三次元弾性解析コード（3/3）
線形ソルバー

2011年夏学期

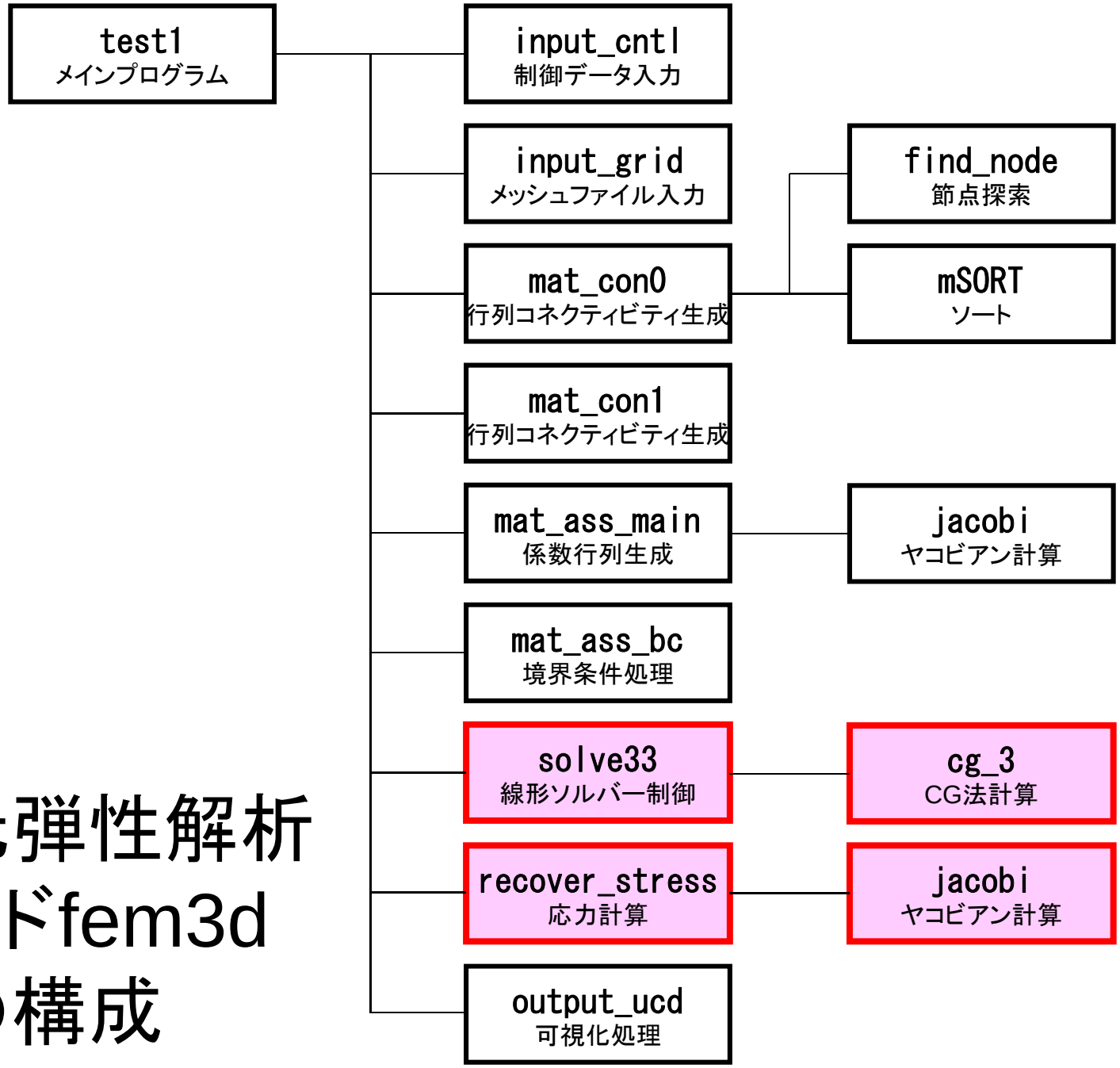
中島 研吾

科学技術計算Ｉ（4820-1027）・コンピュータ科学特別講義Ｉ（4810-1204）

有限要素法の処理：プログラム

- 初期化
  - 制御変数読み込み
  - 座標読み込み⇒要素生成（N:節点数，ICELTOT：要素数）
  - 配列初期化（全体マトリクス，要素マトリクス）
  - 要素⇒全体マトリクスマッピング（Index，Item）
- マトリクス生成
  - 要素単位の処理（do icel= 1, ICELTOT）
    - 要素マトリクス計算
    - 全体マトリクスへの重ね合わせ
  - 境界条件の処理
- 連立一次方程式
  - 共役勾配法（CG）
- 応力計算

三次元弾性解析コードfem3dの構成

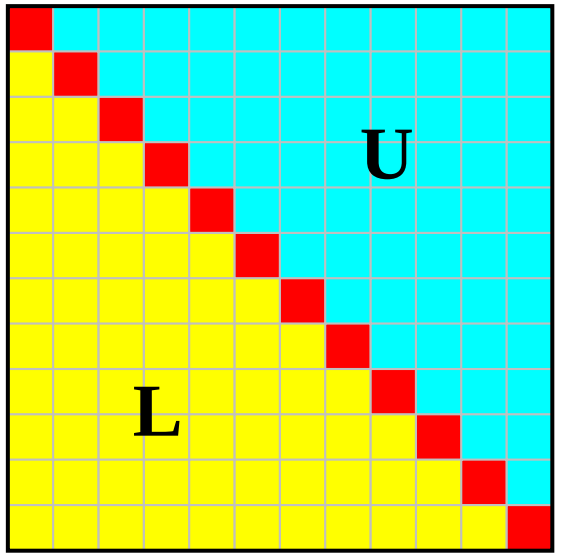


全体処理

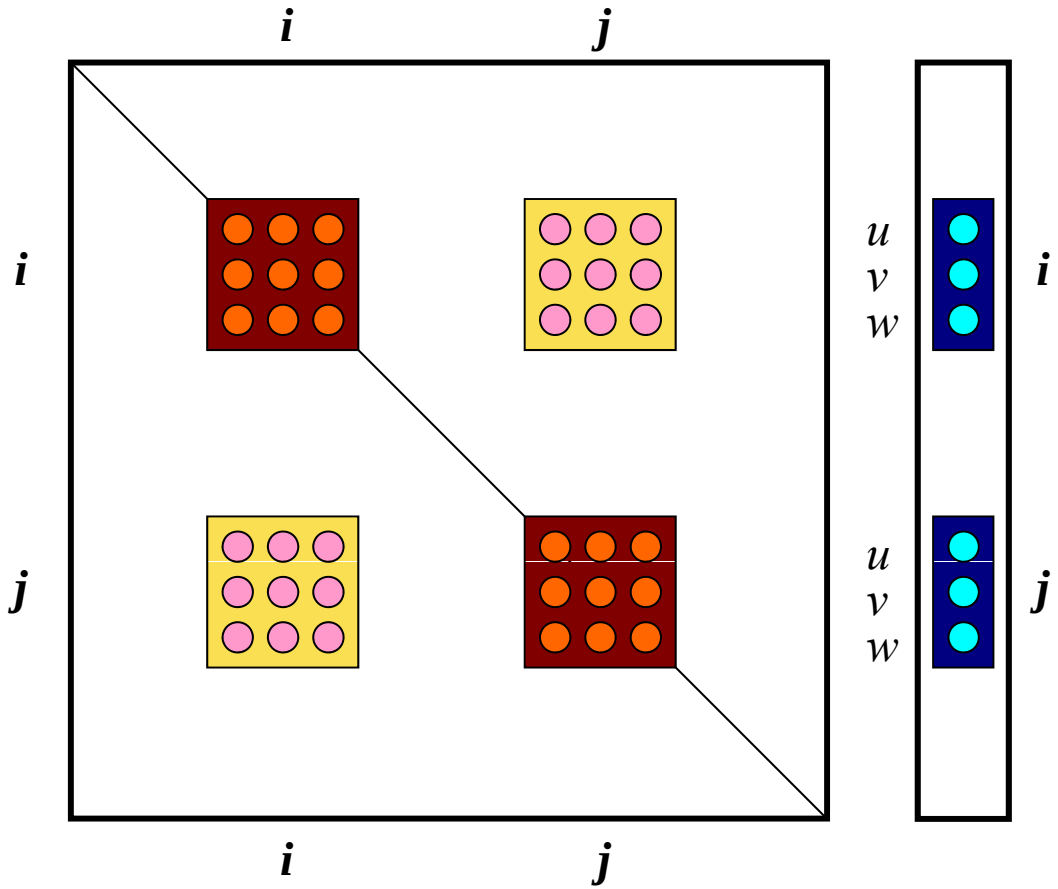
```
#include <stdio.h>
#include <stdlib.h>
FILE* fp_log;
#define GLOBAL_VALUE_DEFINE
#include "pfem_util.h"

extern void INPUT_CNTL();
extern void INPUT_GRID();
extern void MAT_CON0();
extern void MAT_CON1();
extern void MAT_ASS_MAIN();
extern void MAT_ASS_BC();
extern void SOLVE33();
extern void RECOVER_STRESS();
extern void OUTPUT_UCD();
int main()
{

/** Logfile for debug **/
        if( (fp_log=fopen("log.log","w")) == NULL){
                fprintf(stdout,"input file cannot be opened!¥n");
                exit(1);
        }

        INPUT_CNTL();
        INPUT_GRID();


        MAT_CON0();
        MAT_CON1();

        MAT_ASS_MAIN();
        MAT_ASS_BC()  ;

        SOLVE33();

        RECOVER_STRESS();
        OUTPUT_UCD()    ;
}
```

fem3d：いくつかの特徴

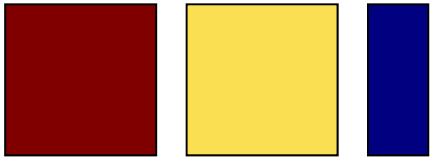
- 非対角成分
  - 上三角，下三角を分けて記憶
    - indexL，itemL，AL
    - indexU，itemU，AU
- ブロックとして記憶
  - ベクトル：1節点3成分
  - 行列：各ブロック9成分
  - 行列の各成分ではなく，節点上の3変数に基づくブロックとして処理する



用語の定義



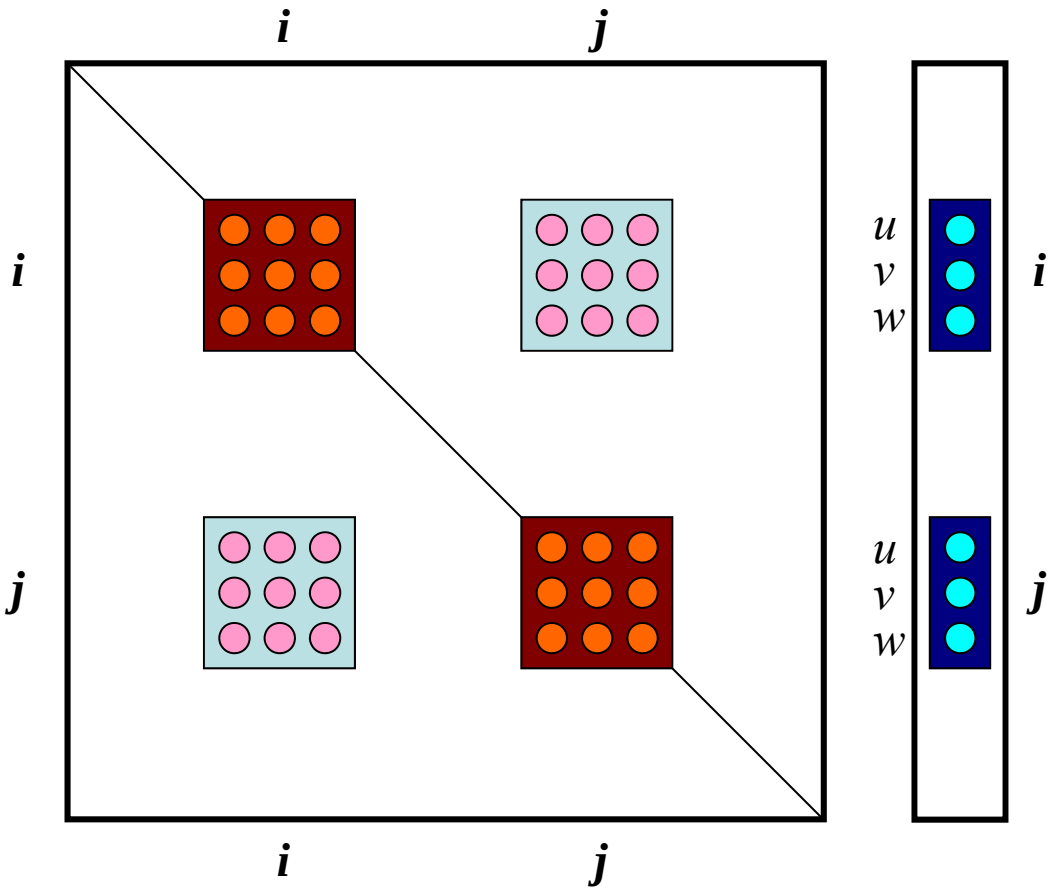
ブロック（節点）:



成分（自由度）:

ブロックとして記憶（1/3）

- 記憶容量が減る
  - index, itemに関する記憶容量を削減できる



ブロックとして記憶（2/3）

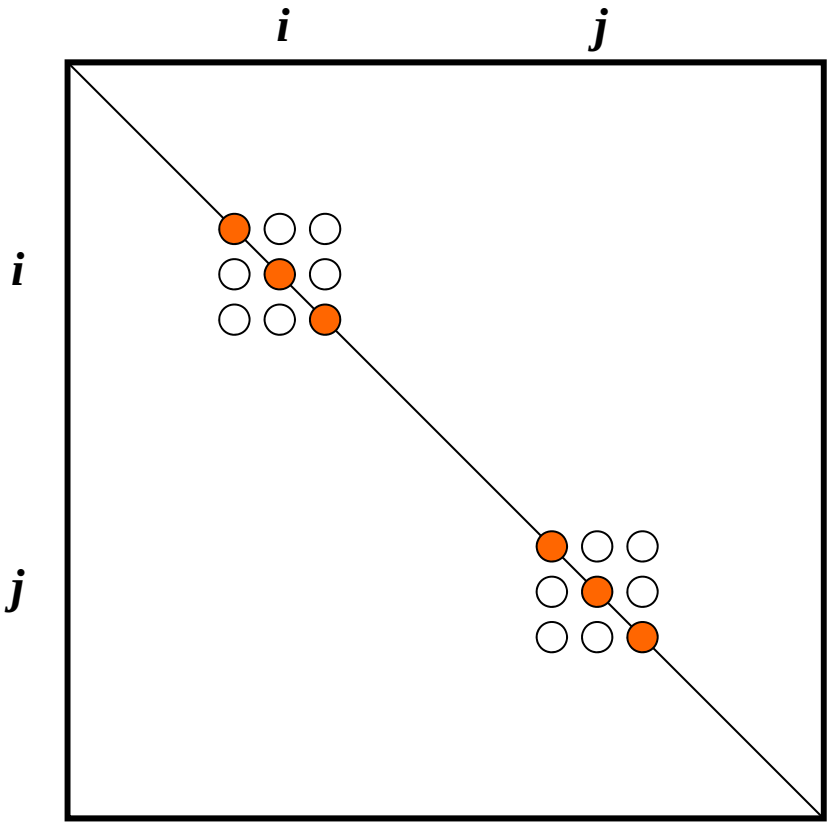
- 計算効率
  - 間接参照（メモリに負担）と計算の比が大きくなる
  - ベクトル，スカラー共に効く：2倍以上の性能
    - 連続領域，キャッシュに載る，ループあたりの計算量増加

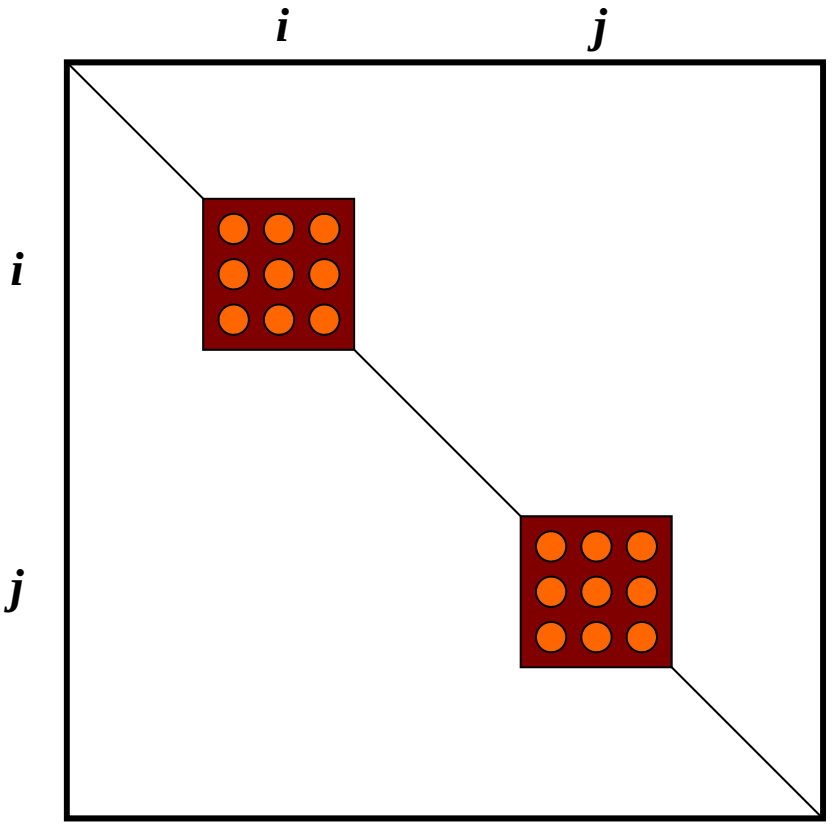
```
do i= 1, 3*N
  Y(i)= D(i)*X(i)
  do k= index(i-1)+1, index(i)
    kk= item(k)
    Y(i)= Y(i) + AMAT(k)*X(kk)
  enddo
enddo
```

```
do i= 1, N
  X1= X(3*i-2)
  X2= X(3*i-1)
  X3= X(3*i)
  Y(3*i-2)= D(9*i-8)*X1+D(9*i-7)*X2+D(9*i-6)*X3
  Y(3*i-1)= D(9*i-5)*X1+D(9*i-4)*X2+D(9*i-3)*X3
  Y(3*I  )= D(9*i-2)*X1+D(9*i-1)*X2+D(9*I  )*X3
  do k= index(i-1)+1, index(i)
    kk= item(k)
    X1= X(3*kk-2)     u
    X2= X(3*kk-1)     v
    X3= X(3*kk)       w
    Y(3*i-2)= Y(3*i-2)+AMAT(9*k-8)*X1+AMAT(9*k-7)*X2 &
                                      +AMAT(9*k-6)*X3
    Y(3*i-1)= Y(3*i-1)+AMAT(9*k-5)*X1+AMAT(9*k-4)*X2 &
                                      +AMAT(9*k-3)*X3
    Y(3*I  )= Y(3*I  )+AMAT(9*k-2)*X1+AMAT(9*k-1)*X2 &
                                      +AMAT(9*k  )*X3

  enddo
enddo
```

ブロックとして記憶（3/3）

- 計算の安定化
  - 対角成分で割るのではなく，対角ブロックの完全LU分解を求めて解く
  - 特に悪条件問題で有効





Global変数表：pfem_util.h/f（1/3）

変数名	種別	サイズ	I/O	内　容
fname	C	[80]	I	メッシュファイル名
N,NP	I		I	節点数
ICELTOT	I		I	要素数
NODGRPtot	I		I	節点グループ数
XYZ	R	[N][3]	I	節点座標
ICELNOD	I	[ICELTOT][8]	I	要素コネクティビティ
NODGRP_INDEX	I	[NODGRPtot+1]	I	各節点グループに含まれる節点数（累積）
NODGRP_ITEM	I	[NODGRP_INDEX[NODGRPTOT+1]]	I	節点グループに含まれる節点
NODGRP_NAME	C80	[NODGRP_INDEX[NODGRPTOT+1]]	I	節点グループ名
NL, NU	I		O	各節点非対角成分数（上三角・下三角）
NPL, NPU	I		O	非対角成分総数（上三角・下三角）
D	R	[9*N]	O	全体行列：対角ブロック
B,X	R	[3*N]	O	右辺ベクトル，未知数ベクトル

Global変数表：pfem_util.h/f（2/3）

変数名	種別	サイズ	I/O	内　容
ALUG	R	[9*N]	O	全体行列：対角ブロックの完全LU分解
AL,AU	R	[9*NPL],[9*NPU]	O	全体行列：上・下三角ブロック成分
indexL, indexU	I	[N+1]	O	全体行列：非零非対角ブロック数
itemL, itemU	I	[NPL], [NPU]	O	全体行列：上・下三角ブロック（列番号）
INL, INU	I	[N]	O	各節点の上・下三角ブロック数
IAL, IAU	I	[N][NL],[N][NU]	O	各節点の上・下三角ブロック（列番号）
IWKX	I	[N][2]	O	ワーク用配列
METHOD	I		I	反復解法（=1に固定）
PRECOND	I		I	前処理手法（=0：ブロックSSOR，=1：ブロック対角スケーリング）
ITER, ITERactual	I		I	反復回数の上限，実際の反復回数
RESID	R		I	打ち切り誤差（1.e-8に設定）
SIGMA_DIAG	R		I	LU分解時の対角成分係数（=1.0に設定）
pfemIarray	I	[100]	O	諸定数（整数）
pfemRarray	R	[100]	O	諸定数（実数）

Global変数表：pfem_util.h/f（3/3）

変数名	種別	サイズ	I/O	内　容
O8th	R		I	=0.125
PNQ, PNE, PNT	R	[2][2][8]	O	各ガウス積分点における $\frac{\partial N_i}{\partial \xi}, \frac{\partial N_i}{\partial \eta}, \frac{\partial N_i}{\partial \zeta} (i=1\sim 8)$
POS, WEI	R	[2]	O	各ガウス積分点の座標，重み係数
NCOL1, NCOL2	I	[100]	O	ソート用ワーク配列
SHAPE	R	[2][2][2][8]	O	各ガウス積分点における形状関数 N_i (i=1~8)
PNX, PNY, PNZ	R	[2][2][2][8]	O	各ガウス積分点における $\frac{\partial N_i}{\partial x}, \frac{\partial N_i}{\partial y}, \frac{\partial N_i}{\partial z} (i=1\sim 8)$
DETJ	R	[2][2][2]	O	各ガウス積分点におけるヤコビアン行列式
ELAST, POISSON	R		I	ヤング率，ポアソン比
SIGMA_N, TAU_N	R	[N][3]	O	節点における垂直，せん断応力成分

- 前処理について
- ソルバー部分
- 応力計算
- レポート課題2

前処理（preconditioning）とは?

