Clustering in large graphs and matrices

P. Drineas* Alan Frieze† Ravi Kannan‡ Santosh Vempala§ V. Vinay¶

Appears in the Proceedings of the Symposium on Discrete Algorithms, SIAM, 1999

Abstract

We consider the problem of dividing a set of m points in Euclidean $n-$space into k clusters (m, n are variable while k is fixed), so as to minimize the sum of distances squared of each point to its "cluster center". This formulation differs in two ways from the most frequently considered clustering problems in the literature, namely, here we have k fixed and m, n variable, and we use the sum of squared distances as our measure; we will argue that our problem is natural in many contexts.

We consider a relaxation of the discrete problem: find the $k-$dimensional subspace V so that the sum of distances squared to V (of the m points) is minimized. We show: (i) The relaxation can be solved by the Singular Value Decomposition (SVD) of Linear Algebra. (ii) The solution of the relaxation can be used to get a 2-approximation algorithm for the original problem. More importantly, (iii) we argue that in fact the relaxation provides a generalized clustering which is useful in its own right. Finally, (iv) we show that the SVD of a randomly chosen submatrix (according to a suitable probability distribution) of the matrix provides an approximation to the SVD of the whole matrix, thus yielding a very fast randomized algorithm. This can be applied to problems of very large size which typically arise in modern applications.

1 Introduction

We consider in this paper the problem of clustering the rows of a given $m \times n$ matrix A - i.e., the problem of dividing up the set of rows into k clusters where each cluster has "similar" rows. Our notion of similarity of two rows (to be discussed in detail below) will be a function of the length of the vector difference of the two rows. So, equivalently, we may view the problem geometrically - i.e., we are given m points in Euclidean n-space and we wish to divide them up into k clusters, where each cluster contains points which are "close to each other". This problem includes as a special case the problem of clustering the

*Computer Science Department, Yale University, New Haven, CT06511. Email: **drineas@cs.yale.edu**.

†Department of Mathematical Sciences, Carnegie Mellon University, Pittsburgh, PA15213. Email: **alan@random.math.cmu.edu**. Supported in part by NSF grant CCR-9530974.

‡Computer Science Department, Yale University, New Haven, CT06511. Email: **kannan@cs.yale.edu**.

§Department of Mathematics, M.I.T., Cambridge, MA02139, and University of California, Berkeley. Email: **vempala@math.mit.edu**.

¶Indian Institute of Science, Bangalore, India. Email: **vinay@csa.iisc.ernet.in**.

vertices of a (directed or undirected) graph where the matrix is just the adjacency matrix of the graph. Here the similarity of two vertices depends on the number of common neighbors.

There are many notions of similarity and many notions of what a good clustering is. In general clustering problems turn out to be NP-hard. In some cases, there are polynomial-time approximation algorithms. Our aim here is to deal with very large matrices (with upwards of 10^5 rows and columns and upwards of a million non-zero entries), where a polynomial time bound on the algorithm is not of much value. Formally, we deal with the case where m, n vary and k, the number of clusters is fixed. We seek linear time algorithms with small constants.

We will show that the basic Singular Value Decomposition (SVD) of Linear Algebra provides us with an excellent tool. We will first show that SVD helps us solve approximately the clustering problems described in the abstract. More importantly, we argue that the SVD itself directly gives us what we call a "continuous clustering" - where each point will belong to a cluster with a certain "intensity" and clusters are not necessarily disjoint. Using basic Linear Algebra, we show several natural properties of this continuous clustering which we argue are useful for our problems. Then we develop a linear time randomized algorithm for approximate SVD which makes the procedure feasible for the very large matrices in modern applications.

1.1 Discrete Clustering.

First consider the following clustering problem: we are given m points $\mathcal{A} = \{A^{(1)}, A^{(2)}, \ldots A^{(m)}\}$ in n-dimensional Euclidean space and a positive integer k where k will be considered to be fixed as m, n vary. The problem is to find k points $\mathcal{B} = \{B^{(1)}, B^{(2)}, \ldots, B^{(k)}\}$ such that

$$f_{\mathcal{A}}(\mathcal{B}) = \sum_{i=1}^{m} (\mathrm{dist}(A^{(i)}, \mathcal{B}))^2$$

is minimized. Here $\mathrm{dist}(A^{(i)}, \mathcal{B})$ is the (Euclidean) distance of $A^{(i)}$ to its nearest point in $\mathcal{B}$. Thus, in this problem we wish to minimize the sum of squared distances to the nearest "cluster center". We call this the "Discrete Clustering Problem" (DCP). The DCP is NP-hard even for $k = 2$ (via a reduction from minimum bisection).

Note that this defines k clusters S_j, $j = 1, 2, \ldots k$. The cluster center $B^{(j)}$ will be the centroid of the points in S_j, $j = 1, 2, \ldots, k$. This is easily seen from the fact that for any set $\mathcal{S} = \{X^{(1)}, X^{(2)}, \ldots, X^{(r)}\}$ and any point B we have

$$\sum_{i=1}^{r} |X^{(i)} - B|^2 = \sum_{i=1}^{r} |X^{(i)} - \bar{X}|^2 + r|B - \bar{X}|^2, \tag{1}$$

where $\bar{X}$ is the centroid $(X^{(1)} + X^{(2)} + \cdots + X^{(r)})/r$ of $\mathcal{S}$.

