SINGULAR VALUE DECOMPOSITION FOR INFORMATION RETRIEVAL, GRAPH BISECTION, AND GENETIC ALGORITHMS

by

JACOB GILMORE MARTIN

(Under the Direction of E. Rodney Canfield)

ABSTRACT

Singular value decomposition's usefulness in graph bisection, genetic algorithms, and information retrieval is studied. An information retrieval theorem about latent semantic indexing (LSI) is presented in detail. Several theorems are proved concerning bisection size guarantees when performing spectral bisection with the singular value decomposition of adjacency matrices of certain graphs. In addition, singular value decomposition is used in many ways to enhance a genetic algorithm's performance.

Highlights of the results include:

- Clarification of a well known LSI theorem, with counterexamples.
- Improvement of heuristics for finding the minimum bisection of a graph.

- Minimum bisection guarantees for graphs with a certain structures using a new proof strategy.
- Empirical evidence that multiple eigenvectors can be useful in spectral bisection.
- Several novel applications of singular value decomposition in genetic algorithms.

INDEX WORDS: Singular Value Decomposition, Graph Bisection, Minimum Graph Bisection, Graph Partitioning, Genetic Algorithm, Spectral Clustering, Spectral Bisection, Spectral, Genetic Engineering, Graph Clustering, Reduced Rank Approximation, Clustering, Latent Semantic Indexing, Latent Semantic Analysis, Gene Decomposition, Information Retrieval

SINGULAR VALUE DECOMPOSITION FOR INFORMATION RETRIEVAL, GRAPH BISECTION, AND GENETIC ALGORITHMS

by

JACOB GILMORE MARTIN
B.S., University of Georgia, 1999

A Dissertation Submitted to the Graduate Faculty of The University of Georgia in Partial Fulfillment of the Requirements for the Degree

DOCTOR OF PHILOSOPHY

ATHENS, GEORGIA
2005

SINGULAR VALUE DECOMPOSITION FOR INFORMATION RETRIEVAL, GRAPH BISECTION, AND GENETIC ALGORITHMS

by

JACOB GILMORE MARTIN

Major Professor: E. Rodney Canfield

Committee: Khaled Rasheed
Robert W. Robinson
Mitch Rothstein

Electronic Version Approved:

Maureen Grasso
Dean of the Graduate School
The University of Georgia
December 2005

ACKNOWLEDGMENTS

I dedicate this dissertation to all those who have given me patience. The most enduringly patient professor I have ever encountered is Rod Canfield. Rod has brightened my educational experience with an easy going, light hearted, and kind methodology. He has been the driving influence for the kind of teacher that I will always aspire to be. His explanations were genius in their simplicity, dispelling any obscurity with ease by knowledge and reason. Most of all, his willingness to sacrifice his time and energy on my behalf went way beyond the call of duty. I am forever indebted and will be eternally grateful for his patience and help, which was always given freely.

Bob Robinson has also shown me patience throughout my scholastic career. He always gave me a chance to be a better student. From him, I learned valuable lessons about teaching, theoretical computer science, and technical writing.

Khaled Rasheed was also an influence in my career here at UGA. It was he who encouraged me to keep writing and working on new projects. He was always willing to give his time at any moment to help me in my endeavors. In addition, the courses I took with him were invaluable for some of the work in this dissertation.

I would also like to thank Mitch Rothstein for lending his time to be on my committee. I have always admired his ability to make the complicated things simple.

I am grateful to Charles Atwood for providing the impetus to start this degree by giving me a job with the University. I'd like to thank him for believing in me and sticking up for me when it counted the most.

I'd also like to thank all of my many friends for being willing to listen to my ideas and for their support. Nathan Jaworski, John Simpson, Paul Ponder, and Tarsem Purewal were particularly patient while letting me try to relate material specific to this dissertation, even though, through no fault of their own, they sometimes rightfully claimed to not understand a word I said. They, and the rest, are true friends indeed.

Finally, my parents, Julia and George, and brothers, Paul and Reeve, have given me the most patience and support throughout my life, and for them I am very thankful and appreciative. They have always been forgiving of my mistakes and illuminating and supportive of my successes. I couldn't have done any of this without them.

TABLE OF CONTENTS

Chapter 1 - Introduction

The technique of singular value decomposition (SVD) has proven itself valuable in several different problem domains: data compression [28], image recognition and classification [34], chemical reaction analysis [68], document comparison [24, 12], cryptanalysis [66, 74], and genetic algorithms [62, 61]. Although these domains are quite different in some aspects, each can be reduced to the problem of ascertaining or ranking relevance in data. Intuitively, the concept of relevance depends critically on the nature of the problem at hand. SVD provides a method for mathematically discovering correlations within data. The focus of this work is to investigate several possible methods of using SVD to improve information retrieval performance in document comparison, to better solve the Minimum Graph Bisection problem, and to improve the performance of a genetic algorithm. In each case, experimental evidence is presented that supports the applicability of the underlying theorems.

A well known information retrieval theorem of Papadimitriou *et al.* is analyzed. In addition, counterexamples to this theorem are constructed, showing that the theorem may not always be true in

practice. Nevertheless, the theorem does give several basic intuitions that are useful in designing experiments and proofs in later chapters of the dissertation.

SVD is also shown to be useful when bisecting certain types of graphs. To obtain a bisection of a graph, SVD is performed directly on the 0,1 adjacency matrix of the graph to be bisected. Next, an eigenvector is chosen and its components are partitioned based on the median of all of the components. Given that each component of an eigenvector represents a vertex of the graph, a partitioning of the graph is achieved. The process of using eigenvectors to partition graphs is called *spectral bisection*. The technique's roots stem from the works of Fiedler [33], who studied the properties of the second smallest eigenvector of the Laplacian of a graph, and Donath and Hoffman [26], who proved a lower bound on the size of the minimum bisection of the graph. Several theorems are proved and experiments conducted with spectral bisection with a view to stimulating further research into the interactions between the eigenvectors of certain graphs.

In addition to applying SVD directly to graphs, it is also used in several ways to guide the search process of a Genetic Algorithm (GA). The first method of using SVD in a GA is to have it expose the most striking similarities between a given individual and a strategically chosen population of individuals. These similarities are used

to influence the direction of the GA's search process by qualifying candidate individuals for reinsertion into the next generation based on their proximity to other individuals whose fitnesses had already been computed. Initial results from the application of this process indicated significant improvements in the GA's performance. The second method of using SVD to help guide the search process of a GA is to use it to expose the most striking similarities between genes in the most highly fit individuals of the optimization history. The GA's optimization operators are then restricted to the locus of the genes corresponding to these striking similarities. In addition, individuals are *engineered* out of the discovered similarities between genes across highly fit individuals. The genes are also reordered on a chromosome in order to group similar genes closer together on a chromosome. The heuristics developed exhibited remarkable performance improvements. In addition, the performance achieved is magnified when the heuristics are combined with each other.

Chapter 2 contains some theoretical background necessary for understanding the proofs in later chapters. Chapter 3 contains a counterexample and a reformulation of an information retrieval theorem by Papadimitriou *et al.*. Chapter 4 contains background information on the graph partitioning problem. Chapter 5 attempts to discover, prove, and use useful facts about new spectral bisec-

tion techniques. Its results will propagate to Chapter 6 by using the new spectral bisection techniques to initially seed the population and guide the evolution of the genetic algorithm. Chapter 6 is the first known attempt at providing strategies for using singular value decomposition in a genetic algorithm for the Minimum Graph Bisection problem. The intent of Chapter 6 is to exploit the spectral properties of SVD to discern approximate information from sets of candidate solutions to the bisection problem. This material is quite different from the material in Chapter 5, which is intended to give a set of good initial solutions for the genetic algorithm in Chapter 6 to work with. Finally, Chapter 7 contains a review of contributions and several open questions for future research.

1.1 Prerequisites

One of the goals of this work is to provide almost all of the prerequisite knowledge for understanding the proofs and ideas contained herein. This is a difficult task, considering the material in this work covers many subjects. The most pervasive subject used is linear algebra, with particular focus on the SVD. The SVD is a subject M.I.T. Professor Gilbert Strang calls "absolutely a high point of linear algebra." In addition, graph theory, probability theory, computer science, and genetic algorithms are also worked with extensively. In each case, attempts are made to provide the reader with

all of the required knowledge directly through proof. Failing this, considerable references are made to proofs in other's works.

The reader should be familiar with the basic asymptotic notation used in complexity theory (big–O, big–Θ, and big–Ω). Basic linear algebra skills are also recommended, but not required because considerable linear algebra background is provided herein. In addition, some knowledge of combinatorics will be helpful.

Chapter 2 - Theoretical Background

> I consider that I understand an equation when I can predict the properties of its solutions, without actually solving it.
> –Paul Adrien Maurice Dirac. Quoted in F Wilczek, B Devine, Longing for the Harmonies

2.1 Probability Theory

Definition 1 *A **probability space** is a finite set of points*

$$\Omega = \{w_1, \cdots, w_m\}$$

with corresponding probabilities $p_1, \cdots, p_m$ *for each point in* Ω. *In effect,*

$$\begin{aligned} & p_i = Pr(w_i) \\ & \sum_{i=1}^{m} p_i = 1 \\ & 1 \geq p_i \geq 0 \end{aligned}$$

Definition 2 *A **random variable** X is a function from the probability space to the real numbers, $X : \Omega \to \Re$.*

Definition 3 *The **expected value** of a random variable X is the real number*

$$E(X) = \sum_{i}^{m} X(w_i) \cdot p_i$$

The following theorem, Markov's Inequality, gives a formula for obtaining the probability that a random variable is greater than or equal to some threshold.

Theorem 4 ***(Markov's Inequality)*** *If a random variable $X \geq 0$ and $a > 0 \in \Re$ then*

$$Pr(X \geq a) \leq \frac{E(X)}{a} \tag{2.1}$$

Proof:

$$\begin{aligned} Pr(X \geq a) &= \sum_{w \in \Omega, X(w) \geq a} Pr(w) \\ &\leq \sum_{w \in \Omega, X(w) \geq a} \frac{X(w)}{a} Pr(w) \\ &= \frac{1}{a} \sum_{w \in \Omega, X(w) \geq a} X(w) Pr(w) \\ &\leq \frac{1}{a} \sum_{w \in \Omega} X(w) Pr(w) \\ &= \frac{E(X)}{a} \end{aligned}$$

■

Let $X_1, X_2, \cdots, X_n$ be independent random variables with finite expectations and variances. Also let

$$\begin{aligned} S &= X_1 + \cdots + X_n \\ \overline{X} &= \frac{S}{n} \\ \mu &= \mathbf{E}[\overline{X} = \mathbf{E}[\frac{S}{n}]] \end{aligned} \tag{2.2}$$

Assuming that for all i, $0 \leq X_i \leq 1$, we have the following upper bound known as the Bienaymé–Chebychev inequality [20].

$$Pr[|\overline{X} - \mu| \geq t] \leq e^{-2nt^2}$$

In [49], Hoeffding extended the previous bound to the case where for each i, $a_i \leq X_i \leq b_i$,

$$Pr[|\overline{X} - \mu| \geq t] \leq e^{-2nt^2 / \sum_{i=1}^{n} (b_i - a_i)^2}$$

What follows is a proof of a similar inequality in the case that $X_i = \{-1, 1\}$.

Theorem 5 *Let $\{X_1, \cdots, X_n\}$ be n independent copies of a random variable X where*

$$X = \begin{cases} 1, & \text{with probability = } \frac{1}{2} \\ -1, & \text{with probability = } \frac{1}{2} \end{cases}$$

Then the probability space for these variables is $\Omega = \{-1, +1\}^n$*, and the probability of each point in* Ω *is* $\frac{1}{2^n}$*. Let* Y *be a function* $\Omega \longrightarrow \Re$ *defined by*

$$Y = \sum_{i=1}^{n} X_i$$

Then for $a > 0$*,*

$$Pr(Y \geq a) \leq e^{-a^2/2n} \tag{2.3}$$

Proof: Because $a > 0$,

$$Pr(Y \geq a) = Pr(e^{\lambda Y} \geq e^{\lambda a})$$

with $\lambda > 0$. By Equation 2.1 we have that

$$Pr(e^{\lambda Y} \geq e^{\lambda a}) \leq \frac{E(e^{\lambda Y})}{e^{\lambda a}}$$

From the definition of the function $E(X)$ and the fact that $e^{\lambda Y} = e^{\lambda(X_1 + \cdots + X_n)} = e^{\lambda X_1} e^{\lambda X_2} \cdots e^{\lambda X_n}$, we have that

$$\begin{aligned} E(e^{\lambda Y}) &= \sum_{\pm 1, \cdots, \pm 1} \frac{1}{2^n} (e^{\lambda X_1} e^{\lambda X_2} \cdots e^{\lambda X_n}) \\ &= \left[\frac{1}{2} (e^{\lambda} + e^{-\lambda}) \right]^n \\ &\leq \left[e^{\lambda^2/2} \right]^n \\ &= e^{\lambda^2 n/2} \end{aligned}$$

Therefore

$$Pr(Y \geq a) \leq \frac{E(e^{\lambda Y})}{e^{\lambda a}} \leq \frac{e^{\lambda^2 n/2}}{e^{\lambda a}} = e^{\lambda^2 n/2 - \lambda a}$$

We want to pick a λ such that $\lambda^2 n/2 - \lambda a$ is as *small* as possible in order to get a tight upper bound. Taking the derivative with respect to λ we see that

$$\frac{d}{d\lambda}(\lambda^2 n/2 - \lambda a) = \lambda n - a$$

The function achieves its minimum when we set it's derivative to 0.

$$\lambda n - a = 0$$

$$\lambda n = a$$

$$\lambda = a/n$$

Substituting the $\lambda = a/n$ into $\lambda^2 n/2 - \lambda a$ we achieve

$$\begin{aligned} \lambda^2 n/2 - \lambda a &= \left(\frac{a}{n}\right)^2 \cdot \frac{n}{2} - \left(\frac{a}{n}\right) \cdot a \\ &= \frac{a^2}{2n} - \frac{a^2}{n} \\ &= -\frac{a^2}{2n} \end{aligned}$$

Therefore

$$Pr(Y \geq a) \leq e^{-a^2/2n} \tag{2.4}$$

■

2.2 Linear Algebra

Definitions

Definition 6 *The **inner product**, or **dot product**, between two vectors $x, y \in \Re$ is denoted by $x \cdot y$ or (x, y) and is defined as follows.*

$$x \cdot y = (x, y) = x_1 y_1 + x_2 y_2 + \cdots + x_n y_n$$

Definition 7 *A **vector norm**, denoted by $\| \cdot \|$, satisfies the following three properties*

1. $\| x \| > 0$ *for any vector* $x \neq 0$
2. *For any scalar* c, $\| cx \| = |c| \, \| x \|$
3. *For any two vectors* x *and* y, $\| x + y \| \leq \| x \| + \| y \|$

Definition 8 *The **length** or **Euclidean norm** of a vector v, denoted by $\| v \|$, is a nonnegative scalar defined by*

$$\| v \| = \sqrt{v \cdot v} = \sqrt{v_1^2 + \cdots + v_n^2}$$

Therefore, the following property also follows by squaring both sides

$$\| v \|^2 = v \cdot v$$

Definition 9 *The* ***matrix norms*** *measure the maximum amount by which changes in a vector* x *are magnified in the calculation of* Ax*. Matrix norms satisfy the same three properties as a vector norm:*

1. $\| A \| > 0$ *for any nonzero matrix* A
2. *For any scalar* c, $\| cA \| = |c| \| A \|$
3. *For any two matrices* A *and* B, $\| A + B \| \leq \| A \| + \| B \|$

Definition 10 *The definitions of the* ***matrix two norm*** $\| A \|_2$ *and* ***Frobenius norm*** $\| A \|_F$ *are*

$$\| A \|_2 = max_{\|x\|=1} \| Ax \|$$
$$\| A \|_F = \sqrt{\sum_{i=1}^{m} \sum_{j=1}^{n} |a_{ij}|^2}$$

Definition 11 *A real matrix* A *is* ***orthogonal*** *if*

$$A^{-1} = A^T \ (AA^T = A^T A = I)$$

Definition 12 *A complex matrix* A *is* ***unitary*** *if*

$$A^{-1} = A^H \ (AA^H = A^H A = I)$$

Here, H denotes the hermitian transpose (conjugate and then transpose). Note that anything said about complex unitary matrices is also true for orthogonal matrices.

Definition 13 *A complex matrix A is **normal** if it commutes with its transpose, In effect, if*

$$AA^H = A^H A$$

Important Theorems

Theorem 14 *If A is a unitary (orthogonal) matrix, then*

1. $(Ax, Ay) = (x, y)$ *for all x and y, so the angle between Ax and Ay equals that between x and y*

   Proof:

   $(Ax, Ay) = (Ax)^H(Ay) = x^H A^H Ay = x^H y = (x, y)$ ■

2. $\| Ax \|$ = $\| x \|$ *for all x, so the length of Ax equals the length of x*

   Proof:

   $\| Ax \| = (Ax, Ax) = (Ax)^H(Ax) = x^H A^H Ax = x^H x = (x, y) = \| x \|$ ■

3. $\| A \|_2 = 1$

   Proof:

   $\| Ax \|_2 = \max_{x \neq 0} \dfrac{\| Ax \|}{\| x \|}$ ■

Theorem 15 *AB is unitary when both A and B are unitary.*

Proof:

$$(AB)^H(AB) = B^H A^H AB = B^H B = I$$
$$(AB)(AB)^H = ABB^H A^H = AA^H = I$$

■

Theorem 16 *For every square $p \times p$ matrix A there exists an upper triangular matrix T and unitary matrix P such that*

$$T = P^H AP \tag{2.5}$$

and

$$A = PTP^H \tag{2.6}$$

*with eigenvalues of A on the main diagonal of T. Equation 2.5 is called the **Schur form** of A. Equation 2.6 is called the **Schur decomposition** of A.*

Proof: The proof will proceed by induction.

- **Base Case**

  Let A be 1×1 so that $p = 1$. Now let $P = [1], P^H = [1]$. This arrangement fulfills the requirements of the theorem because the eigenvalue of any 1×1 matrix is the element itself and the

matrix $[1]$ is unitary.

$$A = [a_{11}] = [1][a_{11}][1] \tag{2.7}$$
$$T = [a_{11}] = [1][a_{11}][1]$$

- **Induction Step**

  Assume the theorem is true for $p = k$ and prove that it is true for $p = k + 1$. This will prove the theorem is true for all $k > 1$ because the base case has already been proved.

  Suppose A is $(k+1) \times (k+1)$. Find the first eigenvector x_1 and eigenvalue λ_1 of A so that

  $$Ax_1 = \lambda_1 x_1$$

  with $\parallel x_1 \parallel = 1$. Use Gram–Schmidt to extend $\{x_1\}$ to an orthonormal basis for $\mathbb{C}^{k+1}$. This gives a set of vectors $\{w_1, \ldots, w_k\}$ such that $W = \{x, w_1, \ldots, w_k\}$ is orthonormal. Let

  $$U = [x_1 W]$$

and compute

$$A^{'} = U^H A U = [x_1 W]^H A [x_1 W] = [x_1 W]^H [A x_1 A W]$$
$$= \begin{bmatrix} x_1^H A x_1 & x_1^H A W \\ W^H A x_1 & W^H A W \end{bmatrix} = \begin{bmatrix} \lambda_1 & b \\ 0 & C \end{bmatrix} = A^{'}$$

for some vector $b = x_1^H A W$ and $k \times k$ matrix C. The fact that $A^{'} = U^H A U$ here implies that $A^{'}$ is unitarily similar to A, which means that the eigenvalues and of A and $A^{'}$ are identical (including the multiplicities).

Now use the induction hypothesis on the matrix C to find a unitary V so that

$$V^H C V = T^{'}$$

with $T^{'}$ upper triangular and with the eigenvalues of C on its main diagonal. Let

$$P^{'} = \left[\begin{array}{c|ccc} 1 & 0 & \dots & 0 \\ \hline 0 & & & \\ \vdots & & \mathrm{V} & \\ 0 & & & \end{array}\right]$$

$P^{'}$ is unitary because V is also unitary by the induction hypothesis. Now,

$$
\begin{aligned}
P'^H U^H A U P' &= P'^H \begin{bmatrix} \lambda_1 & b \\ 0 & C \end{bmatrix} P' = \begin{bmatrix} \lambda_1 & b \\ 0 & V^H C \end{bmatrix} \left[\begin{array}{c|ccc} 1 & 0 & \cdots & 0 \\ \hline 0 & & & \\ \vdots & & \mathrm{V} & \\ 0 & & & \end{array}\right] \\
&= \left[\begin{array}{c|ccc} \lambda_1 & 0 & \cdots & 0 \\ \hline 0 & & & \\ \vdots & & V^H C V & \\ 0 & & & \end{array}\right] = \left[\begin{array}{c|ccc} \lambda_1 & 0 & \cdots & 0 \\ \hline 0 & & & \\ \vdots & & T' & \\ 0 & & & \end{array}\right] = T
\end{aligned}
\tag{2.8}
$$

Equation 2.8 shows that T is an upper triangular matrix because T' is by the induction hypothesis. By Theorem 15, a product of unitary matrices is also unitary. Therefore

$$P = UP'$$

means that P is unitary and

$$P^H A P = T$$

■

Theorem 17 *A $p \times p$ matrix A is normal if and only if A is unitarily similar to a diagonal matrix $D = P^H AP$.*

Proof:

- Diagonal Form Exists $\Longrightarrow$ normal

$$\begin{aligned} A^H A &= (PDP^H)^H (PDP^H) = PD^H P^H PDP^H \\ &= PD^H DP = PDD^H P^H \qquad (2.9) \\ &= PDP^H PD^H P^H \\ &= (PDP^H)(P^H DP)^H \\ &= AA^H \end{aligned}$$

  Equation 2.9 is true because diagonal matrices always commute ($DD^H = D^H D$).

- Normal $\Longrightarrow$ Diagonal Form Exists

  Suppose A is normal. This implies that A is square $p \times p$ by the definition of normal matrices (see Definition 13 on page 13). Theorem 16 says that any square matrix has a Schur form

$$T = U^H AU$$

with U unitary and T upper triangular. Notice that T is also normal

$$\begin{aligned} T^H T &= (U^H A^H U)(U^H A U) \\ &= U^H A^H A U = U^H A A^H U \qquad (2.10) \\ &= (U^H A U)(U^H A^H U) \\ &= T T^H \qquad (2.11) \end{aligned}$$

T is diagonal because T is upper triangular and $TT^H = T^H T$ which says that the dot products of the rows equal the dot product of the columns. Therefore,

$$T = \begin{bmatrix} t_{11} & t_{12} & \cdots & \cdots & t_{1p} \\ 0 & t_{22} & \cdots & \cdots & t_{2p} \\ 0 & 0 & t_{33} & \cdots & t_{3p} \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & t_{pp} \end{bmatrix}$$

$TT^H = T^H T$ implies that

$$\begin{aligned}
|t_{11}|^2 &= |t_{11}|^2 + |t_{12}|^2 + \cdots + |t_{1p}|^2 \\
|t_{12}|^2 + |t_{22}|^2 &= |t_{22}|^2 + |t_{23}|^2 + \cdots + |t_{2p}|^2 \\
|t_{13}|^2 + |t_{23}|^2 + |t_{33}|^2 &= |t_{33}|^2 + |t_{34}|^2 + \cdots + |t_{3p}|^2 \\
\vdots &= \vdots \\
|t_{1p}|^2 + |t_{2p}|^2 + \cdots + |t_{pp}|^2 &= |t_{pp}|^2
\end{aligned}$$

Simplifying gives

$$\begin{aligned}
0 &= |t_{12}|^2 + \cdots + |t_{1p}|^2 \qquad (2.12) \\
0 &= |t_{23}|^2 + \cdots + |t_{2p}|^2 \\
0 &= |t_{34}|^2 + \cdots + |t_{3p}|^2 \\
\vdots &= \vdots \\
0 &= |t_{1p}|^2 + |t_{2p}|^2 + \cdots + |t_{pp-1}|^2
\end{aligned}$$

Equation 2.12 implies that $t_{12} = \cdots = t_{1p} = 0$ because the only time a sum of a group of nonnegative numbers can equal zero is when they are all equal to zero. Similarly, the rest of the times when $i \neq j$, $t_{ij} = 0$. ■

Definition 18 *The **singular values** of an $m \times n$ matrix A are the square roots of the eigenvalues of $A^T A$ (if $m \geq n$) or of AA^T (if $m < n$), denoted by $\sigma_1, \ldots, \sigma_n$*

Theorem 19 *The singular values of A are the square roots of the eigenvalues of AA^T and $A^T A$.*

Proof: Given the existence of SVD by Theorem 23,

$$A^H A = (U\Sigma V^H)^H (U\Sigma V^H) = V\Sigma^H U^H U\Sigma V^H = V(\Sigma^H \Sigma)V^H \qquad (2.13)$$

$$AA^H = (U\Sigma V^H)(U\Sigma V^H)^H = U\Sigma V^H V\Sigma^H U^H = U(\Sigma\Sigma^H)U^H \qquad (2.14)$$

Theorem 16 can be used to see that the eigenvectors of $A^H A$ make up V, with the associated (real nonnegative) eigenvalues on the diagonal of $\Sigma^H \Sigma$. Likewise, Theorem 16 can also be used to see that the eigenvectors of AA^H make up U, with the associated (real nonnegative) eigenvalues on the diagonal of $\Sigma\Sigma^H$.

■

Theorem 20 *Let A be an $m \times n$ matrix. Then $A^T A$ is symmetric and can be orthogonally diagonalized. Let $\{v_1, \ldots, v_n\}$ be an orthonormal basis for $\Re^n$ of eigenvectors of $A^T A$, and let $\lambda_1, \ldots, \lambda_n$ be the associated eigenvalues of $A^T A$. The singular values of A are the lengths of the vectors $Av_1, \ldots, Av_n$*

Proof: For $1 \leq i \leq n$,

$$\begin{aligned} \| Av_i \|^2 &= (Av_i)^T Av_i = v_i^T A^T Av_i \\ &= v_i^T (\lambda_i v_i) \\ &= \lambda_i = \sigma_i^2 \end{aligned}$$

■

Note that this also implies that the eigenvalues of $A^T A$ are all nonnegative. The fact that every eigenvalue of a symmetric matrix is real will be proved next.

Theorem 21 *If A is a real symmetric or complex hermitian matrix ($A = A^T$ or $A = A^H$), then every eigenvalue of A is real.*

Proof: Assume first that A is has only real entries. Let x be an eigenvector of A corresponding to the eigenvalue λ so that

$$Ax = \lambda x \tag{2.15}$$

If we take the complex conjugate of both sides, then the resulting equation is still true.

$$Ax = \lambda x \Longrightarrow A\overline{x} = \overline{\lambda}\overline{x}$$

Now take the transpose of both sides to get

$$\overline{x}^T A^T = \overline{x}^T \overline{\lambda} \tag{2.16}$$

Because $A = A^T$ we have

$$\overline{x}^T A = \overline{x}^T \overline{\lambda} \tag{2.17}$$

Now multiply Equation 2.15 by the vector $\overline{x}^T$ on the left and Equation 2.17 by the vector x on the right to get

$$\overline{x}^T Ax = \overline{x}^T \lambda x$$
$$\overline{x}^T Ax = \overline{x}^T \overline{\lambda} x$$

And so,

$$\overline{x}^T \lambda x = \overline{x}^T \overline{\lambda} x \tag{2.18}$$

The product $\overline{x}^T x$ is positive and will not change the sign of the above equation. Therefore the λ's are all real because the above equation can only hold when there is no imaginary part to flip when λ is conjugated. The complex case follows by letting $A = \overline{A}$. ■

Theorem 22 *If an $m \times n$ matrix A has r nonzero singular values, $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r > 0$ with $\sigma_{r+1} = \sigma_{r+1} = \cdots = 0$, then rank$(A) = r$.*

Proof: Let $\{v_1, \ldots, v_n\}$ be an orthonormal basis of $\Re^n$ of eigenvectors of A^TA, ordered so that the corresponding eigenvalues of A^TA satisfy $\lambda_1 \geq \cdots \geq \lambda_n$. Define the **singular values** of A to be as in Definition 18 on page 20. Then for $i \neq j$

$$(Av_i)^T(Av_j) = v_i^T A^T A v_j = v_i^T \lambda v_j = 0 \tag{2.19}$$

because v_i is orthogonal to v_j for all $i \neq j$ by construction. Thus $\{Av_1, \ldots, Av_n\}$ is an orthogonal set. Let r be the number of nonzero singular values of A. By the definition of singular values, r is also the number of nonzero eigenvalues of A^TA. Because $\| Av_i \|^2 = \sigma_i^2$, we have that $Av_i \neq 0$ if and only if $1 \leq i \leq r$. Therefore, $\{Av_1, \cdots, Av_r\}$ is linear independent and is in the Col A. For any y in Col A, say $y = Ax$, we can write

$$\begin{aligned} x &= c_1v_1 + \cdots + c_nv_n \\ y = Ax &= c_1Av_1 + \cdots + c_rAv_r + c_{r+1}Av_{r+1} + \cdots + c_nAv_n \\ y = Ax &= c_1Av_1 + \cdots + c_rAv_r + 0 + \cdots + 0 \end{aligned}$$

Thus $y \in \text{Span}\{Av_1, \ldots, Av_r\}$, which shows that $\{Av_1, \ldots, Av_r\}$ is an orthogonal basis for Col A. Hence $\text{rank}(A) = r$. ■

Existence of SVD

The existence and theory of singular value decomposition was established by several mathematicians [80]: Beltrami [10], Jordan [55], Sylvester [82], Schmidt [75], and Weyl [86]. Horn and Johnson provide a succinct proof of its existence [52]. Stewart provided an excellent survey of the history of discoveries that lead to the theory of the singular value decomposition [80].

