

三次元弾性解析コード (2/3) マトリクス生成

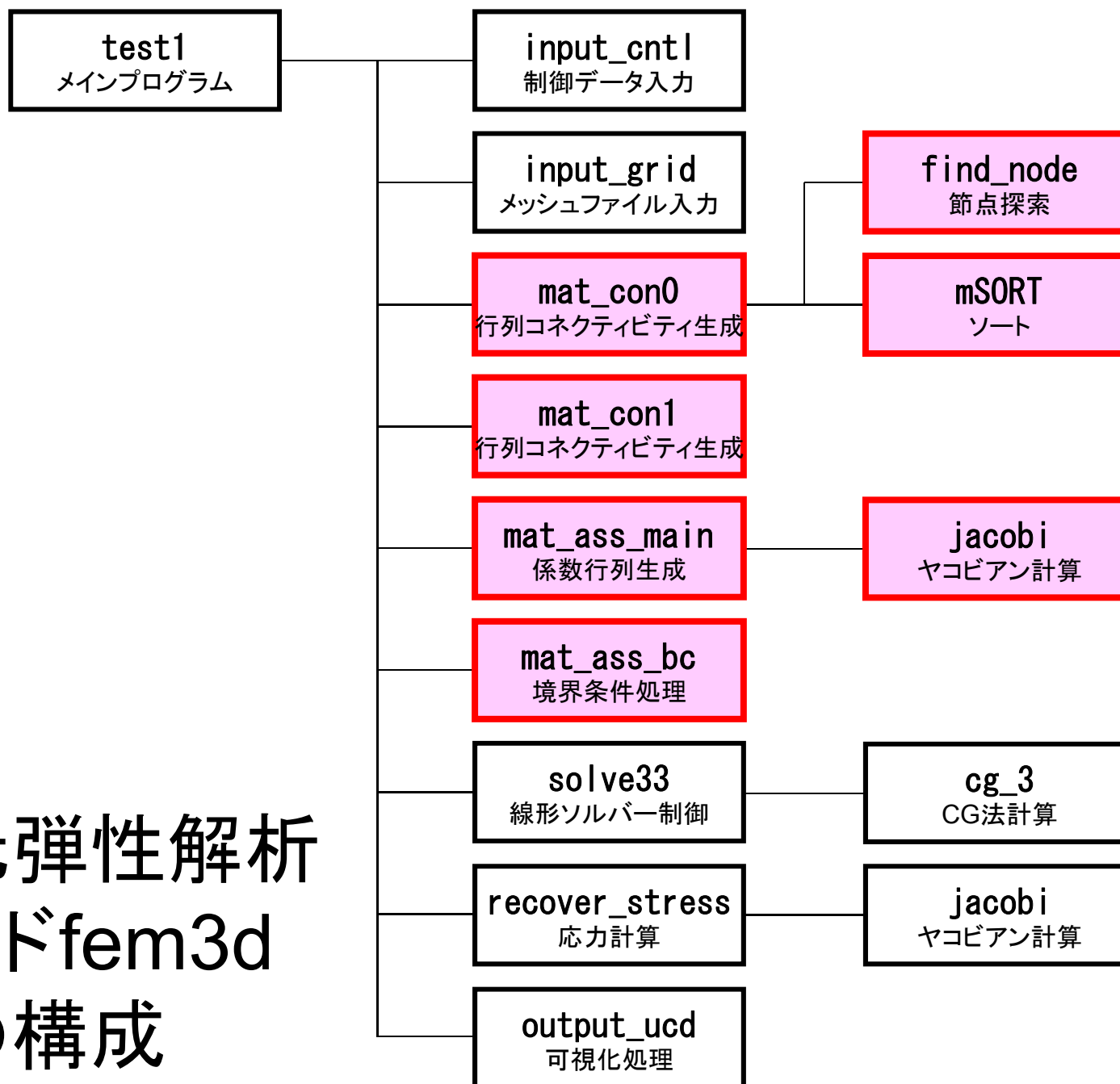
2011年夏学期
中島 研吾

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

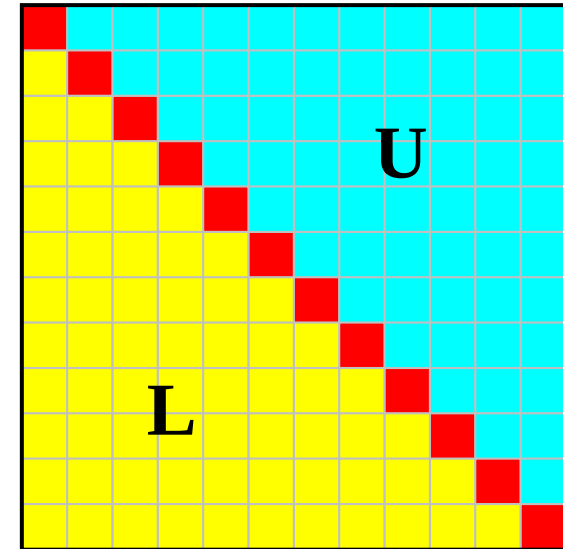
- 初期化
 - 制御変数読み込み
 - 座標読み込み⇒要素生成 (N:節点数, ICELTOT : 要素数)
 - 配列初期化 (全体マトリクス, 要素マトリクス)
 - 要素⇒全体マトリクスマッピング (Index, Item)
- マトリクス生成
 - 要素単位の処理 (do icel= 1, ICELTOT)
 - 要素マトリクス計算
 - 全体マトリクスへの重ね合わせ
 - 境界条件の処理
- 連立一次方程式
 - 共役勾配法 (CG)
- 応力計算

三次元弾性解析 コードfem3d の構成



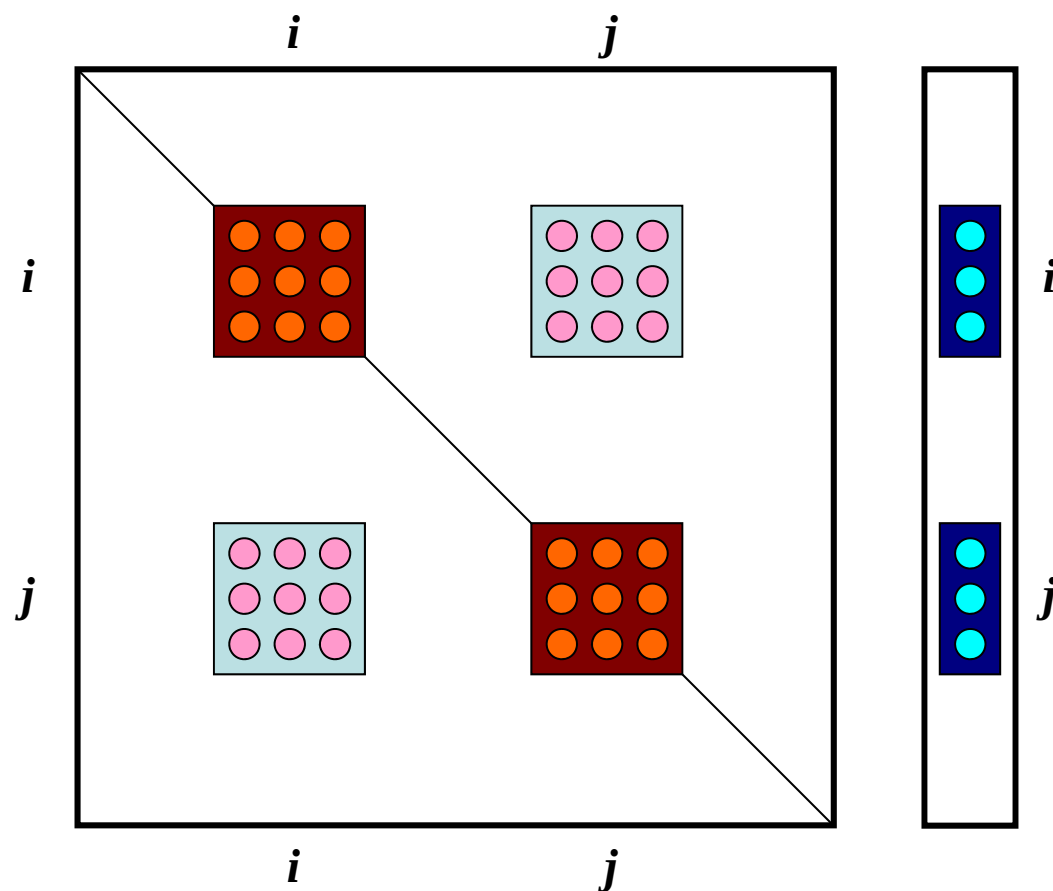
fem3d : いくつかの特徴

- 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



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

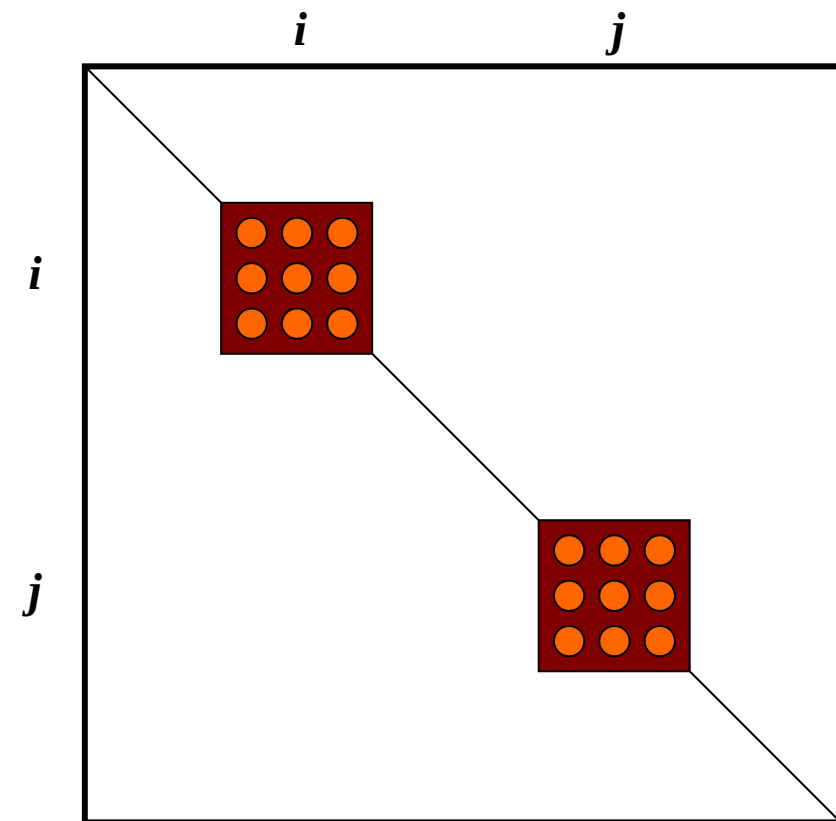
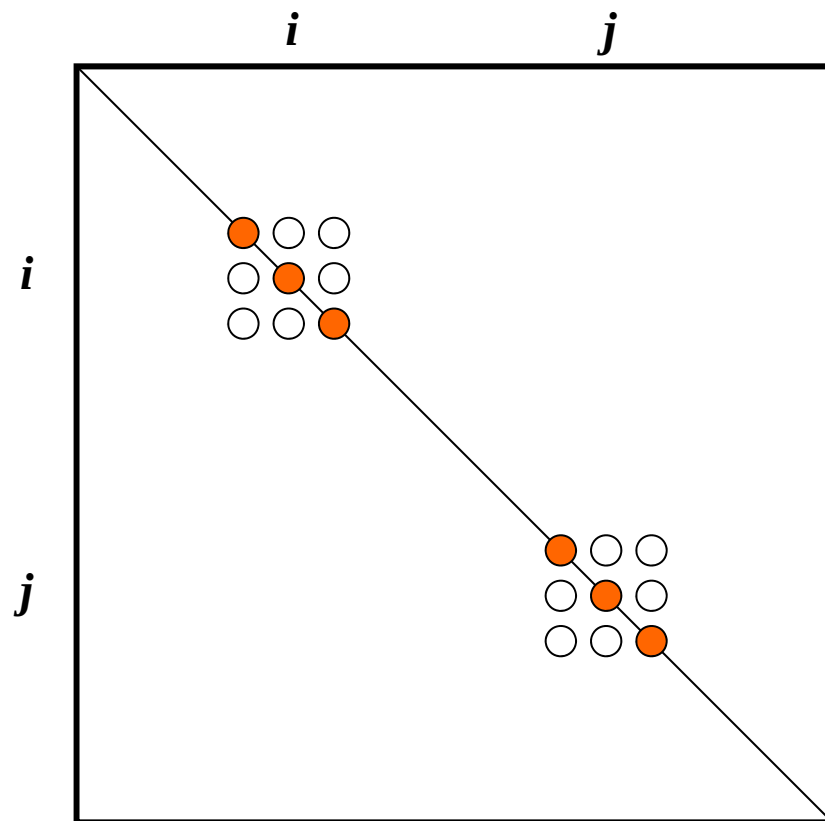
- 間接参照（メモリに負担）と計算の比が大きくなる
- ベクトル，スカラー共に効く：2倍以上の性能
 - 連続領域，キャッシュに載る，ループあたりの計算量増加

```
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)
    X2= X(3*kk-1)
    X3= X(3*kk)
    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

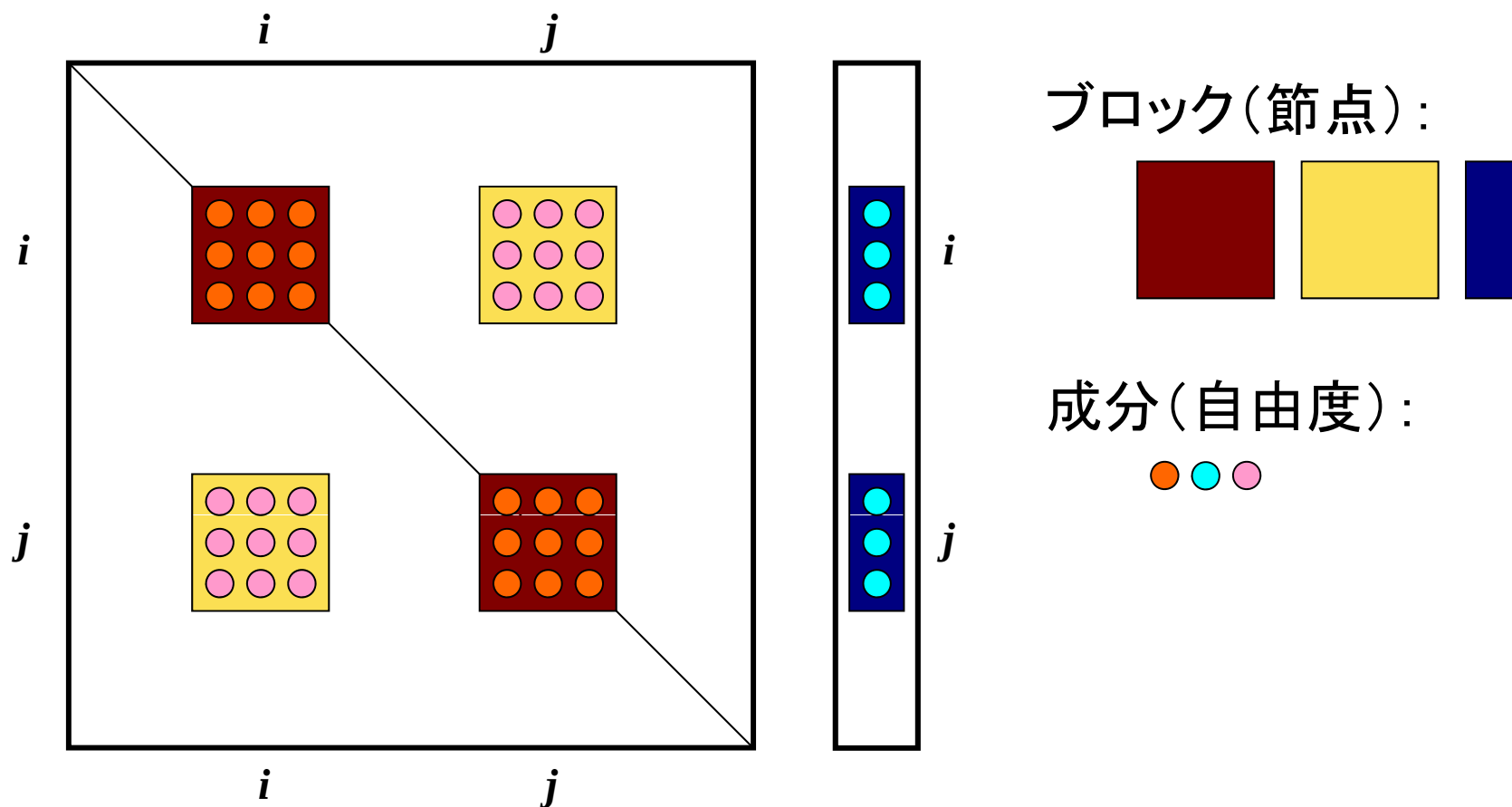
Global変数表 : pfem_util.h/f (2/3)

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)
pfemlarray	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
08th	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 $N_i (i=1\sim 8)$
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

用語の定義



マトリクス生成まで

- 一次元の場合は, index, itemに関連した情報を簡単に作ることができた
 - 非ゼロ非対角成分の数は2
 - 番号が自分に対して : +1と-1
- 三次元の場合はもっと複雑
 - 非ゼロ非対角ブロックの数は7~26 (現在の形状)
 - 実際はもっと複雑
 - 前以て, 非ゼロ非対角ブロックの数はわからない

movie

マトリクス生成まで

- 一次元のときは, index, itemに関連した情報を簡単に作ることができた
 - 非ゼロ非対角成分の数は2
 - 番号が自分に対して : +1と-1
- 三次元の場合はもっと複雑
 - 非ゼロ非対角ブロックの数は7~26 (現在の形状)
 - 実際はもっと複雑
 - 前以て, 非ゼロ非対角ブロックの数はわからない
- INL[N], INU[N], IAL[N][NL], IAU[N][NU]を使って非ゼロ非対角ブロック数 (上下) を予備的に勘定する

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();

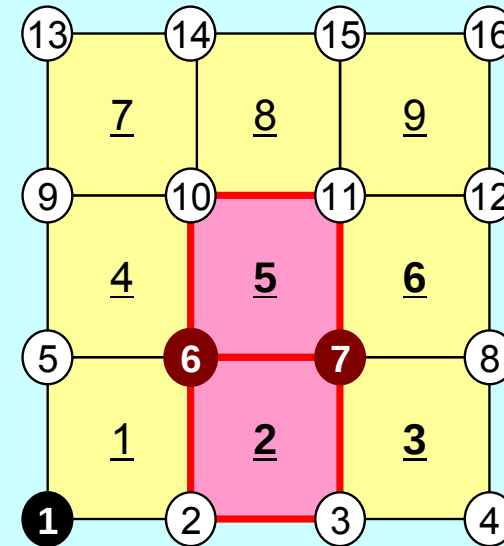
    MAT_CON0();
    MAT_CON1();

    MAT_ASS_MAIN();
    MAT_ASS_BC();

    SOLVE33();

    RECOVER_STRESS();
    OUTPUT_UCD();
}

```

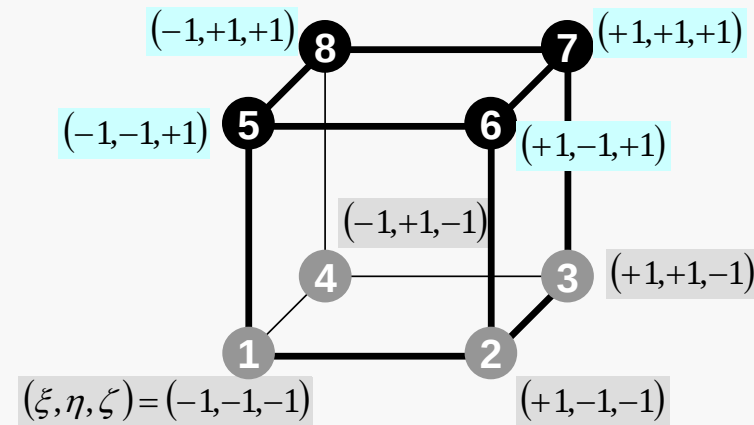
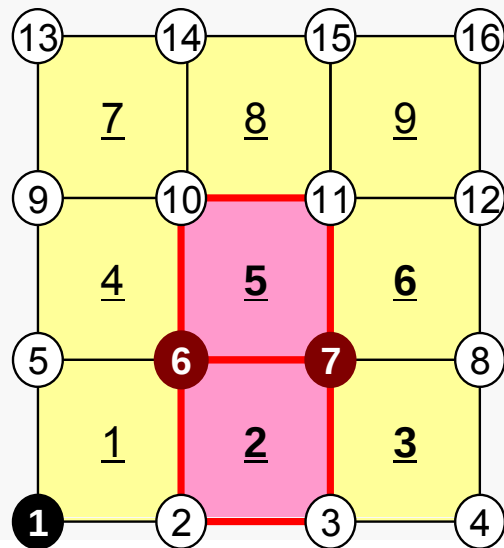


MAT_CON0: INL, INU, IAL, IAU生成
 MAT_CON1: index, item生成

とりあえず1から始まる節点番号を記憶

MAT_CON0 : 全体構成

```
do icel= 1, ICELTOT
  8節点相互の関係から,
  INL, INU, IAL, IAUを生成
  (FIND_NODE)
enddo
```



行列コネクティビティ生成： MAT_CON0 (1/4)

```
#include <stdio.h>
#include "pfem_util.h"
#include "allocate.h"
extern FILE *fp_log;
extern void mSORT(int*, int*, int);
static void FIND_TS_NODE (int, int);

void MAT_CON0 ()
{
    int i, j, k, icel, in;
    int in1, in2, in3, in4, in5, in6, in7, in8;
    int NN;

    N2= 256; (この変数は使用しない)
    NU= 26;
    NL= 26;

    INL=(KINT* ) allocate_vector (sizeof (KINT), N);
    IAL=(KINT**) allocate_matrix (sizeof (KINT), N, NL);
    INU=(KINT* ) allocate_vector (sizeof (KINT), N);
    IAU=(KINT**) allocate_matrix (sizeof (KINT), N, NU);

    for (i=0; i<N; i++) INL[i]=0;
    for (i=0; i<N; i++) for (j=0; j<NL; j++) IAL[i][j]=0;
    for (i=0; i<N; i++) INU[i]=0;
    for (i=0; i<N; i++) for (j=0; j<NU; j++) IAU[i][j]=0;
}
```

NU, NL:
各節点における
(上下)非ゼロ非対角
ブロックの最大数
(接続する節点数)

今の問題の場合は
わかっているので、
このようにできる

不明の場合の実装:
⇒レポート課題

行列コネクティビティ生成： MAT_CON0 (1/4)

```
#include <stdio.h>
#include "pfem_util.h"
#include "allocate.h"
extern FILE *fp_log;
extern void mSORT(int*, int*, int);
static void FIND_TS_NODE (int, int);

void MAT_CON0 ()
{
    int i, j, k, icel, in;
    int in1, in2, in3, in4, in5, in6, in7, in8;
    int NN;

    N2= 256; (この変数は使用しない)
    NU= 26;
    NL= 26;

    INL=(KINT* ) allocate_vector (sizeof (KINT), N);
    IAL=(KINT**) allocate_matrix (sizeof (KINT), N, NL);
    INU=(KINT* ) allocate_vector (sizeof (KINT), N);
    IAU=(KINT**) allocate_matrix (sizeof (KINT), N, NU);

    for (i=0; i<N; i++) INL[i]=0;
    for (i=0; i<N; i++) for (j=0; j<NL; j++) IAL[i][j]=0;
    for (i=0; i<N; i++) INU[i]=0;
    for (i=0; i<N; i++) for (j=0; j<NU; j++) IAU[i][j]=0;
```

変数名	サイズ	内 容
INL, INU	[N]	各節点の上・下三角ブロック数
IAL, IAU	[N] [NL], [N] [NU]	各節点の上・下三角ブロック (列番号)

行列コネクティビティ生成： MAT_CON0 (2/4) : 1から始まる番号

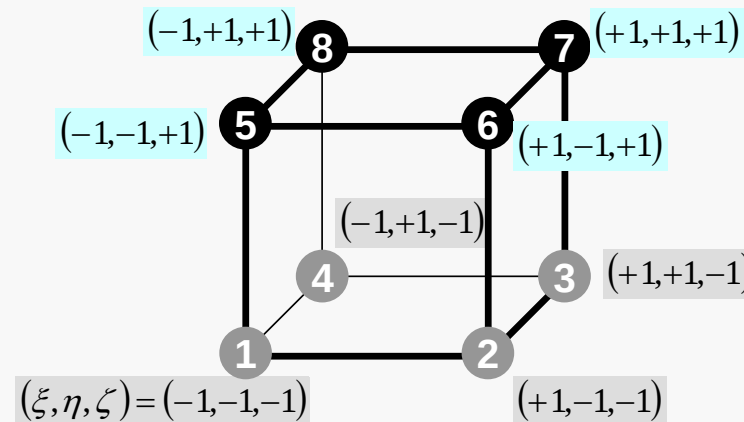
```
for( ice|=0; ice|< ICELTOT; ice|++ ) {
```

```
  in1=|CELNOD[ice|][0];  
  in2=|CELNOD[ice|][1];  
  in3=|CELNOD[ice|][2];  
  in4=|CELNOD[ice|][3];  
  in5=|CELNOD[ice|][4];  
  in6=|CELNOD[ice|][5];  
  in7=|CELNOD[ice|][6];  
  in8=|CELNOD[ice|][7];
```

```
  FIND_TS_NODE (in1, in2);  
  FIND_TS_NODE (in1, in3);  
  FIND_TS_NODE (in1, in4);  
  FIND_TS_NODE (in1, in5);  
  FIND_TS_NODE (in1, in6);  
  FIND_TS_NODE (in1, in7);  
  FIND_TS_NODE (in1, in8);
```

```
  FIND_TS_NODE (in2, in1);  
  FIND_TS_NODE (in2, in3);  
  FIND_TS_NODE (in2, in4);  
  FIND_TS_NODE (in2, in5);  
  FIND_TS_NODE (in2, in6);  
  FIND_TS_NODE (in2, in7);  
  FIND_TS_NODE (in2, in8);
```

```
  FIND_TS_NODE (in3, in1);  
  FIND_TS_NODE (in3, in2);  
  FIND_TS_NODE (in3, in4);  
  FIND_TS_NODE (in3, in5);  
  FIND_TS_NODE (in3, in6);  
  FIND_TS_NODE (in3, in7);  
  FIND_TS_NODE (in3, in8);
```



