Vol.2018-CG-169 No.13
2018/3/3

拡張されたラプラシアン固有関数を用いた流体シミュレーションの計算効率化

石室屋　正人[1,a)]　金井　崇[1,b)]

概要：流体シミュレーションにおいて，計算精度と実行時間はトレードオフの関係にある。計算精度の損失を抑えたまま高速なシミュレーションを実現するために，多数のグリッド速度の集合である流体速度場を，ラプラシアン固有関数や PCA 等の少数の基底で表現することで自由度を削減する手法が提案されてきた。しかし，従来の手法では基底が領域全体をカバーするため，少数の基底では速度の局所的変化に対応できず，領域全体の変化として計算されてしまうという問題がある。本手法では，この問題を解決するために，構造格子内で複雑な領域をカバーするラプラシアン固有関数により領域を分割することで，流体速度の局所変化に対応し，かつ，計算精度を維持するシミュレーション手法を提案する。

1. 背景と関連研究

格子法における流体シミュレーションでは，領域の分割数が大きいほどシミュレーションの精度は向上するが，その反面で，計算時間と使用メモリが大きくなるというトレードオフが存在する。この問題を解決するため，かねてより，精度を維持しつつ，計算を高速化する手法が考えられてきた。自由度を削減してシミュレーションを高速化する手法の一つとして，基本速度場を用いた方法がある。この手法では，流体速度場 $\mathbf{u}$ を，基本速度場 Φ を用いて別形式で表現する。k 番目の基本速度場を $\Phi^k (1 \leq k \leq N)$，各基本速度場に対する重みを $\boldsymbol{w} = (w_1, \cdots w_k \cdots w_N)$ とすると，流体の速度場は，

$$\mathbf{u} = \sum_{k=1}^{N} w_k \Phi^k,$$

と，基底となる基本速度場の線形結合和で表される。CG においては，Treuille らによって PCA を使った基本速度場の生成と，障害物周辺の速度場を組み合わせることで自由度削減を行う流体シミュレーションが提案された [1]。それ以降，シミュレーション領域をタイルによって分割し，それぞれのタイルに PCA による速度場を当てはめる手法 [2]，固体と流体のインタラクションの改良 [3] などが考えられてきた。

一方，De Witt らによってラプラシアン固有関数を用いて自由度削減を行う流体シミュレーションが提案された [4]。この関数は，離散化されたラプラシアン行列の固有ベクトルを計算することで生成できる。ラプラシアン固有関数の利点として，

- ラプラシアン行列は疎行列であるため，固有ベクトルは短時間で計算できる。
- 領域のみに依存する解析解のため，前計算のデータに依存せず他のシーンへの汎用性が高い。
- 関数はインデックス k によって周波数ごとに規則正しく並べられており，流体の見た目の細かさを容易に設定できる。

などが挙げられる。固有関数を用いた手法の多くは正方形領域を対象にしているが，複雑な形状を埋め込んだ構造格子に対するラプラシアン固有関数を計算するために，Liu らは構造格子に DEC によるラプラス演算子を用いることで，領域特有の基本速度場を作成する手法を提案した [5]。しかしこれらの手法は，ラプラシアン行列の作成と固有値の計算に複雑な処理が必要となり，また閉境界に対応する基底のみしか作られていない。

また，基本速度場の問題として，基底はそれぞれが領域全体をカバーしているため，本来は局所的であるはずの速度変化に対しても基底の重みが変わり，結果として領域全体の速度場が変化してしまうというものがある。さらに，1 つの基底はシーンの中のグリッド分のデータ容量を占めるため，基本速度場のデータ集合は（グリッド数）×（速度場の数）の分のメモリを必要とする。よって，大規模なシーンを計算するときや，多くの速度場を使用するときには大量のメモリを消費する。Jones らによる基本速度場の

1　東京大学大学院総合文化研究科
〒 153-8902 東京都目黒区駒場 3-8-1
a)　muroya@graco.c.u-tokyo.ac.jp
b)　kanai@graco.c.u-tokyo.ac.jp

