ALGEBRAIC DISTANCE ON GRAPHS

JIE CHEN* AND ILYA SAFRO†

Abstract. Measuring the connection strength between a pair of vertices in a graph is one of the most important concerns in many graph applications. Simple measures such as edge weights may not be sufficient for capturing the effects associated with short paths of lengths greater than one. In this paper, we consider an iterative process that smooths an associated value for nearby vertices, and we present a measure of the local connection strength (called the algebraic distance, see [25]) based on this process. The proposed measure is attractive in that the process is simple, linear, and easily parallelized. An analysis of the convergence property of the process reveals that the local neighborhoods play an important role in determining the connectivity between vertices. We demonstrate the practical effectiveness of the proposed measure through several combinatorial optimization problems on graphs and hypergraphs.

Key words. Graph connectivity measure, combinatorial scientific computing, stationary iterative process

AMS subject classifications. 05C50, 05C85

1. Introduction.

Measuring the connectivity between two vertices in a graph is one of the central questions in many theoretical and applied areas in computer science. The edge weights are often used to measure how close the vertices are to each other. However, edge weights may not always be available or sufficient because of practical limitations, including a lack of other than Boolean information and the difficulty in determining all pairwise vertex relationships. In many graph algorithms, we often need to choose the most suitable pair of vertices, for example the "heaviest" edge in some sense. In such situations, without a better choice a greedy strategy might have to arbitrarily break the ties due to equal edge weights. A better performance might be possible if the strategy could recognize a pair of vertices that are not connected by an edge but pose a strong indirect connection.

Because of the practical significance of vertex connectivity, many algorithms have been proposed to model it. Examples include the length of the shortest path, the number of simple paths between a pair of vertices, maximum flows, and minimum edge cuts/vertex separators. In a random-walk approach [12, 30], the average first-passage time/cost and average commute time were used. A similarity measure between nodes of a graph integrating indirect paths, based on the matrix-forest theorem, was proposed in [5]. Effective resistance [15] was used to model a connectivity with electrical circuit conductance. A convergence of the compatible relaxation [2] was measured in algebraic multigrid (AMG) schemes [4] in order to detect strong connections between fine and coarse points. A similarity method based on a probabilistic interpretation of a diffusion was introduced in [23]. Our goal is to design a family of algorithms (and measures) that are fast and easy to implement and parallelize and that can be applied locally to the data.

*Department of Computer Science and Engineering, University of Minnesota at Twin Cities, Minneapolis, MN 55455. Current address: Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, IL 60439. Email: jiechen@mcs.anl.gov. Work of this author was supported by NSF grant DMS-0810938, a University of Minnesota Doctoral Dissertation Fellowship, and the CSCAPES institute, a U.S. DOE project.

†Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, IL 60439. Email: safro@mcs.anl.gov. This work was funded by the CSCAPES institute, a U.S. DOE project, and in part by U.S. DOE Contract DE-AC02-06CH11357.

The proposed measure is called the *algebraic distance*. We will give a formal definition (inherited from [25]) in Section 2; for now we note that it is based on a stationary iterative process that smooths an associated value for nearby vertices. The process can also be considered one that propagates some information about every vertex to its neighborhood. After a few iterations, the propagated values define a distance between all pairs of nodes. Conceptually, a small distance means a strong connection because, by the propagation, closely connected vertices will converge to similar values.

The algebraic distance is motivated by the bootstrap algebraic multigrid (BAMG) method [3] for solving linear systems $Ax = b$. In the heart of any multigrid [29] lies a coarsening process that creates a hierarchy of projections of the original problem domain onto smaller spaces. In contrast to the geometric multigrid that exploits a regular geometric pattern (of the underlying domain) in choosing coarse variables, AMG creates a coarse system by automatic exploration of the "geometry" behind the problem using sophisticated rules of "closeness" between variables. Traditional AMG approaches typically use edge weights as an indicator of the closeness. While this is frequently acceptable, there are situations where they don't yield satisfactory results. On the other hand, BAMG defines such a closeness by running several Gauss-Seidel (GS) relaxations with a random initial vector on the corresponding homogeneous system $Ax = 0$. The speed of convergence of the iterate x signifies the closeness between variables, and it is used to determine the rules of aggregation and interpolation. This paper considers a similar process, where GS is replaced by Jacobi overrelaxation (JOR) and A is replaced by the graph Laplacian L.

Recently, the algebraic distance was used as a component of an AMG-based coarsening scheme for graph linear ordering problems [25]. Despite considerable empirical evidence of success in multilevel linear ordering and partitioning algorithms, where the algebraic distance was used as a tool for separating long and short-ranged edges in multilevel aggregation and interpolation, however, the concept of algebraic distance is still not well understood and has not been used widely in combinatorial optimization problems. This paper studies some properties of this relaxation process and interprets the algebraic distances under a mutually influenced environment model, where the neighborhood connectivity information governs the connectivity of the vertices. With this interpretation, the applications of this measure are no longer restricted to multilevel algorithms. Whenever the concept of vertex connectivity is applicable, we can use the algebraic distance to measure the connection strengths between vertices. We show a few such applications in this paper.

2. Notation and preliminaries. Let $G = (V, E)$ denote a weighted undirected graph, where the set of vertices (nodes) V is $\{1, 2, ..., n\}$ and E is the set of edges. Let $W = [w_{ij}]_{n\times n}$ be the weighted adjacency matrix of G, where $w_{ij} \geq 0$ is the weight of the undirected edge ij between nodes i and j; if $ij \notin E$, then $w_{ij} = 0$. Algorithm 1 updates a vector x from a random initialization $x^{(0)}$. We use superscripts to distinguish successive iterates and subscripts to mean vector entries.

We define $s_{ij}^{(k)}$, the *algebraic distance* between vertices i and j, at the kth iteration, to be

$$s_{ij}^{(k)} := \left| x_i^{(k)} - x_j^{(k)} \right|. \tag{2.1}$$

With R initial vectors $x^{(0,r)}$, $r = 1, \ldots, R$, each vector is independently updated by

Algorithm 1 Computing algebraic distances for graphs

Input: Parameter ω, initial vector $x^{(0)}$

1: **for** $k = 1, 2, \ldots$ **do**
2: $\quad \tilde{x}_i^{(k)} \leftarrow \sum_j w_{ij} x_j^{(k-1)} / \sum_j w_{ij}, \ \forall i.$
3: $\quad x^{(k)} \leftarrow (1-\omega) x^{(k-1)} + \omega \tilde{x}^{(k)}$
4: **end for**

using Algorithm 1, and the *extended p-normed algebraic distance* $\varrho_{ij}^{(k)}$ is defined as

$$\varrho_{ij}^{(k)} := \left(\sum_{r=1}^{R} \left| x_i^{(k,r)} - x_j^{(k,r)} \right|^p \right)^{1/p}, \tag{2.2}$$

where the superscript $^{(k,r)}$ refers to the kth iteration on the rth initial random vector. For $p = \infty$, by convention, $\varrho_{ij}^{(k)} = \max_{r=1,\ldots,R} \left| x_i^{(k,r)} - x_j^{(k,r)} \right|$.

The analysis in this paper is based on the spectral graph theory, whereby the spectrum and the eigenvectors of the *graph Laplacian* play a central role. The graph Laplacian matrix is defined as $L = D - W$, where D is the diagonal matrix with diagonal elements d_i equal to $\sum_j w_{ij}$. It is positive semi-definite. Let (λ_i, u_i) denote the eigen-pairs of L ordered in nondecreasing order of λ_i:

$$Lu_i = \lambda_i u_i, \qquad i = 1, 2, \ldots, n. \tag{2.3}$$

It is well known that the smallest eigenvalue(s) of L is zero, and the multiplicity of $\lambda_1 = 0$ is c if and only if the graph has c connected components. In particular, if the graph is connected, λ_1 is simple. Then the smallest nonzero eigenvalue λ_2 is called the *algebraic connectivity* of the graph, and the corresponding eigenvector u_2 is called the *Fiedler vector*.

We will also need to consider the matrix pencil (L, D), which has the same spectrum as the *normalized Laplacian* $\mathcal{L} = D^{-1/2} L D^{-1/2}$. Let (μ_i, v_i) denote the eigenpairs of (L, D) ordered in nondecreasing order of μ_i:

$$Lv_i = \mu_i D v_i, \qquad i = 1, 2, \ldots, n. \tag{2.4}$$

Similar to L, the multiplicity of the zero eigenvalues of (L, D) is the number of connected components of the graph. On the other hand, the largest eigenvalue $\mu_n \leq 2$. In particular, $\mu_n = 2$ if and only if there exists a connected component of the graph that is bipartite. Note that the second smallest eigenvalue $\mu_2 \leq 1$ whenever the graph is not complete. For a comprehensive treatment of the spectral properties of the normalized Laplacian, see, e.g., [6].

It is obvious that Algorithm 1 is the JOR iteration for solving the linear system

$$Lx = 0, \tag{2.5}$$

using the relaxation parameter ω. Of course, we are not interested in actually "solving" this system, which has infinitely many solutions, but the convergence properties of the iteration is key to the understanding of the algebraic distance. A basic result is that JOR converges for any $0 < \omega < 2/\mu_n$. Most results are in the next.

3. Iterative methods for graph Laplacians. Standard iterative methods by matrix splitting for solving a linear system can be written in a general form

$$x^{(k+1)} = Hx^{(k)}, \qquad k = 0, 1, 2, \ldots, \tag{3.1}$$

where H is the iteration matrix. Let the Laplacian $L = D - W_L - W_U$, where W_L and W_U are the strict lower and upper triangular parts of W, respectively. Then the iteration matrices for Gauss-Seidel, Jacobi, SOR, and JOR are, respectively,

$$H_{GS} = (D - W_L)^{-1} W_U, \qquad H_{SOR} = (D/\omega - W_L)^{-1} \left((1/\omega - 1)D + W_U\right),$$
$$H_{JAC} = D^{-1}(W_L + W_U), \qquad H_{JOR} = (D/\omega)^{-1} \left((1/\omega - 1)D + W_L + W_U\right).$$

We will use the notation H when the discussions are general or apply to all the iterative methods; we will add subscripts when an individual method is emphasized.

