非線形固有値問題の固有値密度推定法における適応的並列アルゴリズム

山本 和磨[†1] 前田 恭行[†1]
二村 保徳[†1] 櫻井 鉄也[†1]

非線形固有値問題の指定領域内の固有値密度を推定する方法が提案されている．非線形固有値問題は科学や工学の分野において現れ，その解法では固有値に応じたパラメータ設定が必要となる．そのため，固有値密度を用いて効率的なパラメータ設定をすることで解の精度向上や並列効率の改善が期待できる．本論文では，この固有値密度推定を並列計算機において効率的に実行するためのアルゴリズムの提案を行う．従来の固有値密度推定法は複素平面上の指定領域内に一様に配置された計算点毎の線形方程式を解くことを必要としている．そのため，線形方程式の求解時間のばらつきによる並列効率の悪化と計算量が大きくなるという問題がある．提案するマスターワーカー方式の適応型アルゴリズムは並列効率の改善に加え，指定領域内の固有値分布の粗密に合わせて計算点を配置するため，計算量を削減することができる．さらに，線形方程式計算の途中経過を利用し，より効率的に計算する先読み型のアルゴリズムを提案する．また，数値実験によりこれらの提案アルゴリズムの有効性を確認する．

Adaptive parallel algorithm for stochastic estimation of nonlinear eigenvalue density

Kazuma Yamamoto,[†1] Yasuyuki Maeda,[†1]
Yasunori Futamura[†1] and Tetsuya Sakurai[†1]

A numerical method that estimates the eigenvalue density of nonlinear eigenvalue problems in the specified region has been proposed. Nonlinear eigenvalue problems arise in science and engineering. Since parameter settings for eigensolver that based on eigenvalues are required, accuracy and parallel efficiency can be improved by using eigenvalue density. In the present paper, we propose an algorithm for efficient execution of the estimation method on parallel computers. Conventional approach requires the solutions of linear systems for each integral point that uniformly distributed on the complex plane. Thus, it causes the load imbalance and requires a large computational cost due to the variation of solution time for linear systems. The proposed master-worker type adaptive algorithm improves the load balance and reduces the computational cost by the placing integral points according to the density of eigenvalue in the specified region. Moreover, we propose a look-ahead algorithm that balances the loads more efficiently by recycling the variables in the linear solver. We evaluate the efficiency of the proposed algorithms by several numerical examples.

1. はじめに

行列値関数 $F:\mathbb{C}\rightarrow\mathbb{C}^{n\times n}$ について，

$$F(\lambda)\boldsymbol{x}=\boldsymbol{0},\boldsymbol{x}\neq\boldsymbol{0}$$

を満たす固有値 $\lambda\in\mathbb{C}$，固有ベクトル $\boldsymbol{x}\in\mathbb{C}^n$ を求める問題を固有値問題といい，特に F が λ に対して非線形である場合，非線形固有値問題という．

非線形固有値問題は科学や工学の分野において現れる．例えば，量子ドットの電子状態計算からは多項式固有値問題[4)]，遅延微分方程式からは指数関数を含む固有値問題[5)]，高エネルギー加速器設計からは平方根を含む固有値問題が現れる[6)]．このような問題において一部の固有値，固有ベクトルを必要とする場合，固有値解法に与えるパラメータ設定により計算効率が変わる．このとき，大域的に固有値密度がわかっている場合，効率的なパラメータ設定を行うことが可能となる．例えば，文献 1) の櫻井・杉浦法は複素平面上で指定した領域内部の固有値を求めることができる．このとき，求めようとする領域周辺の固有値に対応する固有ベクトルが張る部分空間サイズに応じたパラメータ

†1 筑波大学
University of Tsukuba

をセットすることで精度良く計算ができる．また，固有値が存在しない領域を考慮することで無駄な計算を避けることができる．そのため，事前に固有値密度がわかっている場合，領域指定に関するパラメータを効率的に設定することで，解の精度改善，演算量や演算時間の削減を行うことができる．Jacobi-Davidson 法や Arnoldi 法はユーザが指定した複素平面上の点（以下，シフト点）周辺の複数の固有値を求めるが，複数のシフト点を設定することで複数の固有値を同時に求めることができる．しかしながら，シフト点は固有値に応じて決めなければならないため難しいが，事前に固有値密度がわかっている場合，適切なシフト点を与えることが可能になる．

