Vol.2017-HPC-160 No.31
2017/7/28

高並列環境下に向けた粗いレベルでニアカーネルベクトルを抽出および追加的に設定する手法の性能評価

野村直也[†1]　中島研吾[†1]　藤井昭宏[†2]　田中輝雄[†2]

概要：大規模連立一次方程式を高速に解く解法として Algebraic Multigrid（AMG）法がある．本研究はその中の SA-AMG（Smoothed Aggregation - AMG）法に着目し研究を行っている．SA-AMG 法は，与えられた問題行列から粗い問題を再帰的に生成し，それを用いることで高い収束性を実現している．ここで，粗い問題を生成する際に問題に応じたニアカーネルベクトルを用いることで，収束性をさらに高められることが知られている．ニアカーネルベクトルの設定に関しては，α SA 法を含めいくつかの関連研究が存在する．これらのほとんどの研究では，階層の全レベルにおいてニアカーネルベクトルの本数を固定し設定を行っている．そこで本研究では，粗いレベルでニアカーネルベクトルを複数本抽出し，それらを用い追加的に設定本数を増やす手法についての分析を行う．この実験では，全階層でニアカーネルベクトルの抽出と追加を行う手法についての，収束性や実行時間に着目し評価を行っている．この手法を用いることで，粗いレベルの行列が全体の収束性に与える影響が大きい時には，収束性が大きく改善することが十分に考えられる．本研究では高並列環境下において，ニアカーネルベクトルの本数の増加による計算コストの増加と，収束性改善の効果のバランスを，数値実験を通して考察する．

1. はじめに

コンピュータによるシミュレーションに基づく計算科学は，工学における設計現場，気象予報，渋滞予測などの身近な場面で幅広く活用されている．このような問題の多くは最終的には連立一次方程式$\boldsymbol{Ax} = \boldsymbol{b}$を解くことに帰着される．近年では，より複雑な問題を正確にシミュレートすることが求められており，それにより問題の大規模化が進んでいる．そのため，大規模連立一次方程式を高速かつ安定に解く手法の開発は急務となっている．

そこで，大規模連立一次方程式を高速に解く手法として Algebraic Multigrid（AMG）法が提案されている[1]．AMG 法は問題行列から階層的に，次元数の異なる行列を作成して解く手法である．AMG 法の中の解法のひとつに，Smoothed Aggregation に基づく AMG 法（SA-AMG）がある[2][3][4]．この手法は多くの問題に対して有用であることが知られており，広く用いられている解法となっている．本研究では，この SA-AMG 法を対象に研究を行っている．

SA-AMG 法は，問題生成部（以下，構築部）と，反復解法部（以下，解法部）に分かれている．構築部では，問題行列に基づくグラフ構造を作成し，それを基にアグリゲートと呼ばれる節点集合を作成する．これを再帰的に行うことで，次元数のより小さい複数の行列を作成する．解法部では，構築部で階層的に生成された行列を用い，Gauss-Seidel 法などの緩和法を用いて問題行列を解く．このとき，元の問題行列などの大規模な問題が設置される階層（レベル）を細かいレベル，問題行列から生成された小規模な行列が設置される階層（レベル）を粗いレベルと呼ぶ．

SA-AMG 法は，収束しにくい成分を粗いレベルで解くことで，高い収束性を実現している．ここで，収束しにくい成分とは，一般にニアカーネルベクトルと呼ばれ，問題行列$\boldsymbol{A}$との行列ベクトル積が 0 に近くなるようなベクトルのことをいう．Gauss-Seidel 法のような通常の定常反復解法で解いた際，この成分が収束の停滞を引き起こす要因となることが知られている．AMG 法は，このような定常反復解法では減衰しにくい誤差成分を，粗いレベルの行列を用いることで効率よく減衰させ，高い収束性を実現している．SA-AMG 法ではさらに，構築部においてニアカーネルベクトルを用いて粗いレベルの行列を生成することにより，小さい行列で効率よくニアカーネルベクトルを解き，残差を効率よく収束させることが可能となる．

本研究では，3 次元弾性体の問題を使用しており，この問題では平行移動成分と回転成分がニアカーネルベクトルとして知られている．SA-AMG 法においてこれらのニアカーネルベクトルを設定することにより，反復回数と実行時間双方で改善がみられることが知られている．そこで我々は，問題行列に応じた適切なニアカーネルベクトルの設定方法と，その効果についての研究を行ってきた．我々の現在までの研究では，α SA 法[5]と呼ばれる手法をもとに V-cycle を用いて問題行列からニアカーネルベクトルを複数本抽出する手法の実装を行った．そして，その手法で抽出したニアカーネルベクトルを SA-AMG 法に適切な本数設定することで，平行移動成分と回転成分を設定した場合よりも，反復回数と実行時間双方で改善がみられることがわかっている[6]．

[5]や[6]を含め，従来の関連研究では，階層の全レベルにおいてニアカーネルベクトルの本数を固定し設定を行っている（例えば，最も粗いレベルで 3 本設定した場合，粗い

†1 東京大学
The University of Tokyo
†2 工学院大学
Kogakuin University

レベルでも 3 本用いて計算を行っている). これは SA-AMG 法では通常，細かいレベルのニアカーネルベクトルをもとに，次の粗いレベルのニアカーネルベクトルを生成しているためである．しかし我々は，細かいレベルから生成されたニアカーネルベクトルだけでは，粗いレベルの行列におけるニアカーネルベクトル成分の減衰が不十分であり，高い収束性を生かしきれていないのではないかと考えた．そこで本研究では[6]の手法を改良し，粗いレベルでニアカーネルベクトルを複数本抽出する手法の実装を行った．そして，SA-AMG 法においてそれらを用い粗いレベルで追加的に設定本数を増やすことによる反復回数や実行時間への影響の分析を行った．また，上記の手法を用いながら，V-cycle に用いる緩和法を変更することによる反復回数や実行時間への影響の調査も行った．

