ボクセル有限要素法

1　ボクセル解析に用いる有限要素

ボクセル解析には，図 1.1 に示す 8 節点長方柱要素を用いる。なお，要素の局所座標系(x, y, z)の原点は要素の中心位置とする。

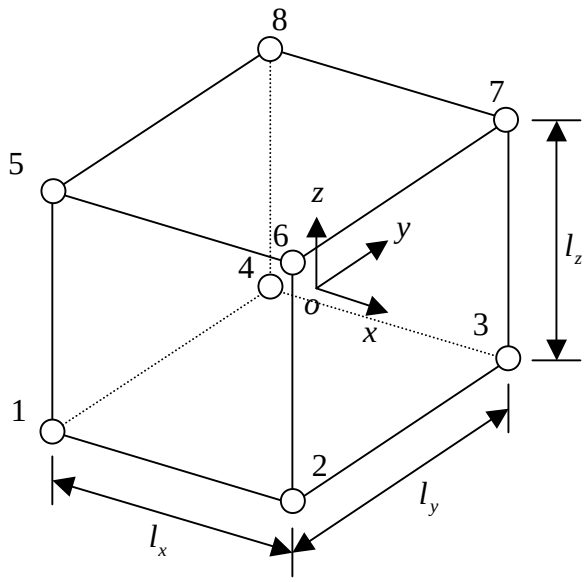

図 1.1　8 節点長方柱要素（1 次要素）

要素内部の変位$\{u\}$を，形状関数を用いて節点の変位$\{\delta\}$で次式のように表す。

$$\{u\}=\begin{Bmatrix} u \\ v \\ w \end{Bmatrix}=\begin{bmatrix} N_1 & 0 & 0 & N_2 & 0 & 0 & \cdots & N_8 & 0 & 0 \\ 0 & N_1 & 0 & 0 & N_2 & 0 & \cdots & 0 & N_8 & 0 \\ 0 & 0 & N_1 & 0 & 0 & N_2 & \cdots & 0 & 0 & N_8 \end{bmatrix}\{\delta\}=[N]\{\delta\} \tag{1.1}$$

ここに，

$$\begin{aligned} N_1&=\frac{1}{8}(1-2x/l_x)(1-2y/l_y)(1-2z/l_z) & N_5&=\frac{1}{8}(1-2x/l_x)(1-2y/l_y)(1+2z/l_z) \\ N_2&=\frac{1}{8}(1+2x/l_x)(1-2y/l_y)(1-2z/l_z) & N_6&=\frac{1}{8}(1+2x/l_x)(1-2y/l_y)(1+2z/l_z) \\ N_3&=\frac{1}{8}(1+2x/l_x)(1+2y/l_y)(1-2z/l_z) & N_7&=\frac{1}{8}(1+2x/l_x)(1+2y/l_y)(1+2z/l_z) \\ N_4&=\frac{1}{8}(1-2x/l_x)(1+2y/l_y)(1-2z/l_z) & N_8&=\frac{1}{8}(1-2x/l_x)(1+2y/l_y)(1+2z/l_z) \end{aligned} \tag{1.2}$$

$$\{\delta\}^T=\{u_1 \quad v_1 \quad w_1 \quad u_2 \quad v_2 \quad w_2 \quad \cdots \quad u_8 \quad v_8 \quad w_8\} \tag{1.3}$$

ひずみ－変位関係式は，

$$\{\varepsilon\}=\begin{Bmatrix}\varepsilon_x\\ \varepsilon_y\\ \varepsilon_z\\ \gamma_{xy}\\ \gamma_{yz}\\ \gamma_{zx}\end{Bmatrix}=\begin{Bmatrix}\partial u/\partial x\\ \partial v/\partial y\\ \partial w/\partial z\\ \partial u/\partial y+\partial v/\partial x\\ \partial v/\partial z+\partial w/\partial y\\ \partial w/\partial x+\partial u/\partial z\end{Bmatrix}=\begin{bmatrix}[B_1] & [B_2] & \cdots & [B_8]\end{bmatrix}\{\delta\} \tag{1.4}$$