節点探索：FIND_TS_NODE

INL, INU, IAL, IAU探索：一次元ではこの部分は手動

```
static void FIND_TS_NODE (int ip1, int ip2)
{
    int kk, icou;
    if( ip1 > ip2 ) {
        for(kk=1; kk<=INL[ip1-1]; kk++) {
            if(ip2 == IAL[ip1-1][kk-1]) return;
        }
        icou=INL[ip1-1]+1;
        IAL[ip1-1][icou-1]=ip2;
        INL[ip1-1]=icou;
        return;
    }
    if( ip2 > ip1 ) {
        for(kk=1; kk<=INU[ip1-1]; kk++) {
            if(ip2 == IAU[ip1-1][kk-1]) return;
        }
        icou=INU[ip1-1]+1;
        IAU[ip1-1][icou-1]=ip2;
        INU[ip1-1]=icou;
        return;
    }
}
```

変数名	サイズ	内 容
INL, INU	[N]	各節点の上・下三角ブロック数
IAL, IAU	[N] [NL], [N] [NU]	各節点の上・下三角ブロック (列番号)

節点探索：FIND_TS_NODE

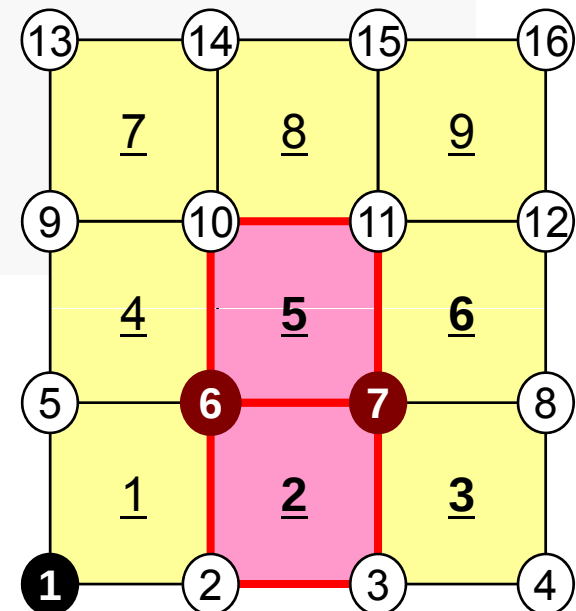
一次元ではこの部分は手動

```
static void FIND_TS_NODE (int ip1,int ip2)
```

```
{
    int kk, icou;
    if( ip1 > ip2 ) {
        for (kk=1; kk<=INL[ip1-1]; kk++) {
            if(ip2 == IAL[ip1-1][kk-1]) return;
        }
        icou=INL[ip1-1]+1;
        IAL[ip1-1][icou-1]=ip2;
        INL[ip1-1]=icou;
        return;
    }

    if( ip2 > ip1 ) {
        for (kk=1; kk<=INU[ip1-1]; kk++) {
            if(ip2 == IAU[ip1-1][kk-1]) return;
        }
        icou=INU[ip1-1]+1;
        IAU[ip1-1][icou-1]=ip2;
        INU[ip1-1]=icou;
        return;
    }
}
```

既にIAL, IAUに含まれている
場合は、次のペアへ

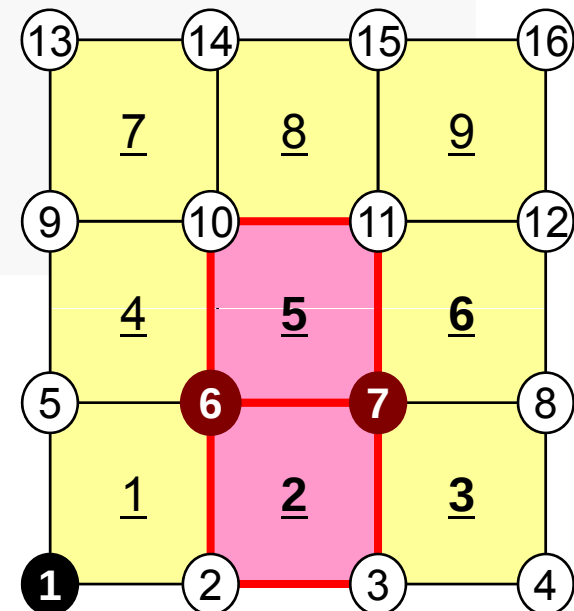


節点探索 : FIND_TS_NODE

一次元ではこの部分は手動

```
static void FIND_TS_NODE (int ip1,int ip2)
{
    int kk, icou;
    if( ip1 > ip2 ) {
        for(kk=1;kk<=INL[ip1-1];kk++) {
            if(ip2 == IAL[ip1-1][kk-1]) return;
        }
        icou=INL[ip1-1]+1;
        IAL[ip1-1][icou-1]=ip2;
        INL[ip1-1]=icou;
        return;
    }
    if( ip2 > ip1 ) {
        for(kk=1;kk<=INU[ip1-1];kk++) {
            if(ip2 == IAU[ip1-1][kk-1]) return;
        }
        icou=INU[ip1-1]+1;
        IAU[ip1-1][icou-1]=ip2;
        INU[ip1-1]=icou;
        return;
    }
}
```

IAL, IAUに含まれていない
場合は, INL・INUに1を加えて
IAL, IAUに格納



行列コネクティビティ生成： MAT_CON0 (3/4)

```

FIND_TS_NODE (in4, in1);
FIND_TS_NODE (in4, in2);
FIND_TS_NODE (in4, in3);
FIND_TS_NODE (in4, in5);
FIND_TS_NODE (in4, in6);
FIND_TS_NODE (in4, in7);
FIND_TS_NODE (in4, in8);

```

```

FIND_TS_NODE (in5, in1);
FIND_TS_NODE (in5, in2);
FIND_TS_NODE (in5, in3);
FIND_TS_NODE (in5, in4);
FIND_TS_NODE (in5, in6);
FIND_TS_NODE (in5, in7);
FIND_TS_NODE (in5, in8);

```

```

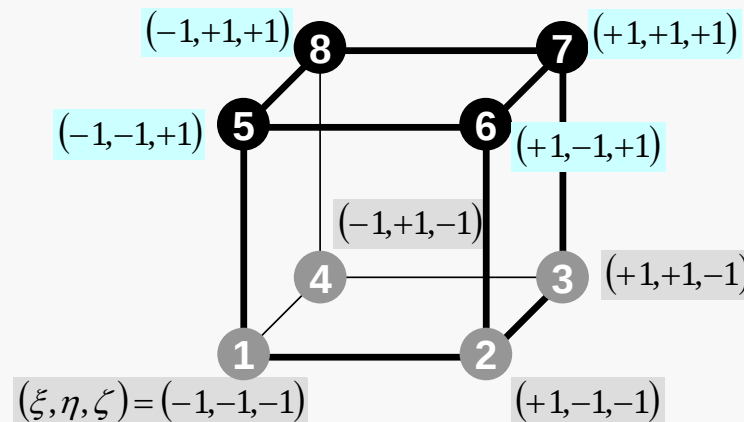
FIND_TS_NODE (in6, in1);
FIND_TS_NODE (in6, in2);
FIND_TS_NODE (in6, in3);
FIND_TS_NODE (in6, in4);
FIND_TS_NODE (in6, in5);
FIND_TS_NODE (in6, in7);
FIND_TS_NODE (in6, in8);

```

```

FIND_TS_NODE (in7, in1);
FIND_TS_NODE (in7, in2);
FIND_TS_NODE (in7, in3);
FIND_TS_NODE (in7, in4);
FIND_TS_NODE (in7, in5);
FIND_TS_NODE (in7, in6);
FIND_TS_NODE (in7, in8);

```



行列コネクティビティ生成： MAT_CON0 (4/4)

```

        FIND_TS_NODE (in8, in1);
        FIND_TS_NODE (in8, in2);
        FIND_TS_NODE (in8, in3);
        FIND_TS_NODE (in8, in4);
        FIND_TS_NODE (in8, in5);
        FIND_TS_NODE (in8, in6);
        FIND_TS_NODE (in8, in7);
    }

    for (in=0; in<N; in++) {
        NN=INL[in];
        for (k=0; k<NN; k++) {
            NCOL1[k]=IAL[in][k];
        }
        mSORT (NCOL1, NCOL2, NN);
        for (k=NN; k>0; k--) {
            IAL[in][NN-k]= NCOL1[NCOL2[k]-1];
        }

        NN=INU[in];
        for (k=0; k<NN; k++) {
            NCOL1[k]=IAU[in][k];
        }
        mSORT (NCOL1, NCOL2, NN);
        for (k=NN; k>0; k--) {
            IAU[in][NN-k]= NCOL1[NCOL2[k]-1];
        }
    }
}

```

各節点において,
IAL[i][k], IAU[i][k]が小さい番号から
大きい番号に並ぶようにソート
(単純なバブルソート)
せいぜい100程度のものをソートする

CRS形式への変換 : MAT_CON1

```

#include <stdio.h>
#include "pfem_util.h"
#include "allocate.h"
extern FILE* fp_log;
void MAT_CON1()
{
    int i, k, kk;

    indexL = (KINT*) allocate_vector(sizeof(KINT), N+1);
    indexU = (KINT*) allocate_vector(sizeof(KINT), N+1);
    for (i=0; i<N+1; i++) indexL[i]=0;
    for (i=0; i<N+1; i++) indexU[i]=0;

    for (i=0; i<N; i++) {
        indexL[i+1] = indexL[i] + INL[i];
        indexU[i+1] = indexU[i] + INU[i];
    }
    NPL = indexL[N];
    NPU = indexU[N];

    itemL = (KINT*) allocate_vector(sizeof(KINT), NPL);
    itemU = (KINT*) allocate_vector(sizeof(KINT), NPU);

    for (i=0; i<N; i++) {
        for (k=0; k<INL[i]; k++) {
            kk = k + indexL[i];
            itemL[kk] = IAL[i][k];
        }
        for (k=0; k<INU[i]; k++) {
            kk = k + indexU[i];
            itemU[kk] = IAU[i][k];
        }
    }
    deallocate_vector(INL);
    deallocate_vector(INU);
    deallocate_vector(IAL);
    deallocate_vector(IAU);
}

```

C

$$\text{indexL}[i+1] = \sum_{k=0}^i \text{INL}[k]$$

$$\text{indexU}[i+1] = \sum_{k=0}^i \text{INU}[k]$$

$$\text{indexL}[0] = \text{indexU}[0] = 0$$

FORTRAN

$$\text{indexL}[i] = \sum_{k=1}^i \text{INL}[k]$$

$$\text{indexU}[i] = \sum_{k=1}^i \text{INU}[k]$$

$$\text{indexL}[0] = \text{indexU}[0] = 0$$

CRS形式への変換：MAT_CON1

```

#include <stdio.h>
#include "pfem_util.h"
#include "allocate.h"
extern FILE* fp_log;
void MAT_CON1()
{
    int i, k, kk;

    indexL = (KINT*) allocate_vector(sizeof(KINT), N+1);
    indexU = (KINT*) allocate_vector(sizeof(KINT), N+1);
    for (i=0; i<N+1; i++) indexL[i]=0;
    for (i=0; i<N+1; i++) indexU[i]=0;

    for (i=0; i<N; i++) {
        indexL[i+1] = indexL[i] + INL[i];
        indexU[i+1] = indexU[i] + INU[i];
    }
    NPL = indexL[N];
    NPU = indexU[N];

    itemL = (KINT*) allocate_vector(sizeof(KINT), NPL);
    itemU = (KINT*) allocate_vector(sizeof(KINT), NPU);

    for (i=0; i<N; i++) {
        for (k=0; k<INL[i]; k++) {
            kk = k + indexL[i];
            itemL[kk] = IAL[i][k];
        }
        for (k=0; k<INU[i]; k++) {
            kk = k + indexU[i];
            itemU[kk] = IAU[i][k];
        }
    }
    deallocate_vector(INL);
    deallocate_vector(INU);
    deallocate_vector(IAL);
    deallocate_vector(IAU);
}

```

NPL=indexL[N]

itemLのサイズ

非ゼロ非対角ブロック(下三角)総数

NPU=indexU[N]

itemUのサイズ

非ゼロ非対角ブロック(上三角)総数

CRS形式への変換：MAT_CON1

```

#include <stdio.h>
#include "pfem_util.h"
#include "allocate.h"
extern FILE* fp_log;
void MAT_CON1()
{
    int i, k, kk;

    indexL=(KINT*)allocate_vector(sizeof(KINT), N+1);
    indexU=(KINT*)allocate_vector(sizeof(KINT), N+1);
    for(i=0; i<N+1; i++) indexL[i]=0;
    for(i=0; i<N+1; i++) indexU[i]=0;

    for(i=0; i<N; i++) {
        indexL[i+1]=indexL[i]+INL[i];
        indexU[i+1]=indexU[i]+INU[i];
    }
    NPL=indexL[N];
    NPU=indexU[N];

    itemL=(KINT*)allocate_vector(sizeof(KINT), NPL);
    itemU=(KINT*)allocate_vector(sizeof(KINT), NPU);

    for(i=0; i<N; i++) {
        for(k=0; k<INL[i]; k++) {
            kk=k+indexL[i];
            itemL[kk]=IAL[i][k];
        }
        for(k=0; k<INU[i]; k++) {
            kk=k+indexU[i];
            itemU[kk]=IAU[i][k];
        }
    }
    deallocate_vector(INL);
    deallocate_vector(INU);
    deallocate_vector(IAL);
    deallocate_vector(IAU);
}

```

itemL, itemUにも1から
始まる節点番号を記憶

CRS形式への変換：MAT_CON1

```

#include <stdio.h>
#include "pfem_util.h"
#include "allocate.h"
extern FILE* fp_log;
void MAT_CON1()
{
    int i, k, kk;

    indexL=(KINT*)allocate_vector(sizeof(KINT), N+1);
    indexU=(KINT*)allocate_vector(sizeof(KINT), N+1);
    for(i=0; i<N+1; i++) indexL[i]=0;
    for(i=0; i<N+1; i++) indexU[i]=0;

    for(i=0; i<N; i++) {
        indexL[i+1]=indexL[i]+INL[i];
        indexU[i+1]=indexU[i]+INU[i];
    }
    NPL=indexL[N];
    NPU=indexU[N];

    itemL=(KINT*)allocate_vector(sizeof(KINT), NPL);
    itemU=(KINT*)allocate_vector(sizeof(KINT), NPU);

    for(i=0; i<N; i++) {
        for(k=0; k<INL[i]; k++) {
            kk=k+indexL[i];
            itemL[kk]=IAL[i][k];
        }
        for(k=0; k<INU[i]; k++) {
            kk=k+indexU[i];
            itemU[kk]=IAU[i][k];
        }
    }

    deallocate_vector(INL);
    deallocate_vector(INU);
    deallocate_vector(IAL);
    deallocate_vector(IAU);
}

```

これらはもはや不要

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_CONO();
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_CONO();
    MAT_CON1();

    MAT_ASS_MAIN();
    MAT_ASS_BC();

    SOLVE33();

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

MAT_ASS_MAIN : 全体構成

```

do kpn= 1, 2      ガウス積分点番号 (ζ方向)
  do jpn= 1, 2    ガウス積分点番号 (η方向)
    do ipn= 1, 2  ガウス積分点番号 (ξ方向)
      ガウス積分点 (8個) における形状関数,
      およびその「自然座標系」における微分の算出
    enddo
  enddo
enddo

```

```

do icel= 1, ICELTOT  要素ループ
  8節点の座標から, ガウス積分点における, 形状関数の「全体座標系」における微分,
  およびヤコビアンを算出 (JACOBI)

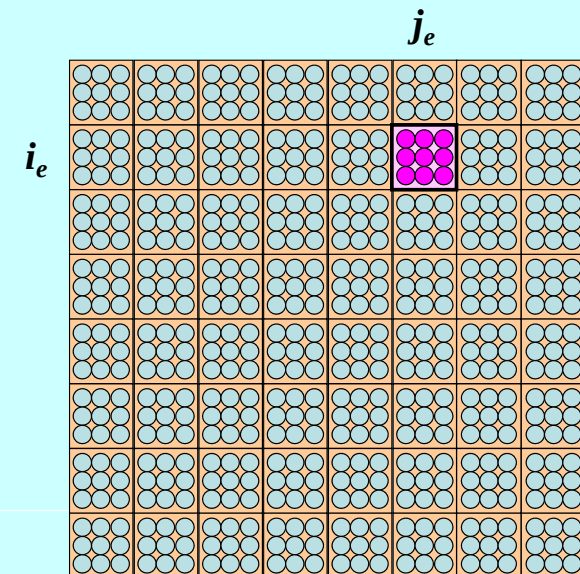
```

```

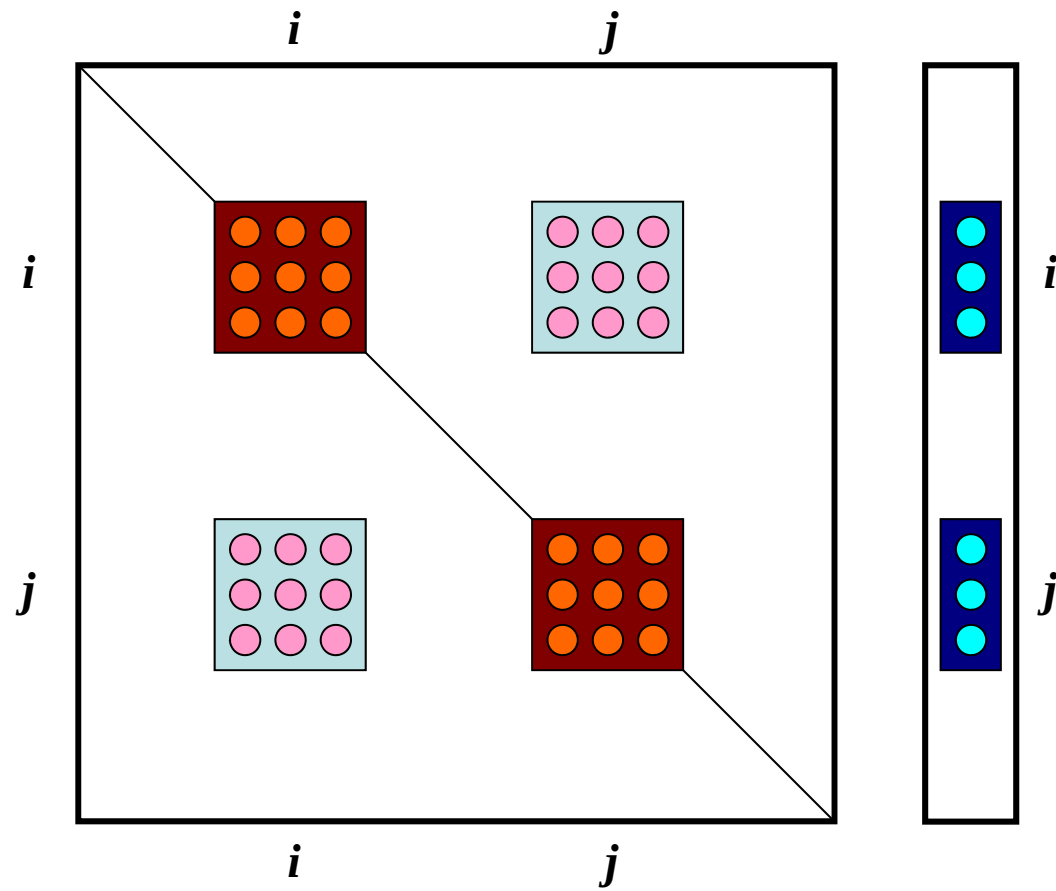
do ie= 1, 8          局所節点番号
  do je= 1, 8        局所節点番号
    全体節点番号 : ip, jp
    Aip, jp の itemL, itemU におけるアドレス : kk

    do kpn= 1, 2      ガウス積分点番号 (ζ方向)
      do jpn= 1, 2    ガウス積分点番号 (η方向)
        do ipn= 1, 2  ガウス積分点番号 (ξ方向)
          要素積分⇒要素行列成分計算, 全体行列への足しこみ
        enddo
      enddo
    enddo
  enddo
enddo
enddo

```



ブロックとして記憶



係数行列 : MAT_ASS_MAIN (1/7)

```
#include <stdio.h>
#include <math.h>
#include "pfem_util.h"
#include "allocate.h"
extern FILE *fp_log;
extern void JACOBI();
void MAT_ASS_MAIN()
{
    int i, k, kk;
    int ip, jp, kp;
    int ipn, jpn, kpn;
    int icel;
    int ie, je;
    int iiS, iiE;
    int in1, in2, in3, in4, in5, in6, in7, in8;
    double valA, valB, valX;
    double QP1, QM1, EP1, EM1, TP1, TM1;
    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 PNXi, PNYi, PNZi, PNXj, PNYj, PNZj;
    double VOL;
    double coef;
    double a11, a12, a13, a21, a22, a23, a31, a32, a33;
```

```
KINT nodLOCAL[8];
```

```
AL=(KREAL*) allocate_vector(sizeof(KREAL), 9*NPL);
AU=(KREAL*) allocate_vector(sizeof(KREAL), 9*NPU);
B =(KREAL*) allocate_vector(sizeof(KREAL), 3*N );
D =(KREAL*) allocate_vector(sizeof(KREAL), 9*N);
X =(KREAL*) allocate_vector(sizeof(KREAL), 3*N);
```

下三角ブロック
上三角ブロック
右辺ベクトル
対角ブロック
未知数ベクトル

```
for (i=0; i<9*NPL; i++) AL[i]=0.0;
for (i=0; i<9*NPU; i++) AU[i]=0.0;
for (i=0; i<3*N ; i++) B[i]=0.0;
for (i=0; i<9*N ; i++) D[i]=0.0;
for (i=0; i<3*N ; i++) X[i]=0.0;
```

係数行列 : MAT_ASS_MAIN (2/7)

WEI[0]= 1.0000000000e0;
WEI[1]= 1.0000000000e0;

POS[0]= -0.5773502692e0;
POS[1]= 0.5773502692e0;

POS: 積分点座標
WEI: 重み係数

/**

INIT.