2. SA-AMG 法

SA-AMG 法は AMG 法の中の解法のひとつであり，与えられた問題行列から階層的に小規模な問題行列を生成し，それらを用いて解く手法である．本章ではまず，AMG 法についての説明を行い，次に SA-AMG 法についての説明を行う．

2.1 AMG 法

AMG 法は，大規模な非構造格子による線形問題を高速に解く数値解法のひとつである．AMG 法ではまず，与えられた問題行列を複数段階に分けて小規模な行列を生成し，これらを用いて問題行列を解く．AMG 法は大きく分けて構築部と解法部の 2 つの処理からなる．構築部は問題行列から未知数間のグラフ構造を作り，次のレベルに残す未知数を選択する．そして，粗いレベルの行列を生成する．これを再帰的に行うことで，階層的に生成された行列を作る．一方，解法部は構築部で生成された行列を用い，実際に問題を解く．

構築部の構造を図 1 に示す．構築部では細かいレベルの問題行列を基に，粗いレベルの問題行列や補間演算子である Prolongation（P）行列と Restriction（R）行列を階層的に

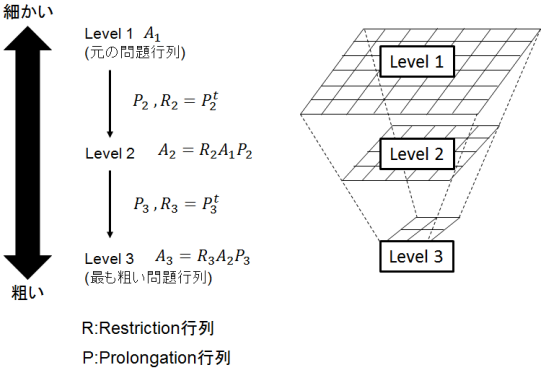

図 1　構築部

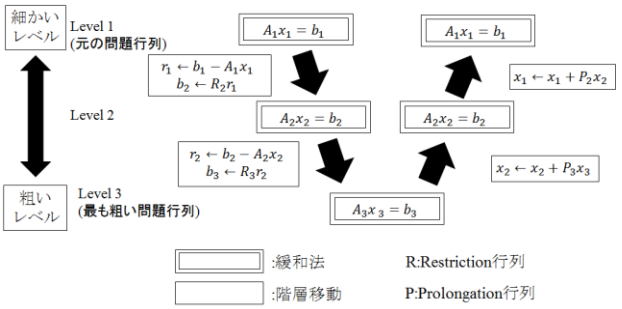

図 2　解法部

生成する．AMG 法は階層型で，最上階に与えられた問題行列を用い，階層が下がるにつれて問題行列より小規模な行列を生成する．階層数は問題サイズによって可変となる．粗いレベルの問題行列は，横長の Restriction 行列と縦長の Prolongation 行列，さらに現階層の問題行列とで行列行列積（RAP）を行うことで作成している．

解法部の構造を図 2 に示す．解法部は，主に行列ベクトル積と緩和法から成り立っている．この図では，最上層に与えられた問題行列が設置され，階層が下がるにつれて，問題行列より小規模な行列が設置される．階層数は問題サイズによって可変となる．階層移動では，構築部で用意した補間演算子の Prolongation 行列と Restriction 行列を使う．階層を下りる際には，現階層の残差を計算し，横長の Restriction 行列と行列ベクトル積を行うことで短いベクトルを生成し，ひとつ下の階層で利用する．階層を上る際は，現階層の解と縦長の Prolongation 行列との行列ベクトル積を行うことで長いベクトルを生成し，ひとつ上の階層の補正解として利用する．複数の階層を行き来する様子が V 字を連想させるため，このような解法部を V-cycle と呼ぶ．

補間演算子から行列の階層構造が生成されるため，この補間演算子の生成手法により様々な AMG 法が存在する[6]．本研究では，その中でも SA-AMG 法を対象とする．この手法は問題行列のみから未知数間の依存関係を定義する．そして，依存関係のある未知数同士で集合を作り，その集合内で重みづけをして補間演算子を生成する．SA-AMG 法はさまざまな分野で利用されており，AMG 法の代表的な手法のひとつとなっている．

2.2 SA-AMG 法

SA-AMG 法では，問題行列に基づく節点と辺で構成されたグラフ構造を用いて粗い問題を作成していく．ここで，問題行列の各行が節点に対応し，非ゼロ要素が辺に対応している．粗い問題を作成する際に，節点全体をアグリゲートと呼ばれる節点集合に分解する．アグリゲートは図 3 のように，次の粗いレベルで 1 つの節点に対応し，グラフ構造である節点を中心に近くの節点をまとめた節点集合と定義される．任意の節点がどこか 1 つのアグリゲートに属するように，アグリゲートが生成される．このアグリゲート