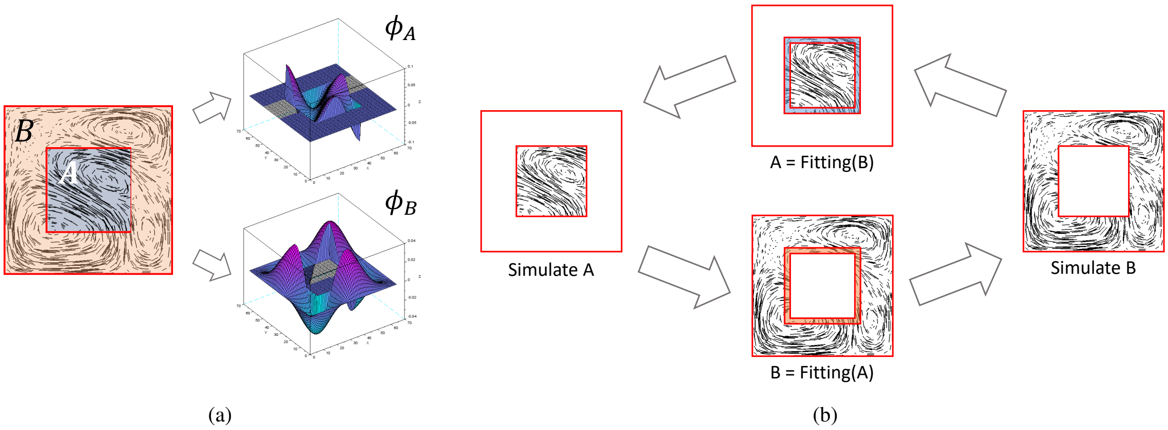


図 1 提案手法の概要。(a) 分割された領域と対応する基底 ϕ。(b) 領域を分割したシミュレーション。

データ圧縮の手法 [6] は，高い圧縮率を発揮するが，データの圧縮・復元に時間がかかるという問題がある。

2. 提案手法

2.1 提案手法の概要

シーン領域を複数の小領域に分割し，それぞれで独立してシミュレーションを行うことによって，速度の変形を小領域内に収め，局所性を維持することを目的とする。

まず，2.2 節において，流れ関数の境界条件を考え，その上で，ユーザが指定した分割領域に従って壁面・流入/流出それぞれの境界条件を満たすラプラシアン固有関数の作成法について説明する（図 1(a)）。

その後，2.3 節において，領域を複数の小領域に分割し，それぞれの小領域でシミュレーションを行う手法を提案する。小領域ごとに所望する流体速度の細かさによって基本速度場の数を変えることで，メモリ・計算速度の効率化を行うことができる。速度場の連続性と流体の非圧縮性を保つために，A と B の分割面での速度を一致させる手法について説明する。シミュレーション実行中では，小領域のシミュレーション → 分割面での速度が一致するように他方の小領域の速度を更新 → ... のサイクルを繰り返す（図 1(b)）。

2.2 基本速度場の生成

ラプラシアン固有関数

ラプラシアン固有関数とは，ある境界条件の下で，

$$\nabla^2\phi = \lambda\phi, \tag{1}$$

を満たすような関数 ϕ のことである。2 次元でのシミュレーションにおいては，ϕ はスカラー場となる。流れ関数を $\psi = -\frac{\phi}{\lambda}$ とし，その回転 $\Phi = \nabla\times\psi$ をとることで，

$$\begin{cases}\nabla^2\Phi = \lambda\Phi, \\ \nabla\cdot\Phi = 0,\end{cases} \tag{2}$$

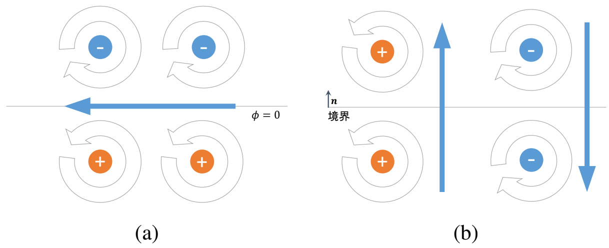


図 2 ψ の境界条件と速度。(a) ディリクレ条件。(b) ノイマン条件。

を満たす基本速度場 Φ を得ることができる。$n_x \times n_y$ に離散化された格子の場合，ϕ は，

$$\phi = \begin{pmatrix}\phi_{0,0} \\ \vdots \\ \phi_{n_x,n_y}\end{pmatrix}, \tag{3}$$

と，格子点上の値の集合として表され，式 (1) は，

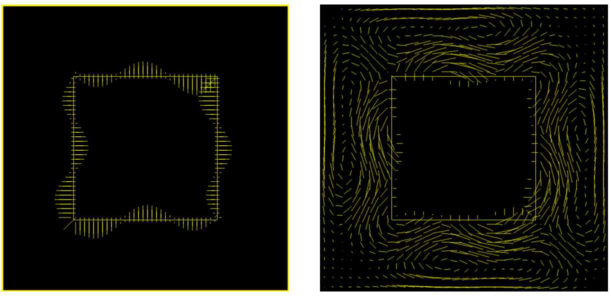
$$L\phi = \lambda\phi, \tag{4}$$

と変換される。ここで，行列 L はラプラシアン行列 $[\nabla^2]$ である。このラプラシアン行列 L の k 番目の固有ベクトルが，各固有関数 ϕ^k に対応する。

