Algebraic Multigrid (AMG) for saddle point systems from meshfree discretizations

K. H. Leem[1], S. Oliveira[2,*] and D. E. Stewart[3]

[1] *Applied Mathematics and Computational Sciences, The University of Iowa, Iowa City, IA 52242, USA*
[2] *Department of Computer Science, The University of Iowa, Iowa City, IA 52242, USA*
[3] *Department of Mathematics, The University of Iowa, Iowa City, IA 52242, USA*

SUMMARY

Meshfree discretizations construct approximate solutions to partial differential equation based on particles, not on meshes, so that it is well suited to solve the problems on irregular domains. Since the nodal basis property is not satisfied in meshfree discretizations, it is difficult to handle essential boundary conditions. In this paper, we employ the Lagrange multiplier approach to solve this problem, but this will result in an indefinite linear system of a saddle point type. We adapt a variation of the smoothed aggregation AMG method of Vaněk, Mandel & Brezina to this saddle point system. We give numerical results showing that this method is practical and competitive with other methods with convergence rates that are $\sim c/\log N$. Copyright © 2003 John Wiley & Sons, Ltd.

KEY WORDS: algebraic multigrid, meshfree methods, Lagrange multiplier method

1. Introduction

Meshfree methods produce approximate solutions of elliptic partial differential equations using basis functions constructed from particles, without the need for a conventional mesh. Of the many meshfree schemes, Reproducing Kernel Particle Methods (RKPM) [10] are used in this paper. For an overview of many of the different kinds of meshfree schemes, see [2]. In RKPM, solving a problem with essential boundary conditions is a difficult task because the basis functions it constructs do not satisfy the nodal basis property: $\Psi_i(\underline{x}_j) = 1$ if $i = j$ and zero if $i \neq j$. There has been some work on modifying the basis function to satisfy nodal basis property such as the singular basis function method. But the singular basis loses some smoothness at the basis nodes [9]. In [13], the essential boundary conditions were enforced by making the correction function, or kernel functions, vanish at the boundary nodes. An alternative approach to deal with essential boundary conditions is the Lagrange Multiplier Method, but this results in indefinite symmetric or unsymmetric semidefinite linear systems.

*Correspondence to: Department of Computer Science, The University of Iowa, Iowa City, IA 52242, USA

Contract/grant sponsor: This work was supported in part by the National Science Foundation; contract/grant number: DMS-9874015

We propose an Algebraic Multigrid (AMG) scheme for solving this indefinite system. AMG methods have been developed for solving linear systems posed on large, unstructured grids because they do not require geometric grid information, and thus are appropriate for meshfree discretizations. Classical AMG was targeted at linear systems with symmetric, positive definite M-matrices [5]. Recently, many new AMG approaches are developed to solve more general linear systems [6, 1]. In this paper we develop a variant of the smoothed aggregation method of Vaněk, Mandel and Brezina [17, 18] and adapted to the saddle point problem we obtained from the Lagrange Multiplier Method and meshfree discretizations.

In AMG methods, finding the set of coarse-grid points is always a difficult but crucial task. Assume we have a set of grid points $\Omega = \{1, 2, \cdots, n\}$. In many AMG methods, including classical AMG, the set of coarse-grid points Ω_c is obtained by partitioning Ω into a set of coarse-grid points Ω_c and a set of fine-grid points Ω_f such that $\Omega_c \cup \Omega_f = \Omega$ and $\Omega_c \cap \Omega_f = \emptyset$. Other versions of AMG assume that $\Omega_f = \Omega$ and the set of coarse-grid points Ω_c is obtained by aggregating fine-grid points in Ω_f. A simple interpolation operator for this coarsening scheme is based solely on aggregation. It is simple to implement, but usually shows slow convergence [12, 3]. We introduce a new AMG approach for our indefinite systems which is based on the latter type of coarsening, but our interpolation operator is constructed by using both aggregation and neighborhood information. Incorporating information about the boundaries, we designed an AMG method which works quite well for the indefinite system generated from meshfree discretizations with essential boundary conditions. When our AMG method is employed as a preconditioner for the well-known Krylov subspace method GMRES, our numerical results show that our AMG preconditioner is very effective, and scales well as the problem size increases.

This paper is organized as follows: RKPM is reviewed in Section 2. In Sections 3 and 4, we introduce our model problem and construct the symmetric indefinite linear system using a Lagrange multiplier method for handling essential boundary conditions. The general framework of AMG method is reviewed in Section 5. Our new AMG approach and its modification for indefinite linear systems are introduced in Sections 6 and 7. In Section 9, we briefly discuss preconditioning using our AMG preconditioner. Finally, the numerical results are presented in Section 10.

2. Reproducing Kernel Particle Methods (RKPM)

Many problems arising in computational mechanics are not well suited to conventional mesh-based methods, such as finite element and finite difference methods. The objective of meshfree methods is to eliminate at least part of this dependence by constructing the approximation entirely in terms of nodes or "particles" without the need for a triangulation or other mesh. It then becomes possible to solve large classes of problems which are awkward for mesh-based methods. In this section, Reproducing Kernel Particle Methods (RKPM) will be reviewed.

We denote $\{\underline{x}_I\}_{I=1}^{NP}$ as the set of points, which are called "*particles*", where NP stands for the number of these "particles". There is one basis function Ψ_I for each "particle", which is constructed as follows. We begin with a a so-called kernel function $\Phi_a(\underline{x} - \underline{x}_I)$, which should be a smooth function with compact support. These functions can be chosen for convenience or ease of computation. Typical choices include radial basis functions, and tensor products of B-splines: $\Phi_a(\underline{z}) = \Phi(z_1/a)\Phi(z_2/a)\cdots\Phi(z_d/a)$. The dilation parameter $a > 0$ is chosen to control the size of the support set, and usually $\Phi_a(\underline{z}) = \Phi_1(\underline{z}/a)$.

