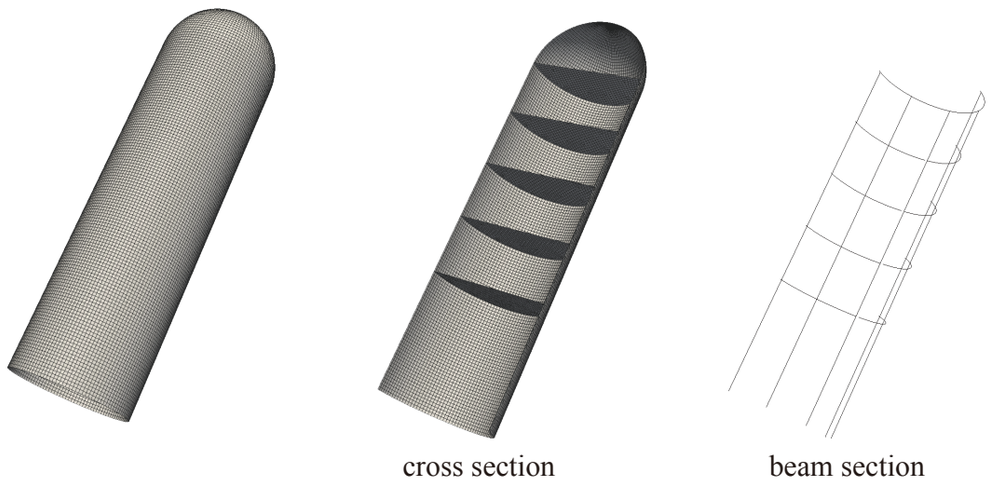


Fig. 6.5 Analysis mesh of a typical cylindrical column which has MITC4 shell elements and 2-node Bernoulli-Euler beam elements

6.3.2 計算結果

数値例として，図 6.5 に示す MITC4 シェル要素と Bernoulli-Euler 梁要素で構成された円筒形状のサイロモデルから得られた係数行列を用いた計算性能の測定結果を示す．境界条件として，モデル底面の変位を完全固定し，重力に相当する体積力をモデルの長手方向に負荷した．解析メッシュは，24,590 ノード, 31,180 要素 (シェル要素 29,620，梁要素 1,560), 147,540 自由度で，非零要素数は 8,010,936 （2.73%）である．計算は，OpenMP 1 スレッドによる逐次計算と，OpenMP 8 スレッドによる並列計算を実施した．この数値例の目的は，提案手法の 3×3 ブロッキング構造要素と，従来手法の 6×6 ブロッキング構造要素を使用した場合の，疎行列ベクトル積のカーネル部分の演算性能を評価することである．疎行列ベクトル積のカーネル部分は，問題の自由度には異存しないが，キャッシュヒット率に影響する．

表 6.4 に，奥田研究室 IC における，疎行列ベクトル積の計算カーネル部分の 1,000 反復あたりの計算時間を示す．表 6.5 に，奥田研究室 IC2 における，疎行列ベクトル積の計算カーネル部分の 1,000 反復あたりの計算時間を示す．表 6.6 に，京コンピュータにおける，疎行列ベクトル積の計算カーネル部分の 1,000 反復あたりの計算時間を示す．ここで，“Num. of cores” は CPU コア数，“Storage format” は適用した行列格納形式，“Storage format” は計算カーネル部分の 1,000 反復あたりの計算時間である．

表 6.4 より，計算カーネル部分を逐次計算した場合，従来手法である 6×6 ブロッキング構造要素の計算時間は 7.28 秒，提案手法である 3×3 ブロッキング構造要素の計算時間は 8.05 秒であり，従来手法の計算時間が 9.6% 短かった．計算カーネル部分を OpenMP 8 スレッドで

並列計算した場合，従来手法である 6×6 ブロッキング構造要素の計算時間は 2.46 秒，提案手法である 3×3 ブロッキング構造要素の計算時間は 2.47 秒であり，計算時間に大きな差は見られなかった．

表 6.5 より，計算カーネル部分を逐次計算した場合，従来手法である 6×6 ブロッキング構造要素の計算時間は 6.48 秒，提案手法である 3×3 ブロッキング構造要素の計算時間は 7.87 秒であり，従来手法の計算時間が 17% 短かかった．計算カーネル部分を OpenMP 8 スレッドで並列計算した場合，従来手法である 6×6 ブロッキング構造要素の計算時間は 1.18 秒，提案手法である 3×3 ブロッキング構造要素の計算時間は 1.21 秒であり，従来手法の計算時間が 2.1% 短かった．

表 6.6 より，計算カーネル部分を逐次計算した場合，従来手法である 6×6 ブロッキング構造要素の計算時間は 24.6 秒，提案手法である 3×3 ブロッキング構造要素の計算時間は 19.0 秒であり，計算時間を 23% 削減できた．計算カーネル部分を OpenMP 8 スレッドで並列計算した場合，従来手法である 6×6 ブロッキング構造要素の計算時間は 6.85 秒，提案手法である 3×3 ブロッキング構造要素の計算時間は 4.65 秒であり，計算時間を 32% 削減できた．

以上の結果から，Xeon 系の CPU で逐次計算をした場合，従来手法である 6×6 ブロッキング構造要素が優位である結果が得られた．計算カーネル部分を OpenMP 8 スレッドで並列計算した場合，従来手法である 6×6 ブロッキング構造要素と，提案手法である 3×3 ブロッキング構造要素の計算時間に大きな差異は見られなかった．

一方，SPARC 系の CPU では，逐次計算・スレッド並列計算とも，提案手法である 3×3 ブロッキング構造要素の方が，計算時間で優位であった．以上の結果から，本研究では，提案手法が優位であった SPARC 系の CPU に対し更なる検証を行うこととした．

Table 6.4 Execution time of SpMV kernel in the CG solver on IC computer

Num. of cores (OMP threads)	Storage format	SpMV time / 1,000 iters [sec]
1	3×3 blocking	8.05
	6×6 blocking	7.28
8	3×3 blocking	2.47
	6×6 blocking	2.46

Table 6.5 Execution time of SpMV kernel in the CG solver on IC2 computer

Num. of cores (OMP threads)	Storage format	SpMV time / 1,000 iters [sec]
1	3×3 blocking	7.87
	6×6 blocking	6.48
8	3×3 blocking	1.21
	6×6 blocking	1.18

Table 6.6 Execution time of SpMV kernel in the CG solver on K computer

Num. of cores (OMP threads)	Storage format	SpMV time / 1,000 iters [sec]
1	3×3 blocking	19.0
	6×6 blocking	24.6
8	3×3 blocking	4.65
	6×6 blocking	6.85

Table 6.7 Performance and memory throughput of SpMV kernel in the CG solver

Num. of cores (OMP threads)	Storage format	Performance [Mflops]	Performance / Peak [%]	Memory thrput. [GB/sec]	Memory thrput. / Peak [GB/sec]
1	3×3 blocking	1,007.	6.30	4.74	7.41
	6×6 blocking	795.	4.97	3.48	5.44
8	3×3 blocking	6,499.	5.08	29.7	46.4
	6×6 blocking	5,440.	4.25	23.2	36.2

表 6.7 に，疎行列ベクトル積の計算カーネル部分の演算性能の測定結果を示す．ここで，“Performance [Mflops]” は SpMV カーネルの演算性能，“Performance / Peak [%]” は理論演算性能に対する演算性能の割合，“Memory thrput. [GB/s]” はメモリスループット，“Memory thrput. / Peak [%]” は理論メモリバンド幅に対するメモリスループットの割合である．

計算カーネル部分を逐次計算した場合，従来手法である 6×6 ブロッキング構造要素の演算性能は理論演算性能の 4.97%，提案手法である 3×3 ブロッキング構造要素の演算性能は理論演算性能の 6.30% であった．提案手法は，STREAM ベンチマークに基づく理論性能 (6.63%) に対し，その 95.0% に相当する演算性能を実現することができた．計算カーネル部分を OpenMP 8 スレッドで並列計算した場合，従来手法である 6×6 ブロッキング構造要素の演算性能は理論演算性能の 4.25%，提案手法である 3×3 ブロッキング構造要素の演算性能は理論演算性能の 5.08% であった．この例では，STREAM ベンチマークに基づく理論性能 (6.63%) に対し，提案手法はその 76.6% に相当する演算性能を実現することができた．

理論演算性能の観点では，疎行列ベクトル積の計算カーネルの理論演算性能は，6×6 BCSR 形式による計算が効率的であることを述べた．

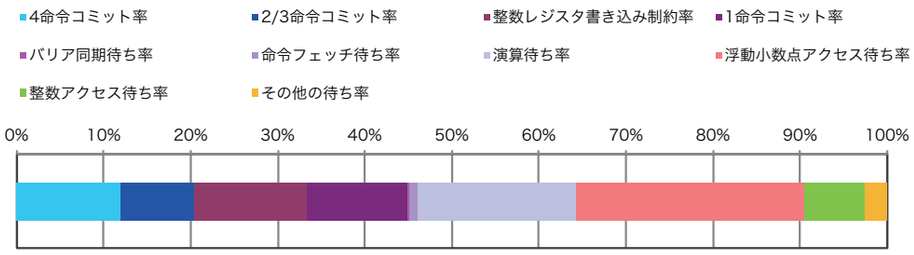


Fig. 6.6 Parcentage of execution time of 3×3 blocking SpMV kernel

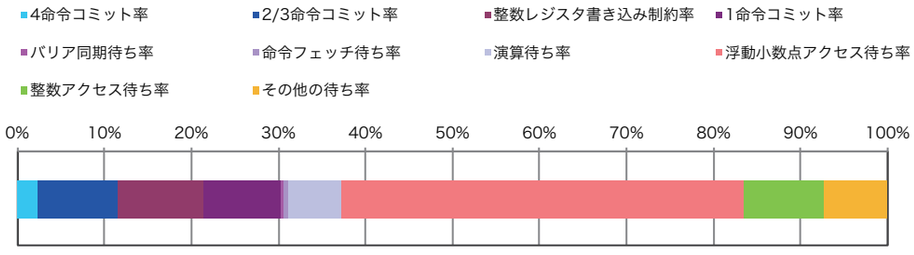


Fig. 6.7 Parcentage of execution time of 6×6 blocking SpMV kernel

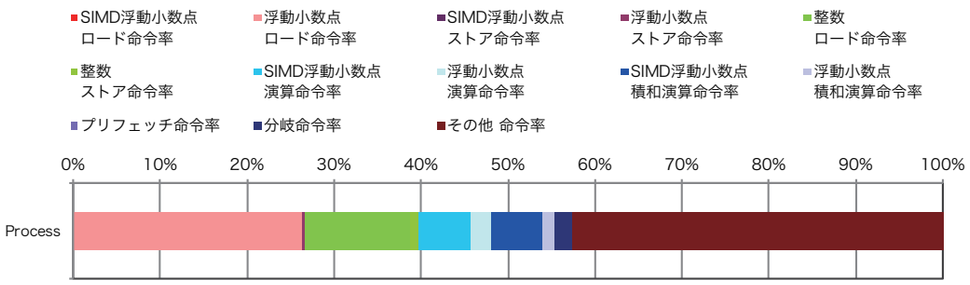


Fig. 6.8 Parcentage of instructions of 3×3 blocking SpMV kernel

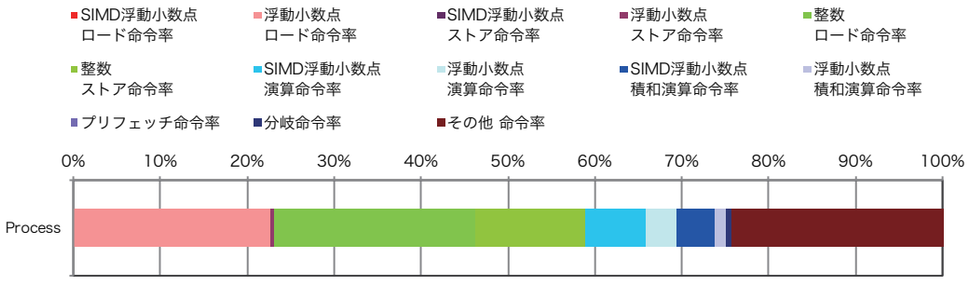


Fig. 6.9 Parcentage of instructions of 6×6 blocking SpMV kernel

図 6.6 に，3×3 ブロッキング構造要素のプロファイリング結果から，計算時間の内訳を示す．図 6.7 に，6×6 ブロッキング構造要素のプロファイリング結果から，計算時間の内訳を示す．図 6.8 に，3×3 ブロッキング構造要素のプロファイリング結果から，命令の内訳を示す．図 6.9 に，6×6 ブロッキング構造要素のプロファイリング結果から，命令の内訳を示す．図 6.6，図 6.7 より，OpenMP 8 スレッドで並列計算した場合，提案手法である 3×3 ブロッキング構造要素の計算時間の内訳のうち 25% が浮動小数点数アクセス待ち時間であり，従来手法である 6×6 ブロッキング構造要素の計算時間の内訳うち 46% が浮動小数点数アクセス待ち時間であることから，浮動小数点数アクセス待ち時間時間が 46% 増加したことが示された．図 6.8，図 6.9 より，OpenMP 8 スレッドで並列計算した場合，提案手法である 3×3 ブロッキング構造要素の命令内訳のうち 12% が整数ロード命令であり，従来手法である 6×6 ブロッキング構造要素の命令内訳のうち 23% が整数ロード命令であることから，整数ロード命令が 48% 増加したことが示された．

プロファイルを取得した部分は，疎行列ベクトル積のカーネルプログラム部分であるため，提案手法である 3×3 ブロッキング構造要素は，効率的に浮動小数点数にアクセスできていることがわかる．また，疎行列ベクトル積のカーネルプログラム部分で整数ロード命令が必要なのは，右辺ベクトルのインデックス計算のみであり，係数行列のインデックス計算はループ変数 i で計算することから，本来キャッシュ上に確保されていることが期待できる．しかしながら，6×6 ブロッキング構造要素では，3×3 ブロッキング構造要素に比べ，整数ロード命令が 2 倍ほど増加していることがわかる．以上の結果から，カーネルプログラム部分にも，本来必要のないメモリのストア・ロードが発生していることから，レジスタスピルが発生していることが考えられる．レジスタスピルは，カーネルの使用するレジスタ数がハードウェア制限を超過した場合，その超過分を一度メモリに退避させる挙動のことを指す．不要なメモリのストア・ロードが発生するため，演算性能に大きく影響を与える．

6.4 結言

本章では，各節点に 3 自由度をもつ連続体要素と各節点に 6 自由度をもつ構造要素が混在する有限要素解析を効率よく実施するために，3×3 ブロッキング構造要素を提案した．数値例より，以下の結果を得た．

- 提各節点あたりの自由度が異なる連続体要素と構造要素が混在する有限要素解析において，並列反復法ソルバの演算性能を劣化することなく計算できる．
- 共役勾配法の疎行列ベクトル積演算において，提案手法の 3×3 ブロッキング構造要素は，従来手法の 6×6 ブロッキング構造要素に比べ，高い演算性能を得ることができた．京コンピュータにおいて，提案手法の理論演算性能はおよそ 6.30% であった．提案手法の演算性能は，疎行列ベクトル積カーネルの理論最大性能に対し，並列スレッド数 1 のとき 95.0% で，並列スレッド数 8 のとき 76.6% であった．

第 7 章　産業界の実問題による解析例

第 7 章は，単行本もしくは雑誌掲載等の形で刊行される予定があるため，インターネット公表できません.