境界条件

壁面境界と流出/流入境界それぞれについて，その境界条件をみたす固有関数 ϕ の作成方法を説明する。ϕ と $\psi = -\frac{\phi}{\lambda}$ の境界条件は一致するため，以降は ψ の境界条件について述べる。流れ関数は流体の渦の方向とその大きさを表すもので，2 次元の場合，$\psi > 0$ ならば，その場所を中心に反時計回りに渦が流れていることを示す。よって，流れ関数がディリクレ条件 ($\psi = 0$ at $\partial\Omega$) を満たすとき，速度は壁面境界の条件を満たす。また流れ関数がノイマン条件 ($\partial\phi/\partial\boldsymbol{n} = 0$ at $\partial\Omega$) を満たすとき，速度は境界に対して垂直となる（図 2）。

本手法では，流体速度のプロジェクションに使われていた手法をラプラシアン行列に応用する。流体セルを境界が



(a) (b)

図 3 (a) 境界速度場 $\tilde{\mathbf{u}}$。(b) 取り除かれる基本速度場の一例。

横切るとき，ラプラシアン行列の係数を

- 壁面境界ではグリッドから境界への距離 [7]
- 流入/流出境界ではグリッド周辺での流体面積 [8]

に応じて決定する（付録 A.1 参照）。

2.3 領域を分割したシミュレーション

提案手法では，複数の小領域に分割してそれぞれ固有関数を作成し，独立にシミュレーションを行うことで，局所的な速度場変化が影響を与える範囲を限定する。基本速度場を用いたシミュレーション方法について，詳しくは De Witt らの論文 [4] を参照されたい。

例として，シミュレーション領域を図 1(a) のように A, B の 2 つに分けるとする。なお，A, B の形状は矩形に限らず任意の形状でよい。シーン全体で速度場の連続性と非圧縮性を保つ必要があるため，一方の小領域でシミュレーションを行った後，分割面での速度場が一致するように他方の速度場を更新する操作（フィッティング）を行う。この操作を A, B それぞれに対して行うため，シミュレーション全体としては A のシミュレーション → 分割面で速度が一致するように B の速度を更新 → B のシミュレーション → 分割面で速度が一致するように A の速度を更新 → ... のサイクルを繰り返す（図 1(b)）。これによって速度場の連続性を保ち，あたかもシーン全体を一気にシミュレーションしたような見た目が得られる。

基本速度場の作成・選択

領域を A, B に分割したのち，それぞれの小領域で基本速度場を作成する。本来は領域は分割されておらず，A, B 間は流体が自由に行き来するため，分割面において流体速度は流入/流出境界となる。本研究では前節の手法を用い，分割面と流体速度が直交するような基本速度場を作成する。

作成された速度場において，分割面における速度ベクトルの集合を境界速度場 $\tilde{\mathbf{u}}$ と定義する（図 3(a)）。領域の分割方法によっては，図 3(b) のような，領域 B 内で大きく渦を巻き，小領域全体の運動エネルギーに対し，境界速度場の運動エネルギーが小さいような基底が現れることがある。このような基底は，後述のフィッティングの操作において小さな差分に境界速度場が入り込み，基底の重みが不当に大きくなる場合があるため，シミュレーション前に事前に取り除く必要がある。本手法では，領域中 B の全ての速度ベクトル $\boldsymbol{u}_{i,j} \in \mathbf{u}_B$ と境界速度 $\tilde{\boldsymbol{u}}_{i,j} \in \tilde{\mathbf{u}}_B$ を比較し，

$$\frac{\max|\tilde{\boldsymbol{u}}|}{\max|\boldsymbol{u}|} < \texttt{threshold}, \tag{5}$$

