MULTILEVEL ADAPTIVE AGGREGATION FOR MARKOV CHAINS, WITH APPLICATION TO WEB RANKING

H. DE STERCK*‡, THOMAS A. MANTEUFFEL†§, STEPHEN F. MCCORMICK†¶, QUOC NGUYEN*††, AND JOHN RUGE†‖

Abstract. A multilevel adaptive aggregation method for calculating the stationary probability vector of an irreducible stochastic matrix is described. The method is a special case of the adaptive smooth aggregation and adaptive algebraic multigrid methods for sparse linear systems, and is also closely related to certain extensively studied iterative aggregation/disaggregation methods for Markov chains. In contrast to most existing approaches, our aggregation process does not employ any explicit advance knowledge of the topology of the Markov chain. Instead, adaptive agglomeration is proposed that is based on strength of connection in a scaled problem matrix, in which the columns of the original problem matrix at each recursive fine level are scaled with the current probability vector iterate at that level. Strength of connection is determined as in the algebraic multigrid method, and the aggregation process is fully adaptive, with optimized aggregates chosen in each step of the iteration and at all recursive levels. The multilevel method is applied to a set of stochastic matrices that provide models for web page ranking. Numerical tests serve to illustrate for which types of stochastic matrices the multilevel adaptive method may provide significant speedup compared to standard iterative methods. The tests also provide more insight into why Google's PageRank model is a successful model for determining a ranking of web pages.

Key words. multilevel method, adaptive aggregation, Markov chain, stationary probability vector, web ranking

AMS subject classifications. 65C40 Computational Markov chains, 60J22 Computational methods in Markov chains, 65F10 Iterative methods for linear systems, 65F15 Eigenvalues, eigenvectors

1. Introduction.

This paper describes a multilevel adaptive aggregation (MAA) method for calculating the stationary probability vector of an irreducible stochastic matrix. Performance of the method is investigated and compared with more traditional iterative methods for stochastic matrices that provide models for web page ranking, including Google's PageRank model.

Our method is a special case of the adaptive smooth aggregation (SA) [1] and adaptive algebraic multigrid (AMG) [2] methods for sparse linear systems. It is, in fact, a variant of the original adaptive method developed in the early stages of the AMG project by A. Brandt, S. McCormick, and J. Ruge [3] (described earlier in [4]). It is also closely related to certain extensively studied aggregation methods for Markov chains. The coarse-level equations we use on aggregated states are essentially the aggregated equations proposed in [5], and the framework of our two-level method is similar to the iterative aggregation/disaggregation (IAD) method for Markov chains that was pioneered in [6] and has since been used and analyzed extensively [7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17]. While the IAD method is normally employed as a two-level method, the use of multilevel versions has been proposed in the literature, and the link to algebraic multigrid methods has been pointed out before [18, 19, 20, 21]. The

*Department of Applied Mathematics, University of Waterloo, Waterloo, Ontario, Canada

†Department of Applied Mathematics, University of Colorado at Boulder, Boulder, Colorado, USA

‡hdesterck@uwaterloo.ca

§tmanteuf@colorado.edu

¶stevem@colorado.edu

‖jruge@colorado.edu

††quocuz@gmail.com

iterative aggregation framework of the algorithm we describe in this paper is thus not new, but the way we choose aggregates is.

In most two-level applications of the IAD method, aggregates are chosen in a prespecified way, based on topological knowledge of the fine-level problem. We mention three examples here: for so-called nearly completely decomposable Markov chains, the aggregates are normally chosen to be the fine-level blocks that are nearly decoupled [7, 8]; in [16], one of the coarse states is an aggregate that contains the dangling nodes of a webgraph; and, in [17], aggregates are formed by the nodes of the webgraph that reside on the same compute node in a distributed computing environment for web ranking. In contrast, in the approach we propose, the aggregation process does not employ any explicit advance knowledge of the topology of the Markov chain, but aggregation is done automatically, based solely on some measure of strength of connection in the stochastic matrix. Moreover, the aggregates are adaptively improved in each step of the iterative process. This adaptive aggregation strategy based on strength of connection relates our approach to the adaptive SA method for linear systems [1], and the resulting algorithm promises to be quite generally applicable, since no *a priori* knowledge of the topology of the Markov chain is needed for the aggregation step. Strength of connection-based aggregation can also be recursively applied on coarser levels, thus enabling multilevel application of the IAD ideas. It is precisely this multilevel procedure that makes multigrid methods so powerful and scalable for many sparse linear systems, and it is our intent in this paper to explore up to which degree such general scalability may be achieved for Markov chain problems. AMG methods can be parallelized efficiently, and AMG scalability has been achieved for linear systems with billions of unknowns [22]. This makes it interesting to explore the use of AMG-like multilevel methods for large Markov chains, such as the Markov chains that result from models for ranking web pages.

Markov chain agglomeration based on strength of connection has been advocated before [18, 19, 21], but it has proven difficult to come up with a strength-based multilevel aggregation strategy that is successful for a wide class of Markov matrices [21]. This is where the main algorithmic contribution of our paper lies: we propose strength-based adaptive agglomeration for Markov chains that is based on AMG-like strength of connection in a scaled problem matrix, in which the columns of the original problem matrix at each recursive fine level are scaled with the current probability vector iterate at that level. This latter choice is motivated by careful consideration of the error equation to be solved in every step and at every level of the iteration procedure. The new aggregation process is thus fully adaptive, with optimized aggregates chosen in each step of the iteration and at all recursive levels.

In the second part of this paper, we investigate the efficiency and scalability of our strength-based multilevel method with adaptive aggregation, on a set of stochastic matrices that provide models for web page ranking. To our knowledge, the use of iterative methods of multilevel adaptive aggregation type has not been explored before for web ranking problems. One of the models studied is the well-known PageRank model [23] employed in the Google search engine [24], and we also apply our method to two different regularizations of the web graph for page ranking. We compare the efficiency of our strength-based multilevel adaptive aggregation method to the efficiency of more standard iterative methods, for problems of various size. These numerical tests serve to illustrate for which types of stochastic matrices our multilevel adaptive method may provide significant speedup, and also to provide more insight into why the PageRank model is a succesful model for determining a ranking of web

pages [25, 26].

2. Multilevel Adaptive Aggregation for calculating the stationary probability vector of an irreducible stochastic matrix.

2.1. Problem formulation. Let $B \in \mathbb{R}^{n\times n}$ be a column-stochastic matrix, *i.e.*, $b_{ij} \geq 0\ \forall i,\ j$ and $\sum_{i=1}^{n} b_{ij} = 1\ \forall j$. (Note that the latter condition may also be written as $\mathbf{e}^T B = \mathbf{e}^T$, with $\mathbf{e}$ the column vector with all ones.) We want to find a vector $\mathbf{x} \in \mathbb{R}^n$ that satisfies

$$B\,\mathbf{x} = \mathbf{x}, \qquad x_i \geq 0\ \forall i, \qquad \|\mathbf{x}\|_1 = 1. \tag{2.1}$$

If matrix B represents the transition matrix of a Markov Chain, then $\mathbf{x}$ is called a stationary probability vector of the Markov Chain. Note that $\mathbf{x}$ is an eigenvector of B associated with eigenvalue 1, the eigenvalue with largest modulus of matrix B.

In what follows, we are concerned with irreducible matrices [11]. Matrix B is irreducible *iff* there exists a path from each vertex i to each vertex j in the directed graph of matrix B. It can be shown that, if B is irreducible, then $\exists!\ \mathbf{x} : B\,\mathbf{x} = \mathbf{x}$, $x_i \geq 0\ \forall i$, and $\|\mathbf{x}\|_1 = 1$. Moreover, this unique stationary probability vector $\mathbf{x}$ satisfies $x_i > 0\ \forall i$. Matrix B is called periodic with period $p > 1$ *iff* the lengths of all cycles in the directed graph of B are multiples of p. If, in addition to irreducibility, B is a-periodic, then the unique stationary probability vector, $\mathbf{x}$, can be obtained from any initial vector $\mathbf{x}_0$ with positive components and $\|\mathbf{x}_0\|_1 = 1$ by repeated multiplication with B:

$$\mathbf{x} = \lim_{n\to\infty} B^n\,\mathbf{x}_0. \tag{2.2}$$

2.2. Relaxation: Power, Jacobi, and Gauss-Seidel methods. A simple and often used iterative method for approximating the unique stationary probability vector of an irreducible and a-periodic column-stochastic matrix B, starting from an initial guess $\mathbf{x}_0$ with $\|\mathbf{x}_0\|_1 = 1$, is the Power method, which is given by:

$$\mathbf{x}_{i+1} = B\mathbf{x}_i. \tag{2.3}$$

Closely related iterative methods are Jacobi (JAC) and Gauss-Seidel (GS). The stationary probability equation (2.1) can be rewritten as

$$A\mathbf{x} = 0, \tag{2.4}$$

with

$$A = B - I, \tag{2.5}$$

where $I \in \mathbb{R}^{n\times n}$ is a unit matrix. Using standard notation for the decomposition of matrix A into its lower and upper triangular parts and its diagonal part $A = L+D+U$, Jacobi for finding the solution of Eq. (2.4) is given by

$$\mathbf{x}_{i+1} = N(D^{-1}(L+U)\mathbf{x}_i), \tag{2.6}$$

and Gauss-Seidel by

$$\mathbf{x}_{i+1} = N((L+D)^{-1}U\mathbf{x}_i). \tag{2.7}$$

Here, we use the normalization operator $N(.)$ defined by

$$N(\mathbf{x}) = \frac{\mathbf{x}}{\|\mathbf{x}\|_1} \qquad (\mathbf{x} \neq 0). \tag{2.8}$$

Note that in the iterative algorithms described in this paper, the normalizations expressed by $N(.)$ do not necessarily have to be carried out in all steps of the algorithms, and may in fact be combined in a single normalization at the end of the iterative process. While the latter strategy is obviously advantageous for efficient computer implementation, we choose to include these normalizations in the descriptions of the algorithmic steps in this paper for reasons of ease of interpretation. Indeed, in this way, all the vectors $\mathbf{x}_i$ can be interpreted as probability vectors, because they satisfy $\sum_{i=1}^{n} x_i = 1$.