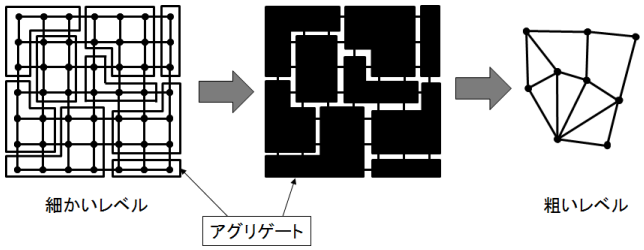

図 3　アグリゲート生成

内の未知数に重み付けをすることで補間演算子を計算し，行列の階層構造を作成する．補間演算子である Prolongation 行列と Restriction 行列は，行列の階層構造を作成する際に用いられる．これらの行列の生成にニアカーネルベクトルを用いることで，収束性をさらに高めることができる．このことについては，次節で説明を行う．

3. ニアカーネルベクトル

この章ではニアカーネルベクトルの定義と，SA-AMG 法においてニアカーネルベクトルがどのように用いられるかを説明する．

ニアカーネルベクトルとは，問題行列Aとの行列ベクトル積が 0 に近くなる 0 ベクトルでないベクトル（ニアカーネルベクトルを$\boldsymbol{v}$とおくと，$A\boldsymbol{v} \approx 0, \boldsymbol{v} \neq 0$）のことを言う．一般的な定常反復解法（例：Gauss-Seidel 法）を用いる場合，残差（$\boldsymbol{b} - A\boldsymbol{x}$）を基に解の修正を行うが，ニアカーネルベクトルの影響により$A\boldsymbol{x}$が 0 に近くなり，残差が停滞してしまう可能性がある．

SA-AMG 法では，補間演算子である Prolongation 行列と Restriction 行列にニアカーネルベクトルを用いることで，粗いレベルに収束しにくい成分を移動させることができる．これにより，粗いレベルでニアカーネルベクトル成分を解くことができるため，効率よく解を収束させることが可能となる[2][3][4].

問題の性質からニアカーネルベクトルをある程度特定できる場合もある．例えば，本研究で用いている弾性体の問題では，平行移動成分と回転成分がニアカーネルベクトルとして知られている．弾性体の問題を対象としている場合，これらを SA-AMG 法に用いることで，収束性が高まることが知られている．

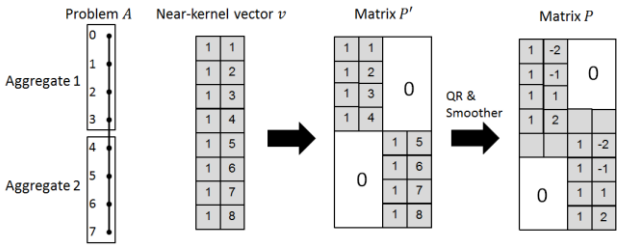

図 4　ニアカーネルベクトルを用いた Prolongation 行列の作成方法

ニアカーネルベクトルの補間演算子への設定方法を図 4 に示す．図 4 は 2 本のニアカーネルベクトルを用いて，問題Aから 2 つのアグリゲートを作成するときの，Prolongation 行列の作成方法を図示している．この図のように行列Pを生成する際には，アグリゲートの節点番号に対応した場所のニアカーネルベクトルの要素を抜き出し，それを並べて QR 分解し，スムーザをかけることで Prolongation 行列を作成する．この時作成された R 行列は，次のレベルのニアカーネルベクトルとして用いられる．SA-AMG 法では，ニアカーネルベクトルを複数本設定することができる．その場合，抜き出す際のベクトル本数が増え補間演算子の行列が大きくなり，結果として計算量が増えることとなる．そのため，ニアカーネルベクトルの本数と実行時間でトレードオフが発生すると考えられる．

図 5 と図 6 に，SA-AMG 法においてニアカーネルベクトルを複数本設定することによる反復回数の変化を示す．図 5 にこの実験で対象とした問題を示す．この問題は 3 次元弾性体問題である．弾性体の問題については，5 節で説明する．この弾性体の問題は，上半分が柔らかい物体に対して，ある一部分に力を加え，どのように変形するかを解く問題となっている．また，ヤング率を上半分が 0.8，下半分を 1 に設定している．反復の終了条件は相対残差が1.0×10^{-7}となったときとした．また，問題サイズについては，1 プロセスあたり 6×15×60 としたウィークスケーリングで計測を行った．図 6 に，収束までに必要とした反復回数のグラフを示す．グラフの横軸はプロセス数であり，縦軸は反復回数となっており，下に行くほど反復回数が少な

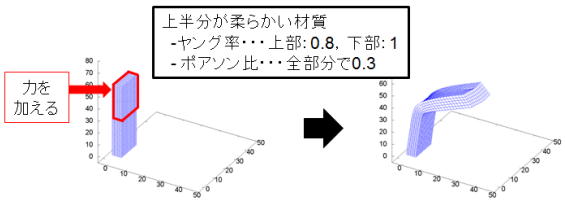

図 5　予備実験での問題設定

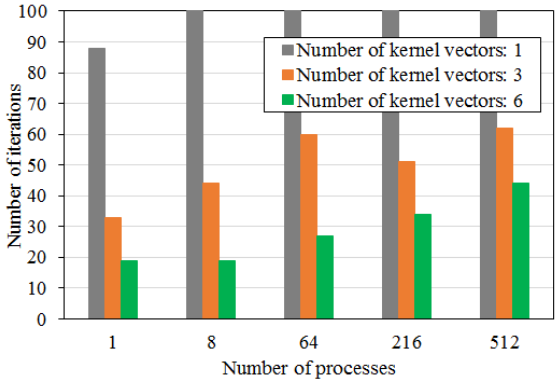

図 6　ニアカーネルベクトルを複数本設定することによる SA-AMG 法の反復回数の変化

く，収束性が良いことを示している．また，グラフの要素である“Number of kernel vectors: 1”はすべての要素が 1 の定数ベクトルをニアカーネルベクトルとして設定した場合，“Number of kernel vectors: 3”は平行移動成分のみを設定した場合，“Number of kernel vectors: 6”は平行移動成分と回転成分を設定したときのグラフとなっている．図 6 より，ニアカーネルベクトルを複数本設定することで，収束性の改善がみられることがわかる．これは，適切なニアカーネルベクトルが設定できているため，このような傾向がみられたと考えられる．本論文では，ニアカーネルベクトルを問題行列から抽出することにより，これらのベクトルよりもさらに反復回数の改善が可能となる性質のよいニアカーネルベクトルが設定できるかの検証も行う．

