Clustering via Matrix Powering

[Extended Abstract]

Hanson Zhou [*]
M.I.T Computer Science and Artificial
Intelligence Laboratory
200 Technology Square
Cambridge, MA 02139
hmzhou@mit.edu

David Woodruff [†]
M.I.T Computer Science and Artificial
Intelligence Laboratory
200 Technology Square
Cambridge, MA 02139
dpwood@mit.edu

ABSTRACT

Given a set of n points with a matrix of pairwise similarity measures, one would like to partition the points into clusters so that similar points are together and different ones apart. We present an algorithm requiring only matrix powering that performs well in practice and bears an elegant interpretation in terms of random walks on a graph. Under a certain mixture model involving planting a partition via randomized rounding of tailored matrix entries, the algorithm can be proven effective for only a single squaring. It is shown that the clustering performance of the algorithm degrades with larger values of the exponent, thus revealing that a single squaring is optimal.

1. INTRODUCTION

Similarity-based clustering partitions a set of points given a matrix of pairwise similarities and finds application in many important problems. One motivating example is clustering web search results. A search for “jaguar” may return numerous pages relevant to either the car or the cat. Given counts of links between pairs of pages as an indicator of similarity, one would like to group the car results together and the cat results together. In the most general form, we are given a set of n points and a matrix M, where M_{ij} gives the distance or similarity between points i and j. The goal is to partition the points such that similar points are grouped together and different points apart. Our approach consists of powering the similarity matrix and comparing rows.

Clustering plays a major role in data mining, with many applications such as scientific data exploration, information retrieval and text mining, spatial database applications, web analysis, CRM (customer relationship management), marketing, medical diagnostics, and computational biology. For surveys and recent work on clustering, see [11, 16, 19, 2, 22, 7, 15, 6, 18, 10, 25]. Traditional clustering problems include the “k-center problem” [13, 14] and the “k-median problem” [7, 15, 6, 22]. The objective of the k-center problem is to minimize the maximum diameter over all k clusters, whereas the goal of the k-median problem is to find k “centers” so that the sum of distances from points to their closest center is minimized.

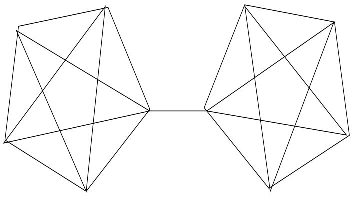

Figure 1: Dumbbell

The spectral and random walk approach to clustering is closely related to our work. For example, Kannan, Vetta, and Vempala define a bi-criterial measure of cluster quality in which the number of clusters is to be minimized while maximizing the minimum cluster conductance [18]. This reflects a desire to keep the number of groups small, while maintaining a high degree of similarity within each group. Previously, Papadimitriou, et al., proved theoretical guarantees for classifying documents to the correct topic under certain assumptions about topic purity and term overlap, via spectral methods [23]. Azar, et al., undertake a similar task, but introduce a more general data mining model [3]. In a different vein, Drineas, et al., give an approximation algorithm for clustering points in Euclidean space so as to minimize the sum of distances squared to each cluster center by first solving a continuous relaxation of the problem using the SVD [12].

To interpret our problem in terms of connectivity in graphs, consider the “dumbbell” of Figure 1, in which there are 2 cliques connected by an edge. The corresponding special case similarity matrix has a 1 in the (i, j) entry if (i, j) is an edge in the graph and 0 otherwise, and grouping into similar clusters corresponds to identifying the well connected components. Partitioning into clusters of high connectivity would yield each of the 2 cliques as a cluster. More generally, each of the parts could be somewhat less densely connected, and the bridging edges could be somewhat more numerous, but still sparse relative to the connectivity within each part. We would like an algorithm that could classify the vertices into the desired clusters with good probability, for suitable ranges of the difference between the number of intra- and inter- cluster edges. This

[*] supported in part by NSF Grant CCR-9820978
[†] supported by an NDSEG Fellowship

PODS 2004, June 14-16, 2004, Paris, France

paper shows that, under a certain generative model for the similarity matrix, our algorithm successfully clusters a large fraction of the nodes with good probability, so long as a certain probability gap in the model is sufficiently large.

Section 2 presents our algorithm. Section 3 discusses the model under which we provide theoretical guarantees, and Section 4 gives these guarantees. Finally, Section 5 details our experiments and Section 6 concludes with some future directions.

2. THE MATRIX POWERING ALGORITHM

We propose the following algorithm for clustering a set of n points with pairwise similarities. For a symmetric matrix M, let M_k^t denote the k-th row (or column) of M^t.

```
Algorithm
Input: A symmetric $\hat{A}$ encoding the pairwise similarities between
       nodes
Output: A partitioning of the nodes into clusters

  1. Select some appropriate exponent t
  2. Select some appropriate threshold $\epsilon$
  3. Compute $\hat{A}^t$
  4. For each pair of yet unclassified nodes i,j
       • if $||\hat{A}_i^t - \hat{A}_j^t||^2 < \epsilon$, then i and j are in the same cluster
       • else, i and j are in different clusters
```

Naturally, this leads to the question of how to select the values in steps 1 and 2. In fact, the very effectiveness and performance of the algorithm hinges on using the right t, and depending on t, the right ϵ. We give analytic results to show that we can correctly cluster for $t = 2$ so long as a certain probability gap under a certain generative model for $\hat{A}$ is $\Omega(n^{-\frac{1}{4}})$. In fact, $t = 2$ turns out to be the optimal value of t in some sense, and we demonstrate that the required gap only becomes larger for greater t. We support our theoretical results with experimental evidence. It is important to note that this algorithm is independent of any generative model and is applicable to an arbitrary matrix of similarities $\hat{A}$, for any measure of similarity, generated from any (possibly randomized) process.

