

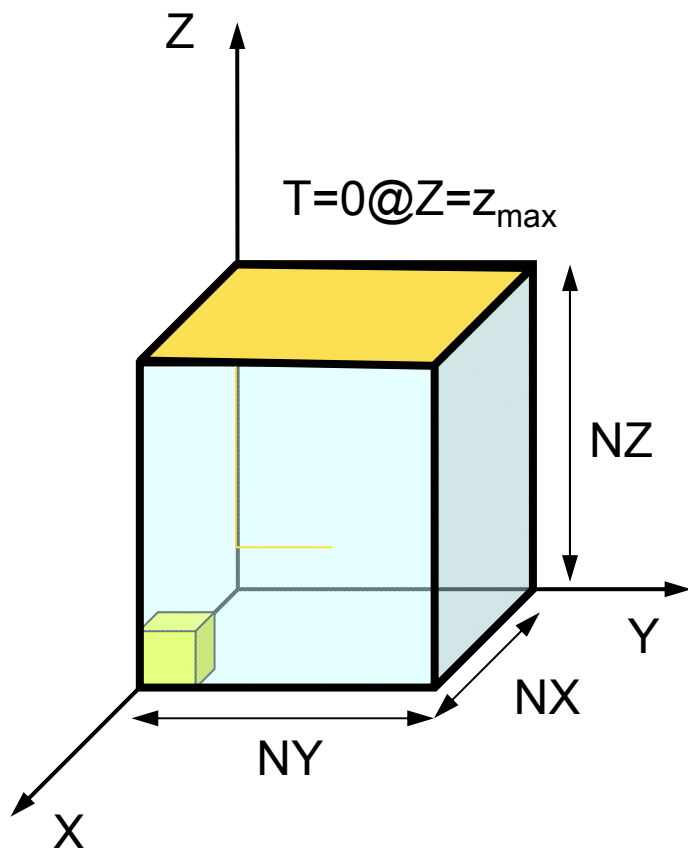
有限要素法による
三次元定常熱伝導解析プログラム
Fortran編

中島 研吾

東京大学情報基盤センター

対象とする問題：三次元定常熱伝導

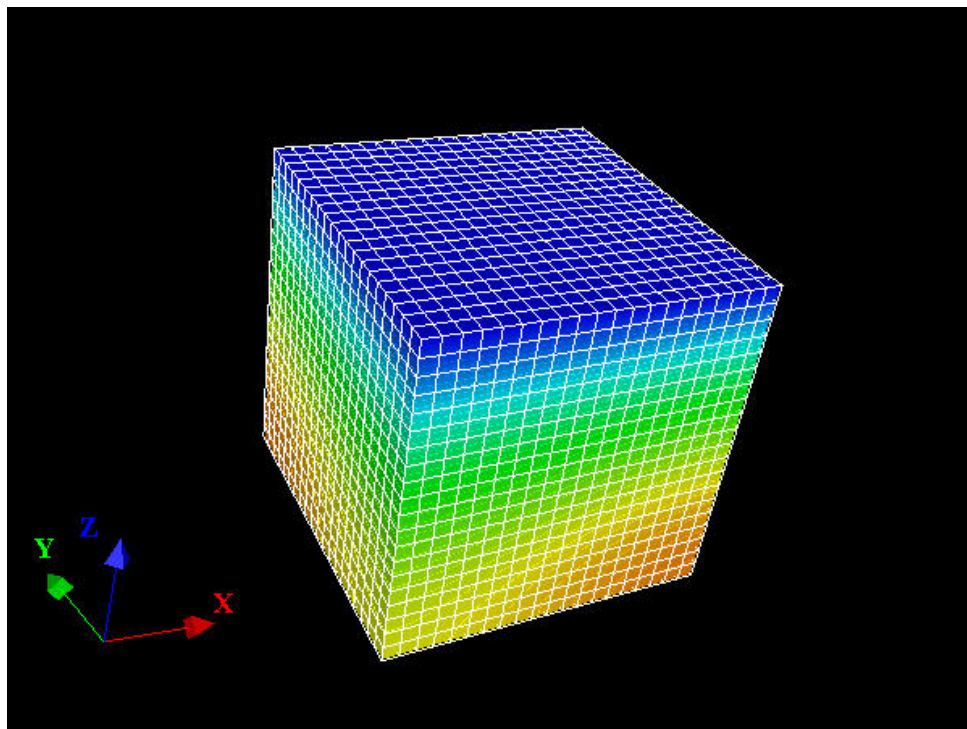
$$\frac{\partial}{\partial x} \left(\lambda \frac{\partial T}{\partial x} \right) + \frac{\partial}{\partial y} \left(\lambda \frac{\partial T}{\partial y} \right) + \frac{\partial}{\partial z} \left(\lambda \frac{\partial T}{\partial z} \right) + \dot{Q}(x, y, z) = 0$$



- 定常熱伝導＋発熱
- 一様な熱伝導率 λ
- 直方体
 - 一辺長さ1の立方体（六面体）要素
 - 各方向に $NX \cdot NY \cdot NZ$ 個
- 境界条件
 - $T=0@Z=z_{\max}$
- 体積当たり発熱量は位置（メッシュの中心の座標 x_c, y_c ）に依存
 - $\dot{Q}(x, y, z) = QVOL|x_c + y_c|$

対象とする問題：三次元定常熱伝導

$$\frac{\partial}{\partial x} \left(\lambda \frac{\partial T}{\partial x} \right) + \frac{\partial}{\partial y} \left(\lambda \frac{\partial T}{\partial y} \right) + \frac{\partial}{\partial z} \left(\lambda \frac{\partial T}{\partial z} \right) + \dot{Q}(x, y, z) = 0$$



- 原点から遠い部分が高温
- 体積当たり発熱量は位置（メッシュの中心の座標）に依存
 - $\dot{Q}(x, y, z) = |x_c + y_c|$

movie

有限要素法の処理

- 支配方程式
- ガラーキン法：弱形式
- 要素単位の積分
 - 要素マトリクス生成
- 全体マトリクス生成
- 境界条件適用
- 連立一次方程式

有限要素法の処理：プログラム

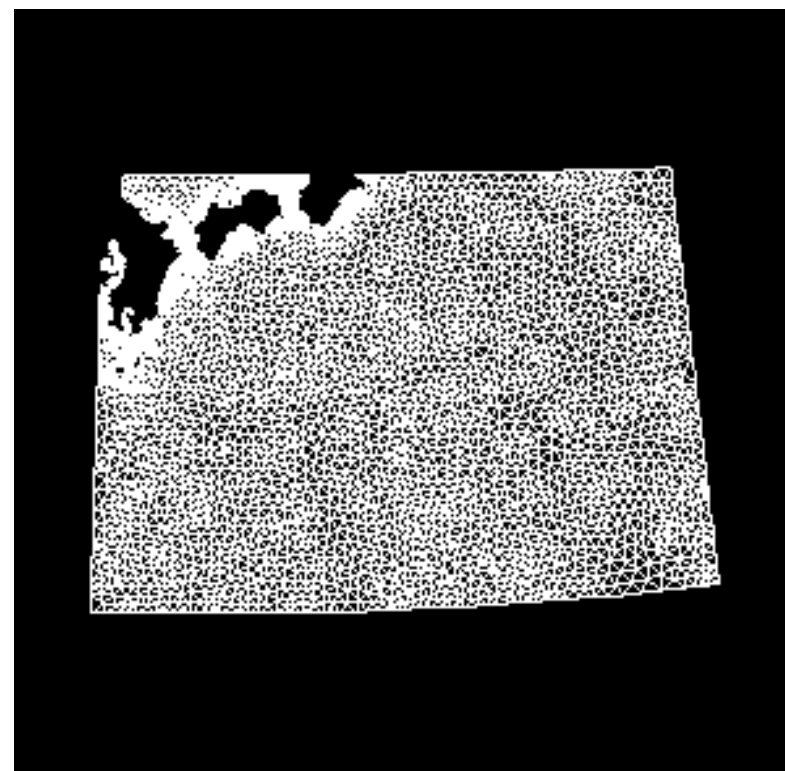
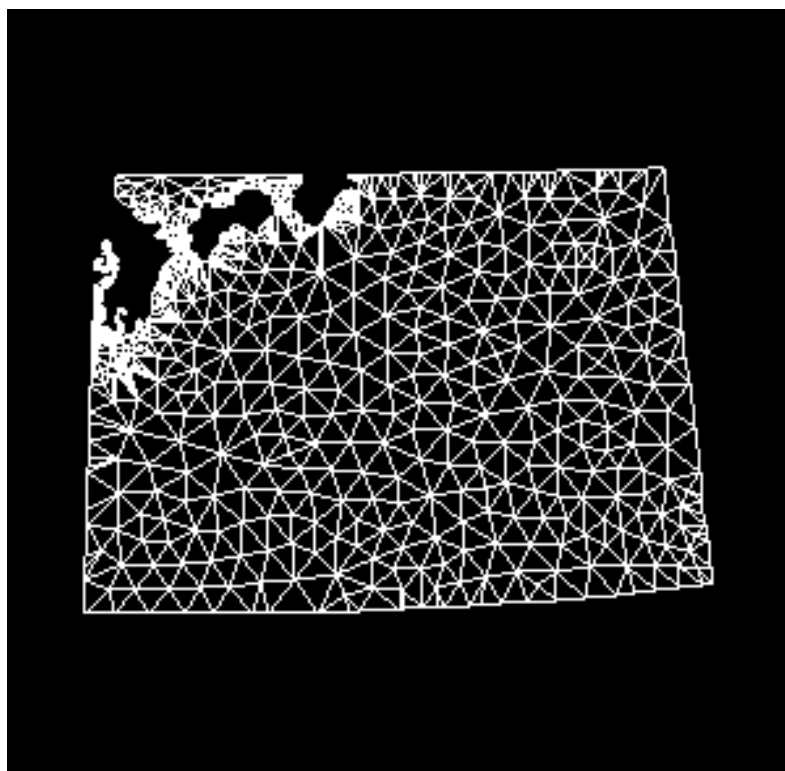
- 初期化
 - 制御変数読み込み
 - 座標読み込み⇒要素生成 (N:節点数, ICELTOT : 要素数)
 - 配列初期化 (全体マトリクス, 要素マトリクス)
 - 要素⇒全体マトリクスマッピング (Index, Item)
- マトリクス生成
 - 要素単位の処理 (do icel= 1, ICELTOT)
 - 要素マトリクス計算
 - 全体マトリクスへの重ね合わせ
 - 境界条件の処理
- 連立一次方程式
 - 共役勾配法 (CG)

- 三次元要素の定式化
- 三次元熱伝導方程式
 - ガラーキン法
 - 要素マトリクス生成

- プログラムの実行
- データ構造
- プログラムの構成

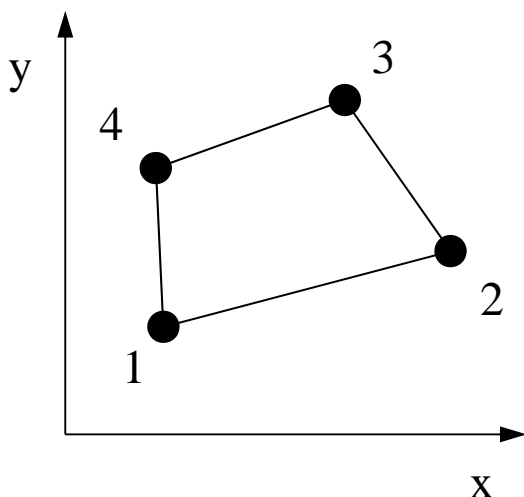
二次元への拡張：三角形要素

- 任意の形状を扱うことができる。
- 特に一次要素は精度が悪く，一部の問題を除いてあまり使用されない。



二次元への拡張：四角形要素

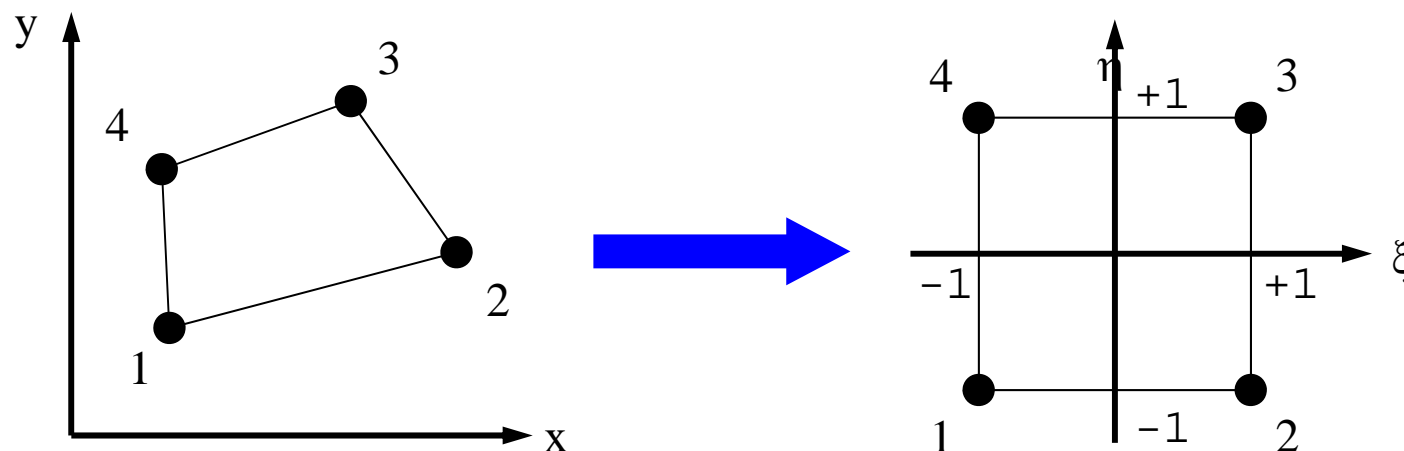
- 一次元要素と同じ形状関数を x, y 軸に適用することによって、四角形要素の定式化は可能である。
 - 三角形と比較して特に低次要素の精度はよい
- しかしながら、各辺が座標軸に平行な長方形でなければならない
 - 差分法と変わらない



- このような形状を扱うことができない。

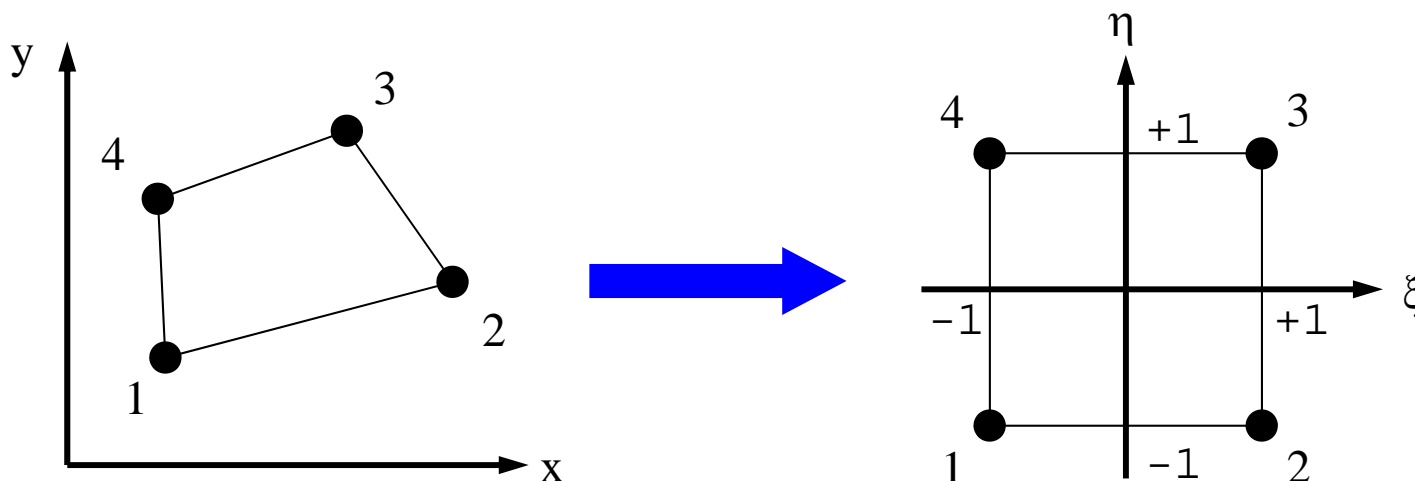
アイソパラメトリック要素 (1/3)

- 各要素を, 自然座標系 (ξ, η) の正方形要素 $[\pm 1, \pm 1]$ に変換する。



- 各要素の全体座標系 (global coordinate) (x, y) における座標成分を, 自然座標系における形状関数 $[N]$ (従属変数の内挿に使うのと同じ $[N]$) を使用して変換する場合, このような要素を**アイソパラメトリック要素 (isoparametric element)** という

アイソパラメトリック要素 (2/3)

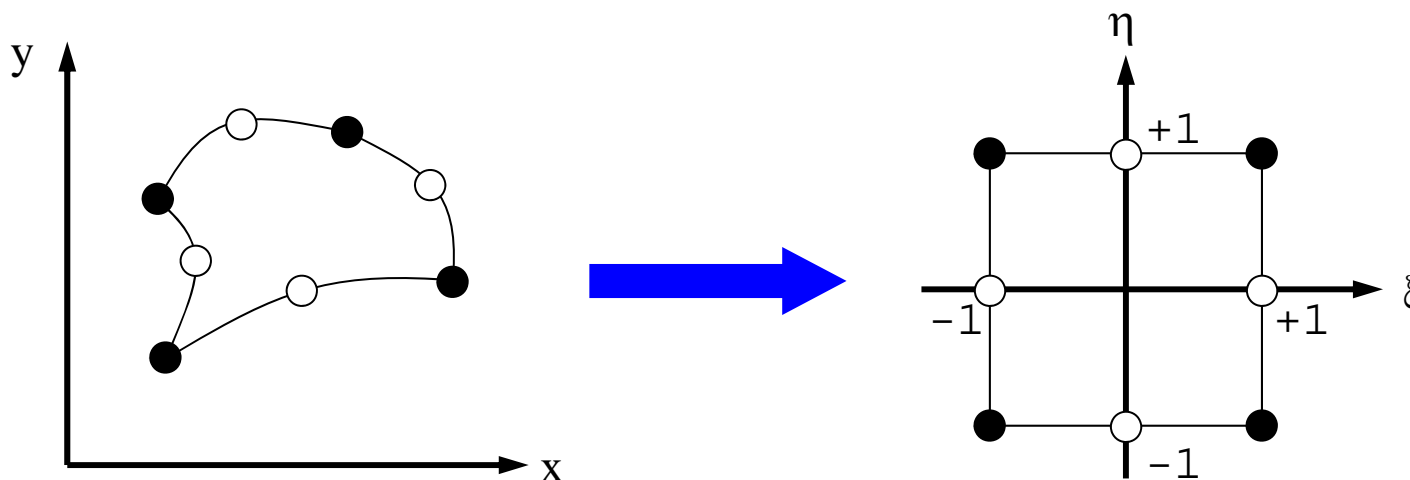


- 各節点の座標 : $(x_1, y_1), (x_2, y_2), (x_3, y_3), (x_4, y_4)$
- 各節点における温度 : T_1, T_2, T_3, T_4

$$T = \sum_{i=1}^4 N_i(\xi, \eta) \cdot T_i$$

$$x = \sum_{i=1}^4 N_i(\xi, \eta) \cdot x_i, \quad y = \sum_{i=1}^4 N_i(\xi, \eta) \cdot y_i$$

アイソパラメトリック要素 (3/3)



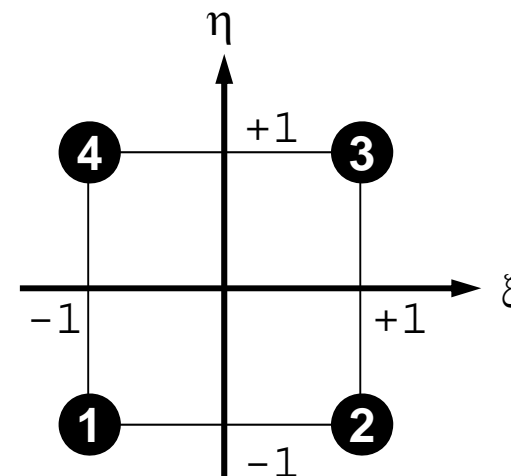
- 高次の補間関数を使えば，曲線，曲面も扱うことが可能となる。
- そういう意味で「自然座標系」と呼んでいる。

Sub-Parametric
Super-Parametric

2D自然座標系の形状関数 (1/3)

- 自然座標系における正方形上の内挿多項式は下式で与えられる：

$$T = \alpha_1 + \alpha_2 \xi + \alpha_3 \eta + \alpha_4 \xi \eta$$



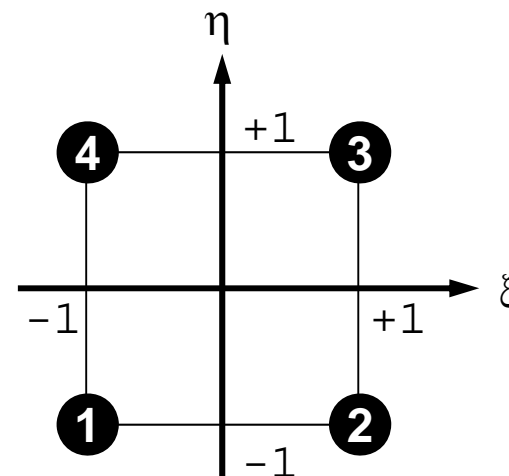
- 各節点での条件より：

$$\alpha_1 = \frac{T_1 + T_2 + T_3 + T_4}{4}, \quad \alpha_2 = \frac{-T_1 + T_2 + T_3 - T_4}{4},$$

$$\alpha_3 = \frac{-T_1 - T_2 + T_3 + T_4}{4}, \quad \alpha_4 = \frac{T_1 - T_2 + T_3 - T_4}{4}$$

2D自然座標系の形状関数 (2/3)

$$\begin{aligned}
 T &= \alpha_1 + \alpha_2 \xi + \alpha_3 \eta + \alpha_4 \xi \eta \\
 &= \frac{T_1 + T_2 + T_3 + T_4}{4} + \frac{-T_1 + T_2 + T_3 - T_4}{4} \xi + \\
 &\quad \frac{-T_1 - T_2 + T_3 + T_4}{4} \eta + \frac{T_1 - T_2 + T_3 - T_4}{4} \xi \eta \\
 &= \frac{1}{4} (1 - \xi - \eta + \xi \eta) T_1 + \frac{1}{4} (1 + \xi - \eta - \xi \eta) T_2 + \\
 &\quad \frac{1}{4} (1 + \xi + \eta + \xi \eta) T_3 + \frac{1}{4} (1 - \xi + \eta - \xi \eta) T_4 \\
 &= \overset{N_1}{\boxed{\frac{1}{4} (1 - \xi)(1 - \eta)}} T_1 + \overset{N_2}{\boxed{\frac{1}{4} (1 + \xi)(1 - \eta)}} T_2 + \\
 &\quad \overset{N_3}{\boxed{\frac{1}{4} (1 + \xi)(1 + \eta)}} T_3 + \overset{N_4}{\boxed{\frac{1}{4} (1 - \xi)(1 + \eta)}} T_4
 \end{aligned}$$



2D自然座標系の形状関数 (3/3)

- 元の式に代入して, T_i について整理すると以下のようなになる:

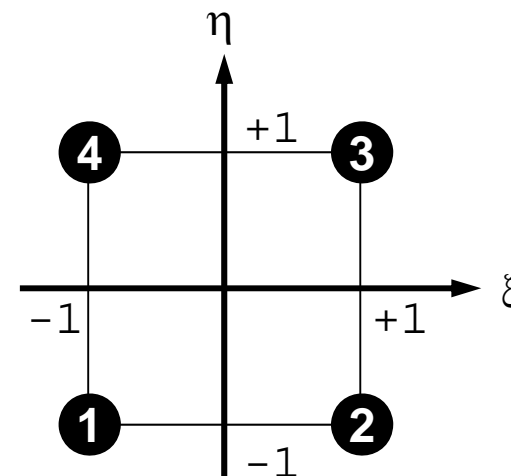
$$T = N_1T_1 + N_2T_2 + N_3T_3 + N_4T_4$$

- 形状関数 N_i は以下のようなになる:

$$N_1(\xi, \eta) = \frac{1}{4}(1-\xi)(1-\eta), \quad N_2(\xi, \eta) = \frac{1}{4}(1+\xi)(1-\eta),$$

$$N_3(\xi, \eta) = \frac{1}{4}(1+\xi)(1+\eta), \quad N_4(\xi, \eta) = \frac{1}{4}(1-\xi)(1+\eta)$$

- 双一次 (bi-linear) 要素とも呼ばれる。
- 各節点における N_i の値を計算してみよ



三次元への拡張

- 四面体要素：二次元における三角形要素
 - 任意の形状を扱うことができる。
 - 特に一次要素は精度が悪く，一部の問題を除いてあまり使用されない。
 - 高次の四面体要素は広く使用されている・・・
- 本講義では低次六面体要素（アイソパラメトリック要素）を使用する。
 - tri-linear
- 自由度：温度@各節点上

3D自然座標系の形状関数

$$N_1(\xi, \eta, \zeta) = \frac{1}{8}(1-\xi)(1-\eta)(1-\zeta) \quad N_5(\xi, \eta, \zeta) = \frac{1}{8}(1-\xi)(1-\eta)(1+\zeta)$$

$$N_2(\xi, \eta, \zeta) = \frac{1}{8}(1+\xi)(1-\eta)(1-\zeta) \quad N_6(\xi, \eta, \zeta) = \frac{1}{8}(1+\xi)(1-\eta)(1+\zeta)$$