A matrix $A \in \mathbb{R}^{n\times n}$ is said to be *convergent* if $\lim_{k\to\infty} A^k$ exists.[1] Let α_i denote the eigenvalues of A, where $i = 1, \ldots, n$. Then, A is convergent if and only if (i) $|\alpha_i| \le 1$, (ii) $|\alpha_i| = 1$ implies $\alpha_i = 1$, and (iii) the algebraic multiplicity of the eigenvalue 1 equals its geometric multiplicity. In other words, the Jordan canonical form of a convergent matrix looks like $\left[\begin{smallmatrix} I_{t\times t} & 0 \\ 0 & J \end{smallmatrix}\right]$, where $t \ge 0$ is the algebraic/geometric multiplicity of the eigenvalue 1, and J consists of Jordan blocks for all the other eigenvalues $|\alpha_i| < 1$. The following theorem implies that unless the graph has a bipartite connected component and Jacobi iterations are used (or equivalently JOR with $\omega = 1$), the iteration matrix H for the system (2.5) is always convergent. A proof of the theorem can be found in [10].

THEOREM 3.1. *The iteration matrix H for the linear system* (2.5) *has the following properties.*

(i) *The matrices H_{GS}, H_{SOR} and H_{JOR} are convergent.*

(ii) *The matrix H_{JAC} is convergent if and only if none of the connected components of the graph is bipartite.*

(iii) *The spectral radii of H_{GS}, H_{JAC}, H_{SOR}, and H_{JOR} are all* 1.

When H is convergent, we denote its similarity transform to the Jordan canonical form as $PHP^{-1} = \left[\begin{smallmatrix} I_{t\times t} & 0 \\ 0 & J \end{smallmatrix}\right]$, where P is nonsingular. The initial vector $x^{(0)}$ can be uniquely decomposed as the sum of a vector in range$(I - H)$ and a vector z in null$(I - H)$, i.e., $x^{(0)} = (I - H)y + z$. Since $z \in$ null$(I - H)$, we have $Hz = z$. Therefore,

$$\begin{aligned} x^{(k)} &= H^k x^{(0)} = H^k(I - H)y + H^k z \\ &= P^{-1} \begin{bmatrix} I_{t\times t} & 0 \\ 0 & J^k \end{bmatrix} \begin{bmatrix} 0 & 0 \\ 0 & I - J \end{bmatrix} Py + z. \end{aligned}$$

Note that the limit of J^k is zero. Thus, $x^{(k)} \to z$. This result is summarized in the following theorem.

THEOREM 3.2. *When the iteration matrix H for the linear system* (2.5) *is convergent, the iterative process* (3.1) *converges. In such a case, the iterate $x^{(k)}$ converges to zero if the initial vector $x^{(0)} \in$ range$(I - H)$; otherwise $x^{(k)}$ converges to a vector in* null$(I - H)$.

[1] Some authors (e.g., [17]) use the term *convergent* for a matrix A where the limit A^k is zero. However, the interesting case in this paper is that the limit is nonzero. Thus, we make a broader inclusion in the definition here.

In particular, if the graph is connected, the eigenvalue 1 of H is simple, and the subspace null$(I-H)$ is spanned by the vector $\mathbf{1}$. Hence in such a case, $x^{(k)}$ converges to either zero or a nonzero scalar multiple of $\mathbf{1}$. An immediate consequence is that the algebraic distance $s_{ij}^{(k)}$ for all pair (i,j) vanishes as k goes to infinity.

As a result, the definition of the algebraic distance as a measure of the connection strength seems to be meaningless. However, we are actually interested in comparing the relative magnitudes of $s_{ij}^{(k)}$ for different (i,j) pairs. In other words, a concurrent scaling of the quantity $s_{ij}^{(k)}$ for all i and j will not compromise the measure.

To derive a suitable scaling, we consider a connected graph and make an additional mild assumption that H is diagonalizable. Let (σ_i, ϕ_i) denote the eigen-pairs of H:

$$H\phi_i = \sigma_i \phi_i, \qquad i = 1, \ldots, n, \tag{3.2}$$

where the eigenvalues are labeled in the order

$$1 = \sigma_1 > |\sigma_2| \geq |\sigma_3| \geq \cdots \geq |\sigma_n|$$

according to Theorem 3.1 (note the absence of the absolute value sign surrounding σ_1 and the strict inequality after σ_1). Correspondingly, the eigenvector $\phi_1 = \mathbf{1}$. Let the initial vector $x^{(0)}$ be expressed as a linear combination of the eigenvectors:

$$x^{(0)} = a_1\phi_1 + a_2\phi_2 + \cdots + a_n\phi_n. \tag{3.3}$$

Then, the kth iterate $x^{(k)} = H^k x^{(0)} = a_1\phi_1 + a_2\sigma_2^k\phi_2 + \cdots + a_n\sigma_n^k\phi_n$. The algebraic distance is then

$$s_{ij}^{(k)} = \left|(e_i - e_j)^T x^{(k)}\right| = \left|(e_i - e_j)^T \sum_{\ell=2}^{n} a_\ell \sigma_\ell^k \phi_\ell\right|,$$

where e_i is the ith column of the identity matrix. Note that the summation starts from $\ell = 2$. Since σ_2^k is a common factor for all the (i,j) pairs, we define the quantity

$$\hat{s}_{ij}^{(k)} := s_{ij}^{(k)}/\sigma_2^k \tag{3.4}$$

and name it *scaled algebraic distance*. It turns out that, in contrast to $s_{ij}^{(k)}$, the scaled quantity $\hat{s}_{ij}^{(k)}$ does not always converge. When it does, however, it converges to some value other than zero. For this, we consider two cases. The first case is that σ_2 is real, and $-\sigma_2$ is not an eigenvalue of H. This case occurs for most of the real-life graphs, and $\hat{s}_{ij}^{(k)}$ converges in such a case. The second case, which is the complement of the first case, also happens for some graphs. A special situation of this case is that σ_2 and σ_3 are a conjugate pair, which makes $\hat{s}_{ij}^{(k)}$ "oscillate".

THEOREM 3.3. *Assume that the graph is connected, with the iteration matrix H convergent and diagonalizable. Let the initial vector $x^{(0)}$ be expanded in the eigenbasis of H as in* (3.3).

(i) *If $\sigma_2 = \sigma_3 = \cdots = \sigma_t$ and $|\sigma_t| > |\sigma_{t+1}|$ for some $t \geq 2$, and if $a_2, \ldots, a_t$ are not all zero, then the quantity $\hat{s}_{ij}^{(k)}$ defined in* (3.4) *approaches the limit $\left|(e_i - e_j)^T\xi\right|$ in the order $O\left(\left|\sigma_{t+1}/\sigma_t\right|^k\right)$, where ξ is an eigenvector corresponding to the eigenvalue σ_2 (with multiplicity $t-1$).*

(ii) *If* $|\sigma_2| = |\sigma_3| = \cdots = |\sigma_t| > |\sigma_{t+1}|$ *for some* $t \geq 3$, *where* $\sigma_2, \ldots, \sigma_t$ *are not all the same,* $a_2, \ldots, a_t$ *are not all zero, and if there exists an integer* m *such that* $(\sigma_\ell/\sigma_2)^m = 1$ *for* $\ell = 3, \ldots, t$, *then the* p*th subsequence* $\{\hat{s}_{ij}^{(mk+p)}\}_{k=0,1,2,\ldots}$ *approaches the limit* $\left|(e_i - e_j)^T \eta_p\right|$ *in the order* $O\left(|\sigma_{t+1}/\sigma_t|^{mk}\right)$, *where* $\eta_p = a_2\phi_2 + a_3(\sigma_3/\sigma_2)^p\phi_3 + \cdots + a_t(\sigma_t/\sigma_2)^p\phi_t$ *for* $p = 0, 1, \ldots, (m-1)$.

Proof. Case (i): Equation (3.4) becomes $\hat{s}_{ij}^{(k)} = \left|(e_i - e_j)^T \xi^{(k)}\right|$, where

$$\xi^{(k)} = a_2\phi_2 + \cdots + a_t\phi_t + \sum_{\ell=t+1}^{n} a_\ell \left(\frac{\sigma_\ell}{\sigma_2}\right)^k \phi_\ell.$$

When k tends to infinity, the summation term in $\xi^{(k)}$ vanishes. Let $\xi = a_2\phi_2 + \cdots + a_t\phi_t$, then $\hat{s}_{ij}^{(k)} \to \left|(e_i - e_j)^T \xi\right|$, and $\hat{s}_{ij}^{(k)}$ converges in the order $O(|\sigma_{t+1}/\sigma_2|^k) = O(|\sigma_{t+1}/\sigma_t|^k)$.

Case (ii): Equation (3.4) becomes $\hat{s}_{ij}^{(k)} = \left|(e_i - e_j)^T \eta^{(k)}\right|$, where

$$\eta^{(k)} = a_2\phi_2 + a_3\tau_3^k\phi_3 + \cdots + a_t\tau_t^k\phi_t + \sum_{\ell=t+1}^{n} a_\ell \left(\frac{\sigma_\ell}{\sigma_2}\right)^k \phi_\ell,$$

and $\tau_\ell = \sigma_\ell/\sigma_2$ for $\ell = 3, \ldots, t$. If there exists a positive integer m such that $\tau_\ell^m = 1$ for all ℓ, then $\eta^{(k)}$ has m limit points $\eta_0, \ldots, \eta_{m-1}$. ∎

4. Further results for Jacobi overrelaxations. For JOR, the eigenvalues and vectors of the iteration matrix H_{JOR} are closely related to those of the matrix pencil (L, D). Thus, results in the preceding section can be further polished by using properties of the graph Laplacian. In particular, the relaxation parameter ω can be chosen such that the eigenvector ϕ_2 coincides with the eigenvector corresponding to the second smallest eigenvalue of (L, D). Meanwhile, because of the distributions of the eigenvalues, the convergence of Algorithm 1 often is slow. Hence, we in addition study the behavior of the iterations at an early stage.

4.1. The limiting case. Observe the following equivalence:

$$H_{JOR}\ \phi_i = \sigma_i\phi_i \iff L\phi_i = \frac{1-\sigma_i}{\omega} D\phi_i.$$

An immediate result is that H_{JOR} is diagonalizable and all the eigenvalues of H_{JOR} are real. More precisely, if μ_j is an eigenvalue of (L, D), then $\mu_j = (1-\sigma_i)/\omega$ for some i. In general, we may not have the exact correspondence $j = i$, since the eigenvalues of H_{JOR} are sorted in the order of their absolute values, whereas the eigenvalues of (L, D) are sorted in their natural order. Figure 4.1 pictorially shows the relative positions of the eigenvalues σ_i of H_{JOR}. Clearly $\sigma_1 = 1 - \omega\mu_1 = 1$ regardless of the value ω, since $\mu_1 = 0$. However, σ_2 can be either $1 - \omega\mu_2$ or $1 - \omega\mu_n$, depending on the choice of ω. A special case is that $1 - \omega\mu_2 = -(1 - \omega\mu_n)$. In this case, $\sigma_2 = -\sigma_3$, and case (ii) of Theorem 3.3 indicates that the scaled algebraic distance $\hat{s}_{ij}^{(k)}$ will not converge; rather, it oscillates when k is large. Otherwise, we enumerate all the other possible cases for ω, and have the following theorem as a corollary of case (i) of Theorem 3.3.