第 8 章　結言

第 8 章は，単行本もしくは雑誌掲載等の形で刊行される予定があるため，インターネット公表できません．

謝辞

本論文は，筆者が東京大学大学院新領域創成科学研究科 人間環境学専攻 博士課程における研究成果をまとめたものです．研究遂行にあたり，多くの方々に多大なご指導，ご協力を頂きましたこと，ここに厚く御礼申し上げます．

はじめに，指導教官である東京大学大学院新領域創成科学研究科 奥田洋司教授に深い感謝を申し上げます．奥田先生には，日頃より温かいご指導賜るとともに，国内外の会議や研究会，課外活動，起業など，多岐にわたる挑戦の機会を与えて頂きました．本研究生活で得た貴重な経験は，奥田先生の懐の深さがあってのものです．重ねまして，心より感謝申し上げます．

そして，副査をお引き受けくださいました東京大学情報基盤センタースーパーコンピューティング研究部門・東京大学大学院情報理工学系研究科兼担 中島研吾教授，東京大学生産技術研究所・東京大学大学院工学系研究科兼担 吉川暢宏教授，東京大学大学院新領域創成科学研究科 割澤伸一教授，東京大学大学院新領域創成科学研究科 橋本学講師に謹んで感謝の意を表します．橋本先生には，日頃より研究会や研究室ゼミにおいて，様々なご指導を頂きました．心より感謝申し上げます．

日頃より，研究生活をともにしてきました奥田研究室の皆様に，感謝申し上げます．心強い研究室の先輩として，合同会社 PExProCS 後藤和哉氏，富士通アドバンストテクノロジ株式会社 稲垣和久氏，株式会社構造計画研究所 三橋祐太氏に感謝申し上げます．研究室の同期として，井原遊氏に心より感謝申し上げます．研究室の後輩として，生野達大氏に感謝申し上げます．様々な議論を通し深い知見をご教示いただきましたこと，大変感謝しております．

井原氏とは，様々な場面を通して，刺激的な経験を共にすることができました．専門分野に関わらない，多岐にわたる深い知識と意思決定のバランス感覚には，大変敬服の思いです．

広義の隣の研究室として，いつも仲良くして頂きました，東京大学大学院工学系研究科 三目直登助教，内田英明氏，金子栄樹氏に感謝申し上げます．三目氏には，公私にわたり，様々な助言を頂きました．サウスベンドからシカゴまでの道程をご一緒できたことは，自身の経験の中でも大きな糧となっています．

大学生活における様々な面でご協力頂きました，渡辺夏実秘書，齊藤麻衣さん，藤枝夕美子さんに感謝申し上げます．プロムスなど研究生活以外の部分でも充実した大学生活を過ごせましたのは，御姉様方あってのものです．

株式会社 Ploutos 菅野朋典氏，株式会社科学計算総合研究所 堀江正信氏に感謝申し上げます．両氏とは，専門分野にとらわれない種々の議論によって，自身の視野を広げることができました．

本研究の一部は，JSPS 特別研究員奨励費 17J08069 の助成を受けたものです．また，本研究の一部は，理化学研究所のスーパーコンピュータ「京」を利用して得られたものです（課題番号：hp130091）．また，本研究の一部は，文部科学省ポスト「京」重点課題 8「近未来型ものづくりを先導する革新的設計・製造プロセスの開発」サブ課題 E「新材料に対応した高度成形・溶接シミュレータの研究開発」の元で実施したものです．本研究の第 6 章は，東京大学・株式会社 IHI との共同研究である「大型多管式反応器のための大規模構造解析技術に関する研究」の元で実施したものです．感謝の意を表します．本研究の第 7 章では，東京大学・鹿島建設株式会社との共同研究である「構造物－地盤連成の大規模非線形応答解析の高速化に関する研究」の元で作成されたモデルを使用させていただきました．深く感謝の意を表します．

最後に，これまで一貫して暖かく応援くださった両親と家族に心から感謝いたします．

本学博士課程終了後も，研究生活，私生活とも今以上に力強く邁進する所存です．これまでご協力くださった皆様に感謝の気持ちを申し上げ，謝辞にかえさせていただきます．

参考文献

[1] Patrick R Amestoy, Timothy A Davis, and Iain S Duff. An approximate minimum degree ordering algorithm. *SIAM Journal on Matrix Analysis and Applications*, Vol. 17, No. 4, pp. 886–905, 1996.

[2] Piotr Berman and Georg Schnitger. On the performance of the minimum degree ordering for gaussian elimination. *SIAM Journal on Matrix Analysis and Applications*, Vol. 11, No. 1, pp. 83–88, 1990.

[3] Elizabeth Cuthill and J McKee. Several strategies for reducing the bandwidth of matrices. *Sparse Matrices and Their Applications*, Vol. 1, pp. 157–166, 1972.

[4] Wai-Hung Liu and Andrew H Sherman. Comparative analysis of the cuthill–mckee and the reverse cuthill–mckee ordering algorithms for sparse matrices. *SIAM Journal on Numerical Analysis*, Vol. 13, No. 2, pp. 198–213, 1976.

[5] Heinz Rutishauser. The jacobi method for real symmetric matrices. *Numerische Mathematik*, Vol. 9, No. 1, pp. 1–10, 1966.

[6] BA Carre. The determination of the optimum accelerating factor for successive over-relaxation. *The computer journal*, Vol. 4, No. 1, pp. 73–78, 1961.

[7] Magnus Rudolph Hestenes and Eduard Stiefel. *Methods of conjugate gradients for solving linear systems*, Vol. 49. NBS, 1952.

[8] Yousef Saad. *Iterative methods for sparse linear systems*. Siam, 2003.

[9] Roger Fletcher. Conjugate gradient methods for indefinite systems. *Numerical analysis*, pp. 73–89, 1976.

[10] Peter Sonneveld. Cgs, a fast lanczos-type solver for nonsymmetric linear systems. *SIAM journal on scientific and statistical computing*, Vol. 10, No. 1, pp. 36–52, 1989.

[11] Henk A Van der Vorst. Bi-cgstab: A fast and smoothly converging variant of bi-cg for the solution of nonsymmetric linear systems. *SIAM Journal on scientific and Statistical Computing*, Vol. 13, No. 2, pp. 631–644, 1992.

[12] Shao-Liang Zhang. Gpbi-cg: Generalized product-type methods based on bi-cg for solving nonsymmetric linear systems. *SIAM Journal on Scientific Computing*, Vol. 18, No. 2, pp. 537–551, 1997.

[13] Youcef Saad and Martin H Schultz. Gmres: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM Journal on scientific and statistical computing*, Vol. 7, No. 3, pp. 856–869, 1986.

[14] William Gropp, Ewing Lusk, Nathan Doss, and Anthony Skjellum. A high-performance, portable implementation of the mpi message passing interface standard. *Parallel computing*, Vol. 22, No. 6, pp. 789–828, 1996.

[15] Pieter Ghysels and Wim Vanroose. Hiding global synchronization latency in the preconditioned conjugate gradient algorithm. *Parallel Computing*, Vol. 40, No. 7, pp. 224–238, 2014.

[16] Youcef Saad. Practical use of some krylov subspace methods for solving indefinite and nonsymmetric linear systems. *SIAM journal on scientific and statistical computing*, Vol. 5, No. 1, pp. 203–228, 1984.

[17] Gerard Meurant. Multitasking the conjugate gradient method on the cray x-mp/48. *Parallel Computing*, Vol. 5, No. 3, pp. 267–280, 1987.

[18] Siegfried Cools and Wim Vanroose. The communication-hiding pipelined bicgstab method for the efficient parallel solution of large unsymmetric linear systems. *arXiv preprint arXiv:1612.01395*, 2016.

[19] Edward Anderson, Zhaojun Bai, Jack Dongarra, Anne Greenbaum, Alan McKenney, Jeremy Du Croz, Sven Hammerling, James Demmel, C Bischof, and Danny Sorensen. Lapack: A portable linear algebra library for high-performance computers. In *Proceedings of the 1990 ACM/IEEE conference on Supercomputing*, pp. 2–11. IEEE Computer Society Press, 1990.

[20] Eduardo N Dvorkin and Klaus-Jürgen Bathe. A continuum mechanics based four-node shell element for general non-linear analysis. *Engineering computations*, Vol. 1, No. 1, pp. 77–88, 1984.

[21] Thomas JR Hughes and F Brezzi. On drilling degrees of freedom. *Computer Methods in Applied Mechanics and Engineering*, Vol. 72, No. 1, pp. 105–121, 1989.

[22] G Pimpinelli. An assumed strain quadrilateral element with drilling degrees of freedom. *Finite elements in analysis and design*, Vol. 41, No. 3, pp. 267–283, 2004.

[23] Craig S Long, Sannelie Geyer, and Albert A Groenwold. A numerical study of the effect of penalty parameters for membrane elements with independent rotation fields and penalized equilibrium. *Finite elements in analysis and design*, Vol. 42, No. 8, pp. 757–765, 2006.

[24] Junuthula Narasimha Reddy. *Mechanics of laminated composite plates and shells: theory and analysis*. CRC press, 2004.

[25] Henry TY Yang, S Saigal, A Masud, and RK Kapania. A survey of recent shell finite elements. *International Journal for Numerical Methods in Engineering*, Vol. 47, No.

1-3, pp. 101–127, 2000.

[26] Suresh Panda and R Natarajan. Analysis of laminated composite shell structures by finite element method. *Computers & Structures*, Vol. 14, No. 3, pp. 225–230, 1981.

[27] Naoki MORITA, Kazuo YONEKURA, Ichiro YASUZUMI, Mitsuyoshi TSUNORI, Gaku HASHIMOTO, and Hiroshi OKUDA. Development of 3× 3 dof blocking structural elements to enhance the computational intensity of iterative linear solver. *Mechanical Engineering Letters*, Vol. 2, pp. 16–00082, 2016.

[28] Stream: Sustainable memory bandwidth in high performance computers. `https://www.cs.virginia.edu/stream/`. 2017 年 10 月 17 日閲覧.

[29] Youcef Saad. *Numerical methods for large eigenvalue problems*. Manchester University Press, 1992.

[30] Louis Komzsik. *The Lanczos method: evolution and application*. SIAM, 2003.

[31] 大崎順彦. 新・地震動のスペクトル解析入門, 1994.

付録

A. 非線形有限要素法による構造解析

A.1 連続体力学

A.1.1 連続体の運動

連続体の運動を記述するために，ある基準時刻 t_0 において領域 Ω_{t_0} を占める物体の配置を基準として考え，各物質点の位置ベクトルを $\boldsymbol{X}$ で表す．これを，基準配置という．現時刻 t_p において，物質点 $\boldsymbol{X}$ が物質点 $\boldsymbol{x}$ に移動し，物体が領域 Ω_{t_p} を占めるとするとき，その変位ベクトル $\boldsymbol{u}$ を，

$$\boldsymbol{u} = \boldsymbol{x} - \boldsymbol{X} \tag{A.1}$$

で定義する．さらに，物質点の速度ベクトル，加速度ベクトルは，それぞれ $\boldsymbol{v} = \dot{\boldsymbol{u}}$，$\boldsymbol{a} = \ddot{\boldsymbol{u}}$ として定義する．これらの関係を図 A.1 に示す．

A.1.2 変形

基準時刻 t_0 における位置ベクトル $\boldsymbol{X}$ と現時刻 t_p における位置ベクトル $\boldsymbol{x}$ を関数 $f(\boldsymbol{X}, t_p)$ を用いて，

$$\boldsymbol{x} = f(\boldsymbol{X}, t_p) \tag{A.2}$$

と表す．式 (A.2) は，物質表示，または Lagrange 表示と呼ばれる．

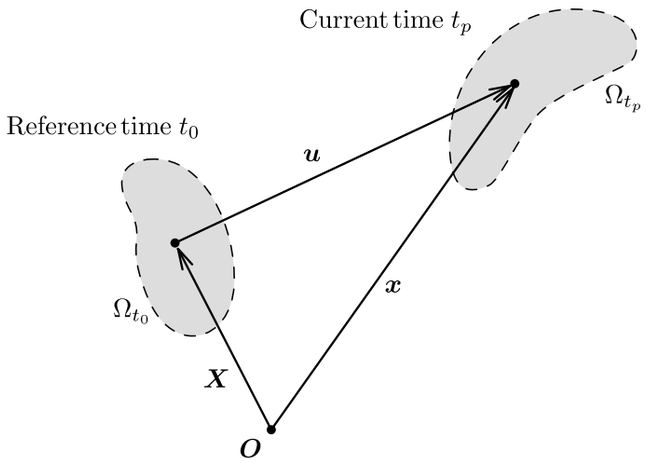


Fig. A.1 Displacement of material points

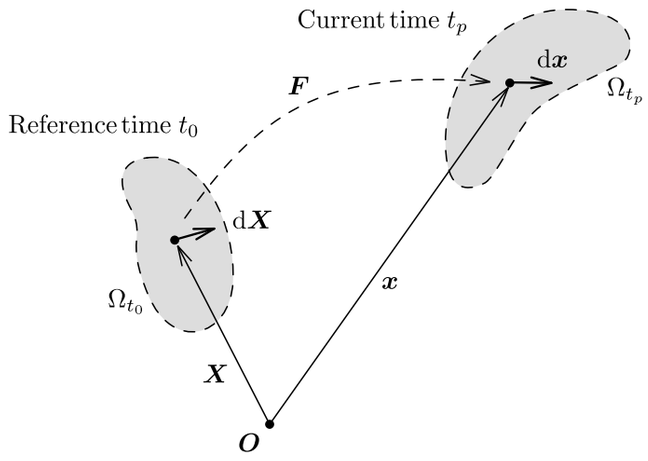


Fig. A.2 Schematic representation of the deformation gradient tensor $\boldsymbol{F}$

基準時刻 t_0 において，ある物質点 $\boldsymbol{X}$ およびその近傍点 $\boldsymbol{X}+\mathrm{d}\boldsymbol{X}$ を考える．$\mathrm{d}\boldsymbol{X}$ が微小であれば，

$$\mathrm{d}\boldsymbol{x} = f(\boldsymbol{X}+\mathrm{d}\boldsymbol{X}, t_p) - f(\boldsymbol{X}, t_p) \tag{A.3}$$

は $\mathrm{d}\boldsymbol{X}$ に対して線形関係にあると考えられるので，

$$\mathrm{d}\boldsymbol{x} = \boldsymbol{F}\cdot\mathrm{d}\boldsymbol{X} \tag{A.4}$$

なる線形変換を定義する．これらの関係を図 A.2 に示す．

このとき，変形勾配テンソル $\boldsymbol{F}$ を，成分表示すれば

$$\boldsymbol{F} = \frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}\boldsymbol{X}} = \frac{\partial x_i}{\partial X_j}\boldsymbol{e}_i\otimes\boldsymbol{e}_j = F_{ij}\boldsymbol{e}_i\otimes\boldsymbol{e}_j \tag{A.5}$$

と表される．

変形勾配テンソル $\boldsymbol{F}$ は，一般に対称ではないが，変形前後で物質点が一対一に対応することを考えると，$\det\boldsymbol{F}=0$ であってはならないため，必ず逆関係

$$\mathrm{d}\boldsymbol{X} = \boldsymbol{F}^{-1}\cdot\mathrm{d}\boldsymbol{x} \tag{A.6}$$

が存在する．

A.1.3 ひずみ

材料力学の範囲では，変形は微小として考えられ，変形前後で境界条件の大きさや向きが変化しないこととして取り扱ってきた．この変形の尺度は，垂直ひずみ（微小ひずみ）として知られている．ひずみ測度は本来，剛体運動に対して不変でなければならないが，垂直ひずみは回転の剛体運動に関して不変ではない性質が知られているため，ここでは有限変形に対し適切なひずみ測度の導出を行う．

