離散ナブラ演算子専用ハードウェア化の検討

A Dedicated Computer for Discrete Nabla Operator

青木 すみえ，慶應大，〒223-8522 横浜市港北区日吉 3-14-1, E-mail: suikoa@hotmail.com
溝口 大介,荒木 健悟,佐々木 徹(株)アプリオリ・マイクロシステムズ,〒223-8522 横浜市港北区日吉 3-14-1
棚橋 隆彦，慶應大，〒223-8522 横浜市港北区日吉 3-14-1, E-mail: taka@mech.keio.ac.jp
Sumie Aoki, Keio University, 3-14-1, Hiyoshi, Kohoku-ku, Yokohama, 223-8522, JAPAN
Takahiko Tanahashi, Keio University, 3-14-1, Hiyoshi, Kohoku-ku, Yokohama, 223-8522, JAPAN

It takes a lot of time to solve Poisson equation in analysis of incompressible flows and calculate discrete nabla operators, which are used for nabla operation in GSMAC FEM. This paper aims to shorten the calculation time dramatically by studying the architecture of the dedicated computer for discrete nabla operators needed many times in solving Poisson equation by GSMAC FEM, considering the speed of providing the data for the calculations, which depends on the speed of memory and the order of providing the data, the way to calculate discrete nabla operators and the structure of floating point adders and memory buffer registers.

１．緒言

大規模流体解析には膨大な計算時間を要する．本研究では GSMAC-FEM の専用ハードウェア化を検討することにより流体解析の高速化を目指す．GSMAC-FEM で用いられる離散ナブラ演算子を用いた演算部分のハードウェア化の検討を行った．

２．GSMAC 有限要素法における Hot Spot

GSMAC-FEM を用いた計算において，計算負荷の高い個所を重点的に高速化するため，計算負荷の高い個所を調べた．例として三次元正方キャビティ強制対流を以下の計算条件について計算した．但しいずれの場合も，時間刻み 5.0×10^{-3} 最大繰り返し回数 500 回 収束判定基準 1.0×10^{-3} である．

①レイノルズ数 100　時間ステップ数　200 回
②レイノルズ数 100　時間ステップ数　50 回
③レイノルズ数 1000　時間ステップ数　200 回
④レイノルズ数 1000　時間ステップ数　50 回

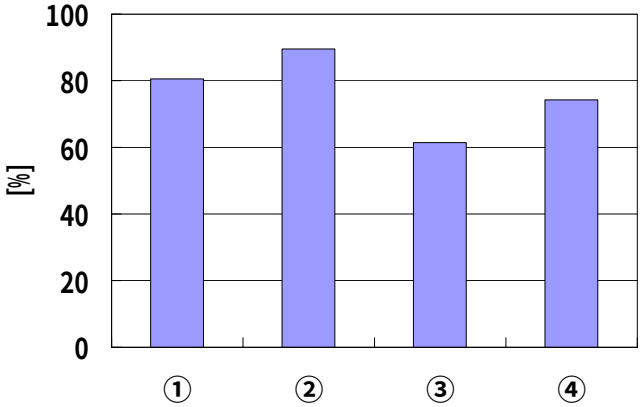

Fig.1 The rate of Poisson equation in GSMAC FEM

レイノルズ数が低いほど，Poisson 方程式の繰り返し回数は増加し，したがって Poisson 方程式の計算時間の占有割合も増加する．また時間ステップの増加にともない定常状態に近づくため，次第に Poisson 方程式の繰り返し回数は減少し，従って計算時間の占有割合も減少する．いずれの場合も全体の計算時間の中で Poisson 方程式の占有割合が圧倒的に多い事が分かる．（Fig.1）従って本研究に於いては Poisson 方程式の高速化に主眼を置いて取り組むことにした．

３．Poisson 方程式

GSMAC 有限要素法において Poisson 方程式は，優対角近似法を用いてラプラスの逆演算子を計算し，速度と圧力の同時緩和法によって解かれる．具体的な計算方法については以下のとおり．（Fig2）

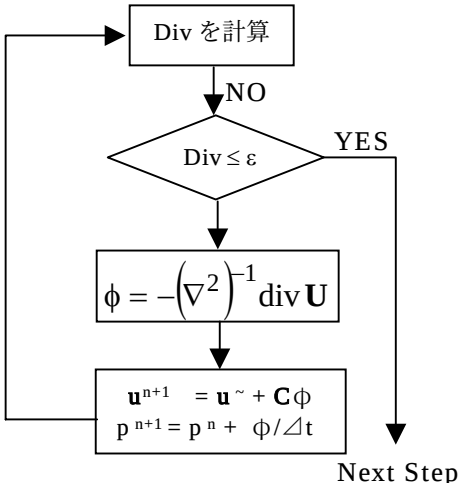

Fig.2 Poisson equation

① 各要素の速度の発散を計算．この際にこう配ベクトルと接点速度を使用する．
② 各要素の修正速度ポテンシャルφを計算．
③ 各要素の各接点ごとに速度の予測子に足しこむ修正部分を **uu** を計算．
それを各接点ごとに予測子の速度に足し込む．
④ 各要素ごとに圧力について修正する．
⑤ 新たに計算された速度と勾配ベクトルを用いて各要素ごとに発散を計算する．ここでその値がある基準を満たしていれば次のステップへ，満たしていないのであれば②へ戻る．