文献 3)，7) では非線形固有値問題における複素平面上での固有値数推定法および，それを用いた固有値密度推定法が提案された．この固有値数推定法は複素平面上の閉曲線に沿って周回積分を行い，閉曲線内部の固有値数を推定する．一方，固有値数推定法を一様に分割された各領域で適用すると固有値密度が得られる．このとき，複数の線形方程式の求解が必要とされ，それらは全て独立に計算することができる．しかしながら，指定領域を一様に分割する場合，固有値の少ない領域は粗く固有値数推定を行うことが望ましいが，固有値分布の粗密はわからないため，事前に決めることは難しく，従来法では計算する積分点が多くなってしまう．また，各小領域内の固有値数推定は確率的推定法を用いるため，線形方程式の求解に高い精度が必要とは限らない．そのため，反復解法を用いることでより高速に解けることが期待されるが，線形方程式ごとの収束までの反復回数の違いから，方法全体の並列効率の低下が発生する．そこで本論文では，計算処理の負荷分散と固有値の粗密に応じた密度推定を適応的に行う並列アルゴリズムを提案する．また，数値実験によりその有効性を確認する．

本論文の構成は以下の通りである．次章で非線形固有値問題の固有値数推定法と固有値密度推定法について説明する．3 章では提案法について説明し，4 章で数値実験を行い，その結果について考察する．最後に 5 章でまとめを行う．

2. 非線形固有値問題の固有値数推定と固有値密度推定

$F(z) \in \mathbb{C}^{n\times n}$ を n 次の解析的行列関数とする．このとき，複素平面上の閉曲線 Γ 内部の固有値数 m は

$$m = \frac{1}{2\pi \mathrm{i}} \oint_\Gamma \mathrm{tr}\left(F(z)^{-1} F'(z)\right) \mathrm{d}z \tag{1}$$

で求められる[7]．(1) 式を数値計算するために，Γ を中心 γ，半径 ρ の円とした N 点積分則を用いて，

$$m \approx \hat{m} = \sum_{j=0}^{N-1} w_j \mathrm{tr}\left(F\left(z_j\right)^{-1} F'\left(z_j\right)\right) \tag{2}$$

と近似する．ここで $F'\left(z_j\right) = \left.\frac{\mathrm{d}F(z)}{\mathrm{d}z}\right|_{z=z_j}$ とし，閉曲線 Γ の N 個の積分点と重みをそれぞれ $z_j = \gamma + \rho\exp(\frac{2\pi \mathrm{i}(j+1/2)}{N}), w_j = \frac{\rho}{N}\exp(\frac{2\pi\mathrm{i}}{N}\left(j+\frac{1}{2}\right))$ である．ただし，$j = 0, 1, \ldots, N-1$ である．このとき，行列のトレースの確率的推定法を用いて，

$$\begin{aligned} &\mathrm{tr}\left(F\left(z_j\right)^{-1} F'\left(z_j\right)\right) \\ &\approx \tilde{t}_j = \frac{1}{L}\sum_{l=1}^{L}\left(\boldsymbol{v}_l^{\mathrm{T}} F(z_j)^{-1} F'(z_j)\boldsymbol{v}_l\right) \end{aligned} \tag{3}$$

と表すことができる．サンプルベクトル $\boldsymbol{v}_l$ は要素が等確率で 1 か -1 である n 次の確率変数ベクトルとし，L は任意の自然数とする．(2) 式に (3) 式を代入することによって，

$$\hat{m} \approx \tilde{m} = \sum_{j=0}^{N-1} w_j \tilde{t}_j \tag{4}$$

と近似できる．このとき，L 本の独立した線形方程式を解くことになるので，$L < n$ において (2) 式のトレースを求めるよりも計算コストが低くなる．

また，この固有値数推定法を応用して固有値密度の推定を行う．図 1 に示すように複素平面上の実軸 $[a_0, a_1]$，虚軸 $[b_0, b_1]$ で囲まれる領域に対して，閉曲線 $\Gamma_1, \ldots, \Gamma_K$ を配置し，各閉曲線に対して固有値数推定を行う．その結果から固有値の多い領域，少ない領域がわかり，固有値密度を推定できる．閉曲線を同じ大きさの円として配置する場合，すべての閉曲線において積分点の数を 4 点とすると図 1 のように積分点を共有することができるので計算量を削減することができる．また，閉曲線を多くとることで，より細かい分布を見ることができる．

3. 固有値密度推定の適応的並列アルゴリズム

3.1 完全型メッシュ

固有値数推定法を用いて，複素平面の固有値密度を推定することを考える．従来法では指定した複素平面上の領域を一様に分割し，各領域で固有値数推定法を用いることで密度を推定した．このとき生成されるメッシュを**完全型メッシュ**とよぶ．本稿では特に固有値数推定法の積分点数を $N = 4$ とし，格子状になる完全型メッシュを考える．このときの格子点数を N_{all} とする．各積分点では線形方程式の求解を必要とし，