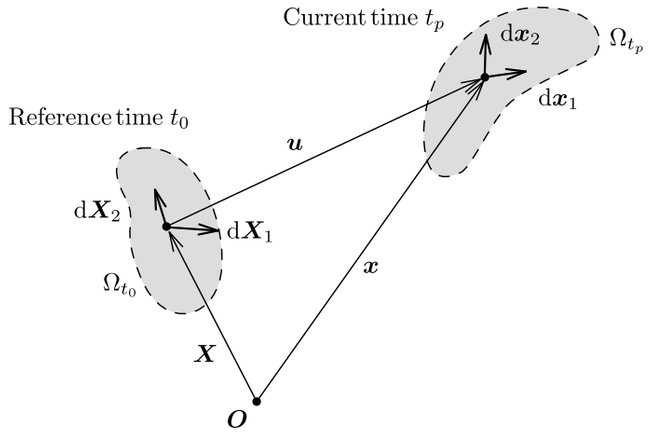


Fig. A.3 Deformation of infinitesimal vectors

基準時刻 t_0 において，図 A.3 に示すように，ある物質点を起点とする任意の微小ベクトル $\mathrm{d}\boldsymbol{X}_1$，$\mathrm{d}\boldsymbol{X}_2$ を考える．これらに対応する現時刻 t_p における微小ベクトルを $\mathrm{d}\boldsymbol{x}_1$，$\mathrm{d}\boldsymbol{x}_2$ と与える．これらの微小ベクトルに対し変形前後の内積変化を考えると，式 (A.4) より，

$$\begin{aligned}
\mathrm{d}\boldsymbol{x}_1 \cdot \mathrm{d}\boldsymbol{x}_2 - \mathrm{d}\boldsymbol{X}_1 \cdot \mathrm{d}\boldsymbol{X}_2 &= (\boldsymbol{F} \cdot \mathrm{d}\boldsymbol{X}_1) \cdot (\boldsymbol{F} \cdot \mathrm{d}\boldsymbol{X}_2) - \mathrm{d}\boldsymbol{X}_1 \cdot \mathrm{d}\boldsymbol{X}_2 \\
&= (\mathrm{d}\boldsymbol{X}_1 \cdot \boldsymbol{F}^t) \cdot (\boldsymbol{F} \cdot \mathrm{d}\boldsymbol{X}_2) - \mathrm{d}\boldsymbol{X}_1 \cdot \mathrm{d}\boldsymbol{X}_2 \\
&= \mathrm{d}\boldsymbol{X}_1 \cdot (\boldsymbol{F}^t \cdot \boldsymbol{F}) \cdot \mathrm{d}\boldsymbol{X}_2 - \mathrm{d}\boldsymbol{X}_1 \cdot \mathrm{d}\boldsymbol{X}_2 \\
&= \mathrm{d}\boldsymbol{X}_1 \cdot (\boldsymbol{F}^t \cdot \boldsymbol{F} - \boldsymbol{I}) \cdot \mathrm{d}\boldsymbol{X}_2 \\
&= \mathrm{d}\boldsymbol{X}_1 \cdot (\boldsymbol{C} - \boldsymbol{I}) \cdot \mathrm{d}\boldsymbol{X}_2
\end{aligned} \tag{A.7}$$

が得られる．このとき，右 Cauchy-Green 変形テンソルを $\boldsymbol{C} = \boldsymbol{F}^t \cdot \boldsymbol{F}$ で定義すると，Green-Lagrange ひずみ $\boldsymbol{E}$ は，右 Cauchy-Green 変形テンソル $\boldsymbol{C}$ を用いて，

$$\boldsymbol{E} = \frac{1}{2}(\boldsymbol{C} - \boldsymbol{I}) = \frac{1}{2}(\boldsymbol{F}^t \cdot \boldsymbol{F} - \boldsymbol{I}) \tag{A.8}$$

で定義される．

次に，Green-Lagrange ひずみ $\boldsymbol{E}$ は，変位 $\boldsymbol{u}$ を用いてどう表記できるか考える．ここで，変位勾配テンソル $\boldsymbol{Z}$ を式 (A.9) で定義する．

$$\boldsymbol{Z} = \boldsymbol{u} \otimes \nabla_{\boldsymbol{X}} = \frac{\partial u_i}{\partial X_j} \boldsymbol{e}_i \otimes \boldsymbol{e}_j \tag{A.9}$$

変位勾配テンソル $\boldsymbol{Z}$ に対し，以下の関係

$$\boldsymbol{Z} \cdot \mathrm{d}\boldsymbol{X} = \left(\frac{\partial \boldsymbol{u}}{\partial X_j} \otimes \boldsymbol{e}_j \right) \cdot \mathrm{d}X_i \boldsymbol{e}_i = \frac{\partial \boldsymbol{u}}{\partial X_i} \mathrm{d}X_i = \mathrm{d}\boldsymbol{u} \tag{A.10}$$

を考えると，式 (A.4) から，式 (A.11)，式 (A.12) の関係が得られる．

$$(\boldsymbol{I} + \boldsymbol{Z}) \cdot \mathrm{d}\boldsymbol{X} = \mathrm{d}\boldsymbol{X} + \mathrm{d}\boldsymbol{u} = \mathrm{d}(\boldsymbol{X} + \boldsymbol{u}) = \mathrm{d}\boldsymbol{x} \tag{A.11}$$

$$\boldsymbol{F} = \boldsymbol{I} + \boldsymbol{Z} \tag{A.12}$$

以上より，Green-Lagrange ひずみ $\boldsymbol{E}$ は，変位 $\boldsymbol{u}$ を用いて式 (A.13) で表される．

$$\begin{aligned}
\boldsymbol{E} &= \frac{1}{2}(\boldsymbol{C} - \boldsymbol{I}) \\
&= \frac{1}{2}(\boldsymbol{F}^t \cdot \boldsymbol{F} - \boldsymbol{I}) \\
&= \frac{1}{2}\{(\boldsymbol{I}^t + \boldsymbol{Z}^t) \cdot (\boldsymbol{I} + \boldsymbol{Z}) - \boldsymbol{I}\} \\
&= \frac{1}{2}(\boldsymbol{Z} + \boldsymbol{Z}^t + \boldsymbol{Z}^t\boldsymbol{Z}) \\
&= \frac{1}{2}\{\boldsymbol{u} \otimes \nabla_{\boldsymbol{X}} + (\boldsymbol{u} \otimes \nabla_{\boldsymbol{X}})^t + (\boldsymbol{u} \otimes \nabla_{\boldsymbol{X}})^t \cdot (\boldsymbol{u} \otimes \nabla_{\boldsymbol{X}})\} \\
&= \frac{1}{2}\left(\frac{\partial u_i}{\partial X_j} + \frac{\partial u_j}{\partial X_i} + \frac{\partial u_k}{\partial X_i}\frac{\partial u_k}{\partial X_j}\right)\boldsymbol{e}_i \otimes \boldsymbol{e}_j
\end{aligned} \tag{A.13}$$

従って，Green-Lagrange ひずみ $\boldsymbol{E}$ を成分表示すれば，式 (A.14) で表される．

$$E_{ij} = \frac{1}{2}\left(\frac{\partial u_i}{\partial X_j} + \frac{\partial u_j}{\partial X_i} + \frac{\partial u_k}{\partial X_i}\frac{\partial u_k}{\partial X_j}\right) \tag{A.14}$$

ここで，Green-Lagrange ひずみ $\boldsymbol{E}$ は，回転の剛体運動に関して不変であることを確認する．一般に特異でないテンソルは極分解可能であるから，変形勾配テンソル $\boldsymbol{F}$ は，直交テンソル $\boldsymbol{R}$ と正定値対称テンソル $\boldsymbol{U}$ を用いて，

$$\boldsymbol{F} = \boldsymbol{R} \cdot \boldsymbol{U} \tag{A.15}$$

のように表される．

このとき，右 Cauchy-Green 変形テンソル $\boldsymbol{C} = \boldsymbol{F}^t \cdot \boldsymbol{F}$ に式 (A.15) を代入すると，

$$\boldsymbol{C} = \boldsymbol{U} \cdot \boldsymbol{R}^t \cdot \boldsymbol{R} \cdot \boldsymbol{U} = \boldsymbol{U}^2 \tag{A.16}$$

が得られることから，Green-Lagrange ひずみ $\boldsymbol{E}$ は，回転の剛体運動に関して不変であることが示された．

さて，ある変形が微小と考えられるのであれば，変位の 2 次の項を無視することができる．例えば，Green-Lagrange ひずみ $\boldsymbol{E}$ に対し微小変形を考えるとき，式 (A.14) の二次の項を無視すれば，

$$E_{ij} = \frac{1}{2}\left(\frac{\partial u_i}{\partial X_j} + \frac{\partial u_j}{\partial X_i}\right) \tag{A.17}$$

となり，微小ひずみテンソルに一致する．

A.1.4 質量保存則

物体の質量 m の保存則は，基準時刻 t_0 における質量密度 ρ_{t_0} と，現時刻 t_p における質量密度 ρ_{t_p} を用いて

$$m = \int_{\Omega_{t_0}} \rho_{t_0} \mathrm{d}V = \int_{\Omega_{t_p}} \rho_{t_p} \mathrm{d}v = const. \tag{A.18}$$

で与えられる．ここで，$\mathrm{d}V$ は Ω_{t_0} の微小体積要素，$\mathrm{d}v$ は Ω_{t_p} の微小体積要素である．

以上から，$J = \det\boldsymbol{F}$ とおくと，微小体積要素について $\mathrm{d}v = J\mathrm{d}V$ の関係から

$$\int_{\Omega_{t_0}} \rho_{t_0}\mathrm{d}v = \int_{\Omega_{t_p}} \rho_{t_p} J\mathrm{d}V \tag{A.19}$$

が与えられる．式 (A.19) より，質量密度について式 (A.20) の関係が成り立つ．

$$\rho_{t_0} = \rho_{t_p} J \tag{A.20}$$

A.1.5　運動量保存則

物体に作用する力として，単位質量あたりの物体力ベクトル $\boldsymbol{g}$ と，表面力ベクトル $\boldsymbol{t}$ を考える．このとき，$\rho_{t_p}\boldsymbol{g}$ は単位体積あたりに作用する力であり，$\boldsymbol{t}$ は単位面積あたりに作用する力である．物体全体におけるこれらの和は，運動量の速度と等しく，式 (A.21) が成り立つ．これを，Euler の第 1 運動法則という．

$$\left(\int_{\Omega_{t_p}} \rho_{t_p}\boldsymbol{v}\mathrm{d}v\right)^{\cdot} = \int_{\Omega_{t_p}} \rho_{t_p}\boldsymbol{g}\mathrm{d}v + \int_{\Gamma_{t_p}} \boldsymbol{t}\mathrm{d}s \tag{A.21}$$

ここで，$\dot{\boldsymbol{v}} = \boldsymbol{a}$ とおき，Reynolds の輸送定理を用いれば，式 (A.21) は

$$\int_{\Omega_{t_p}} \rho_{t_p}(\boldsymbol{a} - \boldsymbol{g})\mathrm{d}v = \int_{\Gamma_{t_p}} \boldsymbol{t}\mathrm{d}s \tag{A.22}$$

と表すことができる．

次に，角運動量保存則について考える．物体の原点に関する物体力のモーメントおよび表面力のモーメントの和は，運動量モーメントの速度と等しく，式 (A.23) が成り立つ．これを，Euler の第 2 運動法則という．

$$\left(\int_{\Omega_{t_p}} \boldsymbol{x} \times \rho_{t_p}\boldsymbol{v}\mathrm{d}v\right)^{\cdot} = \int_{\Omega_{t_p}} \boldsymbol{x} \times \rho_{t_p}\boldsymbol{g}\mathrm{d}v + \int_{\Gamma_{t_p}} \boldsymbol{x} \times \boldsymbol{t}\mathrm{d}s \tag{A.23}$$

ここで，$\dot{\boldsymbol{x}} \times \boldsymbol{v} = \boldsymbol{v} \times \boldsymbol{v} = \boldsymbol{0}$ を考慮し，Reynolds の輸送定理を用いれば，式 (A.23) は

$$\int_{\Omega_{t_p}} \boldsymbol{x} \times \rho_{t_p}\boldsymbol{a}\mathrm{d}v = \int_{\Omega_{t_p}} \boldsymbol{x} \times \rho_{t_p}\boldsymbol{g}\mathrm{d}v + \int_{\Gamma_{t_p}} \boldsymbol{x} \times \boldsymbol{t}\mathrm{d}s \tag{A.24}$$

と表すことができる．

A.1.6　応力

図 A.4 に示すように、現配置における物体内の仮想表面上の微小な面要素 $\mathrm{d}s$ に作用する力が $\mathrm{d}\boldsymbol{f}$ とすると，応力ベクトル $\boldsymbol{t_n}$ は

$$\boldsymbol{t_n} = \mathrm{d}\boldsymbol{f}/\mathrm{d}s \tag{A.25}$$

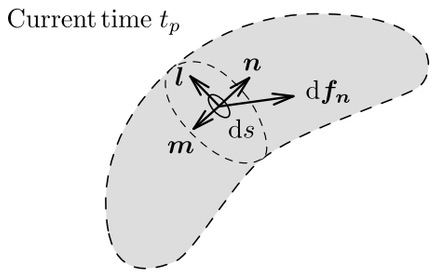


Fig. A.4 Schematic representation of the force loading on the virtual surface in the body

で与えられる．$\mathrm{d}\boldsymbol{f}$，$\boldsymbol{t_n}$ の下付添字 $\boldsymbol{n}$ は，仮想表面の外向単位法線ベクトル $\boldsymbol{n}$ に依存して定まることを示している．

ここで，仮想表面内において直交する単位ベクトル $\boldsymbol{l}$，$\boldsymbol{m}$ を考えると，応力ベクトル $\boldsymbol{t_n}$ は，式 (A.26) のように分解される．

$$\boldsymbol{t_n} = T_{nn}\boldsymbol{n} + T_{nl}\boldsymbol{l} + T_{nm}\boldsymbol{m} \tag{A.26}$$

ここで，基底の組 $(\boldsymbol{n}, \boldsymbol{l}, \boldsymbol{m})$ を，正規直行基底の組 $(\boldsymbol{e}_1, \boldsymbol{e}_2, \boldsymbol{e}_3)$ に一致させると，式 (A.27) を得る．

$$\boldsymbol{t}_i = T_{ij}\boldsymbol{e}_j \tag{A.27}$$

式 (A.27) を書き下せば，式 (A.28) を得る．

$$\begin{aligned} \boldsymbol{t}_1 &= T_{11}\boldsymbol{e}_1 + T_{12}\boldsymbol{e}_2 + T_{13}\boldsymbol{e}_3 \\ \boldsymbol{t}_2 &= T_{21}\boldsymbol{e}_1 + T_{22}\boldsymbol{e}_2 + T_{23}\boldsymbol{e}_3 \\ \boldsymbol{t}_3 &= T_{31}\boldsymbol{e}_1 + T_{32}\boldsymbol{e}_2 + T_{44}\boldsymbol{e}_3 \end{aligned} \tag{A.28}$$

これらによって，Cauchy 応力テンソル $\boldsymbol{T}$ は，式 (A.29) で与えられる．

$$\boldsymbol{T} = T_{ij}\boldsymbol{e}_i \otimes \boldsymbol{e}_j \tag{A.29}$$

また，正規直行基底で張られる 3 面と仮想表面の外向単位法線ベクトル $\boldsymbol{n}$ に垂直な面で構成される微小四面体（Cauchy の四面体）に対し，Euler の第 1 運動法則を適用すると，応力ベクトル $\boldsymbol{t_n}$ と外向単位法線ベクトル $\boldsymbol{n}$ の線形関係

$$\boldsymbol{t_n} = \boldsymbol{T}^t \cdot \boldsymbol{n} \tag{A.30}$$

が得られる．これを，Cauchy の公式という．

Cauchy 応力テンソル $\boldsymbol{T}$ は，現配置において定義される応力であり，真応力と呼ばれる．しかし，有限変形問題においては，未知な配置で定義される応力よりも，基準配置などの既知の配置に基づく応力テンソルを用いると，都合の良い場合がある．ここでは，Kirchhoff 応力テンソル，第 1Piola-Kirchhoff 応力テンソル，第 2Piola-Kirchhoff 応力テンソルについて述べる．