Theorem 23 *Let A be an $m \times n$ matrix with rank r. Then there exists an $m \times n$ diagonal matrix*

$$\Sigma = \begin{bmatrix} D & 0 \\ 0 & 0 \end{bmatrix} \tag{2.20}$$

where the diagonal entries of D are the first r singular values of A, $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r > 0$, and there exist an $m \times m$ orthogonal matrix U and an $n \times n$ orthogonal matrix V such that

$$A = U\Sigma V^T \tag{2.21}$$

Proof: Let $\{v_1, \ldots, v_n\}$ be an orthonormal basis of $\Re^n$ of eigenvectors of $A^T A$, ordered so that the corresponding eigenvalues of $A^T A$

satisfy $\lambda_1 \geq \cdots \geq \lambda_n$. Then for $1 \leq i \leq r$

$$\sigma_i = \sqrt{\lambda_i} = \| Av_i \| > 0$$

and $\{Av_1, \ldots, Av_r\}$ is an orthogonal basis for the column space of A. For $1 \leq i \leq r$, define

$$u_i = \frac{1}{\| Av_i \|} Av_i = \frac{1}{\sigma_i} Av_i$$

so that

$$Av_i = \sigma_i u_i \qquad 1 \leq i \leq r \tag{2.22}$$

Then $\{u_1, \ldots, u_r\}$ is an orthonormal basis of the column space of A. Extend this set to an orthonormal basis $\{u_1, \ldots, u_m\}$ of $\Re^m$, and let U and V be the orthogonal matrices

$$U = [u_1 \cdots u_m] \qquad V = [v_1 \cdots v_n]$$

Also,

$$AV = [Av_1 \cdots Av_r \;\; 0 \cdots 0] = [\sigma_1 u_1 \cdots \sigma_r u_r \;\; 0 \cdots 0]$$

Let D be the diagonal matrix with diagonal entries $\sigma_1, \ \ldots \ , \sigma_r$ And let

$$\Sigma = \begin{bmatrix} D & 0 \\ 0 & 0 \end{bmatrix} \tag{2.23}$$

Then

$$U\Sigma = [u_1 \ \cdots \ u_m] \begin{bmatrix} D & 0 \\ 0 & 0 \end{bmatrix} = [\sigma_1 u_1 \ \cdots \ \sigma_r u_r \ \ 0 \cdots 0] = AV$$

Now, because V is an orthogonal matrix

$$U\Sigma V^T = AVV^T = AI = A$$

■

Summary As Theorem 23 states, singular value decomposition expresses an $m \times n$ matrix A as the product of three matrices, U, Σ, and V^T. The matrix U is an m x m matrix whose first r columns, u_i $(1 \leq i \leq r)$, are the orthonormal eigenvectors that span the space corresponding to the row auto–correlation matrix AA^T. The last $m-r$ columns of U form an orthonormal basis for the left nullspace of A. Likewise, V is an n x n matrix whose first r columns, v_i $(1 \leq i \leq r)$, are the orthonormal eigenvectors that span the space

corresponding to the column auto–correlation matrix $A^T A$. The last $n - r$ columns of V form an orthonormal basis for the nullspace of A. The middle matrix, Σ, is an m x n diagonal matrix with $\Sigma_{ij} = 0$ for $i \neq j$ and $\Sigma_{ii} = \sigma_i \geq 0$ for $\forall i$. The σ_i's are called the singular values and are arranged in descending order with $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n \geq 0$. The singular values are defined as the square roots of the eigenvalues of AA^T and $A^T A$. The singular value decomposition can equivalently be expressed as a sum of rank one matrices

$$A = \sigma_1 u_1 v_1^T + \cdots + \sigma_r u_r v_r^T = \sum_{i=1}^{r=rank(A)} \sigma_i u_i v_i^T \tag{2.24}$$

The u_i's and v_i's are the columns of U and V respectively. Using the Golub–Reinsch algorithm [43, 40], U, Σ, and V can be calculated for an m by n matrix in time $O(m^2 n + mn^2 + n^3)$.

It should be noted that the singular values $\{\sigma_j\}$ are uniquely determined, and, if A is square and the σ_j are distinct, then the left and right singular vectors are determined uniquely up to complex scalar factors of absolute value 1.

Reduced Rank Approximations

The magnitudes of the singular values indicate the weight, or importance, of a dimension. To obtain an approximation of A, all but the $k < r$ largest singular values in the decomposition are set to

zero. This results in the formation of a new lower rank matrix A_k, of rank k, corresponding to the k most influential dimensions.

$$A_k = U_k \Sigma_k V_k^T \tag{2.25}$$

Here, U_k and V_k are the matrices formed by keeping only the eigenvectors in U and V corresponding to the k largest singular values. Equivalently,

$$A_k = \sigma_1 u_1 v_1^T + \cdots + \sigma_k u_k v_k^T = \sum_{i=1}^{k} \sigma_i u_i v_i^T \tag{2.26}$$

Intuitively, the reduced rank matrix A_k amplifies the most important similarities and suppresses the insignificant correlations between the vectors represented in the matrix A. Exactly how much of the original space is preserved is directly related to the amount of reduction performed. A theorem by Eckart and Young states, informally, that the new low–dimensional matrix obtained is the closest matrix, among all matrices of its rank or less, to the original matrix [29, 40]. Formally, it states that among all m x n matrices C with rank at most k, A_k is the one that minimizes

$$\|A - C\|_F^2 = \sum_{i,j} (A_{ij} - C_{ij})^2 \tag{2.27}$$

Eckart and Young's paper was actually a rediscovery of this property, which was first proved by Schmidt [75]. Although the theorem may explain how the reduction *does not deteriorate too much* in performance over conventional vector–space methods, it fails to justify the observed improvement in precision and recall in information retrieval applications [70]. However, several papers have made positive steps towards a rigorous proof that, given an appropriate structure for the matrix A, the benefit is achieved with high probability [70, 25].

Singular vectors and Eigenvectors of Symmetric Matrices

Let A be a symmetric matrix, then the following theorem describes the structure of the singular value decomposition of A.

Theorem 24 *Let A be a symmetric matrix. Then the eigenvectors and eigenvalues of AA^T, A^2, and A^TA are all equal. Moreover, all of these eigenvectors are equal to the eigenvectors of A and the singular values are simply the absolute values of the eigenvalues of A.*

Proof: Since A is symmetric, $A = A^T$. Therefore,

$$AA^T = A^TA = A^2 \tag{2.28}$$

and so their eigenvectors and eigenvalues are all equal. Since by definition, the eigenvectors of A^TA and AA^T are the left and right

singular vectors of A, the singular vectors of A are also equal. Furthermore, the eigenvectors do not move after squaring a matrix. A straight forward calculation shows that only the eigenvalues are squared when multiplying A by A. Concretely, if λ is an eigenvalue of A and x is an eigenvector of A corresponding to this eigenvalue (so that $Ax = \lambda x$), then

$$AAx = A\lambda x = \lambda Ax = \lambda\lambda x = \lambda^2 x \tag{2.29}$$

Therefore, the eigenvectors of $AA^T = A^TA = A^2$ are exactly equal to the eigenvectors of A, but their corresponding eigenvalues are squared. Moreover, Theorem 19 implies that the singular values of A are equal to the absolute values of the eigenvalues of A. ■

Theorem 24 allows a singular value decomposition to be computed for A and still retain the eigenvectors. The SVD already has its singular vectors arranged according to the magnitude of the corresponding singular values. In the symmetric case, the singular vectors are the eigenvectors with their corresponding eigenvalues made positive. One important thing to note about about the difference between eigenvalue decompositions and singular value decompositions in the symmetric case is that when the singular values are arranged in descending order, they will be arranged according to the *absolute value* of the eigenvalue. Therefore, the ordering

of the singular vectors may be slightly different than the ordering of the eigenvectors because two eigenvalues with an opposite sign will square to the same eigenvalue of AA^T or A^TA.

Theorem 25 *If P is an orthogonal $m \times m$ matrix, then PA has the same singular values as A.*

Proof:

$$PA = P(U\Sigma V^T)$$

Now, the product PU is orthonormal because it is square and

$$(PU)^T(PU) = U^TP^TPU = U^TU = I$$

Thus, $PA = (PU)\Sigma V^T$ has the form required for a singular value decomposition. Furthermore, the square roots of the eigenvalues of A^TA and $(PA)^TPA$ are the same because

$$(PA)^TPA = A^TP^TPA = A^TA \tag{2.30}$$

■

Theorem 26 *If P is an orthogonal $n \times n$ matrix, then AP has the same singular values as A.*

Proof:

$$AP = (U\Sigma V^T)P$$

Now, the product $V^T P$ is orthonormal because it is square and

$$(PV^T)^T(PV^T) = VP^T PV^T = VV^T = I$$

Thus, $AP = U\Sigma(V^T P)$ has the form required for a singular value decomposition. Furthermore, the square roots of the eigenvalues of AA^T and $AP(AP)^T$ are the same because

$$AP(AP)^T = APP^T A^T = AA^T \tag{2.31}$$

■

Theorem 27 *The left and right singular vectors, u_1 and v_1, corresponding to the largest eigenvalues of the matrix $A^T A$ and AA^T respectively have (all) non–negative components when A is a non–negative, irreducible matrix.*

Proof: The Perron–Frobenius theorem implies that if A is a non–negative, irreducible matrix, then the eigenvector corresponding to the maximal eigenvalue has positive components [85], [37]. Per-

form the singular value decomposition on A to get

$$U^T A V = \Sigma \tag{2.32}$$

where Σ is the zero matrix with $r = rank(A)$ singular values along the diagonal, and U and V are as described previously. The u_i and v_i are the eigenvectors of AA^T and $A^T A$, respectively. So,

$$AA^T u_1 = \lambda_1 u_1 \text{ and } A^T A v_1 = \kappa_1 v_1 \tag{2.33}$$

with λ_1 and κ_1 being the eigenvalues of AA^T and $A^T A$ respectively. Because the matrix A is non–negative and irreducible, AA^T and $A^T A$ are also non–negative and irreducible. Since σ_1 is the largest singular value ($\sigma_1 \geq \cdots \geq \sigma_n \geq 0$) and $\sigma_i = \sqrt{\lambda_i} = \sqrt{\kappa_i}$, λ_1 and κ_1 are maximal eigenvalues. Therefore, by the Perron–Frobenius theorem, u_1 and v_1 have positive components.

Since $\sigma_1 \geq 0$, both u_1 and v_1 have non–negative components. ■

Theorem 28 *If A is an $m \times n$ real or complex matrix then*

$$\| A \|_2 = \sigma_1$$

Proof: By definition,

$$\| A \|_2 = max_{x \neq 0} \frac{\| Ax \|}{\| x \|}$$

Given any vector $x \neq 0$, let $f(A, x)$ denote the following ratio

$$f(A, x) = \frac{\| Ax \|^2}{\| x \|^2} = \frac{x^T A^T A x}{x^T x}$$

Because $f(A, cx) = f(A, x)$ for any nonzero scalar c, only the *direction* of x affects the value of the function f. Any vector x that maximizes $f(A, x)$ must also give a maximum value for $\frac{\|Ax\|}{\|x\|}$. Let $\{v_1, \cdots, v_n\}$ denote the n columns of V, the orthonormal basis given by the SVD for $\Re^n$. Any vector $x \in \Re^n$ can be written as a linear combination of these columns because they form a basis for $\Re^n$. In effect

$$x = \alpha_1 v_1 + \alpha_2 v_2 + \cdots + \alpha_n v_n = V\alpha \tag{2.34}$$

where $\alpha = [\alpha_1, \cdots, \alpha_n]^T$. Using the properties of the SVD we will develop a compact expression for $f(A, x)$ in terms of the vector α.

$$Ax = U\Sigma V^T(V\alpha) = U\Sigma\alpha = \sum_{i=1}^{m}(\sigma_i \alpha_i)u_i$$

Substituting the above equation for Ax into the following equation yields a formula for $x^T A^T A x$ in terms of the vector α and the singular values of A.

$$x^T A^T A x = \alpha^T \Sigma^T U^T U \Sigma \alpha = \alpha^T \Sigma^2 \alpha = \sum_{i=1}^{n} \alpha_i^2 \sigma_i^2 \tag{2.35}$$

$$x^T x = \alpha^T V^T V \alpha = \alpha^T \alpha = \sum_{i=1}^{n} \alpha_i^2 \tag{2.36}$$

Using the previous two equations, we can give a more expressive formula for $f(A, x)$,

$$f(A, x) = \frac{x^T A^T A x}{x^T x} = \frac{\sum_{i=1}^{n} \alpha_i^2 \sigma_i^2}{\sum_{i=1}^{n} \alpha_i^2}$$

To finish the proof, notice that the above ratio is maximized when $\alpha_1 \neq 0$ and $\alpha_2 = \cdots = \alpha_n = 0$. This is because we have defined the first singular value to be the largest singular value. Therefore, when the ratio is maximized, x is a non–zero multiple of the first right singular vector, v_1. For any non–zero vector x and scalar $c \neq 0$ with $x = cv_1$, we have from Equations 2.35 and 2.36 that

$$x^T A^T A x = c^2 \sigma_1^2$$

$$x^T x = c^2$$

$$f(A, x) = \frac{x^T A^T A x}{x^T x} = \frac{c^2 \sigma_1^2}{c^2} = \sigma_1^2$$

Therefore,

$$max_{x \neq 0} \sqrt{f(A, x)} = max_{x \neq 0} \frac{\| Ax \|}{\| x \|} = \sigma_1 = \| A \|_2 \tag{2.37}$$

Which completes the proof.

■

2.3 Graph Theory

Eigenvalues of Graphs

The spectrum of a graph is the set of eigenvalues of its adjacency matrix. In certain adjacency matrix representations, as a graph's subgraphs become more and more connected, and certain other restrictions are made, the reduced rank SVD has a very good chance of corresponding exactly one singular vector to each subblock of the adjacency matrix. This is due to the assumption that the conductance of each of the subgraphs is high. When the conductance of a graph increases, the separation of the largest two eigenvalues also increases. For examples and background on spectral analysis of data see the survey paper by Noga Alon [3]. General information about the eigenvalues of the adjacency matrix of a graph can be found in a book by Cvetković, Doob, and Sachs [23].

Conductance and Expansion

Definition 29 *A **cut** C, is a partition of a graph, or subgraph, into two disjoint sets of vertices. The partition is defined by picking some $S \subset V$. The other side of the partition is simply $V - S$.*

Definition 30 *The **capacity** of a cut is the number of edges between vertices in the two different sets created by the cut.*

$$\text{capacity}(C) = \sum_{i \in S, j \notin S} \text{weight}(i, j) \tag{2.38}$$

Definition 31 *The **conductance** $\phi(G)$ of a graph G is the minimum ratio, over all cuts of the graph, of the capacity of the cut to the number of vertices in the smaller part created by the cut.*

$$\phi(G) = \min_{(S \subset V)} \frac{\sum_{i \in S, j \notin S} \text{weight}(i, j)}{\min\left\{|S|, |\overline{S}|\right\}} \tag{2.39}$$

Noga Alon and V. D. Milman showed in [2] that if the second largest eigenvalue of a graph is far from the largest eigenvalue, then the graph is a good expander. Alon later showed in another paper that the converse is also true [1]. In effect, if a graph is a good expander, then the largest two eigenvalues will be far apart. The bounds proved between the conductance and the eigenvalue gap are

$$\frac{\phi^2}{2} \leq 1 - \frac{\lambda_2}{\lambda_1} \leq 2\phi \tag{2.40}$$

Figure 2.1 shows that as the conductance increases over the interval $[0..1]$, the separation of the first and second eigenvalues also increases.

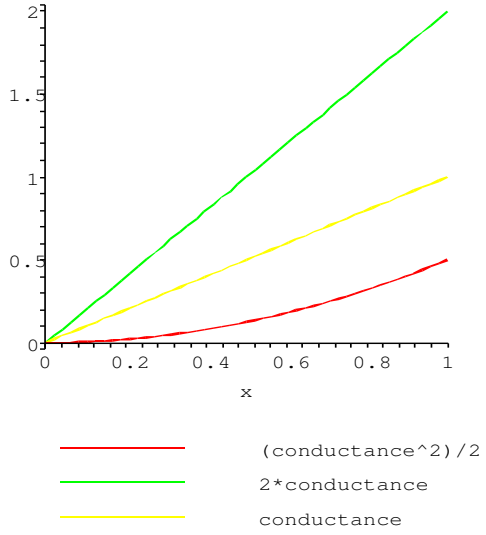

Figure 2.1: The separation of the first two eigenvalues increases with the conductance.

2.4 Genetic Algorithms

Background and Terminology

Genetic Algorithms (GAs) are search and optimization methods that mimic natural selection and biological evolution to solve optimization and decision problems. The books by David Goldberg [42] and Zbigniew Michalewicz [64] provide thorough introductions to the

field of Genetic Algorithms. A brief overview of genetic algorithms and some definitions of terminology follow.

A *chromosome* is a sequence of gene values. In this dissertation, each gene will usually have a value of either a zero or one. A potential solution to a problem is represented by a chromosome. For graph problems, the number of vertices is the size of the chromosome. A *schema* is a pattern of genes consisting of a subset of genes at certain gene positions. If n is the size of a chromosome, a *schema* is an n–tuple $\{s_1, s_2, \cdots, s_n\}$ where $\forall i, s_i \in \{0, 1, \star\}$. Positions in the schema that have a $\star$ symbol correspond to don't–care positions. The non-$\star$ symbols are called *specific symbols*, and represent the defining values of a schema. The number of specific symbols in a schema is called the *order*, and the length between the first and last specific symbols in a schema is called the *defining length* of the schema. Theorem 32 indicates that the smaller the order of a schema, the more copies it will have in the next generation.

Although genetic algorithms do not specifically work with schemata themselves, schemata are a fundamental concept when analyzing the exploratory process of a genetic algorithm. According to the *building block hypothesis* [42, 51], GAs implicitly favor low–order, high–quality, schemas. Furthermore, as evolution progresses, the GA creates higher order, high–quality schemas out of low–order

schemas. This is partially due to the nature of the crossover operator.

The Schema Theorem [51]

Theorem 32 *For a genetic algorithm using binary encoding, proportional selection, one–point crossover, and strong mutation, the following holds for each schema S represented in the population at time t:*

$$n(S,t+1) \geq n(S,t)\frac{f(S,t)}{F(t)}\left(1 - p_{c,s}\frac{\delta(S)}{r-1} - p_m O(S)\right) \tag{2.41}$$

where $n(S,t)$ is the number of representatives of the schema S at generation t, $f(S,t)$ is the average fitness of the chromosomes containing the schema S in the population at generation t, $F(t)$ is the average fitness of all of the chromosomes in the population at generation t, p_c is the probability of crossover, $\delta(S)$ is the defining length of the schema S, $O(S)$ is the order of the schema S, and r is the length of the chromosome.

Hybrid Genetic Algorithms

Hybrid GAs are those that incorporate a local search operator during each generation on the new offspring. They are essentially a hybridization of a genetic algorithm with a suboptimal heuristic that is tailored specifically for solving a certain problem. Several

hybrid GAs are studied that use a trimmed down variant of the Kernighan–Lin [58] algorithm. Additionally, the data structures and implementation of the algorithm are done in constant time and implemented as described in a paper by Fiduccia and Mattheyses [32]. These optimization algorithms perform a limited, low cost, local search when solving various graph bisection problems.

2.5 Computer Science

NP–Complete Problems

Definition 33 *3SAT - Satisfiability With Exactly 3 Literals Per Clause*

***Input**:*

- *A boolean formula ϕ that is conjunction of disjunctive clauses $C_1, C_2, \ldots, C_n$, each containing exactly 3 literals, where a literal is either a variable, or its negation.*

***Property**: There is a truth assignment to the variables that satisfies ϕ.*

Definition 34 *MAXSAT2 - Maximum Satisfiability With At Most 2 Literals Per Clause*

***Input**:*

- *A boolean formula ϕ that is a conjunction of disjunctive clauses $C_1, C_2, \ldots, C_n$, each containing at most two literals.*

- *Positive integer* k*.*

Property*: There is a truth assignment to the variables satisfying* k *or more clauses of* ϕ*.*

If $k = n$ *then the problem can be solved in polynomial time [21].*

Definition 35 *MAX CUT - Max Cut*

Input*:*

- *A Graph* $G = (V, E)$*.*
- *Positive integer* k*.*
- *A weighting function* $w(u, v)$ *that gives the weight of the edge between* u *and* v*.*

Property*: There is a set* $S \subseteq V$ *such that*

$$\sum_{u,v \in E, u \in S, v \in V-S} w(u, v) \geq k \tag{2.42}$$

Definition 36 *SIMPLE MAX CUT - Simple Max Cut*

Input*:*

- *A Graph* $G = (V, E)$*.*
- *Positive integer* k*.*
- *An edge function* $w(u, v)$ *that is one if and only if there is an edge between* u *and* v*.*

***Property**: There is a set $S \subseteq V$ such that*

$$\sum_{u,v \in E, u \in S, v \in V-S} w(u,v) \geq k \tag{2.43}$$

Definition 37 *MINIMUM GRAPH BISECTION - Minimum Graph Bisection*

***Input**:*

- *A Graph $G = (V, E)$ with an even number of vertices.*
- *Positive integer k.*
- *An edge function $w(u, v)$ that is one if and only if there is an edge between u and v.*

***Property**: There is a partition $V = V_1 \cup V_2$ with $|V_1| = |V_2|$ and $V_1 \cap V_2 = \emptyset$ such that*

$$|\{(u,v) \in E : u \in V_1, v \in V_2\}| \leq k \tag{2.44}$$

If no restriction is made that the sizes of the subsets must be equal, then the minimum graph bisection problem can be solved in polynomial time [57].

Chapter 3 - Information Retrieval

> Mathematics is the tool specially suited for dealing with abstract concepts of any kind and there is no limit to its power in this field.
> –Paul Adrien Maurice Dirac. In P. J. Davis and R. Hersh The Mathematical Experience, Boston: Birkhuser, 1981. American, May 1963.

The next few sections are an attempt at understanding the details of a proof in a well known paper on information retrieval written by Papadimitriou, Tamaki, Raghavan, and Vempala [70]. The paper's terminology and investigative efforts are centered around a textual information retrieval framework called Latent Semantic Indexing (LSI). To keep notation and terms synchronous, their terminology has been left mostly unchanged in this dissertation. However, the ideas presented are independent of terminology and are often interchangeable with similar conceptual structures of parts. For example, *corpora* of *documents* are made of *terms*. Similarly, *populations* of *individuals* are made of *genes*. Indeed, the results and ideas that follow can infer performance results about querying

into any type of relations that can be represented by many highly conducting, connected subgraphs. Several papers have investigated the complexity of these models and algorithms and methods for discovering interactions within them [77], [50].

In the first subsection of this chapter, the details of a probabilistic corpus model by Papadimitriou *et al.* are presented [70]. Next, several counterexamples are described for which their theorem is incorrect. Specifically, when the length of each document is kept constant at 1, their theorem that says that the k largest eigenvalues correspond to the k topics, is shown to be incorrect. Further analysis may reveal that as the length of each document is allowed to increase, there will always be cases for which the theorem is incorrect. On the bright side, intuitions obtained from limiting certain variables in their models lead to conditions for which their theorems do hold. Last, proofs show that query performance does not degrade too much when a small perturbation is added to the matrix A and the dimension of the subspace is reduced to rank–k when working with k topics. Although this type of proof has been presented in many papers [8], [25], the proof presented herein is somewhat unique because it directly corresponds to the terminology and probabilistic model described in the seminal paper of Papadimitriou *et al.* [70].

3.1 Probabilistic Corpus Model

$U =$ The universe of all terms $\{t_1, t_2, \ldots\}$

$T =$ A topic; which is a probability distribution on U

$\mathbf{T} =$ The universe of all topics, each a probability distribution on U

$D =$ A probability distribution on $\mathbf{T} \times Z^+$

$\mathbf{C} =$ A corpus model $(U, \mathbf{T}, D)$

Documents are formed by picking some convex combination of topics and a document length l. For each $i = 1 \ldots l$, a term from a topic is chosen based on their corresponding probabilities. In effect, a document is a random sampling of terms from a random sampling of topics. A corpus is simply a collection of documents generated from this process.

Definition: *C*orpus C is **pure** if each document $d \in C$ is on a single topic.

Definition: *C*orpus C is ε**–separable**, where $0 \leq \varepsilon < 1$, if a "primary" set of terms U_T is associated with each topic $T \in \mathbf{T}$ such that

1. U_T are mutually disjoint.

2. For each topic T, the total probability T assigns the terms in U_T is at least $1 - \varepsilon$.

Note that ε–separability implies that terms from other topics may be shared across documents when $\varepsilon \neq 0$. However, the terms shared must not be on any other topic when $\varepsilon = 0$

Definition: *T*he rank–k SVD is δ**–skewed** on a corpus C if, for each pair of documents d_1, d_2 on *different* topics

$$v_{d_1} \cdot v_{d_2} \leq \delta \parallel v_{d_1} \parallel \parallel v_{d_2} \parallel$$

and for each pair of documents d_1, d_2 on the *same* topic

$$v_{d_1} \cdot v_{d_2} \geq (1 - \delta) \parallel v_{d_1} \parallel \parallel v_{d_2} \parallel$$

For small δ, this means that documents on the same topic will be nearly parallel and documents on different topics will be nearly orthogonal. The theorem will show that the rank–k SVD is $O(\varepsilon)$–skewed on a corpus generated from an ε–separable, pure corpus model with k topics.

Let $\Omega_i = \{t_1, \ldots, t_{m_i}\}_i$ be the sample space of topic T_i with

$$p_{ij} = Pr[t_j \in T_i]$$

Notice that the sum of the probabilities in each topic T_i satisfy by definition

$$\sum_{j=1}^{m_i} p_{ij} \geq \begin{cases} 1-\varepsilon, & \text{if C is } \varepsilon\text{–separable} \\ 1, & \text{if C is 0–separable} \end{cases}$$

3.2 A Counterexample To The LSI Theorem of Papadimitriou

One of the goals of the paper was to show that the rank–k SVD is $O(\varepsilon)$–skewed on a corpus generated from an ε–separable corpus model with k topics. For the sake of a simplistic case, the authors first attempted to discover which topics the reduced rank SVD represents when performed on block matrices generated from a **pure** corpus model that is 0–separable. Next, they apply a perturbation to the matrix and show that queries into the reduced SVD space of the perturbation do not move very much.

A main part of Papadimitriou *et al.*.'s work was that it tried to show that for a block diagonal matrix with k blocks, generated from a corpus model to be described in the next section, the k largest of ***all*** of the eigenvalues of the nearly block diagonal matrix $A^T A$ are the maximum eigenvalues of each block $B_i^T B_i$, for $i = 1, \cdots, k$, with high probability [70]. Therefore, when projecting onto the k largest eigenvectors of V, a *document* query vector created mainly from a

block $B_i \in A$ will likely be projected only in the direction of the maximum eigenvector of the i'th block of the document–document Gram (or autocorrelation) matrix $B_i^T B_i$. Likewise, a *term* query vector from a block $B_i \in A$ will only be projected in the direction of the maximum eigenvector of the i'th block of the term–term Gram matrix $B_i B_i^T$. Their idea was that matrices generated from their probabilistic model corresponded to random symmetric matrices. They then made a correspondence between these random symmetric matrices and all graphs. Citing several papers that prove bounds between the conductance and the spectrum of a graph, the authors then contend that the k largest of the eigenvalues of a particular Gram matrix will match up with high probability to their corresponding k topics [8], [1], [2].

Unfortunately, their results do not ensure that the k largest eigenvalues are all greater than the second largest eigenvalue of each block, for all corpuses generated by their model. Their theorem and a short summary are stated below.

Theorem 38 *(Papadimitriou et al. [70]) Let* **C** *be a pure,* 0*–separable corpus model with* k *topics such that the probability each topic assigns to each term is at most* τ*, where* τ *is a sufficiently small constant. Let* C *be a corpus of* m *documents generated from* **C***. Then the rank–*k *SVD is* 0*–skewed on* C *with probability* $1 - O\left(\frac{1}{m}\right)$*.*

Term–document matrices for **pure**, 0–separable corpora contain blocks. Each block represents documents solely drawn from one particular topic. Let A be the $m \times n$ term–document matrix representing the **pure** corpus C. Then A is block diagonal because C is pure and 0–separable. That is, each document contains only terms from the "primary" set of terms, and so will only have positive entries in the block of terms B_i corresponding to the document's one topic, T_i. Any document on a particular topic will have an entry of 0 for terms outside of the topic because we have assumed the corpus model is pure and 0–separable. $A^T A$ is block diagonal with blocks, $B_i^T B_i$. The (i, j) entry of $A^T A$, denoted $A^T A_{ij}$, represents the dot product between the i'th and j'th documents.

$$A^T A_{ij} = d_i \cdot d_j \tag{3.1}$$

Notice that $A^T A_{ij}$ is 0 when

$$d_i \cdot d_j = 0 \tag{3.2}$$

Which happens when d_i and d_j are from different topics. Therefore, $A^T A$ is block diagonal because the corpus model is pure. However, $d_i \cdot d_j = 0$ does not necessarily imply that d_i and d_j are in separate blocks because two samplings from the same topic may produce two documents containing completely different terms from within

that topic. Also note that $A^T A$ is always symmetric because

$$(A^T A)^T = A^T A \tag{3.3}$$

Each block $B_i^T B_i$ contains the dot products between all pairs of documents in the i'th topic, T_i. Because $B_i^T B_i$ is symmetric, the authors say that it can be taken to be the adjacency matrix of a random bipartite multigraph. However, the graphs produced are clearly not bipartite because they are highly connected.

The vertices are the documents. An edge connects two documents if and only if their dot product is greater than zero. The weight of an edge is simply the dot product of the two documents it connects. In effect,

$$weight(i, j) = d_i \cdot d_j \tag{3.4}$$

The subgraphs induced by $B_i^T B_i$ are all disjoint because each document is on a single topic T_i (because C is pure) and all of the topics are disjoint (because C is 0–separable). Within a particular subgraph, if two documents share no common terms, there will not be an edge connecting them. However, if we add a requirement that the maximum term probability is sufficiently small, the probability that two documents on the same topic will not share terms will go to zero. Thus, each subgraph will be highly connected with a high

probability. However, it should be noted that the number of terms in each topic and the number and length of documents drawn from it may force several of its eigenvalues to outweigh another topic's largest eigenvalues.

The approach used by the authors was to show that the k largest of ***all*** of the eigenvalues of the matrix $A^T A$ are exactly the maximum eigenvalues of each block $B_i^T B_i$ for $i = 1, \cdots, k$, where k is the number of topics or *blocks* in A [70]. If this can be shown, then a query document created solely from a topic T_i will be projected only in the direction of the maximum eigenvector of the i'th block $B_i^T B_i$. Two documents from different topics will be perpendicular because their corresponding topic eigenvectors are perpendicular. Therefore the rank–k LSI will be 0–skewed as claimed. However, this is not to say that each singular vector will always correspond to one topic because if any two eigenvalues are the same, the rank–k SVD may produce a set of singular vectors to which all topics do not have a single corresponding eigenvector. Fortunately, the probability that two eigenvalues will be equal goes to zero as the number of documents increases.

The authors used a theorem from spectral graph theory that indicates that as the first and second eigenvalues of a graph separate, the conductance of the graph increases [2], [1]. They reason that since the conductance of the graphs are high, their corresponding

first two eigenvalues will also separate and therefore, the k largest eigenvalues will belong to the k eigenvectors spanning each of the k topics. However, as Theorem 39 shows, this statement is clearly incorrect with high probability for certain choices of the lengths of the documents. In other words, for certain parameters, the top k eigenvectors will *not* represent every topic with high probability. Therefore, although each topic's corresponding graph is highly conducting, topics must have similar conductance values in order to help guarantee that the largest k eigenvalues of the entire matrix are the largest k eigenvalues of each block.