THEOREM 4.1. *Given a connected graph, let* (μ_i, v_i) *be the eigen-pairs of the matrix pencil* (L, D), *labeled in nondecreasing order of the eigenvalues, and assume*

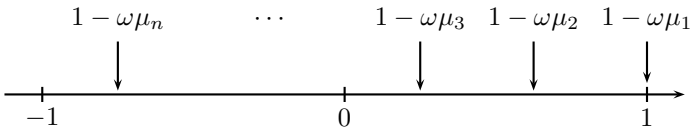


FIG. 4.1. *Pictorial description of the locations of the eigenvalues of* H_{JOR}.

that $\mu_2 \neq \mu_3 \neq \mu_{n-1} \neq \mu_n$. *Unless* $\omega = 2/(\mu_2 + \mu_n)$, *the quantity* $\hat{s}_{ij}^{(k)}$ *defined in* (3.4) *will always converge to a limit* $|(e_i - e_j)^T \xi|$ *in the order* $O(\theta^k)$, *for some* ξ *and* $0 < \theta < 1$.

(i) *If* $0 < \omega < 2/(\mu_3 + \mu_n)$, *then* $\xi \in \text{span}\{v_2\}$ *and* $\theta = (1 - \omega\mu_3)/(1 - \omega\mu_2)$;

(ii) *If* $2/(\mu_3 + \mu_n) \leq \omega < 2/(\mu_2 + \mu_n)$, *then* $\xi \in \text{span}\{v_2\}$ *and* $\theta = -(1 - \omega\mu_n)/(1 - \omega\mu_2)$;

(iii) *If* $2/(\mu_2 + \mu_n) < \omega < \min\{2/(\mu_2 + \mu_{n-1}), 2/\mu_n\}$, *then* $\xi \in \text{span}\{v_n\}$ *and* $\theta = -(1 - \omega\mu_2)/(1 - \omega\mu_n)$;

(iv) *If* $2/(\mu_2 + \mu_{n-1}) \leq \omega < 2/\mu_n$, *then* $\xi \in \text{span}\{v_n\}$ *and* $\theta = (1 - \omega\mu_{n-1})/(1 - \omega\mu_n)$.

A graphical illustration of the dependence of θ on ω is shown in Figure 4.2.

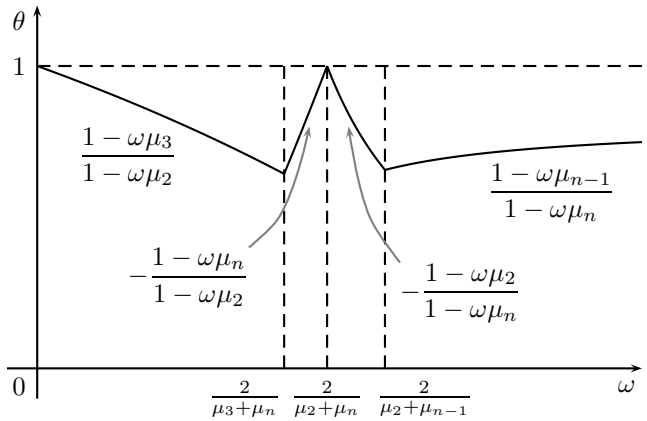


FIG. 4.2. *The* θ *as a function of* ω. *Note that the value* $2/\mu_n$ *can be less than, equal to, or greater than* $2/(\mu_2 + \mu_{n-1})$.

Proof. In case (i), $-(1-\omega\mu_n) < (1-\omega\mu_3) < (1-\omega\mu_2)$; therefore we have $\phi_2 = v_2$, $\sigma_2 = 1 - \omega\mu_2 > 0$, and $\sigma_3 = 1 - \omega\mu_3 > 0$. In case (ii), $(1 - \omega\mu_3) < -(1 - \omega\mu_n) < (1 - \omega\mu_2)$; therefore $\phi_2 = v_2$, $\sigma_2 = 1 - \omega\mu_2 > 0$, and $\sigma_3 = 1 - \omega\mu_n < 0$. In case (iii), $-(1 - \omega\mu_{n-1}) < (1 - \omega\mu_2) < -(1 - \omega\mu_n)$; therefore $\phi_2 = v_n$, $\sigma_2 = 1 - \omega\mu_n < 0$, and $\sigma_3 = 1 - \omega\mu_2 > 0$. In case (iv), $(1 - \omega\mu_2) < -(1 - \omega\mu_{n-1}) < -(1 - \omega\mu_n)$; therefore $\phi_2 = v_n$, $\sigma_2 = 1 - \omega\mu_n < 0$, and $\sigma_3 = 1 - \omega\mu_{n-1} < 0$. The theorem is established by following case (i) of Theorem 3.3. ∎

Sometimes, $\mu_2 = \mu_3$ or $\mu_{n-1} = \mu_n$, for graphs from real-life problems. Thus, Theorem 4.1 does not apply. However, we can use the same technique as in the proof to analyze the convergence of $\hat{s}_{ij}^{(k)}$, by checking the possible values of σ_2 and σ_3.

The above theorem shows two possible limits (v_2 or v_n) for $\hat{s}_{ij}^{(k)}$ depending on the choice of the value ω, which in turn is related to the eigenvalues μ_i. In practice,

the eigenvalues are not numerically computed, but we can analytically derive some upper/lower bounds for the cutting point $2/(\mu_2+\mu_n)$ and estimate which of the cases in Theorem 4.1 applies. A simple bound exploits the fact that $\mu_2 \le \mu_n \le 2$; thus $2/(\mu_2+\mu_n) \ge 1/2$, which indicates that for any $\omega < 1/2$, case (i) or (ii) applies. When the graph is not complete, a slightly better bound is $2/(\mu_2+\mu_n) \ge 2/3$, since in such a case $\mu_2 \le 1$. For more sharper bounds of μ_2 and μ_n, see, for example, [6]. Since in practice we deal with sparse graphs and set $\omega = 1/2$, the quantity $\hat{s}_{ij}^{(k)}$ always converges to $|(e_i - e_j)^T \xi|$ with $\xi \in \text{span}\{v_2\}$.

4.2. At early iterations. For real-life graphs, the θ corresponding to $\omega = 1/2$ is so close to 1 that the theoretical convergence of $\hat{s}_{ij}^{(k)}$ is of little practical use—it takes an enormous number of steps before it gets close enough to the limit. (As observed, θ often can be as high as 0.9999.) However, an interesting phenomenon is that in practice $x^{(k)}$ soon becomes "stable"; that is, the two iterates $x^{(k+1)}$ and $x^{(k)}$ are almost parallel even when k is small.

To make the above statement precise, we want to measure the angle between two consecutive iterates. Specifically, we are interested in how close

$$1 - \left\langle \frac{x^{(k)}}{\left\|x^{(k)}\right\|}, \frac{x^{(k+1)}}{\left\|x^{(k+1)}\right\|} \right\rangle^2$$

is to zero (this is the squared sine of the angle between $x^{(k)}$ and $x^{(k+1)}$). Note that even though $x^{(k)}$ and $x^{(k+1)}$ become close to parallel at early iterations, it does not necessarily mean that $x^{(k)}$ has converged, nor has the quantity $s_{ij}^{(k)}$ or $\hat{s}_{ij}^{(k)}$.

THEOREM 4.2. *Given a graph, let (μ_i, v_i) be the eigen-pairs of the matrix pencil (L, D), labeled in nondecreasing order of the eigenvalues. Denote $V = [v_1, \ldots, v_n]$. Let $x^{(0)}$ be the initial vector of the JOR process, and let $a = V^{-1}x^{(0)}$ with $a_1 \neq 0$. If the following two conditions are satisfied:*

$$1 - \omega\mu_n \ge 0, \tag{4.1a}$$

$$f_k := \frac{\alpha {r_k}^{2k}(1-r_k)^2}{1+\alpha {r_k}^{2k}(1+r_k)^2} \le \frac{1}{\kappa}, \tag{4.1b}$$

where $\alpha = \left(\sum_{i\neq 1} a_i^2\right) / \left(4a_1^2\right)$, r_k is the unique root of the equation

$$2\alpha r^{2k+2} + 2\alpha r^{2k+1} + (k+1)r - k = 0 \tag{4.2}$$

in the interval $[0,1]$, and κ is the condition number of D, then

$$1 - \left\langle \frac{x^{(k)}}{\left\|x^{(k)}\right\|}, \frac{x^{(k+1)}}{\left\|x^{(k+1)}\right\|} \right\rangle^2 \le \frac{4\kappa f_k}{(1+\kappa f_k)^2}. \tag{4.3}$$

Since the proof of the above theorem is long and technical, it is deferred to the Appendix. We address several important issues in this theorem. First, since we use $\omega = 1/2$, condition (4.1a) is satisfied. Second, f_k is defined as a rational polynomial of r_k, which is the unique root of the polynomial (4.2) in the interval $[0,1]$. Therefore, f_k can be easily evaluated and it is typically close to zero. For example, when $\alpha = 100$ and $k = 50$, we have $r_k = 0.9475$, which gives $f_k = 4.6 \times 10^{-4}$. Third, the condition

number κ of D is usually not large. For many graphs arising from application areas such as VLSI design and finite-element meshes, if the graph edges have a uniform weight equal to 1, then d_i is the degree of a vertex, and thus for the whole graph the vertex degrees may not vary too much.[2] All this means is that condition (4.1b) is not a strong requirement. The final bound in (4.3), for $k = 30$ or 50, typically drops to the order of 10^{-4}. Note that $\sin^2(\pi/180) = 3.05 \times 10^{-4}$, which indicates that the angle between $x^{(k)}$ and $x^{(k+1)}$ is $O(1^\circ)$.