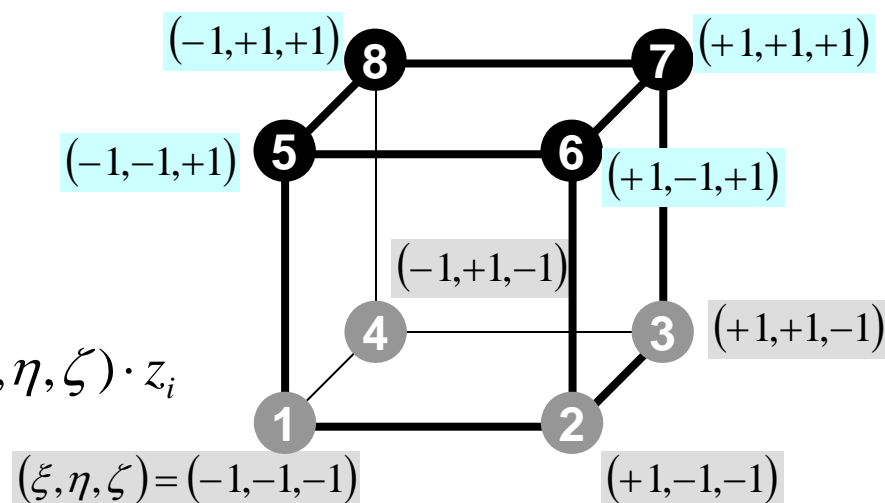
$$N_3(\xi, \eta, \zeta) = \frac{1}{8}(1+\xi)(1+\eta)(1-\zeta) \quad N_7(\xi, \eta, \zeta) = \frac{1}{8}(1+\xi)(1+\eta)(1+\zeta)$$

$$N_4(\xi, \eta, \zeta) = \frac{1}{8}(1-\xi)(1+\eta)(1-\zeta) \quad N_8(\xi, \eta, \zeta) = \frac{1}{8}(1-\xi)(1+\eta)(1+\zeta)$$

$$T = \sum_{i=1}^8 N_i(\xi, \eta, \zeta) \cdot T_i$$

$$x = \sum_{i=1}^8 N_i(\xi, \eta, \zeta) \cdot x_i$$

$$y = \sum_{i=1}^8 N_i(\xi, \eta, \zeta) \cdot y_i, \quad z = \sum_{i=1}^8 N_i(\xi, \eta, \zeta) \cdot z_i$$



- 三次元要素の定式化
- 三次元熱伝導方程式
 - ガラーキン法
 - 要素マトリクス生成

- プログラムの実行
- データ構造
- プログラムの構成

ガラーキン法の適用 (1/3)

- 以下のような三次元熱伝導方程式を考慮する（熱伝導率一定）：

$$\left(\lambda \frac{\partial^2 T}{\partial x^2} \right) + \left(\lambda \frac{\partial^2 T}{\partial y^2} \right) + \left(\lambda \frac{\partial^2 T}{\partial z^2} \right) + \dot{Q} = 0$$

$T = [N]\{\phi\}$ 要素内の温度分布
(マトリクス形式), 節点における温度を ϕ としてある。

- ガラーキン法に従い, 重み関数を $[N]$ とすると, 各要素において以下の積分方程式が得られる:

$$\int_V [N]^T \left\{ \lambda \left(\frac{\partial^2 T}{\partial x^2} \right) + \lambda \left(\frac{\partial^2 T}{\partial y^2} \right) + \lambda \left(\frac{\partial^2 T}{\partial z^2} \right) + \dot{Q} \right\} dV = 0$$

ガラーキン法の適用 (2/3)

- 三次元のグリーンの定理

$$\int_V A \left(\frac{\partial^2 B}{\partial x^2} + \frac{\partial^2 B}{\partial y^2} + \frac{\partial^2 B}{\partial z^2} \right) dV = \int_S A \frac{\partial B}{\partial n} dS - \int_V \left(\frac{\partial A}{\partial x} \frac{\partial B}{\partial x} + \frac{\partial A}{\partial y} \frac{\partial B}{\partial y} + \frac{\partial A}{\partial z} \frac{\partial B}{\partial z} \right) dV$$

- 前式の2階微分の部分に適用 (表面積分省略) :

$$\begin{aligned} & \int_V [N]^T \{ \lambda(T_{,xx}) + \lambda(T_{,yy}) + \lambda(T_{,zz}) \} dV \\ &= - \int_V \{ \lambda([N_{,x}]^T T_{,x}) + \lambda([N_{,y}]^T T_{,y}) + \lambda([N_{,z}]^T T_{,z}) \} dV \end{aligned}$$

- これに以下を代入する :

$$T = [N]\{\phi\}, \quad T_{,x} = [N_{,x}]\{\phi\}, \quad T_{,y} = [N_{,y}]\{\phi\}, \quad T_{,z} = [N_{,z}]\{\phi\}$$

ガラーキン法の適用 (3/3)

- 体積あたり発熱量の項 \dot{Q} を加えて次式が得られる :

$$-\int_V \left\{ \lambda \left([N_{,x}]^T [N_{,x}] \right) + \lambda \left([N_{,y}]^T [N_{,y}] \right) + \lambda \left([N_{,z}]^T [N_{,z}] \right) \right\} dV \cdot \{\phi\} + \int_V \dot{Q} [N] dV = 0$$

- この式を弱形式 (weak form) と呼ぶ。元の微分方程式では2階の微分が含まれていたが、上式では、グリーンの定理によって1階微分に低減されている。
 - 弱形式によって近似関数 (形状関数, 内挿関数) に対する要求が弱くなっている : すなわち線形関数で2階微分の効果を記述できる。
 - 項が増えただけで、次元と同じ

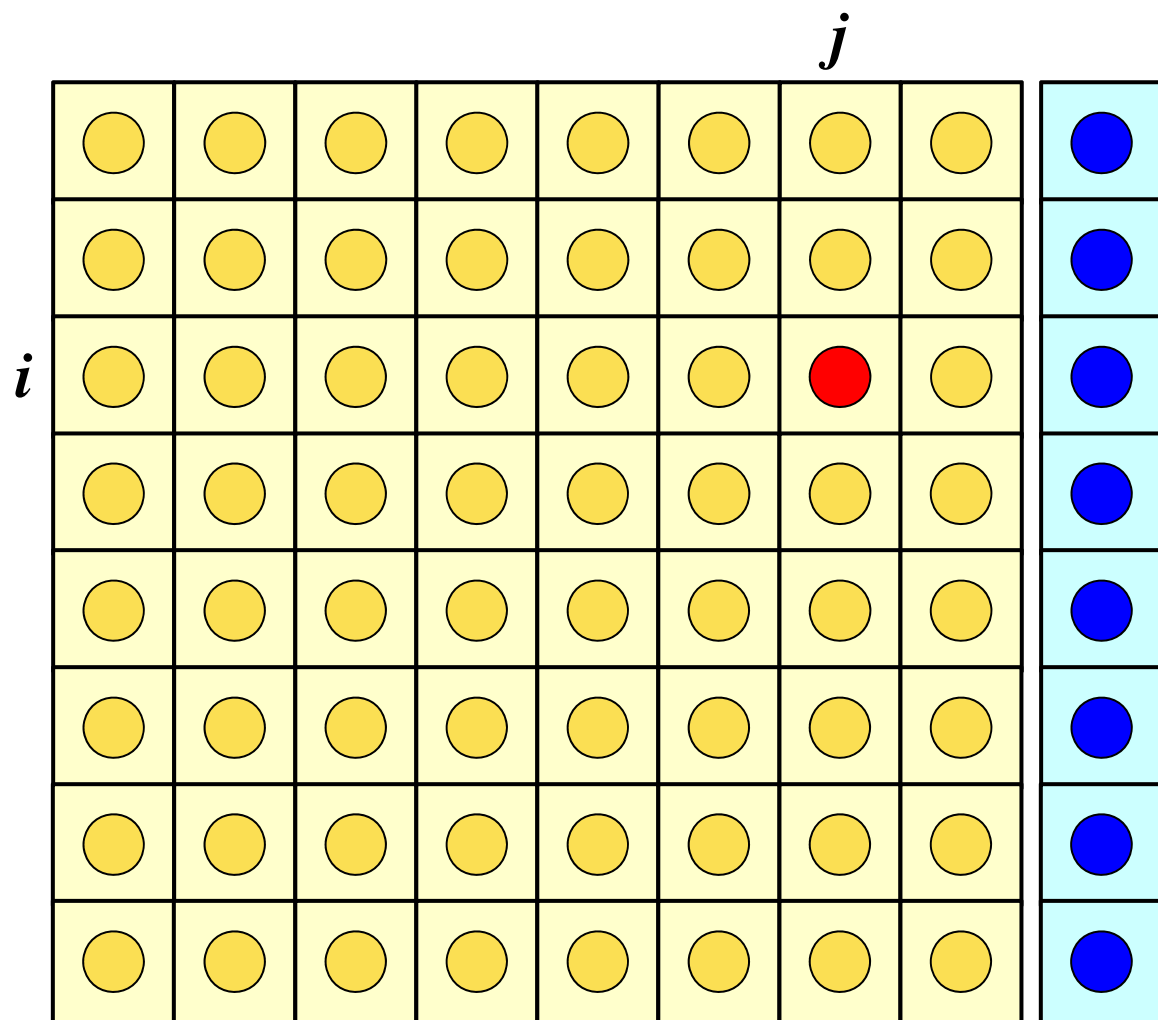
境界条件を考慮した弱形式：各要素

$$[k]^{(e)} \{\phi\}^{(e)} = \{f\}^{(e)}$$

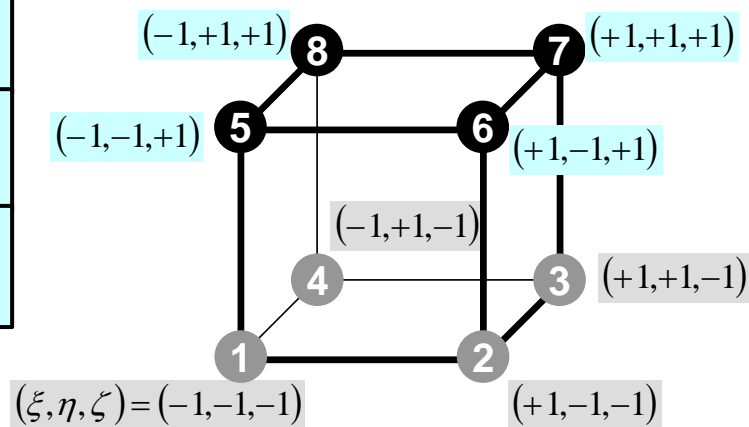
$$[k]^{(e)} = \int_V \lambda \left([N_{,x}]^T [N_{,x}] \right) dV + \int_V \lambda \left([N_{,y}]^T [N_{,y}] \right) dV \\ + \int_V \lambda \left([N_{,z}]^T [N_{,z}] \right) dV$$

$$\{f\}^{(e)} = \int_V \dot{Q} [N]^T dV$$

要素マトリクス : 8 × 8行列

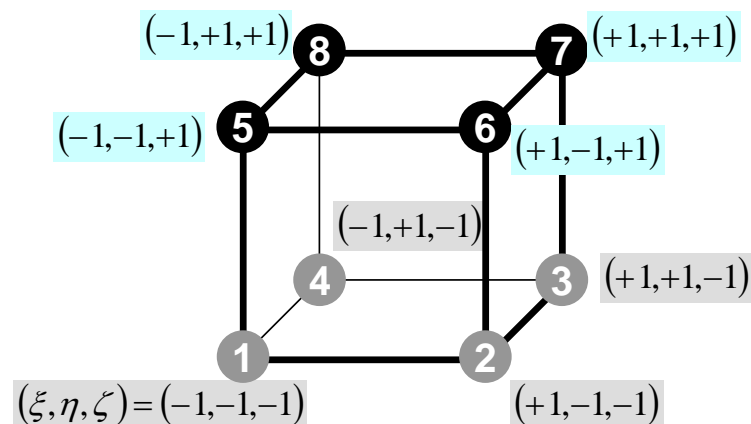


$$[k_{ij}] \quad (i, j = 1 \dots 8)$$



要素マトリクス : k_{ij}

$$[k_{ij}] \quad (i, j = 1 \dots 8)$$



$$[k]^{(e)} = \int_V \lambda ([N_{,x}]^T [N_{,x}]) dV + \int_V \lambda ([N_{,y}]^T [N_{,y}]) dV + \int_V \lambda ([N_{,z}]^T [N_{,z}]) dV$$

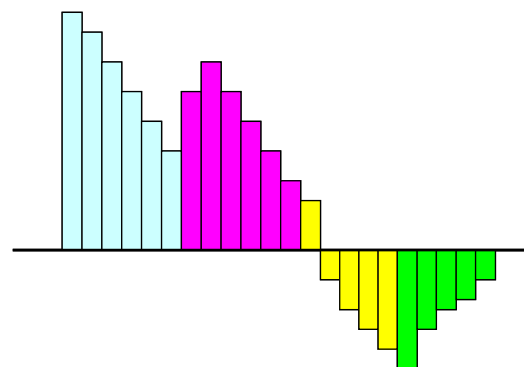


$$k_{ij} = - \int_V \{ \lambda \cdot N_{i,x} \cdot N_{j,x} + \lambda \cdot N_{i,y} \cdot N_{j,y} + \lambda \cdot N_{i,z} \cdot N_{j,z} \} dV$$

数値的に積分を実施する方法

- 台形公式
- シンプソンの公式
- ガウスの積分公式（ガウス＝ルジャンドル（Gauss-Legendre）とも呼ばれる，精度良い）
- 不定積分を解析的に求めるのではなく，有限な数のサンプル点における値を利用する

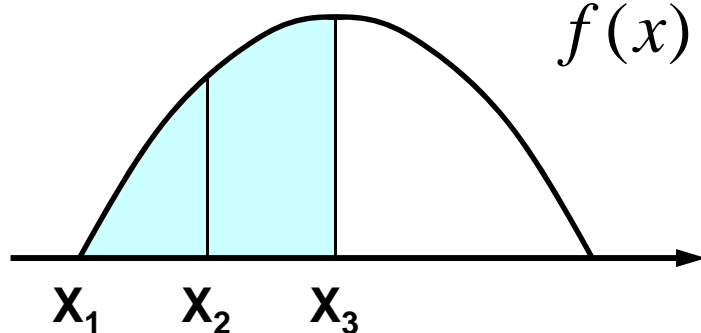
$$\int_{x_1}^{x_2} f(x) dx \Rightarrow \sum_{k=1}^m [w_k \cdot f(x_k)]$$



ガウス積分公式：一次元の例

シンプソンの公式より精度良い

Simpson's



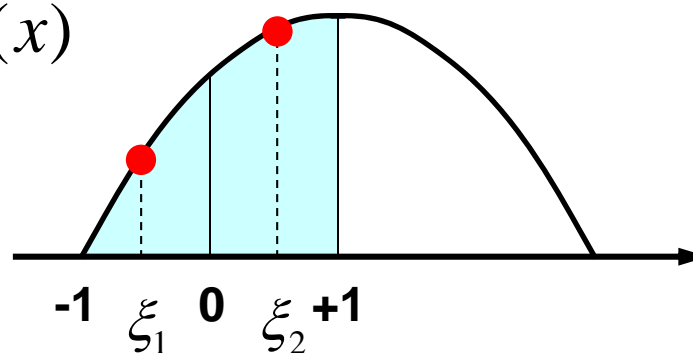
$$f(x) = \sin(x)$$

$$X_1 = 0, \quad X_2 = \frac{\pi}{4}, \quad X_3 = \frac{\pi}{2}$$

$$h = X_2 - X_1 = X_3 - X_2 = \frac{\pi}{4}$$

$$S = \frac{h}{3} [f(X_1) + 4f(X_2) + f(X_3)] = 1.0023$$

Gauss



$$\xi_1, \xi_2 = \pm 0.5773502692$$

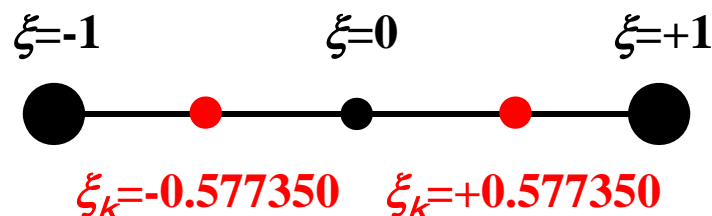
$$S = \int_0^{\pi/2} f(x) dx = \int_{-1}^{+1} f(\xi) h d\xi$$

$$\cong h \sum_{k=1}^2 W_k \cdot f(\xi_k) = 0.99847$$

ガウスの積分公式

- 無次元化された自然座標系 $[-1,+1]$ を対象とする。
- m 個の積分点を使用すると $(2m-1)$ 次の関数までは近似可能（従って1次, 2次の内挿関数（形状関数）を使用するときは, $m=2$ で十分）

$$\int_{-1}^{+1} f(\xi) d\xi = \sum_{k=1}^m [w_k \cdot f(\xi_k)]$$



$$m = 1 \quad \xi_k = 0.00, w_k = 2.00$$

$$m = 2 \quad \xi_k = \pm 0.577350, w_k = 1.00$$

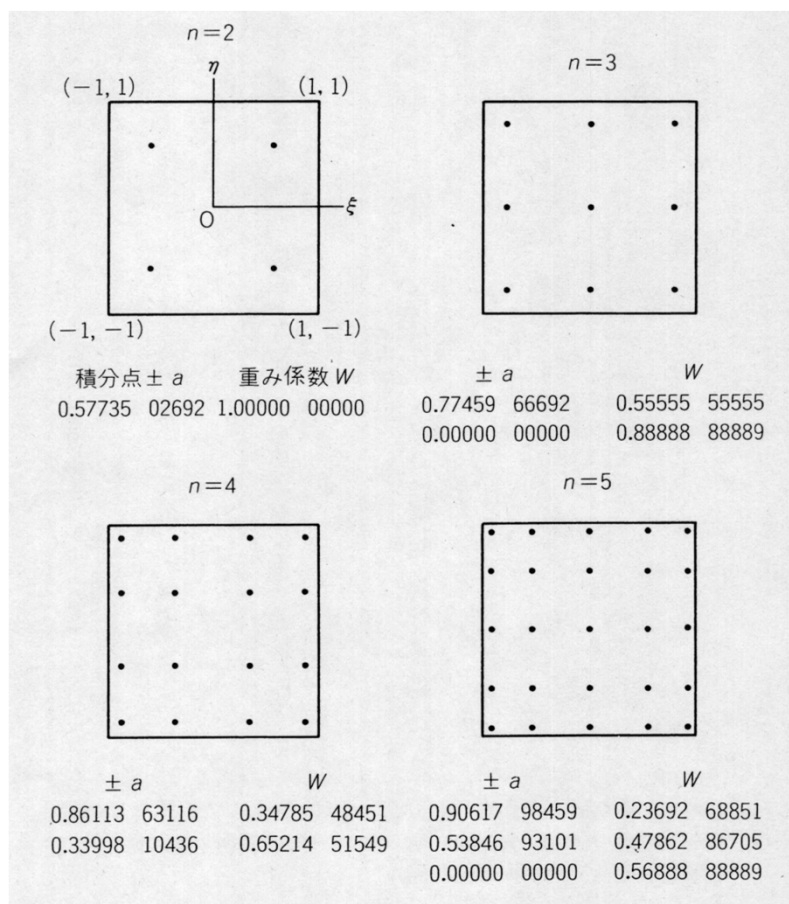
$$m = 3 \quad \xi_k = 0.00, w_k = 8/9$$

$$\xi_k = \pm 0.774597, w_k = 5/9$$

Gaussian Quadrature

ガウスの積分公式

- 自然座標系 (ξ, η, ζ) 上で定義 \Rightarrow ガウス積分公式が使える (三次元)



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

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

L, M, N : ξ, η, ζ 方向の積分点数

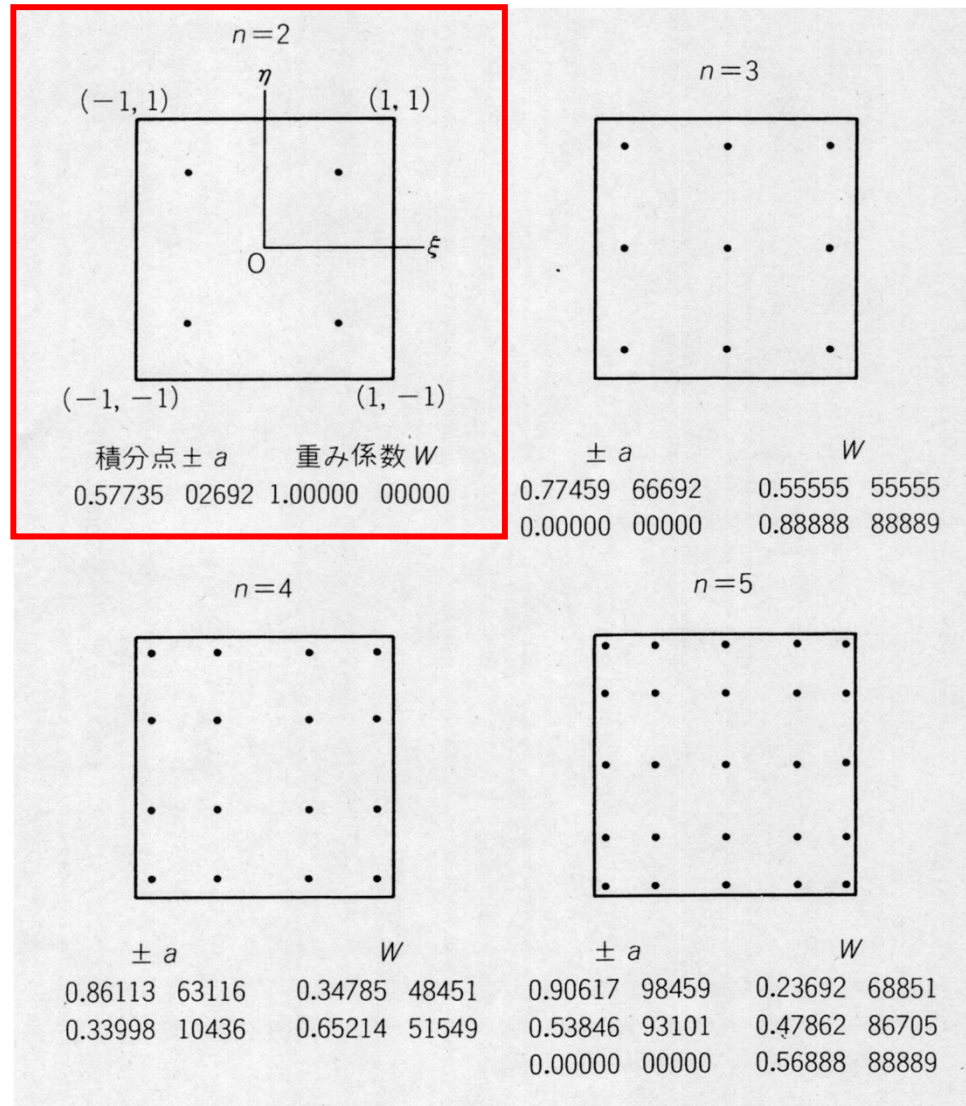
(ξ_i, η_j, ζ_k) : 積分点の座標値

W_i, W_j, W_k : 積分点での重み係数

Gaussian Quadrature

ガウスの積分公式

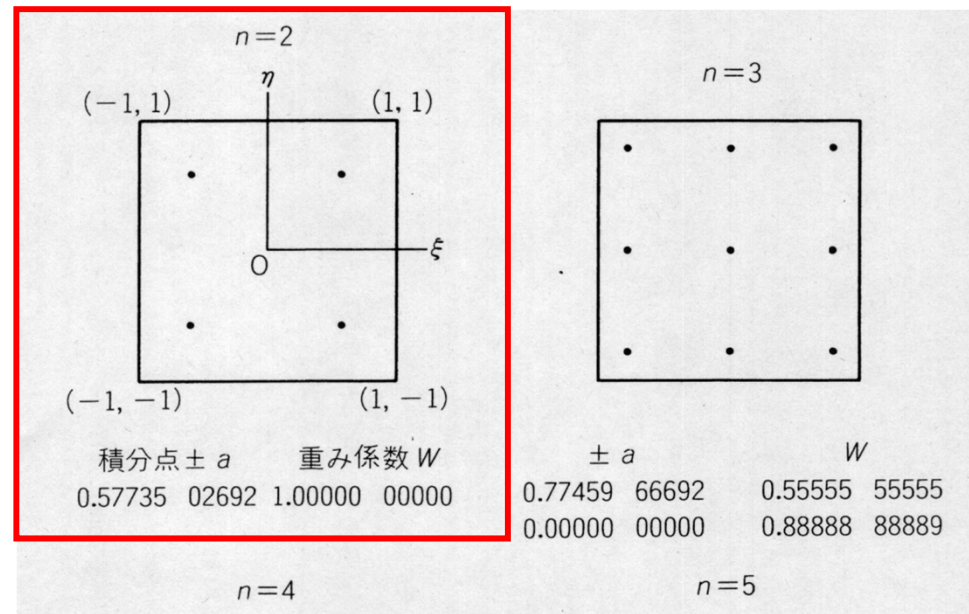
この組み合わせがよく使われる, 二次元では4点における f の値を計算して足せば良い(三次元では8点)



Gaussian Quadrature

ガウスの積分公式

この組み合わせがよく使われる, 二次元では4点におけるfの値を計算して足せば良い(三次元では8点)



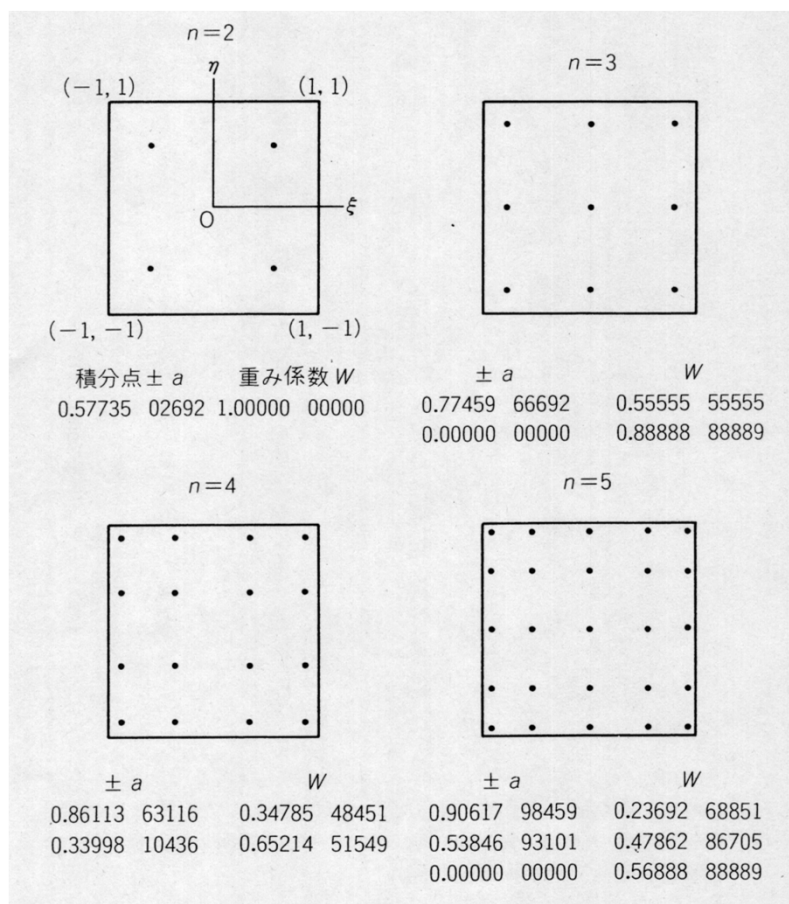
$$I = \int_{-1}^{+1} \int_{-1}^{+1} f(\xi, \eta) d\xi d\eta = \sum_{i=1}^m \sum_{j=1}^n [W_i \cdot W_j \cdot f(\xi_i, \eta_j)]$$

$$= 1.0 \times 1.0 \times f(-0.57735, -0.57735) + 1.0 \times 1.0 \times f(-0.57735, +0.57735) \\ + 1.0 \times 1.0 \times f(+0.57735, +0.57735) + 1.0 \times 1.0 \times f(+0.57735, -0.57735)$$

