計算工学講演会論文集 Vol. 28 (2023 年 5 月)
計算工学会

並列有限要素解析における大域的固有モードを利用した Deflated CG 法の性能評価

Performance Evaluation of the Deflated CG Method using Global Eigen-mode in Parallel Finit Element Method

村井拓海[1)]三目直登[2)]森田直樹[3)]
Takumi Murai, Naoto Mitsume and Naoki Morita

[1)]筑波大学 システム情報工学研究群（〒 305-8573 茨城県つくば市天王台 1-1-1, E-mail: murai.takumi.tm@alumni.tsukuba.ac.jp）
[2)]博 (工) 筑波大学 システム情報系 助教（〒 305-8573 茨城県つくば市天王台 1-1-1, E-mail: mitsume@kz.tsukuba.ac.jp）
[3)]博 (環境) 筑波大学 システム情報系 助教（〒 305-8573 茨城県つくば市天王台 1-1-1, E-mail: nmorita@kz.tsukuba.ac.jp）

The deflated conjugate gradient (CG) method is one of the linear solvers for numerical simulations, which can improve the convergence of the approximate solution since the condition number can be reduced by inputting a known independent basis of the solution space. In this study, the global eigen-mode deflation type deflated CG method, we analyzed problems with different number of conditions using structural analysis as an example and measured the number of iterations and computation time. As the results, by examining the breakdown of computation time in terms of the speed-up factor, it became clear that this method exhibits high parallel computation performance around an appropriate number of contractions.

Key Words : *Deflated CG method, preconditioning, Finite Element Method, eigen-mode deflation, parallelization*

1. 序論

有限要素法の並列計算手法のひとつとして、領域分割法がある。この方法では解析モデルを複数の領域 (sub-domain) に分割し、それぞれの領域について並列計算を行うことができる。またあらかじめ計算データを分割することから、分散メモリ型並列計算機での並列計算に適しており、大規模問題の数値シミュレーションを行う上で大きな課題となる計算時間の削減に有効な手法である。

有限要素法に基づいたシミュレーションは、支配方程式の弱形式化及び離散化を経て最終的に疎な係数行列を持つ連立一次方程式の求解に帰着する。コンピュータ上の数値シミュレーションにおいて、連立一次方程式の求解を行う線形ソルバが解析に要する計算時間の大半を占める。

線形ソルバは直接法と反復法に大別されるが、並列計算を行う場合は並列計算効率の観点から多くの場合反復法が採用される。反復法はある手順に沿って解ベクトルを反復的に更新し、真の解へ漸近させる手法である。この手法は代数的に解を求める直接法と比較すると計算時間の短縮の観点で有利だが、数値誤差の影響を受けやすく、実用にあたっては安定した動作のために問題に適切な処理を施してから求解を行う必要がある。この処理に相当するアルゴリズムを計算する動作を前処理 (preconditioning) と呼称する。効率的な並列計算を実現するためには、反復法前処理についても並列計算に適したアルゴリズムの利用が重要となる。

並列計算における前処理は、局所的前処理と大域的前処理の 2 種類に大別される。局所的前処理とは、分割領域内部の情報のみに前処理を施す方法であり、分割領域間の相互作用を無視することに相当する。局所的前処理では、前処理に相当するアルゴリズムにおいて他領域との通信が発生しないため、前処理による通信コストの増加が抑えられるという利点がある反面、自領域内の情報のみを用いて前処理を行うため、一般的にその効果は限定的である。一方、大域的前処理とは分割領域内部の情報だけでなく分割領域同士の相互作用を考慮した前処理である。領域間の相互作用の情報を前処理に利用するため、局所前処理に比べ高い前処理性能を有すると予想されるが、他領域とのデータ通信が必須となるため、このコスト増加を検討に含める必要がある。大域的前処理と局所的前処理は上述のように異なる特性を持ち、適用する前処理の種類や問題によって適切な方法を選択することが必要になる。