PNQ - 1st-order derivative of shape function by QSI
PNE - 1st-order derivative of shape function by ETA
PNT - 1st-order derivative of shape function by ZET

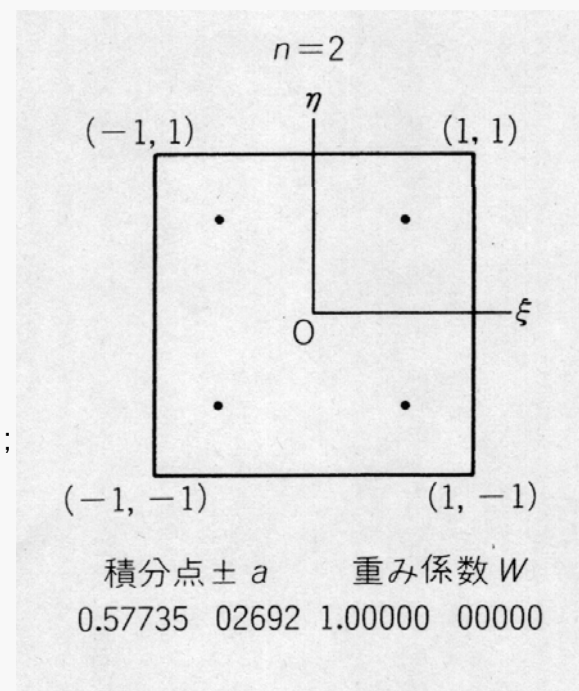
*/

valA= POISSON / (1.e0-POISSON);
valB= (1.e0-2.e0*POISSON)/(2.e0*(1.e0-POISSON));
valX= ELAST*(1.e0-POISSON)/((1.e0+POISSON)*(1.e0-2.e0*POISSON));

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

```
for (ip=0; ip<2; ip++) {
  for (jp=0; jp<2; jp++) {
    for (kp=0; kp<2; kp++) {
      QP1= 1.e0 + POS[ip];
      QM1= 1.e0 - POS[ip];
      EP1= 1.e0 + POS[jp];
      EM1= 1.e0 - POS[jp];
      TP1= 1.e0 + POS[kp];
      TM1= 1.e0 - POS[kp];
```

```
SHAPE[ip][jp][kp][0]= 08th * QM1 * EM1 * TM1;
SHAPE[ip][jp][kp][1]= 08th * QP1 * EM1 * TM1;
SHAPE[ip][jp][kp][2]= 08th * QP1 * EP1 * TM1;
SHAPE[ip][jp][kp][3]= 08th * QM1 * EP1 * TM1;
SHAPE[ip][jp][kp][4]= 08th * QM1 * EM1 * TP1;
SHAPE[ip][jp][kp][5]= 08th * QP1 * EM1 * TP1;
SHAPE[ip][jp][kp][6]= 08th * QP1 * EP1 * TP1;
SHAPE[ip][jp][kp][7]= 08th * QM1 * EP1 * TP1;
```



系数行列：MAT_ASS_MAIN (2/7)

```

WEI[0]= 1.0000000000e0;
WEI[1]= 1.0000000000e0;

POS[0]= -0.5773502692e0;
POS[1]= 0.5773502692e0;

/****
INIT.
PNQ - 1st-order derivative of shape function by QSI
PNE - 1st-order derivative of shape function by ETA
PNT - 1st-order derivative of shape function by ZET
****/

valA=          POISSON /          (1.e0-POISSON);
valB= (1.e0-2.e0*POISSON)/(2.e0*(1.e0-POISSON));
valX= ELAST*(1.e0-POISSON)/((1.e0+POISSON)*(1.e0-2.e0*POISSON));

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

for (ip=0; ip<2; ip++) {
    for (jp=0; jp<2; jp++) {
        for (kp=0; kp<2; kp++) {
            QP1= 1.e0 + POS[ip];
            QM1= 1.e0 - POS[ip];
            EP1= 1.e0 + POS[jp];
            EM1= 1.e0 - POS[jp];
            TP1= 1.e0 + POS[kp];
            TM1= 1.e0 - POS[kp];

            SHAPE[ip][jp][kp][0]= 08th * QM1 * EM1 * TM1;
            SHAPE[ip][jp][kp][1]= 08th * QP1 * EM1 * TM1;
            SHAPE[ip][jp][kp][2]= 08th * QP1 * EP1 * TM1;
            SHAPE[ip][jp][kp][3]= 08th * QM1 * EP1 * TM1;
            SHAPE[ip][jp][kp][4]= 08th * QM1 * EM1 * TP1;
            SHAPE[ip][jp][kp][5]= 08th * QP1 * EM1 * TP1;
            SHAPE[ip][jp][kp][6]= 08th * QP1 * EP1 * TP1;
            SHAPE[ip][jp][kp][7]= 08th * QM1 * EP1 * TP1;
        }
    }
}

```

ひずみ⇒応力関係

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

$[D]$

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

系数行列：MAT_ASS_MAIN (2/7)

```

WEI[0]= 1.0000000000e0;
WEI[1]= 1.0000000000e0;

POS[0]= -0.5773502692e0;
POS[1]= 0.5773502692e0;

/****
INIT.
PNQ - 1st-order derivative of shape function by QSI
PNE - 1st-order derivative of shape function by ETA
PNT - 1st-order derivative of shape function by ZET
****/

valA=          POISSON /          (1.e0-POISSON);
valB= (1.e0-2.e0*POISSON)/(2.e0*(1.e0-POISSON));
valX= ELAST*(1.e0-POISSON)/((1.e0+POISSON)*(1.e0-2.e0*POISSON));

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

for (ip=0; ip<2; ip++) {
    for (jp=0; jp<2; jp++) {
        for (kp=0; kp<2; kp++) {
            QP1= 1.e0 + POS[ip];
            QM1= 1.e0 - POS[ip];
            EP1= 1.e0 + POS[jp];
            EM1= 1.e0 - POS[jp];
            TP1= 1.e0 + POS[kp];
            TM1= 1.e0 - POS[kp];

            SHAPE[ip][jp][kp][0]= 08th * QM1 * EM1 * TM1;
            SHAPE[ip][jp][kp][1]= 08th * QP1 * EM1 * TM1;
            SHAPE[ip][jp][kp][2]= 08th * QP1 * EP1 * TM1;
            SHAPE[ip][jp][kp][3]= 08th * QM1 * EP1 * TM1;
            SHAPE[ip][jp][kp][4]= 08th * QM1 * EM1 * TP1;
            SHAPE[ip][jp][kp][5]= 08th * QP1 * EM1 * TP1;
            SHAPE[ip][jp][kp][6]= 08th * QP1 * EP1 * TP1;
            SHAPE[ip][jp][kp][7]= 08th * QM1 * EP1 * TP1;
        }
    }
}

```

$$\text{valX} = \frac{E(1-\nu)}{(1+\nu)(1-2\nu)}$$

$$\text{valA} = \frac{\nu}{(1-\nu)} \frac{E(1-\nu)}{(1+\nu)(1-2\nu)}$$

$$\text{valB} = \frac{(1-2\nu)}{2(1-\nu)} \frac{E(1-\nu)}{(1+\nu)(1-2\nu)}$$

系数行列：MAT_ASS_MAIN (2/7)

```

WEI[0]= 1.0000000000e0;
WEI[1]= 1.0000000000e0;

POS[0]= -0.5773502692e0;
POS[1]= 0.5773502692e0;

/**
INIT.
PNQ - 1st-order derivative of shape function by QSI
PNE - 1st-order derivative of shape function by ETA
PNT - 1st-order derivative of shape function by ZET
***/

valA=          POISSON /          (1.e0-POISSON);
valB= (1.e0-2.e0*POISSON)/(2.e0*(1.e0-POISSON));
valX= ELAST*(1.e0-POISSON)/((1.e0+POISSON)*(1.e0-2.e0*POISSON));

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

for (ip=0; ip<2; ip++) {
    for (jp=0; jp<2; jp++) {
        for (kp=0; kp<2; kp++) {
            QP1= 1.e0 + POS[ip];
            QM1= 1.e0 - POS[ip];
            EP1= 1.e0 + POS[jp];
            EM1= 1.e0 - POS[jp];
            TP1= 1.e0 + POS[kp];
            TM1= 1.e0 - POS[kp];

            SHAPE[ip][jp][kp][0]= 08th * QM1 * EM1 * TM1;
            SHAPE[ip][jp][kp][1]= 08th * QP1 * EM1 * TM1;
            SHAPE[ip][jp][kp][2]= 08th * QP1 * EP1 * TM1;
            SHAPE[ip][jp][kp][3]= 08th * QM1 * EP1 * TM1;
            SHAPE[ip][jp][kp][4]= 08th * QM1 * EM1 * TP1;
            SHAPE[ip][jp][kp][5]= 08th * QP1 * EM1 * TP1;
            SHAPE[ip][jp][kp][6]= 08th * QP1 * EP1 * TP1;
            SHAPE[ip][jp][kp][7]= 08th * QM1 * EP1 * TP1;
        }
    }
}

```

$$\begin{aligned}
 \text{valX} &= \frac{E(1-\nu)}{(1+\nu)(1-2\nu)} \\
 \text{valA} &= \frac{\nu}{(1-\nu)} \frac{E(1-\nu)}{(1+\nu)(1-2\nu)} = \frac{E\nu}{(1+\nu)(1-2\nu)} \\
 \text{valB} &= \frac{(1-2\nu)}{2(1-\nu)} \frac{E(1-\nu)}{(1+\nu)(1-2\nu)} = \frac{E}{2(1+\nu)}
 \end{aligned}$$

ひずみ⇒応力関係

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

$$[D]$$

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

系数行列：MAT_ASS_MAIN (2/7)

```

WEI[0]= 1.0000000000e0;
WEI[1]= 1.0000000000e0;

POS[0]= -0.5773502692e0;
POS[1]= 0.5773502692e0;

/**
INIT.
PNQ - 1st-order derivative of shape function by QSI
PNE - 1st-order derivative of shape function by ETA
PNT - 1st-order derivative of shape function by ZET
***/

valA= POISSON / (1.e0-POISSON);
valB= (1.e0-2.e0*POISSON)/(2.e0*(1.e0-POISSON));
valX= ELAST*(1.e0-POISSON)/((1.e0+POISSON)*(1.e0-2.e0*POISSON));

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

for(ip=0; ip<2; ip++) {
    for(jp=0; jp<2; jp++) {
        for(kp=0; kp<2; kp++) {
            QP1= 1.e0 + POS[ip];
            QM1= 1.e0 - POS[ip];
            EP1= 1.e0 + POS[jp];
            EM1= 1.e0 - POS[jp];
            TP1= 1.e0 + POS[kp];
            TM1= 1.e0 - POS[kp];

            SHAPE[ip][jp][kp][0]= 08th * QM1 * EM1 * TM1;
            SHAPE[ip][jp][kp][1]= 08th * QP1 * EM1 * TM1;
            SHAPE[ip][jp][kp][2]= 08th * QP1 * EP1 * TM1;
            SHAPE[ip][jp][kp][3]= 08th * QM1 * EP1 * TM1;
            SHAPE[ip][jp][kp][4]= 08th * QM1 * EM1 * TP1;
            SHAPE[ip][jp][kp][5]= 08th * QP1 * EM1 * TP1;
            SHAPE[ip][jp][kp][6]= 08th * QP1 * EP1 * TP1;
            SHAPE[ip][jp][kp][7]= 08th * QM1 * EP1 * TP1;
        }
    }
}

```

$$\begin{aligned}
 QP1(i) &= (1 + \xi_i), & QM1(i) &= (1 - \xi_i) \\
 EP1(j) &= (1 + \eta_j), & EM1(j) &= (1 - \eta_j) \\
 TP1(k) &= (1 + \zeta_k), & TM1(k) &= (1 - \zeta_k)
 \end{aligned}$$

系数行列：MAT_ASS_MAIN (2/7)

```

WEI[0]= 1.0000000000e0;
WEI[1]= 1.0000000000e0;

POS[0]= -0.5773502692e0;
POS[1]= 0.5773502692e0;

/**
INIT.
PNQ - 1st-order derivative of shape function by QSI
PNE - 1st-order derivative of shape function by ETA
PNT - 1st-order derivative of shape function by ZET
***/

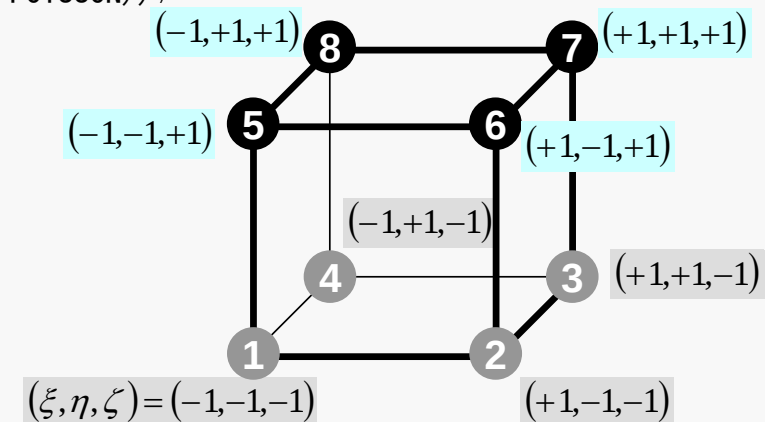
valA= POISSON / (1. e0-POISSON);
valB= (1. e0-2. e0*POISSON)/(2. e0*(1. e0-POISSON));
valX= ELAST*(1. e0-POISSON)/((1. e0+POISSON)*(1. e0-2. e0*POISSON));

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

for (ip=0; ip<2; ip++) {
    for (jp=0; jp<2; jp++) {
        for (kp=0; kp<2; kp++) {
            QP1= 1. e0 + POS[ip];
            QM1= 1. e0 - POS[ip];
            EP1= 1. e0 + POS[jp];
            EM1= 1. e0 - POS[jp];
            TP1= 1. e0 + POS[kp];
            TM1= 1. e0 - POS[kp];

            SHAPE[ip][jp][kp][0]= 08th * QM1 * EM1 * TM1;
            SHAPE[ip][jp][kp][1]= 08th * QP1 * EM1 * TM1;
            SHAPE[ip][jp][kp][2]= 08th * QP1 * EP1 * TM1;
            SHAPE[ip][jp][kp][3]= 08th * QM1 * EP1 * TM1;
            SHAPE[ip][jp][kp][4]= 08th * QM1 * EM1 * TP1;
            SHAPE[ip][jp][kp][5]= 08th * QP1 * EM1 * TP1;
            SHAPE[ip][jp][kp][6]= 08th * QP1 * EP1 * TP1;
            SHAPE[ip][jp][kp][7]= 08th * QM1 * EP1 * TP1;
        }
    }
}

```



系数行列：MAT_ASS_MAIN (2/7)

```

WEI[0]= 1.0000000000e0;
WEI[1]= 1.0000000000e0;

POS[0]= -0.5773502692e0;
POS[1]= 0.5773502692e0;

/****
INIT.
PNQ - 1st-order derivative of shape function by QSI
PNE - 1st-order derivative of shape function by ETA
PNT - 1st-order derivative of shape function by ZET
****/

valA= POISSON / (1.e0-POISSON);
valB= (1.e0-2.e0*POISSON)/(2.e0*(1.e0-POISSON));
valX= ELAST*(1.e0-POISSON)/((1.e0+POISSON)*(1.e0-2.e0*POISSON));

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

for (ip=0; ip<2; ip++) {
    for (jp=0; jp<2; jp++) {
        for (kp=0; kp<2; kp++) {
            QP1= 1.e0 + POS[ip];
            QM1= 1.e0 - POS[ip];
            EP1= 1.e0 + POS[jp];
            EM1= 1.e0 - POS[jp];
            TP1= 1.e0 + POS[kp];
            TM1= 1.e0 - POS[kp];

            SHAPE[ip][jp][kp][0]= 08th * QM1 * EM1 * TM1;
            SHAPE[ip][jp][kp][1]= 08th * QP1 * EM1 * TM1;
            SHAPE[ip][jp][kp][2]= 08th * QP1 * EP1 * TM1;
            SHAPE[ip][jp][kp][3]= 08th * QM1 * EP1 * TM1;
            SHAPE[ip][jp][kp][4]= 08th * QM1 * EM1 * TP1;
            SHAPE[ip][jp][kp][5]= 08th * QP1 * EM1 * TP1;
            SHAPE[ip][jp][kp][6]= 08th * QP1 * EP1 * TP1;
            SHAPE[ip][jp][kp][7]= 08th * QM1 * EP1 * TP1;
        }
    }
}

```

$$N_1(\xi, \eta, \zeta) = \frac{1}{8}(1-\xi)(1-\eta)(1-\zeta)$$

$$N_2(\xi, \eta, \zeta) = \frac{1}{8}(1+\xi)(1-\eta)(1-\zeta)$$

$$N_3(\xi, \eta, \zeta) = \frac{1}{8}(1+\xi)(1+\eta)(1-\zeta)$$

$$N_4(\xi, \eta, \zeta) = \frac{1}{8}(1-\xi)(1+\eta)(1-\zeta)$$

$$N_5(\xi, \eta, \zeta) = \frac{1}{8}(1-\xi)(1-\eta)(1+\zeta)$$

$$N_6(\xi, \eta, \zeta) = \frac{1}{8}(1+\xi)(1-\eta)(1+\zeta)$$

$$N_7(\xi, \eta, \zeta) = \frac{1}{8}(1+\xi)(1+\eta)(1+\zeta)$$

$$N_8(\xi, \eta, \zeta) = \frac{1}{8}(1-\xi)(1+\eta)(1+\zeta)$$

係数行列 : MAT_ASS_MAIN (3/7)

```

PNQ[jp][kp][0] = - 08th * EM1 * TM1;
PNQ[jp][kp][1] = + 08th * EM1 * TM1;
PNQ[jp][kp][2] = + 08th * EP1 * TM1;
PNQ[jp][kp][3] = - 08th * EP1 * TM1;
PNQ[jp][kp][4] = - 08th * EM1 * TP1;
PNQ[jp][kp][5] = + 08th * EM1 * TP1;
PNQ[jp][kp][6] = + 08th * EP1 * TP1;
PNQ[jp][kp][7] = - 08th * EP1 * TP1;
PNE[ip][kp][0] = - 08th * QM1 * TM1;
PNE[ip][kp][1] = - 08th * QP1 * TM1;
PNE[ip][kp][2] = + 08th * QP1 * TM1;
PNE[ip][kp][3] = + 08th * QM1 * TM1;
PNE[ip][kp][4] = - 08th * QM1 * TP1;
PNE[ip][kp][5] = - 08th * QP1 * TP1;
PNE[ip][kp][6] = + 08th * QP1 * TP1;
PNE[ip][kp][7] = + 08th * QM1 * TP1;
PNT[ip][jp][0] = - 08th * QM1 * EM1;
PNT[ip][jp][1] = - 08th * QP1 * EM1;
PNT[ip][jp][2] = - 08th * QP1 * EP1;
PNT[ip][jp][3] = - 08th * QM1 * EP1;
PNT[ip][jp][4] = + 08th * QM1 * EM1;
PNT[ip][jp][5] = + 08th * QP1 * EM1;
PNT[ip][jp][6] = + 08th * QP1 * EP1;
PNT[ip][jp][7] = + 08th * QM1 * EP1;
    }
}

for( icel=0; icel< ICELTOT; icel++) {
    in1=CELNOD[icel][0];
    in2=CELNOD[icel][1];
    in3=CELNOD[icel][2];
    in4=CELNOD[icel][3];
    in5=CELNOD[icel][4];
    in6=CELNOD[icel][5];
    in7=CELNOD[icel][6];
    in8=CELNOD[icel][7];

```

$$PNQ(j,k) = \frac{\partial N_l}{\partial \xi} (\xi = \xi_i, \eta = \eta_j, \zeta = \zeta_k)$$

$$PNE(i,k) = \frac{\partial N_l}{\partial \eta} (\xi = \xi_i, \eta = \eta_j, \zeta = \zeta_k)$$

$$PNT(i,j) = \frac{\partial N_l}{\partial \zeta} (\xi = \xi_i, \eta = \eta_j, \zeta = \zeta_k)$$

$$\frac{\partial N_1}{\partial \xi} (\xi_i, \eta_j, \zeta_k) = -\frac{1}{8} (1 - \eta_j) (1 - \zeta_k)$$

$$\frac{\partial N_2}{\partial \xi} (\xi_i, \eta_j, \zeta_k) = +\frac{1}{8} (1 - \eta_j) (1 - \zeta_k)$$

$$\frac{\partial N_3}{\partial \xi} (\xi_i, \eta_j, \zeta_k) = +\frac{1}{8} (1 + \eta_j) (1 - \zeta_k)$$

$$\frac{\partial N_3}{\partial \xi} (\xi_i, \eta_j, \zeta_k) = -\frac{1}{8} (1 + \eta_j) (1 - \zeta_k)$$

(ξ_i, η_j, ζ_k) における形状関数の一階微分

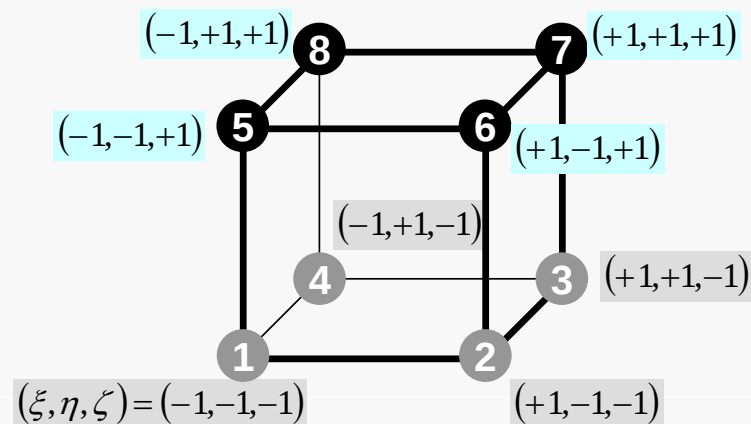
系数行列：MAT_ASS_MAIN (3/7)

```

PNQ[jp][kp][0] = - 08th * EM1 * TM1;
PNQ[jp][kp][1] = + 08th * EM1 * TM1;
PNQ[jp][kp][2] = + 08th * EP1 * TM1;
PNQ[jp][kp][3] = - 08th * EP1 * TM1;
PNQ[jp][kp][4] = - 08th * EM1 * TP1;
PNQ[jp][kp][5] = + 08th * EM1 * TP1;
PNQ[jp][kp][6] = + 08th * EP1 * TP1;
PNQ[jp][kp][7] = - 08th * EP1 * TP1;
PNE[ip][kp][0] = - 08th * QM1 * TM1;
PNE[ip][kp][1] = - 08th * QP1 * TM1;
PNE[ip][kp][2] = + 08th * QP1 * TM1;
PNE[ip][kp][3] = + 08th * QM1 * TM1;
PNE[ip][kp][4] = - 08th * QM1 * TP1;
PNE[ip][kp][5] = - 08th * QP1 * TP1;
PNE[ip][kp][6] = + 08th * QP1 * TP1;
PNE[ip][kp][7] = + 08th * QM1 * TP1;
PNT[ip][jp][0] = - 08th * QM1 * EM1;
PNT[ip][jp][1] = - 08th * QP1 * EM1;
PNT[ip][jp][2] = - 08th * QP1 * EP1;
PNT[ip][jp][3] = - 08th * QM1 * EP1;
PNT[ip][jp][4] = + 08th * QM1 * EM1;
PNT[ip][jp][5] = + 08th * QP1 * EM1;
PNT[ip][jp][6] = + 08th * QP1 * EP1;
PNT[ip][jp][7] = + 08th * QM1 * EP1;
    }
}

for( icel=0; icel< ICELTOT; icel++) {
    in1=CELNOD[icel][0];
    in2=CELNOD[icel][1];
    in3=CELNOD[icel][2];
    in4=CELNOD[icel][3];
    in5=CELNOD[icel][4];
    in6=CELNOD[icel][5];
    in7=CELNOD[icel][6];
    in8=CELNOD[icel][7];

```



係数行列 : MAT_ASS_MAIN (4/7)

```
/**
** JACOBIAN & INVERSE JACOBIAN
**/
```

```
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];
X3=XYZ[in3-1][0];
X4=XYZ[in4-1][0];
X5=XYZ[in5-1][0];
X6=XYZ[in6-1][0];
X7=XYZ[in7-1][0];
X8=XYZ[in8-1][0];
```

```
Y1=XYZ[in1-1][1];
Y2=XYZ[in2-1][1];
Y3=XYZ[in3-1][1];
Y4=XYZ[in4-1][1];
Y5=XYZ[in5-1][1];
Y6=XYZ[in6-1][1];
Y7=XYZ[in7-1][1];
Y8=XYZ[in8-1][1];
```

```
Z1=XYZ[in1-1][2];
Z2=XYZ[in2-1][2];
Z3=XYZ[in3-1][2];
Z4=XYZ[in4-1][2];
Z5=XYZ[in5-1][2];
Z6=XYZ[in6-1][2];
Z7=XYZ[in7-1][2];
Z8=XYZ[in8-1][2];
```

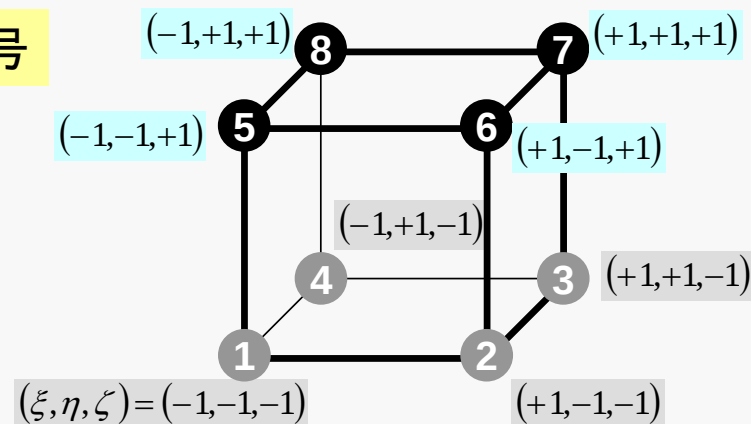
```
JACOBI (DETJ, PNQ, PNE, PNT, PNX, PNY, PNZ, X1, X2, X3, X4, X5, X6, X7, X8,
Y1, Y2, Y3, Y4, Y5, Y6, Y7, Y8, Z1, Z2, Z3, Z4, Z5, Z6, Z7, Z8);
```

8節点の節点番号

8節点のX座標

8節点のY座標

8節点のZ座標



座標値:
節点番号から1引く

JACOBI (1/4)

```

#include <stdio.h>
#include <math.h>
#include "precision.h"
#include "allocate.h"
/**
 *** JACOBI
 ***/
void JACOBI (
    KREAL DETJ[2][2][2],
    KREAL PNQ[2][2][8], KREAL PNE[2][2][8], KREAL PNT[2][2][8],
    KREAL PNQ[2][2][8], KREAL PNY[2][2][8], KREAL PNZ[2][2][8],
    KREAL X1, KREAL X2, KREAL X3, KREAL X4, KREAL X5, KREAL X6, KREAL X7, KREAL X8,
    KREAL Y1, KREAL Y2, KREAL Y3, KREAL Y4, KREAL Y5, KREAL Y6, KREAL Y7, KREAL Y8,
    KREAL Z1, KREAL Z2, KREAL Z3, KREAL Z4, KREAL Z5, KREAL Z6, KREAL Z7, KREAL Z8)
{
/**
    calculates JACOBIAN & INVERSE JACOBIAN
    dNi/dx, dNi/dy & dNi/dz
**/

    int ip, jp, kp;
    double dXdQ, dYdQ, dZdQ, dXdE, dYdE, dZdE, dXdT, dYdT, dZdT;
    double coef;
    double a11, a12, a13, a21, a22, a23, a31, a32, a33;

    for (ip=0; ip<2; ip++) {
        for (jp=0; jp<2; jp++) {
            for (kp=0; kp<2; kp++) {
                PNQ[ip][jp][kp][0]=0.0;
                PNQ[ip][jp][kp][1]=0.0;
                PNQ[ip][jp][kp][2]=0.0;
                PNQ[ip][jp][kp][3]=0.0;
                PNQ[ip][jp][kp][4]=0.0;
                PNQ[ip][jp][kp][5]=0.0;
                PNQ[ip][jp][kp][6]=0.0;
                PNQ[ip][jp][kp][7]=0.0;
            }
        }
    }
}

```

入力

$$\left[\frac{\partial N_l}{\partial \xi}, \frac{\partial N_l}{\partial \eta}, \frac{\partial N_l}{\partial \zeta} \right], (x_l, y_l, z_l) (l = 1 \sim 8)$$