In a multigrid context, the Power, Jacobi, and Gauss-Seidel methods are often referred to as relaxation or smoothing methods, because they attenuate oscillatory error components quickly, leaving smooth error (at least for typical discrete elliptic systems). A variant of Jacobi used further on in this paper is weighted Jacobi (WJAC) with weight w, given by

$$\mathbf{x}_{i+1} = N((1-w)\,\mathbf{x}_i + w\,D^{-1}(L+U)\mathbf{x}_i). \tag{2.9}$$

The Power and Gauss-Seidel methods can be weighted in the same fashion.

It is useful to mention here some of the convergence properties of these relaxation methods for Markov chains [10, 11]. As was stated above, a-periodicity of an irreducible stochastic matrix guarantees convergence of the Power method to the unique stationary probability vector, for any probability vector as initial condition. However, the weighted Power method with weight $w \in (0,1)$ converges to the unique stationary probability vector of an irreducible stochastic matrix regardless of periodicity, for any probability vector as initial condition. In contrast, Jacobi and Gauss-Seidel may fail to converge, even when the matrix is a-periodic, but weighted Jacobi and Gauss-Seidel iteration with weight $w \in (0,1)$ always converge, regardless of the periodicity of the stochastic matrix. For some initial conditions, however, convergence may not be to the unique stationary probability vector. For example, for initial conditions that lie in the kernel of the upper triangular part, U, of B, Gauss-Seidel fails to converge to the unique stationary probability vector, and converges to the trivial solution instead (when no normalization is employed during the iteration process). Some of these convergence properties of the various relaxation methods can be easily seen for the simple 2×2 periodic irreducible matrix given by

$$B = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \tag{2.10}$$

with stationary probability vector $\mathbf{x}^T = [1/2\ 1/2]$, and its a-periodic perturbation

$$\hat{B} = (1-\epsilon)\,B + \epsilon I, \tag{2.11}$$

that has the same stationary probability vector.

2.3. Multilevel Adaptive Aggregation.

2.3.1. Aggregation equations for Markov Chains. It is well-known that Markov chain states can be aggregated, and an equation can be obtained for the

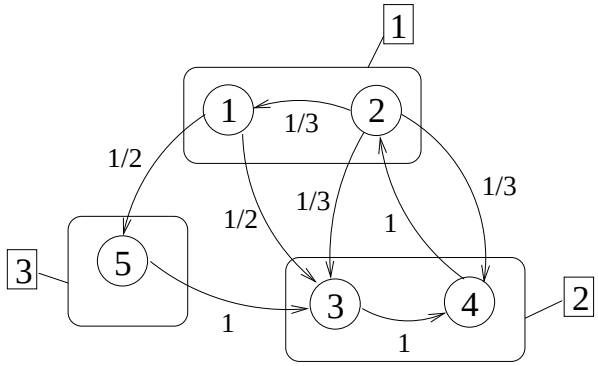

FIG. 2.1. *Aggregation of a fine-level Markov chain with five states into a coarse-level chain with three states. The fine-level states are indicated by numbers in circles, and the aggregated coarse-level states are indicated by numbers in boxes. Fine-level transitions are indicated by arrows with associated transition probabilities. The fine-level Markov chain is an example of a random walk on a directed graph: in each state, the outlinks are followed with equal probability.*

stationary probability vector of the aggregated states that is equivalent to Eq. (2.1). We illustrate this here with an example.

Fig. 2.1 gives a simple example of a Markov chain with five states ($n = 5$). This Markov chain is an example of a random walk on a graph. In each state, the outlinks are followed with equal probability. The transition matrix is given by

$$B = \begin{bmatrix} 0 & 1/3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 1/2 & 1/3 & 0 & 0 & 1 \\ 0 & 1/3 & 1 & 0 & 0 \\ 1/2 & 0 & 0 & 0 & 0 \end{bmatrix}, \tag{2.12}$$

with stationary probability vector $\mathbf{x}^T = [2/19\ 6/19\ 4/19\ 6/19\ 1/19]$.

In Fig. 2.1, the five states are aggregated into three coarse states. The stationary probability vector on the aggregate with index set I is given by

$$x_{c,I} = \sum_{i \in I} x_i. \tag{2.13}$$

Subscript c denotes that the aggregated probability vector applies to a coarse-scale version of the original fine-scale 5×5 problem. For the example of Fig. 2.1, the coarse-scale aggregated stationary probability vector is given by $\mathbf{x}_c^T = [8/19\ 10/19\ 1/19]$.

It follows directly from elementary probability calculus that $\mathbf{x}_c$ satisfies a coarse version of Eq. (2.1):

$$B_c\, \mathbf{x}_c = \mathbf{x}_c, \tag{2.14}$$

with the matrix elements of B_c given by

$$b_{c,IJ} = \frac{\displaystyle\sum_{j \in J} x_j \left(\sum_{i \in I} b_{ij} \right)}{\displaystyle\sum_{j \in J} x_j}. \tag{2.15}$$

For our example,

$$B_c = \begin{bmatrix} 1/4 & 3/5 & 0 \\ 5/8 & 2/5 & 1 \\ 1/8 & 0 & 0 \end{bmatrix}. \tag{2.16}$$

In matrix form, Eq. (2.15) can be written as

$$B_c = P^T B \,\mathrm{diag}(\mathbf{x})\, P \,\mathrm{diag}(P^T \mathbf{x})^{-1}, \tag{2.17}$$

with matrix P given by

$$P = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{2.18}$$

and $\mathrm{diag}(\mathbf{x})$ denoting the diagonal matrix of appropriate dimension with diagonal entries x_i. In analogy with multigrid terminology, we call matrix P an interpolation operator, because it interpolates from the coarse to the fine level, and its transpose P^T a restriction operator. We also refer to P as the agglomeration matrix, because it contains the information that specifies the agglomerates. Operator P^T restricts fine-scale probability vector $\mathbf{x}$ to the coarse-scale vector, $\mathbf{x}_c$:

$$\mathbf{x}_c = P^T \mathbf{x}. \tag{2.19}$$

Coarse-scale matrix B_c is a column stochastic matrix, and it can be shown that B_c inherits the properties of irreducibility and a-periodicity from the fine-scale B [11].

2.3.2. Two-level acceleration of relaxation methods by aggregation. It is well-known that relaxation methods for solving problem (2.1), or, equivalently, problem (2.4), can be accelerated by making use of the aggregation idea described above. It is clear that the aggregated operator, B_c, cannot be constructed without knowledge of the fine-level solution, $\mathbf{x}$, but it turns out that a coarse-level operator constructed using the current fine-level relaxation iterate, $\mathbf{x}_i$, can be used to accelerate convergence in a two-level method as follows:

Algorithm: Two-level acceleration by agglomeration
choose initial guess $\mathbf{x}$
repeat
$\mathbf{x} \leftarrow N(\mathrm{Relax}(A, \mathbf{x}))$ $\quad \nu$ times
$A_c = P^T A \,\mathrm{diag}(\mathbf{x})\, P \,\mathrm{diag}(P^T \mathbf{x})^{-1}$
$\mathbf{x}_c \leftarrow$ solve $A_c \mathbf{x}_c = 0$, $x_{c,i} \geq 0 \,\forall i$, $\|x_c\|_1 = 1$ $\quad$ (coarse-level solve)
$\mathbf{x} = N(\mathrm{diag}(P \,\mathrm{diag}(P^T \mathbf{x})^{-1} \mathbf{x}_c)\, \mathbf{x})$ $\quad$ (coarse-level correction)
until convergence criterion satisfied

Here, $\mathrm{Relax}(A, \mathbf{x})$ stands for one step of relaxation (*e.g.*, the Power or Gauss-Seidel method). The coarse-level solve can be approximate (*e.g.*, by use of a relaxation method, which may employ $P^T \mathbf{x}$ as the initial guess), and the efficiency of the

algorithm hinges on exploiting the fact that the coarse-level solve typically requires much less work than a fine-level solve.