0.33998 10436 0.65214 51549 0.33998 10436 0.65214 51549
0.00000 00000 0.56888 88889 0.00000 00000 0.56888 88889

あとは積分を求めれば良い

- 自然座標系 (ξ, η, ζ) 上で定義 \Rightarrow ガウス積分公式が使える (三次元) . . . しかし, 微分が



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

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

L, M, N : ξ, η, ζ 方向の積分点数

(ξ_i, η_j, ζ_k) : 積分点の座標値

W_i, W_j, W_k : 積分点での重み係数

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 :

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

$[J]$: 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

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

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

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

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

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

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

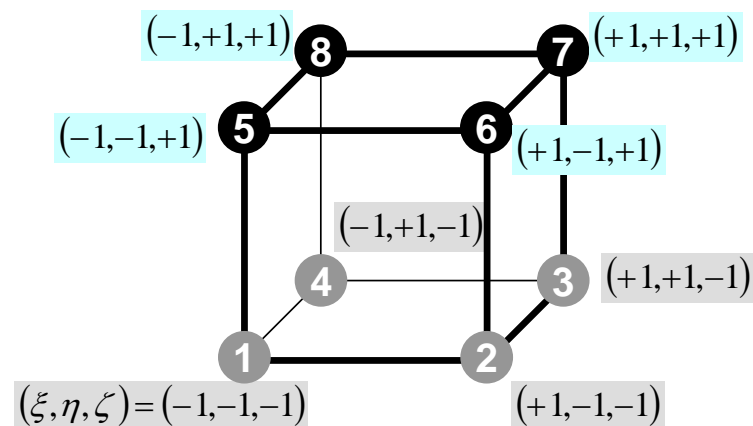
自然座標系における偏微分 (4/4)

- 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

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

要素単位での積分

$$[k_{ij}] \quad (i, j = 1 \dots 8)$$

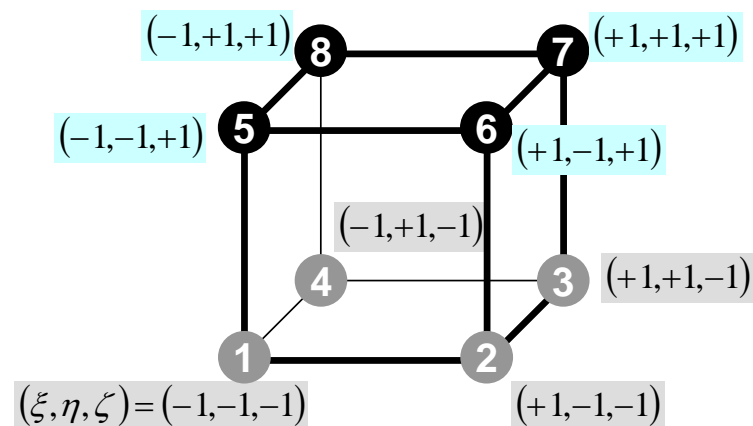


$$k_{ij} = - \int_V \left\{ \lambda \cdot N_{i,x} \cdot N_{j,x} + \lambda \cdot N_{i,y} \cdot N_{j,y} + \lambda \cdot N_{i,z} \cdot N_{j,z} \right\} dV$$

$$= - \int_V \left\{ \lambda \frac{\partial N_i}{\partial x} \frac{\partial N_j}{\partial x} + \lambda \frac{\partial N_i}{\partial y} \frac{\partial N_j}{\partial y} + \lambda \frac{\partial N_i}{\partial z} \frac{\partial N_j}{\partial z} \right\} dV$$

自然座標系での積分

$$[k_{ij}] \quad (i, j = 1 \dots 8)$$



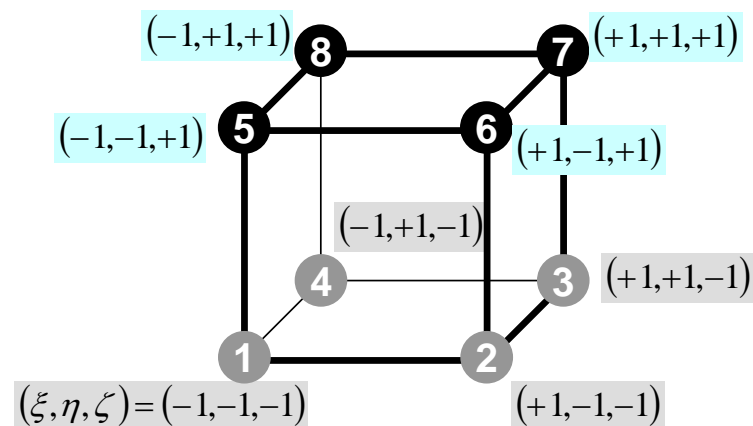
$$- \int_V \left\{ \lambda \frac{\partial N_i}{\partial x} \frac{\partial N_j}{\partial x} + \lambda \frac{\partial N_i}{\partial y} \frac{\partial N_j}{\partial y} + \lambda \frac{\partial N_i}{\partial z} \frac{\partial N_j}{\partial z} \right\} dV =$$

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

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

ガウスの積分公式

$$[k_{ij}] \quad (i, j = 1 \dots 8)$$



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

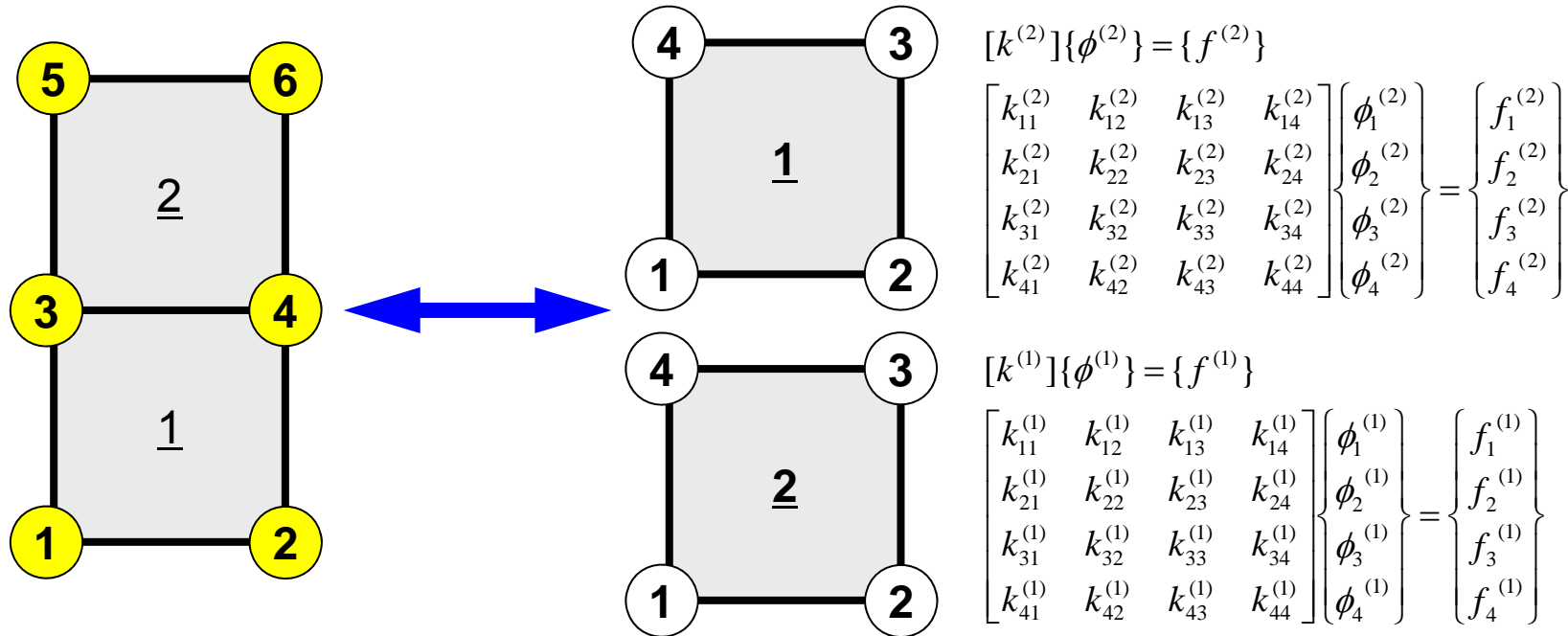
$$I = \int_{-1}^{+1} \int_{-1}^{+1} \int_{-1}^{+1} f(\xi, \eta, \zeta) d\xi d\eta d\zeta$$

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

残りの手順

- ここまでで、要素ごとの積分が可能となる。
- あとは：
 - 全体マトリクスへの重ね合わせ
 - 境界条件処理
 - 連立一次方程式を解く . . .
- 詳細はプログラムの内容を解説しながら説明する。

要素⇒全体マトリクス重ね合わせ



$$[K]\{\Phi\} = \{F\}$$

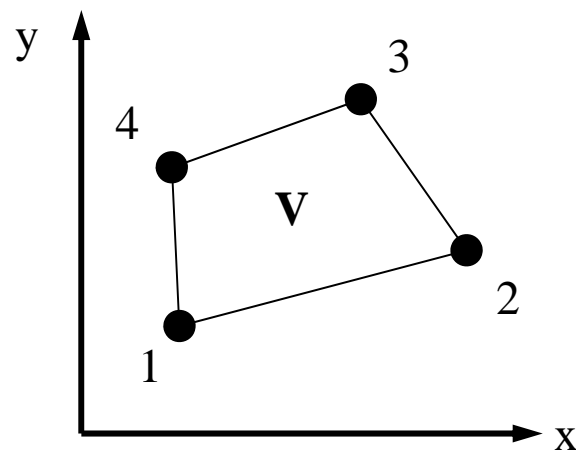
$$\begin{bmatrix} D_1 & X & X & X & & & \\ X & D_2 & X & X & & & \\ X & X & D_3 & X & X & X & \\ X & X & X & D_4 & X & X & \\ & & X & X & D_5 & X & \\ & & X & X & X & D_6 & \end{bmatrix} \begin{Bmatrix} \Phi_1 \\ \Phi_2 \\ \Phi_3 \\ \Phi_4 \\ \Phi_5 \\ \Phi_6 \end{Bmatrix} = \begin{Bmatrix} B_1 \\ B_2 \\ B_3 \\ B_4 \\ B_5 \\ B_6 \end{Bmatrix}$$

- 三次元要素の定式化
- 三次元弾性力学方程式
 - ガラーキン法
 - 要素マトリクス生成
 - 演習

- プログラムの実行
- データ構造
- プログラムの構成

演習

- ガウスの積分公式を使用して以下の四角形の面積を求めよ（プログラムを作って，計算機で計算してください）



1: (1.0, 1.0)
2: (4.0, 2.0)
3: (3.0, 5.0)
4: (2.0, 4.0)

$$I = \int_V dV = \int_{-1}^{+1} \int_{-1}^{+1} \det|J| d\xi d\zeta$$

ヒント (1/2)

- 座標値によってヤコビアン（ヤコビの行列）を計算
- ガウスの積分公式（ $n=2$ ）に代入する。

$$I = \int_{-1}^{+1} \int_{-1}^{+1} f(\xi, \eta) d\xi d\eta = \sum_{i=1}^m \sum_{j=1}^n [W_i \cdot W_j \cdot f(\xi_i, \eta_j)]$$

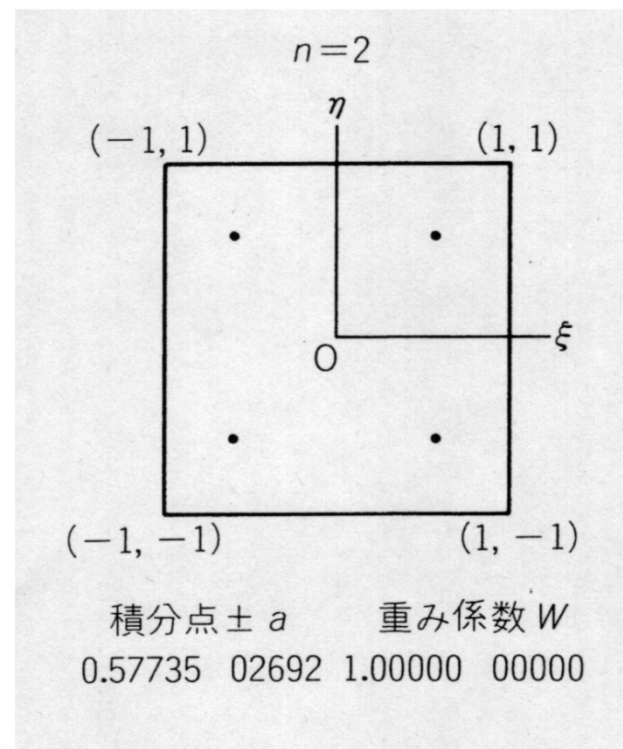
```

implicit REAL*8 (A-H,O-Z)
real*8 W(2)
real*8 POI(2)

W(1)= 1.0d0
W(2)= 1.0d0
POI(1)= -0.5773502692d0
POI(2)= +0.5773502692d0

SUM= 0.d0
do jp= 1, 2
do ip= 1, 2
    FC = F(POI(ip), POI(jp))
    SUM= SUM + W(ip)*W(jp)*FC
enddo
enddo

```



ヒント (2/2)

$$[J] = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} \end{bmatrix}, \quad \det|J| = \frac{\partial x}{\partial \xi} \cdot \frac{\partial y}{\partial \eta} - \frac{\partial y}{\partial \xi} \cdot \frac{\partial x}{\partial \eta}$$

$$\frac{\partial x}{\partial \xi} = \frac{\partial}{\partial \xi} \left(\sum_{i=1}^4 N_i x_i \right) = \sum_{i=1}^4 \frac{\partial N_i}{\partial \xi} x_i, \quad \frac{\partial y}{\partial \xi} = \frac{\partial}{\partial \xi} \left(\sum_{i=1}^4 N_i y_i \right) = \sum_{i=1}^4 \frac{\partial N_i}{\partial \xi} y_i,$$

$$\frac{\partial x}{\partial \eta} = \frac{\partial}{\partial \eta} \left(\sum_{i=1}^4 N_i x_i \right) = \sum_{i=1}^4 \frac{\partial N_i}{\partial \eta} x_i, \quad \frac{\partial y}{\partial \eta} = \frac{\partial}{\partial \eta} \left(\sum_{i=1}^4 N_i y_i \right) = \sum_{i=1}^4 \frac{\partial N_i}{\partial \eta} y_i$$

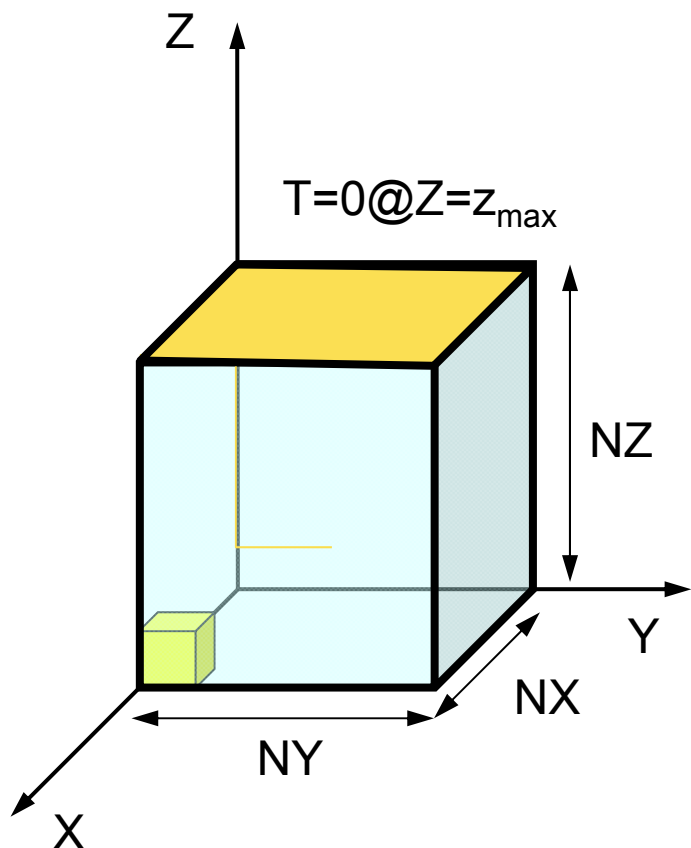
$$N_1(\xi, \eta) = \frac{1}{4}(1 - \xi)(1 - \eta), \quad N_2(\xi, \eta) = \frac{1}{4}(1 + \xi)(1 - \eta),$$

$$N_3(\xi, \eta) = \frac{1}{4}(1 + \xi)(1 + \eta), \quad N_4(\xi, \eta) = \frac{1}{4}(1 - \xi)(1 + \eta)$$

- 三次元要素の定式化
- 三次元弾性力学方程式
 - ガラーキン法
 - 要素マトリクス生成
- プログラムの実行
- データ構造
- プログラムの構成

対象とする問題：三次元定常熱伝導

$$\frac{\partial}{\partial x} \left(\lambda \frac{\partial T}{\partial x} \right) + \frac{\partial}{\partial y} \left(\lambda \frac{\partial T}{\partial y} \right) + \frac{\partial}{\partial z} \left(\lambda \frac{\partial T}{\partial z} \right) + \dot{Q}(x, y, z) = 0$$



- 定常熱伝導＋発熱
- 一様な熱伝導率 λ
- 直方体
 - 一辺長さ1の立方体（六面体）要素
 - 各方向に $NX \cdot NY \cdot NZ$ 個
- 境界条件
 - $T=0@Z=z_{\max}$
- 体積当たり発熱量は位置（メッシュの中心の座標 x_c, y_c ）に依存
 - $\dot{Q}(x, y, z) = QVOL|x_c + y_c|$

インストール (Cygwinでは*.exe)

インストール

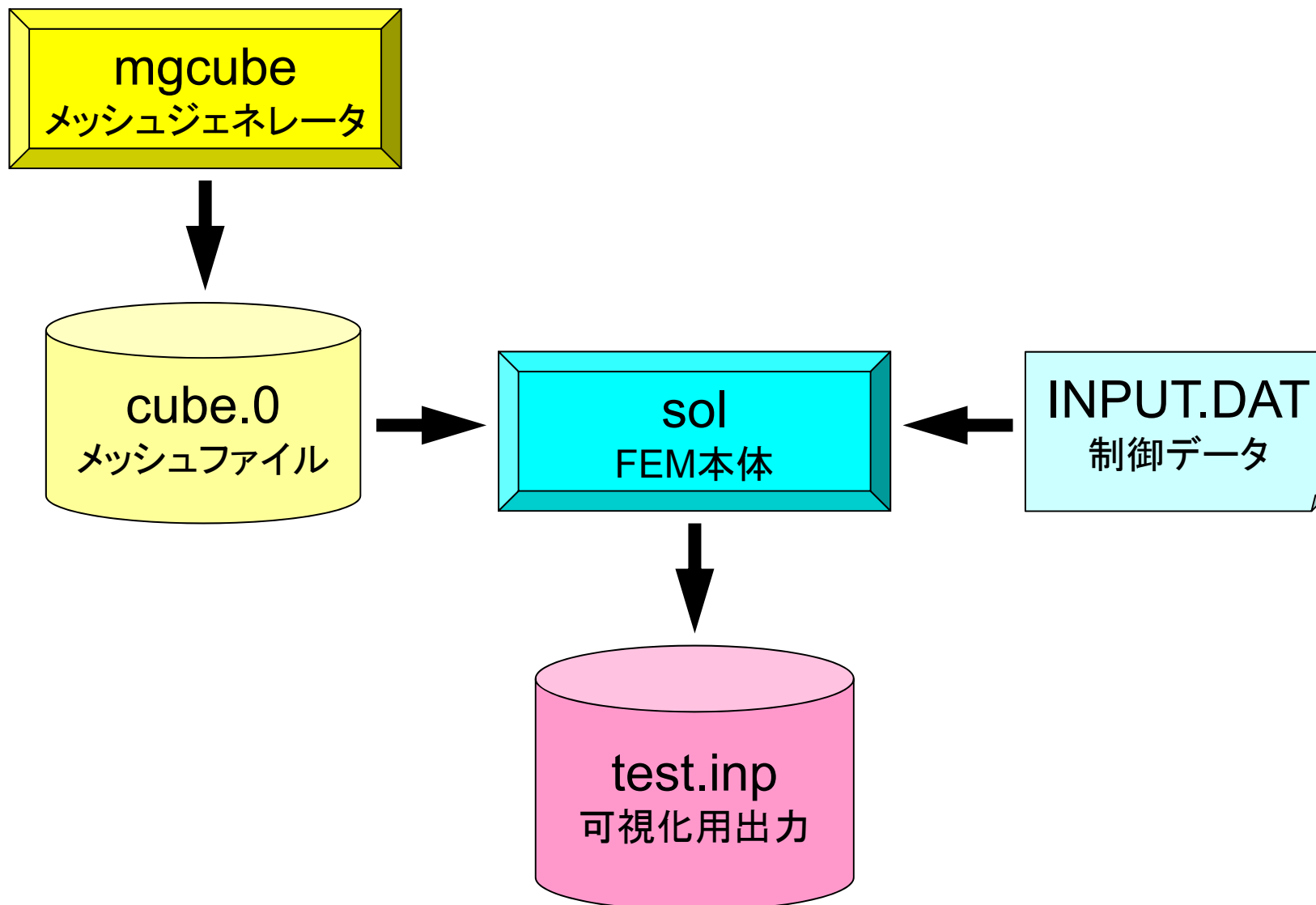
```
>$ cd <$P-TOP>/fem3d/src  
>$ make  
>$ ls ../run/sol  
sol
```

メッシュジェネレータインストール

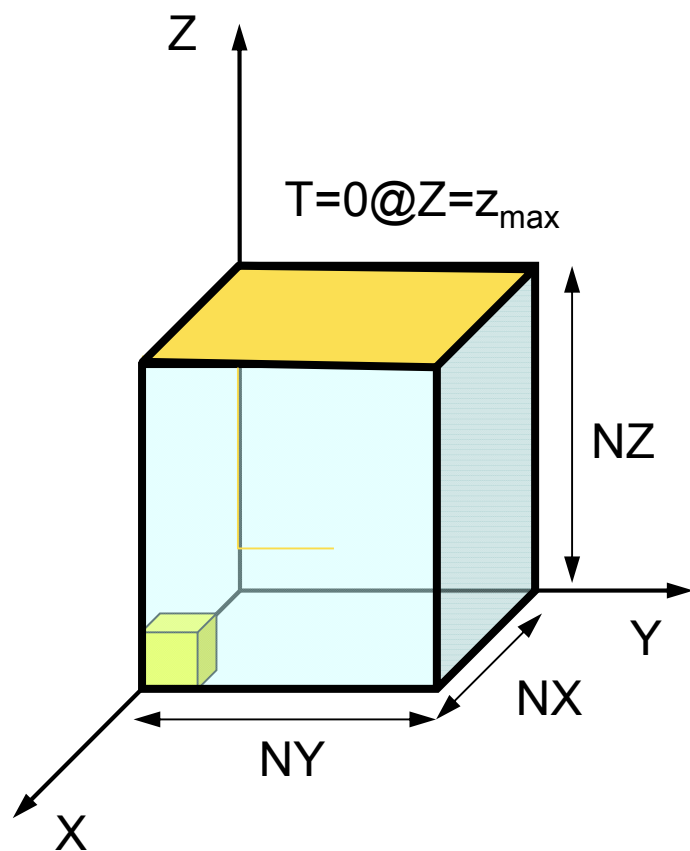
```
>$ cd <$P-TOP>/fem3d/run  
>$ g95 -O3 mgcube.f -o mgcube
```

計算の流れ

メッシュ生成⇒計算, ファイル名称固定



メッシュ生成 (Cygwinでは mgcube.exe)



```
>$ cd <$P-TOP>/fem3d/run  
>$ ./mgcube
```

```
NX, NY, NZ
```

```
20, 20, 20
```

```
>$ ls cube.0
```

```
cube.0
```

← 各辺長さを
訊いてくる
← このように
入れてみる

生成を確認

制御ファイル：INPUT.DAT

INPUT.DAT

```
cube.0      fname
2000        ITER
1.0 1.0     COND, QVOL
1.0e-08     RESID
```

- fname : メッシュファイル名
- ITER : 反復回数上限
- COND : 熱伝導率
- QVOL : 体積当たり発熱量係数
- RESID : 反復法の収束判定値

$$\frac{\partial}{\partial x} \left(\lambda \frac{\partial T}{\partial x} \right) + \frac{\partial}{\partial y} \left(\lambda \frac{\partial T}{\partial y} \right) + \frac{\partial}{\partial z} \left(\lambda \frac{\partial T}{\partial z} \right) + \dot{Q}(x, y, z) = 0$$

$$\dot{Q}(x, y, z) = QVOL |x_c + y_c|$$

実行 (Cygwinではsol.exe

```
>$ cd <$P-TOP>/fem3d/run  
>$ ./sol
```

```
>$ ls test.inp          生成を確認  
test.inp
```

ParaView

- <http://www.paraview.org/>
- ファイルを開く
- 図の表示
- イメージファイルの保存
- <http://nkl.cc.u-tokyo.ac.jp/class/HowtouseParaView.pdf>

UCD Format (1/3)

Unstructured Cell Data

要素の種類

点

線

三角形

四角形

四面体

角錐

三角柱

六面体

二次要素

線2

三角形2

四角形2

四面体2

角錐2

三角柱2

六面体2

キーワード

pt

line

tri

quad

tet

pyr

prism

hex

line2

tri2

quad2

tet2

pyr2

prism2

hex2

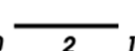
点



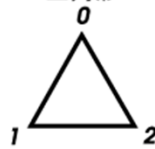
線



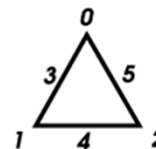
線2



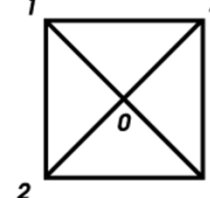
三角形



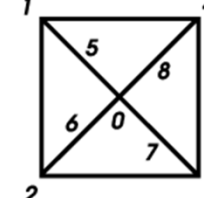
三角形2



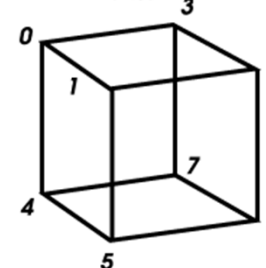
四角錐



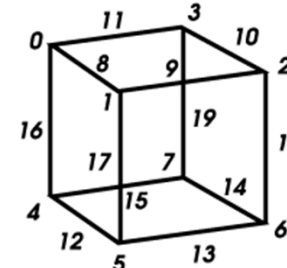
四角錐2



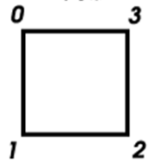
六面体



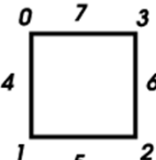
六面体2



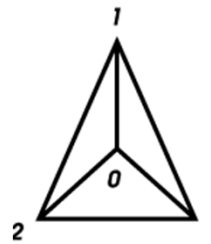
四角形



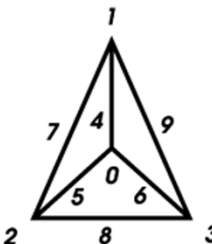
四角形2



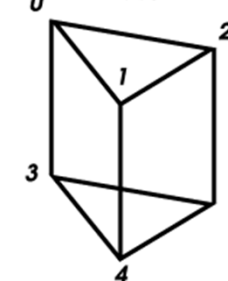
三角錐



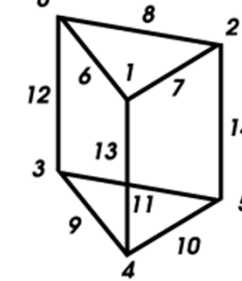
三角錐2



三角柱



三角柱2



UCD Format (2/3)

- Originally for AVS, microAVS
- Extension of the UCD file is “inp”
- There are two types of formats. Only old type can be read by ParaView.