出力

$$\left[\frac{\partial N_l}{\partial x}, \frac{\partial N_l}{\partial y}, \frac{\partial N_l}{\partial z} \right], \det|J|$$

各ガウス積分点[ip][jp][kp]における値

自然座標系における偏微分 (1/4)

- 偏微分の公式より以下のようなになる：

$$\frac{\partial N_i(\xi, \eta, \zeta)}{\partial \xi} = \frac{\partial N_i}{\partial x} \frac{\partial x}{\partial \xi} + \frac{\partial N_i}{\partial y} \frac{\partial y}{\partial \xi} + \frac{\partial N_i}{\partial z} \frac{\partial z}{\partial \xi}$$

$$\frac{\partial N_i(\xi, \eta, \zeta)}{\partial \eta} = \frac{\partial N_i}{\partial x} \frac{\partial x}{\partial \eta} + \frac{\partial N_i}{\partial y} \frac{\partial y}{\partial \eta} + \frac{\partial N_i}{\partial z} \frac{\partial z}{\partial \eta}$$

$$\frac{\partial N_i(\xi, \eta, \zeta)}{\partial \zeta} = \frac{\partial N_i}{\partial x} \frac{\partial x}{\partial \zeta} + \frac{\partial N_i}{\partial y} \frac{\partial y}{\partial \zeta} + \frac{\partial N_i}{\partial z} \frac{\partial z}{\partial \zeta}$$

$\left[\frac{\partial N_i}{\partial \xi}, \frac{\partial N_i}{\partial \eta}, \frac{\partial N_i}{\partial \zeta} \right]$ は定義より簡単に求められるが

$\left[\frac{\partial N_i}{\partial x}, \frac{\partial N_i}{\partial y}, \frac{\partial N_i}{\partial z} \right]$ を実際の計算で使用する

自然座標系における偏微分 (2/4)

- マトリックス表示すると：

$$\begin{Bmatrix} \frac{\partial N_i}{\partial \xi} \\ \frac{\partial N_i}{\partial \eta} \\ \frac{\partial N_i}{\partial \zeta} \end{Bmatrix} = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} & \frac{\partial z}{\partial \xi} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} & \frac{\partial z}{\partial \eta} \\ \frac{\partial x}{\partial \zeta} & \frac{\partial y}{\partial \zeta} & \frac{\partial z}{\partial \zeta} \end{bmatrix} \begin{Bmatrix} \frac{\partial N_i}{\partial x} \\ \frac{\partial N_i}{\partial y} \\ \frac{\partial N_i}{\partial z} \end{Bmatrix} = [J] \begin{Bmatrix} \frac{\partial N_i}{\partial x} \\ \frac{\partial N_i}{\partial y} \\ \frac{\partial N_i}{\partial z} \end{Bmatrix}$$

$$[J] = \begin{bmatrix} J_{11} & J_{12} & J_{13} \\ J_{21} & J_{22} & J_{23} \\ J_{31} & J_{32} & J_{33} \end{bmatrix}$$

$[J]$: ヤコビのマトリクス
(Jacobi matrix
Jacobian)

自然座標系における偏微分 (3/4)

- N_i の定義より簡単に求められる

$$J_{11} = \frac{\partial x}{\partial \xi} = \frac{\partial}{\partial \xi} \left(\sum_{i=1}^8 N_i x_i \right) = \sum_{i=1}^8 \frac{\partial N_i}{\partial \xi} x_i, \quad J_{12} = \frac{\partial y}{\partial \xi} = \frac{\partial}{\partial \xi} \left(\sum_{i=1}^8 N_i y_i \right) = \sum_{i=1}^8 \frac{\partial N_i}{\partial \xi} y_i,$$

$$J_{13} = \frac{\partial z}{\partial \xi} = \frac{\partial}{\partial \xi} \left(\sum_{i=1}^8 N_i z_i \right) = \sum_{i=1}^8 \frac{\partial N_i}{\partial \xi} z_i$$

$$J_{21} = \frac{\partial x}{\partial \eta} = \frac{\partial}{\partial \eta} \left(\sum_{i=1}^8 N_i x_i \right) = \sum_{i=1}^8 \frac{\partial N_i}{\partial \eta} x_i, \quad J_{22} = \frac{\partial y}{\partial \eta} = \frac{\partial}{\partial \eta} \left(\sum_{i=1}^8 N_i y_i \right) = \sum_{i=1}^8 \frac{\partial N_i}{\partial \eta} y_i,$$

$$J_{23} = \frac{\partial z}{\partial \eta} = \frac{\partial}{\partial \eta} \left(\sum_{i=1}^8 N_i z_i \right) = \sum_{i=1}^8 \frac{\partial N_i}{\partial \eta} z_i$$

$$J_{31} = \frac{\partial x}{\partial \zeta} = \frac{\partial}{\partial \zeta} \left(\sum_{i=1}^8 N_i x_i \right) = \sum_{i=1}^8 \frac{\partial N_i}{\partial \zeta} x_i, \quad J_{32} = \frac{\partial y}{\partial \zeta} = \frac{\partial}{\partial \zeta} \left(\sum_{i=1}^8 N_i y_i \right) = \sum_{i=1}^8 \frac{\partial N_i}{\partial \zeta} y_i,$$

$$J_{33} = \frac{\partial z}{\partial \zeta} = \frac{\partial}{\partial \zeta} \left(\sum_{i=1}^8 N_i z_i \right) = \sum_{i=1}^8 \frac{\partial N_i}{\partial \zeta} z_i$$

JACOBI (2/4)

```
PNY[ip][jp][kp][0]=0.0;
PNY[ip][jp][kp][1]=0.0;
PNY[ip][jp][kp][2]=0.0;
PNY[ip][jp][kp][3]=0.0;
PNY[ip][jp][kp][4]=0.0;
PNY[ip][jp][kp][5]=0.0;
PNY[ip][jp][kp][6]=0.0;
PNY[ip][jp][kp][7]=0.0;
```

```
PNZ[ip][jp][kp][0]=0.0;
PNZ[ip][jp][kp][1]=0.0;
PNZ[ip][jp][kp][2]=0.0;
PNZ[ip][jp][kp][3]=0.0;
PNZ[ip][jp][kp][4]=0.0;
PNZ[ip][jp][kp][5]=0.0;
PNZ[ip][jp][kp][6]=0.0;
PNZ[ip][jp][kp][7]=0.0;
```

$$[J] = \begin{bmatrix} J_{11} & J_{12} & J_{13} \\ J_{21} & J_{22} & J_{23} \\ J_{31} & J_{32} & J_{33} \end{bmatrix}$$

```
/**
```

DETERMINANT of the JACOBIAN

```
**/
```

```
dXdQ = PNQ[jp][kp][0]*X1 + PNQ[jp][kp][1]*X2
      + PNQ[jp][kp][2]*X3 + PNQ[jp][kp][3]*X4
      + PNQ[jp][kp][4]*X5 + PNQ[jp][kp][5]*X6
      + PNQ[jp][kp][6]*X7 + PNQ[jp][kp][7]*X8;
dYdQ = PNQ[jp][kp][0]*Y1 + PNQ[jp][kp][1]*Y2
      + PNQ[jp][kp][2]*Y3 + PNQ[jp][kp][3]*Y4
      + PNQ[jp][kp][4]*Y5 + PNQ[jp][kp][5]*Y6
      + PNQ[jp][kp][6]*Y7 + PNQ[jp][kp][7]*Y8;
dZdQ = PNQ[jp][kp][0]*Z1 + PNQ[jp][kp][1]*Z2
      + PNQ[jp][kp][2]*Z3 + PNQ[jp][kp][3]*Z4
      + PNQ[jp][kp][4]*Z5 + PNQ[jp][kp][5]*Z6
      + PNQ[jp][kp][6]*Z7 + PNQ[jp][kp][7]*Z8;
dXdE = PNE[ip][kp][0]*X1 + PNE[ip][kp][1]*X2
      + PNE[ip][kp][2]*X3 + PNE[ip][kp][3]*X4
      + PNE[ip][kp][4]*X5 + PNE[ip][kp][5]*X6
      + PNE[ip][kp][6]*X7 + PNE[ip][kp][7]*X8;
```

$$dXdQ = \frac{\partial x}{\partial \xi} = J_{11}$$

$$dYdQ = \frac{\partial y}{\partial \xi} = J_{12}$$

$$dZdQ = \frac{\partial z}{\partial \xi} = J_{13}$$

JACOBI (3/4)

```

dYdE = PNE[ip][kp][0]*Y1 + PNE[ip][kp][1]*Y2
        + PNE[ip][kp][2]*Y3 + PNE[ip][kp][3]*Y4
        + PNE[ip][kp][4]*Y5 + PNE[ip][kp][5]*Y6
        + PNE[ip][kp][6]*Y7 + PNE[ip][kp][7]*Y8;
dZdE = PNE[ip][kp][0]*Z1 + PNE[ip][kp][1]*Z2
        + PNE[ip][kp][2]*Z3 + PNE[ip][kp][3]*Z4
        + PNE[ip][kp][4]*Z5 + PNE[ip][kp][5]*Z6
        + PNE[ip][kp][6]*Z7 + PNE[ip][kp][7]*Z8;
dXdT = PNT[ip][jp][0]*X1 + PNT[ip][jp][1]*X2
        + PNT[ip][jp][2]*X3 + PNT[ip][jp][3]*X4
        + PNT[ip][jp][4]*X5 + PNT[ip][jp][5]*X6
        + PNT[ip][jp][6]*X7 + PNT[ip][jp][7]*X8;
dYdT = PNT[ip][jp][0]*Y1 + PNT[ip][jp][1]*Y2
        + PNT[ip][jp][2]*Y3 + PNT[ip][jp][3]*Y4
        + PNT[ip][jp][4]*Y5 + PNT[ip][jp][5]*Y6
        + PNT[ip][jp][6]*Y7 + PNT[ip][jp][7]*Y8;
dZdT = PNT[ip][jp][0]*Z1 + PNT[ip][jp][1]*Z2
        + PNT[ip][jp][2]*Z3 + PNT[ip][jp][3]*Z4
        + PNT[ip][jp][4]*Z5 + PNT[ip][jp][5]*Z6
        + PNT[ip][jp][6]*Z7 + PNT[ip][jp][7]*Z8;
DETJ[ip][jp][kp]= dXdQ*(dYdE*dZdT-dZdE*dYdT) +
                   dYdQ*(dZdE*dXdT-dXdE*dZdT) +
                   dZdQ*(dXdE*dYdT-dYdE*dXdT);

```

```

/**
INVERSE JACOBIAN
**/

```

```

coef=1.0 / DETJ[ip][jp][kp];

a11= coef * ( dYdE*dZdT - dZdE*dYdT );
a12= coef * ( dZdQ*dYdT - dYdQ*dZdT );
a13= coef * ( dYdQ*dZdE - dZdQ*dYdE );

a21= coef * ( dZdE*dXdT - dXdE*dZdT );
a22= coef * ( dXdQ*dZdT - dZdQ*dXdT );
a23= coef * ( dZdQ*dXdE - dXdQ*dZdE );

a31= coef * ( dXdE*dYdT - dYdE*dXdT );
a32= coef * ( dYdQ*dXdT - dXdQ*dYdT );
a33= coef * ( dXdQ*dYdE - dYdQ*dXdE );

DETJ[ip][jp][kp]=fabs(DETJ[ip][jp][kp]);

```

$$[J] = \begin{bmatrix} J_{11} & J_{12} & J_{13} \\ J_{21} & J_{22} & J_{23} \\ J_{31} & J_{32} & J_{33} \end{bmatrix}$$

自然座標系における偏微分 (4/4)

- 従って下記のように偏微分を計算できる
 - ヤコビアン (3×3行列) の逆行列を求める

$$\begin{Bmatrix} \frac{\partial N_i}{\partial x} \\ \frac{\partial N_i}{\partial y} \\ \frac{\partial N_i}{\partial z} \end{Bmatrix} = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} & \frac{\partial z}{\partial \xi} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} & \frac{\partial z}{\partial \eta} \\ \frac{\partial x}{\partial \zeta} & \frac{\partial y}{\partial \zeta} & \frac{\partial z}{\partial \zeta} \end{bmatrix}^{-1} \begin{Bmatrix} \frac{\partial N_i}{\partial \xi} \\ \frac{\partial N_i}{\partial \eta} \\ \frac{\partial N_i}{\partial \zeta} \end{Bmatrix} = [J]^{-1} \begin{Bmatrix} \frac{\partial N_i}{\partial \xi} \\ \frac{\partial N_i}{\partial \eta} \\ \frac{\partial N_i}{\partial \zeta} \end{Bmatrix}$$

JACOBI (3/4)

```

dYdE = PNE[ip][kp][0]*Y1 + PNE[ip][kp][1]*Y2
      + PNE[ip][kp][2]*Y3 + PNE[ip][kp][3]*Y4
      + PNE[ip][kp][4]*Y5 + PNE[ip][kp][5]*Y6
      + PNE[ip][kp][6]*Y7 + PNE[ip][kp][7]*Y8;
dZdE = PNE[ip][kp][0]*Z1 + PNE[ip][kp][1]*Z2
      + PNE[ip][kp][2]*Z3 + PNE[ip][kp][3]*Z4
      + PNE[ip][kp][4]*Z5 + PNE[ip][kp][5]*Z6
      + PNE[ip][kp][6]*Z7 + PNE[ip][kp][7]*Z8;
dXdT = PNT[ip][jp][0]*X1 + PNT[ip][jp][1]*X2
      + PNT[ip][jp][2]*X3 + PNT[ip][jp][3]*X4
      + PNT[ip][jp][4]*X5 + PNT[ip][jp][5]*X6
      + PNT[ip][jp][6]*X7 + PNT[ip][jp][7]*X8;
dYdT = PNT[ip][jp][0]*Y1 + PNT[ip][jp][1]*Y2
      + PNT[ip][jp][2]*Y3 + PNT[ip][jp][3]*Y4
      + PNT[ip][jp][4]*Y5 + PNT[ip][jp][5]*Y6
      + PNT[ip][jp][6]*Y7 + PNT[ip][jp][7]*Y8;
dZdT = PNT[ip][jp][0]*Z1 + PNT[ip][jp][1]*Z2
      + PNT[ip][jp][2]*Z3 + PNT[ip][jp][3]*Z4
      + PNT[ip][jp][4]*Z5 + PNT[ip][jp][5]*Z6
      + PNT[ip][jp][6]*Z7 + PNT[ip][jp][7]*Z8;
DETJ[ip][jp][kp]= dXdQ*(dYdE*dZdT-dZdE*dYdT) +
                  dYdQ*(dZdE*dXdT-dXdE*dZdT) +
                  dZdQ*(dXdE*dYdT-dYdE*dXdT);

```

```

/**
INVERSE JACOBIAN
**/

```

```
coef=1.0 / DETJ[ip][jp][kp];
```

```

a11= coef * ( dYdE*dZdT - dZdE*dYdT );
a12= coef * ( dZdQ*dYdT - dYdQ*dZdT );
a13= coef * ( dYdQ*dZdE - dZdQ*dYdE );

```

```

a21= coef * ( dZdE*dXdT - dXdE*dZdT );
a22= coef * ( dXdQ*dZdT - dZdQ*dXdT );
a23= coef * ( dZdQ*dXdE - dXdQ*dZdE );

```

```

a31= coef * ( dXdE*dYdT - dYdE*dXdT );
a32= coef * ( dYdQ*dXdT - dXdQ*dYdT );
a33= coef * ( dXdQ*dYdE - dYdQ*dXdE );

```

```
DETJ[ip][jp][kp]=fabs(DETJ[ip][jp][kp]);
```

$$[J]^{-1} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$$

JACOBI (4/4)

```

/**
  set the dNi/dX, dNi/dY & dNi/dZ components
**/
PNX[ip][jp][kp][0]=a11*PNQ[jp][kp][0]+a12*PNE[ip][kp][0]+a13*PNT[ip][jp][0];
PNX[ip][jp][kp][1]=a11*PNQ[jp][kp][1]+a12*PNE[ip][kp][1]+a13*PNT[ip][jp][1];
PNX[ip][jp][kp][2]=a11*PNQ[jp][kp][2]+a12*PNE[ip][kp][2]+a13*PNT[ip][jp][2];
PNX[ip][jp][kp][3]=a11*PNQ[jp][kp][3]+a12*PNE[ip][kp][3]+a13*PNT[ip][jp][3];
PNX[ip][jp][kp][4]=a11*PNQ[jp][kp][4]+a12*PNE[ip][kp][4]+a13*PNT[ip][jp][4];
PNX[ip][jp][kp][5]=a11*PNQ[jp][kp][5]+a12*PNE[ip][kp][5]+a13*PNT[ip][jp][5];
PNX[ip][jp][kp][6]=a11*PNQ[jp][kp][6]+a12*PNE[ip][kp][6]+a13*PNT[ip][jp][6];
PNX[ip][jp][kp][7]=a11*PNQ[jp][kp][7]+a12*PNE[ip][kp][7]+a13*PNT[ip][jp][7];
PNY[ip][jp][kp][0]=a21*PNQ[jp][kp][0]+a22*PNE[ip][kp][0]+a23*PNT[ip][jp][0];
PNY[ip][jp][kp][1]=a21*PNQ[jp][kp][1]+a22*PNE[ip][kp][1]+a23*PNT[ip][jp][1];
PNY[ip][jp][kp][2]=a21*PNQ[jp][kp][2]+a22*PNE[ip][kp][2]+a23*PNT[ip][jp][2];
PNY[ip][jp][kp][3]=a21*PNQ[jp][kp][3]+a22*PNE[ip][kp][3]+a23*PNT[ip][jp][3];
PNY[ip][jp][kp][4]=a21*PNQ[jp][kp][4]+a22*PNE[ip][kp][4]+a23*PNT[ip][jp][4];
PNY[ip][jp][kp][5]=a21*PNQ[jp][kp][5]+a22*PNE[ip][kp][5]+a23*PNT[ip][jp][5];
PNY[ip][jp][kp][6]=a21*PNQ[jp][kp][6]+a22*PNE[ip][kp][6]+a23*PNT[ip][jp][6];
PNY[ip][jp][kp][7]=a21*PNQ[jp][kp][7]+a22*PNE[ip][kp][7]+a23*PNT[ip][jp][7];
PNZ[ip][jp][kp][0]=a31*PNQ[jp][kp][0]+a32*PNE[ip][kp][0]+a33*PNT[ip][jp][0];
PNZ[ip][jp][kp][1]=a31*PNQ[jp][kp][1]+a32*PNE[ip][kp][1]+a33*PNT[ip][jp][1];
PNZ[ip][jp][kp][2]=a31*PNQ[jp][kp][2]+a32*PNE[ip][kp][2]+a33*PNT[ip][jp][2];
PNZ[ip][jp][kp][3]=a31*PNQ[jp][kp][3]+a32*PNE[ip][kp][3]+a33*PNT[ip][jp][3];
PNZ[ip][jp][kp][4]=a31*PNQ[jp][kp][4]+a32*PNE[ip][kp][4]+a33*PNT[ip][jp][4];
PNZ[ip][jp][kp][5]=a31*PNQ[jp][kp][5]+a32*PNE[ip][kp][5]+a33*PNT[ip][jp][5];
PNZ[ip][jp][kp][6]=a31*PNQ[jp][kp][6]+a32*PNE[ip][kp][6]+a33*PNT[ip][jp][6];
PNZ[ip][jp][kp][7]=a31*PNQ[jp][kp][7]+a32*PNE[ip][kp][7]+a33*PNT[ip][jp][7];
    }
}
}

```

$$\begin{Bmatrix} \frac{\partial N_i}{\partial x} \\ \frac{\partial N_i}{\partial y} \\ \frac{\partial N_i}{\partial z} \end{Bmatrix} = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} & \frac{\partial z}{\partial \xi} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} & \frac{\partial z}{\partial \eta} \\ \frac{\partial x}{\partial \zeta} & \frac{\partial y}{\partial \zeta} & \frac{\partial z}{\partial \zeta} \end{bmatrix}^{-1} \begin{Bmatrix} \frac{\partial N_i}{\partial \xi} \\ \frac{\partial N_i}{\partial \eta} \\ \frac{\partial N_i}{\partial \zeta} \end{Bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \begin{Bmatrix} \frac{\partial N_i}{\partial \xi} \\ \frac{\partial N_i}{\partial \eta} \\ \frac{\partial N_i}{\partial \zeta} \end{Bmatrix}$$

係数行列 : MAT_ASS_MAIN (5/7)

```

/**
    CONSTRUCT the GLOBAL MATRIX
**/
    for (ie=0; ie<8; ie++) {
        ip=nodLOCAL[ie];

        for (je=0; je<8; je++) {
            jp=nodLOCAL[je];

            kk=0;
            if ( jp > ip ) {
                iiS=indexU[ip-1]+1;
                iiE=indexU[ip ];
                for ( k=iiS; k<=iiE; k++) {
                    if ( itemU[k-1] == jp ) {
                        kk=k;
                        break;
                    }
                }
            }

            if ( jp < ip ) {
                iiS=indexL[ip-1]+1;
                iiE=indexL[ip ];
                for ( k=iiS; k<=iiE; k++) {
                    if ( itemL[k-1] == jp ) {
                        kk=k;
                        break;
                    }
                }
            }

            PNXi= 0. e0;
            PNYi= 0. e0;
            PNZi= 0. e0;
            PNXj= 0. e0;
            PNYj= 0. e0;
            PNZj= 0. e0;

            VOL= 0. e0;

```

全体行列の非対角ブロック

$$A_{ip, jp}$$

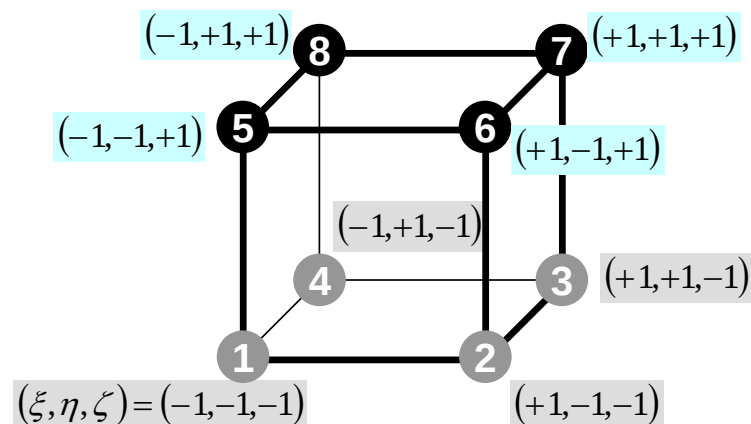
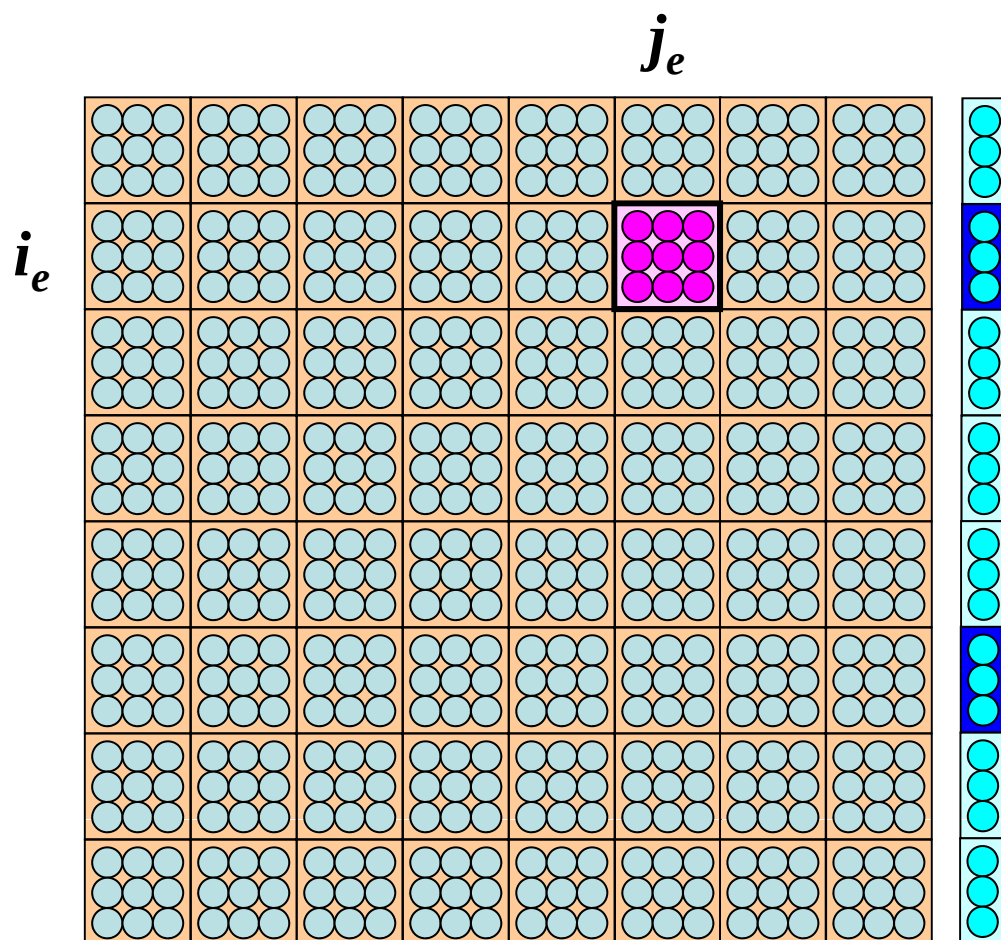
kk: itemL, itemUにおけるアドレス

ip= nodLOCAL[ie]
jp= nodLOCAL[je]