本研究では、精緻なシミュレーションから生じる大規模問題を対象に、係数行列の固有ベクトルをもとに前処理を施す eigen-mode deflation (固有モード縮約) に注目し、その有効性を評価する。Eigen-mode deflation では、既知の固有ベクトルの情報をもとに縮約モデル (reduced order model) を作成し、解の収束性向上を図る。過去に行った検討 [1] において、逐次環境での構造解析における eigen-mode deflation が悪条件問題に対して効果的に機能することを確認した。そこで、検討範囲を逐次計算から分散メモリ型並列計算機を用いた並列計算に拡張し、その並列計算性能を評価する。本論文の範囲では、基礎的検討として大域的前処理を対象とし、計算速度、加速率などの観点から計算性能を比較する。

2. 線形ソルバと前処理

(1) 線形ソルバの概要

連立一次方程式の解を数値計算によって求める線形ソルバは、標準的な有限要素法であれば数値シミュレーションの計算プロセスの多くを占める部分である。今日までに直接法と反復法に大別される様々な手法が提案されており、対象とする問題に適した線形ソルバを選択することが重要である。

直接法は、有限回の計算で代数的に解を求める手法である。動作は極めて安定しており計算結果の精度が高い反面、メモリ消費量や計算時間が膨大となる大規模問題には適さない。

対して反復法は、ある規則に従って数値解を反復更新し正しい解へと漸近させる手法であり、解を適切に更新することで、代数的処理と比較して少ない計算量で解を求めることができる。反復法は、成分のほとんどが 0 である疎行列に対して特に効果を発揮する。0 でない要素のみを格納するデータ構造を用いることでメモリが削減でき、さらに 0 である成分とのベクトル積の計算を省略する実装により大幅に計算量を節約することができる。また反復法はそのアルゴリズムの多くがベクトル演算で構成されるため、MPI (Message Passing Interface) [2] を使用した並列計算と相性がよく、実行時の計算効率が優れる手法である。このような特性から、適切に利用することで直接法と比較して計算時間を大きく短縮できる点で有利であるが、丸め誤差や桁落ちなどの数値誤差に弱く、正しい解が求まらない現象が起こる可能性があるため、安定した運用には後述する反復法前処理を適切に施すことが重要である。

Algorithm 1 Deflated CG method

```
1:  function CALC_DEFLATED_CG(A, b, Z)
2:      r̄^(0) ← b
3:      g ← (Z^T AZ)^{-1} Z^T r̄^(0)
4:      r^(0) ← r̄^(0) − AZg
5:      if (||r^(0)||/||r̄^(0)|| < ε) then
6:          skip main loop
7:      end if
8:      for (i = 0, k++) do
9:          if (i = 0) then
10:             p^(0) ← r^(0)
11:         else
12:             β^(i) ← (r^(i))^T r^(i) / (r^(i−1))^T r^(i−1)
13:             p^(i) ← r^(i) + β^(i) p^(i−1)
14:         end if
15:         v^(i) ← Ap^(i)
16:         v^(i) ← v^(i) − AZ(Z^T AZ)^{-1} Z^T v^(i)
17:         α^(i) ← (r^(i))^T r^(i) / (p^(i))^T v^(i)
18:         x^(i+1) ← x^(i) + α^(i) p^(i)
19:         r^(i+1) ← r^(i) − α^(i) v^(i)
20:         if (||r^(i+1)||/||r̄^(0)|| < ε) then
21:             exit
22:         end if
23:     end for
24:     h ← (Z^T AZ)^{-1} Z^T A(x^(i+1) − x^(0))
25:     x^(i+1) ← x^(i+1) + Z(g − h)
26:     return x^(i+1)
27: end function
```

