Roughed Aggregation

Derrick Cerwinsky[1] and Craig C. Douglas[2]

Abstract

Roughed aggregation is a new AMG method which uses a rougher rather than a smoother to construct the restriction operator. In this case, the rougher used is conjugate gradient. With this simple modification there is a significant performance gain in some problems over smoothed aggregation.

1. Introduction

Aggregation methods represent a large class of coarseners. Using aggregation methods for AMG was first introduced by Vaněk in 1992 [5, 6, 9, 7].

In the years since its introduction many variants have evolved. Notable among these are the Adaptive Smoothed Aggregation (αSA) [1, 2], and the smooth error method [4]. In this paper, the roughed aggregation method is introduced as an alternative to the standard smoothed aggregation methods. Roughed aggregation was first introduced in my Ph.D. dissertation [3] and is under continuous improvement.

Roughed aggregation is a new AMG method which uses a rougher rather than a smoother to construct the restriction operator. In this case, the rougher used is conjugate gradient. With this simple modification there is a significant performance gain in some problems over smoothed aggregation.

2. Roughed Aggregation

Roughed aggregation works in three main phases: the selection of the aggregates, construction of the prolongation operator, and filtering the prolongation operator.

2.1. Definitions

This section introduces the notation used throughout the paper. As is common practice, let $\mathbf{A} = (a_{ij})$ be a $n \times n$ SPD matrix stiffness matrix.

[1]Department of Mathematics, University of Wyoming, dcerwins@uwyo.edu

[2]School of Energy Resources and Department of Mathematics, University of Wyoming, craig.c.douglas@gmail.com

Definition 1. *An aggregate is a collection of elements of* Ω. *The* j*th aggregate is denoted* $\mathcal{C}_j$.

Definition 2. *For a fixed* ε, $0 \leq \varepsilon < 1$, *define the strongly-coupled neighborhood of the point* i *as*

$$\mathcal{N}_i(\varepsilon) = \left\{ i : |a_{ij}| \geq \varepsilon \sqrt{a_{ii} a_{jj}} \right\}. \tag{1}$$

Definition 3. *The matrix* $\mathbf{D}$ *is defined to be the diagonal of the matrix* $\mathbf{A}$. *The filtered matrix* $\mathbf{A}^F = \left(a_{ij}^F\right)$ *is defined as,*

$$a_{ij}^F = \begin{cases} \left\{ \begin{array}{ll} a_{ij} & \text{if } j \in \mathcal{N}_i(\varepsilon) \\ 0 & \text{otherwise} \end{array} \right\} & \text{if } i \neq j \\ a_{ii} - \displaystyle\sum_{j=1, j \neq i}^{n} \left(a_{ij} - a_{ij}^F\right) & \text{otherwise.} \end{cases} \tag{2}$$

2.2. Selecting the Aggregates

Aggregation methods all use a common idea of grouping neighborhoods of points into aggregates. Each aggregate will become a coarse point on the coarse mesh. There are many variations of how this is done depending on the details of the problem being studied. The method presented here is based on the method in [5].

The aggregation works in three phases plus an initialization phase. In the initialization phase a set $\mathcal{R} \subset \Omega$ is constructed to track which nodes have been placed into aggregates as well as filter out any isolated nodes. That is, define the set

$$\mathcal{R} = \{ i : i \in \Omega, \mathcal{N}_i(\varepsilon) \neq \{i\} \}, \tag{3}$$

which is the set of all nodes in Ω that are not isolated.

Phase one creates an approximation of the aggregates. The first node $i \in \mathcal{R}$ for which $\mathcal{N}_i(\varepsilon)$ is disjoint from all other aggregates is called the jth aggregate. When a point is added to an aggregate, it is removed from $\mathcal{R}$. This phase continues until no more aggregates can be created in this way or when $\mathcal{R}$ is empty.

Phase two expands the aggregates to include points that were missed in phase one based on strong connections. Each $i \in \mathcal{R}$ is checked for strong connections to an aggregate. If such an aggregate exists, then i is added to that aggregate and removed from $\mathcal{R}$. If i is connected to multiple aggregates, it should be added to the one with the most strong connections. In the case of a tie, the choice is arbitrary, but should be consistent.

Phase three handles any remaining nodes that may have been ignored in the first two passes. In most cases, this phase will not need to be run. Any remaining points $i \in \mathcal{R}$ should be placed into aggregates based on partial strongly connected neighborhoods.

Once all the points have been aggregated, a prolongation operator is constructed.

2.3. Constructing the Prolongation Operator

The first step of constructing the prolongation operator is constructing a tentative operator $\tilde{\mathbf{P}}$. This operator will then use a single pass of a rougher for refinement. The rougher used here is conjugate gradient.

The tentative prolongation matrix $\tilde{\mathbf{P}}$ is defined entrywise for each of the aggregates $\mathcal{C}_i$.

$$\left(\tilde{\mathbf{P}}\right)_{ij} = \begin{cases} 1 \text{ if } i \in \mathcal{C}_j \\ 0 \text{ otherwise.} \end{cases} \tag{4}$$

The conjugate gradient step requires both the filtered matrix $\mathbf{A}^F$ from Definition 3, and the tentative prolongation matrix $\tilde{\mathbf{P}}$ from (4).