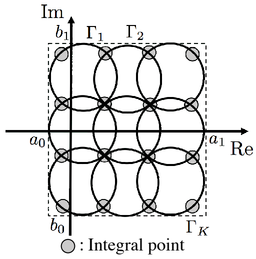

図 1 固有値密度推定法における閉曲線の配置
Fig. 1 Layout of closed curves for estimation method of eigenvalue density

これらは全て独立に計算することができる．この線形方程式の求解は従来，LU 分解などの直接解法を用いてきたが，確率的推定を行うため線形方程式の求解に高い精度が必要とは限らない．そこで Krylov 部分空間反復解法などの反復解法を用い，適当な精度の解が得られたときに反復を停止することにより高速に解けることが期待される．しかしながら，積分点ごとの収束までの反復回数の違いから，方法全体の並列効率の低下が発生することがある．そのため，独立な並列性を持つ線形方程式の求解における負荷分散を考慮したアルゴリズムをマスター・ワーカー方式を用いることで実現する．これを**完全型メッシュアルゴリズム**とよび，図 2 にマスターのアルゴリズム，図 3 にワーカーのアルゴリズムをそれぞれ示す．ここで，マスター・ワーカー方式とは，並列実行時の複数のプロセスをマスターとワーカーに分け，マスターはワーカーへのタスク割り当てと結果の管理を行い，ワーカーは割り当てられたタスクを処理する並列実装方式である．

はじめにマスターは N_{all} 個のタスクをタスクセットに追加する．ここで，タスクはある 1 つの積分点における L 本分の線形方程式の求解とし，タスクセットは未処理のタスクの集合とする．マスターはタスクセットの中からワーカー数に応じて，タスクの処理を各ワーカーに命じる．各ワーカーは受け取ったタスクの情報に基づいて線形方程式を反復解法で解き，収束判定を満たしたとき，(3) 式の計算結果をマスターに送る．ワーカーから計算結果を受け取ったマスターは 4 点分の計算が終わった領域内部の固有値数を推定する．この後，マスターはタスクセットが空でないならばタスクの処理をワーカーに命じる．このタスクはランダムに選択する．マスターが全ての領域について推定固有値数を求めたならば，最後に全ワーカーに終了命令を出して終了する．

3.2 適応型メッシュ

固有値解法では固有値に応じたパラメータ設定が

Complete mesh Master algorithm
Input: number of integral points N_{all}
Output: $\tilde{m}_i$ for $i = 1, \ldots, N_{\mathrm{all}}$

Add N_{all} tasks to TaskSet
Send tasks to all workers
while there exist $\tilde{m}_i$ which have not been computed **do**
  Receive $\tilde{t}_j$ from a worker
  if Converge all integral points in Γ_i **then**
    Compute $\tilde{m}_i$ by Eq.(4)
  end if
  if TaskSet is not empty **then**
    Send a task to free worker
  end if
end while
Send END to all workers

図 2 完全型メッシュアルゴリズムにおけるマスターのアルゴリズム
Fig. 2 Algorithm of master process for complete mesh algorithm

Worker Algorithm 1

Receive task from master
while have not received END **do**
  Solve $F(z_j)^{-1}F'(z_j)\boldsymbol{v}_l$ for $l = 1, \ldots, L$
  Compute $\tilde{t}_j$ by Eq.(3)
  Send $\tilde{t}_j$ to master
  Receive task or END from master
end while

図 3 完全型メッシュアルゴリズムおよび適応型メッシュアルゴリズムにおけるワーカーのアルゴリズム
Fig. 3 Algorithm of worker process for complete mesh algorithm and adaptive mesh algorithm

必要とされる．例えば，櫻井・杉浦法は複素平面上で指定した領域内部の固有値を，Arnoldi 法や Jacobi-Davidson 法はシフト点の周りの固有値を求めるため，それぞれ領域，シフト点を与える必要がある．これらのパラメータはある範囲にいくつの固有値があるかがわかればよく，それをパラメータに用いて正確な値，つまり固有値を求めていくことが可能になる．そのため，固有値が少ない，存在しない範囲を多く存在する領域と同様に細かく指定する必要は低く，固有値の少ない領域はメッシュを粗くし，密に存在するところを細かく指定することが望ましい．しかしながら，一般に固有値分布の粗密は不明であり，事前に決めることは難しいため，本節では固有値分布の粗密に応じて階

層的に求めながら，適応的にメッシュを生成していくアルゴリズムについて述べる．このアルゴリズムでは，はじめに完全型メッシュの積分点のうち，計算資源に応じた点数だけ初期点を選び，粗いメッシュによる初期領域を決定する．各領域で求めた推定固有値数がユーザが与えた閾値 m' よりも大きければ，そのメッシュを分割し新たに計算を必要とする積分点をタスクセットに追加することで，ユーザの求める内部固有値数に応じた密度推定が可能となる．そのため，完全型メッシュと比べて計算する積分点数を減らすことができる．