(2) 反復法前処理

反復法前処理とは、反復法が正常に収束するように入力する問題に予め式変形を施すことに相当するアルゴリズムのことである。反復法の収束性は前処理の有無によって大きく左右される。係数行列の最大固有値と最小固有値の比は条件数（condition number）と呼ばれ、条件数が大きい問題では反復法の収束性が悪化する傾向があるため、前処理により条件数を削減し収束性を向上させることが重要である。前処理のアルゴリズムは、式 (1) に示すように、対象とする連立一次方程式 $Ax = b$ の両辺に前処理行列 P の逆行列を左からかける操作に相当する。ここで A は正定値対称行列（Symmetric Positive Definite : SPD）、x は解ベクトル、b は右辺ベクトルである。

$$\begin{aligned} Ax &= b \\ P^{-1}Ax &= P^{-1}b \end{aligned} \tag{1}$$

このとき、P は $P^{-1}A$ の条件数が A と比べて小さくなるように選定する。前処理行列 P の生成時間よりも多くの時間を前処理の効果で短縮することができれば、その前処理は効果的に動作しているといえる。

(3) 大域的・局所的前処理

領域分割法により並列計算を行う場合、各分割領域は全体剛性行列を分割したブロック行列で表現することができ、これらを分散メモリ型並列計算機の計算プロセスに割り振って代数演算の並列処理を行う。このとき反復法前処理は、前処理を施す際に使用する情報によって大域的前処理と局所的前処理に大別される。

大域的前処理とは分割領域内部の情報だけでなく分割領域同士の相互作用を考慮した前処理である。領域間の相互作用の情報を前処理に利用するため、自領域以外の情報を用いて前処理を行うことが可能であることから比較的高い前処理性能を有すると予想される。ただし、この操作には他領域とのデータ通信による情報の更新が必須となるため、このコスト増加を検討に含める必要がある。

対して局所的前処理とは、分割領域内部の情報のみに前処理を施す方法であり、分割領域間の相互作用を無視することに相当する前処理である。局所的前処理では、前処理に相当するアルゴリズムにおいて他領域との通信が発生しないため、前処理による通信コストの増加が抑えられるという利点があるが、自領域内の情報のみを用いて前処理を行うため、一般的にその効果は限定的である。

構造解析の場面では、解析する問題によってモデル形状や前処理に使用する情報が異なるため、大域的前処理と局所的前処理の性能を十分に調査し、適した手法を選択することが非常に重要となる。

表–1 model 1 における並列数及び縮約次数と反復回数の関係

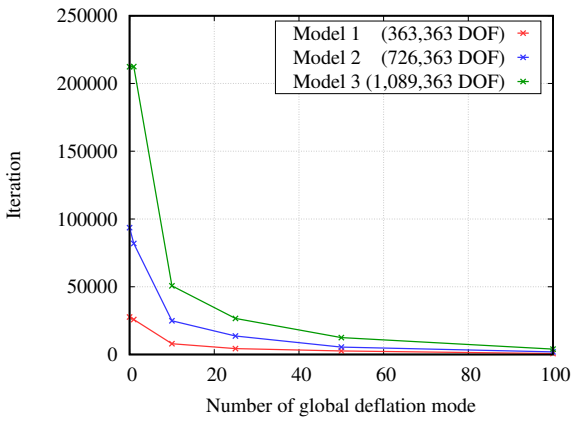
		Number of MPI process									
		1	2	4	8	16	32	64	128	256	CV
Number of deflation mode	0	27,561	27,502	27,745	27,862	27,846	27,813	27,561	27,552	27,488	0.0053
	1	25,042	25,235	25,117	26,399	26,148	26,504	25,213	27,494	25,264	0.0312
	10	8,021	7,998	7,953	8,010	7,933	7,940	7,858	7,946	7,947	0.0058
	25	4,363	4,345	4,348	4,341	4,343	4,341	4,344	4,352	4,361	0.0018
	50	2,628	2,620	2,622	26,21	2,616	2,616	2,622	2,625	2,625	0.0015
	100	663	661	662	662	662	662	662	661	661	0.0009