The coarse-level correction step needs some explanation here. On the coarse level, the current guess for the stationary probability vector, $P^T \mathbf{x}$, is replaced by the new approximation, $\mathbf{x}_c$. The expression $\mathrm{diag}(P^T \mathbf{x})^{-1} \mathbf{x}_c$ gives the ratios of the new and initial probabilities on each of the agglomerates, and the coarse-level correction scales the fine-level probabilities in each agglomerate by the coarse-level correction probability ratio of that agglomerate. As explained in more detail in the next section, the correction is thus a correction of multiplicative type. Depending on the properties of the fine-level matrix, A, the choice of the relaxation method, and the choice of the agglomeration matrix, P, this two-level method may accelerate convergence of the relaxation method considerably. Local convergence properties of the two-level aggregation method are discussed in [15].

2.3.3. Multiplicative coarse-level correction. It is instructive to explain the coarse-level correction step of the two-level algorithm further by considering multiplicative error equations as follows. Given a current iterative approximation, $\mathbf{x}_i$, for $A\,\mathbf{x} = 0$, define the multiplicative componentwise error vector of the ith iterate, $\mathbf{e}_i$, by

$$\mathbf{x} = \mathrm{diag}(\mathbf{x}_i)\,\mathbf{e}_i. \tag{2.20}$$

This yields the following multiplicative error equation

$$A\,\mathrm{diag}(\mathbf{x}_i)\,\mathbf{e}_i = 0. \tag{2.21}$$

At convergence, $\mathbf{x}_i = \mathbf{x}$ and $\mathbf{e}_i = \mathbf{e} = [1 \ldots 1]^T$. If the fine-level error, $\mathbf{e}_i$, is not known, one can seek a coarse-level approximation, $\mathbf{e}_c$, with $\mathbf{e}_i \approx P\,\mathbf{e}_c$. An approximate, coarse-level version of the error equation is then given by

$$P^T\,A\,\mathrm{diag}(\mathbf{x}_i)\,P\,\mathbf{e}_c = 0. \tag{2.22}$$

We can relate the coarse-level error, $\mathbf{e}_c$, to the coarse-level probability vector, $\mathbf{x}_c$, by

$$\mathbf{x}_c = \mathrm{diag}(P^T \mathbf{x}_i)\,\mathbf{e}_c, \tag{2.23}$$

which makes coarse-level error equation (2.22) equivalent to coarse-level stationary probability equation $A_c\,\mathbf{x}_c = 0$, with

$$A_c = P^T\,A\,\mathrm{diag}(\mathbf{x}_i)\,P\,\mathrm{diag}(P^T \mathbf{x}_i)^{-1}, \tag{2.24}$$

as in the two-level acceleration algorithm above. The fine-level error, $\mathbf{e}_i$, is then approximately given by

$$\mathbf{e}_i \approx P\,\mathrm{diag}(P^T \mathbf{x}_i)^{-1}\mathbf{x}_c, \tag{2.25}$$

and this approximation is used in the multiplicative error correction step in the algorithm given above.

2.3.4. Multilevel Adaptive Aggregation method. The efficiency of the two-level acceleration algorithm described above can be further enhanced by using the agglomeration idea in a recursive way, and it turns out that the convergence properties of the resulting algorithm can be improved by choosing the agglomeration matrices,

P, on all levels adaptively based on the current iterates, $\mathbf{x}_i$, and the matrix elements of A. The resulting adaptive multilevel aggregation algorithm is as follows.

Algorithm: Multilevel Adaptive Aggregation method (V-cycle)
$\mathbf{x} = \text{MAA\_V}(A, \mathbf{x}, \nu_1, \nu_2)$
begin
$\mathbf{x} \leftarrow N(\text{Relax}(A, \mathbf{x}))$ ν_1 times
build P based on $\mathbf{x}$ and A
$A_c = P^T A \,\text{diag}(\mathbf{x})\, P \,\text{diag}(P^T \mathbf{x})^{-1}$
$\mathbf{x}_c = \text{MAA\_V}(A_c, N(P^T \mathbf{x}), \nu_1, \nu_2)$ (coarse-level solve)
$\mathbf{x} = N(\text{diag}(P \,\text{diag}(P^T \mathbf{x})^{-1} \mathbf{x}_c)\, \mathbf{x})$ (coarse-level correction)
$\mathbf{x} \leftarrow N(\text{Relax}(A, \mathbf{x}))$ ν_2 times
end

This algorithm uses the simplest type of recursion, resulting in a so-called V-cycle. Similar to the multigrid context, other types of recursive cycles can be considered as well.

Multilevel algorithms of this kind have been considered for Markov Chains before. It turns out, however, that a careful choice of aggregation strategy is crucial for the efficiency of the algorithm, and it appears that there is no known method that produces good results for general matrices A. In fact, efficiency of aggregation strategies may depend on certain properties of A. In the next subsection, we describe a particular aggregation strategy that is based on strength of connection in the scaled matrix $A \,\text{diag}(\mathbf{x}_i)$. Strength of connection is determined as in the AMG algorithm. The proposed aggregation process is fully adaptive, with optimized aggregates chosen in each step of the iteration and at all recursive levels.

Further on in the paper, we investigate the performance of the Multilevel Adaptive Aggregation method using this particular aggregation strategy for problems related to the ranking of web pages, and we compare its performance with more traditional iterative methods. The Multilevel Adaptive Aggregation method described above is also related to adaptive AMG and SA algorithms for the solution of sparse linear systems, as explained in more detail below.

2.3.5. Strength-based aggregation procedure. The main algorithmic contribution of this paper is that we propose an aggregation strategy for Markov chains that is based on strength of connection in the problem matrix scaled by the current iterate, $A \,\text{diag}(\mathbf{x}_i)$, and with strength of connection determined as in the AMG algorithm. A heuristic motivation for basing strength on the scaled problem matrix is as follows.

In each V-cycle, one solves the error equation

$$A \,\text{diag}(\mathbf{x}_i)\, \mathbf{e}_i = 0, \tag{2.26}$$

which indicates that the error of state j is influenced most strongly by the errors in states k that have large matrix elements in row j and columns k of the scaled matrix $A \,\text{diag}(\mathbf{x}_i)$. Denoting the scaled matrix with matrix elements $\bar{a}_{jk}$ by

$$\bar{A} = A \,\text{diag}(\mathbf{x}_i), \tag{2.27}$$

we base the agglomeration procedure on a strength matrix S defined as follows:

$$S_{jk} = \begin{cases} 1 & \text{if } j \neq k \text{ and } \bar{a}_{jk} \geq \theta \max_{l \neq j} (\bar{a}_{jl}), \\ 0 & \text{otherwise,} \end{cases} \tag{2.28}$$

where $\theta \in [0, 1]$ is a strength threshold parameter. In this paper, we choose $\theta = 0.8$.

Note that this is precisely the measure of strength that is employed in the AMG algorithm [3] (except for the column-scaling of A). In fact, the choice of basing strength on the scaled matrix $\bar{A}$ can be further motivated using AMG concepts as follows. Relaxation on $A\,\mathbf{x} = 0$ typically results in a small residual after a small number of relaxation steps: $A\,\mathbf{x}_i \approx 0$. After relaxation, the corresponding error, $\mathbf{e}_i$, in Eq. (2.26) is called algebraically smooth in AMG terminology. When $\mathbf{x}_i$ approaches the exact solution, $\mathbf{x}$, the multiplicative error, $\mathbf{e}_i$, approaches the unit vector. This means that algebraically smooth error approaches the unit vector. The multilevel method can be successful when it manages, after pre-relaxation, to reduce the smooth error by coarse-grid correction, while avoiding contamination of the fine level with a significant amount of error that is not algebraically smooth. To avoid creating large residuals on the fine level after coarse grid correction, it is important to correct, with the same multiplicative factor, the errors of the states that are strongly connected with respect to the scaled matrix, $\bar{A}$; such strong connections would lead to large fine-level residuals if the corresponding states received significantly different coarse-grid corrections. For this reason, it is important to base the agglomeration process on a strength matrix that is determined by considering the matrix elements of the scaled matrix, $\bar{A}$.

After strength matrix S has been determined, the aggregates are formed by the following procedure:

Algorithm: aggregation based on strength matrix S

repeat

- among the unassigned states, choose state j which has the largest value in current iterate $\mathbf{x}_i$ as the seed point of a new aggregate
- add all unassigned states k that are strongly influenced by seed point j (*i.e.*, $S_{kj} = 1$) to the new aggregate

until all states are assigned to aggregates

This aggregation heuristic is based on the notion that 'important states' (states that have a large value in the current iterate $\mathbf{x}_i$) are good candidate seed points for new aggregates, and that states that are strongly influenced by the seed point of an aggregate are good candidates to join that aggregate. This aggregation method can be implemented efficiently by pre-sorting the current iterate vector $\mathbf{x}_i$. The performance of the MAA method employing this agglomeration strategy is examined in Section 4.