- 反復法の収束は係数行列の固有値分布に依存
  - 固有値分布が少なく，かつ1に近いほど収束が早い（単位行列）
  - 条件数（condition number）（対称正定）＝最大最小固有値比
    - 条件数が1に近いほど収束しやすい
- もとの係数行列**[A]**に良く似た前処理行列**[M]**を適用することによって固有値分布を改善する。
  - 前処理行列**[M]**によって元の方程式[A]{x}={b}を [A′]{x′}={b′}へと変換する。ここで[A′]=[M]$^{-1}$[A]，{b′}=[M]$^{-1}${b} である。
  - [A′]=[M]$^{-1}$[A]が単位行列に近ければ良いということになる。
  - [A′]=[A][M]$^{-1}$のように右からかけることもある。
- 「前処理」は密行列，疎行列ともに使用するが，普通は疎行列を対象にすることが多い。

前処理付共役勾配法

Preconditioned Conjugate Gradient Method（PCG）

```
Compute r(0)= b-[A]x(0)
for i= 1, 2, …
    solve [M]z(i-1)= r(i-1)
    ρi-1= r(i-1) z(i-1)
    if i=1
      p(1)= z(0)
     else
      βi-1= ρi-1/ρi-2
      p(i)= p(i-1) + βi-1 z(i-1)
    endif
    q(i)= [A]p(i)
    αi  = ρi-1/p(i)q(i)
    x(i)= x(i-1) + αip(i)
    r(i)= r(i-1) - αiq(i)
    check convergence |r|
end
```

実際にやるべき計算は：

$$\{z\} = [M]^{-1}\{r\}$$

「近似逆行列」の計算が必要：

$$[M]^{-1} \approx [A]^{-1}, \quad [M] \approx [A]$$

究極の前処理：本当の逆行列

$$[M]^{-1} = [A]^{-1}, \quad [M] = [A]$$

対角スケーリング：簡単＝弱い

$$[M]^{-1} = [D]^{-1}, \quad [M] = [D]$$

対角スケーリング，点ヤコビ前処理

- 前処理行列として，もとの行列の対角成分のみを取り出した行列を前処理行列 [M] とする。
  - 対角スケーリング，点ヤコビ（point-Jacobi）前処理

$$[M]=\begin{bmatrix} D_1 & 0 & \cdots & 0 & 0 \\ 0 & D_2 & & 0 & 0 \\ \cdots & & \cdots & & \cdots \\ 0 & 0 & & D_{N-1} & 0 \\ 0 & 0 & \cdots & 0 & D_N \end{bmatrix}$$

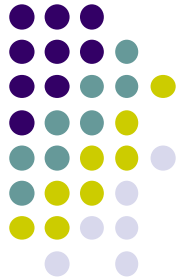
- **solve $[M]z^{(i-1)}= r^{(i-1)}$**という場合に逆行列を簡単に求めることができる。
- 簡単な問題では収束する。

ILU(0), IC(0)

- 最もよく使用されている前処理（疎行列用）
  - 不完全LU分解
    - Incomplete LU Factorization
  - 不完全コレスキー分解
    - Incomplete Cholesky Factorization（対称行列）

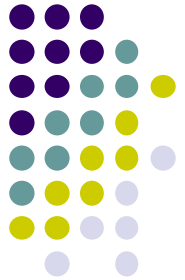
- 不完全な直接法
  - もとの行列が疎でも，逆行列は疎とは限らない。
  - fill-in
  - もとの行列と同じ非ゼロパターン（fill-in無し）を持っているのがILU（0），IC（0）

LU分解法：完全LU分解法



- 直接法の一種
  - 逆行列を直接求める手法
  - 「逆行列」に相当するものを保存しておけるので，右辺が変わったときに計算時間を節約できる
  - 逆行列を求める際にFill-in（もとの行列では0であったところに値が入る）が生じる
- LU factorization

「不」完全LU分解法



- ILU factorization
  - Incomplete LU factorization

- Fill-inの発生を制限して，前処理に使う手法
  - 不完全な逆行列（近似逆行列），少し弱い直接法
  - Fill-inを許さないとき：ILU(0)

LU分解による連立一次方程式の解法

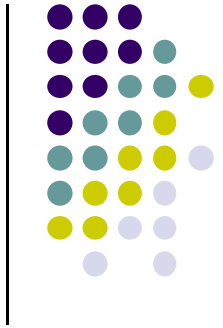
Aがn×n行列のとき、Aを次式のように表すことを
（あるいは、そのようなLとUそのものを）AのLU分解という.

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1n} \\ a_{21} & a_{22} & a_{23} & \cdots & a_{2n} \\ a_{31} & a_{32} & a_{33} & \cdots & a_{3n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & a_{n3} & \cdots & a_{nn} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 \\ l_{21} & 1 & 0 & \cdots & 0 \\ l_{31} & l_{32} & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ l_{n1} & l_{n2} & l_{n3} & \cdots & 1 \end{pmatrix} \begin{pmatrix} u_{11} & u_{12} & u_{13} & \cdots & u_{1n} \\ 0 & u_{22} & u_{23} & \cdots & u_{2n} \\ 0 & 0 & u_{33} & \cdots & u_{3n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & u_{nn} \end{pmatrix}$$

$$\mathbf{A = LU}$$

L : Lower triangular part of matrix A
U : Upper triangular part of matrix A

連立一次方程式の行列表現



n元の連立一次方程式の一般形

$$a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = b_1$$
$$a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n = b_2$$
$$\vdots$$
$$a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nn}x_n = b_n$$

行列表現

$$\begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{pmatrix} \Leftrightarrow \mathbf{Ax} = \mathbf{b}$$

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

LU分解を用いたAx=bの解法

1 $\mathbf{A} = \mathbf{LU}$ となるAのLU分解LとUを求める.

2 $\mathbf{Ly} = \mathbf{b}$ の解yを求める.（簡単！）

3 $\mathbf{Ux} = \mathbf{y}$ の解xを求める.（簡単！）

このxが $\mathbf{Ax} = \mathbf{b}$ の解となる

$$\because \mathbf{Ax} = \mathbf{LUx} = \mathbf{Ly} = \mathbf{b}$$

Ly=bの解法：前進代入

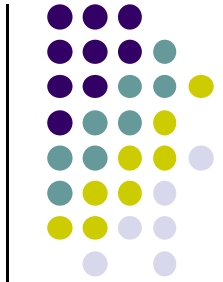
$$\mathbf{Ly = b} \quad \Longleftrightarrow \quad \begin{pmatrix} 1 & 0 & \cdots & 0 \\ l_{21} & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ l_{n1} & l_{n2} & \cdots & 1 \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{pmatrix}$$

$$\Longleftrightarrow$$

$$\begin{aligned} & y_1 = b_1 \\ & l_{21} y_1 + y_2 = b_2 \\ & \vdots \\ & l_{n1} y_1 + l_{n2} y_2 + \cdots + y_n = b_n \end{aligned} \quad \Longleftrightarrow \quad \begin{aligned} & y_1 = b_1 \\ & y_2 = b_2 - l_{21} y_1 \\ & \vdots \\ & y_n = b_n - l_{n1} y_1 - l_{n2} y_2 = b_n - \sum_{i=1}^{n-1} l_{ni} y_i \end{aligned}$$

芋づる式に（one after another）解が求まる.

Ux=yの解法：後退代入



$$\mathbf{Ux} = \mathbf{y} \quad \Longleftrightarrow \quad \begin{pmatrix} u_{11} & u_{12} & \cdots & u_{1n} \\ 0 & u_{22} & \cdots & u_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & u_{nn} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}$$

$$u_{nn} x_n = y_n$$

$$u_{n-1,n-1} x_{n-1} + u_{n-1,n} x_n = y_{n-1}$$

$$\vdots$$

$$u_{11} x_1 + u_{12} x_2 + \cdots + u_{1n} x_n = y_1$$

$\Longleftrightarrow$

$$x_n = y_n / u_{nn}$$

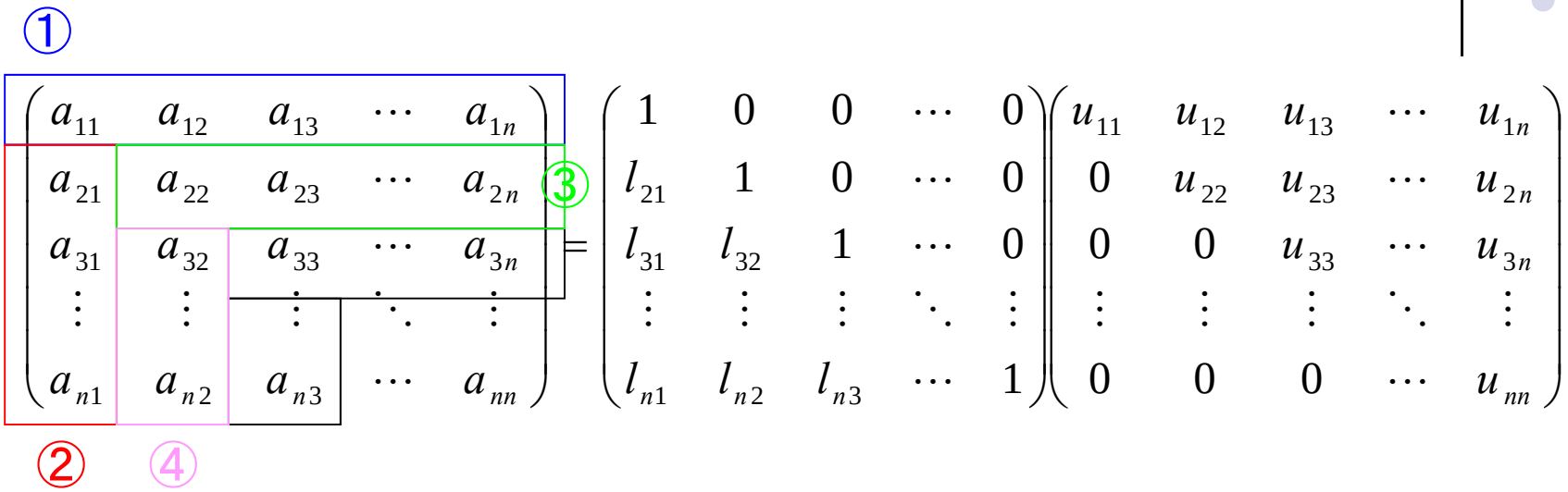
$$x_{n-1} = (y_{n-1} - u_{n-1,n} x_n) / u_{n-1,n-1}$$

$$\vdots$$

$$x_1 = \left(y_1 - \sum_{i=2}^{n} u_{1j} x_j \right) / u_{11}$$

芋づる式に（one after another）解が求まる.

LU分解の求め方



$$\begin{pmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1n} \\ a_{21} & a_{22} & a_{23} & \cdots & a_{2n} \\ a_{31} & a_{32} & a_{33} & \cdots & a_{3n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & a_{n3} & \cdots & a_{nn} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 \\ l_{21} & 1 & 0 & \cdots & 0 \\ l_{31} & l_{32} & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ l_{n1} & l_{n2} & l_{n3} & \cdots & 1 \end{pmatrix} \begin{pmatrix} u_{11} & u_{12} & u_{13} & \cdots & u_{1n} \\ 0 & u_{22} & u_{23} & \cdots & u_{2n} \\ 0 & 0 & u_{33} & \cdots & u_{3n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & u_{nn} \end{pmatrix}$$

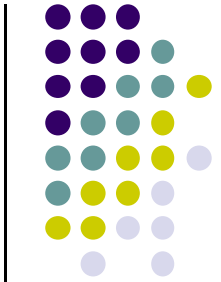
① ➡ $a_{11} = u_{11}, a_{12} = u_{12}, \cdots, a_{1n} = u_{1n} \Rightarrow u_{11}, u_{12}, \cdots, u_{1n}$

② ➡ $a_{21} = l_{21}u_{11}, a_{31} = l_{31}u_{11}, \cdots, a_{n1} = l_{n1}u_{11} \Rightarrow l_{21}, l_{31}, \cdots, l_{n1}$

③ ➡ $a_{22} = l_{21}u_{12} + u_{22}, \cdots, a_{2n} = l_{21}u_{1n} + u_{2n} \Rightarrow u_{22}, u_{23}, \cdots, u_{2n}$

④ ➡ $a_{32} = l_{31}u_{12} + l_{32}u_{22}, \cdots \Rightarrow l_{32}, l_{42}, \cdots, l_{n2}$

数値例



$$\mathbf{A}=\begin{pmatrix}1 & 2 & 3 & 4\\ 2 & 6 & 7 & 10\\ 2 & 2 & 8 & 7\\ 0 & -4 & 7 & 1\end{pmatrix}=\begin{pmatrix}1 & 0 & 0 & 0\\ l_{21} & 1 & 0 & 0\\ l_{31} & l_{32} & 1 & 0\\ l_{41} & l_{42} & l_{43} & 1\end{pmatrix}\begin{pmatrix}u_{11} & u_{12} & u_{13} & u_{14}\\ 0 & u_{22} & u_{23} & u_{24}\\ 0 & 0 & u_{33} & u_{34}\\ 0 & 0 & 0 & u_{44}\end{pmatrix}$$

第1行 ➡ $1=u_{11},2=u_{12},3=u_{13},4=u_{14}$

第1列 ➡ $2=l_{21}u_{11}\Rightarrow l_{21}=2/u_{11}=2,\quad 2=l_{31}u_{11}\Rightarrow l_{31}=2/u_{11}=2$

$0=l_{41}u_{11}\Rightarrow l_{41}=0/u_{11}=0$

第2行 ➡ $6=l_{21}u_{12}+u_{22}\Rightarrow u_{22}=2,\quad 7=l_{21}u_{13}+u_{23}\Rightarrow u_{23}=1$

$10=l_{21}u_{14}+u_{24}\Rightarrow u_{24}=2$

第2列 ➡ $2=l_{31}u_{12}+l_{32}u_{22}\Rightarrow l_{32}=-1,\quad -4=l_{41}u_{12}+l_{42}u_{22}\Rightarrow l_{42}=-2$

数値例(続き)

$$\mathbf{A}=\begin{pmatrix}1 & 2 & 3 & 4\\ 2 & 6 & 7 & 10\\ 2 & 2 & 8 & 7\\ 0 & -4 & 7 & 1\end{pmatrix}=\begin{pmatrix}1 & 0 & 0 & 0\\ l_{21} & 1 & 0 & 0\\ l_{31} & l_{32} & 1 & 0\\ l_{41} & l_{42} & l_{43} & 1\end{pmatrix}\begin{pmatrix}u_{11} & u_{12} & u_{13} & u_{14}\\ 0 & u_{22} & u_{23} & u_{24}\\ 0 & 0 & u_{33} & u_{34}\\ 0 & 0 & 0 & u_{44}\end{pmatrix}$$

第3行 ➡ $8=l_{31}u_{13}+l_{32}u_{23}+u_{33}\Rightarrow u_{33}=3,$

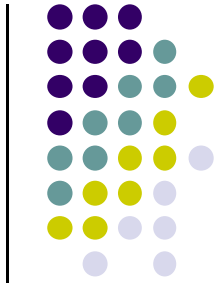
$7=l_{31}u_{14}+l_{32}u_{24}+u_{34}\Rightarrow u_{34}=1$

第3列 ➡ $7=l_{41}u_{13}+l_{42}u_{23}+l_{43}u_{33}\Rightarrow l_{43}=3$

第4行(第4列) ➡ $1=l_{41}u_{14}+l_{42}u_{24}+l_{43}u_{34}+u_{44}\Rightarrow u_{44}=2$

1行、1列、2行、2列、・・・の順に求める式を作っていく.

数値例(続き)

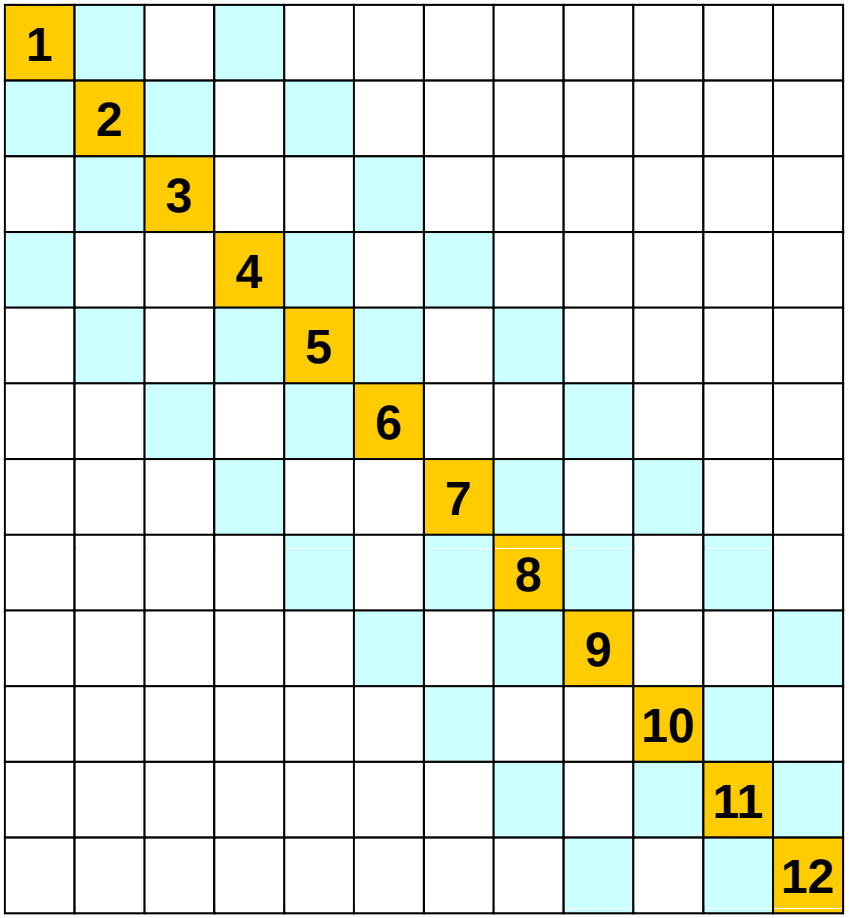


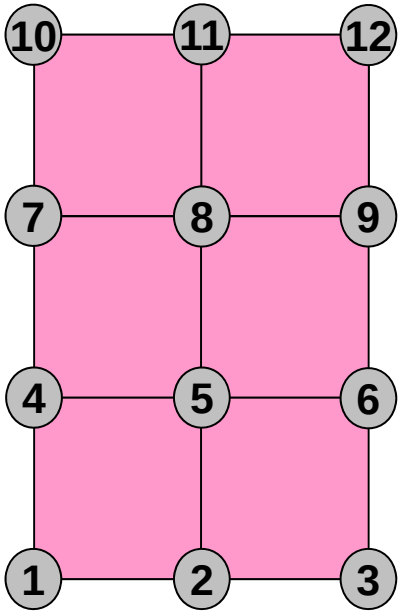
結局

$$\mathbf{A} = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 2 & 6 & 7 & 10 \\ 2 & 2 & 8 & 7 \\ 0 & -4 & 7 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 2 & 1 & 0 & 0 \\ 2 & -1 & 1 & 0 \\ 0 & -2 & 3 & 1 \end{pmatrix} \begin{pmatrix} 1 & 2 & 3 & 4 \\ 0 & 2 & 1 & 2 \\ 0 & 0 & 3 & 1 \\ 0 & 0 & 0 & 2 \end{pmatrix}$$