表–2 model 2 における並列数及び縮約次数と反復回数の関係

		Number of MPI process									
		1	2	4	8	16	32	64	128	256	CV
Number of deflation mode	0	92,280	92,770	95,492	94,762	91,952	93,393	91,461	95,593	94,790	0.0160
	1	79,920	79,558	80,497	85,238	78,867	80,845	79,855	80,083	92,494	0.0502
	10	25,130	25,069	24,537	24,707	24,973	24,560	24,781	25,108	24,795	0.0087
	25	13,691	13,688	13,594	13,639	13,589	13,653	13,561	13,621	13,650	0.0031
	50	5,436	5,411	5,455	5,410	5,401	5,467	5,448	5,467	5,457	0.0045
	100	1,941	1,935	1,938	1,934	1,936	1,938	1,938	1,941	1,927	0.0020

表–3 model 3 における並列数及び縮約次数と反復回数の関係

		Number of MPI process									
		1	2	4	8	16	32	64	128	256	CV
Number of deflation mode	0	212,517	211,891	214,344	212,706	210,294	210,661	213,628	211,977	213,054	0.0058
	1	220,450	217,602	212,344	216,851	217,211	217,765	200,223	203,345	206,320	0.0326
	10	51,631	50,615	50,333	50,792	50,303	49,298	51,114	51,470	50,426	0.0131
	25	26,820	26,488	26,678	26,783	26,424	26,861	26,453	26,361	26,824	0.0071
	50	12,486	12,425	12,521	12,500	12,511	12,455	12,529	12,418	12,443	0.0032
	100	3,664	3,641	3,747	3,738	3,759	3,768	4,982	3,768	4,781	0.1216



図–1 モデル別の反復回数と縮約基底本数の関係

3. Eigen-mode deflation

(1) 共役勾配法

共役勾配法 (CG 法、Conjugate Gradient method) は、正定値対称行列を対象とした非定常反復法である [3]。反復法の最も基本的な手法である最急降下法は目的関数の最小勾配方向への探索を繰り返す手法であるが、既に探索した方向を再度探索してしまう可能性があり、解析対象によっては無駄な反復が発生し計算時間が増加してしまう。この問題を係数行列の情報を用いて対策しているのが本手法である。反復解、残差、探索方向の 3 本のベクトルを更新しながら反復計算を行い、理論上最大 n 回（n は係数行列の階数）で収束することが知られているが、前処理なしでは数値誤差に弱く収束が不安定になる場合もある。

(2) Deflated 共役勾配法

Deflated 共役勾配法 (deflated CG 法、deflated Conjugate Gradient method) とは、係数行列に縮約 (deflation) を施した共役勾配法である [4]。解空間の基底で構成された任意の本数の線形独立なベクトルを入力することでその情報をもとに縮約モデルを作成し、計算を行う。

表–4 縮約次数毎の deflated CG 法における計算時間の割合

Number of deflation mode	0	1	10	25	50	100
SpMV	82.91 %	71.01 %	56.48 %	41.26 %	30.98 %	14.43 %
Inner product	10.94 %	6.98 %	6.01 %	5.35 %	4.76 %	2.76 %
DeMV	0.00 %	9.55 %	11.66 %	14.30 %	16.32 %	19.62 %
DeMV (without comm)	0.00 %	2.72 %	10.51 %	17.51 %	18.83 %	23.21 %
Precond	3.44 %	2.91 %	2.37 %	1.73 %	1.28 %	0.54 %

Deflated 共役勾配法のアルゴリズムを Algorithm 1 に示す。反復アルゴリズム中における CG 法との差分は係数行列から既知の基底情報を縮約する 17 行目のみであり、既知の基底として 0 ベクトルを入力すると CG 法と等価となる。