４．離散ナブラ演算子

４．１　物理空間と計算空間

流体の運動を支配する方程式は物理空間のＣartesian 座標（ｘ，ｙ，ｚ）を用いて表される．しかし，任意形状の物体まわりの流れを調べるときにＣartesian 座標を用いると境界条件の取り扱いが複雑になる．そこで，境界条件を取り扱い易くするため，物体の形状に沿った座標系を選ぶ．これは一般曲線座標（ξ，η，ζ）の一種である．（ξ，η，ζ）で表された空間を計算空間と呼ぶ．実際の計算は物理空間から計算空間に写像されてから実行されることになる．

４．１．１　基底ベクトルと逆基底ベクトル

物理空間における基底ベクトルは（$\boldsymbol{i}$，$\boldsymbol{j}$，$\boldsymbol{k}$）でこれらは互いに直交する単位基底ベクトルである．よって$\boldsymbol{i}$，$\boldsymbol{j}$，$\boldsymbol{k}$から作られる逆基底ベクトルを$\boldsymbol{i}^*$，$\boldsymbol{j}^*$，$\boldsymbol{k}^*$ とすれば

$$\boldsymbol{i}^* = \frac{\mathbf{j}\times\mathbf{k}}{\mathbf{i}\cdot\mathbf{j}\times\mathbf{k}} = \boldsymbol{i}$$

$$\boldsymbol{j}^* = \frac{\mathbf{k}\times\mathbf{i}}{\mathbf{j}\cdot\mathbf{k}\times\mathbf{i}} = \boldsymbol{j}$$

$$\boldsymbol{k}^* = \frac{\mathbf{i}\times\mathbf{j}}{\mathbf{k}\cdot\mathbf{i}\times\mathbf{j}} = \boldsymbol{k}$$

となる．つまり$\boldsymbol{i}=\boldsymbol{i}^*$，$\boldsymbol{j}=\boldsymbol{j}^*$，$\boldsymbol{k}=\boldsymbol{k}^*$となり逆基底ベクトルは基の基底ベクトルに等しい．

計算空間における基底ベクトルは各座標軸に接する接ベクトルである．これは

$$\mathbf{g}_1 = \frac{\partial \mathbf{r}}{\partial \xi^1}, \qquad \mathbf{g}_2 = \frac{\partial \mathbf{r}}{\partial \xi^2}, \qquad \mathbf{g}_3 = \frac{\partial \mathbf{r}}{\partial \xi^3}$$

となる．これは大きさ１ではなく，互いにも直交しない．
次に計算空間の座標の勾配から作られるベクトル

$$\mathbf{g}^1 = \nabla\xi^1 \quad , \quad \mathbf{g}^2 = \nabla\xi^2 \quad , \quad \mathbf{g}^3 = \nabla\xi^3$$

を考える．これは計算空間における逆基底ベクトルになっている．すなわち関係式

$$\nabla\xi^{\mathbf{i}} = \frac{\dfrac{\partial \mathbf{r}}{\partial \xi^{\mathrm{j}}}\times\dfrac{\partial \mathbf{r}}{\partial \xi^{\mathrm{k}}}}{\dfrac{\partial \mathbf{r}}{\partial \xi^{\mathrm{i}}}\cdot\dfrac{\partial \mathbf{r}}{\partial \xi^{\mathrm{j}}}\times\dfrac{\partial \mathbf{r}}{\partial \xi^{\mathrm{k}}}} = \frac{\mathbf{g}_{\mathrm{j}}\times\mathbf{g}_{\mathrm{k}}}{\mathbf{g}_{\mathrm{i}}\cdot\mathbf{g}_{\mathrm{i}}\times\mathbf{g}_{\mathrm{k}}} = \mathbf{g}^{\mathrm{i}}$$

$$\frac{\partial \mathbf{r}}{\partial \xi^{\mathrm{i}}} = \frac{\nabla\xi^{\mathrm{j}}\times\nabla\xi^{\mathrm{k}}}{\nabla\xi^{\mathrm{i}}\cdot\nabla\xi^{\mathrm{j}}\times\nabla\xi^{\mathrm{k}}} = \frac{\mathbf{g}^{\mathrm{j}}\times\mathbf{g}^{\mathrm{k}}}{\mathbf{g}^{\mathrm{i}}\cdot\mathbf{g}^{\mathrm{j}}\times\mathbf{g}^{\mathrm{k}}} = \mathbf{g}_{\mathrm{i}}$$

が成立する．ただしｉ，ｊ，ｋ=1，2，3 とする．計算空間における基底ベクトル$\boldsymbol{g}_{\boldsymbol{i}}$は曲線$\xi^{\boldsymbol{i}}$に接し，計算空間における逆基底ベクトル$\boldsymbol{g}^{\boldsymbol{i}}$は面$\xi^{\boldsymbol{i}}$=const に垂直となる．

４．１．２　ヤコビアン

（ξ，η，ζ）に関する（ｘ，ｙ，ｚ）のヤコビアンは

$$\mathrm{J} = \frac{\partial(\mathrm{x},\mathrm{y},\mathrm{z})}{\partial(\xi,\eta,\varsigma)} = \begin{vmatrix} \dfrac{\partial \mathrm{x}}{\partial \xi} & \dfrac{\partial \mathrm{x}}{\partial \eta} & \dfrac{\partial \mathrm{x}}{\partial \varsigma} \\ \dfrac{\partial \mathrm{y}}{\partial \xi} & \dfrac{\partial \mathrm{y}}{\partial \eta} & \dfrac{\partial \mathrm{y}}{\partial \varsigma} \\ \dfrac{\partial \mathrm{z}}{\partial \xi} & \dfrac{\partial \mathrm{z}}{\partial \eta} & \dfrac{\partial \mathrm{z}}{\partial \varsigma} \end{vmatrix} = \frac{\partial \mathbf{r}}{\partial \xi}\cdot\frac{\partial \mathbf{r}}{\partial \eta}\times\frac{\partial \mathbf{r}}{\partial \varsigma} = \left|\frac{\partial \mathrm{x}^{\mathrm{i}}}{\partial \xi^{\mathrm{j}}}\right|$$