The DCP is thus the problem of partitioning a set of points into clusters so that the *sum of the variances of the clusters* is minimized.

We define a relaxation which we call the *Continuous Clustering Problem* (CCP), as the problem of finding the subspace V of R^n of dimension at most k which minimizes

$$g_{\mathcal{A}}(V) = \sum_{i=1}^{m} \mathrm{dist}(A^{(i)}, V)^2.$$

It is easy to see that the optimal value of DCP is an upper bound for the optimal value of the CCP. Indeed for any set $\mathcal{B}$ of k points,

$$f_{\mathcal{A}}(\mathcal{B}) \geq g_{\mathcal{A}}(V_{\mathcal{B}}) \tag{2}$$

where $V_{\mathcal{B}}$ is the subspace generated by the points in $\mathcal{B}$.

It will follow from standard Linear Algebra that the continuous clustering problem can be solved in polynomial time exactly since the optimal subspace can be read off from the Singular Value Decomposition (SVD) of the matrix A containing $A^{(1)}, A^{(2)}, \ldots A^{(m)}$ as its rows, see Section **??**. Also, we can attempt to solve DCP as follows: we first solve CCP to find a subspace V; then we project the problem to V and now solve the discrete clustering problem in the k-dimensional space (where now k is fixed). we will show that the k-dimensional problem can be solved exactly in polynomial time and this will give us a 2-approximation algorithm for DCP. We will discuss this in Section **??**, but such an approximation algorithm is not our focus.

1.2 Fast SVD.

As stated earlier, our focus is two-fold. Firstly, we will give a very fast randomized algorithm which finds an approximation D^* to A of rank at most k, satisfying (with high probability)

$$||A - D^*||_F^2 \leq \min_{D:\mathrm{rank}(D)\leq k} ||A - D||_F^2 + \epsilon ||A||_F^2 \tag{3}$$

where

$$||A||_F^2 = \sum_{i=1}^{m} \sum_{j=1}^{n} A_{i,j}^2.$$

is the sum of squares of all the entries in the matrix and $\varepsilon > 0$ is a given error parameter. Thus the D^* found is nearly the best rank k approximation to A in the sense described above. (It is known that D^* satisfying (**??**) with even $\epsilon = 0$ can be read off from the SVD and thus such a D^* can be found in polynomial time.)

After making one pass through the entire matrix, our algorithm will run in time

$$O(\frac{k^3}{\epsilon^6} + \frac{k^2}{\epsilon^4} r),$$

where r is the maximum number of non-zero entries in any column of A. [In fact, r be replaced by the average number of entries in a column according to the probability distribution used in the algorithm. (see section **??**).

Instead of computing the SVD of the entire matrix A, we show that it is sufficient to compute the SVD of a matrix consisting of $O(\frac{k}{\epsilon^2})$ randomly picked columns of A; the columns have to be randomly picked according to a distribution satisfying some conditions (see section **??** for details.) We will also see that though we can only prove that if $O(\frac{k}{\epsilon^2})$ columns are picked, then (**??**) holds with high probability, we may pick fewer rows in practice, then check (in a randomized way) whether (**??**) holds; this lets us use less time than the worst case bound if the problem is not worst case.

(We note that [**?**] gives an algorithm which also achieves (**??**), but there the running time is $O(k^{12}/\varepsilon^9)$ which is prohibitively large for even modest values of $k, 1/\varepsilon$. We follow many ideas from that paper. Note that their algorithm has a running time which does not grow with m, n while ours does, but only linearly with m.)

Clearly an approximation of the form (**??**) is only useful if A has a good approximation of "small" rank k and further m, n are large (so exact algorithms are not feasible.) There are many examples of situations where these conditions prevail; we describe briefly two typical examples.

1.2.1 Latent Semantic Indexing.

This is a general technique for analyzing a collection of "documents" which are assumed to be related (for example, they are all documents dealing with a particular subject; see [**?**]). Suppose there are m documents and n "terms" which occur in the documents. The model hypothesizes that (because there are relationships among the documents), there are a small number k of main (unknown) "topics" which the documents are about. The first aim of the technique is to find a set of k topics which best describe the documents. (This is the only part which concerns us here.)

A topic is modeled as an $n-$vector of non-negative reals where the interpretation is that the jth component of a topic vector gives the frequency with which the jth term occurs in (a discussion of) the topic. With this model on hand, it is easy to argue (using Linear Algebra and a line of reasoning similar to the field of "Factor Analysis" in Statistics) that the k best topics are the top k singular vectors of the so-called "document-term" matrix, which is an $m \times n$ matrix A with A_{ij} being the frequency of the jth term in the ith document. Alternatively, one can define A_{ij} as 0 or 1 depending upon whether the jth term occurs in the ith document.