In this method, the prolongation operator is constructed columnwise. The ith column of $\mathbf{P}$ is the result of applying the CG method to $\mathbf{A}^F$ and the ith column of of the tentative prolongation operator. That is, using iterative conjugate gradient, solve for each $i = 1, \ldots, n$

$$\mathbf{A}^F \mathbf{P}_i = \tilde{\mathbf{P}}_i. \tag{5}$$

2.4. Filtering the Prolongation Operator

Once the prolongation operator is constructed it must be filtered to maintain sparsity of the system matrix on the coarse level [8]. There are many forms of filtering that can be used. The one presented in this section is a modification of the method in [9]. This form of filtering requires access to the null space of the matrix. Some alternate filtering methods are found in [1, 2, 4].

The method works by using the prolongation operator found in Section 2.3 and modifying it so that the algebraically rough elements are mapped to zero on the coarse grid.

Definition 4. *Let* $\mathbf{A}$ *be a* $n \times n$ *SPD matrix, and let* α *be a fixed positive number. The near null space of* $\mathbf{A}$ *is the span of the eigenvectors associated with the eigenvalues bounded below by* α*.*

Selecting the correct value for α is something of a challenge, as there does not seem to be any clear method for how this should be done. If α is too large, then not enough is being filtered in the coarsening step and the method will lose effectiveness. However, if α is too small, then extra computations are required that will slow the setup phase. In general an α which is too large is less detrimental then one which is too small.

Once α has been selected, the associated eigenvectors form the columns of the matrix $\mathbf{B}$. Then a matrix $\mathbf{U}$ is constructed. This matrix constitutes the update to the prolongation operator. $\mathbf{U}$ is initialized as

$$\mathbf{U} = \mathbf{D}^{-1} \mathbf{A}^F \mathbf{P}, \tag{6}$$

where $\mathbf{A}$ is the fine system matrix, $\mathbf{D}$ is the diagonal of $\mathbf{A}$, and $\mathbf{P}$ is the prolongation matrix in Section 2.3.

The matrix $\mathbf{U}$ is modified so that

$$\mathbf{U}^T\mathbf{B} = 0. \tag{7}$$

To this end, a Gram–Schmidt process is employed on the columns of $\mathbf{U}$ using the columns of $\mathbf{B}$ as a basis. Then the prolongation operator is modified by the update matrix

$$\mathbf{P}^* = \mathbf{P} - \mathbf{U}. \tag{8}$$

The final step to the algorithm is to impose the sparsity pattern of $\tilde{\mathbf{P}}$ from (4) on $\mathbf{P}^*$ and call this operator $\mathbf{P}$.

3. Numerical Results

This example is the 2D Laplace equation on a unit square domain with Dirichlet boundary conditions. A uniform square mesh is used with 1225 unknowns. AMGLab running in Matlab 7.9.2.529 on Windows 7 (64-bit) is used for the computations.

A three level V-cycle is employed, using 10 Gauss-Seidel smoothing steps at each level, and a direct solve on the coarsest level. At the end of each cycle, the residual is computed and recorded.

Cycle	*RoughedAggregation*
1	$5.298e+000$
2	$6.921e-001$
3	$3.108e-002$
4	$9.979e-004$
5	$6.817e-005$
6	$2.365e-006$
7	$1.087e-007$
8	$6.089e-009$
9	$2.344e-010$
10	$1.092e-011$
11	$7.806e-013$

While there is no proof yet, on all sample problems run to date, roughed aggregation consistently converged quicker than smoothed aggregation.

4. Conclusion

This paper introduced roughed aggregations, a new AMG method for coarsening. The method differs from existing aggregation methods by using a rougher rather than a smoother for the construction of the prolongation method. This new method has shown a performance increase over existing smoothed aggregation methods in sample problems.

Acknowledgments

This research was supported by the National Science Foundation grant EPS-1135483, Air Force Office of Scientific Research grant FA9550-11-1-0341, and Award No. KUS-C1- 016-04, made by King Abdullah University of Science and Technology (KAUST).

References

[1] M. Brezina, R. Falgout, S. MacLachlan, T. Manteuffel, S. McCormick, and J. Ruge. Adaptive smoothed aggregation (αsa). *SIAM J. SCI. COMP*, 25(6):1896–1920, 2004.

[2] M. Brezina, R. Falgout, S. MacLachlan, T. Manteuffel, S. McCormick, and J. Ruge. Adaptive smoothed aggregation (αsa) multigrid. *SIAM Review*, 47(2):317–346, 2005.

[3] Derrick Cerwinsky. *The Theory and Practice of Algebraic Multigrid Methods*. PhD thesis, University of Wyoming, 2012.

[4] E. Chow. An aggregation multilevel method using smooth error vectors. *SIAM Journal on Scientific Computing*, 27(5):1727–1741, 2006.

[5] P. Vaněk. Acceleration of convergence of a two level algorithm by smoothing transfer operators. *Appl. Math.*, 37:265–274, 1992.

[6] P. Vaněk. Fast multigrid solver. *Appl. Math.*, 40:1–20, 1995.

[7] P. Vaněk, J. Mandel, and M. Brezina. Algebraic multigrid by smoothed aggregation for second and fourth order elliptic problems. In N. D. Melson, T. A. Manteuffel, S. F. McCormick, and C. C. Douglas, editors, *Seventh Copper Mountain Conference on Multigrid Methods*, volume CP 3339, pages 721–735, Hampton, VA, 1996. NASA.

[8] Petr Vaněk, Marian Brezina, and Jan Mandel. Convergence of algebraic multigrid based on smoothed aggregation. *Numerische Mathematik*, 88:559–579, 2001. 10.1007/s211-001-8015-y.

[9] P. Vaněk, J. Mandel, and M. Brezina. Algebraic multigrid by smoothed aggregation for second and fourth order elliptic problems. *Computing*, 56:179–196, 1995.