Of course, not every graph with an arbitrary initialization will yield $x^{(k)}$ close to parallel to $x^{(k+1)}$ for a small k. Hence, some assumptions, such as the ones in Theorem 4.2, are needed. On closing this section, we present a "bad" example. In this example, for any k we can construct a corresponding initialization such that $x^{(k)}$ and $x^{(k+1)}$ are far from parallel.

EXAMPLE 4.3. (An adversary example.) Consider a graph with only two vertices and an edge between them. The JOR iteration matrix for this graph is

$$H_{JOR} = \begin{bmatrix} 1-\omega & \omega \\ \omega & 1-\omega \end{bmatrix}.$$

Its two eigenvalues are $\sigma_1 = 1$ and $\sigma_2 = 1 - 2\omega$. Given an initialization $x^{(0)} = [1+\delta, -1+\delta]^T$, the iterations generate the iterates $x^{(k)} = [\sigma_2^k + \delta, -\sigma_2^k + \delta]^T$. Then,

$$1 - \left\langle \frac{x^{(k)}}{\left\|x^{(k)}\right\|}, \frac{x^{(k+1)}}{\left\|x^{(k+1)}\right\|} \right\rangle^2 = \frac{(1-\sigma_2)^2}{1 + \sigma_2^2 + \left(\dfrac{\sigma_2^{2k+2}}{\delta^2} + \dfrac{\delta^2}{\sigma_2^{2k}} \right)}.$$

If we choose $\delta = \sigma_2^{k+0.5} = (1-2\omega)^{k+0.5}$, then the above equation becomes

$$1 - \left\langle \frac{x^{(k)}}{\left\|x^{(k)}\right\|}, \frac{x^{(k+1)}}{\left\|x^{(k+1)}\right\|} \right\rangle^2 = \frac{(1-\sigma_2)^2}{(1+\sigma_2)^2} = \left(\frac{\omega}{1-\omega} \right)^2.$$

When ω approaches $1/2$, the angle between $x^{(k)}$ and $x^{(k+1)}$ can be arbitrarily close to 90°. In other words, when the relaxation parameter ω is not small, for any k, we can choose an appropriate δ such that $x^{(k)}$ and $x^{(k+1)}$ are far from parallel. An example set of such parameters is

$$\omega = 1/8, \qquad k = 20, \qquad \delta = 0.75^{20.5} \approx 0.0027.$$

In this case, the angle between $x^{(20)}$ and $x^{(21)}$ is 8.21°.

5. Connections to spectral partitioning and random walks. Analysis in the preceding section indicates that the scaled algebraic distance $\hat{s}_{ij}^{(k)}$, at convergence, resorts to the second eigenvector v_2 of the matrix pencil (L, D), assuming we adopt the practical choice $\omega = 1/2$. Eigenvector approaches constitute a large component in both theoretical analysis and practical applications of graphs. In particular, the first or second eigenvectors of some matrices related to the graph are widely used in spectral techniques. In this section, we draw connections and distinctions to our approach and two other subjects: spectral partitioning and random walks.

[2] This may not be true for power-law graphs.

The general goal of partitioning a graph is to minimize the edges that cross the two partitions, subject to some size constraints. There exist many variants in the problem formulation (in Section 8 we consider a different one from the one here). A classical formulation is the graph bisection—to cut the vertex set in two parts π and $V\backslash\pi$ such that the sum of the weights of the edges along the cut is minimized; that is,

$$\min_{\pi} f(\pi) = \sum_{i\in\pi, j\in V\backslash\pi} w_{ij},$$

while enforcing the sizes of the two partitions to be the same, that is, $|\pi| = |V\backslash\pi|$. It turns out that if we define a vector q, where $q_i = 1$ if $i \in \pi$ and $q_i = -1$ otherwise, then the objective function becomes $f = \frac{1}{4}q^T Lq$, and the constraint is $q^T\mathbf{1} = 0$. Therefore, an approximate solution is to let $q = u_2$, the second eigenvector of L, and split the entries of the vector q by their median. A slight variant of the formulation, the *normalized cut* [27, 24, 22], uses the eigenvector v_2 as an approximate solution.

The corresponding eigenvalue of the second eigenvector of, either L or (L, D), plays a central role in characterizing the *global* connections between the vertices in the graph. Apart from the basic fact that the magnitudes of λ_2 and μ_2 indicate the connectivity of the graph (i.e., $\lambda_2, \mu_2 \neq 0$ if and only if the graph is connected), μ_2 also bounds the Cheeger's constant

$$h_G := \frac{\sum_{i\in\pi, j\in V\backslash\pi} 1}{\min\left\{\sum_{i\in\pi} d_i,\ \sum_{j\in V\backslash\pi} d_j\right\}},$$

which in a rough term indicates how high a cost will be even when the graph is optimally partitioned, as in the Cheeger's inequality [6]

$$h_G^2/2 < \mu_2 \leq 2h_G.$$

We stress that this is a *global* property of the graph, and it does not represent the local variations of the vertex connectivity.

In contrast to using the second eigenvector in spectral partitioning, the random walk model emphasizes the dominant eigenvector. To be more general, we consider a lazy random walk, where the walker does not change states with probability $1 - \omega$; when he does, the transition probability from state i to state j is w_{ij}/d_i. Thus, the random walk matrix is $(1 - \omega)I + \omega D^{-1}W$, which happens to be the iteration matrix H_{JOR}. Therefore, the stationary distribution of the walk is the dominant left eigenvector of H_{JOR}, and this vector signifies the probability of a random walker landing on each vertex in the final steady state. The convergence rate to the steady state (by iteratively applying a vector to the left-hand side of the matrix) is given by $O(|\sigma_2|^k)$. Indeed, the idea of iteratively applying a vector to the matrix can be exploited to measure a connection of the vertices and to perform a local partitioning of the graph [28, 1]. Our case bears some similarities to the random walk; in particular, we also repeatedly apply a vector on the matrix H_{JOR}. However, the vector is applied to the right of the matrix, and it is not the limit of the vector but, rather, the limit of the entries when the vector undertakes a nontrivial scaling, that actually defines the connectivity measure. The limit of interest is ϕ_2, the second right eigenvector of H_{JOR}, and the rate of convergence is $O(|\sigma_3/\sigma_2|^k)$. In the next section, we offer an explanation of the meaning of this iterative process.

6. Mutually influenced model and the local effect. A practical attraction of the algebraic distance defined based on Algorithm 1 is that the iteration is never run until convergence. A direct saving is in the computational cost. Of course, if one really needs the eigenvector v_2, a more practical method (such as the Lanczos algorithm [20, 26, 16]) will need to be employed; nevertheless, a few steps of JOR are still more efficient than Lanczos. Regardless of the computational aspects, we see that spectral theories based on eigenvectors may not explain why such a "premature" termination is sufficient to yield a good distance estimate. In this section, we consider a mutually influenced model that captures the essence of the measure of a *local* effect.

In a mutually influenced environment, entities are influenced by their neighbors. Therefore, two entities are close, or similar, if they are placed in two similar neighborhoods. In this case, we say that the entities have a strong connection. Concretely, we consider the graph as such an environment, and we denote x_i a quantity associated with each entity/vertex i. We need to build a model for all the x_i's such that the absolute difference $|x_i - x_j|$ indicates the strength of the connection between i and j. Since each vertex is influenced by its neighborhood, we quantify that a $1-\mu$ portion of x_i is a weighted combination of its neighbors x_j. The weight of the influence is naturally indicated by the edge weight w_{ij}, normalized by the sum $\sum_j w_{ij} = d_i$. Therefore, the model is

$$x_i = \mu x_i + \sum_j \frac{w_{ij}}{d_i} x_j \qquad 0 \le \mu \le 1. \tag{6.1}$$

Note that in the model, μ is not a free parameter; rather, it is an innate property of the graph. It indicates in such an environment how strongly a neighborhood acts on each entity. When μ is close to zero, the neighborhood plays a major role, whereas when μ tends to one, a vertex is so stubborn that its neighbors cannot have a strong impact on it. It turns out that in the matrix form, (6.1) is equivalent to

$$Lx = \mu D x. \tag{6.2}$$

The JOR process as presented in Algorithm 1 computes an exact solution for the model at convergence, and it yields an approximate solution even when the iteration is prematurely terminated. According to Theorem 4.1, in the limit, the scaled algebraic distance $\hat{s}_{ij}^{(k)}$ converges to a value proportional to $\left|(e_i - e_j)^T v_2\right|$. In other words, we have $\mu = \mu_2$ and $x = v_2$ in (6.2). On the other hand, even when far from convergence, the normalized iterate $\hat{x}^{(k)} = x^{(k)}/\|x^{(k)}\|$ approximately satisfies the model with $\mu = \mu_2$ and $x = \hat{x}^{(k)}$, namely, $L\hat{x}^{(k)} \approx \mu_2 D\hat{x}^{(k)}$, thanks to the stabilization result of $x^{(k)}$ in Theorem 4.2.

We remark that for a small k the iterate $x^{(k)}$ can be quite different from v_2 and then for different initializations $x^{(k)}$ will be different. However, they all satisfy (or approximately satisfy) the mutually influenced model (6.1). This feature gives us the flexibility to interpret vertex connectivity. The JOR process can be considered one that smooths the associated value for nearby vertices in each iteration. Therefore, in k steps, the values x_i cannot be propagated more than k hops away, and this smoothing process affects only small neighborhoods if k is not large. Thus, the resulting iterate $x^{(k)}$ measures only local influences, and the impact of faraway vertices can come in only when k becomes sufficiently large. As k increases, the range of local impact also expands, and thus the concept of "local connectivity" becomes weaker and weaker. To the extreme, at convergence (which in practice it may require k to be thousands or millions), the "local" effect becomes "global", and that validates one of the reasons

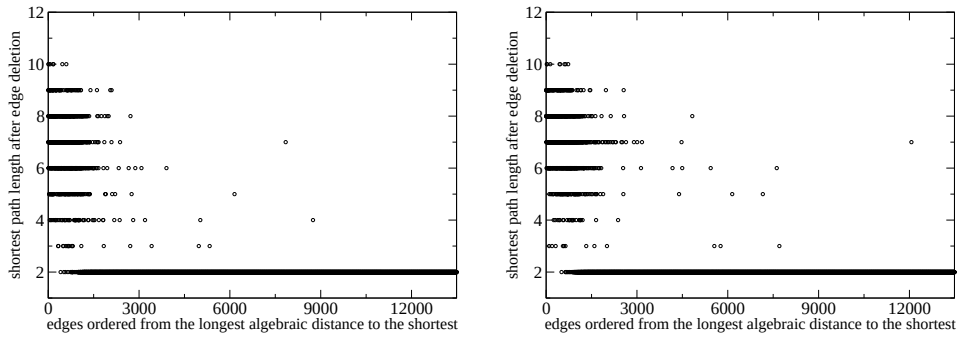
why the second eigenvector can be used for partitioning purposes—it encodes a global measure of the vertex connectivity.