The second example has to do with clustering documents on the web which we discuss in section **??**.

1.3 Generalized Clustering.

The second main focus of this paper is to argue that the approximately best rank k approximation to A yields a "generalized clustering" of A. A "generalized" clustering differs in two respects from normal clustering - first each cluster instead of being a subset (or equivalently a $m-$vector with 0 or 1 components) is a $m-$vector of reals where the ith component gives the "intensity" with which the ith point belongs to the cluster and secondly the requirement that the clusters be disjoint in the discrete clustering is replaced by a requirement that the vectors corresponding to the different clusters be orthogonal. We will see why this notion of clustering is quite natural and allows for some desirable features not allowed by discrete clustering - like having overlapping clusters which is often realistic in practice.

1.4 Review of related literature.

Clustering and location are well-studied subjects and there are many notions of how to measure the effectiveness of a cluster. We discuss them in relation to what we defined above as the problem considered in this paper. See Agarwal and Sharir [?], Bern and Eppstein [?], Drezner [?], Everitt [?], Jambu and Lebeaux [?], Lorr [?], Tryon and Bailey [?] for surveys and [?] for some recent related work.

One traditional clustering problem is the "$k-$center" problem. Here, one defines the cluster size to be the maximum pairwise distance between two points in the cluster and we wish to minimize the maximum cluster size among all divisions into k clusters. In the "$k-$median"problem, we again have to partition into k clusters; we also have to find for each cluster a "cluster center" or median. The optimal solution minimizes the sum of distances of each point to the median of its cluster. The $k-$center problem is easier in that there are constant factor approximation algorithms known for it, Dyer and Frieze [?] and Hochbaum and Shmoys [?]. For the $k-$median problem, good approximation algorithms have been harder to come by. Most notably, Arora, Raghavan and Rao [?] have only very recently discovered Polynomial Time Approximation Scheme when the points are located in the Euclidean plane. (or more generally in a fixed dimensional space) We note that in our problem, the number of dimensions is variable whereas the number of clusters is fixed.

1.5 Notation.

Every real matrix A can be expressed as

$$A = \sum_{t=1}^{r} \sigma_t(A) u^{(t)} v^{(t)^T}$$

where $\sigma_1(A) \geq \sigma_2(A) \geq \ldots \geq \sigma_r(A) \geq 0$ are called the singular values of A and the $u^{(t)}$ form an orthonormal set of vectors and so do the $v^{(t)}$. Also $u^{(t)^T} A = \sigma_t v^{(t)^T}$ and $Av^{(t)} = \sigma_t u^{(t)}$ for $1 \leq t \leq r$. This is the *singular value decomposition* of A.

¿From Linear Algebra, [?] we know that the matrix D_k producing the minimum η of $||A - D||_F$ among all matrices D of rank k or less is given by

$$D_k = \sum_{t=1}^{k} u^{(t)} u^{(t)^T} A = \sum_{t=1}^{k} A v^{(t)} v^{(t)^T}.$$

This implies that

$$\eta = \sum_{t=k+1}^{r} \sigma_t^2.$$

2 The Discrete Clustering Problem

We first show how to solve DCP in $O(m^{k^2 d/2})$ time when the input $\mathcal{A} \subseteq R^d$ – here k, d are considered to be fixed. Each set $\mathcal{B}$ of "cluster centers" defines a Voronoi diagram where cell $\mathcal{C}_i = \{X \in R^d : |X - B^{(i)}| \leq |X - B^{(j)}| \text{ for } j \neq i\}$ consists of those points whose closest point in $\mathcal{B}$ is $B^{(i)}$. Each cell is a polyhedron and the total number of faces in $C_1, C_2, \ldots, C_k$ is no more than $\binom{k}{2}$ (since each face is the set of points equidistant from two points of $\mathcal{B}$).

We have seen in (**??**) that it is the partition of $\mathcal{A}$ that determines the best $\mathcal{B}$ (via computation of centroids) and so we can move the boundary hyperplanes of the optimal Voronoi diagram, without any face passing through a point of $\mathcal{A}$, so that each face contains at least d points of $\mathcal{A}$.

Assume that the points of $\mathcal{A}$ are in general position and $0 \notin \mathcal{A}$ (a simple perturbation argument deals with the general case). This means that each face now contains d affinely independent points of $\mathcal{A}$. We have lost the information about which side of each face to place these points and so we must try all possibilities for each face. This leads to the following enumerative procedure for solving DCP:

Algorithm for DCP in d dimensions

- Enumerate all $\sum_{t=k}^{\binom{k}{2}} \binom{\binom{m}{d}}{t} = O(m^{dk^2/2})$ sets of $k \leq t \leq k(k-1)/2$ hyperplanes, each of which contains d affinely independent points of $\mathcal{A}$.
- Check that the arrangement defined by these hyperplanes has exactly k cells.
- Make one of 2^{td} choices as to which cell to assign each point of $\mathcal{A}$ which is lying on a hyperplane
- This defines a unique partition of $\mathcal{A}$. Find the centroid of each set in the partition and compute $f_{\mathcal{A}}$.