Theorem 39 *A simple counterexample to Theorem 38 is formed by creating a corpus model with the following properties.*

- *The length of every document is* 1*.*
- *There are two topics* T_1 *and* T_2 *each containing* t *disjoint terms respectively, with each term being equally likely to be chosen.*
- *The probability that topic* T_1 *is chosen is* $\frac{2}{5}$ *and the probability that topic* T_2 *is chosen is* $\frac{3}{5}$*.*

The following analysis shows how these conditions provide a counterexample in that the k *largest eigenvalues do not correspond to the* k *topics with high probability, thereby violating the statement in Theorem 38.*

Proof: First focus on a particular topic T_i. Within T_i there are t terms whose probabilities are all equally likely. However the document length is restricted to a length of one. Therefore, within a particular topic, there will only be t types of vectors that are possible to be generated. Concretely, these vectors will be the standard basis vectors e_i, each corresponding to the instance where the i'th term is chosen.

Let $\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_m$ be a sequence of choices of these standard basis vectors, where ε_i represents the i'th choice and is an element of the set $\{e_1, \ldots, e_t\}$. Create a $t \times m$ matrix A by setting the i'th column of A equal to the standard vector represented by ε_i. Next, form the $m \times m$ Gram matrix $A^T A$ whose entries are the following by definition of matrix multiplication and the conditions imposed on ε_i.

$$(A^T A)_{ij} = \varepsilon_i \cdot \varepsilon_j = \begin{cases} 1, & \text{if } \varepsilon_i = \varepsilon_j \\ 0, & \text{otherwise} \end{cases}$$

Let N_i be the number of vectors of type e_i chosen. Then clearly,

$$N_1 + N_2 + \cdots + N_t = m$$

Due to a theorem by McDiarmid concerning the method of bounded differences [63], each of the N_i are equal to $\frac{m}{t}$ with high probability.

Now focus on the Gram matrix A^TA. It will have N_i rows that are the same $\forall i = 1 \dots t$. Therefore, its rank will be at most t. This implies that the dimension of its nullspace is $m - t$. Then, by the definition of the nullspace, 0 will be an eigenvalue exactly $m - t$ times. The other t eigenvalues of the matrix will be N_i with high probability. This is true for the following reasons. The eigenvalues and eigenvectors do not change after permuting the rows and columns of a matrix. Permute the columns of A so that all of the standard basis vectors corresponding to a single term appear together in the sequence $\varepsilon_1.\varepsilon_2, \dots, \varepsilon_m$. Then the Gram matrix A^TA will contain t square blocks containing only ones, each with size $N_1 \times N_1, \dots, N_t \times N_t$. Clearly the all ones vector is an eigenvector for each block with corresponding eigenvalue equal to N_i. Furthermore, when zeroes are padded in for entries outside of the block, the vector becomes an eigenvector of the entire matrix. Since the eigenvalues of A^TA are equal to the union of the eigenvalues of each block of ones in A^TA, the eigenvalues of A^TA are equal to $N_1, \dots, N_t$, with each $N_i = \frac{m}{t}$ with high probability.

Given that topic T_1 is chosen with probability equal to $\frac{2}{5}$, the expected number of times it will be chosen out of m total choices will be $\frac{2}{5}m$. Likewise, the expected number of times topic T_2 will be chosen will be $\frac{3}{5}m$. By the above analysis, the eigenvalues corresponding to topics T_1 and T_2 will be centered around $\frac{\frac{2}{5}m}{t}$ and $\frac{\frac{3}{5}m}{t}$

respectively with high probability, due to another application of McDiarmid's method of bounded differences [63]. Therefore, the second largest eigenvalue of T_2 will be larger than the largest eigenvalue of T_1 with high probability, and so the eigenvectors corresponding to the largest k eigenvalues of $A^T A$ will not account for every topic. ■

One conjecture is that no matter how l is generated or τ is selected, there will always be a way to construct probabilities on the topics such that the k eigenvectors corresponding to the k largest eigenvectors of $A^T A$ do not account for each topic. It remains to be discovered for which choices of the variables l and t and term and topic probabilities that the theorem remains true.

3.3 Matrices drawn from ε–separable corpora

Now consider what happens to the LSI query accuracy when an $O(\varepsilon)$–sized perturbation is added to a pure 0–separable corpus model. The added perturbation allows us to consider corpora generated from a pure corpus model that is $O(\varepsilon)$–separable. What are the properties of the perturbed corpus? One property is that the documents drawn from it may span multiple topics. The goal is to show that the rank–k LSI on the perturbed corpus is only $O(\varepsilon)$–skewed with high probability. This would mean that the rank–k LSI on a perturbed corpus is still able to classify the documents approx-

imately, even though the documents may have non–zero entries for terms contained in several different topics due to the perturbation. Note that most of the weight in a document vector will still be on one topic when ε is small. Essentially, the perturbation is adding edges with a small weight between documents on different topics, making the Gram matrix become less block diagonal. The weight (dot–product) will be small for these added edges because one of the corresponding vector components of two documents on different topics will usually be zero. Presumably, the eigenvalue contributions due to the perturbation will also be small and therefore will be unable to overwhelm the set of maximum eigenvalues of each block.

The next theorem shows that this is indeed the case. Although this type of proof has been presented in many papers [8], [25], the proof is recreated to directly correspond to the terminology and probabilistic model described in the seminal paper of Papadimitriou *et al.* [70]. Before proving it, the following lemma, stated without proof, is used to show that if the k largest singular values of a matrix B are well–separated from the remaining singular values, then the subspace spanned by the corresponding singular vectors is preserved well when a small perturbation is added to B. The lemma stems from a theorem by Stewart about perturbing a symmetric matrix [81], [43].

Lemma 40 *(Stewart) Let B be an $m \times n$ matrix of rank r with singular value decomposition*

$$B = U\Sigma V^T$$

Suppose that, for some k, $1 \leq k < r$, $\frac{\sigma_k}{\sigma_{k+1}} > \frac{c\sigma_1}{\sigma_k}$ for a sufficiently large constant c. Let F be an arbitrary $m \times n$ matrix with $\| F \|_2 \leq \varepsilon$, where ε is a sufficiently small positive constant. Let $A = B + F$ and let $U'\Sigma'V'^T$ be its singular value decomposition. Let U_k and U_k' be $m \times k$ matrices consisting of the first k columns of U and U' respectively. Then, $U_k' = U_k R + G$ for some $k \times k$ orthonormal matrix R and some $m \times k$ matrix G with $\| G \|_2 \leq O(\varepsilon)$.

The next lemma will also be necessary for proving the theorem. In the following two theorems, the Euclidean vector norm will be denoted by $|\cdot|$. Let U_k' and U_k denote the basis matrices of the k–dimensional space that the rank–k SVD applied to A and B respectively identifies. Let A^i denote the transpose of the i^{th} document of A, i.e., the transpose of i^{th} column of the **perturbed** term–document matrix A. Let B^i denote the transpose of the vector corresponding to the i^{th} document in B, i.e., the transpose of i^{th} column of the pure **unperturbed** term–document matrix B. Then for any i,

$$A_k^i = A^i U_k' \quad \text{and} \quad B_k^i = B^i U_k \tag{3.5}$$

Lemma 41 *Given the definitions of the matrices in Lemma 40, and the following assumptions*

- $|F^i U_k| = O(\varepsilon |A^i|)$
- $|F^i| \leq |B^i|$
- $|B^i_k| = \Theta(|B^i|)$

the following is true

$$|A^i_k - B^i_k R| = O(\varepsilon |B^i_k|) \tag{3.6}$$

Proof:

$$
\begin{aligned}
|A_k^i - B_k^i R| &= |A^i U_k^{'} - B^i U_k R| \\
&= |A^i (U_k^{'} - U_k R) + (A^i - B^i) U_k R| && (3.7)\\
&= |A^i G + (F^i) U_k R| && (3.8)\\
&\leq |A^i G| + |(F^i) U_k R| && (3.9)\\
&\leq |A^i| \parallel G \parallel_2 + |F^i U_k| \parallel R \parallel_2 && (3.10)\\
&= |A_i| O(\varepsilon) + |F^i U_k| && (3.11)\\
&= |A_i| O(\varepsilon) + O(\varepsilon |A^i|) && (3.12)\\
&= O(\varepsilon |A^i|) && (3.13)\\
&= O(\varepsilon |B^i + F^i|) && (3.14)\\
&\leq O(\varepsilon (|B^i| + |F^i|)) && (3.15)\\
&= O(\varepsilon |B^i|) && (3.16)\\
&= O(\varepsilon |B_k^i|) && (3.17)
\end{aligned}
$$

Inequality 3.9 follows from the triangle inequality. Inequality 3.10 follows from the definition of the matrix two norm. Equation 3.11 is true because $\parallel U_k^{'} - U_k R \parallel_2 = \parallel G \parallel_2 = O(\varepsilon)$ as per Lemma 40 and because the two norm of any unitary matrix is 1. Equation 3.12 follows because of the assumption that $|F^i U_k| = O(\varepsilon |A^i|)$. Inequality 3.15 follows from the triangle inequality. Equation 3.16 follows

from the assumption that $|F^i| \leq |B^i|$, and Equation 3.17 because of the assumption that $|B^i_k| = \Theta(|B^i|)$. ■

Theorem 42 *Let* **C** *be a pure, ε–separable corpus model with k topics such that the probability each topic assigns to each term is at most τ, where $\tau > 0$ is a sufficiently small constant. Let B be the term–document matrix of a corpus of m documents generated from* **C** *and let C be a corpus whose term document matrix is $A = B + F$ where $\| F \|_2 \leq \varepsilon$. Given the following assumptions,*

- $|F^i U_k| = O(\varepsilon |A^i|)$
- $|F^i| \leq |B^i|$
- $|F^i| \leq |B^i_k|$
- $|B^i_k| = \Theta(|B^i|)$

the rank–k LSI is $O(\varepsilon)$–skewed on C with probability $1 - O(\frac{1}{m})$.

Proof: For any pair of documents i, j the difference between the dot products of two documents in A and two corresponding document's dot products in B will be bounded from above. In effect,

$$|A^i_k \cdot A^j_k - B^i_k \cdot B^j_k| = O(\varepsilon |B^i_k||B^j_k|) \tag{3.18}$$

If this can be proved, then the theorem will follow by using simple substitutions.

$$\begin{aligned} B^i, B^j \text{ in different topics } &\Rightarrow B^i \cdot B^j = 0 \\ &\Rightarrow |A_k^i \cdot A_k^j - B_k^i \cdot B_k^j| = |A_k^i \cdot A_k^j| \\ &\Rightarrow |A_k^i \cdot A_k^j| = O(\varepsilon |B_k^i||B_k^j|) \end{aligned}$$

and

$$\begin{aligned} B^i, B^j \text{ in the same topic } &\Rightarrow B^i \cdot B^j = |B^i||B^j| \\ &\Rightarrow |A_k^i \cdot A_k^j - B_k^i \cdot B_k^j| = |A_k^i \cdot A_k^j - |B^i||B^j|| \\ &\Rightarrow |A_k^i \cdot A_k^j| = |B^i||B^j| \pm O(\varepsilon |B_k^i||B_k^j|) \end{aligned}$$

Therefore, the perturbed matrix A is $O(\varepsilon)$–skewed as required by the theorem.

So, the proof of the theorem reduces to proving the proposition that

$$|A_k^i \cdot A_k^j - B_k^i \cdot B_k^j| = O(\varepsilon |B_k^i||B_k^j|)$$

The following analysis shows that this is indeed the case.

$$
\begin{aligned}
|A_k^i \cdot A_k^j - B_k^i \cdot B_k^j| &= |A_k^i \cdot A_k^j - B_k^i R \cdot B_k^j R| && (3.19)\\
&= |(A_k^i - B_k^i R)A_k^j + B_k^i R \cdot (A_k^j - B_k^j R)| && (3.20)\\
&= |v_i \cdot A_k^j + B_k^i R \cdot v_j| && (3.21)\\
&\leq |v_i \cdot A_k^j| + |B_k^i R \cdot v_j| && (3.22)\\
&\leq |v_i||A_k^j| + |B_k^i R||v_j| && (3.23)\\
&= |v_i||A_k^j| + |B_k^i||v_j| && (3.24)\\
&= O(\varepsilon|B_k^i|)|B_k^j + F^j| + O(\varepsilon|B_k^j|)|B_k^i| && (3.25)\\
&\leq O(\varepsilon|B_k^i|)(|B_k^j| + |F^j|) + O(\varepsilon|B_k^j|)|B_k^i| && (3.26)\\
&= O(\varepsilon|B_k^i|)O(\varepsilon|B_k^j|) + O(\varepsilon|B_k^j|)|B_k^i| && (3.27)\\
&= O(\varepsilon|B_k^j|)(O(\varepsilon|B_k^i|) + |B_k^i|) && (3.28)\\
&= O(\varepsilon|B_k^j|)O(\varepsilon|B_k^i|) && (3.29)\\
&= O(\varepsilon|B_k^j||B_k^i|) && (3.30)\\
&= O(\varepsilon|B_k^i||B_k^j|) && (3.31)
\end{aligned}
$$

The first equation, 3.19 follows because the angle between two vectors is not affected when both vectors are multiplied by a unitary matrix. This fact is proved in Theorem 14 of Chapter 2. Equality 3.21 follows from lemma 41. Here, v_i is a vector with $|v_i| = O(\varepsilon|B_k^i|)$. Likewise, v_j is a vector with $|v_j| = O(\varepsilon|B_k^j|)$. Inequalities 3.22 and 3.23 follow from the triangle and Cauchy–Schwartz inequalities,

respectively. Equality 3.24 follows from the fact that $A^j = B_k^j + F^j$ and because R is orthonormal with $\| R \|_2 = 1$. Inequality 3.26 follows due to the triangle inequality. Equality 3.27 is true because of the assumption that $|F^i| \leq |B_k^i|$. The rest of the equations follow from the relations represented by the asymptotic notation.

■

The paper under consideration [70] provided a good probabilistic model and positive indications that a theorem about LSI's performance may exist. However, the exact conditions and requirements for which the theorem remains true have yet to be described. Since its printing in 1998, several papers have clarified furthered the results obtained (see [56] , [8], [50], and [25]).

Chapter 4 - Graph Partitioning

> The formulation of a problem is often more essential than its solution, which may be merely a matter of mathematical or experimental skill.
> –Albert Einstein

4.1 Problem Statement

A bisection of a graph $G = (V, E)$ with an even number of vertices is a pair of disjoint subsets $V_1, V_2 \subset V$ of equal size with $V_1 \cup V_2 = V$. The cost of a bisection is the number of edges $(a, b) \in E$ such that $a \in V_1$ and $b \in V_2$. The Minimum Graph Bisection problem takes as input a graph G with an even number of vertices, and returns a bisection of minimum cost. The Minimum Graph Bisection problem has been shown to be NP–Complete by the following reductions [38, 39]

$$\begin{aligned} \text{3SAT} &\leq_p \text{MAXSAT2} \\ &\leq_p \text{SIMPLE MAX CUT} \\ &\leq_p \text{MINIMUM GRAPH BISECTION} \end{aligned}$$

The definitions of these problems are given starting on page 42. If no restriction is made that the sizes of the subsets must be equal, then the problem can be solved in polynomial time [57].

Let G be a graph on n vertices and $\alpha > 0$ be given. An α–*edge separator* is a partition of the vertices of G into two disjoint sets A and B such that

- $\max\{|A|, |B|\} \leq \alpha n$

The α–*edge separator problem* is to find an optimal α–edge separator with respect to the number of edges between the two partitions.

The α–*vertex separator problem* is to partition of the vertices of G into three disjoint sets A, B, and C such that

- No edge of G has one endpoint in A and the other endpoint in B
- $\max\{|A|, |B|\} \leq \alpha n$
- $|C|$ is minimized

Additional evidence for the Minimum Graph Bisection problem's difficulty is that it has been shown that, for graphs on n vertices, it is NP–hard to find α–vertex separators of size no more than $OPT + n^{\frac{1}{2}-\varepsilon}$, where OPT is the size of the optimal solution and $\varepsilon > 0$. Specifically, Bui and Jones show that there is no algorithm that guarantees to find a vertex separator of size within $OPT + n^{\frac{1}{2}-\varepsilon}$

for a maximum degree 3 graph with n vertices unless $P = NP$ [17]. It is well known that good edge and vertex separators can be converted back and forth between each other [71], [41]. This implies that a restriction on the degree of the graph will not help solve the Minimum Graph Bisection problem. Unless $P = NP$, the problem is intractable even for graphs of bounded degree.

4.2 Problem Motivation

This Minimum Graph Bisection problem arises in many important scientific problems. Several examples include the splitting of data structures between processors for parallel computation, the placement of circuit elements in engineering design, and the ordering of sparse matrix computations [18]. In addition, the problem is NP–Hard, making it a prime candidate for research and study.

The motivation for using spectral partitioning is that the eigenvalues have been shown to have many relationships to properties of graphs. Moreover, every eigenvalue and eigenvector of a matrix can be computed efficiently in polynomial time. Therefore, eigenvalues and eigenvectors are prime candidates for constructing efficient algorithms for solving various graph problems.

4.3 Literature Survey

Many heuristics have been developed for this problem. Frieze and McDiarmid provide an analysis of the performance of algorithms on random graphs [36]. Perhaps the best known heuristic is the Kernighan–Lin heuristic [58], [16]. The Kernighan–Lin heuristic has a time complexity of $O(n^3)$ and is P–Complete [73], [48]. Fiduccia and Mattheyses gave a simplification of the Kernighan–Lin heuristic that has time complexity $\Theta(E)$ [32]. The efficiency is gained by sorting data efficiently using a method called the bucket sort. A simulated annealing approach was used by Johnson *et al.* [53]. Singular value decomposition has also proved to be a useful tool when clustering graphs [27], [56]. Spectral techniques for graph bisection were motivated by the work of Fiedler [33]. Indeed, spectral techniques are often used to enhance graph algorithms [3], [71], [4], [9]. Donath and Hoffman were among the first to suggest using spectral techniques for graph partitioning [26]. Alpert and Yao showed that more eigenvectors may help improve results [5]. Their main result showed that when all eigenvectors are used, the min–cut graph partitioning and max-sum vector partitioning problems objectives are identical. Graph partitioning with genetic algorithms has been studied extensively [60], [19], [44]. Most GA

methods incorporate some other algorithms and heuristics, such as spectral partitioning or Kernighan–Lin.

Approximation of minimum bisection size was recently studied by Feige *et al.* [31]. They discovered an algorithm that finds a bisection within $O(\sqrt{n}logn)$ of the optimal. The algorithm makes extensive use of minimum–ratio cuts and of dynamic programming. More recently, Andreev and Räcke presented a polynomial time approximation algorithm for the (k, v)–balanced partitioning problem that gives an $O(log^2 n)$–approximation ratio with respect to the number of edges between the different partitions [6]. The (k, v) partitioning problem is to divide the vertices of a graph into k almost equal sized components, each with size less than $|V| \cdot \frac{v}{k}$ so that the number of edges between the different components is minimized. Note that the Minimum Graph Bisection problem is equivalent to the $(2, 1)$–balanced partitioning problem.

4.4 Adjacency Matrix Representations

There are many different ways of representing a graph as an adjacency matrix. One of the goals of this dissertation is to identify and investigate many different representations in order to discover a unifying theorem for spectral bisection that is representation independent. Given a graph G, it is possible to construct an adjacency matrix for the graph with the property that the graph can

be completely constructed solely from the information contained in the adjacency matrix. Unfortunately, because two isomorphic graphs may have a different labeling of their vertices, there may be many adjacency matrices that correspond to the same (unlabeled) graph. However, the eigenvectors of such graphs do not depend on the labeling of the vertices. It is important to note that there exist isomorphic graphs that are cospectral, but do not share the same eigenvectors [83].

Let D be the diagonal matrix obtained by letting the degree of vertex i be located at position D_{ii}. It is interesting to note that almost all of the representations below can be represented by some setting of λ and μ in the following equation [23].

$$F_G(\lambda, \mu) = |\lambda I + \mu D - A| \tag{4.1}$$

However, the Seidel spectrum is not as easily represented using solely this function. Spectral analysis may be able to proceed in general by examining the characteristic polynomial over two variables of the matrix represented by $F_G(\lambda, \mu)$, or other similar functions.

The adjacency matrix of a graph G will be denoted as the representation type symbol with the graph name as a subscript. For example, L_G represents the Laplacian of the graph G. The repre-

sentations used are defined with their symbols as follows.

Type	**Symbol**

0,1 Adjacency **A**

$$A_{ik} = \begin{cases} 0, & \text{if } k = i \text{ or } v_k \text{ is not adjacent to } v_i \\ 1, & \text{if } v_k \text{ is adjacent to } v_i, k \neq i \end{cases} \tag{4.2}$$

Adjacency of the complement $\overline{\mathbf{A}}$

This representation simply switches the connectivity describing roles of the zeroes and ones in the 0,1 Adjacency matrix.

$$\overline{A}_{ik} = \begin{cases} 0, & \text{if } k = i \text{ or } v_k \text{ is adjacent to } v_i \\ 1, & \text{if } v_k \text{ is not adjacent to } v_i, k \neq i \end{cases} \tag{4.3}$$

Laplacian **L**

The Laplacian has been studied extensively. The Laplacian is defined as

$$L = D - A \tag{4.4}$$

Many properties of the Laplacian are listed in a paper on the performance of spectral graph partitioning methods by Guattery and Miller [45].

The Signless Laplacian $|\mathbf{L}|$

The signless Laplacian is defined as

$$|L| = D + A \tag{4.5}$$

The Negative Degree Laplacian $|\overline{\mathbf{L}}|$

This representation is equivalent to the signless Laplacian with negative degrees along the diagonal.

$$|\overline{L}| = A - D \tag{4.6}$$

The Siedel 0,1,-1 Adjacency **S**

Van Lint and Seidel first proposed this representation in their work on equilateral point sets in elliptic geometry [84]. Seidel later visited the representation in a survey on two–graphs [76]. Sometimes these matrices are called Seidel Matrices.

$$S_{ik} = \begin{cases} -1, & \text{if } v_k \text{ is adjacent to } v_i, k \neq i \\ 0, & \text{if k = i} \\ +1, & \text{if } v_k \text{ is not adjacent to } v_i, k \neq i \end{cases} \tag{4.7}$$

Note that this representation is related to the all ones matrix, J, the identity matrix, I, and the regular adjacency matrix A by the

following equation

$$S = J - 2A - I \tag{4.8}$$

The Modified Seidel Adjacency $\overline{S}$

This adjacency matrix representation simply reverses the roles of -1 and 1 in the Seidel representation.

$$\overline{S_{ik}} = \begin{cases} +1, & \text{if } v_k \text{ is adjacent to } v_i, k \neq i \\ 0, & \text{if k = i} \\ -1, & \text{if } v_k \text{ is not adjacent to } v_i, k \neq i \end{cases} \tag{4.9}$$

Note that this representation is related to the all ones matrix, J, the identity matrix, I, and the regular adjacency matrix A by the following equation

$$\overline{S} = 2A - J + I = -S \tag{4.10}$$

This may be an original contribution, but the eigenvectors of $\overline{S}$ are essentially the same as the eigenvectors of S.

The Modified Seidel cD,1,-1 Adjacency $\overline{S}_{(c)}$

This representation may also be an original contribution. Its performance is discussed in Chapter 5's subsection on eigenvector

search and partitioning on page 87.

$$\overline{S_{(c)ik}} = \begin{cases} 1, & \text{if } v_k \text{ is adjacent to } v_i, k \neq i \\ c * deg(v_k), & \text{if k = i} \\ -1, & \text{if } v_k \text{ is not adjacent to } v_i, k \neq i \end{cases} \tag{4.11}$$

Note that this representation is related to the all ones matrix, J, the identity matrix, I, the degree matrix D, and the regular adjacency matrix A by the following equation

$$\overline{S_c} = (2A - J + I) + cD \tag{4.12}$$

4.5 Graph Types

Geometric, random degree, highly clustered, caterpillar, grid, path, and real world graphs are all studied and used as the basis of experiments. A description of the notation and construction details of each type of graph follows.

1. **Random Graphs** - $G_{n.d}$: A graph on n vertices created by placing an edge between two vertices with probability p. The probability p is chosen so that the expected vertex degree of the graph $p(n-1)$ is equal to the input parameter d. Random graphs were introduced in a seminal paper by Erdős and Rényi [30], and have been studied extensively ever since. Ran-

dom graphs were also tested in the simulated annealing graph bisection study of Johnson *et al.* [53].

2. **Random Geometric Graphs** - $U_{n.d}$: A graph on n vertices created by associating n vertices with different locations on the unit square. The unit square is located in the first quadrant of the Cartesian Plane. Therefore, each vertex's location is represented by a pair $(x, y) \in \Re$ for some $0 \leq x, y \leq 1$. An edge is created between two vertices if and only if the Euclidean distance (Definition 8 on page 11) between the two is d or less. The expected average degree for these graphs is approximately $n\pi d^2$ [53]. These graphs were defined and tested in the simulated annealing study by Johnson *et al.* [53].

3. **Caterpillar Graphs** - CAT_n : A caterpillar graph on n vertices. Two of the vertices are the head and tail of the caterpillar. Next, $\lfloor \frac{(n-2)}{7} \rfloor$ vertices are chosen to represent the discs in the spine of the caterpillar. To each of these vertices is then attached 6 legs from the remaining $(n-2) - \lfloor \frac{(n-2)}{7} \rfloor$ vertices. The caterpillar graphs considered here have an even number of discs in their spine. This implies that the only possible caterpillars have an even number of vertices with

$$n \in \{(i * 6 + i) + 2 : \forall i \geq 2, i \bmod 2 = 0\} = \{16, 32, 44, \cdots, 352, \cdots\}$$

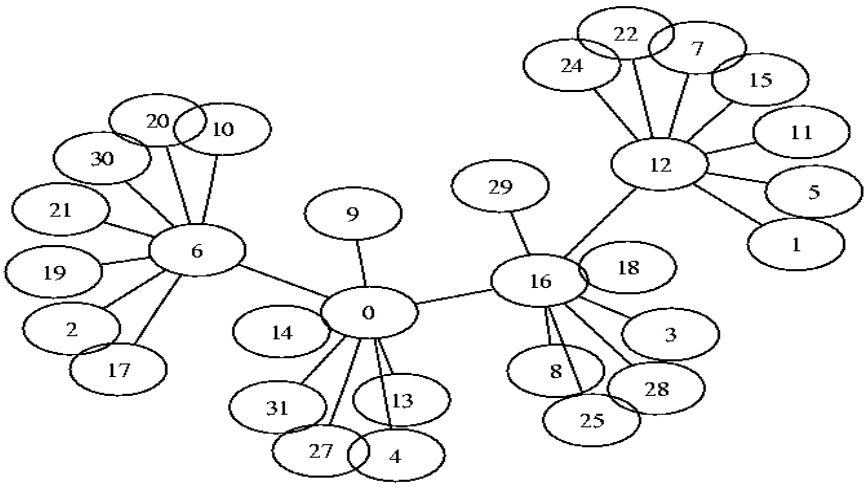

Figure 4.2: A caterpillar of size 32.

Here, i represents the total number of discs on the spine. Caterpillar graphs have been shown to be very difficult for standard graph bisection algorithms such as Kernighan–Lin [54, 19]. In addition, the minimum bandwidth problem for caterpillars with hair length 3 was shown to be NP–Complete by Monien [67].

4. **Grid Graphs** - $GRID_{r.c.b}$: A grid graph on $n = r * c$ vertices. There are r rows and c columns. The optimum bisection the graph is known to be b. For example, a GRID20.25.21 is the graph obtained by constructing a grid of vertices with 20 rows and 25 columns. The best bisection, with size=21, starts between columns 12 and 13 and makes 10 vertical cuts to the center of the graph. Next, 10 vertical cuts are made from the

bottom of the graph to the center, and then one cut to the left. This is a bisection of the graph with a capacity of 21 edges. These graphs were studied in Bui and Moon's work with genetic algorithm's for graph partitioning [19].

5. **Highly Clustered Random Graphs** - $HCG_{n.c.in.out}$: A highly clustered random graph with n vertices containing c clusters with high connectivity. First, the vertices are randomly divided into c clusters, or sets. Next, edges are placed between vertices in the same set with probability equal to in. Edges are placed between two vertices in different sets with probability out. These graphs were studied in conjunction with eigenvector solutions by Bopanna [14]. Bopanna limited his analysis to these types of graphs to ensure that at least one bisection was sufficiently smaller than the average bisection of a graph. Without this limitation, good heuristics were shown to be almost indistinguishable from terrible heuristics in an average case analysis [14].

6. **Path Graphs** - P_n : A graph containing $n-1$ edges between vertices forming a single path from one start vertex to another end vertex, both with degree 1. All other vertices have degree two. This type of graph can be arranged into a one dimensional line.

Chapter 5 - Spectral Graph Bisection

> "It is more important to have beauty in one's equations than to have them fit experiment... If one is working from the point of view of getting beauty in one's equations, and if one has really a sound insight, one is on a sure line of progress. If there is not complete agreement between the results of one's work and experiment, one should not allow oneself to be too discouraged, because the discrepancy may well be due to minor features that are not properly taken into account and that will get cleared up with further development of the theory."
> –Paul Adrien Maurice Dirac. Taken from Scientific American, May 1963.

The relationships between the spectrum of a graph (which are the eigenvalues of its adjacency matrix) and the properties of the graph itself have been popular topics for research and discovery in the last fifty years [23]. The spectrum has been used to help solve the problem of graph isomorphism [83]. However, many important questions regarding graph's spectra still remain open. For

example, the complete set of graphs that are determined by their spectrum is not fully known [83]. Furthermore, it is still interesting to examine the actual eigenvectors associated with eigenvalues in different representations. It turns out that certain eigenvectors of an adjacency matrix sometimes tend to partition its corresponding graph into two halves such that the conductance of the parts is high. Eigenvectors have been used to find good minimum cut partitions and to find good colorings for graphs [14], [9], [7]. However, most studies usually only focus on one eigenvector of one representation type for the adjacency matrix. This eigenvector is called the Fiedler vector, and corresponds to the second smallest eigenvalue of the Laplacian [33].