2.1 A Connection to Random Walks

It is well known that for any probability distribution vector x corresponding to start position, $x^T M^t$ gives the probabilities of being at a node i after t steps of a random walk on the graph, where each step selects an out-going edge with probability proportional to the edge weight. Moreover, as $t \rightarrow \infty$, $\pi = x^T M^t$ gives the stationary distribution, and $\pi_i = \frac{deg(i)}{2m}$, where $deg(i)$ is the degree (sum of incident edge weights) of i, and m is the total weight of all edges [20]. Thus, for t large enough, our algorithm sheds little light on the clusters, other than what can be deduced from the graph's degree sequence.

As may be suggested by the above, there is a pleasing interpretation of the algorithm in the context of random walks. Viewing $\hat{A}$ as a transition matrix and letting e_i be the vector with 1 in the ith position and 0s elsewhere, $\hat{A}_i^t = e_i^T \hat{A}^t$ gives the probability distribution on the position of a random walk starting from node i after t steps. From this we see that the algorithm makes a pairwise grouping decision based on the L2 distance between the probability distributions after t steps of random walks starting from i and j in order to classify nodes i and j as being similar or different. As discussed above, for very large values of t, all probability distributions will be very close to each other, but as will be shown and experimentally verified for reasonably small (constant) values of t, the probability distributions of pairs of nodes from the same cluster will converge more quickly than those of pairs from different clusters. This is the phenomenon that the algorithm exploits to recover the partitioning.

This paper gives a basis for providing theoretical guarantees for clustering algorithms used in practice. Szummer and Jaakkola show how to cluster a large number of points using a random walk given a small set of correctly clustered points [24]. They likewise observe that for t large, very little clustering information can be obtained from the walk since the distribution of points then depends solely on the graph's degree sequence. They go on to provide experimental evidence which suggests their random walk is optimal for small, constant t. In the planted partitions model that we assume, we show a related algorithm which we can prove to be optimal for $t = 2$ steps of a random walk.

3. THE PLANTED PARTITIONS MODEL

We approach this problem from the viewpoint of learning planted partitions from a mixture model. A *mixture model* is a partitioning of a set of nodes into clusters, and a probabilistic generative model for edges between nodes. Pairs of nodes from the same cluster have an edge between them with probability q, and pairs from different clusters have an edge with probability p, where $q > p$. The difference $q - p$ is known as the *probability gap*.

Let $\hat{A}$ be an n by n matrix of Bernoulli random variables, and $A = \mathbf{E}[\hat{A}]$ be the matrix of expectations. The mixture model may be represented by an n by n matrix of random variables $\hat{A}$ with expectation matrix A, where $A_{ij} = q$ for (i, j) in the same cluster, and $A_{ij} = p$ for (i, j) in different clusters. A has k *distinct* rows corresponding to the existence of k different clusters, and k blocks in the expectations matrix. Note that if we assume that the size s_{min} of the smallest block in the matrix is a constant fraction of n, then k is a constant. The randomized rounding of A to obtain $\hat{A}$ preserves symmetry: $\hat{A}_{ij} = \hat{A}_{ji}$. For convenience, we denote the cluster of a node i by $\Psi(i)$. In the analysis we will assume WLOG that A is a block diagonal matrix, and $\Psi(i)$ will refer to the cluster of i or the matrix block of i depending on context.

We receive as input from the real world the 0,1 similarity matrix $\hat{A}$, which we assume to be an instantiation of the matrix of random variables specified by the mixture model. Henceforth, we will refer to both the matrix of random variables and its instantiation as $\hat{A}$, and it should be clear from the context which is intended. Given this input matrix, and under this model, our goal is to partition the rows so that a pair of rows are placed in the same partition if and only if they belong to the same cluster. In other words, given $\hat{A}$, recover Ψ.

The clustering is easy to see in the expected matrix A. However, we are not given A, but rather a perturbed version of A through randomized rounding. Fortunately, this graph is not entirely random, as the desired partitions have been "planted" in some sense, by setting the probabilities appropriately according to the mixture model. Intuitively, we see that larger values of $q - p$ make the partitions easier to learn, as larger gaps cause similar points to be better connected relative to dissimilar points. Similarly, larger values of n make learning easier as we have more samples from which to learn.

3.1 Related Work

Previous work has made use of this mixture model or special cases of it. Boppana gave a spectral algorithm for the problem of graph bisection on randomly generated graphs, though he requires the solution to a convex optimization problem [5]. Blum and Spencer k-color a randomly generated k-colorable graph so long as $p \geq n^{\epsilon - 1}$. They also consider a semi-random model in which a graph generated by an adversary is subject to a small probability of toggling an edge [4]. Condon and Karp partition a random graph into k equal parts, minimizing the number of edges across parts with high probability, so long as $q - p \geq n^{-\frac{1}{2}+\epsilon}$ [8]. Jerrum and Sorkin resolve an open problem of Boppana and Bui by optimally bisecting a random graph with high probability so long as $q - p = \Omega(n^{\delta - 2})$, $\delta \leq 2$, via simulated annealing [17].

Finally, McSherry presents an algorithm [21] to learn a hidden partition in a random graph with high probability so long as $q - p = \Omega(n^{-\frac{1}{2}+\epsilon})$, for any $\epsilon > 0$. The procedure involves a randomized splitting of the columns into two parts and projecting on to the top singular vectors of each part to preserve certain independence properties.

3.2 Our Contribution

In contrast, we show that our algorithm can actually be implemented with only a single squaring of a matrix, and hence is much simpler than the SVD computation required in [21]. This, as well as our experimental evidence, suggests that our algorithm may be very useful in practice. Our simplicity does come at the cost of a slightly larger probability gap, as we require $q - p = \Omega(n^{-\frac{1}{4}})$.