4. ニアカーネルベクトル抽出手法

この節では，本研究で用いたニアカーネルベクトルを問題行列から抽出する手法について説明する．

ニアカーネルベクトルを抽出する手法として，αSA 法と呼ばれる手法が提案されている[5]．αSA 法について説明したものを図 7 に示す．この手法では図 7 のように，まずニアカーネルベクトルの条件である$\boldsymbol{Ax} \approx 0$を，V-cycle を複数回適用することで解く．その後，計算結果であるベクトル$\boldsymbol{x}$に対し Restriction を行い，次のレベルの抽出に用いる初期ベクトルとする．これらの操作を繰り返し，最終的に最下層のひとつ前のレベル（L-1）になったら，現在のベクトル$\boldsymbol{x}$に対し Prolongation を行い，そのベクトルをニアカーネルベクトルし出力を行う．上記の手順により，αSA 法はニアカーネルベクトルの抽出を行う．また我々はαSA 法を基に，αSA 法の 1 階層のみを用いてニアカーネルベクトルの抽出を行う手法の実装および評価を行った．これにより，適切な本数抽出および設定を行うことで，有用となることが確認されている[6].

上記の研究を含め，ニアカーネルベクトルの設定に関する研究では，全階層でニアカーネルベクトルの本数を固定している．これは，SA-AMG 法では通常，細かいレベルのニアカーネルベクトルを基に，粗いレベルのニアカーネルベクトルを生成しているためである．我々は，粗いレベルのニアカーネルベクトルは，細かいレベルのニアカーネルベクトルだけでは不十分ではないかと考えた．そこで，[6]の抽出手法を基に，粗いレベルにおいてもニアカーネルベクトルを抽出および SA-AMG 法にて追加的に設定する手法の実装を行った．抽出手法の概略を図 8 に示す．まず，抽出する際の初期ベクトルとしてベクトル$\boldsymbol{x}$を用意し，乱数で初期化する（3 行目）．そして，V-cycle を$\boldsymbol{Ax} = 0$の問題を解くように適用する．これをμ回繰り返す（4 行目）．そして，以上により出てきたベクトル$\boldsymbol{x}$をニアカーネルベクトルの候補群Bに追加し（5 行目），Bをもとに再度全階層を構築する（6 行目）．これを抽出したい本数分行うことで，ニアカーネルベクトルの複数本抽出を行う（2～7 行目）．そして上記を各レベルで行うことで，複数階層にわたり抽出を行い（1～8 行目），最後にニアカーネルベクトル群Bの出力を行う（9 行目）．上記のようにして，複数階層にわたりニアカーネルベクトルの複数本抽出を行っている．SA-AMG 法に用いる際には，まずは従来と同様に，最も細かいレベルにおいて抽出されたニアカーネルベクトルを設定し，次の粗いレベルの行列とニアカーネルベクトルを生成する．そして，粗いレベルでも同じように細かいレベルのニアカーネルベクトルをもとに，さらに粗いレベルを生成していくが，この時に細かいレベルをもとに生成されたニアカーネルベクトルに追加して，抽出されたニアカーネルベクトルを用いる．この操作を最大レベル数－1 まで行う．このように本手法では，[6]で用いた手法に加え粗いレベルのニアカーネルベクトルも考慮しているため，粗いレベルの行列が全体の収束性に大きな影響を与えている場合，[6]よりもさらに収束性が改善することが見込める．

外部から入力するパラメータとして，$extract_number$とμが存在する．$extract_number$は抽出したい本数であり，実験ではこの値を変更することによる性能への影響の分析を

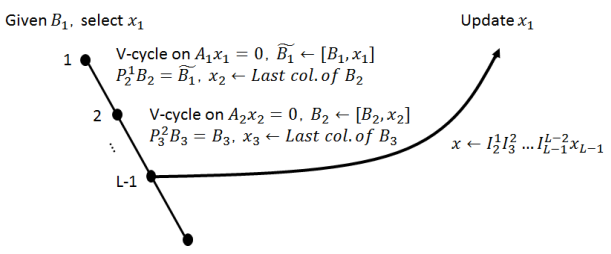

図 7　αSA 法の概略

1	$For\ level = 1\ to\ max\_level - 1$
2	$For\ n = 1\ to\ extract\_number$
3	$Random(\boldsymbol{x})$
4	$V\_cycle^{\mu}\ (A_{level}\boldsymbol{x} = 0)$
5	$B_{level} \leftarrow \boldsymbol{x}$
6	$Multilevel\_creation(B_{level})$
7	$End\ for$
8	$End\ for$
9	B をニアカーネルベクトルとして出力

- max_level：階層の最大レベル数
- $extract\_number$：抽出したい本数
- A_{level}：レベル$level$の問題行列A
- V_cycle^{μ}：V-cycle をμ回繰り返す
- B_{level}：レベル$level$におけるニアカーネルベクトルの候補群の行列

