Vol.2021-HPC-178 No.3
2021/3/15

時間発展Stokes方程式に対する粗格子集約を用いたMultigrid Reduction in Timeの適用

依田 凌[1,a)]　中島 研吾[1]　Matthias Bolten[2]　藤井 昭宏[3]

概要：時間発展偏微分方程式に対する時間並列解法 (parallel-in-time) は，近年の大型計算機の超並列構成により注目されている．その有力解法のひとつに，時間領域に Multigrid Reduction 法を適用して時間並列性を抽出する Multigrid Reduction in Time (MGRIT) がある．しかし一般の Multigrid 法と同様に，超並列環境においては最も粗いレベルの通信コストが増加し，スケーラビリティの悪化が問題となる．そこで本研究では，(1) 粗いレベルで並列度を段階的に落とす粗格子集約 (coarse-grid agglomeration)，(2) 最も粗いレベルにおける緩和法の利用，(3) Krylov 部分空間法への前処理としての利用，の 3 つの手法を検討し，MGRIT のスケーラビリティ向上を図る．数値実験では時間発展 Stokes 方程式を対象とし，3 つの手法を全て組み合わせたソルバが，従来手法に対して約 10.52 倍の高速化を達成した例を示す．

1. はじめに

本研究では時間発展偏微分方程式に対する時間並列解法 (parallel-in-time) を考える．時間方向から並列性を抽出するこの試みは 50 年以上の長い歴史を持ち，Gander による包括的なサーベイ論文がある [1]．また近年の大型計算機が膨大な演算コアにより高性能化を図る超並列構成であることから，超並列環境を効率的に利用するための新たな並列化可能性として時間並列解法は注目を集めている．Ong と Schroder は種々の時間並列解法を実行時間短縮と並列化効率の観点からまとめており [2]，有力解法のひとつとして Multigrid 法に基づく解法を挙げている．具体的には Horton と Vandewalle による Space-Time Multigrid (STMG) [3]，Minion と Emmett による Parallel Full Approximation Scheme in Space and Time (PFASST) [4]，Falgout 等による Multigrid Reduction in Time (MGRIT) [5] である．本研究では MGRIT に焦点を当てる．MGRIT は時間方向に Multigrid Reduction 法 [6] を適用して時間方向から並列性を抽出する手法であり，様々な応用問題でその有効性が確認されている．例として圧縮性 Navier-Stokes 方程式 [7]，eddy-current 問題 [8]，線形弾性方程式 [9]，inviscid Burgers 方程式 [10] への適用が挙げられる．MGRIT は通常の Multigrid 法と同様に最も粗いレベルにおいて直接解法が用いられることが多く，この演算は各プロセスに分散されたデータをマスタープロセスへ集約する必要がある．しかしながら超並列環境で想定される数千から数万プロセスからマスタープロセスへの集約は，通信コスト増加によるスケーラビリティの低下につながる恐れがある．

本研究では MGRIT のスケーラビリティを高めるために，以下の 3 つの手法を用いる．

1 つめは粗格子集約 (coarse-grid agglomeration) [11] である．この手法は，粗いレベルにおいて並列度を段階的に落とし，最も粗いレベルにおける通信コストの削減を図る．Geometric Multigrid 法では中島 [12] が，Algebraic Multigrid 法では Adams 等 [11] や野村等 [13] が知られている．また時間並列に対しては，Speck らが PFASST と空間方向への並列 Multigrid 法の組み合わせを検討し，時空間並列化を最適化における粗格子集約の可能性を示唆している [14]．本研究では MGRIT へ粗格子集約を適用し，時間方向に粗いレベルの効率的な集約を図る．

2 つめは最も粗いレベルにおける緩和法の利用である．前述の通りこのレベルでは直接解法が用いられていることが多い．しかし粗いレベルを再帰的に構築することが困難な場合は，粗いレベルの自由度が増え，直接解法のコストは増加する．この状況は時間並列解法においても対象問題が持つ時間刻み幅の制約などから現実的に考えられる．そこでこの手法は直接解法の代わりに緩和法を用いる．つまり最も粗いレベルにおける逐次処理を排除して，全処理を並列実行可能にする．直接解法を用いた場合と比較してソルバの収束性は悪化するものの，超並列環境を十分に活用

[1] 東京大学 The University of Tokyo
[2] University of Wuppertal
[3] 工学院大学 Kogakuin University
[a)] ryo_yoda@mist.i.u-tokyo.ac.jp.

できる利点を持つ．

3 つめは Krylov 部分空間法への前処理としての利用である．対象問題が線形の場合，時間並列解法は時空間線形方程式を構築する．これを巨大な 1 つの線形方程式とみなすことで Krylov 部分空間法を適用できる．この手法は，McDonald 等の all-at-once approach と呼ばれる手法 [15] の発想と同様である．しかし [15] ではブロック巡回行列に基づく前処理を提案し評価している．本研究では Multigrid 法に基づく時間並列解法 MGRIT を Krylov 部分空間法の前処理として利用することが相違点であり，著者等が提案した手法でもある [16]．

本研究では上述した 3 つの手法を MGRIT に適用しその有効性を検証する．最も粗いレベルにおける緩和法として FCF-relaxation を用いて，Krylov 部分空間法として Generalized Minimal Residual (GMRES) 法を用いる．数値実験では時間発展 Stokes 方程式を対象する．この問題は離散化により鞍点系 (saddle-point systems) が得られる．鞍点系に対する数値解法は多数存在し，Benzi 等によるサーベイが詳しい [17]．Multigrid 法に関するだけでも，強力な緩和法が多数知られており [18][19][20]，近年でも local Fourier analysis による収束性解析が盛んに行われている [21][22]．しかし本研究では時間並列化の影響にのみ焦点を絞り検証を行う．そのため鞍点系に対する上記の並列化手法は利用せずに，その求解には直接解法を用いている．空間並列化と組み合わせた評価は今後の課題である．

以下の章は次のように構成される．2 章では MGRIT のアルゴリズムについて述べる．3 章では本研究で MGRIT へ適用する 3 つの手法を紹介する．4 章では時間発展 Stokes 方程式に対する数値実験を示す．最後に 5 章では結論と今後の課題を述べる．