ここに,

$$[B_i]=\begin{bmatrix}\partial N_i/\partial x & 0 & 0\\ 0 & \partial N_i/\partial y & 0\\ 0 & 0 & \partial N_i/\partial z\\ \partial N_i/\partial y & \partial N_i/\partial x & 0\\ 0 & \partial N_i/\partial z & \partial N_i/\partial y\\ \partial N_i/\partial z & 0 & \partial N_i/\partial x\end{bmatrix} \tag{1.5}$$

形状関数の微分は次式により計算される。

$$\frac{\partial N_1}{\partial x}=-\frac{1}{4l_x}\left(1-\frac{2y}{l_y}\right)\left(1-\frac{2z}{l_z}\right),\quad \frac{\partial N_1}{\partial y}=-\frac{1}{4l_y}\left(1-\frac{2x}{l_x}\right)\left(1-\frac{2z}{l_z}\right),\quad \frac{\partial N_1}{\partial z}=-\frac{1}{4l_z}\left(1-\frac{2x}{l_x}\right)\left(1-\frac{2y}{l_y}\right)$$
$$\frac{\partial N_2}{\partial x}=\frac{1}{4l_x}\left(1-\frac{2y}{l_y}\right)\left(1-\frac{2z}{l_z}\right),\quad \frac{\partial N_2}{\partial y}=-\frac{1}{4l_y}\left(1+\frac{2x}{l_x}\right)\left(1-\frac{2z}{l_z}\right),\quad \frac{\partial N_2}{\partial z}=-\frac{1}{4l_z}\left(1+\frac{2x}{l_x}\right)\left(1-\frac{2y}{l_y}\right)$$
$$\frac{\partial N_3}{\partial x}=\frac{1}{4l_x}\left(1+\frac{2y}{l_y}\right)\left(1-\frac{2z}{l_z}\right),\quad \frac{\partial N_3}{\partial y}=\frac{1}{4l_y}\left(1+\frac{2x}{l_x}\right)\left(1-\frac{2z}{l_z}\right),\quad \frac{\partial N_3}{\partial z}=-\frac{1}{4l_z}\left(1+\frac{2x}{l_x}\right)\left(1+\frac{2y}{l_y}\right)$$
$$\frac{\partial N_4}{\partial x}=-\frac{1}{4l_x}\left(1+\frac{2y}{l_y}\right)\left(1-\frac{2z}{l_z}\right),\quad \frac{\partial N_4}{\partial y}=\frac{1}{4l_y}\left(1-\frac{2x}{l_x}\right)\left(1-\frac{2z}{l_z}\right),\quad \frac{\partial N_4}{\partial z}=-\frac{1}{4l_z}\left(1-\frac{2x}{l_x}\right)\left(1+\frac{2y}{l_y}\right)$$
$$\frac{\partial N_5}{\partial x}=-\frac{1}{4l_x}\left(1-\frac{2y}{l_y}\right)\left(1+\frac{2z}{l_z}\right),\quad \frac{\partial N_5}{\partial y}=-\frac{1}{4l_y}\left(1-\frac{2x}{l_x}\right)\left(1+\frac{2z}{l_z}\right),\quad \frac{\partial N_6}{\partial z}=\frac{1}{4l_z}\left(1-\frac{2x}{l_x}\right)\left(1-\frac{2y}{l_y}\right)$$
$$\frac{\partial N_6}{\partial x}=\frac{1}{4l_x}\left(1-\frac{2y}{l_y}\right)\left(1+\frac{2z}{l_z}\right),\quad \frac{\partial N_6}{\partial y}=-\frac{1}{4l_y}\left(1+\frac{2x}{l_x}\right)\left(1+\frac{2z}{l_z}\right),\quad \frac{\partial N_6}{\partial z}=\frac{1}{4l_z}\left(1+\frac{2x}{l_x}\right)\left(1-\frac{2y}{l_y}\right)$$
$$\frac{\partial N_7}{\partial x}=\frac{1}{4l_x}\left(1+\frac{2y}{l_y}\right)\left(1+\frac{2z}{l_z}\right),\quad \frac{\partial N_7}{\partial y}=\frac{1}{4l_y}\left(1+\frac{2x}{l_x}\right)\left(1+\frac{2z}{l_z}\right),\quad \frac{\partial N_7}{\partial z}=\frac{1}{4l_z}\left(1+\frac{2x}{l_x}\right)\left(1+\frac{2y}{l_y}\right)$$
$$\frac{\partial N_8}{\partial x}=-\frac{1}{4l_x}\left(1+\frac{2y}{l_y}\right)\left(1+\frac{2z}{l_z}\right),\quad \frac{\partial N_8}{\partial y}=\frac{1}{4l_y}\left(1-\frac{2x}{l_x}\right)\left(1+\frac{2z}{l_z}\right),\quad \frac{\partial N_8}{\partial z}=\frac{1}{4l_z}\left(1-\frac{2x}{l_x}\right)\left(1+\frac{2y}{l_y}\right) \tag{1.6}$$