ならばその基底を取り除く操作を行う。今回の実験では，`threshold` を 0.9 に設定する。

境界速度場のフィッティング

分割面での速度場を一致させるための操作について，図 1(b) での `B=fitting(A)` のステップを例に説明する。n ステップ目が終了した時点での A, B それぞれの境界速度場を $\tilde{\mathbf{u}}^n_A, \tilde{\mathbf{u}}^n_B$ とする。ここで，フィッティングを終えた後の B の速度場を $\tilde{\mathbf{u}}_B^{n+\frac{1}{2}}$ とする。毎ステップ，境界面の速度が一致するような重み $\boldsymbol{w}_B$ の値を計算し，$\boldsymbol{w}_B^{n+\frac{1}{2}}$ としたうえで，従来通りのシミュレーションを行う。一致拘束のステップについて，微小変化 Δ を用いて

$$\tilde{\mathbf{u}}_B^{n+\frac{1}{2}} = \tilde{\mathbf{u}}^n_B + \Delta\tilde{\mathbf{u}}_B, \tag{6}$$

とする。ここで，$\tilde{\mathbf{u}}_B^{n+\frac{1}{2}} = \tilde{\mathbf{u}}^n_A$ が成り立つため，

$$\Delta\tilde{\mathbf{u}}_B = \tilde{\mathbf{u}}^n_A - \tilde{\mathbf{u}}_B, \tag{7}$$

となる。この $\Delta\tilde{\mathbf{u}}_B$ をもとに，$\tilde{\mathbf{u}}$ から重み $\boldsymbol{w}$ を計算する関数 `optimize` を用いて，$\Delta\boldsymbol{w}_B = \texttt{optimize}(\Delta\tilde{\mathbf{u}}_B)$ を計算する。その後，$\boldsymbol{w}_B^{n+\frac{1}{2}} = \boldsymbol{w}^n_B + \Delta\boldsymbol{w}$ を用いてシミュレーションを行う。関数 `optimize` は次の式によるリッジ回帰を用いる。

$$\Delta\boldsymbol{w} = \mathop{\arg\min}_{\boldsymbol{w}} \left(\sum_{i,j} \left\| \Delta\tilde{u}_{i,j} - \sum_{k}^{N} w_k \tilde{\Phi}^k_{i,j} \right\|^2 + \alpha \sum_{k}^{N} \left(\frac{\lambda_k}{\lambda_1} w_k \right)^2 \right) \tag{8}$$

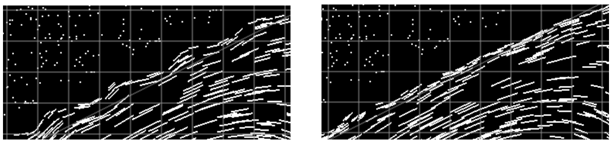
3. 結果と議論

提案手法を用いて基本速度場を生成した結果と，領域を分割してシミュレーションした結果を示す。領域の分割は図 1(a) のようにし，全体の格子分割数は 64×64，中央の小領域 A の分割数は 32×32 としている。なお，A と B の分割幅 Δx は同じである。

3.1 基本速度場の生成

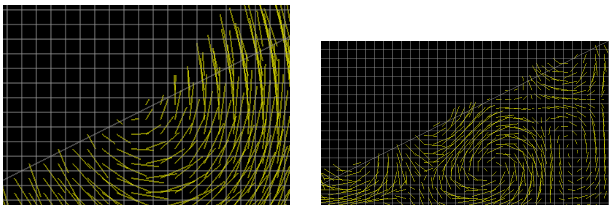
斜めの境界に対して壁面境界・流入/流出境界を設定し，基本速度場を作成した結果を図 4，5 に示す。壁面境界について，ラプラシアン行列の係数を全グリッドで固定の値にすると，境界付近で凹凸のアーティファクトが見られる（図 4(a)）が，係数の値を境界への距離に応じて変えることで境界に沿った滑らかな速度場が生成される (b)。流れ関数のノイマン条件下で，グリッド周辺の流体面積に応じてラプラシアン行列を作成することで，境界に対して速度ベクトルが直交する速度場を作ることができる（図 5）。分

Vol.2018-CG-169 No.13
2018/3/3



(a) (b)

図 4　壁面境界での基本速度場。(a) 係数固定値。(b) 提案手法。



(a) (b)

図 5　流入/流出境界での基本速度場。(a)$k = 1$。(b)$k = 13$。

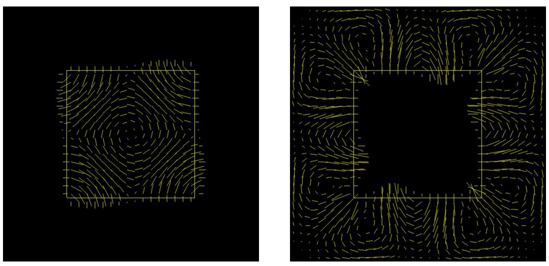
割された小領域について基本速度場を生成した結果を図 6 に示す。どちらの基底も，A と B の境界で速度が分割面に対して直交するようになっている。