の様に表されて，逆基底ベクトルの関係式より

$$\mathbf{g}^{\mathrm{i}} \equiv \nabla\xi^{\mathrm{i}} = \frac{1}{\mathrm{J}}\left(\frac{\partial \mathbf{r}}{\partial \xi^{\mathrm{j}}}\times\frac{\partial \mathbf{r}}{\partial \xi^{\mathrm{k}}}\right) = \frac{1}{\mathrm{J}}(\mathbf{g}_{\mathrm{j}}\times\mathbf{g}_{\mathrm{k}})$$

$$\mathbf{g}_{\mathrm{i}} \equiv \frac{\partial \mathbf{r}}{\partial \xi^{\mathrm{i}}} = \mathrm{J}(\nabla\xi^{\mathrm{j}}\times\nabla\xi^{\mathrm{k}}) = \mathrm{J}(\mathbf{g}^{\mathrm{j}}\times\mathbf{g}^{\mathrm{k}})$$

という関係式が成立する．

ただしここで$\boldsymbol{J}^{-1}$はＪの逆行列で以下の様に表される．

$$\mathbf{J}^{-1} \equiv \frac{\partial \xi^{\mathrm{i}}}{\partial \mathrm{x}^{\mathrm{j}}} = \left(\frac{\partial \mathrm{x}^{\mathrm{j}}}{\partial \xi^{\mathrm{i}}}\right)^{-1} = \frac{1}{|\mathrm{J}|}\begin{bmatrix} \mathrm{A}_{11} & \mathrm{A}_{12} & \mathrm{A}_{13} \\ \mathrm{A}_{21} & \mathrm{A}_{23} & \mathrm{A}_{23} \\ \mathrm{A}_{31} & \mathrm{A}_{32} & \mathrm{A}_{33} \end{bmatrix}$$

ここでA_{ij}はヤコビアンＪの余因子行列である．

成分A_{ij}を表示すると以下の様に表示される．

$$\mathrm{A}_{11} = \begin{vmatrix} \dfrac{\partial \mathrm{y}}{\partial \eta} & \dfrac{\partial \mathrm{z}}{\partial \eta} \\ \dfrac{\partial \mathrm{y}}{\partial \varsigma} & \dfrac{\partial \mathrm{z}}{\partial \varsigma} \end{vmatrix} \qquad \mathrm{A}_{12} = \begin{vmatrix} \dfrac{\partial \mathrm{y}}{\partial \varsigma} & \dfrac{\partial \mathrm{z}}{\partial \varsigma} \\ \dfrac{\partial \mathrm{y}}{\partial \xi} & \dfrac{\partial \mathrm{z}}{\partial \xi} \end{vmatrix} \qquad \mathrm{A}_{11} = \begin{vmatrix} \dfrac{\partial \mathrm{y}}{\partial \xi} & \dfrac{\partial \mathrm{z}}{\partial \xi} \\ \dfrac{\partial \mathrm{y}}{\partial \eta} & \dfrac{\partial \mathrm{z}}{\partial \eta} \end{vmatrix}$$

$$\mathrm{A}_{11} = \begin{vmatrix} \dfrac{\partial \mathrm{y}}{\partial \eta} & \dfrac{\partial \mathrm{y}}{\partial \eta} \\ \dfrac{\partial \mathrm{y}}{\partial \varsigma} & \dfrac{\partial \mathrm{y}}{\partial \varsigma} \end{vmatrix} \qquad \mathrm{A}_{11} = \begin{vmatrix} \dfrac{\partial \mathrm{y}}{\partial \eta} & \dfrac{\partial \mathrm{y}}{\partial \eta} \\ \dfrac{\partial \mathrm{y}}{\partial \varsigma} & \dfrac{\partial \mathrm{y}}{\partial \varsigma} \end{vmatrix} \qquad \mathrm{A}_{11} = \begin{vmatrix} \dfrac{\partial \mathrm{y}}{\partial \eta} & \dfrac{\partial \mathrm{y}}{\partial \eta} \\ \dfrac{\partial \mathrm{y}}{\partial \varsigma} & \dfrac{\partial \mathrm{y}}{\partial \varsigma} \end{vmatrix}$$

$$\mathrm{A}_{11} = \begin{vmatrix} \dfrac{\partial \mathrm{y}}{\partial \eta} & \dfrac{\partial \mathrm{y}}{\partial \eta} \\ \dfrac{\partial \mathrm{y}}{\partial \varsigma} & \dfrac{\partial \mathrm{y}}{\partial \varsigma} \end{vmatrix} \qquad \mathrm{A}_{11} = \begin{vmatrix} \dfrac{\partial \mathrm{y}}{\partial \eta} & \dfrac{\partial \mathrm{y}}{\partial \eta} \\ \dfrac{\partial \mathrm{y}}{\partial \varsigma} & \dfrac{\partial \mathrm{y}}{\partial \varsigma} \end{vmatrix} \qquad \mathrm{A}_{11} = \begin{vmatrix} \dfrac{\partial \mathrm{y}}{\partial \eta} & \dfrac{\partial \mathrm{y}}{\partial \eta} \\ \dfrac{\partial \mathrm{y}}{\partial \varsigma} & \dfrac{\partial \mathrm{y}}{\partial \varsigma} \end{vmatrix}$$