弾性応力－ひずみ関係式は次式のように表せる。

$$\{\sigma\}=\begin{Bmatrix}\sigma_x\\ \sigma_y\\ \sigma_z\\ \tau_{xy}\\ \tau_{yz}\\ \tau_{zx}\end{Bmatrix}=[D]\{\varepsilon\} \tag{1.7}$$

等方弾性体の場合の応力－ひずみマトリックス$[D]$は次式となる。

$$[D]=\frac{E(1-\nu)}{(1+\nu)(1-2\nu)}\begin{bmatrix} 1 & \frac{\nu}{1-\nu} & \frac{\nu}{1-\nu} & 0 & 0 & 0 \\ & 1 & \frac{\nu}{1-\nu} & 0 & 0 & 0 \\ & & 1 & 0 & 0 & 0 \\ & & & \frac{1-2\nu}{2(1-\nu)} & 0 & 0 \\ & & & & \frac{1-2\nu}{2(1-\nu)} & 0 \\ & \text{sym.} & & & & \frac{1-2\nu}{2(1-\nu)} \end{bmatrix} \tag{1.8}$$

ここに，Eはヤング係数，νはポアソン比である。

要素剛性マトリックスは次式のようになる。

$$[k]=\int_{-\frac{l_x}{2}}^{\frac{l_x}{2}}\int_{-\frac{l_y}{2}}^{\frac{l_y}{2}}\int_{-\frac{l_z}{2}}^{\frac{l_z}{2}}[B]^T[D][B]dxdydz \tag{1.9}$$

上式の積分は解析的に積分することが可能である。しかしながら，剛性マトリックス$[K]$は24×24のマトリックスであるため576の成分を有する。したがって，対称性を考慮しても300成分の計算値をプログラムに書く必要がある。この要素剛性マトリックスを何回も計算する必要がある場合は，計算負荷を低減するために解析解を用いる価値があるが，ボクセル解析では高々材料種別数分計算すればよいので，プログラムが複雑になることを考えると数値積分を用いる方が得策である。そこでここでは，(1.9)式をGauss-Legendre求積法で計算することにする。

Gauss-Legendre求積法では，積分点が$(-1,1)$の範囲で与えられるため，座標x,y,zを次式で表す。

$$x=\frac{l_x}{2}\xi,\quad y=\frac{l_y}{2}\eta,\quad z=\frac{l_z}{2}\zeta \tag{1.10}$$

このとき，(10)式は次式で計算される。

$$[k]=\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\left[B(\xi_i,\eta_j,\zeta_k)\right]^T[D]\left[B(\xi_i,\eta_j,\zeta_k)\right]\cdot\frac{l_x}{2}\cdot\frac{l_y}{2}\cdot\frac{l_z}{2}\cdot w_i\cdot w_j\cdot w_k \tag{1.11}$$

