NUMERICAL LINEAR ALGEBRA WITH APPLICATIONS
Numer. Linear Algebra Appl. 2000; **00**:1–6 *Prepared using* nlaauth.cls *[Version: 2002/09/18 v1.02]*

Algebraic multigrid methods for constrained linear systems with applications to contact problems in solid mechanics

Mark F. Adams[1]

[1] *Sandia National Laboratories, MS 9417, Livermore CA 94551 (mfadams@ca.sandia.gov)*

SUMMARY

This paper develops a general framework for applying algebraic multigrid techniques to constrained systems of linear algebraic equations that arise in applications with discretized PDEs. We discuss constraint coarsening strategies for constructing multigrid coarse grid spaces and several classes of multigrid smoothers for these systems. The potential of these methods is investigated with their application to contact problems in solid mechanics. Copyright © 2000 John Wiley & Sons, Ltd.

KEY WORDS: algebraic multigrid, multigrid methods, saddle point problems, parallel multigrid, contact in solid mechanics

1. Introduction

This paper investigates the construction of algebraic multigrid methods for constrained linear systems—saddle point problems or KKT systems. Discretized saddle point problems generate systems of algebraic equations of the form:

$$Ax \equiv \begin{pmatrix} K & C^T \\ C & 0 \end{pmatrix} \begin{pmatrix} u \\ \lambda \end{pmatrix} = \begin{pmatrix} f \\ g \end{pmatrix} \equiv b \tag{1}$$

where K is a large, possibly singular, matrix from a discretized PDE, C is a matrix of constraint equations, u are the primal variables and λ are Lagrange multipliers. These systems arise in many applications from contact in solid mechanics to incompressible flow and problems in mathematical optimization. Several approaches have been developed for these problems, given a solver for the primal matrix K, namely: Uzawa's method [6], projection methods [20], and Schur elimination (or static condensation). These methods generally solve the primal and dual equations separately, and are attractive in that they can effectively use an existing solver

[*]Correspondence to: Sandia National Laboratories, MS 9417, Livermore CA 94551 (mfadams@ca.sandia.gov)
[†]This paper is authored by an employee(s) of the U.S. Government and is in the public domain.

Contract/grant sponsor: Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy; contract/grant number: DE-AC04-94AL85000

implementation for the primal system. Though these methods are useful and have their place in the repertoire of methods for solving these problems, they all suffer from some shortcomings (eg, not general, not scalable, not robust, or costly). This paper explores methods of solving these KKT systems "all-at-once" with algebraic multigrid (AMG) techniques.

Multigrid methods (eg, [14, 23, 12]) are among the most efficient iterative schemes for solving the linear systems associated with elliptic PDEs. Multigrid methods are well known to be theoretically optimal for H^1-elliptic operators, both scalar problems like Poisson's equation, and systems of PDEs like displacement finite element discretizations for elasticity [9]. Hence, multigrid is a natural choice for solving elasticity problems.

Multigrid has been applied to structured grid problems for decades [11]. In the past ten years, algebraic multigrid methods have been developed for unstructured problems as well. One such method, that has proven to be well suited to elasticity problems, is smoothed aggregation [24, 4, 18]. We have used a smoothed aggregation method extensively as the primal solver in the Uzawa algorithm for our engineering applications with contact. See Briggs et al., and the references therein, for an introduction to multigrid and to methods for common PDEs [12].

The remainder of this paper introduces a general framework for applying algebraic multigrid techniques to constrained systems of equations. This includes a general approach to the design of AMG methods for KKT systems in §2, and coarse grid spaces for constraints in §3, and several approaches for constructing smoothers for KKT systems in §4. These techniques are applied to contact problems in solid mechanics, with example problems from an industrial finite element application in §5; and we conclude in §6.

2. An AMG framework for constrained linear systems

A *multigrid*, or multilevel, method for a particular discretized PDE has two primary components: 1) the coarse grid spaces and 2) the smoother for each level. These coarse grid spaces are used to construct a *prolongation* operator P, which maps corrections from the coarse grid $l+1$ to the fine grid l and a *restriction* operator R that maps residuals from the fine grid to the coarse grid ($P = R^T$ for all methods considered here). Note, the columns of P are a discrete representation on the fine grid of the coarse grid functions.

The multigrid smoother is an inexpensive solver (eg, one iteration of Gauss-Seidel) used to reduce the error on each grid that is not effectively reduced by the coarse grid projection. Multigrid methods are designed to be applied recursively until the coarsest grid is small enough to be easily solved—accurately—usually with a direct solver.

Algebraic multigrid (AMG) generally refers to multigrid methods that construct the coarse grid spaces and operators from the fine grid operator, with little or no additional input from the application. The salient feature of AMG is the internal construction of the coarse grid operators, usually via a Galerkin process. For a fine grid l, given a restriction operator R and a prolongation operator P, the Galerkin coarse grid operator is constructed via the matrix triple product $A^{l+1} \leftarrow RA^lP$.