As remarked previously, CCP can be solved by Linear Algebra. Indeed, let V be a k-dimensional subspace of R^n and $\bar{A}^{(1)}, \bar{A}^{(2)}, \ldots, \bar{A}^{(m)}$ be the orthogonal projections of $A^{(1)}, A^{(2)}, \ldots, A^{(m)}$ onto V. Let $\bar{A}$ be the $m \times n$ matrix with rows $\bar{A}^{(1)}, \bar{A}^{(2)}, \ldots, \bar{A}^{(m)}$. Thus $\bar{A}$ has rank at most k and

$$||A - \bar{A}||_F^2 = \sum_{i=1}^{m} |A^{(i)} - \bar{A}^{(i)}|^2 = \sum_{i=1}^{m} (\text{dist}(A^{(i)}, V))^2.$$

Thus to solve CCP, all we have to do is find the first k vectors of the SVD of A (since it is known that these minimize $||A - \bar{A}||_F^2$ over all rank k matrices $\bar{A}$) and take the space V_{SVD} spanned by the first k singular vectors in the row space of A.

We now show that combining the above two ideas gives a 2-approximation to DCP. Let $\bar{\mathcal{A}} = \{\bar{A}^{(1)}, \bar{A}^{(2)}, \ldots, \bar{A}^{(m)}\}$ be the projection of $\mathcal{A}$ onto the subspace V_{SVD} above. Let $\bar{\mathcal{B}} = \{\bar{B}^{(1)}, \bar{B}^{(2)}, \ldots, \bar{B}^{(k)}\}$ be the optimal solution to DCP with input $\bar{\mathcal{A}}$.

Algorithm for general DCP

- Compute V_{SVD}.
- Solve DCP with input $\bar{\mathcal{A}}$ to obtain $\bar{\mathcal{B}}$.
- Output $\bar{\mathcal{B}}$.

It follows from (**??**) that the optimal value $Z_{\mathcal{A}}^{DCP}$ of the DCP satisfies

$$Z_{\mathcal{A}}^{DCP} \geq \sum_{i=1}^{m} |A^{(i)} - \bar{A}^{(i)}|^2. \tag{4}$$

Note also that if $\hat{\mathcal{B}} = \{\hat{B}^{(1)}, \hat{B}^{(2)}, \ldots, \hat{B}^{(k)}\}$ is an optimal solution to the DCP and $\bar{\mathcal{B}}$ consists of the projection of the points in $\hat{\mathcal{B}}$ onto V, then

$$Z_{\mathcal{A}}^{DCP} = \sum_{i=1}^{m} \mathrm{dist}(A^{(i)}, \hat{\mathcal{B}})^2 \geq \sum_{i=1}^{m} \mathrm{dist}(\bar{A}^{(i)}, \bar{\mathcal{B}})^2.$$

Combining this with (**??**) we get

$$\begin{aligned} 2Z_{\mathcal{A}}^{DCP} &\geq \sum_{i=1}^{m} (|A^{(i)} - \bar{A}^{(i)}|^2 + \mathrm{dist}(\bar{A}^{(i)}, \bar{\mathcal{B}})^2) \\ &= \sum_{i=1}^{m} \mathrm{dist}(A^{(i)}, \bar{\mathcal{B}})^2 \\ &= f_{\mathcal{A}}(\bar{\mathcal{B}}) \end{aligned}$$

proving that we do indeed get a 2-approximation. □

3 Generalized Clusters and SVD

In this section, we will argue that there is a natural way of generalizing clusters which leads to singular vectors. To do this, we introduce a typical motivation for this paper. Suppose we wish to analyze the structure of a large portion of the web. Consider the underlying directed graph with one vertex per URL and a edge from vertex i to vertex j if there is a hypertext link from i to j. It turns out to be quite useful to cluster the vertices of this graph. But obviously very large graphs can arise in this application and traditional heuristics (even polynomial time ones) are not good enough. We examine this particular application in some more detail. So we have a directed graph $G(V,E)$ here and we wish to divide the vertex set into "clusters" of "similar" vertices. Since all our information is in the graph, two vertices are "similar" if they share a lot of common neighbors. More generally, the assumption we make is that all the relevant information is captured by the matrix A given to us. We do not dwell on the "modeling" part - the translation of the real problem into the matrix; we assume this has been done already for us.

Going back to the example of clustering the vertices of the graph into clusters of similar vertices, we will examine how "similarity" may be precisely defined. It is useful for this purpose to think of the web example - here an edge from i to j means that i thinks of j as important. So, intuitively, similar vertices "reinforce" each others opinions of what documents are important.

Often by "clustering", one means the partition of the node set into subsets of similar nodes. But a partition is too strict, because it is quite common to have overlapping clusters. Also in traditional clustering, one finds subsets; by going to characteristic vectors, one may view them as 0-1 vectors. But this again is too strict. Different nodes may belong to a cluster with different "intensities". For example, if $N(v)$ (the set of neighbors of v) is large and there are many nodes u such that $N(u)$ is a subset of $N(v)$, a good cluster intuitively would include v and many of the u 's (for reinforcement) but again intuitively, v is more important in the cluster than the u's.