ここに，ξ_i,η_j,ζ_kはx,y,z方向の積分点，w_i,w_j,w_kは積分点に対する重み係数である。また，nは積分点数である。Gauss-Legendre求積法では，$2n-1$次の多項式まで厳密な積分が行えるため，ここでは$n=2$とする。この場合の積分点と重みは次のようになる。

積分点	–0.5773502692	0.5773502692
重み	1.0000000000	1.0000000000

【メモ】

ボクセル解析では，要素サイズはすべての要素について同じであるから，要素剛性マトリックスは，ヤング係数とポアソン比が異なる材料種別数分計算して，それを保存しておけばよい。また，この方法をトポロジー解析に適用する場合，密度法を採用すればこれと同じ計算量ですむ。なぜならば，密度法は，密度の変化によってヤング係数のみが変化するので，剛性マトリックス

に密度係数を掛けて，これを変化させるだけでよい（$\rho^p[k]$，ここに ρ は密度， p はべき係数)。しかし，均質化法を適用した場合，各要素で，応力－ひずみマトリックス $[D]$ が異なるため，すべての要素について剛性マトリックスの計算を行う必要がある。したがって，ボクセル解析に均質化法によるトポロジー解析を応用する場合，設計変数が６倍（実際は３倍，なぜなら角度は応力の主軸方向にするため）になるだけでなく，要素剛性マトリックスを各要素で計算しなければならないという負荷を負うことになる。

２ 簡単な自動メッシュ生成

まず，設計対象領域を囲む長方柱（図 1）を考え，その各辺の長さを L_x, L_y, L_z とする。また，各辺の有限要素分割数を n_x, n_y, n_z とする。このとき長方柱有限要素の各辺の長さは次式で定義される。

$$l_x = \frac{L_x}{n_x}, \quad l_y = \frac{L_y}{n_y}, \quad l_z = \frac{L_z}{n_z} \tag{2.1}$$

また，この場合，長方柱各辺の節点数は， $(n_x+1), (n_y+1), (n_z+1)$ であり，全節点数は，$(n_x+1)(n_y+1)(n_z+1)$ となる。

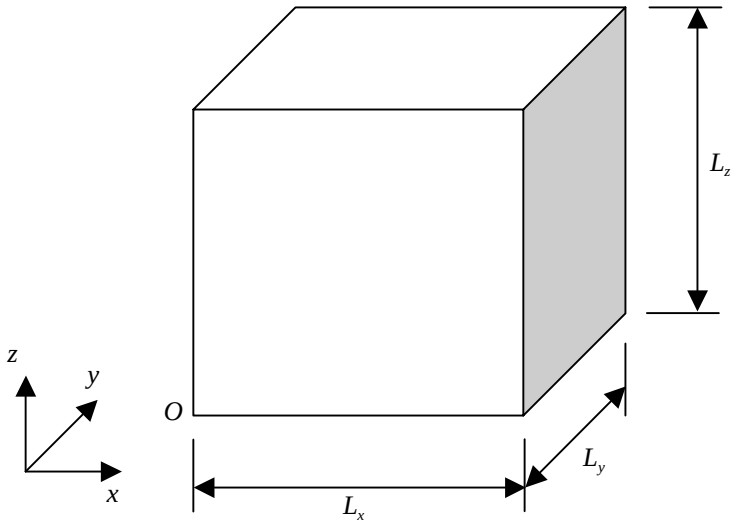

図 2.1　設計領域を囲む長方柱

節点の番号付けは，図 2.1 の O 点を原点とし，まず xy 面に x 方向に節点の番号付けを行い，すべて付け終わったら z 方向を l_z だけ増分させ同じことを繰り返す。すなわち，これをプログラムに書くと次のようになる。

```
do i = 1,nz+1
do j = 1,ny+1
do k = 1,nx+1
n = (ny+1)*(nx+1)*(i-1)+(nx+1)*(j-1)+k    ! 節点番号
x(n) = dble(k-1)*elx      ! 節点の x 座標
y(n) = dble(j-1)*ely      ! 節点の y 座標
z(n) = dble(i-1)*elz      ! 節点の z 座標
```

```
end do
end do
end do
```