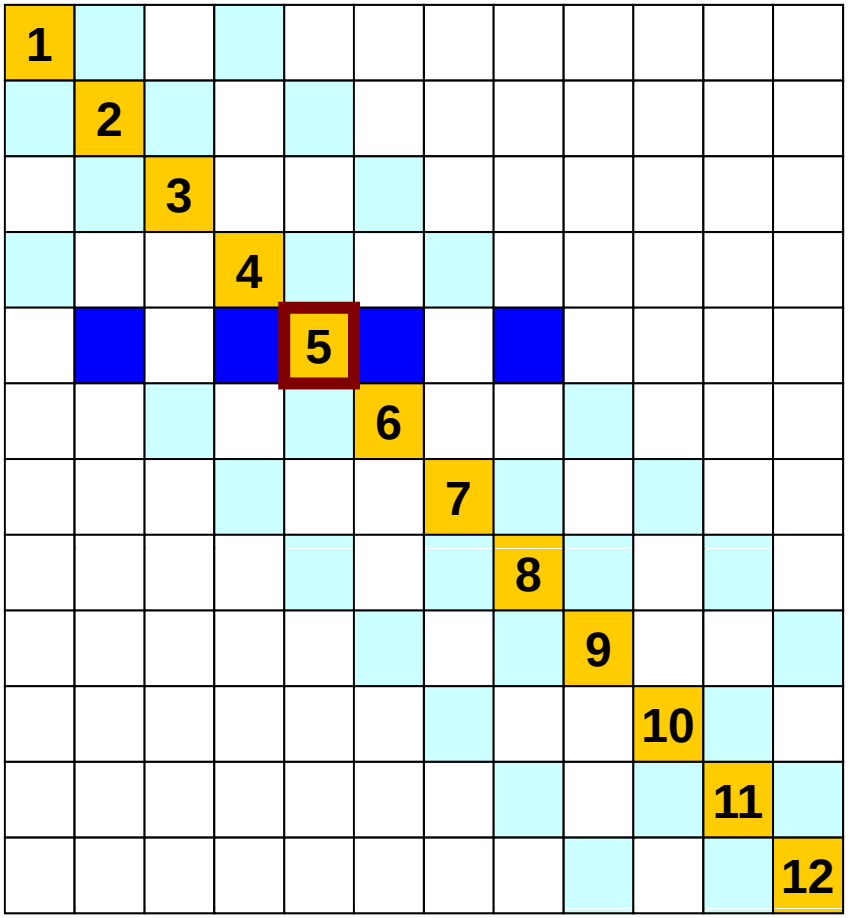
$\mathbf{L}$ $\mathbf{U}$

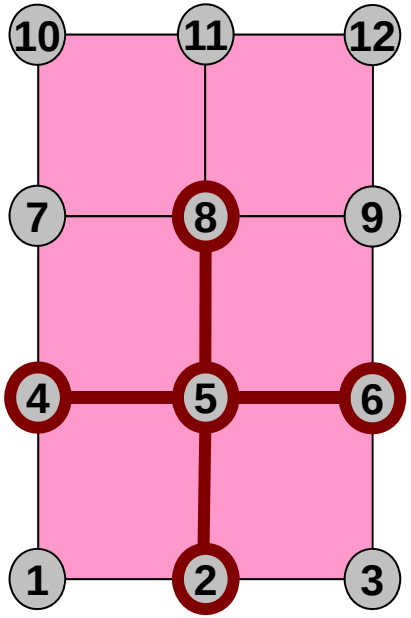
実例:5点差分





実例:5点差分



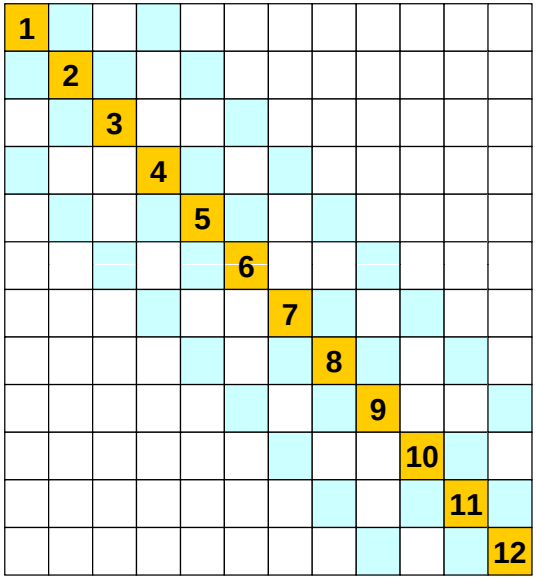


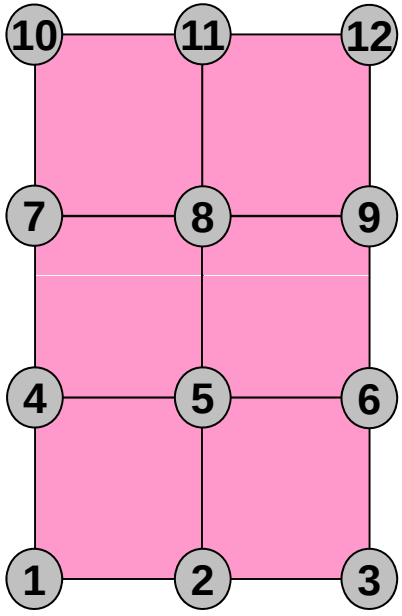
実例：係数マトリクス（対角成分=6.00）

6.00	−1.00	0.00	−1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
−1.00	6.00	−1.00	0.00	−1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
0.00	−1.00	6.00	0.00	0.00	−1.00	0.00	0.00	0.00	0.00	0.00	0.00
−1.00	0.00	0.00	6.00	−1.00	0.00	−1.00	0.00	0.00	0.00	0.00	0.00
0.00	−1.00	0.00	−1.00	6.00	−1.00	0.00	−1.00	0.00	0.00	0.00	0.00
0.00	0.00	−1.00	0.00	−1.00	6.00	0.00	0.00	−1.00	0.00	0.00	0.00
0.00	0.00	0.00	−1.00	0.00	0.00	6.00	−1.00	0.00	−1.00	0.00	0.00
0.00	0.00	0.00	0.00	−1.00	0.00	−1.00	6.00	−1.00	0.00	−1.00	0.00
0.00	0.00	0.00	0.00	0.00	−1.00	0.00	−1.00	6.00	0.00	0.00	−1.00
0.00	0.00	0.00	0.00	0.00	0.00	−1.00	0.00	0.00	6.00	−1.00	0.00
0.00	0.00	0.00	0.00	0.00	0.00	0.00	−1.00	0.00	−1.00	6.00	−1.00
0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	−1.00	0.00	−1.00	6.00

X =

0.00
3.00
10.00
11.00
10.00
19.00
20.00
16.00
28.00
42.00
36.00
52.00

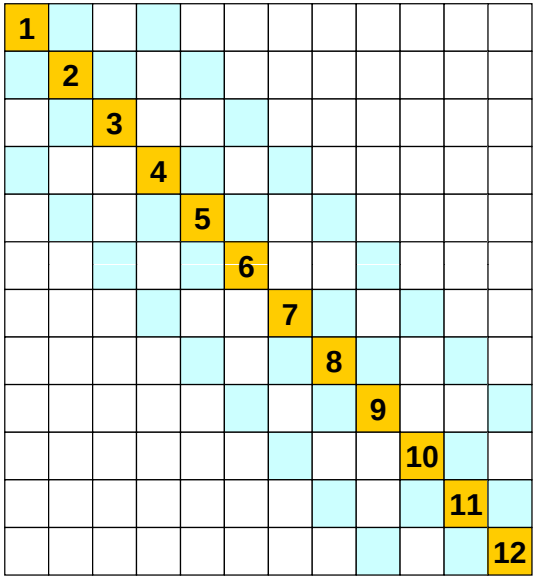


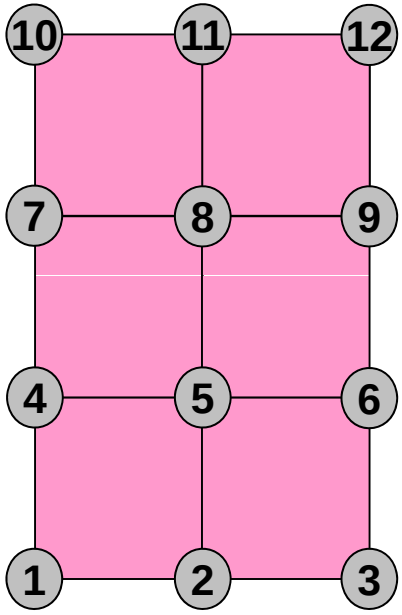


実例：解

6.00	−1.00	0.00	−1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
−1.00	6.00	−1.00	0.00	−1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
0.00	−1.00	6.00	0.00	0.00	−1.00	0.00	0.00	0.00	0.00	0.00	0.00
−1.00	0.00	0.00	6.00	−1.00	0.00	−1.00	0.00	0.00	0.00	0.00	0.00
0.00	−1.00	0.00	−1.00	6.00	−1.00	0.00	−1.00	0.00	0.00	0.00	0.00
0.00	0.00	−1.00	0.00	−1.00	6.00	0.00	0.00	−1.00	0.00	0.00	0.00
0.00	0.00	0.00	−1.00	0.00	0.00	6.00	−1.00	0.00	−1.00	0.00	0.00
0.00	0.00	0.00	0.00	−1.00	0.00	−1.00	6.00	−1.00	0.00	−1.00	0.00
0.00	0.00	0.00	0.00	0.00	−1.00	0.00	−1.00	6.00	0.00	0.00	−1.00
0.00	0.00	0.00	0.00	0.00	0.00	−1.00	0.00	0.00	6.00	−1.00	0.00
0.00	0.00	0.00	0.00	0.00	0.00	0.00	−1.00	0.00	−1.00	6.00	−1.00
0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	−1.00	0.00	−1.00	6.00

1.00	0.00
2.00	3.00
3.00	10.00
4.00	11.00
5.00	10.00
6.00	19.00
7.00	20.00
8.00	16.00
9.00	28.00
10.00	42.00
11.00	36.00
12.00	52.00





ファイルコピー，インストール

```
>$ cd <$fem1>
>$ cp /home03/skengon/Documents/class/fem1/lu.tar .

>$ tar xvf lu.tar
>$ cd lu

>$ g95 lu1.f –o lu1
>$ g95 lu2.f –o lu2
>$ g95 lu3.f –o lu3
```

完全LU分解したマトリクス

./lu1 とタイプ

もとのマトリクス

6.00	−1.00	0.00	−1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
−1.00	6.00	−1.00	0.00	−1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
0.00	−1.00	6.00	0.00	0.00	−1.00	0.00	0.00	0.00	0.00	0.00	0.00
−1.00	0.00	0.00	6.00	−1.00	0.00	−1.00	0.00	0.00	0.00	0.00	0.00
0.00	−1.00	0.00	−1.00	6.00	−1.00	0.00	−1.00	0.00	0.00	0.00	0.00
0.00	0.00	−1.00	0.00	−1.00	6.00	0.00	0.00	−1.00	0.00	0.00	0.00
0.00	0.00	0.00	−1.00	0.00	0.00	6.00	−1.00	0.00	−1.00	0.00	0.00
0.00	0.00	0.00	0.00	−1.00	0.00	−1.00	6.00	−1.00	0.00	−1.00	0.00
0.00	0.00	0.00	0.00	0.00	−1.00	0.00	−1.00	6.00	0.00	0.00	−1.00
0.00	0.00	0.00	0.00	0.00	0.00	−1.00	0.00	0.00	6.00	−1.00	0.00
0.00	0.00	0.00	0.00	0.00	0.00	0.00	−1.00	0.00	−1.00	6.00	−1.00
0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	−1.00	0.00	−1.00	6.00

LU分解したマトリクス
[L][U]同時に表示
[L]対角成分(=1)省略
(fill-inが生じている。もともと0だった成分が非ゼロになっている)

6.00	−1.00	0.00	−1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
−0.17	5.83	−1.00	−0.17	−1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
0.00	−0.17	5.83	−0.03	−0.17	−1.00	0.00	0.00	0.00	0.00	0.00	0.00
−0.17	−0.03	0.00	5.83	−1.03	0.00	−1.00	0.00	0.00	0.00	0.00	0.00
0.00	−0.17	−0.03	−0.18	5.64	−1.03	−0.18	−1.00	0.00	0.00	0.00	0.00
0.00	0.00	−0.17	0.00	−0.18	5.64	−0.03	−0.18	−1.00	0.00	0.00	0.00
0.00	0.00	0.00	−0.17	−0.03	−0.01	5.82	−1.03	−0.01	−1.00	0.00	0.00
0.00	0.00	0.00	0.00	−0.18	−0.03	−0.18	5.63	−1.03	−0.18	−1.00	0.00
0.00	0.00	0.00	0.00	0.00	−0.18	0.00	−0.18	5.63	−0.03	−0.18	−1.00
0.00	0.00	0.00	0.00	0.00	0.00	−0.17	−0.03	−0.01	5.82	−1.03	−0.01
0.00	0.00	0.00	0.00	0.00	0.00	0.00	−0.18	−0.03	−0.18	5.63	−1.03
0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	−0.18	0.00	−0.18	5.63

不完全LU分解したマトリクス（fill-in無し）

./lu2 とタイプ

不完全LU分解した
マトリクス（fill-in無し）
[L][U]同時に表示
[L]対角成分（=1）省略

6.00	−1.00	0.00	−1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
−0.17	5.83	−1.00	0.00	−1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
0.00	−0.17	5.83	0.00	0.00	−1.00	0.00	0.00	0.00	0.00	0.00	0.00
−0.17	0.00	0.00	5.83	−1.00	0.00	−1.00	0.00	0.00	0.00	0.00	0.00
0.00	−0.17	0.00	−0.17	5.66	−1.00	0.00	−1.00	0.00	0.00	0.00	0.00
0.00	0.00	−0.17	0.00	−0.18	5.65	0.00	0.00	−1.00	0.00	0.00	0.00
0.00	0.00	0.00	−0.17	0.00	0.00	5.83	−1.00	0.00	−1.00	0.00	0.00
0.00	0.00	0.00	0.00	−0.18	0.00	−0.17	5.65	−1.00	0.00	−1.00	0.00
0.00	0.00	0.00	0.00	0.00	−0.18	0.00	−0.18	5.65	0.00	0.00	−1.00
0.00	0.00	0.00	0.00	0.00	0.00	−0.17	0.00	0.00	5.83	−1.00	0.00
0.00	0.00	0.00	0.00	0.00	0.00	0.00	−0.18	0.00	−0.17	5.65	−1.00
0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	−0.18	0.00	−0.18	5.65

完全LU分解した
マトリクス
[L][U]同時に表示
[L]対角成分（=1）省略
（fill-inが生じている。もともと0だった成分が非ゼロになっている）

6.00	−1.00	0.00	−1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
−0.17	5.83	−1.00	−0.17	−1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
0.00	−0.17	5.83	−0.03	−0.17	−1.00	0.00	0.00	0.00	0.00	0.00	0.00
−0.17	−0.03	0.00	5.83	−1.03	0.00	−1.00	0.00	0.00	0.00	0.00	0.00
0.00	−0.17	−0.03	−0.18	5.64	−1.03	−0.18	−1.00	0.00	0.00	0.00	0.00
0.00	0.00	−0.17	0.00	−0.18	5.64	−0.03	−0.18	−1.00	0.00	0.00	0.00
0.00	0.00	0.00	−0.17	−0.03	−0.01	5.82	−1.03	−0.01	−1.00	0.00	0.00
0.00	0.00	0.00	0.00	−0.18	−0.03	−0.18	5.63	−1.03	−0.18	−1.00	0.00
0.00	0.00	0.00	0.00	0.00	−0.18	0.00	−0.18	5.63	−0.03	−0.18	−1.00
0.00	0.00	0.00	0.00	0.00	0.00	−0.17	−0.03	−0.01	5.82	−1.03	−0.01
0.00	0.00	0.00	0.00	0.00	0.00	0.00	−0.18	−0.03	−0.18	5.63	−1.03
0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	−0.18	0.00	−0.18	5.63

解の比較：ちょっと違う

不完全LU分解

6.00	−1.00	0.00	−1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
−0.17	5.83	−1.00	0.00	−1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
0.00	−0.17	5.83	0.00	0.00	−1.00	0.00	0.00	0.00	0.00	0.00	0.00
−0.17	0.00	0.00	5.83	−1.00	0.00	−1.00	0.00	0.00	0.00	0.00	0.00
0.00	−0.17	0.00	−0.17	5.66	−1.00	0.00	−1.00	0.00	0.00	0.00	0.00
0.00	0.00	−0.17	0.00	−0.18	5.65	0.00	0.00	−1.00	0.00	0.00	0.00
0.00	0.00	0.00	−0.17	0.00	0.00	5.83	−1.00	0.00	−1.00	0.00	0.00
0.00	0.00	0.00	0.00	−0.18	0.00	−0.17	5.65	−1.00	0.00	−1.00	0.00
0.00	0.00	0.00	0.00	0.00	−0.18	0.00	−0.18	5.65	0.00	0.00	−1.00
0.00	0.00	0.00	0.00	0.00	0.00	−0.17	0.00	0.00	5.83	−1.00	0.00
0.00	0.00	0.00	0.00	0.00	0.00	0.00	−0.18	0.00	−0.17	5.65	−1.00
0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	−0.18	0.00	−0.18	5.65

→

0.92
1.75
2.76
3.79
4.46
5.57
6.66
7.25
8.46
9.66
10.54
11.83

完全LU分解

6.00	−1.00	0.00	−1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
−0.17	5.83	−1.00	−0.17	−1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
0.00	−0.17	5.83	−0.03	−0.17	−1.00	0.00	0.00	0.00	0.00	0.00	0.00
−0.17	−0.03	0.00	5.83	−1.03	0.00	−1.00	0.00	0.00	0.00	0.00	0.00
0.00	−0.17	−0.03	−0.18	5.64	−1.03	−0.18	−1.00	0.00	0.00	0.00	0.00
0.00	0.00	−0.17	0.00	−0.18	5.64	−0.03	−0.18	−1.00	0.00	0.00	0.00
0.00	0.00	0.00	−0.17	−0.03	−0.01	5.82	−1.03	−0.01	−1.00	0.00	0.00
0.00	0.00	0.00	0.00	−0.18	−0.03	−0.18	5.63	−1.03	−0.18	−1.00	0.00
0.00	0.00	0.00	0.00	0.00	−0.18	0.00	−0.18	5.63	−0.03	−0.18	−1.00
0.00	0.00	0.00	0.00	0.00	0.00	−0.17	−0.03	−0.01	5.82	−1.03	−0.01
0.00	0.00	0.00	0.00	0.00	0.00	0.00	−0.18	−0.03	−0.18	5.63	−1.03
0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	−0.18	0.00	−0.18	5.63

→

1.00
2.00
3.00
4.00
5.00
6.00
7.00
8.00
9.00
10.00
11.00
12.00

ILU(0), IC(0) 前処理

- Fill-inを全く考慮しない「不完全な」分解
  - 記憶容量，計算量削減
- これを解くと「不完全な」解が得られるが，本来の解とそれほどずれているわけではない
  - 問題に依存する

Full LU and ILU(0)/IC(0)

Full LU

```
do i= 2, n
  do k= 1,  i-1
    a_ik := a_ik/a_kk
    do j= k+1, n
      a_ij := a_ij - a_ik*a_kj
    enddo
  enddo
enddo
```

ILU(0) : keep non-zero pattern of the original coefficient matrix

```
do i= 2, n
  do k= 1,  i-1
    if ((i,k)∈ NonZero(A)) then
      a_ik := a_ik/a_kk
    endif
    do j= k+1, n
      if ((i,j)∈ NonZero(A)) then
        a_ij := a_ij - a_ik*a_kj
      endif
    enddo
  enddo
enddo
```

Deep Fill-in：ILU(p)/IC(p)

p（fill-inレベル）を大きくすると完全LU分解に近づき，前処理としては安定するが，計算時間を要する：トレードオフ

LEV_{ij}=0 if ((i,j)∈ NonZero(A)) otherwise LEV_{ij}= p+1

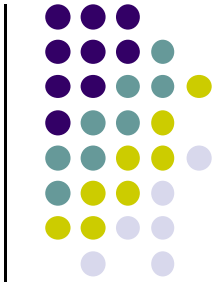
```
do i= 2, n
  do k= 1,  i-1
    if (LEV_ik≦p) then
      a_ik := a_ik/a_kk
    endif
    do j= k+1, n
      if (LEV_ij = min(LEV_ij, 1+LEV_ik+ LEV_kj)≦p) then
        a_ij := a_ij - a_ik*a_kj
      endif
    enddo
  enddo
enddo
```

LU Gauss-Seidel (LU-GS)
LU Symmetric GS (LU-SGS)
本講義で扱う手法

- ILU(0)

```
do i= 2, n
    do k= 1,  i-1
      if ((i,k)∈ NonZero(A)) then
        a_ik := a_ik/a_kk
      endif
      do j= k+1, n
        if ((i,j)∈ NonZero(A)) then
          a_ij := a_ij - a_ik*a_kj
        endif
      enddo
    enddo
  enddo
```