To apply a Galerkin process to the operator in equation 1, care must be taken to ensure that the coarse grid operator preserves the basic structure of the fine grid operator, to allow the process to be applied recursively. One solution to this problem is to construct the coarse grids as follows:

$$\begin{pmatrix} \hat{R}K\hat{P} & \hat{R}C^T\bar{P} \\ \bar{R}C\hat{P} & 0 \end{pmatrix} \Leftarrow \begin{pmatrix} \hat{R} & 0 \\ 0 & \bar{R} \end{pmatrix} \begin{pmatrix} K & C^T \\ C & 0 \end{pmatrix} \begin{pmatrix} \hat{P} & 0 \\ 0 & \bar{P} \end{pmatrix} \tag{2}$$

An attractive property of this formulation is that the coarse grid spaces are, at least formally, separated. The operator $\hat{P}$, the prolongator from the multigrid method for the primal part of the system, is a well studied topic for some classes of PDEs and is assumed to be available. Only the operator $\bar{P}$—the constraint coarse grid spaces need to be defined—the remainder of this section discusses an approach to the construction of $\bar{P}$.

3. Coarse grid spaces for Lagrange multipliers

The design of the coarse grid spaces is critical for any multigrid method. While many methods have been developed for some classes of PDEs, the design of coarse grid spaces for Lagrange multipliers has not been addressed by previous work to our knowledge. The simplest choice for $\bar{P}$, in equation 2, is the identity. The identity can be an appropriate choice for some types of constraints. In particular, constraints that have many nonzero values and are few in number are probably best served by the identity. An example of such a constraint is a kernel or rigid body mode constraint (eg, the solution values sum to zero for Poisson's equation with pure Neumann boundary conditions). Many important classes of constraints, however, require coarsening to avoid unacceptable complexity of the coarse grids and to avoid overdetermined, ill-conditioned, or unstable coarse grid operators.

The simplest type of spaces are piecewise constants—these result in a block diagonal $\bar{P}$ with columns that contain only zeros and ones, and all rows have only one non-zero entry (in practice these columns are normalized to have a 2-norm of 1.0). One attraction of this choice for $\bar{P}$ is that aggregation techniques can be employed to construct these spaces relatively easily. Although piecewise constant coarse grid spaces are not generally optimal they will suffice for this preliminary investigation of these algorithms.

3.1. Aggregation of Lagrange multipliers

Aggregation techniques are often used in AMG algorithms, but their application to constraint equations is not obvious because, for one, the matrix (or graph G) is not square. Though some aggregation strategies can be applied to non-square (or non-symmetric) matrices their behavior is not well understood in the context of aggregates for AMG algorithms. We use greedy aggregation strategies, based on maximal independents sets [1]; these methods are attractive because they are relatively easy to implement in parallel, but general graph partitioning algorithms can also be used for this purpose [4]. Regardless of the aggregation method, symmetric graphs are often desirable, thus, the first task is to define a suitable symmetric matrix or graph G.

3.2. An aggregation graph for contact problems in solid mechanics

A natural choice for a symmetric graph G, for the constraint equations in C, is $G = CC^T$. This choice is not suitable for contact constraints because, for one, CC^T is diagonal if the grids are aligned and common contact formulations are used. We have found in practice that the use of CC^T can lead to slow coarsening and to coarse grid constraint matrices that are nearly rank deficient due to equations being nearly identical under some conditions. A more general approach is the matrix inner product $G \leftarrow CYC^T$, where Y is a symmetric matrix.

We have investigated several choices for Y and have found $Y = \hat{P}\hat{P}^T$ to be very promising. This choice of Y has the effect of applying the coarsening of the primal equations to the columns of C^T (ie, $\hat{P}^T C^T$), and using this matrix inner product to construct the aggregation matrix: $G \leftarrow C\hat{P}\hat{P}^T C^T$.

3.3. Considerations for aggregation algorithms vis-a-vis quality of coarse grid spaces

This choice of G ($G = C\hat{P}\hat{P}^T C^T$) allows the angle between coarse grid constraint equations to be bounded directly. This is due to the construction of the constraint matrix on the coarse grid: $C^{l+1} \leftarrow \bar{P}^T C^l \hat{P}$ (from equation 2). The dot product, and hence the angle, between two coarse grid constraint equations is closely related to the edges of G that are cut in the partitioning. These dot products are the off diagonals of $C^{l+1}\left(C^{l+1}\right)^T = \bar{P}^T C^l \hat{P}\hat{P}^T \left(C^l\right)^T \bar{P} = \bar{P}^T G \bar{P}$, with appropriate scaling. The maximum edge cut, in the partitioning of G, is a bound on the cosine of the minimum angle between coarse grid constraint equations, because $\bar{P}$ is a simple aggregation operator. Thus, the standard graph partitioning principle, of minimizing the maximum edge cut can be employed to directly maximize the minimum angle between coarse grid constraint equations and thereby improve the stability of the coarse grid operator. Additionally it is well known from multigrid (or domain decomposition) theory that it is desirable to have large angles between the functions in the coarse grid space [22].