次に要素も節点と同様に原点Oから，xy面のx方向から順に番号付けを行う。この場合，各要素の節点番号は，プログラムに書くと次のように割り当てられる。

```
do i = 1,nz
do j = 1,ny
do k = 1,nx
n = ny*nx*(i-1)+nx*(j-1)+k      ! 要素番号
indv(n,1) = (ny+1)*(nx+1)*(i-1)+(nx+1)*(j-1)+k
indv(n,2) = (ny+1)*(nx+1)*(i-1)+(nx+1)*(j-1)+k+1
indv(n,3) = (ny+1)*(nx+1)*(i-1)+(nx+1)*j+k+1
indv(n,4) = (ny+1)*(nx+1)*(i-1)+(nx+1)*j+k
indv(n,5) = (ny+1)*(nx+1)*i+(nx+1)*(j-1)+k
indv(n,6) = (ny+1)*(nx+1)*i+(nx+1)*(j-1)+k+1
indv(n,7) = (ny+1)*(nx+1)*i+(nx+1)*j+k+1
indv(n,8) = (ny+1)*(nx+1)*i+(nx+1)*j+k
end do
end do
end do
```

有限要素解析を行うためには，この外に，材料種別番号を要素に与え，個々の材料種別のヤング係数，ポアソン比等の情報が必要となる。また，境界条件，荷重条件を節点自由度に対応する形で与えることが必要である。

３　ボクセル解析の解析例

簡単な自動メッシュ生成法を用いて入力データを作成し，ソルバーとして前処理付き共役勾配法を用いたプログラムを作成した。また，比較のためスカイラインソルバーを用いた解析プログラムもあわせて作成した。そして，簡単な例題として，図 3.1 に示すモデルの解析を行い，連立方程式を解くのに要する計算時間と解析精度を比較した。なお，メッシュ分割数は10×10×10 の1000 要素とした。

表 3.1 は，直接解法（スカイライン法）と，反復法（共役勾配法）の計算時間，および荷重点での変位を比較したものである。表より，反復解法の計算時間が直接解法に比較して約 20 倍速いことがわかる。また，変位も 4 桁までは一致した。

表 3.2 は，反復解法における要素分割数と計算時間の関係を示したものである。また，図 3.2 はこれを図示したものである。

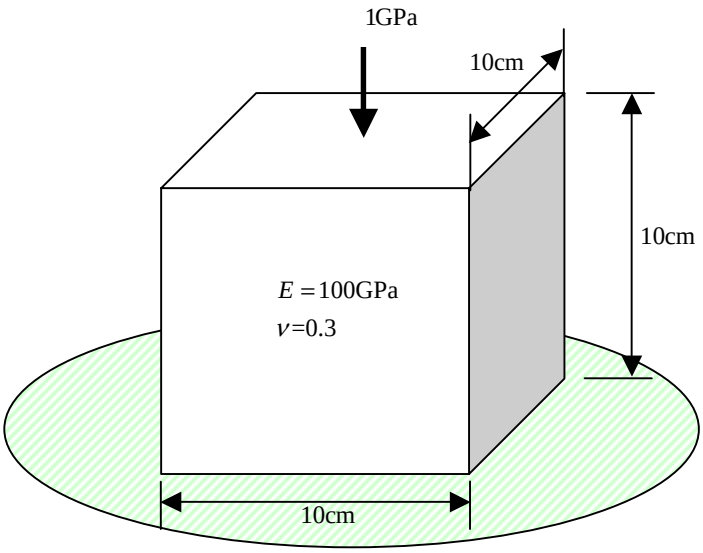

図 3.1　解析モデル

表 3.1　直接解法と反復解法の比較

	PCG solver	Skyline solver
CPU time (Intel Celeron 466MHz)	4.45 sec	91.89 sec
Displacement	0.01551 cm	0.01551 cm