LU Gauss-Seidel (LU-GS)
LU Symmetric GS (LU-SGS)
本講義で扱う手法



- 更なる簡易版，フィルイン無し

```
do i= 2, n
    do k= 1,  i-1
      if ((i,k)∈ NonZero(A)) then
        a_ik := a_ik/a_kk
      endif
      do j= k+1, n
        if ((i,j)∈ NonZero(A)) then
          a_ij := a_ij - a_ik*a_kj
        endif
      enddo
    enddo
  enddo
```

Only do this

LU Gauss-Seidel (LU-GS)
LU Symmetric GS (LU-SGS)
本講義で扱う手法

- 更なる簡易版，フィルイン無し

$$\begin{pmatrix} 1 & 0 & 0 & \cdots & 0 \\ l_{21} & 1 & 0 & \cdots & 0 \\ l_{31} & l_{32} & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ l_{n1} & l_{n2} & l_{n3} & \cdots & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 \\ a_{21}/a_{22} & 1 & 0 & \cdots & 0 \\ a_{31}/a_{33} & a_{32}/a_{33} & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{n1}/a_{nn} & a_{n2}/a_{nn} & a_{n3}/a_{nn} & \cdots & 1 \end{pmatrix}$$

$$\begin{pmatrix} u_{11} & u_{12} & u_{13} & \cdots & u_{1n} \\ 0 & u_{22} & u_{23} & \cdots & u_{2n} \\ 0 & 0 & u_{33} & \cdots & u_{3n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & u_{nn} \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1n} \\ 0 & a_{22} & a_{23} & \cdots & a_{2n} \\ 0 & 0 & a_{33} & \cdots & a_{3n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & a_{nn} \end{pmatrix}$$

```
do i= 2, n
    do k= 1,  i-1
      if ((i,k)∈ NonZero(A)) then
        a_ik := a_ik/a_kk
      endif
      do j= k+1, n
        if ((i,j)∈ NonZero(A)) then
          a_ij := a_ij - a_ik*a_kj
        endif
      enddo
    enddo
  enddo
```

不完全LU分解，LU-GS

./lu3 とタイプ

不完全LU分解した
マトリクス（fill-in無し）

6.00	−1.00	0.00	−1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
−0.17	5.83	−1.00	0.00	−1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
0.00	−0.17	5.83	0.00	0.00	−1.00	0.00	0.00	0.00	0.00	0.00	0.00
−0.17	0.00	0.00	5.83	−1.00	0.00	−1.00	0.00	0.00	0.00	0.00	0.00
0.00	−0.17	0.00	−0.17	5.66	−1.00	0.00	−1.00	0.00	0.00	0.00	0.00
0.00	0.00	−0.17	0.00	−0.18	5.65	0.00	0.00	−1.00	0.00	0.00	0.00
0.00	0.00	0.00	−0.17	0.00	0.00	5.83	−1.00	0.00	−1.00	0.00	0.00
0.00	0.00	0.00	0.00	−0.18	0.00	−0.17	5.65	−1.00	0.00	−1.00	0.00
0.00	0.00	0.00	0.00	0.00	−0.18	0.00	−0.18	5.65	0.00	0.00	−1.00
0.00	0.00	0.00	0.00	0.00	0.00	−0.17	0.00	0.00	5.83	−1.00	0.00
0.00	0.00	0.00	0.00	0.00	0.00	0.00	−0.18	0.00	−0.17	5.65	−1.00
0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	−0.18	0.00	−0.18	5.65

LU-GS
（fill-in無し）

6.00	−1.00	0.00	−1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
−0.17	6.00	−1.00	0.00	−1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
0.00	−0.17	6.00	0.00	0.00	−1.00	0.00	0.00	0.00	0.00	0.00	0.00
−0.17	0.00	0.00	6.00	−1.00	0.00	−1.00	0.00	0.00	0.00	0.00	0.00
0.00	−0.17	0.00	−0.17	6.00	−1.00	0.00	−1.00	0.00	0.00	0.00	0.00
0.00	0.00	−0.17	0.00	−0.17	6.00	0.00	0.00	−1.00	0.00	0.00	0.00
0.00	0.00	0.00	−0.17	0.00	0.00	6.00	−1.00	0.00	−1.00	0.00	0.00
0.00	0.00	0.00	0.00	−0.17	0.00	−0.17	6.00	−1.00	0.00	−1.00	0.00
0.00	0.00	0.00	0.00	0.00	−0.17	0.00	−0.17	6.00	0.00	0.00	−1.00
0.00	0.00	0.00	0.00	0.00	0.00	−0.17	0.00	0.00	6.00	−1.00	0.00
0.00	0.00	0.00	0.00	0.00	0.00	0.00	−0.17	0.00	−0.17	6.00	−1.00
0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	−0.17	0.00	−0.17	6.00

解の比較：更にちょっと違う

不完全LU分解

6.00	−1.00	0.00	−1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
−0.17	5.83	−1.00	0.00	−1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
0.00	−0.17	5.83	0.00	0.00	−1.00	0.00	0.00	0.00	0.00	0.00	0.00
−0.17	0.00	0.00	5.83	−1.00	0.00	−1.00	0.00	0.00	0.00	0.00	0.00
0.00	−0.17	0.00	−0.17	5.66	−1.00	0.00	−1.00	0.00	0.00	0.00	0.00
0.00	0.00	−0.17	0.00	−0.18	5.65	0.00	0.00	−1.00	0.00	0.00	0.00
0.00	0.00	0.00	−0.17	0.00	0.00	5.83	−1.00	0.00	−1.00	0.00	0.00
0.00	0.00	0.00	0.00	−0.18	0.00	−0.17	5.65	−1.00	0.00	−1.00	0.00
0.00	0.00	0.00	0.00	0.00	−0.18	0.00	−0.18	5.65	0.00	0.00	−1.00
0.00	0.00	0.00	0.00	0.00	0.00	−0.17	0.00	0.00	5.83	−1.00	0.00
0.00	0.00	0.00	0.00	0.00	0.00	0.00	−0.18	0.00	−0.17	5.65	−1.00
0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	−0.18	0.00	−0.18	5.65

→

0.92
1.75
2.76
3.79
4.46
5.57
6.66
7.25
8.46
9.66
10.54
11.83

LU-GS

6.00	−1.00	0.00	−1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
−0.17	6.00	−1.00	0.00	−1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
0.00	−0.17	6.00	0.00	0.00	−1.00	0.00	0.00	0.00	0.00	0.00	0.00
−0.17	0.00	0.00	6.00	−1.00	0.00	−1.00	0.00	0.00	0.00	0.00	0.00
0.00	−0.17	0.00	−0.17	6.00	−1.00	0.00	−1.00	0.00	0.00	0.00	0.00
0.00	0.00	−0.17	0.00	−0.17	6.00	0.00	0.00	−1.00	0.00	0.00	0.00
0.00	0.00	0.00	−0.17	0.00	0.00	6.00	−1.00	0.00	−1.00	0.00	0.00
0.00	0.00	0.00	0.00	−0.17	0.00	−0.17	6.00	−1.00	0.00	−1.00	0.00
0.00	0.00	0.00	0.00	0.00	−0.17	0.00	−0.17	6.00	0.00	0.00	−1.00
0.00	0.00	0.00	0.00	0.00	0.00	−0.17	0.00	0.00	6.00	−1.00	0.00
0.00	0.00	0.00	0.00	0.00	0.00	0.00	−0.17	0.00	−0.17	6.00	−1.00
0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	−0.17	0.00	−0.17	6.00

→

0.86
1.60
2.60
3.54
3.99
5.09
6.26
6.52
7.73
9.22
9.70
10.96

LU-GS分解による前進後退代入

$$[M]\{z\}=[\tilde{L}\tilde{U}]\{z\}=\{r\}$$

$$\{z\}=[\tilde{L}\tilde{U}]^{-1}\{r\} \longrightarrow \begin{aligned}[\tilde{L}]\{y\}&=\{r\}\\ [\tilde{U}]\{z\}&=\{y\}\end{aligned}$$

$$[\tilde{L}]=\begin{pmatrix}1 & 0 & 0 & \cdots & 0\\ l_{21} & 1 & 0 & \cdots & 0\\ l_{31} & l_{32} & 1 & \cdots & 0\\ \vdots & \vdots & \vdots & \ddots & \vdots\\ l_{n1} & l_{n2} & l_{n3} & \cdots & 1\end{pmatrix}=\begin{pmatrix}1 & 0 & 0 & \cdots & 0\\ a_{21}/a_{22} & 1 & 0 & \cdots & 0\\ a_{31}/a_{33} & a_{32}/a_{33} & 1 & \cdots & 0\\ \vdots & \vdots & \vdots & \ddots & \vdots\\ a_{n1}/a_{nn} & a_{n2}/a_{nn} & a_{n3}/a_{nn} & \cdots & 1\end{pmatrix}$$

$$[\tilde{U}]=\begin{pmatrix}u_{11} & u_{12} & u_{13} & \cdots & u_{1n}\\ 0 & u_{22} & u_{23} & \cdots & u_{2n}\\ 0 & 0 & u_{33} & \cdots & u_{3n}\\ \vdots & \vdots & \vdots & \ddots & \vdots\\ 0 & 0 & 0 & \cdots & u_{nn}\end{pmatrix}=\begin{pmatrix}a_{11} & a_{12} & a_{13} & \cdots & a_{1n}\\ 0 & a_{22} & a_{23} & \cdots & a_{2n}\\ 0 & 0 & a_{33} & \cdots & a_{3n}\\ \vdots & \vdots & \vdots & \ddots & \vdots\\ 0 & 0 & 0 & \cdots & a_{nn}\end{pmatrix}$$

$$\left[\overline{L}\right]=\begin{pmatrix} 0 & 0 & 0 & \cdots & 0 \\ a_{21} & 0 & 0 & \cdots & 0 \\ a_{31} & a_{32} & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & a_{n3} & \cdots & 0 \end{pmatrix}$$

$$\left[\overline{U}\right]=\begin{pmatrix} 0 & a_{12} & a_{13} & \cdots & a_{1n} \\ 0 & 0 & a_{23} & \cdots & a_{2n} \\ 0 & 0 & 0 & \cdots & a_{3n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 0 \end{pmatrix}$$

$$\left[\overline{D}\right]=\begin{pmatrix} a_{11} & 0 & 0 & \cdots & 0 \\ 0 & a_{22} & 0 & \cdots & 0 \\ 0 & 0 & a_{33} & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & a_{nn} \end{pmatrix}$$

$$[M]=[\tilde{L}][\tilde{U}]=[\overline{L}+\overline{D}][\overline{D}^{-1}][\overline{D}+\overline{U}]=[\overline{L}\,\overline{D}^{-1}+I][\overline{D}+\overline{U}]$$

$$[\overline{L}\,\overline{D}^{-1}]+[I]=\begin{pmatrix} 0 & 0 & 0 & \cdots & 0 \\ a_{21}/a_{22} & 0 & 0 & \cdots & 0 \\ a_{31}/a_{33} & a_{32}/a_{33} & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{n1}/a_{nn} & a_{n2}/a_{nn} & a_{n3}/a_{nn} & \cdots & 0 \end{pmatrix}+\begin{pmatrix} 1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \end{pmatrix}$$

$$=\begin{pmatrix} 1 & 0 & 0 & \cdots & 0 \\ a_{21}/a_{22} & 1 & 0 & \cdots & 0 \\ a_{31}/a_{33} & a_{32}/a_{33} & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{n1}/a_{nn} & a_{n2}/a_{nn} & a_{n3}/a_{nn} & \cdots & 1 \end{pmatrix}=[\tilde{L}]$$

$$[\overline{D}]+[\overline{U}]=\begin{pmatrix} a_{11} & 0 & 0 & \cdots & 0 \\ 0 & a_{22} & 0 & \cdots & 0 \\ 0 & 0 & a_{33} & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & a_{nn} \end{pmatrix}+\begin{pmatrix} 0 & a_{12} & a_{13} & \cdots & a_{1n} \\ 0 & 0 & a_{23} & \cdots & a_{2n} \\ 0 & 0 & 0 & \cdots & a_{3n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 0 \end{pmatrix}=\begin{pmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1n} \\ 0 & a_{22} & a_{23} & \cdots & a_{2n} \\ 0 & 0 & a_{33} & \cdots & a_{3n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & a_{nn} \end{pmatrix}=[\tilde{U}]$$

LU-GSの前進交代代入

$$[M]=[\tilde{L}][\tilde{U}]=[\overline{L}+\overline{D}][\overline{D}^{-1}][\overline{D}+\overline{U}]=[\overline{L}\,\overline{D}^{-1}+I][\overline{D}+\overline{U}]$$

前進代入：**Forward Substitution**

$$[\overline{L}+\overline{D}]\{y\}=\{r\}\Rightarrow\{y\}=[\overline{D}^{-1}](\{r\}-[\overline{L}]\{y\})\Rightarrow y_i=\overline{D}_{ii}^{-1}\left(r_i-\sum_{j=1}^{i-1}\overline{L}_{ij}y_j\right)$$

後退代入：**Backward Substitution**

$$[I+\overline{D}^{-1}\overline{U}]\{z\}=\{y\}\Rightarrow\{z\}=\{y\}-[\overline{D}^{-1}][\overline{U}]\{z\}\Rightarrow z_i=y_i-\overline{D}_{ii}^{-1}\left[\sum_{j=i+1}^{N}\overline{U}_{ij}z_j\right]$$

$$[\overline{L}\,\overline{D}^{-1}]+[I]=\begin{pmatrix}0&0&0&\cdots&0\\ a_{21}/a_{22}&0&0&\cdots&0\\ a_{31}/a_{33}&a_{32}/a_{33}&0&\cdots&0\\ \vdots&\vdots&\vdots&\ddots&\vdots\\ a_{n1}/a_{nn}&a_{n2}/a_{nn}&a_{n3}/a_{nn}&\cdots&0\end{pmatrix}+\begin{pmatrix}1&0&0&\cdots&0\\ 0&1&0&\cdots&0\\ 0&0&1&\cdots&0\\ \vdots&\vdots&\vdots&\ddots&\vdots\\ 0&0&0&\cdots&1\end{pmatrix}=\begin{pmatrix}1&0&0&\cdots&0\\ a_{21}/a_{22}&1&0&\cdots&0\\ a_{31}/a_{33}&a_{32}/a_{33}&1&\cdots&0\\ \vdots&\vdots&\vdots&\ddots&\vdots\\ a_{n1}/a_{nn}&a_{n2}/a_{nn}&a_{n3}/a_{nn}&\cdots&1\end{pmatrix}$$

$$[\overline{D}]+[\overline{U}]=\begin{pmatrix}a_{11}&0&0&\cdots&0\\ 0&a_{22}&0&\cdots&0\\ 0&0&a_{33}&\cdots&0\\ \vdots&\vdots&\vdots&\ddots&\vdots\\ 0&0&0&\cdots&a_{nn}\end{pmatrix}+\begin{pmatrix}0&a_{12}&a_{13}&\cdots&a_{1n}\\ 0&0&a_{23}&\cdots&a_{2n}\\ 0&0&0&\cdots&a_{3n}\\ \vdots&\vdots&\vdots&\ddots&\vdots\\ 0&0&0&\cdots&0\end{pmatrix}=\begin{pmatrix}a_{11}&a_{12}&a_{13}&\cdots&a_{1n}\\ 0&a_{22}&a_{23}&\cdots&a_{2n}\\ 0&0&a_{33}&\cdots&a_{3n}\\ \vdots&\vdots&\vdots&\ddots&\vdots\\ 0&0&0&\cdots&a_{nn}\end{pmatrix}$$

LU-GS分解を使用した前進後退代入

$$[\tilde{L}]\{y\}=\{r\}$$

$$[\tilde{U}]\{z\}=\{y\}$$

```
!C
!C +----------------+
!C | {z}= [Minv]{r} |
!C +----------------+
!C===




      do i= 1, N
        WVAL= W(i,R)
        do k= indexL(i-1)+1, indexL(i)
          WVAL= WVAL -  AL(k) * W(itemL(k),Y)
        enddo
        W(i,Y)= WVAL / D(i)
      enddo

      do i= N, 1, -1
        SW  = 0.0d0
        do k= indexU(i), indexU(i-1)+1, -1
          SW= SW + AU(k) * W(itemU(k),Z)
        enddo
        W(i,Z)= W(i,Y) - SW / D(i)
      enddo
!C===
```

前進代入

自分より番号の小さい成分(下三角)の計算は終了している，しかし自分の値は更新されていない

$$y_i = \overline{D}_{ii}^{-1}\left(r_i - \sum_{j=1}^{i-1}\overline{L}_{ij}y_j\right)$$

後退代入

自分より番号の大きい成分(上三角)の計算は終了している，しかし自分の値は更新されていない

$$z_i = y_i - \overline{D}_{ii}^{-1}\left[\sum_{j=i+1}^{n}\overline{U}_{ij}z_j\right]$$

LU-GS分解を使用した前進後退代入

$$[\tilde{L}]\{y\}=\{r\}$$

$$[\tilde{U}]\{z\}=\{y\}$$

```
!C
!C +----------------+
!C | {z}= [Minv]{r} |
!C +----------------+
!C===




      do i= 1, N
        WVAL= W(i,R)
        do k= indexL(i-1)+1, indexL(i)
          WVAL= WVAL -  AL(k) * W(itemL(k),Y)
        enddo
        W(i,Y)= WVAL / D(i)
      enddo

      do i= N, 1, -1
        SW  = 0.0d0
        do k= indexU(i), indexU(i-1)+1, -1
          SW= SW + AU(k) * W(itemU(k),Z)
        enddo
        W(i,Z)= W(i,Y) - SW / D(i)
      enddo
!C===
```

前進代入

i=1のとき自分より番号の小さい成分は存在しない：

$$y_1 = \overline{D}_{11}^{-1} r_1$$

後退代入

i=nのとき自分より番号の大きい成分は存在しない：

$$z_n = y_n$$

LU-GS分解を使用した前進後退代入

$$[\tilde{L}]\{z\}=\{r\}$$

$$[\tilde{U}]\{z\}=\{z\}$$

```
!C
!C +----------------+
!C | {z}= [Minv]{r} |
!C +----------------+
!C===



      do i= 1, N
        WVAL= W(i,R)
        do k= indexL(i-1)+1, indexL(i)
          WVAL= WVAL -  AL(k) * W(itemL(k),Z)
        enddo
        W(i,Z)= WVAL / D(i)
      enddo

      do i= N, 1, -1
        SW  = 0.0d0
        do k= indexU(i), indexU(i-1)+1, -1
          SW= SW + AU(k) * W(itemU(k),Z)
        enddo
        W(i,Z)= W(i,Z) - SW / D(i)
      enddo
!C===
```

{y},{z}は一種類の配列を使えば済む。
「自分の値」はそれまで更新されない。

前進代入

自分より番号の小さい成分（下三角）の計算は終了している，しかし自分の値は更新されていない

$$z_i = \overline{D}_{ii}^{-1}\left(r_i - \sum_{j=1}^{i-1}\overline{L}_{ij}z_j\right)$$

後退代入

自分より番号の大きい成分（上三角）の計算は終了している，しかし自分の値は更新されていない

$$z_i = z_i - \overline{D}_{ii}^{-1}\left[\sum_{j=i+1}^{n}\overline{U}_{ij}z_j\right]$$

LU-GS分解を使用した前進後退代入

$$[\tilde{L}]\{z\}=\{z\}$$

$$[\tilde{U}]\{z\}=\{z\}$$

```
!C
!C +----------------+
!C | {z}= [Minv]{r} |
!C +----------------+
!C===
      do i= 1, N
        W(i,Z)= W(i,R)
      enddo

      do i= 1, N
        WVAL= W(i,Z)
        do k= indexL(i-1)+1, indexL(i)
          WVAL= WVAL -  AL(k) * W(itemL(k),Z)
        enddo
        W(i,Z)= WVAL / D(i)
      enddo

      do i= N, 1, -1
        SW  = 0.0d0
        do k= indexU(i), indexU(i-1)+1, -1
          SW= SW + AU(k) * W(itemU(k),Z)
        enddo
        W(i,Z)= W(i,Z) - SW / D(i)
      enddo
!C===
```

{r},{y},{z}は一種類の配列を使えば済む。
「自分の値」はそれまで更新されない。

$$z_1 = \overline{D}_{11}^{-1} r_1$$

前進代入

自分より番号の小さい成分（下三角）の計算は終了している，しかし自分の値は更新されていない

$$z_i = \overline{D}_{ii}^{-1}\left(z_i - \sum_{j=1}^{i-1} \overline{L}_{ij} z_j\right)$$

後退代入

自分より番号の大きい成分（上三角）の計算は終了している，しかし自分の値は更新されていない

$$z_i = z_i - \overline{D}_{ii}^{-1}\left[\sum_{j=i+1}^{n} \overline{U}_{ij} z_j\right]$$