Since the approximation properties of the kernel functions $\{\,\Phi_a(\underline{x} - \underline{x}_I)\,\}_{I=1}^N$ are often not very good, we construct new basis functions $\{\,\Psi_I\,\}_{I=1}^N$ in order to have better approximation properties, while maintaining the smoothness and small support of the Φ_a functions. To achieve this, we put

$$\Psi_I(\underline{x}) = C(\underline{x}; \underline{x} - \underline{x}_I)\,\Phi_a(\underline{x} - \underline{x}_I) \tag{1}$$

where $C(\underline{x}; \underline{x} - \underline{x}_I)$ is a suitable correction function. This correction function is determined so that the basis reproduces a certain family of low-order polynomials $\underline{x}^\alpha$, $|\alpha| \leq p$ for some p. Note that here we are using a multi-index notation where $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_d)$ is a vector of non-negative integers, and $\underline{x}^\alpha = x_1^{\alpha_1}\, x_2^{\alpha_2} \cdots x_d^{\alpha_d}$. Also, let $|\alpha| = \alpha_1 + \alpha_2 + \cdots + \alpha_d$. The desired discrete "reproducing kernel" property is that whenever $|\alpha| \leq p$,

$$\underline{x}^\alpha = \sum_I \Psi_I(\underline{x})\,\underline{x}_I^\alpha \tag{2}$$

for all $\underline{x} \in \mathbf{R}^d$. We can construct the correction function $C(\underline{x}; \underline{x} - \underline{x}_I)$ by

$$C(\underline{x}; \underline{x} - \underline{x}_I) = H(\underline{x} - \underline{x}_I)^T b(\underline{x}) = H(\underline{x} - \underline{x}_I)^T \widetilde{M}(\underline{x})^{-1} H(0), \tag{3}$$

where $H(\underline{s}) = [\,\underline{s}^\alpha \mid |\alpha| \leq p\,]$, and $\widetilde{M}(\underline{x}) = \sum_{I=1}^{NP} H(\underline{x} - \underline{x}_I) H(\underline{x} - \underline{x}_I)^T\, \Phi_a(\underline{x} - \underline{x}_I)$. The basis functions $\Psi_I(\underline{x})$ are obtained from equation (1):

$$\Psi_I(\underline{x}) = H(\underline{x} - \underline{x}_I)^T \widetilde{M}(\underline{x})^{-1} H(0)\,\Phi_a(\underline{x} - \underline{x}_I). \tag{4}$$

Note that from this formula we can obtain the derivatives $(\partial\Psi_i/\partial x_i)(\underline{x})$, and also higher-order derivatives if necessary [10].

In our numerical experiment, we employed a cubic spline function as the kernel function. The one dimensional kernel function is defined as follows:

$$\Phi(z_i) = \begin{cases} \frac{2}{3} - 4z_i^2 + 4|z_i|^3 & \text{if } 0 \leq |z_i| \leq \frac{1}{2}; \\ \frac{4}{3} - 4|z_i| + 4z_i^2 - \frac{4}{3}|z_i^3| & \text{if } \frac{1}{2} \leq |z_i| \leq 1; \\ 0 & \text{otherwise.} \end{cases} \tag{5}$$

where $z_i = (x - x_i)/a$. The support of $\Phi_a(x - x_i)$ is $[x_i - a, x_i + a]$ and the dilation parameter a is used in Φ_a to control the size of the support. Then a multi-dimensional kernel function can be obtained from the one dimensional kernel function as follows:

$$\Phi_a(\underline{x} - \underline{x}_I) = \Pi_{i=1}^d \Phi((x_i - (x_I)_i)/a_i), \tag{6}$$

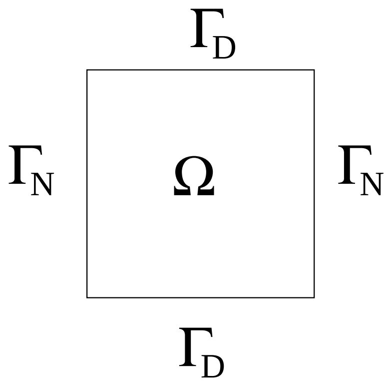
where a_i is the dilation parameter in the ith dimension of the kernel function $\Phi_a(\underline{x} - \underline{x}_I)$. In RKPM, the support of $\Phi_a(\underline{x} - \underline{x}_I)$ is selected to be a small area such that the basis function $\Psi_I(x)$ associated with node I intersects only a small group of surrounding support sets for computational efficiency. Also, note that $\Psi_I(x)$ does not satisfy the nodal basis property, $\Psi_I(\underline{x}_J) \neq \delta_{IJ}$. The smoothness of the shape function $\Psi_I(\underline{x})$ is determined by the smoothness of the kernel function $\Phi_a(\underline{x})$.

Some characteristics of the stiffness matrix generated by RKPM are as follows:

1. Sparseness: If the support sets of node I and J do not overlap, i.e., $supp(\Psi_I) \cap supp(\Psi_J) = \emptyset$, then $K_{IJ} = 0$. This implies that most of the matrix entries K_{IJ} are zero.
2. Symmetry: Interchanging I and J in the integral expression for K_{IJ} does not change the value calculated, so that $K_{IJ} = K_{JI}$ and thus the stiffness matrices are symmetric.

We will see that while the matrices are sparse, they are less sparse than the matrices from conventional finite element method.

Figure 1. Domain



3. Model Problem and Variational Formulation

Consider a second-order partial differential equation defined on a domain Ω contained in $\mathbf{R}^2$:

$$\begin{cases} -\nabla^2 u(\underline{x}) = f(\underline{x}), & \underline{x} \in \Omega \\ u(\underline{x}) = g(\underline{x}), & \underline{x} \in \Gamma_D \\ (\partial u/\partial n)(\underline{x}) = h(\underline{x}), & \underline{x} \in \Gamma_N. \end{cases} \tag{7}$$

where $\Gamma_D \cup \Gamma_N$ is the boundary of Ω. Assume that the domain Ω is $(0,1)\times(0,1)$ in 2 dimensions (see Fig 1).