As a result of the primary focus on algorithms based on the Fielder vector, data and theorems for eigenvectors of other representation types seem to be lacking in the graph partitioning field. This is unfortunate because experimental evidence has shown that many representations provide similar performance, and work better in different situations. The field of spectral partitioning requires an analysis of all the eigenvectors of all of the different representation types in order to determine which eigenvectors and representation types provide the best solution qualities for which problems. For example, a paper by Guattery and Miller describes a family of bounded degree–3 planar graphs called ”roach graphs”

and proves that the "simple spectral algorithm" will produce bad cuts for these types of graphs [46]. However, the analysis is done solely for the Fielder vector of the Laplacian. Surprisingly, empirical results show that the singular vector corresponding to the *third smallest* singular value of the Laplacian, is actually the one that experimentally gives the exact minimum bisection for this type of graph. Furthermore, in the Modified Seidel representation, a near correct answer seems to usually come from the eigenvector that corresponds to the largest eigenvalue. Together, these results give adequate evidence for the examination of all eigenvectors.

Moreover, the discovery of interactions between the eigenvectors themselves should lead to better algorithms for solving or approximating NP–Complete graph problems. In addition, it is hoped that this work will help lead to a unifying theorem for spectral representations in graph theory by identifying patterns between the solutions represented by the eigenvectors of particular representations. It is hypothesized that the information in these patterns will also lead to new ways of combining eigenvectors into better solutions. A description of an exploratory empirical research project and several theorems that guarantee minimal bisections are described in the next two Sections.

5.1 Empirical Results

A description of the traditional spectral graph bisection algorithm and extensive results are provided in this section. The algorithm's focus is to find a minimum bisection of a graph. It works by taking a particular adjacency matrix representation and computing a single partition based on each eigenvector. Each eigenvector creates a bisection of the graph by separating the eigenvector's components based on the magnitude of their values. The vertices corresponding to components that are above the median are placed into one partition. The rest of the vertices are placed in the other partition. If the eigenvector's components are not separated such that each partition contains the same number of vertices, a repair operation is performed that fixes the solution. For each representation, the number of eigenvectors and cuts processed is equal to the number of nodes in the graph.

The algorithms and experiments are not meant to be competitive with new lower time complexity graph bisection approximation algorithms. Instead, they are meant to provide insight into the entire eigenstructure of particular graph problems in order to obtain an overall better solution from spectral methods. The experiments indicate that the eigenvector solution structures seem to have the following properties.

1. They are determined by their representation type.

2. They are consistent with problem type.

3. Their solution search structures are independent of size.

4. Many representations exhibit an oscillation in solution quality when they are arranged by the magnitude of their corresponding eigenvalue.

Full Rank Algorithms

The following algorithm can work with any of the adjacency matrices listed in the previous chapter. However, this may be the first description of an algorithm that bisects a graph using the Modified Seidel matrices $\overline{S}$ and $\overline{S}_{(c)}$. In addition, other adjacency matrix representations have only been mentioned rarely for the bisection problem. The $0,1$ and Laplacian matrix representations are the most commonly studied adjacency matrix for this problem.

The traditional methods only use one eigenvector of the Laplacian matrix, whereas the following algorithm uses every vector, albeit in a simple way. There are already examples of new algorithms that use all of the eigenvectors and beat the old spectral bisection algorithms [5]. The authors in the study used multiple eigenvectors of the Laplacian and turned the graph partitioning problem into a vector partitioning problem.

INPUT (**Adjacency Matrix**)

OUTPUT Partition

1. *Compute* **all** of *the eigenvectors of the input matrix.*

2. **For each** *eigenvector, compute the median of its components and place vertex* i *in partition* A *if the* i*'th component of the eigenvector is less than or equal to the median. Otherwise, place vertex* i *in partition* B*.*

3. *If necessary, repair the partition to make the number of vertices equal by moving vertices from the bigger partition to the smaller partition until the number of nodes in each partition is equal. Start with nodes that are closer to the other partition in terms of their corresponding eigenvector's component.*

Note that the choice of which vertices go in which partition when the corresponding characteristic valuation is equal to the median is essentially arbitrary. A study by Alex Pothen, Horst D. Simon, Kan-Pu Liou describes several techniques for choosing which partition a vertex is placed in when its characteristic valuation is equal to the median [71].

Adjacency Matrix Choice

Every representation was tested to examine its solution properties. In the next section, on theoretical results, theorems are proved

that show that the algorithm will give an optimal minimal bisection given that the graph has a certain structure. More theoretical work is needed to determine which types of graphs are easily split with each representation. It was anticipated that different representations will work better for different graphs. The Seidel representation seems to handle symmetry extremely well. For example, the Modified Seidel representation $\overline{S}$ is extremely good for cutting caterpillars, which are difficult for standard graph bisection algorithms [16]. In fact, it empirically finds the optimum bisection for caterpillars, connected graphs with two big clusters, and connected graphs on four vertices. Its partition pictures are also very symmetrical. Furthermore, results indicate that the eigenvectors of the Modified Seidel representation $\overline{S}$ inhibit the oscillations of solution quality. These results indicate that this representation may have particularly useful properties that lead to better algorithms for using eigenvectors to minimally bisect graphs.

One explanation for Modified Seidel representation's efficiency in symmetric relations may simply be that, in a 0,1 adjacency matrix representation, zero's role is doubled. For example, in the regular adjacency matrix A, it is the convention that $A_{ii} = 0$. On the other hand, zero is already assigned to mean "not connected," and therefore it is burdened by a dual role in the representation. The Modified Seidel representation $\overline{S}$ assigns a zero to the information

about connectivity relations that do not matter anyway because there are no self loops in the input graphs. Note that self loops do not affect bisection sizes.

Another reason that the –1,0,+1 idea seems natural is, roughly, that the dot product between two rows is the number of matches minus the number of mismatches. Namely, if row_i tells about v_i's neighbors, and similarly row_j for v_j:

$$S_{ik} = \begin{cases} 1, & \text{if } v_k \text{ is adjacent to } v_i, k \neq i \\ 0, & \text{if k = i} \\ -1, & \text{if } v_k \text{ is not adjacent to } v_i, k \neq i \end{cases} \tag{5.1}$$

Then the dot product of row_i with row_j equals

$\#\{k : k \neq i, k \neq j, v_k$ has the same status w.r.t. $v_i, v_j\} -$
$\#\{k : k \neq i, k \neq j, v_k$ has different status w.r.t. $v_i, v_j\}$

If two rows or columns share a value, then the corresponding term in the dot product's sum is also positive. Likewise, if two rows or columns have an opposite value, the the corresponding term in the dot product's sum will be negative. Therefore, for the dot product to be non–negative, the two rows or columns must share at least $\frac{n}{2}$ values.

Also of note is that all of the results with the Seidel or Modified Seidel matrices apply to similar adjacency matrix representations selected from the following set as the eigenvectors do not change direction as the variable x varies.

$$\{(0, x, -x)|x \neq 0, \text{x if connected, –x otherwise}\}$$

Eigenvector Search and Updating

It was also discovered that an exploration around the eigenvector solution space can be performed by simply multiplying the diagonal elements of the Modified Seidel adjacency matrix $\overline{S}_{(c)}$ by a constant amount c. Note that this is not a linear operation. The change to the adjacency matrix's trace moves the spectrum because $trace(A) = \lambda_1 + \lambda_2 + \cdots + \lambda_n$. By varying this constant, better solutions are often obtained. Figure 5.3 shows that this is indeed the case for the random and geometric graphs that were tested. Theory should be developed that connects the geometric action of these transitions with their resulting spectral bisections.

Figure 5.4 shows some more performance results that were achieved by first performing spectral bisection and then using the Kernighan–Lin heuristic on the resulting solution. If an eigenvalue decomposition for a square matrix A has already been computed, and the entries along the diagonal of A are then multiplied by a

Graph Type	ALL Laplacian	2nd Laplacian	D,1,-1	2D,1,-1	1.25*D,1,-1
U500.05	**2**	48	33	14	41
U500.10	**37**	**37**	110	**37**	100
U500.20	283	298	**220**	283	225
U500.40	488	488	434	488	**422**
U1000.05	**1**	6	#	4	89
U1000.10	**74**	393	210	**74**	169
U1000.20	**298**	**298**	482	**298**	377
U1000.40	1062	1062	1396	1062	**924**
G500.005	75	149	69	76	**62**
G500.01	302	659	260	300	**259**
G500.02	752	835	**698**	752	718
G500.04	1991	2134	**1882**	1991	1890
G1000.0025	143	241	136	145	**131**
G1000.005	612	868	**540**	612	541
G1000.01	1669	1869	**1518**	1669	1533
G1000.02	3848	4059	**3681**	3848	3750

= Did not converge

Figure 5.3: Results for Bui's Graphs when multiplying the diagonal

Graph Type	2nd Laplacian+KL	ALL Laplacian+KL	-1,1,0 SVD+KL	0,1 SVD+KL
U500.05	29	**2**	20	14
U500.10	**26**	**26**	47	42
U500.20	224	182	**180**	182
U500.40	417	**412**	**412**	**412**
U1000.05	5	**1**	66	11
U1000.10	154	**54**	121	72
U1000.20	232	232	**223**	233
U1000.40	747	**737**	**737**	**737**
G500.005	79	63	**61**	63
G500.01	270	238	**235**	**235**
G500.02	670	**649**	**649**	650
G500.04	1823	1797	1777	**1776**
G1000.0025	149	**115**	123	119
G1000.005	561	493	481	**480**
G1000.01	1472	1428	1405	**1400**
G1000.02	3562	3481	**3454**	3462
Breg500.0	**0**	**0**	**0**	**0**
Breg500.12	**12**	**12**	**12**	**12**
Breg500.16	**16**	**16**	**16**	**16**
Breg500.20	**20**	**20**	**20**	**20**
Cat.352	**1**	**1**	**1**	**1**
Cat.702	**1**	**1**	**1**	**1**
Cat.1052	**1**	**1**	**1**	**1**
RCat.134	**1**	**1**	**1**	**1**
RCat.554	**1**	**1**	**1**	3
RCat.994	**1**	**1**	**1**	3
Grid100.10	14	**10**	**10**	**10**
Grid500.21	**21**	**21**	**21**	#
Grid1000.20	**20**	**20**	**20**	#
W-grid100.20	26	**20**	**20**	**20**
W-grid500.42	**42**	**42**	**42**	**42**
W-grid1000.40	44	**40**	**40**	54

Figure 5.4: Results for Bui's Graphs when running KL afterwards

constant factor, how can the new eigenvectors be computed based solely on the old information? Put another way: If the diagonal entries of a square matrix are all shifted by a constant amount, what happens to the eigenvectors? An explanation for answers to these questions with a formula for the updated vector's computation is described in the section of this chapter containing the theoretical results.

Cut Value Oscillation

Eigenvector solutions are compared by rank and solution quality for several graphs. However, path graphs have a particularly simple adjacency matrix structure in most representations. Furthermore, compositions of two paths by Cartesian product lead to theorems about the eigenvectors and eigenvalues of the resultant grid graph. Papadimitriou and Sideri showed that the bisection width problem for grid graphs is NP–Complete [69], and that it could solved in time $O(n^{5+2h})$, where h is the number of finite connected components of its complement with respect to the infinite grid. The operators described herein already have less time complexity than this result, and so both path and grid graphs were deemed to be a good candidates for this exploratory study. It is known that the eigenvalues of the path graph when represented by the $0, 1$ representation are

equal to the following equation [22], [13].

$$\lambda_i = 2\cos\frac{i\pi}{n+1} \quad (i = 1, \ldots, n)$$

In addition, it is also known that the coordinates of the normalized eigenvector belonging to λ_i are

$$\sqrt{\frac{2}{n+1}} sin \frac{ij\pi}{n+1} \quad (j = 1, \ldots, n)$$

The first surprising empirical result is that there appears to be an oscillation in solution quality that is correlated with the magnitude of the solution eigenvector's corresponding eigenvalue when using certain representations. In fact, the solutions obtained from eigenvectors in the Modified Seidel representation of a path graph form a list of partitions whose solution quality increases approximately linearly in one case. In the other case, the solution quality decreases linearly almost *exactly*. The results show that a regular pattern emerges where the cut size decreases by two for every other eigenvector when the eigenvalues are ordered in increasing order. Figures 5.5 and 5.6 show the value of the cut generated on the y–axis by the eigenvector that corresponds to the x'th largest eigenvalue for two different representations.

Furthermore, the oscillation's shape and structure is empirically shown to be independent of the size and representation of

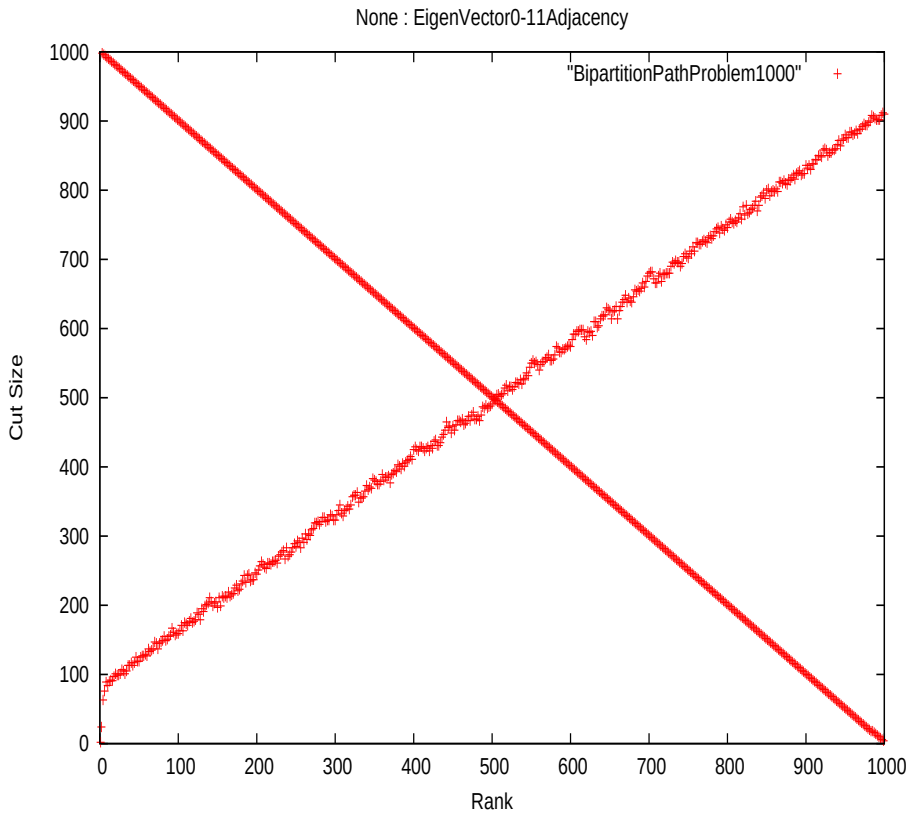

Figure 5.5: Oscillation in the eigenvector solution quality on a path graph of size 1000 in the Modified Seidel representation

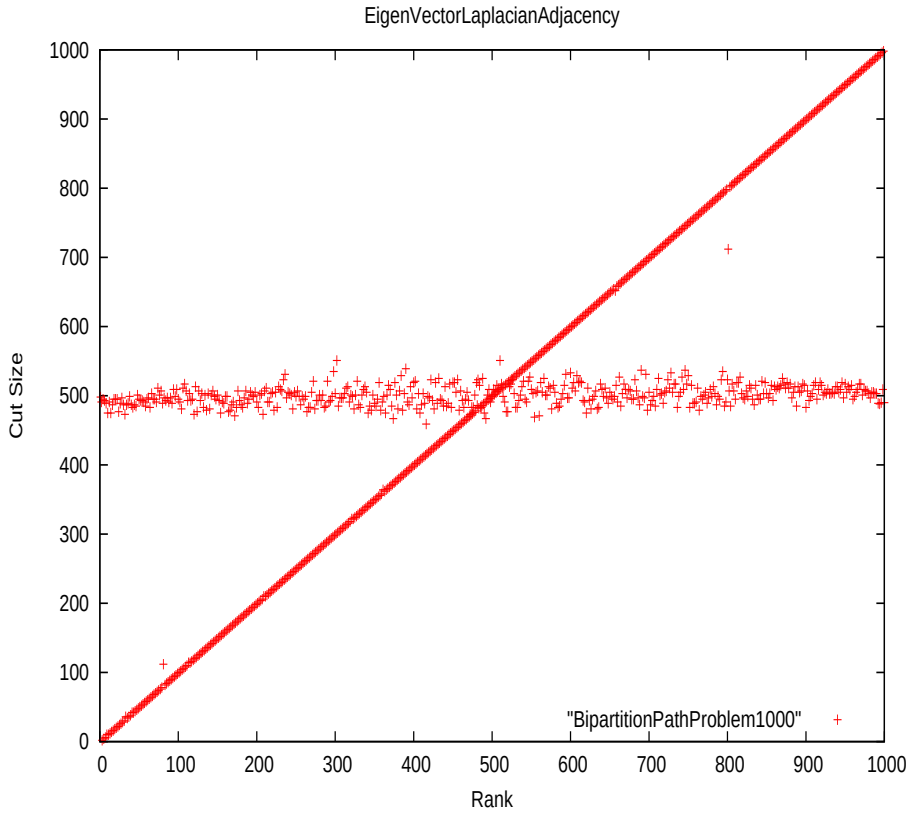

Figure 5.6: Oscillation in the eigenvector solution quality on a path graph of size 1000 in the Laplacian representation

the problem for random graphs in Figures 5.7 and 5.8. In these figures, lines are drawn between both the next largest and next smallest eigenvectors of a particular cut solution value. Figures 5.9 and 5.10 indicate that the cut size oscillation is not influenced by the internal parameters for Bui's B–regular graphs. In addition, these figures indicate that the oscillation does not occur when the graph is not connected, which is the case for the graph labeled "Breg500.0". Figure 5.11 is a plot of the cut sizes for B–regular graphs in the Laplacian representation.

Eigenvector Partition Search Pictures

The next surprising empirical result is that the partitions given by the eigenvectors form a fairly symmetrical partition search picture when the eigenvector solutions are ordered in terms of the magnitude of their corresponding eigenvalue. The partition search pictures are made in the following way. First, the partitions are sorted in terms of the magnitude of the eigenvectors that define them. Next, the algorithm paints values on the i'th row and j'th column of the image based on the magnitude of the j'th component of the i'th sorted eigenvector. Pixels are then assigned a gray scale color based on the magnitude of their corresponding eigenvector component. If the eigenvector's component value is closer to $-\infty$, it is assigned a darker color. Otherwise, the pixel is as-

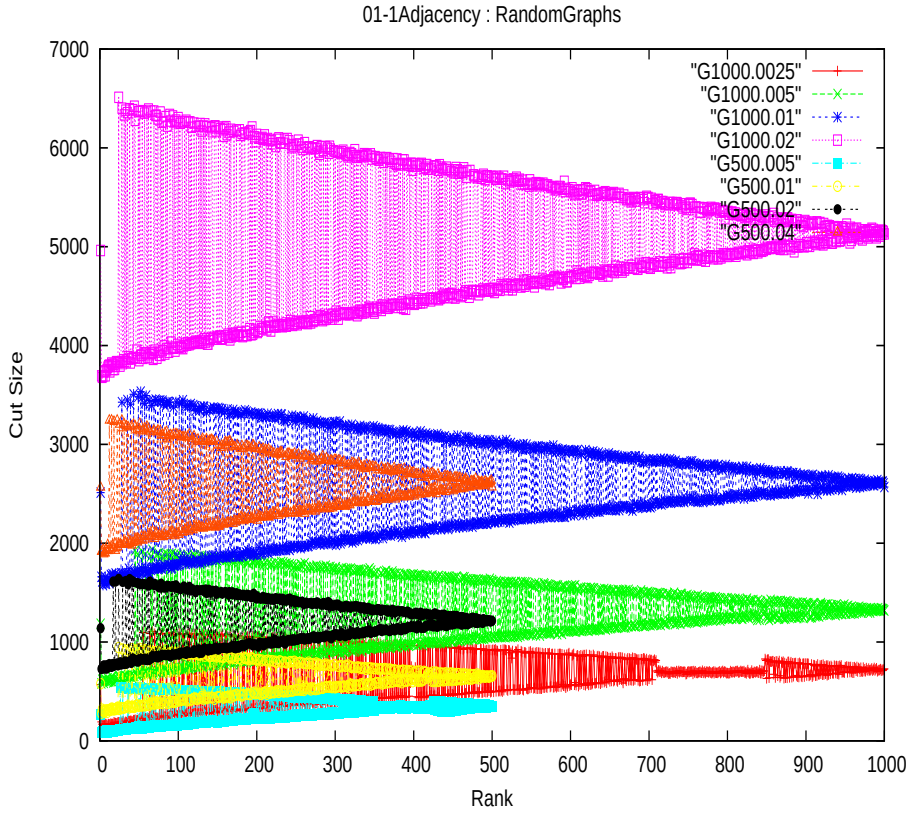

Figure 5.7: Oscillation in the singular vector solution quality on Random Graphs of various sizes in the Modified Seidel Representation. Notice the initial suppression of oscillation in this representation.

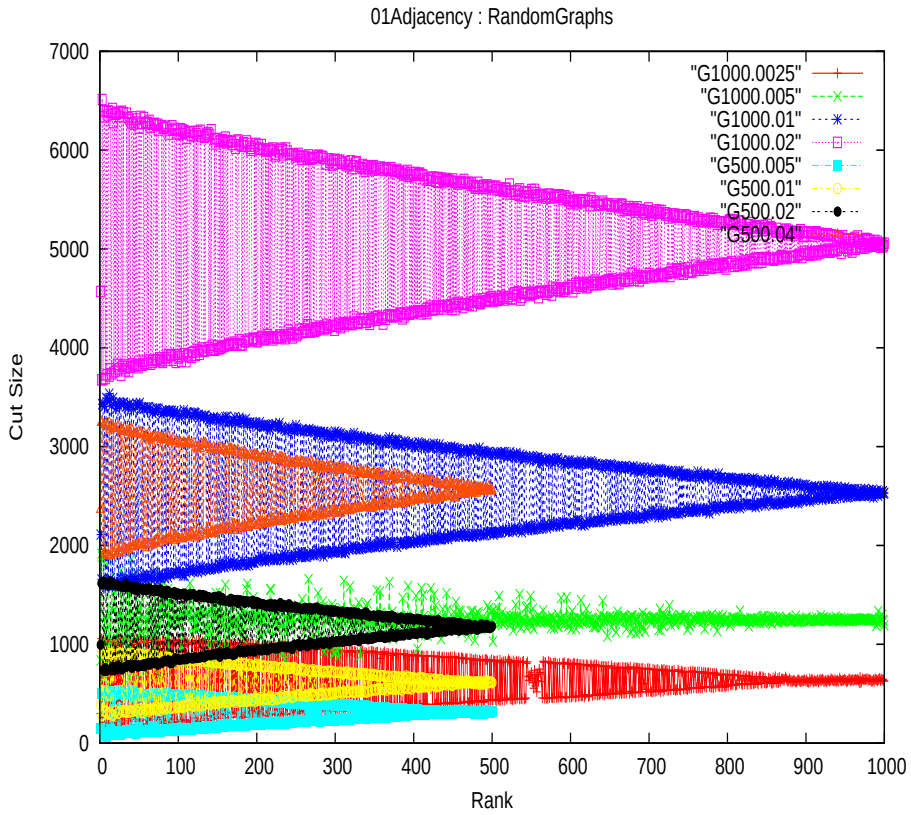

Figure 5.8: Oscillation in the singular vector solution quality on Random Graphs of various sizes in the 0,1 Adjacency Representation.

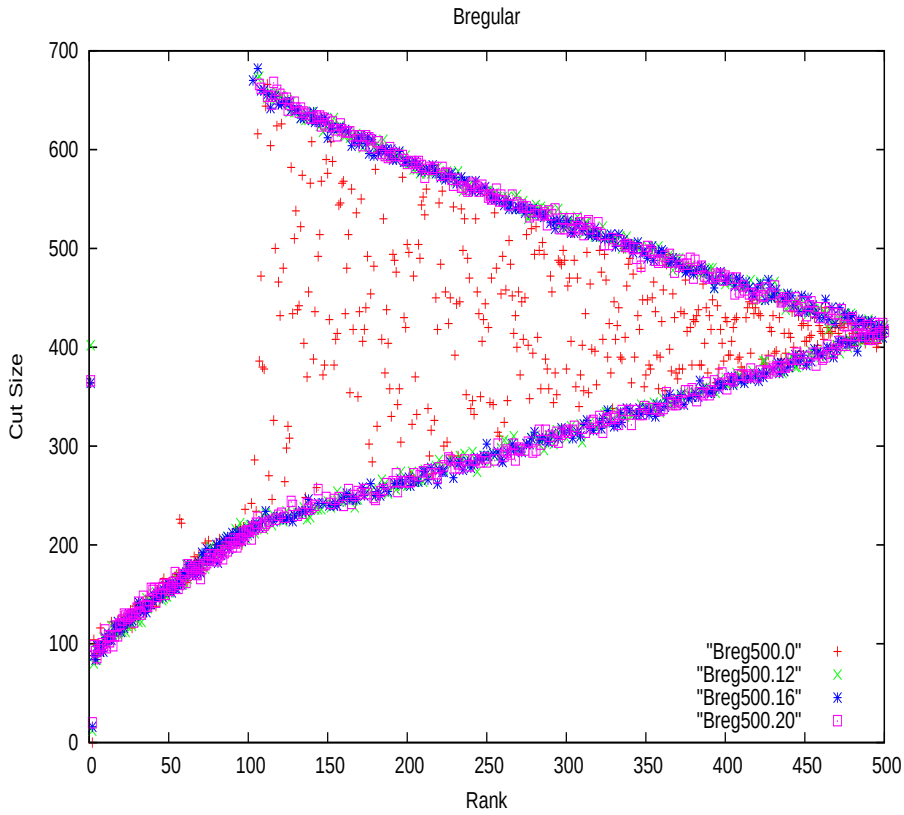

Figure 5.9: Oscillation in the singular vector solution quality on Bui's B–regular Graphs of various sizes in the Modified Seidel Representation. Notice that the cut size does not depend on the parameters *and* the initial suppression of oscillation in this representation.

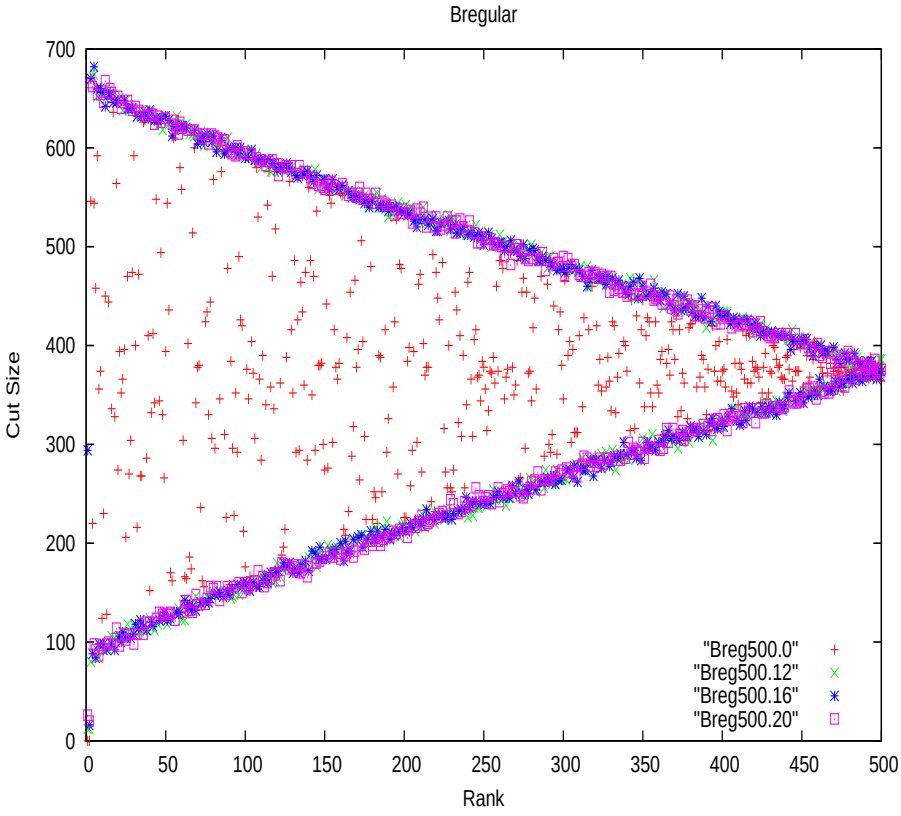

Figure 5.10: Oscillation in the singular vector solution quality on Bui's B–regular Graphs of various sizes in the 0,1 Adjacency Representation. Notice that the cut size does not depend on the parameters.

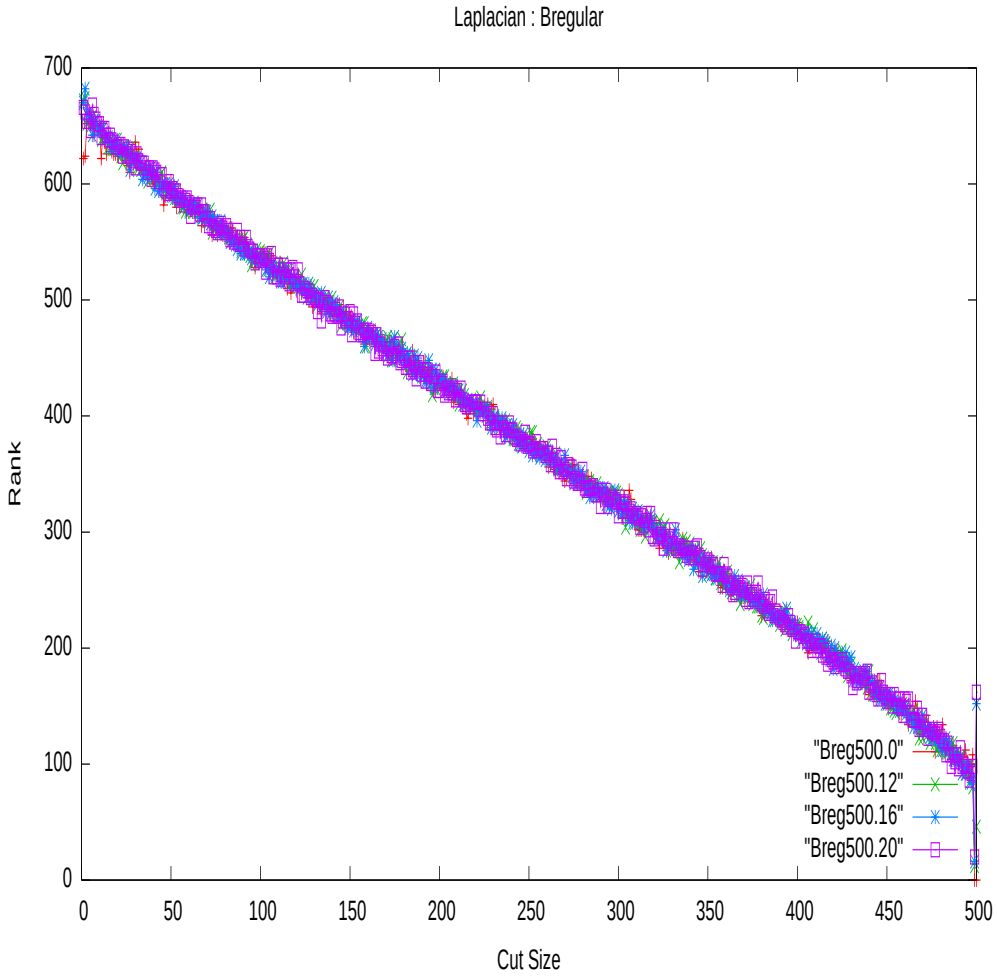

Figure 5.11: Oscillation in the singular vector solution quality on Bui's B–regular Graphs of various sizes in the Laplacian Adjacency Representation. Notice that the cut size does not depend on the parameters.

signed a color that is closer to white. A solution line to the natural consecutive vertex ordering of a path graph problem on 100 vertices would have pixels corresponding to vertices 1 through 50 painted in a darker shade color, and pixels corresponding to vertices 51 through 100 painted in a lighter shade. It is interesting to note that one of the eigenvectors in many of the representations provides a very close approximation to the minimum bisection for the graph bisection problem on path graphs. Therefore, something close to a solution line will appear in the corresponding eigenvector's partition picture. The pictures themselves simply correspond to the partitions that the entire eigenvalue decomposition gives for a particular adjacency matrix representation.