In an experiment in the next section, we show that as long as $x^{(k)}$ stabilizes, it makes no significant difference if the JOR iteration is run further. In other words, the measure of vertex connectivity will yield similar qualities when applied to various graph problems, regardless of whether k is only a small number (yet sufficient to make $x^{(k)}$ stabilize) or k tends to infinity. As such, a practical stopping criterion of JOR is not a prescribed maximum number of iterations, but rather, a threshold of the angle between two successive iterates. This situation comes back to the question why we would run a few steps of JOR rather than computing the eigenvector v_2—the computational cost rules.

7. Numerical examples. In this section we show some numerical examples to assist the interpretation of algebraic distances as a connectivity measure, especially how they represent the local connectivity and neighborhoods. Sometimes, the use of algebraic distances is closely tied to specific applications. However, the interpretation of the measure should not be application dependent. The practical uses of algebraic distances and in different application scenarios are the subject of Section 8.

7.1. Noise edges of a graph. To understand how the algebraic distance measures a *local* connectivity, one may consider perturbing a well-structured graph with random long-ranged edges. The design that these perturbed/noise edges are long-ranged (for example, having a long shortest path) implies that the two end points, although directly connected in the perturbed graph, do not pose a strong connection. For this, we used two unweighted graphs generated from finite element instances (airfoil and 3elt [9]). For every edge ij we define the quantity l_{ij}, which is the length of a shortest path between i and j if ij is deleted. Since the graphs are triangular meshes, $l_{ij} = 2$ for each edge ij. We extended the graph by creating 10% of additional random unweighted edges with $3 < l_{ij} < 11$. Figure 7.1 plots both the original and the noise edges, sorted according to their algebraic distances. It is clear that the noise edges in general have a connection strength much weaker than those of the original edges, and the algebraic distance can serve as a separator between them. The strong correlation between the algebraic distance and the quantity l_{ij} indicates that the former takes into consideration short paths between a pair of vertices when measuring their connectivity. As the number k of iterations increases, even the long-ranged edges with short algebraic distances will eventually pose a long distance.

7.2. A path graph with two regions. Consider a path graph with n vertices. For simplicity let us assume that n is even and call the vertex $n/2$ the "mid point". Each edge to the left of the mid point has a weight 1, whereas each one to the right has a weight ϵ that is smaller than 1. This graph can come from, for example, a discretization of a one dimensional elliptic PDE where the spacing between the grid points in the left region are smaller than that in the right region, but in the same region the spacing is uniform. In a multigrid solver for solving a linear system arising from this grid, it is natural to require a coarsening strategy to recognize that all edge strengths are about equal in each region with the exception of the transition occurring around the mid point. In Table 7.1 we show the algebraic distance for the edge e immediately to the left of the mid point, and that for the edge f immediately to the right of the mid point (two middle columns). We also show the average of (the inverse of) the algebraic distances in each of the two regions (two right columns). One sees that across all iterations $k < \infty$, the connection strength for e is always



(a) airfoil (b) 3elt

FIG. 7.1. *Experiments with finite element meshes. Experimental setup:* $p = 2$, $k = 15$. *Each tick on the x-axis corresponds to one edge, and the edges are ordered from the longest algebraic distance to the shortest. The respective y-coordinate of every point (i.e., edge* ij*) is* l_{ij}.

TABLE 7.1

Experiments with the path graph. Experimental setup: $n = 2000$, $R = 10$, $\epsilon = 0.1$. *Results of experiments with smaller* ϵ *do not differ significantly.*

Iterations k	$1/\varrho_e$	$1/\varrho_f$	avg$(1/\varrho_{ij})$, left	avg$(1/\varrho_{ij})$, right
$2 \cdot 10^1$	220	31	91	89
$2 \cdot 10^2$	829	105	240	249
$2 \cdot 10^3$	3751	487	692	704
$2 \cdot 10^4$	12012	1527	2398	2249
$2 \cdot 10^5$	15135	1861	8225	8143
∞	66678	6668	393024	39302

much stronger than that for f, and the average connection strength for the edges in the left region is more or less the same to that for the right region. The case $k = \infty$ is equivalent to computing the algebraic distances from the exact eigenvector v_2 of (L, D). In such a case, there is also a clear distinction of the connection strengths between e and f; however, the averages of the two regions are no longer close. This comparison has two implications. First, running JOR iterations either in a few steps or until convergence will yield a sufficiently good measure to distinguish the edges e and f. Second, it is arguable whether making comparisons about the connectivity among regions is sensible. On the one hand, since edge weights in the right region are smaller than in the left, one would expect that the connectivities in the two regions are different. On the other hand, a coarsening strategy does not need to recognize the two regions; whether they have similar connectivity or not does not affect coarsening or aggregation. The real need is to be able to distinguish between e and f for mid point coarsening. In this regard, we consider that both the iterate $x^{(k)}$ and the eigenvector v_2 are sufficient to measure the connectivity.

7.3. Comparison with converged eigenvectors. The preceding example implies that the exact eigenvector v_2 has a similar impact to the JOR iterate in measuring vertex connectivity. Here we consider a comparison with the Fiedler vector u_2 in a specific application: graph minimum linear arrangement (MinLA) problem [19]. For

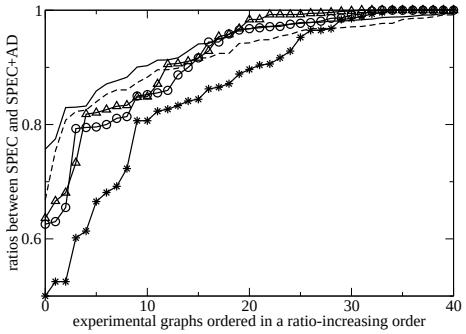
a multilevel solver of MinLA, the strength of connection in a graph coarsening step is computed based on u_2 [19]. We, on the other, computed the strength of connection based on a small number of JOR iterations ($k \leq 100$). Then, the numerical results of MinLA were improved in an average by 4.8%. Note that similar improvements caused by early interrupted iterations have also been seen in [8] for the minimum bandwidth problem and spectral sequencing. These results suggest that practical uses of the algebraic distances do no need to rely on the exact eigenvector. More advanced uses of the algebraic distance in multilevel schemes for linear arrangement problems are described in [25], and in Section 8 we discuss some other uses in more applications.

7.4. Impact of number of iterations. The example in Section 7.2 shows that there is no significant difference between a small k and a large one. Here we further study the impact of k under the context of the classical spectral partitioning method (SPEC) for graphs, where the 2-partitioning is approximated by using the Fiedler vector (see Section 5). Assuming SPEC a block-box solver, we redefined the edge weights as the inverse of the algebraic distances and ran the solver using this redefined graph. Results are as follows.

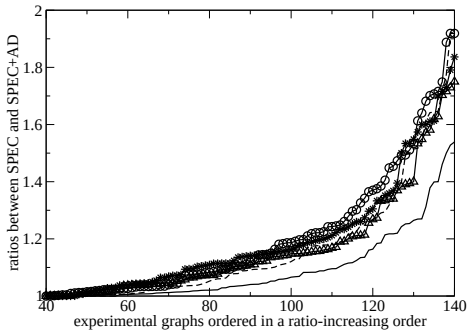
Table 7.2 shows the improvements under edge weight substitution. The row "average improvement" is the ratio of the cut cost of SPEC to that of SPEC plus algebraic distance preprocessing. It suggests that the improvement obtained by using 20 iterations dominates longer runs; however, in many cases the observed difference is insignificant. Thus, a large k is useless in most of the cases from the perspective of expensive computational costs. In the next couple of rows, we show the number of graphs with an average cut-cost ratio larger than 1.03 and smaller than 0.97, respectively. We use a cut-through 1.00 ± 0.03 instead of the strict cutting point 1.00 to differentiate between "real" improvements and worsening, by considering the impact of random effects in the initialization. The row "# graphs where $S_5 \succ S_i$" demonstrates the number of graphs whereby using $10,000$ iterations yields cuts better than using fewer iterations. The numbers show that running longer iterations does not necessarily yield better results. Indeed, this is true for any k as shown in the last row. Here we also take into account only those improvements by at least 3% to minimize the influence of the initial randomization. The improvement ratios for each graph are plotted in Figure 7.2.

Instead of trying to find a suitable k that led to the best result, we terminated the JOR iterations when the angle between two successive iterates $x^{(k)}$ fell below a threshold. In Figure 7.2(d), we show only those graphs that yielded an improvement ratio less than 1.9, as the behavior of other graphs is similar. Clearly, when the angle becomes smaller, the improvements become greater. Thus, in practice, it is advised to specify a small angle threshold to detect the stabilization of the iterate, rather than using a fixed k to terminate the JOR iteration.

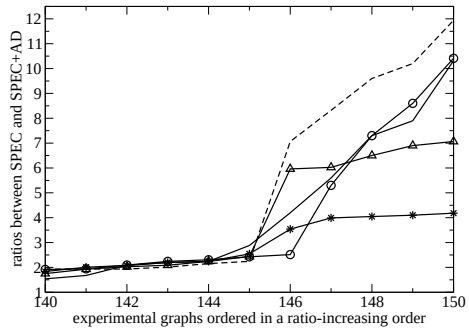
8. Applications. In addition to simple substitutes as edge weights, there are more sophisticated uses of the algebraic distances. In this section we show four applications: maximum weighted matching (MWM), maximum independent set (MIS), and partitioning of graphs (GP) and hypergraphs (HGP). The experimental graphs we used had different sizes ($10^3 < |E| < 10^7$) and structures and were selected from [9, 21] based on their non-triviality for several state-of-the-art solvers for optimization problems. The algebraic distances for MWM, MIS, and GP problems were calculated by using the standard JOR relaxation with $\omega = 1/2$. For the HGP problem, we extended the definition of algebraic distances for hypergraphs and demonstrate the results based on both GS and JOR relaxations.