Denote by $H^s(\Omega)$ the Sobolev space on Ω of order s. Consider the spaces

$$V := H^1(\Omega), \quad Q := H^{-1/2}(\Gamma_D).$$

with norms $\|\cdot\|_V$ and $\|\cdot\|_Q$ respectively. The weak form of (7) can written using the bilinear forms $a(\cdot,\cdot)$ on $V\times V$ and $b(\cdot,\cdot)$ on $V\times Q$ given by

$$\begin{aligned} a(u,v) &= \int_\Omega \nabla v^T \nabla u \; d\underline{x}, \\ b(u,q) &= \int_{\Gamma_D} uq \; d\underline{s}. \end{aligned} \tag{8}$$

Then the $a(\cdot,\cdot)$ defines a linear continuous operator $A : V \to V'$ by

$$\langle Au, v\rangle_{V',V} = a(u,v), \quad \forall v \in V, \; \forall u \in V,$$

and $b(\cdot,\cdot)$ defines $B : V \to Q'$, and its transpose $B^T : Q \to V'$ by

$$\langle Bv, q\rangle_{Q',Q} = \langle v, B^T q\rangle_{V',V} = b(v,q), \quad \forall v \in V, \; \forall q \in Q.$$

To handle essential boundary conditions, we employ a Lagrange Multiplier approach. Then the model problem can be rewritten as the following weak form: find $(u,\lambda) \in V \times Q$ such that

$$\begin{aligned} a(u,v) + b(v,\lambda) &= \tilde{f}(v), \; \forall v \in V, \\ b(u,q) &= \tilde{g}(q), \; \forall q \in Q, \end{aligned} \tag{9}$$

where $\tilde{f}(v) = \int_\Omega fv \; d\underline{x} + \int_{\Gamma_N} hv \; d\underline{s}$ and $\tilde{g}(q) = \int_{\Gamma_D} gq \; d\underline{s}$. The natural conditions on the data are that $f \in H^{-1}(\Omega)$, $h \in H^{-1/2}(\Gamma_N)$ and $g \in H^{1/2}(\Gamma_D)$. In order to find conditions implying the existence and uniqueness of solutions of this problem, the following Babuška-Brezzi condition should be satisfied [8, 15].

Remark 3.1. ***The Babuška–Brezzi condition.***
There exists a constant $\beta > 0$ such that

$$\sup_{v \in V,\, v \neq 0} \frac{|b(v,q)|}{\|v\|_V} \geq \beta \|q\|_{Q/\ker(B^T)} \qquad \textit{for all } q \in Q,$$

where $\|q\|_{Q/\ker(B^T)} = \inf_{q_0 \in \ker(B^T)} \|q - q_0\|_Q$.

It has been shown via the Babuška–Brezzi condition that our model problem (7) has a unique solution and is well-posed [7].

4. Meshfree Discretizations

In this section, we present the meshfree discretizations of continuous problem (9). We construct two finite dimensional subspaces,

$$V^h \subset V, \; Q^h \subset Q$$

where h denotes an appropriate mesh parameter. To approximate (9), we pose it over subspaces of V and Q respectively: find $(u^h, \lambda^h) \in V^h \times Q^h$ such that

$$\begin{aligned} a(u^h, v^h) + b(v^h, \lambda^h) &= \tilde{f}(v^h), \quad \forall v^h \in V^h, \\ b(u^h, q^h) &= \tilde{g}(q^h), \quad \forall q^h \in Q^h. \end{aligned} \tag{10}$$

Similarly, the discrete Babuška-Brezzi condition (Remark 4.1) should also be satisfied to guarantee the convergence of solutions of the discrete problem to the solution of the original problem [15].

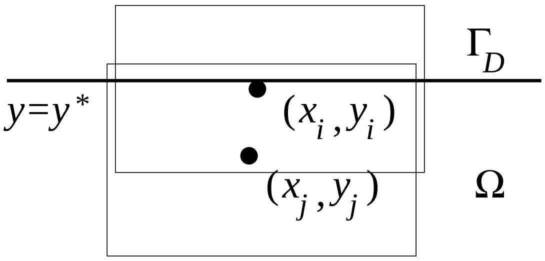
Remark 4.1. ***The discrete Babuška-Brezzi condition***
There exists a constant $\beta > 0$ (independent of h) such that

$$\sup_{v \in V_h,\, v \neq 0} \frac{|b(v^h, q^h)|}{\|v^h\|_{V^h}} \geq \beta \, \|q^h\|_{Q^h/\ker((B^h)^T)}. \tag{11}$$

The discrete Babuška–Brezzi condition (Remark 4.1) can easily fail for the meshfree method if our basis functions for $Q^h$are simply restrictions of the basis functions for V^h restricted to Γ_D. (This is what would happen if we took the standard penalty approach and took the penalty parameter to its limit.) Assume we have constructed the basis functions for Ω and the basis functions for the boundary are simply restrictions to Γ_D as shown in Figure 2. In this Figure, we assume $x_i \approx x_j$ and $y_i \not\approx y_j$. Suppose that the kernel functions

$$\Phi_i(x,y) = \Phi_r((x,y) - (x_i,y_i)) = (1/r^2)\Phi((x-x_i)/r)\Phi((y-y_i)/r), \; \forall i$$

Figure 2. Case when Babuška-Brezzi condition fails



On Γ_D, we have

$$\begin{aligned}
&\Phi_i(x,y^*) = (1/r^2)\Phi((x-x_i)/r)\Phi((y^*-y_i)/r) \\
&\cong (1/r^2)\Phi((y^*-y_i)/r)/\Phi((y^*-y_j)/r)\Phi((x-x_j)/r)\Phi((y^*-y_j)/r) \\
&= \text{const} \cdot \Phi_j(x,y^*)
\end{aligned} \tag{12}$$

This means the basis functions Ψ_i and Ψ_j can be nearly linearly dependent on Γ_D even though they are far from being linearly dependent as functions on Ω. With the near linear dependency, the discrete Babuška-Brezzi condition (Remark 4.1) fails, which means we can't prove convergence of the numerical solutions. As a remedy for this problem, we create new and independent kernel functions $\widetilde{\Phi}_i$'s for the boundary Γ_D separate from the Φ_i's, but using the same RKPM process.

Now, we want to build the meshfree linear system $Kx = F$, where K is a stiffness matrix and F is a load vector. The meshfree linear system $Kx = F$ generated is

$$\begin{bmatrix} A & B^T \\ B & 0 \end{bmatrix} \begin{bmatrix} u \\ \lambda \end{bmatrix} = \begin{bmatrix} c \\ d \end{bmatrix} \tag{13}$$

where $A_{ij} = \int_\Omega ((\nabla\Psi_i)^T \nabla\Psi_j + \Psi_i\Psi_j)\, d\underline{x}$, $B_{ij} = \int_{\Gamma_D} \widetilde{\Psi}_i \Psi_j d\underline{s}$, $c_i = \int_\Omega f\Psi_i\, d\underline{x} + \int_{\Gamma_N} h\Psi_i\ d\underline{s}$, and $d_i = \int_{\Gamma_D} g\widetilde{\Psi}_i\, d\underline{s}$. Note that stiffness matrix $K := \begin{bmatrix} A & B^T \\ B & 0 \end{bmatrix}$ in (13) is symmetric but indefinite, but the submatrix A is symmetric positive semi-definite.