Multilevel strength-based agglomeration methods have been considered before for Markov chains, if only occasionally [19, 21]. The few papers that we are aware of employed significantly different ways to calculate strength than what we propose, especially since their notion of strength was based on the original matrix, A. While these previous approaches have met with some success for particular types of Markov

chains, it has proved difficult to find agglomeration strategies that are successful for large classes of Markov chains [21].

2.4. Relation to adaptive AMG and adaptive SA. Our MAA algorithm is similar in several respects to the original adaptive AMG scheme (now referred to as aAMG) developed in the early stages of the AMG project [3]. The original aAMG approach was based on three fundamental components: Gauss-Seidel relaxation applied to the fine-level homogeneous problem, $A\,\mathbf{x} = 0$; a rescaling of A by the current approximation, $\mathbf{x}$, according to $A \leftarrow \mathrm{diag}(\mathbf{x})^{1/2}\, A\, \mathrm{diag}(\mathbf{x})^{1/2}$, where $\mathrm{diag}(\mathbf{x})$ is the diagonal matrix with entries x_j; and a standard AMG formulation of the coarse-grid correction equation. This is similar to the MAA approach because: the scaling in effect amounts to multiplicative correction; it is adaptive in that the aim is to compute a vector in the (near) null space of A; it takes the scaled matrix into account in the coarsening process; and it is a true multilevel process because this approach was used on all levels. It differs, however, in that the adaptive AMG scheme was applied primarily to real, symmetric positive definite matrices and the coarsening strategy was of AMG type, as opposed to agglomeration type.

The current adaptive smoothed aggregation (aSA) and aAMG methodologies (see [1] and [2], respectively) have been extended in several respects from the early aAMG schemes. This extension is most evident in aSA, where several representative components of the (near) null space of A are computed. This allows the adaptive approach to handle more complicated systems of partial differential equations, such as those that arise in structural mechanics. Treating multiple components is a delicate process in this methodology because many aspects of the approach must be carefully considered. Critical aspects include a clear articulation of what constitutes a representative, mechanisms to prevent global and local redundancy, and procedures for improving the quality of these components. Such a development in the AMG context has been longer in the making because these issues are not as natural as they are in an aggregation setting.

3. Regularization of the web matrix for page ranking. Let G be the column-based adjacency matrix describing links between web pages, *i.e.*, $g_{ij} = 1$ if page j contains a link to page i, and $g_{ij} = 0$ otherwise. Assume that G is derived from a single breadth-first 'webcrawl' that recursively visits the first n pages that are reached from a root page. This link structure can be used to determine an importance ranking of the web pages as follows. For simplicity, first assume that every page has at least one outlink (there are no dangling nodes), or, equivalently, that every column of G has at least one non-zero element. Also, we make the usual assumption throughout the remainder of this paper that self-links are not considered in the original G, although they may be introduced via one of the regularizations discussed below, or into agglomerated equations on coarse levels. Let

$$B = N(G), \tag{3.1}$$

where normalization operator $N(.)$ is now applied to every column of its matrix argument. Consider a 'random surfer' who starts from an arbitrary page and wanders through the network by each time following outlinks with equal probability from the page he resides on. The random surfer basically performs a random walk on the directed graph induced by the link structure. The transition probabilities are given by the columns of B, and, if B is irreducible and a-periodic, a stationary probability

distribution $\mathbf{x}$ will ultimately be approached that satisfies

$$B\,\mathbf{x} = \mathbf{x}. \tag{3.2}$$

This stationary probability density can be used to rank the web pages by importance. One of the crucial properties of rankings defined by this kind of mechanism is that the resulting rankings are robust against 'spamming'.

There are, however, several problems with the general applicability of this approach. The web matrix, G, may have dangling nodes, and may in general not lead to an irreducible and/or a-periodic stochastic matrix B. The link structure has to be regularlized in some way before the stationary probability problem (3.2) can be formulated in a well-posed manner. In the following subsections, we describe a few possible regularizations.

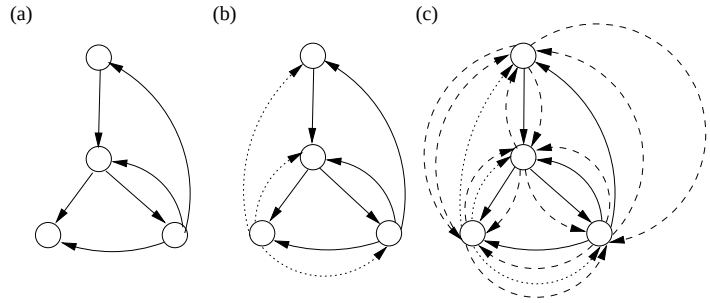

FIG. 3.1. *Example of PageRank regularization. (a) Original, reducible graph. (b) Links from dangling nodes to all other nodes are added with probability* $1/n$ *(dotted lines). (c) Links from all nodes to all nodes are added with coupling factor* α *(dashed lines).*

3.1. PageRank regularization. The PageRank algorithm that is employed by the Google search engine regularizes the link structure as follows (Fig. 3.1). First, the dangling nodes are treated by linking them with all other nodes. Dangling node vector $\mathbf{d}$ is introduced, which is 1 for every dangling node and 0 otherwise. This ensures that $N(G + \mathbf{d}\,\mathbf{e}^T)$ is a stochastic matrix. Second, this stochastic matrix can be made irreducible and a-periodic by adding links from all nodes to all nodes that are to be followed with a small probability, α. The PageRank regularized matrix, B_{PR}, is then given by

$$B_{PR} = (1-\alpha)\,N(G + \mathbf{e}\,\mathbf{d}^T) + \alpha\,N(\mathbf{e}\,\mathbf{e}^T). \tag{3.3}$$

We call α the coupling factor of the regularization. A value of $\alpha = 0.15$ is reported to normally be used for the PageRank algorithm.

3.2. Backlink to the root page. A first alternative way to deal with dangling nodes and reducibility is to add a backlink from every page to the root page of the crawl, with a small coupling probability, α (Fig. 3.2). This, however, does not necessarily make the resulting matrix a-periodic. A-periodicity can be enforced by adding self-links with a small probability, ϵ. In our numerical tests, we choose $\epsilon = 10^{-12}$. Note that this addition does not change the stationary probability vector. In fact, while it guarantees convergence of the Power method without damping, it is not needed for convergence of weighted relaxation methods and the MAA method. We add it here merely because it allows the 'random surfer' in our web ranking analogy

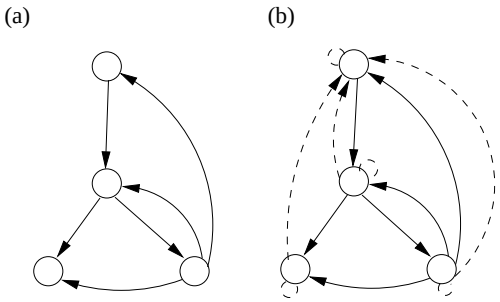

FIG. 3.2. *Example of BackLink regularization. (a) Original, reducible graph. (b) Links from all nodes to to the root node are added with coupling factor α, and selflinks with strength ϵ (dashed lines).*

to converge to the stationary probability vector. The resulting BackLink regularized matrix, B_{BL}, is given by

$$B_{BL} = N((1-\alpha-\epsilon)\,N(G) + \alpha\,\mathbf{e}^{(1)}\,\mathbf{e}^T + \epsilon\,I). \tag{3.4}$$

Here, $\mathbf{e}^{(1)} = [1\ 0 \ldots 0]^T$, and we have assumed that the root page of the crawl is the first page in the page numbering. Note that this regularization adds global links, but only a relatively small amount of them, and all to the same target. Note also that the normalization operator $N(.)$ is now used in a generalized way, in the sense that columns of its matrix argument that are identically zero remain unchanged under the action of $N(.)$. The second, outer normalization $N(.)$ is needed to deal correctly with dangling nodes.

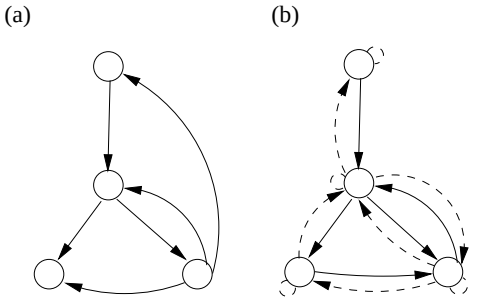

FIG. 3.3. *Example of BackButton regularization. (a) Original, reducible graph. (b) Reverse links are added for all links with coupling factor α, and selflinks with strength ϵ (dashed lines).*

3.3. Adding local backlinks. A second alternative way for dealing with dangling nodes and reducibility is to add backlinks to the pages from which links originate, with a small coupling probability, α (Fig. 3.3). This is a crude model for the use of the back button by actual web surfers. Again, this does not necessarily make the resulting matrix a-periodic, but a-periodicity can again be enforced by adding self-links with a small probability, ϵ. This results in what we call the BackButton regularized matrix B_{BB}:

$$B_{BB} = N((1-\alpha-\epsilon)\,N(G) + \alpha\,N(G^T) + \epsilon\,I). \tag{3.5}$$

Note that regularization (3.5) only adds local links.