４．１．３　変数変換

物理空間$\boldsymbol{x}=(\mathrm{x}^1,\mathrm{x}^2,\mathrm{x}^3)$と計算空間$\boldsymbol{\xi}=(\xi^1,\xi^2,\xi^3)$の間に$\mathbf{x}=\mathbf{x}(\xi)$の対応関係を考える．物理空間$\boldsymbol{x}$と計算空間$\boldsymbol{\xi}$の間に成立する微分関係式は

$$\frac{\partial}{\partial \mathrm{x}} = \frac{\partial \xi^{\mathrm{i}}}{\partial \mathrm{x}^{\mathrm{j}}}\frac{\partial}{\partial \xi^{\mathrm{i}}} \quad \text{つまり，} \quad \nabla = (\nabla\xi^{\mathrm{i}})\frac{\partial}{\partial \xi^{\mathrm{i}}}$$

$$\frac{\partial}{\partial \xi^{\mathrm{i}}} = \frac{\partial \mathrm{x}^{\mathrm{j}}}{\partial \xi^{\mathrm{i}}}\frac{\partial}{\partial \mathrm{x}^{\mathrm{j}}} \quad \text{つまり，} \quad \frac{\partial}{\partial \xi^{\mathrm{i}}} = \left(\frac{\partial \mathrm{x}^{\mathrm{j}}}{\partial \xi^{\mathrm{i}}}\right)\frac{\partial}{\partial \mathrm{x}^{\mathrm{j}}}$$

が成立する．ここで∇とヤコビの行列をＪとすれば

$$\nabla = \mathbf{g}^{\mathrm{i}}\frac{\partial}{\partial \xi^{\mathrm{i}}}$$

$$\frac{\partial}{\partial \xi^{\mathrm{i}}} = \mathbf{g}_{\mathrm{i}}\frac{\partial}{\partial \mathrm{x}^{\mathrm{j}}}$$

のようになる．

４．２　離散ナブラ演算子

関数$\varphi(\boldsymbol{x})$をある節点 I における近似値$\varphi_{\boldsymbol{i}}$と形状関数$\mathrm{N_i(x)}$を用いて表すと

$$\phi(\mathrm{x}) = \sum_{\mathrm{i}} \phi_{\mathrm{i}}\mathrm{N_i(x)}$$

となる．

ここで離散ナブラ演算子の定義は

$$\nabla_{\mathrm{a}} \equiv \frac{1}{\Omega_{\mathrm{e}}}\int_{\Omega_{\mathrm{e}}} \nabla \mathrm{N_a}\,\mathrm{d}\Omega$$

である．つぎに離散ナブラ演算子の近似について述べる．
まず，形状関数のこう配を計算空間から物理空間に変換する．

$$\nabla \mathrm{N_a} = \mathrm{i}\frac{\partial \mathrm{N_a}}{\partial \mathrm{x}} + \mathrm{j}\frac{\partial \mathrm{N_a}}{\partial \mathrm{y}} + \mathrm{k}\frac{\partial \mathrm{N_a}}{\partial \mathrm{z}}$$

$$= \mathbf{g}^1\frac{\partial \mathrm{N_a}}{\partial \xi} + \mathbf{g}^2\frac{\partial \mathrm{N_a}}{\partial \eta} + \mathbf{g}^3\frac{\partial \mathrm{N_a}}{\partial \varsigma}$$

これを考慮すると離散ナブラ演算子の近似は，

$$\nabla_a \equiv \frac{1}{\Omega_e}\int_{\Omega_e} \nabla N_a d\Omega(\mathbf{x})$$
$$= \frac{1}{\Omega_e}\int_{\Omega_e} \mathbf{g}^i \frac{\partial N_a}{\partial \xi^i} J d\Omega(\xi)$$
$$= \frac{1}{\Omega_e} < \mathbf{g}^i J >_e \int_{\Omega_e} \frac{\partial N_a}{\partial \xi^i} d\Omega(\xi)$$
$$= \frac{1}{\Omega_e} < \mathbf{g}^i >_e J_e \int_{\Omega_e} \frac{1}{8}\xi_a^i \left(1+\xi_a^j \xi^j\right)\left(1+\xi_a^k \xi^k\right) d\Omega(\xi)$$
$$= \frac{1}{\Omega_e} < \mathbf{g}_j \times \mathbf{g}_k >_e \cdot \xi_a^i$$
$$= \frac{1}{\Omega_e}\left\{ < \mathbf{g}_2 \times \mathbf{g}_3 >_e \xi_a^i + < \mathbf{g}_3 \times \mathbf{g}_1 >_e \xi_a^j + < \mathbf{g}_1 \times \mathbf{g}_2 >_e \xi_a^k \right\}$$

の様に表す事が出来る．

ここで $< g_j \times g_k >_e$ は基底ベクトルの張る面積であり余因子ベクトルという．つまり，離散ナブラ演算子はそれぞれの余因子ベクトルを計算空間の方向成分に分解してその分解した成分ごとに和を取ったこう配ベクトルを要素体積で割ったものになっている．実際の計算では計算の効率化のためナブラ演算を行う際にはこう配ベクトルを多用するので以下ではこう配ベクトルの生成について議論する．