このメッシュを**適応型メッシュ**とよび，図4に完全型メッシュと適応型メッシュの関係を示す．また，適応型メッシュを用いたマスター・ワーカー方式のアルゴリズムを**適応型メッシュアルゴリズム**とよび，図5にマスターのアルゴリズムを示す．ワーカーのアルゴリズムは完全型メッシュアルゴリズムと同様である．

完全型メッシュアルゴリズムとの違いは以下の2点である．まず，アルゴリズムの1行目で初期タスクの決定を行っている点である．適応型メッシュの初期点は図4右のように指定領域全体をカバーすることが望ましいため，ワーカープロセスに応じた初期点を求める必要がある．次に，アルゴリズムの8行目から10行目の領域分割とタスクの追加である．適応型メッシュでは4点が収束した領域から固有値数推定を行い，m' と比較して分割の必要性を判断する．分割が必要な場合は，新しく計算が必要となる積分点のうち，タスクセットに入っていないものを追加する．このとき，分割された領域の最小サイズを完全型メッシュの領域サイズと同じにすることで，適応型アルゴリズムの最大積分点数は完全型メッシュアルゴリズムの積分点数と等しくなる．適応型メッシュアルゴリズムは完全型メッシュアルゴリズムに比べて計算量を少なくすることができるが，ワーカー数に対してタスク数が少なくなるため，負荷分散がとりづらくなる．

m' の決定は固有値解法への応用により決めればよい．櫻井・杉浦法ならば指定領域内におよそ何個の固有値を含めるか，Arnoldi 法や Jacobi-Davidson 法ならば各シフト点から何個の固有値を求めるかによって m' を定めばよい．これにより，各固有値解法において，求めたい固有値，固有値数に基づいた適切なパラメータ設定が可能になる．

3.3 領域分割の先読み

適応型アルゴリズムで各領域の積分点数を $N = 4$ とするとき，その領域を分割するかどうかは4点めの計算が終わるまで待つ必要がある．このとき，4点めの計算が終わるよりも前に分割の判断をし，分割の必要性があるとわかれば，新しく計算が必要な積分点をより早くタスクセットに追加することができる．これにより，待ち状態のワーカーにより早くタスクを与えることで負荷バランスが向上する可能性がある．そこで積分点3点が収束し，収束していない点の線形方程式解法の反復回数が上限に達したならば，数値積分に必要な重み w_j を収束した3点で計算するよう再定義して，推定固有値数の導出を行う．このとき，積分点が円周上に等間隔に並んでいない場合の w_j の計算は N 次線形方程式

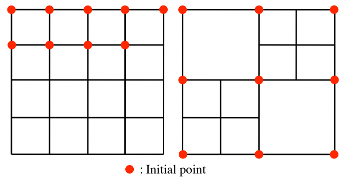

図 4 完全型メッシュと適応型メッシュの関係および計算コア数9のときの初期点の一例（左：完全型メッシュ，右：適応型メッシュ）
Fig. 4 Relationship between complete mesh and adaptive mesh. An example of initial integral points for 9 processes (left:complete mesh, right:adaptive mesh)

Adaptive mesh Master algorithm
Input: threshold value m'
Output: $\tilde{m}_i$ for $i = 1, 2, \ldots$

1: **Add** $n_{\rm p}-1$ initial tasks to TaskSet ($n_{\rm p}$ is number of process)
2: **Send** tasks for all workers
3: **while** there exist $\tilde{m}_i$ which have not been computed **do**
4: **Receive** $\tilde{t}_j$ from a worker
5: **if** Converge all integral points in Γ_i **then**
6: **Compute** $\tilde{m}_i$ by Eq.(4)
7: **end if**
8: **if** $\tilde{m}_i > m'$ **then**
9: **Add** new tasks to TaskSet
10: **end if**
11: **if** TaskSet is not empty **then**
12: **Send** tasks to free workers
13: **end if**
14: **end while**
15: **Send** END

図 5 適応型メッシュアルゴリズムにおけるマスターのアルゴリズム
Fig. 5 Algorithm of master process for adaptive mesh algorithm

$$\sum_{j=0}^{N-1} w_j \zeta_j^{k-1} = \begin{cases} 1, & k=0 \\ 0, & k=1,\ldots,N-1 \end{cases} \tag{5}$$