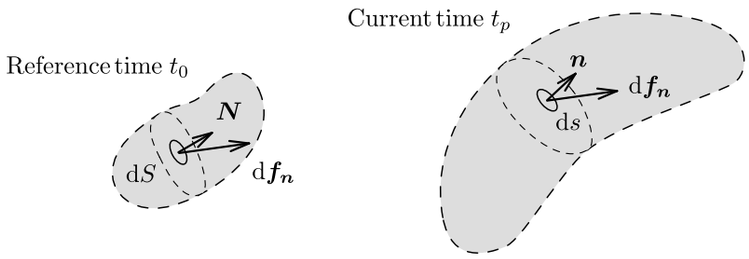


Fig. A.5 Schematic representation of the fitst Piola-Kirchhoff stress tensor $\boldsymbol{P}$

まず，Kirchhoff 応力テンソル $\hat{\boldsymbol{T}}$ は，時刻 t_0（基準配置）から時刻 t_p（現配置）への体積変化率 J を用いて，式 (A.31) で定義される．

$$\hat{\boldsymbol{T}} = J\boldsymbol{T} \tag{A.31}$$

次に，第 1Piola-Kirchhoff 応力テンソル $\boldsymbol{P}$ は，現配置（法線ベクトル $\boldsymbol{n}$ と微小面要素 $\mathrm{d}s$）での微小な面要素に作用する力ベクトルを，基準配置（法線ベクトル $\boldsymbol{N}$ と微小面要素 $\mathrm{d}S$）に並行移動させたときの関係を用いて，式 (A.32) 定義される．この概略図を図 A.5 に示す．また，第 1Piola-Kirchhoff 応力テンソル $\boldsymbol{P}$ は，公称応力とも呼ばれる．

$$\mathrm{d}\boldsymbol{f} = \boldsymbol{P}^t \cdot \boldsymbol{N}\mathrm{d}S \tag{A.32}$$

ここで，Nanson の公式

$$\boldsymbol{F}^t \cdot \boldsymbol{n}\mathrm{d}s = J\boldsymbol{N}\mathrm{d}S \tag{A.33}$$

と，式 (A.34) から，第 1Piola-Kirchhoff 応力テンソル $\boldsymbol{P}$ と Cauchy 応力テンソル $\boldsymbol{T}$ の関係式 (A.35) を得る．

$$\boldsymbol{P}^t \cdot \boldsymbol{N}\mathrm{d}S = \mathrm{d}\boldsymbol{f} = \boldsymbol{T}^t \cdot \boldsymbol{n}\mathrm{d}s \tag{A.34}$$

$$\boldsymbol{T} = \frac{1}{J}\boldsymbol{F} \cdot \boldsymbol{P} \tag{A.35}$$

最後に，第 2Piola-Kirchhoff 応力テンソル $\boldsymbol{S}$ は，現配置（法線ベクトル $\boldsymbol{n}$ と微小面要素 $\mathrm{d}s$）での微小な面要素に作用する力ベクトルを，変形勾配テンソルの逆変換 $\boldsymbol{F}^{-1}$ によって基準配置に写像し，基準配置（法線ベクトル $\boldsymbol{N}$ と微小面要素 $\mathrm{d}S$）の微小面要素との関係を用いて，式 (A.36) で定義される．この概略図を図 A.6 に示す．

$$\boldsymbol{F}^{-1} \cdot \mathrm{d}\boldsymbol{f} = \boldsymbol{S}^t \cdot \boldsymbol{N}\mathrm{d}S \tag{A.36}$$

ここで，式 (A.36) に対し，Nanson の公式 (A.33) から，式 (A.37) を得る．

$$\boldsymbol{T} = \frac{1}{J}\boldsymbol{F} \cdot \boldsymbol{S} \cdot \boldsymbol{F}^t \tag{A.37}$$

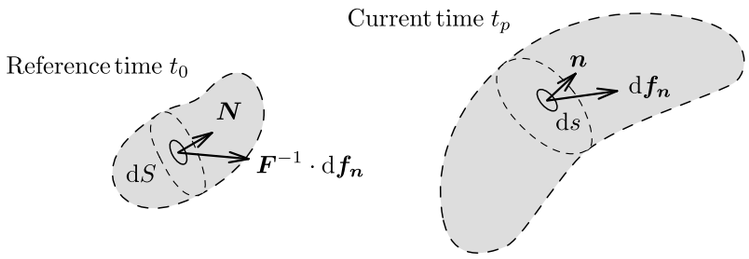


Fig. A.6 Schematic representation of the second Piola-Kirchhoff stress tensor $\boldsymbol{S}$

また，第 1Piola-Kirchhoff 応力テンソル $\boldsymbol{P}$ と第 2Piola-Kirchhoff 応力テンソル $\boldsymbol{S}$ は，

$$\boldsymbol{P} = \boldsymbol{S}\cdot\boldsymbol{F}^t \tag{A.38}$$

のように表される．

A.1.7　平衡方程式

これまでに，いくつかの応力の定義を行った．ここで一度 Cauchy 応力テンソル $\boldsymbol{T}$ に立ち戻り，Cauchy の公式 (A.30) と Euler の第 1 運動法則 (A.21) から平衡方程式を得ることを考える．Euler の第 1 運動法則 (A.21) の $\boldsymbol{t}$ を，Cauchy の公式 (A.30) の $\boldsymbol{t_n}$ に置き換えて，式 (A.39) を得る．

$$\int_{\Omega_{t_p}} \rho_{t_p}(\boldsymbol{a}-\boldsymbol{g})\mathrm{d}v = \int_{\Gamma_{t_p}} \boldsymbol{t_n}\mathrm{d}s \tag{A.39}$$

式 (A.39) に対し，$\boldsymbol{t_n} = \boldsymbol{T}^t\cdot\boldsymbol{n}$ とおき，発散定理

$$\int_V \nabla\cdot\boldsymbol{X}\mathrm{d}V = \int_S \boldsymbol{X}^t\cdot\boldsymbol{n}\mathrm{d}S \tag{A.40}$$

を適用して，式 (A.41) を得る．

$$\int_{\Omega_{t_p}} \rho_{t_p}(\boldsymbol{a}-\boldsymbol{g})\mathrm{d}v = \int_{\Omega_{t_p}} \nabla_{\boldsymbol{x}}\cdot\boldsymbol{T}\mathrm{d}v \tag{A.41}$$

式 (A.41) は，物体の任意の領域において成立することから，平衡方程式 (A.42) が得られる．平衡方程式は，Cauchy の第 1 運動法則としても知られている．

$$\rho_{t_p}\boldsymbol{a} = \nabla_{\boldsymbol{x}}\cdot\boldsymbol{T} + \rho_{t_p}\boldsymbol{g} \tag{A.42}$$

A.1.8　応力の対称性

次に，Cauchy の公式 (A.30) と Euler の第 2 運動法則 (A.23) から応力の対称性を示すことを考える．Euler の第 2 運動法則 (A.23) の $\boldsymbol{t}$ を，Cauchy の公式 (A.30) の $\boldsymbol{t_n}$ に置き換えて，式 (A.43) を得る．

$$\int_{\Omega_{t_p}} \boldsymbol{x}\times\rho_{t_p}\boldsymbol{a}\mathrm{d}v = \int_{\Omega_{t_p}} \boldsymbol{x}\times\rho_{t_p}\boldsymbol{g}\mathrm{d}v + \int_{\Gamma_{t_p}} \boldsymbol{x}\times\boldsymbol{t_n}\mathrm{d}s \tag{A.43}$$

式 (A.43) に対し，$\boldsymbol{t_n} = \boldsymbol{T}^t \cdot \boldsymbol{n}$ とおき，Gauss の定理

$$\int_v \frac{\partial \Psi}{\partial x_i} \mathrm{d}v = \int_s \boldsymbol{n}_i \Psi \mathrm{d}s \quad (i = 1, 2, 3) \tag{A.44}$$

を適用すると，以下の関係

$$\begin{aligned}
\int_s \boldsymbol{x} \times \boldsymbol{t_n} \mathrm{d}s &= \int_s \boldsymbol{x} \times \boldsymbol{T}^t \cdot \boldsymbol{n} \mathrm{d}s \\
&= \int_s x_i \boldsymbol{e}_i \times T_{lj} \boldsymbol{e}_j n_l \mathrm{d}s \\
&= \int_s e_{ijk} x_i T_{lj} n_l \boldsymbol{e}_k \mathrm{d}s \\
&= e_{ijk} \int_s x_i T_{lj} n_l \mathrm{d}s \boldsymbol{e}_k \\
&= e_{ijk} \int_v \frac{\partial}{\partial x_l} (x_i T_{lj}) \mathrm{d}v \boldsymbol{e}_k \\
&= e_{ijk} \int_v \left(\delta_{il} T_{lj} + x_i \frac{\partial T_{lj}}{\partial x_l} \right) \mathrm{d}v \boldsymbol{e}_k \\
&= \int_v \left\{ e_{ijk} T_{ij} \boldsymbol{e}_k + (x_i \boldsymbol{e}_i) \times \left(\frac{\partial T_{lj}}{\partial x_l} \boldsymbol{e}_j \right) \right\} \mathrm{d}v \\
&= \int_v (e_{ijk} T_{ij} \boldsymbol{e}_k + \boldsymbol{x} \times \nabla \cdot \boldsymbol{T}) \mathrm{d}v
\end{aligned} \tag{A.45}$$

から，式 (A.46) を得る．

$$\int_{\Omega_{t_p}} \boldsymbol{x} \times \rho_{t_p} \boldsymbol{a} \mathrm{d}v = \int_{\Omega_{t_p}} \{ \boldsymbol{x} \times (\rho_{t_p} \boldsymbol{g} + \nabla_{\boldsymbol{x}} \cdot \boldsymbol{T}) + \varepsilon_{ijk} \sigma_{ij} \boldsymbol{e}_k \} \mathrm{d}v \tag{A.46}$$

式 (A.46) に対し，平衡方程式 (A.42) を用いると，式 (A.47) の関係が得られる．

$$\int_{\Omega_{t_p}} \varepsilon_{ijk} \sigma_{ij} \boldsymbol{e}_k \mathrm{d}v = 0 \tag{A.47}$$

この関係は，物体の任意の領域において成立することから，式 (A.48) の関係が得られる．

$$\varepsilon_{ijk} \sigma_{ij} = 0 \tag{A.48}$$

式 (A.48) の関係が成り立つには，以下の関係

$$\begin{aligned}
T_{ij} &= T_{ji} \\
\boldsymbol{T} &= \boldsymbol{T}^t
\end{aligned} \tag{A.49}$$

でなければならないことがわかる．従って，Cauchy 応力 $\boldsymbol{T}$ は対称であり，3 次元空間では独立な成分は 6 個となる．この関係は，Cauchy の第 2 運動法則と呼ばれる．

A.1.9 材料構成式

ここまでの連続体力学の理論は，特別な仮説を導入することなく数学的に展開されてきた．これに対し，物体の力学的応答を表現する材料構成式の構築には，実験に基づく物理的考察が必要となる．本小節では，材料構成式として Saint Venant-Kirhhoff 体について述べる．その他弾塑性構成式等については，例えばを参照されたい．

さて，応力とひずみの関係として，第 2Piola-Kirhhoff 応力 $\boldsymbol{P}$ と Green-Lagrange ひずみ $\boldsymbol{E}$ が線形の関係をもつ構成式 (A.50) を考える．この構成式を Saint Venant-Kirhhoff 体と呼び，幾何学的非線形性を考慮した中で最も単純な構成則をもつ．ここで，λ と μ は Lamé 定数である．

$$\boldsymbol{S} = \lambda(\mathrm{tr}\boldsymbol{E})\boldsymbol{I} + 2\mu\boldsymbol{E} \tag{A.50}$$

Saint Venant-Kirhhoff 体は，応力ひずみ関係が線形であるため，ひずみが微小であるときは実際の現象とよく一致する．また，ひずみに幾何学的非線形性を考慮しているので，大回転を生じても不自然な解を得ない．従って，Saint Venant-Kirhhoff 体は大変形微小ひずみ問題を良く表現することができる構成式である．

さて，式 (A.50) を直交座標系で成分表示すると，式 (A.51) が得られる．

$$\begin{aligned} S_{ij} &= \lambda E_{kk}\delta_{ij} + 2\mu E_{ij} \\ &= \lambda E_{kl}\delta_{kl}\delta_{ij} + 2\mu E_{kl}\delta_{ik}\delta_{jl} \end{aligned} \tag{A.51}$$

式 (A.51) を書き直して，式 (A.52) を得る．

$$\begin{aligned} S_{ij} &= C_{ijkl}E_{kl} \\ \boldsymbol{S} &= \mathbf{C} : \boldsymbol{E} \end{aligned} \tag{A.52}$$

ここで，$\mathbf{C}$ は 4 階の弾性テンソルであり，式 (A.53) で表される．

$$\begin{aligned} \mathbf{C} &= \lambda(\boldsymbol{I} \otimes \boldsymbol{I}) + 2\mu\mathbf{I} \\ C_{ijkl} &= \lambda\delta_{kl}\delta_{ij} + 2\mu\delta_{ik}\delta_{jl} \end{aligned} \tag{A.53}$$

A.1.10 平衡方程式の現配置表示

前節で述べた平衡方程式 (Cauchy の第 1 運動法則) を式 (A.54) に再掲する．

$$\int_{\Omega_{t_p}} \rho_{t_p}(\boldsymbol{a} - \boldsymbol{g})\mathrm{d}\Omega_{t_p} = \int_{\Gamma_{t_p}} \boldsymbol{t_n}\mathrm{d}\Gamma_{t_p} \tag{A.54}$$

平衡方程式は現配置で記述される形式が標準的であるが，ここでは平衡方程式の基準配置表示を導く．式 (A.54) に対し，$\rho_{t_0}\mathrm{d}\Omega_{t_0} = \rho_{t_p}\mathrm{d}\Omega_{t_p}$，$\tilde{\boldsymbol{t}} = \dfrac{\mathrm{d}\boldsymbol{f_n}}{\mathrm{d}\Gamma_{t_0}}$ および Cauchy の公式 (A.30) を考慮して，式 (A.55) を得る．

$$\int_{\Omega_{t_0}} \rho_{t_0}(\boldsymbol{a} - \boldsymbol{g})\mathrm{d}\Omega_{t_0} = \int_{\Gamma_{t_0}} \tilde{\boldsymbol{t}}\mathrm{d}\Gamma_{t_0} \tag{A.55}$$

ここで，$\tilde{\boldsymbol{t}} = \boldsymbol{P}^t \cdot \boldsymbol{N}$ を考慮して発散定理を用いると，式 (A.56) を得る．

$$\int_{\Omega_{t_0}} \rho_{t_0} \boldsymbol{a} \mathrm{d}\Omega_{t_0} = \int_{\Omega_{t_0}} \nabla_{\boldsymbol{X}} \cdot \boldsymbol{P} \mathrm{d}\Omega_{t_0} + \int_{\Omega_{t_0}} \rho_{t_0} \boldsymbol{g} \mathrm{d}\Omega_{t_0} \tag{A.56}$$

式 (A.56) は，物体の任意の領域において成立することから，基準配置における平衡方程式 (A.57) が得られる．式 (A.57) は，第 1Piola-Kirchhoff 応力 $\boldsymbol{P}$ で表されている．

$$\rho_{t_0} \boldsymbol{a} = \nabla_{\boldsymbol{X}} \cdot \boldsymbol{P} + \rho_{t_0} \boldsymbol{g} \tag{A.57}$$