5. Basic framework of AMG

We begin introducing the basic framework of AMG. In Sections 5 and 6, we only consider the symmetric positive definite linear system. Solving an indefinite linear system will be dealt with in Section 7. Consider a symmetric positive definite linear system $Au = b$

$$Au = b \tag{14}$$

where A is $n \times n$ matrix. Since AMG does not require the access of physical grids of problems, "grids" will mean sets of indexes of the variables. Hence the grid set for (14) is $\Omega = \{1, 2, \cdots, n\}$. The main idea of AMG is to remove the smooth error by coarse grid

Algorithm 1 AMG two-grid correction cycle:

$$v_f \leftarrow \mathrm{AMG}(v_f, b_f)$$

1. Repeat ν_1 times: $v_f \leftarrow v_f + R_f(b_f - A_f v_f)$ where R_f is the fine-grid smoother.
2. compute the fine-grid residual $r_f \leftarrow b_f - A_f v_f$, and restrict to the coarse grid by $r_c \leftarrow I_f^c r_f$.
3. Solve $A_c e_c = r_c$ for e_c (on coarse grid Ω_c).
4. Interpolate coarse-grid error to the fine grid by $e_f \leftarrow I_c^f e_c$ and correct the fine-grid approximation by $v_f \leftarrow v_f + e_f$.
5. Repeat ν_2 times: $v_f \leftarrow v_f + R_f(b_f - A_f v_f)$.

correction, where smooth error is the error not eliminated by relaxation on the fine grid, which also can be characterized by small residuals, i.e., $Ae \approx 0$.

In order to develop the multi-grid algorithm, we consider first a pair of grids. The coarse-grid level is denoted by c and fine-grid level by f. Equation (14) can be written as $A_f u_f = b_f$. Assume that we have defined a relaxation scheme, a set of coarse-grid points Ω_c, a coarse-grid operator A_c, and inter-grid transfer operators I_f^c (restriction) and I_c^f (interpolation). With this information, we can perform a two-grid AMG cycle as shown in Algorithm 1. Note that the operation $v_f \leftarrow v_f + R_f(b_f - A_f v_f)$ may be implemented as a linear iterative scheme such as Jacobi, Gauss–Seidel or SOR without necessarily breaking it down into the three steps of computing $b_f - A_f v_f$, multiplication by R_f, and addition to v_f.

Having defined the two-grid correction algorithm, we can define other multigrid AMG cycles. For example, to create V-cycle algorithm, we simply replace the direct solution of the coarse-grid problem with a recursive call to AMG on the next coarsest grid, except for the coarsest grid, where we use a direct solver.

In this paper we use the common approach of setting the restriction operator I_f^c to be the transpose of the interpolation operator I_c^f, i.e., $I_f^c = (I_c^f)^T$ and the coarse grid operator A_c is constructed from the fine grid operator A_f by the Galerkin approach:

$$A_c = I_f^c\, A_f\, I_c^f, \tag{15}$$

so that AMG satisfies the principle that the coarse-grid problem needs to provide a good approximation to fine-grid error in the range of interpolation [5].

Thus to specify a particular AMG algorithm we just need to specify for each coarse level the interpolation operator I_c^f and the smoother used. The interpolation operator for A is constructed according to the scheme in Sections 6 and adapted to the saddle point system in 7, while the smoother is described in Section 8. The resulting AMG method is then used as a preconditioner for GMRES, as discussed in Section 9.

6. Smoothed aggregation AMG

One of the main tasks in AMG consists of finding suitable coarsening strategies. Consider the symmetric positive matrix A in (14). A node in the graph of the matrix A is associated with each row. If matrix entry a_{ij} is nonzero, there is an edge from node i to node j in the graph.

Table I. Matrix data

(N_Ω, N_{Γ_D})	particle radius	$(NQ_\Omega, NQ_{\Gamma_D})$	# of Nonzeros	Condition #
(25, 10)	0.4	(400, 20)	513	2.02E+03
(100, 20)	0.2	(400, 40)	3828	1.92E+04
(400, 40)	0.1	(1600, 80)	17,408	1.20E+05
(1600, 80)	0.05	(6400, 160)	73,968	6.38E+05
(6400, 160)	0.025	(25600, 320)	304,688	—
(25600, 320)	0.0125	(102400, 640)	1,236,528	—

Consider the following choice of edge weight ω_{ij}:

$$\omega_{ij} = \int_\Omega (\Psi_i \Psi_j + (\nabla \Psi_i)^T \nabla \Psi_j)\, d\underline{x}, \tag{16}$$

which is same as each a_{ij} entry value. Note that this edge weight depends greatly on the size of overlapped regions of two particles i and j, not taking into account the size the particles. Thus, in order to obtain accurate edge weights, we need to normalize them as follows:

$$\varpi_{ij} = |\omega_{ij}| / \sqrt{\omega_{ii} \cdot \omega_{jj}}, \tag{17}$$

At this point we use a combinatorial clustering algorithm using the normalized edge weights ϖ_{ij} to construct a partition $\{C_1, C_2, \ldots, C_{N_c}\}$ of $\{1, 2, \ldots, n_f\}$. Each C_i is called a cluster. We employed two different strategies for forming clusters. The first was Heavy Edge Matching (HEM) based on ϖ. This is a greedy algorithm which picks the edge (i, j) with the largest value of ϖ_{ij} where i and j are unmatched nodes. Then the cluster or matched pair $\{i, j\}$ is added to the partition, and the process continues until there is no edge between unmatched nodes. This was used for constructing aggregation-based multilevel preconditioners in [14] for a class of eigenvalue problems. Unfortunately, attempting to improve the convergence rates using *smoothed* aggregation methods where the aggregation is done using HEM results in poor performance due to a severe loss of sparsity in the coarse-grid operators. (Pure aggregation methods do not suffer from this problem, but have worse convergence rates.)