2. Multigrid Reduction in Time (MGRIT)

はじめに，線形の時間発展方程式に対する時間逐次解法 (Time-marching method) を考える．これは各タイムステップ毎に逐次的に解を求める手法を指す．ここで支配方程式にある空間離散化と陰解法による時間離散化を適用した場合に注目する．空間方向の自由度を N_x，タイムステップ数を N_t と置く．この時，未知数ベクトル $\boldsymbol{u}^i \in \mathbb{R}^{N_x}$，外部流入ベクトル $\boldsymbol{g}^i \in \mathbb{R}^{N_x}$，離散化に基づく係数行列 $\Phi \in \mathbb{R}^{N_x \times N_x}$ を用いて，時間発展の関係を式 (1) と書ける．なお右肩文字 i は各タイムステップを表す．

$$\begin{cases} u^0 &= g^0 \\ \Phi u^{i+1} &= u^i + g^{i+1} \quad (i = 0, 1, \cdots, N_t - 1) \end{cases} \tag{1}$$

つまり時間逐次解法は式 (1) を逐次的に N_t 回解く手法である．この手法は各タイムステップにおいて，N_x サイズの線形方程式を解く際に空間方向の並列性を持つ一方で，時間方向の並列性は持っていない．

Falgout 等は時間領域に Multigrid Reduction 法 [6] を適用して時間方向から並列性を抽出する Multigrid Reduction in Time (MGRIT) を提案した [5]．この手法は，時間方向に N_t 個のグリッド点を明示的に構築し，全タイムステップの未知数ベクトル u^i をまとめて 1 本の未知数ベクトル $\boldsymbol{u} \in \mathbb{R}^{N_t \times N_s}$ とみなす．そのため反復の対象となる方程式は，式 (2) の巨大な時空間線形システム $A\boldsymbol{u} = \boldsymbol{g}$ であり，N_x サイズのブロック行列で構成される．

$$A\boldsymbol{u} = \begin{bmatrix} I & & & \\ -I & \Phi & & \\ & \ddots & \ddots & \\ & & -I & \Phi \end{bmatrix} \begin{bmatrix} u^0 \\ u^1 \\ \vdots \\ u^{N_t-1} \end{bmatrix} = \begin{bmatrix} g^0 \\ g^1 \\ \vdots \\ g^{N_t-1} \end{bmatrix} = \boldsymbol{g} \tag{2}$$

次に式 (2) に対して時間方向に並列に動作する緩和法を考える．MGRIT は粗格子率 m を導入して，各タイムステップに C 点または F 点とラベル付けする．タイムステップ i $(0 \leq i \leq N_t - 1)$ が，$t \equiv 0 \pmod m$ の場合は C 点に，$t \not\equiv 0 \pmod m$ の場合は F 点に対応する．この F 点，または C 点において時間逐次法を適用する緩和法をそれぞれ F-relaxation と C-relaxation と呼ぶ．**図 1** に $m = 4$ における緩和法の模式図を示す．なお赤の頂点が C 点，黒の頂点が F 点を指す．図 1 より，C 点から始まる m タイムステップを 1 区間とした時，区間の境目で時間方向の依存性が切られていることが確認できる．そのためどちらの緩和法も区間単位で時間方向に並列に動作できる．また F-relaxation，C-relaxation，F-relaxation の順番に適用する緩和法を FCF-relaxation と呼ぶ．

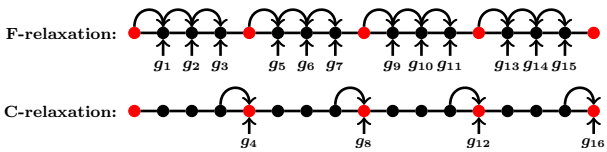


図 1 Sketch of each relaxation with $m = 4$. The red-points indicate the C-points. The black-points denotes the F-points.

次に粗格子補正について考える．粗格子率 m を用いた粗格子化を時間方向に適用するため，粗いレベルにおいて，タイムステップ数が N_t/m である誤差方程式 $A_c\boldsymbol{e}_c = \boldsymbol{r_c} \in \mathbb{R}^{(N_t/m) \times N_x}$ が構築される (式 (3))．A_c 内のブロック行列 $\Phi_c \in \mathbb{R}^{N_x \times N_x}$ は通常，元々の支配方程式を再離散化して構築され，m 倍拡大した時間刻み幅が用いられる．

$$A_c\boldsymbol{e}_c = \begin{bmatrix} I & & & \\ -I & \Phi_c & & \\ & \ddots & \ddots & \\ & & -I & \Phi_c \end{bmatrix} \begin{bmatrix} e_c^0 \\ e_c^1 \\ \vdots \\ e_c^{\frac{N_t}{m}-1} \end{bmatrix} = \begin{bmatrix} r_c^0 \\ r_c^1 \\ \vdots \\ r_c^{\frac{N_t}{m}-1} \end{bmatrix} = \boldsymbol{r}_c \tag{3}$$

さらに細かいレベルと粗いレベルを移動する制限演算子 R と補間演算子 P を定義する．制限演算子 R は細かいレベルの全ての C 点を粗いレベルへ制限し，補間演算子 P は粗いレベルの全ての点を細かいレベルの C 点へ補間する．よって細かいレベルと C 点の個数は，粗いレベルの点の個数と一致する．

以上より MGRIT を導出できる．2 レベルにおけるアルゴリズムを Algo. 1 に示す．1 行目は pre-smoothing の FCF-relaxation を適用している．2 行目では残差を計算し粗いレベルへ制限する．3 行目より制限された残差を打ち消す補正量を求め，4 行目で細かいレベルへ補正する．最後に 5 行目で post-smoothing の F-relaxation を適用する．3 行目の粗格子ソルバには直接解法として時間逐次法が用いられる．本研究ではこの箇所に緩和法である FCF-relaxation を用いる手法を検討するが詳細は 3 章で述べる．また Algo. 1 を再帰的に適用することでマルチレベルを構成できる．

Algorithm 1 Two-level Multigrid Reduction in Time (MGRIT) (A, $\boldsymbol{g}$, $\boldsymbol{u}$, m)

1: Apply FCF-relaxation to $A\boldsymbol{u} = \boldsymbol{g}$.
2: Restrict the residual to the coarse grid from the fine-grid. $\boldsymbol{r}_c = R(\boldsymbol{g} - A\boldsymbol{u})$.
3: Solve $A_c\boldsymbol{e}_c = \boldsymbol{r}_c$.
4: Prolongate the error to the fine-grid from the coarse grid. $\boldsymbol{u} = \boldsymbol{u} + P\boldsymbol{e}_c$.
5: Apply F-relaxation to $A\boldsymbol{u} = \boldsymbol{g}$.