また，式 (A.33) の関係を用いて，式 (A.57) を第 2Piola-Kirchhoff 応力 $\boldsymbol{S}$ で表すと，式 (A.58) が得られる．

$$\rho_{t_0} \boldsymbol{a} = \nabla_{\boldsymbol{X}} \cdot (\boldsymbol{S} \cdot \boldsymbol{F}^t) + \rho_{t_0} \boldsymbol{g} \tag{A.58}$$

A.1.11　仮想仕事の原理に基づく弱形式化

基準配置における平衡方程式 (A.58) に対し，仮想仕事の原理に基づく弱形式を考える．式 (A.58) に対し，仮想変位 $\delta\boldsymbol{u}$ を乗じて領域 Ω_{t_0} で積分することで，式 (A.59) を得る．

$$\int_{\Omega_{t_0}} \delta\boldsymbol{u} \cdot (\nabla_{\boldsymbol{X}} \cdot \boldsymbol{P}) \mathrm{d}\Omega_{t_0} + \int_{\Omega_{t_0}} \delta\boldsymbol{u} \cdot \rho_{t_0} \boldsymbol{g} \mathrm{d}\Omega_{t_0} - \int_{\Omega_{t_0}} \delta\boldsymbol{u} \cdot \rho_{t_0} \boldsymbol{a} \mathrm{d}\Omega_{t_0} = 0 \tag{A.59}$$

このとき，右辺第 3 項は積の微分から以下のように書き換えられる．

$$\begin{aligned} \delta\boldsymbol{u} \cdot (\nabla_{\boldsymbol{X}} \cdot \boldsymbol{P}) &= \delta u_j \frac{\partial P_{ij}}{\partial X_i} \\ &= \frac{\partial}{\partial X_i}(\delta u_j P_{ij}) - \frac{\partial \delta u_j}{\partial X_i} P_{ij} \\ &= \nabla_{\boldsymbol{X}} \cdot (\boldsymbol{P} \cdot \delta\boldsymbol{u}) - \boldsymbol{P}^t : \delta\boldsymbol{F} \end{aligned} \tag{A.60}$$

これを式 (A.59) に代入して，式 (A.61) を得る．

$$\begin{aligned} -\int_{\Omega_{t_0}} \boldsymbol{P}^t : \delta\boldsymbol{F} \mathrm{d}\Omega_{t_0} + \int_{\Omega_{t_0}} \nabla_{\boldsymbol{X}} \cdot (\boldsymbol{P} \cdot \delta\boldsymbol{u}) \mathrm{d}\Omega_{t_0} \\ + \int_{\Omega_{t_0}} \delta\boldsymbol{u} \cdot \rho_{t_0} \boldsymbol{g} \mathrm{d}\Omega_{t_0} - \int_{\Omega_{t_0}} \delta\boldsymbol{u} \cdot \rho_{t_0} \boldsymbol{a} \mathrm{d}\Omega_{t_0} = 0 \end{aligned} \tag{A.61}$$

ここで，式 (A.61) の左辺第 2 項に対して発散定理を適用することで，式 (A.62) を得る．

$$\begin{aligned} \int_{\Omega_{t_0}} \nabla_{\boldsymbol{X}} \cdot (\boldsymbol{P} \cdot \delta\boldsymbol{u}) \mathrm{d}\Omega_{t_0} &= \int_{\Omega_{t_0}} \frac{\partial (\delta u_i P_{ji})}{\partial X_j} \mathrm{d}\Omega_{t_0} \\ &= \int_{\Gamma_{t_0}} \delta u_i N_j P_{ji} \mathrm{d}\Gamma_{t_0} \\ &= \int_{\Gamma_{t_0}} \delta u_i t_i \mathrm{d}\Gamma_{t_0} \\ &= \int_{\Gamma_{t_0}} \delta\boldsymbol{u} \cdot \boldsymbol{t} \mathrm{d}\Gamma_{t_0} \end{aligned} \tag{A.62}$$

式 (A.62) を式 (A.61) に代入して整理することで，式 (A.63) を得る．

$$\int_{\Omega_{t_0}} \boldsymbol{P}^t : \delta\boldsymbol{F}\mathrm{d}\Omega_{t_0} + \int_{\Omega_{t_0}} \delta\boldsymbol{u}\cdot\rho_{t_0}\boldsymbol{a}\mathrm{d}\Omega_{t_0} = \int_{\Gamma_{t_0}} \delta\boldsymbol{u}\cdot\boldsymbol{t}\mathrm{d}\Gamma_{t_0} + \int_{\Omega_{t_0}} \delta\boldsymbol{u}\cdot\rho_{t_0}\boldsymbol{g}\mathrm{d}\Omega_{t_0} \tag{A.63}$$

ここで，式 (A.63) 左辺第 1 項の第 1Piola-Kirchhoff 応力テンソル $\boldsymbol{P}$ は，第 2Piola-Kirchhoff 応力テンソル $\boldsymbol{S}$ で書き換えると式 (A.64) のように表される．

$$\int_{\Omega_{t_0}} \boldsymbol{P}^t : \delta\boldsymbol{F}\mathrm{d}\Omega_{t_0} = \int_{\Omega_{t_0}} (\boldsymbol{S}\cdot\boldsymbol{F})^t : \delta\boldsymbol{F}\mathrm{d}\Omega_{t_0} \tag{A.64}$$

第 2Piola-Kirchhoff 応力テンソル $\boldsymbol{S}$ の対称性を利用すると，以下のように変形できる．

$$\begin{aligned}
(\boldsymbol{S}\cdot\boldsymbol{F})^t : \delta\boldsymbol{F} &= \delta F_{ij}S_{jk}F_{ik} \\
&= \frac{1}{2}\delta F_{ij}S_{kj}F_{ik} + \frac{1}{2}\delta F_{ij}S_{jk}F_{ik} \\
&= \frac{1}{2}\delta F_{ij}S_{jk}F_{ik} + \frac{1}{2}\delta F_{ij}S_{jk}F_{ik} \\
&= S_{jk}\frac{1}{2}\delta(F_{ij}F_{ik}) \\
&= S_{jk}\delta\left\{\frac{1}{2}(C_{jk}-\delta_{jk})\right\} \\
&= S_{jk}\delta E_{jk} \\
&= \boldsymbol{S} : \delta\boldsymbol{E}
\end{aligned} \tag{A.65}$$

以上から，基準配置において第 2Piola-Kirchhoff 応力テンソル $\boldsymbol{S}$ を用いて表した仮想仕事の原理の弱形式 (A.66) を得る．

$$\int_{\Omega_{t_0}} \boldsymbol{S} : \delta\boldsymbol{E}\mathrm{d}\Omega_{t_0} + \int_{\Omega_{t_0}} \delta\boldsymbol{u}\cdot\rho_{t_0}\boldsymbol{a}\mathrm{d}\Omega_{t_0} = \int_{\Gamma_{t_0}} \delta\boldsymbol{u}\cdot\boldsymbol{t}\mathrm{d}\Gamma_{t_0} + \int_{\Omega_{t_0}} \delta\boldsymbol{u}\cdot\rho_{t_0}\boldsymbol{g}\mathrm{d}\Omega_{t_0} \tag{A.66}$$

また特に静解析の場合，加速度項を無視して式 (A.67) を得る．

$$\int_{\Omega_{t_0}} \boldsymbol{S} : \delta\boldsymbol{E}\mathrm{d}\Omega_{t_0} = \int_{\Gamma_{t_0}} \delta\boldsymbol{u}\cdot\boldsymbol{t}\mathrm{d}\Gamma_{t_0} + \int_{\Omega_{t_0}} \delta\boldsymbol{u}\cdot\rho_{t_0}\boldsymbol{g}\mathrm{d}\Omega_{t_0} \tag{A.67}$$

A.2 非線形有限要素法

A.2.1 有限要素離散化

有限要素法による離散化は，ある連続体が占める領域 Ω を，有限要素で占める領域 Ω^h を用いて，

$$\Omega \simeq \Omega^{(h)} = \bigcup_{e}^{N_{\mathrm{elem}}} \Omega^e \tag{A.68}$$

と考えることである．ここで，Ω^e はあるひとつの要素が占める領域，N_{elem} は有限要素の数である．以上から，これまで議論してきた体積要素の積分，表面要素の積分は，それぞれ式

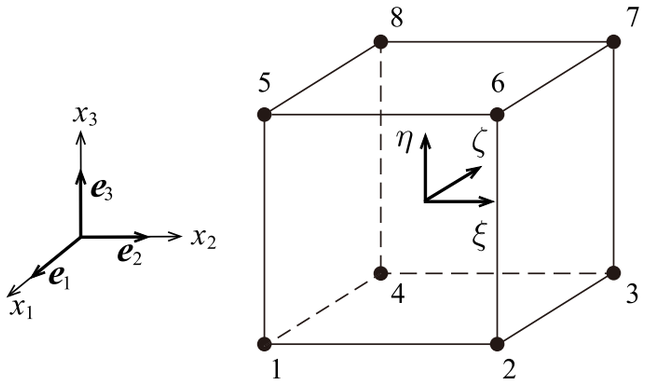


Fig. A.7 8-node hexahedral solid element and coordinate systems

(A.69)，式 (A.70) で表される．ここで，$\bigoplus$ は有限要素離散化におけるアセンブリ作用素であり，要素ベクトルを全体ベクトルに加える演算を表す．アセンブリ作用素は，拡張和としても知られている．

$$\int_{\Omega} \mathrm{d}\Omega \simeq \bigoplus_{e}^{N_{\mathrm{elem}}} \int_{\Omega^e} \mathrm{d}\Omega^e \tag{A.69}$$

$$\int_{\Gamma} \mathrm{d}\Gamma \simeq \bigoplus_{e}^{N_{\mathrm{elem}}} \int_{\Gamma^e} \mathrm{d}\Gamma^e \tag{A.70}$$

A.2.2 形状関数

形状関数 N^e は，いくつかの節点からなる有限要素において，全体座標系における節点の物理量 ϕ_i を式 (A.71) のように，要素領域 Ω^e 内で補間する関数である．

$$\phi^e(\boldsymbol{\xi}) \simeq \sum_{i=1}^{N^e_{\mathrm{node}}} N^e_i(\boldsymbol{\xi})\phi_i \tag{A.71}$$

ここで，ϕ^e は要素内で補間された物理量，N^e_i は節点 i における形状関数，ϕ_i は節点 i における物理量，N^e_{node} は要素 e を構成する節点数である．また，形状関数は 3 次元解析の場合，要素の局所座標系 $\boldsymbol{\xi} = \{\xi, \zeta, \eta\}$ において，$\xi \in [-1, 1]$，$\zeta \in [-1, 1]$，$\eta \in [-1, 1]$ の範囲で定義される．さらに，定義域の任意の座標において，その和が常に 1 となる，partition of unity 条件が満たされる．

$$\sum_{i=1}^{N^e_{\mathrm{node}}} N^e_i(\boldsymbol{\xi}) = 1 \tag{A.72}$$

ここで，全体座標系 $\boldsymbol{x}$ における，補間された物理量 ϕ の微分を考える．式 (A.71) の定義から，N^e_i が $\boldsymbol{x}$ に関して微分可能であればよいので，微分の連鎖則を考慮して，式 (A.73) が得

られる．

$$\frac{\partial N_i^e}{\partial x_j} = \frac{\partial N_i^e}{\partial \xi_k}\frac{\partial \xi_k}{\partial x_j} \tag{A.73}$$

あわせて，全体座標系における $\boldsymbol{x}$ から局所座標系における $\boldsymbol{\xi}$ の写像を考え，その Jacobi 行列を $\boldsymbol{J}$ とすると，以下の関係が得られる．

$$\mathrm{d}\boldsymbol{\xi} = \boldsymbol{J}\mathrm{d}\boldsymbol{x} \tag{A.74}$$

$$\boldsymbol{J} = \frac{\partial \boldsymbol{\xi}}{\partial \boldsymbol{x}} \tag{A.75}$$

式 (A.76) に，形状関数の例として，図 A.7 に示す連続体要素のひとつである 8 節点六面体 1 次要素の形状関数を示す．

$$\begin{cases} N_1^e = \dfrac{1}{8}(1-\xi)(1-\zeta)(1-\eta) \\ N_2^e = \dfrac{1}{8}(1+\xi)(1-\zeta)(1-\eta) \\ N_3^e = \dfrac{1}{8}(1+\xi)(1+\zeta)(1-\eta) \\ N_4^e = \dfrac{1}{8}(1-\xi)(1+\zeta)(1-\eta) \\ N_5^e = \dfrac{1}{8}(1-\xi)(1-\zeta)(1+\eta) \\ N_6^e = \dfrac{1}{8}(1+\xi)(1-\zeta)(1+\eta) \\ N_7^e = \dfrac{1}{8}(1+\xi)(1+\zeta)(1+\eta) \\ N_8^e = \dfrac{1}{8}(1-\xi)(1+\zeta)(1+\eta) \end{cases} \tag{A.76}$$

式 (A.76) は，アイソパラメトリック要素としても知られている．

A.2.3 仮想仕事の原理の離散化

基準配置における仮想仕事の原理の弱形式 (A.66) の離散化を考える．式 (A.14) より，Green-Lagrange ひずみ $\boldsymbol{E}$ の変分 $\delta\boldsymbol{E}$ は，

$$\begin{aligned} \delta\boldsymbol{E} &= \frac{1}{2}(\delta\boldsymbol{Z} + \delta\boldsymbol{Z}^t + \delta\boldsymbol{Z}^t\cdot\boldsymbol{Z} + \boldsymbol{Z}^t\cdot\delta\boldsymbol{Z}) \\ &= \frac{1}{2}\left\{\frac{\partial\delta\boldsymbol{u}}{\partial\boldsymbol{X}} + \left(\frac{\partial\delta\boldsymbol{u}}{\partial\boldsymbol{X}}\right)^t + \left(\frac{\partial\delta\boldsymbol{u}}{\partial\boldsymbol{X}}\right)^t\cdot\frac{\partial\boldsymbol{u}}{\partial\boldsymbol{X}} + \left(\frac{\partial\boldsymbol{u}}{\partial\boldsymbol{X}}\right)^t\cdot\frac{\partial\delta\boldsymbol{u}}{\partial\boldsymbol{X}}\right\} \end{aligned} \tag{A.77}$$

で表される．従って，Green-Lagrange ひずみの変分 $\delta\boldsymbol{E}$ を成分表示すれば，

$$\delta E_{ij} = \frac{1}{2}\left\{\frac{\partial\delta u_i}{\partial X_j} + \frac{\partial\delta u_j}{\partial X_i} + \frac{\partial\delta u_k}{\partial X_i}\frac{\partial u_k}{\partial X_j} + \frac{\partial u_k}{\partial X_i}\frac{\partial\delta u_k}{\partial X_j}\right\} \tag{A.78}$$

で表され，変位 $\boldsymbol{u}$ の関数であることがわかる．以上より，内力仮想仕事について，

$$\begin{aligned}\int_{\Omega_{t_0}} \boldsymbol{S} : \delta\boldsymbol{E}\mathrm{d}\Omega_{t_0} &\simeq \bigoplus_{e}^{N_{\mathrm{elem}}} \int_{\Omega_{t_0}^{(h)}} \boldsymbol{S} : \delta\boldsymbol{E}\mathrm{d}\Omega_{t_0}^{e} \\ &= \bigoplus_{e}^{N_{\mathrm{elem}}} \int_{\Omega_{t_0}^{(h)}} S_{ij}\delta E_{ij}\mathrm{d}\Omega_{t_0}^{e} \\ &= \delta\boldsymbol{u}\cdot\boldsymbol{q}(\boldsymbol{u}) \end{aligned} \tag{A.79}$$