An alternative strategy is given in Vaněk, Mandel and Brezina [17], which also uses the normalized edge weights ϖ_{ij}. The first step of their algorithm iterates through the nodes $i = 1, 2, \ldots, n_f$, creating clusters $\{\, j \mid \varpi_{ij} \geq \varepsilon \,\}$ for a given tolerance $\varepsilon > 0$, provided no node in $\{\, j \mid \varpi_{ij} \geq \varepsilon \,\}$ is already in a cluster. Nodes i and j where $\varpi_{ij} \geq \varepsilon$ are said to be "strongly connected". In the second step, unassigned nodes are assigned to a cluster from step one to which the node is strongly connected, if any. In the third step, the remaining nodes are assigned to clusters consisting of strong neighborhoods intersected with the set of remaining nodes.

Suppose we obtain the partition $\{C_1, C_2, \cdots, C_{N_c}\}$ of the set of fine-grid points $\Omega = \{1, 2, \cdots, n_f\}$ by aggregation. A simple interpolation operator, called E_c^f, can be defined by the full rank $n_f \times n_c$ matrix whose elements are equal to zero or unity where n_f is the number of fine-grid points and N_c the number of coarse-grid points. The kth column of E_c^f corresponding to a cluster C_k is given by

$$(E_c^f)_{ik} = \begin{cases} 1 & \text{if } i \in C_k, \\ 0 & \text{otherwise.} \end{cases} \tag{18}$$

For example, suppose the set of fine-grid points is $\Omega = \{1, 2, 3, 4\}$. Assume we have two clusters $C_1 = \{1, 2\}$ and $C_2 = \{3, 4\}$. Then the interpolation operator E_c^f for these aggregations is given by:

$$E_c^f = \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{bmatrix} \tag{19}$$

Again, the restriction operator E_f^c is defined by $(E_c^f)^T$. Using E_c^f as the interpolation operator and $E_f^c = (E_c^f)^T$ as the restriction operator with the coarse grid operator given by $A_c = E_f^c A_f E_c^f$ results in an AMG method we call a *pure aggregation* AMG method. These pure aggregation AMG methods are easy to use, but display slow convergence. Many different approaches have been introduced to obtain better convergence rates [3, 12, 17, 18]. In [17, 18], Vaněk, Mandel and Brezina developed a *smoothed aggregation* AMG scheme, which is quite close to what we do here. However, there are a number of pertinent differences.

In [17], the interpolation vectors are constructed by applying a damped Jacobi smoother to the matrix E_c^f (see p. 185 of [17]). To obtain our smoothed aggregation scheme we instead solve a "local" linear system to obtain an interpolation vector that interpolates a value for a cluster onto its neighborhood. Second, we apply the method to a saddle point problem, which is discussed in the next section.

For each cluster C_k, we have an associated interpolation vector p_k, which is the kth column of the interpolation operator $I_c^f = [p_1, p_2, \ldots, p_m]$ where m is the number of clusters.

We define the neighborhood N_k of C_k as

$$N_k = \{j \notin C \mid j \text{ connected to } i,\, i \in C_k\},$$

where j connects to i means that $a_{ij} \neq 0$. Denote $A_{IJ} = [\, a_{ij} \mid i \in I, j \in J\,]$ to be an $|I| \times |J|$ matrix where I and J are sets of nodes. For a cluster C_k, its local matrix L_{C_k} is given by

$$L_{C_k} = \begin{bmatrix} A_{C_k C_k} & A_{C_k N_k} \\ A_{N_k C_k} & A_{N_k N_k} \end{bmatrix}. \tag{20}$$

Then the corresponding interpolation vector p_k, the kth column of I_c^f, can be obtained by solving the local linear system below:

$$L_{C_k}\, p_k = \epsilon_{C_k}, \tag{21}$$

where ϵ_{C_k} is the vector given by

$$(\epsilon_{C_k})_i = \begin{cases} 1 & \text{if } i \in C_k, \\ 0 & \text{otherwise.} \end{cases} \tag{22}$$

Now we can set $I_c^f = [p_1, p_2, \ldots, p_m]$ where m is the number of clusters. Using $I_f^c = (I_c^f)^T$ and $A_c = I_f^c A_f I_c^f$ we have defined all components of the AMG method except for the smoothers. In the next section we discuss how we construct the interpolation, restriction, and coarse grid operators for the saddle point problem; in section 8, we describe the smoothers that we use for the saddle point problem.

7. Indefinite Linear System

Suppose the original linear system (13) on the fine grid is written as follows:

$$\begin{bmatrix} A_f & B_f^T \\ B_f & 0 \end{bmatrix} \begin{bmatrix} u_f \\ \lambda_f \end{bmatrix} = \begin{bmatrix} b_f \\ d_f \end{bmatrix} \tag{23}$$

We can construct the coarse-grid operator A_c from the fine-grid operator A_f by using the scheme given in the previous section. To construct the coarse grid operator B_c, we propose a separate coarsening process for the nodes on the boundary Γ_D. For this process, we need to know additional information from the matrix $\widetilde{\omega}$ given by

$$\widetilde{\omega}_{ij} = \int_{\Gamma_D} \widetilde{\Psi}_i \widetilde{\Psi}_j \, d\underline{s}. \tag{24}$$

Then we separately perform aggregations of elements on Γ_D based on the normalized edge weights of $\widetilde{\omega}$. Using the aggregations based on these edge weights, we construct a separate interpolation operator $\widetilde{I}_c^f$ for the nodes on the boundary Γ_D using the local matrices as described in the previous section. Using both I_f^c (the restriction operator for Ω) and $\widetilde{I}_c^f$ (the interpolation operator for boundary Γ_D), the coarse grid matrix B_c can be created using the Galerkin approach:

$$B_c = \widetilde{I}_f^c \, B_f \, I_c^f.$$

Thus

$$\begin{bmatrix} A_c & B_c^T \\ B_c & 0 \end{bmatrix} = \begin{bmatrix} I_f^c & \\ & \widetilde{I}_f^c \end{bmatrix} \begin{bmatrix} A_f & B_f^T \\ B_f & 0 \end{bmatrix} \begin{bmatrix} I_c^f & \\ & \widetilde{I}_c^f \end{bmatrix},$$

as we would expect from using a Galerkin approach.