3.2　領域を分割したシミュレーション

前節によって作成された基本速度場をもとに，領域を分割してシミュレーションした結果を示す。図 7(a) は領域全体をカバーする基底（基底数 $N = 64$）を用いてシミュレーションを行った際の速度場であり，図 7(b) は領域を分割して基底を作成し（A, B それぞれの基底数 $N_A = 64, N_B = 32$），シミュレーションを行った際の速度場である。図 7(c) は A を従来の格子法によるシミュレーション，B を基本速度場によるシミュレーションを行った結果である。シミュレーション中の外力はマウスドラッグによって A 内のみに指定しており，A 内では細かい渦，それ以外では滑らかな流れが発生すると考えられるため，$N_A > N_B$ としてある。(c) の小領域 A におけるシミュレーションでは，中間ステップでの速度 $\boldsymbol{u}^*$ に対して

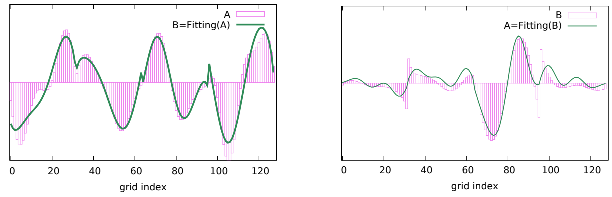
$$\nabla^2 \psi = -\nabla \times \boldsymbol{u}^* \tag{9}$$

によって流れ関数 ψ を計算し，回転 $\boldsymbol{u} = \nabla \times \psi$ をとることによってプロジェクションを行う。このとき，ψ を分割面に対してノイマン条件を適用することで，分割面に直交す



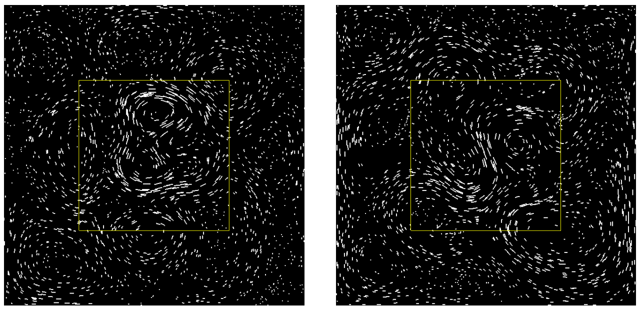
(a) (b)

図 6　分割された領域に対して生成された基本速度場。(a) 小領域 A。(b) 小領域 B。



(a) (b)

図 8　フィッティング結果。(a) 棒グラフはシミュレーションで求めた境界速度 (A) で，折れ線グラフはフィッティングで求めた係数を基に計算した境界速度の値（Fitting(A)）。(b) 同様に棒グラフ（B）と折れ線グラフ（Fitting(B)）。

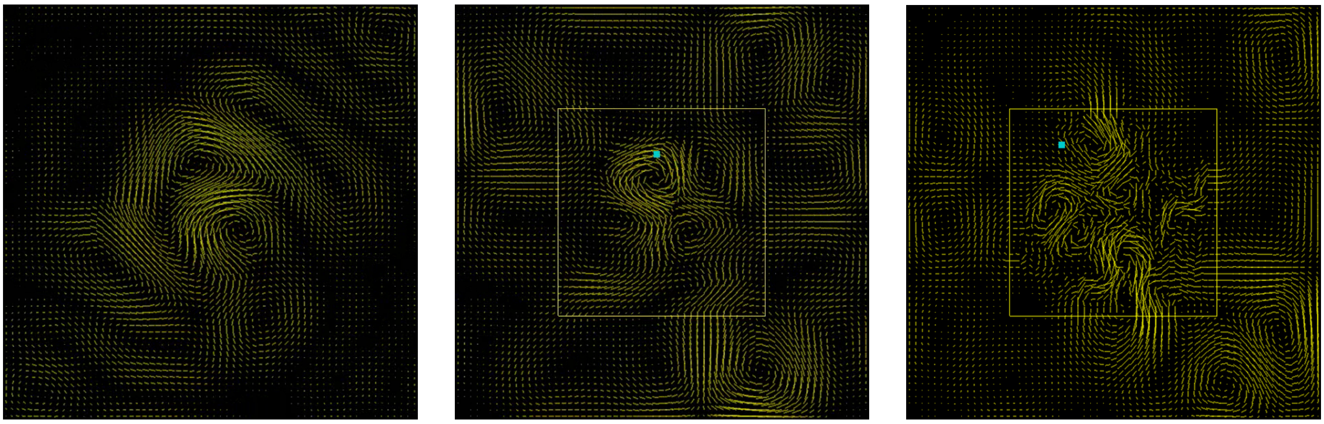


(a) (b)