表 3.2　要素数と計算時間の関係

要素分割	要素総数	CPU time (Intel Celeron 466MHz)
10×10×10	1000	4.45 sec
20×20×20	8000	51.7 sec
30×30×30	27000	232.7 sec
40×40×40	64000	704.1 sec

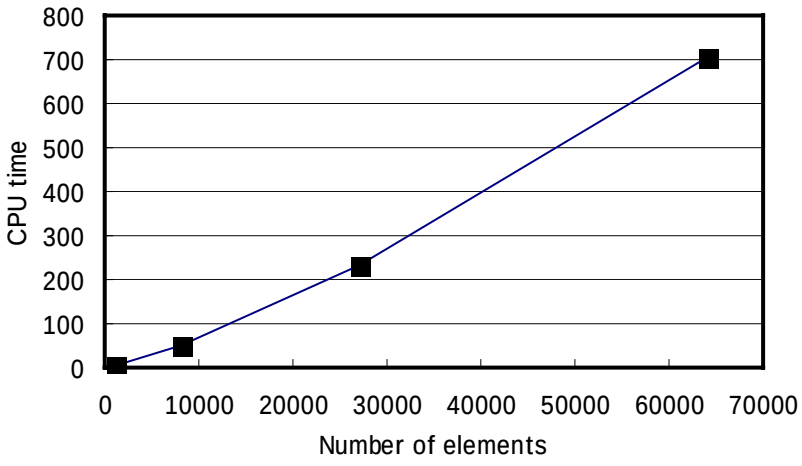

図 3.2　要素数と計算時間の関係

４　要素の節点番号情報を与えるサブルーチン

本プログラムのディメンションの主なものは表 3.1 に示す通りである。表 3.1 から明らかなように, 要素の節点番号情報のディメンションが非常に大きな割合を占めている。ボクセル解析では, 要素を規則正しく分割するので，分割ルールを決めてしまえば要素の節点番号情報を蓄えておく必要はなく，その都度計算すればよい。

ところで，２節の「簡単な自動メッシュ生成」で示したように，今採用している要素の分割は以下のようなものであった。

```
do i = 1,nz
do j = 1,ny
do k = 1,nx
n = ny*nx*(i-1)+nx*(j-1)+k      ! 要素番号
indv(n,1) = (ny+1)*(nx+1)*(i-1)+(nx+1)*(j-1)+k
indv(n,2) = (ny+1)*(nx+1)*(i-1)+(nx+1)*(j-1)+k+1
indv(n,3) = (ny+1)*(nx+1)*(i-1)+(nx+1)*j+k+1
indv(n,4) = (ny+1)*(nx+1)*(i-1)+(nx+1)*j+k
indv(n,5) = (ny+1)*(nx+1)*i+(nx+1)*(j-1)+k
indv(n,6) = (ny+1)*(nx+1)*i+(nx+1)*(j-1)+k+1
indv(n,7) = (ny+1)*(nx+1)*i+(nx+1)*j+k+1
indv(n,8) = (ny+1)*(nx+1)*i+(nx+1)*j+k
end do
end do
end do
```