1から始まる節点番号

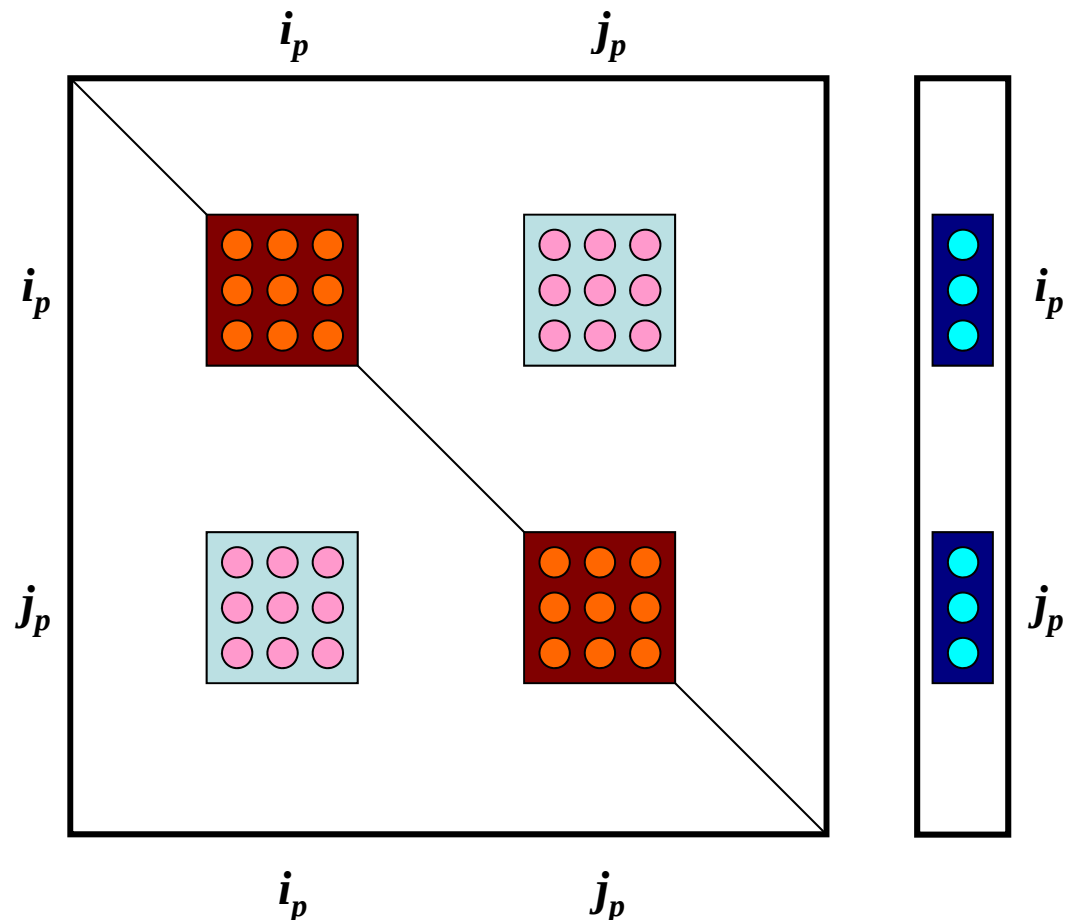
要素マトリクス：24×24行列

各節点上の (u,v,w) 成分が物理的にも強くカップルしているので3自由度をまとめて扱う：8×8行列



$$\begin{bmatrix} a_{i_e j_e 11} & a_{i_e j_e 12} & a_{i_e j_e 13} \\ a_{i_e j_e 21} & a_{i_e j_e 22} & a_{i_e j_e 23} \\ a_{i_e j_e 31} & a_{i_e j_e 32} & a_{i_e j_e 33} \end{bmatrix} (i_e, j_e = 1 \dots 8)$$

全体マトリクス



係数行列 : MAT_ASS_MAIN (5/7)

```

/**
    CONSTRUCT the GLOBAL MATRIX
**/
    for (ie=0; ie<8; ie++) {
        ip=nodLOCAL[ie];

        for (je=0; je<8; je++) {
            jp=nodLOCAL[je];

            kk=0;
            if ( jp > ip ) {
                iiS=indexU[ip-1]+1;
                iiE=indexU[ip];
                for ( k=iiS; k<=iiE; k++) {
                    if ( itemU[k-1] == jp ) {
                        kk=k;
                        break;
                    }
                }

                if ( jp < ip ) {
                    iiS=indexL[ip-1]+1;
                    iiE=indexL[ip];
                    for ( k=iiS; k<=iiE; k++) {
                        if ( itemL[k-1] == jp ) {
                            kk=k;
                            break;
                        }
                    }
                }

                PNXi= 0. e0;
                PNYi= 0. e0;
                PNZi= 0. e0;
                PNXj= 0. e0;
                PNYj= 0. e0;
                PNZj= 0. e0;

                VOL= 0. e0;

```

要素マトリクス ($i_e \sim j_e$)
全体マトリクス ($i_p \sim j_p$) の関係

kk: itemU, itemLにおけるアドレス
に1を足した値(1から始まる)

系数行列：MAT_ASS_MAIN (6/7)

```

a11= 0.0e0; a12= 0.0e0; a13= 0.0e0;
a21= 0.0e0; a22= 0.0e0; a23= 0.0e0;
a31= 0.0e0; a32= 0.0e0; a33= 0.0e0;

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];

            VOL+=coef;
            PNXi= PNX[ipn][jpn][kpn][ie];
            PNYi= PNY[ipn][jpn][kpn][ie];
            PNZi= PNZ[ipn][jpn][kpn][ie];

            PNXj= PNX[ipn][jpn][kpn][je];
            PNYj= PNY[ipn][jpn][kpn][je];
            PNZj= PNZ[ipn][jpn][kpn][je];

            a11+= (valX*PNXi*PNXj+valB*(PNYi*PNYj+PNZi*PNZj))*coef;
            a22+= (valX*PNYi*PNYj+valB*(PNZi*PNZj+PNXi*PNXj))*coef;
            a33+= (valX*PNZi*PNZj+valB*(PNXi*PNXj+PNYi*PNYj))*coef;

            a12+= (valA*PNXi*PNYj + valB*PNXj*PNYi)*coef;
            a13+= (valA*PNXi*PNZj + valB*PNXj*PNZi)*coef;
            a21+= (valA*PNYi*PNXj + valB*PNYj*PNXi)*coef;
            a23+= (valA*PNYi*PNZj + valB*PNYj*PNZi)*coef;
            a31+= (valA*PNZi*PNXj + valB*PNZj*PNXi)*coef;
            a32+= (valA*PNZi*PNYj + valB*PNZj*PNYi)*coef;

        }
    }
}

```

系数行列：MAT_ASS_MAIN (6/7)

```
a11= 0.0e0; a12= 0.0e0; a13= 0.0e0;
a21= 0.0e0; a22= 0.0e0; a23= 0.0e0;
a31= 0.0e0; a32= 0.0e0; a33= 0.0e0;
```

$$-\int_V \left\{ D \frac{\partial N_i}{\partial x} \frac{\partial N_j}{\partial x} + b \left(\frac{\partial N_i}{\partial y} \frac{\partial N_j}{\partial y} + \frac{\partial N_i}{\partial z} \frac{\partial N_j}{\partial z} \right) \right\} dV =$$

$$-\iiint \left\{ D \frac{\partial N_i}{\partial x} \frac{\partial N_j}{\partial x} + b \left(\frac{\partial N_i}{\partial y} \frac{\partial N_j}{\partial y} + \frac{\partial N_i}{\partial z} \frac{\partial N_j}{\partial z} \right) \right\} dx dy dz =$$

$$-\int_{-1}^{+1} \int_{-1}^{+1} \int_{-1}^{+1} \left\{ D \frac{\partial N_i}{\partial x} \frac{\partial N_j}{\partial x} + b \left(\frac{\partial N_i}{\partial y} \frac{\partial N_j}{\partial y} + \frac{\partial N_i}{\partial z} \frac{\partial N_j}{\partial z} \right) \right\} \det|J| d\xi d\eta d\zeta$$

$$\text{coef} = W_i \cdot W_j \cdot W_k \cdot \det|J(\xi_i, \eta_j, \zeta_k)|$$

```
pnj [kpn]) * WEI [ipn] * WEI [jpn] * WEI [kpn];
```

```
] [ie];
] [ie];
] [ie];
```

```
] [je];
] [je];
] [je];
```

$f(\xi, \eta, \zeta)$

$$I = \int_{-1}^{+1} \int_{-1}^{+1} \int_{-1}^{+1} f(\xi, \eta, \zeta) d\xi d\eta d\zeta$$

$$= \sum_{i=1}^L \sum_{j=1}^M \sum_{k=1}^N [W_i \cdot W_j \cdot W_k \cdot f(\xi_i, \eta_j, \zeta_k)]$$

```
a11+= (valX*PNXi*PNXj+valB*(PNYi*PNYj+PNZi*PNZj))*coef;
a22+= (valX*PNYi*PNYj+valB*(PNZi*PNZj+PNXi*PNXj))*coef;
a33+= (valX*PNZi*PNZj+valB*(PNXi*PNXj+PNYi*PNYj))*coef;
```

```
a12+= (valA*PNXi*PNYj + valB*PNXj*PNYi)*coef;
a13+= (valA*PNXi*PNZj + valB*PNXj*PNZi)*coef;
a21+= (valA*PNYi*PNXj + valB*PNYj*PNXi)*coef;
a23+= (valA*PNYi*PNZj + valB*PNYj*PNZi)*coef;
a31+= (valA*PNZi*PNXj + valB*PNZj*PNXi)*coef;
a32+= (valA*PNZi*PNYj + valB*PNZj*PNYi)*coef;
```

系数行列：MAT_ASS_MAIN (6/7)

```

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];
      PNXi= PNX[ipn][jpn][kpn][ie];
      PNYi= PNY[ipn][jpn][kpn][ie];
      PNZi= PNZ[ipn][jpn][kpn][ie];

      PNXj= PNX[ipn][jpn][kpn][je];
      PNYj= PNY[ipn][jpn][kpn][je];
      PNZj= PNZ[ipn][jpn][kpn][je];

      a11+= (valX*PNXi*PNXj+valB*(PNYi*PNYj+PNZi*PNZj))*coef;
      a22+= (valX*PNYi*PNYj+valB*(PNZi*PNZj+PNXi*PNXj))*coef;
      a33+= (valX*PNZi*PNZj+valB*(PNXi*PNXj+PNYi*PNYj))*coef;

      a12+= (valA*PNXi*PNYj + valB*PNXj*PNYi)*coef;
      a13+= (valA*PNXi*PNZj + valB*PNXj*PNZi)*coef;
      a21+= (valA*PNYi*PNXj + valB*PNYj*PNXi)*coef;
      a23+= (valA*PNYi*PNZj + valB*PNYj*PNZi)*coef;
      a31+= (valA*PNZi*PNXj + valB*PNZj*PNXi)*coef;
      a32+= (valA*PNZi*PNYj + valB*PNZj*PNYi)*coef;

    }
  }
}

```

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

X-方向のつりあい式 (3/3)

$$D = \frac{(1-\nu)E}{(1+\nu)(1-2\nu)}, \quad a = \frac{\nu E}{(1+\nu)(1-2\nu)}, \quad b = \frac{E}{2(1+\nu)} \quad \text{とすると}$$

$$\sigma_x = D[N_{,x}]\{U\} + a[N_{,y}]\{V\} + a[N_{,z}]\{W\}$$

$$\tau_{xy} = b[N_{,y}]\{U\} + b[N_{,x}]\{V\}$$

$$\tau_{zx} = b[N_{,z}]\{U\} + b[N_{,x}]\{W\}$$

$$\begin{aligned} (*) = & - \int_V \left\{ D[N_{,x}]^T [N_{,x}] + b([N_{,y}]^T [N_{,y}] + [N_{,z}]^T [N_{,z}]) \right\} dV \{U\} \\ & - \int_V \left\{ a[N_{,x}]^T [N_{,y}] + b([N_{,y}]^T [N_{,x}]) \right\} dV \{V\} - \int_V \left\{ a[N_{,x}]^T [N_{,z}] + b[N_{,z}]^T [N_{,x}] \right\} dV \{W\} \\ & + \int_V [N]^T \{X\} dV = 0 \end{aligned}$$

Y-方向のつりあい式

$$\begin{aligned}
 & - \int_V \left\{ D [N_{,y}]^T [N_{,y}] + b \left([N_{,z}]^T [N_{,z}] + [N_{,x}]^T [N_{,x}] \right) \right\} dV \{V\} \\
 & - \int_V \left\{ a [N_{,y}]^T [N_{,x}] + b \left([N_{,x}]^T [N_{,y}] \right) \right\} dV \{U\} - \int_V \left\{ a [N_{,y}]^T [N_{,z}] + b [N_{,z}]^T [N_{,y}] \right\} dV \{W\} \\
 & + \int_V [N]^T \{Y\} dV = 0
 \end{aligned}$$

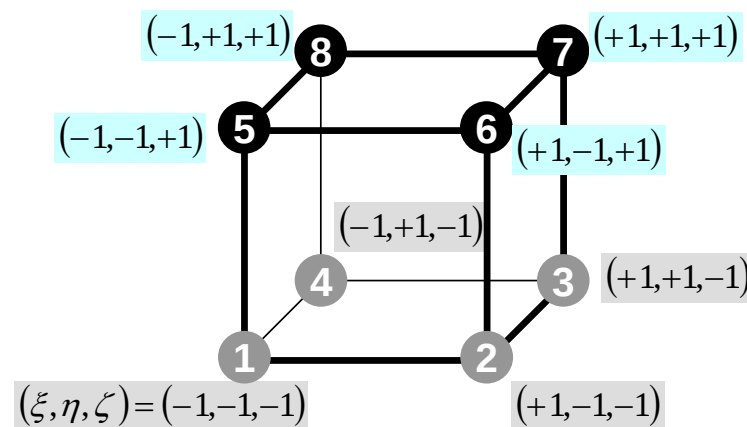
Z-方向のつりあい式

$$\begin{aligned}
 & - \int_V \left\{ D [N_{,z}]^T [N_{,z}] + b \left([N_{,x}]^T [N_{,x}] + [N_{,y}]^T [N_{,y}] \right) \right\} dV \{W\} \\
 & - \int_V \left\{ a [N_{,z}]^T [N_{,x}] + b \left([N_{,x}]^T [N_{,z}] \right) \right\} dV \{U\} - \int_V \left\{ a [N_{,z}]^T [N_{,y}] + b [N_{,y}]^T [N_{,z}] \right\} dV \{V\} \\
 & + \int_V [N]^T \{Z\} dV = 0
 \end{aligned}$$

要素マトリクス： i - j 成分, X 方向 (1/3)

$$\begin{bmatrix} a_{ij11} & a_{ij12} & a_{ij13} \\ a_{ij21} & a_{ij22} & a_{ij23} \\ a_{ij31} & a_{ij32} & a_{ij33} \end{bmatrix} \begin{bmatrix} u \\ v \\ w \end{bmatrix} \quad (i, j = 1 \dots 8)$$

$$\text{valX}=D; \text{valA}=a; \text{valB}=b$$



$$a_{ij11} = - \int_V \left\{ \text{valX} \cdot N_{i,x} \cdot N_{j,x} + \text{valB} \cdot (N_{i,y} \cdot N_{j,y} + N_{i,z} \cdot N_{j,z}) \right\} dV$$

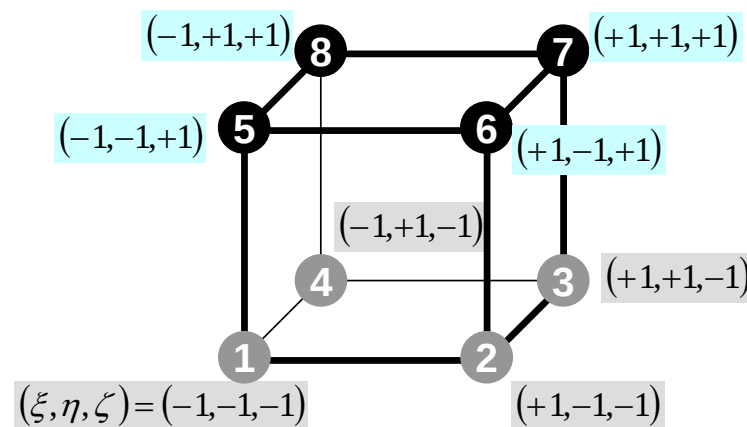
$$a_{ij12} = - \int_V \left\{ \text{valA} \cdot N_{i,x} \cdot N_{j,y} + \text{valB} \cdot N_{i,y} \cdot N_{j,x} \right\} dV$$

$$a_{ij13} = - \int_V \left\{ \text{valA} \cdot N_{i,x} \cdot N_{j,z} + \text{valB} \cdot N_{i,z} \cdot N_{j,x} \right\} dV$$

要素マトリクス： i - j 成分, X 方向 (1/3)

$$\begin{bmatrix} a_{ij11} & a_{ij12} & a_{ij13} \\ a_{ij21} & a_{ij22} & a_{ij23} \\ a_{ij31} & a_{ij32} & a_{ij33} \end{bmatrix} \begin{bmatrix} u \\ v \\ w \end{bmatrix} \quad (i, j = 1 \dots 8)$$

valX=D; valA=a; valB=b

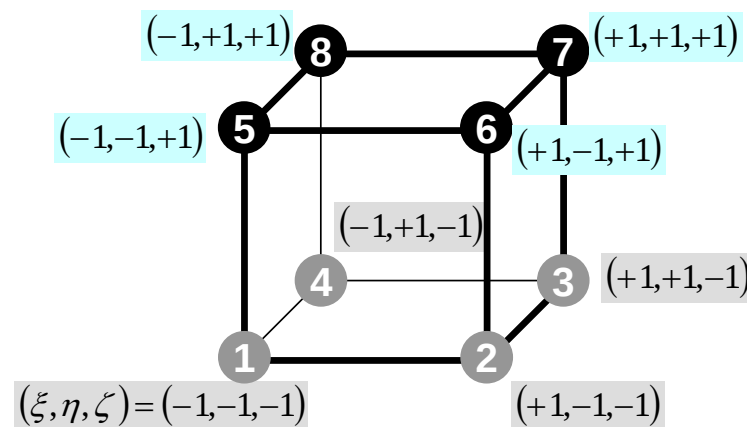


$$\begin{aligned} & - \int_V \left\{ D [N_{,x}]^T [N_{,x}] + b \left([N_{,y}]^T [N_{,y}] + [N_{,z}]^T [N_{,z}] \right) \right\} dV \{U\} \\ & - \int_V \left\{ a [N_{,x}]^T [N_{,y}] + b \left([N_{,y}]^T [N_{,x}] \right) \right\} dV \{V\} \\ & - \int_V \left\{ a [N_{,x}]^T [N_{,z}] + b [N_{,z}]^T [N_{,x}] \right\} dV \{W\} \end{aligned}$$

要素マトリクス： i - j 成分, Y 方向 (2/3)

$$\begin{bmatrix} a_{ij11} & a_{ij12} & a_{ij13} \\ a_{ij21} & a_{ij22} & a_{ij23} \\ a_{ij31} & a_{ij32} & a_{ij33} \end{bmatrix} \begin{bmatrix} u \\ v \\ w \end{bmatrix} \quad (i, j = 1 \dots 8)$$

$valX=D$; $valA=a$; $valB=b$



$$a_{ij21} = - \int_V \{ valA \cdot N_{i,y} \cdot N_{j,x} + valB \cdot N_{i,x} \cdot N_{j,y} \} dV$$

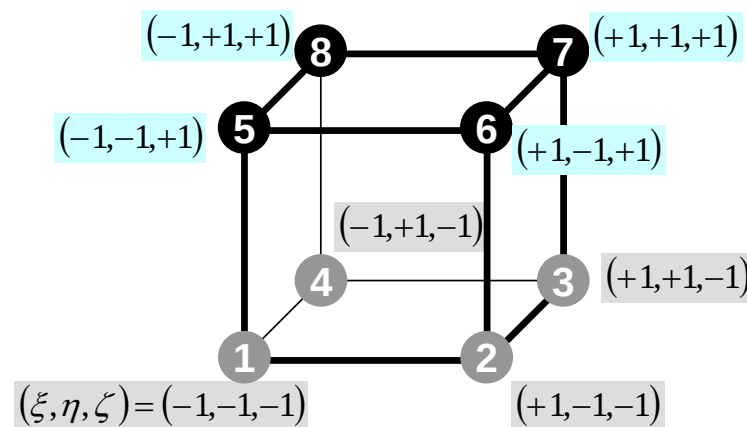
$$a_{ij22} = - \int_V \{ valX \cdot N_{i,y} \cdot N_{j,y} + valB \cdot (N_{i,z} \cdot N_{j,z} + N_{i,x} \cdot N_{j,x}) \} dV$$

$$a_{ij23} = - \int_V \{ valA \cdot N_{i,y} \cdot N_{j,z} + valB \cdot N_{i,z} \cdot N_{j,y} \} dV$$

要素マトリクス： i - j 成分, Y 方向 (2/3)

$$\begin{bmatrix} a_{ij11} & a_{ij12} & a_{ij13} \\ a_{ij21} & a_{ij22} & a_{ij23} \\ a_{ij31} & a_{ij32} & a_{ij33} \end{bmatrix} \begin{bmatrix} u \\ v \\ w \end{bmatrix} \quad (i, j = 1 \dots 8)$$

valX=D; valA=a; valB=b

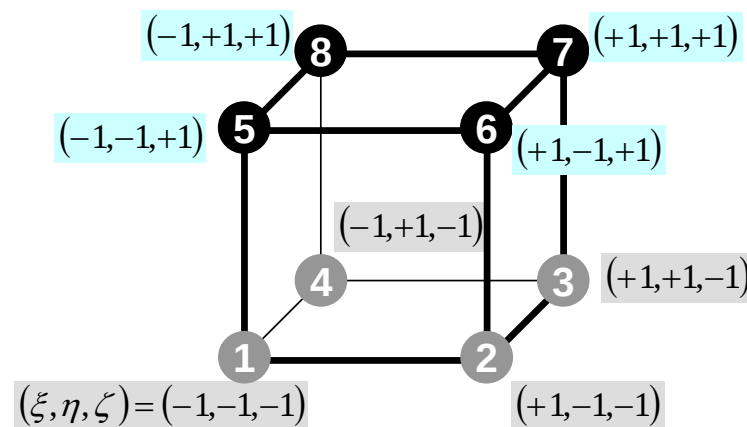


$$\begin{aligned} & - \int_V \left\{ a [N_{,y}]^T [N_{,x}] + b ([N_{,x}]^T [N_{,y}]) \right\} dV \{U\} \\ & - \int_V \left\{ D [N_{,y}]^T [N_{,y}] + b ([N_{,z}]^T [N_{,z}] + [N_{,x}]^T [N_{,x}]) \right\} dV \{V\} \\ & - \int_V \left\{ a [N_{,y}]^T [N_{,z}] + b [N_{,z}]^T [N_{,y}] \right\} dV \{W\} \end{aligned}$$

要素マトリクス： i - j 成分, Z 方向 (3/3)

$$\begin{bmatrix} a_{ij11} & a_{ij12} & a_{ij13} \\ a_{ij21} & a_{ij22} & a_{ij23} \\ a_{ij31} & a_{ij32} & a_{ij33} \end{bmatrix} \begin{bmatrix} u \\ v \\ w \end{bmatrix} \quad (i, j = 1 \dots 8)$$

$valX=D$; $valA=a$; $valB=b$



$$a_{ij31} = - \int_V \{ valA \cdot N_{i,z} \cdot N_{j,x} + valB \cdot N_{i,x} \cdot N_{j,z} \} dV$$

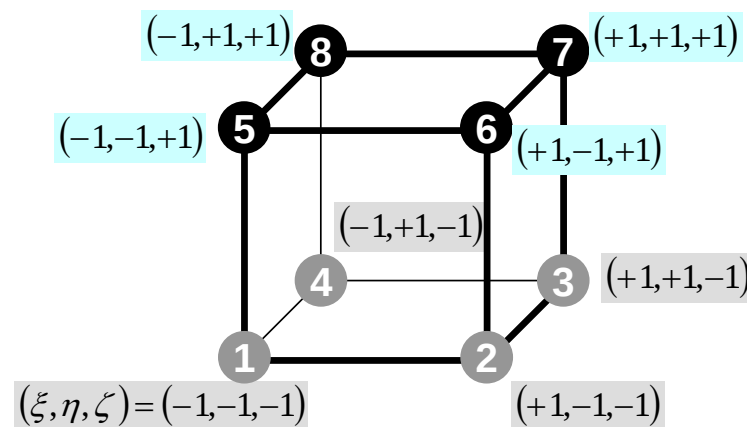
$$a_{ij32} = - \int_V \{ valA \cdot N_{i,z} \cdot N_{j,y} + valB \cdot N_{i,y} \cdot N_{j,z} \} dV$$

$$a_{ij33} = - \int_V \{ valD \cdot N_{i,z} \cdot N_{j,z} + valB \cdot (N_{i,x} \cdot N_{j,x} + N_{i,y} \cdot N_{j,y}) \} dV$$

要素マトリクス： i - j 成分, z 方向 (3/3)