４．３　こう配ベクトル生成の入力

こう配ベクトルを生成する際に考えられる入力は座標，物理空間の座標を計算空間の座標で微分した値，余因子ベクトルの 3 種類がある．これら 3 種類を六面体一要素のこう配ベクトルを生成する際に必要な，データ数，メモリアクセス回数，演算回数の視点から比較して，一番負荷が低い入力を決定した．（Table1，Table2，Table3）

Table.1 The number of data

	データ数
座標値	24
微分値	9
余因子ベクトル	9

Table.2 The number of access to memory

	メモリアクセス回数
座標値	216
微分値	108
余因子ベクトル	36

Table.3 The number of calculations

	演算回数		
	加算	乗算	反転
座標値	168	36	12
微分値	60	36	12
余因子ベクトル	24	0	12

Table.1，Table.2 において明らかなように余因子ベクトルがデータ数，データ参照回数ともに負荷が軽い．Table3 に於いて反転とは符号の反転のことである．こう配ベクトルの対称性を利用できるのでこう配ベクトルは一要素当たり 12 成分の計算で符号反転して 24 成分計算することが可能である．Table3 からも余因子ベクトルから生成する方法が，一番計算負荷が軽いことが分かる．本研究においては三次元六面体要素を検討したので，これよりどの項目に於いてもこう配ベクトルは余因子ベクトルから生成する方法が，比較的効率がよいと言える．従って余因子ベクトルを入力としたこう配ベクトル生成演算器構成について考えた．

４．４　こう配ベクトルの保持方法

４．３で述べたようにこう配ベクトルは余因子ベクトルから生成するのが望ましいことが分かった．ここでは，こう配ベクトルをナブラ演算が出で来る度に余因子ベクトルから逐次生成する場合と，最初に一度だけ余因子ベクトルからこう配ベクトルを生成しておきメモリにためておく場合で，どちらが計算時間の短縮が可能であるかを検討した．三次元正方キャビティイ強制対流問題をレイノルズ数 100，時間ステップ 50，最大繰り返し回数 500，収束判定基準 1.0×10^{-3}，時間刻み 5.0×10^{-3} の条件のもとに計算し，以下の計算時間の比較を行った．

① 余因子ベクトルからこう配ベクトルを逐次生成する方法
② こう配ベクトルを最初に一度だけ計算させる方法

Table.4 The comparison of calculation time

	Time[s]
①	599.05
②	723.07

Table.4 に示すように余因子ベクトルからこう配ベクトルを逐次生成させる方法が，一度だけこう配ベクトルを計算してメモリにおいておく方法よりも計算時間が短縮されている．これはメモリアクセスにかかる時間と計算にかかる時間に由来していると考えられる．現在，計算の速度は高速化されているが，それに対してメモリアクセスの速度は高速化されているとは言いがたい．この問題においても，方法①は一要素のこう配ベクトルを計算するために余因子ベクトルをメモリから取ってくるが，余因子ベクトルは一要素当たりこの場合９データ存在するため，９データをメモリから取ってくる必要がある．それに対し方法②では，こう配ベクトルを直接メモリから取るだけであるが，この場合こう配ベクトルは一要素当たり２４データ存在するため，２４データをメモリから取ってこなければならない．このメモリアクセス回数の違いが Table.4 に示す計算時間の違いの原因と考えられる．

したがって，本研究においてこう配ベクトルの専用ロジックを検討する際，余因子ベクトルから逐次生成する方法を採る．

５．データの保持精度

本研究においてこう配ベクトルの専用ロジックを検討する際に，データ精度を倍精度か単精度にするかを決定するために，3 次元正方キャビティ内強制対流問題を解きその Poisson 方程式の繰り返し回数の変化及び計算時間を参考にした．なお，流体の数値計算において必ずしも，どちらのデータ精度が良いと断言する事は不可能である．ここで述べる結果は，本研究において専用ロジックを検討するさいのデータ精度を決定するためだけである事を言及しておく．調べた問題の計算条件は以下の通り．

①レイノルズ数 100　単精度　時間ステップ 50
②レイノルズ数 100　倍精度　時間ステップ 50
③レイノルズ数 1000　単精度　時間ステップ 50
④レイノルズ数 1000　倍精度　時間ステップ 50
⑤レイノルズ数 100　単精度　時間ステップ 200
⑥レイノルズ数 100　倍精度　時間ステップ 200
⑦レイノルズ数 1000　単精度　時間ステップ 200
⑧レイノルズ数 1000　倍精度　時間ステップ 200

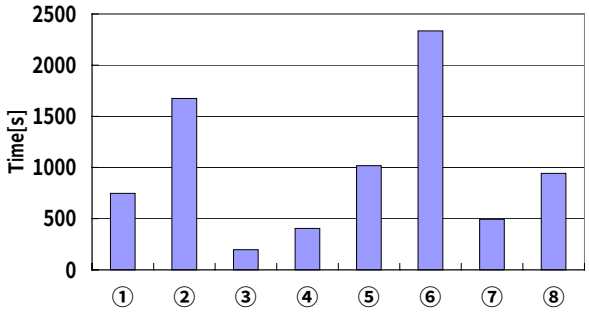

Fig.3 The change of the calculation time

Fig.3 から分かるように計算時間については圧倒的に倍精度のほうが長くかかった．一方，この問題においては倍精度・単精度いずれの場合でも Poisson 方程式の繰り返し回数の変化はほとんどなかった．ここで注意すべきはこの結果はあくまで一つの例に過ぎず，これから一概にすべての問題について単精度で十分な精度であるというわけではない．しかし本研究においては計算の高速化が最大の目標であるため，演算器構成を検討するにあたりデータの精度を単精度にすることに決定した．