となるような内力ベクトル $\boldsymbol{q}$ を定める．

次に，外力仮想仕事については，

$$\begin{aligned}\int_{\Gamma_{t_0}} \delta\boldsymbol{u}\cdot\boldsymbol{t}\mathrm{d}\Gamma_{t_0} + \int_{\Omega_{t_0}} \delta\boldsymbol{u}\cdot\rho_{t_0}\boldsymbol{g}\mathrm{d}\Omega_{t_0} &\simeq \bigoplus_{e}^{N_{\mathrm{elem}}} \left(\int_{\Gamma_{t_0}^{(h)}} \delta\boldsymbol{u}\cdot\boldsymbol{t}\mathrm{d}\Gamma_{t_0}^{e} + \int_{\Omega_{t_0}^{(h)}} \delta\boldsymbol{u}\cdot\rho_{t_0}\boldsymbol{g}\mathrm{d}\Omega_{t_0}^{e} \right) \\ &= \sum_{i}^{N_{\mathrm{node}}} \left(\int_{\Gamma_{t_0}^{(h)}} \delta u_i N_i t_i \mathrm{d}\Gamma_{t_0}^{e} + \int_{\Omega_{t_0}^{(h)}} \delta u_i \rho_{t_0} N_i g_i \mathrm{d}\Omega_{t_0}^{e} \right) \\ &= \delta\boldsymbol{u}\cdot\boldsymbol{f} \end{aligned} \tag{A.80}$$

となるような外力ベクトル $\boldsymbol{f}$ を定める．ここで，

$$\boldsymbol{f} = \int_{\Gamma_{t_0}^{(h)}} N_i t_i \mathrm{d}\Gamma_{t_0}^{e} + \int_{\Omega_{t_0}^{(h)}} \rho_{t_0} N_i g_i \mathrm{d}\Omega_{t_0}^{e} \tag{A.81}$$

である．

最後に，慣性力による仮想仕事については，

$$\begin{aligned}\int_{\Omega_{t_0}} \delta\boldsymbol{u}\cdot\rho_{t_0}\boldsymbol{a}\mathrm{d}\Omega_{t_0} &\simeq \sum_{i}^{N_{\mathrm{node}}}\sum_{j}^{N_{\mathrm{node}}} \delta\boldsymbol{u}\int_{\Omega_{t_0}^{(h)}} \rho_{t_0} N_i N_j \mathrm{d}\Omega_{t_0}^{e}\boldsymbol{a} \\ &= \delta\boldsymbol{u}\cdot\boldsymbol{M}\boldsymbol{a} \end{aligned} \tag{A.82}$$

となるような質量行列 $\boldsymbol{M}$ を定める．ここで，

$$\boldsymbol{M} = \sum_{i}^{N_{\mathrm{node}}}\sum_{j}^{N_{\mathrm{node}}} \int_{\Omega_{t_0}^{(h)}} \rho_{t_0} N_i N_j \mathrm{d}\Omega_{t_0}^{e} \tag{A.83}$$

である．

以上より，仮想内力仕事 W_{int}^{h}，仮想慣性力仕事 W_{inr}^{h}，仮想外力仕事 W_{ext}^{h} をそれぞれ式 (A.84)，式 (A.85)，式 (A.86) に定義する．

$$W_{\mathrm{int}}^{h} = \delta\boldsymbol{u}\cdot\boldsymbol{q}(\boldsymbol{u}) \tag{A.84}$$
$$W_{\mathrm{inr}}^{h} = \delta\boldsymbol{u}\cdot\boldsymbol{M}\boldsymbol{a} \tag{A.85}$$
$$W_{\mathrm{ext}}^{h} = \delta\boldsymbol{u}\cdot\boldsymbol{f} \tag{A.86}$$

基準配置における仮想仕事の原理の弱形式 (A.66) より，

$$W_{\mathrm{int}}^{h} + W_{\mathrm{inr}}^{h} - W_{\mathrm{ext}}^{h} = 0 \tag{A.87}$$

であり，任意の変分 $\delta \boldsymbol{u}$ で成立することから，最終的に非線形方程式

$$\boldsymbol{M}\boldsymbol{a} + \boldsymbol{q}(\boldsymbol{u}) - \boldsymbol{f} = 0 \tag{A.88}$$

の求解に帰着することがわかる．

A.2.4 動的解析

質点系の振動方程式

$$m\boldsymbol{a} + c\boldsymbol{v} + k\boldsymbol{u} = \boldsymbol{f} \tag{A.89}$$

を参考にして，構造解析で解くべき振動方程式 (A.90) にも，減衰行列 $\boldsymbol{C}$ を導入する．減衰行列 $\boldsymbol{C}$ は，何らかのモデル化を施し，実際の減衰現象を模擬するように決定される．

$$\boldsymbol{M}\boldsymbol{a} + \boldsymbol{C}\boldsymbol{v} + \boldsymbol{q}(\boldsymbol{u}) = \boldsymbol{f} \tag{A.90}$$

本章では，減衰項のモデル化として，実験的に数値を定めることのできる Rayleigh 減衰

$$\boldsymbol{C} = R_m\boldsymbol{M} + R_k\boldsymbol{K} \tag{A.91}$$

を導入する．

式 (A.92) に，時刻 $t + \Delta t$ における離散化された運動方程式を示す．

$$\boldsymbol{M}\boldsymbol{a}_{t+\Delta t} + \boldsymbol{C}\boldsymbol{v}_{t+\Delta t} + \boldsymbol{q}_{t+\Delta t} = \boldsymbol{f}_{t+\Delta t} \tag{A.92}$$

ここで，$\boldsymbol{M}$ は質量行列，$\boldsymbol{C}$ は減衰行列，$\boldsymbol{q}_{t+\Delta t}$ は内力ベクトル，$\boldsymbol{f}_{t+\Delta t}$ は外力ベクトル，$\boldsymbol{a}_{t+\Delta t}$ は加速度ベクトル，$\boldsymbol{v}_{t+\Delta t}$ は速度ベクトルである．ここで式 (A.92) が線形問題の範囲であれば，内力ベクトル $\boldsymbol{q}_{t+\Delta t}$ は，剛性マトリクス $\boldsymbol{K}$ を用いて

$$\boldsymbol{q}_{t+\Delta t} = \boldsymbol{K}\boldsymbol{u}_{t+\Delta t} \tag{A.93}$$

と表せられる．

$\boldsymbol{q}_{t+\Delta t}$ は非線形の関数であるため，内力ベクトルの線形化を行って得られる式 (A.94) に対し，Newton-Raphson 反復の反復計算 ($i = 1, 2, 3, \cdots$) を適用する．

$$\boldsymbol{M}\boldsymbol{a}^{(i)}_{t+\Delta t} + \boldsymbol{C}\boldsymbol{v}^{(i)}_{t+\Delta t} + \boldsymbol{K}_{t+\Delta t}\Delta\boldsymbol{u}^{(i)} = \boldsymbol{f}_{t+\Delta t} - \boldsymbol{q}^{(i)}_{t+\Delta t} \tag{A.94}$$

ここで，Newton-Raphson 反復ごとの変位，速度，加速度の修正量は，式 (A.95)，式 (A.96)，式 (A.97) で表される．

$$\boldsymbol{u}^{(i)}_{t+\Delta t} = \boldsymbol{u}^{(i-1)}_{t+\Delta t} + \Delta\boldsymbol{u}^{(i)} \tag{A.95}$$

$$\boldsymbol{v}^{(i)}_{t+\Delta t} = \boldsymbol{v}^{(i-1)}_{t+\Delta t} + \Delta\boldsymbol{v}^{(i)} \tag{A.96}$$

$$\boldsymbol{a}^{(i)}_{t+\Delta t} = \boldsymbol{a}^{(i-1)}_{t+\Delta t} + \Delta\boldsymbol{a}^{(i)} \tag{A.97}$$

$$\boldsymbol{q}^{(i)}_{t+\Delta t} = \boldsymbol{K}\Delta\boldsymbol{u}^{(i)}_{t+\Delta t} \tag{A.98}$$

また，上付き添字 (i) は i 回目の Newton-Raphson 反復であることを示し，(0) の場合は，以下のように時刻 t の値を用いる．

$$\boldsymbol{u}_{t+\Delta t}^{(0)} = \boldsymbol{u}_t \tag{A.99}$$
$$\boldsymbol{v}_{t+\Delta t}^{(0)} = \boldsymbol{v}_t \tag{A.100}$$
$$\boldsymbol{a}_{t+\Delta t}^{(0)} = \boldsymbol{a}_t \tag{A.101}$$
$$\boldsymbol{q}_{t+\Delta t}^{(0)} = \boldsymbol{q}_t \tag{A.102}$$

ここからは簡単のため，線形問題の運動方程式

$$\boldsymbol{M}\boldsymbol{a}_{t+\Delta t} + \boldsymbol{C}\boldsymbol{v}_{t+\Delta t} + \boldsymbol{K}\boldsymbol{u}_{t+\Delta t} = \boldsymbol{f}_{t+\Delta t} \tag{A.103}$$

について議論する．時刻 t から時刻 $t+\Delta t$ における時刻変化に関し，変位 $\boldsymbol{u}_{t+\Delta t}$，速度 $\boldsymbol{v}_{t+\Delta t}$ をそれぞれ時刻 t についてテイラー展開することを考え，式 (A.104)，式 (A.105) が得られる．

$$\boldsymbol{u}_{t+\Delta t} = \boldsymbol{u}_t + \Delta t\boldsymbol{v}_t + \frac{\Delta t^2}{2}\boldsymbol{a}_t + \frac{\Delta t^3}{6}\dddot{\boldsymbol{u}}_t \tag{A.104}$$
$$\boldsymbol{v}_{t+\Delta t} = \boldsymbol{v}_t + \Delta t\boldsymbol{a}_t + \frac{\Delta t^2}{2}\ddot{\boldsymbol{u}}_t \tag{A.105}$$

ここで，式 (A.104) において $\beta = \frac{\Delta t^3}{6}$，式 (A.105) において $\gamma = \frac{\Delta t^2}{2}$ を定義し置き直すと，式 (A.106)，式 (A.107) が得られる．

$$\boldsymbol{u}_{t+\Delta t} = \boldsymbol{u}_t + \Delta t\boldsymbol{v}_t + \frac{\Delta t^2}{2}\boldsymbol{a}_t + \beta\Delta t^3\dddot{\boldsymbol{u}}_t \tag{A.106}$$
$$\boldsymbol{v}_{t+\Delta t} = \boldsymbol{v}_t + \Delta t\boldsymbol{a}_t + \gamma\Delta t^2\dddot{\boldsymbol{u}}_t \tag{A.107}$$

ここで，加速度の変化を線形と仮定すると，式 (A.108) が得られる．

$$\dddot{\boldsymbol{u}}_t = \frac{\boldsymbol{a}_{t+\Delta t} - \boldsymbol{a}_t}{\Delta t} \tag{A.108}$$

式 (A.108) を，式 (A.106)，式 (A.107) にそれぞれ代入すると，式 (A.109)，式 (A.110) が得られる．

$$\boldsymbol{u}_{t+\Delta t} = \boldsymbol{u}_t + \Delta t\boldsymbol{v}_t + \frac{\Delta t^2}{2}\boldsymbol{a}_t + \beta\Delta t^3\frac{\boldsymbol{a}_{t+\Delta t} - \boldsymbol{a}_t}{\Delta t} \tag{A.109}$$
$$\boldsymbol{v}_{t+\Delta t} = \boldsymbol{v}_t + \Delta t\boldsymbol{a}_t + \gamma\Delta t^2\frac{\boldsymbol{a}_{t+\Delta t} - \boldsymbol{a}_t}{\Delta t} \tag{A.110}$$

これを整理して，式 (A.111)，式 (A.112) が得られる．

$$\boldsymbol{u}_{t+\Delta t} = \boldsymbol{u}_t + \Delta t\boldsymbol{v}_t + \left(\frac{1}{2} - \beta\right)\Delta t^2\boldsymbol{a}_t + \beta\Delta t^2\boldsymbol{a}_{t+\Delta t} \tag{A.111}$$
$$\boldsymbol{v}_{t+\Delta t} = \boldsymbol{v}_t + (1-\gamma)\Delta t\boldsymbol{a}_t + \gamma\Delta t\boldsymbol{a}_{t+\Delta t} \tag{A.112}$$

さらに，式 (A.111) を $\boldsymbol{a}_{t+\Delta t}$ について整理すれば，式 (A.113) が得られる．

$$\boldsymbol{a}_{t+\Delta t} = \frac{1}{\beta\Delta t^2}(\boldsymbol{u}_{t+\Delta t} - \boldsymbol{u}_t) - \frac{1}{\beta\Delta t}\boldsymbol{v}_t - \left(\frac{1}{2\beta} - 1\right)\boldsymbol{a}_t \tag{A.113}$$

つぎに，式 (A.113) を式 (A.112) に代入して，式 (A.114) を得る．

$$\begin{aligned}\boldsymbol{v}_{t+\Delta t} &= \boldsymbol{v}_t + (1-\gamma)\Delta t\boldsymbol{a}_t \\ &\quad + \gamma\Delta t\left\{\frac{1}{\beta\Delta t^2}(\boldsymbol{u}_{t+\Delta t}-\boldsymbol{u}_t) - \frac{1}{\beta\Delta t}v_t - \left(\frac{1}{2\beta}-1\right)\boldsymbol{a}_t\right\} \\ &= \frac{\gamma}{\beta\Delta t}(\boldsymbol{u}_{t+\Delta t}-\boldsymbol{u}_{t-\Delta t}) + \left(1-\frac{\gamma}{\beta}\right)\boldsymbol{v}_{t-t\Delta t} + \left(1-\frac{\gamma}{2\beta}\right)\Delta t\boldsymbol{a}v_{t-\Delta t} \quad \text{(A.114)}\end{aligned}$$

式 (A.113)，式 (A.114) を，式 (A.92) に代入して，式 (A.115) を得る．

$$\begin{aligned}&\boldsymbol{M}\left\{\frac{1}{\beta\Delta t^2}(\boldsymbol{u}_{t+\Delta t}-\boldsymbol{u}_t) - \frac{1}{\beta\Delta t}\boldsymbol{v}_t - \left(\frac{1}{2\beta}-1\right)\boldsymbol{a}_t\right\} \\ +\boldsymbol{C}&\left\{\frac{\gamma}{\beta\Delta t}(\boldsymbol{u}_{t+\Delta t}-\boldsymbol{u}_{t-\Delta t}) + \left(1-\frac{\gamma}{\beta}\right)\boldsymbol{v}_{t-t\Delta t} + \left(1-\frac{\gamma}{2\beta}\right)\Delta t\boldsymbol{a}_{t-\Delta t}\right\} \\ &+\boldsymbol{K}\boldsymbol{u}_{t+\Delta t} = \boldsymbol{f}_{t+\Delta t} \quad \text{(A.115)}\end{aligned}$$

式 (A.114) を $\boldsymbol{u}_{t+\Delta t}$ について整理して，式 (A.116) を得る．

$$\begin{aligned}(\frac{1}{\beta\Delta t^2}\boldsymbol{M} + \frac{\gamma}{\beta\Delta t}\boldsymbol{C} + \boldsymbol{K})\boldsymbol{u}_{t+\Delta t} &= \boldsymbol{f}_{t+\Delta t} - \boldsymbol{M}\left(-\frac{1}{\beta\Delta t^2}\boldsymbol{u}_t - \frac{1}{\beta\Delta t}\boldsymbol{v}_t - \frac{1-2\beta}{2\beta}\boldsymbol{a}_t\right) \\ &\quad -\boldsymbol{C}\left\{-\frac{\gamma}{\beta\Delta t}\boldsymbol{u}_t + \left(1-\frac{\gamma}{\beta}\right)\boldsymbol{v}_t + \Delta t\frac{2\beta-\gamma}{2\beta}\boldsymbol{a}_t\right\} \quad \text{(A.116)}\end{aligned}$$