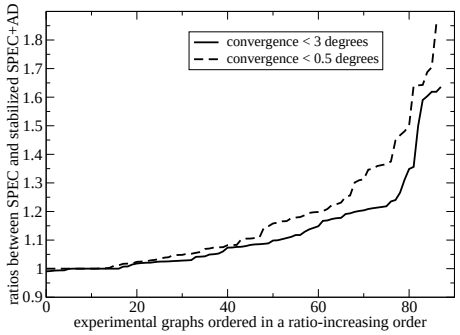
(a) Graphs that were not improved



(b) Graphs with improvement ratio less than 2



(c) Graphs with improvement ratio higher than 2



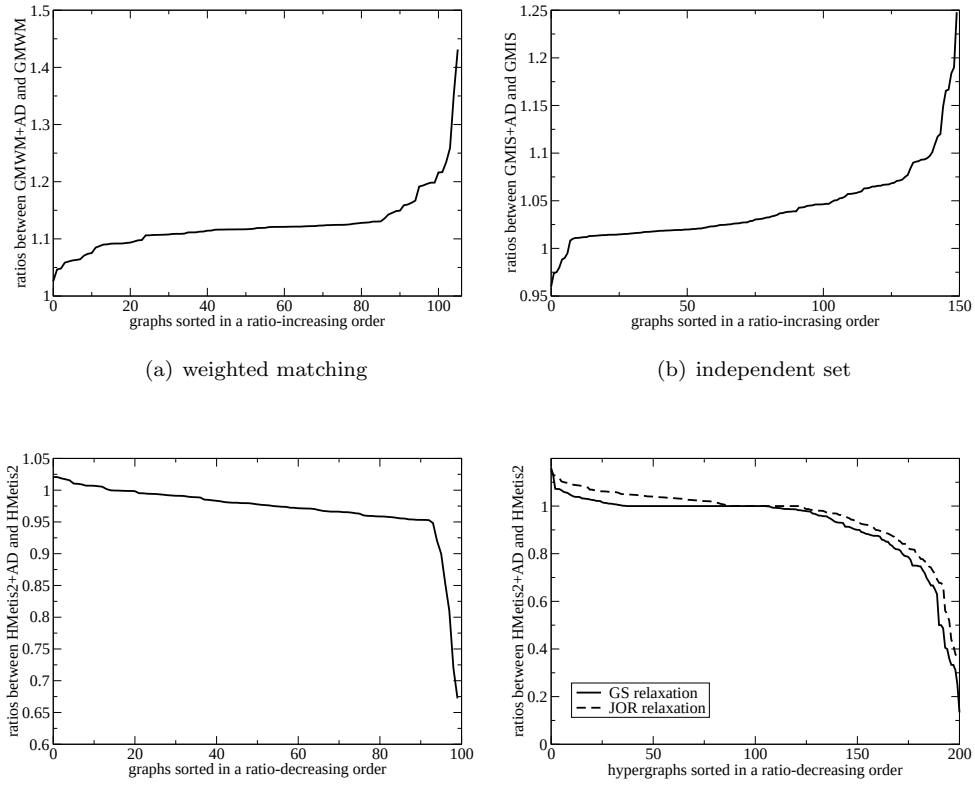
(d) Improvement ratios when JOR stabilized

FIG. 7.2. *Impact of number of iterations. The meaning of each curve for (a) to (c) is shown in Table 7.2. Each point corresponds to the ratio of the cut cost between the baseline algorithm and our algorithm with algebraic distance preprocessing.*

TABLE 7.2
Comparisons on different numbers of iterations. Total number of graphs: 150.

Cases	S_1 (–)	S_2 (- - -)	S_3 (◯)	S_4 (△)	S_5 (∗)
k	5	20	100	1000	10000
Average improvement	1.17	1.24	1.23	1.20	1.15
# improvements ≥ 1.03	63	80	87	89	91
# improvements ≤ 0.97	21	29	22	20	29
# graphs where $S_5 \succ S_i$	59	47	33	23	-
# graphs where $S_i \succ$ others	13	7	8	4	9

In all applications, existing fast and practically used baseline algorithms were modified by performing an algebraic distance processing. The comparison of numerical results is presented in Figure 8.1. Each point in the curves corresponds to a ratio of the numerical results between the modified algorithm and the baseline algorithm. Each ratio is an average of 50 executions of the same algorithm (fixed k) with different random seeds and random permutations of the input (hyper)graphs. This



(a) weighted matching

(b) independent set

(c) graph partitioning

(d) hypergraph partitioning

FIG. 8.1. *Comparisons of our algorithm with algebraic distance preprocessing and the corresponding baseline algorithm for (a) maximum weighted matching, (b) maximum independence set, (c) graph partitioning, and (d) hypergraph partitioning. Each point corresponds to the ratio of the resulting objective value between the two compared algorithms. The objectives are (a) weight of the matching, (b) size of the independence set, (c) and (d) cut costs.*

demonstrates the stability of experiments for different representations of the input.

8.1. Maximum weighted matching (MWM). A matching M of a graph is a subset of the graph edges such that no vertex is incident to more than one edge in M. A matching M is said to be *maximum* if $|M| \geq |M'|$ for any other matching M'. Similarly, a matching M is said to be *maximum weighted* if $w(M) \geq w(M')$ for any other matching M', where $w(M) = \sum_{ij \in M} w_{ij}$.

Although the MWM problem admits a polynomial time solution, often two well-known 2-approximation methods, which have a linear time complexity (without sorting), are used in practice. One is a textbook greedy algorithm, which successively adds a next legal heaviest edge to the existing matching (GMWM); the other is an improved version of the greedy algorithm, based on a path-growing principle. Both algorithms are presented in [11].

In both algorithms, there exists a greedy step in which the next heaviest edge (one that has the largest edge weight) has to be chosen. We modify this criterion according to the following heuristic observation: a better matching can be found in a dense graph

with less effort than in a sparse graph. Thus, we give preference to matching two nodes that are not well connected with other nodes from their neighborhood, to give a chance to the less connected nodes to participate in the matching. Specifically, we first define for each vertex a quantity $a_i = \sum_{ij \in E} 1/\varrho_{ij}^{(k)}$ that captures the connectivity between this vertex and its neighborhood. Then we define another quantity s'_{ij} as the weighted average between a_i and a_j. In each greedy step, we choose the edge that has the smallest s'_{ij}, rather than one with the largest weight. The modified algorithm is summarized in Algorithm 2 (GMWM+AD).

Algorithm 2 Greedy MWM with algebraic distance preprocessing

1: For all edges $ij \in E$ calculate $\varrho_{ij}^{(k)}$ for some k (cf. (2.2)). ▷ Preprocessing
2: For all nodes $i \in V$ compute $a_i = \sum_{ij \in E} 1/\varrho_{ij}^{(k)}$.
3: For all edges $ij \in E$ compute $s'_{ij} = a_i/\delta_i + a_j/\delta_j$, where δ_i is the degree of i.
4: $M \leftarrow \emptyset$. ▷ Start of greedy algorithm
5: **while** $E \neq \emptyset$ **do**
6: $e \leftarrow$ edge with smallest s'_{ij}.
7: Add e to M.
8: Remove e and all its incident edges from E.
9: **end while**
10: **return** M

The experimental results are presented in Figure 8.1(a), which shows for each graph the ratio of the sizes of the matchings between the modified algorithm and the textbook greedy algorithm. All ratios were higher than 1, which indicates that our algorithm yielded a better matching than did the baseline algorithm. Almost identical results were obtained by improving a greedy path growing algorithm from [11]. These results were obtained with $k = 20$, $R = 10$, and $p = 1$. However, results of almost the same quality have been obtained with many different combinations of $R \geq 5$, $10 \leq k \leq 100$, and $p = 2, \infty$.

8.2. Maximum independent set (MIS). An independent set I is a subset of V in which no two vertices are incident. An independent set I is said to be *maximum* if $|I| \geq |I'|$ for any other independent set I'. Finding the maximum independent set in a graph is an NP-complete problem [13], and fast and qualitative approximations are of great interest for many applications.

Although many approximation algorithms for MIS have been proposed, a popular one in practice is still a textbook greedy algorithm [7]. In this algorithm, the vertices are examined in increasing order of their degrees, and the greedy step consists of choosing the next available vertex that does not contradict the current independent set (GMIS). We exploit a similar heuristic to the one in maximum weighted matching. That is, for each vertex i, we define a quantity b_i that indicates the connection strength between the vertex itself and its neighborhood. To be precise, a small b_i means a weak connection, and we sort the vertices in the increasing order of b_i. Hence, we give preference to including a vertex that is weakly connected to its neighbors. The first four lines of Algorithm 3 show the computation of this quantity.

The experimental results are presented in Figure 8.1(b), which shows for each graph the ratio of the sizes of the independent sets between our algorithm (sorting the vertices according to connection strengths) and the textbook greedy algorithm (sorting the vertices according to vertex degrees). A ratio higher than 1 means that

Algorithm 3 Greedy MIS with algebraic distance preprocessing

For all edges $ij \in E$ calculate $s_{ij}^{(k)}$ for some k. ▷ Preprocessing
For all nodes $i \in V$ compute $a_i = \sum_{ij \in E} 1/s_{ij}^{(k)}$.
For all edges $ij \in E$ compute $s'_{ij} = (1/s_{ij}^{(k)})/(a_i + a_j)$.
For all nodes $i \in V$ compute $b_i = \sum_{ij \in E} s'_{ij}$.
Relabel the vertices i in the increasing order of b_i. ▷ Start of greedy algorithm
$I \leftarrow \emptyset$.
for $i = 1, 2, \ldots, n$ **do**
    If $\{i\} \cup I$ is an independent set, add i to I.
end for
return I

our algorithm computes a larger independent set. As can be seen from the figure, for almost all the graphs our algorithm yielded a better result.

8.3. Graph/Hypergraph partitioning (GP/HGP). Both GP and HPG are well known NP-hard problems [14]. Let $\mathcal{H} = (\mathcal{V}, \mathcal{E})$ be a hypergraph, where $\mathcal{V}$ is the set of vertices and $\mathcal{E}$ is the set of hyperedges. Each $h \in \mathcal{E}$ is a subset of $\mathcal{V}$. The goal of HGP is to find a partitioning of $\mathcal{V}$ into a family of τ disjoint nonempty subsets $(\pi_p)_{1 \le p \le \tau}$, restricted to the following:

$$\begin{aligned} &\text{minimize} \sum_{\substack{h \in \mathcal{E} \text{ s.t. } \exists i,j \in h \text{ and} \\ i \in \pi_p \Rightarrow j \notin \pi_p}} w_h \\ &\text{such that } \forall p,\ |\pi_p| \le (1+\alpha) \cdot |V|/\tau\ , \end{aligned} \tag{8.1}$$

where α is a given *imbalance factor*. The GP problem can be considered a special case of HGP with $|h| = 2$ for all $h \in \mathcal{E}$. For both GP and HGP we deal with $\tau = 2$ and as a baseline algorithm a multilevel solver HMetis2 [18] is employed. We define its extension "HMetis2 with algebraic distance preprocessing" in Algorithm 4.