UCD Format (3/3): Old Format

(全節点数) (全要素数) (各節点のデータ数) (各要素のデータ数) (モデルのデータ数)

(節点番号1) (X座標) (Y座標) (Z座標)

(節点番号2) (X座標) (Y座標) (Z座標)

·
·
·

(要素番号1) (材料番号) (要素の種類) (要素を構成する節点のつながり)

(要素番号2) (材料番号) (要素の種類) (要素を構成する節点のつながり)

·
·
·

(節点のデータ成分数) (成分1の構成数) (成分2の構成数) ... (各成分の構成数)

(節点データ成分1のラベル), (単位)

(節点データ成分2のラベル), (単位)

·
·
·

(各節点データ成分のラベル), (単位)

(節点番号1) (節点データ1) (節点データ2)

(節点番号2) (節点データ1) (節点データ2)

·
·
·

(要素のデータ成分数) (成分1の構成数) (成分2の構成数) ... (各成分の構成数)

(要素データ成分1のラベル), (単位)

(要素データ成分2のラベル), (単位)

·
·
·

(各要素データ成分のラベル), (単位)

(要素番号1) (要素データ1) (要素データ2)

(要素番号2) (要素データ1) (要素データ2)

·
·
·

- 三次元要素の定式化
- 三次元弾性力学方程式
 - ガラーキン法
 - 要素マトリクス生成

- プログラムの実行
- データ構造
- プログラムの構成

メッシュファイル構成：cube.0

番号は「1」から始まっている

- 節点データ
 - 節点数
 - 節点番号, 座標
- 要素データ
 - 要素数
 - 要素タイプ
 - 要素番号, 材料番号, コネクティビティ
- 節点グループデータ
 - グループ数
 - グループ内節点数
 - グループ名
 - グループ内節点

メッシュ生成コード : mgcube.f (1/5)

```

implicit REAL*8 (A-H, O-Z)
real(kind=8), dimension(:, :), allocatable :: X, Y
real(kind=8), dimension(:, :), allocatable :: X0, Y0
character(len=80) :: GRIDFILE, HHH
integer, dimension(:, :), allocatable :: IW

|C
|C +-----+
|C | INIT. |
|C +-----+
|C
|C==

```

write (*,*) 'NX, NY, NZ'
read (*,*) NX, NY, NZ

NXP1= NX + 1 X方向節点数
NYP1= NY + 1 Y方向節点数
NZP1= NZ + 1 Z方向節点数

DX= 1. d0

INODTOT= NXP1*NYP1*NZP1 総節点数
ICELTOT= NX *NY *NZ 総要素数
IBNODTOT= NXP1*NYP1 XY平面の節点数

allocate (IW(INODTOT, 4))
IW= 0

メッシュ生成コード : mgcube.f (2/5)

```

icou= 0
ib = 1
do k= 1, NZP1
do j= 1, NYP1
i= 1
icou= icou + 1
ii = (k-1)*IBNODTOT + (j-1)*NXP1 + i
IW(icou, ib)= ii
enddo
enddo

```

```

icou= 0
ib = 2
do k= 1, NZP1
j= 1
do i= 1, NXP1
icou= icou + 1
ii = (k-1)*IBNODTOT + (j-1)*NXP1 + i
IW(icou, ib)= ii
enddo
enddo

```

```

icou= 0
ib = 3
k= 1
do j= 1, NYP1
do i= 1, NXP1
icou= icou + 1
ii = (k-1)*IBNODTOT + (j-1)*NXP1 + i
IW(icou, ib)= ii
enddo
enddo

```

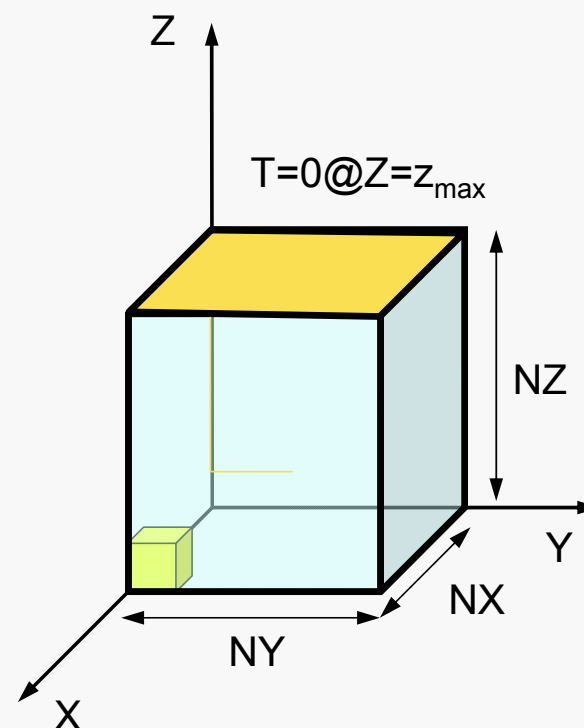
```

icou= 0
ib = 4
k= NZP1
do j= 1, NYP1
do i= 1, NXP1
icou= icou + 1
ii = (k-1)*IBNODTOT + (j-1)*NXP1 + i
IW(icou, ib)= ii
enddo
enddo

```

```
!C===
```

X=Xminの節点をIW(ib, 1)に格納
(ib=1, NYP1*NZP1)



メッシュ生成コード : mgcube.f (2/5)

```

icou= 0
ib = 1
do k= 1, NZP1
do j= 1, NYP1
i= 1
icou= icou + 1
ii = (k-1)*IBNODTOT + (j-1)*NXP1 + i
IW(icou, ib)= ii
enddo
enddo

```

```

icou= 0
ib = 2
do k= 1, NZP1
j= 1
do i= 1, NXP1
icou= icou + 1
ii = (k-1)*IBNODTOT + (j-1)*NXP1 + i
IW(icou, ib)= ii
enddo
enddo

```

```

icou= 0
ib = 3
k= 1
do j= 1, NYP1
do i= 1, NXP1
icou= icou + 1
ii = (k-1)*IBNODTOT + (j-1)*NXP1 + i
IW(icou, ib)= ii
enddo
enddo

```

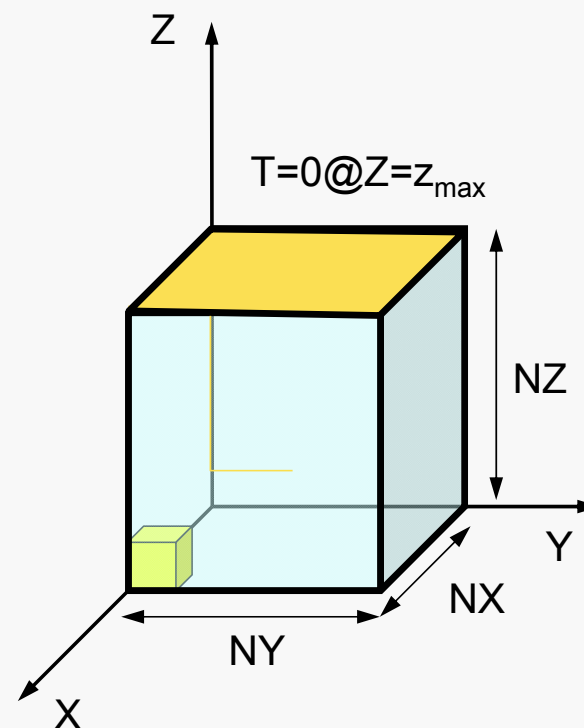
```

icou= 0
ib = 4
k= NZP1
do j= 1, NYP1
do i= 1, NXP1
icou= icou + 1
ii = (k-1)*IBNODTOT + (j-1)*NXP1 + i
IW(icou, ib)= ii
enddo
enddo

```

```
!C===
```

Y=Yminの節点をIW(ib, 2)に格納
(ib=1, NXP1*NZP1)



メッシュ生成コード : mgcube.f (2/5)

```

icou= 0
ib = 1
do k= 1, NZP1
do j= 1, NYP1
i= 1
icou= icou + 1
ii = (k-1)*IBNODTOT + (j-1)*NXP1 + i
IW(icou, ib)= ii
enddo
enddo

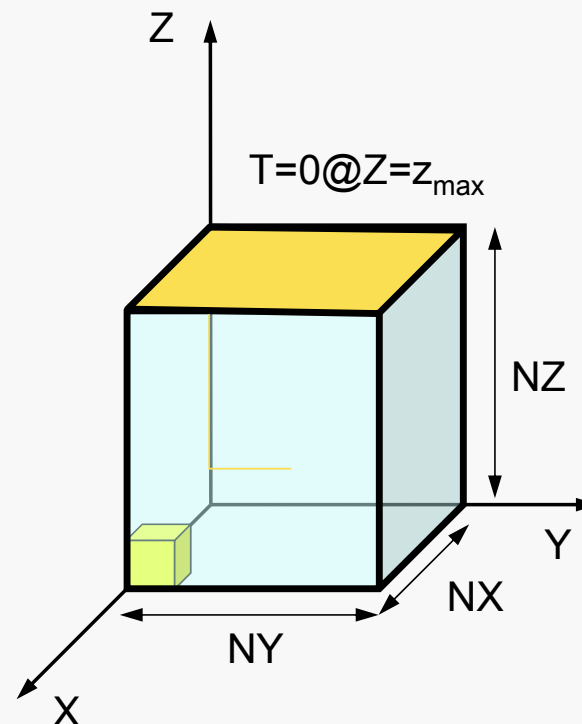
icou= 0
ib = 2
do k= 1, NZP1
j= 1
do i= 1, NXP1
icou= icou + 1
ii = (k-1)*IBNODTOT + (j-1)*NXP1 + i
IW(icou, ib)= ii
enddo
enddo

icou= 0
ib = 3
k= 1
do j= 1, NYP1
do i= 1, NXP1
icou= icou + 1
ii = (k-1)*IBNODTOT + (j-1)*NXP1 + i
IW(icou, ib)= ii
enddo
enddo

icou= 0
ib = 4
k= NZP1
do j= 1, NYP1
do i= 1, NXP1
icou= icou + 1
ii = (k-1)*IBNODTOT + (j-1)*NXP1 + i
IW(icou, ib)= ii
enddo
enddo

```

!C===



Z=Zminの節点をIW(ib, 3)に格納
(ib=1, NXP1*NYP1)

Z=Zmaxの節点をIW(ib, 4)に格納
(ib=1, NXP1*NYP1)

メッシュ生成コード : mqcube.f (3/5)

```
!C
!C +-----+
!C | GeoFEM data |
!C +-----+
!C===
      NN= 0
      write (*,*) 'GeoFEM gridfile name ?'
      GRIDFILE= 'cube.0'

      open (12, file= GRIDFILE, status='unknown', form='formatted')
      write(12, '(10i10)') INODTOT

      icou= 0
      do k= 1, NZP1
      do j= 1, NYP1
      do i= 1, NXP1
         XX= dfloat(i-1)*DX
         YY= dfloat(j-1)*DX
         ZZ= dfloat(k-1)*DX

         icou= icou + 1
         write (12, '(i10,3(1pe16.6))') icou, XX, YY, ZZ
      enddo
      enddo
      enddo

      write(12, '(i10)') ICELTOT

      IELMTYPL= 361
      write(12, '(10i10)') (IELMTYPL, i=1, ICELTOT)
```

節点数
節点番号, 座標

要素数
要素タイプ (使わないデータ)

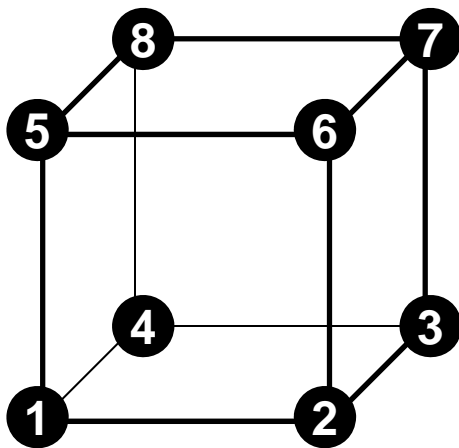
メッシュ生成コード : mgcube.f (4/5)

```

icou= 0
imat= 1
do k= 1, NZ
do j= 1, NY
do i= 1, NX
  icou= icou + 1
  in1 = (k-1)*IBNODTOT + (j-1)*NXP1 + i
  in2 = in1 + 1
  in3 = in2 + NXP1
  in4 = in3 - 1
  in5 = in1 + IBNODTOT
  in6 = in2 + IBNODTOT
  in7 = in3 + IBNODTOT
  in8 = in4 + IBNODTOT
  & write (12, '(10i10)') icou, imat, in1, in2, in3, in4,
    & in5, in6, in7, in8
enddo
enddo
enddo

```

& imat : 材料番号 (=1)



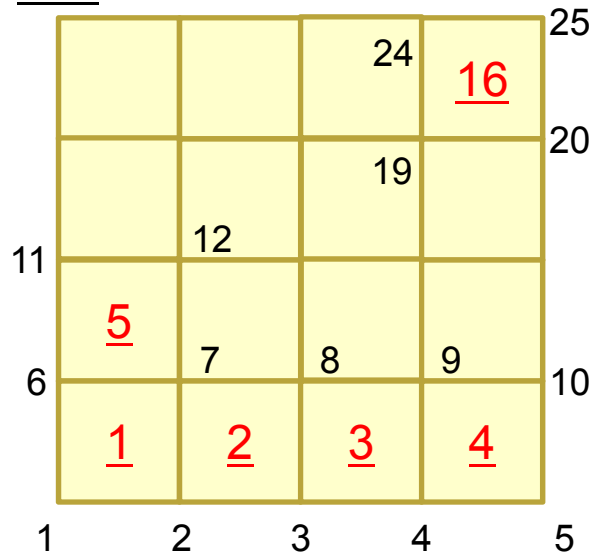
cube.0 : 要素データ

1	1	1	2	7	6	26	27	32	31
2	1	2	3	8	7	27	28	33	32
3	1	3	4	9	8	28	29	34	33
4	1	4	5	10	9	29	30	35	34
5	1	6	7	12	11	31	32	37	36
6	1	7	8	13	12	32	33	38	37
7	1	8	9	14	13	33	34	39	38
8	1	9	10	15	14	34	35	40	39
9	1	11	12	17	16	36	37	42	41
10	1	12	13	18	17	37	38	43	42
11	1	13	14	19	18	38	39	44	43
12	1	14	15	20	19	39	40	45	44
13	1	16	17	22	21	41	42	47	46
(途中省略)									
42	1	62	63	68	67	87	88	93	92
43	1	63	64	69	68	88	89	94	93
44	1	64	65	70	69	89	90	95	94
45	1	66	67	72	71	91	92	97	96
46	1	67	68	73	72	92	93	98	97
47	1	68	69	74	73	93	94	99	98
48	1	69	70	75	74	94	95	100	99
49	1	76	77	82	81	101	102	107	106
50	1	77	78	83	82	102	103	108	107
51	1	78	79	84	83	103	104	109	108
52	1	79	80	85	84	104	105	110	109
53	1	81	82	87	86	106	107	112	111
54	1	82	83	88	87	107	108	113	112
55	1	83	84	89	88	108	109	114	113
56	1	84	85	90	89	109	110	115	114
57	1	86	87	92	91	111	112	117	116
58	1	87	88	93	92	112	113	118	117
59	1	88	89	94	93	113	114	119	118
60	1	89	90	95	94	114	115	120	119
61	1	91	92	97	96	116	117	122	121
62	1	92	93	98	97	117	118	123	122
63	1	93	94	99	98	118	119	124	123
64	1	94	95	100	99	119	120	125	124

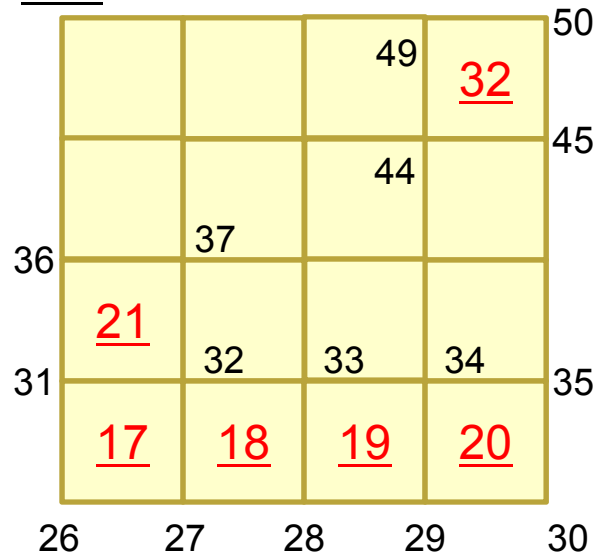
$NX=NY=NZ=4$, $NXP1=NYP1=NZP1=5$

$ICELTOT=64$, $INODTOT=125$, $IBNODTOT=25$

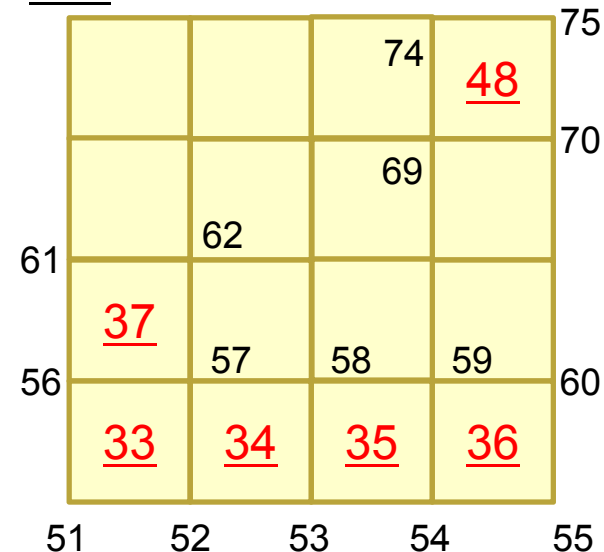
k=1



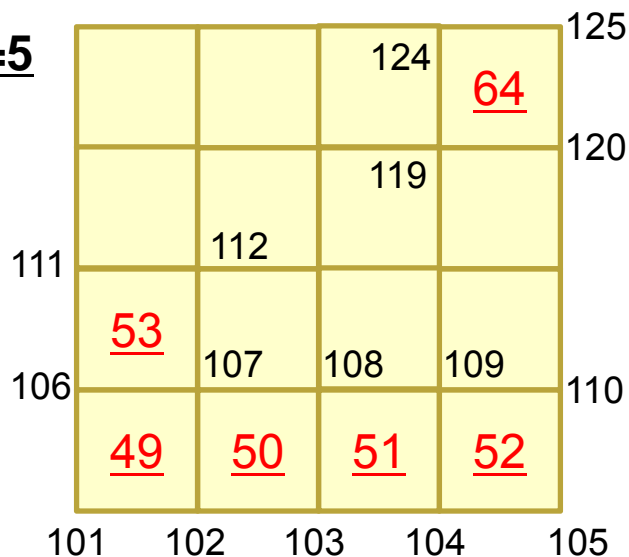
k=2



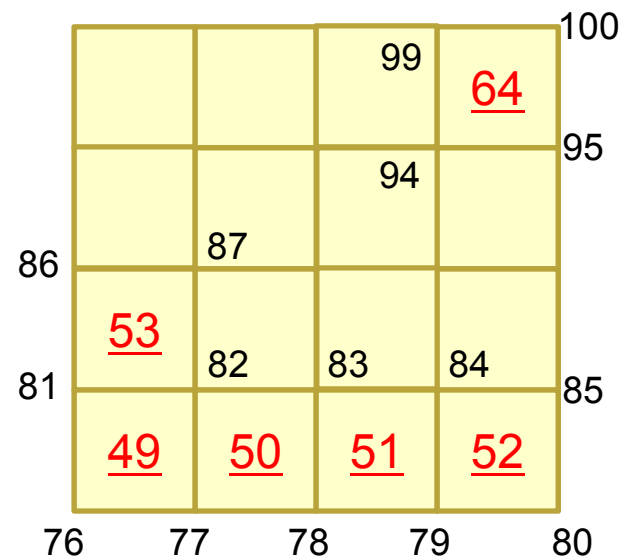
k=3



k=5



k=4



メッシュ生成コード : mgcube.f (5/5)

```
IGTOT= 4
```

```
IBT1= NYP1*NZP1
IBT2= NXP1*NZP1 + IBT1
IBT3= NXP1*NYP1 + IBT2
IBT4= NXP1*NYP1 + IBT3
```

```
write (12, '(10i10)') IGTOT
write (12, '(10i10)') IBT1, IBT2, IBT3, IBT4
```

```
HHH= 'Xmin'
write (12, '(a80)') HHH
write (12, '(10i10)') (IW(ii,1), ii=1, NYP1*NZP1)
HHH= 'Ymin'
write (12, '(a80)') HHH
write (12, '(10i10)') (IW(ii,2), ii=1, NXP1*NZP1)
HHH= 'Zmin'
write (12, '(a80)') HHH
write (12, '(10i10)') (IW(ii,3), ii=1, NXP1*NYP1)
HHH= 'Zmax'
write (12, '(a80)') HHH
write (12, '(10i10)') (IW(ii,4), ii=1, NXP1*NYP1)
```

```
deallocate (IW)
```

(以下略)

```
close (12)
```

```
!C===
```

```
stop
end
```

IGTOT	グループ総数 (Xmin, Ymin, Zmin, Zmax)
IBTx	累積数

cube.0 : 節点グループデータ

Xmin	4 25	50	75	100						
	1	6	11	16	21	26	31	36	41	46
Ymin	51 101	56 106	61 111	66 116	71 121	76	81	86	91	96
	1	2	3	4	5	26	27	28	29	30
Zmin	51 101	52 102	53 103	54 104	55 105	76	77	78	79	80
	1	2	3	4	5	6	7	8	9	10
Zmax	11 21	12 22	13 23	14 24	15 25	16	17	18	19	20
	101	102	103	104	105	106	107	108	109	110
	111	112	113	114	115	116	117	118	119	120
	121	122	123	124	125					

(以下 使用せず)

メッシュ生成

- 実は技術的には大きな課題
 - 複雑形状
 - 大規模メッシュ
- 並列化が難しい
- 市販のメッシュ生成アプリケーション
 - FEMAP
 - CADデータとのインタフェース



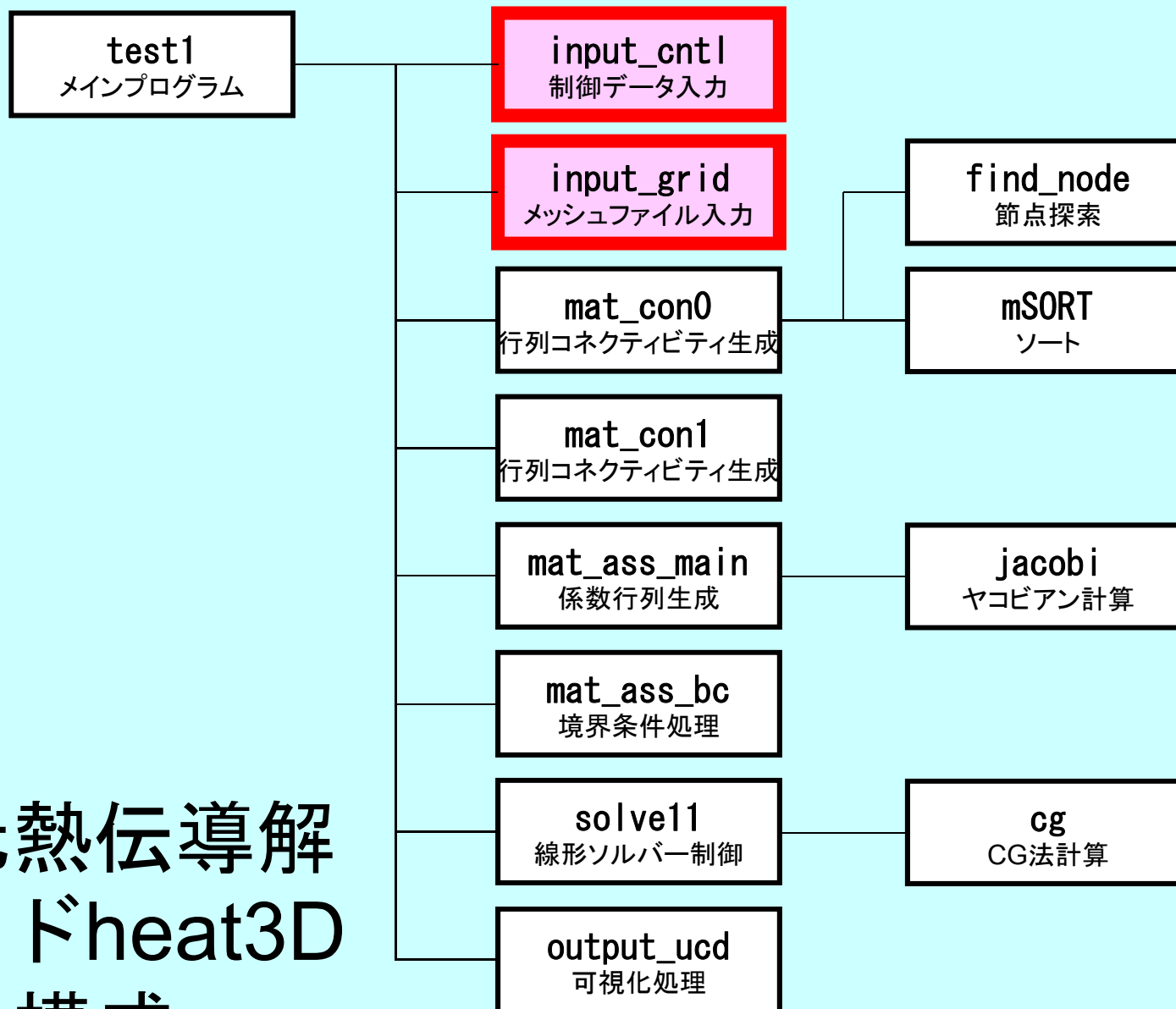
movie

- 三次元要素の定式化
- 三次元弾性力学方程式
 - ガラーキン法
 - 要素マトリクス生成

- プログラムの実行
- データ構造
- プログラムの構成

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, NE: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, NE)
 - 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)