LU-GS分解を使用した前進後退代入

$$\left[\tilde{L}\right]\{z\}=\{z\}$$

$$\left[\tilde{U}\right]\{z\}=\{z\}$$

```
!C
!C +----------------+
!C | {z}= [Minv]{r} |
!C +----------------+
!C===
      do i= 1, N
        W(i,Z)= W(i,R)
      enddo

      do i= 1, N
        WVAL= W(i,Z)
        do k= indexL(i-1)+1, indexL(i)
          WVAL= WVAL -  AL(k) * W(itemL(k),Z)
        enddo
        W(i,Z)= WVAL / D(i)
      enddo

      do i= N, 1, -1
        SW  = 0.0d0
        do k= indexU(i), indexU(i-1)+1, -1
          SW= SW + AU(k) * W(itemU(k),Z)
        enddo
        W(i,Z)= W(i,Z) - SW / D(i)
      enddo
!C===
```

$$z_i = \overline{D}_{ii}^{-1}\left(z_i - \sum_{j=1}^{i-1}\overline{L}_{ij}z_j\right)$$

$$\mathrm{WVAL} = z_i - \sum_{j=1}^{i-1}\overline{L}_{ij}z_j$$

$$z_i = z_i - \overline{D}_{ii}^{-1}\left[\sum_{j=i+1}^{N}\overline{U}_{ij}z_j\right]$$

$$\mathrm{SW} = \sum_{j=i+1}^{N}\overline{U}_{ij}z_j$$

LU-GS分解を使用した前進後退代入

$$[\tilde{L}]\{z\}=\{z\}$$

$$[\tilde{U}]\{z\}=\{z\}$$

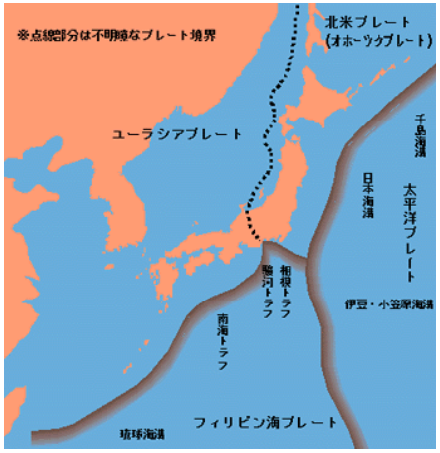
```
!C
!C +----------------+
!C | {z}= [Minv]{r} |
!C +----------------+
!C===
      do i= 1, N
        W(i,Z)= W(i,R)
      enddo

      do i= 1, N
        WVAL= W(i,Z)
        do k= indexL(i-1)+1, indexL(i)
          WVAL= WVAL -  AL(k) * W(itemL(k),Z)
        enddo
        W(i,Z)= WVAL / D(i)
      enddo

      do i= N, 1, -1
        SW  = 0.0d0
        do k= indexU(i), indexU(i-1)+1, -1
          SW= SW + AU(k) * W(itemU(k),Z)
        enddo
        W(i,Z)= W(i,Z) - SW / D(i)
      enddo
!C===
```

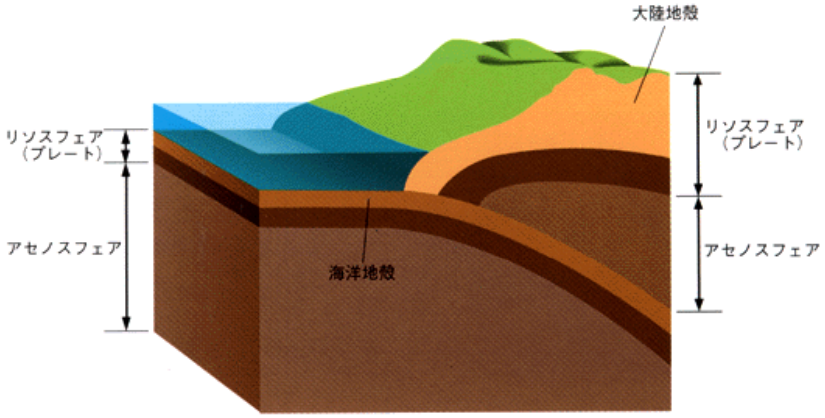
3×3の処理では
対角成分で割る代わりに
対角ブロック[D]の完全LU分解による前進後退代入

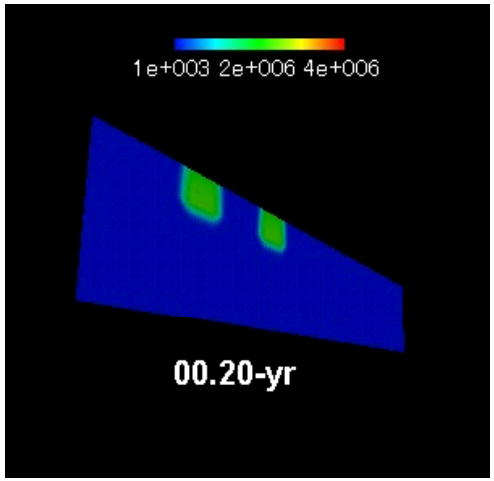
（この部分だけ厳密に行列を解く）

接触問題における前処理手法

- 地震発生サイクルシミュレーションにおける接触問題
  - プレート境界における準静的応力蓄積過程
  - 非線形接触問題をNewton-Raphson法によって解く
  - ALM法（Augmented Lagrangean,拡大ラグランジェ法）による拘束条件：ペナルティ数

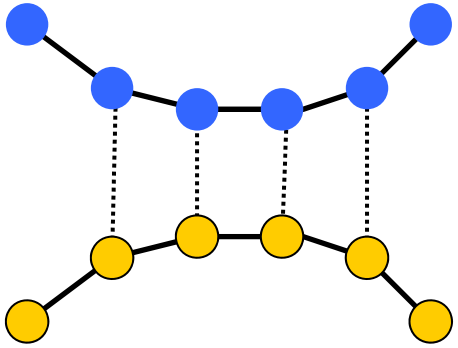




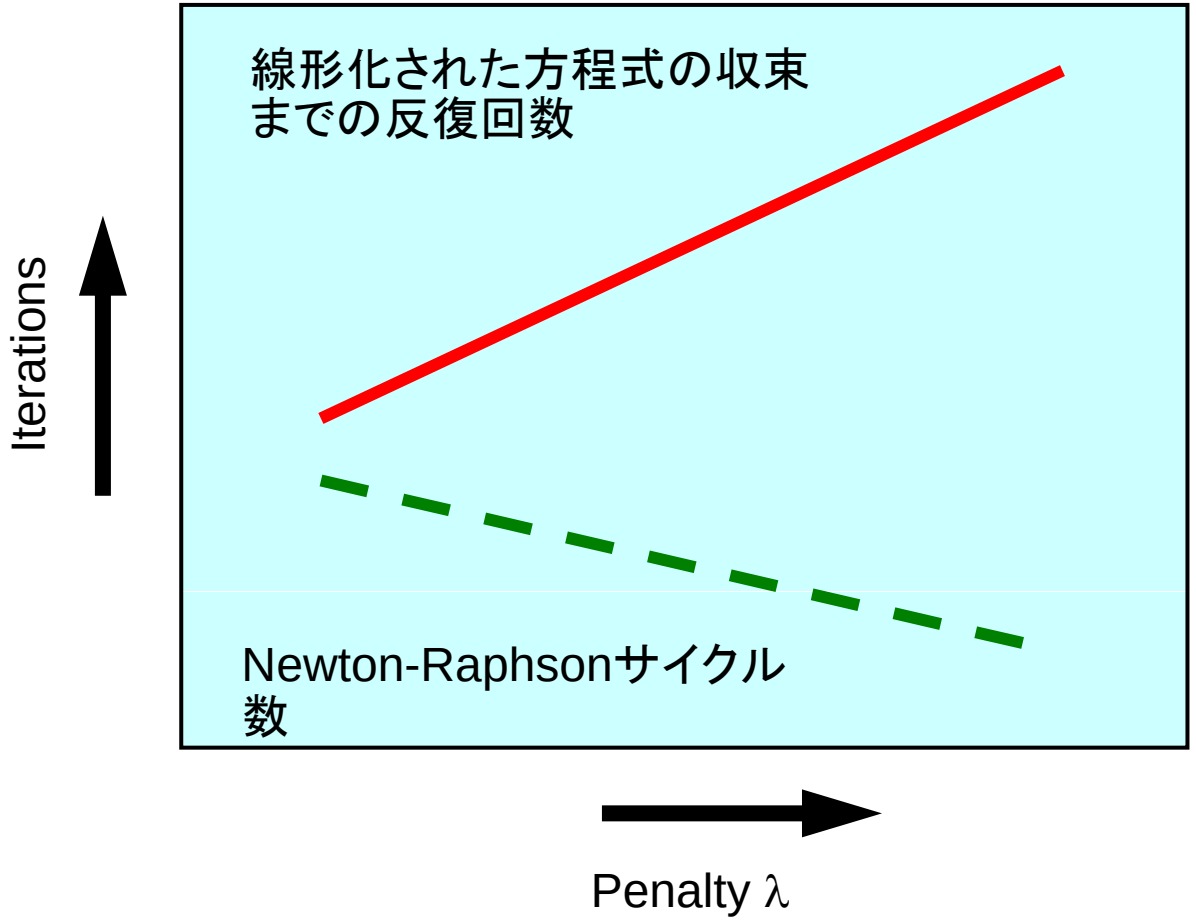


接触問題における前処理手法（続き）

- 仮定
  - 微小変形理論に基づく，静的接触（接触グループに属する節点の座標は同一）
  - 摩擦なし：対称行列（最終的には摩擦ありの場合も計算であったが）
- 特殊な前処理手法を開発：***Selective Blocking.***
  - 三次元接触問題において，効率的に解を得ることのできる，効率的な前処理手法である
- 計算
  - 日立SR2201（東大）：2001～2002
  - 地球シミュレータ：2002～2003
  - IBM SP-3, BG/L：2003～2005



拡大ラグランジェ法
接触問題におけるペナルティ～反復回数の関係
Newton-Raphson / Iterative Solver



ペナルティ数が大きいと，接触条件の精度は高くなり，Newton-Raphsonサイクルも少ない反復回数で収束する。

しかし，線形化された方程式は悪条件となる。

予備的計算結果

ペナルティ拘束条件を含む弾性解析

27,888 nodes, 83,664 DOFs, $\varepsilon=10^{-8}$

Single PE case (Xeon 2.8MHz)

GeoFEM's Original Solvers (Scalar Version)

Preconditioning	λ	Iterations	Set-up (sec.)	Solve (sec.)	Set-up+Solve (sec.)	Single Iteration (sec.)	Memory Size (MB)
Diagonal Scaling	10^2	1531	<0.01	75.1	75.1	0.049	119
	10^6	No Conv.	-	-	-	-	
IC(0) (Scalar Type)	10^2	401	0.02	39.2	39.2	0.098	119
	10^6	No Conv.	-	-	-	-	
BIC(0)	10^2	388	0.02	37.4	37.4	0.097	59
	10^6	2590	0.01	252.3	252.3	0.097	
BIC(1)	10^2	77	8.5	11.7	20.2	0.152	176
	10^6	78	8.5	11.8	20.3	0.152	
BIC(2)	10^2	59	16.9	13.9	30.8	0.236	319
	10^6	59	16.9	13.9	30.8	0.236	
SB-BIC(0)	10^0	114	0.10	12.9	13.0	0.113	67
	10^6	114	0.10	12.9	13.0	0.113	

悪条件問題(Ill-Conditioned Problems)

- 基本的に直接法を使うべき問題。
- しかし，直接法では並列計算において限界がある。
- 安定した前処理手法が必要

- 対策
  - 直接法にできるだけ近い前処理手法
    - 深いFill-in
  - ブロッキング
  - オーダリング

Fill-inが深くなるほど・・・

- 直接法に近づく。
- 必要メモリ量，計算量が増加する。

Blocking：LU-GSの前進交代代入

$$[M]=[\tilde{L}][\tilde{U}]=[\overline{L}+\overline{D}][\overline{D}^{-1}][\overline{D}+\overline{U}]=[\overline{L}\,\overline{D}^{-1}+I][\overline{D}+\overline{U}]$$

前進代入：**Forward Substitution**

$$[\overline{L}+\overline{D}]\{y\}=\{r\}\Rightarrow\{y\}=[\overline{D}^{-1}](\{r\}-[\overline{L}]\{y\})\Rightarrow y_i=\overline{D}_{ii}^{-1}\left(r_i-\sum_{j=1}^{i-1}\overline{L}_{ij}y_j\right)$$

後退代入：**Backward Substitution**

$$[I+\overline{D}^{-1}\overline{U}]\{z\}=\{y\}\Rightarrow\{z\}=\{y\}-[\overline{D}^{-1}][\overline{U}]\{z\}\Rightarrow z_i=y_i-\overline{D}_{ii}^{-1}\left[\sum_{j=i+1}^{N}\overline{U}_{ij}z_j\right]$$

- D^{-1}を乗ずるところで，対角成分で単に割るのではなく，3×3ブロックにLU分解を施す。
  - 三次元固体力学の場合
  - 1節点に強くカップルした変位3成分がある
  - 間接参照が減り，計算効率も上がる

Results in the Benchmark

27,888 nodes, 83,664 DOFs, $\varepsilon=10^{-8}$

Single PE case (Xeon 2.8MHz)

Effect of Blocking/Fill-in

Preconditioning	λ	Iterations	Set-up (sec.)	Solve (sec.)	Set-up+Solve (sec.)	Single Iteration (sec.)	Memory Size (MB)
Diagonal Scaling	10^2	1531	<0.01	75.1	75.1	0.049	119
	10^6	No Conv.	-	-	-	-	
IC(0) (Scalar Type)	10^2	401	0.02	39.2	39.2	0.098	119
	10^6	No Conv.	-	-	-	-	
BIC(0)	10^2	388	0.02	37.4	37.4	0.097	59
	10^6	2590	0.01	252.3	252.3	0.097	
BIC(1)	10^2	77	8.5	11.7	20.2	0.152	176
	10^6	78	8.5	11.8	20.3	0.152	
BIC(2)	10^2	59	16.9	13.9	30.8	0.236	319
	10^6	59	16.9	13.9	30.8	0.236	
SB-BIC(0)	10^0	114	0.10	12.9	13.0	0.113	67
	10^6	114	0.10	12.9	13.0	0.113	

ブロッキングとFill-inレベルの増加により，困難な問題が解けるようになった。

- 前処理について
- **ソルバー部分**
- 応力計算
- レポート課題2

SOLVE33（1/4）：INPUT_CNTL

```
#include <stdio.h>
#include <string.h>
#include <math.h>
#include "pfem_util.h"
#include "allocate.h"
extern FILE *fp_log;
extern void CG_3();
void SOLVE33()
{
        int i,j,k,ii,L;
        KREAL ALU[3][3];
        KREAL PW[3];
        double ALO;

        int  ERROR, ICFLAG=0;
        CHAR_LENGTH BUF;

/**
        +------------+
        | PARAMETERs |
        +------------+
**/
        ITER      = pfemIarray[0];
        METHOD    = pfemIarray[1];
        PRECOND   = pfemIarray[2];
        NSET      = pfemIarray[3];  0
        iterPREmax= pfemIarray[4];  使用せず
        NREST     = pfemIarray[5];  使用せず

        RESID     = pfemRarray[0];
        SIGMA_DIAG= pfemRarray[1];  1.0

        if( iterPREmax  < 1   ) iterPREmax= 1;
        if (iterPREmax  > 4   ) iterPREmax= 4;
```

制御ファイル入力：INPUT_CNTL

```
#include <stdio.h>
#include <stdlib.h>
#include "pfem_util.h"
/** **/
void INPUT_CNTL()
{
        FILE *fp;
        if( (fp=fopen("INPUT.DAT","r")) == NULL){
                fprintf(stdout,"input file cannot be opened!¥n");
                exit(1);
        }
        fscanf(fp,"%s",fname);
        fscanf(fp,"%d %d",&METHOD,&PRECOND);
        fscanf(fp,"%d",&iterPREmax);
        fscanf(fp,"%d",&ITER);
        fscanf(fp, "%lf %lf", &ELAST, &POISSON);
        fclose(fp);

        if( ( iterPREmax < 1 ) ){
                iterPREmax= 1;
        }
        if( ( iterPREmax > 4 ) ){
                iterPREmax= 4;
        }

        SIGMA_DIAG= 1.0;
        SIGMA     = 0.0;
        RESID     = 1.e-8;
        NSET      = 0;

        pfemRarray[0]= RESID;
        pfemRarray[1]= SIGMA_DIAG;
        pfemRarray[2]= SIGMA;

        pfemIarray[0]= ITER;
        pfemIarray[1]= METHOD;
        pfemIarray[2]= PRECOND;
        pfemIarray[3]= NSET;
        pfemIarray[4]= iterPREmax;
}
```

SOLVE33（2/4）

```
/**
        +-----------+
        | BLOCK LUs |
        +-----------+
**/
        if( ICFLAG == 0 ){
                ALUG  =(KREAL*)allocate_vector(sizeof(KREAL),9*N);
                ICFLAG= 1;
                strcpy(BUF.name,"### LINEAR SOLVER:  3x3 Block" );

                if (METHOD == 1) strcat(BUF.name,"ssCG");
                if (METHOD == 2) strcat(BUF.name,"ssBiCGSTAB");

                if (PRECOND == 0) {
                    strcat(BUF.name,"BILU(0)-no ASDD");
                }

                if (PRECOND != 0) strcat(BUF.name,"Block Scaling");

                        fprintf(stdout,"%s¥n",BUF.name);
                        fprintf(fp_log,"%s¥n",BUF.name);
        }
```

SOLVE33（3/4）

```
      if( NSET == 0 ){
           for(i=0;i<9*N;i++) ALUG[i]=0.0;
                for( ii=0;ii<N;ii++){
                     ALU[0][0]= D[9*ii  ]*SIGMA_DIAG;
                     ALU[0][1]= D[9*ii+1];
                     ALU[0][2]= D[9*ii+2];
                     ALU[1][0]= D[9*ii+3];
                     ALU[1][1]= D[9*ii+4]*SIGMA_DIAG;
                     ALU[1][2]= D[9*ii+5];
                     ALU[2][0]= D[9*ii+6];
                     ALU[2][1]= D[9*ii+7];
                     ALU[2][2]= D[9*ii+8]*SIGMA_DIAG;

                     for(k=1;k<=3;k++){
                          L=k;
                          ALO=fabs(ALU[L-1][k-1]);
                          for( i=k+1;i<=3;i++){
                               if( fabs(ALU[i-1][k-1]) > ALO ){
                                    L=i;
                                    ALO=fabs(ALU[L-1][k-1]);}
                          }
                          ALU[k-1][k-1]= 1.e0/ALU[k-1][k-1];
                          for(i=k+1;i<=3;i++){
                               ALU[i-1][k-1]*=ALU[k-1][k-1];
                                    for(j=k+1;j<=3;j++){
                                         PW[j-1]=ALU[i-1][j-1] - ALU[i-1][k-1]*ALU[k-1][j-1];}
                                    for(j=k+1;j<=3;j++){
                                         ALU[i-1][j-1]=PW[j-1];}
                          }
                     }
                     ALUG[9*ii  ]=ALU[0][0];
                     ALUG[9*ii+1]=ALU[0][1];
                     ALUG[9*ii+2]=ALU[0][2];
                     ALUG[9*ii+3]=ALU[1][0];
                     ALUG[9*ii+4]=ALU[1][1];
                     ALUG[9*ii+5]=ALU[1][2];
                     ALUG[9*ii+6]=ALU[2][0];
                     ALUG[9*ii+7]=ALU[2][1];
                     ALUG[9*ii+8]=ALU[2][2];
           }
      }
```

ALUG：Dの完全LU分解
SIGMA_DIAG= 1.0
（INPUT_CNTL）

SOLVE33（3/4）

```
      if( NSET == 0 ){
           for(i=0;i<9*N;i++) ALUG[i]=0.0;
                for( ii=0;ii<N;ii++){
                     ALU[0][0]= D[9*ii  ]*SIGMA_DIAG;
                     ALU[0][1]= D[9*ii+1];
                     ALU[0][2]= D[9*ii+2];
                     ALU[1][0]= D[9*ii+3];
                     ALU[1][1]= D[9*ii+4]*SIGMA_DIAG;
                     ALU[1][2]= D[9*ii+5];
                     ALU[2][0]= D[9*ii+6];
                     ALU[2][1]= D[9*ii+7];
                     ALU[2][2]= D[9*ii+8]*SIGMA_DIAG;

                     for(k=1;k<=3;k++){
                          L=k;
                          ALO=fabs(ALU[L-1][k-1]);
                          for( i=k+1;i<=3;i++){
                               if( fabs(ALU[i-1][k-1]) > ALO ){
                                    L=i;
                                    ALO=fabs(ALU[L-1][k-1]);}
                          }
                          ALU[k-1][k-1]= 1.e0/ALU[k-1][k-1];
                          for(i=k+1;i<=3;i++){
                               ALU[i-1][k-1]*=ALU[k-1][k-1];
                                    for(j=k+1;j<=3;j++){
                                         PW[j-1]=ALU[i-1][j-1] - ALU[i-1][k-1]*ALU[k-1][j-1];}
                                    for(j=k+1;j<=3;j++){
                                         ALU[i-1][j-1]=PW[j-1];}
                          }
                     }
                     ALUG[9*ii  ]=ALU[0][0];
                     ALUG[9*ii+1]=ALU[0][1];
                     ALUG[9*ii+2]=ALU[0][2];
                     ALUG[9*ii+3]=ALU[1][0];
                     ALUG[9*ii+4]=ALU[1][1];
                     ALUG[9*ii+5]=ALU[1][2];
                     ALUG[9*ii+6]=ALU[2][0];
                     ALUG[9*ii+7]=ALU[2][1];
                     ALUG[9*ii+8]=ALU[2][2];
           }
      }
```