8. A JOR smoother for indefinite systems

Another key element in AMG is to choose an appropriate relaxation scheme. As a relaxation scheme for our indefinite linear systems, the Jacobi over-relaxation (JOR) method described in [11] was employed as the relaxation scheme. Consider the splitting for matrix A in (23) as follows:

$$A = M - N. \tag{25}$$

Given an initial guess $(u^{(0)}, \lambda^{(0)})$, then the iterations can be defined by

$$\begin{bmatrix} M & B^T \\ B & 0 \end{bmatrix} \begin{bmatrix} u^{(k)} \\ \lambda^{(k)} \end{bmatrix} = \begin{bmatrix} N & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} u^{(k-1)} \\ \lambda^{(k-1)} \end{bmatrix} + \begin{bmatrix} b \\ d \end{bmatrix} \tag{26}$$

Note that to implement this scheme, we need to compute $\lambda^{(k)} = (BM^{-1}B^T)^{-1}[BM^{-1}(b + Nu^{(k-1)}) - d]$, which means solving the Schur complement system $BM^{-1}B^T\lambda^{(k)} = \cdots$. This system is symmetric and positive definite if M is, and the matrix $BM^{-1}B^T$ is easily computed, but lacks sparsity.

Remark 8.1 below (which is proved in [11]) guarantees the convergence of the iteration scheme (26).

Remark 8.1. *Assume that*

(a) A is a real symmetric nonnegative definite matrix,
(b) B^T is a real matrix with full column rank,
(c) A and B^T have no nontrivial null vectors in common,
(d) $A = D - L - L^T$ where D is a nonsingular diagonal matrix and L is a strictly lower triangular matrix.

Then the following iterative scheme (26) is convergent for the following choice of M and N:

$$M = \frac{1}{\sigma} D, \qquad N = \frac{1-\sigma}{\sigma} D + L + L^T$$

with $\sigma > 0$ chosen small enough that $2D/\sigma - A$ is positive definite.

Table II shows the σ values used in our numerical experiments. We test our AMG V-cycle on the linear system (23) from meshfree discretizations and the convergence results are shown in next section.

9. AMG preconditioning

We employ our AMG scheme as a preconditioner for General Minimal Residual (GMRES) method. To avoid preconditioning an indefinite system, our linear system (13) is modified to the following unsymmetric but semidefinite linear system:

$$\begin{bmatrix} A & B^T \\ -B & 0 \end{bmatrix} \begin{bmatrix} u \\ \lambda \end{bmatrix} = \begin{bmatrix} b \\ -d \end{bmatrix} \tag{27}$$

10. Numerical Results

In this section we present numerical results illustrating the convergence of our method to show its practicability and competitiveness with other methods. The matrices A and B are obtained from the RKPM basis generated with kernel functions $\Phi_a(\underline{x}) = B(x_1/a)\, B(x_2/a)$ where $a = 2\,h$ and B is the B-spline function given in 5. The centers chosen for the "particles" were on a uniform two-dimensional grid with grid spacing h. Basic data on the matrices generated for the saddle point problem are shown in Table I. Note that the matrices generated are substantially less sparse than the standard 5-point stencil for Laplace's equation. This is necessary for meshfree methods because of a "sufficient covering" condition which must be satisfied, and so the basis functions are substantially more "spread out".

We used the combinatorial part of the smoothed aggregation method in Vaněk, Mandel and Brezina (Algorithm 5.1 of [17]); we used the threshold $\varepsilon = 0.1$ for determining "strong neighborhoods"; also since the matrices generated by meshfree methods tend to have a large number of small entries, the neighborhoods N_k used in the construction of the interpolation operators were restricted by incorporating another threshold ϵ^*:

$$N_k = \{\, j \mid \overline{\omega}_{ij} \geq \epsilon^* \text{ and } i \in C_k \,\}.$$

The value used was $\epsilon^* = 0.01$. This helped to reduce the number of non-zeros in the coarse grid matrices.

Table II. Values of σ for JOR smoother

N	25	100	400	1600	6400	25600
σ	1.1	0.49	0.49	0.47	0.47	0.47

The numerical results are indexed by $N := N_\Omega$, the number of basis functions for the domain Ω. The values of N used were 25, 100, 400, 1600, 6400 and 25,600. The number of boundary basis functions were respectively 10, 20, 40, 80, 160, and 320. All computations were done on a Dell Optiplex GX260 with an Intel Pentium 4 CPU running at 3.1GHz and with 1GB of main memory under Red Hat Linux 9. Meschach [16] was used as a matrix library to speed software development. It was possible to solve the problem $N = 102,400$ on this machine. However, this was starting to thrash the machine.

For comparison, we also implemented a number of other preconditioners for GMRES: the identity matrix (which is equivalent to no preconditioner), the JOR smoother as a preconditioner, and an aggregation preconditioner (where the interpolation operator is E_c^f and not I_c^f). These were tested in the same way. The values of σused in the JOR smoother (and preconditioner) are given in Table II.

First, even though the system we are dealing with is symmetric and indefinite or unsymmetric and semi-definite, depending on the sign given to the second block equation, the smoothed aggregation AMG method developed here shows good multigrid-like convergence rates (see Figure 3 and Table III). The convergence rates computed were the $\log(1/(\text{convergence factor}))$ where the convergence factor is the geometric average of the ratios of successive residual norms over all but a few of the starting iterations. The residual norms for the first few iterates were ignored for these calculations as they showed initial transients that would distort the long-term convergence rate estimates. Note that the number of iterations needed for an accuracy of ϵ is asymptotically $\log(1/\epsilon)$ divided by the convergence rate.

The results for no preconditioner for N equal to 6400 and 25600 were excluded because this was taking excessive time. The results for the aggregation preconditioner was excluded for $N = 25600$ because of excessive memory usage due to a memory leak.

The asymptotic estimate of the behavior convergence rate for the smoothed aggregation AMG method of $\approx c/\log N$ was obtained by noting that the product of the observed convergence rate and $\log N$ lay in the range 16 to 20, with most of the products being no more than 17. Similarly, the convergence rate for the JOR preconditioner can be similarly estimated to be $\approx c\,N^{-\alpha}$ with $\alpha \approx 2/3$.

Convergence histories for the smoothed aggregation AMG preconditioner (shown as AMG), the JOR preconditioner, and the pure aggregation AMG preconditioner (labelled as the aggregation preconditioner) are shown in Figure 4.