Matrix squaring can be implemented to run in $O(n^{2.376})$ time using arithmetic progressions [9], and in $O(n^{2.7})$ time using the more practical Strassen's algorithm. This is significantly faster than the $O(n^3)$ time required to compute the SVD, though [21] can be implemented using a sublinear approximate SVD computation via the sampling algorithm of [1]. However, the constants and logarithmic terms of the running time of this approximate SVD(on the order of $11^6(\log n)^6$) seem rather impractical. Furthermore, it may be possible to speed up low-rank matrix multiplication via techniques similar to those in [1], though this is outside the scope of current work. Our algorithm is simple and elegant, and should be well suited to large data sets where the gap requirement of $\Omega(n^{-\frac{1}{4}})$ is easily satisfied.

Ultimately, we will provide the following guarantee that is our main result:

Theorem. *For $t = 2$, the matrix powering algorithm correctly clusters $1 - \delta$ of the rows with probability at least $\frac{1}{2}$, so long as $|q - p| > \frac{2\sqrt{q}(\frac{k^3}{\delta})^{\frac{1}{4}}}{n^{\frac{1}{4}}}$.*

We initially consider the special case of k equal sized blocks of size $s = \frac{n}{k}$ each. We will eventually show that the case of unequal blocks does not deviate too far from the case of equal blocks, and the asymptotics of the performance guarantees given remain the same so long as the minimum block size s_{min} is a constant fraction of n.

4. PERFORMANCE GUARANTEES

4.1 Proof of Main Theorem

We now proceed to show the clustering capability of the algorithm under this mixture model for $t = 2$. The strategy will be to show that the deviation of $\hat{A}_i^t$ from A_i^t is small relative to the distance between A_i^t and A_j^t, where i and j belong in different blocks. If so, then even after perturbation, rows from different clusters should remain well separated for large enough n. Specifically, if $||\hat{A}_i^t - A_i^t||^2 < \gamma$, and $||A_i^t - A_j^t||^2 \geq 16\gamma$, then

$$||\hat{A}_{i_1}^t - \hat{A}_{i_2}^t||^2 \leq (||\hat{A}_{i_1}^t - A_i^t|| + ||\hat{A}_{i_2}^t - A_i^t||)^2 < 4\gamma$$

and

$$\begin{aligned} ||\hat{A}_{i_1}^t - \hat{A}_{j_1}^t||^2 &\geq (||A_i^t - A_j^t|| - ||\hat{A}_{i_1}^t - A_i^t|| - ||\hat{A}_{j_1}^t - A_j^t||)^2 \\ &\geq (4\sqrt{\gamma} - \sqrt{\gamma} - \sqrt{\gamma})^2 \\ &= 4\gamma \end{aligned}$$

for $\Psi(i_1) = \Psi(i_2) = \Psi(i)$ and $\Psi(j_1) = \Psi(j) \neq \Psi(i)$. Thus, if we choose $\epsilon = ||A_i^t - A_j^t||^2/4 \geq 4\gamma$ to be our threshold in the algorithm, then we can cluster correctly in expectation.

First, we present a lemma that shows how block structure is preserved.

LEMMA 1. *Let A be a block diagonal matrix with equally sized blocks of size s, with entries of q_a within the blocks, and p_a without. Let B be a matrix with the same block structure and corresponding entries q_b and p_b. Then, AB has the same block structure with corresponding entries $q_{ab} = sq_aq_b + (n - s)p_ap_b$ and $p_{ab} = sq_ap_b + sq_bp_a + (n - 2s)p_ap_b$.*

PROOF. Let $\Psi(i)$ be the block corresponding to index i. It is clear from $(AB)_{ij} = \sum_k A_{ik}B_{kj} = \sum_k A_{ik}B_{jk}$ that $(AB)_{ij} = sq_aq_b + (n - s)p_ap_b$ when $\Psi(i) = \Psi(j)$, and $(AB)_{ij} = sq_ap_b + sq_bp_a + (n - 2s)p_ap_b$ when $\Psi(i) \neq \Psi(j)$. □

The following theorem calculates the separation between the rows of A^t from different blocks. In some sense, this is the expected "separation" between two rows belonging to different clusters.

THEOREM 2. $||A_i^t - A_j^t||^2 = 2(q - p)^{2t}(n/k)^{2t-1}$*, where* $\Psi(i) \neq \Psi(j)$

PROOF. By Lemma 1, A^t has the same block diagonal structure as A. Let q_t and p_t denote the entries inside and outside of the blocks of A^t, resp., so that $q_1 - p_1 = q - p$. We proceed inductively to show that $q_t - p_t = (q - p)^t s^{t-1}$. By Lemma 1, $q_t = sq_{t-1}q + (n - s)p_{t-1}p$ and $p_t = sq_{t-1}p + sp_{t-1}q + (n - 2s)p_{t-1}p = spq_{t-1} + (sq + (n-2s)p)p_{t-1}$. Hence, by the inductive hypothesis, $q_t - p_t = s(q-p)q_{t-1} - s(q-p)p_{t-1} = s(q-p)(q_{t-1} - p_{t-1}) = (q - p)^t s^{t-1}$. Thus, we know the gap $q_t - p_t$ in general, and this is all we need for the separation:

$$||A_i^t - A_j^t||^2 = 2s(q_t - p_t)^2 = 2(q-p)^{2t}s^{2t-1} = 2(q-p)^{2t}(n/k)^{2t-1},$$

which shows the claim. □

It is easy to see that $||A_i^t - A_j^t||^2 = 0$ when $\Psi(i) = \Psi(j)$, and in the more general case of unequal sized blocks, $||A_i^t - A_j^t||^2 \geq 2(q - p)^{2t}s_{min}^{2t-1}$, where s_{min} is the size of the smallest block.