ALUG：Dの完全LU分解
SIGMA_DIAG= 1.0
（INPUT_CNTL）

SOLVE33（3/4）

```
      if( NSET == 0 ){
           for(i=0;i<9*N;i++) ALUG[i]=0.0;
                for( ii=0;ii<N;ii++){
                     ALU[0][0]= D[9*ii  ]*SIGMA_DIAG;
                     ALU[0][1]= D[9*ii+1];
                     ALU[0][2]= D[9*ii+2];
                     ALU[1][0]= D[9*ii+3];
                     ALU[1][1]= D[9*ii+4]*SIGMA_DIAG;
                     ALU[1][2]= D[9*ii+5];
                     ALU[2][0]= D[9*ii+6];
                     ALU[2][1]= D[9*ii+7];
                     ALU[2][2]= D[9*ii+8]*SIGMA_DIAG;

                     for(k=1;k<=3;k++){
                          L=k;
                          ALO=fabs(ALU[L-1][k-1]);
                          for( i=k+1;i<=3;i++){
                               if( fabs(ALU[i-1][k-1]) > ALO ){
                                    L=i;
                                    ALO=fabs(ALU[L-1][k-1]);}
                          }
                          ALU[k-1][k-1]= 1.e0/ALU[k-1][k-1];
                          for(i=k+1;i<=3;i++){
                               ALU[i-1][k-1]*=ALU[k-1][k-1];
                                    for(j=k+1;j<=3;j++){
                                         PW[j-1]=ALU[i-1][j-1] - ALU[i-1][k-1]*ALU[k-1][j-1];}
                                    for(j=k+1;j<=3;j++){
                                         ALU[i-1][j-1]=PW[j-1];}
                          }
                     }
                     ALUG[9*ii  ]=ALU[0][0];
                     ALUG[9*ii+1]=ALU[0][1];
                     ALUG[9*ii+2]=ALU[0][2];
                     ALUG[9*ii+3]=ALU[1][0];
                     ALUG[9*ii+4]=ALU[1][1];
                     ALUG[9*ii+5]=ALU[1][2];
                     ALUG[9*ii+6]=ALU[2][0];
                     ALUG[9*ii+7]=ALU[2][1];
                     ALUG[9*ii+8]=ALU[2][2];
      }
}
```

LU分解・完全pivoting付き
絶対値の大きい成分が分母となるようにする

ALUG：Dの完全LU分解

Pivoting

完全LU分解

$$\begin{pmatrix} D(9*i) & D(9*i+1) & D(9*i+2) \\ D(9*i+3) & D(9*i+4) & D(9*i+5) \\ D(9*i+6) & D(9*i+7) & D(9*i+8) \end{pmatrix} =$$

$$\begin{pmatrix} d_{11} & d_{12} & d_{13} \\ d_{21} & d_{22} & d_{23} \\ d_{31} & d_{32} & d_{33} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ l_{21} & 1 & 0 \\ l_{31} & l_{32} & 1 \end{pmatrix} \begin{pmatrix} u_{11} & u_{12} & u_{13} \\ 0 & u_{22} & u_{23} \\ 0 & 0 & u_{33} \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ ALUG(9*i+3) & 1 & 0 \\ ALUG(9*i+6) & ALUG(9*i+7) & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ l_{21} & 1 & 0 \\ l_{31} & l_{32} & 1 \end{pmatrix}$$

$$\begin{pmatrix} ALUG(9*i) & ALUG(9*i+1) & ALUG(9*i+2) \\ 0 & ALUG(9*i+4) & ALUG(9*i+5) \\ 0 & 0 & ALUG(9*i+8) \end{pmatrix} = \begin{pmatrix} 1/u_{11} & u_{12} & u_{13} \\ 0 & 1/u_{22} & u_{23} \\ 0 & 0 & 1/u_{33} \end{pmatrix}$$

SOLVE33 (4/4)

```
/***
        +------------------+
        | ITERATIVE solver |
        +------------------+
***/
        if (METHOD == 1 ) {
                CG_3( N, NP, NPL, NPU, D, AL, indexL, itemL, AU, indexU, itemU,
                        B, X,  ALUG, RESID, ITER, &ERROR,
                        PRECOND, iterPREmax);
        }

        ITERactual= ITER;
}
```

CG_3 (1/2)

```
/***
 *** CG_3
 ***/
#include <stdio.h>
#include <math.h>
#include "precision.h"
#include "allocate.h"
extern FILE *fp_log;
/***
        CG_3 solves the linear system Ax = b with 3*3 block matrix
        using the Conjugate Gradient iterative method with the following
        preconditioners for SMP nodes:
 ***/
void  CG_3(
                KINT N,KINT NP,KINT NPL, KINT NPU,KREAL D[],
                KREAL AL[],KINT INL[], KINT IAL[],
                KREAL AU[],KINT INU[], KINT IAU[],
                KREAL B[],KREAL X[],KREAL ALU[],
                KREAL RESID,KINT ITER, KINT *ERROR,
                KINT  PRECOND, KINT iterPREmax)
{
        int i,j,k;
        int ieL,isL,ieU,isU;
        double X1,X2,X3;
        double WVAL1,WVAL2,WVAL3;
        double SW1,SW2,SW3;
        double WV1,WV2,WV3;
        double BNRM20,BNRM2,DNRM20,DNRM2;
        double S1_TIME,E1_TIME;
        double ALPHA,BETA;
        double C1,C10,RHO,RHO0,RHO1;
        int    iterPRE;
        int    indexA,indexB;

        KREAL **WW;
        KINT R=0,Z=1,Q=1,P=2,ZP=3;
        KINT MAXIT;
        KREAL TOL;

        double COMPtime;
```

Variables/Arrays

Global	種別	サイズ	CG_3
N,NP,NPL,NPU	I		N,NP,NPL,NPU
D,B,X	R	[9*N]	D,B,X
AL,AU	R	[9*NPL],[9*NPU]	AL, AU
indexL, indexU	I	[N+1]	INL, INU
itemL, itemU	I	[NPL], [NPU]	IAL, IAU
ALUG	R	[9*N]	ALU
RESID	R		RESID
ITER	I		ITER
PRECOND	I		PRECOND
iterPREmax	I		iterPREmax
	R	[4][3*N]	WW

CG_3 (2/2)

```
/***
        +-------+
        | INIT. |
        +-------+
***/
        ERROR= 0;

        WW=(KREAL**) allocate_matrix(sizeof(KREAL),4,3*N);

        MAXIT  = ITER;
        TOL    = RESID;

        for(i=0;i<3*N;i++){
                X[i]=0.0;
        }
        for(j=0;j<4;j++) for(i=0;i<3*N;i++) WW[j][i]=0.0;
```

```
KINT R =0  WW[0][i]: {r}
KINT Z =1  WW[1][i]: {z}
KINT Q =1  WW[1][i]: {q}
KINT P =2  WW[2][i]: {p}
KINT ZP=3  WW[3][i]
```

Compute $r^{(0)}= b-[A]x^{(0)}$
for i= 1, 2, …
solve $[M]z^{(i-1)}= r^{(i-1)}$
$\rho_{i-1}= r^{(i-1)} z^{(i-1)}$
if i=1
$p^{(1)}= z^{(0)}$
else
$\beta_{i-1}= \rho_{i-1}/\rho_{i-2}$
$p^{(i)}= z^{(i-1)} + \beta_{i-1} p^{(i-1)}$
endif
$q^{(i)}= [A]p^{(i)}$
$\alpha_i = \rho_{i-1}/p^{(i)}q^{(i)}$
$x^{(i)}= x^{(i-1)} + \alpha_i p^{(i)}$
$r^{(i)}= r^{(i-1)} - \alpha_i q^{(i)}$
check convergence |r|
end

前処理：LU-GS（1/3）

```
/**
        +----------------+
        | {z}= [Minv]{r} |
        +----------------+
**/
    if( PRECOND == 0 ){
/**
    Block SSOR
**/
         for( i=1;i<=N;i++){
              WW[ZP][3*i-3]=WW[R][3*i-3];
              WW[ZP][3*i-2]=WW[R][3*i-2];
              WW[ZP][3*i-1]=WW[R][3*i-1];}
         for( i=1;i<=N;i++){
              WW[Z][3*i-3]=0.e0;
              WW[Z][3*i-2]=0.e0;
              WW[Z][3*i-1]=0.e0;}
```

前処理： LU-GS （2/3）

```
/**
     FORWARD
**/
          for( i=1;i<=N;i++){
               SW1= WW[ZP][3*i-3];
               SW2= WW[ZP][3*i-2];
               SW3= WW[ZP][3*i-1];

               isL=INL[i-1]+1;
               ieL=INL[i];
               for(j=isL;j<=ieL;j++){
                    k=IAL[j-1];
                         X1= WW[ZP][3*k-3];
                         X2= WW[ZP][3*k-2];
                         X3= WW[ZP][3*k-1];

                         SW1+=  - AL[9*j-9]*X1 - AL[9*j-8]*X2 - AL[9*j-7]*X3;
                         SW2+=  - AL[9*j-6]*X1 - AL[9*j-5]*X2 - AL[9*j-4]*X3;
                         SW3+=  - AL[9*j-3]*X1 - AL[9*j-2]*X2 - AL[9*j-1]*X3;
               }
               X1= SW1;
               X2= SW2;
               X3= SW3;
               X2= X2 - ALU[9*i-6]*X1;
               X3= X3 - ALU[9*i-3]*X1 - ALU[9*i-2]*X2;
               X3= ALU[9*i-1]*  X3;
               X2= ALU[9*i-5]*( X2 - ALU[9*i-4]*X3 );
               X1= ALU[9*i-9]*( X1 - ALU[9*i-7]*X3 - ALU[9*i-8]*X2);

               WW[ZP][3*i-3]= X1;
               WW[ZP][3*i-2]= X2;
               WW[ZP][3*i-1]= X3;
          }
```

indexL ⇒ INL
itemL ⇒ IAL

$$\left[\overline{L}+\overline{D}\right]\{y\}=\{r\}\Rightarrow\{y\}=\left[\overline{D}^{-1}\right]\left(\{r\}-\left[\overline{L}\right]\{y\}\right)\Rightarrow y_i=\overline{D}_{ii}^{-1}\left(r_i-\sum_{j=1}^{i-1}\overline{L}_{ij}y_j\right)$$

前処理： LU-GS （3/3）

$$\left[I+\overline{D}^{-1}\overline{U}\right]\{z\}=\{y\}\Rightarrow\{z\}=\{y\}-\left[\overline{D}^{-1}\right]\left[\overline{U}\right]\{z\}$$

$$\Rightarrow z_i=y_i-\overline{D}_{ii}^{-1}\left[\sum_{j=i+1}^{N}\overline{U}_{ij}z_j\right]$$

indexU ⇒ INU
itemU ⇒ IAU

```
/**
      BACKWARD
**/
        for(i=N;i>=1;i--){
             isU= INU[i-1]+1;
             ieU= INU[i];
             SW1= 0.e0;
             SW2= 0.e0;
             SW3= 0.e0;

             for(j=isU;j<=ieU;j++){
                  k=IAU[j-1];
                       X1= WW[ZP][3*k-3];
                       X2= WW[ZP][3*k-2];
                       X3= WW[ZP][3*k-1];

                       SW1+=  + AU[9*j-9]*X1 + AU[9*j-8]*X2 + AU[9*j-7]*X3;
                       SW2+=  + AU[9*j-6]*X1 + AU[9*j-5]*X2 + AU[9*j-4]*X3;
                       SW3+=  + AU[9*j-3]*X1 + AU[9*j-2]*X2 + AU[9*j-1]*X3;
             }

             X1= SW1;
             X2= SW2;
             X3= SW3;
             X2= X2 - ALU[9*i-6]*X1;
             X3= X3 - ALU[9*i-3]*X1 - ALU[9*i-2]*X2;
             X3= ALU[9*i-1]*  X3;
             X2= ALU[9*i-5]*( X2 - ALU[9*i-4]*X3 );
             X1= ALU[9*i-9]*( X1 - ALU[9*i-7]*X3 - ALU[9*i-8]*X2);
             WW[ZP][3*i-3]+= -X1;
             WW[ZP][3*i-2]+= -X2;
             WW[ZP][3*i-1]+= -X3;
        }

        for( i=1;i<=N;i++){
             WW[Z][3*i-3]= WW[ZP][3*i-3];
             WW[Z][3*i-2]= WW[ZP][3*i-2];
             WW[Z][3*i-1]= WW[ZP][3*i-1];
        }
    }
```

前処理：Block Scaling

```
      if (PRECOND != 0 ) {
/**
      Block SCALING
**/
        for(i=0;i<N;i++){
             WW[Z][3*i  ]= WW[R][3*i  ];
             WW[Z][3*i+1]= WW[R][3*i+1];
             WW[Z][3*i+2]= WW[R][3*i+2];
        }

        for(i=0;i<N;i++){
             X1=WW[Z][3*i  ];
             X2=WW[Z][3*i+1];
             X3=WW[Z][3*i+2];
             X2= X2 - ALU[9*i+3]*X1;
             X3= X3 - ALU[9*i+6]*X1 - ALU[9*i+7]*X2;
             X3= ALU[9*i+8]*  X3;
             X2= ALU[9*i+4]*( X2 - ALU[9*i+5]*X3 );
             X1= ALU[9*i  ]*( X1 - ALU[9*i+2]*X3 - ALU[9*i+1]*X2);
             WW[Z][3*i  ]= X1;
             WW[Z][3*i+1]= X2;
             WW[Z][3*i+2]= X3;
        }
      }
```

対角スケーリングの代わりに対角ブロックのLU分解による前進後退代入

完全LU分解

$$\begin{pmatrix} D(9*i) & D(9*i+1) & D(9*i+2) \\ D(9*i+3) & D(9*i+4) & D(9*i+5) \\ D(9*i+6) & D(9*i+7) & D(9*i+8) \end{pmatrix} =$$

$$\begin{pmatrix} d_{11} & d_{12} & d_{13} \\ d_{21} & d_{22} & d_{23} \\ d_{31} & d_{32} & d_{33} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ l_{21} & 1 & 0 \\ l_{31} & l_{32} & 1 \end{pmatrix} \begin{pmatrix} u_{11} & u_{12} & u_{13} \\ 0 & u_{22} & u_{23} \\ 0 & 0 & u_{33} \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ ALUG(9*i+3) & 1 & 0 \\ ALUG(9*i+6) & ALUG(9*i+7) & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ l_{21} & 1 & 0 \\ l_{31} & l_{32} & 1 \end{pmatrix}$$

$$\begin{pmatrix} ALUG(9*i) & ALUG(9*i+1) & ALUG(9*i+2) \\ 0 & ALUG(9*i+4) & ALUG(9*i+5) \\ 0 & 0 & ALUG(9*i+8) \end{pmatrix} = \begin{pmatrix} 1/u_{11} & u_{12} & u_{13} \\ 0 & 1/u_{22} & u_{23} \\ 0 & 0 & 1/u_{33} \end{pmatrix}$$

前処理：Block Scaling

```
    if (PRECOND != 0 ) {
/**
    Block SCALING
**/
      for(i=0;i<N;i++){
           WW[Z][3*i  ]= WW[R][3*i  ];
           WW[Z][3*i+1]= WW[R][3*i+1];
           WW[Z][3*i+2]= WW[R][3*i+2];
      }

      for(i=0;i<N;i++){
           X1=WW[Z][3*i  ];
           X2=WW[Z][3*i+1];
           X3=WW[Z][3*i+2];
           X2= X2 - ALU[9*i+3]*X1;
           X3= X3 - ALU[9*i+6]*X1 - ALU[9*i+7]*X2;
           X3= ALU[9*i+8]*  X3;
           X2= ALU[9*i+4]*( X2 - ALU[9*i+5]*X3 );
           X1= ALU[9*i  ]*( X1 - ALU[9*i+2]*X3 - ALU[9*i+1]*X2);
           WW[Z][3*i  ]= X1;
           WW[Z][3*i+1]= X2;
           WW[Z][3*i+2]= X3;
      }
    }
```

対角スケーリングの代わりに対角ブロックのLU分解による前進後退代入

$$\begin{pmatrix} d_{11} & d_{12} & d_{13} \\ d_{21} & d_{22} & d_{23} \\ d_{31} & d_{32} & d_{33} \end{pmatrix} \begin{pmatrix} z_1 \\ z_2 \\ z_3 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ l_{21} & 1 & 0 \\ l_{31} & l_{32} & 1 \end{pmatrix} \begin{pmatrix} u_{11} & u_{12} & u_{13} \\ 0 & u_{22} & u_{23} \\ 0 & 0 & u_{33} \end{pmatrix} \begin{pmatrix} z_1 \\ z_2 \\ z_3 \end{pmatrix} = \begin{pmatrix} r_1 \\ r_2 \\ r_3 \end{pmatrix}$$

前進代入

```
    if (PRECOND != 0 ) {
/**
    Block SCALING
**/
      for(i=0;i<N;i++){
          WW[Z][3*i  ]= WW[R][3*i  ];
          WW[Z][3*i+1]= WW[R][3*i+1];
          WW[Z][3*i+2]= WW[R][3*i+2];
      }

      for(i=0;i<N;i++){
          X1=WW[Z][3*i  ];
          X2=WW[Z][3*i+1];
          X3=WW[Z][3*i+2];
          X2= X2 - ALU[9*i+3]*X1;
          X3= X3 - ALU[9*i+6]*X1 - ALU[9*i+7]*X2;
          X3= ALU[9*i+8]*  X3;
          X2= ALU[9*i+4]*( X2 - ALU[9*i+5]*X3 );
          X1= ALU[9*i  ]*( X1 - ALU[9*i+2]*X3 - ALU[9*i+1]*X2);
          WW[Z][3*i  ]= X1;
          WW[Z][3*i+1]= X2;
          WW[Z][3*i+2]= X3;
      }
    }
```

$$\begin{pmatrix} 1 & 0 & 0 \\ ALUG(9*i+3) & 1 & 0 \\ ALUG(9*i+6) & ALUG(9*i+7) & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ l_{21} & 1 & 0 \\ l_{31} & l_{32} & 1 \end{pmatrix}$$

$$[L]\{y\} = \{r\}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ l_{21} & 1 & 0 \\ l_{31} & l_{32} & 1 \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix} = \begin{pmatrix} r_1 \\ r_2 \\ r_3 \end{pmatrix} \rightarrow$$

$$y_1 = r_1$$
$$y_2 = r_2 - l_{21} \times y_1$$
$$y_3 = r_3 - l_{31} \times y_1 - -l_{32} \times y_2 \rightarrow$$

$$x_1 = x_1$$
$$x_2 = x_2 - l_{21} \times x_1$$
$$x_3 = x_3 - l_{31} \times x_1 - -l_{32} \times x_2$$

後退代入

```
    if (PRECOND != 0 ) {
/**
    Block SCALING
**/
      for(i=0;i<N;i++){
        WW[Z][3*i  ]= WW[R][3*i  ];
        WW[Z][3*i+1]= WW[R][3*i+1];
        WW[Z][3*i+2]= WW[R][3*i+2];
      }

      for(i=0;i<N;i++){
        X1=WW[Z][3*i  ];
        X2=WW[Z][3*i+1];
        X3=WW[Z][3*i+2];
        X2= X2 - ALU[9*i+3]*X1;
        X3= X3 - ALU[9*i+6]*X1 - ALU[9*i+7]*X2;
        X3= ALU[9*i+8]*  X3;
        X2= ALU[9*i+4]*( X2 - ALU[9*i+5]*X3 );
        X1= ALU[9*i  ]*( X1 - ALU[9*i+2]*X3 - ALU[9*i+1]*X2);
        WW[Z][3*i  ]= X1;
        WW[Z][3*i+1]= X2;
        WW[Z][3*i+2]= X3;
      }
    }
```

$$\begin{pmatrix} ALUG(9*i) & ALUG(9*i+1) & ALUG(9*i+2) \\ 0 & ALUG(9*i+4) & ALUG(9*i+5) \\ 0 & 0 & ALUG(9*i+8) \end{pmatrix} = \begin{pmatrix} 1/u_{11} & u_{12} & u_{13} \\ 0 & 1/u_{22} & u_{23} \\ 0 & 0 & 1/u_{33} \end{pmatrix}$$

$$[U]\{z\} = \{y\}$$

$$\begin{pmatrix} u_{11} & u_{12} & u_{12} \\ 0 & u_{22} & u_{23} \\ 0 & 0 & u_{33} \end{pmatrix} \begin{pmatrix} z_1 \\ z_2 \\ z_3 \end{pmatrix} = \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix}$$

$$z_3 = [1/u_{33}] \times [y_3]$$
$$z_2 = [1/u_{22}] \times [y_2 - u_{23} \times z_3]$$
$$z_1 = [1/u_{11}] \times [y_1 - u_{13} \times z_3 - u_{12} \times z_2]$$

$$x_3 = [1/u_{33}] \times [x_3]$$
$$x_2 = [1/u_{22}] \times [x_2 - u_{23} \times x_3]$$
$$x_1 = [1/u_{11}] \times [x_1 - u_{13} \times x_3 - u_{12} \times x_2]$$