Multigrid theory also indicates that aggregates should be composed of strongly connected subdomains and it is common to employ heuristics in the aggregation algorithms to accomplish this [24]. Though, we are not aware of an analysis that proves that this is the case for the constraint equations, it is intuitive to believe that it is so. For instance, the contact constraints considered in this study are enforced with a simple point-on-surface technique in which a “slave” node on one contact surface is constrained to lie on a “master” quadrilateral face of the other contact surface. Additionally, our finite element application implements friction by fixing all three degrees of freedom of a slave node to the master surface, which results in three Lagrange multipliers that involve the five nodes in each contact constraint. These three equations are trivially orthogonal because each equation involves displacements in only one coordinate direction. It is natural to expect that it is advantageous for the aggregation strategy to interpolate these three equations separately (ie, interpolate 'x' direction constraints separately from 'y' and 'z' direction constraints, and so on). This criterion can be achieved by optimizing a standard graph partitioning metric—maximizing the edge weights within an aggregate—to produce strongly connected subdomains. Note, one complication with our choice of Y is that the $\hat{P}$ from many multigrid algorithms (including plain and smoothed aggregation) weakly couple these equations, that is, the information of their strict orthogonality is lost in G.

3.4. An aggregation algorithm for Lagrange multipliers

The choice of piecewise constant interpolation requires a disjoint decomposition of the nodes of the provided aggregation graph G. Our aggregation, or partitioning, strategy is based on a maximal independent set, defined as follows. An *independent set* is a set of vertices $S \subseteq V$ in a graph $G = (V, E)$, in which no two members of S are adjacent (ie, $\forall v, w \in S, (v, w) \notin E$); a *maximal independent set* (MIS) is an independent set for which no proper superset is also an independent set.

We use a variant of a parallel MIS algorithm [1]. Define the *edge weight* between two nodes i and j as $w_{ij} = \frac{|G_{ij}|}{\sqrt{G_{ii}G_{jj}}}$. Given a tolerance $\gamma < 1.0$, the graph G is modified by removing all edges $w_{ij} < \gamma$. Note, $\gamma = 0.2$ is used inn this study. This modified graph is used throughout the process; for simplicity, the symbol G is retained even though the graph as been modified by dropping these weak edges. Define the node weight $w(i)$ for node i, by $w(i) = \sqrt{\sum_j w_{ij}^2}$. The nodes are processed in the greedy MIS algorithm from the highest to the lowest node weight or alternatively these weights are used to assign the "random" numbers used in the parallel randomized MIS algorithm [15]. This defines a set S that is used to construct the aggregates of G.

Given this set S construct sets C_j with each node j in S (ie, $C_j = \{j\}$). Next, for all nodes $i \in G \setminus S$, add i to the aggregate $C_j | \max_{j \in S} w_{ij}$ (the MIS definition ensures that such a node, $j \in S$, exists for all nodes $i \in G \setminus S$). These sets C_j define the prolongator as follows

$$\bar{P}_{ij} = \begin{cases} |C_j|^{-\frac{1}{2}}, & \text{if } i \in C_j \\ 0, & \text{otherwise} \end{cases}$$

This defines the coarse grid spaces for the constraint equations used in this study and leaves only the definition of the KKT smoothers to provide a complete multigrid method.

4. Smoothers for constrained linear systems

Smoothers for constrained systems are a simpler matter than the coarse grid spaces, if for no other reason than previous work on this topic is informative. Schulz used structured geometric multigrid for optimization problems and developed a stable incomplete factorization for M-matrices, by ordering the constraints after all primal equations with which they interact [21]. Note, this concept—of ordering the constraints after all of the primal equations with which they interact—is a component of most solution methods for KKT systems. Segregated methods, or preconditioned Uzawa, have been investigated by Braess and Sarazin for the Stokes problem [10], and generalized by Zulehner [27]. Vanka used structured geometric multigrid for incompressible flow problems with a multiplicative Schwarz, constraint centric method [26].

We investigate variants of these three smoothers: 1) processor block incomplete factorizations (ILU) as preconditioners for a Chebyshev polynomial, 2) segregated (Braess) methods and 3) a constraint centric (Vanka) Schwarz methods. Note, all Chebyshev smoothers used are first order and the spectral radius of the matrix is estimated with a few iterations of conjugate gradient on the primal system $M^{-1}K$, where M^{-1} is the smoothing operator for K (see Adams et al. for details [5]).