The next lemma shows that, relative to the separation between rows from different clusters in A^2, the deviation of $\hat{A}_k^2$ from A_k^2 is small in expectation. Thus, the "error" from perturbation is bounded. The proof is deferred to the appendix and gives an exact analysis of this expectation.

LEMMA 3. $\mathbf{E}||\hat{A}_i^2 - A_i^2||^2 \leq 2q^2n^2$

We are now ready to prove the main theorem.

THEOREM 4. *For $t = 2$ and some fraction $\delta > 0$, the matrix powering algorithm correctly clusters $1-\delta$ of the rows with probability at least $\frac{1}{2}$, so long as $|q-p| > \frac{2\sqrt{q}(\frac{k^3}{\delta})^{\frac{1}{4}}}{n^{\frac{1}{4}}}$.*

PROOF. Simply let $\epsilon = (q-p)^4(n/k)^3/2$. Define a good row to be one for which $||\hat{A}_i^2 - A_i^2|| \leq 4q^2n^2/\delta$. Since $\mathbf{E}||\hat{A}_i^2 - A_i^2||^2 \leq 2q^2n^2$ by the above lemma, $\Pr[\text{row i is good}] \geq 1-\delta/2$ by applying Markov's Inequality to the bad event. Thus, in expectation, at least $1-\delta/2$ of the rows are good, and at most $\delta/2$ of the rows are bad. Again, by a Markov bound applied to the number of bad rows, at least $1-\delta$ of the rows are good with probability at least $\frac{1}{2}$. Now, we see that all the good rows will be classified correctly if $4q^2n^2/\delta < \epsilon/2 = (q-p)^4(n/k)^3/4$, which is equivalent to $|q-p| > \frac{2\sqrt{q}(\frac{k^3}{\delta})^{\frac{1}{4}}}{n^{\frac{1}{4}}}$. Thus, we can correctly cluster $1-\delta$ of the points with probability at least $\frac{1}{2}$ given this probability gap. □

The above analysis can be extended to success probabilities larger than $1/2$ by increasing the requirement on $|q-p|$.

4.2 Optimality of Squaring

A simple calculation shows that for t=1, the gap requirement is $\Omega(1)$, so that squaring does improve the clustering. Unfortunately, further powering of $\hat{A}$ does not improve the situation. In fact, we show that the gap requirement asymptotically increases due to a rapidly growing error. Specifically, we prove the following lemma:

LEMMA 5. $\mathbf{E}||\hat{A}_i^t - A_i^t||^2 = \Theta(n^{2t-2})$ *for all constants* $t \geq 2$.

PROOF. By the definition of matrix multiplication,

$$A_{ij}^t = \sum_{i_1,\cdots,i_{t-1}\in[n]} A_{ii_1}A_{i_1i_2}...A_{i_{t-1}j}$$

Thus,

$$\begin{aligned}
\mathbf{E}||\hat{A}_i^t - A_i^t||^2 &= \sum_{j=1}^{n} \mathbf{E}\left[(\hat{A}_{ij}^t - A_{ij}^t)^2\right] \\
&= \sum_{j=1}^{n} \left(\mathbf{E}\left[(\hat{A}_{ij}^t)^2\right] - 2\mathbf{E}\left[\hat{A}_{ij}^t\right]A_{ij}^t + (A_{ij}^t)^2\right) \\
&= \sum_{j=1}^{n} \sum_{k_1,\ldots,k_{t-1},k'_1,\ldots,k'_{t-1}\in[n]} \\
&\quad \mathbf{E}\left[\hat{A}_{ik_1}^t \cdots \hat{A}_{k_{t-1}j}^t \hat{A}_{ik'_1}^t \cdots \hat{A}_{k'_{t-1}j}^t\right] \\
&\quad -2\mathbf{E}\left[\hat{A}_{ik_1}^t \cdots \hat{A}_{k_{t-1}j}^t\right] A_{ik'_1}^t \cdots A_{k'_{t-1}j}^t \\
&\quad +A_{ik_1}^t \cdots A_{k_{t-1}j}^t A_{ik'_1}^t \cdots A_{k'_{t-1}j}^t \\
&\geq \sum_{j=1}^{n} \sum_{k_1,\ldots,k_{t-1},k'_1,\ldots,k'_{t-1}\in[n]} \\
&\quad \mathbf{E}\left[\hat{A}_{ik_1}^t \cdots \hat{A}_{k_{t-1}j}^t \hat{A}_{ik'_1}^t \cdots \hat{A}_{k'_{t-1}j}^t\right] \\
&\quad -\mathbf{E}\left[\hat{A}_{ik_1}^t \cdots \hat{A}_{k_{t-1}j}^t\right] A_{ik'_1}^t \cdots A_{k'_{t-1}j}^t
\end{aligned}$$

Observe that the above expression is a polynomial in n of degree at most $2t-1$. Also, the number of summands in the inner sum for which $|\{k_1,\ldots,k_{t-1},k'_1,\ldots,k'_{t-1}\}| = l$ is at most $\binom{n}{l}l^{2t-2} = \Theta(n^l)$. Hence, to compute the coefficient of n^{2t-1} in the above it suffices to consider only tuples $(k_1,\ldots,k_{t-1},k'_1,\ldots,k'_{t-1})$ for which

$$|\{k_1,\ldots,k_{t-1},k'_1,\ldots,k'_{t-1}\}| = 2t-2$$

In this case though, the expectations split completely so that the inner sum vanishes. It follows that the above is a polynomial in n of degree at most $2t-2$. To compute the coefficient of n^{2t-2}, it suffices to consider tuples $(k_1,\ldots,k_{t-1},k'_1,\ldots,k'_{t-1})$ for which $|\{k_1,\ldots,k_{t-1},k'_1,\ldots,k'_{t-1}\}| = 2t-3$, i.e., there is exactly one repetition.

