Robust SA-AMG Solver by Extraction of Near-Kernel Vectors

Naoya Nomura
The University of Tokyo
Japan
naoya_nomura@mist.i.u-tokyo.ac.jp

Kengo Nakajima
The University of Tokyo
Japan
nakajima@cc.u-tokyo.ac.jp

Akihiro Fujii
Kogakuin University
Japan
fujii@cc.kogakuin.ac.jp

ABSTRACT

The smoothed aggregation algebraic multigrid (SA-AMG) method is among the fastest solvers for large-scale linear equations, $A\boldsymbol{x} = \boldsymbol{b}$. The SA-AMG method achieves good convergence and scalability by damping various wavelength components efficiently. To achieve this damping, this method creates multi-level matrices which are hierarchically smaller in dimension than the original matrix. Moreover, the convergence can be further improved by setting near-kernel vectors $\boldsymbol{p}$, which satisfy $A\boldsymbol{p} \approx 0$ and $\boldsymbol{p} \neq \boldsymbol{0}$. Generally, the same number of near-kernel vectors are used at each level. In the present work, we propose a method that extracts and adds near-kernel vectors at each level. We evaluate the performance of the solver that extracts the near-kernel vectors and adds them at each level. We use the three-dimensional elastic problem and employ up to 512 processes on the FX10 supercomputer system. By using this method, the performance is improved compared with previous work.

KEYWORDS

Linear solver, Algebraic multigrid method, Near-kernel vectors, Convergence stabilization

1 INTRODUCTION

Iterative methods are widely used for large-scale systems of linear equations $A\boldsymbol{x} = \boldsymbol{b}$ in scientific computing. The smoothed aggregation algebraic multigrid (SA-AMG) method is one of the fastest solvers [1].

The SA-AMG method dumps various wavelength components efficiently by creating multi-level matrices which are hierarchically smaller in terms of dimension than the original matrix. Thereby, this method achieves good convergence and scalability. The SA-AMG method comprises a setup part and a solution part. The setup part creates coarse matrices from a graph structure that is based on the matrix problem. The SA-AMG method then aggregates the unknowns and assigns each aggregate to a node at the next coarser level. The solution part repeatedly relaxes linear equations at each level. The structure is hierarchical, with finer and coarser levels, where the finest level corresponds to the original coefficient matrix problem. Fig. 1 shows the solution part of the SA-AMG method. This process is called the V-cycle. The SA-AMG method can incorporate error components that are difficult to correct using ordinary relaxation methods (such as the Gauss-Seidel method). These components correspond to near-kernel vectors, defined as nonzero vectors $\boldsymbol{p}$ satisfying $A\boldsymbol{p} \approx 0$. The SA-AMG method uses the near-kernel vectors when the multi-level matrices are created to efficiently dump the near-kernel vectors by moving them to coarser levels. In some cases, the near-kernel vectors can be determined from the background physics.

Our prior research indicated that setting a suitable number of near-kernel vectors extracted by the ordinary V-cycle improves convergence [2]. In that previous work, we used the same number of near-kernel vectors at all levels. In the present work, we propose a method that extracts and adds near-kernel vectors at coarser levels and evaluate the robustness and efficiency of the proposed method on the FX10 supercomputer system (using up to 512 processes).

2 NEAR-KERNEL VECTOR EXTRACTION

In the previous work, because we used a near-kernel vector extraction method using the V-cycle at only level 1 [2], we used the same number of near-kernel vectors to create the coarser level matrices at all levels. In the present work, we propose an extraction method that extracts near-kernel vectors at each level. The details of this method are shown in Fig. 2. In this method, we first prepare a random vector as initial vector $\boldsymbol{x}$ (Line 3). Next, the V-cycle is applied μ times to solve $A\boldsymbol{x} = 0$ (Line 4) and input $\boldsymbol{x}$ to matrix B, which is a set of near-kernel vectors (Line 5). A matrix B is prepared at each level, so we can extract the near-kernel vectors at coarser levels. Multi-level matrices are created by near-kernel vectors, so we must recreate the coarser level matrices by using the extracted near-kernel vectors (Line 6). If the number of near-kernel vectors is insufficient, then this method repeats Lines 3 to 6. This method repeats this inner loop up to the level number.

3 NUMERICAL EXPERIMENTS

Experiments were performed on the Fujitsu PRIMEHPC FX10 supercomputer system at the University of Tokyo [3]. We use the Flat MPI model, employing up to 512 cores. The target application was a three-dimensional problem in elasticity. Parallel translation and rotational components are referred to as near-kernel vectors in this problem. Moreover, the experiment

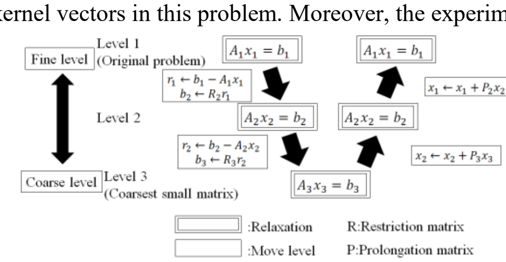

Figure 1: V-cycle of the SA-AMG (Solution part).

```
For level = 1 to max_level − 1
    For n = 1 to extract_number
        Random(x)
        V_cycle^μ (A_level x = 0)
        B_level ← x
        Multilevel_creation(B_level)
    End for
End for
Output B as the near_kernel vectors
```