三次元熱伝導解析コードheat3Dの構成

全体処理

```
program heat3D  
  
use solver11  
use pfem_util  
  
implicit REAL*8 (A-H, O-Z)  
  
call INPUT_CNTL  
call INPUT_GRID  
  
call MAT_CONO  
call MAT_CON1  
  
call MAT_ASS_MAIN  
call MAT_ASS_BC  
  
call SOLVE11  
  
call OUTPUT_UCD  
  
end program heat3D
```


Global変数表 : pfem_util.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	(0:NODGRPtot)	I	各節点グループに含まれる節点数 (累積)
NODGRP_ITEM	I	(NODGRP_INDEX (NODGRPtot))	I	節点グループに含まれる節点
NODGRP_NAME	C80	(NODGRPtot)	I	節点グループ名
NLU	I		O	各節点非対角成分数
NPLU	I		O	非対角成分総数
D	R	(N)	O	全体行列 : 対角ブロック
B, X	R	(N)	O	右辺ベクトル, 未知数ベクトル

Global変数表 : pfem_util.f (2/3)

変数名	種別	サイズ	I/O	内 容
AMAT	R	(NPLU)	O	全体行列 : 非零非対角成分
index	I	(0:N)	O	全体行列 : 非零非対角成分数
item	I	(NPLU)	O	全体行列 : 非零非対角成分 (列番号)
INLU	I	(N)	O	各節点の非零非対角成分数
IALU	I	(N, NLU)	O	各節点の非零非対角成分数 (列番号)
IWKX	I	(N, 2)	O	ワーク用配列
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)
pfemIarray	I	(100)	O	諸定数 (整数)
pfemRarray	R	(100)	O	諸定数 (実数)

Global変数表 : pfem_util.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, 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
COND, QVOL	R		I	熱伝導率, 体積当たり発熱量係数

制御ファイル入力 : INPUT_CNTL

```
subroutine INPUT_CNTL
use pfem_util

implicit REAL*8 (A-H, O-Z)

open (11, file= 'INPUT.DAT', status='unknown')
read (11, '(a80)') fname
read (11, *) ITER
read (11, *) COND, QVOL
read (11, *) RESID
close (11)

pfemIarray(1)= ITER
pfemRarray(1)= RESID

return
end
```

INPUT.DAT

cube . 0	fname
2000	ITER
1.0 1.0	COND, QVOL
1.0e-08	RESID

メッシュ入力 : INPUT_GRID (1/2)

```
!C***  
!C*** INPUT_GRID  
!C***  
!C  
  subroutine INPUT_GRID  
  use pfem_util  
  implicit REAL*8 (A-H, O-Z)  
  
  open (11, file= fname, status= 'unknown', form= 'formatted')  
  
!C  
!C--- NODE  
  read (11,*) N  
  NP= N  
  
  allocate (XYZ(N, 3))  
  XYZ= 0. d0  
  do i= 1, N  
    read (11,*) ii, (XYZ(i, kk), kk=1, 3)  
  enddo
```

allocate, deallocate関数 (C言語)

```
#include <stdio.h>
#include <stdlib.h>
void* allocate_vector(int size, int m)
{
    void *a;
    if ( ( a=(void *)malloc( m * size ) ) == NULL ) {
        fprintf(stdout, "Error:Memory does not enough! in vector %n");
        exit(1);
    }
    return a;
}

void deallocate_vector(void *a)
{
    free( a );
}

void** allocate_matrix(int size, int m, int n)
{
    void **aa;
    int i;
    if ( ( aa=(void **)malloc( m * sizeof(void*) ) ) == NULL ) {
        fprintf(stdout, "Error:Memory does not enough! aa in matrix %n");
        exit(1);
    }
    if ( ( aa[0]=(void *)malloc( m * n * size ) ) == NULL ) {
        fprintf(stdout, "Error:Memory does not enough! in matrix %n");
        exit(1);
    }
    for(i=1; i<m; i++) aa[i]=(char*)aa[i-1]+size*n;
    return aa;
}

void deallocate_matrix(void **aa)
{
    free( aa );
}
```

allocateをFORTRAN並みに
簡単にやるための関数

メッシュ入力 : INPUT_GRID (2/2)

```
!C
!C-- ELEMENT
  read (11,*) ICELTOT
  allocate (ICELNOD(ICELTOT,8))
  read (11,'(10i10)') (NTYPE, i= 1, ICELTOT)

  do icel= 1, ICELTOT
    read (11,'(10i10,2i5,8i8)') ii, IMAT, (ICELNOD(icel,k), k=1,8)
  enddo

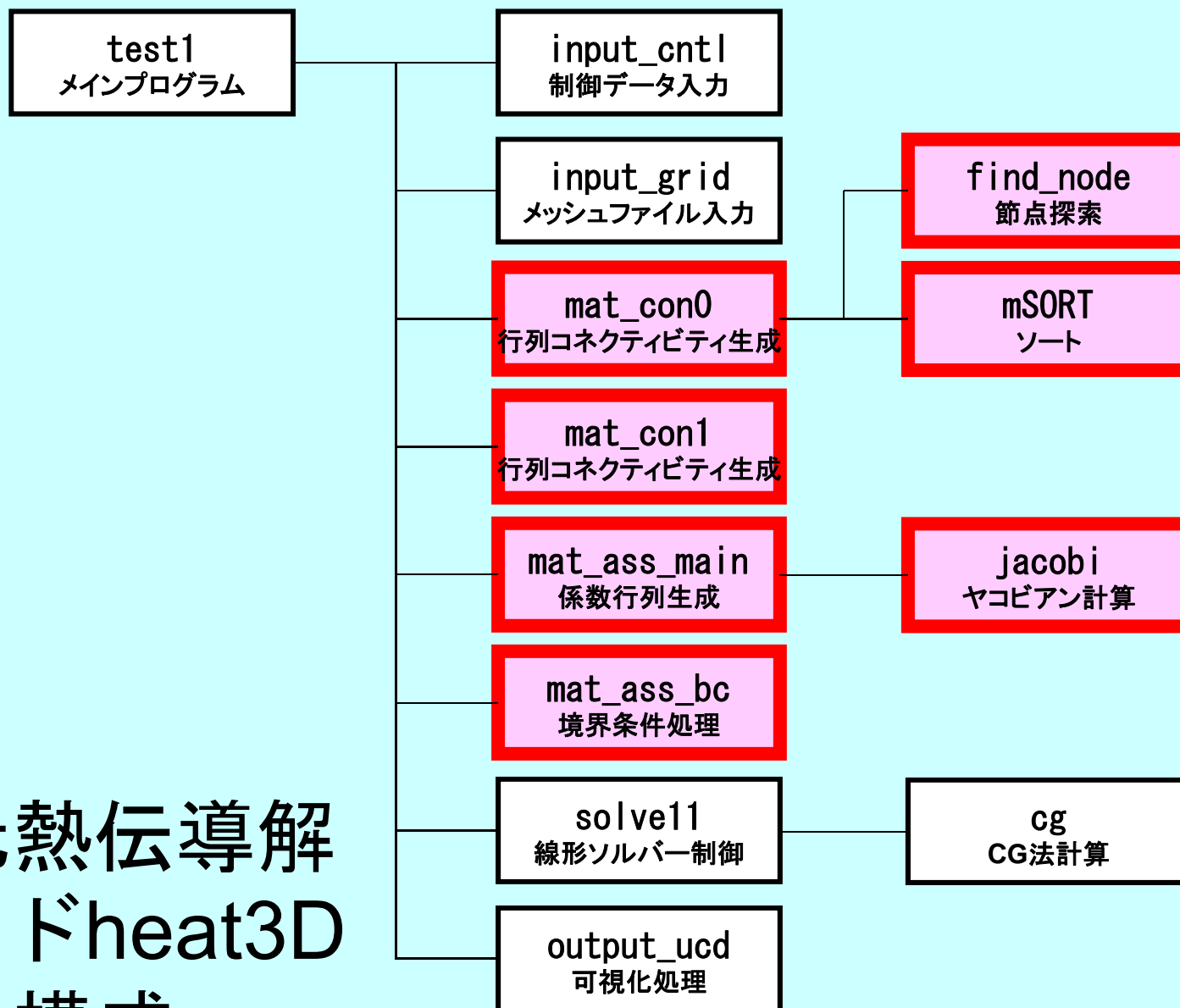
!C
!C-- NODE grp. info.
  read (11,'(10i10)') NODGRPtot
  allocate (NODGRP_INDEX(0:NODGRPtot), NODGRP_NAME(NODGRPtot))
  NODGRP_INDEX= 0

  read (11,'(10i10)') (NODGRP_INDEX(i), i= 1, NODGRPtot)
  nn= NODGRP_INDEX(NODGRPtot)
  allocate (NODGRP_ITEM(nn))

  do k= 1, NODGRPtot
    iS= NODGRP_INDEX(k-1) + 1
    iE= NODGRP_INDEX(k)
    read (11,'(a80)') NODGRP_NAME(k)
    nn= iE - iS + 1
    if (nn.ne.0) then
      read (11,'(10i10)') (NODGRP_ITEM(kk),kk=iS, iE)
    endif
  enddo

  close (11)

  return
end
```



三次元熱伝導解析コードheat3Dの構成

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	(0:NODGRPtot)	I	各節点グループに含まれる節点数 (累積)
NODGRP_ITEM	I	(NODGRP_INDEX (NODGRPtot))	I	節点グループに含まれる節点
NODGRP_NAME	C80	(NODGRPtot)	I	節点グループ名
NLU	I		O	各節点非対角成分数
NPLU	I		O	非対角成分総数
D	R	(N)	O	全体行列 : 対角ブロック
B, X	R	(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
AMAT	R	(NPLU)	O	全体行列 : 非零非対角成分
index	I	(0:N)	O	全体行列 : 非零非対角成分数
item	I	(NPLU)	O	全体行列 : 非零非対角成分 (列番号)
INLU	I	(N)	O	各節点の非零非対角成分数
IALU	I	(N, NLU)	O	各節点の非零非対角成分数 (列番号)
IWKX	I	(N, 2)	O	ワーク用配列
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)
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
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, 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
COND, QVOL	R		I	熱伝導率, 体積当たり発熱量係数

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
 - 非ゼロ非対角成分の数は7~26 (現在の形状)
 - 実際はもっと複雑
 - 前以て, 非ゼロ非対角成分数はわからない



movie

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
 - 非ゼロ非対角成分の数は7~26 (現在の形状)
 - 実際はもっと複雑
 - 前以て, 非ゼロ非対角成分の数はわからない
- INLU(N), IALU(N,NLU) を使って非ゼロ非対角成分数を予備的に勘定する

全体処理

```
program heat3D
use solver11
use pfem_util

implicit REAL*8(A-H, O-Z)

call INPUT_CNTL
call INPUT_GRID

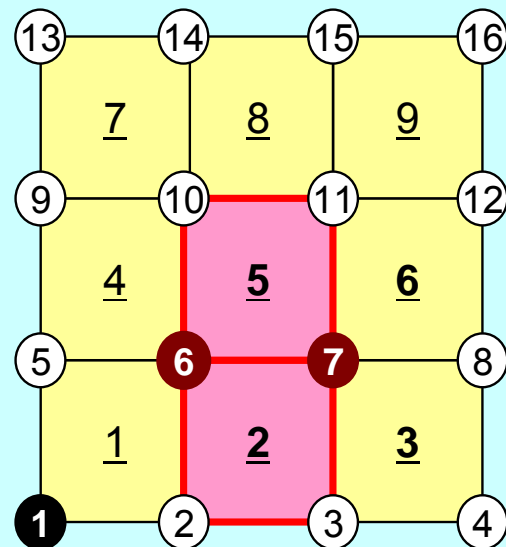
call MAT_CON0
call MAT_CON1

call MAT_ASS_MAIN
call MAT_ASS_BC

call SOLVE11

call OUTPUT_UCD

end program heat3D
```



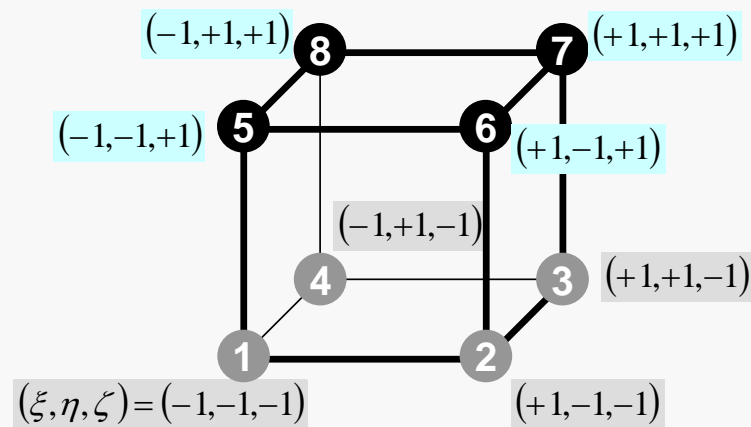
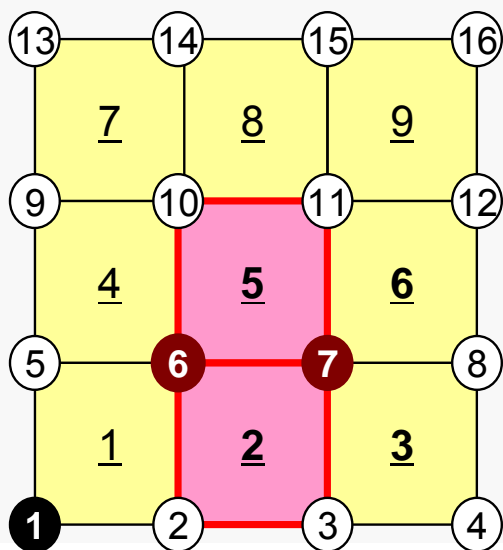
MAT_CON0: INU, IALU生成

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、
  INLU, IALUを生成
  (FIND_NODE)
enddo
```



行列コネクティビティ生成： MAT_CONO (1/4)

```
!C
!C***
!C*** MAT_CONO
!C***
!C
  subroutine MAT_CONO
  use pfem_util
  implicit REAL*8 (A-H, O-Z)

  NLU= 26

  allocate (INLU(N), IALU(N, NLU))

  INLU= 0
  IALU= 0
```

NLU:
各節点における
非ゼロ非対角成分
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_CONO (1/4)

```
!C
!C***
!C*** MAT_CONO
!C***
!C
subroutine MAT_CONO
use pfem_util
implicit REAL*8 (A-H, O-Z)

NLU= 26

allocate (INLU(N), IALU(N, NLU))

INLU= 0
IALU= 0
```

変数名	サイズ	内 容
INLU	(N)	各節点の非零非対角成分数
IALU	(N, NLU)	各節点の非零非対角成分 (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

```

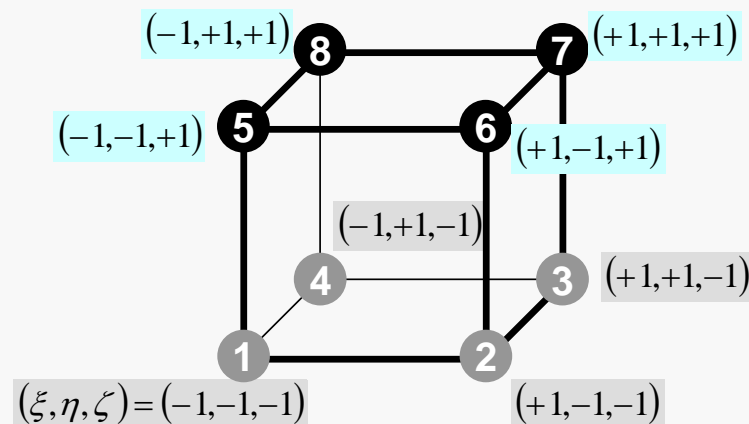
do icel= 1, ICELTOT
  in1= ICELNOD(icel, 1)
  in2= ICELNOD(icel, 2)
  in3= ICELNOD(icel, 3)
  in4= ICELNOD(icel, 4)
  in5= ICELNOD(icel, 5)
  in6= ICELNOD(icel, 6)
  in7= ICELNOD(icel, 7)
  in8= ICELNOD(icel, 8)

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

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

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

```



節点探索 : FIND_TS_NODE

INLU, IAU探索 : 一次元ではこの部分は手動

```

!C
!C***
!C*** FIND_TS_NODE
!C***
!C
      subroutine FIND_TS_NODE (ip1, ip2)
         do kk= 1, INLU(ip1)
            if (ip2. eq. IALU(ip1, kk)) return
         enddo
         icou= INLU(ip1) + 1
         IALU(ip1, icou)= ip2
         INLU(ip1      )= icou
         return
      end subroutine FIND_TS_NODE

```

変数名	サイズ	内 容
INLU	(N)	各節点の非零非対角成分数
IALU	(N, NLU)	各節点の非零非対角成分 (列番号)

節点探索 : FIND_TS_NODE

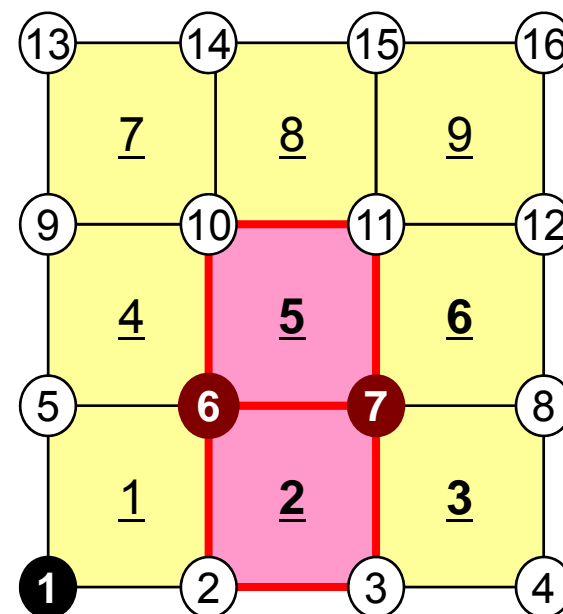
一次元ではこの部分は手動

```

!C
!C***
!C*** FIND_TS_NODE
!C***
!C
      subroutine FIND_TS_NODE (ip1, ip2)
         do kk= 1, INLU(ip1)
            if (ip2. eq. IALU(ip1, kk)) return
         enddo
         icou= INLU(ip1) + 1
         IALU(ip1, icou)= ip2
         INLU(ip1      )= icou
         return
      end subroutine FIND_TS_NODE

```

既にIALUに含まれている
場合は、次のペアへ

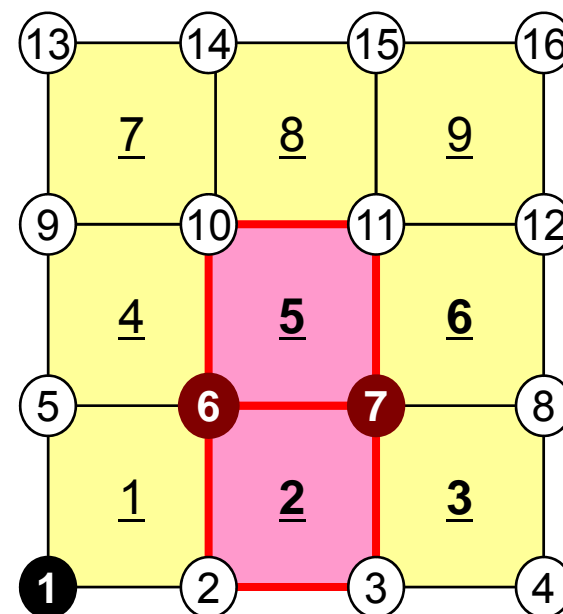


節点探索 : FIND_TS_NODE

一次元ではこの部分は手動

```
!C
!C***
!C*** FIND_TS_NODE
!C***
!C
      subroutine FIND_TS_NODE (ip1, ip2)
        do kk= 1, INLU(ip1)
          if (ip2. eq. IALU(ip1, kk)) return
        enddo
        icou= INLU(ip1) + 1
        IALU(ip1, icou)= ip2
        INLU(ip1      )= icou
        return
      end subroutine FIND_TS_NODE
```

IALUに含まれていない
場合は, INLUに1を加えて
IALUに格納



行列コネクティビティ生成： MAT_CON0 (3/4)

```

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

```

```

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

```

```

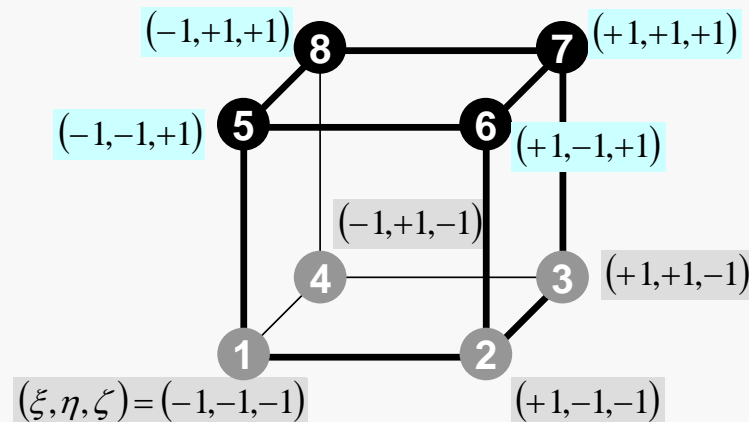
call FIND_TS_NODE (in6, in1)
call FIND_TS_NODE (in6, in2)
call FIND_TS_NODE (in6, in3)
call FIND_TS_NODE (in6, in4)
call FIND_TS_NODE (in6, in5)
call FIND_TS_NODE (in6, in7)
call FIND_TS_NODE (in6, in8)

```

```

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

```



行列コネクティビティ生成： MAT_CON0 (4/4)

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

do in= 1, N
  NN= INLU(in)
  do k= 1, NN
    NCOL1(k)= IALU(in, k)
  enddo
  call mSORT (NCOL1, NCOL2, NN)
  do k= NN, 1, -1
    IALU(in, NN-k+1)= NCOL1(NCOL2(k))
  enddo
enddo
```

各節点において、
IALU(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

```

!C
!C***
!C*** MAT_CON1
!C***
!C
  subroutine MAT_CON1
  use pfem_util
  implicit REAL*8 (A-H, O-Z)

  allocate (index(0:N))
  index= 0

  do i= 1, N
    index(i)= index(i-1) + INLU(i)
  enddo

  NPLU= index(N)

  allocate (item(NPLU))

  do i= 1, N
    do k= 1, INLU(i)
      kk = k + index(i-1)
      item(kk)= IALU(i, k)
    enddo
  enddo

  deallocate (INLU, IALU)

  end subroutine MAT_CON1

```

C

$$\text{index}[i + 1] = \sum_{k=0}^i \text{INLU}[k]$$

$$\text{index}[0] = 0$$

FORTRAN

$$\text{index}(i) = \sum_{k=1}^i \text{INLU}(k)$$

$$\text{index}(0) = 0$$

CRS形式への変換 : MAT_CON1

```
!C
!C***
!C*** MAT_CON1
!C***
!C
  subroutine MAT_CON1
  use pfem_util
  implicit REAL*8 (A-H, O-Z)

  allocate (index(0:N))
  index= 0

  do i= 1, N
    index(i)= index(i-1) + INLU(i)
  enddo

  NPLU= index(N)
  allocate (item(NPLU))

  do i= 1, N
    do k= 1, INLU(i)
      kk = k + index(i-1)
      item(kk)= IALU(i, k)
    enddo
  enddo

  deallocate (INLU, IALU)

  end subroutine MAT_CON1
```

NPLU=index(N)
itemのサイズ
非ゼロ非対角成分総数

CRS形式への変換 : MAT_CON1

```
!C
!C***
!C*** MAT_CON1
!C***
!C
  subroutine MAT_CON1
  use pfem_util
  implicit REAL*8 (A-H, O-Z)

  allocate (index(0:N))
  index= 0

  do i= 1, N
    index(i)= index(i-1) + INLU(i)
  enddo

  NPLU= index(N)

  allocate (item(NPLU))

  do i= 1, N
    do k= 1, INLU(i)
      kk = k + index(i-1)
      item(kk)= IALU(i, k)
    enddo
  enddo

  deallocate (INLU, IALU)

  end subroutine MAT_CON1
```

itemに1から始まる
節点番号を記憶

CRS形式への変換 : MAT_CON1

```
!C
!C***
!C*** MAT_CON1
!C***
!C
  subroutine MAT_CON1
  use pfem_util
  implicit REAL*8 (A-H, O-Z)

  allocate (index(0:N))
  index= 0

  do i= 1, N
    index(i)= index(i-1) + INLU(i)
  enddo

  NPLU= index(N)

  allocate (item(NPLU))

  do i= 1, N
    do k= 1, INLU(i)
      kk = k + index(i-1)
      item(kk)= IALU(i, k)
    enddo
  enddo

  deallocate (INLU, IALU)

end subroutine MAT_CON1
```