６．こう配ベクトル専用ロジックのアーキテクチャの検討

今回は三次元六面体要素に於いてこう配ベクトルを計算することを想定しアーキテクチャの検討を行った．

６．１　前提条件

まずアーキテクチャを検討するに当たり前提条件を決めた．

・浮動小数点加算器は 4 章より 32bit のものを使用し，その動作周波数 300MHz，パイプラインは 3 段設ける．こう配ベクトルは余因子ベクトル（一要素当たり 9 データ）の 2 回の加算によって得られるため浮動小数点加算器を 2 個使用した構成にする．

・また，演算器構成と合わせてデータ供給についてもメモリに SDRAM を想定し，その動作周波数は 100MHz とし，RAS to CAS Latency を 2clk, CAS Latency を 2clk, Burst Length を 8 とした．

・

６．２　アーキテクチャ

本研究で一番懸念される事はメモリからのデータ供給が計算のスピードに対して十分間に合うかどうかにある．この点を解消するためには，データの供給方法の検討が必要である．今回はレジスタファイルを 2 つ用意することによりメモリからデータを読み込む作業と演算を同時に行うことが可能となる構成を考えた．また，こう配ベクトルの一成分は余因子ベクトルの２回の浮動小数点加算によって算出される事から，浮動小数点加算器を２つ設置した．このような構成を取る事により，ある要素のこう配ベクトルを計算している間に，次のこう配ベクトルの生成に必要な余因子ベクトルをもう一方のレジスタファイルに持ってくる事が可能となる．つまり，演算がデータ供給のために妨げられる事が無くなる．この演算器構成を Fig.4 に示す．Fig.4 のような演算器構成であればデータ供給が充分に行われることが可能である．なお，５．１で前述した条件において，この専用ロジックの理論ピーク性能は 0.5Gflops(Floating Operation Per Second)である．

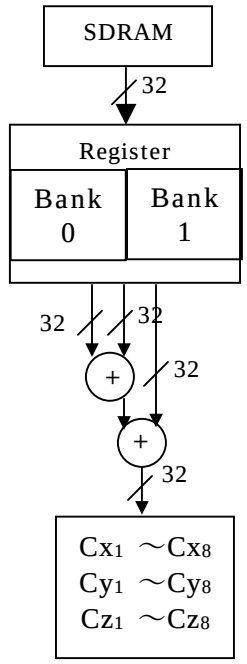

Fig.4 The architecture of operators and memory

７．Poisson 方程式の高速化に向けての検討

前述したとおり GSMAC-FEM は Poisson 方程式の繰り返し計算に多くの計算時間が必要とされる．ここでは，膨大な時間がかかる要因を挙げ，それに対する検討・改良案，及びソフトウェア上での高速化の試みについて述べる．

計算時間のかかる要因として，大きく分けて二点あると考えられる．まず一点目は要素から接点をランダムアクセスする点である．これは SDRAM というメモリ構造上不利になる事がわかる．そして二点目は，速度を修正速度ポテンシャルによって修正する場合に read modified write という動作を行う点にある．この read modified write とはメモリ上から必要なデータを読み出してきてそれを修正して書き戻すという動作であるが，その次の処理にこの修正されたデータが必要になる場合，この動作が終わるまではその次の処理を開始する事が出来ない．その結果多くの計算時間が必要となる．

この２点について以下の７．１でそれぞれの詳細について述べる．さらに，高速化のために Poisson 方程式の計算順序を検討した．この詳細については７．２において述べる．

７．１　高速化の検討Ⅰ

計算時間が多くとられる個所として，要素からその要素回りの接点を参照することがある．これは具体的には，速度を修正速度ポテンシャルによって修正する個所において生じる．前述した read modified write も同じ個所において生じている．この修正速度ポテンシャルにより速度を修正していく過程を Fig.5 に示す．

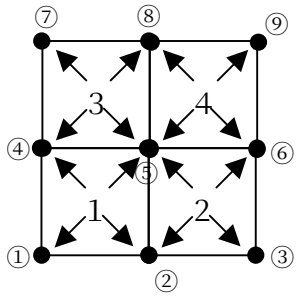

Fig. 5 The connection between nodes and elements (1)

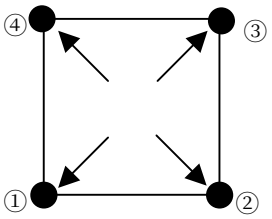

Fig.6 The access from an element to nodes

速度を修正していく際に，まず要素番号の順に修正していく．速度は接点で定義されているので一つの要素について，その要素を構成する接点数分だけ修正することになる．例として Fig.5 に示す 2 次元四角形要素について説明する．要素番号１では接点番号①②④⑤で定義されている速度について修正する．この時，接点番号は必ずしも連続的ではないためメモリアクセスに時間がかかってしまう．さらに要素番号 1 に関する修正が終了した後, 要素番号２に関する修正を行う．要素番号２では接点番号②③⑤⑥において定義されている速度について修正する．この時，接点番号②③では，前の要素番号１の処理が完了しないと修正する事は出来ない. つまり，要素番号１における修正により接点番号②③の速度が修正されてメモリに書き戻された後でなければ，要素番号２における接点番号②③の修正はできない．このために要素番号２における処理は要素番号１における処理が完了するのを待たなくてはならない状況にある．この以上の２点を解決するために別の速度修正方法を検討した.