図 8　本研究でのニアカーネルベクトル抽出手法の概略

行っている．またμに関しては，V-cycle を繰り返す回数であり，この値により抽出したニアカーネルベクトルの性質が決まると考えられる．そのため，μにより性能が変化することが考えられるが，このことについては今後の課題とし，本研究ではμを 20 で固定し，計測を行っている．

5. 数値実験と評価

5.1 実験環境と使用した問題

本研究では，東京大学の Fujitsu PRIMEHPC FX10 スーパーコンピュータシステム（Oakleaf-FX）を使用し，数値実験を行った[7]．FX10 は，1 ノードに 1 個の SPARC64IXfx プロセッサ（16cores, 1.848GHz）と 32GB メモリ（85GB/sec.）を搭載している．また，FX10 では 6 次元メッシュ/トーラス構成（5GB/s/link，bidirectional）を採用している．

数値実験では，最大 512 コアを使用し，計測を行った．また並列化手法については，1 コアに 1 プロセス起動するフラット MPI を使用した．

また本研究では，3 次元弾性体の問題を使用した．この問題は，ある物体に対して，力を加えたとき，どれだけ物体が変形するかを解く問題となっている．またこの問題は 3 次元なため，行列に落とし込む際には 1 節点あたりの行列要素が 3×3 のブロック行列となる．実験 2 で用いた弾性体の問題設定を図 9 に示す．この弾性体の問題は図 9 のように，均質な立方体の物体に対して，ある一部分に力を加えたとき，どのように変形するかを解く問題となっている．また，弾性体の問題において硬さを表すヤング率を，すべての部分で 1 に設定している．ヤング率は値が大きければ物体が硬いことを表している．問題サイズについては，ウィークスケーリングで計測しており，1 プロセスあたり 15×15×15 となるように計測を行った．ウィークスケーリングは，1 プロセスが担当する問題サイズが一定になるように問題サイズを設定するようにしたものである．そのため，プロセス数が増加するほど全体の問題サイズが増加する．また，問題行列の各プロセスへの分割は，各軸方向へ問題を均等に分割し，それにより作成された小行列を各プロセスへ分配する単純な方法で分割を行った．

数値実験では AMGS ライブラリ[8]を使用している．AMGS ライブラリは，大規模な疎行列係数の線型方程式を AMG 法で解くライブラリである．解法部では，GPBiCG 法[9]を使用し，前処理として AMG 法を適用している．AMG 法における解法部の緩和法には，対称ガウス・ザイデル法を適用する．そして，解法部の V-cycle の各レベルで緩和法を 2 回適用する．ただし，領域境界では，依存関係を無視している．また節点数が 100 以下になったとき，最も粗いレベルとし，LU 分解を行い，解を求めている．また，反復の終了条件は相対残差が1.0×10^{-7}とし，反復回数が 500 回となったときに，収束しなかったとした．

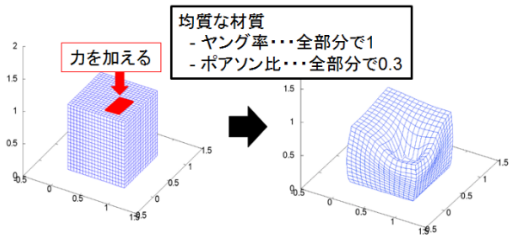

図 9　実験で使用した 3 次元弾性体問題の問題設定

AMGS ライブラリと解法部の GPBiCG 法は，Fortran を用い作成を行っている．また，コンパイラは Fujitsu Fortran コンパイラ，コンパイラオプションは高速化のための最適化オプションである”-Kfast”と，OpenMP を用いるためのオプションである”-Kopenmp”を使用した．

5.2 実験結果

本研究では，2 つの実験を行った．第 1 の実験では，粗いレベルでニアカーネルベクトルの抽出および設定を行うことによる反復回数と実行時間への影響の分析である．もうひとつは上記に加え，緩和法を変更させたときの反復回数と実行時間の比較である．以下，それぞれを実験 1 と実験 2 とする．

まず，実験 1 について述べる．実験 1 の比較対象を表 1 に示す．表 1 のように，この実験では粗いレベルで抽出を行うことによる効果をみるため，我々の研究[6]で従来用いていた粗いレベルで抽出を行わない手法と，今回の手法で

表 1　実験 1 での比較対象

比較対象	詳細
レベル 1 のみ	粗いレベルで抽出なし
粗いレベルで抽出:+1，+5	Level2 以降の階層毎に 1or5 本抽出・追加設定

表 2　第 1 レベルのニアカーネルベクトル抽出本数

表記	詳細
3p	平行移動成分 X,Y,Z (3 本)
6p	3p (3 本)＋回転成分 X,Y,Z (3 本)
3p+1, 3p+2, …	3p (3 本)＋第 1 レベルの抽出本数 (最大 7 本)