The ILU preconditioner is the processor local level fill ILU method in PETSc with level fill of one [7]; the constraint equations are ordered last on each processor and the processor's primal equations are ordered with nested dissection (note, this is not a provably stable method). The following two sections describe the other two smoothers investigated here: segregated methods in §4.1, and a constraint centric Schwarz method in §4.2.

4.1. Segregated KKT smoothers

Consider a block factorization of the KKT system

$$\begin{pmatrix} K & C^T \\ C & 0 \end{pmatrix} = \begin{pmatrix} K & 0 \\ C & -S \end{pmatrix} \begin{pmatrix} I & K^{-1}C^T \\ 0 & I \end{pmatrix} \tag{3}$$

where $S \equiv CK^{-1}C^T$ is the Schur complement. This factorization suggests the use of the preconditioning matrix

$$B \equiv \begin{pmatrix} K & 0 \\ C & -S \end{pmatrix}, B^{-1}A = \begin{pmatrix} I & K^{-1}C^T \\ 0 & I \end{pmatrix} \tag{4}$$

and is attractive because the preconditioned system $B^{-1}A$ is well conditioned in that all of its eigenvalues are 1.0 and the Jordan blocks have size of at most two [17], if an exact solver is used for S^{-1} and K^{-1}.

The action of K^{-1} is approximated with a given smoother for the primal part of the system M^{-1}. The action of $\left(CK^{-1}C^T\right)^{-1}$ is approximated with an approximate Schur complement $Q = C^T D^{-1} C^T$, where D is an approximation to K (eg, D is the diagonal of K). The dual smoother is processor block Jacobi with Q (ie, the entries Q_{ij}, where equations i and j are on different processors, are dropped from Q and Q^{-1} is used as the dual smoother).

These choices for the segregated smoother result in the following algorithm. Give an initial guess λ^j and u^j:

$$u^{j+1} \leftarrow u^j + M^{-1}\left(f - Ku^j - C^T\lambda^j\right)$$

$$\lambda^{j+1} \leftarrow \lambda^j + Q^{-1}\left(Cu^{j+1} - g\right)$$

Note, this is equivalent to one iteration of the preconditioned inexact Uzawa algorithm without regularization [13].

This smoother can be symmetrized with a second application of the primal smoother as discussed by Bank, Welfert and Yserentant [8], as follows,

$$\hat{u}^{j+1} \leftarrow u^j + M^{-1}\left(f - Ku^j - C^T\lambda^j\right)$$

$$\lambda^{j+1} \leftarrow \lambda^j + Q^{-1}\left(C\hat{u}^{j+1} - g\right)$$

$$u^{j+1} \leftarrow \hat{u}^{j+1} - M^{-1}\left(C^T\left(\lambda^{j+1} - \lambda^j\right)\right)$$

This symmetrized form is used in this study.

4.2. A constraint centric Schwarz KKT smoother

This section investigates a constraint centric overlapping Schwarz method. This method is similar to that used by Vanka [26], in that overlapping Schwarz subdomains are constructed from a non-overlapping decomposition of the constraints.

4.2.1. Constraint centric subdomains Vanka uses one constraint per subdomain. We use much larger aggregates—all of the constraint equations on each processor. Each of the sets of constraint equations is augmented with all of the primal equations that interact with any of the constraint equations (this set of equations is used to define the extraction operator R_i in §4.2.2). Thus, each subdomain is a KKT system and can be interpreted in physical terms as the matrix resulting from applying Dirichlet boundary conditions to all of the nodes in the problem that are not touched by any of the constraint equations in the aggregate. These subdomain solves are stable because each subdomain is a well posed KKT system and thus a factorization, with the constraints ordered last, is well defined and does not require pivoting. Thus, a non-overlapping decomposition of the constraint equations is used to construct an overlapping additive Schwarz component (B_d in §4.2.2).

4.2.2. Constraint centric smoother As with the segregated smoother, a smoother for the primal part of the system is assumed to be available and is denoted, in matrix form, as M^{-1}. The primal part (domain) of the smoother is defined by $B_p = R_p^T M^{-1} R_p$ where $R_p = \begin{bmatrix} I & 0 \end{bmatrix}$. The dual part of the smoother is defined by $B_d = (B_1 + B_2 + \cdots + B_k)$, where $B_i = \tilde{R}_i^T \left(R_i K R_i^T \right)^{-1} R_i$, for each subdomain i. The matrices R_i and $\tilde{R}_i$ are extraction operators (matrices with only ones and zeros) as defined in §4.2.1. $\tilde{R}_i$ is block diagonal, with the proper node ordering, but R_i is not because of overlap in the primal equations. That is, $\tilde{R}_i$ is constructed by dropping terms in R_i that require communication and results in a non-overlapped update of the solution.