In the case of GP we substitute the edge weights with the inverses of algebraic distance $s_{ij}^{(k)}$ and use HMetis2 to produce the 2-partitioning. In the case of HGP we consider a bipartite graph model for hypergraphs. We create a bipartite graph $G = (V, E)$ with the vertex set $V = \mathcal{V} \bigcup \mathcal{E}$ and the edge set E with $ih \in E$ if $i \in \mathcal{V}$ appears in hyperedge $h \in \mathcal{E}$. After running k iterations with R random initial vectors on the bipartite model of $\mathcal{H}$, we define $\varrho_S^{(k)}$, the *extended p-normed algebraic distance for a subset of nodes S in $\mathcal{H}$*, as

$$\left(\sum_{r=1}^{R} \max_{i,j \in S} \left| x_i^{(k,r)} - x_j^{(k,r)} \right|^p \right)^{1/p} . \tag{8.2}$$

We can use any reasonable relaxation process to compute the iterates $x^{(k,r)}$. Here we consider GS and JOR with $\omega = 1/2$.

In Figures 8.1(c) and (d) we show for each (hyper)graph the ratio of the cut costs (the objective value in (8.1)) between our extension and the original HMetis2. In the case of GP, our extension yielded a smaller cut for most of the graphs, and in the best case the cut cost was reduced by more than 30%. For HGP, similar results were obtained, and in the best case the cut cost was reduced by around 90%. We

Algorithm 4 HMetis2 with algebraic distance preprocessing, HMetis2+AD

1: (GP) For all edges $ij \in E$ calculate $s_{ij}^{(k)}$ for some k (typically 50).
   (HGP) For all hyperedges $h \in \mathcal{E}$ calculate $\varrho_h^{(k)}$ for some k.
2: For all (hyper)edges $ij \in E$ modify the weight $w_{ij} = 1/s_{ij}^{(k)}$ ($w_h = 1/\varrho_h^{(k)}$).
3: Produce the (hyper)graph cut using HMetis2 with modified (hyper)edge weights.
4: Return the cut weight computed from the original (hyper)edge weights.

also see that a GS relaxation process in general yielded slightly better results than did the standard JOR process. Note that in both cases a better way to apply the algebraic distances is to use them at all levels, as was demonstrated in [25], rather than substituting the edge weights only at the top level. However, even in such a simplified extension, the obtained improvement is systematic and significant. To the best of our knowledge, this is the first evidence that multilevel hypergraph partitioning schemes admit such a great improvement.

The numerical results for comparison have been achieved by using a greedy first-choice coarsening scheme and moderate refinement at uncoarsening for $\alpha = 0.01$. Other coarsening schemes of HMetis2 do not change the picture significantly. The strategy that slightly decreased the gap between baseline and modified algorithms was a "slow" refinement. In many cases, however, more aggressive refinement is expected to reduce this gap for any type of meaningful coarsening.

9. Conclusion. We have presented an iterative process for smoothing initially random values on graph vertices through direct neighbors, and we have defined a notion of algebraic distance as the difference between two smoothed values. The process is equivalent to a JOR relaxation run on the graph Laplacian matrix L. An analysis shows that the scaled algebraic distance between two vertices i and j converges to a value proportional to the difference between the ith and the jth entry of the second smallest eigenvector of the pencil (L, D). Also, it reveals that the convergence is often extremely slow; however, the iterate soon stabilizes: the change between consecutive iterates is very small after only a few iterations. This stabilization effect fits a mutually influenced model, where the range of the impact of a vertex is restricted to its local neighborhood. Thus, the algebraic distance represents a local, instead of global, connectivity between vertices.

We have shown several applications to demonstrate how algebraic distances can be used to define quantities that replace the graph edge weights in algorithms for combinatorial optimization problems. The experiments show that with an algebraic distance preprocessing, the quality of several baseline algorithms can be significantly improved. Furthermore, the computation of the algebraic distances occupies only a small fraction of the overall solution time. Thus, its easy parallelization makes it particularly attractive for dealing with large-scale instances.

10. Acknowledgments. We thank two anonymous referees whose suggestions helped improve this article significantly.

Appendix. Proof of Theorem 4.2. The proof requires two lemmas.

LEMMA A.1. *Define a series of column vectors* (*indexed by* k)

$$\zeta^{(k)} := [a_1\sigma_1^k,\ a_2\sigma_2^k,\ \ldots,\ a_n\sigma_n^k]^T \in \mathbb{R}^n,$$

where $\sigma_1 = 1$, $0 \le \sigma_i \le 1$ *for* $i = 2, \ldots, n$, *and* $a_1 \neq 0$. *If*

$$f_k := \frac{\alpha {r_k}^{2k}(1-r_k)^2}{1+\alpha {r_k}^{2k}(1+r_k)^2} \le 1,$$

where $\alpha = \left(\sum_{i\neq 1} a_i^2\right) / \left(4a_1^2\right)$ *and* r_k *is the unique root of* (4.2) *on* $[0,1]$, *then*

$$1 - \left\langle \frac{\zeta^{(k)}}{\left\|\zeta^{(k)}\right\|}, \frac{\zeta^{(k+1)}}{\left\|\zeta^{(k+1)}\right\|} \right\rangle^2 \le \frac{4f_k}{(1+f_k)^2}. \tag{A.1}$$

Proof. Let $b_i = (1+\sigma_i)/2$ and $c_i = (1-\sigma_i)/2$ for all i. Then,

$$\begin{aligned} 1 - \left\langle \frac{\zeta^{(k)}}{\left\|\zeta^{(k)}\right\|}, \frac{\zeta^{(k+1)}}{\left\|\zeta^{(k+1)}\right\|} \right\rangle^2 &= 1 - \frac{\left(\sum a_i^2 \sigma_i^{2k+1}\right)^2}{\left(\sum a_i^2 \sigma_i^{2k}\right)\left(\sum a_i^2 \sigma_i^{2k+2}\right)} \\ &= 1 - \frac{\left(\sum a_i^2 \sigma_i^{2k}(b_i^2 - c_i^2)\right)^2}{\left(\sum a_i^2 \sigma_i^{2k}(b_i + c_i)^2\right)\left(\sum a_i^2 \sigma_i^{2k}(b_i - c_i)^2\right)} \\ &\le \frac{4\left(\sum a_i^2 \sigma_i^{2k} b_i^2\right)\left(\sum a_i^2 \sigma_i^{2k} c_i^2\right)}{\left(\sum a_i^2 \sigma_i^{2k}(b_i^2 + c_i^2)\right)^2}. \end{aligned} \tag{$*$}$$

Denote $t = \left(\sum a_i^2 \sigma_i^{2k} c_i^2\right) / \left(\sum a_i^2 \sigma_i^{2k} b_i^2\right)$, then,

$$(*) = \frac{4}{(1+t)(1+1/t)}.$$

We would like to find an upper bound for t.

Note that $\sigma_1 = 1$; it is not hard to see that t achieves maximum only when $\sigma_2 = \sigma_3 = \cdots = \sigma_n$. Let them all be equal to r. Then t becomes

$$\frac{\left(\sum_{i\neq 1} a_i^2\right) r^{2k}(1-r)^2}{4a_1^2 + \left(\sum_{i\neq 1} a_i^2\right) r^{2k}(1+r)^2} =: f_k(r).$$

The stationary points of $f_k(r)$ satisfy the first-order condition (4.2). Note that the polynomial on the left-hand side of (4.2) is a monotonically increasing function of r for any $k \ge 1$. Therefore, it's not hard to see that (4.2) has a unique root in the interval $[0,1]$. Compared with the boundaries, we conclude that this root is the global maximum of $f_k(r)$. Thus, $t \le f_k \le 1$, and therefore

$$(*) \le \frac{4}{(1+f_k)\,(1+1/f_k)} = \frac{4f_k}{(1+f_k)^2}. \qquad \square$$

LEMMA A.2. *Let the squared sine of the angle between two unit vectors* $x, y \in \mathbb{R}^n$ *be* ϵ, *i.e.,* $1 - \langle x, y\rangle^2 = \epsilon$, *and let a diagonal matrix* $D \in \mathbb{R}^{n\times n}$ *have condition number* κ. *If* ϵ *and* κ *satisfy*

$$\kappa^2 \left(\frac{1-\sqrt{1-\epsilon}}{1+\sqrt{1-\epsilon}}\right) \le 1, \tag{A.2}$$

then

$$1-\left\langle \frac{Dx}{\|Dx\|},\frac{Dy}{\|Dy\|}\right\rangle^2 \le \frac{4}{\left[1+\kappa^2\left(\frac{1-\sqrt{1-\epsilon}}{1+\sqrt{1-\epsilon}}\right)\right]\left[1+\frac{1}{\kappa^2}\left(\frac{1+\sqrt{1-\epsilon}}{1-\sqrt{1-\epsilon}}\right)\right]}. \quad \text{(A.3)}$$

Proof. Let $z=(x+y)/2$ and $\delta=(x-y)/2$. Then,

$$\begin{aligned}1-\left\langle \frac{Dx}{\|Dx\|},\frac{Dy}{\|Dy\|}\right\rangle^2 &= 1-\frac{(y^TD^2x)^2}{(y^TD^2y)(x^TD^2x)}\\ &= 1-\frac{[(z-\delta)^TD^2(z+\delta)]^2}{[(z-\delta)^TD^2(z-\delta)][(z+\delta)^TD^2(z+\delta)]}\\ &\le \frac{4(z^TD^2z)(\delta^TD^2\delta)}{(z^TD^2z+\delta^TD^2\delta)^2}. \qquad (**)\end{aligned}$$