ここで，Rayleigh 減衰 $\boldsymbol{C} = R_m\boldsymbol{M} + R_k\boldsymbol{K}$ を導入して，式 (A.117) を得る．

$$\begin{aligned}&\left\{\frac{1}{\beta\Delta t^2}\boldsymbol{M} + \frac{\gamma}{\beta\Delta t}(R_m\boldsymbol{M} + R_k\boldsymbol{K}) + \boldsymbol{K}\right\}\boldsymbol{u}_{t+\Delta t} = \\ &\boldsymbol{f}_{t+\Delta t} - \boldsymbol{M}\left(-\frac{1}{\beta\Delta t^2}\boldsymbol{u}_t - \frac{1}{\beta\Delta t}\boldsymbol{v}_t - \frac{1-2\beta}{2\beta}\boldsymbol{a}_t\right) \\ -(R_m\boldsymbol{M} + R_k\boldsymbol{K})&\left\{-\frac{\gamma}{\beta\Delta t}\boldsymbol{u}_t + \left(1-\frac{\gamma}{\beta}\right)\boldsymbol{v}_t + \Delta t\frac{2\beta-\gamma}{2\beta}\boldsymbol{a}_t\right\} \quad \text{(A.117)}\end{aligned}$$

式 (A.117) を $\boldsymbol{u}_{t+\Delta t}$ について整理して，式 (A.118) を得る．

$$\begin{aligned}&\left\{\left(\frac{1}{\beta\Delta t^2} + \frac{\gamma}{\beta\Delta t}R_m\right)\boldsymbol{M} + \left(\frac{\gamma}{\beta\Delta t}R_k + 1\right)\boldsymbol{K}\right\}\boldsymbol{u}_{t+\Delta t} = \boldsymbol{f}_{t+\Delta t} \\ &+\boldsymbol{M}\left[\left\{\frac{1}{\beta\Delta t^2}\boldsymbol{u}_t + \frac{1}{\beta\Delta t}\boldsymbol{v}_t + \left(\frac{1}{2\beta}-1\right)\boldsymbol{a}_t\right\}\right. \\ &\left.+R_m\left\{\frac{\gamma}{\beta\Delta t}\boldsymbol{u}_t + \left(\frac{\gamma}{\beta}-1\right)\boldsymbol{v}_t + \Delta t\left(\frac{\gamma}{2\beta}-1\right)\boldsymbol{a}_t\right\}\right] \\ &+R_k\boldsymbol{K}\left\{\frac{\gamma}{\beta\Delta t}\boldsymbol{u}_t + \left(\frac{\gamma}{\beta}-1\right)\boldsymbol{v}_t + \Delta t\left(\frac{\gamma}{2\beta}-1\right)\boldsymbol{a}_t\right\} \quad \text{(A.118)}\end{aligned}$$

ここで，一時変数 a_1，a_2，a_3，b_1，b_2，b_3，c_1，c_2 を定義して，式 (A.130) を得る．

$$a_1 = \frac{1}{2\beta} - 1 \tag{A.119}$$

$$a_2 = \frac{1}{\beta \Delta t} \tag{A.120}$$

$$a_3 = \frac{1}{\beta \Delta t^2} \tag{A.121}$$

$$b_1 = \Delta t(\frac{\gamma}{2\beta} - 1) \tag{A.122}$$

$$b_2 = \frac{\gamma}{\beta} - 1 \tag{A.123}$$

$$b_3 = \frac{\gamma}{\beta \Delta t} \tag{A.124}$$

$$c_1 = 1 + R_k b_3 \tag{A.125}$$

$$c_2 = a_3 + R_k b_3 \tag{A.126}$$

$$\begin{aligned}(c_2\boldsymbol{M} + c_1\boldsymbol{K})\boldsymbol{u}_{t+\Delta t} = \boldsymbol{f}_{t+\Delta t} \\ +\boldsymbol{M}\{(a_3\boldsymbol{u}_t + a_2\boldsymbol{v}_t + a_1\boldsymbol{a}_t) + R_m(b_3\boldsymbol{u}_t + b_2\boldsymbol{v}_t + b_1\boldsymbol{a}_t)\} \\ +R_k\boldsymbol{K}(b_3\boldsymbol{u}_t + b_2\boldsymbol{v}_t + b_1\boldsymbol{a}_t)\end{aligned} \tag{A.127}$$

ここで一時ベクトル $\boldsymbol{x}_1$，$\boldsymbol{x}_2$ を以下のように置き，式 (A.130) を得る．

$$\boldsymbol{x}_1 = a_3\boldsymbol{u}_t + a_2\boldsymbol{v}_t + a_1\boldsymbol{a}_t \tag{A.128}$$

$$\boldsymbol{x}_2 = b_3\boldsymbol{u}_t + b_2\boldsymbol{v}_t + b_1\boldsymbol{a}_t \tag{A.129}$$

$$(c_2\boldsymbol{M} + c_1\boldsymbol{K})\boldsymbol{u}_{t+\Delta t} = \boldsymbol{f}_{t+\Delta t} + \boldsymbol{M}(\boldsymbol{x}_1 + R_m\boldsymbol{x}_2) + R_k\boldsymbol{K}\boldsymbol{x}_2 \tag{A.130}$$

以上より，式 (A.130) を用いて解ベクトルとなる変位 $\boldsymbol{u}_{t+\Delta t}$ が得られれば，式 (A.113)，式 (A.114) の関係を用いて，速度 $\boldsymbol{v}_{t+\Delta t}$，加速度 $\boldsymbol{a}_{t+\Delta t}$ を算出できる．

この導出過程を，線形化した運動方程式式 (A.94) に対し同様に適用すれば，式 (A.131) の関係が得られる．内力ベクトル $\boldsymbol{q}$ が変位ベクトル $\boldsymbol{u}$ の関数であるから，式 (A.131) は非線形方程式であるため，求解には Newton-Raphson 法等の非線形方程式向けの解法を用いる．

$$(c_2\boldsymbol{M} + c_1\boldsymbol{K}^{(i-1)}_{t+\Delta t})\Delta\boldsymbol{u}^{(i)} = \boldsymbol{f}_{t+\Delta t} - \boldsymbol{q}^{(i-1)}_{t+\Delta t} + \boldsymbol{M}(\boldsymbol{x}_1 + R_m\boldsymbol{x}_2) + R_k\boldsymbol{K}\boldsymbol{x}_2 \tag{A.131}$$

従って，式 (A.132) に示す残差ベクトル $\boldsymbol{\psi}$ が，$\boldsymbol{\psi} = \boldsymbol{0}$ となるような平衡解を求める問題に帰着する．

$$\boldsymbol{\psi} = (c_2\boldsymbol{M} + c_1\boldsymbol{K}^{(i-1)}_{t+\Delta t})\Delta\boldsymbol{u}^{(i)} + \boldsymbol{q}^{(i-1)}_{t+\Delta t} - \boldsymbol{f}_{t+\Delta t} - \boldsymbol{M}(\boldsymbol{x}_1 + R_m\boldsymbol{x}_2) - R_k\boldsymbol{K}\boldsymbol{x}_2 \tag{A.132}$$

A.2.5 固有値解析

固有値解析は，破壊や振動・騒音を制御するために必要な，共振周波数や振動モードを得るための重要な手法である [29, 30]．動的解析では，固有値解析から得られる共振周波数や振動モードから，減衰項の係数が決定される．

有限要素離散化された構造解析の固有値問題は，式 (A.133) に示す固有方程式を解くことに帰着する．

$$\boldsymbol{K}\boldsymbol{x} = \lambda \boldsymbol{M}\boldsymbol{x} \tag{A.133}$$

ここで，$\boldsymbol{K}$ は全体剛性行列，$\boldsymbol{M}$ は全体質量行列，$\boldsymbol{x}$ は固有ベクトル，λ は固有値である．式 (A.133) は，一般固有値問題として分類され，全体剛性行列 $\boldsymbol{K}$ と全体質量行列 $\boldsymbol{M}$ は疎な実対称行列であることが特徴として挙げられる．工学的に低次側の固有値の情報が重要であることから，固有値問題の求解には，Lanczos 逆べき乗法が用いられる．

境界条件による変位の拘束が付加されている場合，全体剛性行列 $\boldsymbol{K}$ は正則となるので，Lanczos 逆べき乗法中に計算される線形方程式 $\boldsymbol{K}\boldsymbol{p} = \boldsymbol{q}$ は，解を得ることができる．

境界条件による拘束を付加しない固有値問題は，自由振動モードを求める問題となる．しかし，全体剛性行列 $\boldsymbol{K}$ は正則ではないため，解が定まらない（剛体モードを求めることになる）．この解決のために，全体質量行列 $\boldsymbol{M}$ に注目する．全体質量行列 $\boldsymbol{M}$ は一般に，整合質量行列・集中質量行列に関わらず逆行列を求めることができるため，この性質に注目し，一般固有値問題の固有方程式 $\boldsymbol{K}\boldsymbol{x} = \lambda\boldsymbol{M}\boldsymbol{x}$ に対し，恒等式 $-\sigma\boldsymbol{M}\boldsymbol{x} = -\sigma\boldsymbol{M}\boldsymbol{x}$ を各々加えて，固有方程式 (A.134) を得る．

$$(\boldsymbol{K} - \sigma\boldsymbol{M})\boldsymbol{x} = (\lambda - \sigma)\boldsymbol{M}\boldsymbol{x} \tag{A.134}$$

ここで，σ はシフト量である．式 (2) に対し，$\boldsymbol{K}' = \boldsymbol{K} - \sigma\boldsymbol{M}$，$\lambda' = \lambda - \sigma$ を定義して，式 (A.135) を得る．

$$\boldsymbol{K}'\boldsymbol{x} = \lambda'\boldsymbol{M}\boldsymbol{x} \tag{A.135}$$

次に，式 (A.133) に示した一般固有値問題に対し，MPC による拘束条件が付加された固有値問題は，式 (A.136) で表される．

$$\boldsymbol{B}^t\boldsymbol{K}\boldsymbol{B}\boldsymbol{B}^t\boldsymbol{x} = \lambda\boldsymbol{B}^t\boldsymbol{M}\boldsymbol{B}\boldsymbol{B}^t\boldsymbol{x} \tag{A.136}$$

ここで，$\boldsymbol{B}$ は正則な拘束行列である．式 (A.136) に対し，$\boldsymbol{K}' = \boldsymbol{B}^t\boldsymbol{K}\boldsymbol{B}$，$\boldsymbol{M}' = \boldsymbol{B}^t\boldsymbol{M}\boldsymbol{B}$，$\boldsymbol{x}' = \boldsymbol{B}^t\boldsymbol{x}$ を定義して，式 (A.137) を得る．

$$\boldsymbol{K}'\boldsymbol{x}' = \lambda\boldsymbol{M}'\boldsymbol{x}' \tag{A.137}$$

最後に，一般固有値問題に対し，MPC による拘束条件が付加された自由振動を求める固有値問題は，式 (A.138) で表される．

$$(\boldsymbol{B}^t\boldsymbol{K}\boldsymbol{B} \ -\sigma\boldsymbol{B}^t\boldsymbol{M}\boldsymbol{B})\boldsymbol{B}^t\boldsymbol{x} = (\lambda-\sigma)\boldsymbol{B}^t\boldsymbol{M}\boldsymbol{B}\boldsymbol{B}^t\boldsymbol{x} \tag{A.138}$$

ここで，$\boldsymbol{K}' = \boldsymbol{B}^t\boldsymbol{K}\boldsymbol{B} \ -\sigma\boldsymbol{B}^t\boldsymbol{M}\boldsymbol{B}$ を定義して，式 (A.139) を得る．

$$\boldsymbol{K}'\boldsymbol{x}' = \lambda'\boldsymbol{M}'\boldsymbol{x}' \tag{A.139}$$

以上に述べたように，構造解析から得られる固有値問題では，境界条件による拘束の有無や MPC による拘束条件の有無によらず，式 (A.133) の形で表された一般固有値問題を解けよい．

A.2.5.1 Lanczos 法

固有値問題を解く手法として，はじめに標準固有値問題

$$\boldsymbol{K}\boldsymbol{x} = \lambda\boldsymbol{x} \tag{A.140}$$

に対する Lanczos 法について述べる．実対称行列は，Arnoldi 法を用いることで，式 (A.141) のように変換可能であることが知られている．

$$\boldsymbol{Q}^{-1}\boldsymbol{K}\boldsymbol{Q} = \boldsymbol{T} \tag{A.141}$$

ここで，$\boldsymbol{Q}$ は正則行列，$\boldsymbol{T}$ は対称な三重対角行列である．

式 (A.141) の関係から，$\boldsymbol{K}\boldsymbol{Q} = \boldsymbol{Q}\boldsymbol{T}$ に対し，$\boldsymbol{Q}$ と $\boldsymbol{T}$ をそれぞれ式 (A.142)，式 (A.143) のように書き下す．

$$\boldsymbol{Q} = [\boldsymbol{q}_1\boldsymbol{q}_2\boldsymbol{q}_3\dots\boldsymbol{q}_n] \tag{A.142}$$

$$\boldsymbol{T} = \begin{bmatrix} \alpha_1 & \beta_2 & & & \\ \beta_2 & \alpha_2 & \beta_3 & & \\ & \beta_3 & \alpha_3 & \beta_4 & \\ & & \beta_4 & \ddots & \beta_n \\ & & & \beta_n & \alpha_n \end{bmatrix} \tag{A.143}$$

ここで，$\boldsymbol{q}_i$ は $\boldsymbol{Q}$ の第 i 番目の縦ベクトル，α_i は三重対角行列 $\boldsymbol{T}$ の第 i 番目の対角要素，β_i は三重対角行列 $\boldsymbol{T}$ の第 i 番目の副対角要素 $(2 \le i \le n)$ である．

このとき，ある縦ベクトル $\boldsymbol{q}_i$ $(2 \le i \le n-1)$ に注目すると，式 (A.141) から式 (A.144) が得られる．

$$\boldsymbol{K}\boldsymbol{q}_i = \beta_i\boldsymbol{q}_{i-1} + \alpha_i\boldsymbol{q}_i + \beta_{i+1}\boldsymbol{q}_{i+1} \tag{A.144}$$

式 (A.144) に対し，左から $\boldsymbol{q}_i$ を乗じると，式 (A.145) が得られる．

$$\boldsymbol{q}_i\boldsymbol{K}\boldsymbol{q}_i = \alpha_i \tag{A.145}$$

1: $\beta_1 = 0$, $\boldsymbol{q}_0 = \boldsymbol{0}$, $\boldsymbol{q}_1 = \boldsymbol{q}_1/||\boldsymbol{q}_1||_2$
2: **for** $i = 1, 2, 3, \ldots, n$ **do**
3: $\quad \boldsymbol{p} = \boldsymbol{K}^{-1}\boldsymbol{q}_i - \beta_i \boldsymbol{q}_{(i-1)}$
4: $\quad \alpha_i = \boldsymbol{p}^t \boldsymbol{q}_i$
5: $\quad \boldsymbol{p} = \boldsymbol{p} - \alpha_i \boldsymbol{q}_i$
6: $\quad \beta_{i+1} = ||\boldsymbol{p}||_2$
7: $\quad \boldsymbol{q}_{i+1} = \boldsymbol{p}/\beta_{i+1}$
8: **end for**