まず速度を修正する際に必要となってくるのは接点で定義された速度と，要素中心で定義された修正速度ポテンシャルである．つまり，接点定義と要素定義のデータが必要になる限りメモリのランダムアクセスは不可避である．そこで少なくとも read modified write という，前の処理が終了するのを待たなくては次の処理が始められない状態を回避する方法について検討した．前述の方法では要素中心に処理が行われている．これにより任意の接点における速度の修正が一度に行われず，複数要素にわたって修正される事になり結果的に一接点について何度か速度を修正してメモリに書き戻さなくてはならない．そこで，接点中心に処理を進めて行く方法について検討した．例として Fig.7 に示す二次元四角形要素について説明する. Fig.7 に示すように接点を中心にして計算して行く．接点番号①で定義された速度を修正するために要素番号１で定義された修正速度ポテンシャルをメモリから読み, 計算を行う．次に接点番号②で定義された速度を修正するために要素番号１，２で定義された修正速度ポテンシャルをメモリから読み，計算を行う．この様にして計算を実行させると境界ではない接点番号⑤の速度を修正するため，要素番号１，２，３，４の修正速度ポテンシャルを読み計算する．つまり, 任意の接点で定義されている速度を修正するためには, その接点の衛星要素において定義されている修正速度ポテンシャルをメモリから読み，計算を行う．この際現在採られている要素中心で計算する方法と比較して，この方法は修正した速度を書き戻すのを待たずして次の処理を開始できる点が大きな利点として上げられる.

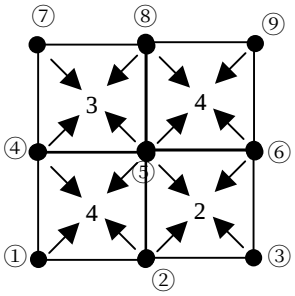

Fig.7 The connection between nodes and elements (2)

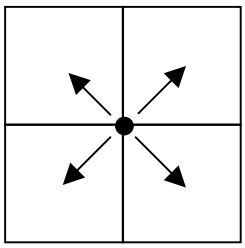

Fig.8 The access from a node to elements

７．２　高速化の検討Ⅱ

ここではさらに Poisson 方程式の計算時間の短縮を図るため検討した方法について述べる．Poisson 方程式の繰り返し計算でナブラ演算が多用されるのは前述の通りである．このナブラ演算は，余因子ベクトルから逐次生成されるこう配ベクトルによって計算されている．ここではこう配ベクトルという形態をとらずに，直接余因子ベクトルを Poisson 方程式のナブラ演算に使用する方法について検討した．例として２次元四角形要素の場合について説明する.

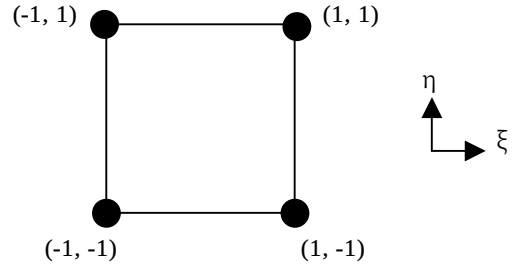

Fig.9 The calculation space

二次元四角形要素についてこう配ベクトル **C** が算出される過程について述べる．まず各物理空間座標値から決定される余因子ベクトル **A** は以下の式で決定される.

$$\mathbf{A}^{\xi}=\begin{pmatrix}A11\\A21\\0\end{pmatrix}=\begin{pmatrix}\dfrac{1}{\Delta\eta}\left(\overline{\Delta y}\right)_{\eta}\\-\dfrac{1}{\Delta\eta}\left(\overline{\Delta x}\right)_{\eta}\\0\end{pmatrix}$$

$$\mathbf{A}^{\eta}=\begin{pmatrix}A12\\A22\\0\end{pmatrix}=\begin{pmatrix}-\dfrac{1}{\Delta\xi}\left(\overline{\Delta y}\right)_{\xi}\\\dfrac{1}{\Delta\xi}\left(\overline{\Delta x}\right)_{\xi}\\0\end{pmatrix}$$

この様にして計算された余因子ベクトルを使用して，こう配ベクトル **C** は以下の様に表示される.

$$\mathbf{C}x_i = A11\xi_i + A12\eta_i$$
$$\mathbf{C}y_i = A21\xi_i + A22\eta_i$$

ここで ξ_i,η_i は計算空間における座標値を示している．また i=1,2,3,4 でありこれは計算空間の各接点番号に対応している．こう配ベクトルを用いて速度の発散は以下の様に表示される.

$$\mathrm{Div}\,\mathbf{v}=\sum_{i=1}^{4}\left(\mathbf{C}x_i u_i+\mathbf{C}y_i v_i\right)$$

ここで，この発散の式をこう配ベクトル **C** と速度を用いて表示するのではなく直接余因子ベクトルを用いて表示すると

$$\mathrm{Div}\,\mathbf{v}=\left(-u_1+u_2+u_3-u_4\right)A11+\left(-u_1-u_2+u_3+u_4\right)A12$$
$$+\left(-v_1+v_2+v_3-v_4\right)A21+\left(-v_1-v_2+v_3+v_4\right)A22$$