を解くことで求められる[8]．(5) 式で得られた w_j を用いて (4) 式の計算を行うことで，先読みした推定固有値数が求められる．この先読みを行うために，各ワーカーにおける線形方程式解法の反復回数上限の設定で反復を途中で停止させ，マスターから命令が来たときに計算を再開するようにする．前述の適応型アルゴリズムに先読みを追加したアルゴリズムを**先読み付き適応型メッシュアルゴリズム**と呼ぶ．図 6 にマスターのアルゴリズム，図 7 にワーカーのアルゴリズムを示す．マスターアルゴリズムの適応型メッシュアルゴリズムとの変更，追加は以下の通りである．5-8 および 13 行目に先読みに関する追加を行った．また，14-18 行目ではワーカーの反復の停止と再開を実現するためにマスターからの命令，ワーカーからの結果の受け取りに変更を行った．ワーカーのアルゴリズムは 4-11 行目で反復の停止と再開を実現するために，マスターとのやり取りに関して変更を行った．

3 点積分による先読みで得られる推定固有値数は一般に 4 点積分の場合と異なる．そのため，先読みの判断の誤りには 2 種類のタイプがある．1 つ目は 3 点では分割の必要がないと判断したが，4 点で求めた場合では分割が必要と判断した場合である．これはタスクセットへの追加のタイミングが先読みなしと同じになるため，先読みのメリットは得られないが，デメリットもない．2 つ目は 3 点では分割が必要と判断したが，4 点で求めた場合では分割が不必要だった場合である．この場合は先読み無しの場合には計算の必要がなかった積分点をタスクセットに追加してしまうため，計算量を増やしてしまうというデメリットがある．

先読みの目的はタスクセットの中身が 0 のために次のタスクが割り振られず，待ち状態となったワーカーを出さないことにあり，ワーカーに割り振るタスクがあるときはリスクを冒してまで先読みをする必要はない．この先読みのリスクを軽減するために，先読みはタスクセットが 0 のときのみ行うようにした．

Adaptive mesh Master Algorithm with Look-ahead
Input: threshold value m'
Output: $\tilde{m}_i$ for $i = 1, 2, \ldots$

1: **Add** $n_{\mathrm{p}}-1$ initial tasks to TaskSet（n_{p} is number of process）
2: **Send** tasks to all workers
3: **while** there exist $\tilde{m}_i$ which have not been computed **do**
4:   **Receive** *result*
5:   **if** Converge integral points in $\Gamma_i \geq 3$ **then**
6:     **if** 1 point NotConverge and TaskSet is empty **then**
7:       **Compute** w_j by Eq.(5)
8:     **end if**
9:     **Compute** $\tilde{m}_i$ by Eq.(4)
10:     **if** $\tilde{m}_i > m'$ **then**
11:       **Add** tasks to TaskSet
12:     **end if**
13:   **end if**
14:   **if** *result* is NotConverge **then**
15:     **Send** CONTINUE
16:   **else if** TaskSet is not empty **then**
17:     **Send** tasks to free workers
18:   **end if**
19: **end while**
20: **Send** END

図 6 先読み付き適応型メッシュアルゴリズムにおけるマスターのアルゴリズム
Fig. 6 Adaptive mesh master's algorithm with look-ahead

表 1 実験で用いた非線形固有値問題
Table 1 Nonlinear eigenvalue problems used in numerical example

	$F(z)$	Dimension
QDotSub2	$\sum_{i=0}^{2} z^i A_i$	245
QDotSub4	$\sum_{i=0}^{4} z^i A_i$	2475
Butterfly	$\sum_{i=0}^{4} z^i A_i$	64

4. 数値実験

4.1 実験環境

数値実験は，CPU: AMD Opteron(tm) Processor 6180 SE(2.5GHz) 12-Core × 4，Memory: DDR3 SDRAM 8GB × 32 (Total 256 GB)，OS: Cent OS 5.4 上で行い，C 言語と MPI を用いて実装した．

表 1 に実験で用いた非線形固有値問題を示す．量子ドット（Quantum Dot，以下 QDot）の電子状態計算[4] から現れる 5 次多項式固有値問題の 2 次の項までを用いた問題を QDotSub2，4 次の項までを用いた問題を QDotSub4 とした．Butterfly は NLEVP[2] の行列“ Butterfly ”である．

また，表 2 に各問題で密度推定を行った領域を示す．本論文では Butterfly は参考文献 2 で示された固有値分布の図の範囲と同様にした．また，QDotSub2 と

Worker Algorithm 2

Receive task from master
while have not received END **do**
  Solve $F(z_j)^{-1}F'(z_j)\boldsymbol{v}_l$ for $l = 1, \ldots, L$
  if equation solver converge **then**
    Compute $\tilde{t}_j$ by Eq.(3)
    $result \leftarrow \tilde{t}_j$
  else
    $result \leftarrow$ NotConverge
  end if
  Send $result$ to master
  Receive task, CONTINUE or END
end while