3. MGRIT のスケーラビリティ改善手法

本章では，本研究で利用する，MGRIT のスケーラビリティ向上を図る 3 つの手法について述べる．

3.1 粗格子集約 (Coarse-grid Agglomeration)

本節では Multigrid 法の並列実装について考える．一般的な Multigrid 法の実装は，最も粗いレベルで全てのプロセスからマスタープロセスへ未知数を集約して，直接解法を適用することが多い．しかしこの実装は，超並列環境においては数千〜数万程度のプロセスから 1 つのプロセスへ集約する必要があり，この際の通信コスト増加は無視できなくなる．この問題に対し，Adams 等は粗格子集約 (coarse-grid agglomeration) の必要性が強調している [11]．この手法は，最も粗いレベルでのみ集約を行うのではなく，その前の粗いレベルから段階的に集約を行う．つまりあるレベルからレベル数を増やす毎に並列度は減少する．よって最も粗いレベルにおける集約では，そのレベルの active プロセスのみが対象となるため，通信コストの増加が抑えられる．**図 2** に，3 レベル 8 プロセス (P0, ..., P7) における粗格子集約の例を示す．青の矢印は集約を表す．例えばレベル 0 からレベル 1 にかけて，P1 は P0 へマージされる．そのためレベル 1 における P1 は，active プロセスから inactive プロセスになり，その状態を赤のバツ印で表している．また一度 inactive となったプロセスは，以降のレベルにおいても同様に inactive となる．粗格子集約は各レベルで異なる acitve プロセス数を持つ階層構造を与える．

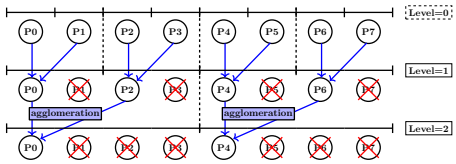


図 2 3 レベル 8 プロセス (P0, ..., P7) における粗格子集約の例．青の矢印は集約を，赤のバツ印は inactive プロセスを表す．

次に，MGRIT に対する粗格子集約を考える．時間方向に割り当てられた各プロセスは，時間領域を分割し，分散されたタイムステップ数を持つ．本研究では，各プロセスが持つタイムステップ数が粗格子率以下になった場合に集約を行う，という基準を採用している．なぜなら MGRIT の緩和法は粗格子率単位で並列性を持っており，これ以上細かい単位で分散しても逐次処理と同じになるためである．ここでタイムステップ数はプロセス数で割り切れるという仮定を置き，時間方向には構造格子であることを踏まえれば，どのレベルにおいて，どのプロセスがどのプロセスとマージされるべきか，は簡単に Algo. 2 で求められる．ここではタイムステップ数 N_t と粗格子率 m，レベル数 L，

Algorithm 2 How to determine the process number to be merged (N_t, m, L, P_t, `rank_id`)

```
1: dist_Nt[0] = N_t / P_t   ▷ Distributed number of time-steps
2: p_width[0] = 1           ▷ Width between active processes
3: for lvl = 1 to L − 1 do
4:     dist_Nt[lvl] = dist_Nt[lvl-1] / m      ▷ Coarsening
5:     p_width[lvl] = 1
6:     if dist_Nt[lvl] < m then
7:         if dist_Nt[lvl] == 0 then
8:             continue              ▷ Process rank_id is inactive.
9:         end if
10:        p_width[lvl] = m/dist_Nt[lvl]*p_width[lvl-1]
11:        merged_id = rank_id - (rank_id%p_width[lvl])
12:    end if
13: end for
```

時間並列度 P_t，プロセス番号`rank_id`を仮定する．各レベルでプロセスに割り当てられるタイムステップ数を表す`dist_Nt`と，active プロセス間の幅を表す`p_width`を導入する．最終的に`rank_id`は，11 行目で求められる`merged_id`へマージされる．そのため Algebraic Multigrid 法における粗格子集約のような複雑な処理は必要ない [11][13]．しかしながら，粗いレベルの active プロセスには，自身の細かいレベルとは異なる時間領域が割り当てられることに注

意が必要である．図 3 に具体例を示す．各色は各プロセスに割り当てる分散された時間領域を指す．プロセス 0 に対応する青色の領域に注目すれば，各レベルにおいて異なる時間領域を持つことが確認できる．この時，対象とする問題の係数行列が時間に対して不変の場合，全てのタイムステップにおいて同じ係数行列を利用できる．しかし逆に係数行列が時間に依存して変化する場合，細かいレベルで利用しなかった係数行列も構築して利用する必要がある．本研究の数値実験では時間に対して不変の係数行列を想定しているため，粗いレベルの右辺，つまり残差ベクトルのみを集約している．

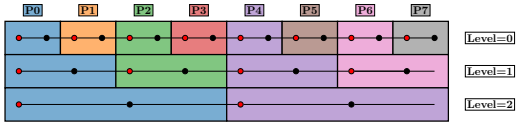


図 3　3 レベル 8 プロセス (P0, . . ., P7) かつ粗格子率 $m = 2$ における MGRIT に対する粗格子集約．各色は各プロセスが持つ分散された時間領域を示す．粗いレベルの active プロセスは細かいレベルの自身と異なる時間領域を持つ．

3.2　最も粗いレベルにおける緩和法の利用

3.1 節で述べた粗格子集約手法を用いることで，超並列環境においても，粗いレベルの効率的な処理が期待されるとはいえ，これとは別に最も粗いレベルにおける直接解法の計算コストを考える必要がある．MGRIT の直接解法とは，最も粗いレベルのタイムステップ数回，時間逐次解法を適用することであるため，タイムステップ数が多いほど計算コストは高くなる．このコストが問題となる状況は，対象とする問題が持つ時間刻み幅の制限により，時間方向にマルチレベルを構築することが困難な場合に考えられる．

この手法は最も粗いレベルにおいて直接解法の代わりに緩和法を利用する．具体的には FCF-relaxation を 1 回適用する．ここで FCF-relaxation の適用回数もパラメータとして考えられるが，複数回適用するほど直接解法の処理に近づくため，本研究では 1 回適用のみを検討する．この手法は時間方向に逐次的な処理である直接解法を排除するため収束性は悪化するものの，全ての演算を時間方向に並列に実行できる．よって高い並列性を十分に活用できる利点を持つ．本研究ではこの手法を Multilevel-FCF と呼ぶ．本来 MGRIT の最も粗いレベルのソルバは規定されていないため，この手法も広義に MGRIT に含まれる．しかし本研究では明確に区別するためこの名称を利用する．Algo. 3 に Multilevel-FCF のアルゴリズムを示す．ここで下付き文字 l はレベル数を表す．通常の MGRIT と異なる箇所は 2 行目のみであり，最も粗いレベルにおいても緩和法である FCF-relaxation が適用されている．