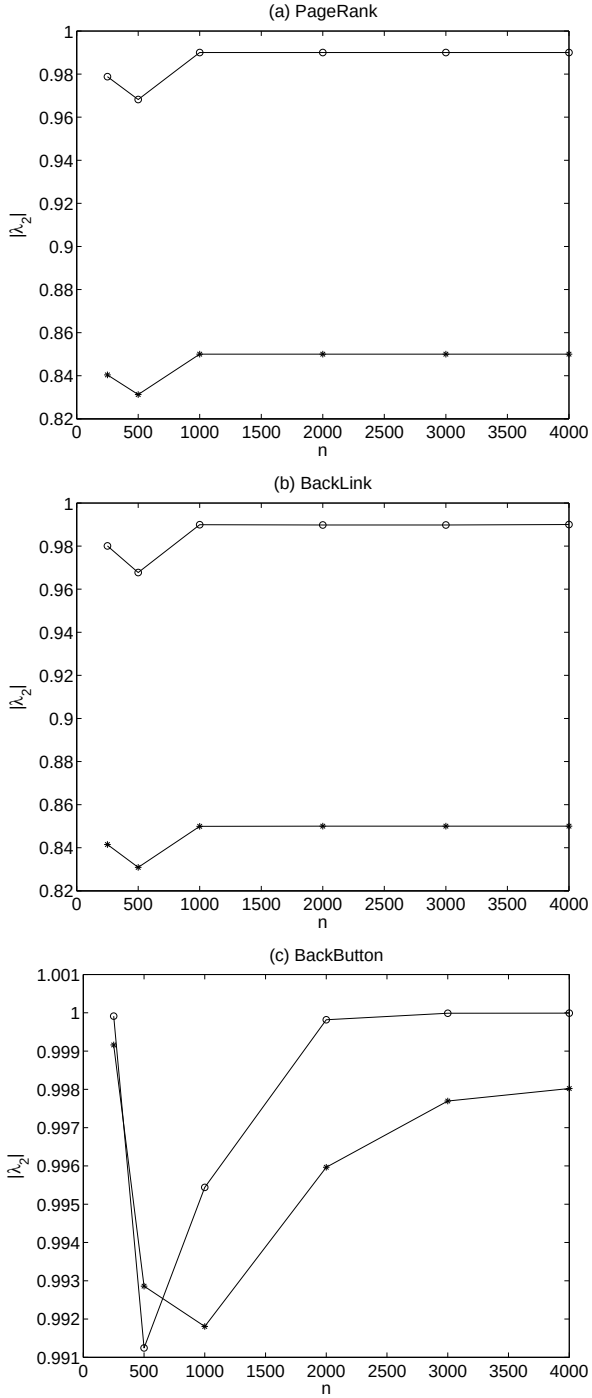

FIG. 3.4. *Modulus of the second eigenvalue, λ_2, for the three types of web matrix regularization as a function of problem size n, for coupling factors $\alpha = 0.15$ (*) and $\alpha = 0.01$ (o).*

3.4. Second eigenvalue of the regularized matrices. The asymptotic convergence factor (*i.e.*, the asymptotic error reduction factor per iteration) of the Power method is bounded by the modulus of the second eigenvalue of B. It can be shown that the second eigenvalue of B_{PR} satisfies $|\lambda_2| \leq 1 - \alpha$, with equality holding when there are at least two irreducible closed subsets in $N(G + \mathbf{e}\,\mathbf{d}^T)$ [27, 26]. One of the advantages of the PageRank regularization is that the Power method converges with asymptotic convergence factor smaller than $1 - \alpha$, independent of the problem size, n. Fig. 3.4 shows the modulus of the second eigenvalue of the three different regularizations of the web matrix, for various problem sizes from $n = 250$ to $n = 4000$, and for coupling factors $\alpha = 0.15$ and $\alpha = 0.01$.

The test problems were generated as subsets of a standard set of real web data, namely, the Stanford Web Matrix with 281903 pages and approximately 2.3 million links, which was gathered in a September 2002 crawl [28]. We reordered the Stanford web matrix such that the page numbering reflects a breadth-first crawl from a root page. After reordering in this way, well-defined subproblems of any size can be obtained by taking appropriate submatrices.

The plots confirm that, for B_{PR}, $|\lambda_2| = 1 - \alpha$ independent of n, for n sufficiently large. The same can be observed for the B_{BL} regularization. However, the B_{BB} regularization, which is achieved by adding only local backlinks, behaves differently: $|\lambda_2|$ is much closer to 1 than $1 - \alpha$, appearing to approach 1 as n increases. This is further investigated in Fig. 3.5, for $\alpha = 0.15$ and larger problem sizes. Fig. 3.5(b) indicates that $\lambda_2 \approx 1 - O(1/n)$, with fitted slope ≈ -1.0480 in the $\log - \log$ plot. This means that slow convergence of the Power method can be expected for the B_{BB} regularization, with convergence speed decreasing as n grows. This could be expected, due to the local link structure of the B_{BB} regularization, but it is somewhat surprising that the slope in Fig. 3.5(b) is so close to linear, because changes in submatrix size for the subproblems of the Stanford web matrix may go along with significant changes in topology, in addition to the effect of the change in size.

3.5. Efficient implementation of the MAA method for the PageRank regularization. One of the disadvantages of the PageRank regularization is that B_{PR} loses its sparsity due to the addition of the all-to-all connections. However, this need not lead to increased computational complexity in the MAA algorithm, because one can keep the additional terms separate on all recursive levels. For the PageRank and BackLink regularizations, matrix A in $A\,\mathbf{x} = 0$ can be written as

$$A = A_{sparse} + \mathbf{h}\,\mathbf{f}^T, \tag{3.6}$$

with $\mathbf{h}$ and $\mathbf{f}$ the column vectors that define the rank-one update to the sparse part of the matrix, A_{sparse}. The coarse operator matrix is then given by

$$A_c = A_{sparse,c} + \mathbf{h}_c\,\mathbf{f}_c^T, \tag{3.7}$$

with

$$A_{sparse,c} = P^T\,A_{sparse}\,\mathrm{diag}(\mathbf{x})\,P\,\mathrm{diag}(P^T\,\mathbf{x})^{-1}, \tag{3.8}$$

$$\mathbf{h}_c = P^T\,\mathbf{h}, \tag{3.9}$$

and

$$\mathbf{f}_c^T = \mathbf{f}^T\,\mathrm{diag}(\mathbf{x})\,P\,\mathrm{diag}(P^T\,\mathbf{x})^{-1}. \tag{3.10}$$

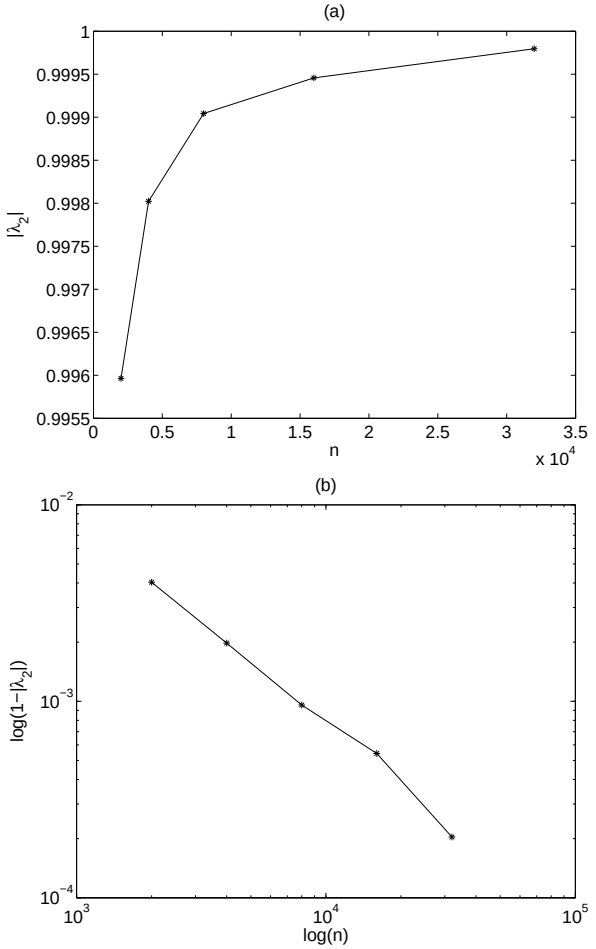

FIG. 3.5. *Modulus of the second eigenvalue, λ_2, for the BackButton regularization as a function of problem size n, for coupling factor $\alpha = 0.15$. The* $\log - \log$ *plot (fitted slope* ≈ -1.0480*) indicates that* $1 - |\lambda_2| \approx O(1/n)$.

The vectors that define the rank-one update can thus be kept separate on all coarse levels. Moreover, presence of the rank-one update does not preclude GS or JAC relaxation sweeps with $O(n)$ complexity, because the relaxation implementations can easily be modified such that $O(n)$ complexity is retained. With respect to agglomeration, we found that satisfactory results can be obtained for the MAA method by only considering the sparse part of A for strength matrix calculation and agglomeration on all levels, and the numerical results reported below use this simplification.