ここで、入力する既知の線形独立なベクトルとして低次の固有ベクトル Z を使用する eigen-mode deflation (固有モード縮約) と呼ばれる手法に着目する。大規模自由度問題を扱う場合、問題の条件数は大きくなる傾向があり、反復法の収束性低下が問題となる。Deflated 共役勾配法では、低次固有値 λ_1 から λ_k ($\lambda_1 < \lambda_k < \lambda_n$) に対応する k 本 ($k < n$) の固有ベクトルが他の固有値解法などによって得られ既知である場合、これらの情報を縮約処理によって反復から省略することができる。この操作により解くべき問題の条件数を $\frac{\lambda_n}{\lambda_1}$ から $\frac{\lambda_n}{\lambda_k}$ に削減することができるため、反復法の収束性の向上が期待される。

本研究では解析領域全体の大域的な固有モードを既知の入力基底として利用し、全解析領域、すなわち全体剛性行列に対して縮約処理を施す global eigen-mode deflation の悪条件問題に対する並列計算性能の評価を行う。解析する問題の条件数、入力基底数及び並列数を変化させ、分散メモリ型並列計算機における計算性能を議論する。

並列計算性能の評価には台数効果の指標である加速率 S_p (speed-up factor) を利用する [2]。加速率 S_p は逐次での実行時間 T_s、並列数 P での実行時間 T_p を用いて式 (2) のように表す。

$$S_p = T_s / T_p \tag{2}$$

4. 構造解析による数値例

(1) 解析条件

構造解析の有限要素法シミュレーションにおいて、既知の基底として解析モデル全体の global な固有モードを利用し、全体剛性行列に対して縮約を行う global eigen-mode deflation の MPI 並列計算環境での性能評価を行うことを目的とする。

構造解析の事象は片持ち梁の曲げ問題を対象とし、縮約数及び並列数を変化させ分散メモリ型並列計算機での計算性能を測定する。片持ち梁モデルは、問題の条件数と線形ソルバの収束性の関係を検討するため、長さの異なる 3 種類の解析モデル model 1 ~ model 3 を使用する。ここで各モデルの寸法諸元は model 1 : (1.0×0.01×0.01 m、 363,363 自由度)、model 2 : (2.0 × 0.01 × 0.01 m、 726,363 自由度)、model 3 : (3.0 × 0.01 × 0.01 m、1,089,363 自由度) である。物性値はヤング率 E : 206, 000 N/mm2、ポアソン比 ν : 0.3 の鋼材とした。

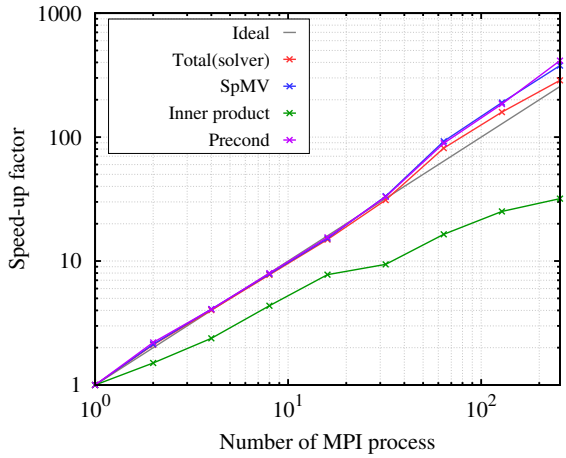
今回は大域的前処理である global eigen-mode deflation の並列計算性能を評価するため、既知の固有モードの次数を 0、1、10、25、50、100 に設定した。並列数は 1、2、4、8、16、32、64、128、256 とし、各モデル毎に測定を行う。線形ソルバでは上記の縮約前処理に加えて対角スケーリング前処理を使用し、固有モードの取得には Lanczos 逆べき乗法を用いた。全てのデータは 3 回分測定を行い、検討にはその平均値を使用する。分散メモリ型並列計算機は東京大学の Oakbridge-CX を利用した。