Observe that the inner sum is always positive, so the above is at least:

$$\sum_{j=1}^{n} \sum_{|\{k_1,\ldots,k_{t-1},k'_1,\ldots,k'_{t-1}\}|=2t-3,\ k_1=k'_1} \mathbf{E}\left[\hat{A}_{ik_1}^t \cdots \hat{A}_{k_{t-1}j}^t \hat{A}_{ik'_1}^t \cdots \hat{A}_{k'_{t-1}j}^t\right] - \mathbf{E}\left[\hat{A}_{ik_1}^t \cdots \hat{A}_{k_{t-1}j}^t\right] A_{ik'_1}^t \cdots A_{k'_{t-1}j}^t,$$

which simplifies to

$$\sum_{j=1}^{n} \sum_{|\{k_1,\ldots,k_{t-1},k'_1,\ldots,k'_{t-1}\}|=2t-3,\ k_1=k'_1} (A_{ik_1}^t - (A_{ik_1}^t)^2)A_{k_1k_2}^t \cdots A_{k_{t-1}j}^t A_{k'_1k'_2}^t \cdots A_{k'_{t-1}j}^t$$

As long as $p, q = \Omega(1)$ and $\max\{p(1-p), q(1-q)\} = \Omega(1)$, each term in the inner sum is a positive constant. There are $\binom{n}{2t-3}(2t-3)! = \Theta(n^{2t-3})$ tuples for which $k_1 = k'_1$, so we have $\mathbf{E}||\hat{A}_i^t - A_i^t||^2 = \Omega(n^{2t-2})$, which completes the proof. □

From the analysis of theorem 4, this lemma implies that the gap requirement $|q-p|$ is $\Omega\left(\frac{1}{n}\right)^{\frac{1}{2t}}$, which is clearly optimal for $t=2$. This is supported by experimental evidence presented later.

4.3 Blocks of Different Sizes

Here, we justify the earlier claim that it suffices to consider blocks of equal sizes, and that blocks of different sizes do not alter the asymptotics of the performance guarantee by more than constant factors, so long as the minimum block size s_{min} is a constant fraction of n. For the separation, for $\Psi(i) \neq \Psi(j)$, we have seen that

$$||A_i^t - A_j^t||^2 \geq 2(q-p)^{2t}s_{min}^{2t-1}$$

It remains to consider the error for unequal blocks. We begin by proving a certain monotonicity property.

LEMMA 6. *Let A be the symmetric block diagonal matrix of expectations defined previously. Let B be the matrix obtained by symmetrically inserting a row and a column of fractional(probability) entries into A. Then, $\mathbf{E}||\hat{A}_i^t - A_i^t||^2 \leq \mathbf{E}||\hat{B}_i^t - B_i^t||^2$, where $\hat{A}$ and $\hat{B}$ are the randomized roundings of A and B, resp., preserving symmetry.*

PROOF. WLOG and for notational convenience, we may assume that we are inserting the last row and column. Let b_{ij} be the (i,j) entry of B and similarly $\hat{b}_{ij}$ for $\hat{B}$. We use $\mathbf{i}$ to denote a tuple $(i_1,\ldots,i_{t-1}) \in [n+1]^{t-1}$. For fixed starting index i, define $b(\mathbf{i},j) = b_{ii_1}b_{i_1i_2}...b_{i_{t-1}j}$, and similarly define $\hat{b}(\mathbf{i},j)$.

$$
\begin{aligned}
& \mathbf{E}||\hat{B}_i^t - B_i^t||^2 \\
= \; & \mathbf{E}\sum_j \sum_{\mathbf{i}} (\hat{b}(\mathbf{i},j) - b(\mathbf{i},j)) \sum_{\mathbf{i}'} (\hat{b}(\mathbf{i}',j) - b(\mathbf{i}',j)) \\
= \; & \mathbf{E}\sum_j \sum_{\mathbf{i},\, \mathbf{i}'} (\hat{b}(\mathbf{i},j) - b(\mathbf{i},j))(\hat{b}(\mathbf{i}',j) - b(\mathbf{i}',j)) \\
= \; & \mathbf{E}\sum_j \sum_{\mathbf{i},\, \mathbf{i}' \in [n]^{t-1}} (\hat{b}(\mathbf{i},j) - b(\mathbf{i},j))(\hat{b}(\mathbf{i}',j) - b(\mathbf{i}',j)) \\
& + \mathbf{E}\sum_j \sum_{\mathbf{i},\mathbf{i}' \in S} (\hat{b}(\mathbf{i},j) - b(\mathbf{i},j))(\hat{b}(\mathbf{i}',j) - b(\mathbf{i}',j)) \\
= \; & \mathbf{E}||\hat{A}_i^t - A_i^t||^2 \\
& + \mathbf{E}\sum_j \sum_{\mathbf{i},\mathbf{i}' \in S} (\hat{b}(\mathbf{i},j) - b(\mathbf{i},j))(\hat{b}(\mathbf{i}',j) - b(\mathbf{i}',j)),
\end{aligned}
$$

where S is the set of tuples $\mathbf{i}$ for which at least one index has value $n+1$. It remains to show that the second summand in the last equation is nonnegative.