3.6. Symmetrization for the regularized problem. It is interesting to point out the following symmetrization for the regularized problem. Let B be column-stochastic and irreducible, and let $\mathbf{x}$, with components $x_i > 0 \ \forall i$, be the unique stationary probability vector of B. Assume that web surfers are allowed to follow links back with probabilities based on where they were likely to have come from. The column-stochastic matrix that describes the probabilities of following backlinks

n	γ_{MAA}	it_{MAA}	$c_{grid,MAA}$	γ_{WJAC}	$f^{(tot)}_{MAA-WJAC}$	$f^{(as)}_{MAA-WJAC}$
PageRank, $\alpha = 0.15$						
2000	0.355124	10	1.67	0.815142	1/2.74	1/3.13
4000	0.335889	9	1.67	0.805653	1/2.52	1/3.59
8000	0.387411	9	1.65	0.821903	1/2.79	1/4.14
16000	0.554686	12	1.78	0.836429	1/4.07	1/6.89
32000	0.502008	11	1.83	0.833367	1/3.94	1/6.20
64000	0.508482	11	1.75	0.829696	1/3.86	1/6.21
128000	0.532518	12	1.75	0.829419	1/4.31	1/7.01
PageRank, $\alpha = 0.01$						
2000	0.321062	10	1.77	0.956362	3.42	1.32
4000	0.658754	20	1.75	0.980665	2.16	1.03
8000	0.758825	22	1.65	0.976889	1.88	1/1.65
16000	0.815774	27	1.77	0.979592	1.45	1/2.31
32000	0.797182	29	1.82	0.979881	1.35	1/2.09
64000	0.786973	33	1.79	0.980040	1.19	1/1.96
128000	0.854340	38	1.72	0.980502	1.05	1/2.88

TABLE 4.1

MAA performance results for the PageRank regularization. For $\alpha = 0.15$, MAA is 2-4 times less efficient than WJAC, and for $\alpha = 0.01$, MAA is 1-3 times more efficient, depending on the particular problem.

is given by

$$\operatorname{diag}(\mathbf{x})\, B^T \operatorname{diag}(B\,\mathbf{x})^{-1}. \tag{3.11}$$

If backlinks are followed with probability α, then

$$B_\alpha = (1-\alpha)\, B + \alpha \operatorname{diag}(\mathbf{x})\, B^T \operatorname{diag}(B\,\mathbf{x})^{-1} \tag{3.12}$$

gives the transition matrix for the resulting Markov chain.

Interestingly, the stationary probability vector, $\mathbf{x}$, of B is also a stationary probability vector of B_α, for any α, as can be verified easily. Moreover, for $\alpha = 1/2$,

$$C = B_{1/2} \operatorname{diag}(\mathbf{x}) = \frac{1}{2}\, B \operatorname{diag}(\mathbf{x}) + \frac{1}{2} \operatorname{diag}(\mathbf{x})\, B^T, \tag{3.13}$$

and thus

$$C = C^T. \tag{3.14}$$

It may be possible to exploit the symmetry of C in iterative algorithms for calculating the stationary probability vector of Markov chains, as it may be expected that algorithms like MAA can be made to perform better when problem matrices are symmetric. This will be explored in future work.

4. Performance of MAA and WJAC for page ranking. In this section we study the performance of the MAA method for the three types of stochastic matrices that provide models for web page ranking that were discussed in Sec. 3. The MAA performance results are compared with the fine-grid WJAC method. For all MAA tests, we use V(1,1) cycles (with one pre-relaxation and one post-relaxation), WJAC

n	γ_{MAA}	it_{MAA}	$c_{grid,MAA}$	γ_{WJAC}	$f^{(tot)}_{MAA-WJAC}$	$f^{(as)}_{MAA-WJAC}$
BackLink, $\alpha = 0.15$						
2000	0.331226	11	1.67	0.839540	1/3.11	1/3.04
4000	0.344225	11	1.75	0.851397	1/3.30	1/3.18
8000	0.361255	11	1.69	0.858532	1/3.04	1/3.24
16000	0.358282	11	2.03	0.866344	1/3.75	1/4.11
32000	0.369351	11	2.26	0.868116	1/3.99	1/4.39
64000	0.368789	11	1.88	0.868889	1/3.30	1/3.53
128000	0.369744	11	1.78	0.871525	1/3.07	1/3.22
BackLink, $\alpha = 0.01$						
2000	0.452383	16	1.89	0.952865	2.01	1/1.21
4000	0.778003	28	1.76	0.953782	1.41	1/4.23
8000	0.749847	20	1.72	0.970096	2.23	1/2.23
16000	0.745776	23	1.96	0.976919	1.87	1/2.11
32000	0.855323	28	1.93	0.981223	1.66	1/3.04
64000	0.868049	32	1.96	0.983076	1.45	1/3.15
128000	0.837747	31	1.83	0.985161	1.65	1/2.09

TABLE 4.2

MAA performance results for the BackLink regularization. For $\alpha = 0.15$, MAA is 3-4 times less efficient than WJAC, and for $\alpha = 0.01$, MAA is 1-2 times more efficient, depending on the particular problem.

relaxation with weight $w = 0.8$, agglomeration strength threshold $\theta = 0.8$, and a direct solve on the final coarse grid with size less than 20 states. For the various test problems, we perform MAA cycles until the error is reduced by a factor 10^{-5}. The error is calculated as the 1-norm of the difference between the current iterate and a highly accurate numerical approximation to the exact solution of $B\,\mathbf{x} = \mathbf{x}$. All errors and convergence factors are measured in the 1-norm. For comparison, we perform WJAC sweeps on the fine grid only, for the same time as the total time of execution of MAA, or until the error is reduced by a factor 10^{-5}, whichever comes first. The legend for the tables in this section is as follows:

- n: problem size
- γ_{MAA}: MAA convergence factor for the final cycle
- it_{MAA}: number of MAA iterations until the error is reduced by a factor 10^{-5}
- $c_{grid,MAA}$: MAA grid complexity for the final cycle
- γ_{WJAC}: WJAC convergence factor for the final cycle
- $f^{(tot)}_{MAA-WJAC}$: total efficiency factor which measures how much faster MAA converges than WJAC, for the same amount of work
- $f^{(as)}_{MAA-WJAC}$: asymptotic efficiency factor which measures how much faster an MAA V-cycle reduces the error than a WJAC sweep, for the same amount of work

The MAA grid complexity, $c_{grid,MAA}$, is defined as the sum of the number of degrees of freedom on all levels, divided by the number of fine-level degrees of freedom.

n	γ_{MAA}	it_{MAA}	$c_{grid,MAA}$	γ_{WJAC}	$f^{(tot)}_{MAA-WJAC}$	$f^{(as)}_{MAA-WJAC}$
BackButton, $\alpha = 0.15$						
2000	0.746000	35	1.74	0.981331	2.36	1/1.41
4000	0.800454	39	1.64	0.982828	2.70	1/1.36
8000	0.786758	40	1.53	0.992129	3.15	1.17
16000	0.851671	50	1.62	0.992330	3.00	1/1.38
32000	0.988423	214	1.64	0.998366	4.92	1/2.88
64000	0.973611	185	1.59	0.999013	9.95	1.40
128000	0.943160	116	1.55	0.999693	34.64	9.90
BackButton, $\alpha = 0.01$						
2000	0.658032	23	1.68	0.999563	106.02	46.05
4000	0.794123	29	1.71	0.999345	73.02	19.78
8000	0.841182	39	1.70	0.997624	23.49	2.64
16000	0.835592	44	1.78	0.998696	19.72	4.42
32000	0.845457	56	1.83	0.999114	39.58	8.22
64000	0.959561	81	1.75	0.999660	75.05	5.74
128000	0.921870	42	1.70	0.999963	816.62	103.79

TABLE 4.3

MAA performance results for the BackButton regularization. For $\alpha = 0.15$, MAA is 2-35 times more efficient than WJAC, and for $\alpha = 0.01$, MAA is 20-817 times more efficient, depending on the particular problem.

The total efficiency factor $f^{(tot)}_{MAA-WJAC}$ is defined as

$$f^{(t)}_{MAA-WJAC} = \frac{\log(r_{MAA})/t_{MAA}}{\log(r_{WJAC})/t_{WJAC}}, \tag{4.1}$$

where r_{MAA} and r_{WJAC} are the factors by which errors are reduced by the MAA and WJAC methods, respectively, and t_{MAA} and t_{WJAC} are their running times. For example, when MAA attains an error reduction $r_{MAA} = 10^{-4}$ in time $t_{MAA} = 2$, and WJAC attains $r_{WJAC} = 10^{-1}$ in time $t_{WJAC} = 1$, the total efficiency factor $f^{(t)}_{MAA-WJAC} = 2$, and MAA can be expected to be approximately twice as effective as WJAC, because WJAC would have to be executed twice as long as MAA in order to obtain the same error reduction. The other total efficiency factors in the tables are calculated in the same way.