行列ベクトル積

```
/***
        +-------------+
        | {q}= [A]{p} |
        +-------------+
***/
     for( j=0;j<N;j++){
          X1=WW[P][3*j  ];
          X2=WW[P][3*j+1];
          X3=WW[P][3*j+2];

          WVAL1= D[9*j  ]*X1 + D[9*j+1]*X2 + D[9*j+2]*X3;
          WVAL2= D[9*j+3]*X1 + D[9*j+4]*X2 + D[9*j+5]*X3;
          WVAL3= D[9*j+6]*X1 + D[9*j+7]*X2 + D[9*j+8]*X3;

          for(k=INL[j]+1;k<=INL[j+1];k++){
               i=IAL[k-1];
                    X1=WW[P][3*i-3];
                    X2=WW[P][3*i-2];
                    X3=WW[P][3*i-1];
                    WVAL1+=   AL[9*k-9]*X1 + AL[9*k-8]*X2 + AL[9*k-7]*X3;
                    WVAL2+=   AL[9*k-6]*X1 + AL[9*k-5]*X2 + AL[9*k-4]*X3;
                    WVAL3+=   AL[9*k-3]*X1 + AL[9*k-2]*X2 + AL[9*k-1]*X3;
          }
          for(k=INU[j]+1;k<=INU[j+1];k++){
               i=IAU[k-1];
                    X1=WW[P][3*i-3];
                    X2=WW[P][3*i-2];
                    X3=WW[P][3*i-1];
                    WVAL1+=   AU[9*k-9]*X1 + AU[9*k-8]*X2 + AU[9*k-7]*X3;
                    WVAL2+=   AU[9*k-6]*X1 + AU[9*k-5]*X2 + AU[9*k-4]*X3;
                    WVAL3+=   AU[9*k-3]*X1 + AU[9*k-2]*X2 + AU[9*k-1]*X3;
          }
          WW[Q][3*j  ]=WVAL1;
          WW[Q][3*j+1]=WVAL2;
          WW[Q][3*j+2]=WVAL3;
     }
```

DAXPY，内積

```
/***
        +---------------------+
        | {x}= {x} + ALPHA*{p} |
        | {r}= {r} - ALPHA*{q} |
        +---------------------+
***/
     for(i=0;i<N;i++){
          X[3*i  ]+=ALPHA *WW[P][3*i  ];
          X[3*i+1]+=ALPHA *WW[P][3*i+1];
          X[3*i+2]+=ALPHA *WW[P][3*i+2];
          WW[R][3*i  ][R]+= -ALPHA *WW[Q][3*i  ];
          WW[R][3*i+1][R]+= -ALPHA *WW[Q][3*i+1];
          WW[R][3*i+2][R]+= -ALPHA *WW[Q][3*i+2];
     }

     DNRM20= 0.e0;
     for(i=0;i<N;i++){
          DNRM20+= WW[R][3*I  ]*WW[R][3*I  ] +
                   WW[R][3*i+1]*WW[R][3*i+1] +
                   WW[R][3*i+2]*WW[R][3*i+2];
     }

     DNRM2= DNRM20;
     RESID= sqrt(DNRM2/BNRM2);
```

DAXPY，内積

```
/***
        +----------------------+
        | {x}= {x} + ALPHA*{p} |
        | {r}= {r} - ALPHA*{q} |
        +----------------------+
***/
     for(i=0;i<N;i++){
          X[3*i  ]+=ALPHA *WW[3*i  ][P];
          X[3*i+1]+=ALPHA *WW[3*i+1][P];
          X[3*i+2]+=ALPHA *WW[3*i+2][P];
          WW[3*i  ][R]+= -ALPHA *WW[3*i  ][Q];
          WW[3*i+1][R]+= -ALPHA *WW[3*i+1][Q];
          WW[3*i+2][R]+= -ALPHA *WW[3*i+2][Q];
     }

     DNRM20= 0.e0;
     for(i=0;i<N;i++){
          DNRM20+= WW[R][3*i  ]*WW[R][3*i  ] +
                   WW[R][3*i+1]*WW[R][3*i+1] +
                   WW[R][3*i+2]*WW[R][3*i+2];
     }

     DNRM2= DNRM20;
     RESID= sqrt(DNRM2/BNRM2);
```

- 前処理について
- ソルバー部分
- **応力計算**
- レポート課題2

応 力

- ここまでで求まるのは，各節点における「変位」
- 工学的に興味のあるのは「応力」
  - 「変位」の微分である「ひずみ」から計算

ひずみ⇒応力関係

$$\begin{Bmatrix}\sigma_x\\ \sigma_y\\ \sigma_z\\ \tau_{xy}\\ \tau_{yz}\\ \tau_{zx}\end{Bmatrix} = \frac{E}{(1+\nu)(1-2\nu)}\begin{bmatrix} 1-\nu & \nu & \nu & 0 & 0 & 0\\ \nu & 1-\nu & \nu & 0 & 0 & 0\\ \nu & \nu & 1-\nu & 0 & 0 & 0\\ 0 & 0 & 0 & \frac{1}{2}(1-2\nu) & 0 & 0\\ 0 & 0 & 0 & 0 & \frac{1}{2}(1-2\nu) & 0\\ 0 & 0 & 0 & 0 & 0 & \frac{1}{2}(1-2\nu)\end{bmatrix}\begin{Bmatrix}\varepsilon_x\\ \varepsilon_y\\ \varepsilon_z\\ \gamma_{xy}\\ \gamma_{yz}\\ \gamma_{zx}\end{Bmatrix}$$

$$[D]$$

$$\{\sigma\} = [D]\{\varepsilon\}$$

ひずみ⇒応力関係

$$\begin{Bmatrix} \sigma_x \\ \sigma_y \\ \sigma_z \\ \tau_{xy} \\ \tau_{yz} \\ \tau_{zx} \end{Bmatrix} = \begin{bmatrix} \mathrm{valX} & \mathrm{valA} & \mathrm{valA} & 0 & 0 & 0 \\ \mathrm{valA} & \mathrm{valX} & \mathrm{valA} & 0 & 0 & 0 \\ \mathrm{valA} & \mathrm{valA} & \mathrm{valX} & 0 & 0 & 0 \\ 0 & 0 & 0 & \mathrm{valB} & 0 & 0 \\ 0 & 0 & 0 & 0 & \mathrm{valB} & 0 \\ 0 & 0 & 0 & 0 & 0 & \mathrm{valB} \end{bmatrix} \begin{Bmatrix} \varepsilon_x \\ \varepsilon_y \\ \varepsilon_z \\ \gamma_{xy} \\ \gamma_{yz} \\ \gamma_{zx} \end{Bmatrix}$$

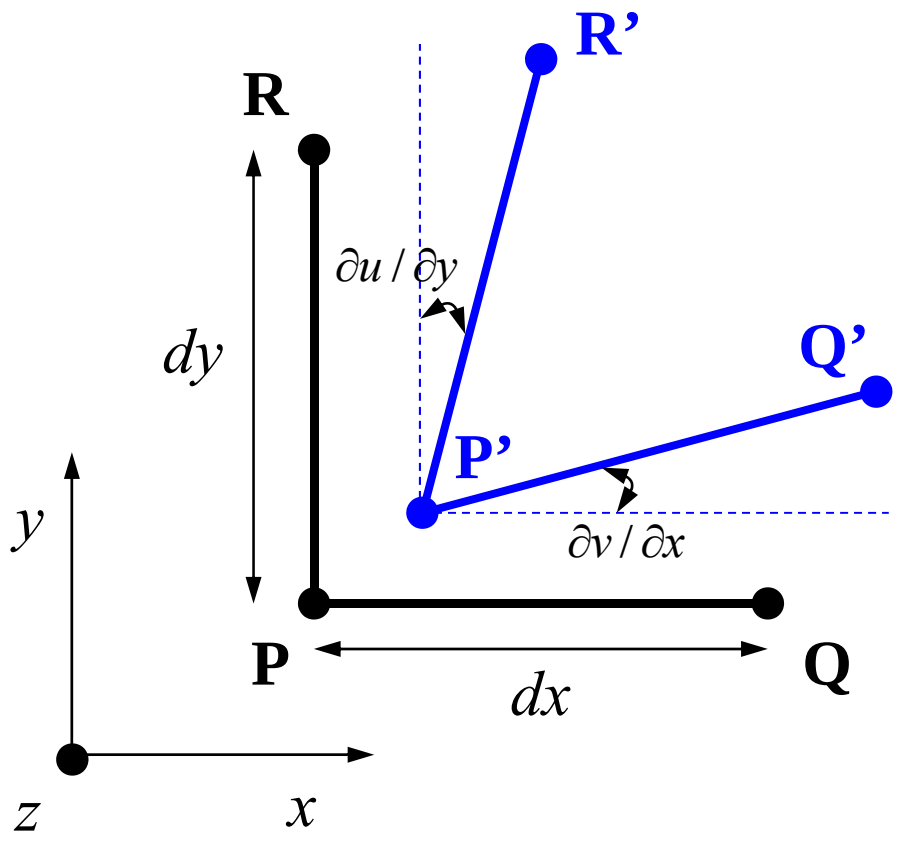
$$[D]$$

$$\{\sigma\} = [D]\{\varepsilon\}$$

垂直ひずみ～変位の関係

- PQ⇒P'Q'

$$\varepsilon_x = \frac{\left\{\left(x + dx + u + \frac{\partial u}{\partial x} dx\right) - (x + u)\right\} - dx}{dx} = \frac{\partial u}{\partial x}$$

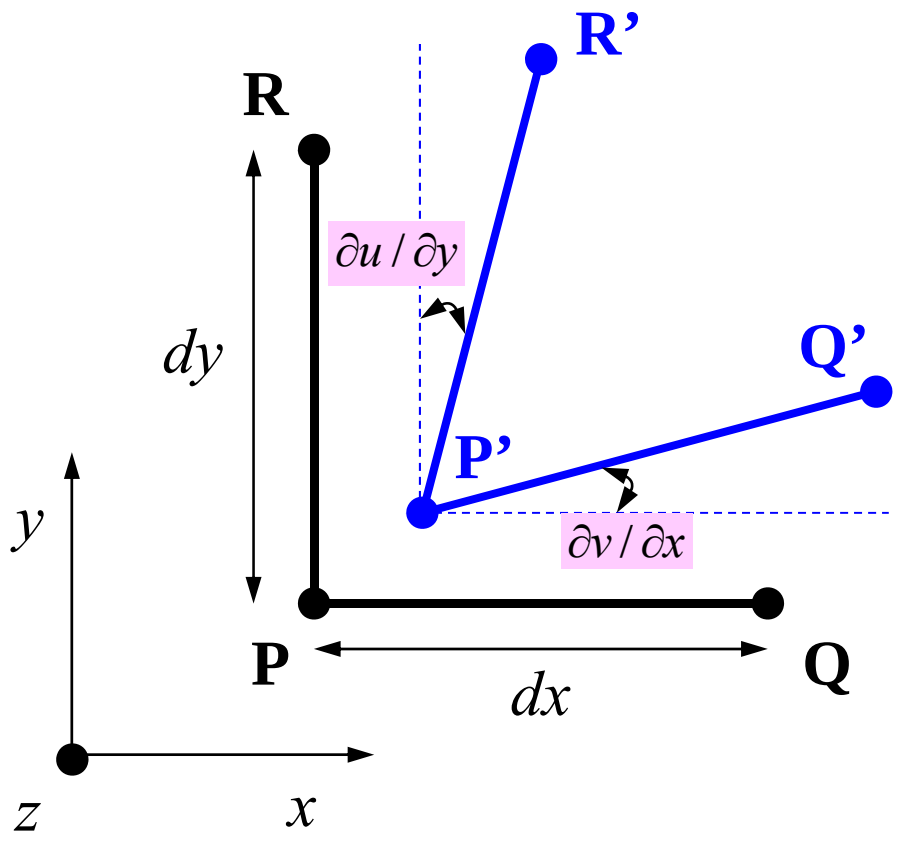


$$\varepsilon_x = \frac{\partial u}{\partial x}$$

$$\varepsilon_y = \frac{\partial v}{\partial y}$$

$$\varepsilon_z = \frac{\partial w}{\partial z}$$

せん断ひずみ～変位の関係



$$\gamma_{xy} = \frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}$$

$$\gamma_{yz} = \frac{\partial v}{\partial z} + \frac{\partial w}{\partial y}$$

$$\gamma_{zx} = \frac{\partial w}{\partial x} + \frac{\partial u}{\partial z}$$

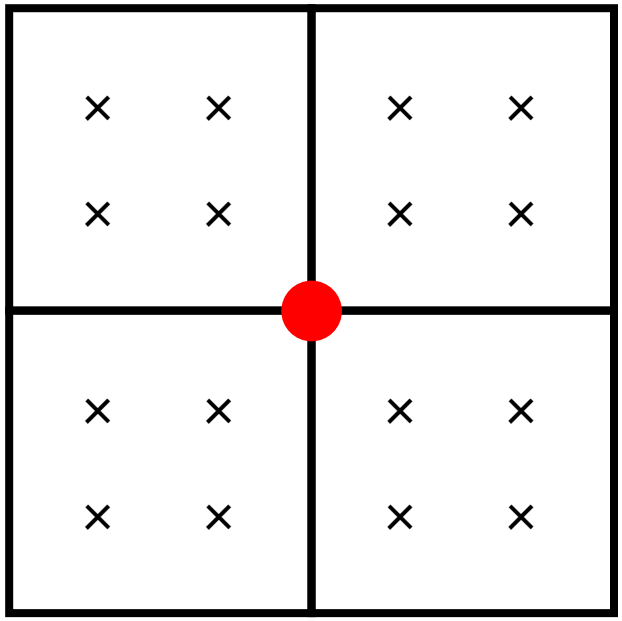
応力の計算

$$\sigma_x = \frac{E}{(1+\nu)(1-2\nu)}\left[(1-\nu)\varepsilon_x + \nu\varepsilon_y + \nu\varepsilon_z\right]$$

$$= \frac{E}{(1+\nu)(1-2\nu)}\left[(1-\nu)\frac{\partial u}{\partial x} + \nu\frac{\partial v}{\partial y} + \nu\frac{\partial w}{\partial z}\right]$$

応 力

- ここまでで求まるのは，各節点における「変位」
- 工学的に興味のあるのは「応力」
  - 「変位」の微分である「ひずみ」から計算



- 正確な応力が求められるのはガウス積分
- 節点における応力
  - 平均値

応力の計算

$$\sigma_x = \frac{E}{(1+\nu)(1-2\nu)}\left[(1-\nu)\varepsilon_x + \nu\varepsilon_y + \nu\varepsilon_z\right]$$

$$= \frac{E}{(1+\nu)(1-2\nu)}\left[(1-\nu)\frac{\partial u}{\partial x} + \nu\frac{\partial v}{\partial y} + \nu\frac{\partial w}{\partial z}\right]$$

- ガラーキン法：要素の平均応力×体積

$$\int_V [N]^T \sigma_x dV = \int_V [N]^T \left\{\frac{E}{(1+\nu)(1-2\nu)}\left[(1-\nu)u_{,x} + \nu w_{,y} + \nu w_{,z}\right]\right\} dV$$

$$= \int_V [N]^T \left\{\frac{E}{(1+\nu)(1-2\nu)}\left((1-\nu)[N_{,x}]\{U\} + \nu[N_{,y}]\{V\} + \nu[N_{,z}]\{W\}\right)\right\} dV$$

1D：要素単位での積分：$\{f\}$

$$N_i = \left(\frac{X_j - x}{L} \right), \quad N_j = \left(\frac{x - X_i}{L} \right) \qquad \frac{dN_i}{dx} = \left(\frac{-1}{L} \right), \quad \frac{dN_j}{dx} = \left(\frac{1}{L} \right)$$

$$\int_V X[N]^T dV = XA \int_0^L \begin{bmatrix} 1 - x/L \\ x/L \end{bmatrix} dx = \frac{XAL}{2} \begin{Bmatrix} 1 \\ 1 \end{Bmatrix}$$

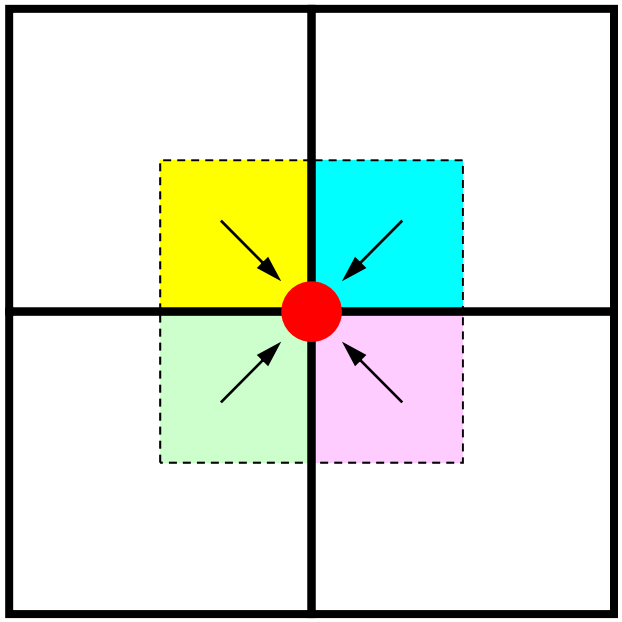
物体力

A：断面積，L：要素長さ

2D：節点応力の求め方

$$\int_V [N]^T \sigma_x dV$$

各節点における，各要素からの
応力×寄与体積



$$\int_V [N]^T dV$$

各節点における寄与体積

$$\therefore \quad \sigma_x = \frac{\sum \int_V [N]^T \sigma_x dV}{\sum \int_V [N]^T dV}$$

各節点における平均応力

RECOVER_STRESS：応力（1/4）

```
#include <math.h>
#include "pfem_util.h"
#include "allocate.h"
extern   void JACOBI();
void RECOVER_STRESS()
{
     int  i,k,kk,icel;
     int  ie,je,ip,jp;
     int  ipn,jpn,kpn;
     int  iiS,iiE;
     double   RB;
     double   UUi,VVi,WWi,UUj,VVj,WWj;
     double   valX,valA,valB,E0,POI0,VOL,coef;
     int  in1,in2,in3,in4,in5,in6,in7,in8;
     double   X1,X2,X3,X4,X5,X6,X7,X8;
     double   Y1,Y2,Y3,Y4,Y5,Y6,Y7,Y8;
     double   Z1,Z2,Z3,Z4,Z5,Z6,Z7,Z8;
     double   SHi,SHj;
     double   EPS_xx,EPS_yy,EPS_zz,GAM_xy,GAM_xz,GAM_yz;

     KINT nodLOCAL[8];

     SIGMA_N=(KREAL*)allocate_vector(sizeof(KREAL),3*N);
     TAU_N  =(KREAL*)allocate_vector(sizeof(KREAL),3*N);

     for(i=0;i<3*N;i++) SIGMA_N[i]=0.0;
     for(i=0;i<3*N;i++) TAU_N[i]=0.0;
     for(i=0;i<3*N;i++) B[i]=0.0;

     for( icel=0;icel<  ICELTOT;icel++){
          E0   =    ELAST;
          POI0=     POISSON;

          valA=     POI0 /    (1.e0-POI0);
          valB=     (1.e0-2.e0*POI0)/(2.e0*(1.e0-POI0));
          valX=     E0*  (1.e0-POI0)/((1.e0+POI0)*(1.e0-2.e0*POI0));

          valA=     valA *    valX;
          valB=     valB *    valX;
```

RECOVER_STRESS：応力（2/4）

```
          in1= ICELNOD[icel][0];
          in2= ICELNOD[icel][1];
          in3= ICELNOD[icel][2];
          in4= ICELNOD[icel][3];
          in5= ICELNOD[icel][4];
          in6= ICELNOD[icel][5];
          in7= ICELNOD[icel][6];
          in8= ICELNOD[icel][7];
         nodLOCAL[0]=  in1;
         nodLOCAL[1]=  in2;
         nodLOCAL[2]=  in3;
         nodLOCAL[3]=  in4;
         nodLOCAL[4]=  in5;
         nodLOCAL[5]=  in6;
         nodLOCAL[6]=  in7;
         nodLOCAL[7]=  in8;
         X1=  XYZ[in1-1][0];
         X2=  XYZ[in2-1][0];
 （中略）
         X7=  XYZ[in7-1][0];
         X8=  XYZ[in8-1][0];
         Y1=  XYZ[in1-1][1];
         Y2=  XYZ[in2-1][1];
 （中略）
         Y7=  XYZ[in7-1][1];
         Y8=  XYZ[in8-1][1];
         Z1=  XYZ[in1-1][2];
         Z2=  XYZ[in2-1][2];
 （中略）
         Z7=  XYZ[in7-1][2];
         Z8=  XYZ[in8-1][2];

/**
    JACOBIAN  &    inv-JACOBIAN
**/
         JACOBI    (DETJ, PNQ, PNE, PNT, PNX, PNY, PNZ,
                    X1, X2, X3, X4, X5, X6, X7, X8,
                    Y1, Y2, Y3, Y4, Y5, Y6, Y7, Y8,
                    Z1, Z2, Z3, Z4, Z5, Z6, Z7, Z8          );
```