$$
\begin{aligned}
& \mathbf{E}\sum_j \sum_{\mathbf{i},\mathbf{i}' \in S} (\hat{b}(\mathbf{i},j) - b(\mathbf{i},j))(\hat{b}(\mathbf{i}',j) - b(\mathbf{i}',j)) \\
= \; & \sum_j \sum_{\mathbf{i},\mathbf{i}' \in \mathbf{S}} \mathbf{E}[\hat{b}(\mathbf{i},j)\hat{b}(\mathbf{i}',j)] - \mathbf{E}[\hat{b}(\mathbf{i},j)b(\mathbf{i}',j)] \\
& - \mathbf{E}[b(\mathbf{i},j)\hat{b}(\mathbf{i}',j)] + \mathbf{E}[b(\mathbf{i},j)b(\mathbf{i}',j)] \\
\geq \; & \sum_j \sum_{\mathbf{i},\mathbf{i}' \in S} \mathbf{E}[\hat{b}(\mathbf{i},j)]\mathbf{E}[\hat{b}(\mathbf{i}',j)] - \mathbf{E}[\hat{b}(\mathbf{i},j)]b(\mathbf{i}',j) \\
& - b(\mathbf{i},j)\mathbf{E}[\hat{b}(\mathbf{i}',j)] + b(\mathbf{i},j)b(\mathbf{i}',j) \\
= \; & \sum_j \sum_{\mathbf{i},\mathbf{i}' \in S} (\mathbf{E}[\hat{b}(\mathbf{i},j)] - b(\mathbf{i},j))(\mathbf{E}[\hat{b}(\mathbf{i}',j)] - b(\mathbf{i}',j)) \\
\geq \; & 0,
\end{aligned}
$$

since for our Bernoulli variables, we have

$$\mathbf{E}[\hat{b}(\mathbf{i},j)\hat{b}(\mathbf{i}',j)] \geq \mathbf{E}[\hat{b}(\mathbf{i},j)]\mathbf{E}[\hat{b}(\mathbf{i}',j)]$$

and $\mathbf{E}[\hat{b}(\mathbf{i},j)] - b(\mathbf{i},j) \geq 0$. □

Let A be the original expectations matrix of unequal blocks. Let C be the matrix obtained from A by contracting each block to size s_{min}, and let B be obtained by expanding each block to size s_{max}. Note that we can symmetrically insert rows and columns to obtain B from A, and A from C. From the above lemma, we deduce that the errors increase monotonically:

$$\mathbf{E}||\hat{C}_i^t - C_i^t||^2 \leq \mathbf{E}||\hat{A}_i^t - A_i^t||^2 \leq \mathbf{E}||\hat{B}_i^t - B_i^t||^2$$

We know that the errors for equal sized blocks are polynomials in n of degree $\leq 2t$. Therefore,

$$\left(\frac{s_{max}}{s_{min}}\right)^{2t} \mathbf{E}||\hat{C}_i^t - C_i^t||^2 \geq \mathbf{E}||\hat{B}_i^t - B_i^t||^2$$

Since s_{min} is a constant fraction of n, $\frac{s_{max}}{s_{min}} = O(1)$. If t is also a constant, then this shows that the error for C is within a constant factor of the error for B, and hence the error for the matrix A of unequal blocks is also within a constant factor of the error for C, the matrix with equal blocks of size s_{min}. This yields the following theorem:

THEOREM 7. *Let A be a symmetric block-diagonal matrix of expectations of unequal blocks, where s_{min} is a constant fraction of n. Let C be obtained from A by contracting each block to size s_{min}. Then, for some constant r depending on the constant t,*

$$\mathbf{E}||\hat{A}_i^t - A_i^t||^2 \leq r\mathbf{E}||\hat{C}_i^t - C_i^t||^2$$

From this theorem, we may conclude that the asymptotics of the performance guarantees are unaffected by taking unequal blocks, and that the algorithm continues to work in this more general setting, for constant values of t.

5. EXPERIMENTS

We run two simulations to experimentally investigate the behavior of the matrix powering algorithm. First and foremost, we would like to examine the performance of the algorithm for different values of the power t. We generate the matrix $\hat{A}$ from the matrix A via randomized rounding preserving symmetry as specified by the mixture model with $q = 0.45$, $p = 0.05$, and $N = 200$ nodes divided evenly into 4 clusters. The success of the algorithm is measured by the percentage of the $\binom{N}{2}$ pairwise relationships (classified as same cluster or different) that it guesses correctly. Note that a score of 75% is not impressive and corresponds to the case where every node is classified to its own cluster. In the other extreme, a score of 25% corresponds to the case in which all of the nodes are classified to the same cluster. The results are shown in Figure 2, percentage correct against t.

Notice that the results basically conform to theoretical expectations, but the algorithm seems to perform unusually well for $t = 3$. We find this to be purely a matter of constant factors, as the power of q in the leading coefficient for the error of $t = 3$ is larger than the corresponding power for $t = 2$. Were q in our experiment much closer to 1 than 0.45, this effect would not be observed.

In addition, we would like to see how performance varies with probability gap, and to verify our intuition that clustering should become easier with larger gaps. We again instantiate the mixture model with N=200 nodes divided evenly into 4 clusters and p=0.05. We plot the percentage correct for t=3 against varying probability gaps($q - p$) in Figure 3.

In practice, implementing the algorithm requires that we know the threshold ϵ, which requires knowledge of $q - p$ and s_{min}, since $\epsilon = (q-p)^4 s_{min}^3/2$. Actually, since s_{min} is a constant fraction of n in our model, the guarantees continue to hold asymptotically if we simply estimate s_{min} by n. One might also imagine that the values of q and p, or just the gap $q - p$, are known to us via experimentation or some understanding of the underlying planted partitions model specific to the particular application. However, if ϵ is unknown to us, we may try a binary search for the correct value ranging between 1 and n^3, incurring an additional $O(\log n)$ factor in running time, and take the best clustering found. This assumes that we have some application specific way of evaluating the quality of a clustering, and is independent of the planted partitions model.

Finally, if new points arrive in an online fashion, one can update the clusters in time $O(n^2)$ incrementally without computing another matrix multiplication.