Similarly, the asymptotic efficiency factor $f^{(as)}_{MAA-WJAC}$ is defined as

$$f^{(as)}_{MAA-WJAC} = \frac{\log(\gamma_{MAA})/t_{MAA}}{\log(\gamma_{WJAC})/t_{WJAC}}, \tag{4.2}$$

where γ_{MAA} and γ_{WJAC} are the asymptotic convergence factors of one MAA V-cycle and one WJAC sweep, respectively, and t_{MAA} and t_{WJAC} are their running times.

Note that these efficiency measures allow us to compare the real efficiency of the methods, as they take into account both work (compute time) and error reduction. The measures may depend somewhat on the implementation of the algorithms, but, for our results, this influence is likely to be small because the same WJAC implementation is used in the MAA V-cycles and in the fine-grid WJAC sweeps.

Tables 4.1, 4.2 and 4.3 show numerical performance results for the MAA method compared with WJAC. Table 4.1 shows that the MAA method converges adequately

for PageRank-normalized test problems of various sizes. MAA convergence factors, numbers of iterations, and grid complexities behave as expected. For $\alpha = 0.15$, MAA is less efficient than WJAC, but MAA is more efficient than WJAC for $\alpha = 0.01$. This is as expected: the numerical results indicate that WJAC convergence factors are bounded by $1 - \alpha$. For $\alpha = 0.15$, WJAC thus converges fast. MAA cycles are much more expensive than WJAC cycles (by a factor of about 20 to 30), and the improved MAA convergence factors cannot make up for this cost increase in this case. For $\alpha = 0.01$, however, WJAC converges more slowly and, in this case, it pays to use MAA. Note that for PageRank regularization the MAA advantage appears to diminish for growing problem size. The BackLink results in Table 4.2 show a similar pattern (WJAC convergence factors are bounded by $1 - \alpha$ in this case as well).

MAA results for the BackButton regularization are more interesting, however (see Table 4.3). For this problem type, MAA shows very large improvements over WJAC: MAA is 2 to 35 times more efficient than WJAC for $\alpha = 0.15$, and 20 to 817 times more efficient for $\alpha = 0.01$. Again, we see that convergence factors and grid complexities are well-behaved. The number of iterations required to reach the relative error tolerance shows some erratic behavior. This is likely due to the fact that increasing problem size in Table 4.3 does not merely reflect a change in size, but also may involve significant changes in link toplogy. Indeed, the problems are nested and the larger problems thus contain all smaller problems, but each increase in size may change the global link topology significantly. Overall, it appears that MAA is far superior to WJAC for this problem type, for which the second eigenvalue of the problem matrix $\lambda_2 \approx 1 - O(1/n)$.

The differences in performance of MAA and WJAC for the various web matrix regularizations can also be understood intuitively as follows. Global links are added in the PageRank and BackLink regularizations, and the resulting Markov chains have global links on all scales. These global links enable traditional single-level iterative methods like WJAC to be effective, so that the multiple levels employed by the MAA algorithm do not lead to a significant advantage. The BackButton regularization, however, adds only local backlinks, and the lack of global links means that single-level iterative methods are not effective. The multilevel nature of the MAA method allows us to bridge the scales for these types of problems, resulting in dramatic gains in performance.

5. Comparison of web matrix regularizations as a function of coupling factor α. It is interesting to compare the behavior of the three different regularizations of the web matrix as a function of the coupling factor α. Intuitively, the ranking would seem to be more realistic when the 'artificial' coupling factor is small, and $1 - \alpha$ approaches 1. Fig. 5.1 shows web page rankings calculated using the three regularizations, for varying coupling factor, α.

Several interesting effects can be noted. First, it turns out that B_{BL} is not a very good choice for web matrix regularization. All backlinks end in the root page of the crawl, which results in too much probability being assigned to this root page. As shown in Fig. 5.2, the root page gets assigned an unrealistically large probability for almost all values of α. (Fig. 5.1(b) is a detail of Fig. 5.2.) Also, pages that are close to the root page in the downstream direction inherit some of the root's inflated pagerank. It is thus clear that links to the root page create an unrealistic asymmetric bias in the page ranking. We therefore do not discuss this regularization further.

Second, it turns out that it may not be such a good idea to take α very close to 0. For the PageRank regularization, some pages acquire spurious probability as α

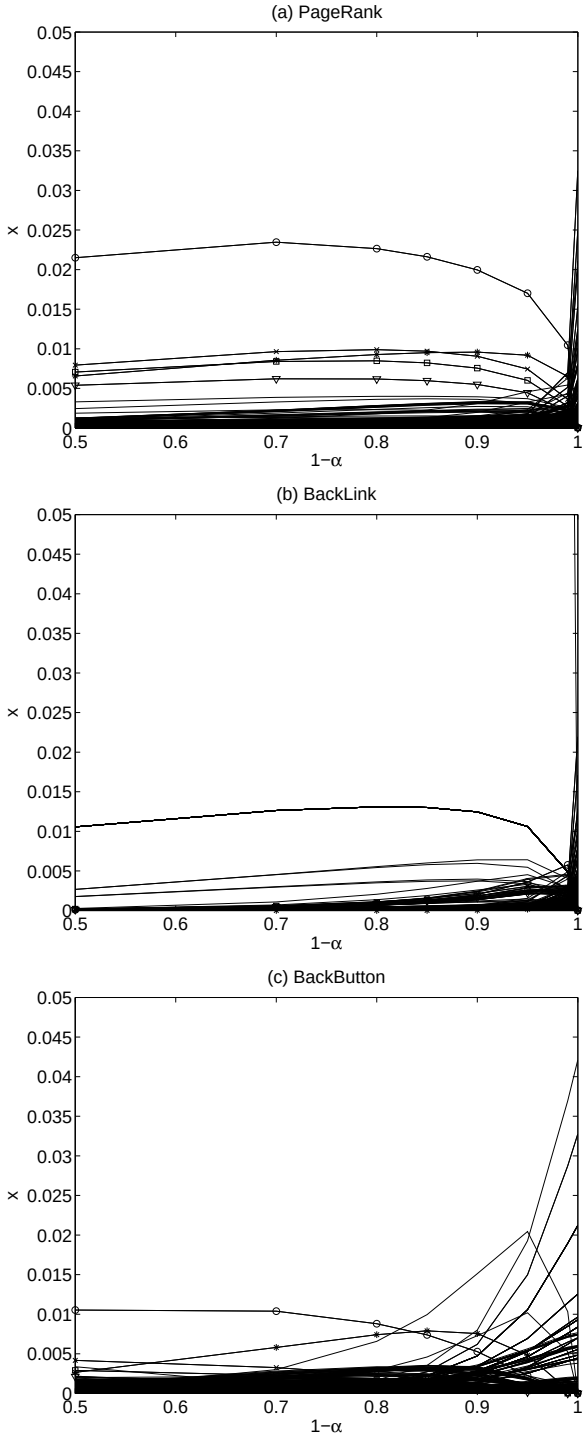

FIG. 5.1. *Stationary probability vector as a function of* $1-\alpha$. *The pages with the five largest PageRanks for* $\alpha = 0.15$ *are indicated by the following symbols:* o, $\times$, *, $\square$, ∇.

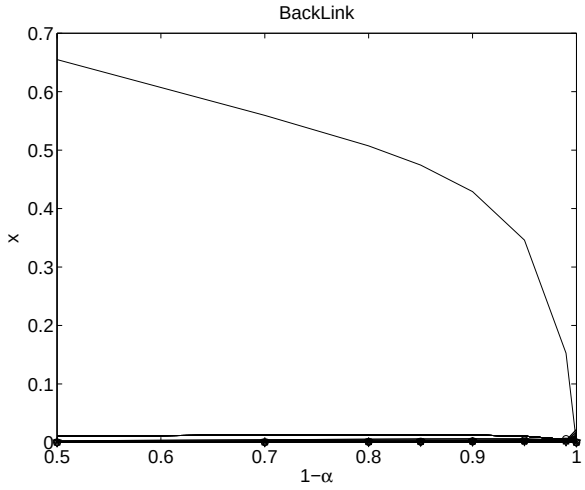

FIG. 5.2. *Stationary probability vector as a function of* $1-\alpha$ *for the BackLink regularization. The root page obtains by far the largest part of the probability, except when* $1-\alpha$ *approaches 1.*

approaches 0. These are not the dangling nodes, which remain connected to all other nodes as the coupling factor α approaches 0. They are clusters of nodes that could be called 'dangling clusters' (irreducible closed subsets), which act as probability sinks. These probability sink clusters start to dominate for small α, as they remove all of the probability from the other nodes. It is clear that this is undesirable, so it seems wise not to take α too close to 0. For the BackButton regularization, this probability sink problem is present as well. The probability sinks now include all of the irreducible closed subsets of the original web matrix, G, including the dangling nodes. Moreover, the probability starts to sink to the dangling nodes and clusters for values of α that are much further away from 0 than for PageRank, because the backlinks only redirect probability locally. It seems that global redistribution of probability is a better way to mitigate the problem of probability sinks.