(2) 計算結果と考察

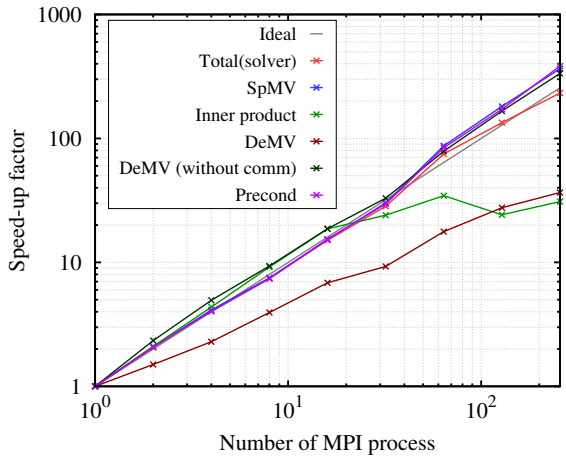
a) 解析モデルの条件数と反復回数

まず model 1 ~ model 3 の基底本数と並列数に対応する反復数を、同一基底本数における標準偏差を平均値で除し無次元化した変動係数 CV と共に表 1 ~ 表 3 に示す。この結果より、同じ縮約次数であれば、多くの場合反復数は並列数が変化してもほぼ一致するが、変動係数 CV に注目すると縮約数が 1 の場合や model 3 における縮約数 100 の場合など、特定の条件においては並列数による反復回数の測定結果に差が生じる場合があることがわかる。今回の検討において使用した global eigen-mode deflation では、解くべき問題を並列化した場合でも、縮約に用いる固有モードは共通であるため数理的な変化はない。よって反復数が並列数によって変化する原因は計算手順の変更による数値誤差の変化に起因するものだと考えられる。この反復数にばらつきが出た事象の原因として、領域分割の過程において、特定の条件でオーダーの差が大きな値同士の和などが発生し、丸め誤差等が発生していると考察する。

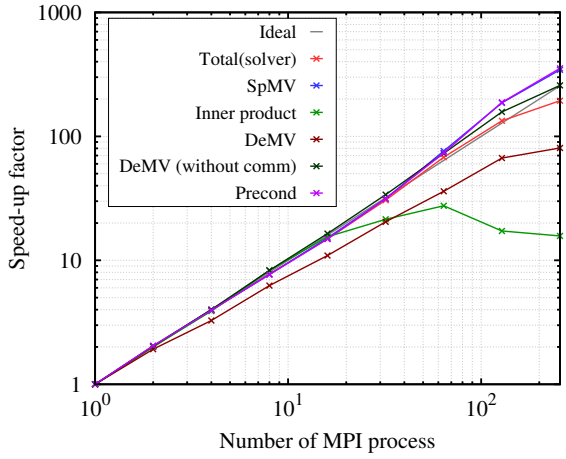
次に model 1 ~ model 3 の反復回数と縮約基底本数の関係を図 1 に示す。この結果より全ての問題において縮約次数の増加に伴って反復回数が単調減少しており、条件数の大きな問題のほうが同一基底数でも反復回数の減少度合いが大きく、悪条件問題で特に eigen-mode deflation が効果的に機能することが確かめられた。これは過去に逐次環境で測定した際の傾向と一致する [1]。またすべてのモデルにおいて今回測定した最大の縮約数 100 で反復回数が縮約なしの場合と比較して 2% 程度まで減少しており、グラフの概形でもこの付近ではほぼ反復数の変化は横這いとなっていることから今回の検討に使用した縮約数のデータセットが妥当なものであることが確かめられた。



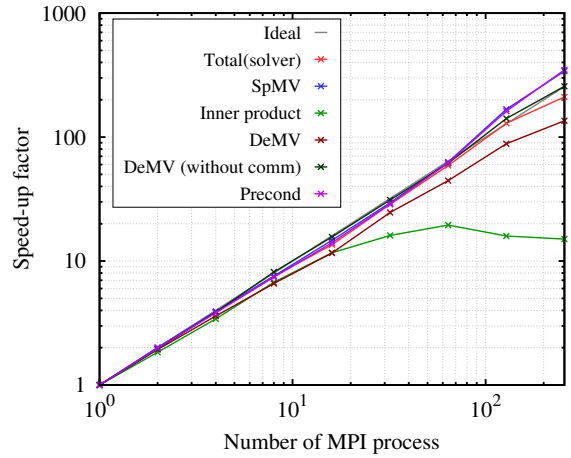
図–2 加速率 Sp (Number of global deflation mode = 0)