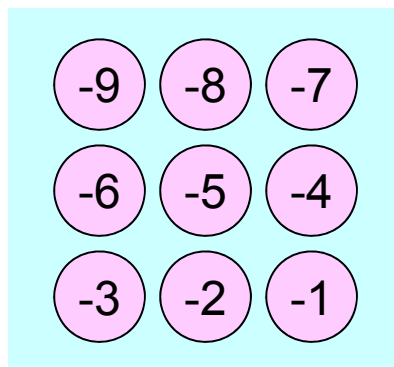
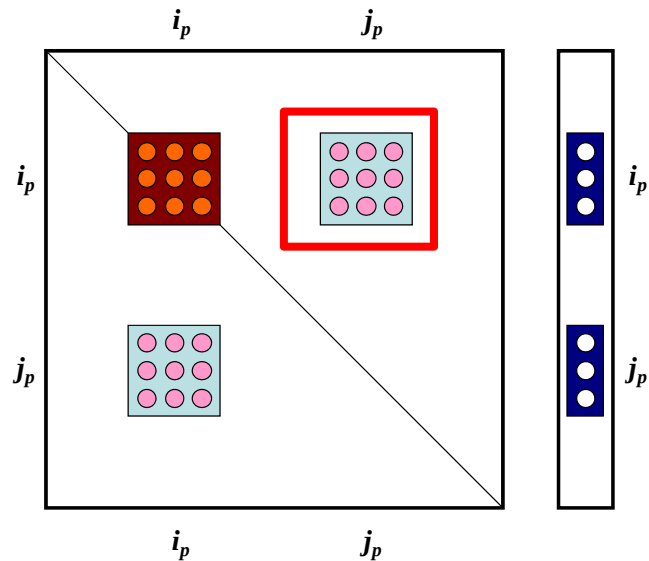
$$\begin{bmatrix} a_{ij11} & a_{ij12} & a_{ij13} \\ a_{ij21} & a_{ij22} & a_{ij23} \\ a_{ij31} & a_{ij32} & a_{ij33} \end{bmatrix} \begin{bmatrix} u \\ v \\ w \end{bmatrix} \quad (i, j = 1 \dots 8)$$

valX=D; valA=a; valB=b



$$\begin{aligned} & - \int_V \left\{ a [N_{,z}]^T [N_{,x}] + b ([N_{,x}]^T [N_{,z}]) \right\} dV \{U\} \\ & - \int_V \left\{ a [N_{,z}]^T [N_{,y}] + b [N_{,y}]^T [N_{,z}] \right\} dV \{V\} \\ & - \int_V \left\{ D [N_{,z}]^T [N_{,z}] + b ([N_{,x}]^T [N_{,x}] + [N_{,y}]^T [N_{,y}]) \right\} dV \{W\} \end{aligned}$$

係数行列 : MAT_ASS_MAIN (7/7)



```

for (icel=0; icel<ICELTOT; icel++) {
    for (ie=0; ie<8; ie++) {
        ip=nodLOCAL[ie];
        for (je=0; je<8; je++) {
            jp=nodLOCAL[je];
            ...
            if (jp > ip) {
                AU[9*kk-9] += a11;
                AU[9*kk-8] += a12;
                AU[9*kk-7] += a13;
                AU[9*kk-6] += a21;
                AU[9*kk-5] += a22;
                AU[9*kk-4] += a23;
                AU[9*kk-3] += a31;
                AU[9*kk-2] += a32;
                AU[9*kk-1] += a33;
            }
            if (jp < ip) {
                AL[9*kk-9] += a11;
                AL[9*kk-8] += a12;
                AL[9*kk-7] += a13;
                AL[9*kk-6] += a21;
                AL[9*kk-5] += a22;
                AL[9*kk-4] += a23;
                AL[9*kk-3] += a31;
                AL[9*kk-2] += a32;
                AL[9*kk-1] += a33;
            }
        }
    }
}

```

要素マトリクス ($i_e \sim j_e$)
 全体マトリクス ($i_p \sim j_p$) の関係

kk: itemU, itemLにおけるアドレス
 に1を足した値(1から始まる)

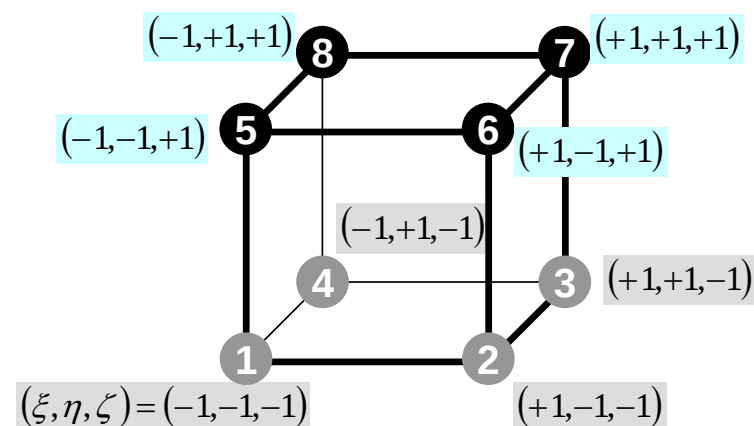
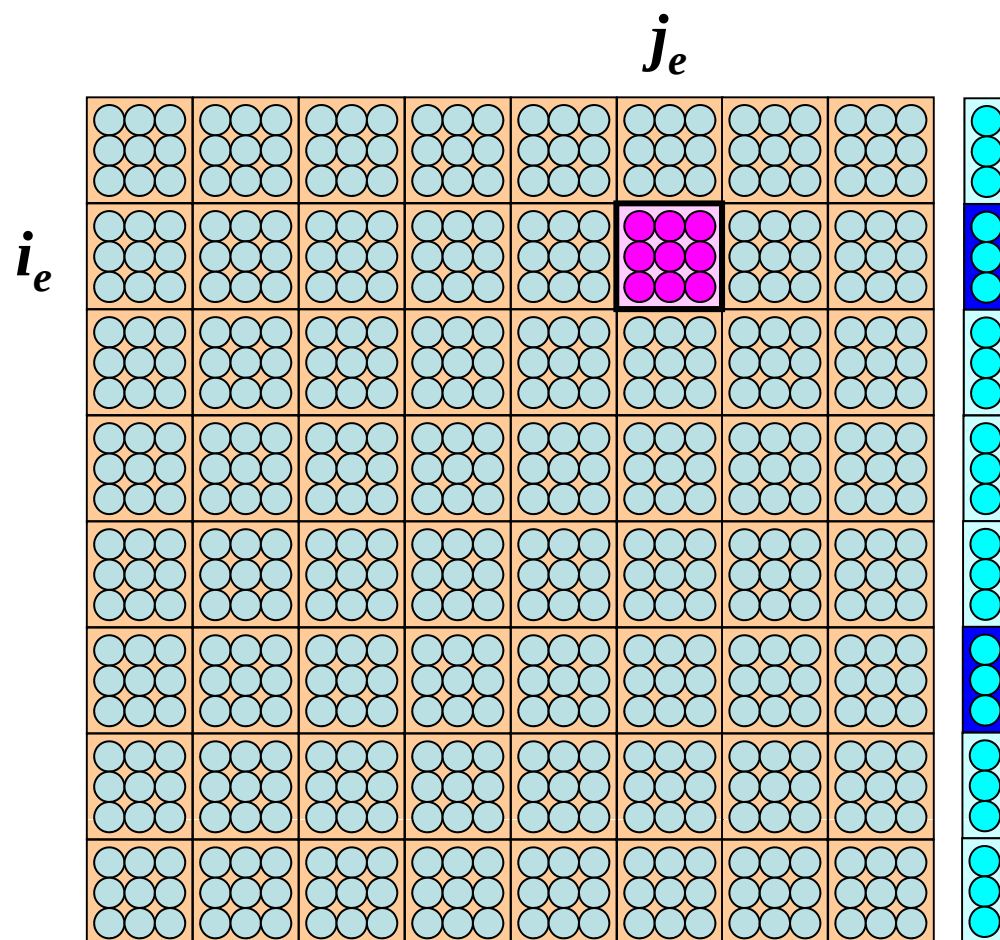
```

...
}
}

```

要素マトリクス：24×24行列

各節点上の (u,v,w) 成分が物理的にも強くカップルしているので3自由度をまとめて扱う：8×8行列



$$\begin{bmatrix} a_{i_e j_e 11} & a_{i_e j_e 12} & a_{i_e j_e 13} \\ a_{i_e j_e 21} & a_{i_e j_e 22} & a_{i_e j_e 23} \\ a_{i_e j_e 31} & a_{i_e j_e 32} & a_{i_e j_e 33} \end{bmatrix} (i_e, j_e = 1 \dots 8)$$

系数行列：MAT_ASS_MAIN (7/7)

```

for (icel=0; icel<ICELTOT; icel++) {
    for (ie=0; ie<8; ie++) {
        ip=nodLOCAL[ie];
        for (je=0; je<8; je++) {
            jp=nodLOCAL[je];
            ...
            if (jp > ip) {
                AU[9*kk-9] += a11;
                AU[9*kk-8] += a12;
                AU[9*kk-7] += a13;
                AU[9*kk-6] += a21;
                AU[9*kk-5] += a22;
                AU[9*kk-4] += a23;
                AU[9*kk-3] += a31;
                AU[9*kk-2] += a32;
                AU[9*kk-1] += a33;
            }

            if (jp < ip) {
                AL[9*kk-9] += a11;
                AL[9*kk-8] += a12;
                AL[9*kk-7] += a13;
                AL[9*kk-6] += a21;
                AL[9*kk-5] += a22;
                AL[9*kk-4] += a23;
                AL[9*kk-3] += a31;
                AL[9*kk-2] += a32;
                AL[9*kk-1] += a33;
            }

            if (jp == ip) {
                D[9*ip-9] += a11;
                D[9*ip-8] += a12;
                D[9*ip-7] += a13;
                D[9*ip-6] += a21;
                D[9*ip-5] += a22;
                D[9*ip-4] += a23;
                D[9*ip-3] += a31;
                D[9*ip-2] += a32;
                D[9*ip-1] += a33;
            }
            ...
        }
    }
}

```

MAT_ASS_BC : 全体構成

```
do i= 1, N      節点ループ
  (ディリクレ) 境界条件を設定する節点をマーク (IWKX)
enddo
```

```
do i= 1, N      節点ループ
  if (IWKX(i,1).eq.1) then  マークされた節点だったら
    対応する右辺ベクトル (B) の成分, 対角ブロック (D) の成分の修正 (行・列)
    do k= indexL(i-1)+1, indexL(i)  下三角ブロック
      対応する非対角 (下三角) ブロック (AL) の成分の修正 (行)
    enddo
    do k= indexU(i-1)+1, indexU(i)  上三角ブロック
      対応する非対角 (上三角) ブロック (AU) の成分の修正 (行)
    enddo
  endif
enddo
```

```
do i= 1, N      節点ループ
  do k= indexL(i-1)+1, indexL(i)      下三角ブロック
    if (IWKX(itemL(k),1).eq.1) then  対応する非対角ブロックの節点がマークされていたら
      対応する右辺ベクトル, 非対角 (下三角) ブロック (AL) の成分の修正 (列)
    endif
  enddo
  do k= indexU(i-1)+1, indexU(i)      上三角ブロック
    if (IWKX(itemU(k),1).eq.1) then  対応する非対角ブロックの節点がマークされていたら
      対応する右辺ベクトル, 非対角 (上三角) ブロック (AU) の成分の修正 (列)
    endif
  enddo
enddo
```

境界条件 : MAT_ASS_BC (1/9)

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "pfem_util.h"
#include "allocate.h"
extern FILE *fp_log;
void MAT_ASS_BC()
{
    int i, j, k, in, ib, ib0, icel;
    int in1, in2, in3, in4, in5, in6, in7, in8;
    int iq1, iq2, iq3, iq4, iq5, iq6, iq7, iq8;
    int iS, iE;
    double STRESS, VAL;

    IWKX=(KINT**) allocate_matrix(sizeof(KINT), N, 2);
    for (i=0; i<N; i++) for (j=0; j<2; j++) IWKX[i][j]=0;
```

境界条件 : MAT_ASS_BC (2/9)

```

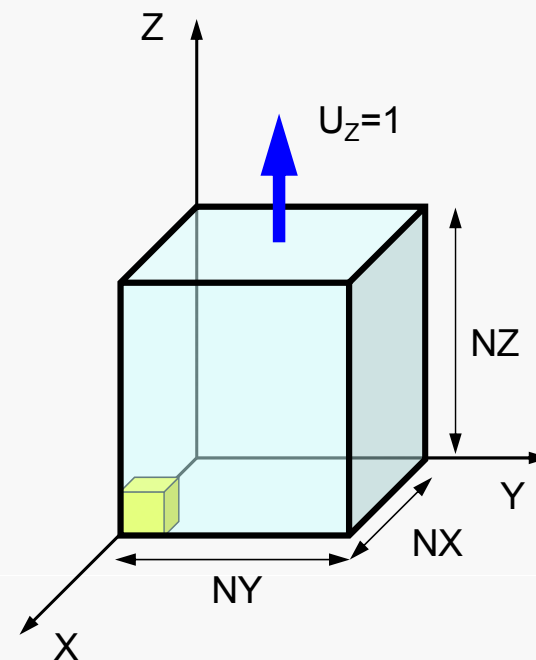
/**
Z=Zmax
**/
for (in=0; in<N; in++) IWKX[in][0]=0;

ib0=-1;
for ( ib0=0; ib0<NODGRPtot; ib0++) {
    if ( strcmp(NODGRP_NAME[ib0].name, "Zmax") == 0 ) break;
}

for ( ib=NODGRP_INDEX[ib0]; ib<NODGRP_INDEX[ib0+1]; ib++) {
    in=NODGRP_ITEM[ib];
    IWKX[in-1][0]=1;
}

for (in=0; in<N; in++) {
    if ( IWKX[in][0] == 1 ) {
        B[3*in] = B[3*in] - D[9*in+2]*1.e0;
        B[3*in+1] = B[3*in+1] - D[9*in+5]*1.e0;
        D[9*in+2] = 0.e0;
        D[9*in+5] = 0.e0;
        D[9*in+6] = 0.e0;
        D[9*in+7] = 0.e0;
        D[9*in+8] = 1.e0;
        B[3*in+2] = 1.e0;
        iS= indexL[in]+1;
        iE= indexL[in+1];
        for (k=iS; k<=iE; k++) {
            AL[9*k-3] = 0.e0;
            AL[9*k-2] = 0.e0;
            AL[9*k-1] = 0.e0;
        }
        iS= indexU[in]+1;
        iE= indexU[in+1];
        for (k=iS; k<=iE; k++) {
            AU[9*k-3] = 0.e0;
            AU[9*k-2] = 0.e0;
            AU[9*k-1] = 0.e0;
        }
    }
}
}

```



境界条件 : MAT_ASS_BC (2/9)

```

/**
Z=Zmax
**/
for (in=0; in<N; in++) IWKX[in][0]=0;

ib0=-1;

for ( ib0=0; ib0<NODGRPtot; ib0++) {
    if ( strcmp(NODGRP_NAME[ib0].name, "Zmax") == 0 ) break;
}

for ( ib=NODGRP_INDEX[ib0]; ib<NODGRP_INDEX[ib0+1]; ib++) {
    in=NODGRP_ITEM[ib];
    IWKX[in-1][0]=1;
}

for (in=0; in<N; in++) {
    if ( IWKX[in][0] == 1 ) {
        B[3*in] = B[3*in] - D[9*in+2]*
        B[3*in+1] = B[3*in+1] - D[9*in+5]*
        D[9*in+2] = 0. e0;
        D[9*in+5] = 0. e0;
        D[9*in+6] = 0. e0;
        D[9*in+7] = 0. e0;
        D[9*in+8] = 1. e0;
        B[3*in+2] = 1. e0;
        iS= indexL[in]+1;
        iE= indexL[in+1];
        for (k=iS; k<=iE; k++) {
            AL[9*k-3] = 0. e0;
            AL[9*k-2] = 0. e0;
            AL[9*k-1] = 0. e0;
        }
        iS= indexU[in]+1;
        iE= indexU[in+1];
        for (k=iS; k<=iE; k++) {
            AU[9*k-3] = 0. e0;
            AU[9*k-2] = 0. e0;
            AU[9*k-1] = 0. e0;
        }
    }
}
}

```

節点グループ名が「Zmax」である
節点において:

$IWKX[in-1][0] = 1$

とする

in: 1から始まる節点番号

境界条件 : MAT_ASS_BC (2/9)

```

/**
    Z=Zmax
**/
for (in=0; in<N; in++) IWKX[in][0]=0;

ib0=-1;
for ( ib0=0; ib0<NODGRPtot; ib0++) {
    if ( strcmp(NODGRP_NAME[ib0].name, "Zmax") == 0 ) break;
}

for ( ib=NODGRP_INDEX[ib0]; ib<NODGRP_INDEX[ib0+1]; ib++) {
    in=NODGRP_ITEM[ib];
    IWKX[in-1][0]=1;
}

for (in=0; in<N; in++) {
    if ( IWKX[in][0] == 1 ) {
        B[3*in] = B[3*in] - D[9*in+2]*1.
        B[3*in+1] = B[3*in+1] - D[9*in+5]*1.
        D[9*in+2] = 0. e0;
        D[9*in+5] = 0. e0;
        D[9*in+6] = 0. e0;
        D[9*in+7] = 0. e0;
        D[9*in+8] = 1. e0;
        B[3*in+2] = 1. e0;
        iS = indexL[in]+1;
        iE = indexL[in+1];
        for (k=iS; k<=iE; k++) {
            AL[9*k-3] = 0. e0;
            AL[9*k-2] = 0. e0;
            AL[9*k-1] = 0. e0;
        }
        iS = indexU[in]+1;
        iE = indexU[in+1];
        for (k=iS; k<=iE; k++) {
            AU[9*k-3] = 0. e0;
            AU[9*k-2] = 0. e0;
            AU[9*k-1] = 0. e0;
        }
    }
}

```

節点グループ名が「Zmax」である
節点では、Z方向変位成分=1に
すなわち:

$$B[3*in+2] = 1.0 \quad (in = 0 \sim N-1)$$

FORTTRANでは

$$B[3*in-2] = 1.0 \quad (in = 1, N)$$

境界条件 : MAT_ASS_BC (3/9)

```

for (in=0; in<N; in++) {
    iS= indexL[in]+1;
    iE= indexL[in+1];
    for (k=iS; k<=iE; k++) {
        if (IWKX[itemL[k-1]-1][0] == 1 ) {
            B[3*in] = B[3*in] - AL[9*k-7]*1.e0;
            B[3*in+1] = B[3*in+1] - AL[9*k-4]*1.e0;
            B[3*in+2] = B[3*in+2] - AL[9*k-1]*1.e0;
            AL[9*k-7] = 0.e0;
            AL[9*k-4] = 0.e0;
            AL[9*k-1] = 0.e0;
        }
    }
    iS= indexU[in]+1;
    iE= indexU[in+1];
    for (k=iS; k<=iE; k++) {
        if (IWKX[itemU[k-1]-1][0] == 1 ) {
            B[3*in] = B[3*in] - AU[9*k-7]*1.e0;
            B[3*in+1] = B[3*in+1] - AU[9*k-4]*1.e0;
            B[3*in+2] = B[3*in+2] - AU[9*k-1]*1.e0;
            AU[9*k-7] = 0.e0;
            AU[9*k-4] = 0.e0;
            AU[9*k-1] = 0.e0;
        }
    }
}

```

一次元

 $x=0$ で成立する方程式

$$u_1=0$$



- x 方向にのみ自由度（変位 u ）
 - 一様な：断面積 A , ヤング率 E
 - 境界条件
 - $x=0$: $u=0$ （固定）
 - $x=x_{max}$: 大きさ F の力（軸力）
- 自重によるたわみ等はナシ：バネと同じ

一次元

プログラム : 1d.c (6/7)

境界条件

```

/*
// +-----+
// | BOUNDARY conditions |
// +-----+
*/

/* X=Xmin */
i=0;
jS= Index[i];
AMat[jS]= 0.0;
Diag[i ]= 1.0;
Rhs [i ]= 0.0;

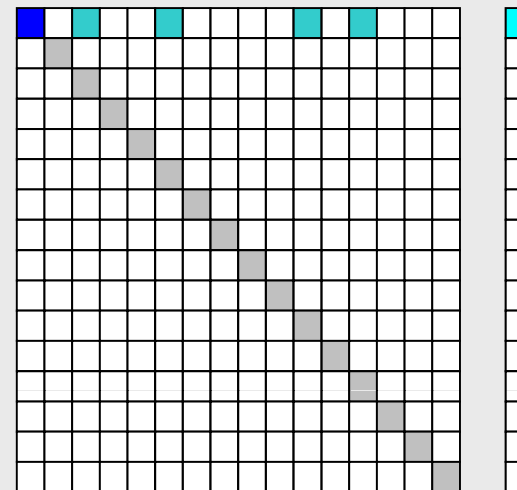
    for (k=0;k<NPLU;k++) {
        if (Item[k]==0) {AMat[k]=0.0;
        }}

/* X=Xmax */
i=N-1;
Rhs[i]= F;

```

 $u_1=0$

対角成分=1, 右辺=0, 非対角成分=0



一次元

プログラム : 1d.c (6/7)

境界条件

```

/*
// +-----+
// | BOUNDARY conditions |
// +-----+
*/

/* X=Xmin */
i=0;
jS= Index[i];
AMat[jS]= 0.0;
Diag[i ]= 1.0;
Rhs [i ]= 0.0;

    for (k=0;k<NPLU;k++) {
        if (Item[k]==0) {AMat[k]=0.0;
        }}

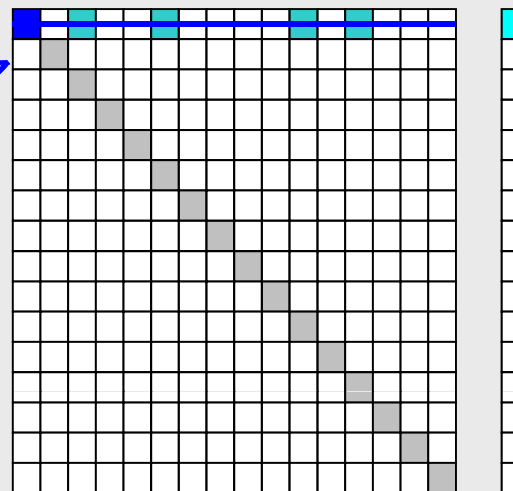
/* X=Xmax */
i=N-1;
Rhs[i]= F;

```

$$u_1=0$$

対角成分=1, 右辺=0, 非対角成分=0

ゼロクリア



一次元

プログラム : 1d.c (6/7)

境界条件

```

/*
// +-----+
// | BOUNDARY conditions |
// +-----+
*/

/* X=Xmin */
i=0;
jS= Index[i];
AMat[jS]= 0.0;
Diag[i ]= 1.0;
Rhs [i ]= 0.0;
    for (k=0;k<NPLU;k++) {
        if (Item[k]==0) {AMat[k]=0.0;
        }}

/* X=Xmax */
i=N-1;
Rhs[i]= F;

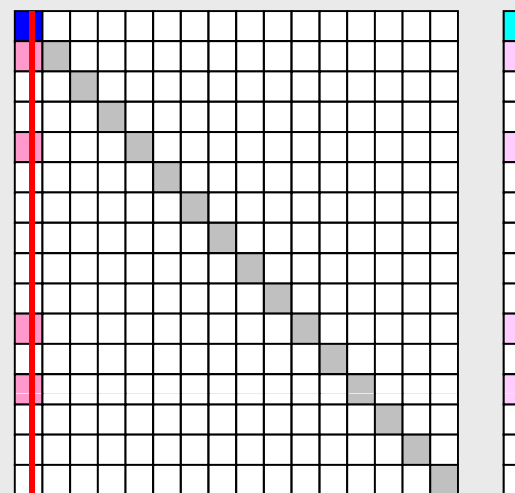
```

行列の対称性を保つため、第一種境界条件を適用している節点に対応する「列」を、右辺に移項して消去する(今の場合は非対角成分を0にするだけで良い)

$$u_1=0$$

対角成分=1, 右辺=0, 非対角成分=0

消去, ゼロクリア



一次元

プログラム : 1d.c (6/7)

境界条件

```

/*
// +-----+
// | BOUNDARY conditions |
// +-----+
*/

/* X=Xmin */
i=0;
jS= Index[i];
AMat[jS]= 0.0;
Diag[i ]= 1.0;
Rhs [i ]= 0.0;

    for (k=0;k<NPLU;k++) {
        if (Item[k]==0) {AMat[k]=0.0;
        }}

/* X=Xmax */
i=N-1;
Rhs[i]= F;

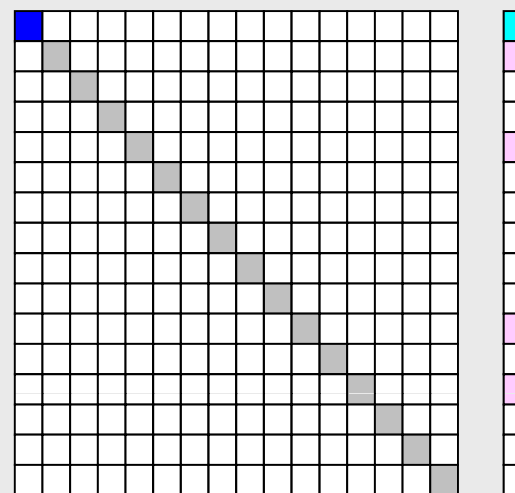
```

行列の対称性を保つため、第一種境界条件を適用している節点に対応する「列」を、右辺に移項して消去する(今の場合は非対角成分を0にするだけで良い)

$$u_1=0$$

対角成分=1, 右辺=0, 非対角成分=0

消去, ゼロクリア



一次元

第一種境界条件が $u \neq 0$ の場合

```

/*
// +-----+
// | BOUNDARY conditions |
// +-----+
*/

```

```

/* X=Xmin */
i=0;
jS= Index[i];
AMat[jS]= 0.0;
Diag[i ]= 1.0;
Rhs [i ]= Umin;

```