Each path graph's vertices are labeled in such a way that the vertices that define the path are connected increasingly.

$$1 \longleftrightarrow 2 \longleftrightarrow 3 \longleftrightarrow \cdots \longleftrightarrow n \tag{5.2}$$

It is important to note that the ordered labeling of the vertices was crucial to the discovery of pictures that exhibited a large amount of symmetry. It is quite easy to see that scrambling the vertex labeling corresponds to scrambling the columns of the picture into an order that is most likely not as symmetrical.

The figures that follow show similar structures in some cases. In particular, they all seem to share the same star in the middle. This may give some hope for the possibility of providing a formula that converts between the eigenvectors of different representations. In addition, the symmetry may provide evidence that algorithms may be found that compute good bisections based on the entire set of eigenvectors. One apparent feature from the results depicted in Figures 5.20 and 5.26 is that the singular vectors have a much different shape than one might predict considering that the singular vectors of symmetric matrices are supposed to be similar to the eigenvectors of the same matrix. The last example of these patterns is shown in Figure 5.27. This figure was produced from maple and depicts in 3D the values of each eigenvector's components, shaded according to their magnitudes. It is readily apparent that the eigenvectors of the path graph interact in a beautiful pattern. The subject of future research should be how these interactions can be used to help solve optimization problems for grids.

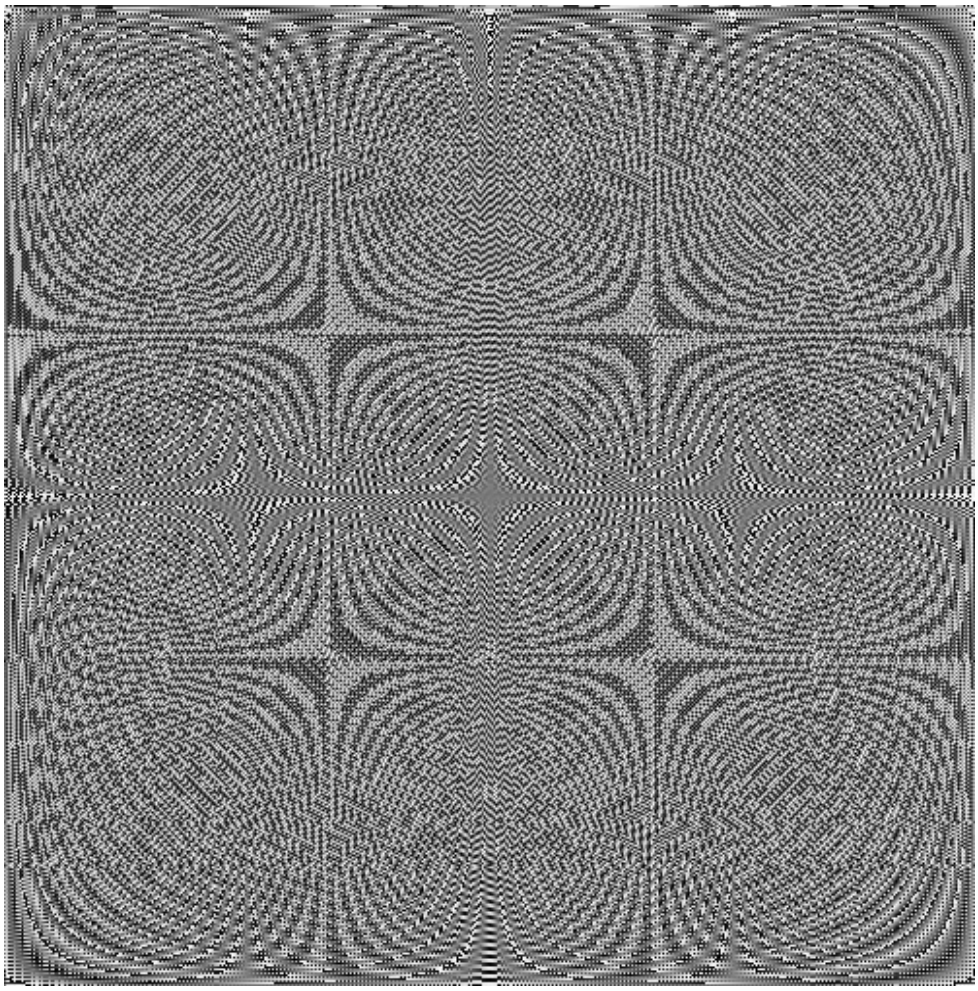

Figure 5.12: Shaded Partition Map for the Path Graph on 500 vertices (P_{500}) in the Modified Seidel Adjacency Representation

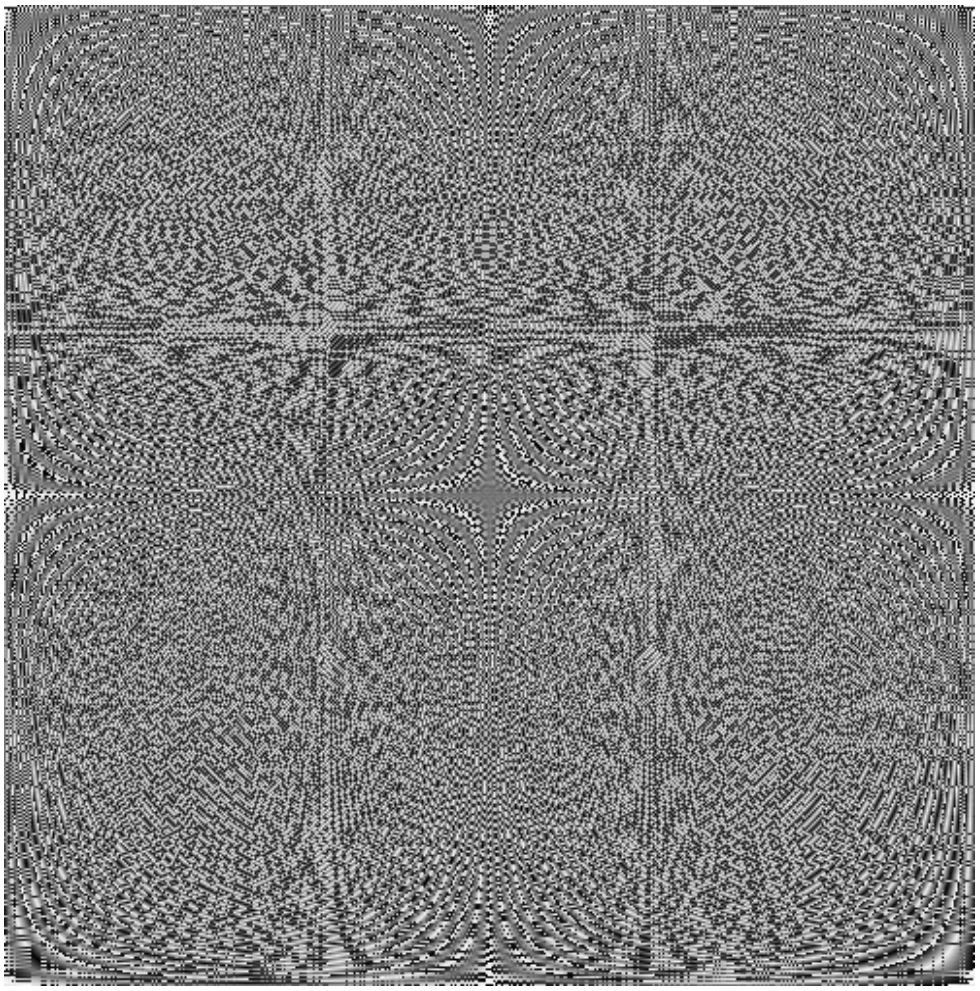

Figure 5.13: Shaded Partition Map for the Path Graph on 500 vertices (P_{500}) in the 0,1 Adjacency Representation

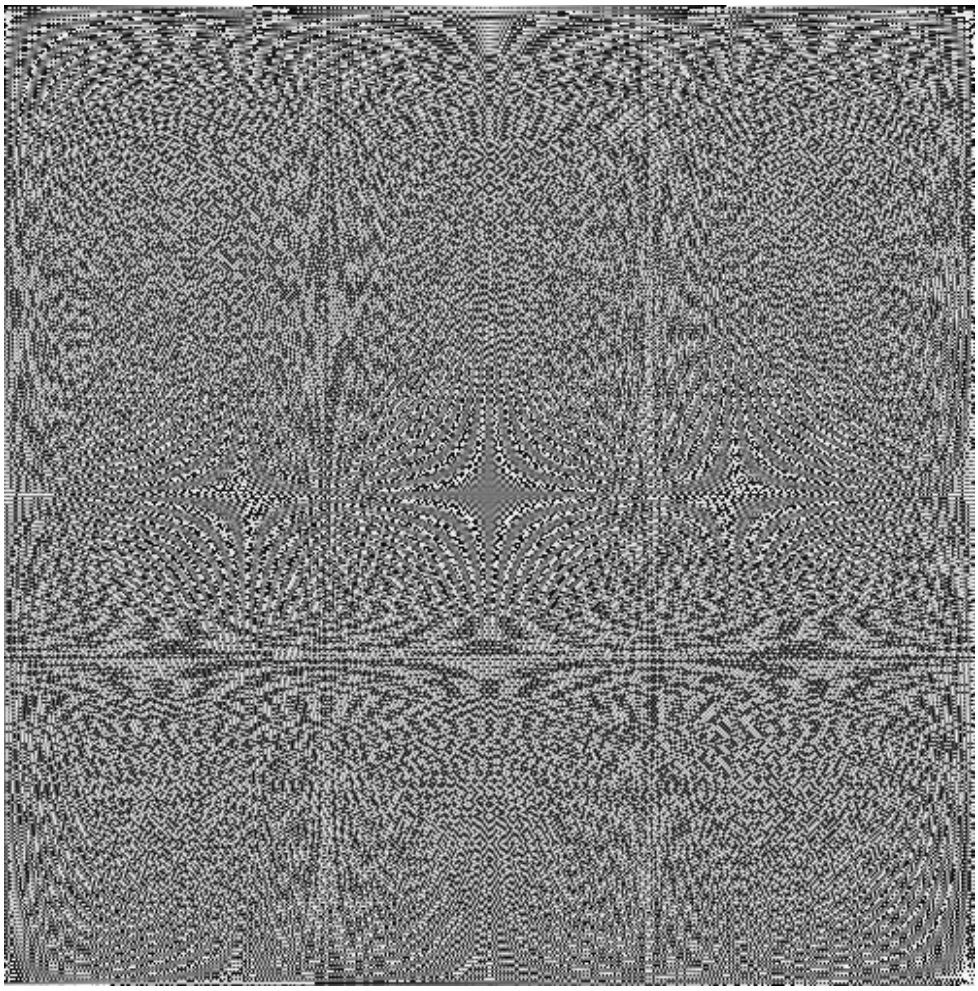

Figure 5.14: Shaded Partition Map for the Path Graph on 500 vertices (P_{500}) in the Negative Degree Laplacian Representation

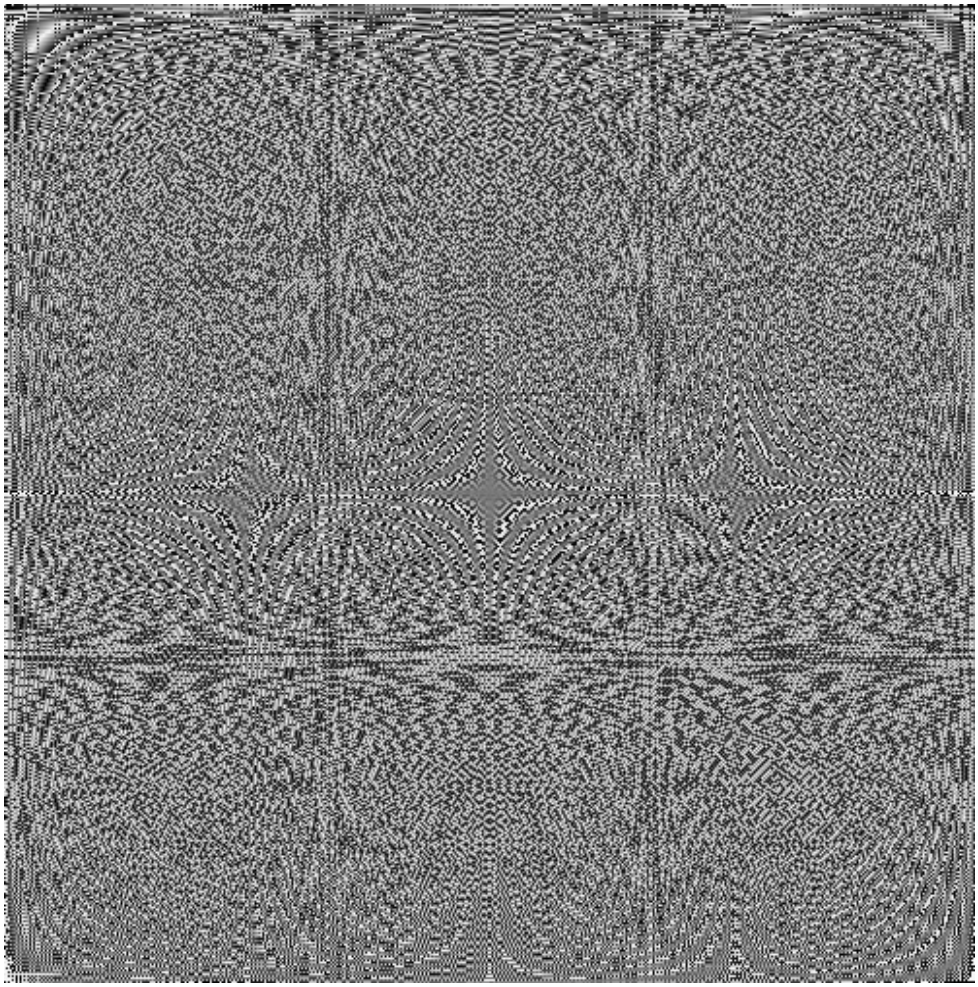

Figure 5.15: Shaded Partition Map for the Path Graph on 500 vertices (P_{500}) in the Laplacian Representation

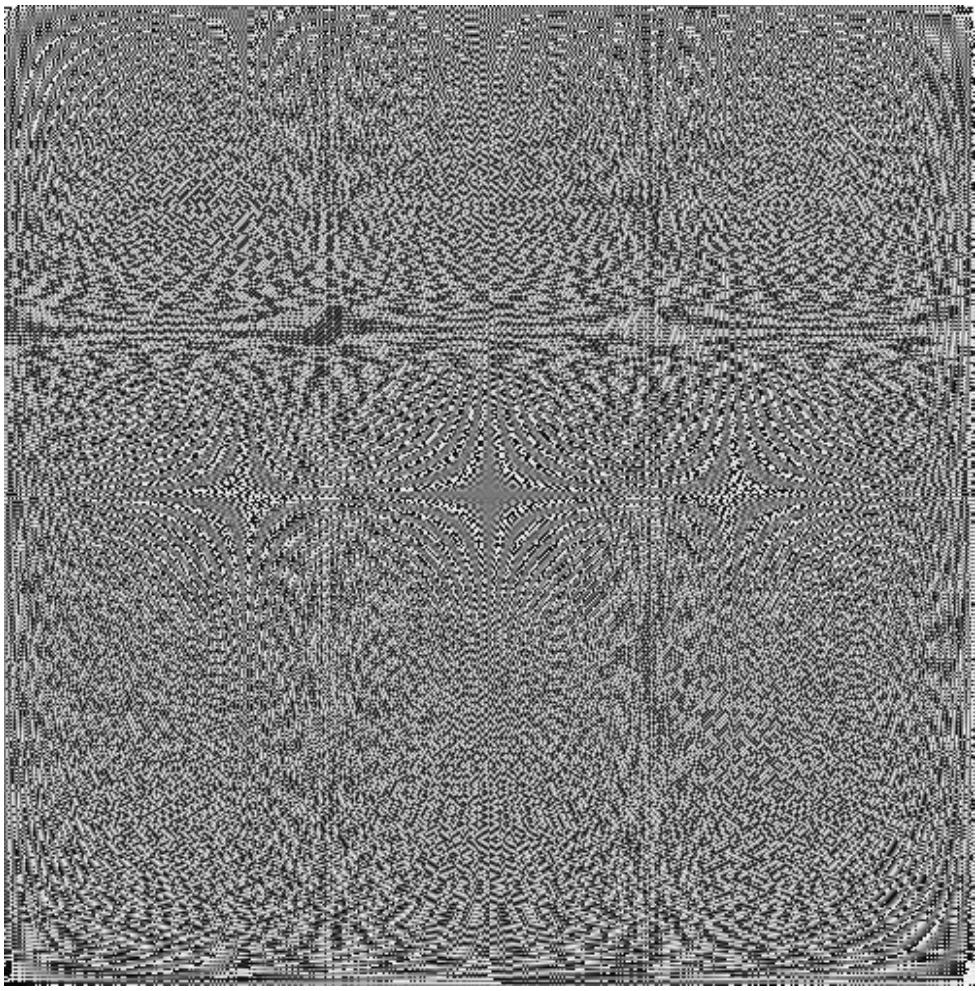

Figure 5.16: Shaded Partition Map for the Path Graph on 500 vertices (P_{500}) in the Signless Laplacian Representation

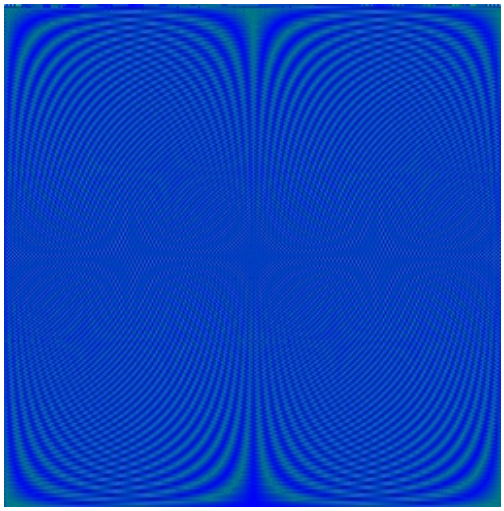

Figure 5.17: Blue shaded Partition Map for the Path Graph on 256 vertices (P_{256}) from eigenvectors of the Seidel –1,0,+1 Representation – Ordered by solution quality

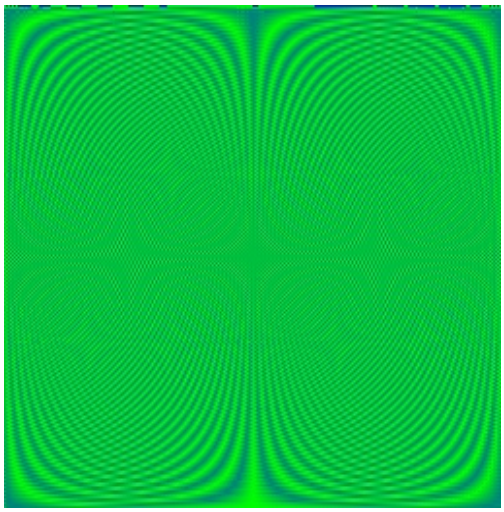

Figure 5.18: Green shaded Partition Map for the Path Graph on 256 vertices (P_{256}) from eigenvectors of the Seidel –1,0,+1 Representation – Ordered by solution quality

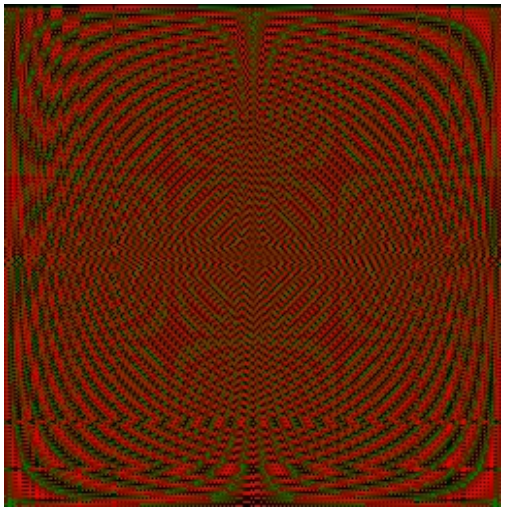

Figure 5.19: Red/Green shaded Partition Map for the Path Graph on 256 vertices (P_{256}) from eigenvectors of the Seidel –1,0,+1 Representation – Ordered by solution quality

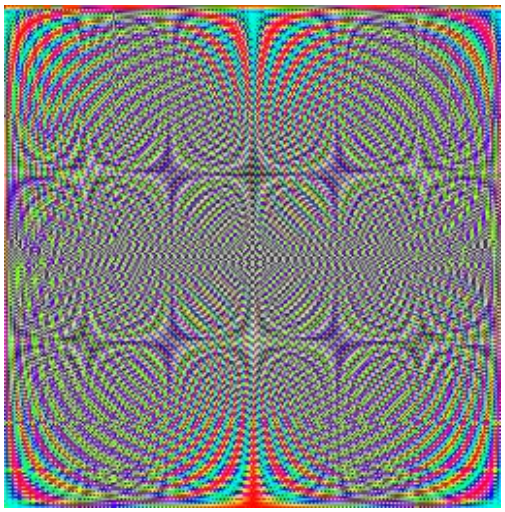

Figure 5.20: Hue shaded Partition Map for the Path Graph on 256 vertices (P_{256}) from eigenvectors of the Seidel –1,0,+1 Representation – Ordered by solution quality

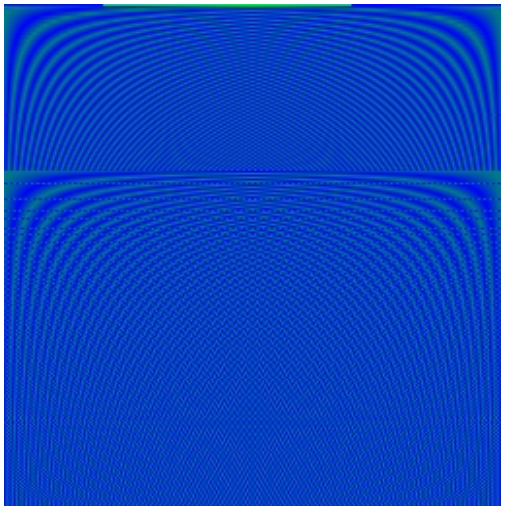

Figure 5.21: Blue shaded Partition Map for the Path Graph on 256 vertices (P_{256}) from singular vectors of the Seidel –1,0,+1 Representation

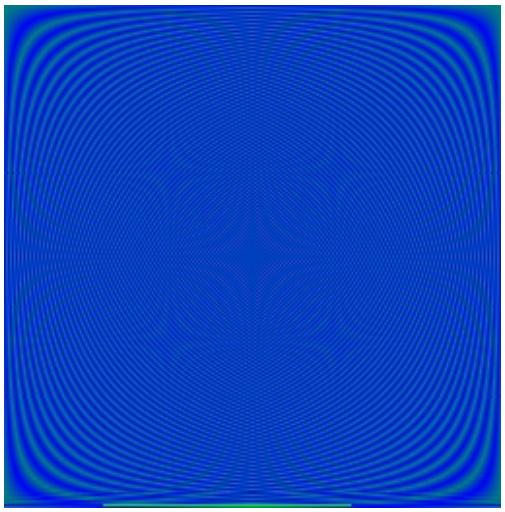

Figure 5.22: Blue shaded Partition Map for the Path Graph on 256 vertices (P_{256}) from singular vectors of the Seidel –1,0,+1 Representation – Ordered by solution quality

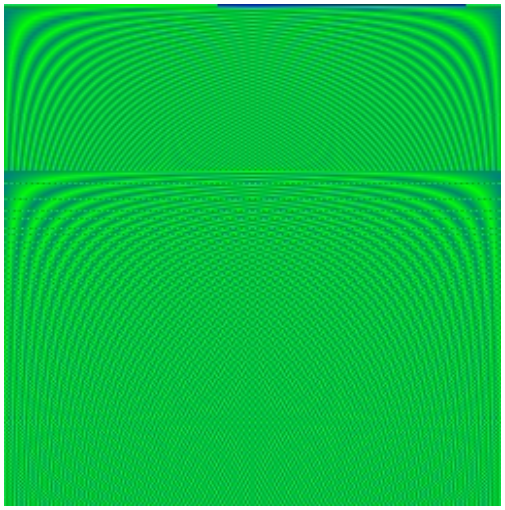

Figure 5.23: Green shaded Partition Map for the Path Graph on 256 vertices (P_{256}) from singular vectors of the Seidel –1,0,+1 Representation

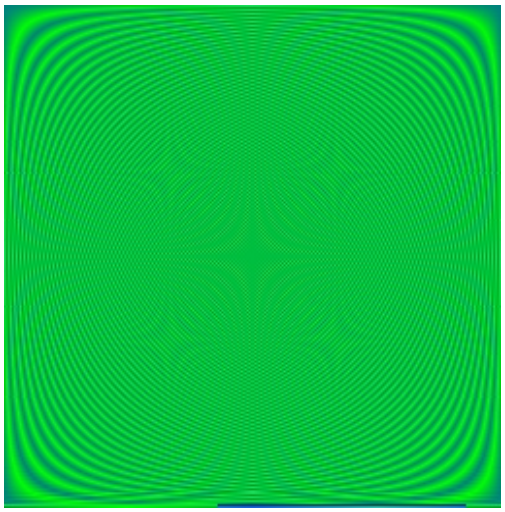

Figure 5.24: Green shaded Partition Map for the Path Graph on 256 vertices (P_{256}) from singular vectors of the Seidel –1,0,+1 Representation – Ordered by solution quality

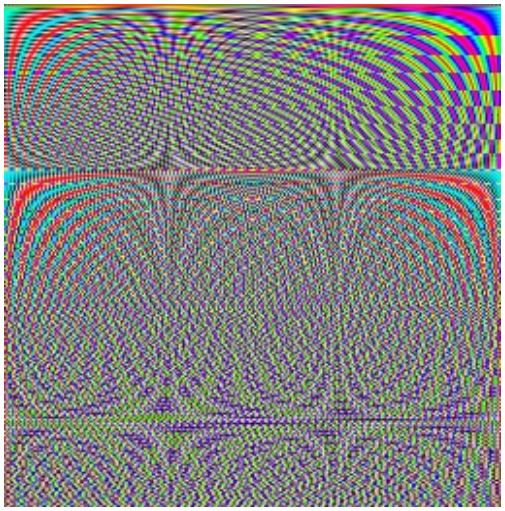

Figure 5.25: Hue shaded Partition Map for the Path Graph on 256 vertices (P_{256}) from singular vectors of the Seidel –1,0,+1 Representation

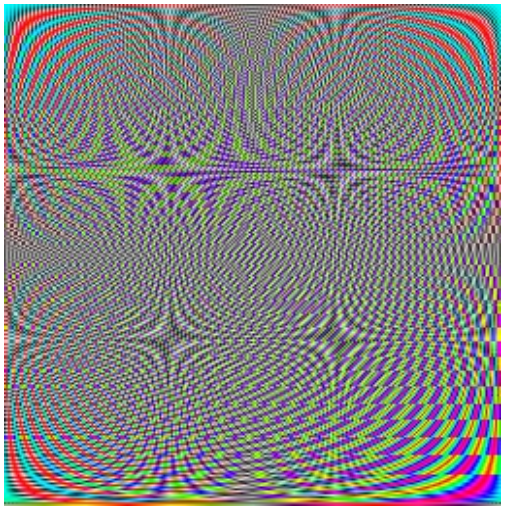

Figure 5.26: Hue shaded Partition Map for the Path Graph on 256 vertices (P_{256}) from singular vectors of the Seidel –1,0,+1 Representation – **Ordered by solution quality**

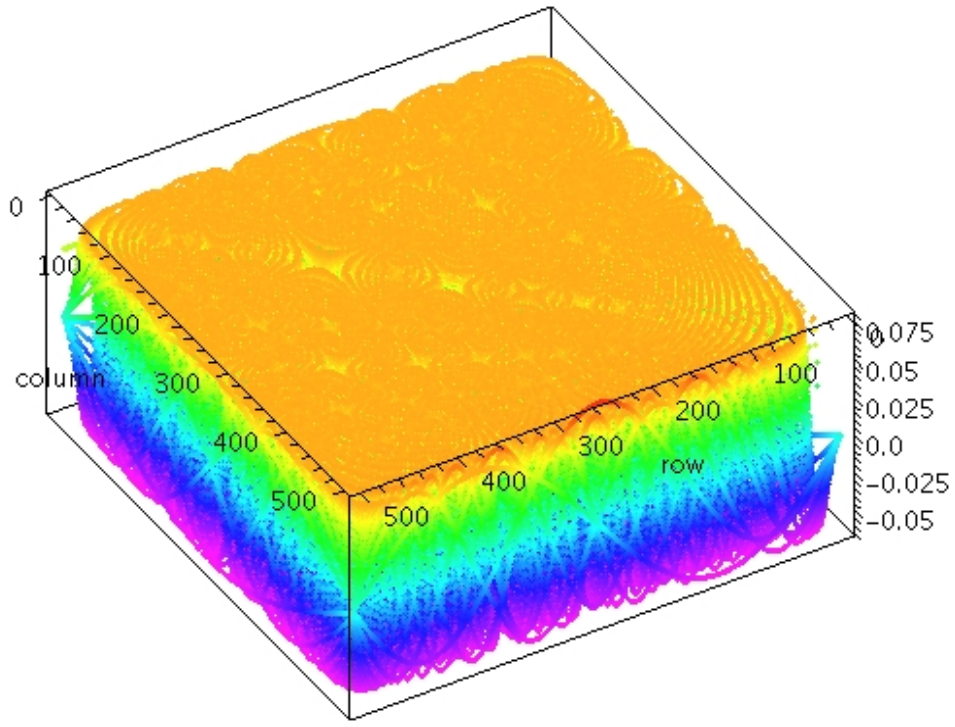

Figure 5.27: 3D Maple matrix plot of the eigenvectors of the Modified Seidel representation $\overline{S}$ of path graph on 500 vertices.