Thus, an additive formulation is used for ease of parallelization, and the solution for the primal variables are updated only from the local solve on each processor to negate the effects of "overshooting" the solution in the overlap region. That is, each subdomain solve first restricts the residual for the entire overlapping domain (thus requiring inter-processor communication), then performs the local subdomains solve (with a factorized matrix that includes rows from other processors), and finally only local solution values are updated and therefore partial solution values, that would otherwise be communicated in a full overlapping additive Schwarz formulation, are discarded.

The constraint centric overlapping Schwarz smoother has two subdomains—the pure primal part B_p and the dual part B_d—and is composed in both additive and multiplicative variants. Note, the primal smoother is not used by Vanka because incompressible flow constraints touch all primal variables and thus, the primal variables are smoothed with the dual smoother. The multiplicative constraint centric Schwarz smoother, with an initial guess x_0 and right hand side b, is defined as

$$\hat{x} \leftarrow x_0 + (B_p + B_d - B_d A B_p)(b - A x_0) \tag{5}$$

that is, standard multiplicative Schwarz with two subdomains. The additive variant is defined as

$$\hat{x} \leftarrow x_0 + (B_p + B_d)(b - A x_0) \tag{6}$$

that is, standard additive Schwarz with two subdomains. The additive variant is used as a preconditioner for an appropriately damped iterative scheme—first order Chebyshev polynomials [5]. The multiplicative variant is used with a multiplicative primal smoother—parallel symmetric Gauss-Seidel [3].

5. Numerical results with contact problems in solid mechanics

This section investigates the effectiveness of the algebraic multigrid methods for constrained systems (AMG/KKT) developed herein with two problems in solid mechanics. The nonlinear quasi-static solid mechanics application *Adagio*, from Sandia National Laboratories [16], is used for these numerical studies. The solver algorithms are implemented in the linear solver package *Prometheus* [19], which is built on the parallel numerical package *PETSc* [7]. Prometheus provides three algebraic multigrid solvers for the primal system: 1) smoothed aggregation, 2) plain aggregation and 3) an unstructured geometric multigrid method [2]. Because the smoothers are not, in general, symmetric and the preconditioned systems are not in general positive GMRES is used as the solver—preconditioned with one iteration of V-cycle AMG/KKT. The Uzawa solver uses conjugate gradient (CG), preconditioned with one multigrid V-cycle, as the primal solver. All solves use one pre and post smoothing step, with an exact solver for the coarsest grid.

We investigate our AMG/KKT method with four KKT smoothers: 1) ILU, 2) the segregated smoother with an additive primal smoother, and both the 3) additive and 4) multiplicative variant of our constraint centric Schwarz method. The ILU preconditioner is the processor local, level fill, ILU method in PETSc with a level fill of one.

The performance of the AMG/KKT methods are compared to a highly refined Uzawa implementation in Prometheus. The augmentation factor for the regularization required with Uzawa's method has been hand optimized to provide the best performance and thus, represents our best effort to solve these systems with Uzawa. Note, Adagio's implementation of friction results in fixing the slave node on the corresponding master surface, resulting in three Lagrange multipliers per contact interaction, before boundary conditions are imposed. The two test problems investigated here are run with friction.

5.1. The "circles" problem

The first test problem, known as the circles problem, is a disk within a ring within a circular cutout as shown in Figure 1. The middle ring segment has an elastic modulus 10^2 times that

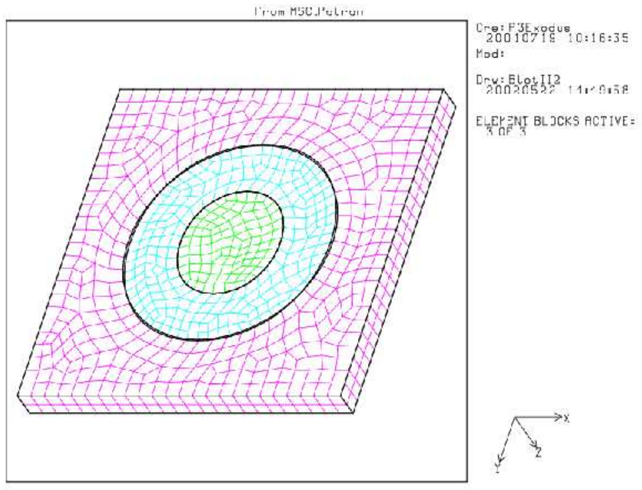

Figure 1. Adagio's "circles" problem

of the “soft” material of the central disc and outer cutout plate. All materials have a Poisson’s ratio of 0.3. This problem has two layers of hexahedral elements through the thickness, 7236 primal equations (minus the Dirichlet boundary conditions) and 180 point on surface contact interactions with a total of 420 Lagrange multipliers. The sliding boundary conditions on the top and bottom surface result in the smoothed aggregation algorithm having nearly the same semantics as the plain aggregation algorithm due to the presence of Dirichlet boundary conditions on all elements (the smoothed aggregation algorithm specifies that equations that are modified by the Dirichlet boundary conditions are not used in the smoothing [25]). Thus, we use plain aggregation for this problem.