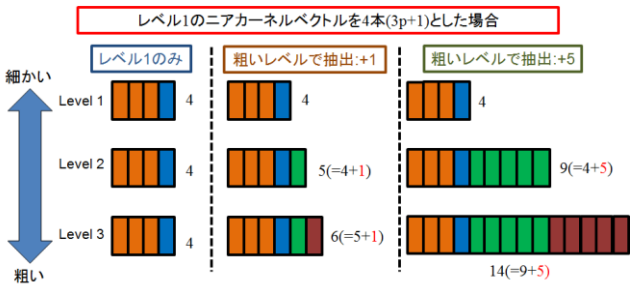

図 10　「粗いレベルで抽出」適用時の SA-AMG 法におけるレベル毎の設定本数

ある粗いレベルで抽出を行う手法で比較を行っている．また，本実験では第 1 レベルのニアカーネルベクトルの設定本数を変化させ，計測を行っている．この詳細を表 2 に示す．表 2 の 3p と 6p は弾性体問題の問題設定から予想されるニアカーネルベクトルである．また，表 2 からわかるように，第 1 レベルにて抽出したニアカーネルベクトルを用いる際には，3p に追加する形で設定を行っている．ここで，本実験の具体例を示した図を図 10 に示す．図 10 は本研究の手法において，第 1 レベルでニアカーネルベクトルを 4 本（3p+1）設定した際に，「粗いレベルで抽出」を適用した際の各レベルのニアカーネルベクトル設定本数を示した例となっている．この図のように，「レベル 1 のみ」では，細かいレベルを基に次のニアカーネルベクトルを生成するため，すべてのレベルで同じ本数となる．一方，「粗いレベルで抽出」では，「+1」の場合はレベル数が増加するごとに，ニアカーネルベクトルの設定本数を 1 本ずつ増やしていく．「+5」では，5 本ずつとなる．

実験 1 の結果を図 11 と図 12 に示す．図 11 と図 12 はそれぞれ 1 並列，512 並列時の，ニアカーネルベクトルの設定本数による反復回数および実行時間の変化を示したグラフとなっている．図 11 と図 12 は双方とも，横軸は第 1 レベルのニアカーネルベクトル設定本数を示している．これらのグラフでは，比較対象として 3p や 6p を設定した時の結果も記載している．また，上のグラフでは縦軸が反復回数を示し，下にいくほど収束までに必要とする反復回数