```

for (j=1; i<N; i++) {
  for (k=Index[j]; k<Index[j+1]; k++) {
    if (Item[k]==0) {
      Rhs [j]= Rhs[j] - AMat[k]*Umin;
      AMat[k]= 0.0;
    }
  }
}

```

行列の対称性を保つため，第一種境界条件を適用している節点に対応する「列」を，右辺に移項して消去する

$$Diag_j u_j + \sum_{k=Index[j]}^{Index[j+1]} AMat_k u_{Item[k]} = Rhs_j$$

一次元

第一種境界条件が $u \neq 0$ の場合

```

/*
// +-----+
// | BOUNDARY conditions |
// +-----+
*/

```

```

/* X=Xmin */
i=0;
jS= Index[i];
AMat[jS]= 0.0;
Diag[i ]= 1.0;
Rhs [i ]= Umin;

```

```

for (j=1; i<N; i++) {
  for (k=Index[j]; k<Index[j+1]; k++) {
    if (Item[k]==0) {
      Rhs [j]= Rhs[j] - Amat[k]*Umin;
      AMat[k]= 0.0;
    }
  }
}

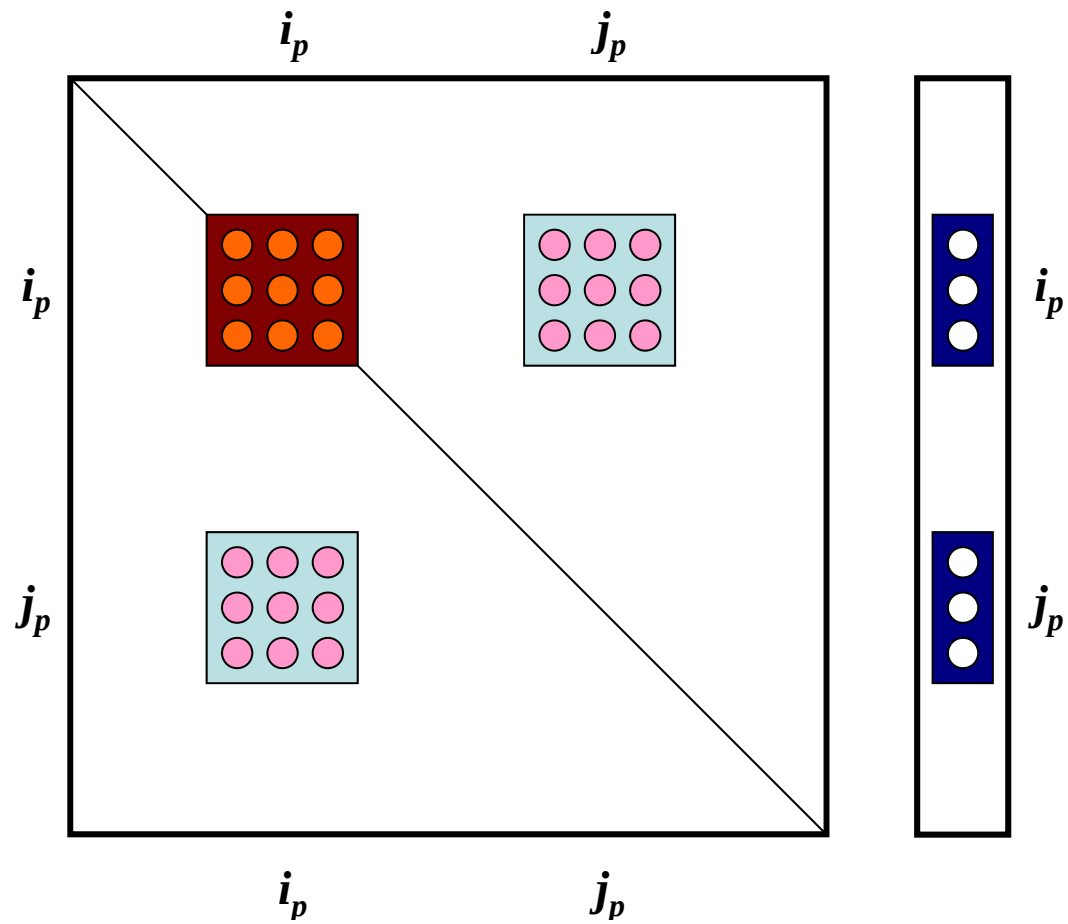
```

行列の対称性を保つため、第一種境界条件を適用している節点に対応する「列」を、右辺に移項して消去する

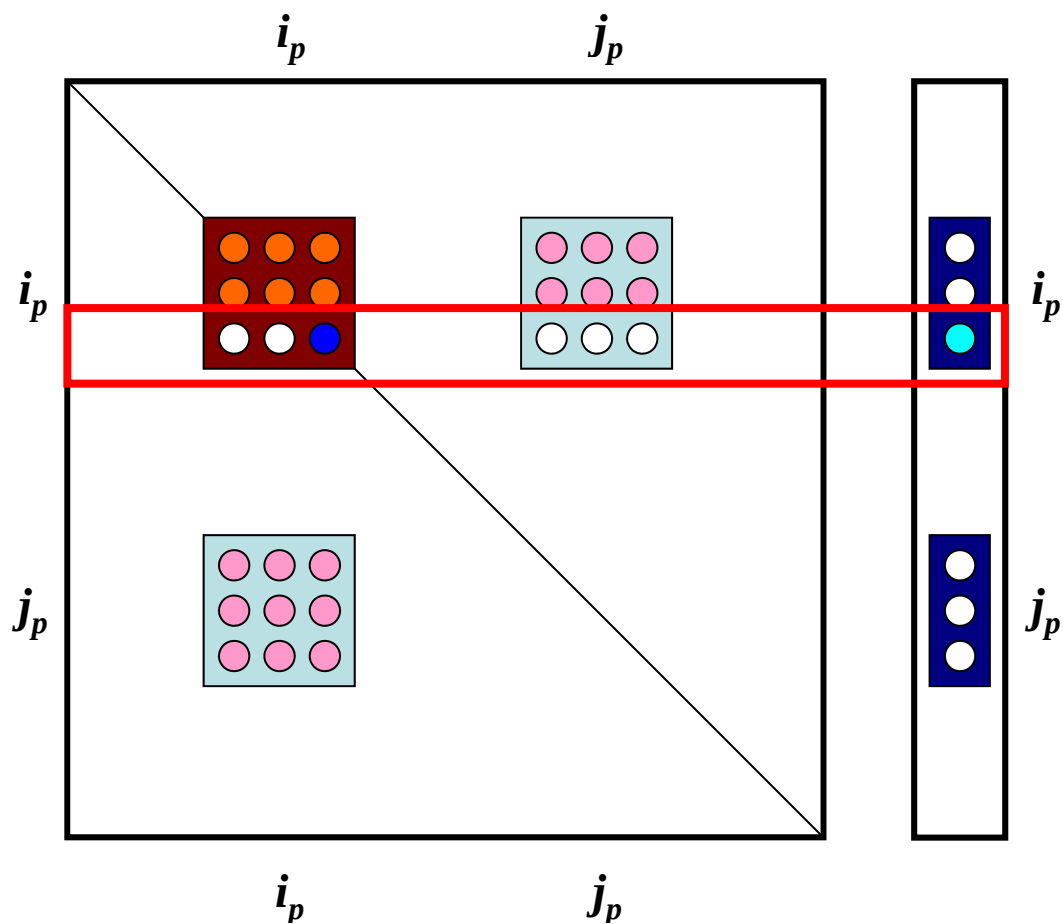
$$Diag_j u_j + \sum_{k=Index[j], k \neq k_s}^{Index[j+1]} Amat_k u_{Item[k]}$$

$$= Rhs_j - Amat_{k_s} u_{Item[k_s]} = Rhs_j - Amat_{k_s} u_{\min} \quad \text{where } Item[k_s] = 0$$

全体マトリクス

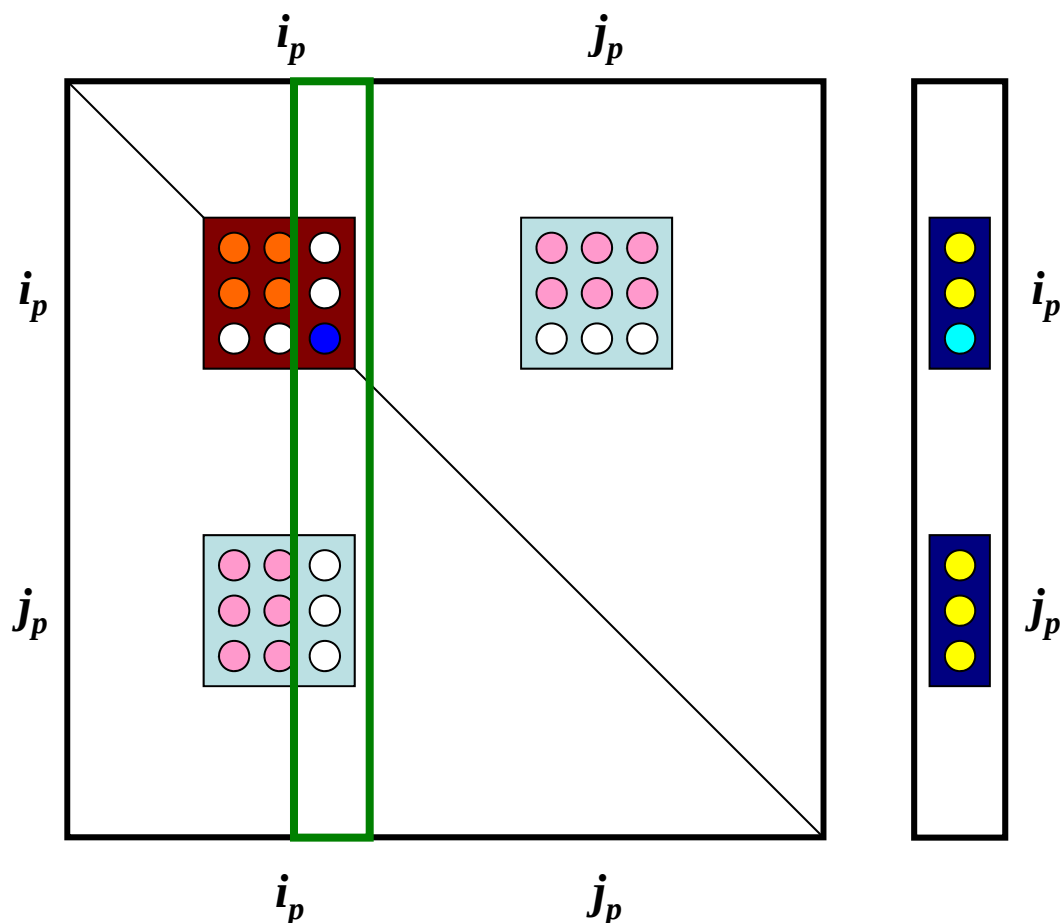


同じことをやればよい
自分の「行」：対角成分=1, それ以外=0



同じことをやればよい

自分の「列」：右辺へ移項，非対角成分=0



境界条件 : MAT_ASS_BC (2/9)

```

/**
Z=Zmax
**/
for (in=0; in<N; in++) IWKX[in][0]=0;

ib0=-1;

for ( ib0=0; ib0<NODGRPtot; ib0++) {
    if ( strcmp(NODGRP_NAME[ib0].name, "Zmax") == 0 ) break;
}

for ( ib=NODGRP_INDEX[ib0]; ib<NODGRP_INDEX[ib0+1]; ib++) {
    in=NODGRP_ITEM[ib];
    IWKX[in-1][0]=1;
}

for (in=0; in<N; in++) {
    if ( IWKX[in][0] == 1 ) {
        B[3*in] = B[3*in] - D[9*in+2]*
        B[3*in+1] = B[3*in+1] - D[9*in+5]*
        D[9*in+2] = 0. e0;
        D[9*in+5] = 0. e0;
        D[9*in+6] = 0. e0;
        D[9*in+7] = 0. e0;
        D[9*in+8] = 1. e0;
        B[3*in+2] = 1. e0;
        iS= indexL[in]+1;
        iE= indexL[in+1];
        for (k=iS; k<=iE; k++) {
            AL[9*k-3] = 0. e0;
            AL[9*k-2] = 0. e0;
            AL[9*k-1] = 0. e0;
        }
        iS= indexU[in]+1;
        iE= indexU[in+1];
        for (k=iS; k<=iE; k++) {
            AU[9*k-3] = 0. e0;
            AU[9*k-2] = 0. e0;
            AU[9*k-1] = 0. e0;
        }
    }
}
}

```

節点グループ名が「Zmax」である
節点において:

$IWKX[in-1][0] = 1$

とする

in: 1から始まる節点番号

境界条件 : MAT_ASS_BC (2/9)

```

/**
    Z=Zmax
**/
for (in=0; in<N; in++) IWKX[in][0]=0;

ib0=-1;
for ( ib0=0; ib0<NODGRPtot; ib0++) {
    if ( strcmp(NODGRP_NAME[ib0].name, "Zmax") == 0 ) break;
}

for ( ib=NODGRP_INDEX[ib0]; ib<NODGRP_INDEX[ib0+1]; ib++) {
    in=NODGRP_ITEM[ib];
    IWKX[in-1][0]=1;
}

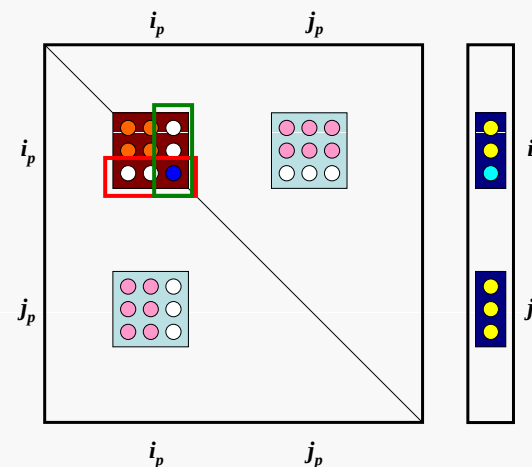
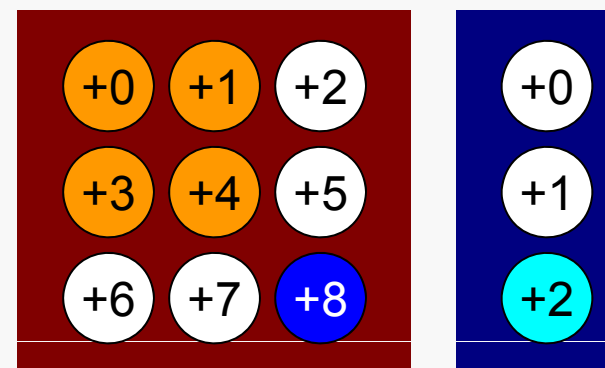
```

```

for (in=0; in<N; in++) {
    if ( IWKX[in][0] == 1 ) {
        B[3*in] = B[3*in] - D[9*in+2]*1.e0;
        B[3*in+1] = B[3*in+1] - D[9*in+5]*1.e0;
        D[9*in+2] = 0.e0;
        D[9*in+5] = 0.e0;
        D[9*in+6] = 0.e0;
        D[9*in+7] = 0.e0;
        D[9*in+8] = 1.e0;
        B[3*in+2] = 1.e0;
        iS = indexL[in]+1;
        iE = indexL[in+1];
        for (k=iS; k<=iE; k++) {
            AL[9*k-3] = 0.e0;
            AL[9*k-2] = 0.e0;
            AL[9*k-1] = 0.e0;
        }
        iS = indexU[in]+1;
        iE = indexU[in+1];
        for (k=iS; k<=iE; k++) {
            AU[9*k-3] = 0.e0;
            AU[9*k-2] = 0.e0;
            AU[9*k-1] = 0.e0;
        }
    }
}

```

B[3*in], B[3*in+1]を
変更してからD[9*in+6],
D[9*in+7]をゼロクリア



境界条件 : MAT_ASS_BC (2/9)

```

/**
    Z=Zmax
**/
for (in=0; in<N; in++) IWKX[in][0]=0;

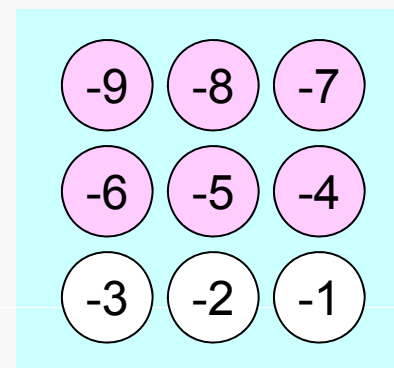
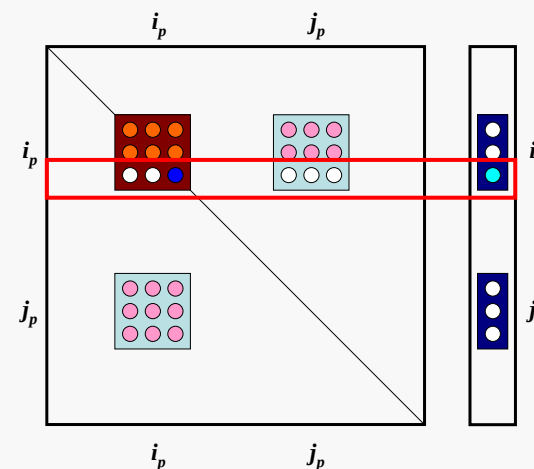
ib0=-1;
for ( ib0=0; ib0<NODGRPtot; ib0++) {
    if ( strcmp(NODGRP_NAME[ib0].name, "Zmax") == 0 ) break;
}

for ( ib=NODGRP_INDEX[ib0]; ib<NODGRP_INDEX[ib0+1]; ib++) {
    in=NODGRP_ITEM[ib];
    IWKX[in-1][0]=1;
}

for (in=0; in<N; in++) {
    if ( IWKX[in][0] == 1 ) {
        B[3*in] = B[3*in] - D[9*in+2]*1.e0;
        B[3*in+1] = B[3*in+1] - D[9*in+5]*1.e0;
        D[9*in+2] = 0.e0;
        D[9*in+5] = 0.e0;
        D[9*in+6] = 0.e0;
        D[9*in+7] = 0.e0;
        D[9*in+8] = 1.e0;
        B[3*in+2] = 1.e0;
        iS= indexL[in]+1;
        iE= indexL[in+1];
        for (k=iS; k<=iE; k++) {
            AL[9*k-3] = 0.e0;
            AL[9*k-2] = 0.e0;
            AL[9*k-1] = 0.e0;
        }
        iS= indexU[in]+1;
        iE= indexU[in+1];
        for (k=iS; k<=iE; k++) {
            AU[9*k-3] = 0.e0;
            AU[9*k-2] = 0.e0;
            AU[9*k-1] = 0.e0;
        }
    }
}

```

“k” starts from “1”



境界条件 : MAT_ASS_BC (3/9)

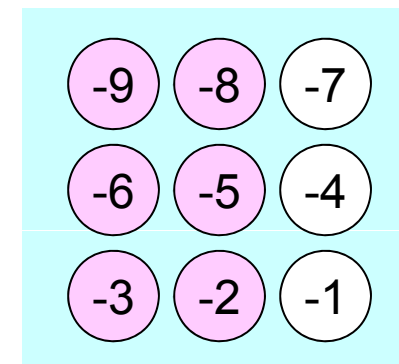
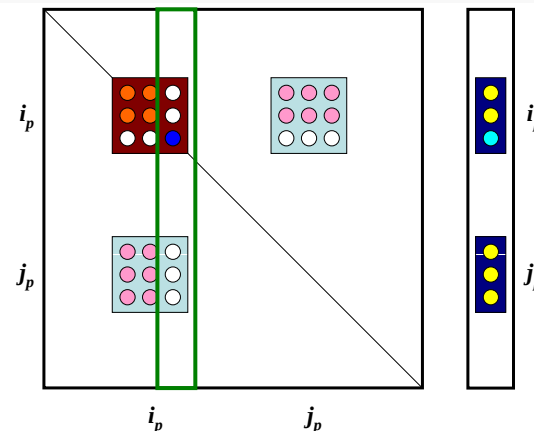
```

for (in=0; in<N; in++) {
    iS= indexL[in]+1;
    iE= indexL[in+1];
    for (k=iS; k<=iE; k++) {
        if (IWKX[itemL[k-1]-1][0] == 1 ) {
            B[3*in] = B[3*in] - AL[9*k-7]*1.e0;
            B[3*in+1] = B[3*in+1] - AL[9*k-4]*1.e0;
            B[3*in+2] = B[3*in+2] - AL[9*k-1]*1.e0;
            AL[9*k-7] = 0.e0;
            AL[9*k-4] = 0.e0;
            AL[9*k-1] = 0.e0;
        }
    }
    iS= indexU[in]+1;
    iE= indexU[in+1];
    for (k=iS; k<=iE; k++) {
        if (IWKX[itemU[k-1]-1][0] == 1 ) {
            B[3*in] = B[3*in] - AU[9*k-7]*1.e0;
            B[3*in+1] = B[3*in+1] - AU[9*k-4]*1.e0;
            B[3*in+2] = B[3*in+2] - AU[9*k-1]*1.e0;
            AU[9*k-7] = 0.e0;
            AU[9*k-4] = 0.e0;
            AU[9*k-1] = 0.e0;
        }
    }
}

```

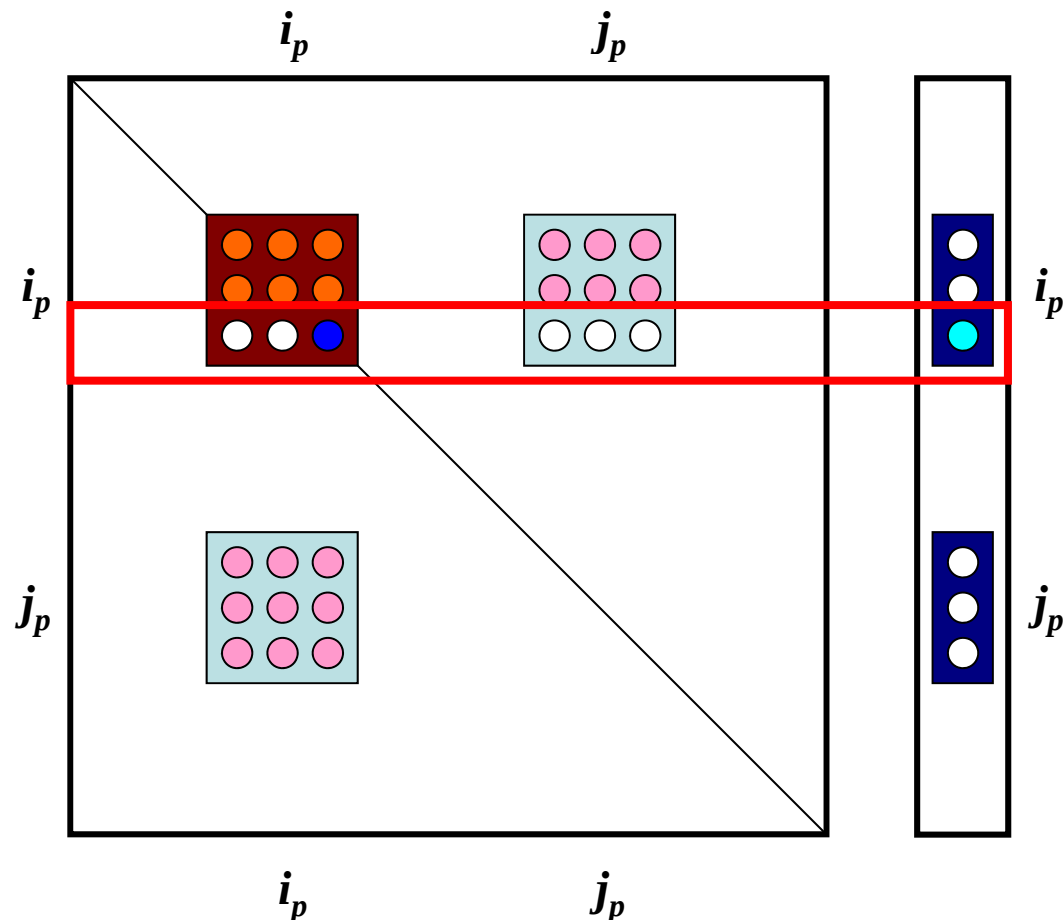
“k” starts from “1”

右辺を変更してから
ゼロクリア



$w=0@Zmin$

自分の「行」：対角成分=1, それ以外=0



境界条件 : MAT_ASS_BC (4/9)