In terms of the number of iterations, the smoothed aggregation preconditioner with the JOR relaxation is completely successful. However, the JOR preconditioner is simpler and faster to apply. The times taken by the different methods are shown in Figure 5. This shows the times per component of the solution against the dimension of the problem. This was done to highlight the difference between the performance observed for the different methods and the ideal behavior of $O(N)$ time to obtain a solution to the desired accuracy.

If we compare the times, then the advantage of the AMG method is reduced but still quite

Figure 3. Convergence rates for the different preconditioners (no preconditioner, JOR preconditioner, pure aggregation AMG preconditioner (shown as aggregation preconditioner), and the smoothed aggregation AMG preconditioner(shown as AMG preconditioner))

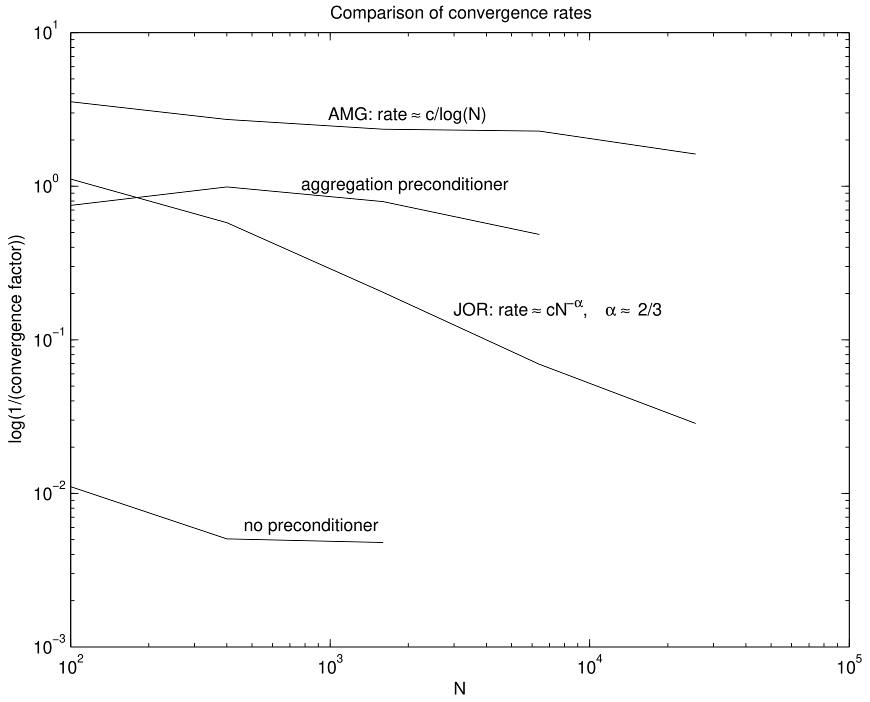
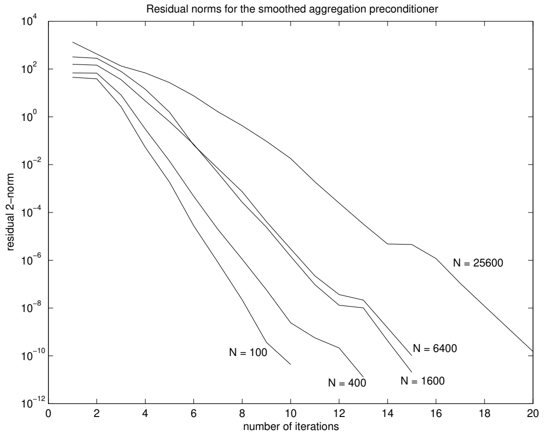


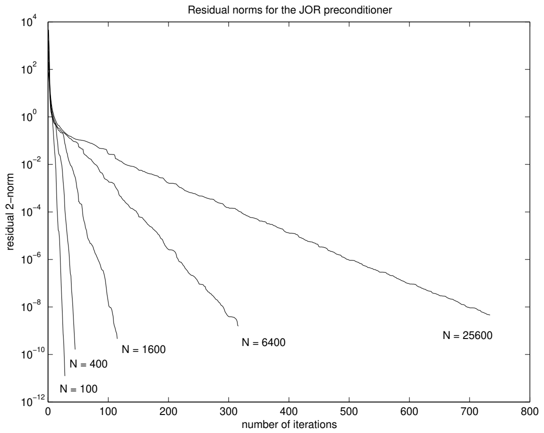
Table III. Convergence rates = log(1/(convergence factor)) as estimated from numerical data (no preconditioner, JOR preconditioner, smoothed aggregation AMG preconditioner (shown as AMG)), pure aggregation AMG preconditioner (shown as aggr'n).

	preconditioner			
N	none	JOR	AMG	agg'n
100	0.011	1.112	3.549	0.750
400	0.005	0.578	2.7181	0.989
1600	0.005	0.204	2.349	0.794
6400	—	0.069	2.282	0.485
25600	—	0.028	1.618	—

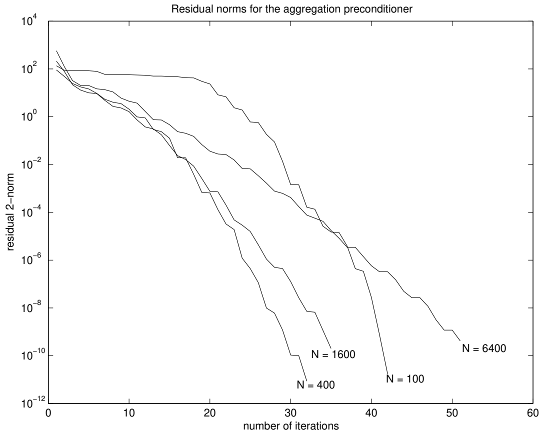
Figure 4. Convergence histories for all preconditioners except the identity matrix.