5.2 Theoretical Results

An Optimality Proof Strategy

The motivation for this section comes from a proof by Moler and Morrison that shows that the second singular vectors correctly partition a rank–2 digram frequency matrix into vowels and consonants [66, 74]. The ideas in Moler and Morrison's paper have already influenced an algorithm for graph coloring by Aspvall and Gilbert [7]. This section will adapt Moler and Morrison's method to create rules for obtaining a partition of a graph with the singular vectors that provably solve the Minimum Graph Bisection problem when the graph contains two main clusters.

Let A be the adjacency matrix of a graph $G = (V, E)$ on n vertices. A candidate solution to the Minimum Graph Bisection problem is given by two vectors l and r each containing $\frac{n}{2}$ ones. Let l denote the *left* partition, let r denote the *right* partition, and let

$$l_i = \begin{cases} 1, & \text{if the i'th vertex is in the left partition} \\ 0, & \text{otherwise} \end{cases} \tag{5.3}$$

$$r_i = \begin{cases} 1, & \text{if the i'th vertex is in the right partition} \\ 0, & \text{otherwise} \end{cases} \tag{5.4}$$

Each vertex is in either one partition or the other, but not both. Therefore, $(l + r)$ is a vector of all ones.

Definition 43 *An **intrapartition** edge is one with endpoints in the same partition.*

Definition 44 *An **interpartition** edge is one with endpoints in different partitions.*

It is easily seen that the total number of edges in the graph is

$$\frac{1}{2}(r^T Al + l^T Ar + r^T Ar + l^T Al) \tag{5.5}$$

and the number of interpartition edges is

$$\frac{1}{2}(r^T Al + l^T Ar) \tag{5.6}$$

Multiplication by $\frac{1}{2}$ ensures that edges between partitions, and edges within a partition, are only counted once.

In terms of A, l, r the minimum bisection problem may be expressed this way:

Find $0, 1$ vectors l and r, each containing $\frac{n}{2}$ 1's, that minimize Equation 5.6

As Theorems 45 and 46 show, the following inequality defines the solution space of the Minimum Graph Bisection problem when working on graphs with two main clusters

$$\text{interpartition edges} < \text{intrapartition edges} \tag{5.7}$$

Equation 5.7 can also be stated by quadratic forms with the vectors l and r. Since $l^T Ar = r^T Al$, the following simplification is a valid synopsis of the requirements of Equation 5.7..

$$l^T Ar < l^T Al \tag{5.8}$$

$$l^T Ar < r^T Ar \tag{5.9}$$

Collectively, the preceding equations (5.8 – 5.9) state that the number of edges between partitions should be less than the number of edges in each partition. Although this may or may not be relevant to the general Minimum Graph Bisection problem, it is relevant when working on graphs with two main clusters because partitions l and r that satisfy these equations also define the minimum bisection.

A slightly relaxed version of Equation 5.7 (in that 5.7 $\Rightarrow$ 5.10) is the following:

$$\frac{\text{inter edges}}{\text{endpoints in left partition}} < \frac{\text{right intra edges}}{\text{endpoints in right partition}} \quad (5.10)$$

Equivalently,

$$\frac{l^T A r}{l^T A (l+r)} < \frac{r^T A r}{r^T A (l+r)} \quad (5.11)$$

Inequality 5.11 says that the ratio of interpartition edges to the number of edges with an endpoint in the left partition is less than the ratio of the number of edges within the right partition to the number of edges with an endpoint in the right partition.

After cross multiplying and expanding, Inequality 5.11 becomes

$$(l^T A r)(r^T A l) + (l^T A r)(r^T A r) - (r^T A r)(l^T A l) - (r^T A r)(l^T A r) < 0 \quad (5.12)$$

The numbers in the parenthesis in inequality 5.12 commute because they are integers. The second and fourth terms cancel giving

$$(l^T A r)(r^T A l) - (r^T A r)(l^T A l) < 0 \quad (5.13)$$

Inequality 5.13 says that the square of the number of interpartition edges is less than the square of the number of edges within

the two partitions.

$$inter^2 < 4intra \tag{5.14}$$

When vectors l and r have been chosen to make this true, l and r represent the minimum bisection of certain graphs. A subclass of these graphs are discovered next.

Theorem 45 *Consider a disconnected graph on $2m$ vertices consisting of two complete subgraphs of size m, K_{m1} and K_{m2}. Let l be the number of vertices from each complete subgraph that are in the opposite partition from the rest of the $m - l$ edges in that subgraph. Also, let $m \geq 2$.*

Then, when $l = 0$ or $l = m$,

$$inter^2 - 4intra < 0 \tag{5.15}$$

otherwise, for $1 \leq l \leq m - 1$,

$$inter^2 - 4intra > 0 \tag{5.16}$$

In effect, the solutions to the Minimum Graph Bisection problem for these graphs are those that make the above equation negative. This occurs exactly when the vertices in a complete graph are all contained within one partition. Moreover, non-solutions to the Minimum Graph Bisection problem make the above equation positive.

Proof: Notice that

$$\begin{aligned}
\text{inter} &= 2l(m-l) \\
\text{intra} &= 2\left[\binom{m-l}{2} + \binom{l}{2}\right] \\
&= (m-l-l)(m-l) + l(l-1) \\
&= (m-l)^2 + l^2 - m
\end{aligned}$$

And so,

$$inter^2 - 4intra = 4\left[l^2(m-l)^2 - ((m-l)^2 + l^2 - m))\right] \tag{5.17}$$

Assume $l = 0$ or $l = m$. Then

$$4\left[l^2(m-l)^2 - ((m-l)^2 + l^2 - m))\right] = -4(m^2 - m)$$

The preceding equation is negative because $m^2 - m > 0$ for all $m \geq 2$. Therefore, when $l = 0$ or $l = m$, the equation is negative as required by the theorem.

Now let $1 \leq l \leq m-1$.

$$\begin{aligned} inter^2 - 4intra > 0 &\iff 4\left[l^2(m-l)^2 - ((m-l)^2 + l^2 - m))\right] > 0 \\ &\iff l^2(m-l)^2 > (m-l)^2 + l^2 - m \\ &\iff l^2(m-l)^2 - (m-l)^2 - l^2 > -m \\ &\iff (l^2-1)((m-l)^2-1) - 1 > -m \end{aligned}$$

This equation is always true when $1 \leq l \leq m-1$ because these values for l make both terms on the left hand side bigger than zero. Therefore, their product is positive, and bigger than $-m$ since $m \geq 2$. ■

Theorem 46 *If k interpartition edges are added between the two cliques in Theorem 45, then the relations in the theorem still hold for all k such that*

$$k < \sqrt{m^2 - m}$$

Proof: Let $inter$ and $intra$ correspond to only the edges originating from K_{m1} and K_{m2}. The goal is to determine for what values of k is

$$(inter + k)^2 - 4intra < 0 \text{ when } l = 0 \text{ or } l = m$$

and

$$(inter + k)^2 - 4intra > 0 \text{ when } 1 \leq l \leq m-1$$

The first equation is true because when $l = 0$ or $l = m$,

$$(inter + k)^2 < 4intra \iff (2l(m-l) + k)^2 < (m-l)^2 + l^2 - m$$
$$\iff k^2 < m^2 - m$$

For the rest of the cases, when $1 \leq l \leq m-1$, the equation $(inter + k)^2 - 4intra$ is made positive because as k gets larger, the equation gets larger. Therefore, the only split that makes the equation negative for graphs containing two complete subgraphs with $0 \leq k < \sqrt{m^2 - m}$ additional edges added between them is the one that places each complete subgraph in its own partition. ■

Additional results to the previous theorem can be achieved by subtracting some number of edges from each complete graph. The union of these types of graphs will represent the graphs that Theorem 47 will provably solve.

Let u_{ij} and v_{ij} be the i'th component of the j'th left and right singular vectors of the symmetric matrix A. Theorem 24 shows that the left and right singular vectors can be taken to be equal for symmetric matrices. As the proof of Theorem 47 shows, the following vertex assignment rules tend to categorize the vertices

into partitions so that Equation 5.13 is satisfied.

$$l_i = \begin{cases} 1, & \text{if } u_{i2} \geq 0 \text{ and } \ v_{i2} \geq 0 \\ 0, & \text{otherwise} \end{cases} \tag{5.18}$$

$$r_i = \begin{cases} 1, & \text{if } u_{i2} < 0 \text{ and } \ v_{i2} < 0 \\ 0, & \text{otherwise} \end{cases} \tag{5.19}$$

Although $l + r$ may not be a vector of all ones for all graphs as a valid partitioning would require, for certain graphs it will produce equal sized partitions when using certain singular vectors. One justification for this behavior is that if a graph contains two main clusters, then A_G is a matrix with two main blocks and is therefore approximately rank–2. Therefore, the rank–2 singular vector's matrix, in the partial sum (see Equation 2.24), must subtract off the strictly positive entries (because of the Perron–Frobenius theory stated in Theorem 27) of the rank–1 matrix in order to make the matrix become closer to the real adjacency matrix (which by assumption had rank ≈ 2).

Theorem 47 *Let A=A_2 be a non–negative rank–2 matrix with the SVD expansion*

$$A_2 = \sigma_1 u_1 v_1^T + \sigma_2 u_2 v_2^T \tag{5.20}$$

Let l and r be defined as in Equation 5.18 and Equation 5.19 respectively. Then inequality 5.13 is satisfied.

Proof: Let

$$L_i = \sigma_i l^T u_i v_i^T l \quad L_i^{'} = \sigma_i l^T u_i v_i^T r \tag{5.21}$$

$$R_i = \sigma_i r^T u_i v_i^T r \quad R_i^{'} = \sigma_i r^T u_i v_i^T l \tag{5.22}$$

Substituting A_2 into Equation 5.13 and expanding produces

$$(L_1^{'} + L_2^{'})(R_1^{'} + R_2^{'}) - (R_1 + R_2)(L_1 + L_2) < 0 \tag{5.23}$$

Expansion gives

$$L_1^{'}R_1^{'} + L_1^{'}R_2^{'} + L_2^{'}R_1^{'} + L_2^{'}R_2^{'} - R_1L_1 - R_1L_2 - R_2L_1 - R_2L_2 < 0 \tag{5.24}$$

All of the terms of the form $L_iR_i - L_i^{'}R_i^{'}$ cancel because

$$\begin{aligned} L_iR_i &= (\sigma_i l^T u_i v_i^T l)(\sigma_i r^T u_i v_i^T r) \\ &= (\sigma_i l^T u_i v_i^T r)(\sigma_i r^T u_i v_i^T l) \\ &= L_i^{'}R_i^{'} \end{aligned}$$

After eliminating terms of the form $L_i R_i - L_i^{'} R_i^{'}$ in Equation 5.23, the following is obtained:

$$L_1^{'} R_2^{'} + L_2^{'} R_1^{'} - R_1 L_2 - R_2 L_1 < 0 \tag{5.25}$$

Equivalently,

$$(\sigma_1 l^T u_1 v_1^T r)(\sigma_2 r^T u_2 v_2^T l) + (\sigma_2 l^T u_2 v_2^T r)(\sigma_1 r^T u_1 v_1^T l) -$$

$$(\sigma_1 r^T u_1 v_1^T r)(\sigma_2 l^T u_2 v_2^T l) - (\sigma_2 r^T u_2 v_2^T r)(\sigma_1 l^T u_1 v_1^T l) < 0 \tag{5.26}$$

How can the equation above be made negative by a suitable partitioning of vertices? Since A is a non–negative matrix, it follows from the Perron–Frobenius theorem [85, 37] and Theorem 27 that u_1 and v_1 have non–negative components. Furthermore, the singular values are always non–negative. Therefore, of all the different inner products appearing in Equation 5.26, only the ones with subscripts corresponding to the second singular vectors can be negative. The goal is to find a partitioning that keeps the equation negative. Given the the partitioning choices of vertices according to Equations 5.18 and 5.19, the only terms that are negative are $r^T u_2$ and $v_2^T r$. Substituting these values into Equation 5.26 forces the first four terms in parentheses to be negative. Furthermore,

the last four terms in parentheses are made positive. Therefore, Equation 5.13 is negative as required. ■

Corollary 48 *As in Theorem 47, let A=A_2 be a non–negative rank 2 matrix with the SVD expansion*

$$A_2 = \sigma_1 u_1 v_1^T + \sigma_2 u_2 v_2^T \tag{5.27}$$

Then the following partition also makes Equation 5.13 negative.

$$l_i = \begin{cases} 1, & \text{if } u_{i2} < 0 \text{ and } \ v_{i2} < 0 \\ 0, & \text{otherwise} \end{cases} \tag{5.28}$$

$$r_i = \begin{cases} 1, & \text{if } u_{i2} \geq 0 \text{ and } \ v_{i2} \geq 0 \\ 0, & \text{otherwise} \end{cases} \tag{5.29}$$

Proof: As in the proof of Theorem 47, substituting A_2 into Equation 5.13 and simplifying gives

$$(\sigma_1 l^T u_1 v_1^T r)(\sigma_2 r^T u_2 v_2^T l) + (\sigma_2 l^T u_2 v_2^T r)(\sigma_1 r^T u_1 v_1^T l) -$$

$$(\sigma_1 r^T u_1 v_1^T r)(\sigma_2 l^T u_2 v_2^T l) - (\sigma_2 r^T u_2 v_2^T r)(\sigma_1 l^T u_1 v_1^T l) < 0$$

Given the the partitioning choices of vertices according to Equations 5.28 and 5.29, the only terms that are negative are $l^T u_2$ and $v_2^T l$. Substituting these values into Equation 5.26 forces the first four terms in parentheses to be negative. Furthermore, the last four terms in parentheses are made positive. Therefore, inequality 5.13 is negative as required. ■

Of course, the adjacency matrix of a graph does not always have a rank of 2. However, the signs of the second singular vectors still tend to approximate the best bisection of graphs whose 0,1 adjacency matrices have a higher rank. This can be justified by the fact that the structure of matrices that have two main clusters are nearly block diagonal. Therefore, the spectral guarantees of Chapter 3 are applicable in that if a graph has two main clusters (topics), then both clusters will be well represented by a rank–2 approximation.

Updating Eigenvectors

This section describes an algorithm for updating eigenvectors. The algorithm's complexity is $O(n^2)$, but it only provides an approximation. The algorithm is very naive because it does not employ any of the typically used optimizations when computing eigenvectors of

symmetric matrices. In addition, partial sums are computed more than is required.

Let x be any one of the eigenvectors of the square matrix A with corresponding eigenvalue λ.

$$Ax = \lambda x \tag{5.30}$$

Let $A^{'}$ be the $n \times n$ matrix formed by multiplying all of A's diagonal entries by a constant value c. Note that this is not a linear operation. Let a_{ij} be the (i, j)'th entry of $A^{'}$.

$$A^{'} = \begin{bmatrix} ca_{11} & a_{12} & \cdots & \cdots & a_{1n} \\ a_{21} & ca_{22} & \cdots & \cdots & a_{2n} \\ a_{31} & a_{32} & ca_{33} & \cdots & a_{3n} \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn-1} & ca_{nn} \end{bmatrix}$$

The goal is to find the eigenvectors of $A^{'}$ using the eigenvectors of A. Let x_i be the i'th component of the eigenvector x of A. By the definition of eigenvalues and eigenvectors, (and assuming a full set

of eigenvectors exists):

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n &= \lambda x_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n &= \lambda x_2 \\ &\vdots \\ a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nn}x_n &= \lambda x_n \end{aligned}$$

The algorithms will be shown for the first component only, but it will be clear that the algorithm will work for any component.

$$a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = \lambda x_1 \tag{5.31}$$

The new component of the eigenvector after multiplying by c is what is required. In effect, what is $x_1^{'}$ in the following equation?

$$ca_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = \lambda^{'} x_1^{'} \tag{5.32}$$

To figure this out, multiply Equation 5.31 by c on both sides to get

$$ca_{11}x_1 + ca_{12}x_2 + \cdots + ca_{1n}x_n = c\lambda x_1$$

Now

$$ca_{11}x_1 = c\lambda x_1 - ca_{12}x_2 - \cdots - ca_{1n}x_n$$

To make everything look more like Equation 5.32, add what is needed to both sides:

$$ca_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = c\lambda x_1 - ca_{12}x_2 - \cdots - ca_{1n}x_n + a_{12}x_2 + \cdots + a_{1n}x_n$$

Simply factoring out λ from the right hand side to get $x_1^{'}$

$$ca_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = \lambda(cx_1 - \frac{1}{\lambda}(ca_{12}x_2 + \cdots + ca_{1n}x_n - a_{12}x_2 - \cdots - a_{1n}x_n)) \tag{5.33}$$

would not give us the new component of the eigenvector.

$$x_1^{'} = cx_1 - \frac{1}{\lambda}(ca_{12}x_2 + \cdots + ca_{1n}x_n - a_{12}x_2 - \cdots - a_{1n}x_n) \tag{5.34}$$

This *would* be true if the new eigenvalue of $A^{'}$ was the same as the old eigenvalue of A.

If new eigenvalue of $A^{'}$ (call it λ_{new}) corresponding to the first new eigenvector was known, then we could discover the new first component of the new eigenvector by factoring λ_{new} out of the right hand side of the equation:

$$x_1^{'} = \frac{c\lambda x_1 - ca_{12}x_2 - \cdots - ca_{1n}x_n + a_{12}x_2 + \cdots + a_{1n}x_n}{\lambda_{new}} \tag{5.35}$$

Experimentally the algorithm seems to be *close*. However, if the exact eigenvectors of the initial matrix (instead of some approximation as most, or all, algorithms give) are known, then the algorithm would work. So, this method loses accuracy as the actual eigenvectors computed initially lose accuracy. Work is needed to experimentally examine the convergence quality of the initial eigenvectors chosen versus the accuracy of this algorithm in comparison to the eigenvector given by computing a new decomposition.

Chapter 6 - Genetic Algorithms

Several ways of using singular value decomposition (SVD), a linear algebra technique typically used for information retrieval, to decompose problems into subproblems are investigated in the genetic algorithm setting. Empirical evidence, concerning document comparison, indicates that using SVD results both in a savings in storage space and an improvement in information retrieval. Combining theoretical results and algorithms discovered by others, several problems are identified that the SVD can be used with to determine a substructure. Subproblems are discovered by projecting vectors representing the genes of highly fit individuals into a new low–dimensional space, obtained by truncating the SVD of a strategically chosen gene $\times$ individual matrix. Techniques are proposed and evaluated that use the subproblems identified by SVD to influence the evolution of the genetic algorithm. By restricting the locus of optimization to the genes of highly fit individuals, the performance of the genetic algorithm is improved. Performance is also improved by using SVD to genetically engineer individuals out of the subproblems. A new SVD schema reordering technique is also

proposed to take advantage of the ideas represented in the schema theorem.

Patterns identified in the theoretical results from earlier chapters are used as a basis for creating an artificial problem that serves as a benchmark for the types of problems that will benefit from this research. The genetic algorithm's subproblem determination performance on several formulations of the NP–Complete Minimum Graph Bisection problem are also presented, giving insight into the structural discovery abilities of SVD. Results from the application of this process to several problems indicated a significant improvement in the GA's performance. In addition, the subproblems are usually determined early in the optimization process. Accordingly, using the discovered subproblems to genetically *engineer* individuals yielded additional performance improvements. It is hoped that this research will be important to help further unify and generalize the types of problems to which SVD can be successfully applied in a GA.

6.1 Block Diagonal Forms

Analyses in this dissertation are typically concerned with matrices whose rows and columns can be permuted to form a block diagonal or near block diagonal matrix. By the definition of block diagonal, two column or row vectors with ones in different blocks

will never have the same component equal to one. Thus, the dot product of any two rows from separate blocks will be zero. Furthermore, the dot product between two columns from different blocks will be zero and the vectors will be perpendicular. If the two columns or rows are in the same block, the dot product will always be one because they are essentially the same vector and are thus obviously parallel. Due to the spectral clustering guarantees to be presented in Chapter 3, these column and row interactions will also hold true with high probability for reductions of block diagonal, or near block diagonal, matrices A down to A_k, with k equal to the number of blocks.

Other matrix representations also share intuitive dot products. Consider the set of block diagonal matrices that are modified to have entries of -1 in the off diagonal instead of zeroes. Dot products between column or row vectors in the same block of this representation will be closer to $+\infty$. Dot products between column or row vectors in different blocks will be closer to $-\infty$.

If some sequence of row and column interchanges can make a matrix be close to block diagonal, with k blocks, then each of the top k singular vectors given by the SVD will correspond to exactly one of the blocks of similarly used rows and columns. Sequences of row and column interchanges do not effect the bases produced when computing the SVD because a matrix's row and

column space remain the same after any sequence of interchanges. Therefore, if the matrix is not in a block diagonal order but some sequence of row and column interchanges can make it block diagonal, then SVD will still recover the blocks. Notice that swapping the columns into block diagonal order does not effect the rows that are used similarly across the columns.

$$\begin{bmatrix} \mathbf{1} & \mathbf{0} & \mathbf{1} & \mathbf{0} \\ \mathbf{1} & \mathbf{0} & \mathbf{1} & \mathbf{0} \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 \end{bmatrix} \xrightarrow{\text{(swap columns 2 and 3)}} \begin{bmatrix} \mathbf{1} & \mathbf{1} & \mathbf{0} & \mathbf{0} \\ \mathbf{1} & \mathbf{1} & \mathbf{0} & \mathbf{0} \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix}$$

This idea is important in the context of information retrieval and genetic algorithms because documents or individuals are typically not ordered in a block form. In the context of graph algorithms, the importance is that the naming of the vertices will not have an affect on the bases produced.

It should be noted that more general forms of matrices have been shown to benefit from clustering with reduced rank SVD [25]. Therefore, it is extremely likely that the SVD operators to be described in this Chapter will prove beneficial for many different problems and representations.

6.2 Implementation Details

Tests are performed using a custom GA, implemented entirely in Java^{TM}. The source code and documentation for the GA may be obtained by e–mailing the author. The SVD was computed using LAPACK routines and the Matrix Toolkits for Java^{TM} (MTJ).

6.3 Spectral Injection

The techniques of Chapter 4 are used to provide initial population seedings for the genetic algorithms. Initially, the SVD of the adjacency matrix of the graph to be bisected is computed. Next, partitions are created using the algorithm described on page 83. These spectrally found partitions are initially and periodically injected into a population in order to influence the GA towards the good partitions. Experiments with this method show that the spectral injection gives the GA a tremendous head start in comparison to not using it at all.

6.4 Local Improvements

Hybrid GAs are those that incorporate a local search procedure during each generation on the new offspring. Local searches are almost always problem specific and take a candidate solution to a

problem and improve it by exploring locally around the solution's values. Hybrid GAs are a hybridization of a genetic algorithm with a heuristic that is tailored specifically for solving a certain problem. Generally, the performance of the local improvement heuristic is compromised to allow for a lower time complexity when creating a hybrid GA. This ensures that the local improvement heuristic does not overwhelm the overall running time of the GA so that the GA will be able to process more generations in less time.

Kernighan–Lin

Several hybrid GAs are studied that use a trimmed down variant of the Kernighan–Lin [58] algorithm. The algorithm's time complexity is trimmed down in the exact way that is described in Bui and Moon's paper on graph partitioning with a GA [19].

Fiduccia–Mattheyses

Additionally, the data structures and implementation of the algorithm are done in constant time based on the methods of Fiduccia and Mattheyses [32]. Fiduccia and Mattheyses gave a simplification of the Kernighan–Lin heuristic that has time complexity $\Theta(E)$ [32]. These optimization algorithms perform a limited, low cost, local search when solving various graph bisection problems. Fig-

ure 6.35 depicts the results from an experiment that used both spectral injection and local improvements.

6.5 Eigenvectors of A^TA

SVD is used to expose the most striking similarities between a given individual and a strategically chosen population of individuals. These similarities are used to influence the direction of the GA's search process by qualifying candidate individuals for reinsertion into the next generation based on their proximity to other individuals, whose fitnesses have already been computed. Initial results from the application of this process indicate significant improvements in the GA's performance. The intent is to evaluate several different tested approaches of using SVD qualifiers to enhance the performance of GAs.

It has been shown experimentally and probabilistically that the SVD should be able to expose the most striking similarities between a given vector and another set of vectors [70]. These similarities are used to influence the direction of the GA's search process by qualifying candidate individuals for reinsertion into the next generation based on their proximity to other individuals. One benefit of this approach is that the fitness function need not be computed in order to determine that an individual closely resembles another individual whose fitness is already known. For problems that require a

computationally expensive fitness function, such as those found in engineering design optimization, this benefit could be significant.

Qualification

The qualification approach involves comparing the candidate to the worst individuals in the current population. The qualification process is initialized for each generation by first creating a matrix containing the individuals to qualify candidates against. This matrix is composed of individuals in the current population whose fitnesses are less than half of the current population's average fitness. Conceptually, the subspace spanned by this matrix outlines the qualification guidelines for the current generation. The qualification subspace is then reduced to k dimensions by computing the SVD and eliminating all but the k largest singular values. A readily apparent criticism of the qualification process is that computing the entire SVD at each generation, for large dimensionalities, may become computationally expensive. However, methods exist for both folding in new vectors and removing old vectors from an existing SVD computation[11].

Qualification for a candidate individual is based on its proximity to the individuals in the qualification space. In order to compute its proximity, a candidate is first converted into a vector, whose components represent the genes of the individual. The vector is then

converted into a unit vector and projected into the qualification space using the diagonal matrix D, which contains the k largest singular values along its diagonal. Assuming a good representation, similar individuals will be represented by nearly parallel vectors and dissimilar individuals by nearly orthogonal vectors. Thus, the concept of similarity is reduced to computing the cosines of the angles between the projected candidate vector and every other vector in the rank–k qualification space. The cosine of an angle is an appropriate function to use because its value approaches one as two vectors become more parallel. Likewise, as two vectors become more perpendicular, the cosine of the angle between them approaches zero. The cosines are then compared to the d–close parameter, which represents the required amount of proximity for qualification ($0 \leq d \leq 1$). If the candidate is at least d–close to any of the worst individuals in the current population, it is discarded. Otherwise, it is allowed a chance to survive to the next generation.

Transformation

The preceding discussion works under the assumption that similar individuals will be represented by nearly parallel vectors, and dissimilar ones by nearly perpendicular ones. This assumption illustrates the need for a good vector based model for each particular optimization problem. A good model would place good individuals

near other good individuals in the individual representation space and likewise for bad individuals. Sometimes, the actual parameters that represent the genes may not lend themselves to accurate categorization of an individual during comparison. For this reason, transformations are applied to each gene of each individual in an attempt to more effectively categorize their values. In continuous function optimization problems, each gene component of an individual is categorized into a section, based on the allowed range of each gene. For example, an individual with ten genes, with gene values varying between zero and ten, could be transformed into an individual with twenty genes. The transformed genes no longer represent the exact gene values. Rather, they represent the location in a partition of the allowed range of values for the gene. Under this example, the first two genes of the new representation indicate whether or not the individual's original gene before transformation is a high (greater than 5) or low (less than or equal to 5) value. The transformations have the effect of categorizing solutions together based on the high and low combinations of the genes. Here, the SVD is used to expose the successful, or unsuccessful, combinations of high and low values of parameters for a particular function. In comparison to the naive approach (with no transformation), the observed improvement in the qualification process is significant for some problems. However, for other problems, the

process was not as beneficial. Presumably, this is because the transformation did not accurately represent what it meant to be a good or bad individual for that particular problem.

Reduction

The amount of reduction performed on the gene–individual matrix, A, directly influences the accuracy achieved. Although the optimal amount of reduction performed is not known *a priori* for a given problem or situation, there is a technique that may provide a good guess. First of all, as shown in Chapter 3, if the "k largest singular values of a matrix A are well–separated from the remaining singular values, then the subspace spanned by the corresponding singular vectors is preserved well when a small perturbation is added to A"[70]. The amount of relative change produced by a given reduction to rank k is based on the k largest singular values. It is easily calculated by the following formula (where the subscript F denotes the Frobenius Norm).

$$percenterror = \frac{\|A - A_k\|_F}{\|A\|_F} * 100 \tag{6.1}$$

It is clear that the amount of perturbation strongly influences the percent error. The results presented in the next section are achieved with the policy of reducing to the lowest rank that causes

no more than 10 percent error. Importantly, this isn't a hard–fast rule, and may change depending on problem domain, representation, and stage of evolution. The results produced by varying the error margin (and therefore the rank chosen) at different stages using strategically chosen conditions should be a topic of study for future works.

Results

An approach similar to the $(\mu + \lambda)$ evolution strategy was used with populations of size 100 generating 100 candidate individuals. Rank–based selection was used to select individuals for breeding. The breeding process used one point crossover and a ten percent chance of mutation. Mutation was performed by first randomly choosing a gene and then increasing or decreasing it by a random amount between 0 and 1. Reinsertion was achieved by picking the best 100 individuals out of the 200 parents and qualified offspring. To handle situations where not enough children are qualified, the qualifier is disabled after 500 unsuccessful attempts at generating the 100 children. The breeding process then finishes by generating the required number of offspring without qualification. A more ideal solution would be to adjust the amount of reduction and d–close parameters based on some function of the qualification pressure.

All of the results are based on the average of the best individual, for each generation, over 30 different random initial populations. The fitness of an individual x is defined in this section as

$$fitness(x) = \frac{1}{1 + |f(x) - target|} \tag{6.2}$$

where $f(x)$ is the function being optimized and $target$ is the value that is desired for the function. For an in depth explanation of how to compute the rank–k cosines between a query vector and the vectors contained in a reduced rank–k model efficiently (see [11]). The GA was tested on a variety of problems, with varying degrees of difficulty and dimensionality, using three different approaches. In the first approach, no rank reduction was performed on the gene–individual matrix at all. The second approach attempted to determine the best amount of rank reduction automatically by analyzing the separation of the singular values in order to select a rank that would cause no more than 10 percent error. In order to compare the performance of the SVD GA to traditional GAs, the final approach did not incorporate the SVD qualification process at all. These tests indicate that the rank reduction cases, on average, outperformed the plain GA, and the no reduction case. The effects of the SVD qualification process are evaluated by testing three different optimization problems. Each problem's goal is to minimize a

function. Recall that a higher fitness value corresponds to a lower function value for each particular problem.