Denote $t=(\delta^TD^2\delta)/(z^TD^2z)$. If $x^Ty\ge 0$, then we have

$$t=\frac{\delta^TD^2\delta}{z^TD^2z}\le\frac{d_{\max}^2\|\delta\|^2}{d_{\min}^2\|z\|^2}=\kappa^2\left(\frac{1-x^Ty}{1+x^Ty}\right)=\kappa^2\left(\frac{1-\sqrt{1-\epsilon}}{1+\sqrt{1-\epsilon}}\right)\le 1,$$

otherwise

$$t=\frac{\delta^TD^2\delta}{z^TD^2z}\ge\frac{d_{\min}^2\|\delta\|^2}{d_{\max}^2\|z\|^2}=\frac{1}{\kappa^2}\left(\frac{1-x^Ty}{1+x^Ty}\right)=\frac{1}{\kappa^2}\left(\frac{1+\sqrt{1-\epsilon}}{1-\sqrt{1-\epsilon}}\right)\ge 1,$$

where $d_{\max}$ and $d_{\min}$ are the maximum and the minimum of the absolute values of the diagonal elements of D, respectively. For both cases, we have

$$(**)=\frac{4}{(1+t)(1+1/t)}\le\frac{4}{\left[1+\kappa^2\left(\frac{1-\sqrt{1-\epsilon}}{1+\sqrt{1-\epsilon}}\right)\right]\left[1+\frac{1}{\kappa^2}\left(\frac{1+\sqrt{1-\epsilon}}{1-\sqrt{1-\epsilon}}\right)\right]}. \qquad \square$$

Proof of Theorem 4.2. From condition (4.1a), we know that all the eigenvalues of H_{JOR}, $1-\omega\mu_i$ are nonnegative. Therefore, we have $\sigma_i=1-\omega\mu_i$ and $\phi_i=v_i$ for all i. Thus, $x^{(k)}=\sum_i a_i\sigma_i^k\phi_i=\sum_i a_i\sigma_i^kv_i$. In the matrix form, we denote

$$x^{(k)}=V\zeta^{(k)},$$

where $\zeta^{(k)}=[a_1\sigma_1^k,a_2\sigma_2^k,\dots,a_n\sigma_n^k]^T$. Then by Lemma A.1,

$$1-\left\langle \frac{\zeta^{(k)}}{\left\|\zeta^{(k)}\right\|},\frac{\zeta^{(k+1)}}{\left\|\zeta^{(k+1)}\right\|}\right\rangle^2 \le \frac{4f_k}{(1+f_k)^2}. \quad \text{(A.4)}$$

To simplify notations, let $x'=\zeta^{(k)}/\left\|\zeta^{(k)}\right\|$ and $y'=\zeta^{(k+1)}/\left\|\zeta^{(k+1)}\right\|$. Then,

$$1-\left\langle \frac{x^{(k)}}{\left\|x^{(k)}\right\|},\frac{x^{(k+1)}}{\left\|x^{(k+1)}\right\|}\right\rangle^2 = 1-\frac{\langle Vx',Vy'\rangle^2}{\|Vx'\|^2\|Vy'\|^2}=1-\frac{\left(y'^TV^TVx'\right)^2}{\left(y'^TV^TVy'\right)\left(x'^TV^TVx'\right)}.$$

Let V^TV have the eigen-decomposition $U^T\Sigma U$, where U is orthogonal and Σ is positive definite diagonal, then

$$1-\left\langle \frac{x^{(k)}}{\left\|x^{(k)}\right\|},\frac{x^{(k+1)}}{\left\|x^{(k+1)}\right\|}\right\rangle^2 = 1-\frac{\left(y'^TU^T\Sigma Ux'\right)^2}{\left(y'^TU^T\Sigma Uy'\right)\left(x'^TU^T\Sigma Ux'\right)}=1-\frac{\left\langle\Sigma^{1/2}Ux',\Sigma^{1/2}Uy'\right\rangle^2}{\left\|\Sigma^{1/2}Ux'\right\|^2\left\|\Sigma^{1/2}Uy'\right\|^2}.$$

Let $x = Ux'$ and $y = Uy'$. Then

$$1 - \left\langle \frac{x^{(k)}}{\left\|x^{(k)}\right\|}, \frac{x^{(k+1)}}{\left\|x^{(k+1)}\right\|} \right\rangle^2 = 1 - \left\langle \frac{\Sigma^{1/2}x}{\left\|\Sigma^{1/2}x\right\|}, \frac{\Sigma^{1/2}y}{\left\|\Sigma^{1/2}y\right\|} \right\rangle^2.$$

Let the right-hand side of the inequality (A.4) be ϵ_k. We have four facts. First, $\|x\| = \|y\| = 1$. Second,

$$1 - \langle x, y\rangle^2 = 1 - \langle Ux', Uy'\rangle^2 = 1 - \langle x', y'\rangle^2 \leq \epsilon_k.$$

Third, by noting that V is D-orthogonal, that is, $V^T DV = I$, we have

$$\kappa^2(\Sigma^{1/2}) = \kappa(\Sigma) = \kappa(V^T V) = \kappa(VV^T) = \kappa(D^{-1}) = \kappa.$$

Fourth, since ϵ_k and f_k satisfy the relation

$$\frac{1 - \sqrt{1-\epsilon_k}}{1 + \sqrt{1-\epsilon_k}} = f_k,$$

we have

$$\kappa^2(\Sigma^{1/2}) \left(\frac{1 - \sqrt{1-\epsilon_k}}{1 + \sqrt{1-\epsilon_k}} \right) = \kappa f_k \leq 1.$$

Therefore, by Lemma A.2, we conclude that

$$1 - \left\langle \frac{\Sigma^{1/2}x}{\left\|\Sigma^{1/2}x\right\|}, \frac{\Sigma^{1/2}y}{\left\|\Sigma^{1/2}y\right\|} \right\rangle^2 \leq \frac{4}{(1+\kappa f_k)(1+1/(\kappa f_k))}. \qquad \square$$

REFERENCES

[1] R. Andersen, F. Chung, and K. Lang, *Local graph partitioning using pagerank vectors*, in Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science, 2006.

[2] A. Brandt, *General highly accurate algebraic coarsening*, Electronic Trans. Num. Anal., 10 (2000), pp. 1–20.

[3] A. Brandt, *Multiscale scientific computation: Review 2001*, in Multiscale and Multiresolution Methods, vol. 20, Springer Verlag, 2002, pp. 3–95.

[4] A. Brandt and D. Ron, *Multigrid solvers and multilevel optimization strategies*, in Multilevel Optimization and VLSICAD, J. Cong and J. R. Shinnerl, eds., Kluwer, 2003.

[5] P. Chebotarev and E. Shamis, *On proximity measures for graph vertices*, Automation and Remote Control, 59 (1998), pp. 1443–1459.

[6] F. R. K. Chung, *Spectral Graph Theory*, American Mathematical Society, 1997.

[7] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, McGraw-Hill, 2nd ed., 2001.

[8] G. M. D. Corso and F. Romani, *Heuristic spectral techniques for the reduction of bandwidth and work-bound of sparse matrices*, Numerical Algorithms, 28 (2001), pp. 117–136.

[9] T. Davis, *University of Florida Sparse Matrix Collection*, NA Digest, 97 (1997).

[10] A. Dax, *The convergence of linear stationary iterative processes for solving singular unstructured systems of linear equations*, SIAM Rev., 32 (1990), pp. 611–635.

[11] D. E. Drake and S. Hougardy, *A simple approximation algorithm for the weighted matching problem*, Inf. Process. Lett., 85 (2003), pp. 211–213.

[12] F. Fouss, A. Pirotte, J.-M. Renders, and M. Saerens, *Random-walk computation of similarities between nodes of a graph with application to collaborative recommendation*, IEEE Transactions on Knowledge and Data Engineering, 19 (2007), pp. 355–369.

[13] M. R. Garey and D. S. Johnson, *Computers and Intractability. A Guide to the Theory of NP-Completeness*, Freemann and Company, 1979.
[14] M. R. Garey, D. S. Johnson, and L. Stockmeyer, *Some simplified NP-complete graph problems*, Theoretical Computer Science, 1 (1976), pp. 237–267.
[15] A. Ghosh, S. Boyd, and A. Saberi, *Minimizing effective resistance of a graph*, SIAM Rev., 50 (2008), pp. 37–66.
[16] G. H. Golub and C. F. Van Loan, *Matrix Computations*, Johns Hopkins University Press, 1996.
[17] R. A. Horn and C. R. Johnson, *Matrix Analysis*, Cambridge University Press, 1990.
[18] G. Karypis and V. Kumar, Metis *A Software Package for Partitioning Unstructured Graphs, Partitioning Meshes, and Computing Fill-Reducing Orderings of Sparse Matrices*, University of Minnesota, Department of Computer Science and Engineering, Army HPC Research Center, Minneapolis, MN, Sept. 1998.
[19] Y. Koren and D. Harel, *Multi-scale algorithm for the linear arrangement problem*, Proceedings of 28th Inter. Workshop on Graph-Theoretic Concepts, (2002).
[20] C. Lanczos, *An iteration method for the solution of the eigenvalue problem of linear differential and integral operators*, Research of the National Bureau of Standards, 45 (1950), pp. 255–282.
[21] J. Leskovec, *Stanford Network Analysis Package (SNAP)*. http://snap.stanford.edu/index.html.
[22] U. Luxburg, *A tutorial on spectral clustering*, Statistics and Computing, 17 (2007), pp. 395–416.
[23] B. Nadler, S. Lafon, R. R. Coifman, and I. G. Kevrekidis, *Diffusion maps, spectral clustering and eigenfunctions of Fokker-Planck operators*, in Advances in Neural Information Processing Systems, vol. 18, MIT Press, 2005, pp. 955–962.
[24] A. Y. Ng, M. Jordan, and Y. Weiss, *On spectral clustering: Analysis and an algorithm*, in Advances in Neural Information Processing Systems 14, 2002.
[25] D. Ron, I. Safro, and A. Brandt, *Relaxation-based coarsening and multiscale graph organization*, Multiscale Modeling & Simulation, 9 (2011), pp. 407–423.
[26] Y. Saad, *Numerical Methods for Large Eigenvalue Problems*, Halstead Press, 1992.
[27] J. Shi and J. Malik, *Normalized cuts and image segmentation*, IEEE Trans. Pattern Anal. Machine Intell., 22 (2000), pp. 888–905.
[28] D. A. Spielman and S.-H. Teng, *A local clustering algorithm for massive graphs and its application to nearly-linear time graph partitioning*, arXiv, cs.DS/0809.3232 (2008).
[29] U. Trottenberg and A. Schuller, *Multigrid*, Academic Press, Inc., Orlando, FL, USA, 2001.
[30] S. Wasserman and K. Faust, *Social Network Analysis: Methods and Applications*, Structural Analysis in the Social Sciences, Cambridge University Press, 1994.