RECOVER_STRESS：応力（3/4）

```
/**
      MATRIX
**/
          for(ie=0;ie<8;ie++){
               ip= nodLOCAL[ie];
               for(je=0;je<8;je++){
                    jp= nodLOCAL[je];
                    UUj= X[3*jp-3];
                    VVj= X[3*jp-2];
                    WWj= X[3*jp-1];
                    UUi= X[3*ip-3];
                    VVi= X[3*ip-2];
                    WWi= X[3*ip-1];

                    EPS_xx=   0.e0;
                    EPS_yy=   0.e0;
                    EPS_zz=   0.e0;
                    GAM_xy=   0.e0;
                    GAM_xz=   0.e0;
                    GAM_yz=   0.e0;
                    GAM_xy=   0.e0;
                    GAM_xz=   0.e0;
                    GAM_yz=   0.e0;

                    VOL  =    0.e0;
                    for( ipn=0;ipn<2;ipn++){
                         for( jpn=0;jpn<2;jpn++){
                              for( kpn=0;kpn<2;kpn++){
                                   coef= fabs(DETJ[ipn][jpn][kpn])*WEI[ipn]*WEI[jpn]*WEI[kpn];
                                   SHi= SHAPE[ipn][jpn][kpn][ie]           *           coef;

                                   EPS_xx+=  SHi*PNX[ipn][jpn][kpn][je];
                                   EPS_yy+=  SHi*PNY[ipn][jpn][kpn][je];
                                   EPS_zz+=  SHi*PNZ[ipn][jpn][kpn][je];
                                   GAM_xy+=  SHi*PNX[ipn][jpn][kpn][je] * VVj
                                        +    SHi*PNY[ipn][jpn][kpn][je] * UUj;
                                   GAM_xz+=  SHi*PNX[ipn][jpn][kpn][je] * WWj
                                        +    SHi*PNZ[ipn][jpn][kpn][je] * UUj;
                                   GAM_yz+=  SHi*PNY[ipn][jpn][kpn][je] * WWj
                                        +    SHi*PNZ[ipn][jpn][kpn][je] * VVj;
                                   VOL  =    VOL + SHi; }}}
```

各節点における変位

RECOVER_STRESS：応力（3/4）

```
/**
      MATRIX
**/
          for(ie=0;ie<8;ie++){
               ip= nodLOCAL[ie];
               for(je=0;je<8;je++){
                    jp= nodLOCAL[je];
                    UUj= X[3*jp-3];
                    VVj= X[3*jp-2];
                    WWj= X[3*jp-1];
                    UUi= X[3*ip-3];
                    VVi= X[3*ip-2];
                    WWi= X[3*ip-1];

                    EPS_xx=   0.e0;
                    EPS_yy=   0.e0;
                    EPS_zz=   0.e0;
                    GAM_xy=   0.e0;
                    GAM_xz=   0.e0;
                    GAM_yz=   0.e0;
                    GAM_xy=   0.e0;
                    GAM_xz=   0.e0;
                    GAM_yz=   0.e0;

                    VOL  =    0.e0;
                    for( ipn=0;ipn<2;ipn++){
                         for( jpn=0;jpn<2;jpn++){
                              for( kpn=0;kpn<2;kpn++){
                                   coef= fabs(DETJ[ipn][jpn][kpn])*WEI[ipn]*WEI[jpn]*WEI[kpn];
                                   SHi= SHAPE[ipn][jpn][kpn][ie]          *          coef;

                                   EPS_xx+=  SHi*PNX[ipn][jpn][kpn][je];
                                   EPS_yy+=  SHi*PNY[ipn][jpn][kpn][je];
                                   EPS_zz+=  SHi*PNZ[ipn][jpn][kpn][je];
                                   GAM_xy+=  SHi*PNX[ipn][jpn][kpn][je] * VVj
                                        +    SHi*PNY[ipn][jpn][kpn][je] * UUj;
                                   GAM_xz+=  SHi*PNX[ipn][jpn][kpn][je] * WWj
                                        +    SHi*PNZ[ipn][jpn][kpn][je] * UUj;
                                   GAM_yz+=  SHi*PNY[ipn][jpn][kpn][je] * WWj
                                        +    SHi*PNZ[ipn][jpn][kpn][je] * VVj;
                                   VOL  =    VOL + SHi; }}}
```

$$u_{,x} = \left[N_{,x}\right]\{U\}, \quad u_{,y} = \left[N_{,y}\right]\{U\}, \quad u_{,z} = \left[N_{,z}\right]\{U\}$$
$$v_{,x} = \left[N_{,x}\right]\{V\}, \quad v_{,y} = \left[N_{,y}\right]\{V\}$$
$$w_{,x} = \left[N_{,x}\right]\{W\}, \quad w_{,z} = \left[N_{,z}\right]\{W\}$$

RECOVER_STRESS：応力（4/4）

```
                    EPS_xx= EPS_xx * UUj;
                    EPS_yy= EPS_yy * VVj;
                    EPS_zz= EPS_zz * WWj;

                    SIGMA_N[3*ip-3]+= valX*EPS_xx+ valA*EPS_yy + valA*EPS_zz;
                    SIGMA_N[3*ip-2]+= valA*EPS_xx+ valX*EPS_yy + valA*EPS_zz;
                    SIGMA_N[3*ip-1]+= valA*EPS_xx+ valA*EPS_yy + valX*EPS_zz;
                    TAU_N[3*ip-3]+= GAM_xy*valB;
                    TAU_N[3*ip-2]+= GAM_xz*valB;
                    TAU_N[3*ip-1]+= GAM_yz*valB;
                    if (ip==jp) B[ip-1]+=VOL;
                }
            }
        }

/****
     NODAL     VALUE
***/

        for(i=0;i<N;i++){
            RB=1.0e0/B[i];

            SIGMA_N[3*i  ]*=RB;
            SIGMA_N[3*i+1]*=RB;
            SIGMA_N[3*i+2]*=RB;

            TAU_N[3*i ]*=RB;
            TAU_N[3*i+1]*=RB;
            TAU_N[3*i+2]*=RB;

            }
        }
```

$$u_{,x} = [N_{,x}]\{U\}, \quad u_{,y} = [N_{,y}]\{U\}, \quad u_{,z} = [N_{,z}]\{U\}$$
$$v_{,x} = [N_{,x}]\{V\}, \quad v_{,y} = [N_{,y}]\{V\}$$
$$w_{,x} = [N_{,x}]\{W\}, \quad w_{,z} = [N_{,z}]\{W\}$$

RECOVER_STRESS：応力（4/4）

```
                    EPS_xx= EPS_xx * UUj;
                    EPS_yy= EPS_yy * VVj;
                    EPS_zz= EPS_zz * WWj;

                    SIGMA_N[3*ip-3]+= valX*EPS_xx+ valA*EPS_yy + valA*EPS_zz;
                    SIGMA_N[3*ip-2]+= valA*EPS_xx+ valX*EPS_yy + valA*EPS_zz;
                    SIGMA_N[3*ip-1]+= valA*EPS_xx+ valA*EPS_yy + valX*EPS_zz;
                    TAU_N[3*ip-3]+= GAM_xy*valB;
                    TAU_N[3*ip-2]+= GAM_xz*valB;
                    TAU_N[3*ip-1]+= GAM_yz*valB;
                    if (ip==jp) B[ip-1]+=VOL;
                }
            }
        }

/****
    NODAL     VALUE
***/

    for(i=0;i<N;i++){
        RB=1.0e0/B[i];

        SIGMA_N[3*i  ]*=RB;
        SIGMA_N[3*i+1]*=RB;
        SIGMA_N[3*i+2]*=RB;

        TAU_N[3*i  ]*=RB;
        TAU_N[3*i+1]*=RB;
        TAU_N[3*i+2]*=RB;

        }
    }
```

$$\begin{Bmatrix}\sigma_x\\ \sigma_y\\ \sigma_z\\ \tau_{xy}\\ \tau_{yz}\\ \tau_{zx}\end{Bmatrix} = \begin{bmatrix}\mathrm{valX} & \mathrm{valA} & \mathrm{valA} & 0 & 0 & 0\\ \mathrm{valA} & \mathrm{valX} & \mathrm{valA} & 0 & 0 & 0\\ \mathrm{valA} & \mathrm{valA} & \mathrm{valX} & 0 & 0 & 0\\ 0 & 0 & 0 & \mathrm{valB} & 0 & 0\\ 0 & 0 & 0 & 0 & \mathrm{valB} & 0\\ 0 & 0 & 0 & 0 & 0 & \mathrm{valB}\end{bmatrix} \begin{Bmatrix}\varepsilon_x\\ \varepsilon_y\\ \varepsilon_z\\ \gamma_{xy}\\ \gamma_{yz}\\ \gamma_{zx}\end{Bmatrix}$$

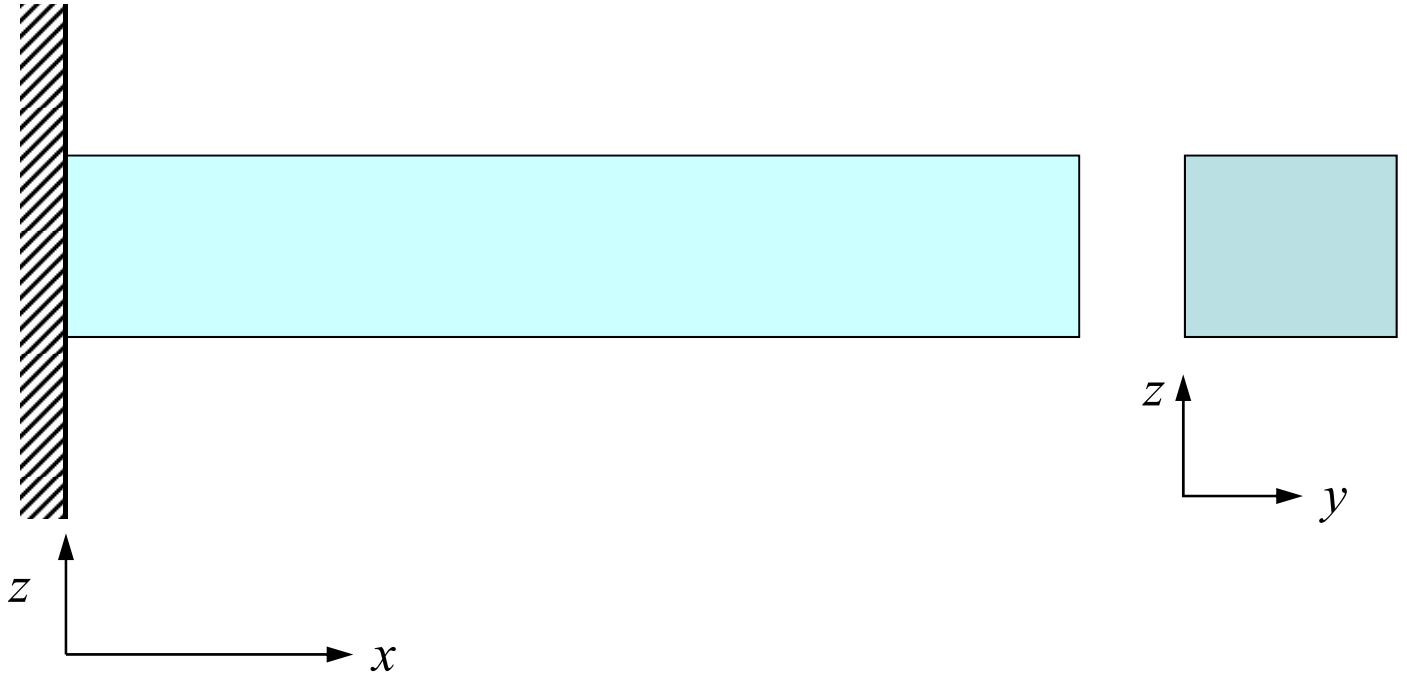
RECOVER_STRESS：応力（4/4）

```
                    EPS_xx= EPS_xx * UUj;
                    EPS_yy= EPS_yy * VVj;
                    EPS_zz= EPS_zz * WWj;

                    SIGMA_N[3*ip-3]+= valX*EPS_xx+ valA*EPS_yy + valA*EPS_zz;
                    SIGMA_N[3*ip-2]+= valA*EPS_xx+ valX*EPS_yy + valA*EPS_zz;
                    SIGMA_N[3*ip-1]+= valA*EPS_xx+ valA*EPS_yy + valX*EPS_zz;
                    TAU_N[3*ip-3]+= GAM_xy*valB;
                    TAU_N[3*ip-2]+= GAM_xz*valB;
                    TAU_N[3*ip-1]+= GAM_yz*valB;
                    if (ip==jp) B[ip-1]+=VOL;
                }
            }
        }

/****
      NODAL     VALUE
***/

      for(i=0;i<N;i++){
            RB=1.0e0/B[i];

            SIGMA_N[3*i  ]*=RB;
            SIGMA_N[3*i+1]*=RB;
            SIGMA_N[3*i+2]*=RB;

            TAU_N[3*i ]*=RB;
            TAU_N[3*i+1]*=RB;
            TAU_N[3*i+2]*=RB;

            }
      }
```

$$\begin{Bmatrix}\sigma_x\\ \sigma_y\\ \sigma_z\\ \tau_{xy}\\ \tau_{yz}\\ \tau_{zx}\end{Bmatrix} = \begin{bmatrix}\mathrm{valX} & \mathrm{valA} & \mathrm{valA} & 0 & 0 & 0\\ \mathrm{valA} & \mathrm{valX} & \mathrm{valA} & 0 & 0 & 0\\ \mathrm{valA} & \mathrm{valA} & \mathrm{valX} & 0 & 0 & 0\\ 0 & 0 & 0 & \mathrm{valB} & 0 & 0\\ 0 & 0 & 0 & 0 & \mathrm{valB} & 0\\ 0 & 0 & 0 & 0 & 0 & \mathrm{valB}\end{bmatrix} \begin{Bmatrix}\varepsilon_x\\ \varepsilon_y\\ \varepsilon_z\\ \gamma_{xy}\\ \gamma_{yz}\\ \gamma_{zx}\end{Bmatrix}$$

- 前処理について
- ソルバー部分
- 応力計算
- **レポート課題2**

レポート課題2（1/2）



- 以下の問題を解く三次元弾性解析コードを作成し，計算を実施せよ
  - 片持ち梁（断面：長方形）
    - ヤング率=1.00，自重（密度）=0.025，ポアソン比=0.30
  - u=v=w=0@x=0
  - mat_con0においてNU=5，NL=5を初期値とせよ

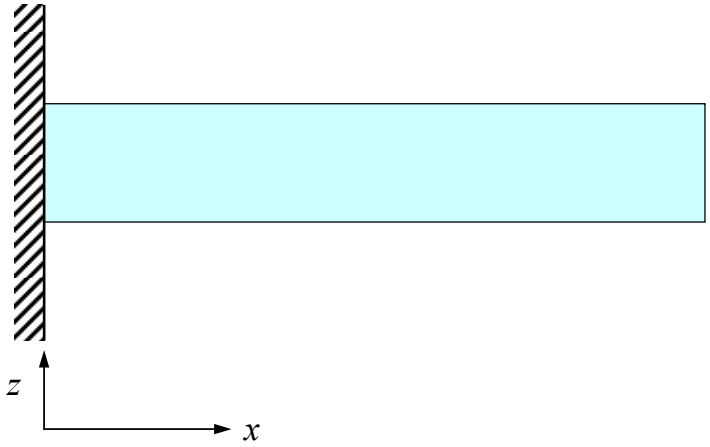
レポート課題2（2/2）

- メッシュジェネレータの作成
- fem3dの改造（境界条件，自重）
- 検証（解析解：次ページ），メッシュ分割の影響
- ガウス積分点の数を変えてやってみよ
  - n=2（fem3d），n=1，n=3
- もし余裕があればILU(0)，IC(0)，GSを試して見よ
- レポート
  - 方針（基本設計）
  - 定式化
  - 実装，結果，考察
  - A4　10枚以内（＋リスト）
  - 提出：2011年9月12日（月）17：00

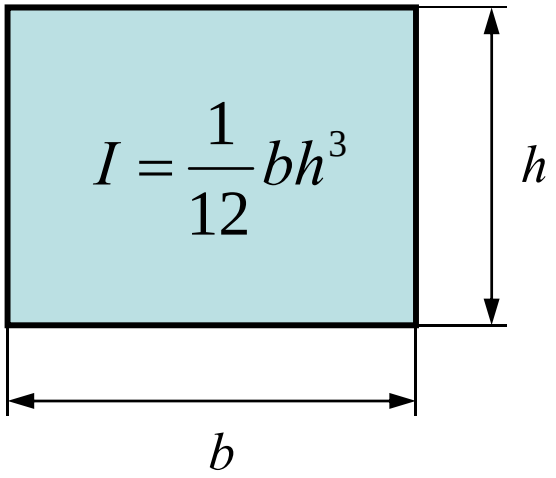
解析解：Lが充分長いときに成立

$$z = -\frac{W}{24EI}(x-L)^2 \cdot \left(x^2 + 2Lx + 3L^2\right)$$

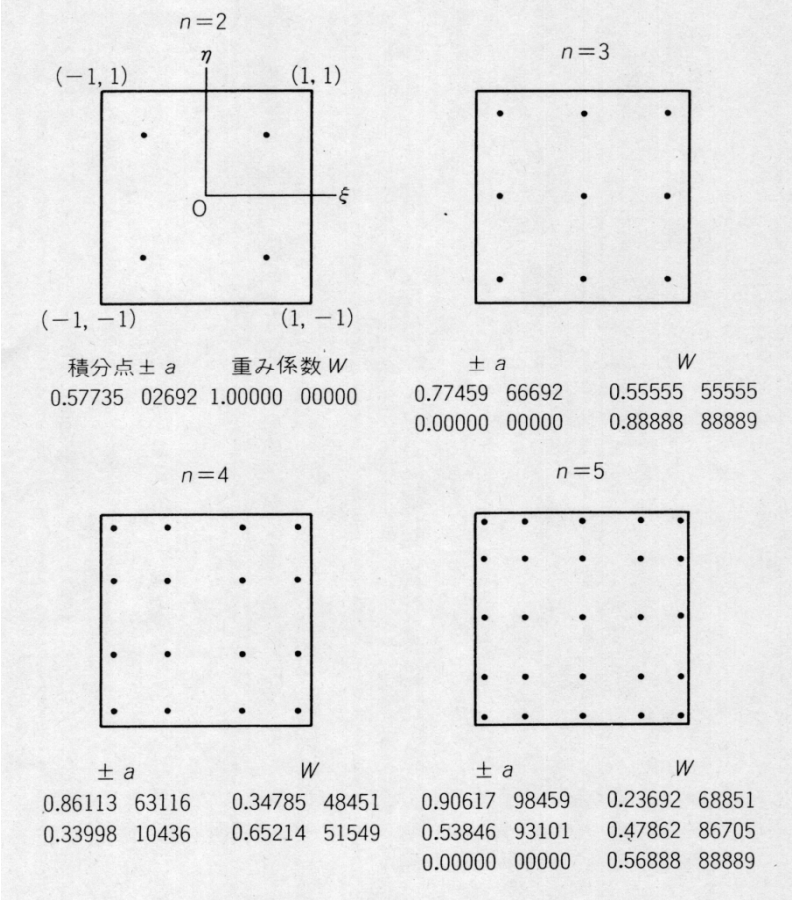
$$z_{\max} = -\frac{WL^4}{8EI} \quad where \quad \frac{L}{h} > 4$$



- z　z方向変位
- L　梁長さ
- W　長さあたり自重
- E　ヤング率
- I　断面二次モーメント
  （長方形の場合）



ガウスの積分公式



n=1

積分点座標：0.000
重み係数　：2.000
（要素中心）

自重：Z方向体積力

$$\frac{\partial \tau_{zx}}{\partial x}+\frac{\partial \tau_{xy}}{\partial y}+\frac{\partial \sigma_{z}}{\partial z}+Z=0$$

$$\tau_{zx,x}+\tau_{xy,y}+\sigma_{z,z}+Z=0$$

↓ ガラーキン法

$$\int [N]^T \{\tau_{zx,x}+\tau_{xy,y}+\sigma_{z,z}+Z\} dV=0$$

$$\int [N]^T \{\tau_{zx,x}\} dV+\int [N]^T \{\tau_{xy,y}\} dV+\int [N]^T \{\sigma_{z,z}\} dV+\int [N]^T \{Z\} dV=0$$