図 9　提案手法 ($N_A = 64, N_B = 32$) に対してパーティクルを移流した結果。

る速度場を得ることができる。また図 8 はフィッティングを行ったときの，各境界グリッドにおける境界速度場の大きさを示したものである。フィッティング結果を見ると，境界速度が概ね一致しており，また提案手法のシミュレーション結果を見ても，分割面で境界速度が一致し，速度の連続性を保っているように見える。これらの速度場にパーティクルを移流した結果が図 9 である。(a) のような場合にはパーティクルがスムーズに流れているように見えるが，分割面での速度の差が大きい時にはフィッティングが機能せず，流体の非圧縮性が保たれないことによってパーティクルの分布に疎密ができてしまっている (b)。

従来手法と提案手法の結果を比較してみると，流体速度の見た目について，同数の基底数で従来手法では大きな渦しか表現できていないのに対して，提案手法では細かな渦も表現できている（図 7(a)(b)）。B での基底数を減らしているため，シミュレーションに用いるメモリの量も減らすことができている（表 1）。しかし小領域内での変化が境界速度場の変化につながり，結果として他の小領域速度場も変化させるため，流体の局所変形には課題が残る。(c) のように一部の注目領域に対してプロジェクションを用いたシミュレーションを，他の領域を基本速度場によるシミュレーションを行うことで局所変形を維持しつつ効率的なシミュレーションを実現できる。しかし基本速度場のみのシミュレーションに比べて計算時間がかかるというデメリットがある。



(a) (b) (c)

図 7 従来手法と提案手法との比較結果。(a) 従来手法 ($N = 64$)[4]。 (b) 提案手法 ($N_A = 64, N_B = 32$)。 (c) 提案手法 (A：プロジェクション，$N_B = 32$)。

手法	基底数		基底サイズ	行列サイズ	合計
従来手法	N	64	1081600	—	1081600
提案手法	N_A	64	278784	20480	717056
	N_B	32	417792		

表 1 メモリ使用量の比較（単位:バイト）。行列サイズは，式 (8) の計算に用いる行列のサイズを指す。

4. まとめと今後の展望

本論文では，ラプラシアン固有関数を用いた自由度削減のシミュレーション手法に対して，複雑な領域をカバーする基底の作成，また流体速度の局所変化に対応し計算精度を維持するシミュレーション手法を提案した。基底の作成について，速度が壁面境界・流入/流出境界を満たすような流れ関数の境界条件を考え，プロジェクションで用いられた精度向上のための手法をラプラシアン行列の作成に応用した。その結果として提案手法で作成された基本速度場は，少ないグリッド分割数でありながら境界条件を満たすようものになり，シミュレーション精度の向上につながった。流体の局所的な変化を反映させたシミュレーションを行うために，領域を複数の小領域に分割してそれぞれでシミュレーションし，シミュレーション後に分割面での速度を各領域で一致させるような手法を提案した。結果として，使用メモリを抑えながら高精細な結果を出すことができたが，小領域内での変化が境界速度場の変化につながり，結果として他の小領域速度場も変化させるため，速度の局所変形という点では課題が残った。

今後の展望としては，現在の手法は分割面に対して速度が直交するという強い制約があるため，流体の挙動に適した分割方法や，分割面に平行な速度場を取り入れることで斜めの速度ベクトルを実現する方法などが考えられる。速度場の合成方法 [9] などが研究されているため，それらとラプラシアン固有関数を組み合わせる手法も研究の余地があると思われる。

参考文献

[1] Treuille, A., Lewis, A. and Popović, Z.: Model Reduction for Real-time Fluids, *ACM SIGGRAPH 2006 Papers*, SIGGRAPH '06, New York, NY, USA, ACM, pp. 826–834 (online), DOI: 10.1145/1179352.1141962 (2006).

[2] Wicke, M., Stanton, M. and Treuille, A.: Modular Bases for Fluid Dynamics, *ACM Trans. Graph.*, Vol. 28, No. 3, pp. 39:1–39:8 (online), DOI: 10.1145/1531326.1531345 (2009).