図 7 先読み付き適応型メッシュアルゴリズムにおけるワーカーのアルゴリズム
Fig. 7 Algorithm of worker process for adaptive mesh algorithm with look-ahead

表 2 推定領域
Table 2 Regions for estimation

	実軸 $[a_0,\ a_1]$	虚軸 $[b_0,\ b_1]$
QDotSub2	[-0.56, 0.04]	[-0.19, 0.21]
QDotSub4	[-1.1, 0.9]	[-0.7, 1.3]
Butterfly	[-2, 2]	[-2, 2]

QDotSub4 は固有値が存在する領域 D 内の大まかな固有値密度を提案法で求め，その結果から比較的複素平面に固有値が広がっている範囲を選択した．領域 D は，多項式固有値問題はコンパニオン行列生成により一般化固有値問題に帰着し[4)]，この一般化固有値問題に Arnoldi 法などを適用することにより絶対値最大の固有値のみを求めることで定めることができる．ただし，多項式固有値問題以外の非線形固有値問題においては固有値数が無限大となることがあり，さらに無限大固有値を持つこともあるため，前述した方法によって範囲を求めることが難しい場合も存在する．なお，科学や工学の分野で現れる非線形固有値問題は物理的背景から求める固有値の範囲がある程度指定できる場合があり，それらの情報を用いて範囲の設定を行うことも考えられる．

密度推定法のサンプルベクトル数は $L = 32$ とした．線形方程式の反復解法にはリスタート付き GMRES 法[10)] を用い，収束条件は相対残差が 10^{-3} を下回ったときとし，リスタート数を 30 とした．前処理には ILU(0) 前処理[9)] を用いた．これ以外のパラメータについては各実験で示す．

表 3 各メッシュの積分点数
Table 3 Number of integral point

	完全型メッシュ	適応型メッシュ（適応型/完全型）
QDotSub2	35	35（1.00）
	117	79（0.67）
	425	167（0.39）
	1617	315（0.19）
	6305	560（0.09）
	24897	859（0.03）
Butterfly	25	25（1.00）
	81	81（1.00）
	289	217（0.75）
	1089	505（0.46）
	4225	1081（0.26）
	16641	2443（0.17）

4.2 アルゴリズムの実装方法

実装では積分点を管理する point 構造体と領域を管理する region 構造体を用いた．region 構造体はメンバに point 構造体へのポインタを 4 つ持ち，point 構造体はメンバに $\tilde{t}_j$ の値を持つ．そのため，ある積分点のタスクが完了し $\tilde{t}_j$ を受け取ったとき，その積分点を含む region について $\tilde{m}_i$ を計算するようにした．

各アルゴリズムのマスターは完全型メッシュの情報を持ち，完全型メッシュのサイズを適応型メッシュの最小サイズとしている．領域分割が必要になったときは完全型メッシュの情報と分割前の領域に関する情報から，新しい領域とそれに必要な point 構造体を求め，$\tilde{t}_j$ が得られていない未計算の積分点のみ新しいタスクとして追加する．

4.3 実 験 結 果

4.3.1 適応型メッシュの有効性

図 8，9 に QDotSub2，Butterfly をそれぞれ適応型メッシュアルゴリズムで解いたときのメッシュ構造と固有値の分布を示す．固有値は MATLAB の関数 `polyeig` で求めた値である．今回の実験では，固有値が存在しないと思われる領域では分割を行わないようにするために，閾値 m' を 0.5 とした．QDotSub2 では初期タスクに初期領域を実軸 3 分割，虚軸 2 分割したときの積分点 12 点を与え，Butterfly では初期タスクに初期領域の端点 4 つを与えた．固有値の存在する周辺の領域ほど分割が行われており，固有値密度に応じたメッシュとなっていることがわかる．

また，表 3 に完全型メッシュの全積分点数を変えたときの適応型メッシュの全積分点数を示す．完全型メッシュでより細かい密度を調べるために分割数を増やしたときでも，適応型メッシュは積分点数の増加が抑えられていることがわかる．実際に要求される詳細度はユーザの用途に応じて異なるが，完全型メッシュ

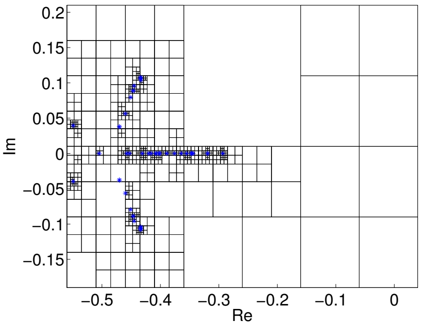

図 8　QDotSub2 の適応型メッシュと固有値分布
Fig. 8　Adaptive mesh and eigenvalue distribution of QDotSub2