6. FUTURE DIRECTIONS

The matrix powering algorithm successfully clusters a large portion of the nodes with good probability given a probability gap of $\Omega(n^{-\frac{1}{4}})$. It is simple, elegant, and runs as fast as a single matrix multiplication. The matrix powering algorithm is independent of the mixture model, and applicable to any matrix of similarities. An interesting direction of future research might be to investigate under

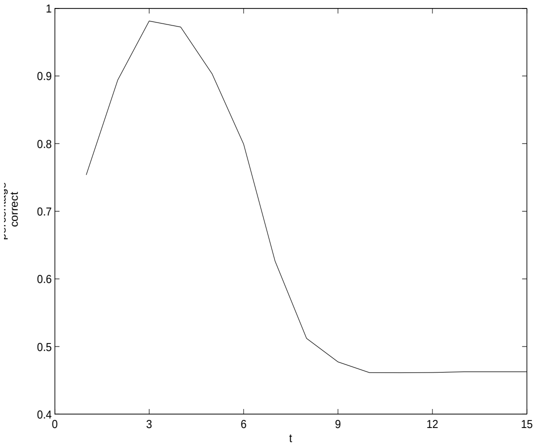

Figure 2: percentage correct vs t

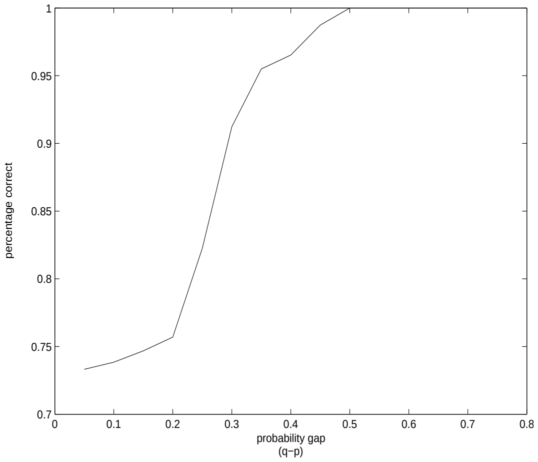

Figure 3: percentage correct vs gap

which other models this algorithm performs well. As our algorithm is likely practical and potentially of high-impact, more involved implementations on typical datasets in various applications should be explored.

7. ACKNOWLEDGMENTS

We would like to thank Santosh Vempala for problem introduction and guidance. Josh Tauber and John Kelner were also helpful in providing tips on the exposition and insightful discussions.

8. REFERENCES

[1] D. Achlioptas and F. McSherry. Fast computation of low rank matrix approximations. In *ACM Symposium on Theory of Computing*, 2001.

[2] S. Arora, P. Raghavan, and S. Rao. Approximation schemes for euclidean k -medians and related problems. In *ACM Symposium on Theory of Computing*, pages 106–113, 1998.

[3] Y. Azar, A. Fiat, A. Karlin, and F. McSherry. Data mining through spectral analysis. In *IEEE Symposium on Foundations of Computer Science*, 2001.

[4] A. Blum and J. Spencer. Coloring random and semi-random k-colorable graphs. In *Journal of Algorithms*, 1995.

[5] R. Boppana. Eigenvalues and graph bisection: An average-case analysis. In *IEEE Symposium on Foundations of Computer Science*, pages 280–285, 1985.

[6] M. Charikar and S. Guha. Improved combinatorial algorithms for the facility location and k-median problems. In *IEEE Symposium on Foundations of Computer Science*, pages 378–388, 1999.

[7] M. Charikar, S. Guha, E. Tardos, and D. B. Shmoys. A constant-factor approximation algorithm for the k -median problem (extended abstract). In *ACM Symposium on Theory of Computing*, pages 1–10, 1999.

[8] A. Condon and R. Karp. Algorithms for graph partitioning on the planted partition model. In *Random Structure and Algorithms*, 1999.

[9] D. Coppersmith and S. Winograd. Matrix multiplication via arithmetic progressions, 1990.

[10] I. S. Dhillon, S. Mallela, and D. S. Modha. Information-theoretic co-clustering. In *9th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2003.

[11] Z. Drezner. *Facility Location: A survey of Applications and Methods*. Springer-Verlag, 1995.

[12] P. Drineas, A. Frieze, R. Kannan, S. Vempala, and V. Vinay. Clustering in large graphs and matrices. In *ACM-SIAM Symposium on Discrete Algorithms*, 1999.

[13] M. Dyer and A. M. Frieze. A simple heuristic for the p-center problem. *Operations Research Letters*, 3:285– 288, 1985.

[14] D. S. Hochbaum and D. B. Shmoys. A best possible approximation algorithm for the k-center problem. *Math. Oper. Res.*, 10:180–184, 1985.

[15] K. Jain and V. V. Vazirani. Primal-dual approximation algorithms for metric facility location and k-median problems. In *IEEE Symposium on Foundations of Computer Science*, pages 2–13, 1999.

[16] M. Jambu and M. O. Lebeaux. *Cluster Analysis and Data Analysis*. North-Holland, New York, 1983.

[17] M. Jerrum and G. Sorkin. Simulated annealing for graph bisection. In *IEEE Symposium on Foundations of Computer Science*, 1993.

[18] R. Kannan, S. Vempala, and A. Vetta. On clusterings: Good, bad, and spectral. In *IEEE Symposium on Foundations of Computer Science*, 2000.

[19] J. Kleinberg, C. Papadimitriou, and P. Raghavan. Segmentation problems. In *(ACM) Symposium on Theory of Computing*, pages 473–482, 1998.

[20] L. Lovasz. Random walks on graphs: a survey. In *Combinatorics*, pages 1–46, 1993.

[21] F. McSherry. Spectral partitioning of random graphs. In *IEEE Symposium on Foundations of Computer Science*, 2001.

[22] R. Ostrovsky and Y. Rabani. Polynomial time approximation schemes for geometric k-clustering. In *IEEE Symposium on Foundations of Computer Science*, pages 349–358, 2000.