[3] Gerszewski, D., Kavan, L., Sloan, P. and Bargteil, A. W.: Basis enrichment and solid-fluid coupling for model-reduced fluid simulation, *Computer Animation and Virtual Worlds*, Vol. 26, No. 2, pp. 109–117 (online), DOI: 10.1002/cav.1612 (2015).

[4] De Witt, T., Lessig, C. and Fiume, E.: Fluid Simulation Using Laplacian Eigenfunctions, *ACM Trans. Graph.*, Vol. 31, No. 1, pp. 10:1–10:11 (online), DOI: 10.1145/2077341.2077351 (2012).

[5] Liu, B., Mason, G., Hodgson, J., Tong, Y. and Desbrun, M.: Model-reduced Variational Fluid Simulation, *ACM Trans. Graph.*, Vol. 34, No. 6, pp. 244:1–244:12 (online), DOI: 10.1145/2816795.2818130 (2015).

[6] Jones, A. D., Sen, P. and Kim, T.: Compressing Fluid Subspaces, *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '16, Aire-la-Ville, Switzerland, Switzerland, Eurographics Association, pp. 77–84 (online), available from ⟨http://dl.acm.org/citation.cfm?id=2982818.2982830⟩ (2016).

[7] Gibou, F., Fedkiw, R. P., Cheng, L.-T. and Kang, M.: A Second-Order-Accurate Symmetric Discretization of the Poisson Equation on Irregular Domains, *Journal of Computational Physics*, Vol. 176, No. 1, pp. 205 – 227 (online), DOI: https://doi.org/10.1006/jcph.2001.6977 (2002).

[8] Ng, Y. T., Min, C. and Gibou, F.: An efficient fluid?solid coupling algorithm for single-phase flows, *Journal of Computational Physics*, Vol. 228, No. 23, pp. 8807 – 8829 (online), DOI: https://doi.org/10.1016/j.jcp.2009.08.032 (2009).

[9] Sato, S., Dobashi, Y. and Nishita, T.: A Combining Method of Fluid Animations by Interpolating Flow Fields, *SIGGRAPH ASIA 2016 Technical Briefs*, SA '16, New York, NY, USA, ACM, pp. 4:1–4:4 (online), DOI: 10.1145/3005358.3005382 (2016).

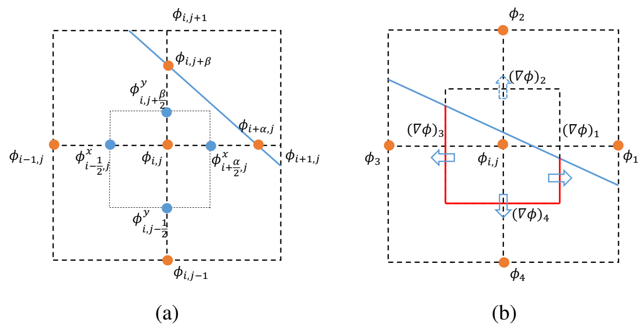


図 A·1 (a) ディリクレ条件でのラプラシアン係数。(b) ノイマン条件でのラプラシアン係数

付　　録

A.1 ラプラシアン行列の求め方

関数 ϕ のラプラシアン，

$$\nabla^2\phi = \nabla\cdot\nabla\phi = \frac{\partial}{\partial x}\left(\frac{\partial\phi}{\partial x}\right) + \frac{\partial}{\partial y}\left(\frac{\partial\phi}{\partial y}\right) = \frac{\partial\phi^x}{\partial x} + \frac{\partial\phi^y}{\partial y},$$

をグリッドで離散化すると，

$$\begin{aligned}
\nabla^2\phi_{i,j} &= \left.\frac{\partial\phi^x}{\partial x}\right|_{i,j} + \left.\frac{\partial\phi^y}{\partial y}\right|_{i,j} \\
&= \frac{1}{\Delta x}\left(\phi^x_{i+\frac{1}{2},j} - \phi^x_{i-\frac{1}{2},j}\right) + \frac{1}{\Delta x}\left(\phi^y_{i,j+\frac{1}{2}} - \phi^y_{i,j-\frac{1}{2}}\right) \\
&= \frac{1}{\Delta x}\left\{\frac{1}{\Delta x}\left(\phi_{i+1,j} - \phi_{i,j}\right) - \frac{1}{\Delta x}\left(\phi_{i,j} - \phi_{i-1,j}\right)\right\} \\
&+ \frac{1}{\Delta x}\left\{\frac{1}{\Delta x}\left(\phi_{i,j+1} - \phi_{i,j}\right) - \frac{1}{\Delta x}\left(\phi_{i,j} - \phi_{i,j-1}\right)\right\} \\
&= \left(\frac{1}{\Delta x}\right)^2\left(\phi_{i+1,j} + \phi_{i-1,j} + \phi_{i,j+1} + \phi_{i,j-1} - 4\phi_{i,j}\right),
\end{aligned} \tag{A.1}$$