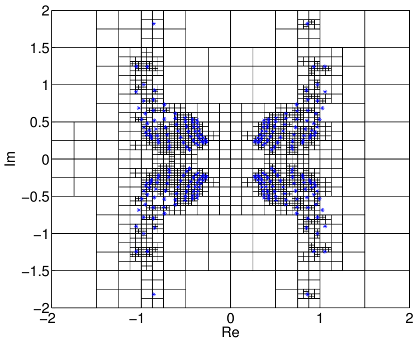

図 9　Butterfly の適応型メッシュと固有値分布
Fig. 9　Adaptive mesh and eigenvalue distribution of Butterfly

表 4　先読みの正誤
Table 4　Efficiency of look-ahead algorithm

	先読みで分割する	先読みは分割しない
実際に分割する	42	2
実際に分割しない	14	48

の積分点数を広く変化させても適応型メッシュでは完全型メッシュより計算量を増やさずに固有値が密に集まっている部分の詳細な推定密度を得ることができている．

実際に固有値解法のパラメータ設定を考える場合には，固有値が少ない領域は細かい分布を調べる必要性は少ないため，適応型メッシュでも十分な情報が得られている．例えば，本実験で得られる結果は固有値が多く存在する領域には計算資源を多く割り当てる，存在しない領域は計算を行わないなど，効率的な固有値計算に役立てることができる．

4.3.2　先読みの有効性

次に，先読みの有効性を調べる．問題は QDotSub4 を用い，並列数 n_{p} は 48 とした．このとき，マスタープロセスは 1 個，ワーカープロセスは 47 個である．密度推定の範囲は実軸 [-1.1, 0.9]，虚軸 [-0.7, 1.3] とし，初期領域を各軸ともに 4 分割し，25 点を初期タスクとした．また，m' は 20 とした．

図 10 は先読み付き適応型アルゴリズムの各ワーカープロセスの実行状況である．図 11 は先読みを行わない適応型アルゴリズムの各ワーカープロセスの実行状況である．横軸はマスタープロセスの経過時間で，縦軸が各ワーカープロセスである．ワーカーが線形方程式を解いている時間を「実行時間（busy time）」と呼び，各図で赤い線で示した．また，解いていない時間を「待ち時間（wait time）」と呼び，各図の線のない部分にあたる．グラフの結果から初期タスクが割り当てられなかったワーカーに対し，先読みをすることでより早くタスクを渡し，始まりを早くすることができている．また，先読み無しでは 3 秒前後においてタスクリストが空になり，待ち状態のワーカーが多数現れたが，先読み付きでは同じ時間において十分にワーカーが働いている．これらから，先読みが約 4 秒までの結果において有効であったことがわかる．マスタープロセスの実行時間は先読み無しが 11.82 秒，先読み付きが 11.26 秒と 0.56 秒速くなった．ただし，本実験は行列のサイズが科学や工学の実用レベルの問題と比べて小さいことや本実験が先読みの有効性の確認であることから，固有値解法の効率化にどの程度寄与するかの評価については今後の課題である．本実験では最後にいくつかの線形方程式の反復数が非常に多くなり，実行時間が長くなった．これは領域を細かくした際に積分点が固有値に近い値をとってしまったため，反復数が非常に増えたと考えられる．このような状況に対する改善案として，線形方程式の最大反復回数を分割前の 1 つ粗い領域における各積分点の反復回数から定めるといった方法が考えられる．表 4 は先読みの正誤を示した．先読みは全部で 106 回行われた．「先読みで分割し実際は分割する」と「先読みで分割せず実際は分割しない」場合は先読みが正しく行われており，合計で 90 回あった．そのため，正答率は約 84.9%となった．先読みが失敗した内訳は「先読みで分割せず実際は分割する」という場合は 2 回発生した．これは先読み無しと同じであるため，問題はない．しかし，「先読みで分割し実際は分割しない」という判断が 14 回行われてしまった．この場合は本来必要のない計算が発生するため，計算量が増加してしまう．失敗の原因としては 3 点積分による先読みにおける推定値の精度悪化が考えられる．この改善案として，対象とする領域の分割前の一階層粗い領域の推定数と，そこから

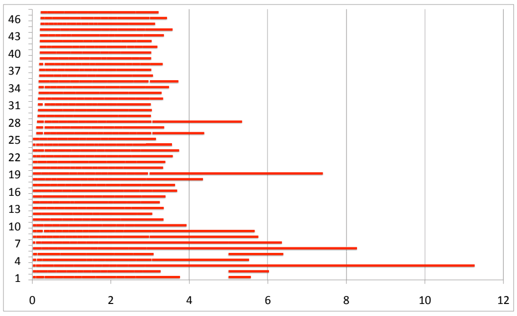

図 10 先読み付き適応型アルゴリズムの各ワーカープロセスの実行状況

Fig. 10 Timeline of busy time and wait time for each worker process on adaptive mesh algorithm with look-ahead.