Two multigrid levels are used and only the first linear solve is investigated. The problem is run on eight Sun processors and PETSc achieves about 400 Mflops/second in the matrix vector product on the fine grid matrix. The systems are solved with a tolerance of 10^{-12}, ie, convergence is declared when the solution $\hat{x}$ satisfies $\|b - A\hat{x}\|_2 < 10^{-12}\|b\|_2$.

Figure 2 shows the residual history, as a function of time (measured in units of the time for one matrix vector product on the fine grid), of the four smoothers in the AMG/KKT algorithm, of the Uzawa solver, and of the additive constraint centric smoother used as the preconditioner. This data shows that the KKT smoother used alone (as the GMRES preconditioner) is

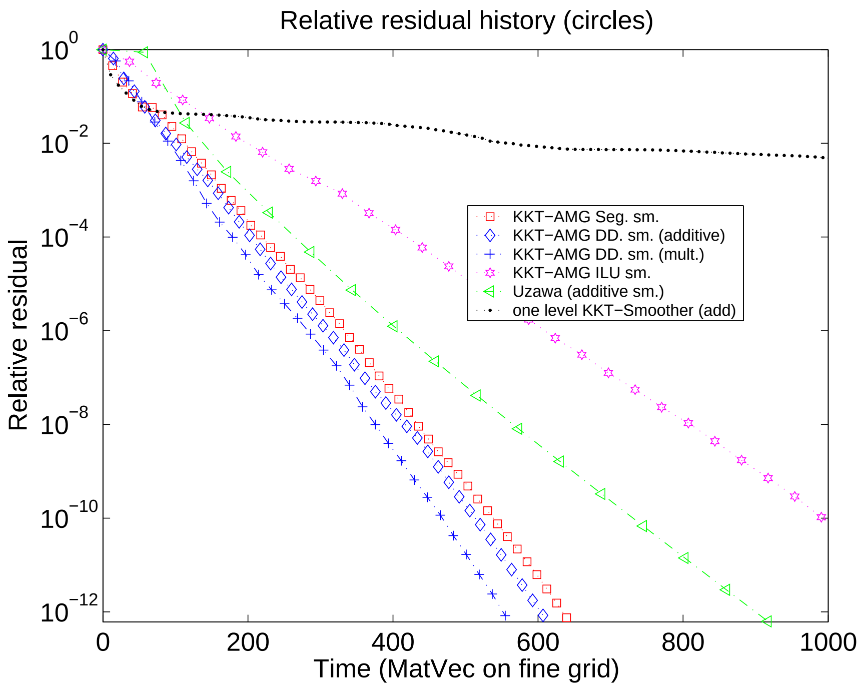

Figure 2. Residual vs. Time (in matrix vector products on the fine grid) for the “circles” problem

stagnating as is expected because it reduces the high frequency error effectively but is slow at reducing the low frequency content of the error. The AMG/KKT method with constraint centric smoothers, and the segregated smoother, solve the problem in about two thirds the time of the Uzawa solver. The ILU smoother is noticeably slower than the other methods.

5.1.1. Scalability issues All four of the AMG/KKT smoothers use the processor subdomains, in a domain decomposition method, and are hence less powerful as more processors are used for a given problem. This is not necessarily a problem, because smoothers are only responsible for the high frequency content of the error which can be reduced effectively with small Schwarz subdomains. To determine if this is indeed the case in practice, the iteration counts for this problem on 1, 2, 4 and 8 processors are measured. Table I shows the iteration counts for the Uzawa solver, the AMG/KKT preconditioner with the constraint centric Schwarz (CCS) smoother, both the additive and multiplicative variants, the segregated smoother, and the ILU smoother. This data shows that the smoothers are indeed essentially invariant to the number

Table I. AMG/KKT iteration counts vs. number of processors

Smoothers	1	2	4	8
Segregated	51	57	52	47
CCS (additive)	47	43	47	42
CCS (multiplicative)	35	34	37	31
ILU	30	33	35	32
Uzawa	100	122	127	106

of processors (ie, the size of the subdomains) in this range of processors.