となる。このラプラシアン係数を各グリッドごとに並べたものが，ラプラシアン行列 L となる。行列の各固有ベクトルが直交するためには，行列は対称性を持つ必要がある。本手法ではラプラシアン行列を作成した後，$L = L^T$ となるように各係数を対角成分に転写する操作を行っている。

壁面境界

壁面境界では，流れ関数はディリクレ条件 $\phi = 0$ at $\partial\Omega$ となる。境界がグリッドを横切る場合，境界上で $\phi = 0$ となるようにラプラシアン行列の係数を調整する必要がある。この図で，境界より下を流体セル，境界より上を固体セルとする。グリッド (i, j) と隣接グリッド $(i+1, j), (i, j+1)$ との間に境界が通り，グリッド (i, j) と境界との x 方向，y 方向の距離をそれぞれ $\alpha\Delta x, \beta\Delta x$，また境界と格子線の交点をそれぞれ $(i+\alpha), (i, j+\beta)$ とする。図 4 に示すように，計算点が $(i+1, j)$ から $(i+\alpha, j)$ に，$(i, j+1)$ から $(i, j+\beta)$ に移動するため，勾配 $\frac{\partial\phi^x}{\partial x}, \frac{\partial\phi^y}{\partial y}$ の値が変わる。ϕ のラプラシアン係数は，

$$\begin{aligned}
\nabla^2\phi &= \frac{\partial\phi^x}{\partial x} + \frac{\partial\phi^y}{\partial y} \\
&= \frac{\phi^x_{i+\frac{\alpha}{2},j} - \phi^x_{i-\frac{1}{2},j}}{\frac{1+\alpha}{2}\Delta x} + \frac{\phi^y_{i,j+\frac{\beta}{2}} - \phi^y_{i,j-\frac{1}{2}}}{\frac{1+\beta}{2}\Delta x} \\
&= \left(\frac{1}{\Delta x}\right)^2\left\{\frac{2}{\alpha(1+\alpha)}\phi_{i+\alpha,j} + \frac{2}{1+\alpha}\phi_{i-1,j}\right. \\
&\left. + \frac{2}{\beta(1+\beta)}\phi_{i,j+\beta} + \frac{2}{1+\beta}\phi_{i,j-1} - 2\left(\frac{1}{\alpha} + \frac{1}{\beta}\right)\phi_{i,j}\right\},
\end{aligned} \tag{A.2}$$

となる。

流出/流入境界

流入・流出境界では，流れ関数はノイマン条件 $\partial\phi/\partial\boldsymbol{n} = 0$ at $\partial\Omega$ となる。ノイマン条件では，小領域 $[i-\frac{1}{2}, j-\frac{1}{2}]\times[i+\frac{1}{2}, j+\frac{1}{2}]$ に流体が含まれる全グリッドについて計算を行う。流れ関数場のラプラシアン $\nabla^2\phi$ にガウスの発散定理を適用することで，

$$\frac{1}{V}\int_V \nabla^2\phi = \frac{1}{V}\oint \nabla\phi\cdot\boldsymbol{n}, \tag{A.3}$$

を得る。これをグリッドによって離散化すると，法線ベクトル $\boldsymbol{n}$ は縦・横の 4 方向になるため，これらの方向を $n = 1, 2, 3, 4$ とすると，式 (A.3) は

$$\frac{1}{V}\oint \nabla\phi\cdot\boldsymbol{n} = \sum_n (\nabla\phi)_n = \frac{1}{\Delta x}\sum_n (\phi_n - \phi_{i,j}), \tag{A.4}$$

のように書き換えられる。隣接グリッドとの間に境界が通る場合 (図 4)，$(\nabla\phi)_n$ に流体が占める長さ A_n (図中赤線) をかける。よって，最終的に，

$$\nabla^2\phi = \frac{1}{\Delta x}\left\{\sum_n A_n\phi_n - \left(\sum_n A_n\right)\phi_{i,j}\right\}, \tag{A.5}$$

となる。