[23] C. Papadimitriou, P. Raghavan, H. Tamaki, and S. Vempala. Latent semantic indexing: a probabilistic analysis. In *ACM Conference on Principles of Database Systems*, 1998.

[24] M. Szummer and T. Jaakkola. Partially labeled classification with markov random walks. In *Advances in Neural Information Processing Systems*, volume 14, 2001.

[25] S. Virtanen. Clustering the chilean web. In *LA-WEB*, 2003.

APPENDIX

A. PROOF OF LEMMA 3 (ERROR BOUND)

PROOF.

$$\begin{aligned}
\mathbf{E}||\hat{A}_i^2 - A_i^2||^2 &= \mathbf{E}\left[\sum_m (\sum_l \hat{a}_{il}\hat{a}_{lm} - a_{il}a_{lm})^2\right] \\
&= \sum_m \sum_{l_1,l_2} \mathbf{E}\left[(\hat{a}_{il_1}\hat{a}_{l_1m} - a_{il_1}a_{l_1m})(\hat{a}_{il_2}\hat{a}_{l_2m} - a_{il_2}a_{l_2m})\right] \\
& \quad \text{the product terms in the expectation are dependent when } l_1 = l_2 \\
& \quad \text{ OR if } m \neq i \text{ and } (l_1, l_2) = (m, i) \text{ or } (i, m) \\
&= \sum_m \sum_{\substack{l_1 \neq l_2 \\ (l_1,l_2)\neq(m,i) \\ (l_1,l_2)\neq(i,m)}} \mathbf{E}[\hat{a}_{il_1}\hat{a}_{l_1m} - a_{il_1}a_{l_1m}]\mathbf{E}[\hat{a}_{il_2}\hat{a}_{l_2m} - a_{il_2}a_{l_2m}] \\
& \quad + \sum_m \sum_{l_1=l_2} \mathbf{E}[(\hat{a}_{il}\hat{a}_{lm} - a_{il}a_{lm})^2] \\
& \quad + \sum_{\substack{m\neq i \\ (l_1,l_2)=(m,i)\text{or}(i,m)}} \mathbf{E}[(\hat{a}_{il_1}\hat{a}_{l_1m} - a_{il_1}a_{l_1m})(\hat{a}_{il_2}\hat{a}_{l_2m} - a_{il_2}a_{l_2m})] \\
& \quad \text{Note that } \hat{a}_{il} = \hat{a}_{lm} \text{ only when m=i} \\
&= \sum_{l_1 \neq l_2} (a_{il_1} - a^2_{il_1})(a_{i_l2} - a^2_{i_l2}) + \sum_m \sum_l \mathbf{E}[\hat{a}^2_{il}\hat{a}^2_{lm} - 2a_{il}a_{lm}\hat{a}_{il}\hat{a}_{lm} + a^2_{il}a^2_{lm}] \\
& \quad +2\sum_{m\neq i} \mathbf{E}[(\hat{a}_{im}\hat{a}_{mm} - a_{im}a_{mm})(\hat{a}_{ii}\hat{a}_{im} - a_{ii}a_{im})] \\
&= \sum_{l_1,l_2} a_{il_1}(1 - a_{il_1})a_{il_2}(1 - a_{il_2}) - \sum_l a^2_{il}(1 - a_{il})^2 + \sum_m \sum_l \mathbf{E}[\hat{a}^2_{il}\hat{a}^2_{lm}] \\
& \quad -2\sum_m \sum_l a_{il}a_{lm}\mathbf{E}[\hat{a}_{il}\hat{a}_{lm}] + \sum_m \sum_l a^2_{il}a^2_{lm} \\
& \quad +2\sum_{m\neq i} a_{ii}a_{mm}a_{im} - 2a_{ii}a_{mm}a^2_{im} + a_{ii}a_{mm}a^2_{im}
\end{aligned}$$

We are now in position to expand out all of the expectations.

$$\begin{aligned}
\mathbf{E}||\hat{A}_i^2 - A_i^2||^2 &= \left(\sum_l a_{il}(1 - a_{il})\right)^2 - \sum_l a^2_{il}(1 - a_{il})^2 + \sum_m \sum_l a_{il}a_{lm} - \sum_l a^2_{il} + \sum_l a_{il} \\
& \quad -2(\sum_m \sum_l a^2_{il}a^2_{lm} - \sum_l a^4_{il} + \sum_l a^3_{il}) + \sum_m \sum_l a^2_{il}a^2_{lm} \\
& \quad +2\sum_{m\neq i} a_{ii}a_{mm}a_{im}(1 - a_{im}) \\
&= \left(\sum_l a_{il}(1 - a_{il})\right)^2 - \sum_l a^2_{il}(1 - a_{il})^2 + \sum_m \sum_l a_{il}a_{lm} + \sum_l a_{il}(1 - a_{il}) \\
& \quad -\sum_m \sum_l a^2_{il}a^2_{lm} - 2\sum_l a^3_{il}(1 - a_{il}) \\
& \quad +2[(\sum_m a_{ii}a_{mm}a_{im}(1 - a_{im})) - a^3_{ii}(1 - a_{ii})] \\
&= (sq(1-q) + (n-s)p(1-p))^2 - (sq^2(1-q)^2 + (n-s)p^2(1-p)^2) \\
& \quad +(sq + (n-s)p)^2 + sq(1-q) + (n-s)p(1-p) - (sq^2 + (n-s)p^2)^2 \\
& \quad -2(sq^3(1-q) + (n-s)p^3(1-p)) \\
& \quad +2(q^2(sq(1-q) + (n-s)p(1-p)) - q^3(1-q)) \\
&\leq 2q^2n^2 \text{ for large enough } n
\end{aligned}$$