Algorithm 3 Multilevel-FCF $(A, \boldsymbol{g}, \boldsymbol{u}, m)$

1: **if** l is the corasest level, L **then**
2:　　Apply FCF-relaxation to $A_L \boldsymbol{e}_L = \boldsymbol{r}_L$.　▷ Approximate solves instead of the direct method
3: **else**
4:　　Apply FCF-relaxation to $A_l \boldsymbol{u}_l = \boldsymbol{g}_l$.
5:　　Restrict the residual to the coarse-grid from the fine-grid. $\boldsymbol{r}_{l+1} = R_l(\boldsymbol{g}_l - A_l \boldsymbol{u}_l)$.
6:　　Solve on next level: Multilevel-FCF$(A_{l+1}, \boldsymbol{r}_{l+1}, \boldsymbol{e}_{l+1}, m)$.
7:　　Prolongate the error to the fine-grid from the coarse-grid. $\boldsymbol{u}_l = \boldsymbol{u}_l + P_l \boldsymbol{e}_{l+1}$.
8:　　Apply F-relaxation to $A_l \boldsymbol{u}_l = \boldsymbol{g}_l$.
9: **end if**

3.3　Krylov 部分空間法への前処理

3.2 節の Multilevel-FCF では収束性の代わりに並列性を優先した．しかし可能であるならば，収束性悪化による極端な反復回数の増加は避けるべきである．そこで本節では，高い並列性を維持した上で収束性の向上を図るために，Krylov 部分空間法との併用を考える．すなわち式 (2) を 1 つの線形システムとみなして Krylov 部分空間法を適用し，MGRIT や Multilevel-FCF をその前処理として利用する．ソルバ単体で利用した場合と比較して，Krylov 部分空間法による収束性の加速が期待される．

式 (2) の係数行列は非対称であるため，本研究では Krylov 部分空間法として Generalized Minimal Residual (GMRES) 法 [23] を用いる．GMRES は残差ベクトルの L2 ノルムを最小化する多項式を選択し，残差単調減少性を持つ．GMRES は右前処理が可能であり，Algo. 4 に右前処理付き GMRES のアルゴリズムを示す．3 行目が前処理部分に対応しており，ここで MGRIT や Multileve-FCF を適用する．すなわち，初期値ベクトルを $\boldsymbol{z}_k = \boldsymbol{0}$ として，Algo.1 の MGRIT$(A, \boldsymbol{v}_k, \boldsymbol{z}_k, m)$ や Algo.3 の Multilevel-FCF$(A, \boldsymbol{v}_k, \boldsymbol{z}_k, m)$ を呼ぶ．13 行目の最小化問題では，Hessenberg 行列 $\tilde{H}$ 行に対して Givens 回転を適用し上三角化して解いている．

Algorithm 4 Preconditioned GMRES $(A, \boldsymbol{g}, \boldsymbol{u}_0, M, k)$

1: $\boldsymbol{r}_0 = \boldsymbol{g} - A\boldsymbol{u}_0$, $\beta = ||\boldsymbol{r}_0||_2$, and $\boldsymbol{v}_1 = \boldsymbol{r}_0/\beta$
2: **for** $k = 1, \cdots,$ maxiter **do**
3:　　$\boldsymbol{z}_k = M^{-1}\boldsymbol{v}_k$　▷ Preconditioning part.
4:　　$\boldsymbol{\omega} = A\boldsymbol{z}_k$　▷ SpMV for the space-time matrix.
5:　　**for** $j = 1, \cdots, n$ **do**　▷ Gram-Schmidt process.
6:　　$h_{j,k} = (\boldsymbol{\omega}, \boldsymbol{v}_j)$
7:　　$\boldsymbol{\omega} = \boldsymbol{\omega} - h_{j,k}\boldsymbol{v}_j$
8:　　**end for**
9:　　$h_{k+1,k} = ||\boldsymbol{\omega}||_2$
10:　　$\boldsymbol{v}_{k+1} = \boldsymbol{\omega}/h_{k+1,k}$　▷ Normalization.
11: **end for**
12: $Z_k = [\boldsymbol{z}_1, \cdots, \boldsymbol{z}_k]$, $\tilde{H}_k = \{h_{i,j}\}_{1 \le i \le k+1, 1 \le j \le k}$, $\boldsymbol{e}_1 = (1, 0, \cdots, 0)^*$.
13: $\boldsymbol{u}_k = \boldsymbol{u}_0 + Z_k \boldsymbol{y}$ $\left(\boldsymbol{y} = \mathrm{argmin}||\beta \boldsymbol{e}_1 - \tilde{H}_k \boldsymbol{y}_k||\right)$

最後に MGRIT や Multilevel-FCF と GMRES を併用した場合の並列性と追加演算コストについて考える．GMRES の主要な計算は，疎行列ベクトル積 (sparse matrix-vector multiplication, SpMV) である．この SpMV は式 (2) の行単位で並列性を持っており，MGRIT や Multilevel-FCF の緩和法より高い並列性を持っている．そのため Krylov 部分空間法の併用によって，ソルバの並列性は低下しない．さらにその計算量は，陰解法による離散化の場合，MGRIT や Multilevel-FCF の緩和法と比較すると比較的小さい．なぜなら各緩和法は空間サイズの線形方程式を解く必要があるのに対し，式 (2) に対する SpMV は空間サイズの行列による数回の SpMV に対応するためである．

4. 数値実験

4.1 時間発展 Stokes 方程式

2 次元時間発展 Stokes 方程式は未知数 (u, v, p) を持つ．それぞれ u は x 方向の速度，v は y 方向の速度，p は圧力を表す．支配方程式は式 (4) から式 (6) で与えられる．

$$\partial_t u - \Delta u + \partial_x p = f_x \tag{4}$$
$$\partial_t v - \Delta v + \partial_x p = f_y \tag{5}$$
$$-\partial_x u - \partial_y v = 0 \tag{6}$$

ここで Δ はラプラシアンを指し，$\partial_t := \partial/\partial t$，$\partial_x := \partial/\partial x$，$\partial_y := \partial/\partial y$ である．式 (4) と式 (5) は運動方程式を表し，式 (6) は連続の式に対応する．この支配方程式は，流体速度が非常に遅く，粘性が非常に大きい現象を記述している．離散化にはスタガード格子 (staggered grid) と呼ばれる，異なる箇所に未知数 (u, v, p) を配置する格子を用いる．二次元領域 $\Omega = [0, 1]^2$ における具体例を**図 4** に示す．圧力 p の離散点は各セルの中心に配置される．また垂直方向の辺の中点に x 方向の速度 u が，水平方向の辺の中点に y 方向の速度 v が配置される．本実験では全ての境界で Dirichlet