```

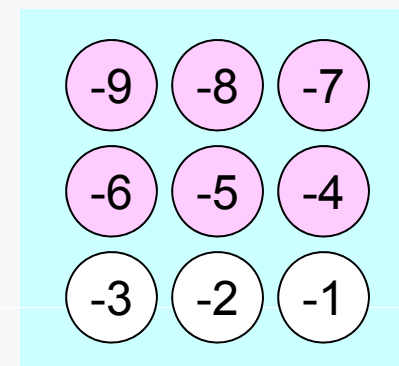
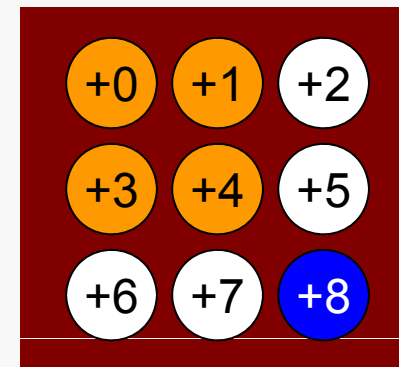
/**
**/
    Z=Zmin
    for (in=0; in<N; in++) IWKX[in][0]=0;

    ib0=-1;
    for ( ib0=0; ib0<NODGRPtot; ib0++) {
        if ( strcmp(NODGRP_NAME[ib0].name, "Zmin") == 0 ) break;
    }
    for ( ib=NODGRP_INDEX[ib0]; ib<NODGRP_INDEX[ib0+1]; ib++) {
        in=NODGRP_ITEM[ib];
        IWKX[in-1][0]=1;
    }
    for (in=0; in<N; in++) {
        if ( IWKX[in][0] == 1 ) {
            D[9*in+2]= 0. e0;
            D[9*in+5]= 0. e0;
            D[9*in+6]= 0. e0;
            D[9*in+7]= 0. e0;
            D[9*in+8]= 1. e0;
            B[3*in+2]= 0. e0;
            iS= indexL[in]+1;
            iE= indexL[in+1];
            for (k=iS; k<=iE; k++) {
                AL[9*k-3]= 0. e0;
                AL[9*k-2]= 0. e0;
                AL[9*k-1]= 0. e0;
            }
            iS= indexU[in]+1;
            iE= indexU[in+1];
            for (k=iS; k<=iE; k++) {
                AU[9*k-3]= 0. e0;
                AU[9*k-2]= 0. e0;
                AU[9*k-1]= 0. e0;
            }
        }
    }
}

```

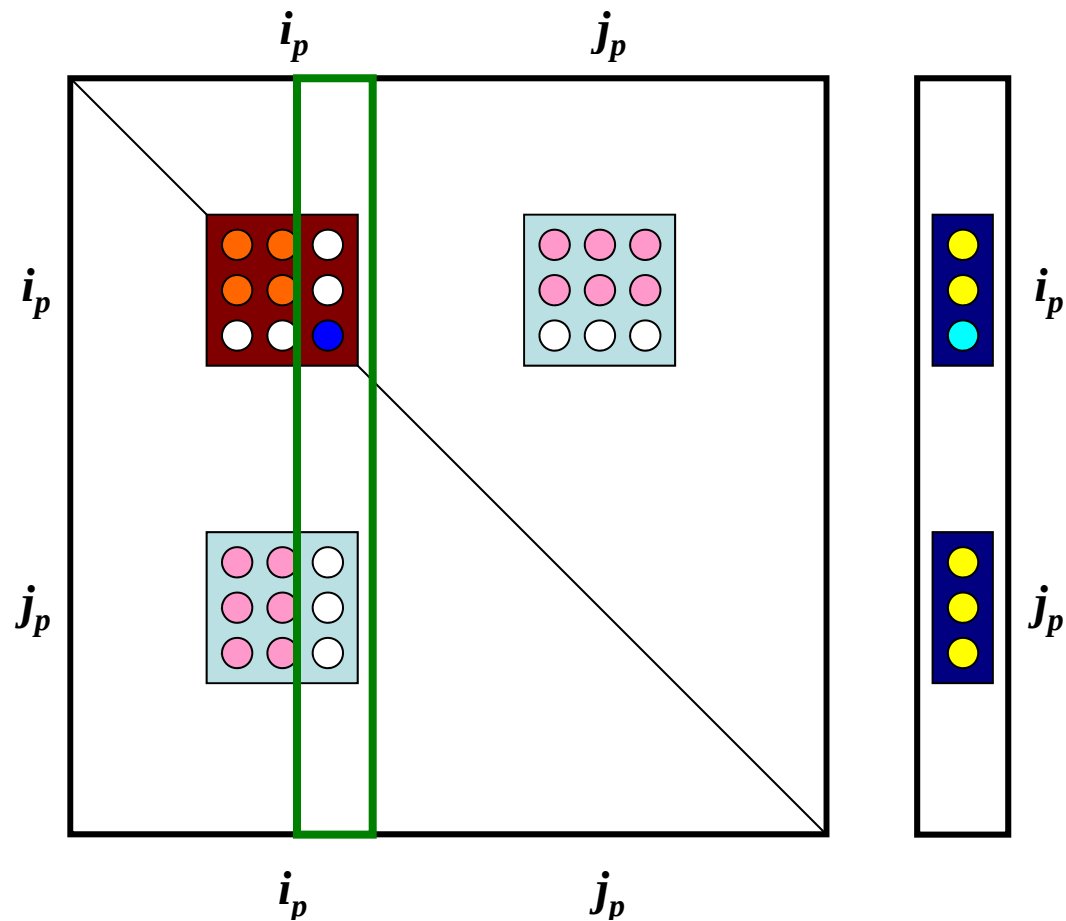
w=0@Zmin

“k” starts from “1”



$$w=0@Zmin$$

自分の「列」：右辺へ移項，非対角成分=0



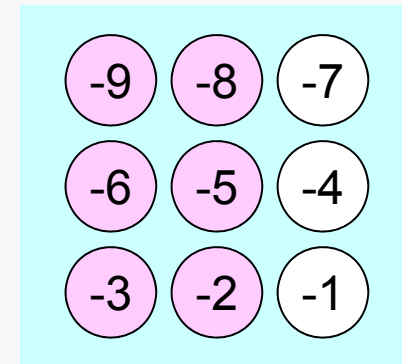
境界条件 : MAT_ASS_BC (5/9)

```

for (in=0; in<N; in++) {
    iS= indexL[in]+1;
    iE= indexL[in+1];
    for (k=iS; k<=iE; k++) {
        if (IWKX[itemL[k-1]-1][0] == 1 ) {
            AL[9*k-7]= 0. e0;
            AL[9*k-4]= 0. e0;
            AL[9*k-1]= 0. e0;
        }
    }
    iS= indexU[in]+1;
    iE= indexU[in+1];
    for (k=iS; k<=iE; k++) {
        if (IWKX[itemU[k-1]-1][0] == 1 ) {
            AU[9*k-7]= 0. e0;
            AU[9*k-4]= 0. e0;
            AU[9*k-1]= 0. e0;
        }
    }
}

```

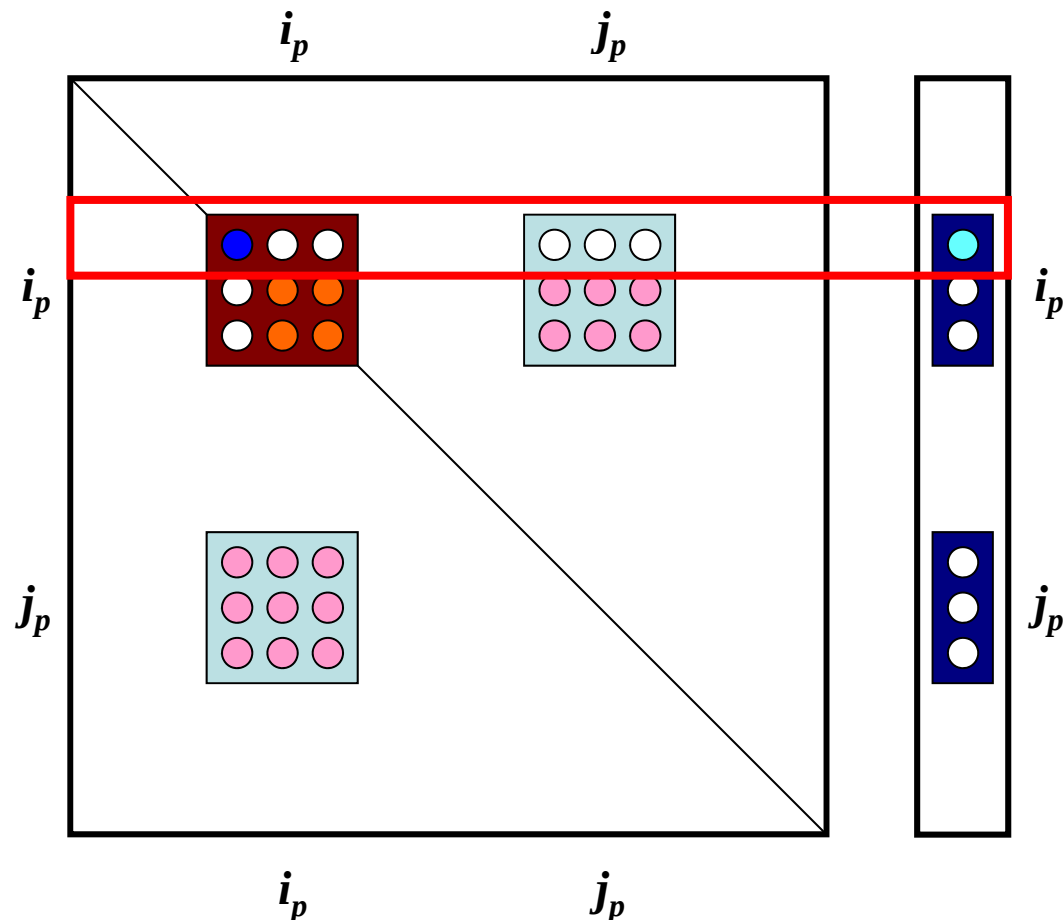
w=0@Zmin



“k” starts from “1”

$u=0@Xmin$

自分の「行」：対角成分=1, それ以外=0



境界条件 : MAT_ASS_BC (6/9)

```

/**
X=Xmin
**/
for (in=0; in<N; in++) IWKX[in][0]=0;

ib0=-1;
for ( ib0=0; ib0<NODGRPtot; ib0++) {
    if ( strcmp(NODGRP_NAME[ib0].name, "Xmin") == 0 ) break;
}

for ( ib=NODGRP_INDEX[ib0]; ib<NODGRP_INDEX[ib0+1]; ib++) {
    in=NODGRP_ITEM[ib];
    IWKX[in-1][0]=1;
}

for (in=0; in<N; in++) {
    if ( IWKX[in][0] == 1 ) {
        D[9*in] = 1. e0;
        D[9*in+1] = 0. e0;
        D[9*in+2] = 0. e0;
        D[9*in+3] = 0. e0;
        D[9*in+6] = 0. e0;
        B[3*in] = 0. e0;

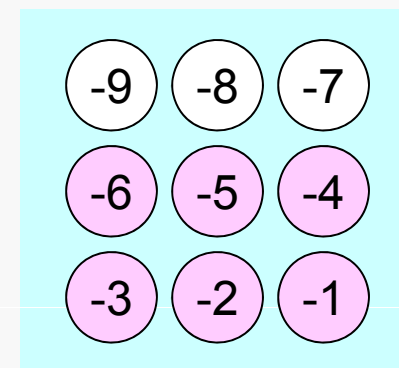
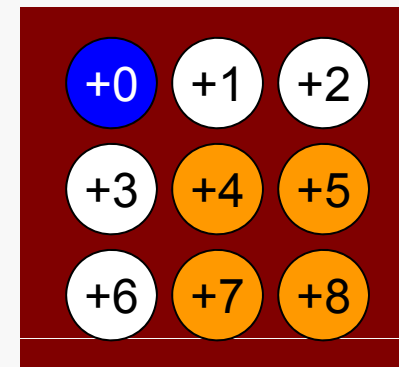
        iS= indexL[in]+1;
        iE= indexL[in+1];
        for (k=iS; k<=iE; k++) {
            AL[9*k-9] = 0. e0;
            AL[9*k-8] = 0. e0;
            AL[9*k-7] = 0. e0;
        }

        iS= indexU[in]+1;
        iE= indexU[in+1];
        for (k=iS; k<=iE; k++) {
            AU[9*k-9] = 0. e0;
            AU[9*k-8] = 0. e0;
            AU[9*k-7] = 0. e0;
        }
    }
}

```

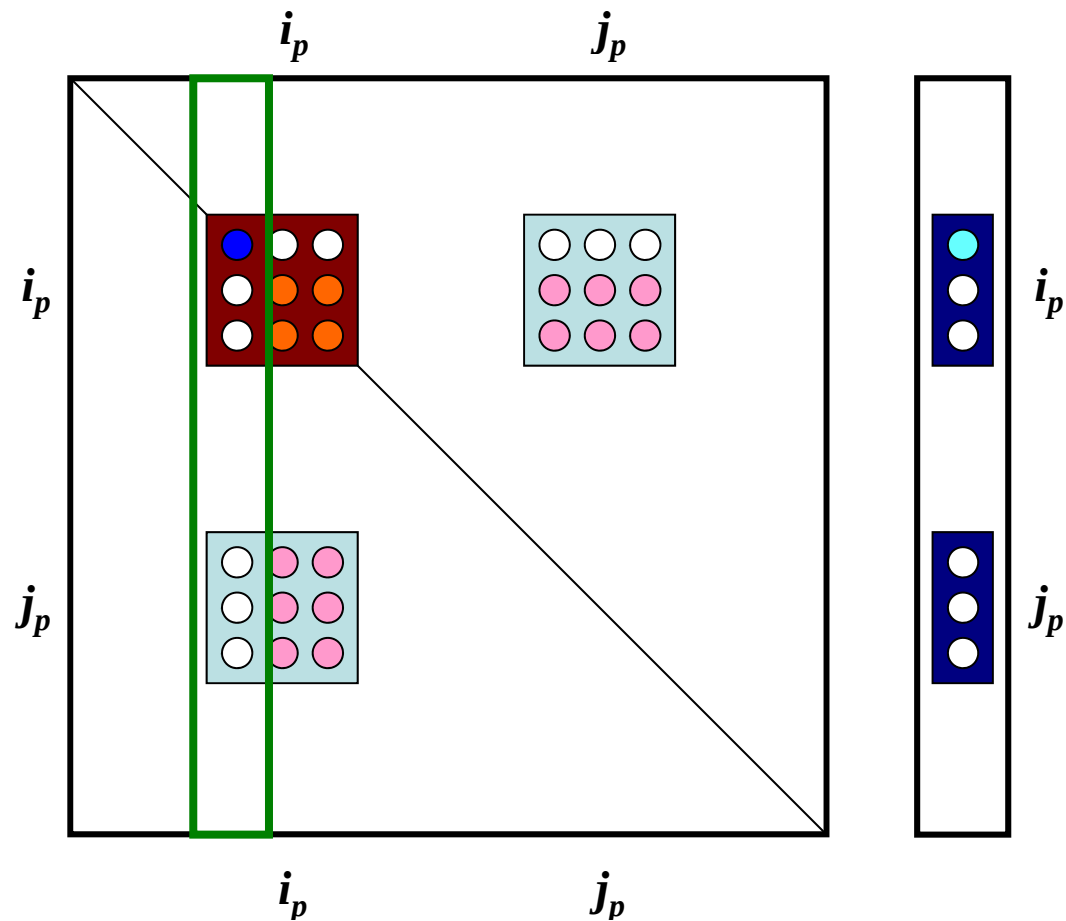
$u=0@Xmin$

“k” starts from “1”



$$\mathbf{u}=\mathbf{0}@X_{\min}$$

自分の「列」：右辺へ移項，非対角成分=0



境界条件 : MAT_ASS_BC (7/9)

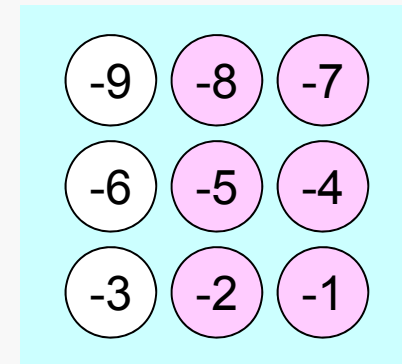
```

for (in=0; in<N; in++) {
    iS= indexL[in]+1;
    iE= indexL[in+1];
    for (k=iS; k<=iE; k++) {
        if (IWKX[itemL[k-1]-1][0] == 1 ) {
            AL[9*k-9]= 0. e0;
            AL[9*k-6]= 0. e0;
            AL[9*k-3]= 0. e0;
        }
    }

    iS= indexU[in]+1;
    iE= indexU[in+1];
    for (k=iS; k<=iE; k++) {
        if (IWKX[itemU[k-1]-1][0] == 1 ) {
            AU[9*k-9]= 0. e0;
            AU[9*k-6]= 0. e0;
            AU[9*k-3]= 0. e0;
        }
    }
}

```

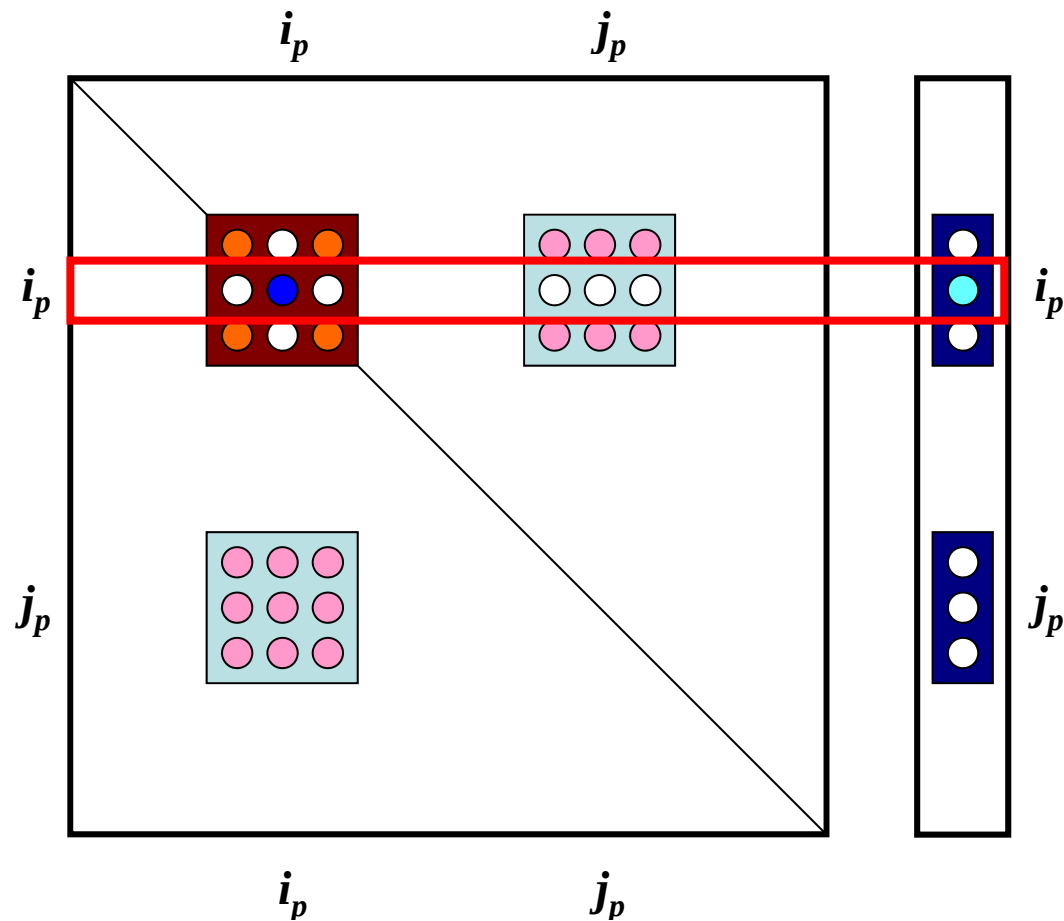
u=0@Xmin



“k” starts from “1”

$\mathbf{v}=\mathbf{0}@Y_{min}$

自分の「行」：対角成分=1, それ以外=0



境界条件 : MAT_ASS_BC (8/9)

```

/**
**/
    Y=Ymin
    for (in=0; in<N; in++) IWKX[in][0]=0;

    ib0=-1;
    for ( ib0=0; ib0<NODGRPtot; ib0++) {
        if ( strcmp(NODGRP_NAME[ib0].name, "Ymin") == 0 ) break;
    }

    for ( ib=NODGRP_INDEX[ib0]; ib<NODGRP_INDEX[ib0+1]; ib++) {
        in=NODGRP_ITEM[ib];
        IWKX[in-1][0]=1;
    }

    for (in=0; in<N; in++) {
        if ( IWKX[in][0] == 1 ) {
            D[9*in+1]= 0. e0;
            D[9*in+4]= 1. e0;
            D[9*in+7]= 0. e0;
            D[9*in+3]= 0. e0;
            D[9*in+5]= 0. e0;
            B[3*in+1]= 0. e0;

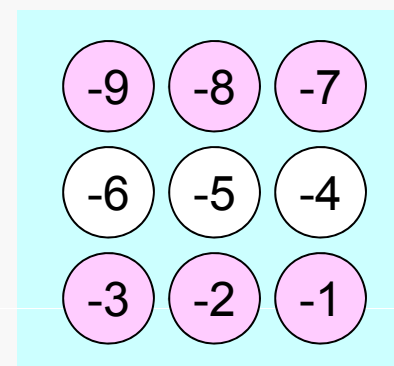
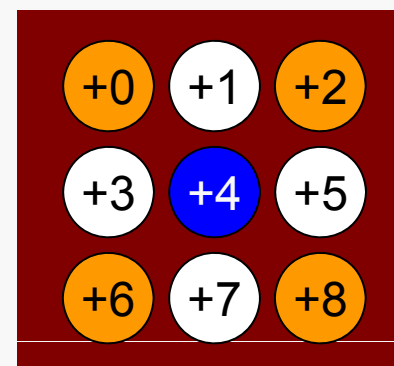
            iS= indexL[in]+1;
            iE= indexL[in+1];
            for (k=iS; k<=iE; k++) {
                AL[9*k-6]= 0. e0;
                AL[9*k-5]= 0. e0;
                AL[9*k-4]= 0. e0;
            }

            iS= indexU[in]+1;
            iE= indexU[in+1];
            for (k=iS; k<=iE; k++) {
                AU[9*k-6]= 0. e0;
                AU[9*k-5]= 0. e0;
                AU[9*k-4]= 0. e0;
            }
        }
    }

```

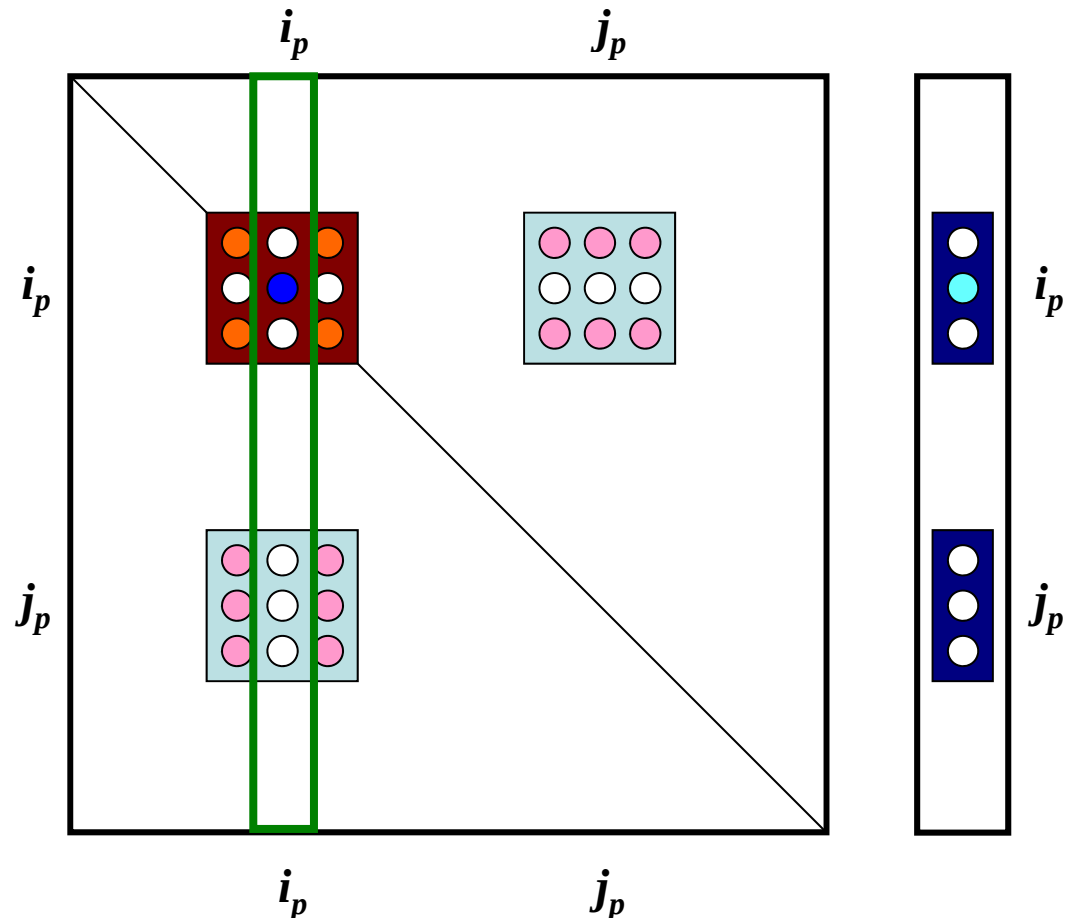
v=0@Ymin

“k” starts from “1”



$$\mathbf{v} = \mathbf{0} @ \mathbf{Y}_{\min}$$

自分の「列」：右辺へ移項，非対角成分=0



境界条件 : MAT_ASS_BC (9/9)

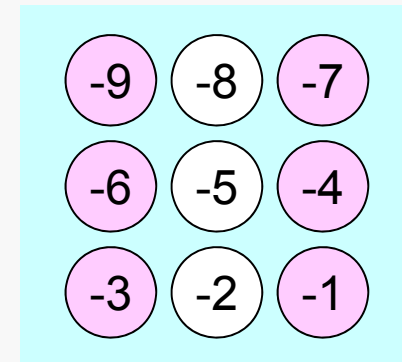
```

for (in=0; in<N; in++) {
    iS= indexL[in]+1;
    iE= indexL[in+1];
    for (k=iS; k<=iE; k++) {
        if (IWXX[itemL[k-1]-1][0] == 1 ) {
            AL[9*k-8]= 0. e0;
            AL[9*k-5]= 0. e0;
            AL[9*k-2]= 0. e0;
        }
    }

    iS= indexU[in]+1;
    iE= indexU[in+1];
    for (k=iS; k<=iE; k++) {
        if (IWXX[itemU[k-1]-1][0] == 1 ) {
            AU[9*k-8]= 0. e0;
            AU[9*k-5]= 0. e0;
            AU[9*k-2]= 0. e0;
        }
    }
}
}

```

v=0@Ymin



“k” starts from “1”