5.2. *The forging problem*

The second test problem is shown in Figure 3. This problem is from a simulation of a forging

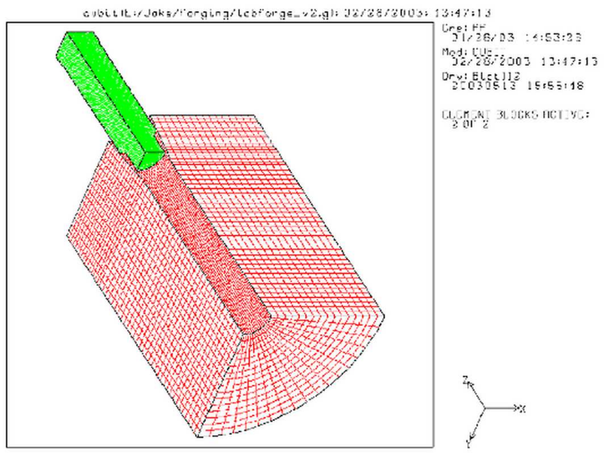

Figure 3. Adagio "forge" problem

process and has about 127K degrees of freedom and 64 constraint equations after the first five "time" steps. This is a nonlinear pseudo-static analysis with five steps, with the first contact occurring in the third step.

The forge problem is run on 16 processors of an SGI Origin 2000 and the matrix vector product on the fine grid attains about 1900 Mflops/sec. Adagio uses a non-linear CG method that can be preconditioned with a linear solver, Prometheus is used in this case with a residual tolerance of 10^{-2}. Three levels of smoothed aggregation multigrid are used. Table II shows the total ("end-to-end") time for the simulation along with the time for the solution phase (with the number of linear solves) and the solver setup phases. This data shows that the ILU

Table II. Solve and run times (sec) for the forge problem

Smoothers	end-to-end	setup	solve (# of solves)
Segregated	1060	259	293 (309)
CCS (additive)	1140	262	383 (306)
CCS (multiplicative)	910	257	182 (306)
ILU	2096	607	1025 (304)
Uzawa	1057	238	355 (304)

smoother is not competitive and that the segregated and constraint centric Schwarz smoothers are slightly faster than the Uzawa solver. In particular, the solve time for AMG/KKT with the multiplicative constraint centric Schwarz smoother is about one half of that of the Uzawa solver.

5.2.1. Scalability issues One measure of scalability for a multigrid method is to test grid independence by running the same problem with different number of levels. This is best done with versions of the problem with different levels of resolution. We do not have scaled version of our test problems and so we must make do with running the same problem with different numbers of levels. That is, the coarsening process is stopped with two, three and four levels. Note, the two level solve, with only one coarse grid, is very slow due to the large coarse grid operator that is factored and repeatedly solved in the multigrid process, the convergence rate (iteration count) provides a base case with which to compare the rates using more coarse grids. Table III shows the number of GMRES iterations for the first linear solve with Lagrange multipliers in the forging problem (and the total number of CG iteration in the inner loop of the Uzawa solver).

This data shows some growth in the iteration counts with the additive constraint centric Schwarz smoother. The segregated and multiplicative constraint centric Schwarz smoothers, however, demonstrate grid independence. This data is encouraging, though we do not see this as conclusive evidence of the grid independence of these methods.

Table III. Iteration counts of first linear solve of the forging problems

	Iterations (1st solve)				dof on each level (approx.)	
Levels	Uzawa	CCS (add.)	CCS (mult.)	Segregated	primal	Lagrange mult.
2	29	7	4	8	7872	17
3	29	8	5	9	276	5
4	28	10	4	8	30	2

6. Conclusions

We have presented a general framework for applying algebraic multigrid techniques to constrained systems of linear algebraic equations that shows promise as an effective approach to designing solvers for saddle point problems. This includes a novel method for automatically constructing the coarse grid spaces for constraint equations. We have developed one AMG method within this framework, along with three classes of smoothers, and applied them to contact problems in solid mechanics.

Two of the smoothers, the segregated and constraint centric Schwarz methods, are at least competitive to a well optimized Uzawa solver on two test problems. The third smoother, ILU, is not competitive—but we have not explored the design space of ILU methods, to a significant degree, and hence ILU methods should not be discarded from consideration in our opinion. We have demonstrated that our framework for applying algebraic multigrid techniques to constrained linear systems has the potential of supporting fast, scalable solvers for this class of constrained linear systems.

Though we have demonstrated the potential of these methods there are many areas of future work on this topic. First, the convergence rates, that we observe, are not as fast as the convergence rate in the primal solves in the Uzawa iterations. We believe that this is achievable and this issue deserves further study. In particular, the damping for the additive KKT smoothers deserves further investigation.

Piecewise constant coarse grid spaces are well known to be suboptimal for the primal equations and it is natural to expect this is the case for the constraint equations as well, though we are not aware of any analysis on this subject. Thus, developing smoothing techniques for the constraint coarse spaces will likely be necessary for ultrascalability. It is unlikely that these techniques can be made as robust as the Uzawa algorithm (which we have never seen fail), but further investigation is needed to asses the robustness of these methods and address any deficiencies.

And finally, we see problems that are currently intractable, such as challenging constrained optimization problems with PDEs, as potentially benefiting from the techniques discussed here.

ACKNOWLEDGEMENTS

The author would like to thank the Adagio development team at Sandia National Laboratories for providing the test problems in this study, and the referees and editor for their thoughtful and constructive criticisms.