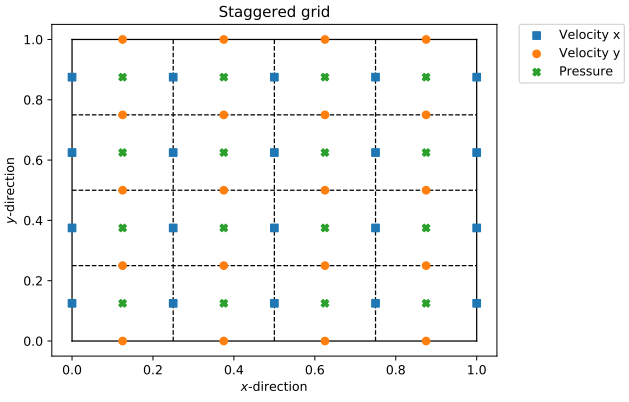


図 4 二次元領域 $\Omega = [0, 1]^2$ におけるスタガード格子．青点と橙点，緑点はそれぞれ u, v, p に対応する．

境界条件を仮定する．そのためこの離散化により線形の鞍点系が構築されるものの，その係数行列はフルランクで一意な解を持つことが保証される．

本数値実験では，支配方程式の 1 つの解析解に基づく右辺ベクトル f_1 と f_2 を用いる．それは sin タイプの解析解で次式で与えられる．

$$u = +\sin(2\pi x)\sin(2\pi y)\cos(t) \tag{7}$$
$$v = +\cos(2\pi x)\cos(2\pi y)\cos(t) \tag{8}$$
$$p = -\cos(2\pi x)\sin(2\pi y)1/(t+1) \tag{9}$$
$$\begin{aligned} f_x = &-\sin(t)\sin(2\pi x)\sin(2\pi y) \\ &+8\pi^2 \sin(2\pi x)\sin(2\pi y)\cos(t) \\ &+\frac{2\pi \sin(2\pi x)\sin(2\pi y)}{t+1} \end{aligned} \tag{10}$$
$$\begin{aligned} f_y = &-\sin(t)\cos(2\pi x)\cos(2\pi y) \\ &+8\pi^2 \cos(t)\cos(2\pi x)\cos(2\pi y) \\ &-\frac{2\pi \cos(2\pi x)\cos(2\pi y)}{t+1} \end{aligned} \tag{11}$$

図 5 に式 (7) の u，式 (8) の v，式 (9) の p の関数の概形を示す．タイムステップ 0 における初期推定値または Dirichlet 境界条件における境界値は，解析解の関数値を用いて定められる．また他のタイムステップの初期推定値にはゼロベクトル $\mathbf{0}$ を用いている．

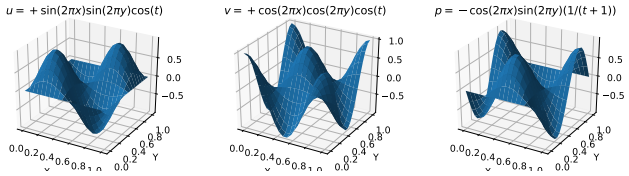


図 5 Sin-type solutions for time-dependent Stokes problems at initial time-step $t = 0$.

4.2 比較対象とするソルバと実行環境の設定

本研究で検討する 3 つの手法の性能評価のために，従来手法も含めた次の 6 つのソルバを比較対象とする．本節以降も同じソルバの表記を用いることに注意されたい．

(1) Time-marching method: 従来手法．時間逐次解法
(2) simple MGRIT: 従来手法．粗格子集約無し MGRIT
(3) MGRIT(AGG)
(4) MGRIT(AGG)-GMRES
(5) Multilevel-FCF(AGG)
(6) Multilevel-FCF(AGG)-GMRES

従来手法には時間逐次解法と，粗格子集約機能を持たない MGRIT を用いており，後者は「simple MGRIT」と表記する．また「(AGG)」はそのソルバが粗格子集約を持つことを表し，「-GMRES」は前処理付き GMRES を表す．

計測環境には計算ノードに Intel Xeon Platinum 8280 (CascadeLake) を 2 ソケット搭載している Oakbridge-CX (OBCX) を利用し，1 コア 1 プロセスの Flat MPI モード

で実行した．またバージョン 2019.5.281 の Intel コンパイラを用いた．各反復ソルバにおける収束判定には相対残差の 2 ノルムが 10^{-12} 以下を用いた．またどのソルバも各タイムステップにおいて空間サイズの線形方程式を解く必要がある．本研究では時間並列化の影響に焦点を絞るため空間並列化は行わずに LU 分解を利用した．それは求解コストが全てのタイムステップで同じ計算量であり，時間並列化の分析が容易になるためである．また本数値実験では空間サイズの係数行列は時間に対して不変で一度分解した行列は再利用できるため，分解コストは計測時間に含めずに前進・後退代入部のみを計測した．

4.3 粗格子集約の強スケーリング性能

粗格子集約手法の強スケーリング性能を検証するために，MGRIT(AGG) の実行時間に注目する．比較対象は時間逐次解法と simple MGRIT とする．空間自由度 $N_x^2 = 13^2$ とタイムステップ数 $N_t = 65,536$，粗格子率 $m = 4$ を固定し，時間並列度 P_t を変化させた場合の実行時間を**図 6** に示す．

図 6 では時間逐次解法は時間方向に並列性を持たないため一定の値がプロットされている．この問題設定では simple MGRIT は 5 レベルを構築すると，最終的に各プロセスに 1 タイムステップしか割り当てられないため，効率的に実行できる最大のレベル数と並列数は $L = 5$ と $P_t = 256$ に制限される．そのため simple MGRIT は $P_t = 256$ において，時間逐次解法と比較して約 3 倍の実行時間削減を達成したものの，最も粗いレベルのコストが増加しており，これ以上の高速化は難しい．これに対し MGRIT(AGG) はより多いレベル数とより高い並列数で動作できる．そのためプロセス数の増加に伴い実行時間を削減し，最終的に $L = 7$ かつ $P_t = 4096$ において，時間逐次解法と比較して約 25 倍の実行時間削減を達成した．図 6 より粗格子集約の適用による MGRIT のスケーラビリティ改善を確認した．

4.4 Multilevel-FCF の収束性への影響

最も粗いレベルにおいて緩和法の利用したソルバの収束性を評価するため，Multlevel-FCF(AGG) と MGRIT(AGG) の反復回数に注目する．空間自由度 $N_x^2 = 13^2$ とタイムステップ数 $N_t = 14,336$，さらに時間並列度 $P_t = 896$ も固定し，粗格子率 m とレベル数 L を変化させた場合の反復回数と実行時間を**表 1** に示す．