So, we define a cluster x as just an $m-$vector of reals. [So, $x(u)$ is the "intensity" with which u belongs to x.] What should we assign as the "weight" or importance of the cluster x ? A crucial quantity in this regard is the vector x^TA because $(x^TA)_i$ is the "frequency" of occurrence of node i in the neighborhood of the cluster x. So high values of $|(x^TA)_i|$ mean high reinforcement. So,

$$\sum_{i=1}^{n}(x^TA)_i^2 = |x^TA|^2$$

is a measure of the importance of the cluster. We note also that if x is scaled by some λ, so is every component of x^TA. So we make the following central definition:

DEFINITION A *cluster* (of A) is an $m-$vector x with $|x| = 1$. The weight of the cluster x, denoted $W(x)$ is $|x^TA|$.

[Here $|x|$ denotes the Euclidean length.]

Reasoning behind the Definition

Why Euclidean lengths? While we cannot exhaustively discuss all other possible measures, we look at the two other obvious norms possible; (l_∞ and l_1) the examples illustrate the advantage of Euclidean norm over these also carry over for many other norms.

We have used the Euclidean norm for both x and x^TA. First suppose we used instead l_1 norm for x. Then if there are k nodes of G all with the same neighborhood set, putting $x_i = 1$ for one of them and zero for the others as well as putting $x_i = 1/k$ for each gives us the same x^TA and so the same weight. However, we prefer larger clusters (greater reinforcement). It is easy to see that if we restrict to $|x| = 1$, then we would choose the larger cluster. Similarly if the l_∞ is used for x, then we will always have $x_i = 1$ for all i being the maximum weight cluster, which obviously is not always a good choice. It is also easy to see that if the l_∞ norm is used for $|x^T A|$, then x will be based only on the highest in-degree node which is not always desirable. A similar example can be provided for the case when the l_1 norm is used for $x^T A$.

Having defined the weight of a cluster, we next want to describe the decomposition process mentioned earlier which successively removes the maximum weight cluster from the graph. Let u be the maximum weight cluster and v any other cluster. If we write v as $v = \lambda u + w$ where λ is a scalar and w is orthogonal to u, it is known from basic Linear Algebra that $w^T A$ is also orthogonal to $u^T A$, so $|v^T A|^2 = \lambda^2 |u^T A|^2 + |w^T A|^2$. So the larger the λ the higher the weight of v. So if we only require v to be different from u, it may be arbitrarily close to u. This leads us to the correct requirement - namely that v be (required to be) orthogonal to u. [Orthogonality replaces disjointness in the traditional partition.] With this, we make another crucial definition:

DEFINITION An optimal clustering of A is a set of orthonormal vectors $x^{(1)}, x^{(2)}, \ldots$ so that $x^{(i)}$ is a maximum weight cluster of A subject to being orthogonal to $x^{(1)}, \ldots x^{(i-1)}$.

We will argue (directly from Linear Algebra) in the final paper that corresponding to "removing" the first k clusters is the operation of subtracting the $m \times n$ matrix

$$\sum_{t=1}^{k} x^{(t)} {x^{(t)}}^T A$$

from A. So if

$$R^{(k)} = A - \sum_{t=1}^{k} x^{(t)} {x^{(t)}}^T A,$$

$R^{(k)}$ defines a "residual" graph after removing the first k clusters. (Actually it represents a weighted graph with edge weights.) The intuition is that if the first few clusters are of large weight, then the residual matrix will have small norm (sum of squared entries). There is a way of quantifying this from Linear Algebra: (note that $\sum_{t=1}^{k} x^{(t)} {x^{(t)}}^T A$ is a matrix of rank k)

Fact 1 $R^{(k)}$ has the least sum of squares of entries among all matrices of the form $A - D$ where rank$(D) \leq k$. Also, $R^{(k)}$ has the least 2-norm (the 2-norm of a matrix M is $\max_{u:|u|=1} |Mu|$) among all matrices of the form $A - D$ where rank$(D) \leq k$.

So, the optimal clustering makes the "error" matrix $R^{(k)}$ as small as possible in two natural ways.

We defined the weights of clusters by looking at the out-degrees. Symmetrically, we may look at the in-degrees. Luckily Linear Algebra tells us (and we elaborate further in the final paper)

Fact 2 An optimal clustering with respect to in-degrees yields also an optimal clustering with respect to out-degrees and vice versa.

We close this section with one more important property of optimal clusterings. Recapping our discussion of "reinforcement of opinions", we note that reinforcement comes from large sets S, T of nodes (not necessarily disjoint) with many edges of the form $(u, v), u \in S; v \in T$.