6. Conclusion. We have presented the Multilevel Adaptive Aggregation method for calculating the stationary probability vector of an irreducible stochastic matrix. The method uses the same agglomeration equations as iterative aggregation/disaggregation methods for Markov chains, but the adaptive multilevel agglomeration procedure we propose is new: it is based on strength of connection in a scaled problem matrix, and it is fully adaptive, with optimized aggregates chosen in each step of the iteration and at all recursive levels. As such, the method is closely related to adaptive smooth aggregation and adaptive algebraic multigrid methods for sparse linear systems.

We have applied the MAA method to a set of stochastic matrices that provide models for web page ranking. In our numerical results, we compared three regularizations of the web matrix that can be used to calculate a ranking of web pages. The PageRank regularization with α not too close to 0 (*e.g.*, $\alpha = 0.15$) seems to be the best regularization of the web graph for page ranking purposes of the three regularizations we tested. The value of $\alpha = 0.15$ seems to be a good choice for two reasons: it mitigates the probability sink problem without diffusing probability differences too much, and it allows for fast convergence of the Power method.

It was shown that MAA can be more efficient than the Power or Weighted Jacobi

methods by very large factors for Markov chains for which the modulus of the second largest eigenvalue, $|\lambda_2|$, is close to 1, and especially when $|\lambda_2(n)| \to 1$ for large n. When $|\lambda_2|$ is constant in n and significantly smaller than 1, WJAC outperforms the MAA method presented. It may be possible to improve the aggregation strategy, but it is probably hard to find an adaptive aggregation strategy that would make MAA more efficient than WJAC for problems of this kind. For example, for the PageRank-regularized matrix B_{PR} with $\alpha = 0.15$, the convergence factor $\gamma_{WJAC} \approx 0.85$. It is fair to expect that MAA-like algorithms will always require the equivalent of at least 10, and probably rather 20, fine-level WJAC sweeps per V-cycle. That means that the MAA convergence factor has to be smaller than $0.85^{10} \approx 0.2$, in order for the MAA approach to become competitive with WJAC. This has to be held up in a scalable way on parallel computers, which is difficult for large numbers of processors. AMG can achieve convergence factors of 0.1 for some problems, but it may be hard to develop an MAA method that can consistently achieve convergence factors of around 0.1 for general irreducible Markov chains, or for the subclass of general PageRank matrices (which in itself is a subclass of the Markov chains that have $|\lambda_2|$ bounded away from 1). However, MAA can be expected to achieve results far superior to the WJAC and Power methods for general Markov chains that have $|\lambda_2(n)| \to 1$ for large n. This was demonstrated for a regularization of the web graph that adds local backlinks with a small coupling probability.

In future work we plan to refine the strength-based adaptive agglomeration approach introduced in this paper. In particular, we want to investigate more systematically how it performs for general Markov matrices for which the second eigenvalue approaches one, and we intend to explore theoretically the convergence properties of the MAA method for general Markov chains. While we have shown in this paper that order of magnitude speedups can be achieved for certain Markov chain problems compared to traditional one-level iterative methods, it remains an open question whether the convergence factors of the MAA method are bounded away from one as problem size increases for general Markov chains.

Future work will also include parallelization of the MAA method. Efficient parallelizations of AMG methods exist that are nearly linearly scalable up to very large problem sizes with billions of unknowns on parallel computers with thousands of processors [22]. Many components of these existing parallel AMG implementations can be reused for parallelization of the MAA method. The WJAC relaxation used in this paper is fully parallelizable, but a parallel version of the MAA aggregation procedure needs to be developed.

REFERENCES

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

[2] M. Brezina, R. Falgout, S. MacLachlan, T. Manteuffel, S. McCormick, and J. Ruge, *Adaptive algebraic multigrid*, SIAM J. Sci. Comp. 27:1261-1286, 2006.

[3] A. Brandt, S. F. McCormick, and J. W. Ruge, *Algebraic multigrid (AMG) for sparse matrix equations*, in Sparsity and Its Applications, D. J. Evans, ed., Cambridge University Press, Cambridge, 1984.

[4] J. Ruge, *Algebraic multigrid (AMG) for geodetic survey problems*, in Proceedings of the International Multigrid Conference, Copper Mountain, CO, 1983.

[5] H.A. Simon and A. Ando, *Aggregation of variables in dynamic systems*, Econometrica 29:111-138, 1961.

[6] Yukio Takahashi, *A lumping method for numerical calculations of stationary distributions of Markov chains*, Research Report B-18, Department of Information Sciences, Tokyo

Institute of Technology, 1975.

[7] J.R. Koury, D.F. McAllister, and William J. Stewart, *Iterative Methods for Computing Stationary Distributions of Nearly Completely Decomposable Markov Chains*, SIAM Journal of Algebraic and Discrete Methods 5:164-186, 1984.

[8] William J. Stewart and W.L. Cao, *Iterative Aggregation/Disaggregation Techniques for Nearly Uncoupled Markov Chains*, Journal of the ACM 32:702-719, 1985.

[9] Paul J. Schweitzer and Kyle W. Kindle, *An iterative aggregation-disaggregation algorithm for solving linear equations*, Applied Mathematics and Computation 18:313-354, 1986.

[10] U. R. Krieger, B. Mller-Clostermann, and M. Szcittnick, *Modeling and Analysis of Communication Systems Based on Computational Methods For Markov Chains*, IEEE Journal on Selected Areas in Communication, 8-9:1630-1648, 1990.

[11] William J. Stewart, *An Introduction to the Numerical Solution of Markov Chains*, Princeton University Press, Princeton, 1994.

[12] U. R. Krieger, *On a two-level multigrid solution method for finite Markov chains*, Linear Algebra and its Applications 223/224:415-438, 1995.

[13] Ivo Marek and Petr Mayer, *Convergence analysis of an iterative aggregation/disaggregation method for computing stationary probability vectors of stochastic matrices*, Numerical Linear Algebra with Applications 5:253-274, 1998.

[14] Tuugrul Dayar and William J. Stewart, *Comparison of Partitioning Techniques for Two-Level Iterative Solvers on Large, Sparse Markov Chains*, SIAM J. Sci. Comput. 21:1691, 2000.

[15] Ivo Marek and Petr Mayer, *Convergence theory of some classes of iterative aggregation/disaggregation methods for computing stationary probability vectors of stochastic matrices*, Linear Algebra and its Applications 363:177-200, 2003.

[16] Amy N. Langville and Carl D. Meyer, *Updating the Stationary Vector of an Irreducible Markov Chain with an Eye on Google's PageRank*, SIAM Journal on Matrix Analysis 27:968-987, 2005.

[17] Yangbo Zhu, Shaozhi Ye, and Xing Li, *Distributed PageRank computation based on iterative aggregation-disaggregation methods*, Proceedings of the 14th ACM international conference on Information and knowledge management, 578-585, 2005.

[18] Graham Horton and Scott T. Leutenegger, *A Multi-Level Solution Algorithm for Steady-State Markov Chains*, ACM SIGMETRICS, 191-200, 1994.

[19] Scott T. Leutenegger and Graham Horton, *On the Utility of the Multi-Level Algorithm for the Solution of Nearly Completely Decomposable Markov Chains*, In W. Stewart, ed., Numerical solution of Markov chains, Kluwer Publishers, 425-443, 1995.

[20] U. R. Krieger, *Numerical solution of large finite Markov chains by algebraic multigrid techniques*, in W. Stewart, ed., Numerical solution of Markov chains, Kluwer Publishers, 403-424, 1995.

[21] Claudia Isensee and Graham Horton, *A Multi-Level Method for the Steady State Solution of Markov Chains*, Simulation und Visualisierung, SCS European Publishing House, 2004.

[22] Hans De Sterck, Ulrike Meier Yang, and Jeffrey J. Heys, *Reducing Complexity in Parallel Algebraic Multigrid Preconditioners*, SIAM Journal on Matrix Analysis and Applications 27:1019-1039, 2006.

[23] L. Page, S. Brin, R. Motwani and T. Winograd, *The PageRank Citation Ranking: Bringing Order to the Web*, Technical Report 1999-0120, Computer Science Department, Stanford, 1999.

[24] Sergey Brin and Lawrence Page, *The Anatomy of a Large-Scale Hypertextual Web Search Engine*, Computer Networks and ISDN Systems 33:107-117, 1998.

[25] Amy N. Langville and Carl D. Meyer, *A Survey of Eigenvector Methods of Web Information Retrieval*, SIAM Review 47:135-161, 2005.

[26] Amy N. Langville and Carl D. Meyer, *Deeper Inside PageRank*, Internet Mathematics 1:335-380, 2005.

[27] Taher H. Haveliwala and Sepandar D. Kamvar, *The Second Eigenvalue of the Google Matrix*, Technical Report 2003-0020, Computer Science Department, Stanford, 2003.

[28] Stanford Web Matrix, http://nlp.stanford.edu/∼sdkamvar/data/stanford-web.tar.gz.