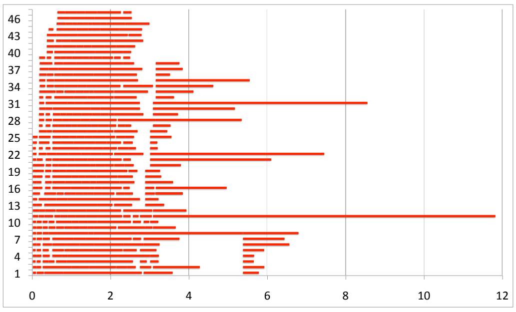

図 11 適応型アルゴリズムの各ワーカープロセスの実行状況

Fig. 11 Timeline of busy time and wait time for each worker process on adaptive mesh algorithm.

分割され計算の終わった別の領域における推定数を用いて対象とする領域の推定数の精度を改善していくことが挙げられる．

5. まとめ

本論文では非線形固有値問題の固有値密度推定法を計算コストと負荷バランスを考慮し並列に実行するアルゴリズムを提案した．完全型メッシュアルゴリズム，先読み無し適応型メッシュアルゴリズム，先読み付き適応型メッシュアルゴリズムの３つのアルゴリズムについて議論を行った．さらに数値実験によって適応型メッシュアルゴリズムは完全型メッシュアルゴリズムに対して計算量が少なくなること，先読みでより早くタスクを追加できたために待ち状態になるワーカー数を減らせたこと，先読みの正解率が 84.9%であったことを確認し，提案法の有効性を示した．しかしながら，最後に反復数が非常に多くなるタスクが発生し実行時間が長くなっていたこと，先読み付きアルゴリズムでは先読みの失敗がしばしば発生することも確認した．

今後の課題としては提案法の結果に基づく固有値解法のパラメータ設定および大規模問題への適用，先読み付き適応型メッシュアルゴリズムの正答率改善，線形方程式の最大反復回数の設定方法の検討などが挙げられる．

謝辞 本研究は科研費（21246018，23105702）及び CREST（ポストペタスケール高性能計算に資するシステムソフトウェア技術の創出）の助成をうけた．

参考文献

1) J. Asakura, T. Sakurai, H. Tadano, T. Ikegami and K. Kimura, A numerical method for polynomial eigenvalue problems using contour integral, Japan J. Indust. Appl. Math., 27:73-90, 2010.
2) T. Betcke, N.J. Higham, V. Mehrmann, C. Schroder and F. Tisseur, NLEVP: A Collection of Nonlinear Eigenvalue Problems, MIMS EPrint 2010.98, 2010.
3) Y. Futamura, H. Tadano, and T. Sakurai, Parallel stochastic estimation method for eigenvalue distribution, JSIAM Letters, 2:127-130, 2010.
4) F.-N. Hwang, Z.-H. Wei, T.-M. Huang and W. Wang, A Parallel Additive Schwarz Preconditioned Jacobi-Davidson Algorithm for Polynomial Eigenvalue Problems in Quantum Dot Simulation, J. Comput. Phy., 229:2932-2947, 2010.
5) E. Jarlebring, K. Meerbergen and W. Michiels, An Arnoldi method with structured starting vectors for the delay eigenvalue problem, Proceedings of the 9th IFAC Workshop on Time Delay Systems, Prague, June 7-9, 2010
6) B.S. Liao, Subspace projection methods for model order reduction and nonlinear eigenvalue computation, PhD thesis, Department of mathematics, Univ. of California at Davis, 2007.
7) Y. Maeda, Y. Futamura and T. Sakurai, Stochastic estimation method of eigenvalue density for nonlinear eigenvalue problem on the complex plane, JSIAM Letters, 3:61-64, 2011.
8) H. Ohno, Y. Kuramashi and T. Sakurai, A quadrature-based eigensolver with a Krylov subspace method for shifted linear systems for Hermitian eigenproblems in lattice QCD, JSIAM Letters, 2:115-118, 2010.
9) Y. Saad, Iterative methods for sparse linear systems, SIAM, Philadelphia, 2003.
10) Y. Saad and M.H. Schultz, GMRES: A generalized minimal residual algorithms for solving nonsymmetric linear systems, SIAM J.Sci.Stat.Comput, 7:856-869, 1986.