表 1 より，レベル数 L の増加，あるいは粗格子率 m の増加に伴い，Multilevel-FCF(AGG) の反復回数は，MGRIT(AGG) の反復回数と同等になることを確認した．前者の場合，レベル数 L の増加に伴い，最も粗いレベルのタイムステップ数 $N_{tc} = N_t/m^{L-1}$ は減少している (表 1 の列 “N_{tc}” を参照)．そのため最も粗いレベルにおける緩和法はそのレベルの直接解法を良く近似し，同等の収束性をもたらした．具体的に $m = 2$ かつ $L = 2$ と $L = 8$ の場合，Multilevel-FCF(AGG) と MGRIT(AGG) の反復回数はそれぞれ，$L = 2$ では 26 回と 15 回であったが，$L = 8$ ではどちらも同じ 15 回となった．後者の場合は，粗格子率 m の増加に伴い，C 点の区間幅が広がるため，より収束性の高い緩和法となるためである．

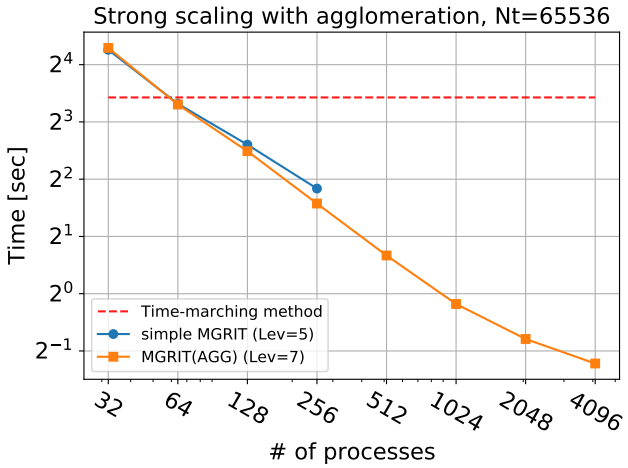


図 6 Run times of MGRIT (blue, circle marker) and MGRIT(AGG) (orange, squared marker) with $N_x^2 = 13^2$, $\Omega = [0,1]^2$, $N_t = 65536$, $T = [0,1]$, and a coarsening factor $m = 4$. The dashed red line indicates the time-marching method without parallelization.

Multilevel-FCF(AGG) が MGRIT(AGG) と同等の収束性を与える場合，より一反復のコストが軽い Multilevel-FCF(AGG) が全体の実行時間が短いことも表 1 より確認できる．しかし Multilevel-FCF(AGG) の収束性は，最も粗いレベルのタイムステップ数 N_{tc} に依存することに注意する必要がある．つまり N_{tc} が十分に小さく直接法を良く近似する場合，また高い並列度を踏まえて反復回数の増加が許容される場合に有効だと考えられる．

4.5 前処理による収束性向上

GMRES への前処理による収束性への影響を評価するために，ソルバとして用いた場合と前処理として用いた場合を比較する．すなわち，Multilevel-FCF(AGG) と Multilevel-FCF(AGG)-GMRES，MGRIT(AGG) と MGRIT(AGG)-GMRES を比較する．特に表 1 内で最も反復回数の削減が大きい場合に絞り，**図 7** に各ソルバの収束履歴を示す．図 7 の左図は $m = 2$ かつ $L = 9$ に対応し，右図は $m = 4$ かつ $L = 4$ に対応する．どちらの場合も前処理として利用することで，2 回から 4 回の反復回数削減を確認した．これは劇的な改善では無いものの，3.3 節で述べたように GMRES との併用による追加演算コストは比較的小さいため (表 1 の列 “time/it.” を参照)，たとえ数回の削減であったとしても全体の実行時間の削減につながった．結果とし

表 1 Time-to-solution in seconds and number of iteartions with $N_x^2 = 13^2$, $N_t = 14336$, and $P_t = 896$ (16 nodes / 56 cores). m: coarsening factor. L: number of levels. N_{tc}: number of time-steps on the coarsest level. The rightmost column ("time/it.") indicates the averaged run time of one iteration.

m	L	N_{tc}	solver	time [sec]	it.	time/it.
2	5	896	simple MGRIT	2.7046	13	0.2080
			simple MGRIT-GMRES	2.5215	12	0.2101
2	8	112	Multilevel-FCF(AGG)	0.4505	26	0.0173
			Multilevel-FCF(AGG)-GMRES	0.4334	24	0.0181
			MGRIT(AGG)	0.6010	15	0.0401
			MGRIT(AGG)-GMRES	0.5322	13	0.0409
2	9	56	Multilevel-FCF(AGG)	0.3069	17	0.0181
			Multilevel-FCF(AGG)-GMRES	0.2656	14	0.0190
			MGRIT(AGG)	0.4416	15	0.0294
			MGRIT(AGG)-GMRES	0.3917	13	0.0301
2	10	28	Multilevel-FCF(AGG)	0.2848	15	0.0190
			Multilevel-FCF(AGG)-GMRES	0.2586	13	0.0199
			MGRIT(AGG)	0.3649	15	0.0243
			MGRIT(AGG)-GMRES	0.3255	13	0.0250
4	4	224	Multilevel-FCF(AGG)	0.4089	27	0.0151
			Multilevel-FCF(AGG)-GMRES	0.3697	23	0.0161
			MGRIT(AGG)	0.9823	16	0.0614
			MGRIT(AGG)-GMRES	0.8630	14	0.0616
4	5	56	Multilevel-FCF(AGG)	0.2783	16	0.0174
			Multilevel-FCF(AGG)-GMRES	**0.2570**	**14**	**0.0184**
			MGRIT(AGG)	0.4414	16	0.0276
			MGRIT(AGG)-GMRES	0.3995	14	0.0285
8	3	224	Multilevel-FCF(AGG)	0.2922	16	0.0183
			Multilevel-FCF(AGG)-GMRES	0.2837	15	0.0189
			MGRIT(AGG)	0.9331	15	0.0622
			MGRIT(AGG)-GMRES	0.9519	15	0.0635
8	4	28	Multilevel-FCF(AGG)	0.3411	15	0.0227
			Multilevel-FCF(AGG)-GMRES	0.3546	15	0.0236
			MGRIT(AGG)	0.3836	15	0.0256
			MGRIT(AGG)-GMRES	0.3941	15	0.0263

て $m = 4$ かつ $L = 4$ における Multilevel-FCF(AGG) は，前処理として利用することで，反復回数を 27 回から 23 回へと約 14.81% 削減し，実行時間を 0.4089 秒から 0.3697 秒へと約 9.59% 削減した．