N–Queens. The first problem tested was the n–queens problem. The goal is to find a way to place n queens on an n x n chess board such that no two queens attack each other. This means that no two queens may be placed on the same row, column, or diagonal. For this reason, the representation chosen is a permutation of the set $\{1, \ldots, n\}$. This representation restricts the search to cases in which queens are on separate rows and columns. The function f of an individual x is computed by counting the number of diagonal conflicts in its representation. The fitness of the individual is then computed with equation 6.2, where $f(x)$ is the number of pairs of queens that are attacking each other in the solution x, and the target is zero. The results achieved when $n = 30$ are shown in 6.28. The results indicate that domains with permutation representations are amenable to the SVD qualification process without transformation. An added bonus is also incurred because the act of reducing the qualification matrix, thereby bringing out its latent characteristics, outperforms the no–reduction cases. The transformation step was skipped for this problem because the discrete nature of the domain make it doubtful that significant improvements could be achieved.

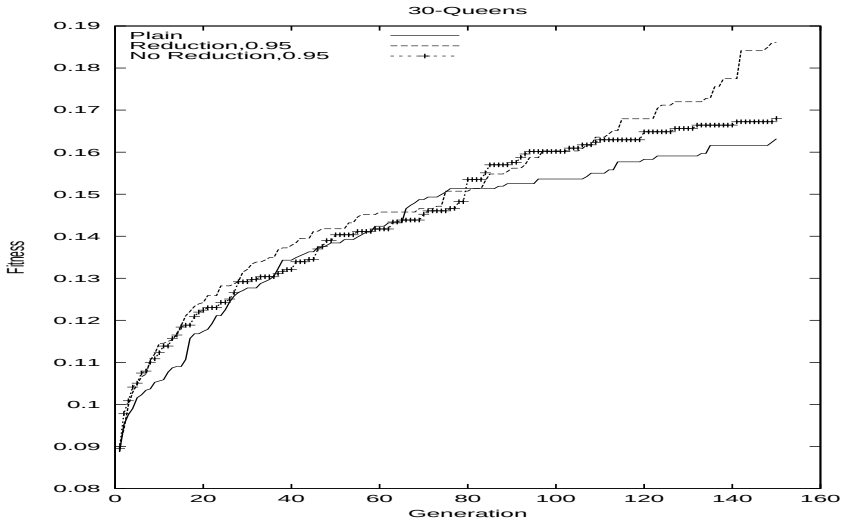

Figure 6.28: Results for the 30 Queens function

Sphere Model Minimization. The second problem was to minimize the sphere model function, given below.

$$f(x) = \sum_{i-1}^{30} (x_i - 1)^2, x_i \in [-5, 5] \tag{6.3}$$

The representation chosen for individuals in this problem was a vector of doubles. Transformations were applied by categorizing each of the thirty genes into one of four equal sized partitions of the interval $[-5, 5]$. From 6.29, it can be seen that the SVD GA outperformed the plain GA once again. However, the act of reduction and transformation did not provide significantly better results over the plain GA in this problem. Presumably this is because the trans-

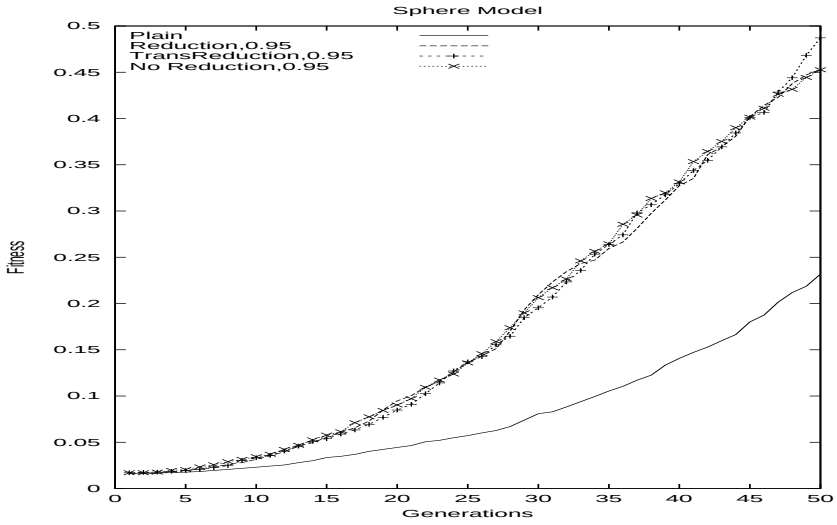

Figure 6.29: Results for the sphere model

formation chosen for comparison via the SVD did not adequately capture what it meant to be a good solution in this problem domain.

Langermann Function Minimization. The third minimization problem tested was a modified Langermann function.

$$f(x) = -\sum_{i=1}^{N} c(i)(e^{\frac{-1}{\pi}\|x-A(i)\|_f^2} cos(\pi \|x - A(i)\|_f^2) \tag{6.4}$$

$$x_i \in [0, 10] \tag{6.5}$$

The GA was tested on the 10 dimensional formulation of this function ($N = 10$). The global minimum for this function is ap-

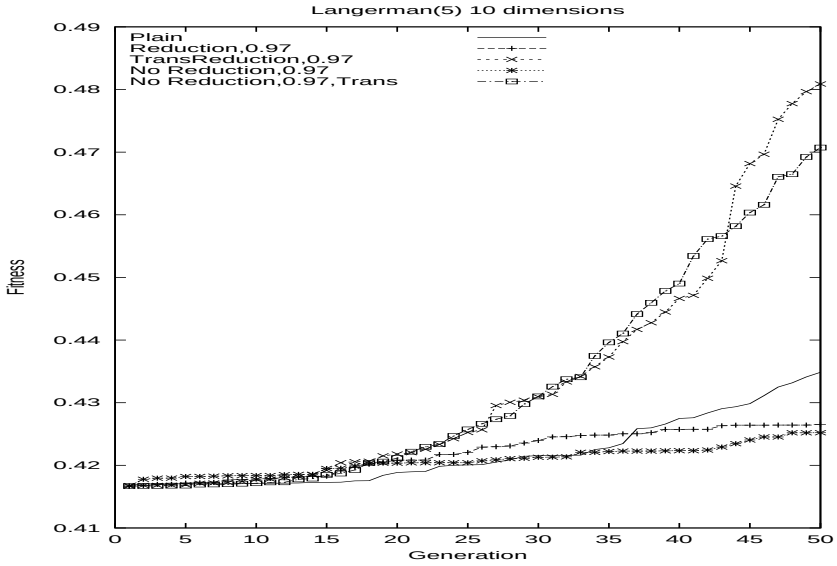

Figure 6.30: Results for the 10 dimensional Langerman–5 function

proximately -1.4999. The representation chosen for individuals in this problem was also a vector of doubles. Transformations were applied by categorizing each gene into one of four equal sized partitions of the interval $[0, 10]$. From 6.30, it is apparent that GAs using the rank–k SVD qualifier outperformed the no reduction cases. In addition, the transformation processes outperformed their counterparts by a significant margin. The SVD process without transformation did not perform as well as the Plain GA. Therefore the type of representation is an extremely important factor for success.

Future Directions

Skewness Using an idea described in [70], we may be able to discover many pieces of important information about a given qualification space. Let Q represent the qualification space. For each individual $q_i \in Q$, let v_q be the vector assigned to the individual by the rank–k SVD on Q. The rank–k SVD is δ–skewed on the qualification space Q if, for each pair of individuals q and q', $v_q \cdot v_{q'} \leq \delta \|v_q\| \|v_{q'}\|$ if q and q' belong to different categories and $v_q \cdot v_{q'} \geq (1-\delta) \|v_q\| \|v_{q'}\|$ if they belong to the same category. From Papadimitriou *et al.*s findings in [70], the SVD should be able to provide several, probabilistically verified, search techniques for problems with appropriate representation. Several pieces of key information about the GA's current state and qualification pressure can be discovered by averaging the values for δ for every pair of vectors between the two qualification spaces Q_g and Q_b. The spaces Q_g and Q_b are composed of good and bad individuals, respectively. First of all, the average of these values describes how well the chosen representation categorizes individuals. Secondly, if the amount of δ–skewness is high then the good aren't far from the bad, and therefore there is more qualification pressure on candidate individuals. This information could be used to indicate when the population has become stale. Furthermore, an algorithm could deduce how skewed the current

views of good and bad are, and adjust the amount of the d–close parameter appropriately.

Gene Clustering The preceding discussion only makes use of the individual qualification space. Using the eigenvectors that span the gene–gene autocorrelation matrix AA^T, the results could be improved by clustering the different types of good or bad individuals into groups. The information provided from this matrix would be able to show which genes are used similarly across a collection of individuals. From this information, it should be determinable which parts of the problem are decomposable into sub–problems. The GA could then focus its work on optimizing these subproblems, instead of the entire problem. The eigenvectors of AA^T are the basis of the algorithms in the next section.

Parameter Choices Unfortunately, it is hard to predict, *a priori*, the optimal transformation and parameter choices. What is needed is a concrete function with the ability to compute good parameters for a given situation. What makes parameters or representations "good," and under what assumptions for a given problem and situation? The amount of error used should depend strongly on which stage of evolution the GA is in. In the early stages, the GA should be allowed to explore the search space as widely as possible, in or-

der to find as many basins of attraction as possible. In effect, the GA should avoid having too many similar individuals in the first stages, be they poor or good individuals.

Conclusion

In this section, methods for improving a genetic algorithm's performance by using singular value decomposition to qualify candidate individuals are presented. Results from several application domains showed that using the SVD qualifier is significantly beneficial. Furthermore, it was observed that the d–close parameter, the amount of rank reduction, and choice of transformation greatly influence the amount of performance improvement achieved. It is clear that further testing and development on several different types of problems and parameter strategies will be required in order to go beyond these primitive attempts of exploiting the SVD in such a way as to exhibit positive benefits in genetic algorithms.

6.6 Eigenvectors of AA^T

SVD Incorporation

The goal is to discover the genes that are used *similarly* across the best individuals. The ideas to be presented next can be generalized

to other methods of determining similarly used genes. However, SVD yields accurate identification of subproblems in optimization problems whose solutions have a block representation. The SVD of a matrix containing the best few individuals in the entire optimization history was computed. Instead of aiming for the sole fittest individual, the GA used SVD to decompose the few fittest individuals and therefore directed the search towards a *combination* of the best individuals. Tests using large sets of individuals were not as beneficial. Perhaps this was because the SVD could not discover a single pattern for which to aim during operator restriction.

The computational complexity of computing the SVD may outweigh the complexity of the problem being solved. However, problems with a computationally expensive fitness function may benefit from the methods to be described. In particular, if complex problems can be decomposed into smaller and simpler subproblems, then the benefit will outweigh the cost of computing the SVD. Several time optimizations can also be made to decrease the amount of time used computing the SVD. For example, existing SVDs can be updated using special algorithms for adding or removing rows and columns [11]. Also, random projections are a fast alternative to singular value decomposition [70].

Restricted Mutation and Crossover

At every other generation, the mutation operator is restricted to a specific subset of the genes. This isolates the search process to the blocks in highly fit solutions, facilitating the determination of the local optimum. Similarly, the crossover operator is restricted to a specific group of genes. After crossover is applied to the reduced gene set, the unrestricted genes are replaced in their corresponding positions in the generated children. In both techniques, the restriction only happened every other generation. This enables the mutation and crossover operators to fully explore the entire space of possible chromosomes.

Genetic Engineering

A simple genetic engineering approach is tested at every generation. First, the rank–2 SVD of the top 50 best individuals is computed. Then, using a process to be described in the next subsection, a set of subproblems is generated. Next, a random subproblem with the correct size (the parameters of the problem are known) is selected and a new individual constructed by placing ones in the corresponding positions of the genes in the subproblem, and zeros everywhere else. For example, given the subproblem $\{1, 3\}$ the individual constructed would only have the first and third genes equal

to one ($[1010\cdots0]$). If no subproblem had the correct size, then no individual is engineered during that generation. Future research could also develop problem dependent heuristics to engineer good individuals out of subproblems with an arbitrary size.

Schema Reordering

Due to the nature of the problems addressed, good schema are apt to be destroyed during crossover if the locations forming the schema are scattered apart on the chromosome. To combat the disruptive nature of crossover, chromosomes are reordered to group the similar genes closer together on a chromosome. This helps to create higher–quality schemas with shorter defining lengths. SVD is used to define the reordering at every generation during optimization. The reordering groups similar genes together, allowing the GA to benefit from the building block hypothesis. This is in contrast to a strategy that only performs an initial schema preprocessing once before the GA for the Minimum Graph Bisection problem starts [19]. As the building block hypothesis suggests, the computational power of genetic algorithms largely comes from manipulating the solutions of subproblems, i.e., building blocks. Hence, identifying subproblems has been a center of many subfields within genetic and evolutionary computation. Three examples of related fields that should be studied to better connect the use of SVD to

current GA research are Linkage Learning [47], Probabilistic Model Building Genetic Algorithms [72], and Learnable Evolution Models [65].

Further Work

Future work should concentrate on several issues. First, there have been several papers that generalize the categorization powers of reduced rank SVD to situations that are not specifically transformable to block diagonal form [8]. Problem types with structures other than a block diagonal matrix need to be considered to determine additional representations that the SVD can be used with to benefit a genetic algorithm. Second, heuristics for rank choice should be identified to improve the overall subproblem determination performance. Finally, it would be interesting to create heuristics for choosing different subsets of individuals that determine the subproblems at each generation. For example, the worst, the best, or even the most diverse solutions in the optimization history could each be valid choices for the subsets of individuals that direct the optimization process.

Subproblem Determination

After the formation of a matrix of good individuals, the following steps are taken to group genes into subproblems. For every gene,

the cosines of the angles between it and every other eigenvector of AA^T are put in a matrix. In order for gene i and gene j to belong to the same subproblem, the cosines of the angle between the i^{th} and j^{th} eigenvectors of the gene–gene autocorrelation matrix have to be greater than 0.92. That is, the vectors have to be close to parallel. The cosine of an angle is an appropriate function to use because its value approaches one as two vectors become more parallel. Likewise, as two vectors become more perpendicular, the cosine of the angle between them approaches zero. The bound of 0.92 was chosen *a priori* by testing values between zero and one. However, strategies could be produced to vary this amount in a heuristic manner. For example, if the problem's solutions are required to have subproblems of genes with a particular size, then the parameter could be adjusted to favor retrieving subsets of genes with the correct size.

If ω_{ij} is the angle between the i^{th} and j^{th} gene vector then,

$$\cos\omega_{ij} = \frac{(e_i^T U\Sigma V^T)(V\Sigma U^T e_j)}{\parallel e_i^T U\Sigma V^T \parallel_2 \parallel V\Sigma U^T e_j \parallel_2} \tag{6.6}$$

For gene i and gene j to be clustered into the same subproblem, the following relation had to hold

$$\cos\omega_{ij} > 0.92 \tag{6.7}$$

Here, e_i denotes the i^{th} standard vector, which contains all zeroes except for a one in the i^{th} position. $\| \cdot \|_2$ denotes the Euclidean vector norm. The U, Σ, and V are the matrices found by the SVD.

In the document comparison domain, reduction of rank actually *improves* the quality of the information retrieved [11, 70]. Using reduced rank versions on problems with a block diagonal representation gives an approximation of what it means for two genes to be used similarly across a group of individuals. Therefore, various rank reductions are also tested. To calculate the cosines between genes in a reduced rank model, A_k is substituted for A in all of the above calculations.

$$\cos \omega_{ij} = \frac{(e_i^T U_k \Sigma_k V_k^T)(V_k \Sigma_k U_k^T e_j)}{\| e_i^T U_k \Sigma_k V_k^T \|_2 \| V \Sigma_k U_k^T e_j \|_2} \tag{6.8}$$

Berry, Drmač and Jessup provide an in depth explanation of how to efficiently compute the rank–k cosines between a query vector and the vectors contained in a reduced rank–k model [11].

Subproblem Selection Strategies

Two methods are tested for determining the subproblems the GA should work on. In the first method, *Maximum Subproblem*, the largest sized subproblem is selected. In the second method, *Subproblem Rotation*, a subproblem is chosen at random.

Low Rank Approximations

Two forms of the SVD are tested. The first is the full rank version of the SVD. The second is based on the reduced rank version, where all but the first k largest singular values are set to zero, giving A_k. As expected, the reduced rank strategies generally discover the subproblems more efficiently than the full rank versions. This is due in part to the theoretical results mentioned in the section that contains a probabilistic analysis of reduced rank spectral clustering in Chapter 3 on page 57. The performance may also have improved because, in the application domains tested, the GA is only seeking one block in the solution space. Reduction to a lower rank correctly directs the search towards the correct block because a lower value of k in A_k increases the cosines of the angles between vectors of similar types [15]. Another reason may be that in comparison with higher rank reductions, lower rank reductions are less restrictive and will identify larger subsets of related genes as the rank is reduced. Therefore, lower rank reductions allow the restrictive mutation and crossover operators to have more freedom during exploration. However, lowering the rank too much may not always increase the performance because all genes will be seen as similar to all other genes.

Experiments

As mentioned previously, SVD should perform well when analyzing problems that have a solution space that can be made block diagonal. The first problem tested was the Block Sum Partitioning problem. This problem was created and tested to provide a benchmark for the types of problems that will benefit from this research. The solution vectors of this problem can be arranged to form a block diagonal matrix. When two genes are used similarly across the solution individuals, they often contain the same value across their rows. When the SVD subproblem clustering process is applied to a matrix of correct solutions for this problem, the clusters returned are exactly the subproblems that define where a solution should have the value one.

A problem's individual type will be referred to as *symmetric* if whenever the vector obtained by applying the Boolean NOT to every gene in an individual represents the same solution to the problem. For example, the Minimum Graph Bisection problem's individual type is *symmetric* because $\{1, 0\}$ represents the same bisection as $\{0, 1\}$. SVD is not confused by solutions that are symmetric. In other words, SVD is not affected by the possible namings of a partition. This is because in problems with a *symmetric* individual type, similar genes are still used similarly across individuals

drawn from the same block. Furthermore, subproblems will be correctly discovered regardless of the order the rows or columns of the matrix are in. The Minimum Graph Bisection problem's solution vectors are symmetric and can be arranged to form a block diagonal matrix. When two genes are used similarly across the solution individuals in this problem, it means that the vertices that the genes represent are frequently placed in the same partition. SVD helps obtain an approximate *consensus* from the best invidiuals as to which vertices should be placed in the same partition.

Implementation Details

An approach similar to the $(\mu + \lambda)$ evolution strategy was used, with populations of size 100 generating 100 candidate individuals. Reinsertion was achieved by picking the best 100 individuals out of the 200 total parents and children. The results are based on the average of the best individual at each generation, over 100 different random initial populations. Let $f(x)$ be the value of the function that is being optimized when applied to an individual x. The log fitness of an individual is defined as

$$logfitness(x) = \ln \frac{1}{1 + |f(x) - target|} \leq 0 \tag{6.9}$$

In this fitness function, the function value $f(x)$ approaches its target (for example the minimum) as the fitness function approaches zero. Individuals with higher fitness represent better solutions than those with lower fitness. An individual with a fitness equal to zero is an exact solution because only then will $f(x) = target$.

The Kernighan–Lin local improvements are carefully tailored to take linear time in the number of edges of the graph. In the case of singular value decomposition, only the traditional computations with time complexity $O(m^2n+mn^2+n^3)$ for an $m \times n$ matrix are used. To improve this polynomial bound, algorithms can be employed that find low rank approximations quickly. A technique of random projection was described by Papadimitriou *et al.* in a seminal paper [70]. Some algorithms for computing approximate SVDs have time complexity *independent* of m and n [35]. These approximate SVD algorithms could be used to provide lower complexity local search methods in hybrid genetic algorithms.

Intuitively, the number of generations it takes to find a solution is the greatest factor in proving a genetic algorithm's performance. In order to assess the amount of benefit achieved using the SVD heuristics, all comparisons are made to a plain genetic algorithm that did not use the SVD heuristics. The Plain GA serves as a strawman for the SVD methods. Sometimes the Plain GA is augmented with local search and, in some cases, the spectral injection

heuristics discussed in Chapter 5. The GA is compared with various combinations of several of the current state of the art genetic operators, local search functions, and techniques used for solving the Minimum Graph Bisection problem.

6.7 Block Sum Partitioning

Problem Statement

Let $(x)_{ik}$ denote the i^{th} block of size k in a binary string. Furthermore, let $(x)_{ik}^{m}$ denote the numeric value of the m^{th} position in the i^{th} block. The (n,k)–block partition problem is defined on binary strings of length n as follows:

$$\max_{i=1,2,\ldots,\frac{n}{k}} \left(\sum_{m=1,2,\ldots,k} (x)_{ik}^{m} \quad - \sum_{l\neq i, m=1,2,\ldots,k} (x)_{lk}^{m} \right) \tag{6.10}$$

In words, the (n,k)–block partition problem is the maximum, over all the $1,2,\cdots,\frac{n}{k}$ blocks of k genes, of the sum of the elements in the block minus the sum of the elements not in the block. This problem will be called the Block Sum Partitioning problem (BSP). An individual is considered a solution when its function value is equal to k, the length of every block. By the problem's construction, this can only happen when an individual contains all ones in one block and all zeroes in every other block. Therefore, the set of individuals

that are solutions form a block diagonal matrix. From the analysis presented in Section 2.2, the SVD should perform well on this type of problem because it will be able to accurately categorize the similar genes of highly fit individuals. The highly correlated genes of good individuals will correspond to the genes that should belong to the same block.

Implementation Details

Tests were performed on the binary valued version of the $(100, 10)$–Block Sum Partitioning problem. That is, the problem of structuring 100 genes into a form with one of the 10 blocks containing 10 ones, and the rest of the 9 blocks containing all zeroes. The genetic operators do not know the value of k. Hence, the full representation space was used and not restricted to individuals with k ones during optimization. The mutation rate was set at 12%. Restricted mutation was performed by flipping a gene in a subproblem to its opposite value of either one or zero. Restricted one point crossover was used in both of the genetic subproblem strategies. The plain GA used both one point crossover and mutation without restrictions. Genetic engineering was not tested with this problem.

Results

Figure 6.32 is a plot of the average best individual at every generation for this problem using various ranks. The maximum subproblem's performance is very similar to the corresponding variations of the subproblem rotation's performance. Both of the subproblem methods outperformed the plain GA. The subproblem rotation strategy using a rank equal to 2 performed best. Furthermore, rank reduction increased the performance of the genetic algorithm in all cases.

Call a set of genes *involved* in a solution if setting each gene in the set to one and each gene out of the set to zero yields a correct solution to the problem. The following tables compare the first generation the GA discovered a solution and the first generation that each subproblem determination method correctly identified at least $\frac{2}{3}$ of a set of genes that is *involved* in a correct solution. In addition, the subproblems are only counted as being found when their size is at least $\frac{2}{3}$ of the size of a correct subproblem. They are not counted when their size is greater than the correct subproblem's size. The results in the following tables were obtained by collecting the average over 100 runs, using full rank, restricted crossover, and restricted mutation. Notice that the subproblems are discovered much earlier than the first solution.

BSP (full rank)	Solution	Subproblem found
Maximum	103.87	**63.64**
Rotation	86.92	**75.76**

Although the overall first solution performance is better with the rank 1 reduction for this problem, the subproblems are not typically discovered until after the first solution is found. A possible explanation for this is that the size of the subproblem found when the rank is reduced is much larger than the size of the subproblem when using full rank. This is because under a reduced rank model, genes are more likely to be similar to other genes. While the correct subproblem is likely still represented in the set, the size of the set is usually much larger than the size of a correct subproblem. In these cases the subproblem is not counted as being found because it is too big.

BSP (rank 1)	Solution	Subproblem found
Maximum	87.27	**263.74**
Rotation	75.59	**231.5**

As the following table indicates, the rank 2 reductions resulted in the best overall performance.

BSP (rank 2)	Solution	Subproblem found
Maximum	77.79	**45.02**
Rotation	73.4	**78.06**

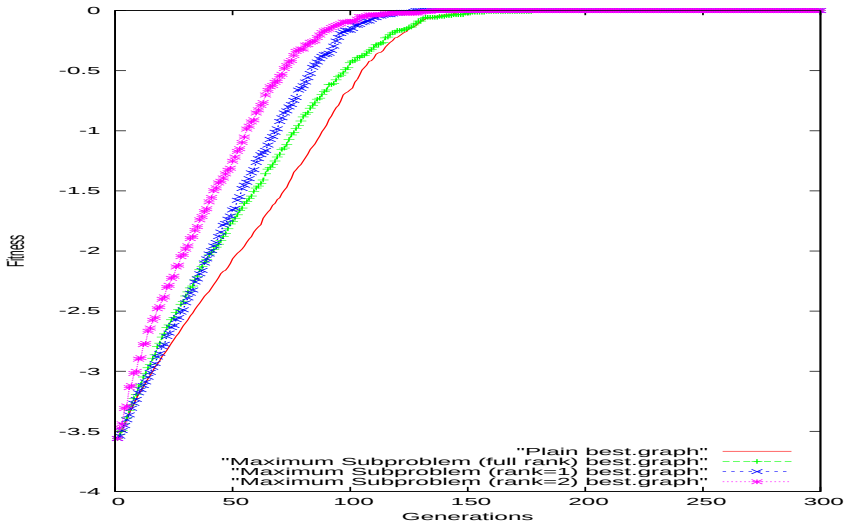

Figure 6.31: The average best individual per generation for the BSP problem using the Maximum Subproblem strategy.

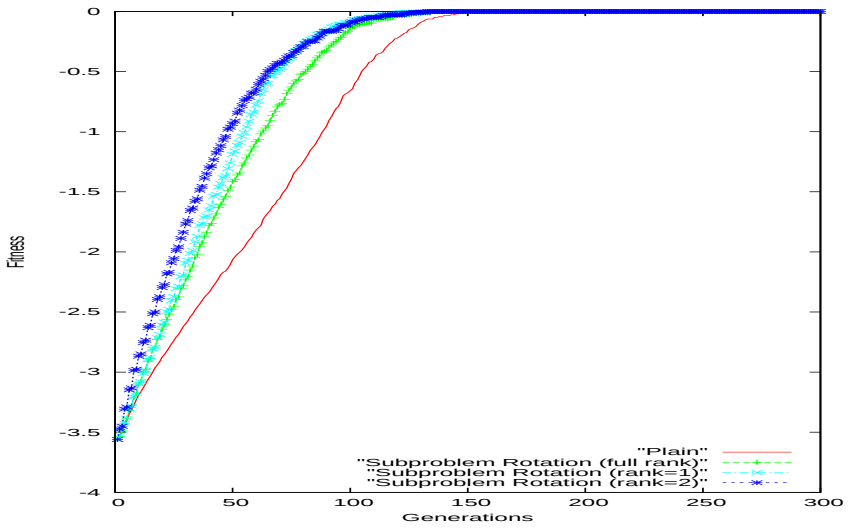

Figure 6.32: The average best individual per generation for the BSP problem using the Subproblem Rotation Strategy.

6.8 Minimum Graph Bisection

Problem Statement

A bisection of a graph $G = (V, E)$ with an even number of vertices is a pair of disjoint subsets $V_1, V_2 \subset V$ of equal size. The cost of a bisection is the number of edges $(a, b) \in E$ such that $a \in V_1$ and $b \in V_2$. The Minimum Graph Bisection problem takes as input a graph G with an even number of vertices, and returns a bisection of minimum cost. The Minimum Graph Bisection problem has been shown to be NP–Complete [39]. Many heuristics have been developed for this problem. Perhaps the best known is the Kernighan–Lin heuristic [58, 16]. Graph partitioning with genetic algorithms has been studied extensively [60], [19], [59], [78], [79]. Singular value decomposition has also proved to be a useful tool when clustering graphs [27], [56]. However, this dissertation contains the first attempt to combine these results, providing strategies for using singular value decomposition in a genetic algorithm for the Minimum Graph Bisection problem.

Implementation Details

Individuals were represented in binary. If the i^{th} component of an individual was one, then the i^{th} vertex was placed in the set V_1. Otherwise, if the i^{th} component of an individual was zero, then the i^{th} vertex was put in the set V_2. Notice that individuals are symmetric in this representation. The mutation rate was set at 12%. A modified mutation method of switching two random genes was implemented to keep the number of ones and zeroes in an individual equal. In the case of subproblem evolution, a gene from the subproblem area was flipped and an opposite gene from the non–subproblem area is also flipped. In plain GAs, the mutation operator simply exchanged the values of two opposite genes. The crossover operator was adapted from an earlier paper on graph bisection with GAs [19]. It is a modified five point crossover that attempts to account for the symmetric nature of graph bisection solutions. No restriction on the locus of crossover was used in this problem. The highly correlated genes correspond to vertices that the current population believes should be clustered into the same partition. Before the GA started, the ordering of the vertices was permuted in order to prevent the results from containing any possible bias on the input.

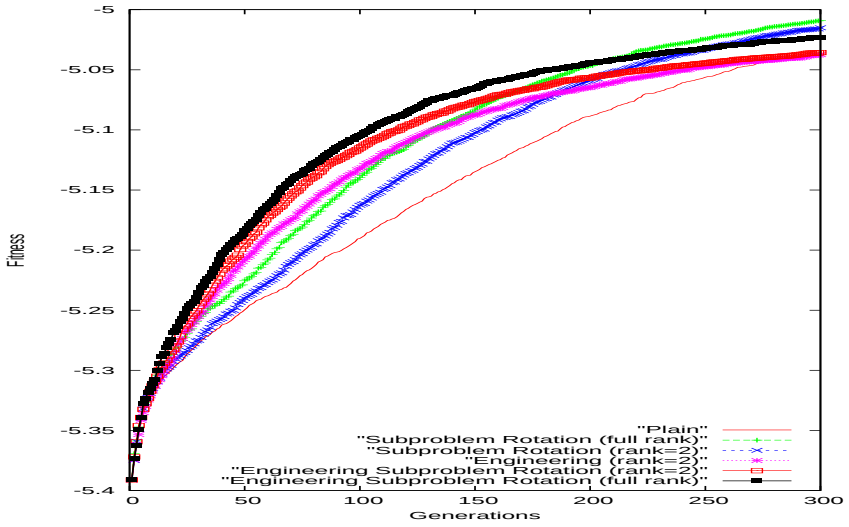

Figure 6.33: The average best individual per generation for the Minimum Graph Bisection problem on random graphs with 100 vertices and 5% edge probability.

Random Graphs

Figure 6.33 contains the average fitness of the best individual at each generation over 100 different random graphs. An edge between two vertices was created with a 5% chance. In this problem, the subproblem methods outperformed the plain GA, but only by a slight margin. Graphs with higher chances of an edge occurring between vertices produced very similar results. The decrease in performance in the engineering results after the first 100 generations indicates that the populations may have become genetically stale.