(a) Smoothed aggregation AMG preconditioner

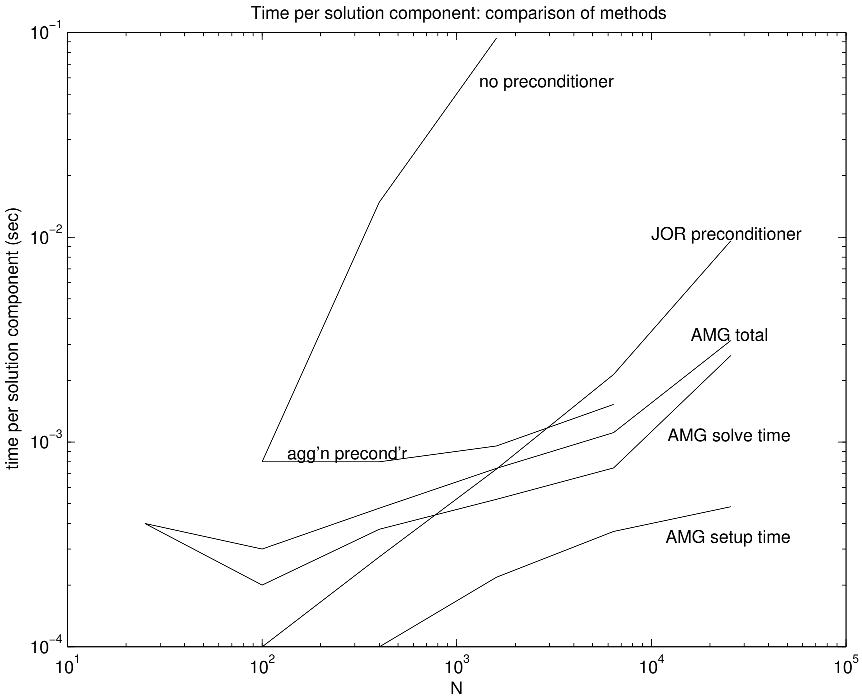


(b) JOR preconditioner



(c) Pure aggregation AMG preconditioner

Figure 5. Times taken by the different preconditioners to solve the problem (no preconditioner, JOR preconditioner, pure aggregation AMG preconditioner (shown as aggr'n precond'r), and the smoothed aggregation AMG preconditioner(shown as AMG))



significant, especially for large problems. The "break-even" point for the AMG method against the JOR method is at about $N = 1600$.

The times for single iterations are mainly controlled by the sparsity of the coarse-grid matrices. This information is given in Table IV. While the method has given good performance, it is clear that there is loss of sparsity occurring in the coarsening process. Improving this process is a matter for future work.

Acknowledgements

The authors would like to thank the National Science Foundation for their generous support, and also the referees for their suggestions which have improved the presentation of this paper.

REFERENCES

1. Bank RE and Smith RK. An algebraic multilevel multigraph algorithm. *SIAM J. Sci. Computing.* 2002; **23**:1572–1592.

Table IV. Size and sparsity of coarse-grid operators. Entries $x.xx(yy)$ denote $x.xx \times 10^{yy}$.
Under A we first list n where A is $n \times n$, and then the number of non-zeros in A.
Under B we first list m where B is $m \times n$, and then the number of non-zeros in B.

	level 0		level 1		level 2		level 3		level 4	
N	A	B	A	B	A	B	A	B	A	B
25	25 361	10 76	4 16	4 16						
100	100 3.36(3)	20 232	19 357	8 76						
400	400 1.64(4)	40 512	73 2.79(3)	14 198	18 324	6 81				
1600	1600 7.18(4)	80 1072	274 1.36(4)	28 378	55 3025	10 483				
6400	6400 3.00(5)	160 2192	1093 6.18(4)	54 910	828 1.92(5)	18 852	225 4.24(4)	18 1181		
25600	25,600 1.23(6)	320 4432	4247 2.53(5)	108 1659	3186 7.98(5)	36 1723	779 1.96(5)	36 3169	464 1.76(5)	36 5247

2. Belytschko T, Krongauz Y, Organ D, Fleming M, and Krysl P. Meshless methods: An overview and recent developments. *Computer Methods in Applied Mechanics and Engineering.* 1996; **139**:3-47.
3. Braess D. Towards algebraic multigrid for elliptic problems of second order. *Computing.* 1995; **55**:379–393.
4. Brandt A. Algebraic multigrid theory: The symmetric case. *Appl. Math. Comput.* 1986; **19**:23–56.
5. Brandt A, McCormick S, and Ruge J. Algebraic multigrid (AMG) for sparse matrix equations, in *Sparsity and its applications (Loughborough, 1983)*, Cambridge Univ. Press, Cambridge, 1985, pp. 257–284.
6. Brezina M, Cleary AJ, Falgout RD, Henson VE, and etc. Algebraic multigrid based on element interpolation (AMGe). *SIAM J. on Scientific Computing.* 2000; **22**:1570–1592.
7. Brezzi F and Fortin M., *Mixed and Hybrid Finite Element Methods*, vol. 15 of Springer Series in Computational Mathematics, Springer-Verlag, 1991.
8. Carey GF and Oden JT., *Finite Elements*, vol. II, Prentice Hall, Inc, 1983.
9. Chen JS, Han W, You Y, and Meng X. A reproducing kernel method with nodal interpolation property. *Int. J. for Numerical Methods Engineering.* 2003; **56**:935-960.
10. Chen JS, Pan C, Wu CT, and Liu WK. Reproducing kernel particle methods for large deformation analysis of non-linear structures. *Compt. Methods Appl. Engrg.* 1996; **139**:195–227.
11. Dyn N and Ferguson WE, Jr.. The numerical solution of equality-constrained quadratic programming problems. *Math. Comp.* 1983; **41**:165–170.
12. Fish J and Belsky V. Generalized aggregation multilevel solver. *Int. J. for Numerical Methods in Engineering.* 1997; **40**:4341–4361.
13. Gosz J and Liu WK. Admissible approximations for essential boundary conditions in the reproducing kernel particle method. *Comput. Mech.* 1996; **19**:120–135.
14. Holzrichter M and Oliveira S. A graph based Davidson algorithm for the graph partitioning problem. *International Journal of Foundations of Computer Science.* 1999; **10**:225–246.
15. Oden JT and Carey GF., *Finite Elements*, vol. IV, Prentice Hall, Inc, 1983.
16. Stewart DE and Leyk Z., *Meschach: Matrix Computations in C*, Australian National University, Canberra, 1994. Proceedings of the CMA, #32.
17. Vaněk P, Mandel J, and Brezina M. Algebraic multigrid by smoothed aggregation for second and fourth order elliptic problems. *Computing.* 1996; **56**:179–196. International GAMM-Workshop on Multi-level Methods (Meisdorf, 1994).
18. Vaněk P, Brezina M, and Mandel J. Convergence of algebraic multigrid based on smoothed aggregation. *Numer. Math.* 2001; **88**:559–579.