This aspect of such clustering as well as the introduction of SVD technique to cluster such graphs was pioneered by Kleinberg [**?**]. (See also Gibson, Kleinberg and Raghavan [**?**]). Kleinberg [**?**] considered the ubiquitous problem of how to glean the most relevant documents from the (usually large) set of documents returned by a standard Web Search program for a key word. The intuition is to define a document to be an "authority" if a lot of other documents (returned by the search) point to (have a hypertext link to) it. He argues why it is not a good idea just to take documents which are pointed to by a lot of others. He defines a dual notion - a document is a "hub" if it **points to** a lot of other documents. More generally, suppose n documents are returned by the search engine. Then, he defines an $n \times n$ matrix A where A_{ij} is 1 or 0 depending upon whether the ith document points to the jth. [He does not explicitly deal with this large matrix.]

He sets out to find two n-vectors - x, y where x_i is the "hub weight" of document i (the weight is higher if the document is a good hub) and y_j is the "authority weight" of document j. With the normalization $|x| = |y| = 1$, he argues (and we do not reproduce the argument here) that it is desirable to find $\max_{|x|=|y|=1} x^T Ay$, (since in the maximizing x, y we expect the hub weights and authority weights to be mutually consistent.)

This is of course the problem of finding the singular vectors of A. Since A is large (in his examples, in the hundreds or thousand) , he judiciously chooses a submatrix of A and computes only the singular vectors of it. Our approach here would be to choose the submatrix according to a probability distribution.

He also points out that especially in the case when the key word has multiple meanings, not only the top, but some of the other singular vectors (with large singular values) are interesting. For example, when the key word is "JAVA" the top few singular vectors put high authority weights on documents about the programming language JAVA whereas another set of singular vectors put high weights on documents about the Island, others on documents about the coffee etc. So, it is of interest to find the largest k singular vectors for some small k; this is indeed the problem we consider here.

¿From Linear Algebra, then, we know that after removing the top k clusters, in the residual matrix $R^{(k)}$, the maximum weight of a cluster is at most $\sigma_{k+1}(A)$. We use this to argue below that in $R^{(k)}$, there are no submatrices with large sums. In the above context (of the web application) this means that after removing the top k clusters, there is no appreciable set of hubs and authorities.

Lemma 1 With the notation as above, for any subsets $S, T \subseteq V$,

$$\left| \sum_{i \in S, j \in T} R_{ij}^{(k)} \right| \leq \sigma_{k+1}(A) \sqrt{|S||T|}.$$

Corollary If $|S| = |T| = l$, then the average degree of of a node of S in the induced subgraph (S, T) of $R^{(k)}$ is at most $\sigma_{k+1}(A)$.

Thus if $\sigma_k(A)$ falls off rapidly with k then the residual graphs do not have large reinforcing subgraphs. In other words, in this case, all large reinforcing subgraphs are caused by relatively few clusters.

4 Fast SVD Algorithm

Suppose $P_i, i = 1, 2, \ldots n$ are nonnegative reals summing to 1 and satisfying

$$P_i \geq c|A_{(i)}|^2/||A||_F^2, \tag{5}$$

where c is a constant, such that we have the ability to sample the columns of A with probabilities $\{P_i\}$. Thus a sampler which samples the columns with probabilities proportional to their length squared would do. [Then we may take $c = 1$.] A sampler with this property is available if one has an idea of the lengths of the columns, but in any case, it is easy to set up such a sampler after one pass through the matrix A [**?**] :

Remark : For any matrix at all, we claim that after making one pass through the entire matrix, we can set up data structures so that after that we can sample the entries fast - $O(1)$ time per sample with $\{P_i\}$ satisfying (**??**).

Suppose M is such that for all i, j

$$A_{ij}^2 \leq M$$

$$A_{ij}^2 = 0 \qquad \text{OR} \qquad |A_{ij}|^2 \geq 1/M.$$

We create $O(\log M)$ bins; in the one-pass, we put into the lth bin all the entries (i, j) such that $(1/M)2^{l-1} \leq |A_{ij}|^2 \leq \frac{1}{M}2^l$. We also keep track of the number of entries in each bin. After this, we pretend all entries in a bin are of equal absolute value and then it is easy to set up a sampler - the details of the data structures are elementary and left to the reader.

The algorithm below will pick a certain number of columns of A according to $\{P_i\}$ satisfying (**??**). Then it will scale the columns in a certain way (according to the $\{P_i\}$). The $\{P_i\}$ ensure that "heavier" columns are more likely to be picked; the scaling can be looked on as "compensating" for the "over-weighting" of the heavier columns. Of course, we supply a proof that our method works.

4.1 Algorithm.

Let $0 < \varepsilon, \delta < 1$ be given.

(1) Let $s = 4k/(\varepsilon^2 c\delta)$. Select independently s columns of A (according to the distribution $\{P_i\}$); if column i is selected, include

$$A_{(i)}/\sqrt{sP_i}$$

as one of the columns of a matrix S.

So, S is a $m \times s$ matrix after the random selection.

(2) Find $S^T S$. (This takes time $O(s^2 m)$ or time $O(s^2 r)$ where r is the maximum number of entries in a column of A.)

(3) Find the top k eigenvectors $p^{(i)}$, $i = 1, 2, \ldots, k$ of the $s \times s$ matrix $S^T S$ and return $q^{(i)} = Sp^{(i)}/|Sp^{(i)}|$, $i = 1, 2, \ldots, k$ as the clusters. The $q^{(i)}$ are $m-$vectors.