4.6 3 つの手法を全て組み合わせたソルバの有効性

上記の節では 3 つの手法それぞれに注目してその影響を確認した．本節では最後に表 1 における最も実行時間が短いソルバに注目する．それは $m = 4$ かつ $L = 5$ の Multilevel-FCF(AGG)-GMRES であった．これを表 1 内に太字で示している．Multilevel-FCF(AGG)-GMRES は本研究で検討した 3 つの手法を全て組み合わせたソルバである．粗格子集約，最も粗いレベルにおける緩和法の利用，GMRES への前処理としての利用，という順番に適用した場合の実行時間の変化を見ると，それぞれの手法は約 6.13 倍，約 1.59 倍，約 1.08 倍の高速化を達成した．結果として，Multilevel-FCF(AGG)-GMRES は，従来手法である simple MGRIT と比較して約 10.52 倍の高速化を達成した．この傾向は問題サイズと並列数を増やした場合でも同様に確認された．空間自由度 $N_x^2 = 21^2$，タイムステップ数 $N_t = 57344$，並列度 $P_t = 3584$ における反復回数と実行時間を**表 2** に示す．この問題設定における最も実行時間が短いソルバは $m = 2$ かつ $L = 11$ における Multilevel-FCF(AGG)-GMRES であり，時間逐次解法として比較して 36.97 倍の高速化を達成した．

5. おわりに

本研究では時間発展 Stokes 方程式に対して，粗格子集約手法，最も粗いレベルにおける緩和法の利用，GMRES への前処理としての利用，の 3 つの手法を MGRIT に適用した．粗格子集約を行うことで，高並列度環境においても粗いレベルの効率的な並列処理を可能にし，その改善効

表 2 Time-to-solution in seconds and number of iterations with $N_x^2 = 21^2$, $N_t = 57344$, and $P_t = 3584$ (64 nodes / 56 cores). m: coarsening factor. L: number of levels. N_{tc}: number of time-steps on the coarsest level. The rightmost column ("time/it.") indicates the averaged run time of one iteration.

m	L	N_{tc}	solver	time [sec]	it.	time/it.
-	-	-	Time-marching method	93.4030	-	-
2	5	3584	simple MGRIT	88.4082	13	6.8006
			simple MGRIT-GMRES	81.7821	12	6.8152
2	10	112	Multilevel-FCF(AGG)	4.0378	22	0.1835
			Multilevel-FCF(AGG)-GMRES	3.7205	20	0.1860
			MGRIT(AGG)	5.7740	15	0.3849
			MGRIT(AGG)-GMRES	5.0513	13	0.3886
2	11	56	Multilevel-FCF(AGG)	2.8648	15	0.1910
			Multilevel-FCF(AGG)-GMRES	**2.5265**	**13**	**0.1943**
			MGRIT(AGG)	4.3359	15	0.2891
			MGRIT(AGG)-GMRES	3.7921	13	0.2917
2	12	28	Multilevel-FCF(AGG)	2.9867	15	0.1991
			Multilevel-FCF(AGG)-GMRES	2.6299	13	0.2023
			MGRIT(AGG)	3.6647	15	0.2443
			MGRIT(AGG)-GMRES	3.2160	13	0.2474
4	5	224	Multilevel-FCF(AGG)	3.8426	23	0.1671
			Multilevel-FCF(AGG)-GMRES	3.3902	20	0.1695
			MGRIT(AGG)	8.5754	15	0.5717
			MGRIT(AGG)-GMRES	8.0294	14	0.5735
4	6	56	Multilevel-FCF(AGG)	2.9724	16	0.1858
			Multilevel-FCF(AGG)-GMRES	2.6386	14	0.1885
			MGRIT(AGG)	4.7192	16	0.2949
			MGRIT(AGG)-GMRES	4.2667	14	0.3048
4	7	14	Multilevel-FCF(AGG)	3.2796	16	0.2050
			Multilevel-FCF(AGG)-GMRES	2.9102	14	0.2079
			MGRIT(AGG)	3.5275	16	0.2205
			MGRIT(AGG)-GMRES	3.1321	14	0.2237
8	4	112	Multilevel-FCF(AGG)	3.2141	15	0.2143
			Multilevel-FCF(AGG)-GMRES	3.0413	14	0.2172
			MGRIT(AGG)	6.7425	15	0.4495
			MGRIT(AGG)-GMRES	6.3415	14	0.4530

果が最も高かった．また最も粗いレベルにおける緩和法の利用では，そのレベルのタイムステップ数が十分小さい場合には緩和法は直接解法を良く近似し，高い並列性を維持したまま同様の収束性を提供することが分かった．さらにGMRES への前処理として利用することで，反復回数の削減は僅かであるものの，少ない追加コストで収束性を改善した．結果としてこの 3 つの手法全てを組み合わせることで，従来の MGRIT と比較して実行時間を 10.52 倍短縮した．本研究では，この組み合わせが MGRIT のスケーラビリティを向上させる例を示した．

また本研究では時間並列化のみに焦点を当てて実験を行った．そのため検討した各手法を空間並列化と組み合わせて評価する必要がある．今後の課題として，[17] に挙げられるような鞍点系に対する Multigrid 法を適用することが挙げられる．また応用問題としてジオダイナミクスに基づく大きな粘性係数を持つ Stokes 流れへの適用も検討している．

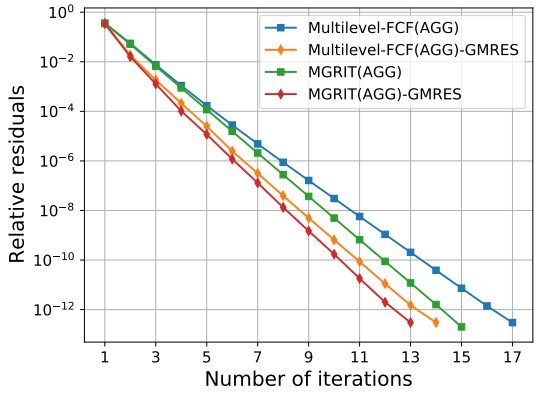
参考文献

[1] Gander, M. J.: 50 years of time parallel time integration, *Multiple shooting and time domain decomposition methods*, Springer, pp. 69–113 (online), DOI: 10.1007/978-3-319-23321-5_3 (2015).

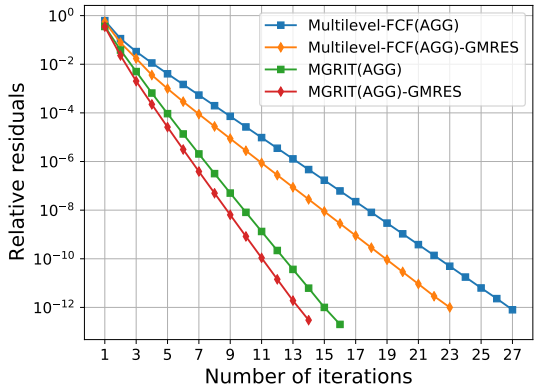
[2] Ong, B. and Schroder, J.: Applications of time parallelization, *Computing and Visualization in Science*, Vol. 23, No. 11 (online), DOI: 10.1007/s00791-020-00331-4 (2020).