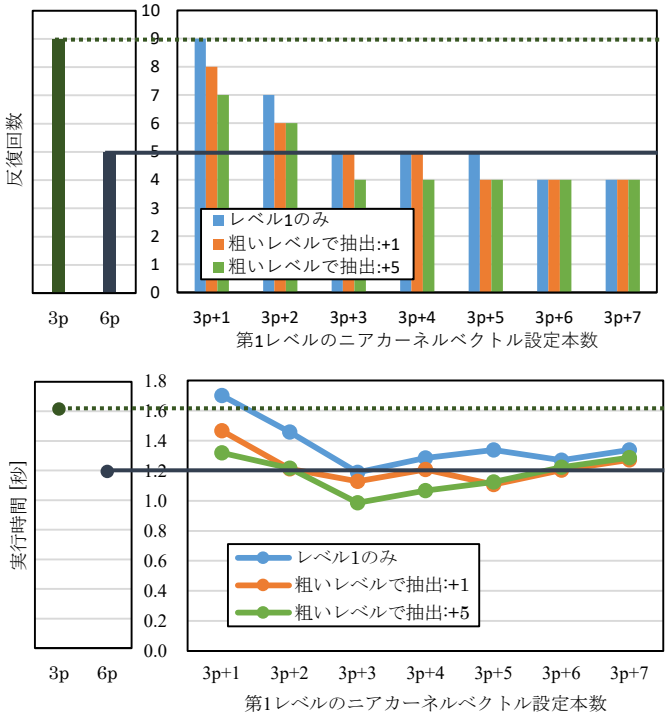

図 11　ニアカーネルベクトルの設定本数による反復回数（上）と実行時間（下）の変化（1 並列）

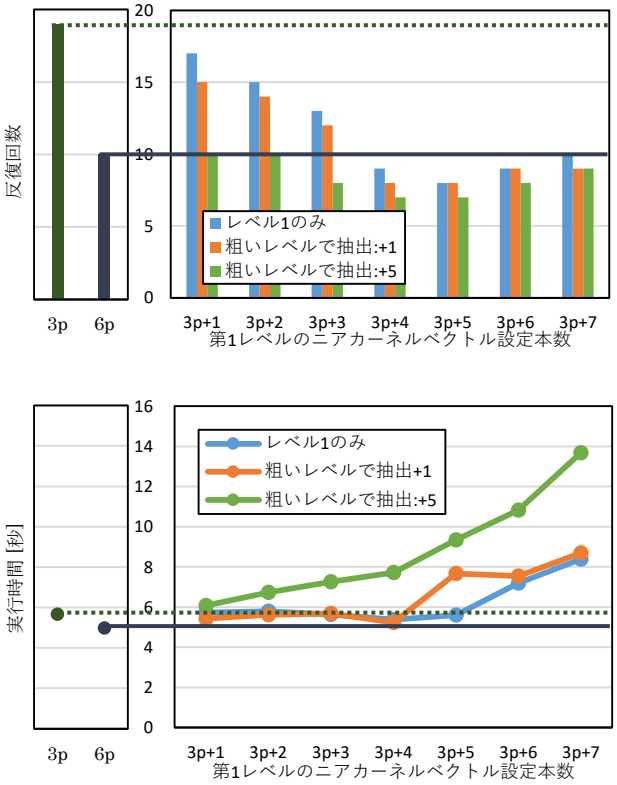

図 12　ニアカーネルベクトルの設定本数による反復回数（上）と実行時間（下）の変化（512 並列）

が少なく，収束性がよいことを示している．また，下のグラフは実行時間を示しており，下にいくほど実行時間が少なく，性能が良いことを示している．反復回数に着目すると，第 1 レベルおよび粗いレベルにおいてニアカーネルベクトルを適切な本数設定することで，反復回数を削減できることがわかる．また，実行時間に着目しても，1 並列においては粗いレベルで適切なニアカーネルベクトルを設定することで，実行時間が削減されることがわかる（「レベル 1 のみ」の最良値である 3p+3 の場合と，粗いレベルで抽出の最良値である「粗いレベルで抽出 : +5」の 3p+3 を比較すると，「粗いレベルで抽出」を用いることで約 20%削減されている）．これは，粗いレベルで追加的にニアカーネルベクトルを設定することで，粗いレベルの行列においても効率的にニアカーネルベクトル成分の減衰が行え，それが全体の反復回数や実行時間改善につながったものと考えられる．しかし，512 並列では，6p の実行時間よりも悪くなってしまっていることがわかる．これは，ニアカーネルベクトルの本数を増やすことで，粗いレベルの行列生成にかかる時間が増加してしまい，結果として収束性改善の効果以上に性能が悪化してしまったためであると考えられる．

ここで，プロセス数を変化させることによる反復回数の比較を図 13 に示す．このグラフでは横軸がプロセス数，縦軸が反復回数となっており，下にいくほど収束性が良いことを示す．また，グラフ内の要素である「従来最良」は，

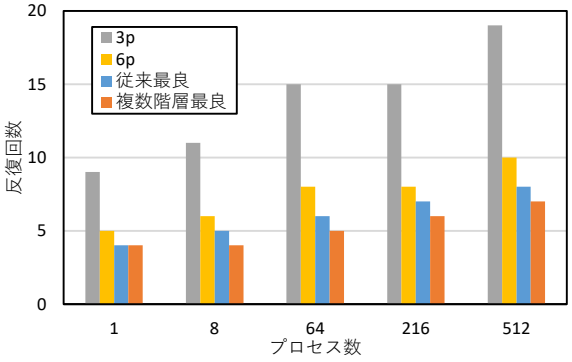

図 13　プロセス数による反復回数の変化

「レベル 1 のみ」において第 1 レベルのニアカーネルベクトル設定本数で最良のものを設定した時の値であり,「複数階層最良」は「粗いレベルで抽出」において，第 1 レベルに加え，粗いレベルでのニアカーネルベクトル設定本数で最良のものを設定した際の値となっている（例えば 512 プロセスでは,「従来最良」は「レベル 1 のみ」の 3p+5 の値を示し,「複数階層最良」では「粗いレベルで抽出:+5」の 3p+4 の値を示している）. 図 13 より，粗いレベルでニアカーネルベクトルを抽出することにより，どのプロセス数においても，問題設定から予想されるニアカーネルベクトルを設定したものや，レベル 1 のみ抽出したものを設定するよりも収束性の改善がみられることがわかる（複数階層最良を用いることで,「6p」と比べ 30%,「従来最良」と比べても約 14%の削減）. これより，粗いレベルでニアカーネルベクトルを適切な本数抽出および設定を行うことで，ニアカーネルベクトル成分を効率よく減衰させることができ，収束性が改善することがわかった.

次に，実験 2 の結果を図 14 に示す. 図 14 は緩和法の変更による反復回数および実行時間を示したグラフとなっている. このグラフの横軸はプロセス数，縦軸は反復回数となっており, 下にいくほど収束性が良いことを示している. この実験では，緩和法に実験 1 で用いた対称ガウス・ザイデル法（SGS）と，Fill-in なし ILU 分解と前進後退代入（ILU(0)）を用いた場合で比較を行っている. また，ILU(0)の前進後退代入では，並列化時においても収束性を安定させるため，加法シュワルツ法（Overlapped Additive Schwartz Domain Decomposition Method）[10]を，SGS の適用回数と同じ 2 回適用している. さらに，図 14 は図 13 の「複数階層最良」と同じく，第 1 レベルに加え，粗いレベルでのニアカーネルベクトル設定本数で最良のものを設定した際の結果を記載している. また, 抽出時の V-cycle に用いる緩和法と，SA-AMG 法適用時の緩和法は同じものを使用している. 図 14 より，反復回数に着目すると，実験 1 で用いていた SGS よりも，ILU(0)を用いることで、収束性の改善がみられることがわかる. しかし，実行時間に着目すると，SGS とあまり変化がないという結果となった. これは収束

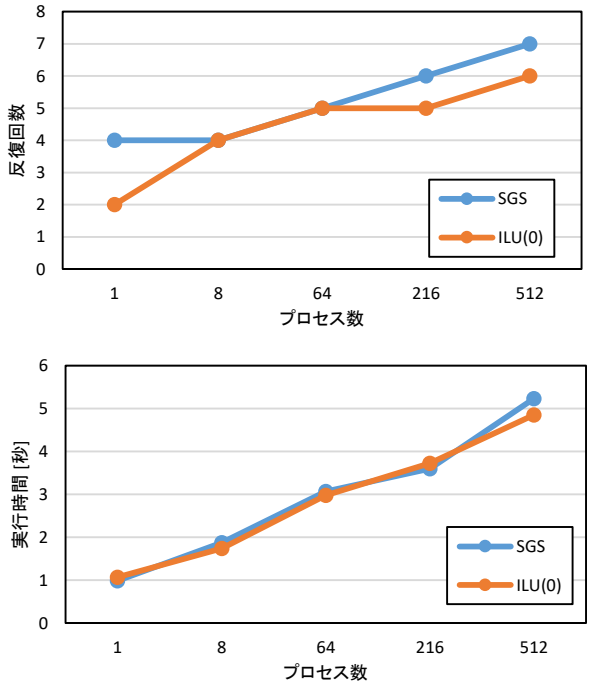

図 14　緩和法による反復回数（上）と実行時間（下）の比較（「複数階層最良」と同じ最良値の結果を使用）

性の改善による実行時間削減の効果よりも，ILU 分解のコストが大きくなってしまったためであると考えられる. これより，ニアカーネルベクトル抽出手法を用いた場合において，緩和法を適切に選択することで，さらに収束性の改善が行える可能性があることがわかった.