4.2 Analysis of the Algorithm.

Note that since the $q^{(t)}$ are orthonormal, we have from Linear Algebra,

$$||A - \sum_{t=1}^{k} q^{(t)} q^{(t)^T} A||_F^2 = ||A||_F^2 - \sum_{t=1}^{k} |A^T q^{(t)}|^2. \tag{6}$$

Also from Linear Algebra,

$$\min_{D:\mathrm{rank}(D) \leq k} ||A - D||_F^2 = \sigma_1^2(A) + \sigma_2^2(A) + \ldots \sigma_k^2(A).$$

We prove

Lemma 1 *For any positive real θ, we have*

$$\mathbf{Pr}\left(\sum_{t=1}^{k} \left(|A^T q^{(t)}|^2 - \sigma_t(A)^2 \right)^2 \geq 2\theta^2 ||A||_F^4 \right) \leq \frac{1}{c\theta^2 s},$$

*where c is as in (**??**). From this, we get that with probability at least $1 - \delta$, we have (with s as in the algorithm)*

$$||A - \sum_{t=1}^{k} q^{(t)} q^{(t)^T} A||_F^2 \leq \min_{D:rank(D) \leq k} ||A - D||_F^2 + \varepsilon ||A||_F^2.$$

Proof Now $q^{(t)}$, $t = 1, 2, \ldots, k$ are singular vectors of S. Writing AA^T and SS^T both in a coordinate system with $q^{(1)}, \ldots q^{(k)}$ as the first k coordinate vectors, we see that $q^{(t)^T}(AA^T - SS^T)q^{(t)}$ is the (t, t) entry of $AA^T - SS^T$. So we have

$$\sum_{t=1}^{k} (q^{(t)^T}(AA^T - SS^T)q^{(t)})^2 \leq ||AA^T - SS^T||_F^2.$$

¿From this we have

$$\sum_{t=1}^{k} \left(|A^T q^{(t)}|^2 - |\sigma_t(S)|^2 \right)^2 \leq ||AA^T - SS^T||_F^2. \tag{7}$$

Applying the the Hoffman-Wielandt inequality (see Golub and Van Loan [?]), we see that

$$\begin{aligned}\sum_{t=1}^{k}(\sigma_t(SS^T)-\sigma_t(AA^T))^2 &= \sum_{t=1}^{k}(\sigma_t(S)^2-\sigma_t(A)^2)^2 \\ &\leq ||AA^T-SS^T||_F^2.\end{aligned}$$

Adding the two and using the inequality $(a+b)^2 \leq 2a^2+2b^2$ for any real a, b, we get

$$\sum_{t=1}^{k}\left(|A^Tq^{(t)}|^2-\sigma_t(A)^2\right)^2 \leq 2||AA^T-SS^T||_F^2. \tag{8}$$

Re-phrasing Lemma 1 of [?]: For all $\theta > 0$,

$$\mathbf{Pr}(||AA^T-SS^T||_F \geq \theta||A||_F^2) \leq \frac{1}{\theta^2 cs}.$$

This gives us the first assertion of the lemma.

Applying the last with $\theta = \varepsilon/(2k)^{1/2}$, we get that with probability at least $1-\delta$,

$$||AA^T-SS^T||_F \leq \varepsilon||A||_F^2/(2k)^{1/2}.$$

Putting this into (??), applying the Cauchy-Schwartz inequality and using (??), we get the second assertion of the lemma. □

4.3 Doing better than worst-case.

Note that even though we prove that s at the value we have does the job, it is possible that in an actual problem, the situation may be far from worst-case. In fact in practice, it suffices to pick s rows (where s is at first much smaller than the worst-case bound). Then we may check for the resulting approximation $\sum_{t=1}^{k} q^{(t)}q^{(t)^T}A$ to A whether it is sufficiently close to A. We can do this in a randomized fashion, namely, sample the entries of $A-\sum_{t=1}^{k} q^{(t)}q^{(t)^T}A$ to estimate the sum of squares of this matrix. If this error is not satisfactory, then we may increase s. The details of variance estimates on this procedure are routine and left to the final paper.

4.4 Preliminary experimental results.

While the algorithm has asymptotically a linear time upper bound, the bound still depends polynomially on $k, 1/\epsilon$. There is some hope that in practice it is better than the theoretically provable bound on $k, 1/\epsilon$. We have performed some limited experiments to date on dense matrices and plan to perform more experiments on larger sparse matrices.

Since the algorithm is only of interest in the case when the matrix has a good low rank approximation, we tried the algorithm on certain "randomly" generated dense matrices with known singular values. We generated 1000×1000 random matrices of the form $A = U\Sigma V^T$, where U and V are random orthogonal matrices and Σ is a diagonal matrix containing the specified singular values of A. To generate U and V we generated some perfectly random orthogonal matrices using a $O(n^3)$ procedure described by Stewart ([**?**]) and subsequently we performed a random walk in the set of orthogonal matrices by applying $O(n)$ random Givens rotations ([**?**]) in order to get new elements of the set.