[3] Horton, G. and Vandewalle, S.: A Space-Time Multigrid Method for Parabolic Partial Differential Equations, *SIAM Journal on Scientific Computing*, Vol. 16, No. 4, pp. 848–864 (online), DOI: 10.1137/0916050 (1995).

[4] Emmett, M. and Minion, M. L.: Toward an Efficient Parallel in Time Method for Partial Differential Equations, *Communications in Applied Mathematics and Com-*



(a) $m = 2$ and $L = 9$



(b) $m = 4$ and $L = 4$

図 7 Number of iterations of unpreconditioned solvers (blue and green, squared) and preconditioned solvers (orange and red, diamond) in Table. 1.

putational Science, Vol. 7, pp. 105–132 (online), DOI: 10.2140/camcos.2012.7.105 (2012).

[5] Falgout, R. D., Friedhoff, S., Kolev, T. V., MacLachlan, S. P. and Schroder, J. B.: Parallel Time Integration with Multigrid, *SIAM Journal on Scientific Computing*, Vol. 36, No. 6, pp. C635–C661 (online), DOI: 10.1137/130944230 (2014).

[6] Ries, M., Trottenberg, U. and Winter, G.: A note on MGR methods, *Linear Algebra and its Applications*, Vol. 49, pp. 1–26 (online), DOI: 10.1016/0024-3795(83)90091-5 (1983).

[7] Falgout, R. D., Katz, A., Kolev, T. V., Schroder, J. B., Wissink, A. M. and Yang, U. M.: Parallel Time Integration with Multigrid Reduction for a Compressible Fluid Dynamics Application, Technical report, Lawrence Livermore National Laboratory (2014).

[8] Friedhoff, S., Hahne, J., Kulchytska-Ruchka, I. and Schöps, S.: Exploring Parallel-in-Time Approaches for Eddy Current Problems, *Progress in Industrial Mathematics at ECMI 2018*, Springer International Publishing, pp. 373–379 (online), DOI: 10.1007/978-3-030-27550-1_47 (2019).

[9] Hessenthaler, A., Nordsletten, D., Röhrle, O., Schroder, J. B. and Falgout, R. D.: Convergence of the multigrid reduction in time algorithm for the linear elasticity equations, *Numerical Linear Algebra with Applications*, Vol. 25, No. 3, p. e2155 (online), DOI: 10.1002/nla.2155 (2018).

[10] Howse, A., Sterck, H., Falgout, R., MacLachlan, S. and Schroder, J.: Parallel-In-Time Multigrid with Adaptive Spatial Coarsening for The Linear Advection and Inviscid Burgers Equations, *SIAM Journal on Scientific Computing*, Vol. 41, No. 1, pp. A538–A565 (online), DOI: 10.1137/17M1144982 (2019).

[11] May, D. A., Sanan, P., Rupp, K., Knepley, M. G. and Smith, B. F.: Extreme-Scale Multigrid Components within PETSc, *Proceedings of the Platform for Advanced Scientific Computing Conference*, PASC '16, New York, NY, USA, Association for Computing Machinery, (online), DOI: 10.1145/2929908.2929913 (2016).

[12] Nakajima, K.: OpenMP/MPI Hybrid Parallel Multigrid Method on Fujitsu FX10 Supercomputer System, *2012 IEEE International Conference on Cluster Computing Workshops*, pp. 199–206 (online), DOI: 10.1109/ClusterW.2012.35 (2012).

[13] Nomura, N., Fujii, A., Tanaka, T., Marques, O. and Nakajima, K.: Algebraic Multigrid Solver Using Coarse Grid Aggregation with Independent Aggregation, *2018 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, pp. 1104–1112 (online), DOI: 10.1109/IPDPSW.2018.00170 (2018).

[14] Speck, R., Ruprecht, D., Emmett, M., Bolten, M. and Krause, R.: A space-time parallel solver for the three-dimensional heat equation, *Parallel Computing: Accelerating Computational Science and Engineering (CSE), Proceedings of the International Conference on Parallel Computing, ParCo 2013*, Advances in Parallel Computing, Vol. 25, IOS Press, pp. 263–272 (online), DOI: 10.3233/978-1-61499-381-0-263 (2013).

[15] McDonald, E., Pestana, J. and Wathen, A.: Preconditioning and Iterative Solution of All-at-Once Systems for Evolutionary Partial Differential Equations, *SIAM Journal on Scientific Computing*, Vol. 40, No. 2, pp. A1012–A1033 (online), DOI: 10.1137/16M1062016 (2018).

[16] Ryo, Y., Akihiro, F. and Teruo, T.: MGRIT preconditioned Krylov subspace method, *IEEE/ACM Proceedings of SC18, research poster* (2018).

[17] Benzi, M., Golub, G. and Liesen, J.: Numerical solution of saddle point problems, *Acta Numerica*, Vol. 14, pp. 1 – 137 (2005).

[18] Vanka, S.: Block-implicit multigrid solution of Navier-Stokes equations in primitive variables, *Journal of Computational Physics*, Vol. 65, No. 1, pp. 138–158 (online), DOI: 10.1016/0021-9991(86)90008-2 (1986).

[19] Wittum, G.: Multi-grid methods for stokes and navier-stokes equations, *Numerische Mathematik*, Vol. 54, pp. 543–563 (1989).

[20] Braess, D. and Sarazin, R.: An efficient smoother for the Stokes problem, *Applied Numerical Mathematics*, Vol. 23, No. 1, pp. 3–19 (online), DOI: 10.1016/S0168-9274(96)00059-1 (1997).

[21] He, Y. and MacLachlan, S. P.: Local Fourier analysis of block-structured multigrid relaxation schemes for the Stokes equations, *Numerical Linear Algebra with Applications*, Vol. 25, No. 3, p. e2147 (online), DOI: 10.1002/nla.2147 (2018).

[22] Claus, L.: Multigrid smoothers for saddle point systems, PhD Thesis, University of Wuppertal (2019).

[23] Saad, Y. and Schultz, M. H.: GMRES: A Generalized Minimal Residual Algorithm for Solving Nonsymmetric Linear Systems, *SIAM J. Sci. Stat. Comput.*, Vol. 7, No. 3, pp. 856–869 (online), DOI: 10.1137/0907058 (1986).