これらはもはや不要

全体処理

```
program heat3D

use solver11
use pfem_util

implicit REAL*8(A-H, O-Z)

call INPUT_CNTL
call INPUT_GRID

call MAT_CON0
call MAT_CON1

call MAT_ASS_MAIN
call MAT_ASS_BC

call SOLVE11

call OUTPUT_UCD

end program heat3D
```

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)

```

```

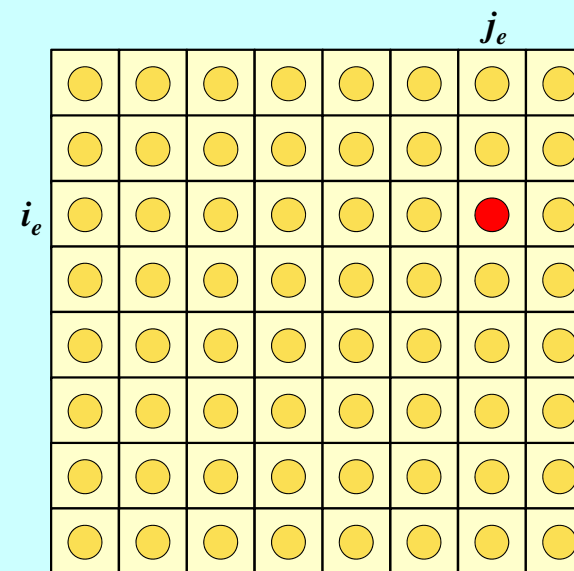
do ie= 1, 8          局所節点番号
  do je= 1, 8        局所節点番号
    全体節点番号 : ip, jp
    Aip, jp の item におけるアドレス : kk

```

```

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

```



係数行列 : MAT_ASS_MAIN (1/6)

```
!C
!C***
!C*** MAT_ASS_MAIN
!C***
!C
subroutine MAT_ASS_MAIN
use pfem_util
implicit REAL*8 (A-H,O-Z)
integer(kind=kint), dimension( 8) :: nodLOCAL

allocate (AMAT(NPLU))
allocate (B(N), D(N), X(N))

AMAT= 0. d0
B= 0. d0
X= 0. d0
D= 0. d0

WEI (1)= +1. 0000000000D+00
WEI (2)= +1. 0000000000D+00

POS (1)= -0. 5773502692D+00
POS (2)= +0. 5773502692D+00
```

係数行列 (非零非対角成分)
右辺ベクトル
未知数ベクトル
係数行列 (対角成分)

係数行列 : MAT_ASS_MAIN (1/6)

```
!C
!C***
!C*** MAT_ASS_MAIN
!C***
!C
subroutine MAT_ASS_MAIN
use pfem_util
implicit REAL*8 (A-H,O-Z)
integer(kind=kint), dimension( 8) :: nodLOCAL

allocate (AMAT(NPLU))
allocate (B(N), D(N), X(N))

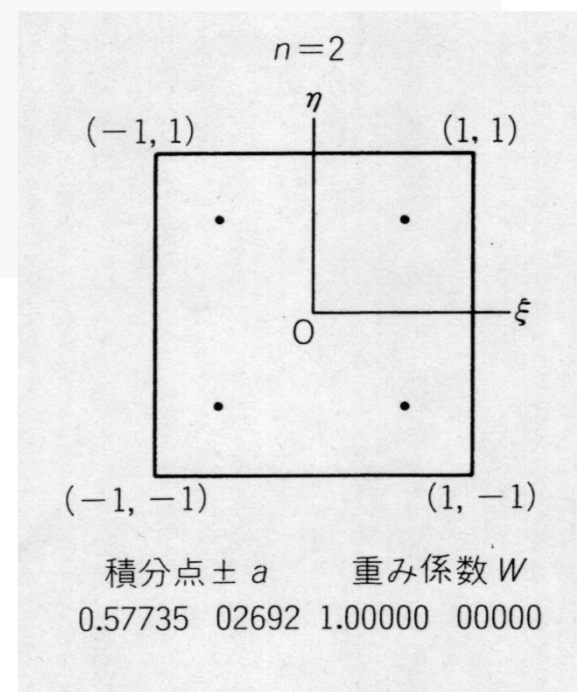
AMAT= 0. d0
B= 0. d0
X= 0. d0
D= 0. d0
```

```
WEI (1)= +1. 0000000000D+00
WEI (2)= +1. 0000000000D+00
```

```
POS (1)= -0. 5773502692D+00
POS (2)= +0. 5773502692D+00
```

POS: 積分点座標

WEI: 重み係数



系数行列：MAT_ASS_MAIN (2/6)

```
!C
!C-- INIT.
!C   PNQ - 1st-order derivative of shape function by QSI
!C   PNE - 1st-order derivative of shape function by ETA
!C   PNT - 1st-order derivative of shape function by ZET
!C
```

```
do kp= 1, 2
do jp= 1, 2
do ip= 1, 2
```

```
QP1= 1. d0 + POS(ip)
QM1= 1. d0 - POS(ip)
EP1= 1. d0 + POS(jp)
EM1= 1. d0 - POS(jp)
TP1= 1. d0 + POS(kp)
TM1= 1. d0 - POS(kp)
```

```
SHAPE(ip, jp, kp, 1) = 08th * QM1 * EM1 * TM1
SHAPE(ip, jp, kp, 2) = 08th * QP1 * EM1 * TM1
SHAPE(ip, jp, kp, 3) = 08th * QP1 * EP1 * TM1
SHAPE(ip, jp, kp, 4) = 08th * QM1 * EP1 * TM1
SHAPE(ip, jp, kp, 5) = 08th * QM1 * EM1 * TP1
SHAPE(ip, jp, kp, 6) = 08th * QP1 * EM1 * TP1
SHAPE(ip, jp, kp, 7) = 08th * QP1 * EP1 * TP1
```


系数行列：MAT_ASS_MAIN (2/6)

```
!C
!C-- INIT.
!C   PNI  - 1st-order derivative of shape function by QSI
!C   PNE  - 1st-order derivative of shape function by ETA
!C   PNT  - 1st-order derivative of shape function by ZET
!C
```

```
do kp= 1, 2
do jp= 1, 2
do ip= 1, 2
```

```
QP1= 1. d0 + POS(ip)
QM1= 1. d0 - POS(ip)
EP1= 1. d0 + POS(jp)
EM1= 1. d0 - POS(jp)
TP1= 1. d0 + POS(kp)
TM1= 1. d0 - POS(kp)
```

```
SHAPE(ip, jp, kp, 1) = 08th * QM1 * EM1 * TM1
SHAPE(ip, jp, kp, 2) = 08th * QP1 * EM1 * TM1
SHAPE(ip, jp, kp, 3) = 08th * QP1 * EP1 * TM1
SHAPE(ip, jp, kp, 4) = 08th * QM1 * EP1 * TM1
SHAPE(ip, jp, kp, 5) = 08th * QM1 * EM1 * TP1
SHAPE(ip, jp, kp, 6) = 08th * QP1 * EM1 * TP1
SHAPE(ip, jp, kp, 7) = 08th * QP1 * EP1 * TP1
```

$$QP1(i) = (1 + \xi_i), \quad QM1(i) = (1 - \xi_i)$$

$$EP1(j) = (1 + \eta_j), \quad EM1(j) = (1 - \eta_j)$$

$$TP1(k) = (1 + \zeta_k), \quad TM1(k) = (1 - \zeta_k)$$

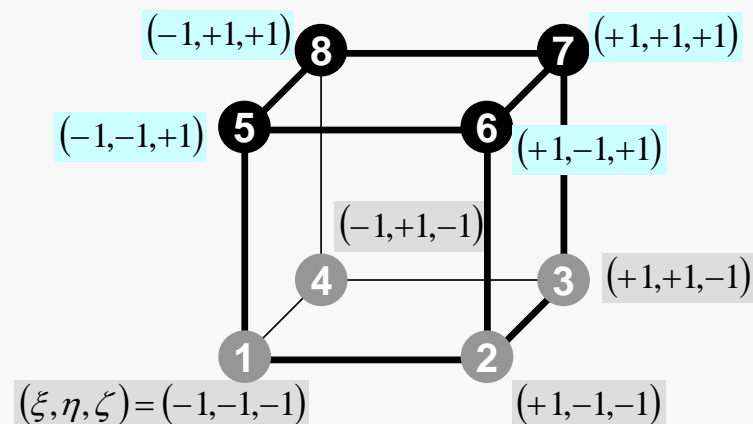
系数行列：MAT_ASS_MAIN (2/6)

```
!C
!C-- INIT.
!C   PNQ  - 1st-order derivative of shape function by QSI
!C   PNE  - 1st-order derivative of shape function by ETA
!C   PNT  - 1st-order derivative of shape function by ZET
!C
```

```
do kp= 1, 2
do jp= 1, 2
do ip= 1, 2
```

```
QP1= 1. d0 + POS(ip)
QM1= 1. d0 - POS(ip)
EP1= 1. d0 + POS(jp)
EM1= 1. d0 - POS(jp)
TP1= 1. d0 + POS(kp)
TM1= 1. d0 - POS(kp)
```

```
SHAPE(ip, jp, kp, 1) = 08th * QM1 * EM1 * TM1
SHAPE(ip, jp, kp, 2) = 08th * QP1 * EM1 * TM1
SHAPE(ip, jp, kp, 3) = 08th * QP1 * EP1 * TM1
SHAPE(ip, jp, kp, 4) = 08th * QM1 * EP1 * TM1
SHAPE(ip, jp, kp, 5) = 08th * QM1 * EM1 * TP1
SHAPE(ip, jp, kp, 6) = 08th * QP1 * EM1 * TP1
SHAPE(ip, jp, kp, 7) = 08th * QP1 * EP1 * TP1
```



系数行列：MAT_ASS_MAIN (2/6)

```
!C
!C-- INIT.
!C   PNQ   - 1st-order derivative of shape function by QSI
!C   PNE   - 1st-order derivative of shape function by ETA
!C   PNT   - 1st-order derivative of shape function by ZET
!C
```

```
do kp= 1, 2
do jp= 1, 2
do ip= 1, 2
```

```
QP1= 1. d0 + POS (ip)
QM1= 1. d0 - POS (ip)
EP1= 1. d0 + POS (jp)
EM1= 1. d0 - POS (jp)
TP1= 1. d0 + POS (kp)
TM1= 1. d0 - POS (kp)
```

```
SHAPE (ip, jp, kp, 1) = 08th * QM1 * EM1 * TM1
SHAPE (ip, jp, kp, 2) = 08th * QP1 * EM1 * TM1
SHAPE (ip, jp, kp, 3) = 08th * QP1 * EP1 * TM1
SHAPE (ip, jp, kp, 4) = 08th * QM1 * EP1 * TM1
SHAPE (ip, jp, kp, 5) = 08th * QM1 * EM1 * TP1
SHAPE (ip, jp, kp, 6) = 08th * QP1 * EM1 * TP1
SHAPE (ip, jp, kp, 7) = 08th * QP1 * 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 (3/6)

```

PNQ (jp, kp, 1) = - 08th * EM1 * TM1
PNQ (jp, kp, 2) = + 08th * EM1 * TM1
PNQ (jp, kp, 3) = + 08th * EP1 * TM1
PNQ (jp, kp, 4) = - 08th * EP1 * TM1
PNQ (jp, kp, 5) = - 08th * EM1 * TP1
PNQ (jp, kp, 6) = + 08th * EM1 * TP1
PNQ (jp, kp, 7) = + 08th * EP1 * TP1
PNQ (jp, kp, 8) = - 08th * EP1 * TP1
PNE (ip, kp, 1) = - 08th * QM1 * TM1
PNE (ip, kp, 2) = - 08th * QP1 * TM1
PNE (ip, kp, 3) = + 08th * QP1 * TM1
PNE (ip, kp, 4) = + 08th * QM1 * TM1
PNE (ip, kp, 5) = - 08th * QM1 * TP1
PNE (ip, kp, 6) = - 08th * QP1 * TP1
PNE (ip, kp, 7) = + 08th * QP1 * TP1
PNE (ip, kp, 8) = + 08th * QM1 * TP1
PNT (ip, jp, 1) = - 08th * QM1 * EM1
PNT (ip, jp, 2) = - 08th * QP1 * EM1
PNT (ip, jp, 3) = - 08th * QP1 * EP1
PNT (ip, jp, 4) = - 08th * QM1 * EP1
PNT (ip, jp, 5) = + 08th * QM1 * EM1
PNT (ip, jp, 6) = + 08th * QP1 * EM1
PNT (ip, jp, 7) = + 08th * QP1 * EP1
PNT (ip, jp, 8) = + 08th * QM1 * EP1

```

```

enddo
enddo
enddo

```

```

do icel= 1, ICELTOT
  CONDO= COND

```

```

in1= ICELNOD (icel, 1)
in2= ICELNOD (icel, 2)
in3= ICELNOD (icel, 3)
in4= ICELNOD (icel, 4)
in5= ICELNOD (icel, 5)
in6= ICELNOD (icel, 6)
in7= ICELNOD (icel, 7)
in8= ICELNOD (icel, 8)

```

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

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

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

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

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

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

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

(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 (3/6)

```

PNQ (jp, kp, 1) = - 08th * EM1 * TM1
PNQ (jp, kp, 2) = + 08th * EM1 * TM1
PNQ (jp, kp, 3) = + 08th * EP1 * TM1
PNQ (jp, kp, 4) = - 08th * EP1 * TM1
PNQ (jp, kp, 5) = - 08th * EM1 * TP1
PNQ (jp, kp, 6) = + 08th * EM1 * TP1
PNQ (jp, kp, 7) = + 08th * EP1 * TP1
PNQ (jp, kp, 8) = - 08th * EP1 * TP1
PNE (ip, kp, 1) = - 08th * QM1 * TM1
PNE (ip, kp, 2) = - 08th * QP1 * TM1
PNE (ip, kp, 3) = + 08th * QP1 * TM1
PNE (ip, kp, 4) = + 08th * QM1 * TM1
PNE (ip, kp, 5) = - 08th * QM1 * TP1
PNE (ip, kp, 6) = - 08th * QP1 * TP1
PNE (ip, kp, 7) = + 08th * QP1 * TP1
PNE (ip, kp, 8) = + 08th * QM1 * TP1
PNT (ip, jp, 1) = - 08th * QM1 * EM1
PNT (ip, jp, 2) = - 08th * QP1 * EM1
PNT (ip, jp, 3) = - 08th * QP1 * EP1
PNT (ip, jp, 4) = - 08th * QM1 * EP1
PNT (ip, jp, 5) = + 08th * QM1 * EM1
PNT (ip, jp, 6) = + 08th * QP1 * EM1
PNT (ip, jp, 7) = + 08th * QP1 * EP1
PNT (ip, jp, 8) = + 08th * QM1 * EP1

```

```

enddo
enddo
enddo

```

```

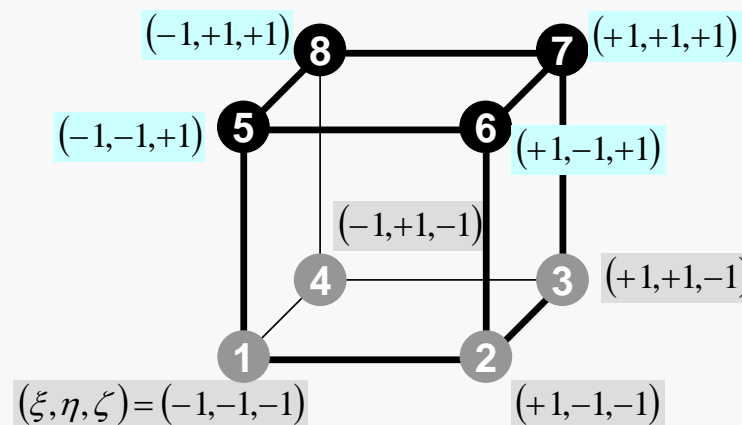
do icel= 1, ICELTOT
  CONDO= COND

```

```

in1= ICELNOD (icel, 1)
in2= ICELNOD (icel, 2)
in3= ICELNOD (icel, 3)
in4= ICELNOD (icel, 4)
in5= ICELNOD (icel, 5)
in6= ICELNOD (icel, 6)
in7= ICELNOD (icel, 7)
in8= ICELNOD (icel, 8)

```



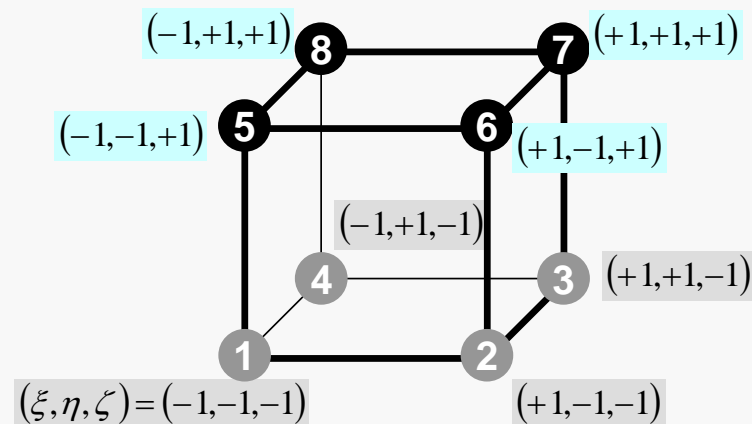
係数行列 : MAT_ASS_MAIN (4/6)

```

nodLOCAL (1) = in1
nodLOCAL (2) = in2
nodLOCAL (3) = in3
nodLOCAL (4) = in4
nodLOCAL (5) = in5
nodLOCAL (6) = in6
nodLOCAL (7) = in7
nodLOCAL (8) = in8

```

8節点の節点番号



```

X1= XYZ (in1, 1)
X2= XYZ (in2, 1)
X3= XYZ (in3, 1)
X4= XYZ (in4, 1)
X5= XYZ (in5, 1)
X6= XYZ (in6, 1)
X7= XYZ (in7, 1)
X8= XYZ (in8, 1)
Y1= XYZ (in1, 2)
Y2= XYZ (in2, 2)
Y3= XYZ (in3, 2)
Y4= XYZ (in4, 2)
Y5= XYZ (in5, 2)
Y6= XYZ (in6, 2)
Y7= XYZ (in7, 2)
Y8= XYZ (in8, 2)
QVC= 08th * (X1+X2+X3+X4+X5+X6+X7+X8+
&           Y1+Y2+Y3+Y4+Y5+Y6+Y7+Y8)

```

```

Z1= XYZ (in1, 3)
Z2= XYZ (in2, 3)
Z3= XYZ (in3, 3)
Z4= XYZ (in4, 3)
Z5= XYZ (in5, 3)
Z6= XYZ (in6, 3)
Z7= XYZ (in7, 3)
Z8= XYZ (in8, 3)

```

```

& call 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 )

```

```

&
&
&

```

係数行列 : MAT_ASS_MAIN (4/6)

```

nodLOCAL (1)= in1
nodLOCAL (2)= in2
nodLOCAL (3)= in3
nodLOCAL (4)= in4
nodLOCAL (5)= in5
nodLOCAL (6)= in6
nodLOCAL (7)= in7
nodLOCAL (8)= in8

```

```

X1= XYZ (in1, 1)
X2= XYZ (in2, 1)
X3= XYZ (in3, 1)
X4= XYZ (in4, 1)
X5= XYZ (in5, 1)
X6= XYZ (in6, 1)
X7= XYZ (in7, 1)
X8= XYZ (in8, 1)

```

8節点のX座標

```

Y1= XYZ (in1, 2)
Y2= XYZ (in2, 2)
Y3= XYZ (in3, 2)
Y4= XYZ (in4, 2)
Y5= XYZ (in5, 2)
Y6= XYZ (in6, 2)
Y7= XYZ (in7, 2)
Y8= XYZ (in8, 2)

```

8節点のY座標

```

& QVC= 08th * (X1+X2+X3+X4+X5+X6+X7+X8+
& Y1+Y2+Y3+Y4+Y5+Y6+Y7+Y8)

```

```

Z1= XYZ (in1, 3)
Z2= XYZ (in2, 3)
Z3= XYZ (in3, 3)
Z4= XYZ (in4, 3)
Z5= XYZ (in5, 3)
Z6= XYZ (in6, 3)
Z7= XYZ (in7, 3)
Z8= XYZ (in8, 3)

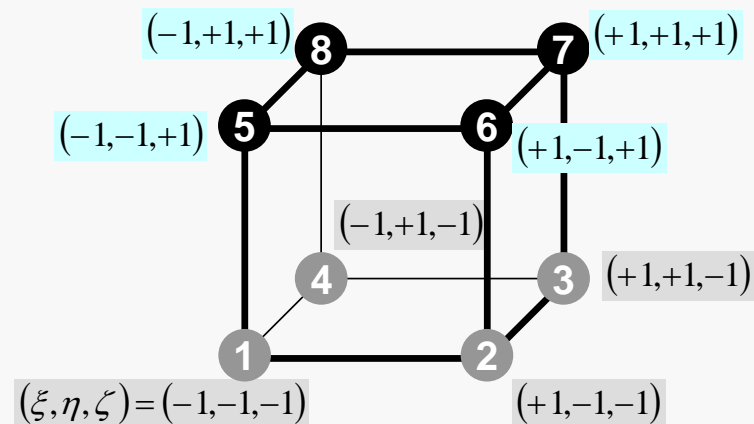
```

8節点のZ座標

```

& call 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 )

```



係数行列 : MAT_ASS_MAIN (4/6)

```

nodLOCAL (1)= in1
nodLOCAL (2)= in2
nodLOCAL (3)= in3
nodLOCAL (4)= in4
nodLOCAL (5)= in5
nodLOCAL (6)= in6
nodLOCAL (7)= in7
nodLOCAL (8)= in8

```

```

X1= XYZ (in1, 1)
X2= XYZ (in2, 1)
X3= XYZ (in3, 1)
X4= XYZ (in4, 1)
X5= XYZ (in5, 1)
X6= XYZ (in6, 1)
X7= XYZ (in7, 1)
X8= XYZ (in8, 1)
Y1= XYZ (in1, 2)
Y2= XYZ (in2, 2)
Y3= XYZ (in3, 2)
Y4= XYZ (in4, 2)
Y5= XYZ (in5, 2)
Y6= XYZ (in6, 2)
Y7= XYZ (in7, 2)
Y8= XYZ (in8, 2)

```

```

& QVC= 08th * (X1+X2+X3+X4+X5+X6+X7+X8+
              Y1+Y2+Y3+Y4+Y5+Y6+Y7+Y8)

```

```

Z1= XYZ (in1, 3)
Z2= XYZ (in2, 3)
Z3= XYZ (in3, 3)
Z4= XYZ (in4, 3)
Z5= XYZ (in5, 3)
Z6= XYZ (in6, 3)
Z7= XYZ (in7, 3)
Z8= XYZ (in8, 3)

```

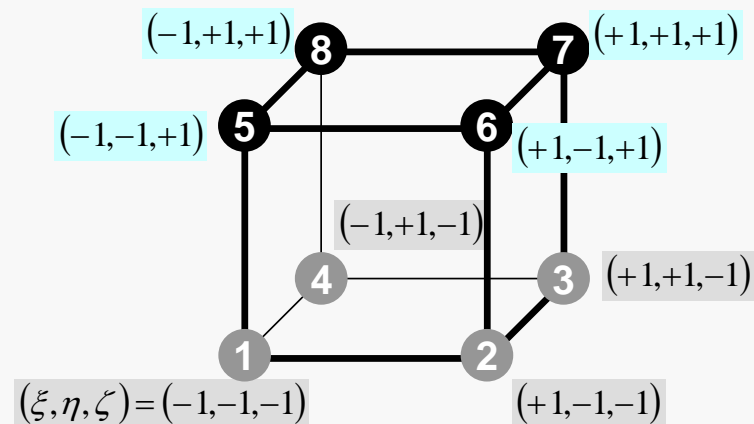
```

& call 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X座標

8節点のY座標



$$\frac{\partial}{\partial x} \left(\lambda \frac{\partial T}{\partial x} \right) + \frac{\partial}{\partial y} \left(\lambda \frac{\partial T}{\partial y} \right) + \frac{\partial}{\partial z} \left(\lambda \frac{\partial T}{\partial z} \right) + \dot{Q}(x, y, z) = 0$$

$$\dot{Q}(x, y, z) = QVOL |x_c + y_c|$$

体積当たり発熱量は位置（メッシュの中心の座標 x_c, y_c ）に依存

系数行列：MAT_ASS_MAIN (4/6)

```

nodLOCAL (1)= in1
nodLOCAL (2)= in2
nodLOCAL (3)= in3
nodLOCAL (4)= in4
nodLOCAL (5)= in5
nodLOCAL (6)= in6
nodLOCAL (7)= in7
nodLOCAL (8)= in8

```

```

X1= XYZ (in1, 1)
X2= XYZ (in2, 1)
X3= XYZ (in3, 1)
X4= XYZ (in4, 1)
X5= XYZ (in5, 1)
X6= XYZ (in6, 1)
X7= XYZ (in7, 1)
X8= XYZ (in8, 1)
Y1= XYZ (in1, 2)
Y2= XYZ (in2, 2)
Y3= XYZ (in3, 2)
Y4= XYZ (in4, 2)
Y5= XYZ (in5, 2)
Y6= XYZ (in6, 2)
Y7= XYZ (in7, 2)
Y8= XYZ (in8, 2)

```

& **QVC= 08th * (X1+X2+X3+X4+X5+X6+X7+X8+
Y1+Y2+Y3+Y4+Y5+Y6+Y7+Y8)**

```

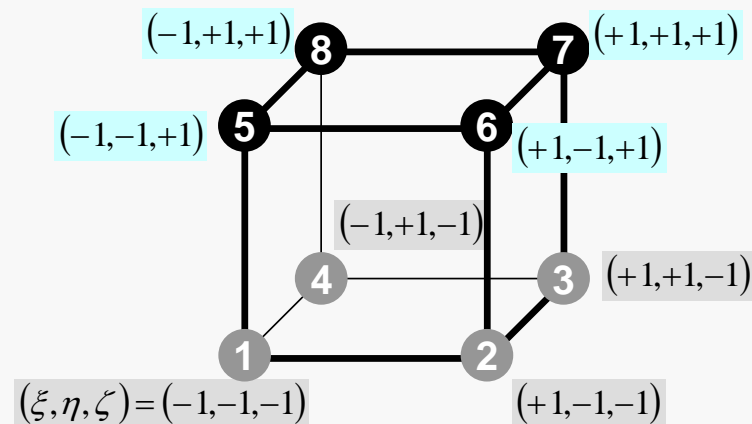
Z1= XYZ (in1, 3)
Z2= XYZ (in2, 3)
Z3= XYZ (in3, 3)
Z4= XYZ (in4, 3)
Z5= XYZ (in5, 3)
Z6= XYZ (in6, 3)
Z7= XYZ (in7, 3)
Z8= XYZ (in8, 3)

```