In our experiments, we varied the percentage of the Frobenius norm of the matrix A that was contained within the top k singular values of A. We had the following parameters : k and q, where k was a number between 10 and 50 (and the top k singular values were large) and

$$q = \frac{\sigma_1^2 + \sigma_2^2 + ... + \sigma_k^2}{||A||_F^2}$$

where σ_i are the singular values of A. In other words, q is the fraction of the Frobenius norm captured by the first k singular values. We varied q between 0.4 and 0.8.

For a given k, q, we set the singular values $\sigma_1, \sigma_2, \ldots \sigma_k$ to be $c \cdot k, c \cdot (k-1), \ldots c \cdot 1$ respectively for some constant c (i.e., i th one was proportional to $k+1-i$). The other singular values were set to be equal. A little calculation shows that subject to these conditions, given $||A||_F^2$, the singular values are completely determined. The reason for choosing the top k in this way was that we did not want them to fall off too steeply (this would make the computation of singular values using the power method or better Lancos method easy), but at the same time, we wanted them to occupy a constant fraction (q) of the Frobenius norm sqaured.

Our experiments were to find how many rows of A we would have to pick (at random) so that our approximation would have an error of at most 3 %, i.e., we wanted to ensure that if A is the input matrix and D^* the rank K approximation we find, then we had

$$||A - D^*||_F^2 \le \min_{D:\text{rank}(D)\le k} ||A - D||_F^2 + (0.03)||A||_F^2.$$

Note that since we know all the singular values of A, we can compute $\min_{D:\text{rank}(D)\le k} ||A - D||_F^2$ and we can determine if the above requirement is met. We varied s, the number of random rows picked until the requirement was met.

Following are our results (they were obtained from running our experiments over 15 random matrices for each entry of the table generated as described above):

k	q	*Number of rows*
10	0.8	66
10	0.6	124
10	0.4	179
20	0.8	127
20	0.6	210
20	0.4	238
30	0.8	188
30	0.6	249
30	0.4	391
40	0.8	226
40	0.6	373
40	0.4	436
50	0.8	248
50	0.6	432
50	0.4	448

Note that the theoretical bound only guarantees the requirement will be met if we use s of the order $k/(.03)^2$ which is very large. The limited experiments here are much more promising. We plan to do more experiments with much larger sparse matrices which are also generated to have specific singular values.

Acknowledgment : We wish to thank Alan Edelman, Stan Eisenstat, Michael Littman, Prabhakar Raghavan and R. Ravi for their comments.

References

[1] P.Agarwal and M.Sharir, Planar geometric location problems, DIMACS 1990.

[2] S.Arora, P.Raghavan and S.Rao, Approximation schemes for k-medians and related problems, Proceedings of the 30th Annual ACM Symposium on Theory of Computing (1998) 106-113.

[3] M.Bern and D.Eppstein, Approximation algorithms for geometric problems, in *Approximation algorithms for hard problems*, D.Hochbaum Ed., PWS Publishing, 1996.

[4] M.W.Berry, S.T.Dumais, and G.W.O'Brien. "Using linear algebra for intelligent information retrieval", SIAM Review, 37(4), 1995, 573-595, 1995.

[5] Z.Drezner, Facility location : a survey of applications and methods, Springer, 1995.

[6] M.E.Dyer and A.M.Frieze, A simple heuristic for the p-centre problem, *Operations Research Letters 3* (1985) 285-288.

[7] B.Everitt, Cluster analysis, published on behalf of the Social Science Research Council by Heinemann Educational Books; New York, Halsted Press, 1980.

[8] A.M.Frieze, R.Kannan and S.Vempala, Fast Monte-Carlo algorithms for finding low rank approximations, to appear in FOCS '98

[9] G.H.Golub and C.F.Van Loan, Matrix Computations, Johns Hopkins University Press, London, 1989.

[10] D.Gibson, J.Kleinberg and P.Raghavan, Clustering Categorical Data: An Approach Based on Dynamical Systems, *Very Large Data Bases (VLDB)* (1998) 311-322.

[11] D.S.Hochbaum and D.B.Shmoys, A best possible heuristic for the k-center problem, *Mathematics of Operations Research* (1985) 180-184.

[12] M.Jambu and M-O.Lebeaux, Cluster analysis and data analysis, North-Holland, 1983.

[13] J.Kleinberg, Authoritative sources in a hyperlinked environment, Proceedings of 9th Annual ACM-SIAM Symposium on Discrete Algorithms, (1998) 668-677.

[14] J.Kleinberg, C.H.Papadimitriou and P. Raghavan, *Segmentation Problems* in STOC 98.

[15] M.Lorr, Cluster analysis for social scientists, Jossey-Bass, 1983.

[16] G.W. Stewart, The efficient generation of random orthogonal matrices with an application to condition estimators, SIAM J. of Numerical Analysis, Vol. 17, No. 3, June 1980, pp. 403-409.

[17] R.C.Tryon and D.E.Bailey, Cluster analysis, McGraw-Hill, 1970.