REFERENCES

1. M. F. Adams, *A parallel maximal independent set algorithm*, in Proceedings 5th Copper Countain Conference on Iterative Methods, 1998.
2. ———, *Parallel multigrid solvers for 3D unstructured finite element problems in large deformation elasticity and plasticity*, International Journal for Numerical Methods in Engineering, 48 (2000), pp. 1241–1262.
3. ———, *A distributed memory unstructured Gauss–Seidel algorithm for multigrid smoothers*, in ACM/IEEE Proceedings of SC2001: High Performance Networking and Computing, Denver, Colorado, November 2001.
4. ———, *Evaluation of three unstructured multigrid methods on 3D finite element problems in solid mechanics*, International Journal for Numerical Methods in Engineering, 55 (2002), pp. 519–534.
5. M. F. Adams, M. Brezina, J. J. Hu, and R. S. Tuminaro, *Parallel multigrid smoothing: polynomial versus Gauss–Seidel*, J. Comp. Phys., 188 (2003), pp. 593–610.
6. K. Arrow, L. Hurwicz, and H. Uzawa, *Studies in nonlinear programming*, Stanford University Press, 1958.
7. S. Balay, W. D. Gropp, L. C. McInnes, and B. F. Smith, *PETSc 2.0 users manual*, tech. rep., Argonne National Laboratory, 1996.
8. R. E. Bank, B. D. Welfert, and H. Yserentant, *A class of iterative methods for solving mixed finite element equations*, Numer. Math., 56 (1990), pp. 645–666.
9. D. Braess, *On the combination of the multigrid method and conjugate gradients*, in Multigrid Methods II, W. Hackbusch and U. Trottenberg, eds., Berlin, 1986, Springer–Verlag, pp. 52–64.
10. D. Braess and R. Sarazin, *An efficient smoother for the Stokes problem.*, Appl. Numer. Math., 23 (1997), pp. 3–20.
11. A. Brandt, *Multi-level adaptive solutions to boundary value problems*, Math. Comput., 31 (1977), pp. 333–390.
12. W. L. Briggs, V. E. Henson, and S. McCormick, *A multigrid tutorial, Second Edition*, SIAM, Philadelphia, 2000.
13. H. C. Elman and G. H. Golub, *Inexact and preconditioned Uzawa algorithms for saddle point problems*, SAIM J. Numer Anal., 31 (1994), pp. 1645–1661.
14. W. Hackbusch, *Multigrid methods and applications*, vol. 4 of Computational Mathematics, Springer–Verlag, Berlin, 1985.
15. M. Luby, *A simple parallel algorithm for the maximal independent set problem*, SIAM J. Comput., 4 (1986), pp. 1036–1053.
16. J. A. Mitchell, A. S. Gullerud, W. M. Scherzinger, R. Koteras, and V. L. Porter, *Adagio: non-linear quasi-static structural response using the SIERRA framework*, in First MIT Conference on Computational Fluid and Solid Mechanics, K. Bathe, ed., Elsevier Science, 2001.
17. M. F. Murphy, G. H. Golub, and A. J. Wathen, *A note on preconditioning for indefinite linear systems*, SIAM J. Sci. Comput., 21 (2000), pp. 1969–1972.
18. G. Poole, Y. Liu, and J. Mandel, *Advancing analysis capabilities in ANSYS through solver technology*, Electronic Transactions in Numerical Analysis, 15 (2003), pp. 106–121.
19. *Prometheus*. www.cs.berkeley.edu/∼madams.
20. P. Saint-Georges, Y. Notay, and G. Warzee, *Efficient iterative solution of constrainted finite elemenet analyses*, Comput. Methods Appl. Mech. Engrg., 160 (1998), pp. 101–114.
21. V. Schulz, *Incomplete indefinite decompositions as multigrid smoothers for KKT systems*, ISNM, 133 (1999), pp. 257–266.
22. B. Smith, P. Bjorstad, and W. Gropp, *Domain Decomposition*, Cambridge University Press, 1996.
23. U. Trottenberg, C. Oosterlee, and A. Schüller, *Multigrid*, Academic Press, London, 2001.
24. P. Vaněk, M. Brezina, and J. Mandel, *Convergence of algebraic multigrid based on smoothed aggregation*, Numerische Mathematik, 88 (2001), pp. 559–579.
25. P. Vaněk, J. Mandel, and M. Brezina, *Algebraic multigrid by smoothed aggregation for second and fourth order elliptic problems*, in 7th Copper Mountain Conference on Multigrid Methods, 1995.
26. S. Vanka, *Block-implicit multigrid solution of Navier-Stokes equations in primitive variables*, J. Comp. Phys., (1986), pp. 138–158.
27. W. Zulehner, *A class of smoothers for saddle point problems*, Computing, 65 (2000), pp. 227–246.