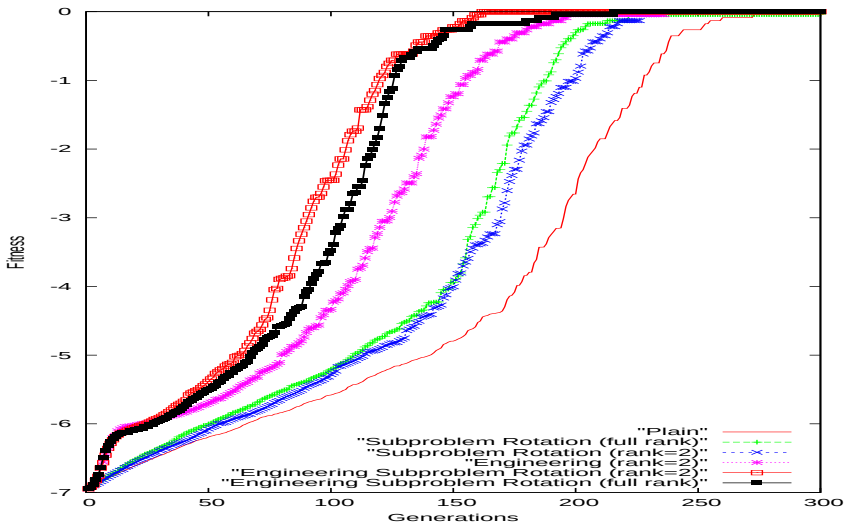

Figure 6.34: The average best individual per generation for the Minimum Graph Bisection Cluster problem on random graphs with 100 vertices and two main clusters with no local improvement.

Highly Clustered Random Graphs

The random graphs for this problem were created by first randomly dividing all of the vertices into two disjoint and equal sized sets. Next, edges within a set are created with a 98% probability. Then, edges between vertices in different sets are created with probability equal to 5%. This problem will be called the Minimum Graph Bisection Cluster problem. Presumably, SVD will perform remarkably better in the cases where the random graph contains two main clusters. Accordingly, tests performed on random graphs that are explicitly constructed to contain most of their weight in two clus-

ters, indicate an increase in the performance of the SVD subproblem and engineering methods. Figure 6.34 is a plot of the results from these highly clustered random graphs. Once again, all SVD methods outperform the plain GA. Furthermore, the combination of restricted operators with genetic engineering yields better results than using either restricted operators or genetic engineering alone. Engineering consistently gave a significant performance boost during the first fifty generations of optimization. Additionally, engineering is improved by reducing the rank to 2.

The following table lists the average first generation the GA subproblem determination methods identified at least $\frac{2}{3}$ of the genes *involved* in a solution. The results in the following table were obtained by collecting the average over 100 runs. The subproblems were discovered much earlier than the first solution is obtained.

GBC (full rank)	Solution	Subproblem found
Plain	206.58	NA
Maximum	176.22	**99.92**
Rotation	154.64	**97.38**

For this problem, rank one reduction did not improve the GA's overall solution performance for either subproblem strategy. This indicates that the reduction to rank one may have deteriorated the solution space too much. On the other hand, the subproblems were typically found earlier than the full rank version found them.

GBC (rank 1)	Solution	Subproblem found
Maximum	181.29	**74.46**
Rotation	204.45	**86.7**

Figure 6.35 depicts the results from an experiment that compares most of the heuristics that have been described. In addition, local searches were performed on the entire population at each generation. Appropriately, many of the heuristics and hypotheses of this dissertation are given support by the applicability of theorems presented in earlier chapters. Reduction of rank, spectral injection, subproblem rotation, engineering, and schema reordering are all verified to positively impact the performance of the genetic algorithm separately for this graph. Figure 6.36 shows that the performance increase is much more dramatic when the the local search operator is not performed. However, Figure 6.37 shows that when the Kernighan–Lin local improvement is used with problems for which KL does not perform well, the performance improvement when engineering individuals outperforms the plain GA by a more significant margin.

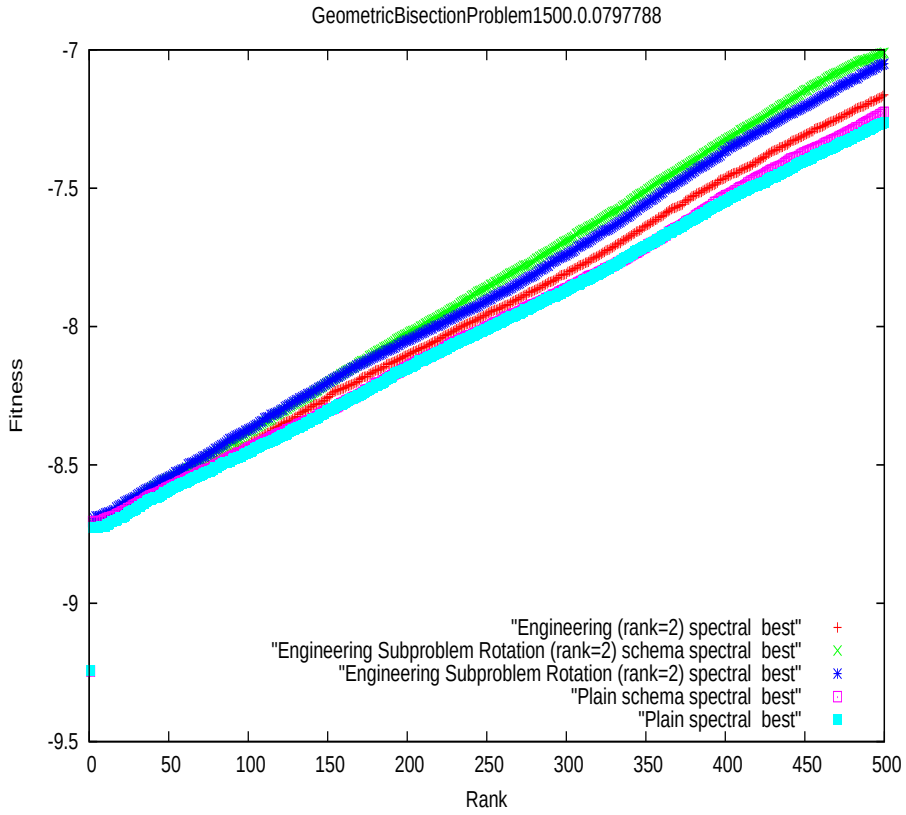

Figure 6.35: With spectral injection and local improvements using a modified Kernighan and Lin approach.

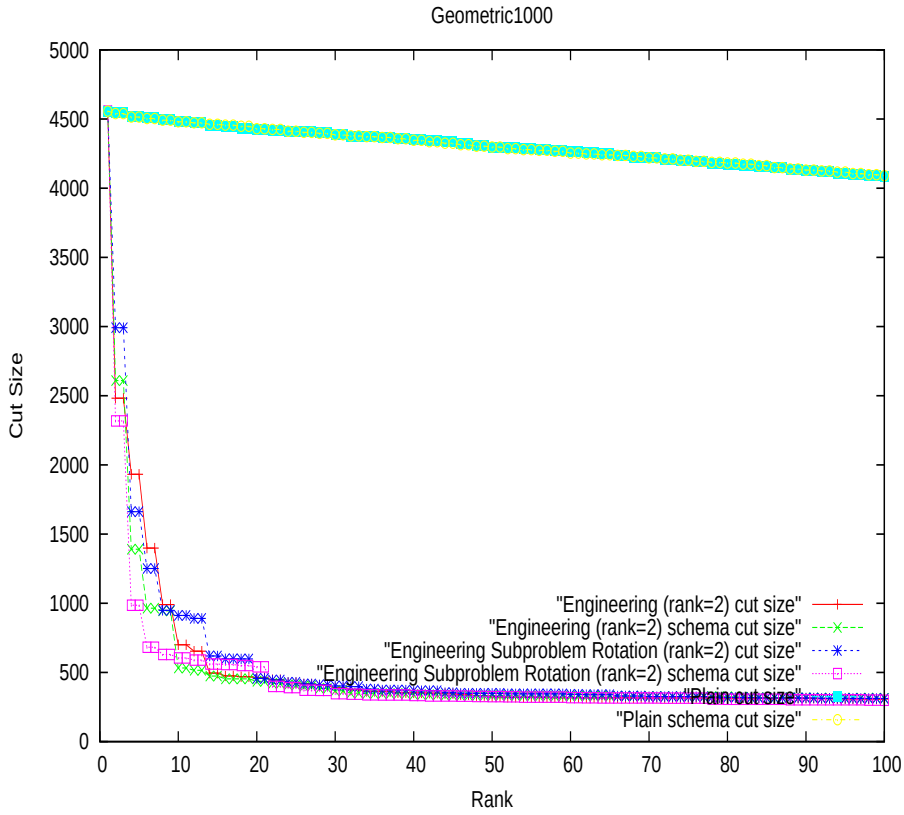

Figure 6.36: Cut size results corresponding to no spectral injection and no local improvements for Bui's Geometric graph U1000.20.

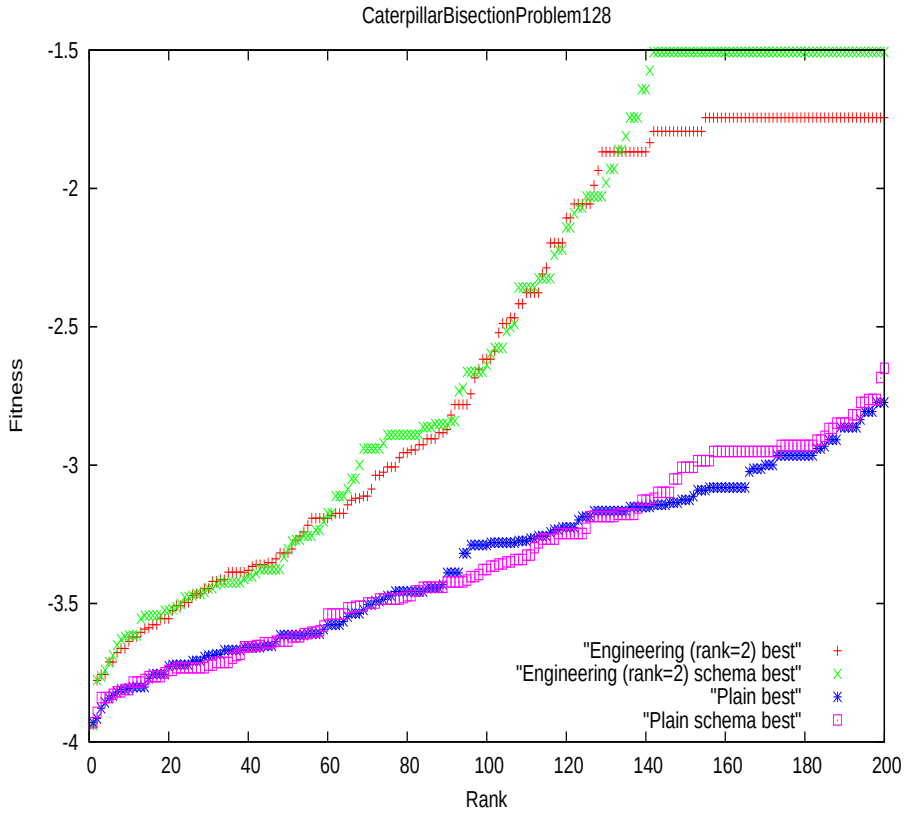

Figure 6.37: Results corresponding to local improvements for a Caterpillar Bisection Problem on 128 vertices.

Chapter 7 - Conclusion

In conclusion, there remains a tremendous amount of work to be done with singular value decomposition for information retrieval, graph bisection, and genetic algorithms. It is hoped that the work in this dissertation will provide a head start to future researchers willing to tackle these important research tasks. A review of the contributions of this dissertation and a few open questions raised by it follow.

7.1 Future Research

A few open questions that have been raised by the work in this dissertation are listed below.

- Is there a single proof strategy for identifying which singular vectors give minimum bisection solutions when using different adjacency representations on different graphs?
- What is the difference between the singular vectors in different representations? Given the algebraic relations between the matrices themselves, is there an algebraic equation to convert between their singular vectors?

- Do algorithms exist that can take advantage of the cut size solution quality oscillations by combining eigenvectors to give better solutions to graph bisection problems?
- How do convergence and alternate eigenvector finding algorithms influence partitionings?

7.2 Review of Contributions

Below is a brief summary of some of the major contributions of this dissertation.

- Clarification of a well known LSI theorem, with counterexamples.
- Improvement of heuristics for finding the minimum bisection of a graph.
- Minimum bisection guarantees for graphs with a certain structures using a new proof strategy.
- Empirical evidence that multiple eigenvectors can be useful in spectral bisection.
- Several novel applications of singular value decomposition in genetic algorithms.

Bibliography

[1] N. Alon, *Eigenvalues and expanders*, Combinatorica **6** (1986), no. 2, 83–96. MR MR875835 (88e:05077)

[2] N. Alon and V. D. Milman, λ_1, *isoperimetric inequalities for graphs, and superconcentrators*, J. Combin. Theory Ser. B **38** (1985), no. 1, 73–88. MR MR782626 (87b:05092)

[3] Noga Alon, *Spectral techniques in graph algorithms (invited paper)*, LATIN'98: theoretical informatics (Campinas, 1998), Lecture Notes in Comput. Sci., vol. 1380, Springer, Berlin, 1998, pp. 206–215. MR MR1635529 (99d:68180)

[4] Charles J. Alpert, Andrew B. Kahng, and So-Zen Yao, *Spectral partitioning with multiple eigenvectors*, Discrete Appl. Math. **90** (1999), no. 1-3, 3–26. MR MR1665987 (99m:68145)

[5] Charles J. Alpert and So-Zen Yao, *Spectral partitioning: The more eigenvectors, the better*, DAC, 1995, pp. 195–200.

[6] Konstantin Andreev and Harald Räcke, *Balanced graph partitioning*, SPAA, 2004, pp. 120–124.

[7] Bengt Aspvall and John R. Gilbert, *Graph coloring using eigenvalue decomposition*, SIAM J. Algebraic Discrete Methods **5** (1984), no. 4, 526–538. MR MR763982 (86a:05044)

[8] Yossi Azar, Amos Fiat, Anna R. Karlin, Frank McSherry, and Jared Saia, *Spectral analysis of data*, STOC, 2001, pp. 619–626.

[9] Earl R. Barnes, *An algorithm for partitioning the nodes of a graph*, SIAM J. Algebraic Discrete Methods **3** (1982), no. 4, 541–550. MR MR679649 (84b:90036)

[10] E. Beltrami, *Sulle funzioni bilineari*, Giornale di Matematiche ad Uso degli Studenti Delle Universita **11** (1873), 98–106, An English translation by D. Boley is available in Technical Report 90–37, Department of Computer Science, University of Minnesota, Minneapolis, 1990.

[11] Michael W. Berry, Zlatko Drmač, and Elizabeth R. Jessup, *Matrices, vector spaces, and information retrieval*, SIAM Rev. **41** (1999), no. 2, 335–362 (electronic). MR MR1684547 (2000b:15001)

[12] Michael W. Berry, Susan T. Dumais, and Gavin W. O'Brien, *Using linear algebra for intelligent information retrieval*, SIAM Rev. **37** (1995), no. 4, 573–595. MR MR1368388 (96j:68061)

[13] Norman Biggs, *Algebraic graph theory*, Cambridge University Press, London, 1974. MR MR0347649 (50 #151)

[14] Ravi B. Boppana, *Eigenvalues and graph bisection: An average-case analysis (extended abstract)*, FOCS, 1987, pp. 280–285.

[15] M. Brand and K. Huang, *A unifying theorem for spectral embedding and clustering*, International Workshop On Artificial Intelligence and Statistics, January 2003, International Workshop On Artificial Intelligence and Statistics (AI & Statistics), January 2003, AI & Statistics 2003.

[16] T. N. Bui, S. Chaudhuri, F. T. Leighton, and M. Sipser, *Graph bisection algorithms with good average case behavior*, Combinatorica **7** (1987), no. 2, 171–191. MR MR905164 (88k:05119)

[17] Thang Nguyen Bui and Curt Jones, *Finding good approximate vertex and edge partitions is NP–hard*, Inform. Process. Lett. **42** (1992), no. 3, 153–159. MR MR1168771 (93h:68111)

[18] Thang Nguyen Bui and Curt Jones, *A heuristic for reducing fill-in in sparse matrix factorization*, PPSC, 1993, pp. 445–452.

[19] Thang Nguyen Bui and Byung Ro Moon, *Genetic algorithm and graph partitioning*, IEEE Trans. Comput. **45** (1996), no. 7, 841–855. MR MR1423913

[20] P. CHEBYCHEV, *Sur les valeurs limites des intégrales*, Jour. Math. Pur. Appl. **19** (1874), no. 2, 157–160.

[21] Stephen A. Cook, *The complexity of theorem-proving procedures*, STOC, 1971, pp. 151–158.

[22] D. Cvetković, P. Rowlinson, and S. Simić, *Eigenspaces of graphs*, Encyclopedia of Mathematics and its Applications, vol. 66, Cambridge University Press, Cambridge, 1997. MR MR1440854 (98f:05111)

[23] Dragoš M. Cvetković, Michael Doob, and Horst Sachs, *Spectra of graphs*, Pure and Applied Mathematics, vol. 87, Academic Press Inc. [Harcourt Brace Jovanovich Publishers], New York, 1980. MR MR572262 (81i:05054)

[24] Scott C. Deerwester, Susan T. Dumais, Thomas K. Landauer, George W. Furnas, and Richard A. Harshman, *Indexing by latent semantic analysis*, Journal of the American Society of Information Science **41** (1990), no. 6, 391–407.

[25] Chris H. Q. Ding, *A similarity-based probability model for latent semantic indexing*, SIGIR, 1999, pp. 58–65.

[26] W. E. Donath and A. J. Hoffman, *Lower bounds for the partitioning of graphs*, IBM J. Res. Develop. **17** (1973), 420–425. MR MR0329965 (48 #8304)

[27] Petros Drineas, Alan M. Frieze, Ravi Kannan, Santosh Vempala, and V. Vinay, *Clustering large graphs via the singular value decomposition*, Machine Learning **56** (2004), no. 1–3, 9–33.

[28] R. O. Duda and P. E. Hart, *Pattern classification and scene analysis*, Wiley, New York, 1972.

[29] C. Eckart and G. Young, *The approximation of one matrix by another of lower rank*, Psychometrika **1** (1936), 211–218.

[30] P. Erdős and A. Rényi, *On the evolution of random graphs*, Bull. Inst. Internat. Statist. **38** (1961), 343–347. MR MR0148055 (26 #5564)

[31] Uriel Feige, Robert Krauthgamer, and Kobbi Nissim, *Approximating the minimum bisection size (extended abstract)*, Proceedings of the Thiry-Second Annual ACM Symposium on Theory of Computing (New York), ACM, 2000, pp. 530–536 (electronic). MR MR2115290

[32] C. M. Fiduccia and R. M. Mattheyses, *A linear-time heuristic for improving network partitions*, DAC '82: Proceedings of the 19th conference on Design automation (Piscataway, NJ, USA), IEEE Press, 1982, pp. 175–181.

[33] Miroslav Fiedler, *Algebraic connectivity of graphs*, Czechoslovak Math. J. **23(98)** (1973), 298–305. MR MR0318007 (47 #6556)

[34] Myron Flickner, Harpreet Sawhney, Wayne Niblack, Jonathan Ashley, Qian Huang, Byron Dom, Monika Gorkani, Jim Hafner, Denis Lee, Dragutin Petkovic, David Steele, and Peter Yanker, *Query by image and video content: The QBIC system*, IEEE Computer **28** (1995), no. 9, 23–32.

[35] Alan Frieze, Ravi Kannan, and Santosh Vempala, *Fast Monte-Carlo algorithms for finding low-rank approximations*, J. ACM **51** (2004), no. 6, 1025–1041 (electronic). MR MR2145262 (2005m:65006)

[36] Alan Frieze and Colin McDiarmid, *Algorithmic theory of random graphs*, Random Structures Algorithms **10** (1997), no. 1–2, 5–42. MR MR1611517 (99c:05177)

[37] F. R. Gantmacher, *The theory of matrices. Vols. 1, 2*, Translated by K. A. Hirsch, Chelsea Publishing Co., New York, 1959. MR MR0107649 (21 #6372c)

[38] M. R. Garey, D. S. Johnson, and L. Stockmeyer, *Some simplified np–complete problems*, STOC '74: Proceedings of the sixth annual ACM symposium on Theory of computing (New York, NY, USA), ACM Press, 1974, pp. 47–63.

[39] M. R. Garey, D. S. Johnson, and L. Stockmeyer, *Some simplified NP–complete graph problems*, Theoret. Comput. Sci. **1** (1976), no. 3, 237–267. MR MR0411240 (53 #14978)

[40] G.Golub and C. Reinsch, *Handbook for matrix computation ii, linear algebra*, Springer–Verlag, 1971.

[41] John R. Gilbert and Earl Zmijewski, *A parallel graph partitioning algorithm for a message-passing multiprocessor*, Internat. J. Parallel Programming **16** (1987), no. 6, 427–449. MR MR980804 (90d:68029)

[42] D. E. Goldberg, *Genetic algorithms in search, optimization, and machine learning*, Addison–Wesley, Reading, Mass., 1989.

[43] Gene H. Golub and Charles F. Van Loan, *Matrix computations*, Johns Hopkins Studies in the Mathematical

Sciences, Johns Hopkins University Press, Baltimore, MD, 1996. MR MR1417720 (97g:65006)

[44] William A. Greene, *A genetic algorithm with self-distancing bits but no overt linkage*, GECCO, 2002, pp. 367–374.

[45] Stephen Guattery and Gary L. Miller, *On the performance of spectral graph partitioning methods*, Proceedings of the Sixth Annual ACM-SIAM Symposium on Discrete Algorithms (San Francisco, CA, 1995) (New York), ACM, 1995, pp. 233–242. MR MR1321854 (95m:05225)

[46] Stephen Guattery and Gary L. Miller, *On the performance of spectral graph partitioning methods*, SODA, 1995, pp. 233–242.

[47] Georges R. Harik and David E. Goldberg, *Learning linkage*, Foundations of Genetic Algorithms, 1996, pp. 247–262.

[48] Bruce Hendrickson and Robert W. Leland, *A multi-level algorithm for partitioning graphs*, Supercomputing '95: Proceedings of the 1995 ACM/IEEE conference on Supercomputing, 1995.

[49] Wassily Hoeffding, *Probability inequalities for sums of bounded random variables*, J. Amer. Statist. Assoc. **58** (1963), 13–30. MR MR0144363 (26 #1908)

[50] Thomas Hofmann, *Probabilistic latent semantic indexing*, SIGIR, 1999, pp. 50–57.

[51] John H. Holland, *Adaptation in natural and artificial systems*, University of Michigan Press, Ann Arbor, Mich., 1975. MR MR0441393 (55 #14256)

[52] Roger A. Horn and Charles R. Johnson, *Topics in matrix analysis*, Cambridge University Press, Cambridge, 1991. MR MR1091716 (92e:15003)

[53] David S. Johnson, Cecilia R. Aragon, Lyle A. McGeoch, and Catherine Schevon, *Optimization by simulated annealing: an experimental evaluation. part i, graph partitioning*, Oper. Res. **37** (1989), no. 6, 865–892.

[54] Curt Allen Jones, *Vertex and edge partitions of graphs*, Ph.D. thesis, Pennsylvania State University, University Park, PA, USA, 1992.

[55] C. Jordan, *Mémoire sur les formes bilinéaires*, Journal de Mathématiques Pures et Appliquées, Deuxième Série **19** (1874), 35–54.

[56] Ravi Kannan, Santosh Vempala, and Adrian Vetta, *On clusterings: good, bad and spectral*, J. ACM **51** (2004), no. 3, 497–515 (electronic). MR MR2145863

[57] Richard M. Karp, *Reducibility among combinatorial problems*, Complexity of computer computations (Proc. Sympos., IBM Thomas J. Watson Res. Center, Yorktown Heights, N.Y., 1972), Plenum, New York, 1972, pp. 85–103. MR MR0378476 (51 #14644)

[58] B. Kernighan and S. Lin, *An Efficient Heuristic Procedure for Partitioning Graphs*, Bell Systems Journal **49** (1972), 291–307.

[59] Jong-Pil Kim and Byung-Ro Moon, *A hybrid genetic search for multi-way graph partitioning based on direct partitioning*, Proceedings of the Genetic and Evolutionary Computation Conference (GECCO–2001) (San Francisco, California, USA) (Lee Spector *et al.*, ed.), Morgan Kaufmann, 7–11 July 2001, pp. 408–415.

[60] Harpal S. Maini, Kishan G. Mehrotra, Mohan Mohan, and Sanjay Ranka, *Genetic algorithms for graph partitioning and incremental graph partitioning*, Tech. Report CRPC–TR94504, Center for Research on Parallel Computation, Rice University, Houston, TX, 1994.

[61] Jacob G. Martin, *Subproblem optimization by gene correlation with singular value decomposition*, GECCO '05: Proceedings

of the 2005 conference on Genetic and evolutionary computation (New York, NY, USA), ACM Press, 2005, pp. 1507–1514.

[62] Jacob G. Martin and Khaled Rasheed, *Using singular value decomposition to improve a genetic algorithm's performance*, Proceedings of the 2003 Congress on Evolutionary Computation CEC2003 (Canberra), IEEE Press, 8-12 December 2003, pp. 1612–1617.

[63] Colin McDiarmid, *On the method of bounded differences*, Surveys in combinatorics, 1989 (Norwich, 1989), London Math. Soc. Lecture Note Ser., vol. 141, Cambridge Univ. Press, Cambridge, 1989, pp. 148–188. MR MR1036755 (91e:05077)

[64] Zbigniew Michalewicz, *Genetic algorithms + data structures = evolution programs*, Springer–Verlag, Berlin, 1994. MR MR1329091 (96h:68001)

[65] Ryszard S. Michalski, *Learnable evolution model: Evolutionary processes guided by machine learning*, Mach. Learn. **38** (2000), no. 1–2, 9–40.

[66] Cleve Moler and Donald Morrison, *Singular value analysis of cryptograms*, Amer. Math. Monthly **90** (1983), no. 2, 78–87. MR MR691178 (84c:68080)

[67] Burkhard Monien, *The bandwidth minimization problem for caterpillars with hair length* 3 *is NP–complete*, SIAM J. Algebraic Discrete Methods **7** (1986), no. 4, 505–512. MR MR857587 (88b:68064)

[68] Ben Noble and James W. Daniel, *Applied linear algebra*, third ed., Prentice–Hall, Englewood Cliffs, NJ, USA, 1988.

[69] C. H. Papadimitriou and M. Sideri, *The bisection width of grid graphs*, Math. Systems Theory **29** (1996), no. 2, 97–110. MR MR1368793 (97d:68093)

[70] Christos H. Papadimitriou, Prabhakar Raghavan, Hisao Tamaki, and Santosh Vempala, *Latent semantic indexing: a probabilistic analysis*, J. Comput. System Sci. **61** (2000), no. 2, 217–235. MR MR1802556 (2001m:68039)

[71] Alex Pothen, Horst D. Simon, and Kang-Pu Liou, *Partitioning sparse matrices with eigenvectors of graphs*, SIAM J. Matrix Anal. Appl. **11** (1990), no. 3, 430–452. MR MR1054210 (91h:65064)

[72] Kumara Sastry and David E. Goldberg, *Probabilistic model building and competent genetic programming*, Genetic Programming Theory and Practise (Rick L. Riolo and Bill Worzel, eds.), Kluwer, 2003, pp. 205–220.

[73] John E. Savage and Markus G. Wloka, *Parallelism in graph–partitioning*, J. Parallel Distrib. Comput. **13** (1991), no. 3, 257–272. MR MR1136211 (92g:68058)

[74] Bruce R. Schatz, *Automated analysis of cryptogram cipher equipment*, CRYPTOLOGIA **1** (1977), no. 2, 116–142.

[75] Erhard Schmidt, *Zur Theorie der linearen und nichtlinearen Integralgleichungen*, Math. Ann. **63** (1907), no. 4, 433–476. MR MR1511415

[76] J. J. Seidel, *A survey of two-graphs*, Colloquio Internazionale sulle Teorie Combinatorie (Rome, 1973), Tomo I, Accad. Naz. Lincei, Rome, 1976, pp. 481–511. Atti dei Convegni Lincei, No. 17. MR MR0550136 (58 #27659)

[77] Jouni K. Seppänen, Ella Bingham, and Heikki Mannila, *A simple algorithm for topic identification in 0–1 data*, PKDD, 2003, pp. 423–434.

[78] A. J. Soper, C. Walshaw, and M. Cross, *A combined evolutionary search and multilevel optimisation approach to*

graph–partitioning, J. Global Optim. **29** (2004), no. 2, 225–241. MR MR2092958 (2005k:05228)

[79] Alan J. Soper, Chris Walshaw, and Mark Cross, *A combined evolutionary search and multilevel approach to graph partitioning*, Proceedings of the Genetic and Evolutionary Computation Conference (GECCO–2000) (Las Vegas, Nevada, USA) (David Goldberg Darrell Whitley *et al.*, ed.), Morgan Kaufmann, 10–12 July 2000, pp. 674–681.

[80] G. W. Stewart, *On the early history of the singular value decomposition*, SIAM Rev. **35** (1993), no. 4, 551–566. MR MR1247916 (94f:15001)

[81] G. W Stewart, *Matrix algorithms. Vol. I*, Society for Industrial and Applied Mathematics, Philadelphia, PA, 1998. MR MR1653546

[82] J. J. Sylvester, *On the reduction of a bilinear quantic of the* n^{TH} *order to the form of a sum of* n *products by a double orthogonal substitution*, Messenger of Mathematics **19** (1889), 42–46.

[83] Edwin R. van Dam and Willem H. Haemers, *Which graphs are determined by their spectrum?*, Linear Algebra Appl. **373** (2003), 241–272. MR MR2022290 (2005a:05135)

[84] J. H. van Lint and J. J. Seidel, *Equilateral point sets in elliptic geometry*, Nederl. Akad. Wetensch. Proc. Ser. A 69=Indag. Math. **28** (1966), 335–348. MR MR0200799 (34 #685)

[85] Richard S. Varga, *Matrix iterative analysis*, Prentice–Hall Inc., Englewood Cliffs, N.J., 1962. MR MR0158502 (28 #1725)

[86] Hermann Weyl, *Das asymptotische Verteilungsgesetz der Eigenwerte linearer partieller Differentialgleichungen (mit einer Anwendung auf die Theorie der Hohlraumstrahlung)*, Math. Ann. **71** (1912), no. 4, 441–479. MR MR1511670