したがって，要素番号 n から逆に，i,j,k が割り出せれば，要素の節点番号を与えるサブルーチンを作ることができる。このサブルーチンは以下のようになる。

```
      subroutine ElmNum(n,indv,nx,ny,nz)
      implicit real*8(a-h,o-z)
      dimension indv(1)
c
      i   = int(n/(nx*ny)) + 1
      mi  = mod(n,nx*ny)
      if( mi.eq.0 ) i = int(n/(nx*ny))
      n1  = n - nx*ny*(i-1)
      j   = int(n1/nx) + 1
      mj  = mod( n1,nx )
      if( mj.eq.0 ) j = int(n1/nx)
      k  = n1 - nx*(j-1)
c
      indv(1) = (ny+1)*(nx+1)*(i-1)+(nx+1)*(j-1)+k
      indv(2) = (ny+1)*(nx+1)*(i-1)+(nx+1)*(j-1)+k+1
      indv(3) = (ny+1)*(nx+1)*(i-1)+(nx+1)*j+k+1
      indv(4) = (ny+1)*(nx+1)*(i-1)+(nx+1)*j+k
      indv(5) = (ny+1)*(nx+1)*i+(nx+1)*(j-1)+k
      indv(6) = (ny+1)*(nx+1)*i+(nx+1)*(j-1)+k+1
      indv(7) = (ny+1)*(nx+1)*i+(nx+1)*j+k+1
      indv(8) = (ny+1)*(nx+1)*i+(nx+1)*j+k
c
      return
      end
```

なお，プログラム内の int は少数点以下を切り捨てて整数にする組み込み関数である。この組み込み関数は整数のわり算には必要ないと思われるが，念のため使用している。

以上のサブルーチンを組み込むことにより，要素の節点情報のディメンションが必要なくなる。したがって，本プログラムで必要なディメンションは以下のようになる。

表 4.1　解析に必要な主なディメンション数とメモリー数

種　別	ディメンション数	メモリー数(byte)
要素の材料種別情報(整数)	[要素数]	[要素数]×4
境界条件の情報(整数)	[節点数]×3	[節点数]×3×4
連立方程式の解法(実数)	[節点数]×3×4	[節点数]×3×4×8

したがって，必要メモリー数は，[要素数]×4byte，[節点数] ×108byte である。128Mbyte のメモリーで約 100 万要素の解析が可能である。したがって，1000 万要素の解析には，1.28Gbyte の計算機が必要となる。

以上の改良を行った後に，図 3.1 の解析モデルで 50 万～100 万要素までの解析を行ってみた。その結果を表 4 に示す。図 4.1 は，表 3.2 に表 4.1 のものを合わせて，要素数と計算時間の関係を表したものである。なお，100 万要素で急に計算時間が増えているのは，実際には必要メモリーが 128Mbyte では不足であり（128Mbyte フルに計算に使用することができないため），ハードディスクにスワップしたためである。

表 4.1　50 万要素以上の解析時間

要素分割	要素総数	CPU time (Intel Celeron 466MHz) [sec]	Displacement [cm]
80×80×80	512000	819.4	0.1232
90×90×90	729000	1712	0.1386
100×100×100	1000000	6213	0.1540

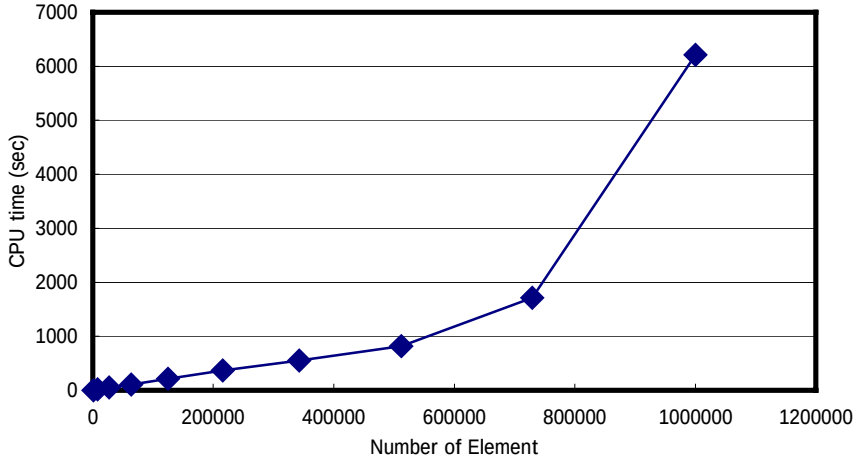

図 4.1　計算時間と要素数の関係