```

& call 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 )

```



$$\frac{\partial}{\partial x} \left(\lambda \frac{\partial T}{\partial x} \right) + \frac{\partial}{\partial y} \left(\lambda \frac{\partial T}{\partial y} \right) + \frac{\partial}{\partial z} \left(\lambda \frac{\partial T}{\partial z} \right) + \dot{Q}(x, y, z) = 0$$

$$\dot{Q}(x, y, z) = QVOL |x_c + y_c|$$

$$QVC = |x_c + y_c|$$

&
&
&

系数行列 : MAT_ASS_MAIN (4/6)

```

nodLOCAL (1)= in1
nodLOCAL (2)= in2
nodLOCAL (3)= in3
nodLOCAL (4)= in4
nodLOCAL (5)= in5
nodLOCAL (6)= in6
nodLOCAL (7)= in7
nodLOCAL (8)= in8

X1= XYZ (in1, 1)
X2= XYZ (in2, 1)
X3= XYZ (in3, 1)
X4= XYZ (in4, 1)
X5= XYZ (in5, 1)
X6= XYZ (in6, 1)
X7= XYZ (in7, 1)
X8= XYZ (in8, 1)
Y1= XYZ (in1, 2)
Y2= XYZ (in2, 2)
Y3= XYZ (in3, 2)
Y4= XYZ (in4, 2)
Y5= XYZ (in5, 2)
Y6= XYZ (in6, 2)
Y7= XYZ (in7, 2)
Y8= XYZ (in8, 2)
QVC= 08th * (X1+X2+X3+X4+X5+X6+X7+X8+
&          Y1+Y2+Y3+Y4+Y5+Y6+Y7+Y8)
Z1= XYZ (in1, 3)
Z2= XYZ (in2, 3)
Z3= XYZ (in3, 3)
Z4= XYZ (in4, 3)
Z5= XYZ (in5, 3)
Z6= XYZ (in6, 3)
Z7= XYZ (in7, 3)
Z8= XYZ (in8, 3)

& call JACOBI (DETJ, PNO, PNE, PNT, PNQ, 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 )
&&&

```

JACOBI (1/4)

```
subroutine JACOBI (DETJ, PNQ, PNE, PNT, PNQ, 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 )
```

```
!C
!C
!C
!C
```

```
calculates JACOBIAN & INVERSE JACOBIAN
dNi/dx, dNi/dy & dNi/dz
```

```
implicit REAL*8 (A-H, O-Z)
dimension DETJ(2, 2, 2)
dimension PNQ(2, 2, 8), PNE(2, 2, 8), PNT(2, 2, 8)
dimension PNQ(2, 2, 2, 8), PNY(2, 2, 2, 8), PNZ(2, 2, 2, 8)
```

```
do kp= 1, 2
do jp= 1, 2
do ip= 1, 2
  PNQ(ip, jp, kp, 1)=0. d0
  PNQ(ip, jp, kp, 2)=0. d0
  PNQ(ip, jp, kp, 3)=0. d0
  PNQ(ip, jp, kp, 4)=0. d0
  PNQ(ip, jp, kp, 5)=0. d0
  PNQ(ip, jp, kp, 6)=0. d0
  PNQ(ip, jp, kp, 7)=0. d0
  PNQ(ip, jp, kp, 8)=0. d0
  PNY(ip, jp, kp, 1)=0. d0
  PNY(ip, jp, kp, 2)=0. d0
  PNY(ip, jp, kp, 3)=0. d0
  PNY(ip, jp, kp, 4)=0. d0
  PNY(ip, jp, kp, 5)=0. d0
  PNY(ip, jp, kp, 6)=0. d0
  PNY(ip, jp, kp, 7)=0. d0
  PNY(ip, jp, kp, 8)=0. d0
  PNZ(ip, jp, kp, 1)=0. d0
  PNZ(ip, jp, kp, 2)=0. d0
  PNZ(ip, jp, kp, 3)=0. d0
  PNZ(ip, jp, kp, 4)=0. d0
  PNZ(ip, jp, kp, 5)=0. d0
  PNZ(ip, jp, kp, 6)=0. d0
  PNZ(ip, jp, kp, 7)=0. d0
  PNZ(ip, jp, kp, 8)=0. d0
```

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)

!C
!C== DETERMINANT of the JACOBIAN

&	dXdQ =	+ PNQ (jp, kp, 1) * X1 + PNQ (jp, kp, 2) * X2
&		+ PNQ (jp, kp, 3) * X3 + PNQ (jp, kp, 4) * X4
&		+ PNQ (jp, kp, 5) * X5 + PNQ (jp, kp, 6) * X6
&		+ PNQ (jp, kp, 7) * X7 + PNQ (jp, kp, 8) * X8
&	dYdQ =	+ PNQ (jp, kp, 1) * Y1 + PNQ (jp, kp, 2) * Y2
&		+ PNQ (jp, kp, 3) * Y3 + PNQ (jp, kp, 4) * Y4
&		+ PNQ (jp, kp, 5) * Y5 + PNQ (jp, kp, 6) * Y6
&		+ PNQ (jp, kp, 7) * Y7 + PNQ (jp, kp, 8) * Y8
&	dZdQ =	+ PNQ (jp, kp, 1) * Z1 + PNQ (jp, kp, 2) * Z2
&		+ PNQ (jp, kp, 3) * Z3 + PNQ (jp, kp, 4) * Z4
&		+ PNQ (jp, kp, 5) * Z5 + PNQ (jp, kp, 6) * Z6
&		+ PNQ (jp, kp, 7) * Z7 + PNQ (jp, kp, 8) * Z8
&	dXdE =	+ PNE (ip, kp, 1) * X1 + PNE (ip, kp, 2) * X2
&		+ PNE (ip, kp, 3) * X3 + PNE (ip, kp, 4) * X4
&		+ PNE (ip, kp, 5) * X5 + PNE (ip, kp, 6) * X6
&		+ PNE (ip, kp, 7) * X7 + PNE (ip, kp, 8) * X8
&	dYdE =	+ PNE (ip, kp, 1) * Y1 + PNE (ip, kp, 2) * Y2
&		+ PNE (ip, kp, 3) * Y3 + PNE (ip, kp, 4) * Y4
&		+ PNE (ip, kp, 5) * Y5 + PNE (ip, kp, 6) * Y6
&		+ PNE (ip, kp, 7) * Y7 + PNE (ip, kp, 8) * Y8
&	dZdE =	+ PNE (ip, kp, 1) * Z1 + PNE (ip, kp, 2) * Z2
&		+ PNE (ip, kp, 3) * Z3 + PNE (ip, kp, 4) * Z4
&		+ PNE (ip, kp, 5) * Z5 + PNE (ip, kp, 6) * Z6
&		+ PNE (ip, kp, 7) * Z7 + PNE (ip, kp, 8) * Z8

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

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

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

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

JACOBI (3/4)

```

dXdT =
&      + PNT (ip, jp, 1) * X1 + PNT (ip, jp, 2) * X2
&      + PNT (ip, jp, 3) * X3 + PNT (ip, jp, 4) * X4
&      + PNT (ip, jp, 5) * X5 + PNT (ip, jp, 6) * X6
&      + PNT (ip, jp, 7) * X7 + PNT (ip, jp, 8) * X8

dYdT =
&      + PNT (ip, jp, 1) * Y1 + PNT (ip, jp, 2) * Y2
&      + PNT (ip, jp, 3) * Y3 + PNT (ip, jp, 4) * Y4
&      + PNT (ip, jp, 5) * Y5 + PNT (ip, jp, 6) * Y6
&      + PNT (ip, jp, 7) * Y7 + PNT (ip, jp, 8) * Y8

dZdT =
&      + PNT (ip, jp, 1) * Z1 + PNT (ip, jp, 2) * Z2
&      + PNT (ip, jp, 3) * Z3 + PNT (ip, jp, 4) * Z4
&      + PNT (ip, jp, 5) * Z5 + PNT (ip, jp, 6) * Z6
&      + PNT (ip, jp, 7) * Z7 + PNT (ip, jp, 8) * Z8

&
&
&
&
DETJ (ip, jp, kp) = dXdQ*(dYdE*dZdT-dZdE*dYdT) +
&                  dYdQ*(dZdE*dXdT-dXdE*dZdT) +
&                  dZdQ*(dXdE*dYdT-dYdE*dXdT)

```

```

!C
!C==

```

INVERSE JACOBIAN

coef= 1. d0 / 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) = dabs (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)

```

dXdT =
&      + PNT (ip, jp, 1) * X1 + PNT (ip, jp, 2) * X2
&      + PNT (ip, jp, 3) * X3 + PNT (ip, jp, 4) * X4
&      + PNT (ip, jp, 5) * X5 + PNT (ip, jp, 6) * X6
&      + PNT (ip, jp, 7) * X7 + PNT (ip, jp, 8) * X8

dYdT =
&      + PNT (ip, jp, 1) * Y1 + PNT (ip, jp, 2) * Y2
&      + PNT (ip, jp, 3) * Y3 + PNT (ip, jp, 4) * Y4
&      + PNT (ip, jp, 5) * Y5 + PNT (ip, jp, 6) * Y6
&      + PNT (ip, jp, 7) * Y7 + PNT (ip, jp, 8) * Y8

dZdT =
&      + PNT (ip, jp, 1) * Z1 + PNT (ip, jp, 2) * Z2
&      + PNT (ip, jp, 3) * Z3 + PNT (ip, jp, 4) * Z4
&      + PNT (ip, jp, 5) * Z5 + PNT (ip, jp, 6) * Z6
&      + PNT (ip, jp, 7) * Z7 + PNT (ip, jp, 8) * Z8

DETJ (ip, jp, kp) = dXdQ*(dYdE*dZdT-dZdE*dYdT) +
&                  dYdQ*(dZdE*dXdT-dXdE*dZdT) +
&                  dZdQ*(dXdE*dYdT-dYdE*dXdT)

```

```
!C
!C==
```

INVERSE JACOBIAN

coef= 1. d0 / 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)= dabs(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)

!C

!C== set the dNi/dX, dNi/dY & dNi/dZ components

```

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)
PNX(ip, jp, kp, 8) = a11*PNQ(jp, kp, 8) + a12*PNE(ip, kp, 8) + a13*PNT(ip, jp, 8)

```

```

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)
PNY(ip, jp, kp, 8) = a21*PNQ(jp, kp, 8) + a22*PNE(ip, kp, 8) + a23*PNT(ip, jp, 8)

```

```

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)
PNZ(ip, jp, kp, 8) = a31*PNQ(jp, kp, 8) + a32*PNE(ip, kp, 8) + a33*PNT(ip, jp, 8)

```

```

enddo
enddo
enddo

```

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

係数行列 : MAT_ASS_MAIN (5/6)

```

!C
!C== CONSTRUCT the GLOBAL MATRIX
do ie= 1, 8
  ip = nodLOCAL (ie)
do je= 1, 8
  jp = nodLOCAL (je)

  kk= 0
  if (jp.ne.ip) then
    iiS= index(ip)-1 + 1
    iiE= index(ip )
    do k= iiS, iiE
      if ( item(k).eq.jp ) then
        kk= k
        exit
      endif
    enddo
  endif
endif

```

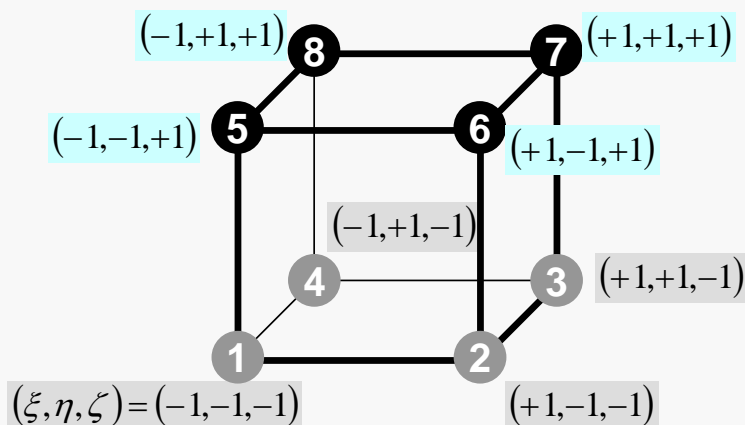
全体行列の非対角成分

$$A_{ip, jp}$$

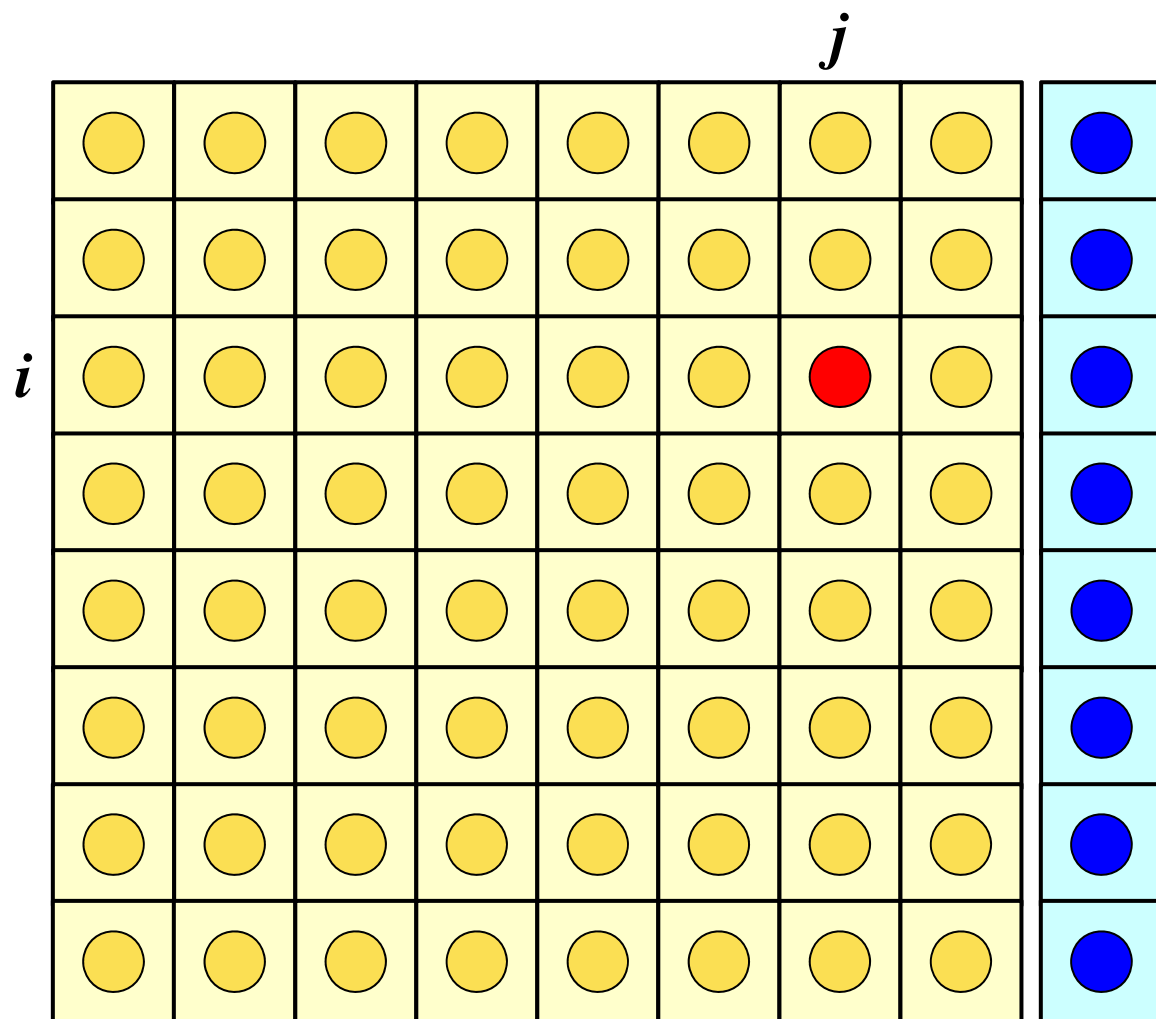
kk: itemにおけるアドレス

$$ip = \text{nodLOCAL}(ie)$$

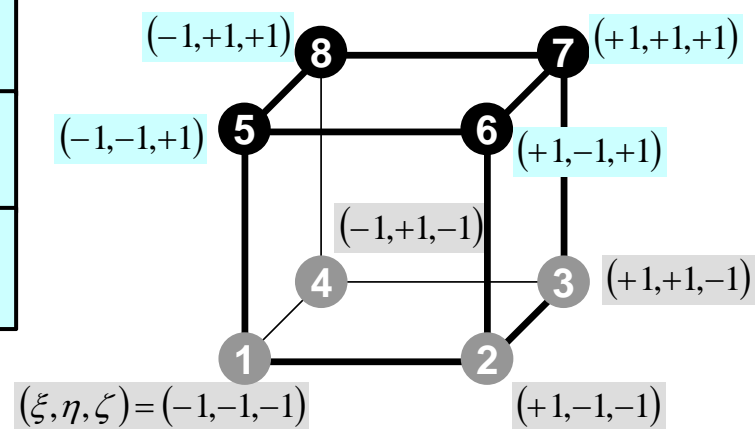
$$jp = \text{nodLOCAL}(je)$$



要素マトリクス : 8×8 行列



$$[k_{ij}] \quad (i, j = 1 \dots 8)$$



係数行列 : MAT_ASS_MAIN (5/6)

```

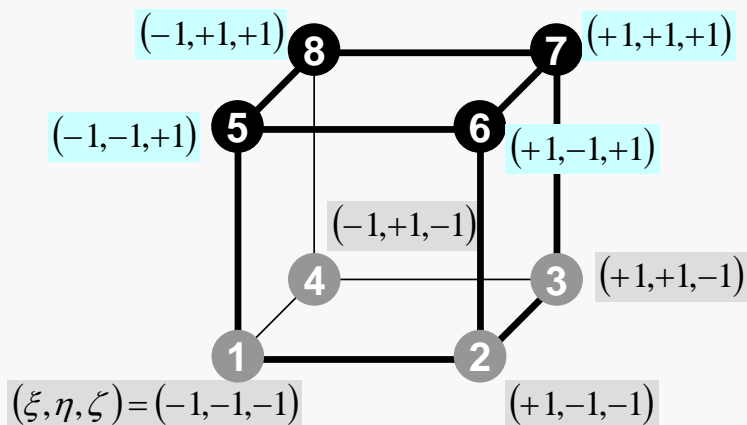
!C
!C== CONSTRUCT the GLOBAL MATRIX
do ie= 1, 8
  ip = nodLOCAL (ie)
do je= 1, 8
  jp = nodLOCAL (je)

  kk= 0
  if (jp.ne.ip) then
    iiS= index(ip-1) + 1
    iiE= index(ip )
    do k= iiS, iiE
      if ( item(k).eq.jp ) then
        kk= k
        exit
      endif
    enddo
  endif
endif

```

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: itemにおけるアドレス



系数行列：MAT_ASS_MAIN (6/6)

```

QVO = 0. d0
COEFij= 0. d0
do kpn= 1, 2
do jpn= 1, 2
do ipn= 1, 2
  coef= dabs (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)

  & COEFij= COEFij + coef * CONDO *
      (PNXi*PNXj+PNYi*PNYj+PNZi*PNZj)

  SHi= SHAPE (ipn, jpn, kpn, ie)
  QVO= QVO + SHi * QVOL * coef
enddo
enddo
enddo

if (jp. eq. ip) then
  D(ip)= D(ip) + COEFij
  B(ip)= B(ip) + QVO*QVC
else
  AMAT(kk)= AMAT(kk) + COEFij
endif
enddo
enddo
enddo

return
end

```

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

系数行列：MAT_ASS_MAIN (6/6)

```

QVO = 0. d0
COEFij= 0. d0
do kpn= 1, 2
do jpn= 1, 2
do ipn= 1, 2
coef= dabs (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)

& COEFij= COEFij + coef * CONDO *
(PNXi*PNXj+PNYi*PNYj+PNZi*PNZj)

SHi= SHAPE (ipn, jpn, kpn, ie)
QVO= QVO + SHi * QVOL * coef
enddo
enddo
enddo

if (jp. eq. ip) then
D(ip)= D(ip) + COEFij
B(ip)= B(ip) + QVO*QVC
else
AMAT(kk)= AMAT(kk) + COEFij
endif
enddo
enddo
enddo

return
end

```

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

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

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

系数行列：MAT_ASS_MAIN (6/6)

```

QVO = 0. d0
COEFij= 0. d0
do kpn= 1, 2
do jpn= 1, 2
do ipn= 1, 2
coef= dabs (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)

& COEFij= COEFij + coef * CONDO *
(PNXi*PNXj+PNYi*PNYj+PNZi*PNZj)

SHi= SHAPE (ipn, jpn, kpn, ie)
QVO= QVO + SHi * QVOL * coef
enddo
enddo
enddo

if (jp. eq. ip) then
D(ip)= D(ip) + COEFij
B(ip)= B(ip) + QVO*QVC
else
AMAT(kk)= AMAT(kk) + COEFij
endif
enddo
enddo
enddo

return
end

```

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

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

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

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

系数行列：MAT_ASS_MAIN (6/6)

```

QV0 = 0. d0
COEFij= 0. d0
do kpn= 1, 2
do jpn= 1, 2
do ipn= 1, 2
  coef= dabs (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)

  & COEFij= COEFij + coef * CONDO *
      (PNXi*PNXj+PNYi*PNYj+PNZi*PNZj)

  SHi= SHAPE (ipn, jpn, kpn, ie)
  QV0= QV0 + SHi * QVOL * coef
enddo
enddo
enddo

if (jp. eq. ip) then
  D(ip)= D(ip) + COEFij
  B(ip)= B(ip) + QV0*QVC
else
  AMAT(kk)= AMAT(kk) + COEFij
endif
enddo
enddo
enddo

return
end

```

$$[k]^{(e)} \{\phi\}^{(e)} = \{f\}^{(e)}$$

$$\{f\}^{(e)} = \int_V \dot{Q} [N]^T dV$$

$$\dot{Q}(x, y, z) = QVOL |x_C + y_C|$$

$$QVC = |x_C + y_C|$$

$$QV0 = \int_V QVOL [N]^T dV$$

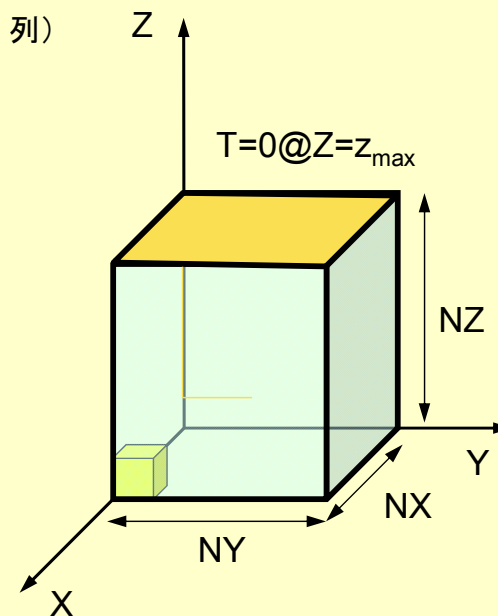
$$\{f\}^{(e)} = QV0 \cdot QVC$$

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, 対角成分 (D) の成分の修正 (行・列)
    do k= index(i-1)+1, index(i)
      対応する非零非対角成分 (AMAT) の成分の修正 (行)
    enddo
  endif
enddo
```

```
do i= 1, N  節点ループ
  do k= index(i-1)+1, index(i)
    if (IWKX(item(k),1).eq.1) then  対応する非零非対角成分の
      節点がマークされていたら
        対応する右辺ベクトル, 非零非対角成分 (AMAT) の成分の修正 (列)
    endif
  enddo
enddo
```



境界条件 : MAT_ASS_BC (1/2)

```
subroutine MAT_ASS_BC
use pfem_util
implicit REAL*8 (A-H, O-Z)

allocate (IWKX(N, 2))
IWKX= 0

!C
!C== Z=Zmax

do in= 1, N
  IWKX(in, 1)= 0
enddo

ib0= -1
do ib0= 1, NODGRPtot
  if (NODGRP_NAME(ib0).eq.'Zmax') exit
enddo

do ib= NODGRP_INDEX(ib0-1)+1, NODGRP_INDEX(ib0)
  in= NODGRP_ITEM(ib)
  IWKX(in, 1)= 1
enddo
```

節点グループ名が「Zmax」である
節点inにおいて:

$$IWKX(in,1)= 1$$

とする

境界条件 : MAT_ASS_BC (2/2)

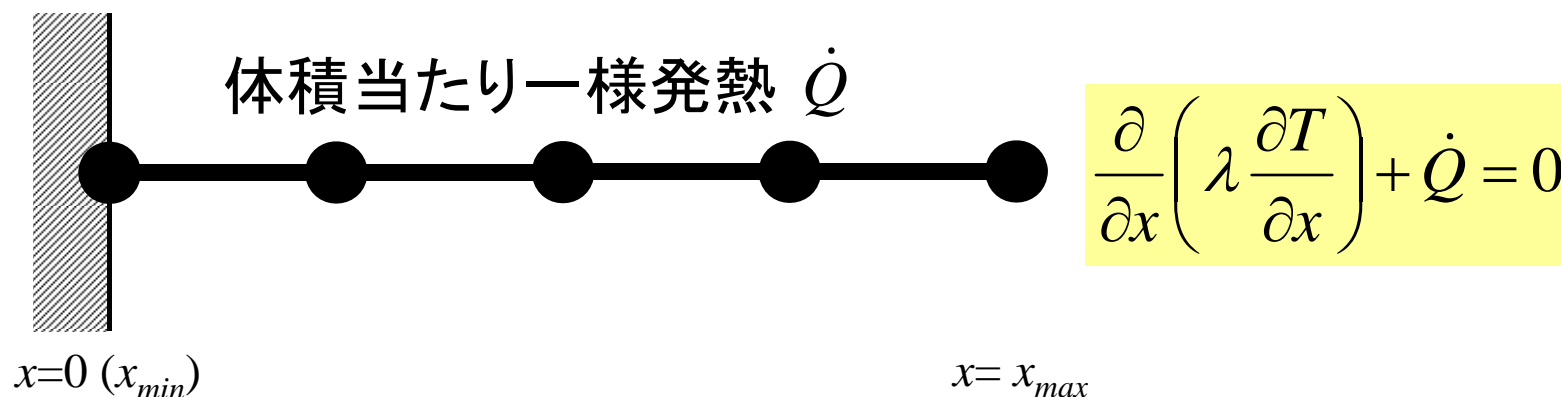
```
do in= 1, N
  if (IWXX(in,1).eq.1) then
    B(in)= 0.d0
    D(in)= 1.d0

    iS= index(in-1) + 1
    iE= index(in )
    do k= iS, iE
      AMAT(k)= 0.d0
    enddo
  endif
enddo

do in= 1, N
  iS= index(in-1) + 1
  iE= index(in )
  do k= iS, iE
    if (IWXX(item(k),1).eq.1) then
      AMAT(k)= 0.d0
    endif
  enddo
enddo

!C==
return
end
```