- $extract\_number$: Number of extracted vectors
- A_{level}: Matrix A of level *level*
- V_cycle^{μ}: Iterate the V-cycle μ times

Figure 2: Details of the near-kernel vector extraction.

was performed as a weak scaling (with a local domain per process of 15^3). The experiment was implemented using the AMGS library [4]. The solution part was executed by the AMG preconditioned GPBiCG method [5]. The relaxation of the solution part was performed twice per level by the Symmetric Gauss-Seidel method. The termination criterion for the 2-norm of the relative residuals was set to 1.0×10^{-7}.

We evaluate the performance of the SA-AMG method by setting the extracted near-kernel vectors at coarser levels. The target of this experiment is shown in Table 1. In addition, we vary the number of near-kernel vectors at level 1. The details are shown in Table 2. Fig. 3 shows the number of iterations in the 512-process case. As shown, convergence is improved by setting a suitable number of near-kernel vectors at coarser levels. Fig. 4 shows the number of iterations and the execution time for various numbers of processes. In Fig. 4, “Best in Case 2,3” gives the best results for Case 2 and Case 3. From this graph, “Best in Case 2,3” achieves good convergence regardless of the number of processes, but the execution time does not improve after the 216-process case. This is due to the trade-off between the increasing cost and the improvement of the convergence by using near-kernel vectors.

Table 1: Experimental cases.

Candidates	Details
Case 1	No extraction at coarser levels (previous work)
Case 2, Case 3	Number of additional near-kernel vectors by each coarser level: 1 (Case 2), 5 (Case 3) (proposed method)

Table 2: Settings of near-kernel vectors at level 1.

Candidates	Details
3p	Parallel translation in X,Y,Z(3 vectors)
6p	3p(3 vectors) + rotation in X,Y,Z(3 vectors)
3p+1,3p+2,…	3p(3 vectors) + extracted near-kernel vectors at level 1($\leq$ 7 vectors)

e.g.: The number of near-kernel vectors at Case 3 and 3p+1
⇒level 1:4, level 2:9, level 3:14

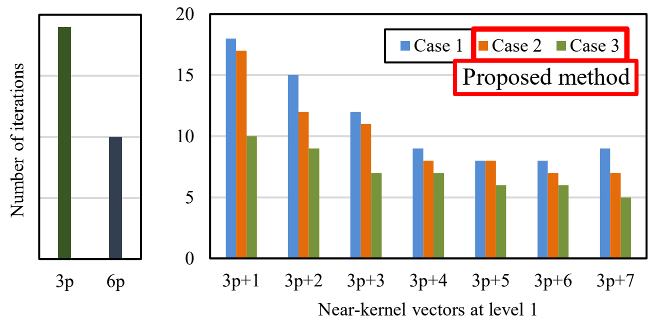

Figure 3: Number of iterations in 512-process case.

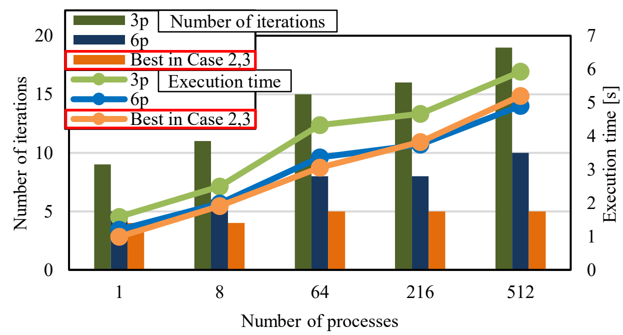

Figure 4: Number of iterations and execution time for various numbers of processes.

4 CONCLUSIONS

In the present work, we propose a method that extracts near-kernel vectors at coarser levels, and investigate the performance for the SA-AMG method. The convergence is improved by using our method, but the calculation cost of per iteration is increased. This is because the problem is easy, and the cost of setup part accounts for most of the execution time. Therefore, we evaluate our method in difficult problem. Moreover, we must find how to decide the optimal number of near-kernels as further research. Further, we also must investigate how to stabilize the convergence (e.g., changing the relaxation, taking a larger overlap size between process domains).

ACKNOWLEDGMENTS

This work was partially supported by Intel(R) Parallel Computing Centers (IPCC), and the Japan Society for the Promotion of Science KAKENHI (grant number 15K15998).

REFERENCES

[1] Vanek, P., et al.: Convergence of Algebraic Multigrid Based on Smoothed Aggregation, Numerische Mathematik 2001, vol 88, pp.559-579, (2001).
[2] Nomura, N. et al.: Performance Analys is of SA-AMG Method by Setting Extracted Near-kernel Vectors, VECPAR2016 (2016).
[3] Information Technology Center: The University of Tokyo, http://www.cc.u-tokyo.ac.jp/.
[4] AMGS Library:http://hpcl.info.kogakui n.ac.jp/lab/software/amgs.
[5] 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).