6. おわりに

本研究では，粗いレベルでニアカーネルベクトルを複数本抽出する手法を実装し, それらを SA-AMG 法に用いることによる反復回数や実行時間への影響の分析を行った. この研究では，2 つの実験を行った. 第 1 の実験は粗いレベルでニアカーネルベクトルの抽出および設定を行うことによる反復回数と実行時間への影響の分析である. もうひとつは上記に加え，緩和法を変更したときの反復回数と実行時間の比較である.

本実験では，3 次元弾性体の問題を使用しており，この問題では平行移動成分と回転成分がニアカーネルベクトルとして知られている. 第 1 の実験では，平行移動成分と回転成分を設定する場合と，我々の研究で従来用いていた，ニアカーネルベクトルをレベル 1 のみで抽出したものを設定した場合，そして本研究での提案手法である粗いレベルで抽出し，追加的に設定した場合の比較を行った. この実験により，粗いレベルでのニアカーネルベクトル抽出手法を用い, 抽出されたベクトルを適切な本数設定することで，平行・回転成分を使用する場合や，レベル 1 のみ抽出する手法と比べ，反復回数と実行時間双方で性能の改善がみられることがわかった. これより，粗いレベルにおいてもニ

アカーネルベクトル成分の減衰は重要であり，粗いレベルにおいてもニアカーネルベクトルを追加で設定することで，効率的に減衰させることができ，全体の収束性の改善につながることがわかった．しかし，ニアカーネルベクトルを多く用いることが必ずしも収束性の改善につながらないことや，ニアカーネルベクトルを複数本設定することによる計算コストの増加が，収束性改善の効果を上回ってしまい，結果として実行時間が悪化してしまう場合も見受けられた．第 2 の実験では，粗いレベルでのニアカーネルベクトル抽出手法を用いた場合において，第 1 の実験でも用いている対称ガウス・ザイデル法と，Fill-in なし ILU 分解と前進後退代入をそれぞれ V-cycle の緩和法として用いた場合の，反復回数や実行時間の比較を行った．この実験により，ILU 分解を用いることで，対称ガウス・ザイデル法よりも収束性の改善がみられることがわかった．これより，ニアカーネルベクトル抽出手法を用いる場合においても，緩和法の選択でさらなる収束性の改善が見込めることがわかった．

今後の課題としては，まず抽出したニアカーネルベクトルを多く用いることが，必ずしも性能改善につながらない原因を調査することが挙げられる．これに関しては，V-cycle の適用回数であるμの値（4 章の図 8 を参照）による収束性への影響の分析や，緩和法により収束性が変化することから，用いる緩和法の変更により，改善がみられるかを検証していくことを考えている．またその上で，本手法が広い範囲の問題に対して有効であるか調査していくことも考えている．本研究では 3 次元弾性体問題のみの適用であったが，この問題以外の悪条件で解きにくい問題に対して適用し，本手法が有用となるかを調査していくことを考えている．

謝辞 本研究の一部は，JSPS 科研費 15K15998 の助成を受けたものです．

参考文献

[1] Pereira, F. H., Verardi, S. L. L., and Nabeta, S. I.: “A fast algebraic multigrid preconditioned conjugate gradient solver”, Applied Mathematics and Computation 179, pp.344-351, 2006.
[2] Vanek, P., Brezina, M. and Mandel, J.: “Convergence of Algebraic Multigrid Based on Smoothed Aggregation”, Numerische Mathematik 2001, vol 88, pp.559-579, 2001.
[3] Vanek, P., Mandel, J. and Brezina, M.: “Algebraic Multigrid by Smoothed Aggregation for Second and Fourth Order Elliptic Problems”, Computing, Vol.56, pp.179-196, 1998.
[4] Chan, T. F. and Vanek, P.: “Multilevel algebraic Elliptic Solvers”, UCLA Math, Dept. CAM Report, 1999.
[5] Brezina, M., Falgout, R., Maclachlan, S., Manteuffel, T., Mccormick, S. and Ruge, J.: “Adaptive Smoothed Aggregation (α SA)”, SIAM J. Sci. Comput, Vol. 25, No.6, pp.1896-1920 (2004).
[6] Nomura, N., Fujii, A., Tanaka T., Nakajima, K. and Marques, O.: “Performance Analysis of SA-AMG Method by Setting Extracted Near-kernel Vectors”, VECPAR2016 (2016).
[7] Information Technology Center: The University of Tokyo, http://www.cc.u-tokyo.ac.jp/.
[8] AMGS Library: http://hpcl.info.kogakuin.ac.jp/lab/software/amgs
[9] Zhang, S.-L.: “GPBi-CG: Generalized Product-type Methods Based on Bi-CG for Solving Nonsymmetric Linear Systems” SIAM J. Sci. Comput, Vol. 18, No.2, pp.537-551 (1997).
[10] Smith, B., Bjorstad, P., and Gropp, W.: “Domain decomposition: parallel multilevel methods for elliptic partial differential equations”, Cambridge university press (2004).