Fig. A.8 Inverse Lanczos method for standard eigenvalue problem

このとき，標準固有値問題における固有ベクトルの直交性から ${}^t\boldsymbol{q}_i\boldsymbol{q}_j = 0 \quad (i \neq j)$ である．また，式 (A.144) を移項すると，式 (A.146) を得ることができる．

$$\beta_{i+1}\boldsymbol{q}_{i+1} = \boldsymbol{K}\boldsymbol{q}_i - \alpha_i \boldsymbol{q}_i - \beta_i \boldsymbol{q}_{i-1} \tag{A.146}$$

式 (A.145)，式 (A.146) に対し，$\boldsymbol{q}_0 = \boldsymbol{0}$，$\beta_0 = 0$ の条件を加えることで，α_i, β_i が再帰的に求められる．図 A.8 に標準固有値問題に対する Lanczos 逆べき乗法を示す．ここで，$\boldsymbol{p}$ は一時ベクトルである．

次に，一般固有値問題

$$\boldsymbol{K}\boldsymbol{x} = \lambda \boldsymbol{M}\boldsymbol{x} \tag{A.147}$$

に対する Lanczos 法について述べる．一般固有値問題を Lanczos 法を用いて解く場合，一般固有値問題の固有ベクトルは M 直交性 ${}^t\boldsymbol{q}_i\boldsymbol{M}\boldsymbol{q}_j = 0 \quad (i \neq j)$ があることを考慮する必要がある．従って，標準固有値問題に対する Lanczos 法中の，内積演算として行われていた 2 ノルム $(\boldsymbol{r}, \boldsymbol{r})$ の計算を，M ノルム $(\boldsymbol{M}\boldsymbol{r}, \boldsymbol{r})$ に置き換えればよい．図 A.9 に，一般固有値問題に対する Lanczos 逆べき乗法を示す．ここで，$\boldsymbol{s}$，$\boldsymbol{r}$ は一時ベクトルである．

標準固有値問題 $\boldsymbol{A}\boldsymbol{x} = \lambda \boldsymbol{x}$ に対する Lanczos 法の収束判定を考える．

Lanczos 法によって行列 $\boldsymbol{A}$ と三重対角行列 $\boldsymbol{T}$ の相似変換 $\boldsymbol{A} = \boldsymbol{Q}\boldsymbol{T}\boldsymbol{Q}^{-1}$ が得られると，固有値 λ_i に対する行列 $\boldsymbol{A}$ の固有ベクトル $\boldsymbol{x}_i$ と，行列 $\boldsymbol{T}$ の固有ベクトル $\boldsymbol{y}_i$ には，

$$\boldsymbol{x}_i = \boldsymbol{Q}\boldsymbol{y}_i \tag{A.148}$$

の関係がある．

このとき，残差 $\boldsymbol{r} = \boldsymbol{A}\boldsymbol{x} - \lambda \boldsymbol{x}$ は，以下の式で得られる．

$$\begin{aligned} \boldsymbol{r} &= \boldsymbol{A}\boldsymbol{x} - \lambda \boldsymbol{x} \\ &= \boldsymbol{A}\boldsymbol{Q}\boldsymbol{y} - \lambda \boldsymbol{Q}\boldsymbol{y} \\ &= \boldsymbol{A}\boldsymbol{Q}\boldsymbol{y} - \boldsymbol{Q}\boldsymbol{T}\boldsymbol{y} \\ &= (\boldsymbol{A}\boldsymbol{Q} - \boldsymbol{Q}\boldsymbol{T})\boldsymbol{y} \end{aligned} \tag{A.149}$$

1: $\beta_1 = 0$, $\boldsymbol{q}_0 = \boldsymbol{0}$
2: $\boldsymbol{q}_1 = \boldsymbol{q}_1 / ||\boldsymbol{q}_1||_B$
3: $\boldsymbol{p} = \boldsymbol{M}\boldsymbol{q}_1$
4: **for** $i = 1, 2, 3, \ldots, n$ **do**
5: $\quad \boldsymbol{K}\boldsymbol{s} = \boldsymbol{p}$
6: $\quad \boldsymbol{s} = \boldsymbol{s} - \beta_i \boldsymbol{q}_{(i-1)}$
7: $\quad \alpha_i = {}^t\boldsymbol{p}\boldsymbol{s}$
8: $\quad \boldsymbol{s} = \boldsymbol{s} - \alpha_i \boldsymbol{q}_i$
9: $\quad \boldsymbol{r} = \boldsymbol{M}\boldsymbol{s}$
10: $\quad \beta_{i+1} = \sqrt{{}^t\boldsymbol{r}\boldsymbol{s}}$
11: $\quad \boldsymbol{p} = \boldsymbol{r} / \beta_{i+1}$
12: $\quad \boldsymbol{q}_{i+1} = \boldsymbol{s} / \beta_{i+1}$
13: **end for**

Fig. A.9 Inverse Lanczos method for generalized eigenvalue problem

一方，Lanczos 法を反復回数 m で打切った場合の誤差は，

$$\boldsymbol{A}\boldsymbol{Q}_m = \boldsymbol{Q}_m\boldsymbol{T} + \beta_{m+1}\boldsymbol{q}_{m+1}\boldsymbol{e}_m^t \tag{A.150}$$

と表せる．従って，Lanczos 反復の残差 $\boldsymbol{r}_m$ について

$$\begin{aligned} \boldsymbol{r}_m &= (\boldsymbol{A}\boldsymbol{Q}_m - \boldsymbol{Q}_m\boldsymbol{T})\boldsymbol{y} \\ &= \beta_{m+1}\boldsymbol{q}_{m+1}\boldsymbol{e}_m^t\boldsymbol{y} \end{aligned} \tag{A.151}$$

が得られる．

以上により，Lanczos 反復を打ち切るには，閾値 ε に対し $|\beta_m| < \varepsilon$ を満たせばよい．

A.2.5.2　QL 法　Lanczos 法は，行列 $\boldsymbol{A}$ から三重対角行列 $\boldsymbol{T}$ の相似変換 $\boldsymbol{A} = \boldsymbol{Q}\boldsymbol{T}\boldsymbol{Q}^{-1}$ を得る手法である．従って，解くべき問題の固有値・固有ベクトルを求めるには，Lanczos 法から得た三重対角行列 $\boldsymbol{T}$ を更に行列変換する必要がある．

ここでは，その手法として QL 法の概略を示す．ここで，ギブンス行列 $\boldsymbol{G}$ は，以下のように表される．

$$\boldsymbol{G} = \begin{bmatrix} 1 & & & & & & \\ & \ddots & & & & & \\ & & \cos\theta & & \sin\theta & & \\ & & & \ddots & & & \\ & & -\sin\theta & & \cos\theta & & \\ & & & & & \ddots & \\ & & & & & & 1 \end{bmatrix} \tag{A.152}$$

$$\boldsymbol{G}_{i,j,\theta} = \begin{cases} g_{kk} = 1 & (1 \leq k \leq n; k \neq i, j) \\ g_{kk} = \cos\theta & (k = i, j) \\ g_{ij} = \sin\theta & \\ g_{ji} = -\sin\theta & \\ g_{kl} = 0 & (1 \leq k, l \leq k; l \neq i, j) \end{cases} \tag{A.153}$$

行列 $\boldsymbol{T}$ にギブンス行列行列 $\boldsymbol{G}$ を次々と左から乗じ，下三角行列 $\boldsymbol{L}$ に変換できるとする．これを，

$$\boldsymbol{R}^{-1}\boldsymbol{T} = \boldsymbol{L} \tag{A.154}$$

と表す．ここで $\boldsymbol{R}^{-1}$ は，

$$\boldsymbol{R}^{-1} = \boldsymbol{G}_{1,2,-\theta}\boldsymbol{G}_{2,3,-\theta}\ldots\boldsymbol{G}_{n-1,n,-\theta} \tag{A.155}$$

である．

このとき $\boldsymbol{T}' = \boldsymbol{L}\boldsymbol{R}$ を考えると，$\boldsymbol{T} = \boldsymbol{R}\boldsymbol{L}$ より，$\boldsymbol{T}' = \boldsymbol{R}^{-1}\boldsymbol{T}\boldsymbol{R}$ と変換できる．$\boldsymbol{T}'$ の固有値は $\boldsymbol{T}$ の固有値と等しく，$\boldsymbol{T}'$ は対角行列に漸近することが知られている．

固有値解析では，以上に述べたように，行列の三重対角化と三重対角行列の対角化の過程を経て，固有値・固有ベクトルを求める．

A.2.6 周波数応答解析

モード法による周波数応答解析は，正弦波の入力に対する定常的な応答を簡便に求める手法であり，ある周波数領域での構造物全体の応答特性を把握するのに重要である [31]．式 (A.156) に，減衰を考慮した振動方程式を再掲する．

$$\boldsymbol{M}\boldsymbol{a} + \boldsymbol{C}\boldsymbol{v} + \boldsymbol{K}\boldsymbol{u} = \boldsymbol{g} \tag{A.156}$$

ここで，$\boldsymbol{g}$ は外力ベクトルである．外力 $\boldsymbol{g}$ として，式 (A.157) のように正弦波加振を考える．

$$\boldsymbol{g} = \boldsymbol{f}e^{j\omega t} \tag{A.157}$$

この荷重 $\boldsymbol{g}$ による応答は，正弦波として表すことができると考えられるので，変位 $\boldsymbol{u}$，速度 $\boldsymbol{v}$，加速度 $\boldsymbol{a}$ はそれぞれ式 (A.158)，式 (A.159)，式 (A.160) で表される．

$$\boldsymbol{u} = \boldsymbol{\phi}\boldsymbol{z}e^{j\omega t} \tag{A.158}$$

$$\boldsymbol{v} = j\omega\boldsymbol{\phi}\boldsymbol{z}e^{j\omega t} \tag{A.159}$$

$$\boldsymbol{a} = -\omega^2\boldsymbol{\phi}\boldsymbol{z}e^{j\omega t} \tag{A.160}$$

ここで，$\boldsymbol{\phi} = \{\boldsymbol{q}_1, \boldsymbol{q}_2, \ldots, \boldsymbol{q}_m\}$ は固有ベクトル $\boldsymbol{q}_i$ をまとめた行列，$\boldsymbol{z}$ は変位ベクトル $\boldsymbol{u}$ をよく表す係数ベクトルである．

式 (A.158)，式 (A.159)，式 (A.160) を式 (A.156) に代入して，式 (A.161) が得られる．

$$\begin{aligned} -\omega^2\boldsymbol{M}\boldsymbol{\phi}\boldsymbol{z}e^{j\omega t} + j\omega\boldsymbol{C}\boldsymbol{\phi}\boldsymbol{z}e^{j\omega t} + \boldsymbol{K}\boldsymbol{\phi}\boldsymbol{z}e^{j\omega t} &= \boldsymbol{f}e^{j\omega t} \\ (-\omega^2\boldsymbol{M}\boldsymbol{\phi} + j\omega\boldsymbol{C}\boldsymbol{\phi} + \boldsymbol{K}\boldsymbol{\phi})\boldsymbol{z}e^{j\omega t} &= \boldsymbol{f}e^{j\omega t} \\ (-\omega^2\boldsymbol{M}\boldsymbol{\phi} + j\omega\boldsymbol{C}\boldsymbol{\phi} + \boldsymbol{K}\boldsymbol{\phi})\boldsymbol{z} &= \boldsymbol{f} \end{aligned} \tag{A.161}$$

ここで，Rayleigh 減衰を式 (A.162)，構造減衰を式 (A.163) のように与える．

$$\boldsymbol{C} = \alpha\boldsymbol{M} + \beta\boldsymbol{K} \tag{A.162}$$
$$\boldsymbol{K} = (1 + j\gamma)\boldsymbol{K} \tag{A.163}$$

式 (A.162)，式 (A.163) を式 (A.161) に代入して，式 (A.164) が得られる．

$$\{-\omega^2\boldsymbol{M}\boldsymbol{\phi} + j\omega(\alpha\boldsymbol{M} + \beta\boldsymbol{K})\boldsymbol{\phi} + (1 + j\gamma)\boldsymbol{K}\boldsymbol{\phi}\}\boldsymbol{z} = \boldsymbol{f} \tag{A.164}$$

式 (A.164) に対し，左から $\boldsymbol{\phi}^t$ を乗じて，式 (A.165) を得る．

$$\{-\omega^2\boldsymbol{\phi}^t\boldsymbol{M}\boldsymbol{\phi} + j\omega(\alpha\boldsymbol{\phi}^t\boldsymbol{M}\boldsymbol{\phi} + \beta\boldsymbol{\phi}^t\boldsymbol{K}\boldsymbol{\phi}) + (1 + j\gamma)\boldsymbol{\phi}^t\boldsymbol{K}\boldsymbol{\phi}\}\boldsymbol{z} = \boldsymbol{\phi}^t\boldsymbol{f} \tag{A.165}$$

ここで，固有方程式 $\boldsymbol{K}\boldsymbol{\phi} = \boldsymbol{\Lambda}\boldsymbol{M}\boldsymbol{\phi}$ を考えると，式 (A.166)，式 (A.167) が得られる．

$$\boldsymbol{\phi}^t\boldsymbol{M}\boldsymbol{\phi} = \boldsymbol{I} \tag{A.166}$$
$$\boldsymbol{\phi}^t\boldsymbol{K}\boldsymbol{\phi} = \boldsymbol{\Lambda} = \mathrm{diag}(\lambda_1, \lambda_2, \cdots, \lambda_n) = \mathrm{diag}(\bar{\omega}_1^2, \bar{\omega}_2^2, \cdots, \bar{\omega}_n^2) \tag{A.167}$$

式 (A.165)，式 (A.166)，式 (A.167) から，式 (A.168) を得る．

$$\{-\omega^2\boldsymbol{I} + j\omega(\alpha\boldsymbol{I} + \beta\boldsymbol{\Lambda}) + (1 + j\gamma)\boldsymbol{\Lambda}\}\boldsymbol{z} = \boldsymbol{\phi}^t\boldsymbol{f} \tag{A.168}$$

式 (A.168) に対し，対角行列であることを考慮すると，i 番目モードについて式 (A.169) が得られる．

$$\begin{aligned}\{-\omega^2 + j\omega(\alpha + \beta\bar{\omega}_i^2) + (1 + j\gamma)\bar{\omega}_i^2\}z_i &= \boldsymbol{\phi}_i^t\boldsymbol{f} \\ \{-\omega^2 + j(\alpha\omega + \beta\omega\bar{\omega}_i^2 + \gamma\bar{\omega}_i^2) + \bar{\omega}_i^2\}z_i &= \boldsymbol{\phi}_i^t\boldsymbol{f}\end{aligned} \tag{A.169}$$

式 (A.169) を z_i について式変形して，式 (A.170) が得られる．

$$z_i = \frac{\boldsymbol{\phi}_i^t\boldsymbol{f}}{\bar{\omega}_i^2 - \omega^2 + j(\alpha\omega + \beta\omega\bar{\omega}_i^2 + \gamma\bar{\omega}_i^2)} \tag{A.170}$$

式 (A.171) の関係を用いて，式 (A.170) を変形すると，式 (A.172) が得られる．

$$\frac{1}{a + jb} = \frac{(a - jb)}{(a + jb)(a - jb)} = \frac{(a - jb)}{a^2 + b^2} \tag{A.171}$$

$$z_i = \frac{\boldsymbol{\phi}_i^t\boldsymbol{f} + (\bar{\omega}_i^2 - \omega^2) - j(\alpha\omega + \beta\omega\bar{\omega}_i^2 + \gamma\bar{\omega}_i^2)}{(\bar{\omega}_i^2 - \omega^2)^2 + (\alpha\omega + \beta\omega\bar{\omega}_i^2 + \gamma\bar{\omega}_i^2)^2} \tag{A.172}$$

式 (A.172) によって係数ベクトル $\boldsymbol{z}$ を得ることで，式 (A.158)，式 (A.159)，式 (A.160) から，周波数応答を得ることができる．