となる．これらの 2 式は全く等価である．余因子ベクトルから直接発散を計算する方法の利点については二つ考えられる．第一に，余因子ベクトルをこう配ベクトルという形式をとらずに直接，発散の計算に使用することにより演算数が減少し，こう配ベクトルの算出時間の短縮が可能となる．第二に，速度をまとめてメモリからデータを読み出してそれを使用して各方向二回ずつ演算を行うためメモリアクセス負荷が軽くなる事が考えられる．

さらに第一の利点について，検討を加える．なおこの検討は三次元六面体要素について行う．

まず以下のそれぞれの方法について，余因子ベクトルから 1 要素当たりの速度の発散を求めるまでの総演算回数について検討した．

方法

① 余因子ベクトルからこう配ベクトルを計算し，こう配ベクトルから速度の発散を求める方法

② 余因子ベクトルから直接速度の発散を計算する方法

具体的に行われる計算

①

（ｉ）こう配ベクトルの計算

まず余因子ベクトル $\mathbf{A}$ からこう配ベクトル $\mathbf{C}x_i, \mathbf{C}y_i, \mathbf{C}z_i$ を計算する．

$$\mathbf{C}x_i = \mathbf{A}11\xi_i + \mathbf{A}12\eta_i + \mathbf{A}13\varsigma_i$$
$$\mathbf{C}y_i = \mathbf{A}21\xi_i + \mathbf{A}22\eta_i + \mathbf{A}23\varsigma_i$$
$$\mathbf{C}z_i = \mathbf{A}31\xi_i + \mathbf{A}32\eta_i + \mathbf{A}33\varsigma_i$$

ここで i=1,2,3,4,5,6,7,8 で $\xi_i, \eta_i, \varsigma_i$ は計算空間における座標値を示している．

（ii）速度の発散の計算

接点 i における速度を u_i, v_i, w_i とすると，ある要素における速度の発散は

$$\mathrm{Div}\,\mathbf{v} = \sum_{i=1}^{8}\left(\mathbf{C}x_i u_i + \mathbf{C}y_i v_i + \mathbf{C}z_i w_i\right)$$

のように表示できる．

②

（ｉ）速度の計算

余因子ベクトルから直接計算をする場合には，計算空間での座標値を考慮して，一要素回りに存在する８接点で定義された各方向速度について加算を行う．

$$u^{\xi} = \sum_{i=1}^{8} u_i\xi_i \qquad v^{\xi} = \sum_{i=1}^{8} v_i\xi_i \qquad w^{\xi} = \sum_{i=1}^{8} w_i\xi_i$$
$$u^{\eta} = \sum_{i=1}^{8} u_i\eta_i \qquad v^{\eta} = \sum_{i=1}^{8} v_i\eta_i \qquad w^{\eta} = \sum_{i=1}^{8} w_i\eta_i$$
$$u^{\varsigma} = \sum_{i=1}^{8} u_i\varsigma_i \qquad v^{\varsigma} = \sum_{i=1}^{8} v_i\varsigma_i \qquad w^{\varsigma} = \sum_{i=1}^{8} w_i\varsigma_i$$

（ii）速度の発散の計算

余因子ベクトル $\mathbf{A}$ と（ｉ）で求めた値を利用して速度の発散は

$$\mathrm{Div}\,\mathbf{v} = u^{\xi}A11 + u^{\eta}A12 + u^{\varsigma}A13$$
$$+ v^{\xi}A21 + v^{\eta}A22 + v^{\varsigma}A23$$
$$+ w^{\xi}A31 + w^{\eta}A32 + w^{\varsigma}A33$$

の様に表示できる．

以上の具体的な計算式からそれぞれの方法で１要素の速度の発散を求める際に，加算，乗算の回数を Table.8 に示した．

Table.5 The number of floating operations needed for gradient vector of one element

Type \ Calculation	Floating point add	Floating point multiple
①	71	24
②	71	9

以上の事項を踏まえて速度の発散の専用 Logic を検討した場合の応答時間を，パラメータを用いて検証する．余因子ベクトルからこう配ベクトルを計算して速度の発散を求める方法①と，余因子ベクトルから直接速度の発散を計算する方法②について浮動小数点演算器を用いて計算した場合の計算時間を式で表示する．浮動小数点加算器，浮動小数点乗算器を一つずつ仮定してそれぞれの応答時間を m，n とおく．３次元六面体要素１要素あたりの速度の発散を求める際にかかる計算時間を方法①，方法②についてそれぞれ $\mathbf{t}_1, \mathbf{t}_2$ とおくと，以下のように表示される．但しこの場合は，データは理想的に与えられるものとし，パイプラインは設けていない．

$$t_1 = 71m + 24n$$
$$t_2 = 71m + 9n$$

このように表示されるのは Table.5 からの演算数がそのまま反映されている形になっていることが分かる．したがって余因子ベクトルを直接速度の発散の計算に使用する方法②はソフトウェア上でも十分な高速化が見られた上, 今後の Poisson 方程式全体を視野に入れた専用ロジックの検討に有効であるといえる．

参考文献

（１）棚橋隆彦 “流れの有限要素法解析Ｉ”，朝倉書店

（２）ジョン・L・ヘネシー, デイビット・A・パターソン “コンピュータの構成と設計[下]”，日経 BP