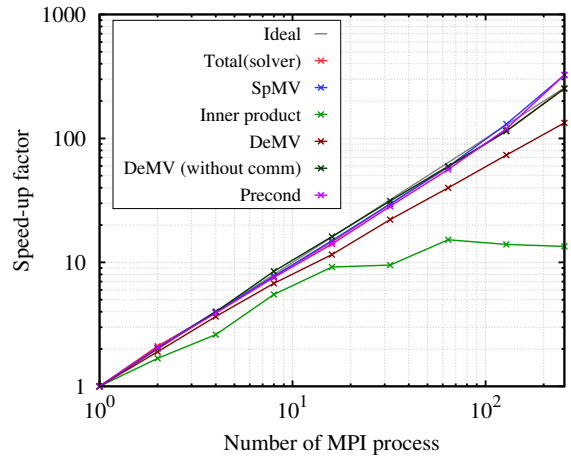
図–3 加速率 Sp (Number of global deflation mode = 1)



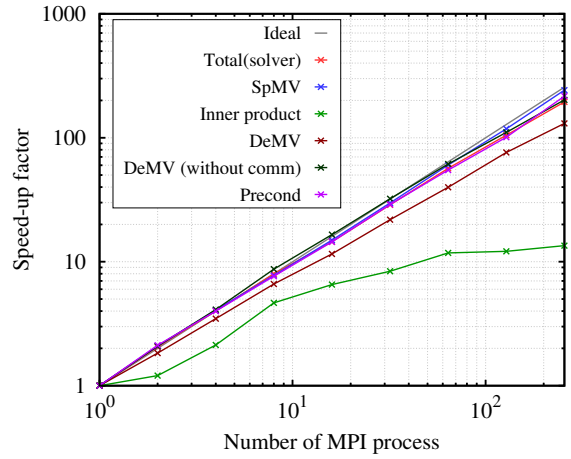
図–4 加速率 Sp (Number of global deflation mode = 10)



図–5 加速率 Sp (Number of global deflation mode = 25)



図–6 加速率 Sp (Number of global deflation mode = 50)



図–7 加速率 Sp (Number of global deflation mode = 100)

b) 並列計算性能

プログラムの実行にあたり、線形ソルバである deflated CG 法の計算時間をその成分毎に測定した。総計算時間 (Total) の内訳は、分割領域間での通信を要する疎行列ベクトル積 (SpMV)、ベクトル内積 (Inner product)、密行列ベクトル積 (DeMV) と分割領域間での通信を要さない密行列ベクトル積 (DeMV (without comm))、対角スケーリング前処理 (Precond) である。これらの計算時間内訳の割合と縮約次数の関係を表 4 に示す。また model 1 についての測定結果を加速率として図 2〜図 7 に示す。

表 4 より、疎行列ベクトル積 (SpMV) とベクトル内積 (Inner product) の計算時間割合は縮約次数の増加に伴って減少し、逆に密行列ベクトル積 (DeMV) と分割領域間での通信を要さない密行列ベクトル積 (DeMV (without comm)) の割合は増加することがわかる。これは縮約次数 M が大きくなると固有モード Z のサイズが $N \times M$ 増加し、algorithm1 中 17 行目の縮約処理に要する密行列ベクトル積の演算量が $O(M^2)$ のオーダーで増加し、