対象とする問題：一次元熱伝導問題

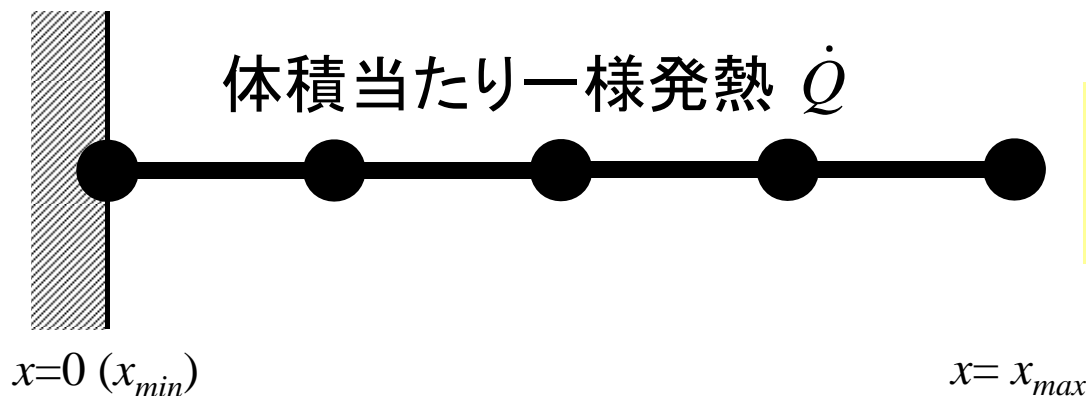


- 一様な：断面積 A ，熱伝導率 λ
- 体積当たり一様発熱（時間当たり） $[QL^{-3}T^{-1}]$ \dot{Q}
- 境界条件
 - $x=0$: $T=0$ （固定）
 - $x=x_{max}$: $\frac{\partial T}{\partial x} = 0$ （断熱）

復習：一次元問題

$x=0$ で成立する方程式

$$T_1=0$$



$$\frac{\partial}{\partial x} \left(\lambda \frac{\partial T}{\partial x} \right) + \dot{Q} = 0$$

- 一様な：断面積 A ，熱伝導率 λ
- 体積当たり一様発熱（時間当たり） $[QL^{-3}T^{-1}]$ \dot{Q}
- 境界条件
 - $x=0$: $T=0$ (固定)
 -
 - $x=x_{max}$: $\frac{\partial T}{\partial x} = 0$ (断熱)

復習：一次元問題

プログラム: 1d.f(6/6)

第一種境界条件 @x=0

```

!C
!C +-----+
!C | BOUNDARY CONDITIONS |
!C +-----+
!C===

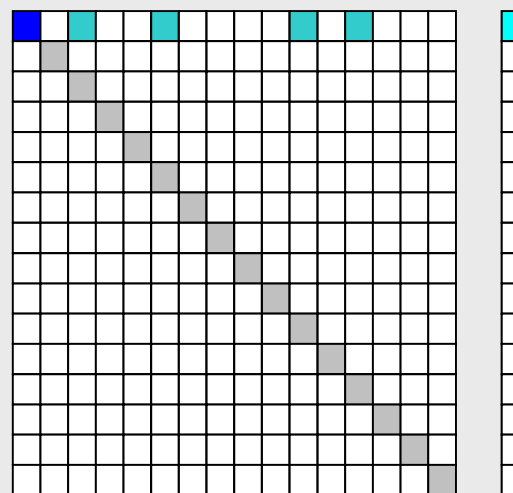
!C
!C-- X=Xmin
      i= 1
      js= INDEX(i-1)

      AMAT (js+1)= 0. d0
      DIAG (i)= 1. d0
      RHS (i)= 0. d0

      do k= 1, NPLU
        if (ITEM(k).eq. 1) AMAT (k)= 0. d0
      enddo
!C===

```

$T_1=0$
 対角成分=1, 右辺=0, 非対角成分=0



復習: 一次元問題

プログラム: 1d.f(6/6)

第一種境界条件 @x=0

```

!C
!C +-----+
!C | BOUNDARY CONDITIONS |
!C +-----+
!C===

!C
!C-- X=Xmin
      i= 1
      js= INDEX(i-1)

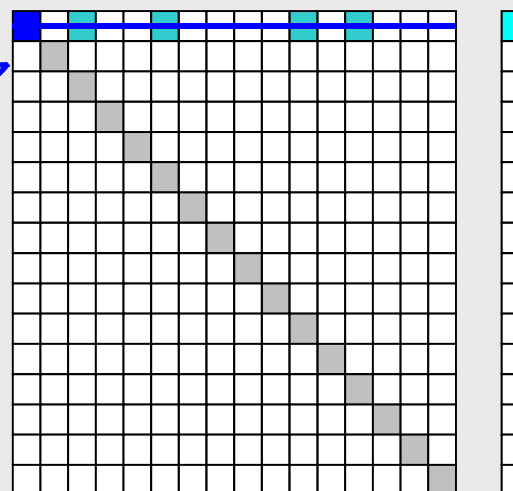
      AMAT (js+1)= 0. d0
      DIAG (i)= 1. d0
      RHS (i)= 0. d0

      do k= 1, NPLU
        if (ITEM(k).eq. 1) AMAT (k)= 0. d0
      enddo
!C===

```

$T_1=0$
 対角成分=1, 右辺=0, 非対角成分=0

ゼロクリア



復習: 一次元問題

プログラム: 1d.f(6/6)

第一種境界条件 @x=0

```

!C
!C +-----+
!C | BOUNDARY CONDITIONS |
!C +-----+
!C===

!C
!C-- X=Xmin
      i= 1
      jS= INDEX(i-1)

      AMAT(jS+1)= 0. d0
      DIAG(i)= 1. d0
      RHS (i)= 0. d0

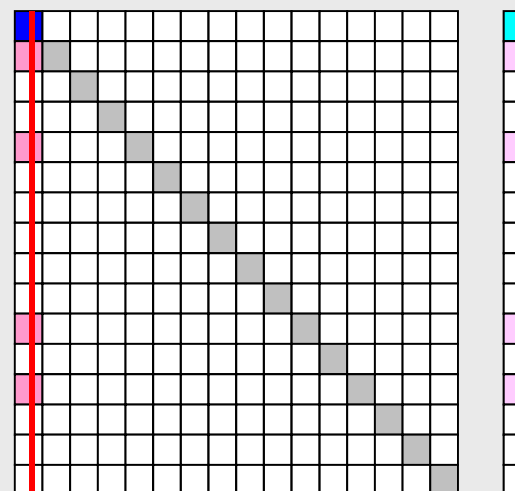
      do k= 1, NPLU
        if (ITEM(k).eq. 1) AMAT(k)= 0. d0
      enddo
!C===

```

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)

$T_1=0$
対角成分=1, 右辺=0, 非対角成分=0

消去, ゼロクリア



復習: 一次元問題

第一種境界条件が $T \neq 0$ の場合

```
!C
!C +-----+
!C | BOUNDARY CONDITIONS |
!C +-----+
!C===
```

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

```
!C
!C-- X=Xmin
      i= 1
      jS= INDEX(i-1)
```

$$Diag_j \phi_j + \sum_{k=Index[j-1]+1}^{Index[j]} Amat_k \phi_{Item[k]} = Rhs_j$$

```
      AMAT(jS+1)= 0. d0
      DIAG(i)= 1. d0
      RHS (i)= PHImin
```

```
      do i= 1, N
        do k= INDEX(i-1)+1, INDEX(i)
          if (ITEM(k).eq.1) then
            RHS (i)= RHS(i) - AMAT(k)*PHImin
            AMAT(k)= 0. d0
          endif
        enddo
      enddo
```

```
!C===
```

復習：一次元問題

第一種境界条件が $T \neq 0$ の場合

```
!C
!C +-----+
!C | BOUNDARY CONDITIONS |
!C +-----+
!C===
```

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

```
!C
!C-- X=Xmin
      i= 1
      jS= INDEX(i-1)
```

```
      AMAT(jS+1)= 0. d0
      DIAG(i)= 1. d0
      RHS (i)= PHImin
```

```
      do i= 1, N
        do k= INDEX(i-1)+1, INDEX(i)
          if (ITEM(k).eq.1) then
            RHS (i)= RHS(i) - AMAT(k)*PHImin
            AMAT(k)= 0. d0
          endif
        enddo
      enddo
```

```
!C===
```

$$\begin{aligned}
 & \text{Diag}_j \phi_j + \sum_{k=\text{Index}[j-1]+1, k \neq k_s}^{\text{Index}[j]} \text{Amat}_k \phi_{\text{Item}[k]} \\
 & = \text{Rhs}_j - \text{Amat}_{k_s} \phi_{\text{Item}[k_s]} \\
 & = \text{Rhs}_j - \text{Amat}_{k_s} T_{\min} \quad \text{where } \text{Item}(k_s) = 1
 \end{aligned}$$

復習：一次元問題

境界条件 : MAT_ASS_BC (2/2)

```

do in= 1, N
  if (IWKX(in,1).eq.1) then
    B(in)= 0.d0
    D(in)= 1.d0

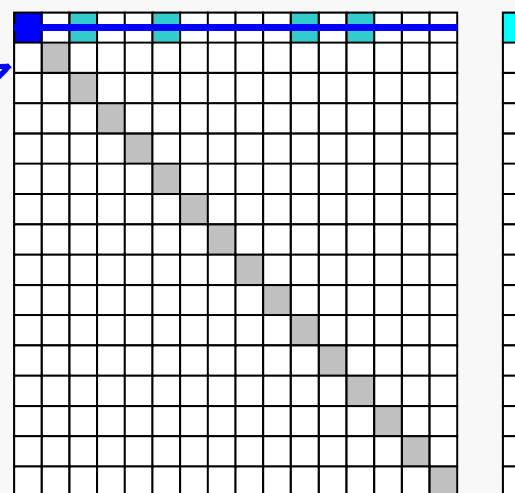
    iS= index(in-1) + 1
    iE= index(in )
    do k= iS, iE
      AMAT(k)= 0.d0
    enddo
  endif
enddo

do in= 1, N
  iS= index(in-1) + 1
  iE= index(in )
  do k= iS, iE
    if (IWKX(item(k),1).eq.1) then
      AMAT(k)= 0.d0
    endif
  enddo
enddo
!C==
return
end

```

IWKX(in,1)=1となる節点に対して
対角成分=1, 右辺=0, 非零対角成分=0

ゼロクリア



ここでやっていることも一次元の時と
全く変わらない

境界条件 : MAT_ASS_BC (2/2)

```

do in= 1, N
  if (IWKX(in,1).eq.1) then
    B(in)= 0. d0
    D(in)= 1. d0

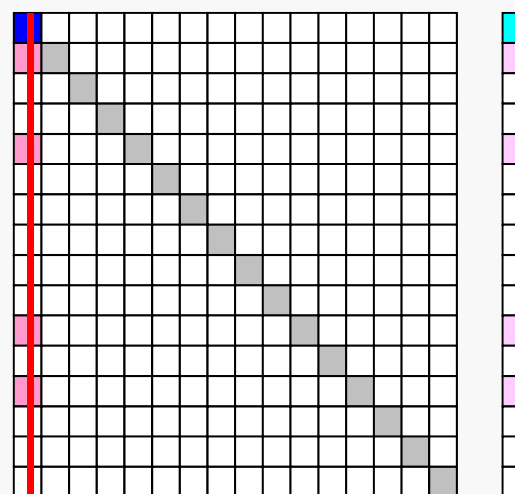
    iS= index(in-1) + 1
    iE= index(in )
    do k= iS, iE
      AMAT(k)= 0. d0
    enddo
  endif
enddo

do in= 1, N
  iS= index(in-1) + 1
  iE= index(in )
  do k= iS, iE
    if (IWKX(item(k),1).eq.1) then
      AMAT(k)= 0. d0
    endif
  enddo
enddo

!C==
return
end

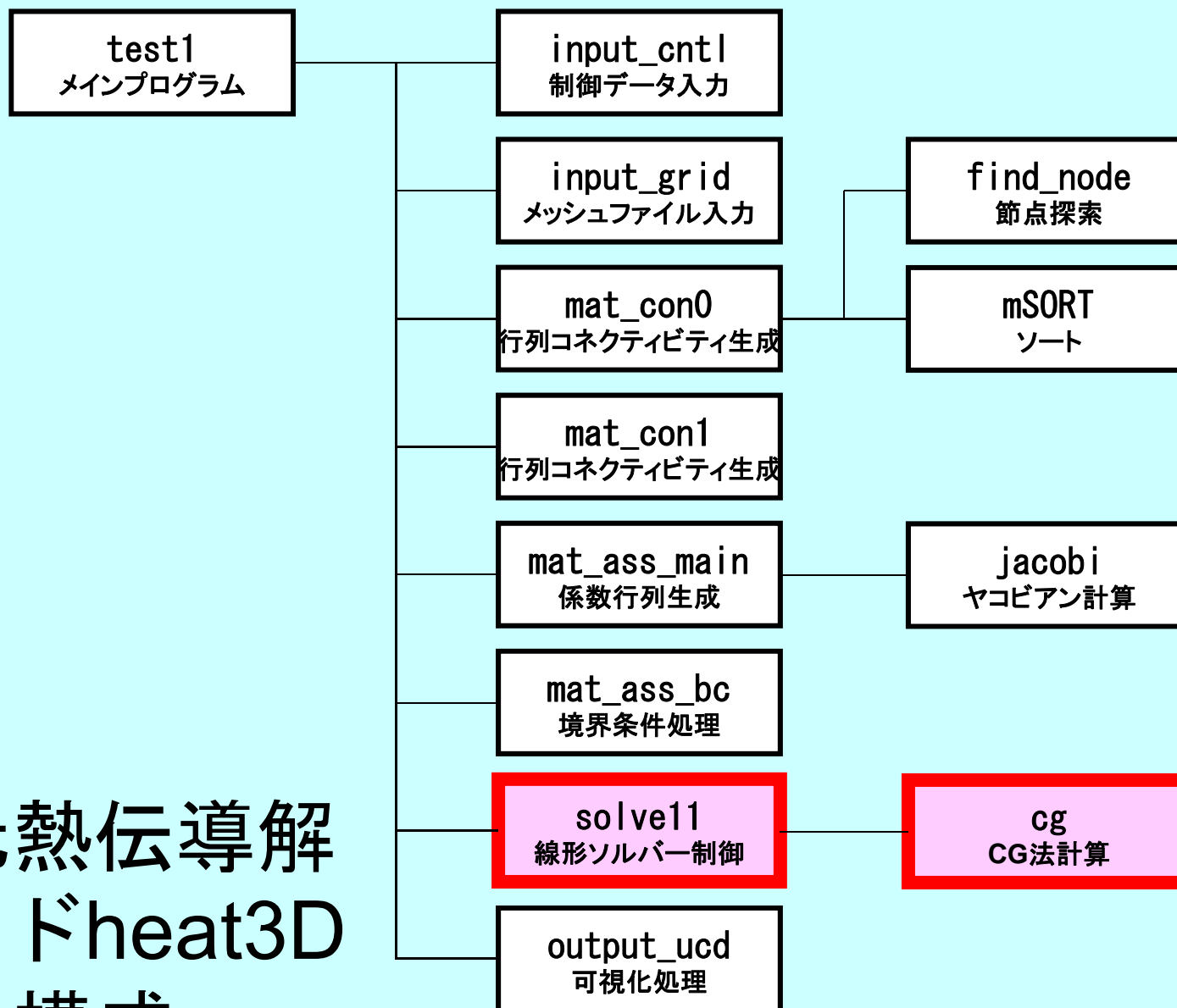
```

IWKX(in,1)=1となる節点を非零非対角成分として有している節点に対して、右辺へ移項, 当該非零非対角成分=0



消去, ゼロクリア

ここでやっていることも一次元の時と全く変わらない



三次元熱伝導解析コードheat3Dの構成

全体処理

```
program heat3D

use solver11
use pfem_util

implicit REAL*8(A-H, O-Z)

call INPUT_CNTL
call INPUT_GRID

call MAT_CONO
call MAT_CON1

call MAT_ASS_MAIN
call MAT_ASS_BC

call SOLVE11

call OUTPUT_UCD

end program heat3D
```

SOLVE11

```

module SOLVER11
contains
  subroutine SOLVE11

    use pfem_util
    use solver_CG

    implicit REAL*8 (A-H, O-Z)

    integer :: ERROR, ICFLAG
    character(len=char_length) :: BUF

    data ICFLAG/0/

!C
!C +-----+
!C | PARAMETERS |
!C +-----+
!C==
!C
!C
!C == ITER      = pfemIarray(1)      CG法の最大反復回数
!C == RESID    = pfemRarray(1)      CG法の収束判定値
!C
!C==
!C
!C +-----+
!C | ITERATIVE solver |
!C +-----+
!C==
!C
!C == call CG
!C == &      ( N, NPLU, D, AMAT, index, item, B, X, RESID, ITER, ERROR ) &
!C
!C == ITERactual= ITER
!C==

    end subroutine SOLVE11
  end module SOLVER11

```

前処理付き共役勾配法

Preconditioned Conjugate Gradient Method (CG)

```

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

```

前処理: 対角スケーリング

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

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

- **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。

CGソルバー(1/6)

```

module solver_CG
contains

  subroutine CG                                     &
    & (N, NPLU, D, AMAT, index, item, B, X, RESID, ITER, ERROR)

    implicit REAL*8(A-H, O-Z)
    include 'precision.inc'

    integer(kind=kint ), intent(in):: N, NPLU
    integer(kind=kint ), intent(inout):: ITER, ERROR
    real (kind=kreal), intent(inout):: RESID
    real(kind=kreal), dimension(N), intent(inout):: B, X, D
    real(kind=kreal), dimension(NPLU), intent(inout):: AMAT
    integer(kind=kint ), dimension(0:N ), intent(in) :: index
    integer(kind=kint ), dimension(NPLU), intent(in) :: item

    real(kind=kreal), dimension(:, :), allocatable :: WW

    integer(kind=kint), parameter :: R= 1
    integer(kind=kint), parameter :: Z= 2
    integer(kind=kint), parameter :: Q= 2
    integer(kind=kint), parameter :: P= 3
    integer(kind=kint), parameter :: DD= 4

    integer(kind=kint ) :: MAXIT
    real (kind=kreal) :: TOL, W, SS

```

CGソルバー (1/6)

```
module solver_CG
```

```

WW (i, 1) = WW (i, R)   ⇒ {r}
WW (i, 2) = WW (i, Z)   ⇒ {z}
WW (i, 2) = WW (i, Q)   ⇒ {q}
WW (i, 3) = WW (i, P)   ⇒ {p}
WW (i, 4) = WW (i, DD)  ⇒ 1/{D}

```

```

integer (kind=kint ), intent(inout) :: ITER, LRROR
real (kind=kreal), intent(inout) :: RESID
real (kind=kreal), dimension(N), intent(inout) ::
real (kind=kreal), dimension(NPLU), intent(inout) ::
integer (kind=kint ), dimension(0:N ), intent(in) ::
integer (kind=kint ), dimension(NPLU), intent(in) ::

```

```
real (kind=kreal), dimension(:, :), allocatable ::
```

```

integer (kind=kint), parameter :: R= 1
integer (kind=kint), parameter :: Z= 2
integer (kind=kint), parameter :: Q= 2
integer (kind=kint), parameter :: P= 3
integer (kind=kint), parameter :: DD= 4

```

```

integer (kind=kint ) :: MAXIT
real (kind=kreal) :: TOL, W, SS

```

```

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

```

CGソルバー (2/6)

```

!C
!C +-----+
!C | INIT. |
!C +-----+
!C===
      ERROR= 0
      allocate (WW(N,4))
      MAXIT = ITER
      TOL   = RESID
      X = 0. d0
!C===

!C +-----+
!C | {r0}= {b} - [A] {xini} |
!C +-----+
!C===

      do j= 1, N
      WW(j, DD)= 1. d0/D(j)
      WVAL= B(j) - D(j)*X(j)
      do k= index(j-1)+1, index(j)
      i= item(k)
      WVAL= WVAL - AMAT(k)*X(i)
      enddo
      WW(j, R)= WVAL
      enddo

```

$$\begin{aligned}
 WW(i, 1) &= WW(i, R) && \Rightarrow \{r\} \\
 WW(i, 2) &= WW(i, Z) && \Rightarrow \{z\} \\
 WW(i, 2) &= WW(i, Q) && \Rightarrow \{q\} \\
 WW(i, 3) &= WW(i, P) && \Rightarrow \{p\} \\
 WW(i, 4) &= WW(i, DD) && \Rightarrow 1/\{D\}
 \end{aligned}$$

対角成分の逆数(前処理用)
 その都度、除算をすると効率が
 悪いため、予め配列に格納

CGソルバー(2/6)

```

!C
!C +-----+
!C | INIT. |
!C +-----+
!C===
      ERROR= 0
      allocate (WW(N, 4))
      MAXIT  = ITER
      TOL    = RESID
      X = 0. d0
!C===

!C +-----+
!C | {r0} = {b} - [A] {xini} |
!C +-----+
!C===
      do j= 1, N
        WW(j, DD)= 1. d0/D(j)
        WVAL= B(j) - D(j)*X(j)
        do k= index(j-1)+1, index(j)
          i= item(k)
          WVAL= WVAL - AMAT(k)*X(i)
        enddo
        WW(j, R)= WVAL
      enddo

```

Compute $\mathbf{r}^{(0)} = \mathbf{b} - [\mathbf{A}]\mathbf{x}^{(0)}$

```

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) = z(i-1) + βi-1 p(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

```


CGソルバー (3/6)

```

BNRM2= 0. d0
do i= 1, N
  BNRM2= BNRM2 + B(i)**2
enddo

```

```

BNRM2= BNRM2

```

```

if (BNRM2. eq. 0. d0) BNRM2= 1. d0
ITER = 0

```

```

!C===

```

```

do iter= 1, MAXIT

```

```

!C

```

```

!C***** Conjugate Gradient Iteration

```

```

!C

```

```

!C +-----+

```

```

!C | {z} = [Minv] {r} |

```

```

!C +-----+

```

```

!C===

```

```

do i= 1, N

```

```

  WW(i, Z) = WW(i, R) * WW(i, DD)

```

```

enddo

```

```

!C===

```

$BNRM2 = |b|^2$

あとで収束判定に使用

CGソルバー (3/6)

```

BNRM2= 0. d0
do i= 1, N
  BNRM2= BNRM2 + B(i)**2
enddo

BNRM2= BNRM2

if (BNRM2. eq. 0. d0) BNRM2= 1. d0
ITER = 0
!C===

do iter= 1, MAXIT
!C
!C*****
!C
!C +-----+
!C | {z}= [Minv] {r} |
!C +-----+
!C===
do i= 1, N
  WW(i, Z)= WW(i, R) * WW(i, DD)
enddo
!C===

```

```

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

```

CGソルバー(4/6)

```

!C
!C +-----+
!C | {RHO}= {r} {z} |
!C +-----+
!C===
      RH00= 0. d0
      do i= 1, N
        RH00= RH00 + WW(i, R)*WW(i, Z)
      enddo
      RHO= RH00
!C===
!C +-----+
!C | {p} = {z} if      ITER=1      |
!C | BETA= RHO / RH01  otherwise  |
!C +-----+
!C===
      if ( ITER. eq. 1 ) then
        do i= 1, N
          WW(i, P)= WW(i, Z)
        enddo
      else
        BETA= RHO / RH01
        do i= 1, N
          WW(i, P)= WW(i, Z) + BETA*WW(i, P)
        enddo
      endif
!C===

```

```

Compute  $r^{(0)} = b - [A]x^{(0)}$ 
for  $i = 1, 2, \dots$ 
  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

```

CGソルバー (5/6)

```

!C +-----+
!C | {q} = [A] {p} |
!C +-----+
!C===
      do j= 1, N
          WVAL= D(j)*WW(j,P)
          do k= index(j-1)+1, index(j)
              i= item(k)
              WVAL= WVAL + AMAT(k)*WW(i,P)
          enddo
          WW(j,Q)= WVAL
      enddo
!C===
!C +-----+
!C | ALPHA= RHO / {p} {q} |
!C +-----+
!C===
      C10= 0.d0
      do i= 1, N
          C10= C10 + WW(i,P)*WW(i,Q)
      enddo
      C1= C10
      ALPHA= RHO / C1
!C===

```

```

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

```

CGソルバー (6/6)

```

!C
!C +-----+
!C | {x} = {x} + ALPHA*{p} |
!C | {r} = {r} - ALPHA*{q} |
!C +-----+
!C===
      do i= 1, N
          X(i) = X (i)  + ALPHA * WW(i, P)
          WW(i, R) = WW(i, R) - ALPHA * WW(i, Q)
      enddo
!C===
      DNRM2= 0. d0
      do i= 1, N
          DNRM2= DNRM2 + WW(i, R)**2
      enddo
      DNRM2= DNRM2
      RESID= dsqrt(DNRM2/BNRM2)

      if ( RESID. le. TOL  ) exit
      if ( ITER .eq. MAXIT ) ERROR= -300

      RH01 = RHO
    enddo
!C===

```

```

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

```

CGソルバー (6/6)

```

!C
!C +-----+
!C | {x} = {x} + ALPHA*{p} |
!C | {r} = {r} - ALPHA*{q} |
!C +-----+
!C===
      do i= 1, N
          X(i) = X (i)  + ALPHA * WW(i,P)
          WW(i,R)= WW(i,R) - ALPHA * WW(i,Q)
      enddo
!C===
      DNRM2= 0. d0
      do i= 1, N
          DNRM2= DNRM2 + WW(i,R)**2
      enddo
      DNRM2= DNRM2
      RESID= dsqrt(DNRM2/BNRM2)

      if ( RESID.le.TOL  ) exit
      if ( ITER .eq. MAXIT ) ERROR= -300

      RH01 = RH0
    enddo
!C===

```

```

Compute  $r^{(0)} = b - [A]x^{(0)}$ 
for  $i = 1, 2, \dots$ 
    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

```

$$\text{Resid} = \sqrt{\frac{\text{DNorm2}}{\text{BNorm2}}} = \frac{|r|}{|b|} = \frac{|Ax - b|}{|b|} \leq \text{Tol}$$