これに含まれる密行列ベクトル演算の計算割合が徐々に卓越するためである。

続いて図 2〜図 7 より、いずれの基底本数の場合でも、総計算時間はおよそ理想的な加速率を示すことがわかる。しかし、ベクトル内積 (Inner product) に注目すると、並列数が増加するにつれ加速率の増加が緩やかになり、一定の値に漸近する。これはベクトル内積の速度向上の限界 (saturation) が起きていることを表す。表 4 に示す通り、本手法と検討範囲においてベクトル内積が全体の計算時間に占める割合は大きくないため、この現象が全体の並列計算性能に与える影響は限定的である。また密行列ベクトル積 (DeMV) に注目すると、縮約数が比較的小さい図 3 や図 4 の場合では加速率が比較的悪いことがわかる。これは上述の通り縮約数 M によって deflated CG 法のアルゴリズム上で計算する密行列ベクトル積の計算オーダーが決定されることによるものであり、縮約数が小さい場合領域内での計算量が小さく、通信時間の割合が卓越すると考えられ、これは得られた測定結果の傾向と一致する。

また並列プロセス数が大きな領域では加速率が理想値のラインを超えており、スーパーリニア・スピードアップが起きていることがわかる。これは並列化により分割領域のデータ量が小さくなることで CPU のキャッシュ・ヒット率が増加し、計算が高速化したためだと考えられる。

これらの結果から global eigen-mode deflation は、検討したモデルに対して実用で想定される反復数となる縮約次数においても良好な並列計算性能を示すことが確認された。

5. 結論

本論文では、並列有限要素法に基づいた数値シミュレーションの計算速度向上のために、計算量の大きな割合を占める線形ソルバに着目し、低次の固有モードを利用して deflated 共役勾配法の性能向上を図る大域的前処理である global eigen-mode deflation を deflated 共役勾配法に適用し、性能評価を行った。

まず条件数の異なる複数の解析モデルを用いた検証によって、逐次環境と同様に MPI 並列環境でも条件数の大きな問題であるほど eigen-mode deflation が反復法の収束性の向上に大きく寄与することが確認できた。

次に線形ソルバ deflated CG 法の計算時間の内訳を総計算時間に対する割合とそれぞれの加速率を用いてその性能を評価した。この検討により、縮約前処理のために必要となる密行列ベクトル積の並列性能評価を行い、これが想定される縮約数の範囲で理想的な性能を発揮することを確認した。

ただし、解析する問題によって収束性が向上する特性は大きく異なることが予想され、並列計算性能が大きく低下する領域での計算が必要となる可能性は現段階では否定できない。また本論文の検討では固有モードの取得のため直接法のアルゴリズムを含む lanczos 逆べき乗法を使用したため、並列計算においてはこの部分の計算コストが大きくなるため、固有対を反復計算で取得することができる LOBPCG 法 [5] などのより優れた固有値解法に関して調査し、適した手法を選択する必要がある。これらを考慮して実用に向けた総合的な評価を行うことに加え、分割領域の集合単位で局所的な縮約処理を行う subdomain eigen-mode deflation の性能評価を今後の課題とする。

6. 謝辞

本研究は JSPS 科研費 20K19813 の助成を受けたものである。

参考文献

[1] 村井拓海, 三目直登, 森田直樹: 有限要素解析における縮約モデルを用いた反復法前処理, 日本計算工学会第 27 回計算工学講演会, 2022.
[2] 片桐孝洋: 「スパコンプログラミング入門 並列処理と MPI の学習」, 東京大学出版会 , 2013.
[3] Magnus, R. H.,& Eduard, S.: Methods of Conjugate Gradients for Solving Linear Systems, Journal of Research of the National Bureau of Standards, 49(6), 409-436, 1952.
[4] Yousef, S., Yeung. M., Jocelyne, E.,& Frédéric. G.: A Deflated Version of the Conjugate Gradient Algorithm, SIAM Journal on Scientific Computing, Society for Industrial and Applied Mathematics, 21(5), 1909-1926, 2000.
[5] Andrew, V. K.: Toward the Optimal Preconditioned Eigensolver: Locally Optimal Block Preconditioned Conjugate Gradient Method, SIAM J. Sci. Comput., 23(2), 517-541, 2006.